ICMC2012 NON-COCHLEAR SOUND

WELCOME NOTES
ORGANISATION
PARTICIPANTS

ICMC2012 NON-COCHLEAR SOUND
WELCOME FROM THE ICMA

Dear ICMC2012 Delegates,

On behalf of the ICMA, it is with great pleasure to welcome you to the 2012 ICMC. Since 1974, we have convened annually at the ICMC conferences at various locations and countries all over the globe including Canada, China, Cuba, Denmark, France, Ireland, Japan, Germany, Greece, Netherlands, Scotland, Singapore, Spain, and the United States. We are excited that Slovenia has joined the expanding list of countries that have hosted our annual meeting as it shows that computer music is reaching more diverse audiences and communities.

The theme for this year’s conference is “Non-Cochlear Sound,” a concept coined by Seth Kim-Cohen which can be “understood as an analogy to the Duchampian notion of non-retinal art.” The aim of “Non-Cochlear Sound” is to “…investigate the potential of sound as a medium and further, the potentials of music in conjunction with new technologies to create new possibilities of artistic expression, which could create a closer relation to language and consequentially, enable generation of meaning and production of knowledge.”

As is always the case, the conference organizers have put in long hours, days, and months into bringing this conference to you. The conference would not have been possible without the dedication by Miha Ciglar’s team, the music/paper reviewers, and many other volunteers who have generously donated their time and effort tackling this enormous task. We look forward to enjoying an engaging a fun week of computer music in Ljubljana.

Welcome back to those who have been to previous ICMCs, dobrodošli to newcomers, and bienvenue to all. Let the music begin!

Sincerely.

Tae Hong Park
President, ICMA

WELCOME FROM IRZU INSTITUTE FOR SONIC ARTS RESEARCH

We are very glad to present the 38th edition of the International Computer Music Conference, taking place in collaboration with the EarZoom sonic arts festival, within the framework of IRZU – Institute for Sonic Arts Research in Ljubljana, Slovenia. IRZU is a small private institute and was founded in 2008 with the aim of filling a local content gap in the interdisciplinary field of electronic music / sonic arts. Our activities cover the following three main areas: audio technology research (digital signal processing, sonic interaction design, music information retrieval, non-linear acoustics, etc.), artistic productions (performances, exhibitions, installations, films) and educational projects (lectures, workshops), which are mostly aimed at transmitting basic knowledge on the use of digital technologies and conceptual approaches in sonic arts, to local composers and inter-media artists. The scientific research at IRZU is oriented towards technologies that can be used in an artistic context, but which also exhibit a commercial potential, through which we are actively working towards establishing an alternative funding resource for promoting experimental electronic arts and music in Slovenia.

One of IRZU’s main projects is the EarZoom festival, which is taking place annually, since 2009. The aim of the festival is to create a referential international platform for discussing the latest developments within audio technology research and the artistic trends in sonic practices. EarZoom is featuring international artists and scientists, presenting state of the art research in exciting disciplines like machine listening, algorithmic composition, new interfaces for musical expression, open source music production software, computer vision systems, spatial audio systems, etc.

EarZoom is continuously growing in size and in this year (2012), when the festival is held in collaboration with the ICMC we have reached a peak, which will not be matched any time soon. All of IRZU’s activities strongly depend on public funding in the context of cultural projects. In 2012 we have witnessed big budget cuts in all sectors, including the field of culture and arts, which have recently also lost an autonomous Ministry of culture in Slovenia. We can therefore only hope that in the future our efforts will be recognized – possibly also outside artistic contexts – and that we will be able to continue with our work after the ICMC2012 is finished.

It is the first time in the history of the ICMC that it is hosted by a non government organization, however, the realization of ICMC2012 was also supported by the Faculty of Computer and Information Science as well as the Faculty of Arts, at the University of Ljubljana. In addition to that, we have received generous support on the part of all participating venues: Kino Šiška, Španski Boro | En knap, Stara Elektrarna | Blaueke, Dijadet Dom Tabor, Galerija ŠKUC, Galerija Jakopič, Galerija Kapelica, and Kiberpipa.

Like in previous years, the ICMC2012 is featuring a wide range of papers and music, covering all aspects of technical, creative and aesthetic issues around the use of computers in music. The theme of the 2012 conference is “Non-Cochlear Sound”, which can be understood as an analogy to the Duchampian notion of non-retinal art. The phrase “Non-Cochlear Sound” was coined by artist and writer, Seth Kim-Cohen, in his 2009 book “In The Blink Of An Ear: Toward A Non-Cochlear Sonic Art”. We are proud to announce that Seth Kim-Cohen as well as the German theorist Diedrich Diederichsen are actively participating in this year’s ICMC as keynote speakers. As it was anticipated prior to the call for works, there were actually not many submissions referring to the conference theme. The aim of the thematic frame however is still to investigate the potential of sound as a medium and further, the potentials of music in conjunction with new technologies to create new possibilities of artistic expression. We still hope that during the conference, we will be able to trigger some fruitful discussions and together examine the possibilities for creating a closer relation between music and language, which would consequentially, enable generation of meaning and production of knowledge.

Miha Ciglar
Conference Chair, ICMC2012
WELCOME FROM THE MUSIC PROGRAM CHAIRS

We are proud to present the proceedings of the 2012 International Computer Music Conference, held in Ljubljana, the capital of Slovenia. We accepted to coordinate the program of this conference with a great sense of responsibility and tried our best to continue building on the strong record of previous ICMC conferences and support the interdisciplinary community of researchers and practitioners in the field of Computer Music. This was not an easy task as we received a large number of high quality submissions and had to make some difficult decisions assisted by the hard work of the music-reviewers and reviewers.

We employed a double-blind reviewing process, where each paper was reviewed by at least two to three reviewers. Meta-reviewers thoroughly reviewed the assigned reviews and did their best to resolve any disagreements between reviewers, providing a summary of the overall submission quality through their meta-reviews. Based on the reviews and submitted category, the accepted submissions were grouped into one of the following categories: long papers (up to 8 pages in the proceedings, short papers (4 pages), posters, demos and studio reports (4 pages). Because of the double-blind review process we did not impose any authorship limitations.

ICMC 2012 received a total of 165 submissions out of which 107 were accepted, resulting in an acceptance rate of 64%. 26 long papers were selected (the number of long paper presentation slots is limited to 16, so 12 are actually presented as a short paper), 36 short papers, 35 posters, 7 demos and 3 studio reports. 22 meta-reviewers and two Chair pairs coordinated the efforts of 169 reviewers. Before paper assignments, reviewers were invited to indicate their preferences for papers to review. These preferences directly informed the paper assignment process, thus ensuring knowledgeable reviews and assessor confidence. Each submission received at least two reviews and a meta-review. The final decisions were made by the Paper Chairs primarily taking into account the recommendation of the meta-reviewers and the reviewers as well as balance of the overall program.

Matija Marolt and Martin Kaltenbrunner
Paper Chairs, ICMC2012

WELCOME FROM THE MUSIC PROGRAM CHAIR

The ICMC2012 music steering committee launched an open call for musical works in late 2011. Between December 15th 2011 and February 15th 2012, we received 539 successfully submitted applications from 42 different countries via the EasyChair online interface. The submissions were done in 12 categories and were coordinated by a program committee of 18 meta-reviewers who invited a total of 198 reviewers, all of which were highly experienced and internationally acknowledged composers, sound artist and researchers.

Like in previous years, the music reviewing and selection process was performed in a double-blind manner, where the authors as well as the reviewers identities were undisclosed. Each submission was reviewed by at least two, but in most cases by three reviewers. The reviewers were asked to give grades based on the following criteria: Overall Evaluation; Reviewer’s Confidence; Artistic Merit; Technical Quality and Feasibility of Performance. Furthermore, the meta-reviewers took some considerations to resolve the status of pieces with contrasting reviews, and helped also to review and grade pieces, with missing reviews. At the end of the reviewing process, we had 520 accepted submissions. The final selection of 126 works was completed by the four music chairs, who carefully examined all the selected pieces, in order to prepare a well-balanced program, which, we believe represents a strong image of the current state in computer music composition and performance.

I would like to thank the co-chairs Steven Lay, Gregor Pompe and Brane Zorman for their help and support. Their knowledge and dedication to this venture contributed to creating a program we will all enjoy during the Conference.

Mauricio Valdés
Music Chair, ICMC2012
We are pleased to announce that the best paper award of the International Computer music Conference 2012 goes to:

**ANDREA VALLE and DARIO SANFILIPPO: "TOWARDS A TYPOLOGY OF FEEDBACK SYSTEMS"**

Each year the ICMA recognizes the best paper submitted with this award. The top eight scoring papers were given to a panel of computer music experts, including last year's winner. The panel was not given the original reviewers' scores; they scored each paper individually, and then were instructed to come to consensus. The quality of the submissions made this process particularly difficult, but eventually one clear winner emerged.

**PANEL:**
Chair: Margaret Schedel
Rebecca Fiebrink
Lonce Wyse

We are pleased to announce that the best paper award of the International Computer music Conference 2012 goes to:

**DARIO SANFILIPPO and ANDREA VALLE: "TOWARDS A TYPOLOGY OF FEEDBACK SYSTEMS"**

**THE **ICMC2012** PAPER PROGRAM COMMITTEE**

**THE **ICMC2012** PAPER PROGRAM REVIEWERS**

**THE **ICMC2012** PAPER PROGRAM COMMITTEE**

**THE **ICMC2012** PAPER PROGRAM REVIEWERS**
THE _ICMC2012 MUSIC PROGRAM COMMITTEE

Alex McLean, IGRSM University of Leeds, UK
Andreas Weisler, Bruckner-University, Linz, Austria
Christopher Hawthorn, Sonic Arts Research Centre, Queen’s University Belfast, UK
Daniel Schorno, STEIM, Amsterdam, The Netherlands
Edgar Barroso, Harvard University, Cambridge, MA, USA
Fernando Schroeder, Sonic Arts Research Centre, Queen’s University Belfast, UK
Georg Boenn, Cardiff School of Creative & Cultural Industries Univ. of Glamorgan, UK
Gregorio Jimenez, Conservatorio Superior de Música de Valencia
Maja Cerar, Columbia University, New York, NY, USA
Robert Ratcliffe, Research Institute for the Humanities, Kiel University, Staffs, UK
Rodrigo Sigal, Centro Mexicano para la Música y las Artes Sonoras, Morelia, Mexico
Rogelio Sosa, General director of Festival Aural, Mexico city, Mexico
Scott McLaughlin, University of Huddersfield, UK
Se-Lien Chuang, Linz, Austria
Tom McLean, ICSRiM University of Leeds, UK

THE _ICMC2012 MUSIC PROGRAM REVIEWERS

carole charguier
Chris Saltor
Christopher Anta
Christopher Bailey
Christopher Hoddinott
Clay Chaplin
Dan Overholt
Dan Stowell
Gerald Bennet
Gerard Ermelay
Germán Torro-Pérez
Gilles Gobert
Gorbach Thomas
Graham Colman
Gregorio Jimenez
Harriët Urban
Henry Vega
Holmertz Klaus
Hugo Solís
Iñigo Ibáñez Barriaga
Ismael Martínez
Juan Narango
JÀVÀN SANZÉ
Jicca Ico Bukvic
J. Anthony Allan
Jack W. Stamps
Jan Schacher
Jane Dickson
Jaroslav Kapucinski
Jason Freeman
Jean-Claude Capella
Jean-Francois Charles
Jean Bagis 1 Rubi
Johannes Kretz
Johannes Zellbrin
John Richards
Johannson Kirk
Jon Christopher Nielsen
Jonathan C Reus
Jonathan Katz
José Lázaro Montes
José Luis Galiana
José Manuel Benínguer
José Orló Graus
Josué Moreno
Juan A. Romero
Juan Cristóbal Gariñelo
Juan José Elías de Cabanas
Judith Shatin
Judy Klein
Julian Brooks
Julien-Robert Legault Salvati
Justin Yang
Kassian Dut
Katherine Rosenberger
Kauffmann Dieter
Kevan Patton
Linz-Maues Igó
Luc Doerberiner
Ludger Brümmer
Luke Dubois
Manuel Bernina Medina
Manuel Roña Turturu
Manuela Meier
Marcel Warick
Mari Kimura
María Basilla
Mark Haulicy
Marco Cittanti
Marta Gentilucci
Martín Clarke
Martin Rumori
Matthew Sansoni
Max Neaport
Mccallan Peter
Michael Edwards
Michael Kingpling
Michael Matthews
Miguel Angel Berbós López
Miguel Ortiz Perez
Mike Loevenstein
Mike Vaughan
Mike Vonovsky
Musil Wolfgang
Orestis Karamanlis
Otto Cadore
Pablo García-Valezuela
Panayiotis Kokoras
Pasoula Aki
Paul Rudy
Paul Wilson
Paula Mattusszen
Pedro Carniño
Por Bloland
Raúl Minsburg
Rebecca Flerkink
Ricardo Cortes
Ricardo de Armas
Richard Elger
Richard Glover
Rob Canning London
Robert MacKay
Roberto Morales
Rodgri Cadiz
Rodrigo Sigal
Rogelio Sosa
Rupert Toll
San Purcuell
Scott Hewitt
Sebastian Berwick
Seppo Grünild
Shintaro Imai
Simon Kihlax
Simon Vincent
Simon Waters
Simon Bokesssy
Stephen Thomas
Takao Maiza
Tara Rodgers
Theodore Lottis
Thomas Gardner
Thomas Görelz
Thomas Grill
Tom Mud
Veronika Mayer
Vincent Gómez Pons
Volkan Kihn
Will Schricht
Winfried Ritsch
Yair López
Yen-Ting Cho
Yuri Spitsyn

LIST OF PREVIOUS CONFERENCE SITES

2001 University of Huddersfield, UK
2008 New York, United States
2009 McGill University, Montreal, Quebec, Canada
2008 Sonic Arts Research Centre, Belfast, N. Ireland
2009 Aalborg University Esbjerg, Denmark
2008 Hong Kong University of Science and Technology, China
2009 Gothenburg, Sweden
2009 Havana, Cuba
2009 Berlin, Germany
1999 Beijing, China
1998 University of Michigan, Ann Arbor, United States
1998 Aristotle University of Thessaloniki, Greece
1998 Hong Kong University of Science and Technology, China
1995 Barbif Centre for the Arts, Canada
1994 diem, Danish Institute of Electroacoustic Music, Denmark
1993 Wasada University, Japan
1992 San Jose State University, United States
1991 McGill University, Canada
1990 University of Glasgow, Scotland
1989 Ohio State University, United States
1988 gmmik, Cologne, Germany
1987 University of Illinois at Champaign-Urbana, United States
1986 Royal Conservatory of Music, Den Haag, Netherlands
1985 Simon Fraser University, Canada
1984 ICRM, France
1983 Eastman School of Music, United States
1982 The Venice Biennal, Italy
1981 New York University, United States
1980 Queens College, New York City, United States
1980 Northwestern University, Illinois, United States
1979 Aristotle University, Thessaloniki, Greece
1978 Massachusetts Institute of Technology United States
1976 University of California, Los Angeles, United States
1975 University of Illinois at Chicago, United States
1974 University of Michigan, East Lansing, United States

1992 San Jose State University, United States
1991 McGill University, Canada
1990 University of Glasgow, Scotland
1989 Ohio State University, United States
1988 gmmik, Cologne, Germany
1987 University of Illinois at Champaign-Urbana, United States
1986 Royal Conservatory of Music, Den Haag, Netherlands
1985 Simon Fraser University, Canada
1984 ICRM, France
1983 Eastman School of Music, United States
1982 The Venice Biennal, Italy
1981 New York University, United States
1980 Queens College, New York City, United States
1980 Northwestern University, Illinois, United States
1979 Aristotle University, Thessaloniki, Greece
1978 Massachusetts Institute of Technology United States
1976 University of California, Los Angeles, United States
1975 University of Illinois at Chicago, United States
1974 University of Michigan, East Lansing, United States
SESSION D2: DEMOS

PROBAO MUSIC: A MULTIMODAL ONLINE MUSIC LIBRARY
Verena Thomas, David Damm, Christian Frenaney, Michael Clausen, Frank Kurth and Meinard Müller

CONTROLLING MOZART’S DICE MUSIC USING ACCELERATION SENSORS
Ciril Bohak, Masahiro Nitsuma and Martja Marolt

TOOLS AND ABSTRACTIONS FOR SWARM BASED MUSIC AND ART
Daniel Bisig and Philippe Kocher

SESSION 6A: NETWORK MUSIC

SOURCENODE: A NETWORK SOURCED APPROACH TO NETWORK MUSIC PERFORMANCE (NMP)
Robin Renwick

OSCTHULHU: APPLYING VIDEO GAME STATE-BASED SYNCHRONIZATION TO NETWORK COMPUTER MUSIC
Curtis McKinney and Chad McKinney

REAL-TIME WEB TECHNOLOGIES IN THE NETWORKED PERFORMANCE ENVIRONMENT
Rob Canning

SESSION D3: DEMOS

AIRDUINO: AN INEXPENSIVE DIY MIDI WIND CONTROLLER
Timothy Anderson

TACTILE SOUND AESTHETIC EXPERIENCE
Justyna Zubrycka and Paweł Cyrta

SESSION 6B: EDUCATION

A PIANO LEARNING SUPPORT SYSTEM CONSIDERING RHYTHM
Yoshinari Takegawa, Tsutomu Terada and Masahiko Tsukamoto

MAESTRO: USING TECHNOLOGY TO IMPROVE KINESTHETIC SKILL LEARNING OF MUSIC CONDUCTORS
Andrea Brown and Yonatan Sasson

NUANCE: A SOFTWARE TOOL FOR CAPTURING SYNCHRONOUS DATA STREAMS FROM MULTIMODAL MUSICAL SYSTEMS
Daniele Salvati, Sergio Canazza and Gian Luca Foresti

SESSION 7: COMPOSITION SYSTEMS

AN INTRODUCTION TO SLIPPERY CHICKEN
Michael Edwards

MANUSCORE: MUSIC NOTATION-BASED COMPUTER ASSISTED COMPOSITION
James B. Maxwell, Arne Eigenfeldt and Philippe Pasquier

SESSION 8A: COMPOSITION TECHNIQUES

THE XYOLIN: A 10-OCTAVE CONTINUOUS-PITCH XYLOPHONE, AND OTHER EXISTEMOLOGICAL INSTRUMENTS
Steve Mann and Ryan Janzen

THE INVESTMENT OF PLAY: EXPRESSION AND AFFORDANCES IN DIGITAL MUSICAL INSTRUMENT DESIGN
Joanne Cannon and Stuart Favilla

SESSION 8B: NEW INTERFACES FOR MUSICAL EXPRESSION

THE XOVI: A 18-OCTAVE CONTINUOUS-PITCH XYLOPHONE, AND OTHER EXISTEMOLOGICAL INSTRUMENTS
Steve Mann and Ryan Janzen

THE INVESTMENT OF PLAY: EXPRESSION AND AFFORDANCES IN DIGITAL MUSICAL INSTRUMENT DESIGN
Joanne Cannon and Stuart Favilla

ADAPTING GENERAL PURPOSE INTERFACES TO SYNTHESIS ENGINES USING UNSUPERVISED DIMENSIONALITY REDUCTION TECHNIQUES AND INVERSE MAPPING FROM FEATURES TO PARAMETERS
Stefano Fasciani and Enrico Wyse

A MICROPHONE ARRAY INTERFACE FOR REAL-TIME INTERACTIVE MUSIC PERFORMANCE
Daniele Salvati, Sergio Canazza and Gian Luca Foresti
SESSION 9A: ARTIFICIAL INTELLIGENCE IN MUSIC

478 STATISTICAL PARSING FOR HARMONIC ANALYSIS OF JAZZ CHORD SEQUENCES
Mark Granroth-Wilding and Mark Steedman

504 A NEW APPROACH FOR CONSTRAINT PROGRAMMING IN MUSIC USING RELATION DOMAINS
Sascha Van Cauwelaert, Gustavo Gutiérrez and Peter Van Roy

523 A METHOD FOR COMPUTER CHARACTERIZATION OF “GESTURE” IN MUSICAL IMPROVISATION
Christopher Doharian

541 DEVELOPING AND COMPOSING FOR A ROBOTIC MUSICIAN USING DIFFERENT MODES OF INTERACTION
Mason Bretan, Marcello Cicconet, Ryan Nikolaidis and Gil Weinberg

SESSION 9B: SPATIAL AUDIO

504 A FRAMEWORK FOR THE CHOREOGRAPHY OF SOUND
Gerhard Eckel, Martin Rumori, David Pirro and Ramón González-Arroyo

516 SPATOSC: PROVIDING ABSTRACTION FOR THE AUTHORIZATION OF INTERACTIVE SPATIAL AUDIO EXPERIENCES
Mike Wozniakowski, Zuck Seet, Alexandre Quezay, Tristan Matthews and Luc Courchesne

531 CYCLICAL FLOW: SPATIAL SYNTHESIS SOUND TO AS MULTICHLANNE COMPOSITION TOOL
Andrew Dolphin

543 SOUNDS OF SIMULATIONS: DATA LISTENING SPACE
Katharina Vogt, David Pirro, Martin Rumori and Robert Heidrich

SESSION P4: POSTERS

531 SEMANTIC MOVIE SCENE ANNOTATION FOR RAPID PROTOTYPING OF SCORE MUSIC
Julian Rubisch, Jakob Doppler, Stefan Schuster and Hannes Raffaseder

534 PHRASE BOUNDARY ESTIMATION IN MUSIC PERFORMANCE WITH HMM-BASED UNSUPERVISED LEARNING
Tae Houn Kim and Stefan Weinzierl

541 AUDIO-TO-AUDIO ALIGNMENT USING PARTICLE FILTERS TO HANDLE SMALL AND LARGE SCALE PERFORMANCE DISCREPANCIES
Bo Xiong and Ozgur Izmirli

543 BROWSING MUSIC AND SOUND USING GESTURES IN A SELF-ORGANIZED 3D SPACE
Gabrielle Odowski-chuk and George Tzanetakis

547 ADVANCING EXPERT HUMAN-COMPUTER INTERACTION THROUGH MUSIC
Benjamin Smith and Guy E. Gannett

551 SURSOUND
Kun-Ying Tsai, Che-Wei Liu and Yu-Chung Tsang

555 MOTIVEVIEWER: HIERARCHICAL PATTERN DETECTION
Verona Thomas and Michael Clausen

559 PARTIAL TRACKING IN TWO STEPS
Adkr Sriska

563 COMPUTER AIDED MELODIC ANALYSIS USING SUFFIX TREES
Matevž Jakolc, Janez Demšar and Andrej Brednik

SESSION 10: COMPOSITION TECHNIQUES 2

567 ALLOTROPE: WORKS FOR SOLO TRUMPET, LAPTOP, PEDALS AND GUITAR AMPLIFIER
Peter Knight

573 ITERATIVE SYNAESTHETIC COMPOSING WITH MULTIMEDIA SIGNALS
Angus Forbes and Kiyomitsu Osita

579 VUZIK: A PAINTING GRAPHIC SCORE INTERFACE FOR COMPOSING AND CONTROL OF SOUND GENERATION
Aura Pon, Juniko Ishino, David Eagle, Ehud Sharlin, Nicolas D’Alessandro and Sheelah Carpendale

584 PRECISE PITCH CONTROL IN REAL TIME CORPUS-BASED CONCATENATIVE SYNTHESIS
Aaron Einbond, Christopher Trapani and Diemo Schwarz

SESSION 11: REPRESENTATION AND MODELS FOR COMPUTER MUSIC

589 VISUALIZATION OF PERCEPTUAL QUALITIES IN TEXTURAL SOUNDS
Thomas Grill and Arthur Flaxer

597 THE YIN YANG THEORY IN SOUND AND MUSIC: A FIRST EXPLORATION
Leonardo Gabrielli and Daniela Gabrielli

604 NAVIGATING VARIATION: COMPOSING FOR AUDIO MOSAICING
Diemo Schwarz and Benjamin Hackbarth

608 COMPRESSED MULTIDIMENSIONAL TREES FOR EVOLUTIONARY MUSIC REPRESENTATION
Abbas Pirmia and Jon McCormack

ICMC2011: LOST PAPER

614 FLEXIBILITY, SUITABILITY AND SPONTANEITY IN NEW INSTRUMENT DESIGN: THE FEEDBACK JOYPAD
Tom Mudd
A COGNITIVE APPROACH TO ELECTRONIC MUSIC: THEORETICAL AND EXPERIMENT-BASED PERSPECTIVES

Amı Çamcı
Academy for Creative and Performing Arts
Leiden University, The Netherlands
a.camci@umail.leidenuniv.nl

ABSTRACT
Meaning in music has been a topic of discussion throughout the history of the art form. Influenced chiefly by the trends in scientific methodology, perspectives on the semantics of music manifested considerable shifts in focus over time. Especially in the last century, the upsurge in cognitive research in the field of psychology has yielded a growing body of new studies that probe the processes at play during the perception of sound and music. However, research that specifically focuses on electronic music within this respect has been sparse and largely theoretical.

As both the material and the language of electronic music diverge from that of traditional musical practices, the composer's engagement with the listener's cognitive faculties is fundamentally altered. This necessitates idiosyncratic approaches towards electronic music that take into account the act of meaning attribution on both compositional and analytical levels. Stemming from the author's compositional practice, research discussed here incorporates an environmental event perception model to motivate a gesture-based approach to the study of cognitive units in electronic music. Current paper provides theoretical perspectives on this topic as well as preliminary experiment results which substantiate the proposed ideas.

1. INTRODUCTION
This paper is a product of an ongoing research project which investigates the cognitive mechanisms that operate during the experiencing of electronic music. The project takes into account both composer's (poietic) and listener's (esthesic) points of view to deal with the subject matter while utilizing perspectives and methods inherited from fields as diverse as narratology and cognitive psychology.

The paper presents current theoretical inferences and experimental findings: Section 2 offers a historical context for compositional and analytical approaches to electronic music. Section 3 provides an overview of the cognitive idiosyncrasies of electronic music in relation to instrumental practices while Section 4 discusses a gesture-based framework for electronic music informed by the discussed cognitive phenomena. Section 5 presents experimental data from listening tests to materialize the theoretical discourse.

2. HISTORICAL CONTEXT
Having now grown approximately a hundred years old, electronic music has spawned a myriad of sub-genres while benefiting greatly from the advancements in analog electronics in its earlier periods, and computer technology during the latter half of its existence. Today, studios are equipped with consumer-grade machines that can, not only handle legacy techniques of electronic music-making, but furthermore, push the borders of one's imagination in creating unique ways to process audio. The modern medium affords the composer with a plethora of technology that liberates the artistic expression to a greater extent as the translation from idea to sound becomes increasingly transparent.

Earlier periods of electronic music, however, was subjected to many technical limitations, which influenced artists on both practical and conceptual levels, thus effectively defining the aesthetic directions of the time. In contrary to the intuitiveness of the vision of “opening music to all sounds” [1] at the genesis of the genre, the technology at the time to materialize this vision was highly intricate and, therefore, accountable for artistic defenses which were often inherited from instrumental music practices. Herbert Eimert, one of the prominent figures of the Cologne studio in the 1950,s, writes: “It is certain that no means of musical control could have been established over electronic material had it not been for the revolutionary thought of Anton Webern (...)” [1], referring to the Serialist Movement of the Second Viennese School, which mandates the designation of musical parameters through serial permutations. However, this highly parametric approach practiced in certain schools of electronic music was gradually abandoned over the course of subsequent decades in favor of composer's instincts [3] as the arts gained control over and furthermore, started to influence technology. Zanpronha reflects that, “non-motivated” parametric procedures from the earlier periods were replaced, over time, by approaches which acknowledged the “complexity of listening and sound references” [17].

This transformation on the compositional side of the music has been paralleled by the perceptual approaches towards it as well. The emerging cognitively-oriented methods of analysis began to take into account the characteristics of the electronic music experience [5] [12][16]. Some of the ontological perspectives which surfaced as a part of this trend will be further discussed.

3. COGNITIVE IDIOSYNCRASIES OF ELECTRONIC MUSIC
In this section, electronic music will be situated in a broader context of musical practices in order to delineate the contrasting cognitive aspects between electronic and instrumental music. This will, in return, highlight the idiosyncrasies of electronic music in relation to common ideas pertinent to traditional musical practices.

3.1. Cognition of Instrumental Music
There exist intrinsic cognitive differences for both the listener and the composer, between the experiences of
instrumental and electronic music. The principal factors behind this differentiation are material and language: instrumental music refers to electroacoustic eras, utilizes fabricated sound sources, which have been refined to their formal forms as the most widely accepted instruments to craft sounds that do not exist in the nature [6], sometimes referred to as harmonic or pure sounds. Electronic music is also synthetic in a similar fashion: Abstract concepts like melody, harmony, and rhythm are modeled and widely acknowledged over the course of centuries. These fabricated structures have formed a musical language that is now engrained to our deep-seated mechanisms of music perception. It is therefore that, music is understood to be a form of artistic expression [9], which breeds a self-sustaining, abstract dialect. There remains no delegation in between the material and the experience of instrumental music. Although referentialist composers of program music might impel the dynamics of the orchestra to new ends, the listener is left to imagine the music in concordance with the extramusical narrative, the abstraction proposed by the musical sounds opens up for the listener, a world of affects rather than representations. This is the primary reason for research on music perception to be largely focused on discovering associations between music and emotions.

3.2. Cognition of Electronic Music

However, in the case of electronic music, there is witnessed an acceleration in the so-called musical sound. With the constraints of physical instruments and the limitations of human perception, whether it may be recorded, processed or synthesized, becomes a material for composition. Undoubtedly, this change in material mandates a change in language as well. Just as the sounds propagating in our daily environments cannot be described via the traditional musical language, sonic events harbored within electronic music display features that fall outside the realm of language. Concurrently, the experience of electronic music, as well, diverges significantly from that of instrumental music. With the introduction of new sound sources that are not distinguished by the listener as being musical, the cognitive response to the genre evolves into a distinct process of sonic attribution that is not sufficed by the culturally embedded mechanisms of music perception. As the electronic music medium expands and the listener encounters with this music prompts a more encompassing cognitive process: The listeners’ “ear-witness accounts” [13] of previously experienced daily events, or their “aural perspicacity” as Trevor Wishart refers to them [16], constitute the main reference in describing the experience of this music. Therefore, amid the material’s ascent to affect which was previously described to be evoked during a compositional process, we observe the emergence of a mediating layer as a new continuum from sonic event to affect to materializer. A relationship this nature of electronic music yields a distinct form of communication between the composer and the listener. The composer (being a listener herself), experience electronic music within a broader domain of cognitive associations to sounds:

The representation of reality is now a compositional task of a gesture which (as model) is found at the heart of many contemporary electroacoustic approaches; be they acoustical, soundscape/ ecological, or even musique concrète. There is no longer any need for composers or listeners to ignore the extramusical connotations of electroacoustic sounds. [5]

However, the communication between the composer and the listener is largely yet to be explored. The concept to percept bridge is obscured by the breadth of possibilities offered by the medium. Smalley explains that the listener is left to “imagine the music” [7], references to “how to cut an aesthetic path and discover a stability in a world of changing events” [8]. From middle-open sound to the extensive recombination of sound-making methods. He further elaborates that “[the] problems of representation are complex and combined with the multiple, often similarly and fairly universally applicable analytic tools have evidently inhibited electroacoustic music’s accessibility in more intellective musical circles” [12].

4. GESTURE/EVENT PERSPECTIVE

The designs of the electronic music composer are inherently connected to her knowledge of how sounds evolve over time. Objects of the ear-witness accounts, which construct knowledge of the listener as being gesture, in a rather general sense, is the inherent intentionality of a gesture on the composition” [10]. Gestures in electronic music are compared to events in nature; the semantic distinction is that the entireity of a piece and co-exist in multi-layered forms, similar to temporal and textural varieties evident in our daily soundscape. Such correlation between sounding environmental events and gestures in electronic music are demonstrated and substantiated with the data obtained from preliminary experiments.

5. PRELIMINARY EXPERIMENTS

Subsequent to extensive studies on developing a test method which is both idiosyncratic to electronic music listening and capable of extracting a comprehensive report of this act, preliminary experiments were conducted to investigate the intercorrelations between the aforementioned theoretical framework of the gesture/event perspective and the actual experience of electronic music.

5.1. Material and Method

The piece chosen for subject group studies is titled Birdsong, the second piece to come out of a tetralogy which explores evolutionary phenomena. Subsequent to its first round of listening, an analysis of the dynamics of the underwater, Birdsong engages with organic morphologies that transcend the surface of the ocean. The piece is an outcome of investigations on the characteristics of aquatic and avian sounds. Constituent sound designs of the macro-level structure of the piece follow the aforementioned gesture model in which sonic events denote events within the narrative at varying time-scales and texture formations. Gestural design was inspired by devices that served foreground functions (i.e. animal vocalizations) are shorter in duration and are aimed at stimulating different parts of the brain. From a continuous abstraction from background, while ground gestures (i.e. water sounds) operate at meso-scales to set semantic and spatial contexts.

The experiments were administered a single participatory time. Participants were not in any way influenced or informed regarding the piece. Each participant was asked to listen to the piece, prior to being provided with any information regarding the piece or achieving an initial listening experience that was least affected by experience. Each participant was asked to write down general impressions in free form, examples of which will be discussed in the following sections. The subsequent part involved a software-based test, during which the participants were able to listen to the piece in every second, and then type in random words he or she was able to identify, in order to acquaint the participant with the interface and monitor possible software and database malfunctions. Prior to the actual experiment, it was explained that the participant was expected to type in their impressions as to anything they would feel or imagine, instantaneously to its occurrence within the piece.

5.2. Analysis of Data

The subject profiles display a large diversity in musical backdrops as the group of 12 participants included composition professionals, music engineers and people with no prior training in sound or music. The participants input 224 descriptive terms, which average to 18.6 words per subject within the 408” duration of the piece. The responses varied from single words to up to 4-word segments, although no limitations were imposed during the explanation of the experiment. The responses were later on grouped into four general categories as to whether they described sound sources, scenes (locations), emotions, concepts or perceptual descriptors. Responses that could be prephrased by the phrase "sound of" were categorized as sound sources, sound sources mainly consisted of sound producing objects (i.e. insect) or events (i.e. creature moving). Several responses which did not describe a source but rather implied resonant environments as in "cave" and "underwater" was categorized as sound environments. Affctively-loaded inputs such as "anger" and "hope" were excluded from the study, as well as any set of inputs that did not exhibit direct sonic or emotional connotations but rather implied abstractions which could be related to the sounds, such as "star wars" and "language" were listed under conceptual descriptors. Sound descriptions of the sounds, such as "low" and "pan", were categorized as perceptual descriptors.

Figure 1 Real-time input distributions.

Certain responses that exhibit strong semantic similarities will be grouped together. The highest rating semantic groups were liquid sounds (i.e. water, liquid, waterfall, ocean), speech-like sounds (i.e. bird, dog, bat, mouse, amphibian creature) and sci-fi sounds (i.e. lightbeam, laser, ice/glass, reptile/amphibian creature). Notice that displayed high inter-participant occurrences at approximate moments in the piece were the “water”, “bubbles”, “laser”, “love”, “vocalization/speech”, “cave” and “swarm”; these responses indicate sounds which were specifically localized in a particular background or emotion.
ends of the musical background spectrum displayed strong similarities while the only instrumental music correspondences between compositional strategies and resulting listener experiences were highlighted. These results offered valuable insight as to how a composer's narrative concepts translated into a listener's perceptions. Further experiments will be conducted in reciprocity with compositional practice and theoretical discourse in order to explore the details of the communication between the composer and the listener. This investigation will establish a further understanding of the cognitive nature of the genre, and will bear strategies to harness the potential of this nature to a greater extent.

REFERENCES


1. Eventually this research resulted in my dissertation titled “Dirty Light. The application of musical principles to the organization of light as an extension of musical expression into the non-figurative visual realm”, that I completed at Brunel University London in 2010.
Leicester. He was interested in creating a visual counterpart to Conceptual Art, which was new at that time and widely discussed amongst his students. The material of The Sinking of the Titanic consists of various things that are related to the sinking of the luxury liner. Part of it consists of compositions that might have been played by the ship’s band the night of the sinking. Furthermore Bryars collected Morse codes that have been exchanged between the Titanic and other ships during the accident, as well as interviews with survivors and some other materials. Simplified, it can be said that the composition is a collage of these materials. However, interestingly the first audio version of the piece was realized in 1972, three years after the original version. The first version consisted of a visual presentation of the mentioned materials in the form of an exhibition. By choosing this form of presentation, Bryars intended to let the musical experience happen in the imagination of the audience.

All of these different elements on display had a potential acoustic reality. Instead of using them as concrete sound – as Bryars did in the later concert version of the work – in the 1969 version he opted for letting them manifest themselves acoustically in the inner ear of the beholder. Each viewer could therefore experience the material in a very personal version. Bryars thus succeeded in creating a straightforward musical realisation of some of the core ideas of Conceptual Art. From the point of view of visual art, Conceptual Art can be described as a non-retinal form of art, in the sense that artistic practice is not primarily interested in the materiality of a given object, but in its potential as a signifier in an expanded context. The retinal or visual aspect is thereby only a gateway into a mode of perception were a large part of the experience of a particular work of art happens in the imagination. The mental expansion of the art-object and the spinning out of ideas that it incites is often the most significant aspect of the work.

Marcel Duchamp coined the term non-retinal art and his so-called Ready-mades are often referred to as the first objects of Conceptual Art. Probably the most famous amongst them is Fountain (1917).1

In these experiments music has been played to test persons while gaps of silence have been cut in with a duration of 2-5 seconds. When the test persons were familiar with the music played, a rise of neuronal activity has been measured during those gaps, which was not present when the music was unfamiliar to the test persons. This has been interpreted that in the former case the music is filled in by the imagination of the listener.2

However, if the music is unfamiliar the listener cannot amend it. Hence, auditory imagery cannot take place and the neuronal activity remained low. This also shows that auditory imagery relies on the previous knowledge of the person. What has never been experienced cannot be imagined as a sound, no matter whether it is a piece of music, a particular sound or a foreign language. Even when we invent new sounds or compositions in our imagination, we draw on our experience and recombine it in various ways.

The interesting thing about auditory imagery is the multifaceted role it can play for the experience of music, no matter whether the music is actually sounding or imagined. It is the vehicle for many forms of musical experiences that take place in the inner ear. For instance, when listening to a cover version of a pop song auditory imagery plays a crucial part of the musical understanding. Many cover versions are based on the assumption that the listener knows the original version of the song. Their appeal is often based on detailed differences to the original version. If the receiver is not familiar with the original and therefore fails to recognize that a cover is being played, a significant part of the artistic intention cannot be understood. What actually takes place when we listen to a cover version is a constant comparison to the original, as if it was running as a parallel soundtrack in the imagination of the listener. This largely sub-conscious process is therefore an essential part of the musical experience.

4. EDWIN GORDON’S AUDITION

The comparison between sounds that are remembered and expected to those that are actually perceived is not something confined to the reception of cover versions but rather a general phenomenon when we listen to any familiar style of music. For instance, when an experienced listener hears the beginning of a classical piece of music, he or she might soon recognize it as a sonata. This entails the expectation that some form of development section will follow, as well as a recapitulation. Some composers, like for example Beethoven, have consciously played with such expectations and occasionally fooled the listener by composing pseudo-recapitations.2 Such tricks show that the composer expected a certain level of sophistication and expertise from his listeners. Why else would he want to surprise them with a pseudo-rcap, if he would not expect them to recognize such a cunning move? In order to distinguish such sorts of style-dependent and genre-dependent recognitions and anticipations from more ordinary types of auditory imagery, the term Audition comes in handy, which was coined in 1975 by the music educator Edwin Gordon. He explained in his own words:

…when you are audiating as you are listening to music, you are summarizing and generalizing from the specific music patterns you have just heard as a way to anticipate or predict what will follow.3

Edwin Gordon developed a specific learning theory on the basis of recognition and anticipation processes, which, in turn, relied on the agglomeration of a specific musical knowledge. It is implicit in this understanding of music education that memory plays an essential part when we listen to music. Whether or not expectations are fulfilled determines an important part of the musical

1 At the occasion of the Turner Prize 2004 this work has been selected as the most influential work of art of the 20th century, in a survey amongst artists and art critics http://news.bbc.co.uk/2/hi/entertainment/4059997.stm (14/06/11).
4 www.nature.com/nature/journal/v434/n7030/full/434158a.html (13.06.11).
5 At the occasion of the Turner Prize 2004 this work has been selected as the most influential work of art of the 20th century, in a survey amongst artists and art critics http://news.bbc.co.uk/2/hi/entertainment/4059997.stm (14/06/11).
8 All three examples are presenting an absence of performative actions, rather than an absence of sound. Nevertheless they address and question the traditional idea of silence.
9 At the occasion of the Turner Prize 2004 this work has been selected as the most influential work of art of the 20th century, in a survey amongst artists and art critics http://news.bbc.co.uk/2/hi/entertainment/4059997.stm (14/06/11).
experience. Whenever we recognize intrinsic references within a work of music, like for example the recombination of a particular musical motif, we drawing on the short-term memory that has been built up since the piece started. However, when we deal with the recognition of a genre, as in the aforementioned example with the classical sonata, we reach back to our long-term memory. In this case we are comparing what we are hearing with listening experiences that we have made in the more distant past.

Both forms of memory are an essential part of the listening process. They are musical experiences from the past. Triggered by an actual sound, they are not sounding themselves but they are added to the experience of the sound as a mental activity.

With Audiation, recognition and anticipation stem from a particular notion of musical culture. These processes are however almost the same as the inner ongoing comparison of a cover version with its original. In both instances inner listening processes are determining factors in how the music is perceived.

5. EXTRINSIC REFERENCES

The entire domain of inner listening processes becomes even more complex if we add extrinsic references to the equation. Extrinsic listening refers to all sorts of associative references entailed by sounds – references that point to things that lie beyond the objective material condition of a sound. We experience extrinsic listening for example if we recognize a locomotive in Pierre Schaeffer’s seminal work Cinq études de bruits. In the first of these études the sound of a train is used as abstract material for the composition. At the same time the sound’s origin is identified by the listener, which creates a link to the world outside of the intrinsic structure of the composition.

Another example of extrinsic reference is the use of the mandolin in Gustav Mahler’s 7th Symphony. The function of this instrument is more than just to add another colour to the orchestral apparatus. Its function is also to evoke a Mediterranean atmosphere since this is the culture and ambience this instrument is usually associated with.

A more unusual extrinsic reference is at hand with the example of Frédéric Chopin’s piano étude op.10 Nr.1. This piece opens a cycle of 24 études with large arpeggios in the C-major key that are spanning practically across the entire keyboard. Even though this is a genuine composition, Chopin is at the same time referring to Johann Sebastian Bach’s Prelude Nr.1 of the Well-tempered piano. It is no coincidence that this is another cycle of 24 pieces that is opened with arpeggios in the C-major key. Chopin deliberately started his cycle in a corresponding fashion to Bach, in order to show his deep admiration for his music. This is an entirely different sort of reference than the aforementioned ones as it can only be deciphered by the real connoisseur who has a very solid knowledge of piano literature. Such an extrinsic reference has resemblance to Audiation because it points to a concrete body of knowledge. Here, the borders between extra-musical references and musical literacy become blurred. Auditory Imagery, Audiation and extrinsic listening are therefore areas that cross into each other seamlessly.

6. THE VACUITY OF REDUCED LISTENING

The different sorts of non-cochlear listening that thus far have been discussed show that what our ear perceives is only part of what we hear. Our memory and imagination are constantly adding to what reaches our ears. Because of the constant activity of our inner hearing, we are incapable of hearing the sound ‘as such’, as every sound comes along like Duchamp’s Fountain: ready to be filled with associations and meanings according to a given context. Even when any sort of associative listening is actively suppressed, as Pierre Schaeffer proposed as part of reduced listening, no listener can create a cultural vacuum in himself or herself, which would be the precondition for an unbiased phenomenological perception. Schaeffer’s practicing of reduced listening was an attempt to purify music of non-musical elements. Non-musical sound was traditionally described as noise – an undesired sound in a particular context. With Schaeffer the non-musical is not located in the sounding material anymore, but inside the listener in the form of associative responses to sound. Reduced listening therefore describes a mental behaviour that is imposed on the listener, a disciplined way of hearing.

7. THE RELEVANCE OF EXTRINSIC REFERENCES FOR ELECTRO-ACOUSTIC COMPOSITION

The blurring of the edges between music and environmental sounds may eventually prove to be the most striking feature of all twentieth-century music.

This quote by Raymond Murray Schafer implies that music has increasingly taken on qualities that are referential. All environmental sound communicates something about itself and refers to its cause or source. If music becomes more similar to environmental sound it therefore also gains a higher amount of referential qualities.

I would argue that an awareness for sounds’ referential potential, which entails inner forms of listening, is even more relevant for artists working in the field of electro-acoustic music, than those working with traditional instruments. The computer is a tool, which is largely devoid of medium specificities and is therefore often referred to as a post-medium device. This means that with the computer we can create sounds that do not reveal their source in such an obvious way as for example the sound of a violin points to a strung wooden sound-board played with horse-hair. Not only does the computer offer practically inexhaustible possibilities for the creation of sounds, it also enables the artist to artificially add layers of associative meaning to sounds that point beyond their immediate sonic condition. To give a simple example, the Scottish electronica duo Boards of Canada early on specialized in the use of sounds that evoke a nostalgic “retro” atmosphere. This is achieved by manipulating sounds to give them a sort of “patina” that makes them sound old. Post-medium devices make it possible to artificially create or at least apply such patinas and add them to any sound.

Such additions of extra musical references constitute a level of musical organisation which is not accessible to instrumental composition in such a nuanced way. When compared to traditional categories of musical organisation, a reflected application of sounds with referential qualities opens a whole new way of thinking about sound and its organisation. This is something which has already for quite a while been embraced by video artists, sound artists and pop musicians but which is only slowly finding its way into the discourse of more academic electro-acoustic music. In the latter, associative listening processes have so far mainly been discussed as the enemy of the aforementioned Schaefferian reduced listening. For a full assessment of the artistic potential of sounds’ extrinsic qualities, a thorough understanding of the various forms of inner-hearing processes can provide a fruitful point of departure.

1 See: Barbanti, Roberto, Les Origines des Arts Multimédia, Nîmes: Leduc editions, 2009

2 For an analysis of Bill Viola’s use of sound in some of his video installations, see: Rogers, Holly, Visualizing Music, Saarbrücken: Lambert Academic Publishing, 2010, p.144-191

3 See: Labbé, Brandon, Acoustic Territories/Sound culture and Everyday Life, New York: Continuum, 2010


5 More recently such an approach has been discussed for example by James O’Callaghan in: O’Callaghan, James: “Soundscape Elements in the Music of Denis Smalley: negotiating the abstract and the mimetic” in Organised Sound 16(1), Cambridge University Press, 2011, p.54-62

6 Several writings by US-West-Coast composers who were involved in the World Soundscape Project are also offering a starting point. See: Truax, Barry: Acoustic Communication, Norwood: Ailes, 1984


8 More recently such an approach has been discussed for example by James O’Callaghan in: O’Callaghan, James: “Soundscape Elements in the Music of Denis Smalley: negotiating the abstract and the mimetic” in Organised Sound 16(1), Cambridge University Press, 2011, p.54-62

9 Several writings by US-West-Coast composers who were involved in the World Soundscape Project are also offering a starting point. See: Truax, Barry: Acoustic Communication, Norwood: Ailes, 1984

10 Other relevant writings are Robert Morris: Soundsuits, Townes: Citizen of the Heaven, Christensen: Music as a Soundscape, Mulligan: Music and the Soundscape, Wasserman: The Ecologies of Soundspace, etc.

11 This argument strongly contradicts the notion of many trends in more recent electronic music that deliberately focus on sound as a medium that is believed to be devoid of any referentiality. See Demers, Joanna: Listening through the Noise, New York: Oxford University Press, 2010, p.69-113


8. CONCLUSION
With this paper I would like to point out that musical experience is only partly determined by the actual sound that is transmitted. Various forms of mental activities – like interpretations, remembrances, anticipations, or recognitions of intrinsic and extrinsic references – are playing as much their part in the hearing process as the sonic condition of the sound. The artistic application of referential qualities to sound is something what is especially relevant for, and applicable in electro-acoustic music. Even though it has already been explored and applied, theoretically it has not yet been fully assessed and reflected. A thorough analysis of inner listening processes would be a good starting point. Even though we have seen that musical experience is possible without the occurrence of any sound at all, it would be questionable to conclude that music generally does not rely on acoustic vibration. However, it does not depend on it in such an exclusive way as it is usually assumed. Inner hearing processes are abundant and they do not fully depend on sonic stimulation. Therefore we never hear only what the ear perceives.

From this perspective a number of other interesting questions emerge, as for example a definition of music that accounts for the fact that music is not necessarily sonic. Already more than 50 years ago Robert Ashley wrote:

It seems to me that the most radical redefinition of music that I could think of would be one that defines ‘music’ without reference to sound.1

Hence, the idea of music as something that can be sensed and experienced without being audible is not new at all. In fact, it is as old as the culture of Western music. With the concept of a Harmonia Mundi the Pythagoreans introduced a model of thinking about sound where the musical experience does not rely on its acoustic presence.2 An inter-sensory and inner-sensory conception of music was therefore placed in the very cradle of Western culture.

9. REFERENCES

2 Haase, Rudolf, Geschichte des Harmonikalnen Pythagoreismus, Vienna: Verlag Elisabetha Lafite, 1969

ON THE NON-COCHLEARITY OF THE SOUNDS THEMSELVES
Dr Malcolm Riddoch
West Australian Academy of Performing Arts
Edith Cowan University

ABSTRACT
What is non-cochlear sound? This open question is followed by way of an initial exploration of the psychophysiology of audition. Non-cochlear in sound is posited firstly in terms of synaesthesia and the skin and body cavity reception of infrasonic and low frequency sound waves. The auditory imagination is a further example that can produce a perception of sound without any direct acoustic stimulation of either the ear or skin and body. However, one’s imagination still retains a relation to the sounds of the world we live in. From a phenomenological perspective this worldly relation is a fundamental characteristic of sound as something that is heard. On this basis the causality associated with empirical accounts of auditory perception as a product of biological processes are contrasted with an interrogation of sound qua sound. It is posited that the sounds themselves are non-cochlear in the sense of being non-physical phenomena disclosed in the lived experience of hearkening to the meaningful sounds one hears in the world.

1. INTRODUCTION
“Only he who already understands can listen (zuhören)!” Martin Heidegger, Being and Time

1.1. An Empirical Perspective
What is non-cochlear sound? This question is presented in the negative sense, ‘non’ being the negation of the adjective ‘cochlear’, meaning an absence or lack of the cochlea in sound. In order to answer this, one would then presumably first have to ask - what is cochlear sound? From an empirical perspective this question asks about the role of the cochlea in the perception of sound, for which we have a reasonably thorough psychophysiological understanding. From acoustic energy propagating through air and other media, reflected via the pinnae through air and other media, reflected via the pinnae to the coiled fluid filled bag of the cochlea, its tapered basilar membrane and hair cells – such is the cochlear perception of actual sounds. The physical complexity of this neuronal network is gigantic as each neuron can have from 1-100 axonal connections firing off an electrochemical impulse 40-1000 times per second. The axons themselves are very dynamic and can grow new connections or whither away depending on the signals they receive in a neurological network of around 55,000,000,000 neurons.

We also know that the auditory pathways in the human brain are especially adapted to organized sounds, exhibiting ‘pattern sensitivity’ to different sequences and ‘sensitive tuning’ to different discrete signals.3 This sensitivity is exhibited even in newborns indicating that an attunement to organized sound is an evolutionary adaptation in the human species. Furthermore, the brain exhibits plasticity in frequency and pattern discrimination such that aural training can improve sensitivity to the trained frequencies and produce measurable cortical changes in the brain.4 We are not only born with a developed auditory cortex but the brain continues to adapt to our evolving soundscapes throughout life.

It would seem then, at least from an empirical perspective, that we have a preliminary definition of ‘cochlear sound’. It is the perceived sound associated with the kinetic energy vibrations within the cochlea that produce electrochemical signals in the brain. Just precisely what this ‘association’ is, between the perception of actual sounds and the evolved, biological mechanisms of our audition, remains somewhat ambiguous. One might be tempted to simply conclude that all perceived sound is merely a subjective psychological affect caused by neuronal in audition. Yet where is the ‘cochlear sound’ in this biological and kinetic mechanism?

1.2. Neurological Sound
After the cochlear structure of course there is the electrochemical propagation of binary signals triggered by the hair cell excitation of the auditory nerve through to the brain’s auditory cortex as well as the cerebrum, limbic system and beyond. Audition being a whole brain phenomenon, these acoustic vibrations traveling via the cochlea and their approximately 3500 hair cells produce highly complex neuronal excitations that are intimately related to the perception of actual sounds. The physical complexity of this neuronal network is gigantic as each neuron can have from 1-100 axonal connections firing off an electrochemical impulse 40-1000 times per second. The axons themselves are very dynamic and can grow new connections or whither away depending on the signals they receive in a neurological network of around 55,000,000,000 neurons.

It would seem then, at least from an empirical perspective, that we have a preliminary definition of ‘cochlear sound’. It is the perceived sound associated with the kinetic energy vibrations within the cochlea that produce electrochemical signals in the brain. Just precisely what this ‘association’ is, between the perception of actual sounds and the evolved, biological mechanisms of our audition, remains somewhat ambiguous. One might be tempted to simply conclude that all perceived sound is merely a subjective psychological affect caused by neuronal in audition. Yet where is the ‘cochlear sound’ in this biological and kinetic mechanism?

1 From Weinberger, pp. 88-95 [2]
2 From section 34, “Being-there and Discourse. Language”, p. 208 [1] on which the notions of ‘hearing’, ‘hearkening’ and ‘listening’ in this paper are based.
3 Cf. Pante et al. [3]
4 In Weinberger, pp. 88-95 [2]
excitation. However, the precise character of this presumed causality and its ‘effects’ still remains ambiguous. Any attempt to define this ambiguous relation between naive psycholinguistics and its inherent to seemingly logical assumptions. Nevertheless, given this ambiguity, we can still proceed in a provisional sense to ask: What then might be a ‘non-cochlear’ sound be?

2. WHAT IS NON-COCHLEAR SOUND?

Following on from the above empirical definition of a ‘cochlear’ sound one might say that non-cochlear sounds are perceived sounds associated with the excitation of the auditory cortex in the human brain by means other than cochlear vibrations transmitted through the hair cells to the auditory nerve. Three such cases of non-cochlear excitation should suffice to demonstrate this particular psychophysiological phenomenon.

2.1. Synaesthetic Sound

The first example deals with synaesthesia or the perception of one sense via the stimulation of other senses. Most commonly this presents itself in the form of black printed characters appearing coloured, however there is another form of synaesthesia for which colour is heard as sound. A Miami University study of 572 synaesthetes also found that 1% of the subjects heard sound via smell, taste or touch4. The mechanism for this effect is thought to be some form of cross talk in the brain between auditory and other sensory pathways.

2.2. Infrasonic Sound

A related but somewhat more problematic example is that of infrasonic or very low frequency sound waves (generally below 20Hz) that can be felt via the body rather than the ear. This is problematic in the sense that the bodily perception of very low frequency sound may not be considered as a form of hearing. At very low frequencies, around 16-18Hz or lower depending on the subject, cochlear auditions fails to detect tonal information at all and sensitivity to the sound falls away the lower the frequency. This is simply a physical limitation of the biomechanics of the middle and inner ear at very low frequencies5. Amplitude thresholds for perception vary depending on age and other physiological factors but there is a crossover in perception between the auditory pathways of the cochlea and the bodily sensation of such low frequency sound. Various studies have pointed to the Merkel cell, Meissner corpuscles and Pacinian corpuscles in the body’s largest organ, the skin, as possible receptors for low frequency sonic vibrations6.

Alongside these skin receptors the chest cavity can resonate from around 80Hz and lower, depending on one’s physical build, and thus also plays a role in low frequency, non-cochlear perception of sound. These effects are commonplace in loud, amplified music venues and are especially emphasized in contemporary electronic music genres such as Hip Hop or Techno as well as in various forms of experimental music, sound art and so on. One has to feel the bass, and it is an embodied, somatic and synaesthesic auditory experience.

Such effects can also be felt in the ultra low frequency range, which are meaningful vibrations that we hear but not as sound. This is especially manifest in the imperceptible rumble of machinery that can cause various public health and safety problems for workers. The rumble of the car in the street and the voice of a dear friend7. The mechanism for this effect is thought to be some form of cross talk in the brain between auditory and other sensory pathways.

2.3. Auditory Imagination

The third example of non-cochlear sound in this empirical sense is demonstrated by one’s own auditory imagination. The remembrance of past events and their signatory sounds, auditory dreams or hallucinations (including hypnotagogic and hypnopompic experiences), the recollection of a performance via one’s own personal or cultural performative work of music, and for composers the contemplation of a new work and its possible auditory aesthetic: All these works of the imagination can involve the perception of sound and the excitation of the auditory cortex without cochlear input.

The inner world of one’s own imagination is replete with auditory perceptions that have no direct cochlear or somatic input although undoubtedly they have evolved along with all our other capacities in the brain. This everyday fact implies that the sounds themselves are not necessarily an artefact of cochlear or even acoustic excitation, although the sounds of our imagination are largely derived from those of the world we live in. Even the fantastic sounds heard in hallucinations and dreams with seemingly no worldly counterparts can be perceived by us to be heard by others than the worldly sounds we hear. That is, they derive their unique strangeness only in contrast to and in the context of the worldly sounds of our everyday experience.

This relation between the sounds themselves, as in the perceived sounds we actually hear, and both the world we live in as well as our auditory brain functions, is an interesting conundrum. Where does the sound, non-cochlear or otherwise, actually occur? Is it in the brain, in the dynamic cascade of electrochemical impulses that never ceases until death? Or do sounds occur in the world, in a sense yet to be fully explicated? And what is the logical difference between these two propositions?

3. ON THE MEANINGFULNESS OF SOUND

3.1. Sound as Concept

An interesting example of this ambiguity (at least interesting for myself as a philosopher and musician) and an extrapolation of this notion of non-cochlear-ity of sound can be found in John Cage’s composition 4’33”. In the performance of this work the sounds themselves are called to presence simply by their absence. In place of all the music one’s musical imagination is challenged by a void, by a silence that is filled with worldly sounds, and by a musical relation that reveals itself as a tension between audience and performers and the mutual space they occupy. The composition is non-cochlear due to the absence of any organized sounds and yet the silence is not silent, one still hears. More to the point one hears! Attention is given to both the absence of the work and the loudness of the silence that fills the musical void.

Beyond this apparent silence of the work one’s attention is drawn to the musical relations at work in a traditional performative context between audience and performers on stage.

Here we have come to a fourth possible definition of non-cochlear sound, via the imagination, in terms of the conceptual or meaningfulness of sound. The sounds themselves - whether associated with cochlear vibrations, with synaesthesics or with the imagination - can direct our attention to something other than the sound. Such a conceptual hearing of a work’s sound, is especially adept at this form of sonic manipulation of the imagination and its relation to the world and perhaps it is from sound art that this non-empirical notion of non-cochlear sound might best be demonstrated. The question here has become: What is the relation between the worldly meaningfulness of the sounds themselves and the biological basis of sound?

3.2. Sound as a Worldly Phenomenon

One possible answer might be to postulate a chain of events starting with the electrochemical stimulation of the auditory nerve, resulting in a chain of neuronal activity which in turn is identified in terms of other sounds with various aspects of what we cognize as the world, resulting in a chain of electrochemical events to cochlear and somatic apparatus that allows us to piece together the flux of perceptions that makes up our everyday world. What is it that we actually hear and understand? Is it the sounds producing raw noises and tone complexes? Or is it the meaningful sounds themselves? Does one hear the familiar melody first or a pattern sequence of tones that are identified after the fact as this or that popular tune?

From a phenomenological perspective what one first hears is the melody as it flows and never a series of tones, noises and abstract timbres1. One hears Wagner’s symphonic crescendo, the wind in the trees, the car in the street and the voice of a dear friend2. Sounds, whether associated with cochlear vibrations or not, are always in the first instance meaningful sounds.

Even so-called noise music is precisely as meaningful as noise music, as an attempt to negate or transcend the unbal, harmonious and rhythmic limitations of the Occidental tempered scale and metric rhythm tradition from which the genre largely derives its reversed, mirrored and inverted image. Which is to say, the concept of noise music is defined by the world in which it is perceived as noise. Likewise, if one awakes suddenly at night on hearing an inexcitable sound, its inexplicability is meaningful only in the context of everything else that is happening around them. This is especially so in the case of musical perception where the sounds themselves are lifted up from the background noise and brought to presence as the musical work - unless of course one is bored with the performance.
the music in which case attention may lapse and one hearkens to the sound of another drum, perhaps one’s own thoughts or perhaps the hum of the stage lights and so on.

4. BACK TO THE SOUNDS THEMSELVES

But where has this particular argument taken us? From the empirical notion of non-cochlear sound in terms of synaesthesia and infrasonic vibrations to sounds of the imagination with no immediate acoustic input; then to sound in terms of artistic conceptuality and on to sounds as first and foremost meaningful, worldly phenomena. Each of these art points us in the direction of the world within which we hear and hearken then is this merely an artistic contrivance or does the world that art discloses bring us to a far more fundamental phenomenon in non-cochlear sound?15

The fact that the sounds we hearken to are already meaningful would indicate that the conceptual in sound is not merely an afterthought, an artistic abstraction, or a subjective, psychological construction. The meaningfulness of what we hear is a fundamental aspect of the sounds themselves as we encounter them in the first instance. The non-cochlear conceptuality of sound in this everyday sense is thus both an abstract and yet also a most concrete phenomenon associated with hearing sounds in our world.

4.1. Scientific Limits of Sound

Yet what is the relation of the actual worldly sounds of our lived experience to the psychophysiological processes associated with our everyday perceptions? Which is to restate our initial question: Where is the sound in a (non)-cochlear vibration and its neurological effects? In a colloquial sense one might claim that self-evidently sound is perceived ‘in’ the brain or ‘in’ the mind and leave it at that. However, a practitioner in sound for whom the sounds themselves are the very medium of their craft may not be satisfied with such an easy answer. Of course there are no worldly sounds ‘in’ a brain for the brain is quite simply a biological network of neurons floating in cerebrospinal fluid in a skull.

Also, at least from a phenomenological perspective, there are no sounds ‘in’ a mind for the mind or ego itself is not a thing, a receptacle, but rather it is an ongoing process of perception and reflection forming the lived experience of one’s own being in this world. Which is to say, sounds are heard in the world we already live in and understand in one sense or another – sounds occur in the world.

Furthermore, the sciences (biology, natural physics or chemistry and so on) still have as yet been incapable of demonstrating a causal mechanism linking our neurological processes with the supposed subjective effect - the world of our perception. There are of course concrete associations between our biological organism and the sounds themselves as evidenced in the medical sciences, but there is no empirically verifiable mechanism for how electrochemical patterns in the brain actually become something heard.

The strictly empirical sciences by definition deal with objectively verifiable physical phenomena and the mathematically calculable data associated with these. Other phenomena such as the everyday world of our lived experience and its ongoing flux of perceptions are by definition ‘subjective’ and not directly amenable to empirical analysis. Science, from this perspective, is a discipline the limits of which are clearly defined by its physical scope. All other non-physical phenomena such as the sounds themselves are thus beyond the scope of strictly empirical science. Put another way, there is no empirically verifiable causal relation but rather a strictly associative relation between physical processes such as neuronal activity and the perceptions associated with those processes.

4.2. The Question of Sound’s Causality

Given these empirical difficulties it would be easy to propose that there is therefore no such thing as a cochlear sound in our demonstrable empirical time, there are only in the first instance the sounds themselves we hear and hearken to. By simple inference all sound, as something heard in the world, is therefore non-cochlear (or more precisely a non-physical phenomenon).

One might however object that this conclusion regarding the non-cochlearly of sound is itself merely an empty exercise in semantics, and if we were to leave the matter there I would tend to agree. Yet what is at stake in this argument is not merely a specific interpretation of the terms “non-cochlear” and “sound” but also the notions of causality that inform our everyday understanding of sound in general and the ways we talk about it.

For those of us with a scientific background, including those of us schooled in a modern education industry, it can be easy to assume a form of psychological causality in regard to the causal relation between scientific reality and the phenomenal world it attempts to describe, the world of lived experience. Thus it might appear self-evident that the perception of sound occurs ‘in’ the brain or mind and is caused by electrochemical stimulation. Such a basic presupposition can be the cause of a good deal of confusion when talking about sound, for sound in this case is explained in terms of something other than the phenomenon of sound itself. In fact here the sounds themselves are relegated to an ‘inner’ sensitivity that remains mysterious while the biological ‘cause’ assumes priority in terms of understanding those very sounds.

While an understanding of the psychophysiological processes associated with audition is obviously useful, such as in the psychoacoustics of digital reverber modelling, it does not necessarily require a psychologistic worldview. If as a musician – composer or performer alike – one wishes to understand sound in terms of the sounds themselves, then it might also be useful to deconstruct one’s own presuppositions about sound and its relation to the physical and perceptual phenomena associated with hearing in the first instance.

From this critical perspective the term ‘sound’ stands for an open question disclosed in the lived experience of hearing and hearkening. The causality of sound is here proximally related to one’s directed attention (intentionality) within the world, or in other words, one hearkens and thus hears. This is a very different relation to causality than the psychologistic notion of the biological provenance of sound, and it is a relation that perhaps opens up the possibility of talking about sound qua sound.

Such an open perspective also problematizes the causal relation between sound and our biology. For example, one could say that in an evolutionary sense we do not hear because we have evolved the biological mechanisms for audition. Rather, precisely the opposite, we have evolved the biological mechanisms for audition because we already hearkened to sound.

Why else would the human species have developed a complex auditory cortex if not for the evolutionary advantage of hearing and hearkening to the sounds themselves?

One has to first already have come to understand what sound is by having heard and by hearkening to sounds in our everyday world. It is not that sound causality is itself merely an empty exercise in semantics, but rather the artwork “opens up a world” in an originary sense, cf. Heidegger’s ‘Origin of the Work of Art’ p. 169 [10].

15 The function of art in this sense is not as mere entertainment or craft, rather the artwork ‘opens up a world’ in an originary philosophical sense, cf. Heidegger’s ‘Origin of the Work of Art’ p. 169 [10].

5. REFERENCES


GENERATIVE SOUND ART AS POETIC POETRY FOR AN INFORMATION SOCIETY

Alice Eldridge
Centre for Creative Research into Sound Arts Practice
University of the Arts, London, UK
alice@eclla.org

Abstract

Recent conversation in both the sonic and generative art communities envision a practice which engages with both formal and cultural or conceptual concerns. At the same time there is a sense amongst practitioners in the generative arts community that there is room to develop the methods of the practice. This paper sketches a simple, practical approach to generative sound art which draws from philosophical observations of the emerging information society and implements a sound-based generative scheme as a simple, literal illustration of a practice which looks outward to society and inward to the materials of its practice. It is suggested that a digital sound art of this nature has potential as a discourse for contemporary society, a poietic playground for coming to terms with the implications and challenges of the information age.

1. INTRODUCTION

Seth Kim-Cohen’s recent vision of a conceptual sonic art which engages “both the non-cochlear and the cochlear, and the constituting trace of each in the other.” [15], p.xxi, resonates loudly with an ongoing debate in the generative and the sonic arts. This paper introduces a simple, practical approach to generative sound art which draws from philosophical observations of the emerging information society and implements a sound-based generative scheme as a simple, literal illustration of a practice which looks outward to society and inward to the materials of its practice. It is suggested that a digital sound art of this nature has potential as a discourse for contemporary society, a poietic playground for coming to terms with the implications and challenges of the information age.

These debates in both the sonic and software arts in- prise considerations of how practice at the interface, a generative sound art, might approach this vision. This paper offers a simple, practical illustration. Section 1.1 outlines Whitehead’s system story. The exposition of this critical device is an attempt to open up the formal, aesthetic concerns of the generative art community to a broader cultural context, so bridging Cramer’s formalist-culturalist divide. Section 1.2 examines current issues in generative practice. It is suggested that some of the problems currently identified in the field may be a symptom of importing methods, models and their attendant perspectives from the engineering sciences that could be fielded from more exploratory, experimental approaches to design and implementation of generative systems. Inspiration for an alternative is drawn from Luciano Floridi’s observations of the impact of digital technologies in the broader information society. Section 2 introduces a simple class of generative system inspired by Floridi’s concept of ontological friction. This is presented as possible alternative to the standard practice of algorithmic composition. Rather than specifying the numerical outputs of a formally imple- mented algorithm, the generative process is constituted in the sound in which it is also manifest. In section 3, it is proposed that the poietic and creative nature of computer music, and generative sound art in particular, has unique potential as a discourse for coming to terms with the challenges of contemporary society.

1.1. System stories and critical generativity

The potential for software art, in general, to represent and interrogate the cultural complexities of contemporary society has been suggested in the past (e.g. [17]). In adopting formal, complex systems as a basic generative tool, software art has the potential to convey not only an image of cultural situations, but more powerfully, to present a systemic abstraction, or model. “Abstract generative art”, suggests Whitehead “performs cosmogenesis: it brings forth a whole artificial world, saying here is my world and here is how it works.” [21], p.5. In collapsing the concept notion and execution, such works are in quite a unique position to illustrate, explore and critique “how it works”. In Kim-Cohen’s proposed terms, we appear to be tooled up and ready to talk about.

Whitehead observes, however, that this potential remains largely unfulfilled. Surveying a number of visual artworks he suggests that where complex systems are deployed, the predominant interest is in their formal, generative potential, an inward-looking, utilitarian and non-reflexive concern with the formal form of the emergent structure. He sug- gests that what is needed is a critical approach to genera- tive art, in order to open up these formal systems to dis- cussion and critique, drawing out relations between their internal workings and the outside world. Whereas previous- ous discourse focused on the materials and process (e.g. [4]), Whitelaw argues that a culturally-relevant critique might need necessarily to look into the formal systems themselves, rather than their sensory outputs.

Drawing from the way in which Artificial Life simu- lations have been critiqued in the humanities (e.g. [12], [14]), Whitelaw develops the system story as a critical de- vice. For Whitelaw the ‘system’ about which the story is told very specifically refers to the abstract, formal struc- ture(s), the objects, relations, actions and processes of the formal system, as distinct from either the language-specific text in which it is implemented, or it’s material (sonic, visual, sculptural etc.) manifestation. The system story then, is a retelling of the narrative of the entities, rela- tions, ontologies and processual structures of the software system. The system story can be used to engage with the formal object and draw out its implications. It provides a means to “connect - critically, prospectively, specula- tively - entities and relations within the system, with en- tities and relations outside it.” [21], p.3. The cultural cri- tique afforded by system story is the bridge he builds in an attempt to span Cramer’s formalist-culturalist divide.

This feeling that there are as yet unexplored opportuni- ties for contemporary sonic and software practices is also evident amongst practitioners in the generative art community. It has recently been suggested that it is time to develop our generative art methods “Generative art must do more than simply implement formal systems imported from the sciences.” [7], p.18. The recent population explosion in artist and programmer communities (and the recent interest in generative software art) has created lively communities around emergent structures or behavioural dynamics to a particular emergent structure. This feeling that there are as yet unexplored opportuni- ties for contemporary sonic and software practices is also evident amongst practitioners in the generative art community. It has recently been suggested that it is time to develop our generative art methods “Generative art must do more than simply implement formal systems imported from the sciences.” [7], p.18. The recent population explosion in artist and programmer communities (and the recent interest in generative software art) has created lively communities around emergent structures or behavioural dynamics to a particular emergent structure.

2. Beyond Algorithmic ready-mades

This paper sketches a simple, practical illustration of generative sound art of this nature. It is suggested that generative sound art might approach this vision. This paper offers a simple, practical illustration. Section 1.1 outlines Whitehead’s system story. The exposition of this critical device is an attempt to open up the formal, aesthetic concerns of the generative art community to a broader cultural context, so bridging Cramer’s formalist-culturalist divide. Section 1.2 examines current issues in generative practice. It is suggested that some of the problems currently identified in the field may be a symptom of importing methods, models and their attendant perspectives from the engineering sciences that could be fielded from more exploratory, experimental approaches to design and implementation of generative systems. Inspiration for an alternative is drawn from Luciano Floridi’s observations of the impact of digital technologies in the broader information society. Section 2 introduces a simple class of generative system inspired by Floridi’s concept of ontological friction. This is presented as possible alternative to the standard practice of algorithmic composition. Rather than specifying the numerical outputs of a formally implemented algorithm, the generative process is constituted in the sound in which it is also manifest. In section 3, it is proposed that the poietic and creative nature of computer music, and generative sound art in particular, has unique potential as a discourse for coming to terms with the challenges of contemporary society.

1.1. System stories and critical generativity

The potential for software art, in general, to represent and interrogate the cultural complexities of contemporary society has been suggested in the past (e.g. [17]). In adopting formal, complex systems as a basic generative tool, software art has the potential to convey not only an image of cultural situations, but more powerfully, to present a systemic abstraction, or model. “Abstract generative art”, suggests Whitehead “performs cosmogenesis: it brings forth a whole artificial world, saying here is my world and here is how it works.” [21], p.5. In collapsing the concept notion and execution, such works are in quite a unique position to illustrate, explore and critique “how it works”. In Kim-Cohen’s proposed terms, we appear to be tooled up and ready to talk about.

Whitehead observes, however, that this potential remains largely unfulfilled. Surveying a number of visual artworks he suggests that where complex systems are deployed, the predominant interest is in their formal, generative potential, an inward-looking, utilitarian and non-reflexive concern with the formal form of the emergent structure. He sug- gests that what is needed is a critical approach to genera- tive art, in order to open up these formal systems to dis- cussion and critique, drawing out relations between their internal workings and the outside world. Whereas previous- ous discourse focused on the materials and process (e.g. [4]), Whitelaw argues that a culturally-relevant critique might need necessarily to look into the formal systems themselves, rather than their sensory outputs.

Drawing from the way in which Artificial Life simu- lations have been critiqued in the humanities (e.g. [12], [14]), Whitelaw develops the system story as a critical de- vice. For Whitelaw the ‘system’ about which the story is told very specifically refers to the abstract, formal struc- ture(s), the objects, relations, actions and processes of the formal system, as distinct from either the language-specific text in which it is implemented, or it’s material (sonic, visual, sculptural etc.) manifestation. The system story then, is a retelling of the narrative of the entities, rela- tions, ontologies and processual structures of the software system. The system story can be used to engage with the formal object and draw out its implications. It provides a means to “connect - critically, prospectively, specula- tively - entities and relations within the system, with en- tities and relations outside it.” [21], p.3. The cultural cri- tique afforded by system story is the bridge he builds in an attempt to span Cramer’s formalist-culturalist divide.

This feeling that there are as yet unexplored opportuni- ties for contemporary sonic and software practices is also evident amongst practitioners in the generative art community. It has recently been suggested that it is time to develop our generative art methods “Generative art must do more than simply implement formal systems imported from the sciences.” [7], p.18. The recent population explosion in artist and programmer communities (and the recent interest in generative software art) has created lively communities around emergent structures or behavioural dynamics to a particular emergent structure.
of form or behaviour is preconceived and the genetic op-
plications. The ubiquity of ICTs in many areas of society,
lighting both the metaphysical and practical, ethical im-
plications of the given sonic material with an apparent intentional-
ity that belies the simplicity of the generative process. In
Sprids, the harmonic simplicity of the original drift develop-
s a coherent harmonic progression as each drift triggers a
change in rate according to its harmonic content, alter-
ing the pitch of the subsequent drips. Over time, the long
term structure (albeit musically trivial) unfolds according to
the spectral content of individual sound events.
In Balckridr there is a basic (and comic) exposition of
the gestures contained in the original recording. This
gesture exposition would be effective in a live interac-
tive setting. Swpong provides a short excerpt from initial
experiments with implementing the algorithm and also
exploring the gestural combinations and their generative
potential. In performance setting, the self-directed exploration of material has
the potential for a lively trading of gestures between instrumen-
tal performers and digital components of a performance system.

2. SELF-OBSERVING SYSTEMS

Self-observing Systems is an ongoing project which explores
generative sound art methods. The concern in the current study is the construction of a simple generative
mechanism which exhibits an ontological alignment com-
parable to that between ‘processor’ and ‘processed’ as
seen in Infosphere at large, a system which relates a story about the world, through the material of sound. This is
approached by implementing a simple system in which the generative process is constituted in the sound.

2.1. A self-determined sample player

The illustrative mechanism is based on a sample player. This familiar unit generator, which is the digital homo-
logue of various physical music players, represents the ‘processors’ in Floridi’s analysis and caricatures the im-
 pact of the de-physicalisation of devices more broadly. The sample player is furnished with a listening module. Rather than playing in a linear fashion, its playback rate (and direction) are determined by features of the audio it plays.

2.1.1. Implementation

The system was built in beads and comprises a variable rate sample player, listener module and short audio files
as shown schematically in Fig. 1. The sample is played at rate, R. When an onset is detected, absolute playback
rate is updated as a function of the most dominant fre-
quency in the current audio buffer and its direction reversal. Noise is added by means of a probabilistic update
regime. The probability is inversely proportional to the current rate. This balances the effects of playback rate: at slower speeds, significant audio features will be further apart in time (and less prominent). See Fig. 2 for details.

The listener module is an onset detector comprising the following analysis chain: short frame ‘Wienmeyer, fast fourier transform, peak detector and spectral difference measure. The peak detector followsthe algorithm described in [6]. In this implementation the audio file is not overwriten.

A resynthesis module is included for ‘decoration’ (it plays no functional role in the process), highlighting the
moments of change. When an onset is detected, an osc-
cillator bank is triggered, creating a complex chord. The frequency, duration and amplitude of each of N oscillators
is set according to the frequency and strength of the corresponding first N partials in the current buffer. In this
implementation, N = 10.

The basic design principle adopted here is therefore one of ciruc-
lar causality. This is nothing new. The approach lies at the heart of the cybernetic enterprise [20]. Its generative and interactive potential was evidenced in early electronic pieces (consider e.g. Gordon Mumma’s ‘Cybernetic’ Horn piece, Horseplay (1967) and it has experienced some-
what of a revival in recent years, following Dr Sciop’s Audible Ecos-
system Interface series [5].

2.2. Sonic examples

The system is intentionally playful and lively and this is reflected in the output. A range of samples containing varying degrees of harmonic and gestural complexity were explored. The examples given here include a dripping tap, Sprids, a field recording of blackbird song, Balckridr and a live example with pizzicato ‘cello, Swpong.

What we hear is an engaging and active exploration of the given sonic material with an apparent intentional-
ality that belies the simplicity of the generative process. In Sprids, the harmonic simplicity of the original drift develops a coherent harmonic progression as each drift triggers a change in rate according to its harmonic content, alter-
ing the pitch of the subsequent drips. Over time, the long term structure (albeit musically trivial) unfolds according to the spectral content of individual sound events.

In Balckridr there is a basic (and comic) exposition of the gestures contained in the original recording. This
gesture exposition would be effective in a live interac-
tive setting. Swpong provides a short excerpt from initial experiments with implementing ‘cells’. In performance setting, the self-directed exploration of material has the po-
tential for a lively trading of gestures between instrumen-
tal performer and digital components of a performance system, an aesthetic design principle which guides many
electro-acoustic improvisation systems (e.g. [19]).

2.3. Other self-observing systems

The principle of self-observing systems is open ended. A
further hypothetical example is outlined below.

---

*The basic design principle adopted here is therefore one of circu-
lar causality. This is nothing new. The approach lies at the heart of the cybernetic enterprise [20]. Its generative and interactive potential was evidenced in early electronic pieces (consider e.g. Gordon Mumma’s ‘Cybernetic’ Horn piece, Horseplay (1967) and it has experienced some-
what of a revival in recent years, following Dr Sciop’s Audible Ecos-
system Interface series [5].

**beads** is a Java library for real time audio: www.beadsproject.net
Self-directed feedback circuits illustrates how a simi-
lar generative process can be constituted across all
levels of a digital audio system, extending out beyond the con-
fines of a formal algorithm and digital representation of
sound through the DAC, speakers, room, mic and back
again. It explores ways of auto-maintaining and directing
the Larozen effect to create complex, adaptive resonances.

In an analogue system, characteristics of feedback arte-
facts are determined by the frequency response of the au-
dio system (mic and speakers) and the distance between them.
They can be modelled digitally using delay lines. In a simple
system, the effective gain can be man-
gaged by implementing a proportional control algorithm
(such as a watt governor) which monitors the amplitude
in its input buffers and adjust the delay time (at audio rates).
The gain on each channel can be similarly adjusted. Ini-
tial experiments suggest that even this simple mechanism
Can sustain frequencies other than those promoted by the
frequency response of the audio system, creating subtle
shifts which are sensitive to tiny changes in the ambient
environment. Further variation can be imagined by im-
plementing frequency-dependent delay lines and setting,
or, evolving, the delay times according to the impulse re-
sponse of the physical environment, frequency response of
audio system and spectro-temporal characteristics of the
surrounding acoustic environment. Recent research in
automated methods of avoiding feedback (outlined in 181)
point to several potentially interesting avenues for explo-
ation.

3. DISCUSSION

The self-observing systems offer a simple illustration of a
generative sound art scheme which implements a sonics
version of Floridi’s ontological alignment. In the design
of the generative schema, sound (or its digital represen-
tation) is integral to the generative process such that the
two are mutually constituted. What we listen to is not
the sonification of some numerical process at runtime,
but the ‘empirical epiphenomenon’ of interaction between the
two. Under this approach, we circumvent the need to
define either mappings to, or representations of, a final
sonic manifestation. This is offered in response to calls
from the generative art community, that as the field ma-

tures, it should move beyond the implementation of algo-
rithms lifted from the sciences; it aims to illustrate how we
can explore changes in the broader cultural world, by
representing systemic principles of the changing infosphere
in the materials, both formal and sensory, of a generative
practice.

3.1. Generative sound art as a poetic playground for an
information society

The self-observing systems outlined above are also in-
tended as a cartoon-like illustration of how a generative sound
practice could reflect upon and explore the impli-
cations of significant changes in our broader infosphere.
The closing proposition of this paper is that this represents
a potent role for generative sound art in contemporary so-
ciety. To return to Floridi’s analysis introduced in section 1.3,
the metaphysical shift he identifies can be seen in the
following key changes: Objects and processes are be-
coming de-physicalised, causing a shift from a materialist
to information-based metaphysics in which they become
typed, clonable and support independent; As the world
fills with intelligent agents, we cease to be discrete indi-
viduals and become networked info-gets, part artificial, part
human, connected informational organisms; Interactabil-
ity, rather than immobility or perceivability, becomes the
criterion for existence, even if the interaction is only vir-
tual; The global infosphere is merging with the analogue
world, creating a world populated by intelligent, active
agents which challenge our Newtonian world view.

For the computer musician who has spent the last half
decent design intelligent, interactive, modular perfor-
ance systems toward a networked digital practice, this
is nothing new. Artistically, the implications of such de-
velopments could even be summarised as an increase in
creative opportunity. To the general public, however, this
may read as a science fiction scenario. For society at large
such changes bring about significant ethical challenges,
not least an urgent need to reconcile the technological and
natural worlds [8].

The public need for new ways to come to terms with the
existential and ethical impact of an increasingly technologised
society can be seen in the emergence of new forms of hybridised discourse. New Scientist, for exam-
ple, have just launch a digital publication, Arc, which
merges literature (science-fiction) and science (futurology)
to explore the impact that technology is having on our lives. “Fiction gives us the chance to explore and be ec-
centric” says Simon Ings, a novelist, science writer and
editor of Arc. “If one thing is for sure, the future is not
going to be agreed by committee. The future is going to be
eccentric. And the best way of predicting the future is to
make it up.” [16]. New branches of social science are also emerging which adopt the formal modelling and
simulation practices similar to those used in generative art

The simulation sciences provide ways of understand-
ing the world through modelling. Science-faction pro-
vides a speculative literary discourse for toying with pos-
sible futures, but, as Manovich and others have suggested,
the generative arts have the potential to speculatively toy
with models of possible futures. The development of a
generative sound art practice has a unique potential as both
a poetic (imaginative, symbolic, figurative) and poie-
 tic (from the Greek, ποιεικός, meaning productive, pos-
itive) discourse, one which can engage the ears, intel-
lect and imagination in equal measure.

4. CONCLUSION

Now, more than ever, we need new forms of discourse
which enable us to come to terms with the complexities of
contemporary society. Computer music, and more specifi-
cally, generative sound art, represents a possible discourse
through which we can model our world reflectively, tin-
kering with these models, critically, and present these stories
in the material of sound itself: to talk about the world
through sound.

5. REFERENCES

[1] N. K. Gilbert, N. Abd-Tarizich,
[5] A. Di Scipio, “Sound is the interface: From inter-
[6] S. Kim-Cohen, In the Blink of an Ear: Toward a Non-
[9] ——, “A look into the future impact of ICT on our
[15] S. Kim-Cohen, In the Blink of an Ear: Toward a Non-
[18] E. Perez-Gonzalez and J. Reiss, “An automatic max-
imum gain normalization technique with applica-
tions to audio mixing,” in 124th AES Convention
[19] D. Van Nort, P. Olivierots, and J. Brautsch,
[20] H. Von Foerster, M. Mead, and H. L. Teuber, Cybe-
[22] M. Whitelaw, “System stories and model worlds:
A critical approach to generative art,” in Readme
100: Temporary Software Art Factory. Norderst-
[23] M. Yee-King. “An autonomous timbre matching im-
prover,” in Proceedings of the 2011 International
THE ELECTRONIC MUSIC OF ROBERTO GERHARD
Monty Adkins  Carlos Duque  Gregorio Karman
Centre for Research in New Music
University of Huddersfield

ABSTRACT
Roberto Gerhard was a pioneer of electronic music in England creating it so soon after the substantial concert, theatre and radio works from as early as 1954. However, for various political, cultural and personal reasons Gerhard’s electronic music has not been published or widely disseminated. Gerhard’s electronic music is one of the richest repositories for understanding the development of the composer’s late compositional technique as well as the early development of electronic music in the UK. As a result of an AHRC study of the tapes held in the Gerhard Archive at the Cambridge University Library it is possible to understand the composer’s technique and thoughts on electronic music and how they evolved as his work with magnetic tape became more and more refined.

1. INTRODUCTION
Roberto Gerhard considered himself an explorer of sound rather than someone who merely experimented with it. Central to this exploration in the final two decades of his life (1950-1970) was electronic music. In his writings from 1930, Gerhard is as prophetic regarding the future of music as Cage and Varese’s ‘territory, untested as to its aesthetic value to the musician. [1]

Although Gerhard writes in Concrete Music and Electronic Sound Composition that he approached ‘the electronic medium strictly as a sideline’ [2], the importance of this work and its impact on his instrumental composition has thus far received scant academic interest. Gerhard himself maintained that working in the electronic medium had resulted in a number of far-reaching morphological changes in the manner of composing sound and it seems to me that these changes are bound to affect methods of composition in the traditional field of instrumental composition as well. [3]

Gerhard’s approach to electronic music traversed the aesthetic paradigms that polarized early musique concrète and Électronique Musik, often using instrumental, concrete and electronic sound materials. Working very much on his own (the BBC Radiophonic Workshop was not opened until 1958, some four years after Gerhard had started working in the medium) he was critical of the mainstream avant-garde, writing that,

most of us had already noticed for some time that, whether German, Italian, Dutch or Belgian, electronic music sounds curiously alike in its timbral aspect. If the possibilities were really unlimited, one couldn’t help feeling that these composers were strangely coincident and repetitive in the use they made of them [4].

and that the sine tone has a ‘rigid, cold, dead-signal quality. It is utterly unsuited to convey anything warm, tender, vivid, alive in human experience’ [5]. Gerhard was rather interested in the transformation of acoustic sound materials, stating that ‘the microphone captures the living spark of the natural acoustic source’ [6]. Gerhard was, however, more circumspect than either Edgar Varèse or John Cage in his use of such acoustic sources. In an unpublished notebook entry from 1957 Gerhard writes that he considers that, ‘the term ‘musique concrète’ is ridiculous’ [7] and later in 1959 he wrote that,

in principle, anything that comes from an acoustic source is possible material for musique concrète. This, of course, throws the gates wide open – too wide, perhaps – to material of all sorts, musical and not so musical. The French themselves, for instance, are not above using pots and pans for their exercices aux cassebroles as they describe them.’ [8]

Gerhard’s approach to electronic music with its emphasis on the abstract ‘musical’ quality of concrete sounds rather than their associative meaning and the sampling and transformation of his own instrumental compositions is akin to the work of Iannis Xenakis and Bruno Maderna – two composers for whom electronic music and its techniques were to play a central part in informing their compositional aesthetic. For instance, Gerhard’s use of concrete, instrumental and electronic sound sources in Audiomobile II DNA (1963) has a kinship in approach with Maderna’s La Rire (1962) which incorporates the sounds of voices, footsteps in rain, white noise and sine-tone generators, as well as transformed timpani, flute and piccolo.

2. SOURCES
Whilst Schaeffer, Stockhausen and their respective colleagues at the GRM and WDR studios propagated concert electronic music and produced significant theoretical output on their work and the new medium, Gerhard was a more practical composer. Gerhard’s experiments were carried out in the public glare initially through composing incidental music (The Prisoner (1954), King Lear (1955) and Pericles (1958) being some of his earliest such works).

One of the disadvantages of not working professionally in a major radio or state-funded studio meant that there was no archival administrative structure to preserve Gerhard’s electronic works. Apart from the electronic component of the Symphony no. 3, ‘Collages’ neither of the publishers of Gerhard’s instrumental music (Boosey & Hawkes and OUP) hold copies of his electronic works, or his incidental works incorporating electronics. The major repository of Gerhard’s electronic music is the archive held in the Cambridge University Library. A small number of recordings and cues of theatrical productions are held at the British Sound Archive and the Archive of the Royal Shakespeare Company.

During the 1950s and 1960s, Gerhard gathered a significant magnetic tape collection in his studio, corresponding to a major repository of historical sound recordings of his own work in which all areas of his compositional activity are represented. Following Gerhard’s death in 1970, Poldi Gerhard continued to play back the recordings, helping to identify their contents with her own annotations and comments. After her own death in February 1994, the studio was dismantled and the tapes were deposited at the Cambridge University Library with the rest of Gerhard’s archive. In 2008 the inventory of the tape collection took place, and later that year, Gerhard’s archive was donated to the Cambridge University Library.

A preliminary catalogue of Gerhard’s Tape Archive comprising 714 items was compiled by one of the authors (Karman, 2008). During this stage, the annotations on boxes and other materials found on the tape containers were documented and the general state of the collection was assessed. Different problems were identified including on-going chemical degradation processes (see Fig.1), and a number of tapes were found to be incorrectly labeled or misplaced. The current research project involves to digitize all of the tapes as well as to produce a complete catalogue of the contents of the archive. The current research project has digitized all of the tapes as well as documenting the annotations on boxes and other materials found with the tapes to produce a full catalogue of the archive.

For Gerhard’s electronic works, the magnetic tape collection at the Cambridge University Library is the primary source. Over half of the tapes in the collection are directly related to Gerhard’s sound compositions [9] – the rest comprising a considerable number of recordings of his own instrumental works and a library of music by his contemporaries (including Schoenberg, Webern, Berg, Bartok, Stockhausen and Nono). Excluding one remarkable exception, the Symphony no 3 ‘Collages’ (1960) [10], most of this work remains unpublished, and in a number of cases is not available from other sources. The tapes contain all different stages of production, from initial source recordings to multilevel compound mixes [11] and completed compositions.

3. GERHARD’S STUDIO
According to the International Electronic Music Catalogue (1968) compiled by Hugh Davies, the first informal activities in Gerhard’s private permanent studio are listed as having been initiated in 1954. The

1 ‘The Electronic Music of Roberto Gerhard’ funded by the Art and Humanities Research Council 2012.

2 ‘The Electronic Music of Roberto Gerhard’ funded by the Art and Humanities Research Council 2012.
official foundation of what Gerhard termed his ‘Home Office’ (perhaps the first private ‘home’ studio in the UK), can be dated to 1958, coinciding with the composer’s change of address to 14 Madingley Road, Cambridge on October 1st 1958.

Gerhard’s close friend, Joaquim Homs, visited Cambridge in September 1959 and provides a first-hand impression of the studio one year after the Gerhard’s move to Madingley Road.

The study was ample and, at the back, near the window that lead to the garden, there was a grand piano. [...] By now Gerhard had constructed an electronic laboratory [...] with the aid of the Radiophonic Workshop, and it was full of tape-loops of concrete music [12].

A series of undated black and white portraits of Gerhard at his workplace [13], perhaps simultaneous to Homs’ visit, present varied perspectives of four open-reel tape recorders, together with numerous reels on shelves and an unusual image of hundreds of tape splices fixed on hooks to the lid of the grand piano (Fig.2).

Although Gerhard maintained that I’ve always been working with shoe-string equipment in electronics. It comprises: one microphone, five tape recorders, a track mixer of five channels, and that is all. I’ve never used oscillators or white noise generators. I’m allergic to sine tones. When I needed certain types of white noise, the BBC Radiophonic Workshop has kindly provided lengths of tape. I’ve always been working with shoe-string equipment in electronics. It comprises: one microphone, five tape recorders, a track mixer of five channels, and that is all. I’ve never used oscillators or white noise generators. I’m allergic to sine tones. When I needed certain types of white noise, the BBC Radiophonic Workshop has kindly provided lengths of tape. I would have been happy to have been able to install envelope control. I could not afford it. But I have been able to develop some measure of envelope modification by a manual means. I have no visual or audio monitoring. I wish I could have had some modulators. No automatic switching devices. On occasion their absence has been very trying. [14]

A closer investigation of these photographs (see Fig.2 and Fig.3) [15] supplies further information about the recording equipment in Gerhard’s studio c.1958-59.

There were two EMI TR50 mono recorders, an early Vortexion WVA’ mono recorder and a Ferrograph Series 66 mono recorder¹. In the early 1960s, Gerhard incorporated a new Ferrograph Series 48 mono recorder and a five-channel mixer into his studio. It would not have been uncommon to find a similar set of open-reel tape recorders in the facilities of the BBC [16].

With this in mind, and though Gerhard was eager to underline the most essential equipment with which he worked in the ‘Home Office’, it would be better to characterize his studio as one that contained some of the best commercially available equipment at the time.

4. BBC RADIOPHONIC WORKSHOP

For Gerhard, his contact with the BBC Radiophonic Workshop, was vital and the only external support he had for his work. It opened on 1 April, 1958 some four years after Gerhard had started work in the medium; the technicians working in Room 13 at Madingley Road (the Radiophonic Workshop headquarters) included, among others: Daphne Oram (who resigned in January 1959, after 15 years with the BBC, to follow a career as a composer); Delia Derbyshire (who joined the BBC in 1960 and collaborated with Gerhard on his 1965 Prix Italia winning Anger of Achilles) and Dick Mills (who assisted with producing Gerhard’s work, particularly the Symphony no.3 ‘Collages’) at the Royal Albert Hall and also the Royal Festival Hall). When the Radiophonic Workshop opened, an invitation was sent out to numerous composers to come and see the new facilities with a view to discussing the possibilities for composition opened up by the studio. Apart from Gerhard, only two other composers accepted the invitation. Peter Manning writes,

‘The closed door’ policy of the BBC Radiophonic Workshop, and the continuing lack of support from other quarters, severely retarded developments in Britain during the 1960s. Indeed, Roberto Gerhard was the only established composer from the broader community to be granted reasonable access to the BBC facilities during the decade. This permitted him to produce a number of pieces, primarily for radio, working both at the BBC and at his own private studio in Cambridge [17].

The years 1958-1965 were the most productive regarding Gerhard’s electronic music output. It is perhaps because of the regular commissions (The Unexpected Country (1957), Asylum Diary (1959), The Cage and The Prisoner (1961)) and the arrival of the Cologne Studio for Electronic Music (1961-64) that Gerhard received from the BBC for music for radio plays and the William Glock’s admiration of Gerhard’s work that allowed him to work in his home studio and in the BBC Studio with great flexibility.

5. GERHARD’S ELECTRONIC MUSIC

Hugh Davies, in his 1981 Tempo article on Gerhard’s electronic music wrote that, Gerhard was not only the first important British composer to adopt electronic music techniques; it seems probable that he was, by a few months, the creator of the first British score to involve tape [18].

Gerhard’s pioneering achievements can be put in a broader, less localized, perspective. The first musique concrète work, l’étude aux chemins de fer, was produced by Pierre Schaeffer in 1948 at the Club d’Essai, RTF (later IANA-GRM). In 1950 Schaeffer and his then assistant Pierre Henry produced their first substantial work in the genre: the collaborative Symphonie pour un homme seul. The NWDR studio opened in 1953, where Stockhausen produced his first experiments with Elektronische Musik, the Studie I & II (1953 and 1954). The first acknowledged work that combined instruments and electronic sounds was Madonna’s Musica in due dimensioni produced in Bonn, in 1952 for flute, cymbal and electronic tape. One of the most famous early works incorporating electronics was Varèse’s Déserts (1954) for ensemble and tape. Varèse’s work alternates rather than integrates the instruments and electronics, having three tape ‘interpolations’. It was in the same year, 1954, that Gerhard completed his first ensemble and tape work, the incidental music for Bridget Boland’s play, The Prisoner.

Gerhard was well aware of the techniques of electronic music on the continent: transposition, looping and layering of sounds, cutting and splicing to create rhythms or dynamic envelopes, feedback, filters and ring modulators, were thoroughly described in a special number of the technical magazine of the Nordwestdeutschen Rundfunk devoted to the Colognne Studio for Electronic Music [19], part of the composer’s book collection along with other seminal texts relating to the early days of electronic music by composers such as Pierre Schaeffer, Karlheinz Stockhausen and Milton Babbitt. While always suspicious of studios operated by sound technicians, Gerhard, on occasion regretted his lack of more sophisticated devices, envelope controllers and modulators. It is therefore not surprising that one of his favourite resources was the use of transposition (see Table 1). Although Gerhard wrote that he was primarily interested in electronic music for ‘applied works... to works of radio and television, for the stage and screen’ completing twelve substantial scores for ensemble or tape and three substantial works for theatre, he produced a number of works with or for electronics not intended as incidental music.

Table 1: Tape-speed combinations found in King Leaf’s sound score.

It should be noted that Gerhard often annotated his tape boxes 7″ rather than 7½″ i.p.s.
Gerhard’s electronic works for concert include the Audiomobiles series of works (1958-63), the second of which became the soundtrack for Hans Boys and Anand Sarabhai’s film DNA in Reflection (1963); Lament for the Death of a Bullfighter, for speaker and tape (1959); Symphony no.3, ‘Collages’ for orchestra and tape (1960); Caligula (1961), the projected Sculptures series (1963) utilising sounds recorded from a sculpture by John Youngman (only one work in this cycle was completed although there are multiple tapes that contain substantial compounds, and the epic, though unfinished one work in this cycle was completed although there are multiple tapes that contain substantial compound mixes), and the epic, though unfinished Vox Humana (1966-67). The Ten Pieces for tape are extracts from Audiomobiles II: DNA in Reflection.

The last work in this category is a live electronic work entitled Claustrophilia – a page to John Cage (1966) scored for orchestra and tape (1963). It is the authors’ worklist officially contains an entry for Audiomobiles I-IV, from preliminary study of the archive it appears that only three were completed. Audiomobiles I was completed in 1958. The second in the series was composed for the film DNA in Reflection in 1963. A third Audiomobiles appears in the tape catalogue often listed as Audiomobiles 3 ‘Sculpture’. It is the authors’ belief that Gerhard having recorded and processed the sounds of John Youngman’s metal sculpture realized that their sonic potential was considerable. Therefore, having originally intended the material merely to be used for the next in the series of Audiomobiles works decided rather to create a new cycle of works based solely on Youngman’s sculpture.

In private letters to Davies, Gerhard indicates that he has 27 ‘...an accumulation of work in a state of near-readiness, I mean ready for com-po-sition, namely ready for com-positing’ [21]. One such example is tape CUL_0RO1_011601 on the box of which Gerhard has written ‘very good bits of electronic music’ and contains twenty-four minutes of highly developed (almost) continuous electronic music derived from the Youngman sculpture. Utilized were by no means limited to instrumental sources. Production notes reveal the regular use of daily objects for making sounds (packing paper, paper tissue, cellophane, abrading), as well as a wide range of incidental noises (birds, dogs, axe strokes, cracking tree, thunder, wind, rain and storm, whipping grass, crowds, chatter, laughter, screams), which could be home-made 10 or taken from the everyday environment. In his notebooks, Gerhard writes, ‘...we all have got to start in the same way: by building up a repertoire of sounds which are stored on tape. [...] The sounds selected may either be appropriate in their original form to the sound-picture one has in mind or else require further treatment before being used. Most of my stored sounds are of instrumental origin, recorded on tape through microphone. The next step - what I called my second stage - is directed towards a certain transformation of that original sound, ideally towards a metamorphosis of the sound [in] which [its] origins are blurred, and a far-reaching change of identity might be achieved.’ [23]

Gerhard’s methods for obtaining such source materials for his compositions are documented by Lindsay Anderson and Dick Mills. Anderson writes, ‘...I remember visiting Roberto in Cambridge, talking about the score, and even assisting him in throwing various objects down the stairs, in an effort to produce the right kind of abstract sounds which he felt he needed.’ [24]

As a sound composition, Gerhard’s work was primarily intended to be used for the next in the series of Audiomobiles projects. By creating a sound composition was to gather a preliminary study of the archive it appears that only three were completed. Audiomobiles I was completed in 1958. The second in the series was composed for the film DNA in Reflection in 1963. A third Audiomobiles appears in the tape catalogue often listed as Audiomobiles 3 ‘Sculpture’. It is the authors’ worklist officially contains an entry for Audiomobiles I-IV, from preliminary study of the archive it appears that only three were completed. Audiomobiles I was completed in 1958. The second in the series was composed for the film DNA in Reflection in 1963. A third Audiomobiles appears in the tape catalogue often listed as Audiomobiles 3 ‘Sculpture’. It is the authors’ belief that Gerhard having recorded and processed the sounds of John Youngman’s metal sculpture realized that their sonic potential was considerable. Therefore, having originally intended the material merely to be used for the next in the series of Audiomobiles works decided rather to create a new cycle of works based solely on Youngman’s sculpture.

Gerhard’s working processes are fairly well documented in his notebooks. They contain numerous annotations of source materials and comments on these. For Gerhard, the first step toward creating a sound composition was to gather a repertoire of raw materials on tape. This process is described in ft. 1-10 of the sound score [22] for the incidental music to King Lear (1955), which contains detailed instructions for recording a catalogue of instrumental sounds using different dynamics and modes of attack, including: maracas, cymbals, xylophone, tubular cymbal, tam-tam, piano, chromatic timpani, bass drum, gong and mbira. In his studio, Gerhard had a microphone available for making recordings of piano effects - or smaller percussion instruments. But the sound materials he employed were by no means limited to instrumental sources. Production notes reveal the regular use of daily objects for making sounds (packing paper, paper tissue, cellophane, abrading), as well as a wide range of incidental noises (birds, dogs, axe strokes, cracking tree, thunder, wind, rain and storm, whipping grass, crowds, chatter, laughter, screams), which could be home-made or taken from the everyday environment. In his notebooks, Gerhard writes, ‘...we all have got to start in the same way: by building up a repertoire of sounds which are stored on tape. [...] The sounds selected may either be appropriate in their original form to the sound-picture one has in mind or else require further treatment before being used. Most of my stored sounds are of instrumental origin, recorded on tape through microphone. The next step - what I called my second stage - is directed towards a certain transformation of that original sound, ideally towards a metamorphosis of the sound [in] which [its] origins are blurred, and a far-reaching change of identity might be achieved.’ [23]

Gerhard’s methods for obtaining such source materials for his compositions are documented by Lindsay Anderson and Dick Mills. Anderson writes, ‘...I remember visiting Roberto in Cambridge, talking about the score, and even assisting him in throwing various objects down the stairs, in an effort to produce the right kind of abstract sounds which he felt he needed.’ [24]

Dick Mills, who worked at the BBC Radiophonic Workshop describes recording sessions in which Poldi Gerhard was fond of participating too, writing that:

Roberto had a rather difficult problem to overcome when attempting to record his basic sounds, as he lived on a busy trunk road in Cambridgeshire and the only quiet period was around 3.30 in the morning. One can imagine the scene as Roberto twanged and banged and bonked metallic objects as his wife Poldi acted as recording engineer. Both of them were in their sixties at that time. [25]

Aside from sound sources recorded in his own studio, Gerhard also recycled fragments of recordings of his own instrumental works. Where the materials he needed could not be easily recording or created in his own studio Gerhard would resort to commercial instrumental sounds was recorded. One such example is the music for the Royal Shakespeare Company’s performance of Pericles (1958) for which Gerhard produced the incidental music for ensemble and electronics. The box of tape CUL_0RO1_02540112 credits ‘Studio Black, Queens Way’ for the recording of percussion and exotic instruments. The multiplicity of sound sources meant Gerhard could not always obtain sounds also include his close friend Joaquim Homs for recordings of castanets which were required for the part of Symphony no. 3, ‘Collages’, an Australian friend bringing recordings of fishes. Although Gerhard had a preference for sources of acoustic origin, this did not rule out the occasional use of synthetic sounds, such as white noise or sine tones.

In the second stage of the production process, Gerhard listened intently to the internal characteristics of his material, abstracting the sounds from their physical sources through various means of processing. During this stage of processing the primacy of the original sound as a means of grouping material developed from it became redundant as a means of classification. As Gerhard processed his material he regrouped it so that the timbral or gestural relationship between the sounds now assumes the most important means of classification. This processing stage allowed Gerhard to re-classifying the transformed sounds into sound-families – what Gerhard referred to as his ‘theory of change of family’ [26] in which material is grouped together because of its similar sound behavior or timbre. From these sound-families Gerhard developed a series of clear compositional stages and his own terminology for each:

- small mixes Gerhard termed ‘sound images’ or ‘sound aggregates’;
- these ‘aggregates’ were mixed to form ‘compounds’;
- numerous ‘compounds’ were mixed to form ‘multilevel compounds’;
- these multilevel compounds the final ‘assembly’ would be mixed through editing.

Gerhard’s working processes are fairly well documented in his notebooks. They contain numerous annotations of source materials and comments on these. For Gerhard, the first step toward creating a sound composition was to gather a repertoire of raw materials on tape. This process is described in ft. 1-10 of the sound score [22] for the incidental music to King Lear (1955), which contains detailed instructions for recording a catalogue of instrumental sounds using different dynamics and modes of attack, including: maracas, cymbals, xylophone, tubular cymbal, tam-tam, piano, chromatic timpani, bass drum, gong and mbira. In his studio, Gerhard had a microphone available for making recordings of piano effects - or smaller percussion instruments. But the sound materials he employed were by no means limited to instrumental sources. Production notes reveal the regular use of daily objects for making sounds (packing paper, paper tissue, cellophane, abrading), as well as a wide range of incidental noises (birds, dogs, axe strokes, cracking tree, thunder, wind, rain and storm, whipping grass, crowds, chatter, laughter, screams), which could be home-made or taken from the everyday environment. In his notebooks, Gerhard writes, ‘...we all have got to start in the same way: by building up a repertoire of sounds which are stored on tape. [...] The sounds selected may either be appropriate in their original form to the sound-picture one has in mind or else require further treatment before being used. Most of my stored sounds are of instrumental origin, recorded on tape through microphone. The next step - what I called my second stage - is directed towards a certain transformation of that original sound, ideally towards a metamorphosis of the sound [in] which [its] origins are blurred, and a far-reaching change of identity might be achieved.’ [23]

In the instances in which Gerhard required variable speed playback, the transformation would again be organised in an external facility (often the BBC Radiophonic Workshop). By working between his own studio and the BBC Radiophonic Workshop Gerhard was able to achieve such processes as the mixing of three different recordings at different speeds and applying simultaneous glissandi to the recordings. In his notebooks, Gerhard used capital letters to identify the sound patterns that resulted from the combination of multiple sources in such
For Gerhard, the final part of the composition process was the ‘sound-montage’ – the assembling, editing and juxtaposition of ‘compound mixes’. Gerhard considered the sound-montage ‘something of a game; something like a jigsaw puzzle with pieces upside-down or the wrong way around, bumping into one another and thus emphasizing their isolation, rather than giving them a common purpose which would lift them onto a plane of poetic imagery’ [28]. Working with electronic music gave Gerhard – an immediate statement is the intuitive freedom that working in the electronic medium gave him – an immediate tactility of working with, and transforming, sound. Here a further comparison with Maderna may be drawn. About electronic music, Maderna once said, ‘we no longer have a linear time – our consciousness casts various projections of time that can no longer be represented with the logic of one dimension’ [29]. Working with electronic music made Maderna trust in his compositional intuition. The influence of electronic music in Maderna’s instrumental composition can be found in works such as the Sewerellite. Gerhard himself wrote that ‘the way time is felt in electronic music differs entirely from the way time is experienced in traditional music.’ Gerhard was adamant that there is a fundamental difference between working with electronics and instruments. He uses the term sound-composition (1959)’ in BOWEN, M. (ed.), (2000) Gerhard on music: selected writings, Aldershot, Ashgate, pp.79-80

In the Symphony no. 4 Gerhard often uses four cymbals playing simultaneously to emulate electronic sounds. Figure 7 creates a ‘noise’ crescendo similar gestures developed from white noise in Coligula and Audioambios II. Later in the work (bb.485-486 – Figure 8) Gerhard creates a similarly striking sound by drawing screw-rods over suspended cymbals indicating that

The electronic medium, in effect, makes the work received its world premiere performed by the authors et al., on 26th April 2012 at the 2nd International Roberto Gerhard, Conference, Barcelona.

For the operatic work ‘behaviour’, it will be noticed, not colour; colour is never of decisive importance. Instead of ‘behaviour’ I might have used the term sound-activity. The electronic medium, in effect, makes possible new modes of action with sound which have greater freedom of tonal movement, of configuration and of structural weaving than those which our traditional instruments permit. [30]

Gerhard’s sound-of-behaviour bears a close conceptual resemblance to what Denis Smalley would later term spectromorphology [31] – literally the shaping of sound through time. In line with thinking in fields of sound-activity the electronic works are driven by gesture and texture seen within a structural chord system. Although Gerhard did not care for Schaeffer’s term for the basic perceptual unit in musique concrète, the objet sonore, it is clear that in his electronic works and increasingly in his later instrumental works, he nevertheless moved away from the ‘tone’ as the essential unit, to his own notion of the sound object or sound field as building blocks for his works. This is particularly evident in the Symphony no. 4 ‘New York’ (1966). Examples occur at the beginning of the Symphony no.4 are the ‘structural chords’ and Gerhard’s use of percussion. The structural chords are played in the brass and are used to articulate the structure of the work, in particular the transition from one texture to another and cadential points at the end of sections, for instance in bb.14-15 (Figure 6).

The identification of the machines and their description has been possible thanks to the kind collaboration of Mr. Georg West.

When Gerhard was adamant that there is a fundamental difference between working with electronic music and offers a critique of the French and German schools of musique concrète and Electronik Musik. The tape archive at the Cambridge University Library contains the work of a dynamic and original composer. One who was at the forefront of the exploration of the new medium of electronic music and one who deserves to be recognized as such.

REFERENCES


3. Ibid. 2 p.180

4. Ibid. 2 p.183

5. Gerhard unpublished notebooks, CUL

6. Gerhard Symphony no. 4 ‘New York’ (1966). Examples occur at the beginning of the Symphony no.4 are the ‘structural chords’ and Gerhard’s use of percussion. The structural chords are played in the brass and are used to articulate the structure of the work, in particular the transition from one texture to another and cadential points at the end of sections, for instance in bb.14-15 (Figure 6).

The work received its world premiere performed by the authors et al., on 26th April 2012 at the 2nd International Roberto Gerhard, Conference, Barcelona.

8. REFERENCES


3. From a transcription of Maderna’s 1957 presentation at Darmstadt (made by Horst Weber, 1984)


8. Using Gerhard’s own terminology.

9. Ibid. 2 p.180

10. Ibid. 2 p.183

11. Ibid. 2 p.183

12. Gerhard unpublished notebooks, CUL

13. Ibid. 2 p.184


20. Ibid. 2 p.183

21. Ibid. 2 p.180

22. GERHARD.7.115 f.20

23. GERHARD.7.102, ff. 1-10, Cambridge University Library.

24. GERHARD.7.115 f.45i


27. GERHARD, R., ‘Collages’, of which at least three commercial recordings are currently available: HMV ASD 2427, (1967); Montagu Aviside MO 782103, (1997); Chandos Records 1013104, (1997).

28. Ibid. 2 p.180


32. GERHARD.7.102, ff. 1-10, Cambridge University Library.


38. Ibid. 2 p.183

39. From a transcription of Maderna’s 1957 presentation at Darmstadt (made by Horst Weber, 1984)


41. Ibid. 2 p.180

42. Ibid. 2 p.183

43. Ibid. 2 p.183

44. Ibid. 2 p.183

45. Ibid. 2 p.184

46. Ibid. 2 p.184

47. Ibid. 2 p.184

48. Ibid. 2 p.184

49. Ibid. 2 p.184

50. Ibid. 2 p.184


52. GERHARD, R., Symphony no. 4 (OUP, London, 1971) p.1
TOWARDS A TYPOLOGY OF FEEDBACK SYSTEMS

Dario Sanfilippo
Scuola di Musica Elettronica
Conservatorio di Musica San Pietro a Majella di Napoli

Andrea Valle
CIRMA - StudiUn
Università degli Studi di Torino

ABSTRACT

Feedback-based systems for audio synthesis and processing have been in use since the 60s, resulting both from the theoretical reflection on Cybernetics and System theory, and from experimentation on analog circuits. The advent of computers has made possible the implementation of complex theoretical systems into audio-domain oriented applications, in some sense bridging the gap between theory and practice in the analog domain, and further increasing the range of audio/musical applications of feedback systems. In this paper, we first briefly introduce feedback systems; then, we propose a set of features to characterize them; finally we propose a typology targeted at feedback systems used in the audio/musical domain, and discuss some relevant examples.

1. INTRODUCTION: FEEDBACK PROPERTIES

The notion of feedback is widely used both as a technical feature in the design of audio and music application and as an aesthetic key in their description (see e.g. [14] [26] [9] [18], as well as the musical works based on feedback that we will discuss later). In short, many feedback-based systems for audio and music production/composition exist and many sound artists/composers consider feedback as a crucial notion at the basis of their work. Rather than being a monolithic category, feedback is a complex notion that presents many different features and aspects that are grouped together because of a family resemblance. A minimal definition of feedback takes into account the configuration of a system, provided with input/output, in which some kinds of transformation are carried out, where the output is connected (fed back) to the input after a delay (namely greater than 0 seconds) [13]. In negative feedback the input-output relation is inverse: an increase in the output causes a decrease in the input, and vice versa. Thus, the response of the system to stimulus is that of compensation, and it will tend to be in equilibrium around a desired target. In a positive feedback configuration, the input-output relation is direct: if the output increases, the input increases; vice versa, if the output decreases, the input decreases. In this case, a deviation of the system towards a direction will produce a further shifting in the same direction, and the behavior will be that of magnifying the effect (Cage) [1]. He implemented a system based on a network of eight delay lines in a bidirectional circular audio feedback configuration, where two microphones are connected to two of the eight delay lines, feeding the network with the sound from the piano, and where each node’s output is connected to a loudspeaker (see Figure 1). The nodes have an independent time-varying delay, and they also contain sound transformations like resonators and ring modulators. Apart from the technical implementation, the behavior of the system provides a practical example of how interactive feedbacks may happen. The network acts as a Larsen effect which is triggered by perturbations of the sound from the piano. Although the system design is relatively simple, a high number of loops and sub-loops are activated between microphones and loudspeakers. As a result, the output of each node is dependent from the piano and from all of the other nodes, in turn feeding back the network. It thus becomes possible to hear the sound of the single nodes together with their mutual influence.

1.2. Interaction, interdependency and synergy

A fundamental property in feedback configurations is that of coupling [2]. Two or more elements within a feedback loop are coupled because they operate in a condition where they mutually affect each other. From a systemic perspective the concepts of interaction, interdependency and synergy are crucial in order to understand feedback systems. A totality which is made up of different components, interconnected by specific relations, shows a certain behavior and identity thanks to the cooperation of all its parts. Any small change in the organization of the relational network can potentially change the identity of the system and radically alter its behavior. Any system of this type, thus, relies on all its components, and each of the parts has a fundamental role in the global functioning of the system. The strict interaction between the components allows for the combination of their properties, leading to new entities which are not the result of a mere summation of the properties of their parts, but, rather, result from their synergy. In most cases, interaction in performances is described as a level occurring between the human and the machine, where -typically- gestural devices let the performer define actions to be followed by reactions in the machine, without taking into account a real mutual influence. On the contrary, Di Scipio has been able to provide an interesting perspective on interaction in music by describing it as a condition that takes place in the sound domain [14]: interaction occurs among sound materials. Christopher Burns followed Di Scipio’s path in his realization of Electronic Music for Piano by John Cage [9]. He implemented a system based on a network of eight delay lines in a bidirectional circular audio feedback configuration, where two microphones are connected to two of the eight delay lines, feeding the network with the sound from the piano, and where each node’s output is connected to a loudspeaker (see Figure 1). The nodes have an independent time-varying delay, and they also contain sound transformations like resonators and ring modulators. Apart from the technical implementation, the behavior of the system provides a practical example of how interactive feedbacks may happen. The network acts as a Larsen effect which is triggered by perturbations of the sound from the piano. Although the system design is relatively simple, a high number of loops and sub-loops are activated between microphones and loudspeakers. As a result, the output of each node is dependent from the piano and from all of the other nodes, in turn feeding back the network. It thus becomes possible to hear the sound of the single nodes together with their mutual influence.

1.1. Nonlinearity and circular causality

A system is said to be linear when its output (effects) is proportional to its input (causes). As an example, let’s consider a pool ball not subject to friction forces. If the ball is hit with a force f, it will have a velocity v. When the force is twice greater, the velocity is doubled. Actually, many natural phenomena and systems in the world are intrinsically nonlinear, thus with no proportional relation between causes and effects. As a result, in a nonlinear system, causes of reduced size can have greater effects, and, on the other hand, causes of greater size can have smaller effects. A feedback system is typically nonlinear, nonlinearity being the result of a process with circular causality [19][21]. In such a configuration, effects are also causes [21], and there is a mutual relation between them. The causes are fed back to themselves through their effects, and the effects are the result of their combination with the causes, thus breaking the input-output linear proportion. Another important feature of feedback configuration and circular causality is that processes become iterated, leading to systems which will be capable of self-alimentation. From a musical perspective, nonlinearity clearly emerges in feedback-based systems where the change of internal variables can result in very different behaviors in the final output. A clear example is provided by the work of John Cage: he implemented a system based on a network of eight delay lines in a bidirectional circular audio feedback configuration, where two microphones are connected to two of the eight delay lines, feeding the network with the sound from the piano, and where each node’s output is connected to a loudspeaker (see Figure 1). The nodes have an independent time-varying delay, and they also contain sound transformations like resonators and ring modulators. Apart from the technical implementation, the behavior of the system provides a practical example of how interactive feedbacks may happen. The network acts as a Larsen effect which is triggered by perturbations of the sound from the piano. Although the system design is relatively simple, a high number of loops and sub-loops are activated between microphones and loudspeakers. As a result, the output of each node is dependent from the piano and from all of the other nodes, in turn feeding back the network. It thus becomes possible to hear the sound of the single nodes together with their mutual influence.

As with physical models, the parameters for each component section do not correspond to waveguide elements (networks without a particular beginning or end), including structures with “spokes” connecting microphones and loudspeakers. As a result, the output of each node is dependent from the piano and from all of the other nodes, in turn feeding back the network. It thus becomes possible to hear the sound of the single nodes together with their mutual influence.

The positive and negative feedback concepts can also be grouped together because of a family resemblance. The positive feedback occurs when an increase (decrease) in A produces an increase (decrease) in B. In the contrary, a negative feedback occurs when an increase (decrease) in A produces a decrease (increase) in B. For example, in the relation infection → virus, an increase in the infected people will lead to an increase in viruses, which will in turn lead to an increase in infected people (positive feedback) [22]. In the relation rabbits → grass, more rabbits eat more grass, grass decreases and so will the rabbits, but a decrease in the rabbits allows more grass to grow, eventually leading to more rabbits, and so on (negative feedback) [22]. Negative feedback is widely used in control and self-regulating systems (from thermostats to living organisms), and its major role is that of creating stability. Positive feedback, instead, has a typically unstable behavior and causes exponential variations. A set of specific features emerges from a wide literature on the subject of feedback systems, also in the audio/musical domain. As a consequence, a specific corpus of works and practices sharing these features can be identified in electronic/electroacoustic music. Even if the specific corpus can be identified by means of these features, still its internal variety is very high, and a classification schema can be proposed in order to clarify in which way some systems differ from others. In the following we first introduce the set of features that can be recognized in feedback systems; then we propose a classification schema for the identified works.

1.3. Self-organization, homeorhesis and homeostasis

Self-organization has received many definitions in different contexts, such as Cybernetics, Information Theory, Thermodynamics, Synergetics and others, and although the term is widely used, there is not a generally accepted meaning [20]. Here, we will describe the main features of self-organization so that it becomes possible to apply the concept to the musical domain, as a property characterizing feedback systems. Intuitively, self-organization happens when a system is capable of organizing itself autonomously: the sound captured from a microphone connected to a speaker is reproduced and again captured, recursively, resulting in a positive feedback activating pitch tones from the steaded amplification of a signal [7].

defined as the emergence of coherent patterns at a

level out of local interactions between the elements of a

system [21][19]. Because of the recursive relations be-
tween the system’s components, the self-organization pro-
cesses [20], [19], [18], [17], [16], [15], [14], [13], [12], [11], [10], [9], [8], [7], [6], [5], [4], [3], [2], [1] occur, and it takes place through

the simultaneous action of all the elements, none of them

playing the role of a coordinator. Self-organization, in

deep, is the trajectory of a centralization: it excludes the

presence of an external element regulat-

ing the system. From this perspective, music systems in

which the elements are independent (with no interaction

among them, as it often happens), and in which automa-
tions are high-level processes of sound organization, can-

not be considered as self-organized. If a state of a system

is any configuration of its variables (i.e. its overall out-

put), hence self-organization can be thought as the shift

from one state to another, including the different behav-

iors arising from the process of state shifting. A system
can enter a stable state, in which either the behavior is sta-
tical or it shows dynamic equilibrium. An opposite sit-

uation is that of a dynamic unstable behavior, in which

the system continuously shifts from one state to another.

Such a distinction allows to oppose self-organization (sta-

tical and dynamic, leading to an increase in order (a decrease in statistical entropy) [21][22]) to self-
disorganization (dynamic instability), leading to an in-

crease in disorder and/or disorganization in some respect analogous to homeostasis and homeo-

rheesis, the two terms indicating respectively a tendency
towards stability and a tendency towards a certain point
(an “attractor”) while the system shifts through different states (following a “trajectory”). An example of

work which is based on the concept of self-organization is Ephemerons by Phivos-Angelos Kollias. The composer
describes the system as representing the organism, its cells be-
ing sonic units in the process of adapting to the surround-

ing audio environment. Adaptation occurs thanks to the

capability of the cells to sense the loudness features from

the environment [25]. The system is initially fed with the

sound captured from the audience’s applause to the piece

performed just before Ephemerons. Cells’ perception of

the applause is used to dynamically control the signal pro-
cessing inside each cell. The work emerges from this self-

organization of a mass of cells interacting mutually and

with the environment, and evolving through the time.

1.4. Chaos, complexity and emergence

Chaos is a widely diffused term which is often used as a

synonym to “unpredictability”, yet the two terms do not

seematically coincide, as, although chaos implies unpre-
dictability, the reverse relation is not always true. First,

chaotic behaviors can be unpredictable even if no random-

ness is involved. Second, in chaos, what happens now, is

the effect caused from what happened before. More
generally, chaos can be thought as a highly dynamical be-

havior where order and disorder coexist and “compete”,
and where a causal connection between past, present and

future is established. Feedback can be modeled as a non-
linear iterated process, a formalism that, under certain condi-
tions, allows for the development of mathematical models of chaotic systems [24]. In feed-

back systems, chaotic behaviors can occur at two differ-

tent levels. In a situation of dynamical equilibrium, while

there is an overabundance of information, the system is

chaotic. On the other hand, if considering homoeo-

rheesis, the two terms indicating respectively a tendency
towards stability and a tendency towards a certain point
(an “attractor”), exactly the opposite occurs. Complexity is yet another important concept that can be

used to characterize feedback systems [24]. The paradigm of Complexity states that a mass of very simple processes can

do things very complex and unpredictable. In feedback systems, chaotic behaviors can occur at two differ-

tent levels. In a situation of dynamical equilibrium, while

there is an overabundance of information, the system is

chaotic. On the other hand, if considering homoeo-

rheesis, the two terms indicating respectively a tendency
towards stability and a tendency towards a certain point
(an “attractor”), exactly the opposite occurs. Indeed, feedback is an interesting case of a simple behav-

ior that leads to unexpected results (due to nonlinearity)

through iteration. In this sense, it can be described in the

framework of Complexity. The notion of complex-

ity is strictly related to that of emergence, the first defin-

ing the structural organization of the process, the second

the quality of unexpectedness of the results. Emergence

can refer to organizational levels [29], to self-organization
[39], to entropy variation [41], to nonlinearity [28], or ex-
clusively to complexity [5][6] [10] [23] or synergy [12].
Here we will focus on the description of emergence re-
ferring to the organizational levels approach, as it seems
referring to the organizational levels approach, as it seems
more relevant in relation to the features described in this

study on feedback systems used in a wide cor-
pus of musical projects dating back from the ‘60s that is

still flourishing at the present day. Although to-
gether by the use of feedback as a common denominator,
these works present a rich and complex phenomenology
which is the basis for our research, up to now unarticulated.
First of all, it is possible to propose a general schema of audio feedback systems that aims at summing
up all the key features emerging in our analysis corpus.
Figure 2 shows a feedback system (System) following the previously introduced minimal definition of feedback, System’s audio output (audio, resulting from Out) is re-

jected into the same system input (In). Starting from Figure 2, it is possible to propose five main features to
classify feedback-based audio/music systems. These fea-
tures can be organized into couple of oppositions, defining six categories. The other elements represented in Figure 2 will be introduced while discussing the categories.

2. TOWARDS A TYPOLOGY OF MUSIC FEEDBACK SYSTEMS

The previous discussion, though very general, allows to

narrow the field to feedback systems used in a wide cor-

in use). These systems are implemented by means of audio feedback networks of non-random and non-automated DSP modules like reverb, ring modulation, frequency shift-

ing and waveshaping, a design explicitly thought as tech-
nick in the field of sound art, capable of unfolding the

generative behaviors and emergent properties. The work by the Aus-

tralian sound artist Malcolm Riddoch, instead, is an ex-
ample where both digital and analog devices are used. His approach is based on improvisation, with an impor-
tant focus on environmental factors, soundscapes transfor-
mations, and intermedia. His aim is to classify feedback systems using different types of input/output trans-
ducers (microphones, electric bass/guitar), and analog/digital modules for sound manipulation, even using the computer to turn sound into control signals. In this way, the artist creates feedback chains aimed at exploring the spaces where the performance takes place.

2.2. Information rate: audio/control

The output of an audio feedback system is an audio stream. Audio information rate can be described in turn in relation to perception (e.g. in terms of temporal resolution of hear-
ing [31] or to technology (e.g. in terms of sampling rate in a digital system). Feedback can indeed take place in the audio domain, as it happens in Larsen tones, where acoustic information from a loudspeaker is used as input to a microphone, diffused again through the loudspeaker and so on. But the rate of the signal fed back into System’s input can be sub-audio, i.e. occurring in the control domain when, for example, information is extracted from sound and is used to drive processes of sound transformation. In Figure 2 the thick line (audio) represents the audio flow re-injected into System in case of audio feedback. Related to control in the case of control feedback, an analysis component (Analyzer) is required in order to perform the extraction of information from audio and to generate a control flow (control). The latter is then used to turn sound into control signals that are used to control digital devices (e.g. to turn sound into control signals that are used to control digital devices (e.g. http://malcolmriddoch.com/

such as amplitude, spectral flatness, spectral centroid and pitch tracking which are processed in a control DSP engine (what he calls the "brain"), and finally used to perform control DSP algorithms at the control level (see Figure 3). An example of adaptive systems is the Audible Ecosystems project, a remarkable set of projects. In order to classify feedback systems, each category deals with the openness to environment. The sound is analyzed with a feature extraction algorithm: the control signal thus obtained is used to drive digital processes of sound synthesis based on psychoacoustic criteria. The resulting audio is reintroduced in the environment, thus captured and analyzed again, recursively. A fundamental condition for the system to "survive" (i.e. remain active) is to be coupled with the environment (see Figure 3).

2.4. Trigger modality: internal/external

In positive feedback systems, the initial conditions are particularly relevant, as some energy is injected in order to trigger the amplifying feedback loop. As shown in Figure 2, this initialization step (trigger, to speak with computing parlance) can result from the internal activity of the system itself or be operated by some external agent. In relation to the first case, an analog system always has residual noise in its components, that can be used as the only source of alimentation. In the digital domain, it may be possible to have numerical garbage within the software that can be used in a similar way. Otherwise, the system can e.g. be excited with an impulse and then let run with no external sources. In either case, the system features some kind of external perturbation as an element to alter its spontaneous behavior. This situation could also be considered as a particular hybrid case for the audio/control category, as the external signal is often preferred to a fixed set of instructions. Also, as in any other kind of control loop, it is not easy to define the value for the category. Starting from the categories discussed above, it is possible to describe feedback systems by encoding the values that each system assumes. In a relevant number of cases it is not easy to define the value for the categories. On one side, the system may appear ambiguous to the observer because of its complexity or of the lack of information on its internal processes. On the other side, hybrid configurations are indeed possible, that do not clearly allow to place the system with respect to the category (the typical case being that of mixed analógital/digital configurations). In order to classify feedback systems, each category can receive a value in the range [0, 1], where 0 represents the opposite feature, and 1 the case of un-ambiguous, hybrid systems in that category. In short, a feedback system can be represented by a ternary string encoding its properties. Table 1 shows a comparison of the previously discussed examples. Columns represent categories, rows show values for each example. As there are six categories to be taken into account, each one with three possible values, the total number of combinations is very typicallly the case when System is coupled with Environment. In this case, it may be possible of extracting information from Environment (see the dashed meta path in Figure 2) in order to adapt its state to changing environment and system conditions. This is performed by analyzing system states at the control level (Analyze) but, as it determines a change to a different state of the system and not only a variation in its actual state, is placed at a higher level (hence the name meta). Two examples are shown in Figure 3: the performer is present: as s/he is forced to dynamically interact with the dynamical machine system, in turn, the audio signal output by the analog device is often preferred to a fixed set of instructions. Also, as in any other kind of control loop, it is not easy to define the value for the category. Starting from the categories discussed above, it is possible to describe feedback systems by encoding the values that each system assumes. In a relevant number of cases it is not easy to define the value for the categories. On one side, the system may appear ambiguous to the observer because of its complexity or of the lack of information on its internal processes. On the other side, hybrid configurations are indeed possible, that do not clearly allow to place the system with respect to the category (the typical case being that of mixed analógital/digital configurations). In order to classify feedback systems, each category can receive a value in the range [0, 1], where 0 represents the opposite feature, and 1 the case of un-ambiguous, hybrid systems in that category. In short, a feedback system can be represented by a ternary string encoding its properties. Table 1 shows a comparison of the previously discussed examples. Columns represent categories, rows show values for each example. As there are six categories to be taken into account, each one with three possible values, the total number of combinations is very typical.
high, resulting in $3^6 = 729$ types of feedback system. As a consequence, such an analytical, even if minimal, framework allows to include many different works, sharing a common reference to feedback but coming from different traditions and practices, and to specify their mutual relations.

4. CONCLUSIONS

The use of feedback clearly identifies a specific group of works that, starting from the ’60s, have explored, with a various degree of awareness, a tightly related set of notions, such as nonlinearly, circular causality, interdependence, self-organization, complexity and emergence. Even if the external boundaries that define this corpus of works are, if not clear-cut, anyhow sufficiently evident, still feedback-based musical systems show a wide internal variety, that must be tackled in order to shed light on their richness. The discussed typology, resulting from the set of six general categories, is intended to provide an analytical tool that would allow description and comparison among feedback-based audio/music works. The categories can be further expanded. As an example, interaction includes many different performing modalities that can lead to other subcategories. On the other side, there is indeed a trade-off between analytical detail and overall manageability. Our future work will focus on expanding the corpus of examples in order to test the typological device and eventually modify the schema and/or redefine/refine the categories.

5. REFERENCES


THE PRESENCE OF A TRI-POLAR DYNAMIC IN SONIC ART INSTALLATION

Samantha Horseman
University of Huddersfield
Queensgate
Huddersfield
West Yorkshire
HD1 3DH

ABSTRACT

This paper explores the concept that in the mediation of a sonic artwork, the individual becomes a complete component of the work, and syntactic activation of the dormant syncretic potential of the artistic materials. From this basis, it considers further issues of temporality and semiotics referring to the work of Kramer, Nattiez, Petrie and Deleuze. In viewing a sonic artwork as comprising of a tri-polar dynamic between sonic elements, physical elements and the perceiving individual, it continues to contextualise the creative processes and relationships of the sonic artist within this framework informed by my own creative practice as a sonic installation artist.

1. SYNCRETISM IN SONIC ART INSTALLATION

In considering artworks currently allied with the genre of sonic art, its particular artistic resources and influences becomes easily evident. For example, there is an incredible breadth of variance in the artistic materials used between the renowned landmarks in the work of Bill Fontana, the presence of Thelonius ‘Spoo’ Spem in Allum (1570) in Janet Cardijn’s The Forty Part Motet (2001) or the influence of minimalist art in the work of Max Neuhaus. Yet these works are all frequently discussed under the umbrella genre of sonic art. And so, perhaps the difficulties encountered when considering the perception and dynamic relationships of sonic art are symptomatic of this seemingly infinite variability in the artistic materials at the sonic artist’s disposal. David Toop has outlined sonic art as, ‘sound combined with visual practices’ in his catalogue for the sonic art exhibition, Sonic Boom [22]. However, it is not within the scope of this text to engage in a detailed discussion regarding what exactly constitutes a sonic artwork. I am not arguing a position that Toop’s pluralism has already incited many good texts on this topic such as those by Alan Licht [13] and Tony Gibbs [9]. I intend to work from the premise that the sonic artwork constitutes a new species of art and it is within this framework that a revelatory narrative arises. In embarking on this revelatory narrative, it may be suggested that the individual becomes a completing signifying force alongside the aforementioned aural and physical components of an artwork. Therefore the presence of a tri-polar dynamic has become, for me, a most intriguing crux within sonic art. In understanding sonic art installation as a syncretic art form, assimilated from these three signifying forces, we may continue to ask: what are the implications of this conceptual framework on issues regarding temporality in sonic art installation?

2. TEMPORALITY IN THE DIALOGUE BETWEEN SONIC ART AND ITS AUDIENCE

The existence of multiple temporalities operating within music and the visual arts has long been in discussion amongst contemporary circles. As a genre utilizing both sound and visual practices, it is therefore logical to assume that sonic art can also be seen as potentially possessing multiple or in some cases, multiple temporalities accompanying the sonic artwork in this context concerning the individual directly! I wish to further explore temporality accompanying the individual, referring to J.T Fraser’s Umwelt model of temporality as discussed by J. Kramer in his seminal text, The Time of Music: New Meanings, New Temporalities, New Listening Strategies [11]. Fraser’s hierarchy model of time outlines levels of temporal consciousness or Umwelts in which he describes temporality in the evolution of nature and the human consciousness. This conceptual framework is often posited against the arts, however it is especially relevant to our discussion here on account of its correlation to conscious thought. When immersed in a work of sonic art, the individual is in space and time alongside aural and physical phenomena, and it is within this framework that a revelatory narrative occurs. I would like to explore the idea of a sonic artwork going from existing in a state of syncretic consciousness of the perceiving individual and the sonic artwork although it may not be time conducive to encompass further, on an exhaustive list of these here. Therefore, in the absence of a perceiving individual, a sonic artwork possesses a great deal of potential conceptual energy. It is in the context of this continually evolving soundscape. With each loud speaker inhabiting its own pendulum and an on-going variation in swing patterns the artworks current allied with the genre of sonic art is used between the renowned landmarks in the work of Bill Fontana, the presence of Thelonius ‘Spoo’ Spem in Allum (1570) in Janet Cardijn’s The Forty Part Motet (2001) or the influence of minimalist art in the work of Max Neuhaus. Yet these works are all frequently discussed under the umbrella genre of sonic art in current literature. And so, perhaps the difficulties encountered when considering the perception and dynamic relationships of sonic art are symptomatic of this seemingly infinite variability in the artistic materials at the sonic artist’s disposal. David Toop has outlined sonic art as, ‘sound combined with visual practices’ in his catalogue for the sonic art exhibition, Sonic Boom [22]. However, it is not within the scope of this text to engage in a detailed discussion regarding what exactly constitutes a sonic artwork. I am not arguing a position that Toop’s pluralism has already incited many good texts on this topic such as those by Alan Licht [13] and Tony Gibbs [9]. I intend to work from the premise that the sonic artwork constitutes a new species of art and it is within this framework that a revelatory narrative arises. In embarking on this revelatory narrative, it may be suggested that the individual becomes a completing signifying force alongside the aforementioned aural and physical components of an artwork. Therefore the presence of a tri-polar dynamic has become, for me, a most intriguing crux within sonic art. In understanding sonic art installation as a syncretic art form, assimilated from these three signifying forces, we may continue to ask: what are the implications of this conceptual framework on issues regarding temporality in sonic art installation?

2. TEMPORALITY IN THE DIALOGUE BETWEEN SONIC ART AND ITS AUDIENCE

The existence of multiple temporalities operating within music and the visual arts has long been in discussion amongst contemporary circles. As a genre utilizing both sound and visual practices, it is therefore logical to assume that sonic art can also be seen as potentially possessing multiple or in some cases, multiple temporalities accompanying the sonic artwork in this context concerning the individual directly! I wish to further explore temporality accompanying the individual, referring to J.T Fraser’s Umwelt model of temporality as discussed by J. Kramer in his seminal text, The Time of Music: New Meanings, New Temporalities, New Listening Strategies [11]. Fraser’s hierarchy model of time outlines levels of temporal consciousness or Umwelts in which he describes temporality in the evolution of nature and the human consciousness. This conceptual framework is often posited against the arts, however it is especially relevant to our discussion here on account of its correlation to conscious thought. When immersed in a work of sonic art, the individual is in space and time alongside aural and physical phenomena, and it is within this framework that a revelatory narrative occurs. I would like to explore the idea of a sonic artwork going from existing in a state of syncretic consciousness of the perceiving individual and the sonic artwork although it may not be time conducive to encompass further, on an exhaustive list of these here. Therefore, in the absence of a perceiving individual, a sonic artwork possesses a great deal of potential conceptual energy. It is in the context of this continually evolving soundscape. With each loud speaker inhabiting its own pendulum and an on-going variation in swing patterns the artworks current allied with the genre of sonic art is used between the renowned landmarks in the work of Bill Fontana, the presence of Thelonius ‘Spoo’ Spem in Allum (1570) in Janet Cardijn’s The Forty Part Motet (2001) or the influence of minimalist art in the work of Max Neuhaus. Yet these works are all frequently discussed under the umbrella genre of sonic art in current literature. And so, perhaps the difficulties encountered when considering the perception and dynamic relationships of sonic art are symptomatic of this seemingly infinite variability in the artistic materials at the sonic artist’s disposal. David Toop has outlined sonic art as, ‘sound combined with visual practices’ in his catalogue for the sonic art exhibition, Sonic Boom [22]. However, it is not within the scope of this text to engage in a detailed discussion regarding what exactly constitutes a sonic artwork. I am not arguing a position that Toop’s pluralism has already incited many good texts on this topic such as those by Alan Licht [13] and Tony Gibbs [9]. I intend to work from the premise that the sonic artwork constitutes a new species of art and it is within this framework that a revelatory narrative arises. In embarking on this revelatory narrative, it may be suggested that the individual becomes a completing signifying force alongside the aforementioned aural and physical components of an artwork. Therefore the presence of a tri-polar dynamic has become, for me, a most intriguing crux within sonic art. In understanding sonic art installation as a syncretic art form, assimilated from these three signifying forces, we may continue to ask: what are the implications of this conceptual framework on issues regarding temporality in sonic art installation?
In line with the very nature of evolution itself, the occurrence fleetingly before temporal consciousness evolves. For example, Eotemporality is likely to only occur along a linear progression, order and affective relationships successions, moving in different directions, are presented as it were at once, in the same composition [11]. This notion can clearly be extended to apply to the sonic artwork on account of its synesthetic nature and the capability of both aural and physical elements to exist and display multiple facets of temporality mediated in the linear coexistence of the individual. Subsequently, Biotemporality may be seen to occur as we cognitively ‘lock on’ to the work. The resultant dialogue between aural and physical phenomena and the individual may initially regard the intrinsic characteristics of artistic media and the comprehension of meaningful syntax within and between them. Consequently, a sense of temporality progressively affects the individual, but several different intrinsic relationships, moving in different directions, are presented as it were at one time (or rather, simultaneously occurring) [10]. It is not my proposal that an individual would not necessarily occur in stages of equal relevancy to our discussion however. Perhaps it would be useful to reference Nattiez and in particular, his tripartition model of semiotic analysis [15]. Not only does this serve as a useful framework with which to discuss temporality, but it also provides an introductory model from which to consider semiotics and sonic artwork.

It is not my proposal that an individual — who, in reaching Nootemporal existence that may even exist subconsciously, the individual is most certainly an esthesic and neutral levels of artistic semiosis, then the individual is to be argued as having a place in both the esthetic and neutral levels of artistic semiosis, then the Nootemporal state of the individual renders a neutral existence of the artwork as impossible. In addition to this argument, if the poietic strategies used in its creation are in part driven by a shared cultural history, it would be useful to reference Nattiez and particularly, his tripartition model of semiotic analysis [11]. The concision of an artwork means that, even in the case of clearly referential physical phenomena, all of which can vary from spatial to sculptural; purely anecdotal to wholly abstract. As such, the boundaries of this conceptual potentiality are vast with their exact interpretation at the discretion of each different mediating individual. The literal individuality of this third and final signifying force within the sonic, the poetic and mediatic individual. The sonic and physical media have the potential to act as meaningful signifiers, which are then activated by the mediating individual on account of their presence in the sonic artwork. The sonic and physical phenomena can consist of both discrete and non-discrete values combined with a huge possibility of discursive nature. As such, the boundaries of this conceptual potentiality are vast with their exact interpretation at the discretion of each different mediating individual. The literal individuality of this third and final signifying element of the sonic artwork means that, even in the case of clearly referential artistic material, the outcome of the artwork experience is largely a matter of hermeneutics. Therefore, when attempting to discuss aspects of sonic and physical phenomena and the individual, it is most important to discuss the dispositional scope of artistic media of the sonic artwork.

As discussed above, the proposed evolution of temporal consciousness occurring in the mediating individual is not as straightforward as a pure experience, as discrete phenomena of mind and content, the semantic potentiality of the truly open-ended, inconclusive, emergent quality of interactive art. This gives rise to the question of how time and space are to be contextualized in the presence of an individual, their personal history and culture conditioned responses. It is possible to conceptualize that an individual experiences the boundaries between Eotemporality as distinct. In line with the very nature of evolution itself, the evolution of the individual’s revelatory narrative occurs and that sonic art kinetically ‘becomes’ in the presence of a third signifying force: the individual who, in achieving a virtual state of likeness, completes the syncretic artistic whole. As Ascott writes: ‘It is the action of observing that creates meaning – but also provides a parallel to the interaction that is both temporal and spatial because the individual is the basic process’. Here, the sonic artwork is fully engaged in a dialogue with the individual in the widened context of their processes and ‘unique personal history’ [11]. It is within this dialogue that the disparate components of the sonic, physical and self begin to function as an artistic whole conceptually greater than the sum of its parts and reminiscent of the Gesamtkunstwerk aesthetic. Here the work is mediated through both intrinsic and extrinsic facets, experienced as aural and physical, and non-cochlear, retinal and non-retinal. Perhaps, it is in this light that the focus of sonic art as a hermeneutically hinged art form becomes apparent, in the acknowledgment that it becomes kinetically activated when re-contextualized in the presence of an individual, their personal history and culture conditioned responses.

As discussed above, the proposed evolution of temporal consciousness occurring in the mediating individual is not as straightforward as a pure experience, as discrete phenomena of mind and content, the semantic potentiality of the truly open-ended, inconclusive, emergent quality of interactive art. This gives rise to the question of how time and space are to be contextualized in the presence of an individual, their personal history and culture conditioned responses. It is possible to conceptualize that an individual experiences the boundaries between Eotemporality as distinct. In line with the very nature of evolution itself, the evolution of the individual’s revelatory narrative occurs and that sonic art kinetically ‘becomes’ in the presence of a third signifying force: the individual who, in achieving a virtual state of likeness, completes the syncretic artistic whole. As Ascott writes: ‘It is the action of observing that creates meaning – but also provides a parallel to the interaction that is both temporal and spatial because the individual is the basic process’. Here, the sonic artwork is fully engaged in a dialogue with the individual in the widened context of their processes and ‘unique personal history’ [11]. It is within this dialogue that the disparate components of the sonic, physical and self begin to function as an artistic whole conceptually greater than the sum of its parts and reminiscent of the Gesamtkunstwerk aesthetic. Here the work is mediated through both intrinsic and extrinsic facets, experienced as aural and physical, and non-cochlear, retinal and non-retinal. Perhaps, it is in this light that the focus of sonic art as a hermeneutically hinged art form becomes apparent, in the acknowledgment that it becomes kinetically activated when re-contextualized in the presence of an individual, their personal history and culture conditioned responses.

As discussed above, the proposed evolution of temporal consciousness occurring in the mediating individual is not as straightforward as a pure experience, as discrete phenomena of mind and content, the semantic potentiality of the truly open-ended, inconclusive, emergent quality of interactive art. This gives rise to the question of how time and space are to be contextualized in the presence of an individual, their personal history and culture conditioned responses. It is possible to conceptualize that an individual experiences the boundaries between Eotemporality as distinct. In line with the very nature of evolution itself, the evolution of the individual’s revelatory narrative occurs and that sonic art kinetically ‘becomes’ in the presence of a third signifying force: the individual who, in achieving a virtual state of likeness, completes the syncretic artistic whole. As Ascott writes: ‘It is the action of observing that creates meaning – but also provides a parallel to the interaction that is both temporal and spatial because the individual is the basic process’. Here, the sonic artwork is fully engaged in a dialogue with the individual in the widened context of their processes and ‘unique personal history’ [11]. It is within this dialogue that the disparate components of the sonic, physical and self begin to function as an artistic whole conceptually greater than the sum of its parts and reminiscent of the Gesamtkunstwerk aesthetic. Here the work is mediated through both intrinsic and extrinsic facets, experienced as aural and physical, and non-cochlear, retinal and non-retinal. Perhaps, it is in this light that the focus of sonic art as a hermeneutically hinged art form becomes apparent, in the acknowledgment that it becomes kinetically activated when re-contextualized in the presence of an individual, their personal history and culture conditioned responses.

As discussed above, the proposed evolution of temporal consciousness occurring in the mediating individual is not as straightforward as a pure experience, as discrete phenomena of mind and content, the semantic potentiality of the truly open-ended, inconclusive, emergent quality of interactive art. This gives rise to the question of how time and space are to be contextualized in the presence of an individual, their personal history and culture conditioned responses. It is possible to conceptualize that an individual experiences the boundaries between Eotemporality as distinct. In line with the very nature of evolution itself, the evolution of the individual’s revelatory narrative occurs and that sonic art kinetically ‘becomes’ in the presence of a third signifying force: the individual who, in achieving a virtual state of likeness, completes the syncretic artistic whole. As Ascott writes: ‘It is the action of observing that creates meaning – but also provides a parallel to the interaction that is both temporal and spatial because the individual is the basic process’. Here, the sonic artwork is fully engaged in a dialogue with the individual in the widened context of their processes and ‘unique personal history’ [11]. It is within this dialogue that the disparate components of the sonic, physical and self begin to function as an artistic whole conceptually greater than the sum of its parts and reminiscent of the Gesamtkunstwerk aesthetic. Here the work is mediated through both intrinsic and extrinsic facets, experienced as aural and physical, and non-cochlear, retinal and non-retinal. Perhaps, it is in this light that the focus of sonic art as a hermeneutically hinged art form becomes apparent, in the acknowledgment that it becomes kinetically activated when re-contextualized in the presence of an individual, their personal history and culture conditioned responses.

As discussed above, the proposed evolution of temporal consciousness occurring in the mediating individual is not as straightforward as a pure experience, as discrete phenomena of mind and content, the semantic potentiality of the truly open-ended, inconclusive, emergent quality of interactive art. This gives rise to the question of how time and space are to be contextualized in the presence of an individual, their personal history and culture conditioned responses. It is possible to conceptualize that an individual experiences the boundaries between Eotemporality as distinct. In line with the very nature of evolution itself, the evolution of the individual’s revelatory narrative occurs and that sonic art kinetically ‘becomes’ in the presence of a third signifying force: the individual who, in achieving a virtual state of likeness, completes the syncretic artistic whole. As Ascott writes: ‘It is the action of observing that creates meaning – but also provides a parallel to the interaction that is both temporal and spatial because the individual is the basic process’. Here, the sonic artwork is fully engaged in a dialogue with the individual in the widened context of their processes and ‘unique personal history’ [11]. It is within this dialogue that the disparate components of the sonic, physical and self begin to function as an artistic whole conceptually greater than the sum of its parts and reminiscent of the Gesamtkunstwerk aesthetic. Here the work is mediated through both intrinsic and extrinsic facets, experienced as aural and physical, and non-cochlear, retinal and non-retinal. Perhaps, it is in this light that the focus of sonic art as a hermeneutically hinged art form becomes apparent, in the acknowledgment that it becomes kinetically activated when re-contextualized in the presence of an individual, their personal history and culture conditioned responses.
acknowledge that unstable relationships within the tri-polar dynamic are unavoidable. Jean-Jacques Nattiez acknowledges the difficulty for stable relationships to exist with the infinite multiplicity of interpreters, signified in music within his 1987 text, Music and Discourse: Toward a Semiology of Music [15]. Nattiez references the work of Charles Sanders Peirce as semiotically accounting the unstable nature of what has been discussed here as the individual’s personal Nocturnal history when considering semiotics. Peirce’s triangular model of semiotics in languages maintains that, “A sign is anything which is related to a second thing, its object in respect to quality, in such a way as to bring a third thing, its interpreters, into relation to the same object, and that in such a way as to bring a fourth into relation to that same object in the same form, ad infinitum” [15].

In Nattiez’s discussion of Peirce he observes that the virtual nature of the signifying object exists only, ‘within and outside the infinite multiplicity of interpreters, signified in music within his 1987 text, Music and Discourse: Toward a Semiology of Music [15]. Nattiez references the work of Charles Sanders Peirce as semiotically accounting the unstable nature of what has been discussed here as the individual’s personal Nocturnal history when considering semiotics. Peirce’s triangular model of semiotics in languages maintains that, “A sign is anything which is related to a second thing, its object in respect to quality, in such a way as to bring a third thing, its interpreters, into relation to the same object, and that in such a way as to bring a fourth into relation to that same object in the same form, ad infinitum” [15].

Peirce’s diagram shows his model of semiotics consisting of a sign having an infinite number of possible interpreters, signifying in a signified virtual object. This is how the theory likely to transpire if we are to consider the sonic artwork as essentially a virtual, liminal space. The upward trajectory on the diagram through which the object is formed (from the sonic sign, physical sign and individual) represents the brief transition through Prototemporality in which the disparate forces of sonic, physical and self unite as an artistic unit. The further evolution of temporal consciousness that eventually reaches Nocturnal occurs as the interpreters on the upper plane of the diagram follow a line of flight. Also, the potential for the individual to engage in an ongoing narrative of revelation within the tri-polar dynamic fits neatly with the lines of flight described as part of the rhizome. As such, we are given the impression that there is a transformative element within the dynamic as the three signifying forces collide: “The diagrammatic or abstract machine does not function to represent, even something real, but rather constructs a real that is yet to come, a new type of reality” [6]. And so this diagram presents the neutral and esthetic levels collide and potentially belonging to one infrastructure when viewed in this context.

Figure 3 shows singular, fixed lines of flight. However, it is important to recognise that these diagrams are intended to represent just one possible semantic chain for the sake of making the conceptual crux of the rhizomatic argument is that it represents multiple lines of semantic flight, capable of journeying in any direction, with multiple points of entry and exit. However, to attempt to illustrate all potential rhizomatic lines of flight would of course be impossible and more importantly would result in a chaotically unintelligible diagram. It is here that I would suggest the interpretant levels of the diagrams are viewed with consideration of what Delueze and Guattari refer to as the first principle of the rhizome: “the thought of the rhizome is that of a network in which it is impossible to connect anything other, and must be, with no point plotted or order fixed [6].

There is currently one more facet that could be explored diagrammatically: the poetic. Figure 4 takes into account our previous discussion concerning the past of the sonic artwork, in which the artist develops physical and sonic expressions of a conceptual crux resulting in what Nattiez refers to as the artistic trace. It

Figure 3
extends the base level of figure 3 to include a new lower level of possible semantic chains: interpretations that led to the culmination and choice of artistic media as it exists in the sonic artwork. In other words, it includes the creative processes that comprise the sonic artwork’s past in a process of poiesis. Like the upper level on its mirror image, the lines of flight here exist in the Nootemporal state of the creative artist, with their creative practice influenced by their own personal history and individuality. Now we can see the diagram as showing sonic art as having three states of existence: the poetic level existing in the artwork’s past, the potential level of infinite interpretants yet to be realized and the actual lines of semantic flights occurring within the tri-polar dynamic.

Based on these notions, it could be possible to see a more tangible connection between the artist and his audience. Could it be argued that the sonic artwork acts as a gateway between the two individuals engaged in a poetic tri-polar dynamic and esthetic tri-polar dynamic respectively. In this context, the sonic artwork can be seen as a plateau in which nodal points of collision are possible. It could then be argued that both lower and upper levels are part of the same semantic rhizome in which the artist has an exit point at the Neutral level and the mediating individual has an entry point. During the poetic process the artist has travelled a semantic rhizome in the development of the piece, culminating in the artistic trace. However, this does not necessarily mean that the rhizome ends there, as Deleuze and Guattari may also argue. Instead, the mediating individual enters to continue the rhizome, deterritorializing from the artists original rhizomatic pathway. The sonic artwork provides a context in which these nodal departures and collisions occur between the artist, their artwork and the sonic art audience. In his book, Art Encounters: Delueze and Guattari (2006), Simon O’Sullivan discusses the relationship between art and the philosophical texts of Deleuze and Guattari. He verifies my account of the tri-polar dynamic between the sonic artist, their art and the audience above stating, ‘The virtual or actualization of the virtual, is then the creative act – precisely the production, or actualization of difference and thus diversity from a pre-existing field of potentialities’ [17]. Speaking on a broader level, Brian Massumi defines meaning in art as a ‘network of enveloped material processes’ in his text, A User’s Guide to Capitalism and Schizophrenia: Deviations from Delueze and Guattari (1992). He continues to state that meaning can be seen as ‘the envelopment of a potential, a contraction of the past, and the future, in an event/object that has the capacity to affect or be affected’ [14]. Here it is the work of ‘interpretation’ to unravel these ‘virtual’ processes encapsulated in the object. Whilst speaking of a meaning on a general level, this viewpoint has clear resonances with the sonic artwork in this context [14].

In considering sonic art installation as a genre hinged on the tri-polar dynamics between self, sonic and physical components, we can begin to understand the significance of hermeneutics in its analysis: the same artwork becomes capable of multiple identities dependant on the particular individual completing the syncretic whole. This multiplicity seems to be a most relevant topic to many of today’s intermedia art forms. Propelled by an ‘age of confront’ [26], which refers to the idea that information concerning all cultures and subjects is readily available across geographical and temporal boundaries, our arts culture is evolving — almost cybernetically and self-perpetuating — into an expanding network of interconnected art forms and with it, an exciting rhizomatic network of taxonomical possibilities seems to be the replacement of clear-cut genres for the future of intermedia arts.

5. REFERENCES

**ABSTRACT**

In this paper we propose the design and implementation of a Turing Test (TT) for the research of the singing voice. Although the TT is mainly related to the research field of Artificial Intelligence (AI), being used both as a criterion and an operational guide by the scientists of this field, with the present paper we attempt to introduce a rather different approach to the TT. Given the fact of various disputes over the validity of the TT as a criterion of AI, one might argue that the TT is nothing more than a ‘philosophical fossil’, a left-over and remainder of past and outdated philosophical assumptions about the nature of human intelligence. The problem of subjectivity in the results of TT experiments has strengthened the question about the usefulness of the TT as a research means. Our goal is to introduce a new scope for the use of the TT not as a criterion of intelligence but as an ‘instrument’ for tracing certain features of human judgment in various fields. Pretty much in the fashion of a transcendental philosophical status we face the construction of an operational TT procedure in which what is judged is judgement itself. Specifically, in the present paper we attempt to exhibit the way in which a TT can be used to trace and highlight features of human judgment regarding the singing voice.

1. INTRODUCTION

1.1. Turing Test and Artificial Intelligence

The TT is named after the famous mathematician Alan Turing. After the TT was first introduced in 1950, it was much discussed and taken up by many researchers in the field of artificial intelligence. This interest in the TT is related to the idea that the TT is a criterion for the evaluation of artificial intelligence.

Artistic creativity seems also to satisfy those who demand a holistic approach to the problem of evaluating an observed action in terms of intelligence [7], [15]. From all the forms of art, music seems to be the one that most attracts the interest of the TT designers. After all, as mentioned by Gareth Loy, the abstract nature of music, thought to be its susceptibility to mathematical and logical analysis, makes it an ideal medium for testing the computational capabilities of machines [31, 32].

Wiggins report that one of the basic motives for the use of music as a medium for the TT has led to a variety of ‘musical TTs’ [46, 47]: ‘Musical Directive Test’ (MDT), ‘Musical Output Test’ (MOT), ‘stylistic MDIT’s’, ‘musical Total TT’s’ (musical TT), the musical version of the Harnad’s ‘TotalITT’. What use could all this research have?

2. THE TURING TEST: FROM A CRITERION TO A COMMENT. FROM A COMMENT TO A RESEARCH ‘INSTRUMENT’

Apart from those who propose, design and organise TT’s, there are also researchers who doubt the efficiency of the TT as a criterion for intelligence. Various attempts against the TT can be divided into four major problems. First there is the problem of what we could call ‘copyright of intelligence’ or ‘authorship’. When the critics in a TT observe an action that they characterise as intelligent, to whom should they attribute the origins of this action? To the observed computing system or to its programmers? There is always the possibility of ‘mimeis’ [49] and in this case, what is intelligent is not the observed system but its programmers [16]. Ariza thinks of the TT as insufficient due to the fact that it encloses the idea of deception [3]. Even some researchers involved in the TT research deny that deceiving the critics has any relation with the potential of a computer to present intelligent behaviour [3, 39]. This problem of deception regarding the authority was already mentioned by Turing himself [46] and can be related to what Searle attempts to state with his ‘Chinese Room Argument’: pure syntax can produce the same behavioural result with meaning [34]. Syntax can deceive us and pass as being meaning (therefore intelligent treatment of symbols) [16]. As Boden states, the success of a machine in a TT is a matter of our own political and ethical decision [14].

The latter statement brings us to the second problem of the TT, which is the unavoidable subjectivity in the judgment of the critics. In a Wittgensteinian manner we would say that the critics participate to a certain “form of life” therefore their judgment is guided by the values of this “form of life”. On can find a similar line of critique in the work by Halpern [16], while Miche stresses the importance of “social intelligence” and proposes a ‘TT for social intelligence’ [31]. Again, Turing had already foreseen this problem when he mentioned that surprise would not occur with our social stereotypes-can easily be considered as originality [46]. The effects of cultural subjectivity were shown clearly in the mistaken judgment reported by Halpern [16] considering a Loebner Test that took place in 1991.

The subjectivity of human judgment is doubled when we adopt for instance music as a medium of our TT. This is actually the third problem regarding the TT which is specified as a problem of all the ‘artistic TTs’: an aesthetical subjection which presents its self both as a subjection in the notion of art and as a subjection of the artistic taste. Dalig and Safrin have actually organised a series of TTs the results of which showed that critics’ aesthetical preferences affected their judgment regarding the intelligence of the participants [12]. Wiggins observes that in a ‘musical TT’ the definition and tracing of intelligence results in a problem equal to the problem of defining ‘good music’: a problem of aesthetics [48]. Similar views are expressed by Laurie Spiegel and David Cope [11].

But embracing art as a medium for the TT can reveal another problem. By giving to the TT a musical ‘face’ researchers might do justice to Turing’s initial proposal to introduce the TT as a medium for testing the capabilities of machines. Back in the 16th century Shaftesbury observed that there can be aesthetic objects which are not artistic objects, they are not made by humans at all and are simply physical objects made by nature [5]. Shaftesbury’s observation points at something that in modern philosophy is marked out by Dennett as ‘intentional stance’: ‘humans’ tendency to personalize everything, therefore to attribute intentional, aesthetic etc) to entities that are considered ‘soulless’ like machines [13]. Hofstadter has called our intentional stance towards the AI computing systems an “ELIZA effect” [20], while Ariza points out that intentional stance could make the critics of a ‘musical TT’ attribute intelligence to non intelligent entities [3].

The above analysis of the problems of the TT has turned the focus of our discussion from the evaluated participants to the evaluating interrogator. Our interest seems to be shifted from the ontological evaluation of the evaluated entity to the ontology of our evaluation process itself. In our view this shift can be already found in the proposals for ‘Musical Discrimination Tests’ (Musical DT’s) [12] and ‘reverse TTs’ or ‘fully automated TTs’ of ‘CAPTICH’ [33, 34].

This turn of interest from the participants to the interrogators might do justice to Turing’s initial intentions. Some researchers believe that with ‘Computing, Machinery and Intelligence’ Turing did not intend to introduce a sufficient criterion of intelligence but rather wanted to trigger a discussion about the
subjectivity of human judgment regarding the attribution of intelligence to others [31, 47] or even about the subjectivity of human judgment in general [19]. After all, Turing lived at the end of the 20th century when the least educated people have changed their views regarding the definition of intelligence and the potential for machine intelligence, not because of a technological progress but because of a change in our beliefs [46]. Thus, Turing’s introduction of a TT does not seem to be a ‘manual guide’ for the mechanical reproduction of our subjectivity in general, but rather an exploration of the philosophical potential for the satisfaction of our stereotypes, finally a statement for the arbitrariness of human judgment, a critique of human judgment.

Thus in this rather Kantian way of thinking the TT is the only way in which the TT can be proved fruitful in any research, that is as a means of revealing the way our subjectivity is structured and employed so, in other words as an ‘anthropological tool’. This is exactly the treatment of the TT that we intend to adopt in our proposal for a ‘singing voice TT’.

3. 3. Theoretical background

It is striking that, in the above mentioned research on TT, none TT for singing voice has been carried out until now. There have been only some TTs regarding speech [28, 38, 24, 40]. We propose the design and implementation of a ‘singing voice TT’ as an extension and follow-up work of the psychosocial experiment carried out by the singing voice research group of Kounourtzoglou and Georgaki, for the evaluation of a Greek voice score-to-singing voice score synthesis system [28]. In that experiment, 30 human participants, aged between 20 and 50 years old, evaluated and compared an older Greek diphone database (GR2) produced by using the MBROLA synthesizer with a newly improved and extended Greek diphone database (GR3) produced also with MBROLA. The experiment was actually a test of MOS (Mean Opinion Score) in which participants evaluated two singing voice samples based on GR2 and GR3 respectively according to singing voice ‘qualities’ characterised as ‘vocalness’, ‘naturalness’, ‘intelligibility’ and ‘expressivity’.

However, that experiment lacked for a comparison between synthetic and natural singing voice, since both the evaluated samples were synthetic. GR3 was found better than the GR2 in all aspects but a complete evaluation of GR3 demands its comparison with samples of natural singing voice. It is exactly the comparison between mechanistic (synthetic) and human (natural) samples that could give to our experiment the ‘flavour’ of a TT.

But what could be really evaluated with this TT?

First of all, this is a test that does not aim at the evaluation of intelligence but seems to aim at the evaluation of aesthetical quality. Therefore it already presents a significant difference to what is usually believed to be the original conception of the TT. On the other hand, as we presented in section 2, it seems that Turing’s intentions concerned not the ontology of the evaluated entities but the ontology of the evaluation itself. Therefore the TT should be conceived as a comment on human judgment and could be used as an ‘anthropological tool’ for the exploration of the experimental truth about human judgment. In this sense our proposal is fully aligned with the concept of the TT, aiming actually not at the evaluation of GR3 but at the research on the potential of human judgment.

Thus in this ‘singing voice TT’ that we propose we will shift our interest from the evaluation of the singing voice qualities (‘vocalness’, ‘naturalness’, ‘intelligibility’ and ‘expressivity’) to the factors that govern the attribution of these qualities. Such factors are: gender, age, education, culture and familiarity.

Age is an acknowledged factor for the perception and appreciation of music in general [34, 26]. But also it is a factor acknowledged in the case of perception, appreciation and production of the singing voice [22, 10].

Education and especially music education has been also widely considered as another important factor that affects the perception and production of the singing voice during childhood [21].

Familiarity and specifically familiarity with technology is expected to be one more factor affecting the interlocutors in our ‘singing voice TT’. Stimulus familiarity in general has been interesting the musicologists since the late 60’s [8]. What about familiarity with music technology? There is a research left in the field of AI ethics gives as the initiative to wonder whether the familiarity with technology can be a crucial factor for the perception of a singing voice. Swisher, Dobromir and Chemero organised an ‘embodied TT’ in the field of AI ethics in the case of singing voice. In our view these two directions are interconnected. The first direction is that of a theoretical and finally ‘anthropological’ interest. Our research aims at the exploration of factors that are widely believed to affect human perception of the singing voice: gender, age, education, familiarity and cultural diversity. In this sense, it falls within the scope of cognitive musicology and social psychology of music. The second direction is rather practical since the evaluation of GR3 concerns the field of development and improvement of singing voice synthesizers. Thus it concerns the field of Music Technology. In our view these two directions should not be conceived differently, since imitating the vocal tract could be equal to deceiving the ear and deceiving the ear requires the knowledge of ‘how’ the ear perceives.

Finally, all these three groups of samples will be subdivided to two subgroups: a subgroup with samples of female singing voice and a subgroup with samples of a male singing voice. With this subdivision we wish to explore further the role of gender in the perception of the singing voice.

4. CONCLUSIONS-PERSPECTIVES

The TT has been one of the most popular topics in AI and Philosophy of Mind and, in general, in the field of artificial intelligence. It seems to suffer from serious problems even when the so believed measure of artistic creativity is employed. ‘Authorship’, ‘subjectivity of human judgment’ (regarding intelligence or artistic value) and ‘intentional stance’ are problems which in their appearance make us realize that what can be judged in a TT is judgment itself. In this sense, the use of the TT can be altered from a disputed and rather dubious criterion of intelligence to a successful ‘instrument’ for the introspection of human judgment. Specifically, in this paper we propose a treatment of the TT as a research tool for the introspection of aesthetic judgment. Since no TT has been carried out for the singing voice we intend to explore the ability of TT in being an introspective ‘instrument’ in the case of singing voice perception and evaluation.

Our proposal for a ‘singing voice TT’ aims in two directions that we nevertheless believe should be interconnected. The first direction is that of a theoretical and finally ‘anthropological’ interest. Our research aims at the exploration of factors that are widely believed to affect human perception of the singing voice: gender, age, education, familiarity and cultural diversity. In this sense, it falls within the scope of cognitive musicology and social psychology of music. The second direction is rather practical since the evaluation of GR3 concerns the field of development and improvement of singing voice synthesizers. Thus it concerns the field of Music Technology. In our view these two directions should not be conceived differently, since imitating the vocal tract could be equal to deceiving the ear and deceiving the ear requires the knowledge of ‘how’ the ear perceives.
STRANGE LOOPS IN CFML, A LIVECODER’S RIDDLE

Adam M. Smith  
Santa Cruz, CA, USA  
adam@adamsmith.as

ABSTRACT
The practice of livecoding borrows heavily from the techniques and vocabulary of music and computer programming. In a setting where the design, implementation, execution, and reflective redesign of software systems are simultaneously overlaid and entangled with sonic creativity in the moment, traditional vocabularies fail to offer more than ambiguous metaphor. This paper uses examples from cfml, a minimal livecoding system, to probe the livecoder’s conceptual landscape, revealing unfamiliar structures and processes reminiscent of Hofstadter’s strange loops. Results from even rudimentary practice with an intentionally impoverished tool point at the need for further inquiry into these natively-livecoding concepts that are at the very edge of expression within the terminology of livecoding’s computer music origins.

1. INTRODUCTION
Canonically, livecoding challenges an artist/programmer to interactively develop a software system that will generate musical entertainment for an audience in an real-time fashion. At a relatively obscure extreme of computer music, practitioners of livecoding are left to adopt and awkwardly apply loan words from the parent disciplines of music (e.g. “composition”) and engineering (e.g. “program”). In this work, my goal is to highlight the unique aspects and processes inherent and native to livecoding that are at the very edge of expression within the terminology of livecoding’s computer music origins. When algorithms are defined and redefined in an organic manner [11], it quickly becomes difficult to point out exactly where is the score for a particular piece. Some fundamental things we have yet to name.

Hofstadter sketches a strange loop as “a paradoxical feedback loop” or when “there is a shift that goes beyond far this, something difficult to express in our available musical and engineering terminology.” Livecoding performances are (to single out just one feature) pathologically indeterminant. The flavor of a livecoding piece hinges not only on what new code is injected during performance but when and where that injection occurs with respect to the sonic and computational state of the piece. If the audio stream computed in generative music is only an epiphenomenal shadow of an underlying algorithm, then the succession of ephemeral algorithms at play in a livecoded music piece are but shadows of something we have yet to name.

2. PROGRAMMING IN CFML
At first glance, cfml is a straightforward generative music programming language. Where cfml diverges from the languages that inspired it (described later) is in its use of a strange construct without clear precedent in other languages. The key structure that every expression in cfml assembles is that which the programmer composes, that which composes the music heard, and that which is composed of other instances of the same type: a data/code structure that I call (for lack of a more transparent name) a comp. This is not intended as a universal concept for computer music: comps are simply the peculiar creatures that occur in cfml. To understand the accidental emergence of comps and their unexpected affordances for live performance, it is important to understand the original intent of cfml and its surface features.

2.1. Development Context
Originally developed for use as a supporting example in a lecture for an undergraduate computer science audience, cfml was conceived in late 2009 as the Context Free Musical Language. Cfml was to be a lightweight sonic analog of Chris Coyne’s cf notebook and a language and interpreter for Context Free Design Grammars in the domain of two-dimensional visual arts and Mikhail Cherstvenk’s Structure Synth [3] (a three-dimensional adaptation of cfml). The grammar constructs used in each of these tools are derived from shape grammars, a concept from architecture [1]. Where shape grammars were historically used descriptively (as a way to reflect on spatial decomposition and the reuse of geometric motifs), the intended use of grammars in cfml and Structure Synth is the generation of new artifacts. As such, these systems include pragmatic features (such as context-sensitive, automatic termination of would-be infinite recursion) that distance them from their context-free namesakes in the Chomsky hierarchy of formal languages [2].

The analogy between the visual and sonic arts embodied by cfml and cf notebook is not only a metaphor for how these practices work in tandem, but also a way to reflect on the design process in cfml and its surface features: deceptively simple to use but richly expressive. How the music emerges from the algorithms is easily recognizable as the domain of generative music, but how reactions to this music turn into code-splinters or where those splinters achieve artistic effect is all unknown.

In this paper, I use examples from my livecoding system cfml [9] as a way to point at those strange structures and processes that seem to be at the core of livecoding. In section 2, I review cfml as a generative music system and show some of the strange structures and processes that seem to be at the core of livecoding. In section 3, I give a walkthrough of the virtual machine at the heart of cfml’s performance engine to explain the particular processes that the system requires a programmer to reason through. Finally, in section 4, I describe a sequence of three studies that portray cfml as a trivial generative music system, an expressive live performance tool, and a curiously abstract artifact beyond the understanding of its creator.

2.2. Literals
The generation of music in cfml starts with literal notes. The simplest type of comp holds a small collection of notes to be played back through Impromptu’s scheduling and synthesis back end. Such a comp is constructed using the literal keyword:

(literal duration note-descriptors)

The duration of a note bundle is expressed in beats and the notes themselves are composed of 5-element lists (c, o, v, d, p) where c is a relative onset (in beats), o is a relative pitch number (used to compute an absolute pitch number given a separately
specified scale), \( v \) is a relative velocity, and \( a \) is relative duration (also in beats).

Comps are performed using the `perform` command, passing the comp, a tempo (in beats-per-minute), and a scale (a list of MIDI pitch offsets within each octave). This expression plays a single half-second note on middle-C through whichever synthesizers are responding on MIDI channel 1.

\[
\text{perform (literal 1 '((0 0 0 0 0))) 120 (pcscale 0 'dorian)}
\]

This slight variation has radically different semantics (note the parentheses):

\[
\text{define beep (literal 1 '((0 0 0 0 0)))}
\]

In the first version, `beep` is bound to the comp returned by `literal`. In the second version, `beep` is bound to a `procedure` that will return a comp when invoked. For consistency purposes, comp-returning procedures are also considered comps. In most cfml programs, nearly all definitions in this second form: defining Scheme procedures that take no arguments (making them context free, in a sense). Being able to delay the evaluation of the body of a procedure until a later time (during which various other global names have been redefined) is critical for livecoding uses of cfml.

2.4. Modifiers

Hand-entering the parameters for a note bundle is tedious. Thus, cfml includes operators for modifying how a pre-defined comp will be performed. Currently, transpose (volume scaling), duration, and tempo (tempo scaling) are the only modifiers:

\[
\text{define beep (literal 1 '((0 0 0 0 0)))}
\]

This expression describes a comp that will perform `beep` at triple length, half volume, and shifted up by two pitches in the working scale:

\[
\text{(tra 2 (vol 1/2 (dur 3 beep)))}
\]

2.5. Combinators

Cfml has three operators (called combinators) for building more complex comps from simpler ones: `after`, `during`, and `choose`. Like modifiers, expressions using combinators can be nested arbitrarily. The following expression yields a comp encoding the nondeterministic performance of three possibilities (a triple-beep, a short chord, or a long beep):

\[
\text{(choose (after beep beep beep) (dur 1/2 (during (tra 2 beep)) (tra 4 beep)) (dur 2 beep))}
\]

Like the other combinators, `after`, is an operator that can be applied to any number of comps. As the name suggests, the comp returned by an `after` expression will resolve in the performance of the first of its arguments followed by successive performance of the remaining elements. For the purposes of sequencing, the duration (first argument) to a `literal` comp is used to determine the length of a note bundle (allowing for bundles with silence or notes that extend beyond the nominal boundaries of a local pattern).

The `during` combinator yields similar results to `after` except that all comps are performed in parallel instead of in a series. The performance length of a comp produced with `during` is the length of the longest comp it performs. Finally, to allow the expression of aleatoric composition rules, the `choose` operator accepts an arbitrary number of comps as arguments and will decide randomly (with a uniform distribution) which one to perform. The length of a comp resulting from `choose` is the length of the randomly chosen comp it performed.

2.6. Live Execution State

At this point, I have introduced all of cfml’s surface features. Any comp can be produced by some combination of literals, modifiers, combinators, and procedure definitions. Fluent performance with this language, however, requires understanding more. During live execution of a cfml program inside of Impromptu, there are four primary types of program state that can effect both how audio is synthesized or which program edits a programmer should consider making to achieve a certain effect.

The first type of execution state is the queue of notes to be played. This data structure, holding a set of fully-resolved MIDI events, is maintained by Impromptu and is not directly visible to the programmer. Once notes enter the queue, they are committed for synthesis and cannot be directly affected by the programmer. The next type of execution state is also an invisible, internal queue. When a comp is delayed in time (through the use of an `after` combinator), a reference to that comp and its intended execution environment are saved in a data structure associated with Impromptu’s callback scheduler. While the programmer cannot easily un-schedule future performance tasks, the programmer does have direct control over the environment in which these future tasks will run. Because of the use of procedure definitions to delay computation, one version of a comp can be made to refer to a future version of itself (a state unheard of in conventional programming tasks).

The third and most obvious type of live execution state is the global symbol table (the structure that stores the mappings created by the compiler). By redefining a modified version of a definition, the newly defined comp will be made available to any procedures attempting to look up the old comp by name (though it is also possible to retain a reference to the older version of the definition, a fact exploited in the example described in subsection 4.3).

The final (and often overlooked) type of execution state is the state of the programmer’s text editor. Even though this state exists outside of composition rules in play at any time, it is still data stored in Impromptu’s memory. Knowing which of the definitions visible on screen is still valid with respect to the global symbol table takes programmer effort (though an alternate graphical interface might ease this burden). When considering a redefinition of an existing procedure definition, the programmer must choose between destructively editing the current definition in place (losing access to the old version) or writing a new definition below (possibly with the help of copy/paste) so that the old definition can be restored or re-edited with ease. Opting to leave more code-at-hand requires time to create new code, consumes valuable screen-space while performing, and increases the cognitive burden of remembering which definitions are active when there are more to choose between.

Mentally tracking each of these types of execution state and using that knowledge to make strategic decisions about what with, how, and when to redefine names in the global symbol table seems to be a skill unique to livecoding.

3. UNPACKING PERFORMANCE

In the preceding discussion, I have referred to compositions as meaning, performance, and execution as if they were nearly separable concepts. In livecoding, however, this separation can never be complete. In this section, I aim to give an accurate description of the distinct artistic and technical processes that are carefully attended to and overlaid terms when talking about cfml at a high level.

When I use cfml as notation, an instrument, or a prop for performance, I am referring to performance as the process of being a liveworker in the moment of play. Second-by-second, my performance will consist of creating and naming comps. When I use cfml as a compiler, interpreter, synthesizer, or computation engine, I’m referring to the performance as in the generative music mode, where a virtual machine is consuming comps as machine instructions, performing the operations they describe, and yielding result data (some of which is directed to a real-time synthesizer).

As comps are the glue between these two very distinct senses of performance in cfml, I should describe how they are created and consumed in more detail.

3.1. Comps as Data Structures, Code Structures

Having initially gotten cfml to a functioning state in an exploratory mode of software development, I must admit that comps, as the centerpiece of cfml’s internal structure in my system, are more the product of accident than conscious design. In a Scala-based reimplementation of the core ideas of cfml, I took the opportunity to reflect on what a comp is and what it means. After this, I best understood comps as data structures that describe a single action to be performed by a strange kind of virtual machine (VM). This reading is supported by cfml’s interpreter being organized as an opcode dispatch loop (albeit for a slightly different language than the programmer sees on the surface).

When a cfml expression such as `after beep (tra 2 beep)` is evaluated (as if it were no different from any other Scheme expression) the result is actually a graph of comps. This is how (a root comp and all of the other comps it points to, transitively) that is actually interpreted by the cfml VM. Depending on the type of the comp, the action performed is sometimes better read as code compilation, high-level composition work, or low-level score realization (i.e. even at the machine level, performance is not a simple concept).

Comps can be read as instructions for how and when to compose new musical material; they are data structures that can be interpreted to yield syntactically or semantically. Contrast this with the cfml expressions seen throughout the paper so far. None of these were really programs for composing music; they were programs for composing comps (in the sense of needing to translate to formal whole). Comps are not just scores or rules for generating scores (one meta-level up), but potentially rules-for-rules-for-rules... for generating scores at an ambiguous metalevel. Cfml expressions say one thing and do another, but seem to end up doing what they say eventually. Understanding the mixing meta-levels of languages within languages and interpreters for interpreters is another facet of the riddle cfml unintentionally presents to those who would use it (and similar systems) fluently.

3.2. VM Instructions

There are six types of comps: four have a relatively clean mapping to cfml’s surface language, one is an understandable abstraction, and the final type stretches the limits of the VM metaphor for cfml’s operation.

The internal state of the cfml VM is encoded in seven registers: `score` (a reference to a single comp to be performed), `tempo` (set in `perform` and altered by `vol`), `velocity scaling ratio altered by `vol`), and altered by `trk`, `time` (logical time in samples that this comp should be performed, incremented by nominal durations), volume (over the scale) and altered by `trk` (a continuation to be called when performance of the current comp completes).
3.2.1. op-literal
Unsurprisingly, op-literal instructions are produced by expressions using the literal operator. That is, when I evaluate (literal 1 "((0 0 1 0 1))"), the result I see is a new list data structure: (op-literal 1 (0 0 1 0 1)) (no sound is produced until this VM instruction is passed to perform, the function that kicks off temporally recursive VM execution).

To execute an op-literal instruction, the VM interprets the note descriptors in the context of the current values of the VM registers (mapping relative pitch to in-scale absolute pitch, determining duration in samples from the current tempo, etc.) and passes the resulting data to Impromptu’s MIDI event scheduler. Knowing the exact duration of the literal data, the VM uses Impromptu’s callback command to schedule the new note (which will begin producing any sounds that might have been scheduled after this one) for the appropriate delay.

3.2.2. op-after
Evaluating an expression like (after beep beep beep) results in a value like (op-after $beep $beep $beep) where $beep is a reference to the current value of $beep in the global symbol table. That is to say, the after combinator dynamically translates its list of arguments into a chain of two-argument op-after instructions.

To execute an op-after instruction, the VM recursively executes the first argument to the instruction, passing the current continuation (extracted with call/cc) for the new value of the r register. This way, when the child instruction completes performance, the VM will resume execution in the state it had upon entrance into this instruction (any transpositions or volume adjustments made in a subordinate comp cannot affect the parent). Upon completion of performance of the left child, the right child is executed.

So far, I have just followed the standard idiom for delayed computation in Impromptu: temporal recursion[10]. The essence of temporal recursion is to perform some operation right now and then schedule a delayed callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later. To overcome jitter and callback to your own code, passing only the remainder of the work to be performed later.

3.2.3. op-during
The relation between during expressions and the op-during VM instruction is analogous to the relation between after expressions and op-after instructions. The only twist is that, instead of waiting for the left child of an op-during construction to execute before executing the right child, both are immediately scheduled for execution in parallel. Once both children have completed (in any order), the execution/termination of the op-during instruction is considered complete.

3.2.4. op-choose
Comps created with the choose combinator (op-choose instructions) are simple enough to execute: a random argument of the instruction is selected and then that instruction is recursively executed.

3.2.5. op-tweak
To capture the environment adjustment (register tweaking) required by the tra, vol, and dis modifiers, a single type of instruction suffices. Evaluating an expression like (tra +2 beep) results in a comp like (op-tweak $tra $beep) where $tra is a dynamically created function that takes the current assignment of the note-parameter related VM registers and computes a new value for each of them.

To execute an op-tweak instruction, the tweaking function is applied to the current VM registers and the child comp is recursively executed by a VM in the newly described state.

3.2.6. (procedure)
The final type of instruction is not associated with an op-code tag. Recall that we previously defined procedures as a kind of comp. Execution of comps that are really procedure references is trivial: run the procedure (assumed to take no arguments) and recursively execute the comp it returns.

When used as part of a livecoding piece, the only kind of procedures passed to the cfml VM are procedures defined by cfml surface language expressions. So when the VM executes this type of comp, it is essentially stepping back a meta-level to ask the outer Scheme interpreter to dynamically compile programmer-entered expressions down to the VM’s language (another strange loop), potentially collecting up recently defined values out of the global symbol table.

3.3. Human Input
Armed with a better understanding of what it means for the cfml VM to perform a comp (a stretched sense of machine instruction execution), I should return to the sense of performance in being a human livecoder, in the moment. I have shown that the programmer’s input is clearly scoped to putting new procedure definitions into named slots in the global symbol table. This operation is non-destructive and cannot modify any comp that is directly referenced by another comp. Outside of inspecting the global symbol table, the VM is insulated from the programmer’s activity.

Consider the evaluation of this snippet of cfml (the focus of subsection 4.2):

```cfml
(define (song) (after phrase song))
```

The symbol song now points to a new procedure. This procedure, when executed, will produce an op-after instruction with the procedures currently pointed to by phrase and song in the global symbol table as its two arguments (explicitly taking the snapshot of the supporting composition rules active at that point in time). With no other changes, execution of (song) always produces identical results. However, if song is redeﬁned during the performance of phrase, the new value of song will be picked up when the VM executes the instruction’s second argument in an attempt to resolve it into a concrete instruction. This is not just a recap of instruction execution; it is an example of the mental code-tracing process that is required to decide when to evaluate a redeﬁnition in order to get it picked up on the appropriate musical cycle.

This required thinking through future-self-reference, yet another strange loop, is simultaneously at the core of what makes livecoding unsettling from traditional pro- grammers and what makes it comprehensible as a medium for live musical expression. It is indeed true that song plays after phrase (just as the surface language code said it would), but it does it as part of a cyclic determination process that wraps up even the livecoder’s internal thought processes!

The ultimate meaning of op-after instructions is tied to meta-circular interpretation of self-referential structures that are generated on the fly from interactively modiﬁed declarative code. This situation is clariﬁed little by the use of continuations in the VM’s implementation (continuations are a programming language feature often explained with reference to time-travel[8]).

Despite this wild complexity, the affordances for human expression are clear: redeﬁning phrase will alter the future of the current piece with the effects becoming immediately audible after the completion of the current version of phrase.

4. TROIS ÉTUDES
Explanations of conceptual landscapes aside, the nominal purpose of cfml is to allow a livecoder to entertain an audience with some simple, synthesized music. In this section, I describe three elementary performances that pro vide a concrete realization of cfml’s expected and not-so-expected features.

To simplify the examples below, imagine the following definition has been evaluated before any performance:

```cfml
(define (run comp) (perform comp 120 (pensala 0 ‘dorian)))
```

4.1. The Escalator, a Disposable Ditty
My first example, The Escalator, is a rather unreliable mechanized stairway. See Figure 1 for a visual analogy of the music generated by the deﬁnitions below. This piece exhibits only the most generic music faculties of the cfml language, and it can be performed without interactive live coding. It was adapted from the self-contained piece, included with the cfml source distribution, that plays once after all of the core deﬁnitions are loaded.

First, I deﬁne a generic bump of the MIDI keyboard for use as a building block in larger comps:

```cfml
(define (bump) (literal 3/2 ‘((0 3 0 1 2) 1 4 2 1 1)))
```

Performing this comp (a bump-generating procedure), by evaluating (run bump), yields a quarter-second gui tar pluck on middle-C. Sequencing four transposed copies of this pattern into a more interesting unit requires the following deﬁnition:

```cfml
(define (lump) (after bump) (tra 2 bump) (tra 5 bump) (tra 4 bump))
```

Before assembling the lumpy staircase, I deﬁne two patterns for the string instrument on another channel.

```cfml
(define (string-step) (lITERAL 2 ‘((0 4 0 2 1 1 4 2 1 1))))
```

```cfml
(define (string-end) (lITERAL 2 ‘((0 4 4 2 1 1 4 4 2 1 1))))
```

Throwing caution to the wind, I deﬁne a single future self-referential comp that describes choosing between the option of immediately ending the piece and the option of the up-transposed sequence of an interesting phrase with the future performance of the same comp. In Scheme, this is a construct that enables binding of local variables.

```cfml
(define (song) (let ((phrase (define (string-step) (after lump) (tra 4 lump)))) (define (string-end) (after phrase song)))))
```

Evaluating (run song) yields, 50% of the time, simply performance of the string-end pattern. However, repeated attempts will often reveal ascending chains of notes that rise in pitch (mind-bendingly satisfying the how many times in a row I can flip heads on a fair coin).

Had cfml been implemented as a batch generation system, performances of The Escalator would sound no different. In this piece, the strange loops in cfml lie dormant, masquerading as a harmless recursive decomposition of a generative music piece.

4.2. The Noodle, a Livecoded Jam Session
The obvious edge livecoding has over generative music is that the rules of composition are malleable; the livecoder can adapt these rules to the evolving tastes of a live audience. In The Noodle, I demonstrate how repeating a very simple music fact can provide the foundation for many moments of enjoyment when the conditions of that repetition are actively steered by the programmer.
with a staccato note on the piano synthesizer:

\[
\text{define (blip)}
\begin{array}{l}
\text{| (literal 1 (4 (0 0 1 0 1))/(4))} \\
\hline
\end{array}
\]

As in The Escalator, I build the overall composition on a four-note ostinato:

\[
\text{define (phrase)}
\begin{array}{l}
\text{| (after blip (tra 4 blip) (bla 9 blip))} \\
\hline
\end{array}
\]

The initial song structure is the staid trope of temporal right-recursion:

\[
\text{define (song) (after phrase song)}
\]

At this point, I evaluate (run song) to kick off live performance of the definitions thus far. This begins a sta- ble arpeggio that can hold the audience’s interest for per- haps a few iterations. Left unattended, it would continue to self-referentially unfold for eternity. To keep the piece alive, I begin an upward progression with this destructive, in-place redetermination:

\[
\text{define (song) (after phrase (tra +1 song))}
\]

The interest created by the introduction of an upward trend turns to tension as the notes creep into uncomfort- ably high register. Again, I step in to rescue the piece from boredom and increase drama with another redetermi- nation, this time starting a much steeper downward trend:

\[
\text{define (song) (after phrase (tra -2 song))}
\]

By now, the trick of steering the noodling melody up and down by tweaking the transposition parameter is be- coming clear. As the note goes to zero, I begin work on fac- toring out a tweakable knob and catch the rhythm just before it enters an uncomfortably low register, reinstating the gentle upward trend with this refactored definition:

\[
\text{define delta +1)}
\]

From here, sharply tweaking delta between -3 and +3 can create a few more moments of interest, but the di- rect control requires too much attention to maintain while thinking through larger scale flourishes. Capturing and automating my manual steering with a random walk is simple enough:

\[
\text{define delta +1)}
\]

With the current algorithms creating some complex- ity on their own, I need rareweak delta except for when the piece wanders too far off course. Between these sparse tweaks, I am free to adjust the spread of notes in phrase, add an alternative phrase2, build larger phrases with interactively-swappable substructure, and begin to automate incremental variation of the dynamics (note ve- locity) using a similar random-walk scheme.

No time during the performance of this piece is there even a particularly interesting algorithm executing. Further, most of the algorithms encountered, when left unattended even for a few seconds, would quickly be pro- dict control requires too much attention to maintain while...
CHUGENS, CHUGBGRAPHS, CHUGINS: 3 TIERS FOR EXTENDING CHUCK

Spencer Salazar  
Ge Wang

Center for Computer Research in Music and Acoustics
Stanford University

{spencer, ge}@ccrma.stanford.edu

ABSTRACT

The ChucK programming language lacks straightforward mechanisms for extension beyond its built-in programming and processing facilities. Chugens address this issue by allowing programmers to live-code new unit generators in ChucK in real-time. Chugbgraphs allow new unit generators to be built in ChucK, by defining specific arrangements of existing unit generators. Chugins allows ChucK to provide a wide array of high-performance unit generators and general functionality to be exposed in ChucK by providing a dynamic binding between ChucK and native C/C++ compiled code. Performance and code analysis shows that the most suitable approach for extending ChucK is situation-dependent.

1. INTRODUCTION

Since its introduction, the ChucK programming language [14] has become a popular tool for computer music composers, educators, and application software developers. However, to date, its catalogue of audio processing unit generators and general programming functionality has been largely limited to those that are built-in when the ChucK binary executable is compiled. Adding new unit generators mandates recompilation of the entirety of ChucK, requiring a level of expertise and motivation reserved for an elite group of power-users. Furthermore, community-based development efforts are hampered by this centralization of functionality, as developers of new unit generators have no easy way to share their work.

The aim of the work described herein is to provide ChucK with multiple levels of extensibility, each essential and appropriate to specific tasks and levels of user expertise. On the one hand, ChucK’s pervasive ethos of on-the-fly development creates the desire to design and implement new audio processors in ChucK itself in real-time, working down to the per-sample level if necessary. Furthermore, implementing these components in ChucK allows their use on any operating system ChucK supports with no additional effort from the developer. For these cases we have developed Chugens and Chugbgraphs.

On the other hand, real-time performance requirements often mandate the use of compiled native machine code for complex audio-rate processing. There also exists a wealth of C/C++ based software libraries for audio synthesis and effects, such as FluidSynth [4] and Faust [9]. These situations can be straightforwardly handled given portable bindings between ChucK and native compiled code, which is precisely the intent of Chugins.

2. RELATED WORK

Extensibility is a primary concern of music software of all varieties. The popular audio programming environments Max/MSP [15], Pure Data [10], SuperCollider [8], and Csound [4] all provide mechanisms for developing C/C++-based compiled sound processing functions. Max also allows control-rate functionality to be encapsulated in-situ in the form of Javascript code snippets. Max’s Gen facility dynamically compiles audio-rate processors, implemented as either data-flows or textual code, to native machine code. Csound allows the execution of Tcl, Lua, and Python code for control-rate and/or audio-rate manipulation and synthesis. The Impromptu environment supports live-coding of audio-rate processors in a Scheme-like language [3], and LuaAV does the same for Lua [13].

Pure allows programmers to dynamically write and execute audio signal processors using Faust [5]. A performance of compiled machine code is written and tied to a virtual machine architecture will cause Chugens to be exposed in ChucK by providing a dynamic binding between ChucK and native C/C++ compiled code. Performance and code analysis shows that the most suitable approach for extending ChucK is situation-dependent.

3. CHUGENS, CHUGBGRAPHS, CHUGINS

3.1. Chugens

The goal of Chugens is to facilitate rapid prototyping of audio synthesis and processing algorithms. Additionally, Chugens provide a basic framework for extending ChucK’s built-in audio processing functionality. Using the Chugen system, a programmer can implement stand-alone, audio algorithms within the ChucK development environment, utilizing the full array of programming facilities provided by ChucK. These processing units can be naturally integrated into standard ChucK programs, even in the same script file, providing seamless control of audio-rate processing, control-rate manipulation, and higher-level compositional organization.

Chugens are created first by subclassing the built-in Chugen class. This subclass is required to implement a function, which accepts a single floating-point argument (the input sample) and returns a single floating-point value (the output sample). For example, this code uses a Chugen to synthesize a sinusoid using the cosine function:

3.2. Chugbgraphs

A Chugen defined so may be integrated into audio graphs like any standard ChucK ugen. Since the tick function is just a standard ChucK class member function, it can be as simple or as elaborate as required. Standard library calls, file I/O, multiprocessing (using spawn), and other general ChucK programming structures can be integrated into the tick function and supporting code. For performance reasons, its important to consider that the tick function will be called for every sample of audio, so tick functions will typically perform better. Moreover, the intrinsic overhead of ChucK’s virtual machine architecture will cause Chugens to underperform compared to a native C/C++ implementation. Lastly, since Chugens are fundamentally a specialization of a Chugen class, it may make more sense to provide structured access to whichever parameters it wishes to expose to the programmer.

3.3. Chugins

Chugins (pronounced “chug-in”) allow near limitless possibilities for expansion of ChucK’s capabilities. A Chugin is a distributable dynamic library, typically written in C or C++ compiled to native machine code, which ChucK can be instructed to load at runtime. When loaded, the Chugin defines one or more classes that are subsequently available to ChucK programs. These classes may define new unit generators or provide general-purpose programming functionality beyond that built in to ChucK. Since these classes are normal ChucK classes implemented with native code, member functions and variables can be used to provide an interface to control parameters. Chugins are best suited for audio algorithms that are reasonably well understood and stand to gain from the writing of compiled machine code. The "write-compile-run" development cycle and C/C++-based programming mandated by Chugins makes implementing audio processors require comparatively more effort than the Chugbgraph or Chugen approaches. However for ugens a programmer intends to use over an extended period of time, the effort to implement a Chugin will quickly pay off in the form of lower CPU usage.

An additional advantage of Chugins is that they may provide functionality far outside the intrinsic capabilities of ChucK. Complex synthesis C/C++-based synthesis packages can be imported wholesale into ChucK, opening up an abundance of sonic possibilities. For example, Chugins have been implemented to bring audio processing programs from the Faust programming language into ChucK. Similarly, the SoundFont renderer FluidSynth has been implemented as a Chugin. This functionality is not limited to audio processing; a serial port import/output Chugin is under development, as are other general-purpose programming libraries.

Development of a Chugin is somewhat more complex than Chugbgraphs or Chugens, and does not lend itself to explicit presentation of code herein. Using a set of convenience macros, a Chugin developer first defines a function, which ChucK calls upon first loading the Chugin. Chugin provides the function with routines with which to define unit generators and classes, specify what member variables and functions
are associated with these, and indicate a `tick` function in the case of unit generators. These functions are then defined in C/C++, using predefined macros for interactions with the upper-level Chuck runtime, such as retrieving function arguments, getting and setting member variables, and handling input/output samples. This code is then compiled into a dynamic library using the standard facilities for doing so on the target computing platform (gcc for Mac OS X and Linux systems, Visual C++ for Windows systems). Additional C/C++ code or libraries, such as STK, may be compiled into the dynamic library using the mechanisms standard for those operations on the target platform. Furthermore, Faust DSP code can be automatically converted to ChucK using Faust's just architecture file and an accompanying compilation script.

### 4. PERFORMANCE CASE STUDIES

Execution speed of audio software is typically a vital metric for computer musicians, as real-time audio synthesis requires timely production of tens of thousands of samples per second. To evaluate the performance of our mechanisms for extending Chuck, we designed and implemented several reference unit generators using our mechanisms for extending ChucK, we designed and implemented several reference unit generators using each method. For each method, we measured the time required to offline-render 5 minutes of audio using the resulting unit generator.

In many situations not only is CPU time at a premium, but a programmer's time to implement a specified application is also limited. Therefore, often “less complex” programs are more desirable than “more complex” programs because they can be developed faster and be better understood by other programmers. While quantifying code complexity is a nuanced and challenging task of finding and installing ChucK extensions. Similar to Debian APT [1] or RubyGems, to simplify the task of finding and installing ChucK extensions. Our current system of importing ChucKs does not scale well, as ChucKs cannot be loaded on demand. Rather, ChucK will load every ChucK that is installed on a system, which may take a noticeable amount of time if there are many ChucKs present. In the future we wish to more intelligently load ChucKs only when their component units and classes are invoked by active ChucK code.

ChucKs generate audio by executing ChucK virtual machine code at audio-rate; dynamically compiled ChucKs instead to native machine code may introduce significant performance gains. Environments such as LuaAV, Gen, and Pure do just this, compiling high-level programs to native code for audio-rate execution. Additional improvements may be required in the form of process and memory safety in ChucKs. Currently, ChucKs execute in the same process space as Chuck itself, which means that buggy ChucKs can crash Chuck outright. Technological approaches can alleviate these problems somewhat, but organizational solutions may also be desirable, such as having registries of vetted and thoroughly tested ChucKs.

<table>
<thead>
<tr>
<th>Chugen</th>
<th>10.690</th>
<th>21</th>
<th>223</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chugraph</td>
<td>6.544</td>
<td>41</td>
<td>258</td>
</tr>
<tr>
<td>Chucin</td>
<td>1.965</td>
<td>84</td>
<td>165</td>
</tr>
</tbody>
</table>

Table 2. Performance and complexity measurements for Bitcrusher.

### 5. FUTURE WORK

Further developments we are pursuing include mechanisms for creating a general purpose programming libraries in ChucK. Additionally we are investigating a unified ChucK repository and distribution system, similar to Debian APT [1] or RubyGems, to simplify the task of finding and installing ChucK extensions.

### 6. ACKNOWLEDGEMENTS

Special thanks to Kassen Oud, Casper Schipper, Jorge Herrera, and Hongchen Choi for their invaluable beta testing, bug finding, and feature requesting. Additional support for this research was made possible by a National Science Foundation Creative IT Grant, No. IIS-0855758.

### 7. REFERENCES

GIBBER: LIVE CODING AUDIO IN THE BROWSER

Charles Roberts, JoAnn Kuchar-Morin
University of California at Santa Barbara
Media Arts & Technology Program
charlie@charlie-roberts.com, jkm@create.ucsb.edu

ABSTRACT

The performance practice of live coding has grown dra-

matically over the course of the last decade and the in-
creasing number of practitioners has been accompanied by
a growing number of languages and environments. Into
this context we introduce Giber: a pure JavaScript library for
performing high-level audio synthesis and sequencing in
web browsers. In addition to its JavaScript implementation
Gibber also includes a performance environment that can
be run in browsers implementing a realtime audio API.
Google’s Chrome browser currently provides the
best audio experience but other browsers are also sup-
ported. Gibber is built on top of JavaScript libraries for
realtime audio APIs, the most important of these being audi-

oLib.js by Jussi Kalliokoski[1]. We layered an extremely
lightweight syntax on top of audioLib.js and extended it to include
a variety of sequencing options, FM synthesis, gran-

ular synthesis, Karplus-Strong physical modeling, flang-
ing, waveshaping, buffer shuffling and a number of other
oscillators and effects. Some of these have already been
contributed back to audioLib.js, we hope to contribute
more in the future.

1. INTRODUCTION

The performance practice of live coding has grown dra-

matically over the course of the last decade and the in-
creasing number of practitioners has been accompanied by
a growing number of languages and environments. Into
this context we introduce Giber: a pure JavaScript library for
performing high-level audio synthesis and sequencing in
web browsers. In addition to its JavaScript implementation
Gibber also includes a performance environment that can
be run in browsers implementing a realtime audio API.
Google’s Chrome browser currently provides the
best audio experience but other browsers are also sup-
ported. Gibber is built on top of many open-source Java-

Script libraries, the most important of these being audi-

oLib.js by Jussi Kalliokoski[1]. We layered an extremely
lightweight syntax on top of audioLib.js and extended it to include
a variety of sequencing options, FM synthesis, gran-

ular synthesis, Karplus-Strong physical modeling, flang-
ing, waveshaping, buffer shuffling and a number of other
oscillators and effects. Some of these have already been
contributed back to audioLib.js, we hope to contribute
more in the future.

1.1. Motivation

The creation of Gibber was motivated by the desire to use
JavaScript as a language for browser-based live coding
environments. When work on Gibber began, no musical
live coding environments using JavaScript existed. Al-
though there are many visual environments that let you
change JavaScript code and quickly see the updated re-
sults, few are geared towards live performance[1]. JavaScript,
although sometimes maligned[2], is an excellent language
for live coding. Its prototypical nature makes it easy to
extend and combine objects, it is dynamic, features object
introspection and meta-programming and has first-class
functions with closures. This flexibility enabled us to cre-
ate a syntax in which the creation and playback of a sine
wave as simple as:

```
Slow(440, .5);
```

In the above example, 440 is the frequency and .5 is the
amplitude of the wave form. Throughout the creation of
Gibber we strove for this level of simple, declarative
syntax. The next example creates a triangle oscillator and
adds distortion and reverb to it:

```
t = Tri(220, .5);
//Fx.add( Dist(1), Reverb() );
```

By coupling this syntax with an accessible web-based
environment, we hope to provide a vehicle for students
to easily experiment with synthesis. Towards this end, we
included a number of tutorials on additive, subtractive
and FM synthesis with Gibber.

Gibber also supports networked live coding perfor-
manences. To join a networked performance performers sim-
ply launch Gibber in a browser, choose a username and enter
the IP address of a remote computer running the Gib-
ner environment. Any code sent to this remote instance
will be displayed for the audience and other performers to see
(assuming video from the remote computer is pro-
jected as per the typical live coding performance) and ex-
ecuted in the remote context.

1.2. The Browser As A Realtime Synthesis Platform

Over the past twenty years digital audio practitioners have
witnessed the gradual evolution of the browser as a re-
altime music platform. From its humble beginnings as a
player of (often aesthetically questionable) general MIDI
files, to the use of Java and Flash plugins to synthesize
audio, to the current trend of using native JavaScript for
sample-level synthesis, the browser has, almost since its
inception, been an environment for realtime music cre-
aton and consumption.

Support for JavaScript based audio synthesis varies wildly across current web browsers. Google’s Chrome and
Firefox provide the best support; Firefox is a close
second but appears to operate less efficiently and with
inferior timing. Gibber’s default script (feature-features, delays, FM synthesis, physical modeling, drum samples and
various other audio effects) uses an average of 39%
of one core on a 2.8 GHz Intel Core i7 processor under
Chrome while Firefox uses an average of 46%. JavaScript
does receive access to audio output buffers in Internet Ex-
plorer and realtime audio is only available in a limited form
in beta versions of Safari. No mobile web browsers currently
support JavaScript audio synthesis. Apple’s security restric-
tions against JIT compilation in mobile web browsers have stopped the WebGL standard from being implemented
in iOS for many years and similarly block realtime audio
implementations in the browser; there is little reason to hope that these restrictions will change in the near future.

These concerns aside, at some point in the future a
standard for web audio will be agreed upon and provide
browser developers with greater motivation to properly
implement realtime audio capabilities. Various standards
have already been suggested; they range from a relatively
simple API providing access to input and output stream
[2] to a much more complex API suggested by
Google[8] that provides C++ implementations of convolu-
tion and other expensive DSP algorithms in order to obtain
performance improvements over JIT-compiled JavaScript.

Gibber has been designed to provide a simple and
straightforward syntax to enable our coding environment, per-
form networking for distributed performances and synthe-
size realtime audio. Although there are efficiency limita-
tions for JavaScript-based audio synthesis when compared
to C/C++, these limitations are balanced by the benefits
of an accessible platform with a unified development lan-
guage.

1.3. Related Work

At the time of this writing, toplap.org (an umbrella web-
site for performers interested in live coding) lists thirty-
four active live coding environments; this is by no means
a comprehensive list. In order to position Gibber within
this extensive and growing community, we compare the
following attributes and draw upon a number of live cod-
ing implementations that were influential in the creation of
Gibber:

- **Language - Is a new programming language imple-
  mented or is an existing language used?**
- **Synthesis Source - Are synthesis algorithms writ-
  ten in the same language used for performance?**
- **High-Level / Low-Level - Does the environment man-
  age an audio graph? Do you have access to low-
  level sample output?**

In terms of language, Gibber is implemented in pure
JavaScript with no syntactic additions or modifications. This
is similar to live coding environments using languages
descended from Lisp: Impromp[l0], which is written in
Scheme, and Overtone[l3], which uses Clojure. JavaScript
itself borrows a great deal from these functional languages,
including functions as first-class objects and closures. Other
languages such as superCollider[l2] and ChucK[l2] implement
new languages for live coding. Another JavaScript live
coding environment was recently announced[l4] focusing
on graphics and low-level audio synthesis. This environ-
ment adds special keywords to JavaScript; code written
with these syntactical additions cannot be run using a
standard JavaScript interpreter without preprocessing.

Many live coding environments include synthesis algo-
rithms written in the same language that is used for per-
formance while others are used to control external audio
sources. Impromp[l, for example, controls Audio Unit
plugins in OSX; this provides access to an extremely rich
palette of commercial and open-source synthesizers and
effects. ixi and Overtone both control the audio server
ixi found in SuperCollider[7]. Low-level unit generators in
are programmed in C++ in ChucK, however, these can be
combined at a higher level using the ChucK language
coding synthesis where graphs of operator expres-
sions are defined using the interpreted Lua language, and
then JIT compiled to machine code to obtain the best per-
formance possible. In Gibber, synthesis is performed in-
ternally using JavaScript. The audio quality of Gibber
currently suffers when compared to platforms that rely
on richly developed synthesis tools like the SuperCollider
server or vst plugins.

Gibber’s primary focus is on pattern-based music gen-
eration; it features high-level methods that allow sample-
accurate sequencing and synthesis capabilities. In this
way it is very similar to and directly inspired by
Clojure, the aforementioned language layered on top of Su-
perCollider. Although performance practice using Gibber
primarily consists of executing high-level commands to
control sequence and synthesis parameters, Gibber does
provide the ability to override the default audio callback so
that users can directly fill output buffers with samples using
Gibber unit generators.

2. IMPLEMENTATION

Gibber is a JavaScript library for the declarative creation
and sequencing of audio synthesis graphs. It also con-
tributes a performance environment that can be run in
web browsers and affords simple networked music perfor-
manences. We will discuss the JavaScript syntax and performance en-
vironment separately.

One exception to this is the excellent Lowcode.js found at
http://www.5kits.net/pix/lowered.js

2http://sofljs.org
2.1. Performance Environment

Gibber runs in web browsers that implement an API for real-time audio synthesis. Users can either download the performance environment from GitHub to run on their own computers or they can simply visit the Gibber website. The performance environment features a code editor built using the open-source project CodeMirror. The editor provided by the CodeMirror library affords syntax highlighting, line numbering, customized GUI themes and keystroke shortcuts amongst many other features.

One important addition to the code editor in Gibber is sample-accurate delay of code execution. Although selected code can be executed immediately via a keystroke combination, other keystroke combinations allow code execution to be delayed until the first sample of the next musical measure or beat. This ensures that the various sequences that comprise a Gibber performance will be executed precisely on the downbeat of the next measure if desired, regardless of network jitter and latency. At worst, under extremely high latency, the code will be executed a measure later than intended but will still generate results that are in phase with the rest of the performance.

In addition to displaying code, the remote computer also displays a real-time chatroom that allows performers to communicate during performances; this practice of displaying dialogue amongst performers can also be found in the current network setup used by the Hub and in the recently introduced live coding environment LOLC[5].

In addition to the code editor, the performance environment also features a menubar containing quickstart instructions, GUIs for loading/saving files and joining distributed performances, a list of key commands and general information about the Gibber project. A sample Gibber session is presented in Figure 1.

2.2. Features

2.2.1. Syntax

As mentioned in the introduction, the audio synthesis in Gibber is layered on top of the audioLib.js library. This library comes with a number of unit generators and also handles browser discrepancies in providing access to output sample buffers. Although excellent for general use, the audioLib.js syntax is unnecessarily verbose for live coding. For example, compare the process of defining a 440 Hz square wave in audioLib.js and Gibber:

```javascript
// audioLib.js
var s = new audioLib.Oscillator(440, 440);
s.waveShape = "square";
// Gibber
s = Square(440);
```

Note that the above audioLib.js code does not generate any audio output; a programmer would need to manually append the oscillator output to a buffer feeding the DAC, presumably inside a sample loop. In Gibber, the square wave oscillator is automatically added to a signal processing graph and begins outputting audio immediately.

To further improve terseness and decrease the potential for typos during performance, we abandon some JavaScript best practices and take advantage of questionable language features. The Gibber environment makes extensive use of the global namespace; in doing so we avoid the need to prefix constructors with a namespace identifier and also mitigate the need to use the var keyword when declaring variables. We also promote the use of single letter variable names; audio and sequencing objects held in single letter variables are automatically replaced in the audio / control graphs when reassembled. Consider the following lines of code:

```javascript
t = Tri();
s = new audioLib.Oscillator(440, 440);
s.waveShape = "square";
// Gibber
s = Square(440);
```

These two lines of code create a triangle oscillator and sequence it to alternate between pitches of 440 and 880 Hz every quarter note (sequencing will be discussed in greater depth shortly). By using single letter variable names, audio and sequencing objects held in single letter variables are automatically replaced in the audio / control graphs when reassembled. Consider the following lines of code:

```javascript
v = Tri();
s = new audioLib.Oscillator(440, 440);
s.waveShape = "sine";
// Gibber
v = Sine(440);
```

2.2.2. Modulation and Effects

Modulation and effects are also managed automatically by the Gibber audio graph. To apply modulation, a programmer specifies the parameter to modulate, a source for the modulation and how the modulation output should be applied. Below is a simple example for vibrato.

```javascript
s = Sine(440);
// Vibrato : +/- 8 Hz at a rate of 4 Hz
s.mod("frequency", Sine(4, 8), "+*-");```

The "+" shows that we add the output of the modulating sine wave to the frequency component; other operators include assignment, multiplication, division etc. As another example, here is an unevolved FM bell:

```javascript
carFreq = 250;
c = Sine(carFreq, .15);
cmRatio = 1.4;
index = 190;
```

```javascript
c.mod("freq",
    Sine(carFreq * cmRatio, index, "+*-"));
```

Each oscillator has an effects chain that we can add to and remove from:

```javascript
v = Sine(440);
s.fx.add( Flanger(), Delay(), Reverb() );
```

```javascript
// remove flanger
s.fx.remove("Flanger");
```

```javascript
// remove first effect in chain
s.fx.remove(0);
```

```javascript
// remove all affects
s.fx.remove();
```

Any effect parameter and any modulation parameter can be modulated; modulations are applied recursively within the audio signal graph. There is also a global variable that allows us to apply effects to the summed audio output of all oscillators.

2.2.3. Rhythm and Timing

Choosing a system for notating time and rhythm in Gibber has been an interesting challenge. Early on we decided that there would be an initial emphasis on using 4/4 time. With this emphasis it makes sense that there would be an easy way to indicate both divisions and multiples of a musical measure; however, we also wanted to retain the freedom to specify precise durations in samples. The initial solution to this was to create a set of variables holding the number of samples for subdivisions of a measure ranging from a whole note to a sixty-fourth note. These variables were preceded by an underscore character: thus, _1 was a quarter note, _4 was a sixteenth note, etc. Durations lasting longer than a measure could also be easily notated, for example, _1 * 2.25 represents the number of samples in two measures and a quarter note.

Although this system worked well for a number of both live performances and screen casts, it always felt contrived and was visually unappealing. When considering possible alternatives the potential visual representations of durations hinted at a better solution. Instead of using variables to represent subdivisions of a measure, we now assume that a musical measure is the standard unit of duration and simply pass a value of _1 to functions accepting a duration as a parameter in order to indicate a measure in length. In this system _1/8 represents an eighth note, _1/16 represents a sixteenth note and _1 represents sixteens measures in length. This is visually much more elegant than our previous solution and much treier for declaring lengths greater than one measure in length, for example, _1 + _4.5 simply becomes 4.5.

There are of course problems that arise when using integer values to represent both durations measured in samples and durations measured as multiples of a musical measure. In Gibber...
2.2.4. Sequencing

Although there are a number of objects used to sequence parameters in Gibber, each is derived from the fundamental Seq (sequencer) object. The Seq object can be used to sequence any type of data. As examples, one Seq object might pass chord identifiers as strings to an Arpeggiator while another might call a series of different functions, perhaps alternating every measure between telling another Seq to randomize its values and then re-setting them to their original contents. Virtually any aspect of Gibber can be controlled by a Seq object. In the code sample below, the first Seq controls the pitch of an oscillator while the second randomizes the order of values in the first and resets them to their original positions every four measures.

```javascript
s = Sine(440);

// pass two methods to alternate calling
// slave the previously created sine ugen
// pass an array, a duration for each step
s = Seq([440,660,880], 1/4).slay(s);

// pass two methods to alternate calling
// every four measures
s = Seq([s.shuffle, s.reset], 4);
```

The Seq object makes a distinction between the sequence of objects or functions it holds and the durations that define when these functions are executed. An array of durations can be passed to the Seq constructor instead of a single duration enabling arbitrary rhythms to be sequenced. This separation also allows for greater control of algorithmic processes as sequence and duration values can be accessed programmatically using different techniques.

By default, progression through the sequence and durations arrays occurs linearly. However, users can assign an arbitrary pick function to either the sequence or durations array that can instead determine which array position to use when the sequencer advances. Gibber includes a `surpriseMe()` function that randomly picks an item from an array, and a `weight()` function that allows users to weight the likelihood of each item being chosen. In the example below, we see a Seq object where frequencies are randomly selected while durations are heavily weighted towards eighth notes as opposed to quarter and half notes.

```javascript
// pass arrays of frequencies and durations
s = Seq([440,660,880], [1/8,1/4,1/2]);

// tell Seq to randomly pick frequencies
s.sequence.pick = surpriseMe();

// use a weighted random function to pick
// durations
s.durations.pick = weight([.8, .1, .1]);
```

Gibber defines other objects extending Seq that provide additional functionality:

- **Arp** - allows you to easily arpeggiate chords identified by nomenclature such as “G4m7” and “CMaj7b9”.
- **Drums** - in addition to providing drum samples this object can be quickly sequenced using a string such as “toto” to represent kick and snare hits.
- **ScaleSeq** - allows you to sequence a series of notes in a particular mode that are offsets of a root note. For example in a Aeolian 0 = A, 1 = B, 2 = C etc.

All Seq objects register to receive messages from the Gibber master clock so that they update their step durations whenever the master tempo changes.

3. CONCLUSIONS AND FUTURE WORK

Gibber has been used in a number of performances since its introduction. The first formal concert performance was by the CREATE Ensemble at UC Santa Barbara, in which six performers submitted code to a remote computer for execution. It is worth noting that one of the ensemble members had no programming experience but was still able to participate by copy and pasting code and modifying variable values. The first author has also performed solo with Gibber both in formal and informal settings.

We are excited about the educational potential of a syntactically clear live coding environment presented on the web. We imagine classroom scenarios where a lecturer instructs students to explore a synthesis topic, after providing some time for experimentation the lecturer could then ask for volunteers to submit code to a remote instance of Gibber where it could be projected, executed and critiqued by the class. To further this idea we have included tutorials on various synthesis techniques in Gibber and plan to incorporate more.

In terms of future work the audio synthesis capabilities of Gibber need to be expanded as the palette of available sounds currently run at audio rate; control-rate modulation would greatly improve efficiency. We look forward to adding generative visualization capabilities to Gibber and exploring strategies for controlling both audio and visuals concurrently. Finally, we hope to further expand Gibber’s networked performance capabilities with the addition of audio graph visualizations that inform users about the activities of their fellow performers.

4. ACKNOWLEDGMENTS

Special thanks to all the open-source projects that contribute to Gibber. This research was supported by NSF grant IIS-0855279 / IIS-1047678 and by a graduate fellowship from the Robert W. Deutsch Foundation.

5. REFERENCES

LIVE NOTATION: ACOUSTIC RESONANCE?

Alex McLean
ICRiM, University of Leeds
alex.mclean@icrim.org.uk

Hester Reeve
C3RI, Sheffield Hallam University
h.reeve@shu.ac.uk

ABSTRACT

The present paper acts as a viewpoint shared by two authors: a live coder who uses their programming language in live music, and a live artist who uses their body in live fine art. We provide background to these practices, before entering a dialogue exploring their confluence. The subject of the dialogue is a hypothetical collaborative performance, from which a shared platform of live notation could be explored. The relation between code and body is confronted from both perspectives, looking for a role for live notation as an intrinsic part of live work, both for body and code. In this we consider notation as not being something that prefigures, defines or is created by a performance, but as activity that resonates within a performance.

I. INTRODUCTION

The present paper is a confluence of two performance arts, finding a viewpoint for a live artist and live coder to reflect upon their practice, and look for means of collaboration. There is some existing context for this work, for example Nick Collins has applied the principles of live coding to dance, working with choreographer Tessa Priya, and separately with Matthew Yee-King as Wrong-Headed [1]. Similarly, some fine artists have worked with public performance and programming in the construction of multimedia and performed work, such as Simon Biggs in Body Text (2010), working with dancer Sue Hawksley and sound artist Fraser famous for his work Balloons and ‘performing’, a rejection of tradition and entertainment as a strategy to grab a handle on a complex, difficult to model to anatomy, but to witness the artist’s body labouring in a meaningful or non-everyday fashion in front of the visitors to the space, fronting the visitors to the space, sometimes for the duration of the performance, sometimes for the duration of the performance, and sometimes for the duration of the performance...
comes from (now in reference specifically to practices such as Reeve’s) is to recognise that it is decidedly anti-theatrical, site-specific and usually performed once only. The action risks unfolding live over time un-rehearsed, experimentation is the aesthetic as opposed to formal beauty or ‘excellence’. In fact, a live work’s meaning emerges through the very fact that the performer is not ‘pretending’ or even necessarily ‘skilled’ in terms of using the body; live artists are not ‘dancing’ the idea, for example, but that would not stop them dancing, if the idea of an horizon of an idea demanded it. Intention and idea-hood is everything. So, the live artist is an ordinary human being prepared to do something in front of others in order to open up the potential of liveliness per se.

Perhaps unexpectedly, a synchronicity/confuence between the practices of live coding and live art is now im-mediately apparent, not so much in terms of the aesthetics of the performances produced but in terms of the committed labour of the creative practitioner intensively involved in producing before an audience. Similar too is a quasi-sto-rriness in refusing to employ standard conventions – the live coder rejecting standard forms of classical music and the live artist rejecting literal communication of ideas, instead seeking to embody them. This is not to be agent provocateur or punk rock – we would suggest instead that this ‘anti-representation’, arises from an investment in the form of the work for its own sake, coupled with a belief that to do so is not indulgent but often a way of intensifying the audience experience.

Just as live coders might benefit from considering their body-at-work in their performances and their code as ex-tended meaning full body (because the code is an integral aspect of the live work and has a presence that exceeds ‘bits of information’), live artists might benefit from con-sidering the incorporation of abstracted marks and sym-bols within their performances as a form of extended meaning-full body. Indeed some live artists are already moving toward such a suggestion – giving agency to mark making or written language within their work. In Brigid McLean’s work Vections-Writing (2007-8), for example, the action of living with a piano for a month and learning to play it as much performed through large diagrams – part musical score, part genesis of erasures – as it is through fingers on the keys. More subtly, Yuan-Fong Ling transposes the tra-di-tional life class into a participatory performance where often he as ‘master’ is the naked body to be drawn by stu-dents whose labour processed again we see the body at work in conjunction with markings being produced and, in line with live art strategies, ordered by an idea-hood that can only be communicated via critical liveliness. A few live art events are even considering private performance in the studio, in making a drawing or painting, to be something relevant to focus in upon (as in the case of Andre Stitt’s most recent explorations). Whilst live work is often an array of issues related to the meaningful documentation of their works, recently extended to the ‘material traces’ left over after an action ³, the consideration of ‘notation’ within the execution of a live art work or the ‘liveliness’ of notation to have affect is entirely novel and may open up new possibilities in live art practice and theory.

2. DIALOGUE

Hester Reeve (HR): Despite apparent differences between live coding and live art, we’ve had some very engendered philosophical conversations and always seem to tune in to similar areas of value. Does this mean if we were to start work on a collaborative performance it would be straightforward?

Alex McLean (AM): Well I think the problems that might emerge could be interesting, and so in a way it would be a shame if it was straightforward. I think it would also be unlikely, as I think issues of the body are some-what alien to computer programmers.

HR: And that’s obviously not the case with live artists. And yet, I always start my process for a work from ab-stract ideas – philosophical ideas as opposed to camal ones. But of course, I am very interested in what my body then has to do in order to embody or be true to those ideas. This conceptual idea basis to live art might mark another difference between our two practices. So, exciting but not straightforward?

AM: Yes it is interesting that in live coding there are two different agents; the programmer who codes and the music generator that the code generates. The code then generates it, and no-one really seems sure about what the pur-pose of showing the code is. Some suggest it is to en-hance the experience of the music, I think it might just distract from the music. But the code does something which might be the alternative of showing nothing is worse. So the place that code has in a live coding performance might be understood better on considering why live artists show the body.

HR: For me as a live artist, the body-being of a live art work is obviously central, but it’s not the core essence of that centre (it certainly can not be removed). I am not sure straight off what I do think is more elemental to the work than my body making it happen. Interestingly, I get that same sense of something ‘elemental’ (for want of a better expression) when I look at the code in a live coding performance. I also respond to the coder’s body invested in labour at the laptop, yes, but the code is not mere result or documentation, because it has live agency but only as you witness it. The fact that there’s that flicker of agency without-a body is somehow deeply stirring.

AM: Yes I agree there is a question of what a body is. I have been rather enthusiastic about this for a couple of decades, to the point that code has been an important window of experience for me. Is my code part of my body? I am reminded of a discussion I had with my student, when we realised that we sometimes show parts of ourselves, for our bodies and minds. For my sister, her body was ra-tional and her mind irrational, so for example her mind would want to jump on the next Eurostar train to Paris, but her responsible body would stop her. For me my body wants to go to Paris and my rational mind stops it. This kind of basic mismatch in bodily experience might also be a problem in our (as yet hypothetical) collaborative perfor-mance. Do you think it is possible and you think that the code could realise it? If such an incompatibility unfolded during a performance, that could be exciting, or it could result in ‘failure’.

HR: For starters, I am not sure that our collaboration means that as a live artist my job is to confront you with your body or make you consider its ‘performance’. For me that would be somehow to miss the point. And, as I have said, the notion of ‘live’ is not ‘dramatic’ or ‘presented’ in any way, that concentration in the task which is witnessed through the coder’s body carries its own potentiality which I am interested in about it. That is mean-ingful somehow from a live art perspective, so I would see one aspect of my collaborative effort to affirm that in your practice. I use my body as a medium in my work because I feel it is (in our culture) extremely honest and somehow it demands people consider potentiality, the po-tential depth of human agency, which freaks most people out. For example, when Oleg Kulig runs around naked in a gallery or city centre as a savage dog, often baring people, audiences may claim to be shocked by his coarse actions which over turn social norms but to my view of things, the real shock that such live art actions force people to face is that the music is not something ‘direct’ that the body has to do, it is music, in this sense it is true: we decided IT, we uphold IT and look, I am undecided IT. It gets undecided relatively easily ergo you have the capacity to undecide IT. And of course, un-deciding is as disruptive to programming as something radically new as it is an abandonment of something.4 But, even so, more of importance is how is it that a live art action I do enables an experience of significance for the audience, but I don’t in any way mean ‘Hester Reeve’ as significant. It’s almost ritualistic, without meaning that my actual performance mimics so called ethnic performance rituals. It’s ritualis-tic on an ‘elemental’ level. I still can’t get to what I mean by that other than (to return to your opening comment) it seems to link to questioning after what is the body? Not simply the flesh that makes me me and you you?

AM: I suppose the body is something through which we may speak, creating channels of communication for example by articulating the mouth or gesturing with hands. Something of substance, perhaps, but if we ig-nore that then the code could be seen as an extension of the body, as could other forms of writing[8]. And I think ex-hposing the code is a gesture of honesty too, not caring how it looks to others, but just showing how you were. The code is not the body but I do not get the sense that it is moving, however, something is moving inside. The code is being constructed, in the process chopped up and reformed in different guises, and the computer pro-cessing the code is not just following the instructions in the code. At times the listening audience in the room might-Oleg Kulik often carries out his dog action, many cite as legendary ‘Dog’ at Interpal, Felpulifeben, 1996.

4Oleg Kulik often carries out his dog action, many cite as legendary ‘Dog’ at Interpal, Felpulifeben, 1996.
ing the time structures I want to construct through my code, shift away from the wholly grid-based acid techno I have become preoccupied with for the past couple of years and back towards looser time, not in terms of inaccuracy but in terms of events which organise themselves into heterarchies, rather than imposing a fixed grid upon them. Repetitions, and multiples of the number four are psychologically extremely important to me, but to the point where I need to fight against them through polyrhythmic patterns and musical arcs that end in chaos. This is an opportunity to create an important shift in my musical activity, influenced by your actions but as part of my own musical development.

HR: Is it possible to relate more about the coding aspect of your performance work to me - a non-programming person. Your type of coding is a curious mixture of abstract language and concrete making born in the same moment. Is it a logical process then?

AM: Well code takes much the same form as written text such as this, discrete alphabetic marks on a page. When we read a novel the text might evoke sensory experience, and when a computer turns code into sound it is an analogous process. So when that code is interpreted live, the senses can be manipulated through the text. The history of computer programming is in both mathematics and textiles, and I think the latter still offers the best metaphors for programming, following a knitting pattern, or giving a machine instructions which are woven into results. But of course with programming, I don’t feel the wood moving between your fingers, and code involves extra steps of abstraction; not just patterns, but patterns of patterns. Live coding reconnects these levels of abstraction with tangible experience, so half of you is in discrete language, and the other half is lost in a musical moment. This is a lot like speech, discrete words interwoven with musical prosody. So for you, how does this kind of notion of abstraction relate to body work, or notations of body work, in live art?

HR: Except with live coding, your brain is not in command of the end result in the way that it is when the weaver weaves or someone speaks to another. Or do you know exactly what sounds will emerge as you code, in the way that a pianist can anticipate what will happen with her fingers in a certain arrangement on the key board? It’s hard to know how to answer your question. Notation is just not a term that has ever been used in association with live art, or not yet. The term has concrete relevance to the planning stages of a performance, a behind the scenes plotting of the end result in the way that it is when the weaver weaves or someone speaks to another. But for only so long as the notation exists.

AM: I might not know exactly what will happen when I change the code, but I anticipate what might happen. This is the same as a painter, who makes a mark on canvas, experiences the results, and decides whether it is good before making the next mark [6, p. 33]. So my brain is locked in a feedback loop with the code, output and the body – the brain has an idea, writes some code, the code is turned into output, and experienced through the body.

HR: So, are you insinuating that the value of the body in our respective performances is to maintain the ‘world,’ ‘matter,’ ‘other humans’ somehow as constituent in the form of the work but without the need to represent them? No resonance without any of those things?

AM: Yes I think I agree with that, at least as a starting point. Shall we leave it there for now and write a conclusion?

3. CONCLUSION: LIVE NOTATION: ACOUSTIC RESONANCE AND NON-COCHLEAR SOUND

‘Non-Cochlear Sound addresses sound as a conceptual, contextual construct. Non-Cochlear Sound might function in a sound-like fashion without specifically referencing or making sound, it might use sound as a vehicle for transporting ideas or materials from point A to point B, it might even make sound but only as an excuse for initiating other activities. Sound always makes meaning by interacting with other things in proximity: geographic proximity, ideological proximity, philosophical proximity. Non-Cochlear Sound is nothing more – and nothing less – than the acknowledgement of this reality.’


As our dialogue between live coding and live art demonstrates, ‘Live Notation’ is not about an exchange of techniques or skills (although we are open to the need arising) but instead, and perhaps a little surprisingly, more about excavating ontological concerns in order to understand and support deeper, concealed ‘shared space’ that lies at the heart of both practices (at least in terms of the particular exercise of them). This shared space seems linked to resonance rather than any type of content, a resonance that can only come to bear through the presence of both. However, it is not clear that there is an association between ancient and contemporary. In these ancient caves sound and marking were used as part of ritualised communication. Often red dots of colour are daubed around the hand prints, and scholars suggest that these mark points of aural resonance within the caves that the tribes people would have exploited in their rituals. Some scholars have gone as far to name such events as the first ever rock concerts [11]. It is hard not to think that aspects of these performance-rituals were about inhabiting – if temporarily – the world as extended-body.

4. ACKNOWLEDGEMENTS

This work is supported by Arts and Humanities Research Council award AH/J013056/1 on the Digital Transformations programme.

5. REFERENCES


PLAYING WITH TIME - MANIPULATION OF TIME AND RATE IN A MULTIPLE-RATE SIGNAL PROCESSING PIPELINE

Georg Essl
Computer Science & Engineering and Music
University of Michigan
Ann Arbor, Michigan, USA
gessl@eecs.umich.edu

ABSTRACT
Time is a central notion in synthesis engines, and manipulating time is an important part of structuring an instrument, a sound or a performance. We discuss how time can be treated in a flexible-rate and multi-rate dataflow engine that does not operate on a preferred rate. We describe how rates can be locally controlled, now interweaving rates can be managed. A multi-rate pipeline has benefits both for computational load as well as ease of building dataflow interactivity in live performance.

1. INTRODUCTION
In this paper we address the manipulation of time and rate within a multi-rate signal processing pipeline. Time is of course a central notion to audio processing, as it is intrinsically a time-based medium. Many perceptual qualities change critically with time differences such as the transition of temporal to frequency hearing [16] and the transition of temporal to frequency hearing [16] and the characteristics of the pipeline design that help deduce local timing and rates. And we will discuss the kinds of time and rate based processing units we suggest for such an audio processing pipeline.

2. RELATED WORK
The work presented here is currently in use in the mobile programming environment UrMus [7]. Audio processing engines have a long-standing history going back to its origin with Music by Max Matthews. Ultimately multiple paradigms have emerged addressing how to allow users to generate and process music. The most dominant paradigms are time-based systems, such as Csound [3], Max/MSP [20] or Pure Data (Pd) [19] on the one hand, and graphical patching systems, such as SuperCollider [15] or Chuck [21] on the other hand. The importance of time has been recognized for a long time and many systems described above have a wealth of mechanisms to deal with timing, rate and changes in dataflow.

However a single audio-rate is the dominant paradigm even in systems that do offer rich primitives to control time itself, as is the case in Chuck or SuperCollider. Even though Pure Data is in principle a single-rate dataflow system, aspects of multi-rate processing can be implemented in Pd via the sub-patching mechanism [14].

The question of incorporating timeliness into multi-rate dataflow architectures, as well as a range of time-based flow patterns was addressed by Atzeni and co-workers [2, 1] and they also addressed suitable visual representations for time-based patterns [10]. The use of serialization, vectorization, decimation and selection operators to offer pathways to multi-rate processing was explored in the contexts of the designs of Faust [12, 13]. Our concern here is somewhat different to these prior proposals as we discuss in the following section.

Perhaps the closest to the current work is the recent work by Norilo [18] on the Functional Reactive Paradigm invented by Nordland in the context of functional programming [17]. The key idea here is that from the call of a function, final evaluation, the local frame rate requirements can be deduced when considering sources (he used the word spring) are found in the functional evaluation path and their rate propagated to points of intersection which is the case for all the way to the remaining hardware called sinks. In our work we arrive at similar conclusions, however outside the functional paradigm and we arrive at a structure that is directional. In addition our implementation of directionality, the approaches also differ in the way they resolve rate conflicts. Norilo proposes a priority scheme. Here we will consider direct interven-ence to deduce locally within the dataflow graph how rate is determined.

3. WHERE DOES TIMING INFORMATION COME FROM?
Here we want to address the question: Where should the local time advancement in a dataflow come from? Hence we want to address how to reason about timing control locally, how to understand that input data propagates, or stops propagating within the network and how it can be controlled and manipulated.

There are numerous sources of timed information. Input happens over time, or the source of input may itself be driven by a regular or irregular time-pattern. For example microphone input happens at audio rates, while ac-
update of the other. We call this property coupled. For example standard linear time-invariant filters are coupled. A new output has to be computed when a new input arrives, or if updated on the output side, a new input has to be acquired if a new output is requested. Coupled flowboxes occur often in dataflow and they typically correspond to being fed through in a fashion that is dictated by arriving (or leaving data). Usually that means that the sample rate, or irregular timing is unchanged, though one can design coupled flowboxes that do not have this characteristic. For example a down-sampler may emit only half the samples that arrive at its input but the rate of emission is still directly linked to the rate at the input.

However, not all relationships between input and output have to be coupled. Take for example a simple gain with a gain input and a signal input and output. Clearly changes in the gain input do not require an immediate change in the signal input. It is sensible to only apply the current gain when the signal path is being updated. We call this property of the gain input with regard to the gain output decoupled.

Decoupling has an important consequence for timing, as it breaks the relationship of input and output with respect to timing. Timing does not have to, but could be forced across a decoupled input-output pair. This property is central to deciding and controlling timing in various sub-flows of the dataflow network.

For example a gain is set to a certain value. After that, no change in the signal flow through the gain requires an update of the gain value. Another way to think about the difference between coupled and decoupled input-output pairs is with respect to required computation (and its cost). A coupled pair may not.

> Figure 2. Flowboxes can have different behavior with respect to propagating timing. (a) Coupled input-output pairs propagate timing. (b) A decoupled pair will not. (c) A fully decoupled flowbox has no coupling input-output pairs.

Fully decoupled are actually also regular entities within typical dataflow processing pipelines. Consider traditional unit generators, such as a sine oscillator. It has as output a signal, and a number of input parameters that control the nature of that signal (amplitude, frequency, phase). Notice that none of these inputs has to be a signal and no update of any of them necessarily means that a new output has to be computed right away. Hence a sine oscillator is fully decoupled.

4.1. Consequences of Directionality and Coupling

Most generally, a flowbox may indeed be used in both direction even in a concurrent or interleaved fashion. In principle invoking a coupled input-output pair simply invokes the internal computation of the unit. However, given that a flowbox is connected to other flowboxes there is a propagating consequence to directionality. If a coupled flowbox is updated in one direction and connected to another flowbox, that flowbox is invoked with this directionality as well. As long as all flowboxes in the chain are coupled, clearly all of the flowboxes will be updated (see Figure 3).

A decoupled flowbox ends this requirement to propagate updates. Hence coupled chains of flowboxes have their timing determined at either the output, or the input of the chain (or perhaps a rate generating flowbox in the middle of the chain, to be discussed later). A decoupled flowbox can make the timing undetermined. It is easy to construct examples with undetermined timing for a subflow of a dataflow network. Consider Figure 4. We have two cascaded sine oscillators connected to an accelerometer input on one side, and the digital analogue converter (dac) for audio playback on the other side. The input of the first oscillator will be updated at the accelerometer rate, while the output of the dac oscillator is updated at audio rates. Notice however that due to the fully decoupling nature of both sine oscillators, the flow between the two has no determined timing.

The idea of the pump is that the rate at those special connectors (whether pushed into it, if it is an input, or pulled from it, if it is an output) will dictate the rate of the coupled connection pair. Figure 5 shows example pumps inserted between the two sine oscillators of our example. If a pulling connection is used to drive a pump we call it a drain. Here the visual update rate is used to control the timing. Notice how the link between the two oscillators operates on a different timing schedule than either input or intended output. The second pump is driven by a fixed flowbox. This is a flowbox that can be programmed or linked to event-based user input (like touching the screen) to create timing information. Hence the update between the two sine oscillators is now determined but completely arbitrary.

One can define pumps with or without side-effects. Pumps without side effects simply use the rate of the signal at the pump input to drive timing but ignore the data. These pumping mechanisms without side-effect have in special form already appeared. For example in ChucK, a blackhole [21] is a pulling pump (drain) without side effects linked to the audio rate. In our multi-rate system this can be accomplished using a drain as seen in Figure 5 but connecting it to the dac to get audio rates.

Pumps with side-effect use this data to additionally affect the pumped signal. For example one can define a gain-pump, which is a hybrid of a gain and a pump. When the gain is set, the pump also update the coupled input-output. If both gain and input-output are connected
To create a new source and a new sink, which we call be usable within the dataflow. In order to allow this, we of a drumming performance. This timing clearly should be usable within the dataflow network. To allow this, we create a new source and a new sink, which we call Push and Pull. These are flowboxes that can have their input set programmatically as is the case for a Push (see Fig. 6). That program can either be linking the push to a user action such as a button press, or some other event, or a script that generates timing. A Pull does the same except that it allows the program to read out data using programmed timing. Here too the timing can come from user action on the interface. Their behavior is similar to bangs or number events in MAX/MSP or Pd in principle [19] except that due to the directionality choice we intrinsically consider dual pairs (Push and Pull).

4.4. User, event, or script based timing
So far we have only discussed a certain set of timing sources in detail, namely rated inputs (like accelerometers) and rated outputs (like audio output). However there clearly is a need for timing that is driven more directly by the user. Say a user pushes a button on the screen as part to audio-rate data, this would be the same as a ring modulator [6] whose overall rate is driven by the timing at the gain input.

5. HANDLING MULTIPLE SOURCE OF TIMING
An immediate consequence of a multi-rate pipeline is the presence of more than one source of timing and that those sources of timing can have different patterns. For example a pipeline may want to have two sinks update the over all flow. We allow this by simply allowing multiple pull links to connect to an output. However that means that the effective time pattern is the interweaving of the time pat-}

**Figure 6**. User input can provide timing information. Push (a) and Pull (b) flowboxes allow user-based, or scripted timing to be feed into the dataflow.

**Figure 7**. The depiction of directionality in our multi-rate dataflow. (a) A Push refers to the rate being derived from the input side. (b) A Pull refers to the rate being derived from the output side.

terns of each of the sinks. In some cases this may be un-problematic, or even desirable. For example imagine that two user-driven events can progress time. Then the interweaving of those events is precisely the desired behavior. If two interweaved timing are steady rates, then the interweaved rate is very likely an irregular time pattern with an effective rate different from either of the interweaved rates. The exception is the case when both rates are the same, in which case we get a doubled regular rate.

Consider the example depicted in Figure 7. Here both the dac and the visual output are connected to a dataflow network as sinks. The dac updates at 48kHz and the visual output updates at an irregular but averaged steady rate of 60Hz, depending on CPU load. Hence with interweaving, the dataflow network is updated at an effective irregular rate of 480060Hz. Clearly this is not desirable for audio playback due to artifacts introduced by the injected up rates from the visual side and it may not be the desired behavior for the visualization either. In fact what is likely the desirable outcome in such a case is that indeed one source of timing dictates the timing of the dataflow net- work, while the other sink only wants to read out information.

However we already have a mechanisms that in prin- ciple can address this: decoupling. Hence by taking any decoupled output of a timed dataflow that we want to ob- serve without affecting the rate of that dataflow we can do this. However we may want to provide a decoupled flow- box with that explicit function without additional side- effects. We call this flowbox Sniff. It consists of a coupled input-output pair, as well as a decoupled output. The ef- fect of this flowbox is to observe a running timed dataflow at some chosen times. Hence it acts as a resampling of one stream by another. In fact resampling in this nature fashion may not be the best choice in some situations, hence the simple Sniff serves as the simplest prototypical examples of a general class of resampling flowboxes, that may not only observe but use a derived observation (such as inter- polations) to achieve a desired resampling outcome (see Figure 8).

The dual argument also applies for inputs. Input-side timing if connected to the same input will interweave. However pragmatically we found this to be not a partic- ularly useful. Inputs tend to either be well mixed by exist- ing duplicated flowboxes, such as gain, or terminate in a decoupled input, hence the interleaved pattern having no particularly relevant impact.

However in principle a merged input with a steady rate can be achieved by synchronizing injection of one dataflow with the other and this synchronization too may be subject to processing such as interpolation. There is a wide choice of thinkable injection functions. In fact the gain flowbox we have discussed earlier is a form of inter- polation.

Finally one can give control to the rate choice to the dataflow network itself by providing. This can be achieved through a pair of flowboxes that allow the selection of the input or output, respectively that is currently used to de- rived rate. For inputs lines that are not selected to provide rate information can either be ignored, or output. For output, those lines too can either be ignored or allow ob- servation of the data stream.

6. MANIPULATING TIME AND RATES DIRECTLY

We call a flowbox time-manipulating if part of its effect is the change in temporal pattern between coupled input- output pairs. A range of such effects is thinkable. An illuminat- ing example is the zPuls flowbox. zPuls observes a data stream and emits an impulse whenever it observes a zero- crossing. More generally we call flowboxes who emit data based on conditions on the data stream conditional [9]. It will, however be silent if no such zero crossing occurs. Carefully, flowboxes time manipulating. The timing of the pulse pattern generated by it is very likely very differ- ent from the rate of the signal observed. zPuls in fact is a very general flowbox that can be used to generate rich regular and irregular timing patterns with the pattern con- trolled by the wave form. The property here is of course that an impulse and hence followup computation is only invoked when it is relevant, and this can be made arbitrarily rate. zPuls is a prototype for a mechanism that we call retrigerring. Not all incoming timed data propa- gates. Rather whether some outcome timed event is trig- gered is conditional on some property.

Another class of time-manipulating flowboxes have to do with downsampling. Consider downsampling by a fac- tor of two. The simplest possible way to achieve down- sampling is to simply omit every other sample from a sig- nal. Traditionally in this is often seen as setting the omit- ted samples to zero. We however to the option here to simply only send data half of the time, hence manipu- late time at the same time as we reduce the actual data. A general down-sample at an arbitrary downsampling rate will have to interpolate, hence leading to a general solution to down-sample to arbitrarily lower rates. An adaptive downsampler can be build from a simple differentiator, which we call DiffGate. This flowbox will only emit data if there is change input-side from previous data. If the data is indeed slow-changing and over large stretches constant this flowbox will remove the constant- rate behavior that does not actually give changed data. A related flowbox QuantGate will only emit a new sample if the accumulated difference to the last emitted data is larger than some specifiable threshold. The effect is that only changes above a specified quanta will create a timed event. This flowbox can be used to adaptively downsam- ple slowly changing but constant signals. This is a very general flowbox that works well to adaptively downsample slow-changing effects such as amplitude en- velopes.

7. UNIVERSALIZATION OF PROCESSING UNITS WITH TIME-VARIABILITY

Already in [9] we discussed the property of universal plug- ability with respect to being able to connect processing units with different input and output semantics by defining a normed data-stream and making each processing unit understand how to translate the normed data into its own suitable semantics. Here we also get a universalization with respect to time primitives. For example consider that a drum ma- chine is to be built, where different weights of impact should be stored in a circular buffer. However we already have flowboxes that are meant to store circular buffered data and allow its playback and rate control. This flow- box is called Looper and was originally designed to allow recording and looped playback of audio samples. How- ever, in our pipeline design there is nothing privileged about the audio rate, or even a regular rate. We can use the same flowbox another, if we want significantly slower, or irregular rate. Hence we can use the samples stored in Looper as the intensity levels of a drum and play it back at rates appropriate for a typical drum rhythm. We can even choose to make the timing irregular, to perhaps al- low a swing pattern to apply, without having to define a Looper block different from the one we already have. In this sense all flowboxes that we do have now can be used in different timing situations. For example an averaging filter can be used on irregular data.

8. REALIZATION IN URMUS

What we have discussed here is realized in urMus, an in- teraction environment for commodity mobile devices [8]. However urMus hides many aspects of the rate control from the user in the domain of the interaction. For example the directionality of the flow is not visible in the visual rep- resentation of the interface (see Figure 9). This is purely
Figure 9. (a) An amplitude-modulation patch in urMus. Notice how the directionality of the connectivity is hidden in the representation. (b) The actual patch structure as implemented, using our symbolic notation.

between control and signal flows, but often knowledge of rates is actually not explicitly required. Notice also that individual inputs are presented as separate interaction entities (SinOsc(1) Amp and SinOsc(1) Freq are separate widgets). This too is justified to organize interactions into finger-sized elements.

8.1. The Lua interface
UrMus itself has a Lua layer. Lua is a fast, embeddable script language with certainly attractive properties for malleable user interface design [11]. Within the Lua language, aspects of UrMus that are otherwise separate can be programmatically related. The dataflow pipeline discussed here is exposed to lua through API functions. Key aspects of the interface is the possibility to create new instances of flowboxes, and to change the connectivity between them. The function to create a new instance CreateFlowBox (prototype) creates a new instance of the SinOsc flowbox from the global prototype FBSinOsc. Some flowboxes are stateless and hence do not require instancing. Many hardware sources and sinks are of this type. For example we can use dac = _G["FBDac"] to get the global instance of the audio playback sink. In order to establish a pull link between the sin oscillator and the dac we use the method outletBox:setPullLink:In (index, infobox, outindex).

dac:setFullLink(0, mySinOsc, 0)

Alternatively one can use member functions of the flowbox itself.
dac:in:SetPull (mySinOsc:Out)

The moment this is executed, a sine will play at default frequency of 440 Hertz and at normed amplitude. To connect this to an accelerometer that pushes into the frequency we do the following:
accel = _G["FBAccel"]
accel:setFullLink(0, mySinOsc, 0)

And one can of course also use the flowbox member notation.

accelxFSetPush (mySinOsc:Freq)

To undo connections one can use :removePullLink() and :removePushLink() or :removePull() and :removePush() in flowbox member notation, with the same arguments that one used to create the connection. These sets of functions and methods are already sufficient to generate fully functional dataflow networks and through directionality control rates. However there are further methods provided that allow an interface to sensibly represent a flowbox in whichever form it prefers. So can a flowbox be queried for the number and labels of each input and output it offers via the methods :numIns() and :getNumIns() :getNumOuts(). One can find out if a flowbox can be instanciated by querying via :isInstantiable(). There are also methods to query the (de-)coupling property via :getCoupled() and :isCoupled(), and to push numbers into or pull numbers from a data network via :push() and :pull().

9. CONCLUSIONS
In this paper we presented the design of a flexible-rate and multi-rate dataflow architecture that does not have an intrinsic master rate. The rates of local sub-flows of the dataflow network are derived from the rates of the hardware and interaction elements that are connected to it. We gave how the variable rates within the network can be determined and controlled and gave a range of mechanisms to manipulate time.

There are a number of benefits to this design. One is that it is well-behaved for live-patching [9]. Furthermore it allows subparts of the network to run at computationally relevant speeds, not the usually higher audio rates, allowing better performance specifically on the still slower mobile hardware. Finally having flexible control over time within a dataflow leads to a universalization of the use of flowboxes that is liberated from the assumption of a dominant single rate.

10. REFERENCES
The “warm chorus” effect described in this paper was initially designed for a string quartet concert in which a simulated string orchestra effect was requested by the performers. Due to the fact that the various “industry-standard” chorus effects which were available to us at the time left a lot to be desired from a musical standpoint, it was decided to create a tailor-made effect for the concert, using the Max/MSP environment, which could be used subsequently as a generic real-time chorsusing tool. In order to overcome the mechanical quality inherent in many generic chorus algorithms, the “warm chorus” uses a model based on the performer arrangement in the string section. This novel approach inspired by human behaviour is further enhanced with additional signal processing techniques to help reduce the perceived “phasiness” of the processed sound, resulting in a chorus effect that is more natural than traditional off-the-shelf chorsing effects.

1. INTRODUCTION

The quest for a more realistic chorsing effect appropriate for instrumental (rather than exclusively vocal) sound sources was initiated while collaborating with the New Asia String Quartet in 2007 [11]. The quartet was performing George Crumb’s Black Angels alongside Haydn’s Seven Last Words, Op.51, which additionally exists in a prior orchestral version and a later choral version by the composer. Since we were already amplifying and adding artificial reverb to the instruments for the performance of the Crumb piece, the members of the quartet thought it would be appropriate to also process the Haydn work in order to create a larger, ensemble-like sound, thereby alluding to the various pieces of the piece.

Our unattainable dissatisfaction, however, with the credibility of the chorsing effects available to us in both hardware and software, coupled with the apparent lack of chorsing instruments for instrumental sound, led the author to embark on the creation of a dedicated string orchestra simulation effect for the concert. Since the objective was to create a rich string orchestra sound, the author (also an amateur string player) decided to use the player arrangement of an orchestral string section as an architectural model for the algorithm. The original version of the resulting “warm chorus” was first used in concert in the Spring of 2007, and has since undergone several updates, modifications and improvements.

2. THE TRADITIONAL CHORUS EFFECT

Artificial chorsing effects can be traced back to the early days of electronic music [1] [8]. These effects were implemented electronically and can still be found today in many analog guitar pedals and electronic effect units, due to their having henceforth furnished a quintessential (and sometimes clichéd) “sound” within many musical genres.

2.1. The Basic Chorus

In the realm of audio digital signal processing, the most basic chorus effects imitate their analog predecessors and are thus implemented with a series of modulated taps on a single delay line, summed together to form a chorsused output (see Figure 1) [3] [10]. Although this processing scenario is computationally efficient, it has several drawbacks including unwanted notch filtering throughout the spectrum and a perceived mechanical regularity to the resultant output sound, due to the Low Frequency Oscillators (LFOs) used for modulation.

Figure 1. Block diagram of a traditional chorus effect.

2.2. The Industry Standard Chorus

The “industry standard” version of this effect attempts to compensate for the unwanted comb filtering by basing the chorus, paradoxically, on a comb filter — albeit one that includes separate delay-line taps for its feedforward and feedback loops (see Figure 2) [4]. This important modification, along with using the inverse of the signal in the feedback loop, both improves the thickness of the sound and mitigates to some extent the perceived “phasiness” in the upper end of the spectrum.

Two or more of these units may be used in parallel for added sound density, although the result still has a mechanical effect more closely resembling its analog forebears rather than a genuine ensemble performance.

2.3. Pitch Synchronous Overlay Add Techniques

More recently, chorsisng algorithms based on Pitch Synchronous Overlay Add (PSOLA) techniques have been implemented [9]. Using pitch-synchronous techniques tend to reduce the unwanted artefacts found in traditional chorsusing algorithms, though they are not eliminated completely. Some PSOLA-based chorsing algorithms require pre-calculation of the input sound, limiting their use to non-real-time situations, but the main drawback from a user’s perspective — even in the real-time implementations — is that, due to the pitch-synchronous nature of the algorithm used, they require a monophonic input source for proper pitch analysis and waveform segmentation.

2.4. Spectral Techniques

Some spectral-domain techniques for chorsusing have also been explored in recent years. The most successful of these use cross-synthesis techniques to combine a solo voice and the sound of a pre-recorded chorus [2]. Alas, the spectral-domain techniques have hitherto been primarily limited to the modulation of vocal sounds; to extend them to the processing of instrumental sounds would otherwise require pre-recording an appropriate actual instrumental ensemble, something which is not particularly difficult, but not always practically feasible.

3. AN ORCHESTRAL SECTION MODEL

In contrast to traditional chorsing algorithms, which for the most part attempt to design the effect on computational terms, the design of the “warm chorus” is fundamentally based on a human-inspired model: that of the placement and comportment of performers in an orchestral string section. The idea is that, in a string section, the first stand will probably have performers who are the best players in the section: they are the leaders for those sitting behind them and will likely play with the best intonation, have more accurate timing and, being more confident performers, will play slightly more prominently. Additionally, the sound of these players will be the least delayed and subjected to air absorption since they are seated closest to the audience. The players in the back of the section may not be such proficient players as those in front of them, so we presume that with each subsequent stand, their transposition from ideal intonation will be farther than that of the players in front of them, and their timing may be slightly more lax. Since these players are also likely to be more timid or uncertain about the accuracy of their playing, their volume will be conseqently lower on the average, but have a larger range of variation than those players ahead of them. Finally, the sound of each stand will be slightly filtered and delayed due to air absorption and distance from the audience.

Although this is a gross over-simplification of the make-up of an orchestral section, and does justice neither to performers, nor to conductors and artistic directors (neither of whom would congregate all of the weakest players in the back of the section), it does, notwithstanding, provide a simple and effective human-based model upon which to design a chorus effect in much the same way that the general description of the dimensions and materials of a concert hall may offer a simple, effective model for a reverberation algorithm.

4. DESIGNING THE “WARM CHORUS”

Instead of being built upon the traditional modulated delay line, the “warm chorus” uses a set of time-domain granular harmonizers with optional input delay, whose transpositions are set to different minute (generally in the range of 2 to 40 cents) values. Although this is significantly more computationally expensive than traditional algorithms, it more accurately models a multiple-player scenario. Using such a time-domain transposition method is considerably cheaper than using spectral-domain pitch shifting, and the somewhat “blurry” quality of the pitch-asynchronous overlap-add of the harmonizers actually helps achieve a thicker overall sound with fewer transpositions (and less of a regular “woozy” vibrato-like feeling arising from sinusoidally modulated delay taps). The granular harmonizers used for the first version of the “warm chorus” were simply taken from the standard Max/MSP harmonizer example, transpose~[1], but subsequently a new 4-overlap granular harmonizer with built-in allpass filtering was designed for the purpose.

4.1. Harmonizer Structure

The modified harmonizer used for the chorus effect is designed with four sequential overlapping sinusoidal windows each 90 degrees out of phase from the previous. These windows are controlled by a single phase~ object[2] which simultaneously synchronizes the read locations of the four delay taps and the windowing functions, and acts as a trigger signal for four sample-and-hold units. The delay line read locations have some degree of random variance that is held constant for the duration of the window; this helps create irregular phase cancellation and reinforcement in the upper end of the spectrum. A set of four 8th order allpass filters inserted before summing the overlapping windows further contributes to shifting phases throughout the spectrum, so any beating that occurs after summing the closely.

1The MSP transpose~ example is an implementation of the earliest and most basic granular digital pitch shifters, which were modeled on earlier analog rotary tape head devices See [1] and [8] for a detailed history.

2The phase~ object in Max/MSP generates a periodic ramping function from 0 to 1.

Richard Dudas
Hanyang University College of Music
222 Wangsimi-ro, Seongdong-gu
133-791 Seoul, South Korea

ABSTRACT

The “warm chorus” effect described in this paper was initially designed for a string quartet concert in which a simulated string orchestra effect was requested by the performers. Due to the fact that the various “industry-standard” chorus effects which were available to us at the time left a lot to be desired from a musical standpoint, it was decided to create a tailor-made effect for the concert, using the Max/MSP environment, which could be used subsequently as a generic real-time chorsing tool. In order to overcome the mechanical quality inherent in many generic chorus algorithms, the “warm chorus” uses a model based on the performer arrangement in the string section. This novel approach inspired by human behaviour is further enhanced with additional signal processing techniques to help reduce the perceived “phasiness” of the processed sound, resulting in a chorus effect that is more natural than traditional off-the-shelf chorsing effects.

The “warm chorus” effect described in this paper was initially designed for a string quartet concert in which a simulated string orchestra effect was requested by the performers. Due to the fact that the various “industry-standard” chorus effects which were available to us at the time left a lot to be desired from a musical standpoint, it was decided to create a tailor-made effect for the concert, using the Max/MSP environment, which could be used subsequently as a generic real-time chorsing tool. In order to overcome the mechanical quality inherent in many generic chorus algorithms, the “warm chorus” uses a model based on the performer arrangement in the string section. This novel approach inspired by human behaviour is further enhanced with additional signal processing techniques to help reduce the perceived “phasiness” of the processed sound, resulting in a chorus effect that is more natural than traditional off-the-shelf chorsing effects.

The “warm chorus” effect described in this paper was initially designed for a string quartet concert in which a simulated string orchestra effect was requested by the performers. Due to the fact that the various “industry-standard” chorus effects which were available to us at the time left a lot to be desired from a musical standpoint, it was decided to create a tailor-made effect for the concert, using the Max/MSP environment, which could be used subsequently as a generic real-time chorsing tool. In order to overcome the mechanical quality inherent in many generic chorus algorithms, the “warm chorus” uses a model based on the performer arrangement in the string section. This novel approach inspired by human behaviour is further enhanced with additional signal processing techniques to help reduce the perceived “phasiness” of the processed sound, resulting in a chorus effect that is more natural than traditional off-the-shelf chorsing effects.
spaced transpositions will not have an obvious regularity. Using a lower-order allpass filters saves only minimal CPU, and did not appear to be as effective upon repeated testing.

This harmonizer unit, diagrammed for the sake of clarity in Figure 3, is not very useful for the purposes of general audio transposition (where 2-overlap seems to be both more practical and produce better results). Nonetheless, for the purposes of a chorusing algorithm it works quite well: the transposed sound is stable within the -100 to +100 cent transposition range and contains some necessary degree of spectral complexity resulting from irregular delicate amplitude modulation in different regions of the spectrum.

4.2. Harmonizer Intonation

It is worth mentioning that, quite unlike an LFO-modulated delay tap — which produces alternating sharp and flat transpositions of the input sound — all of the transposition values used for the harmonizers are positive values (i.e. sharp). Young musicians in student orchestras are often told that sharply tuned notes are perceived to be less “out of tune” than flat ones. Whether or not this is true may be open to debate, but research on intonation has shown that professional musicians have an overall tendency toward higher-pitched intonation for the majority of both equally-tempered and just intervals [6], so this sharp-tuning scenario has been used in the “warm chorus” effect.

4.3. Additional Time-Domain Processing

The output of the more greatly detuned harmonizers is subjected to slight lowpass filtering to simulate air absorption in an exaggerated manner. This also mitigates eventual aliasing for the majority of both equally-tempered and just intervals [6], so this sharp-tuning scenario has been used in the “warm chorus” effect.

4.4. Magic Numbers

The relationships of the transposition values, as well as the relationships of delay times, window sizes and random variance of the various parameters, are generated using numbers from the prime number series. This was intuitively decided upon in order to further help reduce perceptually regular beating, modulation and phasing between multiple closely-spaced transpositions, delays and window durations. It seems to be fairly effective. The parameters for the allpass filters in the harmonizer units are randomly set, empirically.

For the moment this works satisfactorily, although it is certainly an area for future research and improvement.

5. ADVANTAGES AND DRAWBACKS

The resulting effect is aurally convincing, though computationally expensive. Processing can be reduced, and the result is still passable if only 4x4 (instead of 8x8) transpositions are used. It is possible that stages of 4 harmonizers could be used to achieve 64 transpositions, but this has yet to be tested. The other major drawback is a fair amount of input/output latency due to the inclusion of the STFT.

On the positive side, the principal advantage of the algorithm is that it can accept any kind of input sound — monophonic, polyphonic, noisy or reverberant — and the processed result is equally effective. The “warm chorus” is also easy to use: it is fitted with a few simple user controls to adjust the intonation, sloppiness and fullness as appropriate for different types of audio input.

6. CONCLUSION

There is further work to be done refining this algorithm, but it is nonetheless musically effective in its current form. Above all, the author believes its novelty lies in the underlying orchestral section and human behavioural model. This will hopefully provide inspiration for re-thinking the design of other traditional audio effects.

7. ACKNOWLEDGMENTS

The author would like to thank violinist Eui-Myang Kim for instigating the project, as well as composer and colleague Jongwoo Yim for his help and advice during the initial stages of conception and development.

8. REFERENCES

A SPECTROGRAPHIC ANALYSIS OF VOCAL TECHNIQUES IN EXTREME METAL FOR MUSICOLOGICAL ANALYSIS

Eric Smialek (1, 2) Philippe Depalle (2, 3) David Brackett (1) CIRMMT, McGill University, Montréal, QC, Canada

ABSTRACT

Extreme metal genres such as death metal and black metal rely on music analysis to seek alternative methods to Western notation-based analysis, especially when one asks what means of expression their vocalists may draw from in order to seem convincing and powerful to fans. Using spectrograms generated by AudioSculpt, a powerful sound analysis, processing, and re-synthesis program, this paper demonstrates a mixed application of spectrograms and conventional music analysis to vocals in two separate contexts: an a cappella recording in a soundproof laboratory and a commercial recording with a full band. The results support an argument for the utility of spectrograms in revealing articulations and expressive nuances within extreme metal vocals that have thus far passed unnoticed in popular music scholarship.

1. INTRODUCTION

Popular music scholars have long insisted that details of musical sound such as rhythmic and melodic inflections or timbral characteristics are central in importance to popular musicians and their audiences [1, 2, 3, 4]. Accordingly, some of the leading popular music scholars such as Richard Middleton [1] and Philip Tagg [2] have expressed criticism of traditional analyses of popular music that overlook these features, often due to a reliance on Western music notation. One alternative is to visually represent sound through spectrograms, and it has now been nearly thirty years since Robert Cogan made his case for the utility of spectrographic technology to reveal information about musical expression in a genre of music for which little to no analytical methods have been established. Using real-time spectrographic displays, this paper will demonstrate how spectrograms can be useful research tools for scholars who require or seek alternatives to notation-based analysis.

2. THE EXTREME METAL VOICE

2.1. Basic Aspects of Vocal Production

To produce the vocal sounds characteristic of death metal and black metal (as well as related sub-genres), extreme metal vocalists pass air through the ventricular folds (or “false vocal chords”) located a few millimetres above the vocal folds (see [12] for anatomical details). This allows extreme metal vocalists to achieve the large spectral spread of energy visible in spectrograms.1 Extreme metal screams can be performed by either inhaling or exhaling, resulting in two very distinct styles of screaming. The different directions of air flow can be thought of as akin to the linguistic distinction made between voiced and unvoiced methods of articulating consonants: when performing exhaled vocals, one’s larynx vibrates, indicating that the vocal cords are vibrating—rather forcefully—whereas this vibration does not occur with inhaled vocals. This basic difference has a profound effect on the overall sound quality of expressive ways. To lengthen their vocal tract and produce a lower sounding voice, vocalists may simply lower their chin or conversely angle their chin upwards to shorten the vocal tract length for a higher scream. It is also possible to change the vocal tract length by raising or lowering the larynx (or voice box), a process which happens automatically in conjunction with the rounded or spread lip shapes we use to articulate different vowels (see Figure 2) [12].

Figure 1. Some basic distinctions between extreme metal vocal techniques as demonstrated by a volunteer extreme metal vocalist.

The extreme metal voice and their roles in music pieces, we decided not only to perform spectrographic analyses, but also to extract parameters that allow for a partial resynthesis of key sound components in order to validate our assumptions.

To explain, the following is a brief description of the basic principles underlying AudioSculpt functionalities: AudioSculpt generates a short-term Fourier transform (STFT) representation of the sound.1 In order to optimally display the sonogram image, STFT parameters must be chosen including analysis window size and shape, step size (that determines how the STFT fits over the signal characteristics), and fast Fourier transform (FFT) sampling. For each sound example in this study, we used a Blackman window of size Ms=2400 with a step size of M/8 and a number of channels equal to 4096. Once a satisfactory image is in view, it is ready to be analyzed.

Qualitative judgments can be made based on the obtained spectrographic image but more precise measurements of vowel formant quality are also taken by first synthesizing a new sound file based on the original sound’s vowel formants using one of AudioSculpt’s several spectral gain filters (in each case during this study, the pencil filter tool). These filters change the gain of certain frequency regions defined by the user within the resynthesized signal. For this study,

This research was greatly facilitated through funds from the Social Sciences and Humanities Research Council (SSHRC) and a CIRMMT Student Award.

1 A similar process is used in Mongolian throat singing where singing voice formants are used to convey melodic information [13].
specifically, AudioSculpt’s "Partial Tracking" feature was used to track partials using multiple breakpoint functions for each sinusoidal component (a procedure that works with inharmonic signals as well as harmonic ones). A new sound was then synthesized by amplifying the formant regions of the original signal.

AudioSculpt’s diapason tool was then used to take frequency readings from the synthesized partials. When the tool is pointed at a particular place on the sonogram, its frequency and amplitude are displayed, a corresponding sine tone sounds, and a two-dimensional projection of the synthesized signal appears. This procedure allows for an analysis of the original sound by hypothesizing that certain components of it appear to be important, resynthesizing those components, and, consequently, analyzing them.

4. A RECORDED IMPROVISATION
In order to investigate the features of extreme metal voices presented in section 2 in a more musical context, we asked a vocalist to perform a vocal improvisation using whatever extreme metal vocal techniques he wished. Given the somewhat artificial performance environment, the vocal technique might make in a concert setting or full-band recording, devoid of reverberation and there were no instruments to accompany the vocalist—the volunteer vocalist’s improvisation shown in Figure A1 (see URL), if not an exact reproduction of what was virtually a void of reverberation and there were no instruments to accompany the vocalist—the volunteer vocalist’s improvisation shown in Figure A1 (see URL), if not an exact reproduction of what was virtually a

4.1 A Spectrographically-Informed Music Analysis
In addition to the spectrogram, Figure A1 also contains phonetic transcriptions as well as analytical annotations provided at the bottom of the figure. As indicated by the rhythms given below the spectrogram, the vocals fit neatly into a regular 4/4 meter, indicating that the vocalist kept a regular pulse in the regions to be filtered were defined according to a computer-based analysis of the original sound file’s dynamic markings at measure 7); each half contains a recurring, slightly varying rhythm (labelled x and y) that generally does not occur in the other half; and, following the contours visible in the spectrogram, both divisions exhibit an arch-like quality of intensification whereby the vocalist’s screams steadily increase in volume and high spectral frequencies (measures 2–4, 9–11) before rapidly calming with quieter, lower vowels (measures 6–8, 12–15). The total result is a sample of improvised extreme metal vocals which demonstrates controlled musical techniques in a regular manner; one that could be heard as loosely narrative or rhetorical in the sense that it creates regularly spaced climaxes, moments of repose, and gradual variations. If there is a clear sense of musicality to be found here, what can be inferred about the vocal techniques used to achieve it?

4.1.2. Inhaled vs. Exhaled Vocals
Inhaled vocals in Figure A1 are shown by boxes around the notation below the spectrogram, indicating that an inhaled and inhaled vocals are employed about equally during the improvisation. Though aurally distinguishable from exhales and with extremely similar, the distinguishing characteristic of the inhales are not immediately apparent from the spectrogram (at least at the resolution given in the example). Nevertheless, it can be used to slightly alter vowels. Specifically, when a vowel is rhotacized, i.e. coloured by /ɹ/, the third formant becomes lowered.2 Even if this third-formant lowering is not as directly tied to an impression of heaviness as the lowering of the first two formants, it nevertheless provides a way for the vocalist to create variety and, on a social-perceptual note with regards to paralanguage, it seems more than a coincidence that the /ɹ/ sound is often i to stop the flow of air through the vocal tract.3 Having drawn a number of inferences as to why certain patterns appeared in the improvisation, the most notable is that there exists a consistency to the vocalist’s use of particular phonemes in such a way that they seem to be fundamental to the most salient musical features of the improvisation.

4.2.2. The Importance of /ɹ/ Sounds
This last point raises one of the most conspicuous consistencies observed throughout the recording sample. The vocalist very frequently alternates between /ɹ/ and /ɹ/ sounds with both inhaled and exhaled vocals. These phonemes occur for nearly all the articulations contained within the solid box given in the left-centre column and, in the case of the one exception, the /ɹ/ combination is merely displaced by a beat, and this is marked by the arrow in the left-centre column. The versatile phonetic (”/) of Standard North American English is especially worth of emphasis here for both the acoustical and physiological advantages it offers the vocalist. This sound does not require a stoppage in air flow so it can be used as a vowel (e.g. /ɹ/ as in “fit”) or as a consonant (e.g. /ɹ/ as in “rapid”). As a result, it is especially suited to the physiological difficulties of articulating consonants with vocals (note again that nearly all of the other consonants are exhaled). Because it can be articulated with a continuous air flow, the /ɹ/ can be used to slightly alter vowels. Specifically, when a vowel is rhonahcized, i.e. coloured by /ɹ/, the third formant becomes lowered.2

3 Paralanguage can roughly be understood to include all forms of non-verbal communication. These can include facial expressions, vocal tone, gestures, intonation, pitch, volume, pauses, pauses, and silences, as well as prosaic and timbral modifications to ordinary speech that shade meaning [16].
5. AN EXCERPT FROM “THE VOWEL SONG”

Framed as a public service promoting literacy, “The Vowel Song” by death metal band Zimmer’s Hole begins with vocalist Chris Valagao reciting the vowels of the alphabet (henceforth “letters”) to distinguish from phonetic vowels in long sustained screams, harmonized by three guitars in homorhythm (see Figure A3). The slow punctuations of each homorhythmic attack, combining voice, low power chords (shown only as roots in the example), and two harmonized lead guitars, not only lend a certain satirical grandiosity to song, they also help to create the sensation of Valagao’s unpitched screams possessing a kind of melody, drawn by a precise control of formant frequency locations.

5.1.1. Formants in Flux

Although it may not be immediately evident in the spectrograms given here, there is quite a great deal of variation to the formant frequencies used in the example. In order to illustrate this variation, Figure 3 plots the position of each letter that Valagao screams within vowel space, i.e., a graph which plots vowels according to the frequency of the first formant on the y-axis and second formant on the x-axis. Of course, some of the letters Valagao screams are actually diphthongs or triphthongs. Accordingly the most steady-state vowel within each letter is identified with a dot on the graph and arrows leading up to or away from it depending on how the phonetic transitions occur. To illustrate an example, Valagao’s screamed letter “u,” represented by plot #5, is performed in such a way that it traverses vowel space beginning near /ʊ/ (as in “hit”), reaching its most steady-state point near /u/ (as in “harm”), and finally moving towards the lower formant frequencies in between /ʊ/ and /u/ (as in “hook” and “caught” respectively). It becomes clear how much formant variation is involved in this song introduction from only taking into account all the steady-state plot points, let alone taking into account all the quick diphthong transitions indicated by the arrows.

5.1.2. Interactions between the Voice and Guitar

If these changes in formant frequency are compared with the pitch contour of the guitar parts (which move in parallel motion), a surprising correspondence appears between the guitars’ changes of pitch direction and the movements of the first formant. As the first letter changes to the next, the lower formant decreases in frequency paralleling the descent of the guitars (see the “changes of direction” arrows in Figure A3). With the next letter, the lower formant reverses direction just as the guitar does. This pattern of alternating upwards and downwards directions, shared between the guitars and the voice’s first formant, continues until the guitars and voice break the homorhythm. What’s more, there is an even stronger correspondence between the highest lead guitar part and the voice’s first formant movements.

is shown in Table 1 (see next page) which compares the high lead guitar melody with the first formant of each vowel at its point of greatest stability and sustain (the dotted points in Figure 3). Both are shown as frequency values in Hz and in terms of pitches that correspond to those frequencies. A comparison of these pitch values (including their octave position) reveals a striking relationship. With a margin of about one semitone above and below the upper guitar part, the voice’s first formant parallels the exact contour of the guitar part.

![Figure 3](image-url)

**Figure 3. Formant positions from the opening of “The Vowel Song” plotted in vowel space** [17]. Each letter that Valagao screams is assigned a number. Arrows indicate phonetic transitions before and after a vowel is stabilized and sustained.

Taking into consideration that there is usually a frequency range of around 100Hz over which each formant’s energy is significant (the values in Table 1 sample the average frequency for the first formant) and that the voice is in rhythmic unison with the guitars, it does not seem far-fetched for a listener to perceptually connect the guitar melody and the formant movements, thereby imagining a kind of melodic motion assigned to a series of unpitched screams. Even after the homorhythm breaks off, one can discern further frequency-oriented connections between the voice and its surrounding musical contexts. As the vocalist finishes preparing the last letter by screaming the words “and sometimes,” a diving upper harmonic merges with his third formant beginning with the onset of the final letter Y. In this brief intro to “The Vowel Song,” the widely taken for granted division between timbre and pitch appears especially blurred.

6. CONCLUSION

Having observed extreme metal vocal techniques in both a relatively controlled recording session and at work in a commercial studio recording, it should now be clear that the extreme metal voice is far from the simplistic percussive device that it is often assumed to be. If such assumptions are in no small part the result of deeply entrenched habits of describing vocal music primarily in terms of pitched melodies, the extreme metal voice can serve as an invitation to approach the study of musical expression in new ways. Having taken an interdisciplinary approach to the extreme metal voice, the results of this paper support ongoing arguments for the musicalological utility of spectrograms in drawing attention to subtle means of musical expression that can easily be overlooked.

### Table 1. A comparison of frequency values between the highest lead guitar’s melody and the first vocal formant in its point of greatest stability.

<table>
<thead>
<tr>
<th>No. 1 A</th>
<th>No. 2 E</th>
<th>No. 3 I</th>
<th>No. 4 O</th>
<th>No. 5 U</th>
</tr>
</thead>
<tbody>
<tr>
<td>C6/1050 Hz</td>
<td>G5/780 Hz</td>
<td>Eb/6/1240 Hz</td>
<td>C6/1050 Hz</td>
<td>Eb/6/1240 Hz</td>
</tr>
<tr>
<td>C5/540 Hz</td>
<td>G4/380 Hz</td>
<td>F5/750 Hz</td>
<td>B4/510 Hz</td>
<td>E5/650 Hz</td>
</tr>
</tbody>
</table>

7. REFERENCES

SOUND SYNTHESIS WITH AUDITORY DISTORTION PRODUCTS

Gary Kendall
Sonic Arts Research Centre
School of Creative Arts
Queen’s University Belfast
garykendall@me.com

Christopher Haworth
Sonic Arts Research Centre
School of Creative Arts
Queen’s University Belfast
chaworth01@qub.ac.uk

Rodrigo F. Cadiz
Center for Research in Audio Technologies, Music Institute Pontificia Universidad Católica de Chile recardz@ac.cl

ABSTRACT

This paper describes a method of sound synthesis based on auditory distortion products, often called combination tones. In 1856 von Helmholtz was the first to identify sum and difference tones as products of auditory distortion. Today these tones can best be understood as otoacoustic emissions, the ‘distortion’ being produced by the cochlear amplifier. The use of distortion tones in musical composition has largely been rudimentary and dependent on very high amplitudes in order for the distortion products to be audible and musically useful. Also discussed here are methods whereby the spectra of subjective distortion tones can be matched to target spectra of up to 5 harmonics, thus making auditory distortion a controllable method of sound synthesis.

1. INTRODUCTION

This paper describes a method of sound synthesis based on auditory distortion products, often called combination tones—a method that creates controlled auditory illusions of sounds that are not present in the physical acoustic signals. Numerous composers have used distortion tones (Niblock, Kirkegaard and Amacher) and audiences have experienced these effects, which would often be described as fuzzy, ghostly tones near to the head, in concert halls and in art galleries. Combination tones are typically described as products of non-linearities in the auditory system produced only at high sound levels. In actuality, these sounds are evoked distortion product otoacoustic emissions (DPOAEs) that are produced by the active amplification of the basilar membrane and are active even at low sound levels. Approaches to the use of subjective distortion tones in technological music have largely been rudimentary and haphazard, and the subject of how to utilize subjective combination tones in musical contexts has not been approached in a truly systematic fashion.

2. DISTORTION PRODUCT OTOACOUSTIC EMISSIONS

2.1 Studies with Two Pure Tones

There is a long history of research into what was commonly called combination tones (CTs). Most studies of combination tones have used a combination of two pure tones as stimuli and studied the listener’s perception of a third tone, not present in the original stimulus, but clearly audible to the listener. In 1856 von Helmholtz was the first to identify sum and difference tones [15]. For two sinusoidal signals with frequencies \( f_1 \) and \( f_2 \) such that \( f_2 > f_1 \), the sum and difference tones have the frequencies \( f_1 + f_2 \) and \( f_2 - f_1 \) respectively. Later on Plomp [10] identified many additional combination tones with the frequencies \( f_1 + Nf_2 \) and \( Nf_2 \). Originally it was thought that CTs occurred only at high intensity levels that then drove the essentially linear mechanics of the physical auditory system into a non-linear region. The original theory was that a mechanical non-linearity was located in the middle ear or in the basilar membrane. Goldstein [5] provided a particularly thorough investigation of combination tones produced by two pure tones. The frequency, amplitude and phase of the distortion tones were determined using a method of acoustic cancellation, first introduced by Zwicker [16]. Importantly, Goldstein demonstrated that combination tones were present at even low stimulus levels and thus could not be products of mechanical nonlinearity in the way it was originally conceived.

The theory of mechanical non-linearity has been displaced after the recognition that parts of the inner ear, specifically the outer hair cells of the basilar membrane, act as an active amplification system. So, the ear rather than being a passive system with nonlinearities is an active system. Nonlinearities are best explained in terms of the workings of the cochlear amplifier. Seen from this perspective, CTs can best be understood as otoacoustic emissions (OAEs), that is, signals generated by the active components of the cochlea. Specifically, they are distortion product otoacoustic emissions (DPOAEs) evoked by physical acoustic stimuli. Incidentally, DPOAEs also propagate back through the middle ear and can be measured in the ear canal. They are typical of healthy hearing systems and their testing has become a common diagnostic tool for identifying hearing disorders [8, 7].

Of the many distortion products, two types are potentially the most useful to music and sound synthesis because of the ease with which listeners can recognize them: the quadratic difference tone \((f_1 - f_2)\), QDT, which obeys a low-law distortion and the cubic difference tone \((2f_1 - f_2)\), CDT, which obeys cubic-law distortion. Despite the commonalities of their origins, there are considerable differences between the two. The cubic difference tone is the most intense distortion product and is directly observable to the listener even when acoustic stimuli are at relatively low-intensity levels. Because the tone’s frequency \((2f_1 - f_2)\) generally lies close to \( f_1 \), it has seldom been commented on in musical contexts (a significant exception being the Sibelius First Symphony [3]). The quadratic difference tone \((f_1 - f_2)\) requires a higher stimulus intensity to be audible, but because the resultant tone’s frequency generally lies far below the stimulus frequencies and thus is more easily recognized, it has been a topic of musical discourse. As its discovery by Tartini in 1754. Even simple characterizations of the differences between the CDT and QDT are subject to debate and our understanding is constantly being updated by research.

2.2 Studies with Musical Tones

In a study more closely related to musical tones, Pressnitzer and Patterson [12] focused on the combination of combination tones to pitch, especially to the missing fundamental. They utilized a harmonic tone complex instead of the usual pair of pure tones. They employed a series of pure tones between 1.5 and 2.5 kHz with a spacing of 100 Hz (1500 Hz, 1600 Hz, etc. to produce simultaneously 30 pairs of distinct tones) at 100 Hz, 200 Hz, 300 Hz, etc. One consequence was that each adjacent pair of sinusoids in the stimulus tones contributed to the gain of the resulting fundamental. They report: ‘an harmonic complex tone . . . can produce a sizeable DS [distortion spectrum], even at moderate to low sound levels.’ They go on to establish that the level of the fundamental is a summation of contributions from all of the pure tones and that it was essentially “the vector sum of the quadratic distortion tones . . . produced by all possible pairs of primaries.” (This is a good first approximation in which the influence of CDTs is ignored.) Another consequence was that the resultant DS contained multiple harmonics of the fundamental (1700-1500–200 Hz; 1800-1500–300 Hz, etc.). This too were vector sums of the corresponding pairs of pure tones. Employing the same computer technique as Goldstein, they measured the level of each harmonic of the distortion tone and quantified the relatively small amount of inter-subject variability.

To the listener, the QDT could just as well be external sound, albeit external sound with some illusive perceptual properties. For example, QDTs tend to localize much closer to the head than to the location of the loudspeakers, and in this way they expand capabilities for spatial sound. The predictability of QDTs provides a basis for the synthesis of tones that are heard by the listener but which are totally absent from the acoustic stimulus. This predictability also makes it possible to generate acoustic signals that produce complex QDTs matching harmonic target spectra of up to five harmonics.

3. MUSICAL APPLICATIONS

3.1 Background

It is important to note that combination tones are not rarefied, obscure artifacts that only the most trained listeners and psychoacousticians care about. Most musicians are aware of them, and many composers and performers actively welcome them in their work. Furthermore, although the computer musician is technically better positioned to exploit such phenomena, this observation is by no means restricted to electronic music. Yoshi Wada, Matt Ingalls, John Butcher and Tony Conrad are just some of the instrumentalists who describe the deployment of CTs in their live performances. Indeed, from a purely creative standpoint one could argue that the improviser is historically better situated than any other for the exploration of CTs in his performance. CTs are a product of listening, and listening drives improvisation. So where accidents and artifacts can be accepted or rejected, enhanced or attenuated immediately, the opportunity for a subjectively heard music to be developed is greatest, that is, when the performer is free of a score. There are exceptions to this, such as the aforementioned reference to the Sibelius First Symphony [12]. But the example of ear-guided improvisation is emblematic of the overall status of CTs in musical history. When they appear, the tendency is for them to be treated as rarefied, obscure artifacts — rather than as directly controlled musical material.

Kark [9] and Haworth [6] have both described several instances in 20th Century music where this is not the case, and the CT has been treated as a musical material in itself. Maruyama and Jacob Kierkegaard achieve this with the aid of computers, and for accurate control of the CT, the use of a pure tone generator at the very least is essential. Phill Niblock is particularly worthy of note, an artist whose approach to CTs falls squarely between the ear-guided instrumental work of Wada, and the more exacting approach of somebody like Amacher. His work is comprised of dense layers of electronically
treated instrumental drones. He applies microtonal pitch shifts and spectral alterations, which serve to enhance the audibility and predominance of the 'natural' occurring combination tones, as well as introduce new ones. In his 2008 analysis of Niblock’s work, Staedler [14] counted as many as 21 CTs of different frequencies in 3-7 169 (1974) for Cello and Tape.

Drones music is prevalent in the musical history of combination tones, a tendency that is linked to its oft-invoked ‘architectural’ form, which elicits the quality of ‘sound within sound’. A formally static, apparently non-moving combination can reveal a multiplicity of acoustic detail when listened to intently, and CTS are often noticed here. When free to move your head, one can easily hear how head movement changes the intensity and localization of the DPOAE. Were the musical form changing and developing, this kind of comparison would not be possible, and so in many cases CTS may simply not be recognized as something separate from acoustic sound. Niblock’s approach merely magnifies the conditions for the discrimination of CTS. The aforementioned serendipitous quality of CTS – their being ‘happy accidents’ – is therefore subtly effaced, as they are engineered during the editing process.

Like most techniques for creating DPOAEs, Niblock’s approach can be considered to be ‘inside out’ inasmuch as the acoustic sound and manipulates it until the DPOAE is rendered audible. Whether one is playing the violin or digitally pushing partials to within close ratios, the fact remains that the DPOAE as a material entity is fundamentally elusive, only controllable in terms of pitch and loudness. In order to achieve fine-grained control, one needs to reduce the acoustic variables to just those that are necessary to create DPOAEs. In this sense, a DPOAE synthesis would start with the same sound materials as are used in psychoacoustic experiments and hearing tests.

For this reason, the most effective work that has been created using this approach bears comparison to the kinds of examples one hears on psychoacoustic demonstration CD’s. To an extent, this is dictated by the instruments electronic composers have at their disposal: oscillators, clicks and white noise generators are as much the classic sounds of electronic music as they are the stimulus sounds used by hearing science researchers. But one should not underestimate the imagination of the first musical experiments with DPOAEs. Electronic musicians were quick to see the musical possibilities of the evolving notions of auditory non-linearity. For instance, the British Radiophonic Workshop composer, Daphne Oram, devotes two chapters to the consideration of sum and difference tones in her 1972 book, _An Individual Note_ [10]. And some years later these ideas are born into fruition by another artist, Maryanne Amacher, who makes the solicitation of CTS into an arform in its own right. Her sound installations and live performances became notorious for their utilization of aggregated, gated sine waves at very high volumes. Her 1977 article, ‘perceptual geography’ [1], describes an intuitive practice of composing with ‘additional tones’ which resist notation; a bringing to the surface of subliminal sounds often experienced but usually ignored, or taken to be a trick of the senses. After having been introduced to this hearing science literature in the 90s by getting access to the Internet, Amacher revised her earlier article, christening her technique ‘ear tones’ because they are created inside the ear. Amacher’s is the first sound work to elicit a truly separate musical stream from the CTS, a subjective ‘third layer’ (which she referred to as the ‘third ear’) [2]. It is this objectification of the subliminal sounds that is most successful about this work, and the point that needs taking forward.

3.2 Practical Observations

In order for subjective distortion tones to be musically meaningful, the listener must be able to distinguish them from ordinary acoustic sounds (otherwise why not simply use ordinary acoustic sounds). Sustained pure-tone acoustic signals will produce audible sustained, single-frequency distortion tones. This explains why musical use of distortion tones has favored the situation in which the subjective tones form an independent auditory stream or musical sequence that appears to the listener to be independent of the stimulus tones. Changing sequences of acoustic frequencies make difference tone sequences a bit more noticeable musically.

Amacher’s _Head Rhythm/Plaything_ is an effective example of this. A repetitive sequence of crude, pure tone chips elicits a disorientating, subtly shifting rhythm pattern of combination tones at different frequencies. But among the musical properties of the CTS, there are some musical settings by utilizing a series of sinusoidal components (otherwise why not simply use ordinary acoustic signals). Sustained two-pure-tone acoustic signals will produce audible sustained, single-frequency distortion tones. This explains why musical use of distortion tones has favored the situation in which the subjective tones form an independent auditory stream or musical sequence that appears to the listener to be independent of the stimulus tones. Changing sequences of acoustic frequencies make difference tone sequences a bit more noticeable musically.

Amacher’s _Head Rhythm/Plaything_ is an effective example of this. A repetitive sequence of crude, pure tone chips elicits a disorientating, subtly shifting rhythm pattern of combination tones at different frequencies. But among the musical properties of the CTS, there are some musical settings by utilizing a series of sinusoidal components (otherwise why not simply use ordinary acoustic signals). Sustained two-pure-tone acoustic signals will produce audible sustained, single-frequency distortion tones. This explains why musical use of distortion tones has favored the situation in which the subjective tones form an independent auditory stream or musical sequence that appears to the listener to be independent of the stimulus tones. Changing sequences of acoustic frequencies make difference tone sequences a bit more noticeable musically.

Amacher’s _Head Rhythm/Plaything_ is an effective example of this. A repetitive sequence of crude, pure tone chips elicits a disorientating, subtly shifting rhythm pattern of combination tones at different frequencies. But among the musical properties of the CTS, there are some musical settings by utilizing a series of sinusoidal components (otherwise why not simply use ordinary acoustic signals). Sustained two-pure-tone acoustic signals will produce audible sustained, single-frequency distortion tones. This explains why musical use of distortion tones has favored the situation in which the subjective tones form an independent auditory stream or musical sequence that appears to the listener to be independent of the stimulus tones. Changing sequences of acoustic frequencies make difference tone sequences a bit more noticeable musically.

4. Modeling Auditory Distortion as a Non-Linear System

In order to develop a systematic approach to synthesis with DPOAEs, the auditory distortion products can be modeled as products of a general nonlinear system. We start with a classical power series representation [15]:

\[ y = a_1 x + a_2 x^2 + a_3 x^3 + \ldots \]

where x is the input and y the output of the system. The \( a_n \) are constants. The nonlinearity of the output increase as input level of x increases.

4.1. Quadratic Difference Tone (QDT)

The quadratic component, \( a_2 x^2 \), contributes the difference tone, \( f_1 - f_2 \) and also components at \( 2f_1, f_1 + f_2 \) and \( 2f_2 \), although at lower subjective levels. The level of the quadratic distortion tone (as measured by the cancellation method) is given by

\[ L_{QDT} = L_1 + L_2 - C \]

where \( L_1 \) and \( L_2 \) represent the levels of the acoustic signals and \( C \) depends on the relative amplitude of the quadratic distortion, \( C = 130 \text{ dB} \) [4]. Experimental data in which a cancellation tone is used to determine the amplitude of the QDT show a fairly regular behavior. With increasing \( L_1 \) or \( L_2 \), the cancellation level is almost exactly what is predicted and this happens when the difference between the frequencies of the acoustic signals is large or small. (It should be mentioned that there is a percentage of listeners for whom this observation breaks down.) For our purposes, these variances in the effective amplitudes will have a relatively small effect on perceived timbres, especially dynamic ones.

4.2. Cubic Difference Tone (CDT)

The cubic component, \( a_3 x^3 \), contributes the cubic distortion tone, \( 2f_1 - f_2 \) and also \( 2f_1 - f_3 \) etc. The level of the cubic distortion tone is given by

\[ L_{CDT} = 2L_1 + L_2 - C \]

where \( L_1 \) and \( L_2 \) represent the levels of the acoustic signals and \( C \) again depends on the relative amplitude of the cubic distortion tone [4]. Experimental test data do not conform well with what would be predicted for regular cubic distortions because the level of the CDT is very dependent on frequency separation and frequency range, both on frequency separation and frequency range, another reason why CDTs are difficult to use in a controlled way for synthesis.

When multiple pure tones are created at sequential upper harmonics of a fundamental (\( f_1, 2f_1, 3f_1 \), etc.), then the QDT spectrum is a summation of distortion products produced by all pairs of pure tones. For example, it can be shown that a composite signal with a 100 Hz fundamental and with upper harmonics 15 to 25 each at 54 dB SPL produces a fundamental QDT only 15-10 dB lower than the gain of the acoustic signals [12].

Once we have a clearly audible QDT, we find that the harmonic nature of the QDT spectrum is preserved if the QDT is added to other QDTs at levels that the listener can recognize, the acoustic sinusoids have to be at a level that is uncomfortable for most listeners, especially for any extended period of time.

Haworth solved the problem for QDTs in concert settings by utilizing a series of sinusoidal components with constant difference frequencies (akin to the stimuli of [12]). In Correlation Number One, each adjacent pair of sinusoids produces the identical QDT frequency, summing linearly to its total gain and thereby amplifying the level of the subjective tone [6]. Not only did the combination of acoustic sinusoids produce increased gain, but also components that were harmonics of the primary QDT.

The harmonics are created by what can be modeled as intermodulation distortion. If we consider the situation in which there are two sinusoidal inputs to the simple quadratic equation:

\[ y = x^2 \]

we find:

\[ y = a_2 x^2 \cos(\omega_2 t) + a_2 x^2 \cos(\omega_2 t) \]

\[ + a_2 x^2 \cos(\omega_2 t) \]

\[ = \frac{a_2}{2} \frac{2}{2} \frac{2}{2} \cos(2\omega_2 t) = \frac{2}{2} \cos(2\omega_2 t) \]

where \( a_1 \) and \( a_2 \) are the sum and difference frequencies. The complete output signal of the squarer contains DC and components at twice the input frequencies in addition to the sum and difference frequencies.

If multiple pure tones are created at sequential upper harmonics of a fundamental (\( f_1, 2f_1, 3f_1 \), etc.), then the QDT spectrum is a summation of distortion products produced by all pairs of pure tones. For example, it can be shown that a composite signal with a 100 Hz fundamental and with upper harmonics 15 to 25 each at 54 dB SPL produces a fundamental QDT only 15-10 dB lower than the gain of the acoustic signals [12].
amplitude-modulated distortion product. Modulating both gives a fused, amplitude modulated sound. The latter operation is preferable, as it avoids the persistent pure tone, but it ought to be noticed that in the former operation, there is slightly better timbral segregation if only a few acoustic tones are being used. The un-modulated acoustic tone remains fully audible, and the amplitude modulation appears to only affect the distortion product, whereas when both are being simultaneously modulated, the two appear fused, amplitude modulated sound. The general strategy is to use the difference combination tones in the input signal $x(t)$ to reinforce a fundamental frequency on the output signal $y(t)$. In order to do this, the component sinussoids of $x(t)$ need to be harmonic, so that the difference of the frequencies involved always yield a unique frequency or an harmonic of it.

As an example, let’s assume that we want to specify $y(t)$ with a fundamental frequency $f$ and a spectra of five harmonics, with amplitudes $A_1, A_2, A_3, A_4,$ and so on. Let’s denote the amplitudes of the sinussoids of the input signal as $A_1, A_2, A_3, A_4,$ ... Then, if we extend equation (5) and (6) to allow four sinussoidal components, the problem reduces to solving the following system of equations:

$$T_1 = A_1 A_2 + A_2 A_3 + A_3 A_4$$

$$T_2 = A_1 A_3 + A_2 A_4$$

$$T_3 = A_1 A_4$$

$$A_1 = 1$$

This seems to be a fairly simple set of equations, that at a first sight should not present much of a problem, but indeed it is a non-linear system, as it involves solving a second order polynomial. Indeed, the solutions for some of the coefficients come in a pair of conjugates, and are the following:

$$A_1 = 1$$

$$A_2 = 1$$

$$A_3 = \sqrt{(2 - 3T_2 - T_3 + 4T_4)}$$

$$A_4 = T_3 - 2T_4$$

If we extend the problem to allow for five sinussoidal components, we have the system:

$$T_1 = A_1 A_2 + A_2 A_3 + A_3 A_4 + A_4 A_5$$

$$T_2 = A_1 A_3 + A_2 A_4 + A_3 A_5$$

$$T_3 = A_1 A_4 + A_2 A_5$$

$$T_4 = A_1 A_5$$

$$A_1 = 1$$

In this particular case, the solutions for each amplitude $A_i$ consist on one real solution and a pair of complex conjugate solutions, as solving the system of equations in (9) involve finding the root of a polynomial of order $3$. The solutions are so long and intricate that it is impossible to include them within this paper. If the number of harmonics is increased to six, the network of equations is impossible to solve analytically, only numerically. We have found that the method of calculating the $A_i$ coefficients by direct solution is possible for up to five harmonics in the DPOAE.

6. CONCLUSIONS

At this point we have been able to generate DPOAEs that are clearly audible in concert settings. We have been able to apply the common time-domain processes of AM and FM. And, we have been able to control timbre by manipulating up to 5 harmonics. We intend to extend our control of timbre by utilizing other mathematical methods and by applying perceptual criteria as a way of simplifying the complexity of the computation. This may be particularly important in order to support real-time applications in which coefficients are calculated on the fly.

Among our accomplishments is the creation of DPOAEs based on sinusoidal analysis of recorded sounds, including acoustic instruments and singing. The subjective impression produced by these sounds extends the range of DPOAE possibilities well past those typical of psychoacoustic experiments and into a domain where DPOAEs share the dynamic properties of traditional acoustic sound, albeit within the practical limitations imposed by DPOAE synthesis. Then too, among the most attractive perceptual attributes of DPOAEs is the illusory quality of their localization, which places the phantom sources closer to the listener than to the loudspeakers. We intend to report on this and others extensions of DPOAE techniques in the near future.

7. ACKNOWLEDGEMENTS

This research was partially funded by Fondecyt Grant 11109193, Conicyt, Government of Chile.

8. REFERENCES


CONTROLLING DYNAMIC STOCHASTIC SYNTHESIS WITH AN AUDIO SIGNAL

Gordan Kreković
Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia
gordan.krekovic@fer.hr

Igor Brikči
Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia
igor.brikci@fer.hr

ABSTRACT

Dynamic stochastic synthesis is a sound synthesis technique devised by Iannis Xenakis. This technique produces music by manipulating a set of breakpoints. At every repetition of the waveform, the breakpoints change their positions accordingly to random walks. In all the previous implementations of this method, synthesis parameters were either fixed or controllable through a graphical user interface. Our intention was to hide most of the parameters from musicians and let them use an input audio signal to intuitively control the synthesis process. We proposed and implemented a novel solution based on a mapping between audio features of an input signal and the synthesis parameters. The amplitude random walk is limited by the minimum and maximum amplitudes of the input signal and parameterized by the spectral centroid for noisy signals. It is possible to change the interpolation method and the number of breakpoints in a waveform using the graphical user interface. The solution was developed as a real-time system and demonstrated to professional composers. They confirmed that the realization is suitable for practical tasks, especially for live performances.

1. INTRODUCTION

Motivated by the desire to create music with complex sonorities, Iannis Xenakis began his research on stochastic synthesis in the late 1960s. By that time, this composer, music-theorist and architect-engineer had already experimented with stochastic processes for building musical forms and choosing note attributes. Xenakis perceived an opportunity to apply the same approach to the microstructural level and use it for sound synthesis [10]. His early concrete ideas about stochastic synthesis were published in the book Formalized Music in 1971. After several years of research he developed a new technique called dynamic stochastic synthesis (DSS). This technique devised by Ianis Xenakis. This technique produces music by manipulating a set of breakpoints. At every repetition of the waveform, the breakpoints change their positions accordingly to random walks. In all the previous implementations of this method, synthesis parameters were either fixed or controllable through a graphical user interface. Our intention was to hide most of the parameters from musicians and let them use an input audio signal to intuitively control the synthesis process. We proposed and implemented a novel solution based on a mapping between audio features of an input signal and the synthesis parameters. The amplitude random walk is limited by the minimum and maximum amplitudes of the input signal and parameterized by the spectral centroid for noisy signals. It is possible to change the interpolation method and the number of breakpoints in a waveform using the graphical user interface. The solution was developed as a real-time system and demonstrated to professional composers. They confirmed that the realization is suitable for practical tasks, especially for live performances.

Such organic sound quality is the result of the constant waveform evolution. However, the evolution is non-deterministic, so it is not possible to predict or control the generated sound. The synthesis parameters define only restrictions within which the waveform non-deterministically changes, but do not specify the concrete pitch, amplitude and timbral characteristics of the sound. Moreover, the first design of the dynamic stochastic synthesizer did not even support changing the parameters during the synthesis process. These problems were addressed by several authors who proposed different interface designs for parameter automation and real-time control [1, 3, 6]. Rich graphical user interfaces, keyboard shortcuts, and MIDI control were implemented to help in the creative process. However, musicians still had to be aware of synthesis parameters and understand their technical meaning.

We believe that more intuitive parameter manipulation could provide a more direct way from ideas to their realization. Musicians would spend less time on non-musical tasks such as setting parameters and learning how they affect the synthesis process. In this paper we proposed a novel approach to controlling parameters of the DDS using an input audio signal. This way, a musician does not need to think about numerical values, but can play a musical instrument, synthesizer, or even use his own voice to control the synthesis process.

The idea of employing audio signals to control processes is already represented in adaptive digital audio effects and audio driven synthesis [7, 8]. Unlike the existing approaches, our algorithm does not include an input signal in the synthesis process, but only uses its audio features to render the needed parameters. The goal was to define a mapping from an input signal to synthesis parameters that is the most intuitive, but still maintains flexibility of the DDS process. We designed an algorithm to fulfill these requirements and implemented it as a real-time system which is ready to be used by composers and live performers.

2. DYNAMIC STOCHASTIC SYNTHESIS

A waveform generated by the DSS process is defined by a set of breakpoints. Each breakpoint has its amplitude and the position in the time domain which is relative to the preceding breakpoint in the set. Initial values of amplitudes and positions are usually chosen randomly or taken from a trigonometric function. At every repetition of the waveform, the values are varied stochastically using random walks as illustrated by Figure 1. A particular breakpoint changes its amplitude and position independently of the other breakpoints.

3. CONTROL OF SYNTHESIS PARAMETERS

The DSS process can be parameterized with the following parameters:

1. number of breakpoints in a waveform,
2. barriers of the amplitude random walk,
3. probability distribution of the amplitude random walk and its parameters,
4. barriers of the duration random walk,
5. probability distribution of the duration random walk and its parameters,
6. type of interpolation between breakpoints.

To use the full potential of this synthesis technique, a musician should understand the system. We wanted to overcome this problem by making an intuitive and inspiring interface. Using an arbitrary input audio signal for controlling the synthesis process, a musician can easily achieve desired results expressing his ideas in the musical domain. Such approach is straightforward and opens numerous possibilities for experimentation with different audio sources.

Our goal was to establish a natural relation between input sounds and parameters of the DDS. For each synthesis parameter we searched for the most appropriate audio feature which could be extracted from an input signal and used for parameter control.

3.1. Feature extraction

Extraction of local audio features from the input signal is based on time framing. At each iteration, our algorithm calculates a set of features from a fixed-size time frame and uses that to set synthesis one cycle of the resulting sound. For the step size between two successive input frames the algorithm takes a length of the synthesized cycle. The step size will never exceed the frame size because of the restrictions on the waveform frequency (equations 1 and 2).

3.2. Amplitude random walk

The amplitude of the synthesized sound is expected to follow the amplitude of the input signal. For that reason, we used the peak-to-peak value of an input frame to control the range of the amplitude random walk. This is the most straightforward mapping, because our algorithm simply takes the minimum and maximum values of an input frame and sets them as amplitude barriers.

In contrast, selecting a probability distribution for the amplitude random walk cannot be done directly from features of an input signal. Forcing any kind of unnatural mapping would make the system less intuitive for musicians. On the other hand, limiting the synthesis process just to one probability distribution would be ignorant, because Xenakis and other researchers expended considerable effort exploring
is defined with the following equation:

\[ H_i = \frac{1}{\log(N)} \sum_{j=1}^{N} z_i \log z_i , \]

where \( i \) is a frequency index and \( z_i \) is a normalized spectral density:

\[ z_i = \frac{S(o)}{\sum_{j} S(o)} \]

Here \( S(o) \) represents the magnitude of the discrete Fourier transform with the transform size \( N \).

3.3. Duration random walk

As explained earlier, the barriers of the duration random walk can be calculated directly from the desired frequency limits. Those are expected to follow the changes of the input signal, so our task was to establish mapping between a certain audio feature of the input signal and the frequency limits.

For harmonic signals, the most appropriate feature for such mapping is fundamental frequency. However, if a musician uses a technique like noise morphing to control the DSS process, much better results would be obtained with spectral centroid. This measure indicates the position of the gravity center of a frequency spectrum. It appears in automatic sound classification, because it is perceptually related to the impression of timbral brightness. In our algorithm spectral centroid is used to calculate the frequency limits when the input signal is non-periodic.

The mapping between the audio features of an input signal and the barriers of the duration random walk is described by the following steps:

1. calculate the fundamental frequency \( f_o \) and the total spectral energy \( E_i \) of the current input block
2. calculate the fundamental frequency was detected:
   2.1. assign the value of \( f_o \) to \( f_{max} \) and \( f_{min} \)
   2.2. if the fundamental frequency was not detected:
     2.3. decrease \( f_{max} \) until there is at least 10% of the total spectral energy between \( f_{o} \) and \( f_{max} \)
     2.4. decrease \( f_{min} \) until there is at least 10% of the total spectral energy between \( f_{o} \) and \( f_{min} \)
3. if the fundamental frequency was not detected:
   3.1. calculate the spectral centroid \( f_c \) of the input block
   3.2. assign the value of \( f_c \) to \( f_{max} \) and \( f_{min} \)
   3.3. decrease \( f_{max} \) until there is at least 30% of the total spectral energy between \( f_{c} \) and \( f_{max} \)
   4. calculate the barriers of the duration random walk from frequency limits \( f_{max} \) and \( f_{min} \) using Equations 1 and 2.

To detect the fundamental frequency the system uses the modified autocorrelation function method [5]. If there is no peak higher than threshold in the normalized autocorrelation function, the algorithm returns zero. In such cases we assume that the input signal does not show significant periodicity and switch to spectral centroid as explained in the step 3. The frequency limits are selected to be wider for the input sounds with more complex spectrum.

For the duration random walk, the algorithm uses the same distribution as for the amplitude random walk. The parameters are also calculated from the spectral entropy of the input signal, but they are scaled with different experimentally obtained coefficients \( \lambda \).

3.4. Other parameters

The number of breakpoints in a waveform affects the richness of the synthesized sound. We decided to let musicians control this parameter through the graphical interface, so that they can experiment with different numbers of breakpoints without changing the input audio signal.

Motivated by improvements of the original DSS proposed by other researchers, we included additional interpolation methods for further enhancement of timbral variety. Along the linear interpolation for linking breakpoints, we implemented zero-order and quadratic interpolations. A musician can choose among these methods through the graphical interface.

4. IMPLEMENTATION

The application was developed in C++ using Qt toolkit. For handling audio data we implemented support for JACK Audio Connection Kit [4] which enables easy connection with other audio processing software. Such software can be used for pre-processing the input signal, adding effects to the synthesized signal, or recording the output. For calculating fast Fourier transforms we employed KissFFT library [2]. Thanks to the selected set of technologies, our application is fully platform independent.

5. EXPERIMENTS AND DISCUSSIONS

We tested the application with different types of input signals. Figure 2 shows an example of a synthesized sound controlled by a speech signal.

![Figure 2: Spectrograms of an example input signal (upper) and the synthesized signal (lower). In this case we used the linear interpolation and 5 breakpoints in a waveform.](image)

To ascertain the efficiency of our tool in practical tasks of musical composition and performance, we presented it to four professional composers involved in computer music. They tested the application using an acoustic piano and their voices as sound sources. The composers agreed that the approach is interesting and straightforward. They expected more elements on the graphical interface, because their practice to manipulate parameters explicitly. However, all of them noticed the benefits of direct control from input audio, especially for live performances.

In practical cases they would use our system along with other means of sound synthesis or acoustic instruments, because the DSS is inadequate as the only synthesis technique for longer compositions.

6. CONCLUSION

The possibility to control the dynamic stochastic synthesis using an input audio opens many opportunities for computer musicians. This novel approach is intended to keep the creative process in the musical domain and let musicians express their ideas easily. During the research, we have extensively experimented with different mappings from an input audio to the synthesis parameters. We believe that we have found the most intuitive way to control the DDS using an input audio signal.

7. REFERENCES

Sound morphing figures prominently as one of the most interesting sound transformation techniques due to its enormous creative potential. Most authors pose the problem of morphing sounds using perceptual requirements, but hardly ever evaluate their results mainly because perceptual evaluations are cumbersome and costly, and there are no standard objective evaluation criteria established for sound morphing. In this work we propose a formal evaluation procedure for sound morphing algorithms following three criteria, namely correspondence, intermediateness, and smoothness. The adoption of the proposed evaluation framework will help formalize the results towards more perceptually relevant morphed sounds.

1. INTRODUCTION

Sound morphing encompasses several models and techniques whose common goal is to obtain gradual transformations between sounds. Many sound transformations are called morphing in the literature [11], ranging from music compositions [10, 8] to the design of sound synthesizations between sounds. Many sound transformations are techniques whose common goal is to obtain gradual transformations between sounds. If we interpolate the parameters of representation. This assumes that there exists a sequence of intermediate sounds that will be perceived as a gradual transition between source and target. However, the perceptual impact of the interpolation of parameters depends largely on what information the parameters represent and how the sound material is perceived. On the one hand, if the parameters encode perceptually irrelevant information, the result of the interpolation of these parameters will very likely have little perceptual significance. On the other hand, if sound perception is categorical, we would try to achieve an impossible perceptually continuous transformation.

Morphing can be viewed as a transformation that involves hybridization to obtain intermediate form (among other features) [12]. The term hybridization is applied in many areas generally to refer to a process that involves the combination of two (or more) objects, individuals, variables, etc., depending on what is being considered. A first important aspect of the problem of hybridization is to understand that there are several possible ways of combining two things. We are going to consider two simplified hybridization processes, one commonly found in nature and the other one usually only accessible to artificial means, which we call morphing.

2. MORPHING SOUNDS

The aim of sound morphing is to obtain results that are perceptually intermediate between source and target. The classic morphing technique is based on the interpolation principle, which supposes that we should obtain a gradual transition between the sounds if we interpolate the parameters of representation. This assumes that there exists a sequence of intermediate sounds that will be perceived as a gradual transition between source and target.

Morphing can be viewed as a transformation that involves hybridization to obtain intermediate form (among other features) [12]. The term hybridization is applied in many areas generally to refer to a process that involves the combination of two (or more) objects, individuals, variables, etc., depending on what is being considered. A first important aspect of the problem of hybridization is to understand that there are several possible ways of combining two things. We are going to consider two simplified hybridization processes, one commonly found in nature and the other one usually only accessible to artificial means, which we call morphing.

2.1. Hybridization and Morphing

The hybridization process called sexual reproduction is ubiquitous in nature. In general terms, when a couple has children, we can usually easily recognize who the parents are because of physiological similarities. In other words, the kids take after their parents, and we usually say that they might have their mother’s nose, the father’s eyes, etc. This is a specific case of hybridization where the hybrid individual (the child) consists of a combination of parts from either parent, illustrated in figure 1.4.

On the right, figure 1b depicts the process called morphing, where each constituent parent is now a combination of the corresponding parts from the parents. Therefore, each part is intermediate in shape (among other features). Figure 2a illustrates that sometimes there is more than one possible way of defining intermediate (or morphed) shapes. What is the intermediate shape between the circle and the square? When we are only considering shape, both transformations shown in figure 2a intuitively satisfy the requirement of a gradual transformation. The “best” or “most appropriate” transformation is application-dependent when we use objective criteria to evaluate the transformation or user-dependent when we use subjective criteria, that is, a user’s personal taste or aesthetics.

3. EVALUATION CRITERIA

3.1. Correspondence

Morphing requires a description of the entities being morphed (shapes, images, sounds, etc.) followed by establishing the correspondence between these descriptions, as depicted in figure 2b. The morph is achieved by a description whose elements are intermediate. Elements without correspondence in the description render the transformation complicated. If we are morphing faces and one of them has a mole, we will have to decide how we are going to represent intermediate versions of the unmatched feature. One of the consequences of the lack of correspondence between the objects being morphed is that we can have multiple possible transformations depending on how we decide to deal with the free feature. When morphing sounds, most works address the correspondence between model parameters [14, 11].

3.2. Intermediateness

The morphed objects should be perceived as intermediate. For example, when transforming a square and a circle we want to avoid transforming the square into another recognizable shape first (say, a triangle) that is not perceived as intermediate between the square and the circle and then finishing the transformation from this shape into the circle. Intuitively, when transforming between a child’s and a man’s face, we expect all the hybrids to be human faces because it would be counterintuitive otherwise. Conceptually, the transformation should be the face of a person getting older.

Figure 3a illustrates the requirement of intermediateness. Points that are connected by a line are intermediate between the segment AC connecting the points, such that the distances from A to B plus the distance from B to C be equal to the distance from A to C. Notice that in figure 3a point D lies at the same distance from points A and C, yet, it is not intermediate between them.

In practice, when morphing shapes, images, and sounds, we make use of the interpolation principle as a convex combination [3, 4] of the parameters to achieve intermediateness. A convex combination leads to intermediate representations in the space of parameters. We argue that
3.3.3. Conceptual Distance
Let us suppose for a moment that the perception of faces is continuous. If we establish correspondence between the faces and interpolate parameters of a perceptually relevant representation (model), we should expect the morph to be smooth. However, some examples of face morphs will lead to artificial hybrids simply because the faces are conceptually very far from each other. Figure 4 illustrates that the morph between a man’s and a cat’s face looks less natural than one between two human faces because of the conceptual distance between them. As a general rule of thumb, the naturalness of the morph is inversely proportional to the conceptual distance between the two. The farther apart the objects are in the conceptual space, the more challenging it is to obtain convincing morphs.

4. EVALUATION PROCEDURE

Figure 5 depicts a flowchart with subjective and objective evaluation procedures for morphing. When the purpose of morphing is artistic, the evaluation of the results is usually subjective. Each individual will evaluate the results according to their own aesthetic criteria. When we want to obtain morphed results for technical applications, such as sound synthesizers or to study perceptual aspects of morphing (such as continuous timbre spaces), we need to establish an objective way of measuring the quality of the results. In figure 5 we see the three criteria proposed, namely correspondence, intermediateness, and smoothness. The objective evaluation procedure can use perceptual (listening tests) and automatic (feature values) means.

First of all, we must ensure correspondence between the representations of the sounds being morphed. Tellman [14] is among the first to investigate correspondence in sound morphing. The model used to represent the sounds plays a crucial role in this step [4, 3] because model parameter correspondence does not necessarily guarantee correspondence of perceptual features of sounds. Osaka [11] examines the problem of matching partials when interpolating the parameters of a sinusoidal model. Caetano [4], in turn, proposes a source-filter model to represent musical instrument sounds that guarantees temporal and spectral correspondence. More specifically, correspondence between the frames of the sounds and between spectral envelope parameters for each frame.

The perceptual intermediateness of the morph depends largely on how perceptually relevant the representation is. When interpolating the parameters of physical models, Hikichi [9] recognizes that the linear interpolation of the parameters does not lead to perceptually linear morphed sounds. So they propose to construct MDS spaces using the source, target and morphed sounds to study how to warp the interpolation factor to obtain perceptually linear morphed sounds. Naturally, this approach renders the results very difficult to obtain and to evaluate. Also, the warping function is model dependent and probably user dependent too.

Finally, it is important to investigate the smoothness of the transition. Caetano [4, 3] proposes to investigate how accurately the morphing factor α controls the morph guided by perceptually salient features such as spectral centroid and attack time. Caetano varied the morphing factor α linearly and studied which of several representations leads to linear variation of the feature values.

5. CONCLUSIONS AND FUTURE PERSPECTIVES

Most works about sound morphing skip the evaluation of the results because perceptual evaluations are cumbersome and expensive and there is no standard objective evaluation criteria or procedure when morphing. In this work, we proposed correspondence, intermediateness, and smoothness as evaluation criteria. We discussed each criterion in turn, giving examples to illustrate their importance. Then we presented an objective evaluation framework using the proposed criteria, along with perceptual assessments and quantitative measures. Future perspectives of this work include the development of a standard evaluation procedure for sound morphing that defines the perceptual tests and the feature values to investigate the correspondence, intermediateness, and smoothness of sound morphing algorithms.

6. ACKNOWLEDGEMENTS

This work is funded by the Marie Curie IAPP “AVID-MODEL” grant within the European Commissions FP7 and was partially carried out at IRCAM.

7. REFERENCES


REAL-TIME SIMULATION OF PREHISPANIC ANTARAS FROM SOUTHAMERICA

Patricio de la Cuadra
Pontificia Universidad Católica de Chile
Centro de Investigación en Tecnologías de Audio

Benoit Fabre
Université Pierre et Marie Curie
Institut Jean Le Rond d’Alembert

José Pérez de Arce
Museo Chileno de Arte Precolombino

Arnaud Gérard
Universidad Mayor de San Andrés
Instituto de Investigaciones Físicas

ABSTRACT

Pre-Hispanic instruments from South America are of great interest to understand the sound landscape from the cultures around the Andes. In this paper an Antara from the Aconcagua culture is measured and modeled using a one-dimensional waveguide model. Particular interest is devoted to the details of the so-called complex resonator which is characteristic of many instruments found in the region. The input impedance is obtained from the parametrized model and fitted to a waveguide simulation of the instrument. The model is implemented in a real-time platform which will allow performing these ancient instruments no longer available for playing.

1. INTRODUCTION

The sound landscape in America before the Spanish conquest was very different from today’s. Not only in the instruments used but also in the meaning and purpose of music. A great deal of attention was devoted to the timbre of the sound and its loudness, without giving much importance to the idea of a scale with accurate pitch. Sound effects like beatings were appreciated for its strong emotional power and music was conceived as a communitarian rite. Most important instruments included winds and percussion [1].

The Andean’s sound has survived the cultural invasion in several places throughout Latin America, such as Bolivia, Peru and central Chile where in the ritual Bailes de chinos is possible to observe a sound power very contrasting with the European music [5, 10]. Among wind instruments there was a variety of panpipes built with different materials. Some stone panpipe like flutes, called Antaras, have been found in excavations together with the sepulture of their owners [6]. It is believed that in some cases the instruments were muted by opening holes in the resonator when the owner died. Thus, the article’s main goal is to create a real-time simulation of that particular instrument and to describe a methodology to reproduce resonators of complex bores based on input impedance fitting.

The instrument chosen for this study (fig.1) belongs to the Diaguita culture (900 until the Spanish invasion 1536), located in where currently is central Chile. It was found in a cemetery beside his owner’s head, who was facing upwards in contrast with all other corpses that were facing downwards. It was handcrafted in stone with an extraordinary precise technique; the inner walls width is about 0.7 mm, close to the material resistance limit. The instrument is in very good condition although one of its tubes was muted by opening holes in the sides. The resonator is build to produce a complex resonator defined, in a one-dimensional plane acoustic wave description as:

\[ r_k = \frac{S_1 - S_3}{S_1 + S_3} \]  

where \( S_1 \) and \( S_3 \) are the transversal section of the two cylinders as observed in fig.3.

2. ACOUSTICS OF THE RESONATOR

The resonator is built to produce a raijado sound, a very loud, noisy and rich-in-harmonics sound that often oscillates at low frequency producing effect similar to an exaggerated vibrato or flutter-tongue. To achieve this sound, suitable for the timbre complexity searched by the Andean cultures, the resonator cannot be simply cylindrical. Often it is made out of two cylinder of different diameters and approximately the same length, as shown in fig.3. This type of resonator is called complex resonator [1]

![Figure 2](image-url)  
Figure 2. Scanned image showing the internal shape of the tubes. One tube is muted by opening holes in the sides. Tube 4 to 1 from left to right.

![Figure 3](image-url)  
Figure 3. Complex resonator scheme. The left end is where the player blows, the right end is closed. Its acoustical impedance has been analytically modeled in [1]. Different from both open and closed cylinders, the properly built complex resonator shows admittance peaks at frequencies \( f_0, 2f_0, 4f_0, 5f_0, 7f_0, 8f_0, \ldots \) as can be observed in fig.7. The position of the resonances is very sensitive to the relation of lengths and surfaces of the two cylinders, and it is impressive to observe that the four tubes of the antara are precisely carved to produce the desired sound raijado in all of them.

3. INSTRUMENT MEASUREMENTS

The instrument was scanned by a computer tomography scanner, SOMATOM Emotion, providing high quality images from the inside bore, as shown in fig.2. The images were calibrated and the relevant dimensions extracted from them as detailed in table 1.

<table>
<thead>
<tr>
<th>Tube</th>
<th>L1 [mm]</th>
<th>L2 [mm]</th>
<th>S1 [mm²]</th>
<th>S2 [mm²]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>87.8</td>
<td>104.4</td>
<td>78.5</td>
<td>19.63</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>115.1</td>
<td>78.5</td>
<td>19.63</td>
</tr>
<tr>
<td>3</td>
<td>110.3</td>
<td>124</td>
<td>78.5</td>
<td>19.63</td>
</tr>
<tr>
<td>4</td>
<td>120.6</td>
<td>138.6</td>
<td>78.5</td>
<td>19.61</td>
</tr>
</tbody>
</table>

Table 1. Dimensions for the four tubes, Tube 1 being the shorter.

4. SIMULATION MODEL

A one-dimensional waveguide simulation of the complex resonator has been implemented using lumped elements (fig.4). The cylinders are connected through a two-port junction as modeled in [8]. Using an optimized version with only one-multiplication, fig.5 shows its implementation. Where \( r_k \) is the reflection coefficient of the junction defined, in a one-dimensional plane acoustic wave description as:

![Figure 4](image-url)  
Figure 4. Waveguide model for a complex resonator.

\[ r_k = \frac{S_1 - S_3}{S_1 + S_3} \]  

where \( S_1 \) and \( S_3 \) are the transversal section of the two cylinders as observed in fig.3.

![Figure 5](image-url)  
Figure 5. One-multiplier junction model for pressure signal [8].
The losses block represents a filter that captures the visco-thermal losses inside the resonator, while the Reflection filter models the frequency dependent reflection of the pressure wave at the open end of the bore.

The complete model has been programmed using FAUST [4] as development platform due to its flexibility, modularity and the possibility of having a visual representation of the code. Fig.6 shows the visual display for the central part of the resonator.

5. FITTING THE MODEL

The resonator simulation can be seen as a black box with only a few parameters to adjust. These parameters are the lengths of the upper cylinder $L_1$ and the poles and zeros for the losses and reflection filter. $L_1$ and the reflection filter have to be adjusted in order to reflect the effect of the performer’s lips acting at the open end. Reducing its size and therefore affecting the radiation and reflection filter, and altering the effective length of $L_1$ due to the impedance variation. The losses filter needs to be adjust to reflect the irregularities of the inner walls of the bore.

By adjusting these parameters we can heuristically approximate the acoustic impedance read from the model to the one obtained by the analytical model.

To obtain the acoustic impedance from the simulating model, the input is assumed to be a volume velocity impulse [7]. Thus, the input impedance coincides with the acoustic pressure $p(t)$ read from the simulation, but scaled by the characteristic impedance. The time signal is converted into frequency-domain to be compared with the impedance from the analytical model. Parameters are changed and the process is recomputed in an iterative loop until a good similarity is reached.

The comparison between the simulation and the analytical model is shown in fig.8. We observe a good matching for the resonant frequencies. The amplitude of the higher partials are smaller, mainly due to the effect of the filters.

6. CONCLUSIONS AND FUTURE WORK

An efficient model of the complex-tube antaras has been implemented. The results obtained by comparing input impedances are considered correct, but for a final tuning of the parameters it would be desirable to have an input impedance measurement from the real instrument to compare with.

It would be interesting to develop an intelligent system that adaptively moves the poles and zeros from the filter searching for the values producing a close match between model and measurements.

The current model coupled with the appropriate excitation will allow us to recover the sound of an interesting group of instruments that, due to damage or preservation, are not available for playing and will allow us to experiment with parameters changes that could help us understand the details of the complex resonators and their especial sound.

7. ACKNOWLEDGEMENTS

The authors would like to acknowledge the support from the “Fondo para el Fomento de la Música Nacional, 2012” and FONDECYT No.11090142. The Museo Arqueológicos de La Serena was very kind in letting us make measurements and recording with their instruments. Finally we would like to thank Jimmy Campilay for the scanned images of the antaras.

8. REFERENCES

TIME/PITCH MODIFICATION USING NARROWBAND AM-FM SIGNALS

Naotoishi Osaka
Atsushi Mita
Tokyo Denki University
School of Science and Technology for Future Life
Tokyo, Japan
osaka@im.dendai.ac.jp, mita@stil.im.dendai.ac.jp

ABSTRACT

Time-scale modification and pitch modification are two of the most fundamental effects in telecommunications and music information processing. The phase vocoder provides a good framework for such modifications. However, it has long been known that the technique introduces an undesired degrading factor known as “phasiness”. To remove this phenomenon we propose a new phase vocoder, which adopts an FIR narrowband filter bank for analysis, and an AM-FM signal for the synthesis of each band. It is applied to time-scale/pitch conversion, vibrato extraction and addition for reading and singing voices. The impairment was removed and consequently the modified sound quality improved.

1. INTRODUCTION

The phase vocoder was originally proposed by Flanagan [1]. The waveform of a signal is divided into several channels using a filter bank. Each band signal is represented in terms of instantaneous amplitude and phase derivative, instantaneous frequency. The phase vocoder was first expressed digitally by Portnoff using the STFT [2]. In reference [1], a narrow-bandpass filtered and windowed signal is defined in terms of the STFT, so resynthesis and its modification becomes realistic applications, particularly time/frequency modification, i.e. temporal stretching/compression and pitch shifting.

Although perfect resynthesis can be achieved theoretically, there is a drawback when applied to temporal modification known as phasiness, which is undesired reverberation added to the signal. This is produced because energy is scattered in multiple channels using a filter bank. Each band-limited signal is defined in terms of the phase derivative, instantaneous frequency. The phaseness produced because energy is scattered in multiple channels is expressed as

Mathematical method is that the phase function for each band is a monotonically increasing function. This prevents the derivative of the phase function with respect to time, i.e., instantaneous frequency, from becoming a negative value that is meaningless in terms of auditory perception. By remaining a positive integer, the values correspond to instantaneous pitch. This makes interpretation of the digital manipulation introduced here easy to understand.

A block diagram of the method is shown in Fig.1. In this section we introduce the analysis and synthesis procedure of the proposed method in greater detail.

2. ANALYSIS/SYNTHESIS SYSTEM

We propose a new technique: a new phase vocoder based on a different framework. The characteristics are:

1. An FIR filter is adopted to make a narrowband filter bank.
2. In analysis, instantaneous amplitude (IA) and instantaneous frequency (IF) are derived using the Hilbert transform, and modification is basically done on these parameters.
3. In resynthesis, a sinusoidal representation using IAs and IFs is done for each band, and assembled, and the overlap-add method is used for frame-to-frame concatenation.

In Fig.1, a very steep slope in the frequency-amplitude characteristics is necessary so that there is hardly any effect on the neighboring bands for both sides of the band of interest. This prevents an overlapped representation among multiple bands for a single sinusoid.

In items 2 and 3, by adopting a narrow-enough channel, we can acquire a signal in each band such that if the signal is expressed in terms of time-variant amplitude and frequency, it corresponds to auditory perception. A mathematical condition is that the phase function for each band is a monotonic decreasing function. This prevents the derivative of the phase function with respect to time, i.e., instantaneous frequency, from becoming a negative value that is meaningless in terms of auditory perception. By remaining a positive integer, the values correspond to instantaneous pitch. This makes interpretation of the digital manipulation introduced here easy to understand.

A block diagram of the method is shown in Fig.1. In this section we introduce the analysis and synthesis procedure of the proposed method in greater detail.

2.1. Analysis

First the input signal is divided into band-limited signals using an FIR perfect reconstruction filterbank. The impulse response of k-th filter h(n) at discrete time n is shown in equation 1, where K is the number of frequency bands, and s(n) is a window function. Its direct interpretation is a low-pass filter (LPF) and a band-lower LPF which cancels passed signal except a band in problem. By tedious algebra it is converted to and interpreted as a band-shifted filter from an ideal LPF. A Blackman window is used with 2047 filter taps. The bandwidth ranges from about 100 to 400 Hz.

Filter computation is done fast by OLS (Overlap-and-Save), which multiplies filter transfer function and spectrum, derived from impulse response and a signal by FFT, respectively, and get a circular convolution by computing an inverse FFT. From the discussion in the previous section, the band-limited signal can be represented as a sinusoidal function with IAs and IFs. This is equivalent to saying the signal is an AM-FM signal. The components of AM and FM are IA and IF, respectively. First, we compute an analytical signal from the input signal x(n) at digital time n using the Hilbert transform Ψ(x)(n).

\[ A(n) = \sqrt{2 + x^2(n)} \]

\[ \theta(n) = \arctan \frac{x(n)}{\sqrt{2 + x^2(n)}} \]

An IF, Φ(n) is a derivative of a phase \( \theta(n) \) with respect to continuous time t. This procedure is run for all the bands.

2.2. Synthesis

The Band-limited signal is reconstructed, as well as sinusoidal model, by

\[ y_{kn}(n) = A_{kn}(n) \cos \theta_{kn}(n) \]

where \( A_{kn}(n) \) is an IA of the k-th channel and \( \theta_{kn}(n) \) is an integral of an IF of the k-th channel at time n. This procedure is done for each channel.

3. TIME SCALE/PITCH MODIFICATION

A time-scale/pitch modification is possible by modifying IAs and IFs before the synthesis procedure. In this section we introduce a time-scale transformation, pitch transformation, vibrato elimination, and vibrato addition.

3.1. Time-Scale Modification

A time-scale modification is achieved by interpolating IAs and IFs. First we substitute the time axis with the new transformed axis. That is, if time is stretched to 1.47 times the original length, the time at 1 second is substituted with 1.47 seconds. Then linear interpolation is done for both parameters. Theoretically, this is not a correct interpolation. However, the rate of change of these parameters is much lower than that of a signal wave, and this procedure gives a good approximation. The example is shown in Fig. 2.

Then the initial phase value in the current phase function is computed from the phase value in previous synthesis frame. However, only the stationary part is modified, and transient portions such as unvoiced sounds and onsets of sounds are not modified to obtain more natural sounds. The decision procedure for unvoiced sounds and onsets uses threshold values, α and β, and are defined as equations (6) and (7), respectively.

\[ \alpha = \sum \frac{X(j)}{\tilde{X}} \]

\[ \beta = \sum \left[ p_i(i + 1) - p_i(i) \right] \]

(6)

(7)

Where \( X(j) \) is the power spectrum of the input signal \( x(n) \) acquired from FFT, \( \tilde{X} \) is its maximum value, and \( J \) is the number of FFT bins. \( p_i(i) \) is a local peak or minimum value sequences for \( i=0,1,\ldots,J-1 \), where I is the number of local values, detected from \( f_s(n) \), the instantaneous frequency at time \( n \) of the k-th channel within an analysis frame, and \( \alpha \) and \( \beta \) are judged as an unvoiced or an onset of speech, respectively, if they exceed thresholds. Time stretching is executed for the intervals where both of these parameters are under thresholds.

Otherwise the signal in the current frame is identified as either unvoiced or an onset of speech and remain unchanged since such a transitive part impairs the quality if modified temporally.
3.2. Pitch Modification

A pitch modification is achieved by multiplying conversion coefficients to IFs. Acquired IFs consequently decide the new bands where the modified signal belongs. In order not to alter the spectral envelope from the original sound, the IA is modified so that it becomes closer to that of the original sound for the new channel. This is to preserve the spectral envelope, and as a result a timbre the same as the original one is obtained. For this purpose, the square mean values of each channel are derived and IA is multiplied by their ratio using equation 8.

\[ R_{ns} = \text{frame shift, 0.2s. We define } \theta = \frac{\text{width of frame shift}}{20 \text{ ms}}, \text{ and estimating vibrato is achieved as follows:} \]

1. Estimate the DC component \( d_{k,n} \) of \( f_{k}(n) \), and find their difference for each 0.4 second frame \( l \). This makes a residual signal, \( r_{k} (n) \).
2. Compute spectrum of \( r_{k} (n) \) and find the max value \( \tilde{\lambda}_{k} \) and its frequency \( f_{k} \). Note that this is not the spectrum of a signal, but that of an IF, and discussed in terms of very low frequency, i.e., vibrato rate in Hz.
3. Interpolation of \( \tilde{\lambda}_{k} \) and \( f_{k} \) gives vibrato information \( \tilde{\lambda}(n) \) and \( f(n) \) in terms of vibrato amplitudes and vibrato frequencies.
4. Vibrato is computed using equation 11.

\[ v(n) = \tilde{\lambda}(n) \cos(f(n)) \]  

Where \( \tilde{\lambda}(n) \) is an integral of \( \tilde{\lambda}(n) \) at time \( n \).

An obtained vibrato \( v(n) \) is applied to any non-vibrato IF, \( f_{k}(n) \) by \( f_{k}(n) + v(n) \). The applied non-vibrato signal can even be a different vibrato-removed IFs. Two types of lines are vibrato-removed sequence and Vibrato added sequence. The output sound preserved the same timbre as that of non-vibrato signal. These sounds can be heard at website [5].

3.3. Vibrato Elimination

Since vibrato deals with very slow vibration phenomenon, we set up another frame length, 0.4s and frame shift, 0.2s. We define \( f \) as a frame number. Vibrato-eliminated signal for each channel is achieved through the following steps.

1. Estimate the DC component \( d_{k,n} \) of the IF, \( f_{k}(n) \) for each 0.4 seconds and compute their mean \( \bar{f} \) for the longer interval.
2. Derive harmonic number \( m \) from equation 9.

\[ m = \text{round}\left(\frac{\bar{f}}{f_{0}}\right) \]  

Where \( \text{round}(A) \) is a function that returns nearest integer to \( A \) and \( f_{0} \) is the fundamental frequency of the input signal.

3. Correct \( d_{k} \) that does not satisfy equation 10 to \( m \). \n
\[ 0.97f_{m} \leq d_{k} \leq 1.03f_{m} \]  

4. The linearly interpolated \( d_{k,n} \) is newly defined as vibrato eliminated IFs.

Final vibrato eliminated signal is acquired by summing up the sinusoids derived using new IFs for all the channels.

3.4. Vibrato Elimination and Its Transcription to Other Sounds

Vibrato estimation is done only from the fundamental frequency of the input signal. Transcription of the vibrato is done by adding vibrato information multiplied by \( m \), derived from equation 9, to analyzed IFs. Then estimating vibrato is achieved as follows:

1. Estimate the DC component \( d_{k,l} \) of \( f_{k}(n) \), and find their difference for each 0.4 second frame \( l \). This makes a residual signal, \( r_{k,l} \) (n).
2. Compute spectrum of \( r_{k,l} (n) \) and find the max value \( \tilde{\lambda}_{k,l} \) and its frequency \( f_{k,l} \). Note that this is not the spectrum of a signal, but that of an IF, and discussed in terms of very low frequency, i.e., vibrato rate in Hz.
3. Interpolation of \( \tilde{\lambda}_{k,l} \) and \( f_{k,l} \) gives vibrato information \( \tilde{\lambda}(n) \) and \( f(n) \) in terms of vibrato amplitudes and vibrato frequencies.
4. Vibrato is computed using equation 11.

\[ v(n) = \tilde{\lambda}(n) \cos(f(n)) \]  

Where \( \tilde{\lambda}(n) \) is an integral of \( \tilde{\lambda}(n) \) at time \( n \).

An obtained vibrato \( v(n) \) is applied to any non-vibrato IF, \( f_{k}(n) \) by \( f_{k}(n) + v(n) \). The applied non-vibrato signal can even be a different vibrato-removed IFs. Two types of lines are vibrato-removed sequence and Vibrato added sequence. The output sound preserved the same timbre as that of non-vibrato signal. These sounds can be heard at website [5].

5. CONCLUSIONS

We proposed a new phase vocoder-based sound modification technique. It adopts an FIR narrowband filter-bank for analysis and extracts instantaneous amplitudes and instantaneous frequencies for each band. Then the signal for each band is expressed as an AM-FM signal, i.e. a sinusoidal representation for synthesis. This theoretically removes the degradation produced in modification, known as phasiness, since the window’s effect caused by STFT-based analysis is removed. We were able to achieve a high sound quality by time-scale/pitch modification, and vibrato elimination/addition. However, our results are still limited to these case studies.

6. REFERENCES


THE SUM TOOL AS A VISUAL CONTROLLER FOR IMAGE-BASED SOUND SYNTHESIS

Mika Kaukankare
Sibelius Academy – DocMus
mkaukank@siba.fi

Sara Adhitya
EHESS/ IUAV/ STMS: IRCAM/CNRS/UPMS
sara.adhitya@ehess.fr

ABSTRACT

This paper presents an image-based system for the algorithmic control of sound synthesis. Our system allows for the superimposition of multiple spatio-temporal trajectories (paths) on top of layered images. The color data retrieved from the images can be converted into, and interpreted as, control information for controlling the parameters of a synthesis algorithm.

The work is conducted within the PWGL visual programming environment, within which we have developed a new user-library called SUM. The SUM tool is aimed at the integration of image and sound and was originally developed for the sonification of urban maps and the analysis of urban structure.

Lately, SUM has been developed towards a more general image-based compositional tool. Unlike most image-to-sound toolkits, which are limited to a single horizontal time axis, SUM allows for the access of the image data from multiple, user-definable spatio-temporal perspectives and the mixing of both vector and raster images.

In this technical example, we build an image-based ‘mixer’ to control the parameters of a granular synthesizer. Granular synthesis has a long history in the field of computer music and it is frequently used as both a synthesis technique and a compositional method.

1. INTRODUCTION AND BACKGROUND

In this paper, an image-based data mapping scheme for controlling sound synthesis is presented. The scheme originates from the data sonification tool SUM[3]. SUM is a PWGL[5] user-library, aimed at the integration of image and sound, and originally developed as a tool for urban structural analysis.

Image sonification is currently an active research field and of interest from both scientific and artistic perspectives. There are several systems that aim specifically at image-to-sound translation. Hight[1] and Iannitti[2] can be seen as attempts to revitalize Xenakis’ UPIC system. Hight provides the users with a traditional piano roll representation and allows the drawing of curves, lines, and other geometric objects. Other image-based audio software include, for example, Audiosp and MetaSynth. Audiosp is essentially an additive synthesizer which transforms the pixel information retrieved from the image to frequency, time, and amplitude. However, it does not offer any editing tools. MetaSynth is arguably one of the most comprehensive software packages of its class, providing its users with filters, spectral analysis, effects, a sequencer editor, and a timeline editor for large-scale structural organization. Furthermore, AudioSculpt can also be considered an image sonification application but with functionality limited only to the sonification of spectral sonograms. Finally, SumArt[12] is an open, network-based image sonification framework. The data retrieved from images is transmitted through OSC and can be used for synthesis in, for example, MaxMSP and Pd.

When translating information between the visual and auditory domains, the problem of representing and organizing time needs to be addressed. Most of the aforementioned applications are limited to a single horizontal time axis. However, SUM allows us to access the images from multiple spatio-temporal perspectives by superimposing a vector polylines over the areas (i.e., data) of interest.

Most of the toolkits also operate on raster images exclusively and do not support the creation of graphical information within the tool itself. SUM, however, allows the co-existence of both raster and vector images. It supports the superimposition of multiple images, arranged in layers like a 3D matrix of data, which allows for the synthesis of spatially-concurrent graphic information. The images can also be used to mask each other, covering parts of other images, allowing the selective use and alteration of an image without its direct transformation. Thus, SUM allows the multi-layered composition of user-definable images, which can potentially be played as open graphical scores.

SUM can generate MIDI output and it can be connected to external sound sources through OSC[10]. In this paper, the data mapping scheme of the SUM tool is adapted to synthesis control. We use the information, generated by the SUM tool, to control the low-level DSP parameters of a synthesis process defined with the help of PWGLSynth[6]. Here, we have chosen a rather unorthodox approach, where we create an ‘image of a mixer’ from which we retrieve the control information. Due to our path animation scheme, we are also able to simulate the movement of the controls synchronized with the sound.

2. THE SUM APPLICATION

Most image-to-sound toolkits approach the sonification process through different scanning methods, usually restricted to left-to-right scanning. However, for compositional purposes a more flexible representation of time is needed. In this respect, the major contribution of the SUM tool in this respect is its ability to superimpose, on one or more images, an arbitrary number of paths. These paths are spatio-temporal objects which define the connection between the graphical and musical spaces. In addition to time and location, paths define, among others, direction, delay, duration, and speed. SUM supports the co-existence of multiple paths each having their own internal temporal properties.

2.1. SUM Graphical User Interface

Figure 1 shows the SUM user interface. Internally, there are two layers: an image layer; and a path layer. In the given example, we use gray-scale images. The image layer is composed of three different transparent (PNG) images: (a) the ‘spatializer controller’, drawn as a circle with gradient color; (b) the ‘slider controllers’, consisting of four rectangles with a gradient from black to white; and (c) the ‘external controller’, displayed at the top of the figure.

The path layer consists of seven individual paths which model the desired motion profiles of the controllers as a function of time (much like in the case of servo-based mixers). For example, the temporal behavior of the grain duration can be defined graphically by superimposing a path on top of one of the ‘slider controllers’. In our example, the paths travel in both directions. Here, all of the points of the path lie on the same vertical line making editing the paths inconvenient. However, as shown in Figure 2, the paths can also be displayed and edited as breakpoint functions, thus allowing us to define the movement of our imaginary controls more easily.

In this section we give a brief example utilizing the SUM application shown in Figure 1. We aim to use the images as controllers to generate control information for a granular synthesis instrument realized with the help of PWGLSynth. Granular synthesis is an established sound synthesis method that creates sounds or sound textures from small segments of sound. These segments are called grains and they are usually below 100ms in duration. A typical granular synthesizer might allow the superposition of multiple grains and their playback at different speeds. By varying the grain position, duration, spatial position, and density of the grains, different sounds and sound textures can be created.

Figure 2. The paths can be edited in a breakpoint function mode, facilitating the definition of points which lie parallel to either of the two axes.

While the example shown is purely technical, the principle of image-based retrieval of control parameters can also be applied to more artistic applications. This may involve any number and combination of image layers, and the user is free to define the relationship between image and sound.

2.2. SUM Sonification Process

The SUM sonification process is represented in Figure 3. Multiple paths can access the image data. The paths are rasterized according to the Bresenham line algorithm[4] to produce a list of discrete data-points (the paths are single pixel wide). A mapper is then applied to each of the paths to transform the raw data into discrete or continuous control events. The same mapper can be applied to multiple paths, in which case they can have the same timbral quality, but of variable temporal quality will be generated. Conversely, application of the same path to multiple mappers will produce events of the same spatio-temporal quality, but of variable timbral quality. The mappers provide us with two ways of transforming the image data into discrete or continuous control events:

1. using one-to-one mapping, i.e., there is one user value corresponding to each RGB color value, or
2. using an arbitrary transfer function, which allows for the access and treatment of all the attributes contained by a data-point, e.g., RGB color value, time, duration, speed, spatial coordinates, etc.

3. SYNTHESIS CONTROL

In this section we give a brief example utilizing the SUM application shown in Figure 1. We aim to use the images as controllers to generate control information for a granular synthesis instrument realized with the help of PWGLSynth. Granular synthesis is an established sound synthesis method that creates sounds or sound textures from small segments of sound. These segments are called grains and they are usually below 100ms in duration. A typical granular synthesizer might allow the superposition of multiple grains and their playback at different speeds. By varying the grain position, duration, spatial position, and density of the grains, different sounds and sound textures can be created.
In the given example, the disk-shaped object (a) is used to control spatialization. The gradient colors (normalized between 0.0–1.0) are converted into angles (in degrees) and used as an input for a VBA-based[7] spatializer. The four sliders can then be used to control parameters such as grain duration and master volume. Finally, the textured image (c) is used as a structural controller: the image is read from left to right and the color information is used to control the width of the window inside which the grains are randomly chosen.

3.1. The Synthesis Patch

The main PWGLSynth patch is shown in Figure 4(a). It defines the interface between SUM and PWGLSynth. The box named “granular player” is used to receive the information sent by the SUM player. Here, we use the internal real-time message dispatching scheme of PWGL, where SUM sends pitch, velocity, and duration information through the box, which then interprets the incoming data and converts it into synthesis control information. The actual granular synthesizer definition is shown in Figure 4(b). Here, the real-time inlets, marked as ‘:start’, ‘:end’, and ‘:len’, are shown. During playback, SUM sends, through the “granular player” box, the appropriate control information to these inlets which, in turn, change the parameters of the synthesis algorithm in real-time.

3.2. SUM Playback/Animation

SUM provides image-to-sound synchronization during the playback. It animates the pertinent points along the given paths, thus making it possible to follow the ‘graphical score’ and verify the relationship between the image and the resulting sound. Furthermore, the points can be given a custom appearance. Here, we use rectangular shapes when animating the ‘slider’ paths, a loudspeaker-like image for the spatializer path, and a vertical ‘playback line’ for the rest.

4. CONCLUSIONS AND FUTURE WORK

Our preliminary experiments with image-based synthesis control are presented. The SUM tool, a PWGL user-library developed originally for the sonification of urban maps, is here extended to allow for synthesis control. We implement a simple image-based controller application that consists of traditional controllers using a disk-shaped spatialization controller, and a textured image for a more ‘artistic’ controller source.

Future improvements include more accurate and customizable path handling. First, we will define intelligent (meta)path classes that are able to access the image using different predefined methods, such as left-to-right scanning, circular scanning, and potentially also raster scanning[11]. Second, we aim to improve our path-sampling algorithms to allow for better timing and synchronization. For example, it should be possible to attach a tempo function to a path. Third, the paths should be able to send messages to each other, for example, to initiate playback. Fourth, we should allow for the mapping of the image data to arbitrary parameters. Due to SUM’s original MIDI-centric approach, we currently support only pitch, duration, velocity, and channel (and also color for visualization). Fifth, we need to implement more elaborate elementary path types, e.g., based on Bézier curves, and also allow for the definition of arbitrary algorithmic paths. All the code related to the raw image access and path handling (e.g., rasterization) should eventually find its way to an external open-source library written in C or C++. This would allow for the use of the data in other applications and it would eventually allow for the scripting of SUM.

Finally, we are actively researching the possibilities of native 3D formats and we are also in the process of moving from proprietary 2D graphics API to OpenGL[9]. The development of an .obj[8] format importer is on the way and would eventually allow us to create and experiment with various multidimensional graphical scores.

Acknowledgements

The authors would like to thank IRCAM and CCRMA for hosting the research. The work of Mika Kuuskankare has been supported by the John Crampton Scholarship Trustees of Australia. The work of Mika Kuuskankare has been supported by the Academy of Finland (SA137619).

5. REFERENCES

ACOUSTIC-AGGREGATE-SYNTHESIS

Paul Clift
Department of Music
Columbia University, New York, NY

ABSTRACT

Acoustic Aggregate Synthesis describes a real-time performance format which attempts to fuse synthetic and acoustic sound sources in order to achieve a semi-acoustic re-synthesis of a pre-defined acoustic model. At the heart of this project is the desire to maintain the diffuseness patterns, attack/sustain/release characteristics, acoustic amplification etc. of a given instrument whilst ‘overriding’ its timbral characteristics in favour of the creation of a contrasting, readily identifiable, secondary timbre.

1. BACKGROUND

1.1 Additive analysis/re-synthesis

The process described in this essay is essentially one of additive re-synthesis; such processes, even those which make use of the most up-to-date analysis/synthesis tools are well-known and require little elaboration here. Nonetheless, while such procedures have long proved effective as a means of producing/manipulating complex timbres (or, more pertinent here, hybrid timbres) in deferred-time, or, when used in real-time, as the basis of orchestrated synthetic accompaniment to instrumental performance, the notion of unifying acoustic & synthetic sound sources in the creation of a perceived single, hybrid timbre with historically-defined semantic characteristics (e.g. recognisable as a flute) remains a fecund territory for exploration. Worthy of note is the ambitious research in the hybridisation of timbre undertaken by Jonathan Harvey et al. which culminated in the premiere of Speakings (2007-8) for orchestra.

1.2 Augmented instruments: the integration of transducers into acoustic instruments

AAS also follows upon a long tradition of ‘augmentation’ of acoustic instruments through the integration of diffusors, transducers, indeed a potent factor in the success of timbral homogeneity (in particular when a combination of acoustic & synthetic sounds is used) is perceived directionality. Although relatively little has been written on this subject, countless attempts have been made, with varying degrees of success, at establishing fusion between acoustic & electronic sounds through maximal spatial proximity of multiple sound sources. This phenomenon may be enhanced when, as is practicable in certain cases, electronic sounds are subject to a given instrument’s acoustic diffusion patterns (i.e. the electronic sound source is diffused not directly, but rather after passing through one or more instrumental mechanisms). This notion is discussed further in section 2.5 of this essay.

2. ACOUSTIC AGGREGATE SYNTHESIS (AAS)

This process, using the MaxMSP interface, makes a real-time comparison between an instrumental-template (in actuality a list of the 64 most prominent sinusoidal components of a given timbre, created in deferred-time) and an incoming signal; in response to each component detected in the incoming sound-source, three possibilities exist:

- where the incoming signal contains a component which is present (or is within relative-proximity to a component, as determined in the ‘margin-of-frequency-deviation’ variable) in the instrumental-template but of lower intensity, the difference in intensity is calculated and applied to the electronic diffusion of that frequency;

- where the incoming signal contains a component which is absent in the instrumental-template, or vice-versa, nothing is diffused;

- where the incoming signal contains a component which is present (or is within relative-proximity to a component) in the instrumental-template but of greater intensity, nothing is diffused.

Thus, in the re-synthesis of the sound source used in the creation of the instrumental-template, a variable proportion is generated acoustically and is therefore subject to the source’s aforementioned characteristic nuances of acoustic diffusion, attack, sustain, decay etc.

2.1 The creation of instrumental-templates

![Figure 1. Screenshot of AAS instrument-templatemake; user interface](image1)

In this case, the most commonly-found example is, for obvious reasons, the use of loudspeakers inside a piano; one such example is Alvin Curran’s Twentieth Century (1994) for diskklavier & electronics. Instrumental templates are simply lists of frequencies & intensities organised according to MIDI pitch-values. I have already developed a number of instrumental-templates which span the entire pitch and intensity range of common orchestral instruments. They may be implemented statically, i.e. taking just one pitch/dynamic from the template data and transposing it to match the FO of the incoming signal, or dynamically, by approximating the FO of the incoming signal to the nearest match in the template, then performing minor corrective transpositions (read: multiplications) of template frequency data. Also, multiple instances of a given pitch (2-32) may be created in order to avoid the sensation of timbral ‘stasis’. An incoming signal with a progressive change in timbre, such as a note struck on a piano, or a bell (or indeed any other sound with a natural decay) may be used to great effect. It goes without saying that the comparative model will generate no output once the comparative model reaches zero-intensity. Sounds which change timbre sporadically (such as polyphonic textures or combinations of percussion instruments) have generally proven to be ineffective with 64-component analysis/re-synthesis.

2.2 Interface

![Figure 2. Screenshot of AAS interface; in this example, a template for clarinet in Bb is dynamically matched to an incoming signal of a horn playing Bb4](image2)

2.2.1 Interface

If the intention is to achieve a faithful reproduction of the timbre used to generate the instrumental-template, it is of course essential match it with an incoming signal whose fundamental is relatively close; obviously, attempting to reproduce a flute by means of a contrabass’ low-E would generate a timbre which is denatured beyond recognition. Assuming the incoming signal has as its fundamental a frequency corresponding to a pitch which is present (or, as is most often the case, one within relative proximity to a pitch which is present) in the template, the process of dynamic-matching will ensure maximal preservation of the timbral characteristics of the acoustic model.

![Figure 3. spectrogramme visualisations of: (left) a flute template (B♭4, mf), selected dynamically to best match pitch & intensity of the incoming signal; (right) the same template ‘fine-tuned’ (transposed & attenuated) amplified to match pitch/intensity of the incoming signal precisely](image3)

It may, of course, also be desirable to match a template with a sound source of contrasting register. In this case, a given template pitch may be selected and transposed dynamically to match the FO of the incoming signal; the generated timbre will, of course, become progressively more denatured the further it deviates from its original pitch, which can be used to great effect in certain cases.

2.3 Summary of the process of synthesis

Below is a series of examples which illustrates the multi-step process in the generation of output:

![Figure 4. spectrogramme visualisations of: (left) an analysis of incoming signal, an oboe (B♭4, mf); (right) an analysis of incoming signal, an oboe (B♭4, mf)](image4)

1. flue, oboe, clarinet in B♭, bassoon, trumpet (ordinario, w/ harmon mute), horn (ordinario, stopped, brassy) & trombone (ordinario, w/ harmon mute) etc.

2. The most commonly-found example is, for obvious reasons, the use of loud-speakers inside a piano; one such example is Alvin Curran’s Twentieth Century (1994) for diskklavier & electronics.
The asterisk in the above example denotes an ambiguity concerning the source of incoming information. The combinations of instrumental-templates and incoming signal sources must be made with regards to the timbres of the output: of course, an informed choice is required. As such, I have compiled a simplified flow-chart summary of the process.

2.4. Empirical results in controlled tests

The asterisk in the above example denotes an ambiguity concerning the source of incoming information. The combinations of instrumental-templates and incoming signal sources must be made with regards to the timbres of the output: of course, an informed choice is required. As such, I have compiled a simplified flow-chart summary of the process.

2.4.1 Choice of timbres

It should be obvious at this point that it is desirable to emulate a sound which is of ‘greater’ timbral complexity than that of the incoming signal. For example, generally speaking a flute template functions effectively with a clarinet as sound-source, but less so with an oboe. Nonetheless, we find that anomalies and exceptions are commonplace. For example, the template for a flute playing fortissimo in the ‘third’ register (D6 - A7) is effective when used with an oboe mezzo-piano in the same register; the flute template contains multiple components which are stronger than the corresponding components in the incoming oboe. One may complement any timbre with any other provided that there is some degree of timbral inequity. As mentioned, there will be further discussion of this principle in subsequent, more comprehensive articles.

2.5. Diffusion

As mentioned earlier, in order that the resulting aggregate timbre be perceived as such (and not a mere juxtaposition of acoustic sound-source and electronic synthesis), it is beneficial to place the loud-speaker diffusing the aggregate-synthesis in very close proximity to the acoustic sound source.

2.5.1 Incorporation of loud-speaker into the instrument

Notably with the bass clarinet and all members of the saxophone family (with the exception of the soprano), it is possible to incorporate the loud-speaker & microphone into the instrument itself.

2.5.2 Use of loud-speaker in proximity

In the case of instruments for which the integration of a loud-speaker is impractical, such as flute or violin, a speaker placed directly in front of the performer at the vertical acme of the instrument’s average-maximum acoustic-diffusion has proven effective in achieving a high level of fusion.

3. ACKNOWLEDGEMENTS

I would like to thank Miller Puckette, René Causé, Jean-Baptiste Barrière, Gérard Bertrand & Alain Terrier for their ongoing assistance in the realisation of this project.

SELECTED BIBLIOGRAPHY


A comprehensive discussion of this is beyond the scope of this short article, but will be included in future writing on this procedure.
MEASURING THE PERFORMANCE OF REALTIME DSP USING PURE DATA AND GPU

André Jacovsky Bianchi¹, Marcelo Queiroz⁴

Computer Science Department, University of São Paulo, Brazil
{ab,mq}@ime.usp.br

ABSTRACT

In order to achieve greater amounts of computation while lowering the cost of artistic and scientific projects that rely on realtime digital signal processing techniques, it is interesting to study the performance of commodity parallel processing GPU cards coupled with commonly used software for realtime DSP. In this article, we describe the measurement of data roundtrip time using the Pure Data environment to outsource computations to GPU cards. We analyze memory transfer times to/from GPU and compare a pure FFT roundtrip with a full Phase Vocoder analysis/synthesis roundtrip for several different DSP block sizes. With this, we can establish the maximum DSP block sizes for which each task is feasible in realtime by using different GPU card models.

1. INTRODUCTION

The highly parallel nature of many Digital Signal Processing (DSP) techniques makes the use of commodity hardware for parallel processing increasingly useful for realtime scenarios of artistic performances, small technical applications and prototyping. To make better use of parallel processing devices it is interesting to study if different combinations of hardware and software can meet specific criteria.

Wideely used by digital artists, Pure Data (Pd) is a realtime DSP software licensed under free software terms, which can easily be extended and combined with control hardware through wired or wireless interfaces. Also, Pd is able to handle audio and video signals making it possible to build and control arbitrary DSP algorithms.

To enhance Pd with parallel processing capabilities, one of the lowest cost solutions nowadays is to attach to it a Graphics Processing Unit (GPU), to which Pd will then be able to transfer data back and forth and request (parallel) computation to be performed over it. If all this can be done in a time period of less than one DSP cycle (the period for one block of samples that is act upon by the DSP computing time), then it may in fact be worth it to combine Pd and GPU for realtime DSP performances.

This work focuses on measurement of common parallel tasks, such as memory transfer and kernel execution times, and uses Pd extensible design to implement interaction with the GPU using C and CUDA C code compiled as shared libraries.

The use of GPU for realtime audio signal processing has been addressed in recent work [4, 2, 3, 5, 6]. For instance, by measuring the performance of the GPU against that of a commodity CPU, Tsingos et. al. showed that for several applications it is possible to achieve dramatic speedups (factors of 5 to 100) [6]. A similar approach to ours was carried by Savioja et. al., who analyzed the GPU performance on additive synthesis, FFT (and convolution in frequency domain) and convolution in time domain [5]. Our work differs from these in two ways. First, we make use of Pure Data as the software environment for interaction with the GPU API, thus providing a look into the use of parallelism on a widely adopted realtime computer music platform. Additionally, we provide a fine-grained measurement of memory transfer times, so we can compare these with actual processing time.

Our computation outsourcing model assumes that the GPU will be used by Pd in a synchronous manner. At every DSP cycle, Pd will contact the GPU, transfer a portion of memory to it, call kernel functions over that portion of data, wait for these kernel calls to end, and then transfer data back to the host's main memory. Our aim is to perform the following measurements, in order to establish the feasibility of using a GPU-aided environment with Pd on realtime performances:

- **Memory transfer time.** Since a GPU only processes data that reside on its own memory, memory transfer can represent a bottleneck for parallel applications that use GPU: generally, the fewer the amount of data transferred the better.
- **Kernel execution time.** This is the total time used by all instructions performed on the GPU, after memory is transferred from the host and before memory is transferred back to it.
- **Kernel execution time.** This is the total time taken by all instructions performed on the GPU, after memory is transferred from the host and before memory is transferred back to it.
- **Full roundtrip time.** This is the total time taken to transfer data from host to device, operate on that data, and then transfer it back to the host. This is the single most important value to compare with the DSP cycle period, in order to establish the feasibility of using the GPU in realtime environments.

On the following section, we describe the setup used for performing time measurements for these tasks.

2. METHODS

In order to measure kernel and memory transfer time using Pd and GPU, we set up a Pd external that communicates with the GPU and keeps track of elapsed time between operations. The external behaves as a normal Pd object that receives input signals and produces output signals, but actual DSP is delegated to the GPU.

Our test environment is an Intel(R) Core(TM) i7 CPU 920 2.67GHz with 8 cores and 6 GB RAM, running Ubuntu GNU/Linux 11.04 with linux kernel version 2.6.28-13-generic, and equipped with two models of NVIDIA GPU cards: GeForce GTX 275 (240 cores, 896 MB RAM, 127.0 GB/s memory bandwidth) and GeForce GTX 470 (448 cores, 1280 RAM, 133.9 GB/s).

For implementing the Pd external we used standard C and CUDA C. As a basic DSP algorithm implementation, we make use of CUFFT5. NVIDIA’s implementation of the FFT algorithm which is compatible with the widely used FFTW collection of C routines3.

2.1. Implementations

To be able to evaluate the performance of our setup for realtime DSP, we started with an implementation of a pure FFT external that transfers data into the GPU, runs the FFT over it and then transfers data back to the host, thus providing the host with a frequency domain description of the signal block. The results for different block sizes on the GTX 470 can be seen in Figure 1, and will be discussed in the next section.

In order to estimate how much more computation we may export to the GPU while preserving realtime operation, we implemented a full Phase Vocoder1 analysis and synthesis engine. This also allows for a comparison of the time spent by the GPU in regular user code versus device-specific professionally-engineered library code.

An implementation of the Phase Vocoder for the GPU can use parallelism in two ways. First, it can estimate the instantaneous amplitude and frequency for each oscillator by making use of the parallel FFT as we just saw. After that, as the result for each synthesized output sample does not depend on the calculation of other sample values, the PV can perform one additive synthesis for each output sample of a DSP block in parallel. Thus, an implementation of the Phase Vocoder on the GPU uses the same amount of data transfer between host and device as the pure FFT algorithm, but comprises more kernel calls and more computation inside each kernel. The results for memory transfer and kernel time of our Phase Vocoder tests for different block sizes can be seen in Figure 2, and will also be discussed in the next section.

2.2. Tests

We ran the FFT and PV algorithms for a period equal to 100 DSP blocks, for block sizes of 2ⁿ with 6 ≤ n ≤ 17, and then calculated the mean time taken for data transfer (back and forth) and full PV analysis and synthesis. The maximum block size considered was 2¹⁷ = 131072 samples corresponds to a period of about 3 seconds of audio. The execution time of the full Phase Vocoder for block sizes of more than 2¹⁶ samples largely exceeds the corresponding DSP period, so this block size seems enough to provide upper bounds for feasibility of computation as a function of block sizes.

As observed by Savioja et. al. [5] the additive synthesis snippet is computationally intensive and very sensitive regarding the method used to obtain each sample value of the sinusoidal oscillators. We have compared the performance of 5 different implementations: (1) 4-point cubic interpolation, (2) linear interpolation, (3) table lookup with truncated index, (4) GPU’s trigonometric primitives (specifically, we used the sincf() function of CUDA API, which computes a double precision floating point number), and (5) GPU’s built-in texture lookup functions. The results for each implementation can be seen in Figure 2 and will be discussed in the next section.

3. RESULTS

To illustrate the results obtained, we present graphs for the tests made with batch processing, i.e. letting Pd run as fast as it can without waiting for the full periods of DSP cycle to end to produce new samples. Wave files were used as input and output of the FFT and PV algorithms were carried out for different block sizes as specified in the last section. Test results were generated for both models of GPU, but as memory transfer times and FFT kernel execution times vary according to expectations between models (due to the differences of transfer speed and number of cores), we show these results just for the faster model.

Figure 1 presents the result of memory transfer times and FFT kernel execution times for different block sizes on the GTX 470. Figure 2 shows kernel times for the full

---

1This work has been supported by the funding agencies CAPES and FAPE/ESP (grant 2008/08632-8).
2http://www.puredata.info/
3http://developer.nvidia.com/cufft
4http://developer.nvidia.com/cuda-toolkit
5http://www.fftw.org/
6http://www.lttng.org/
PV analysis and synthesis as a function of block size for both cards. By comparing Figures 1 and 2, we can see that there is a noticeable difference, of some orders of magnitude, between the time it takes to run the pure FFT and the full PV. Comparing the time taken by the two different algorithms on the same model of card, we see that the FFT itself takes time comparable to the memory transfers time, of about tens of milliseconds, while the full PV implementation will take many seconds for larger block sizes. This means that hundreds of pure FFT executions could occur in a DSP cycle while only few full PV analysis and synthesis can actually be performed on the same amount of time.

3.1. Memory transfer times

Regarding memory transfer, we can observe on Figure 1 that the time it takes to transfer a certain amount of memory from host to device and back sees approximately linear in relation to block size. This seems reasonable once the bandwidth of data transfer between the host and the GPU is constant (see also Section 2). Despite that, we should notice that it is unwise to use the memory bandwidth value to make assumptions regarding memory transfer speed, because this relation depends on the architecture implementation of the memory hierarchy. Also, we could determine that memory transfer results are very close for the FFT and PV implementations. This also seems reasonable given that the amount of memory transferred on both implementations is the same. This indicates that the number of kernel calls or the amount of time kernels take working on data seems to have no influence on memory transfer speed.

Finally, we should notice that the difference of time taken for memory transfer on the two different GPU models is very small, and in both scenarios the back trip takes a little longer than host-to-device transfer. Again, this difference is expected to few memory transfers in each direction, then there is no need to bother with memory transfer time as its magnitude has something to do with differences in texture memory fetching implementations on different card models. From these plots we can see that, for the GTX 275, blocks with size bigger than $2^{14}$ take more time to compute any of the Phase Vocoder implementations than the time available for real-time applications. A similar result was found for the GTX 470 for blocks bigger than $2^{17}$ samples.

We can also observe that Phase Vocoder times grow superlinearly for all implementations. Since the number of oscillators in the additive synthesis is roughly half the number of samples in a block, a quadratic computational complexity ($\propto n^2$) is expected. The degree of parallelism brought by the GPU is not enough to produce differences in profile for the variant oscillator lookup methods implemented, and the differences in scale are accountable by the hidden constant in the big-O notation.

4. DISCUSSION

Given the results described in the last section, we can conclude that small implementation differences can have significant results regarding kernel time consumption on the GPU. It is not clear whether the numerical quality of the built-in sine function is better than our 4-point cubic interpolation, but it is clear that conscious choices have to be made in order to use the GPU’s full potential for larger block sizes.

We can also conclude that, if we restrict the roundtrip to few memory transfers in each direction, then there is no need to bother with memory transfer time as its magnitude is of the order of tens of milliseconds while DSP block period is a bit difficult to explain. On the GTX 275, it has the worse behaviour of all methods, taking 40% longer than the second most expensive, cubic interpolation. On the other hand, on the GTX 470 implementation (5) has a performance comparable to the others, being a little bit faster than cubic interpolation. We assume that this behaviour has something to do with differences in texture memory fetching implementations on different card models.

Figure 2. Kernel times for different implementations of oscillator calculation running on different card models.

Figure 2. Kernel times for different implementations of oscillator calculation running on different card models.
A SYSTEM FOR RECORDING ANALOG SYNTHESISERS WITH THE WEB

Lucas Zawacki
Federal University of Rio Grande do Sul
Instituto de Informática
lzawacki@inf.ufrgs.br

Marcelo Johann
Federal University of Rio Grande do Sul
Instituto de Informática, PGMICRO
johann@inf.ufrgs.br

ABSTRACT

This paper proposes a system composed by hardware and software for remote accessing real analog synthesisers using the World Wide Web. In this system, a file with the user’s submission is submitted to the site, being played with an actual analog synthesizer, and a high-quality recording is made available back to the user. The proposed service model is characterized by being played with an actual analog synthesizer, and a real-time access to an analog synthesizer implies that whenever the system is being used, or available to a given user, it cannot be accessed by others. In other words, we need to serialize the users, something that cannot be implemented efficiently. The only way to do that is to reserve big time frames (like half-hours) for each user, resulting in poor system performance. In the proposed system, instead of being able to play with the synthesizer in real time, like in [4], the user is expected to prepare a job and submit it to a queue, where it will be processed shortly. To our knowledge this is the first work that addresses batched access to analog synthesisers.

Today, three decades later, we must acknowledge the importance that analog synthesisers still have. The synthesisers and computer software (plug-ins) available today are not only used to produce sounds like only digital technology can generate, but also to mimic the old technologies, acoustical instruments, and the sounds of many analog synthesisers. This is already a sign that they form a class by themselves, real musical instruments capable of characteristic performances and nuances. Actual analog synthesisers are still heavily employed in modern pop music [1]. Descendents of the Moog synthesisers are used to provide the deep basses in genres like Rap, Dance, effects in electronic music, as well as solos in countless Jazz and Rock groups, just to mention a few examples. In summary, it is safe to say that while the digital revolution brought us incredible possibilities, better control, new sounds, everything widely accessible from simple consumer computers, the original analog machines that are already part of the music aesthetics will continue to be superior and irreplaceable in many aspects for years to come.

There is an accessibility issue, however. With few exceptions, analog synthesisers, both new and old, are more expensive than before. Therefore, this construction must employ old techniques, components or practices that are not the mainstream today, while the original ones are getting older, so finding them in good shape is becoming harder. There are indeed collectors that have a good set of original machines, but just a few people can access them, and new generations might not even get the chance to know they sound, led to think that the plug-ins are accurate reproductions.

The contribution of this paper lies in this context. We propose a system that puts an analog synthesizer, or a collection of them, accessible through the Web, for recording performances submitted by the public. Furthermore, we propose a batched access model to the service so as to achieve a much higher efficiency, which is not possible with real-time access.

The remainder of this paper is organized as follows. Section 2 explains the top-level system architecture and organization, highlighting key elements, while a first fully working prototype is described in section 3. Section 4 covers many aspects related to how a system like this should and should not be used, pointing out the real aggregate value, as well as the viability of commercial exploration. Section 5 discusses many options to address the most important challenge, which is the sound preview. Finally, section 6 enumerates goals and new perspectives that this system opens up.

1. INTRODUCTION

We might think that every new technology inevitably replaces the old. This is not true in many cases, especially in arts and music. Consider for example the traditional acoustical instruments used in classical music. Despite the impressive advancements in digital audio technology, possible with digital audio, they were not replaced either for artistic expression neither for many commercial productions. It is true that most tracks in television advertisement are made with samplers and synthesisers, but on the other hand many major movies still use soundtracks composed and recorded by real orchestras with acoustical instruments.

The same can be said about analog synthesisers. They started to be replaced with the advent of the digital technology in the 80’s. There are many reasons for this trend to have been so hardly pursued. Powerful analog machines were big and expensive, their programming could not be easily stored, and they did not have a perfectly stable tuning, among others, all very important factors. They were also limited in their ability to reproduce complex initial transients that are characteristic of instruments like the piano.

2. SYSTEM ARCHITECTURE

A key aspect in the proposed system is the batched access model. It works like an “on demand server”, to which the user submits a job and retrieves the expected result after some short time. This is a different scenario compared to real-time access, which requires tight timing constraints, and server, with latency/throughput issues, as addressed in many works on network online music [6]. The main reason for the batched access is the increase in system usage. Real-time access to an analog synthesizer implies that whenever the system is being used, or available to a given user, it cannot be accessed by others. In other words, we need to serialize the users, something that cannot be implemented efficiently. The only way to do that is to reserve big time frames (like half-hours) for each user, resulting in poor system performance. In the proposed system, instead of being able to play with the synthesizer in real time, like in [4], the user is expected to prepare a job and submit it to a queue, where it will be processed shortly. To our knowledge this is the first work that addresses batched access to analog synthesisers.

The system must be composed of a front-end, a hardware infrastructure and a software processing system that implements the desired functionalities. Starting with hardware, at least the real synthesizer, a computer, an audio interface and a MIDI interface are needed. The software system must be able to play the MIDI performances, and to accept any changes the user wants to make, back the resulting audio with an audio interface. The synthesizer must be MIDI playable. However, there are standard MIDI to CV converters available, that convert audio (Voltage) that can be used to play older analog synthesisers that do not support MIDI directly. The software system has at least two other attributes, which is to implement a queue, dispatching each job with the proper synchronization, and to make a set of pre-processing tasks. Among these, it is important to check the user submitted MIDI file for errors, bounds and filter out undesired messages.

The last component is the user front-end. It must allow the user to submit its performance, select a set of parameters and get access to the resulting recording. The MIDI submission can be implemented from inside a plug-in running in a given DAW (Digital Audio Workstation), or with a standard Web page, and we choose the later as a first option. It requires the user to export each one of its MIDI performances from its DAW to a MIDI file on its computer’s file-system, from which it can be uploaded to our system.

There are many alternatives to implement more sophisticated interactions with the system. Feedback can be employed in two ways: first, with a MIDI renderer, reporting to the user, and this should not be implemented, pointing out the real aggregate value, as well as the viability of commercial exploration. Section 5 discusses many options to address the most important challenge, which is the sound preview. Finally, section 6 enumerates goals and new perspectives that this system opens up.

3. THE FIRST PROTOTYPE

The first prototype was implemented as a proof of concept for this service and is currently working privately. The goal for this implementation was to check its viability using standard technologies and also test the access model and usage issues. The system offers access to a Studio Electronics ATC-1 with the Moog-type filter installed [7], a synthesizer that inherits most of its circuits from the Minimoog [5]. The synthesizer is recorded using the Steinberg Cubase Elements DAW, and custom A/D hardware. The whole setup is running in an iMac connected to the synthesizer via the MOTU Timepiece A/V MIDI USB interface.

The Web interface is a PHP script that: receives the upload form, ask for a MIDI file. It lets the user select a sound program from the available sound bank. After the upload is done the system offers access to the full-quality audio files, and so on. There is a separate worker thread that synchronizes the job execution. To execute each job, it takes the request from the queue and runs a custom AppleScript sequence. This sequence provides the automation of the DAW to create a new project, import the MIDI file, play it, record the audio and close the project after completion. The automation is possible using a combination of shortcut keystrokes and GUI operations, possible with the AppleScript language. It is partially performed as if someone was using the computer, and that includes a set of delays (0.5s or 1s) so that the GUI system have time to create its elements before the next command is asked.

4. AGGREGATE VALUE AND BUSINESS OPPORTUNITIES

As mentioned before, the biggest challenge for such a batched system lies in the absence of instant sound feedback. Some interesting options to compensate for this problem will be addressed in the next section, but let us start analysing the typical usage cases, interpreting the possibilities that this service can bring.

4.1 Typical use

A typical use would be to have a mixed MIDI/audio production in the user’s DAW, where some of the MIDI tracks need emphasis and call for a higher quality, like in a solo or bass line. The user can then browse a set of sound samples from the library, select one and submit its job to record one or two tracks with a real analog synthesizer, increasing their quality in terms of timbre and articulation, for example. Musicians already know that such sounds stand up in the mix [1], unlike their digital emulations. If used for a few tracks or shorter passages, even moderate costs would be perfectly acceptable, mostly because the user would pay only for the duration of the performances.
If the accessing costs are really low, as we will propose ahead, bedroom or garage musicians, as well as experimentalists, can risk more trying to get different sounds and techniques that can provide additional reasons to seek for analog synthesizers and plug-ins, helping to increase the overall market.

4. Simple cost estimate

As it was said before, a key issue for the system’s economic viability is the high scalability attainable with the batched access. It is yet to be verified whether users will adapt themselves to the system or not, and it certainly depends on the accessing costs. If we can prove that the cost for accessing the service is low, it is expected that people will recognize the possibilities, the aggregate value and the returns in terms of quality that they can get from the system and use it.

Let us assume that the market exists and is larger than what can be provided by a single synthesizer of a given kind. Based on this assumption we find that the system should be sized according to the demand so as to keep its utilization bound by some parameter. Assuming a basic M/M/1 queue model, for simplicity, with utilization rate $\rho = \lambda / \mu$ (of 50%, from basic queue theory we get that the number of clients in the queue is expected to be $N \approx \rho = 0.5$). Therefore for a given average job size the client is expected to wait twice this time (including the time to process the 1 client in the system, and its own job), which is a good quality of service. Assume now that a US$2,500.00 synthesizer is being operated by a computer system and accompanying audio/MIDI interfaces, also in the market value of US$2,500.00. Next let us suppose that the system will charge only US$1.00 for each minute of sound played and recorded. At 50% of utilization, the system should suffer no day, generating a revenue of around US$500 (remember that there is the overhead of the delays when processing each job). In such hypothetical scenario, the equipment would pay itself in just 10 days.

While we are ignoring the other costs of operating the service, the above numbers were presented to show that a US$1.00/minute target price is not unrealistic. Now by comparing this price (and the easy Web access) to the option of renting a keyboard or renting a high-quality recording studio, that can reach hundreds of dollars per hour, to do the same job, it becomes clear that this is an attractive alternative to many users. We can expect that a lot of people that would never be able to use the standard alternatives will try the new service.

5. SOUND SELECTION AND PREVIEW

One of the biggest differences from having access to a synthesizer through this system as opposed to sitting in front of it is the preview of the rendered sound. Real time interaction with the synthesizer is not possible. Another problem is the programmability of the synthesizer parameters. Even though MIDI is a standard way to communicate parameter changes to the hardware, the way synthesizers implement these changes does vary. A composer might want to have a more accurate, and a tighter control of the pitch changes of the program aspects of a synth and this is something such a system should be able to provide. There are a few options that can be used to work around both problems:

- Pre-recorded samples - there must be at least a standard sample for each sound, stored in a different server, that the user can browse and listen to them to choose the one it needs;
- User feedback before recording - a short segment of the submitted job can be recorded and given to the user before processing it completely. In this way he can check the sound and parameters, as volume, base pitch, and so on. Another option would be to allow a given number of re-recordings for the same performance with different parameters;
- VST plugins, software emulators - Software plug-ins can be provided as to mimic the sound of each program in the actual synthesizer. With this option, the musician can prepare its production before sending it to be processed. A user that knows from the beginning that she/he will be able to access the same sound from a real analog machine for its recordings.

6. GOALS AND PERSPECTIVES

The main contribution of this paper is the proposal of a new service of batched access to analog synthesizers operated as analog servers. The batched access (not in real time) is key to enable high utilization, increasing hardware efficiency and decreasing the access costs. The service can make expensive or rare analog machines easily available to high quality recordings for everyone interested. We identified that it is quite straightforward to put a modern MIDI-equipped analog synthesizer online, as it was demonstrated in our working prototype. A new service like this should help to improve the quality of new recordings ranging from the most modest beginners up to top artists, in accordance with new trends in social music production [8], as it provides access to a larger variety of sound sources for all.

The system may also help to preserve the entire sound palette that many generations of synthesizers could create by promoting the continuous use of real analog synthesizers of all times. Many synthesizers that are not MIDI capable can be controlled with specific hardware. Even synthesizers that do not have CV inputs and cannot be programmed remotely could also be operated with a dedicated hardware work on them. This investment might be interesting not only from the economic point of view, but also to keep iconic instruments alive and working for future generations in an affordable and cost-effective way. The variety of analog machines that can be placed available as servers is very large, and we can think of the system as an analog farm or analog cloud.

Finally, we can also extend the idea to automate real acoustic instruments so that expensive ones could be accessed by the public. This is an idea that can include pianos, electric pianos, organs of all types and varieties, and the list goes on. As the system’s economic advantage lies in the higher utilization rates, there are also other analog sound processes employed in audio productions that can be analyzed as alternatives to be installed as analog servers. As future work, it is our intention to open a new version of the system to the public as soon as possible so that we can check the demand, see how it is actually going to be used and evaluate the impact that it has in the field.

REFERENCES

TOWARDS INTERACTIVE SONIFICATION BASED ON REAL-TIME PHYSICS SIMULATIONS

Rhys Perkins
Anglia Ruskin University
Department of Music and Performing Arts,
Cambridge, CB1 1PT, UK

ABSTRACT

There has been an increasing amount of research utilising 3D virtual environments as a core component of interactive sonification. While showing considerable potential for their ability to produce both real-time visualisation and sound, they often come with constraints as a result of their design decision processes.

This paper presents developments of a prototype that has arisen out of my attempts to address some of the issues involved in bringing sonification to a wider audience, with an eye to making it more flexible and immersive environment for composition and sound design.

1. INTRODUCTION

The emergence of improving technology has provided an opportunity to overcome the limitations of previous work [1] where computational requirements would produce significant latency between the audio and the visual, inhibiting real-time interaction. Dedicated hardware, such as the graphics processing unit (GPU), allow for the sharing of resources and have recently shifted their focus towards more general purpose systems such as CORDIS-ANIMA [3] allowing for the sharing of resources and have recently shifted their focus towards more general purpose hardware, such as the graphics processing unit (GPU), providing an opportunity to overcome the limitations of previous work [1] where computational requirements would produce significant latency between the audio and the visual, inhibiting real-time interaction. Dedicated hardware, such as the graphics processing unit (GPU), allow for the sharing of resources and have recently shifted their focus towards more general purpose systems such as CORDIS-ANIMA [3] allowing for the sharing of resources and have recently shifted their focus towards more general purpose hardware, such as the graphics processing unit (GPU), providing an opportunity to overcome the limitations of previous work [1] where computational requirements would produce significant latency between the audio and the visual, inhibiting real-time interaction.

2. KEY COMPONENTS

2.1. Human Interaction

Investigation by Hunt and Herrmann [7] has shown that applying human interaction in real-world contexts to sonification can help improve interface design. They state: “we aimed at interaction using only their physical environment and making continuous use of all their senses. When we perform an action on an object we expect some kind of reaction and our perception of objects builds up over time through this interactive process. The objects that we find in this system adhere to the unanchoring laws of physics that our neural hardware has been effectively programmed to deal with over many years of evolution. This enables the user to utilise their acquired skills in order to manipulate high-dimensional data via objects with predictable behaviour and response. The user can interact through inference relating to expectation due to the behaviour of known physical counterparts [3] reducing the gap between reality and simulation. This is turn means we can naturally map several physical gestures to manipulate objects and influence data streams. Interaction with the objects has a direct effect on the procedurally generated simulation in a similar manner to model-based sonification [7] where the user supplies the initial excitation. By grabbing, moving and throwing objects it is feasible to perform a wide range of actions from plucking; each of which results in changes to the data dimensions. This direct process introduces information manipulation to the average user at a more accessible level when compared to other similar applications [8], since no coding knowledge is required, allowing us to visualise the underlying algorithms with a natural mental model [4].

Presentation of the data in this manner brings its own set of problems in that there is an increased potential to overwhelm the viewer with information. When considering high-dimensional spaces our data study [9] argues for a mental model simpler than brute-force awareness of every detail to avoid cognitive overload. Parameters should be cross-coupled so that the performer naturally thinks of certain parameters as varying together in predefined patterns. The high-dimensional data, encapsulated visually by each model, allows us to intrinsically understand how the parameters vary together. Throwing an object would imply a change in velocity that would be influenced by the mass of the object. Spherical objects are more naturally envisaged as representing a physical world, while cubes are associated with more static, less dynamic behaviour. This enables the user to utilise their acquired skills in order to manipulate high-dimensional data via objects with predictable behaviour and response. The user can interact through inference relating to expectation due to the behaviour of known physical counterparts [3] reducing the gap between reality and simulation. This is turn means we can naturally map several physical gestures to manipulate objects and influence data streams. Interaction with the objects has a direct effect on the procedurally generated simulation in a similar manner to model-based sonification [7] where the user supplies the initial excitation. By grabbing, moving and throwing objects it is feasible to perform a wide range of actions from plucking; each of which results in changes to the data dimensions. This direct process introduces information manipulation to the average user at a more accessible level when compared to other similar applications [8], since no coding knowledge is required, allowing us to visualise the underlying algorithms with a natural mental model [4].

2.2. Objects

Objects provide a modular approach to the way the user experiences the underlying data bringing more granularity to the environment they inhabit. Depending on the object’s configurable physical parameters the computer will automatically simulate subsequent interactions as the object reacts to its current environment and user intervention. However, it can be argued that there are parameters that have no direct effect on the simulation which are just as important for the user to exploit. These properties can enhance a...
users’ experience, and encourage them to learn, by creating associations through further visual abstractions that can be audibly reinforced. As one example, the colour of an object could be changed in order to present information in a new manner. According to research in Gestalt laws of grouping [12] there is a stronger tendency to group local elements by common colour than by similar objects in shape. This would imply that, in some cases, our brains are more receptive to the material that encompasses each shape, rather than the shape itself. By involuntarily grouping similar coloured entities, the audience’s attention would be drawn to a single contrasting colour object, perceiving it as being outside of the group. The performer could then take advantage of this visual phenomenon by using it to introduce a solo theme or demonstrating object-specific sonic behaviour

In a typical physics simulation most objects will likely come to rest until excitation provides the impetus for movement. If frequent changes in data are desired then we can introduce objects that encourage movement without relying on direct user interaction or define a series of events through scripted logic configured via object specific context menus (Figure 1). By generating a script, a user can change any number of object attributes at a specific time allowing for the creation of visual algorithms or a gestural score. Cador [10] refers to musical structure and the process of composition, being made of assembled simpler entities that can be put together by a generative process, an animated process or a dynamic gesture. The range of objects provided by the software, along with the various means of interaction, aim to reflect those compositional methods within an automated simulation of mechanical physics.

![Figure 1: A selected object with its context menu](https://example.com/image1)

2.3. Mapping

In the prototype I describe here the parametric mapping process grants an insight into the composer’s conceptual understanding of the data dimensions. It has been suggested that metaphors help create more intuitive mappings [13] and are well suited to parameter mapping sonification [5]. When discussing his musical creation “picco... TERA” [10], Cador uses the metaphor of the maraca for the simulation of free particles in a box. This could be seen as a restriction of the right coupling of the mechanical and acoustic systems only allowing for the generation of noise given the number of vibrating bodies involved. Separation of the two systems provides the means for more freedom to generate alternative acoustic signals but introduces the need for a mapping system to bridge the gap. Mapping of simulation data circumvents the need for restricting vibrating masses to an audible frequency range in order to produce an acoustic signal. Instead, scaling the object attributes means any data value can be made audible or inaudible, regardless of the object's state and behaviour. Considering this alternative approach we are presented with a visual metaphor that fits our everyday observations but the sound representation has become more subjective. That is to say the user must now decide how the data dimensions should be linked to the audio-dimensions.

The mapping of the objects serves to reflect the musical decisions made by the user making it difficult to produce general metaphors that are valid in any context. Those from another cultural background could judge differently what may be coherent and intuitive. There have been attempts to create online databases [14] suggesting mappings based on experimental evidence although it is widely accepted that an effective mapping can’t always be predicted [5][13]. A heuristic approach to this area should be adopted to allow for a compositional process that encourages experimentation in order to express creativity where the audience can reflect on the implications of a musician’s cultural and physical experiences. After interviewing physicists at CERN [5], the authors discovered that strong metaphors emerge from their professional experience. They found that more mapping associations were suggested for the well-known particles and fewer for the rarer particles proposing that perhaps this arose from fewer encounters, lack of interaction, and therefore becoming less prominent in the mind. They also observed that everyday properties such as mass were cited more often than abstract ones. This would imply that an object visually described through a recognisable metaphor, encapsulating everyday properties, can be easier to map. Our experience with physical objects allows us to inherently determine complex data relations. Properties are implied by a rigid body’s response to collisions with its surroundings. By referring to the visual behaviour during this event we perhaps reduce the need to refer to the linking of the parameters for interpretation.

Walker [15] states that in order to achieve an effective mapping choice one must go beyond that of polarity and linear scaling functions while avoiding restrictions placed on the user through bad design. The mapping window controls were devised to encourage flexibility by employing a messaging system (based on the OSC and MIDI protocols) to allow the user to map exposed parameters to potentially any input of a synthesiser and increase sonic diversity. In conjunction with these GUI controls (Figure 2) I created a function editor that serves to display the relationship between the two dimensions. The editor itself contains two permanent breakpoints that define the input domain (x axis) and the output range (y axis). Further breakpoints can be added and removed in order to construct a bijective mapping curve or polyline. The curvature of the segments can also be configured in order to account for both a linear and non-linear response.

![Figure 2: The mapping window](https://example.com/image2)

3. CONCLUSION

This work sets out to add to the range of tools for interacting, composing and experimenting with a data set using a combination of the visual and aural modalities. By utilising mainstream open source physics engines we have access to highly optimised algorithms that govern inherently recognisable objects where the data provided by the unchanging laws of classical mechanics serves as common ground for sharing and communicating ideas. Separating the audio from the mechanical system introduces new ways of interpreting data dimensions where, along with a flexible mapping system, we can choose to highlight or ignore individual streams in order to discover new ways of creating music.

4. REFERENCES


SUAC STUDIO REPORT
Yoichi Nagashima
Shizuoka University of Art and Culture
2-2-1, choo, naka-ku, Hamamatsu, Shizuoka JAPAN

INTRODUCTION
SUAC (Shizuoka University of Art and Culture) was founded in April 2000. Hamamatsu City, Shizuoka Prefecture, where SUAC is located, is situated almost midway between Tokyo and Osaka, and has numerous globally famous companies [Yamaha, Roland, Kawai, Suzuki, Honda, etc.]. SUAC has two faculties and six departments - the Faculty of Cultural Policy and Management (three Departments) and the Faculty of Design (Dept. of Industrial Design, Dept. of Art and Science [1] and Dept. of Space and Architecture). We have a WS room, a Mac room, as well as many Studios and Ateliers. The 2004 International Conference on New Interfaces for Musical Expression (NIME04) was hosted by SUAC [2]. We also organized the Media Art Festival (MAF) from 2001 to 2010 [3-5]. For example, the first MAF2001 featured the following: (1) two Live Computer Music Concerts with 12 composers, (2) a Symposium of IPSJ (Information Processing Society of Japan), (3) an Installation Gallery exhibiting 15 works, (4) A movie theater with 14 works, and (5) a CG Gallery exhibiting 12 works. I presented the paper "Students' projects of interactive media-installations at SUAC" at NIME06 [5]. We also organized three workshops about "Sketching / Physical Computing" at MAF2008-2010 with Prof. Shigeru Kobayashi, who is well-known as the inventor of Gainer/Funnel. In this paper, I will mainly report on four recent student's projects, because space limits. There are also some interesting research projects from professors of media psychology.

ABSTRACT
This is a studio report of student projects at SUAC (Shizuoka University of Art and Culture) in Hamamatsu Japan. SUAC was founded in April 2000, and hosted NIME04. All students in the "Faculty of Design, Department of Art and Science" are studying computer sound, interaction and media arts. SUAC organized the Media Art Festival (MAF) from 2001 to 2010, and many student's works have been created - live performances, installations and interactive contents - Flash, Web, Game, etc. In this paper, I will report on four recent student projects - two live performances and two sound installations created 2010-2012.

1. INTRODUCTION
SUAC (Shizuoka University of Art and Culture) was founded in April 2000. Hamamatsu City, Shizuoka Prefecture, where SUAC is located, is situated almost midway between Tokyo and Osaka, and has numerous globally famous companies [Yamaha, Roland, Kawai, Suzuki, Honda, etc.]. SUAC has two faculties and six departments - the Faculty of Cultural Policy and Management (three Departments) and the Faculty of Design (Dept. of Industrial Design, Dept. of Art and Science [1] and Dept. of Space and Architecture). We have a WS room, a Mac room, as well as many Studios and Ateliers. The 2004 International Conference on New Interfaces for Musical Expression (NIME04) was hosted by SUAC [2]. We also organized the Media Art Festival (MAF) from 2001 to 2010 [3-5]. For example, the first MAF2001 featured the following: (1) two Live Computer Music Concerts with 12 composers, (2) a Symposium of IPSJ (Information Processing Society of Japan), (3) an Installation Gallery exhibiting 15 works, (4) A movie theater with 14 works, and (5) a CG Gallery exhibiting 12 works. I presented the paper "Students' projects of interactive media-installations at SUAC" at NIME06 [5]. We also organized three workshops about "Sketching / Physical Computing" at MAF2008-2010 with Prof. Shigeru Kobayashi, who is well-known as the inventor of Gainer/Funnel. In this paper, I will mainly report on four recent student's projects, because space limits. There are also some interesting research projects from professors of media psychology.

2. "CANON"
A live computer music performance, "CANON", was composed and performed by Momo Imamurra (4th year student) in 2010-2011. The title "CANON" refers not only to the musical style "Canon" but also to "Flower Sound" in Japanese. There were four big "flower" objects on the stage. Each flower object had a speaker, foot-switch and lighting system whose brightness was controlled by the loudness of the speaker sound (Figure 1). This work was composed with Max5 environment, and performed in the "Inter-college Computer Music Concert" (Showa College of Music) in December 2010 and the Graduate Exhibition of SUAC in February 2011.

When the performance started, the performer (composer herself) played the Tenor Sax with one foot-switch of the "flower" object. Then, a performed Sax phrase (ad lib) is sampled in realtime into the system, and Sax sound is generated(processed) with harmony, repeat, arpeggio and echo, and the "flower" object sounds and lights and then the first part repeats with some interval timing. This sampling step means "I am giving water to this plant, and the flower starts singing". Next, the performer plays and samples the other phrase to another "flower" object, then the processed second part is generated by synchronized timing with the first part as musical style of "Canon". After she plays/samples the third part and the fourth part, all four "flower" objects generate and play each part like "Canon". Finally, the performer stands at the centre of flowers, and plays improvisation with BGM/lighting of four "flower" objects, a complete quintet ensemble.

3. "JAMI-GIRLS’ BAND"
This was a special collaboration-education project of 5 first year students and myself in 2011-2012. Firstly, I got many "junk" Jaminators in e-auctions at very low prices. Then, the students (Ayano Kazuma, Chika Suzuki, Yuriko Tosaya, Mai Morikawa and Akiko Yamada) and I opened the Jaminator, removed the parts and analyzed the system (Figure 3).

When the performance started, the performer (composer herself) played the Tenor Sax with one foot-switch of the "flower" object. Then, a performed Sax phrase (ad lib) is sampled in realtime into the system, and Sax sound is generated(processed) with harmony, repeat, arpeggio and echo, and the "flower" object sounds and lights and then the first part repeats with some interval timing. This sampling step means "I am giving water to this plant, and the flower starts singing". Next, the performer plays and samples the other phrase to another "flower" object, then the processed second part is generated by synchronized timing with the first part as musical style of "Canon". After she plays/samples the third part and the fourth part, all four "flower" objects generate and play each part like "Canon". Finally, the performer stands at the centre of flowers, and plays improvisation with BGM/lighting of four "flower" objects, a complete quintet ensemble.

As creators, students produced movies and images for graphic part of the performance, and we recorded many sounds from mobile phones as sonic materials [8]. The performance was a kind of battle-session game of sounds and graphics on stage. The graphic part was projected to a big screen behind the stage. The duration of this performance was about 7 minutes.

4. "SOUNDVOROUS PLANT"
This work is a sound installation produced by Chiaki Ikuma, a 3rd year student. Figure 6 shows this work, and the title means a "sonic version" of "insectivorous plant". It has an old-fashioned record player box and a big flower object connected to the box. This flower eats "sounds" with a microphone at the centre of it, and...
makes the "bech" sound converted from incoming sound. The system is written in Max/MSP environment and using GAINER to detect the movements (directions and speed) of the turntable controlled by people.

5. "OTOKAKECCO"

The installation work "OTOkakecco" was produced by Haruka Misaki as the final project of her master's course, and was exhibited at the SUAC graduation exhibition in February 2012. The title means Sound Hide and Seek in Japanese. Figure 8 shows the concept sketch, and Figure 9 shows the work.

Figure 6. Installation work "Soundvorous Plant".

Figure 7 shows some sketches of this idea. The creator tested many materials to produce the flower petals. Finally, she used a vacuum forming machine to press two panels of vinyl chloride, and Japanese traditional thin paper with red color. This work was specially made of two panels of vinyl chloride, and the sound is specially designed like a "Shepard Tone" - the infinite scale without pitch jump in the ring.

This project took about one year. To begin with, she tested the layout (size of rings and balls and heights with human) simulation with 3D animation software (Fig. 10), and tested the lighting interaction simulation and produced Shepard Tones before production.

People turn the turntable to the right while saying/singing, and these sounds are "eaten" by the flower. Then, after stopping the turntable once and people re-turn the turntable again, the flower generates a strange processed sound from sampled sound. If the turntable playback direction is right, the generated sound is modulated. If the playback direction is left, the sound is reverse-effected.

Figure 8. Concept sketch of "OTOkakecco".

Figure 9. Installation work "OTOkakecco".

In the center of SUAC's special big dark ball, there are three floors of big iron rings (2 meters in diameter), set at a height of 1 meter, 1.5 meters and 2 meters from the ground. Each ring has seven plastic balls (24 cm in diameter) which light seven types of colors. In each ring floor, there are seven different colors on balls and these colors change at 5-10 second intervals. Normally, the brightness of each ball is very weak.

People can hit the balls - from inside or outside of the ring, and then the ball flashes at maximum brightness for a moment, and a diatonic scale sound is generated related to each color. Each ring floor has a different timbre, and the sound is specially designed like a "Shepard Tone" - the infinite scale without pitch jump in the ring.

After the simulation processes, she started production. The control system was written by Max/MSP/jitter, and I produced the special interface system as a common tool for interactive systems for students. Figure 11 shows this interface, it controls 63 LEDs (3 colors * 7 balls * 3 floors) outputs with individual PWM from MIDI, and 21 Analog inputs from small microphones inside the balls to MIDI. Each ball contains two electronic boards of two high-power RGB LED, small microphones and OP-amp. MIDI. Each ball contained two electronic boards of two high-power RGB LED, small microphones and OP-amp. Cables were about 200 meters, and the main rings were hung with hard steel wires.

Figure 10. Layout simulation by 3D studio Max.

6. CONCLUSION

In this paper I have introduced some projects by SUAC students. The environment/platform of these four works is Max/MSP/jitter, but there are also many works with "Flash+Gainer", "Processing+Gainer" and "Propeller Processor", of course. The work of the upperclassmen informs the output of the underclassmen. The mentoring of the younger students also provides the upperclassmen with life skills and personal growth outside the classroom. At SUAC, there are many exciting collaborative projects by - [students and teacher] and [teachers]. It is also possible to design psychological experiments using our students as experimental subjects as they have good visual/aural senses. In a future paper, I would like to report on media psychology experiments at SUAC [10-11].

7. REFERENCES

PRESENTING ‘COSMOSF’ AS A CASE STUDY OF AUDIO APPLICATION DESIGN IN OPENFRAMEWORKS

Sinan Bökesoy
Doğan Bey Sok. 34971
Büyükada-Istanbul

ABSTRACT

Since the introduction of the open source toolkits for multimedia interaction programming, artists were encouraged to start to develop their tools in C++. The C++ language has considerable advantages in performance, gain and other frameworks such as Openframeworks¹ and Cinder² are C++ programming toolkits allowing us to encounter an interesting combination of artistic approach and programmer’s perspective/design. Cosmosƒ¹ is a stochastic sound synthesis engine, which does integrate a bottom-up sonic organization into a top-down organizing event generation system with a complex modulation routing and recursive audio structure. Cosmosƒ is made within Openframeworks. This paper introduces how its structural components are integrated within OF. The purpose is to share this experience with the computer music community as a case study.

1. INTRODUCTION

1.1. Cosmosƒ: advanced stochastic synthesizer

Cosmosƒ¹ is a real-time dynamic stochastic synthesis engine, which does generate sonic textures, where discrete sonic events of certain density are distributed in a time space with their onset time and duration parameter calculated with stochastic/deterministic functions. Each macro event defines the duration of a meso-space, and the sub micro events are distributed inside it. The overall goal is to achieve control on each event space and perform the process of change on the appropriate operation level. The user can intervene with the system in real-time on different time scales by inputting a sound source or accessing different type of synthesis/modulation generators by controlling the parameters for the sonic event distribution. (Figure 2)

It has a recursive structure with an audio feedback loop, offering emergent sonic behavior within a hierarchy of multiple time scales. The output of the system is fed back again to the input, as the micro-event audio data.

1.2. Audio Coding in Openframeworks

One can find the existing C++ frameworks being relatively poor when considering the audio programming features offered in their bundles compared to MaxMSP³, Csound⁴ etc. However with some existing C++ Synth Tool kits or various open source DSP code snippets, 3rd party expansions are always possible. For instance Cosmosƒ DSP code is based on the functions provided by Maximilian C++ Synth Tool Kit [2], an MIT licensed library. The offMaxim add-on for OF is simply wrapping the Maximilian C++ Toolkit.

Openframeworks[3] gives the opportunity of easily accessing the system audio by defining an object of.OutputStream class and by setting the basic parameters bufferSize, nChannels and sampleRate to initiate it:

```cpp
OutputStream setUpOutputStream(2, 0, sampleRate, initialBufferSize, 4);
```

¹ Jannis Xenakis has gathered his pioneering ideas of stochastic music in "Formalized Music: Thought and Mathematics in Composition" 1971.

On Figure 3, the communication of the program sections and classes are given. The setup() method initializes the program parameters.

The update() method updates all the program variables in interaction with the user interface and calls the necessary update routines such as the event distribution mechanism again in accordance to the parameter input of the user. The LFO’s and stochastic LineGen’s are generated inside the audioRequest() method and are operating with micro and meso event audio processing classes CosmosCell and CosmosCellM. The stochastic function generator ClassStochF is called by the event generation mechanism and by the modulation sources.

3. C++ TRANSLATION OF THE STRUCTURE

3.1. Event distribution process

On Figure 3, the communication between the code segments organized as C++ classes.

Figure 2. Event generation (top-down) and audio routing (bottom-up) in Cosmosf

Figure 3. The communication between the code segments organized as C++ classes.

Figure 4. The Cosmosf cycle is a circular field, where all the events are visualized and positioned with their onset and duration time references shown on concentric circles. The full circle signifies the cycle length like the clock of the day.

The micro and meso space parameters are assigned with the user interface elements and then the stochastic functions calculate the onset and duration of each event in these spaces. For instance the CellLD class, which generates the micro events inside a meso event has the following significant methods for its operation:

```cpp
void CellLD::setup(int Length, int density, int OnsetOffset, int
OnsetDuration, int ChannelSelect) {};
```

void testApp::audioRequested(float*output, int bufferSize, int nChannels) { for (int i = 0; i < bufferSize; i++) {
```cpp
output[i*nChannels] = (for left channel audio sample); output[i*nChannels + 1] = (for right channel audio sample);
```

Therefore the for loop fills an output array with its size equals to 2*BufferSize for the stereo audio format.

2. DEFINITION OF THE COMPONENTS

Cosmosƒ has following program routines organized in classes of C++ language to maintain different processes inside the application. For instance,

```cpp
CellLD.cpp / CellLD.h : for generating the micro event data with stochastic functions.
CellLDMeso.cpp / CellLDMeso.h : for generating the meso event data with stochastic functions.
ClassStochF.cpp / ClassStochF.h : for generating the stochastic function calculations.
CosmosCell.cpp / CosmosCellM.h : for generating the micro event audio data.
CosmosCellM.cpp / CosmosCellM.h : for generating the meso event audio data.
```

The main control panel Cosmosf

Cosmosf has following program routines organized in classes of C++ language to maintain different processes inside the application. For instance;

CellLD.cpp / CellLD.h : for generating the micro event data with stochastic functions.
CellLDMeso.cpp / CellLDMeso.h : for generating the meso event data with stochastic functions.
ClassStochF.cpp / ClassStochF.h : for generating the stochastic function calculations.
CosmosCell.cpp / CosmosCellM.h : for generating the micro event audio data.
CosmosCellM.cpp / CosmosCellM.h : for generating the meso event audio data.

LFO.cpp / LFO.h : Low frequency modulation sources program code.

LinearGen.cpp / LinearGen.h : Stochastic linear modulation generators in Xenakinian way¹.

Figure 3. The communication between the code segments organized as C++ classes.

3. C++ TRANSLATION OF THE STRUCTURE

3.1. Event distribution process

On Figure 3, the communication between the code segments organized as C++ classes.

Figure 2. Event generation (top-down) and audio routing (bottom-up) in Cosmosf

Figure 3. The communication between the code segments organized as C++ classes.

Figure 4. The Cosmosf cycle is a circular field, where all the events are visualized and positioned with their onset and duration time references shown on concentric circles. The full circle signifies the cycle length like the clock of the day.

The micro and meso space parameters are assigned with the user interface elements and then the stochastic functions calculate the onset and duration of each event in these spaces. For instance the CellLD class, which generates the micro events inside a meso event has the following significant methods for its operation:

```cpp
void CellLD::setup(int Length, int density, int OnsetOffset, int
OnsetDuration, int ChannelSelect) {};
```

The update() method updates all the program variables in interaction with the user interface and calls the necessary update routines such as the event distribution mechanism again in accordance to the parameter input of the user. The LFO’s and stochastic LineGen’s are generated inside the audioRequest() method and are operating with micro and meso event audio processing classes CosmosCell and CosmosCellM. The stochastic function generator ClassStochF is called by the event generation mechanism and by the modulation sources.

3. C++ TRANSLATION OF THE STRUCTURE

3.1. Event distribution process

On Figure 3, the communication between the code segments organized as C++ classes.

Figure 2. Event generation (top-down) and audio routing (bottom-up) in Cosmosf

Figure 3. The communication between the code segments organized as C++ classes.

Figure 4. The Cosmosf cycle is a circular field, where all the events are visualized and positioned with their onset and duration time references shown on concentric circles. The full circle signifies the cycle length like the clock of the day.

The micro and meso space parameters are assigned with the user interface elements and then the stochastic functions calculate the onset and duration of each event in these spaces. For instance the CellLD class, which generates the micro events inside a meso event has the following significant methods for its operation:

```cpp
void CellLD::setup(int Length, int density, int OnsetOffset, int
OnsetDuration, int ChannelSelect) {};
```

The update() method updates all the program variables in interaction with the user interface and calls the necessary update routines such as the event distribution mechanism again in accordance to the parameter input of the user. The LFO’s and stochastic LineGen’s are generated inside the audioRequest() method and are operating with micro and meso event audio processing classes CosmosCell and CosmosCellM. The stochastic function generator ClassStochF is called by the event generation mechanism and by the modulation sources.

3. C++ TRANSLATION OF THE STRUCTURE

3.1. Event distribution process

On Figure 3, the communication between the code segments organized as C++ classes.

Figure 2. Event generation (top-down) and audio routing (bottom-up) in Cosmosf

Figure 3. The communication between the code segments organized as C++ classes.

Figure 4. The Cosmosf cycle is a circular field, where all the events are visualized and positioned with their onset and duration time references shown on concentric circles. The full circle signifies the cycle length like the clock of the day.

The micro and meso space parameters are assigned with the user interface elements and then the stochastic functions calculate the onset and duration of each event in these spaces. For instance the CellLD class, which generates the micro events inside a meso event has the following significant methods for its operation:

```cpp
void CellLD::setup(int Length, int density, int OnsetOffset, int
OnsetDuration, int ChannelSelect) {};
```

The update() method updates all the program variables in interaction with the user interface and calls the necessary update routines such as the event distribution mechanism again in accordance to the parameter input of the user. The LFO’s and stochastic LineGen’s are generated inside the audioRequest() method and are operating with micro and meso event audio processing classes CosmosCell and CosmosCellM. The stochastic function generator ClassStochF is called by the event generation mechanism and by the modulation sources.
The phasor points to the absolute time since the beginning of each cycle. The phasor frequency is indeed the inverse of 1 Cosmosƒ cycle duration. If the current cycle length is defined with the floating variable Loop then the Maxpoint variable in the below expression gives us the current phasor point indicating the absolute time in the cycle.

\[
\text{Maxpoint}=\text{mainCounter}\times\text{phaser}(1000)\times\text{Loop}+1,\text{Loop}+1;
\]

When the Maxpoint reaches the end of the cycle, new meso events and micro events are distributed for the next Cosmosƒ cycle.

\[
\text{if}\left(\text{Maxpoint} \geq \text{Loop}+1\right) / \{ / \text{if it is in end of the loop,}
\]

regenerate the cell distribution...

CellMesoGenTripl() // generate the mesocell distribution
CellMicroGenTripl() // generate the microcell distribution

3.4. Audio rendering in Cosmosƒ main program

After realizing the event distribution for the new cycle, the program transfers this data to the micro and meso event audio classes. This happens in cascade for loop’s. First the AudioMicros class instances, which generate the audio for the micro-events, will be updated. The parameters are;

- the feedback audio (the address pointer to the feed variable). It is calculated as;

\[
\text{feed}=\left(\text{output}[0]*\text{output}[1]\right)\times\text{feedback}\times\text{sample} + \text{sample};
\]

This expression tells us, that the stereo output of Cosmosƒ is being summed and multiplied with the feedback amount value set by the user and then added with the livelpump sample data maintained at dataBN.

- The phasor value pointing to the absolute time index.
- Various micro-cell parameters as floating point and integer arrays (only the addresses are passed).
- Synthesis parameters and LFO (6 LFOs for a micro event) parameter arrays.
- Pointers to the Cosmosƒ output buffers bufData and bufDataB (two buffers which records the output of Cosmos in turn).

\[
\text{for}\left(\text{int i} = 0;\ i < \text{numCell}\cdot\text{CellDensity};\ i++\right)\{\text{sumLo} = \text{sumLo} + \text{AudioMicros}[i].\text{update}\left(\text{feed}[2],\ &\text{point},\right)\};\text{CellParam}[i],\ &\text{CellParam}[0],\ &\text{GrnParam}[i],\ &\text{FilParam}[0],\ &\text{LFOValues}[0],\ &\text{bufData}[0],\ &\text{bufData}[0];\right)
\]

The next significant code in the CosmosCellM::update method is the if-then conditional loop, which compares the current phasor value with the onset time of the micro event inside the cycle. When they match, then the micro event DSP routine starts and generates the event audio until the end of the relevant micro event.

\[
\text{if}\left(\text{unsigned}\text{point} > \text{posLo}&&\text{unsigned}\text{point} < \text{posLo}+\text{posLo}+1\right)\{\text{DSP code}\}
\]

The posLo variable here is the event start time and the posLo is the event duration with the variable type long. There are various DSP routines and waveform generating functions inside the DSP code. For example to playback a loaded sample from the buffer with certain speed and start offset, we use the ofMaxSample object from the ofMaxMsd with quadratic interpolation. The code expression below is for the calculation of the Sample Start point considering the values from modulation sources like LineGen and LFO. Likewise if Cosmosƒ is not in recursive buffer playing mode; the beats play method, being a member of the ofMaxSample class, plays the sample with the Speedindex value from the sample buffer dataBufB.

\[
\text{dataBufB} = \text{speedup}\times\text{INDEX}\times\text{sizeBN}(1-\text{speedup}\times\text{sizeBN})\times\text{dataBufB};
\]

The expression Speedindex*INDEX*sizeBN(1- speedup*INDEX) calculates the revised playback rate value according to the demanded start offset point considering the sampleRate and the size of the sample as given with the sizeBN variable. Hence according the code, the playback rate decreases when the performed chunk size of the sample data decreases too.

4. CONCLUSION

Despite the difficulty of expressing/explaining such complex code structures, which is generally the case with C++, we had a simple overlook to the significant design features implemented in Cosmosƒ as a case study of audio application development in Openframeworks. More references of such these will encourage the composers to develop their own tools in such platforms.

5. REFERENCES


KRONOS AS A VISUAL DEVELOPMENT TOOL FOR MOBILE APPLICATIONS

Vesa Norilo
Sibelius Academy
Centre for Music & Technology, Helsinki, Finland
vnorilo@siba.fi

ABSTRACT
Kronos is a programming language and a compiler suite, recently enhanced with a visual front end. It is designed to facilitate programming of digital signal processors for music. Kronos patches can be compiled either for real time playback or into an intermediate language such as C++, for integration in several third party frameworks.

This paper introduces the visual patcher for Kronos along with productivity-enhancing aspects of functional program-ming and metaprogramming that are unique in the visual domain. Higher order functions, closures and captured variables are examined pertaining to visually pro-grammed signal processors. As a case study, a resonator bank synthesizer is built in the Patcher and subsequently exported as a component for iOS SDK.

1. INTRODUCTION
Many creative aspects of music technology involve pro-gramming tasks. Instrument and effect design as well as live electronics depend on the ability of the creative pro-fessional to fashion custom signal processors. There are several musical programming environments. Among them, Supercollider[4] is a good example of textual programming harnessed for musical DSP. Pure Data[7] offers a visual front end but is simplistic as a programming language. PWGL[3] features a proper language and visual interface, but the signal processing component is limited.

The aim of Kronos is to provide a powerful, easily approachable language and combine it with a visual user interface. The entire compiler suite is built around opti-mization algorithms that make it possible to write high-performance DSP programs in a high level language.

The rest of this paper is organized as follows; in Sec-tion 2, a brief description of the core language is given. In Section 3 the integration of textual and visual program-ming is discussed. Section 4 deals with translating user patches into code, with an iOS application as a case study. Finally, Section 5 concludes the paper and presents future avenues of research.

2. LANGUAGE OVERVIEW
The Kronos back end is a server application capable of compiling textual programs. Several compiler targets are supported, including real time processing. This section gives a brief overview of the language in order to contextu-alize the visual patching environment. For an extended discussion, the reader is referred to previous publications[5]. A full-featured functional language is provided. Ab-structions such as higher order functions and closures[2] are supported, allowing sophisticated constructs to be used.

2.1. The Functional Reactive Paradigm
A central efficiency aspect of signal processing algorithms is the multirate problem. Significant optimization can be achieved by updating parts of the algorithm at lower up-date rates. Kronos proposes functional reactive programming[8] as a solution to the multirate problem. Nodes can act as clock sources, causing the graph downstream to update synchronously. A priority system ensures that more im-portant clocks can override less important ones; a user inter-face slider can be a local clock source, but once the con-trol signal reaches the audio path, the audio clock should take over. The reactive system has previously been dis-cussed in detail[5].

The reactive aspect is designed to be ignored by the beginner and intermediate user. The compiler includes a factorizer capable of optimizing user functions for heterogenous signal rates with no effort on the part of the programmer.

2.2. Type System
Kronos offers an algebraic type system[2] to deal with data structure. In addition, data can be decorated with user types to denote semantical meaning. For example, a vector of two real numbers could equally well stand for a complex number or a stereophonic signal. However, very different behavior is expected in these two cases. When multiplying, we expect the complex number case to follow the laws of complex arithmetic. For a stereophonic signal, a more sensible result is an element-wise multiplication.

By utilizing polymorphism[2], the appropriate multi-plication can be selected automatically based on the type of the incoming signal. A function library can be designed so that the end user need not worry about data types, yet the functions behave as expected.

2.3. Generic Programming
Generic programming follows from the capabilities of the type system. At source level, Kronos programs are type-less and generic. For performance reasons they are stat-ically typed during execution.

This is accomplished by a specialization pass reminis-cent of the C++ template metacompiler[1]. When com-piling a program, the data types of the signal flow are in-ferred throughout, and appropriate forms of polymorphic functions are selected at each point.

This process can generate versions of “specialize: all” functions in the system for any given argument. The only requirement is that all the function calls the specialization candidate makes can themselves be specialized. Eventu-ally this requirement will propagate to the leaves of the call tree, where arithmetic and logical primitives tend to be found. Defining those for a certain user type can thus enable previously written libraries to accept the said type.

This is very useful due to the number of permutations encountered in DSP. A single filter function can be used for single or double precision, real or complex, mono or multichannel signals or any combination of these.

3. VISUAL USER INTERFACE
Since Kronos is aimed at musicians and music technol-o-gists in addition of programmers, a visual patching en-vironment is provided to lower the barrier to entry. The visual interface acts as a client, connecting to the back end server over OSC[9].

Every effort is made to provide feature parity between textual and visual programming. As certain tasks are more natural to perform in writing while others are better suited for visualization, programmers may choose which domain to work in.

3.1. Integration of Text and Patching
The leading principle of the visual interface is that any expression can be typed into a patch node. Whenever the user starts typing, the node creation window pops up.

The free text entry guarantees that all language con-structs are available to the programmer. However, it is just as confusing for the beginner as writing the entire program in textual form in the first place. This is addressed by the interactive suggestion list displayed during text entry.

Beginners can use the text entry widget like a live search box into a predefined menu hierarchy. Advanced users may wish to employ it more like a code completion aid. Further inlets can be added to the expression by in-serting an inlet prefix with $, such as $input. The token itself is displayed as a tooltip for the inlet.

This scheme makes it simple to enter literals and con-stants, as the user just type them in to obtain the de-sired node. Some examples of nodes are shown in Figure 1.

3.2. Lambda Arrow and Captured Variables
Higher order functions, anonymous functions and closures are essential to productive programming in the functional paradigm[2]. While well understood in textual format, their application to visual programming may be less obvi-ous.

The most common way to construct anonymous func-tions in Kronos is to use the lambda arrow syntax. It is an infix operator =>, where the left hand side defines func-tion arguments and the right hand side defines the body. A syntax such as $x => $x + $y can be read as x into square root of x.

This lambda arrow results in an anonymous function that takes the square root of a real number. Such a func-tion can then be passed to a higher order function like Al-gorithm.Map, which applies it to a vector of elements. Similar constructs are used for most iterative tasks.

An essential aspect of anonymous functions is the con-cpt of captured variables. Traditionally, anonymous func-tions can refer to variables outside of their scope. As a result, the patch lacks the concept of variables, a slightly different method of capturing is required.

Figure 1. Constant, literal, tuple and a function node with inlets.

3.1.1. Patching
Nodes with inlets can be connected to other nodes via patch cords familiar to users of visual environments. In the generated program, connections are represented by variables. The value at the outlet is assigned to a unique symbol. Subsequently, all inlets connected to the outlet are replaced by references to this symbol.

3.2. Embedding
A visual connection is a powerful representation of data flow. However, sometimes the data flow can be so obvious or trivial that visualizing each connection only reduces readability. As an alternative to patch cords, the Kronos Patcher allows user to embed nodes into each other. By dragging a node into an inlet, the contents of that node replace the inlet.

Figure 2 shows an example, where a small subpatch is embedded entirely within a single node. Embedded nodes can be detached with a shake gesture.

Figure 2. Embedded nodes as a part of an expression.
In the visual patcher, adding inlets to a lambda arrow expression results in a capture. A capturing inlet is demonstrated in Figure 3. A value from a slider and a lambda transformation form a closure which is then applied over a vector of numbers.

Figure 3. Example of a captured inlet in an anonymous function

3.3. Convenience Nodes

To assist in visual programming, several nodes are provided to help in situations where the desired effect is cumbersome to achieve with the visual interface.

3.3.1. Tuple

The Tuple node is created with a single inlet. Whenever something is connected or embedded into the socket, an additional socket is created automatically. A tuple data type is built from all the connected inlets.

3.3.2. Tie

Tie nodes are used to split algebraic types[2]. Created with the identifier Tie, it can deconstruct an algebraic type into distinct outlets.

3.3.3. Slider

A slider node allows the user to include an interactive control in the patch. For real time synthesis, the slider connects to the server back end via a dedicated OSC[9] method.

Figure 4. Slider, Tuple, Tie and a multi-output node embedded in a Tie node

4. CODE GENERATION

To run the patch, the user may issue a build command. The visual patch is then converted into a textual Kronos program, and sent to the server. The server will perform specialization, type inferential and reactive factorization as described in Section 2. Finally, executable code is generated.

Various code emitters can be added to the Kronos compiler back end. Currently, it supports immediate compilation over a vector of numbers for real time sound synthesis, as well as generation of portable code in C++. This is most useful for integration with third party software. As a case study, the integration of Kronos-generated C++ into a mobile iOS application will be examined.

4.1. Case study: an iOS Application

4.1.1. Designing a Patch

First, the design of a simple resonator bank synthesizer will be examined. The implemented filter is a typical two-pole biquad resonator. The outline of the patch is as follows:

- Compute coefficients from frequency and bandwidth
- Implement a Direct Form II biquad filter
- Use the resonator as a component of a filter bank
- A function to compute resonator coefficients is given in Figure 5. Three argument inlets are specified: signal, frequency and bandwidth. The signal is required to obtain the sample rate via Reactive::SampleRate. Standard resonator coefficients b1 and b2 are computed and passed as a tuple to the function root node Forms.

Figure 5. Computation of resonator coefficients from frequency and bandwidth

The implementation of a Direct Form II biquad is shown in Figure 6. It makes use of the previously defined coefficient computation routine, embedded in a Tie node to provide outlets for both coefficients. The filtering consists of two stages of unit delay into multiply-accumulate and recursion. The feed-forward section and peak gain normalization follow.

Finally, a filter bank is built out of the resonator. One slider will be used to adjust the bandwidth of all the filters, while a frequency control is assigned to each. A pseudo-random noise source will act as an excitation signal.

A lambda transformation is defined where the noise signal along with the bandwidth slider are captured (see Section 3.2.1). These signals partially specify a signal transform, leaving a single lambda argument, freq.

The subsequent function, Algorithm::Map, receives a vector of frequencies from a set of sliders embedded in a tuple node. The resulting vector is processed with the lambda transform to obtain four channels of resonant noise. The channels are summed by Algorithm::Reduce.

4.1.2. Integration with XCode

XCode is the tool used to develop iOS applications. The Kronos C++-emitter converts the visual patch to code, and produces an application delegate class containing the signal processor for the iOS target. This way, Kronos can be used to develop signal processing algorithms while the refined user interface design tools of XCode kit can be fully applied.

To provide audio I/O and user interaction, the generated app delegate is annotated with IBAction keywords that are recognized by XCode. This allows the user interface elements to be connected visually in XCode Interface Builder. In fact, one can create a fully functional iOS application without writing a single line of code.

Figure 6. Direct Form II realization of a biquad resonator

5. CONCLUSION

This paper has presented the current state of Kronos, a programming environment for musical signal processing. Its key features were discussed, and a visual user interface for comprehensive functional programming was presented. The system was applied to the task of generating a simple mobile audio application.

Many of the capabilities of Kronos are present in various software packages. SuperCollider[4] supports a wide range of high level programming constructs. Faust[6] can generate efficient C-code for various frameworks. Pure Data[7] has a visual patching interface. However, none of these are able to combine all the features. Further, the

6. REFERENCES

THE HISSTOOLS IMPULSE RESPONSE TOOLBOX: CONVOLUTION FOR THE MASSES

Alexander Harker, Pierre Alexandre Tremblay
CeReNeM
University of Huddersfield
Huddersfield, HD1 3DH, United Kingdom
ajharker@gmail.com/p.a.tremblay@hud.ac.uk

ABSTRACT

This paper introduces the HISSTools project, and its first release, the HISSTools Impulse Response Toolbox (HIRT), a set of tools for solving problems relating to convolution and impulse responses (IRs). Primarily, the aims and design criteria for the HISSTools project are discussed. The elements of the HIRT are then outlined, along with motivating factors for its development, underlying technologies, design considerations and potential applications.

1. THE HISSTOOLS PROJECT

1.1. The HISS

The term HISS (Huddersfield Immersive Sound System) refers simultaneously to the multi-channel loudspeaker system, the academic community using it and the research lab attached to it. The common thread is a focus on the composition and performance of electronic music. More details can be found on the HISS website [1].

1.2. HISSTools

1.2.1. Aims

The HISSTools project aims to create powerful modular tools to address specific issues related to the composition, performance and presentation of electronic music. In particular, this project aims to bring complex technologies to mid-level users, who would not otherwise have access to them. Pre-existing tools may not be accessible or suitable to many users. Typically, this applies to tools that enable the design of solutions to a number of specific issues of concern to HISS members. The toolbox can be downloaded from the HISS website [12].

1.2.2. Availability

Although the focus of the HISSTools reflect the research activities of HISS members, they engage with problems that affect the wider community. The goal is to further understand of DSP techniques, and widen the creative possibilities available to practitioners of electronic music. Thus, these tools are licensed for free use, and are fully open-source.

1.3. HISSTools Design Criteria

1.3.1. Lightweight / Comprehensive Modules

The HISSTools are intended to be used in a variety of contexts, as part of musical workflows that cannot necessarily be predicted in advance. Thus, minimal, lightweight implementations, comprehensively dealing with a well-defined and coherent task are called for. Solutions should be sufficiently powerful but also flexible so that the user can concentrate on their specific musical needs, rather than on underlying technical details. Modularity is key to supporting a variety of musical and technical needs with minimal development time. Modules should place minimal requirements on the user in terms of setup and learning curve.

1.3.2. Environment Appropriate Implementation

It is desirable to maintain consistency with the behaviour of the host environments, in order to facilitate easy integration into a pre-existing workflow. This concerns core engineering decisions such as memory-access/state management models as well as more superficial issues, such as object and method/message naming conventions.

1.3.3. Efficient

For real-time applications, peak CPU usage (rather than average or amortised values) is the limiting factor, even a momentary overload can result in audio dropout. Thus, all possible steps should be taken to minimise the use of resources. These include careful algorithm choice/design, minimising unnecessary recalculation, use of SIMD instructions, algorithmic approximation (where appropriate) and avoidance of branching within loops.

Memory efficiency is also a concern as even 64-bit spaces do not prevent excessive paging and out due to a lack of physical RAM. The consequent data starvation can be disastrous for real-time performance.

1.3.4. Stable and Reliable

Stability is such an obvious concern that one should not need to raise it, yet unfortunately it is common to encounter tools for electronic music that are suffer significant deficiencies in this area. Lack of stability can be catastrophic in performance, amounting to absolute and unrecoverable failure. For the HISSTools we distribute pre-release tools to a set of international ‘power user’ testers and carry out extensive internal unit testing, in order to ensure both application stability, as well as reliable, accurate results.

1.3.5. Pragmatic

A pragmatic approach has previously been advocated by the authors in [20]. We favour only solutions that are suitable straightforward for the end use, and for which the benefits significantly outweigh any drawbacks. This may mean rejecting theoretically optimal solutions in cases where they prove unsuitable for quick and practical deployment in a musical context.

2. THE HISSTOOLS IR TOOLBOX

2.1. Overview

HISSTools first public release is a set of tools for working with convolution and IRs in MaxMSP. This set of objects addresses tasks such as measuring IRs, spectral display from realtime input/buffers, and non-realtime convolution, deconvolution and inversion. Although the primary development for development has been MaxMSP (chosen for its overwhelming popularity amongst practitioners), much of the code-base is generic, so as to support porting to other environments.

These tools enable the design of solutions to a number of specific issues of concern to HISS members. The toolbox can be downloaded from the HISS website [12].

2.2. Toolbox Concept

The primary motivation for the HIRT was to provide tools for improving the concert presentation of electronic music (especially when combined with acoustic instruments). Any combination of room, loudspeaker, speaker position, and listener position will affect the sound heard by the listener. Acknowledging this was the focus of earlier research [28], in which the approach was to simulate the effect of the concert hall within the studio, so as to produce music that would not suffer unexpected and potentially highly detrimental alterations in concert. The reverse perspective of this approach is to treat the system in question so as to achieve results closer to those heard in the studio. This is the first goal of the HIRT project: to allow for loudspeaker/room corrections to be generated and applied rapidly and efficiently for use in a concert environment. By extension, such corrections can also be employed in a studio setting.

The second motivation for the HIRT was to improve the frequency balance of close microphone or pick-up capture. This work was inspired by work by Alban Basstein [3], who advocates a system based on the relative frequency profiles of two microphones at different distances from the instrument. The idea is to take a close capture of the instrument that is relatively free from issues of spill and feedback, and improve the frequency balance prior to electronic processing, thus achieving better tonal balance.

Whilst these motivations are well-defined, it was felt that specific solutions would be both inflexible for the given tasks, and also limited in terms of re-application. Thus, a highly modular design was preferred, suited to a large number of convolution and impulse response related problems. The HIRT is thus suitable for use in relation to several general problems, including multichannel IR measurement, multichannel convolution, FIR Design and crosstalk cancellation.

2.3. Underlying Technical Approaches

2.3.1. FFT-based Processing

The HIRT takes an FFT-based approach to convolution and deconvolution problems, using a 64-bit precision FFT for all frequency domain processing. This is an efficient means of calculating convolutions, but it also has important implications for deconvolution. Firstly, one must be aware that the circularity of the FFT means there is potential for time-aliasing, in which energy wraps around the end points of the FFT window. Some user awareness of this issue is thus required when using the HIRT.

This problem might be avoided by using a least-squares time domain deconvolution, which produces optimal results in the time domain for a given filter length. However, calculation times are impractical for even moderate filter lengths, whereas it is quite practical on modern computers to calculate FFT-based convolutions on inputs of several tens of seconds within a relatively short time (a few seconds at most). Furthermore, whilst a time domain approach results in filters that is optimal in the time domain sense, rather than in a frequency domain. On modern machines minimising filter length is not of great concern. Typically, we over-specify the filter length and control the optimality in the frequency domain, subsequently truncating and fading the resultant IR to remove near-zero coefficients. This gives direct control over the frequency specification of the filter, which is of highest concern for our applications.

2.3.2. Fast Deconvolution with Regularisation

Using an FFT-based approach, deconvolution between time-domain inputs is possible simply by taking the FFT of each input, performing complex division, and trans-
forming the result back into the time domain.\footnote{Generally the FFT is quite large as it is required to be at least large enough to contain the longer of the two inputs, plus padding to deal with phase shifts in the output.} It is often also appropriate to introduce a delay to the output in order to deal with non-minimum phase components, which result in a non-causal output.

However, depending on the divisor the convolver can result in extremely long output filter lengths, often with significant time-consuming if the FFT is not large enough to hold the result. Additionally, filter blow-up may result, in which the output is unusable due to very large coefficients. In fact, there is no guarantee that the output will be finite or causal, even given that the input is an impulse response that is both finite and also a measurement of a causal system. In order to ensure meaningful results, it is necessary to circumvent such problems by ‘regularising’ the deconvolution\footnote{This object is based on code from the pre-existing irconvolve and partconvolve objects from the MaxMSP Externals package\cite{AHarker Externals}.}. Using frequency dependent regularisation it is possible to reduce the output of the deconvolution in frequency extremes, or other deficient frequency regions. Thus it is possible to control the length of the resultant filter, as well as its accuracy in different frequency ranges, according to the area of interest.

Regularisation as proposed in\cite{17} is equivalent to convolution between the direct result of the complex division and a linear phase IR\footnote{The overall technique is similar to that in\cite{8}, and combines the benefits of both convolution and deconvolution (as proposed in \cite{17,18}).} (which adjusts the amplitude response, but not phase). As linear phase FIRs are symmetrical, they exhibit equal amounts of pre- and post-ring. Thus, the implicit convolution can result in undesirable pre-ring in the output. A rearrangement of the process is proposed in\cite{4} which makes this implicit convolution explicit. The linear phase convolution can then be replaced by a minimum phase convolution of the same amplitude response to reduce pre-ringing. The HRT allows both linear and minimum phase forms of regularisation.

2.3.3. IR Measurement

Currently, the preferred method of impulse response measurement for most applications is the exponentially swept sine (ESS)\cite{6}. The ESS takes a constant amount of time per octave, with longer sweeps resulting in improvements to the signal-to-noise ratio. This method offers a very high response time per octave, with longer sweeps resulting in improvements to the signal-to-noise ratio. This method offers a very high. We have achieved usable results using both coloured noise signals, and even a music programme (given a sufficiently long measurement period). Whilst the ESS signal offers very good measurement results, it is unpleasant to listen to, especially at high volumes, and thus it is not viable for use in occupied venues.

2.3.4. Frequency Smoothing

Several applications require that the overall frequency profile of an IR be corrected (for instance, room equalisation, which should not correct every detail of a room response, as this is equivalent to deverberation). In such instances, it is desirable to work on a smoothed approximation of the frequency response, rather than the response in raw form\cite{13,14}. The key formulation is convolution of the IR in the frequency domain, with a sliding Hann window, which increases in width as the central frequency increases. Thus, there is more smoothing (in a linear sense) at higher frequencies than at lower ones. This relates to perception in that the ear is more sensitive to small frequency differences at lower frequencies. In the HRT the size of window at each frequency bin is controlled by the following formula: \[ \text{ws} = \text{smoothh} \times 0.5 \times \text{smoothw} \times 0.5 \] where \( \text{smoothh} \) is the window size, \( \theta \) is the central frequency of the bin, and \( \text{smoothh} \) and \( \text{smoothw} \) are the smoothing amounts at 0Hz and the Nyquist frequency respectively. The unit in all cases is normalised frequency.

For this purpose we favour the dismissal of phase entirely, and smooth the power spectrum only. This is in contrast to the complex smoothing proposed in\cite{14}, which arguably confines phase and amplitude information. Using large filter IRs, we found this approach did not produce suitable results. This is also appropriate to our concerns which are biased towards correcting the amplitude spectrum over the phase spectrum.

2.3.5. Phase Alteration

For some applications it is desirable to create an IR with a known power spectrum, but with a controllable group delay, somewhere between the minimum and maximum possible values. The minimum and maximum group delay for a given power spectrum correspond to the minimum of maximum phase filters with the specified power spectrum. Exactly between these lies is a linear phase filter that has a constant group delay for all frequencies (and a symmetrical form). The HRT employs homomorphic prediction\cite{22} to produce minimum phase spectra from normal minimum phase input data. The homomorphic interpolation it is then possible to interpolate between minimum and maximum phase forms of a given filter\cite{16].

2.4. Object Overview

2.4.1. IR Measurement

\textit{imeasure} performs IR measurement of a Multiple Input Multiple Output (MIMO) system using a range of known signals. The ESS signal is the technically optimal method, but may not always be practical for use (see 2.3.3). Alongside the swept sine method, MLS and coloured noise signals are offered as more pleasant sounding signals, with close to optimal performance. Measurement parameters are fully customisable. The object internally handles the IR retrieval from the recorded measurement (either by deconvolution, or optionally by convolution with an appropriate inverse signal for swept sine measurements). N convolutions are necessary to model the first N harmonics. The \textit{irphase} calculates the deconvolution between two real-time input signals. The resultant IR represents an estimate of the convolution necessary to transform the reference signal into the measurement input. The assumption is that the two input signals have the same source, but are picked up through different systems. Possible applications are estimating IRs with any broadband signal (for instance, music in an occupied room), and deriving relative IRs for systems such as microphones.

In situations where only the overall frequency profile needs to be modelled, \textit{irreference} can smooth the spectra of the two signals before deconvolution. In this case phase information is dismissed, and two minimum phase spectra with the smoothed amplitude profiles of the recorded inputs are deconvolved to produce the output. This may be preferable to post-deconvolution smoothing (with the \textit{iraveragex} object – see below), especially in the case where the input signal is not particularly broadband.

\textit{iraverages} performs exponential sweeps to a buffer\cite{11} (and inverse sweeps for IR retrieval by convolution if desired). Sweeps can be saved to disk and IRs measured using any system capable of simultaneous PCM playback and recording in cases where it is not practical to run MaxMSP. The \textit{bufconvolve} object can later be used to do the appropriate deconvolution using a copy of the original sweep or its inverse.

2.4.2. Offline Convolution and Deconvolution

\textit{bufconvolve} performs both convolution and deconvolution in the frequency domain. In the later case, there are options for different kinds of regularisation.

\textit{irinvert} performs IR inversion. The single-channel case, this is essentially deconvolution of a Dirac delta function (a single sample spike in the digital domain) by an arbitrary input signal. This method is preferable to post-deconvolution smoothing for minimal latency filters (i.e. minimum phase), or creating custom filters with a specific phase specification.

\textit{irinlinorm} is the non-linear component retrieved from an ESS measurement relate to the individual harmonics of a system. If these are transformed appropriately, it becomes possible to model the non-linear system by a Hammerstein model which operates on consecutive powers of the input signal\cite{2,21}, rather than harmonic transformations. N convolutions are necessary to model the first N harmonics. The \textit{irinlinorm} object, takes N input IRs for the harmonics (taken with the \textit{irrmsignals} object) and returns N new IRs for use in a Hammerstein model.

2.4.3. Realtime Convolution

\textit{multiconvolve} is a single channel object for zero-latency convolution using a fixed partitioning scheme. This object combines time domain convolution for the early portion of an IR with more efficient FFT-based partitioning convolution for the latter parts of the IR\cite{7}. This object also supports multi-channel behaviour directly. There are two modes, the first of which is a parallel multi-
mode in which each input is convolved once only to drive the respective output. In true multichannel mode there are up to \( N^2 \) convolutions (one for every combination of individual inputs and outputs). This is appropriate for scenarios such as true multichannel reverbs, or crosstalk cancellation networks. For efficiency it is possible to specify IRs for only certain routings within the convolution network.

2.4.5. Visualisation

A comprehensive spectral display (e.g. the spectral shape of an IR), or a combination of the two. For real and windowing parameters dynamically alterable, and there are several display modes, such as peak hold and time averaging.

2.4.6. Misc

irtransaural - calculates impulse responses for stereo transaural crosstalk cancellation for a given system in the viable frequency range of the system in question. Finally, the resultant filters are converted to minimum phase to reduce latency. Optionally the filter may be truncated, faded and normalised at this stage (using \( \text{irfave}\)). Figure 1 shows the proposed approach in graphical form.

Figure 1. Loudspeaker/Room Correction

The measurement microphone is ideally totally flat, as the full system being measured (and corrected) includes the microphone. This is likely to become noticeable in the case that the microphone is more coloured than the loudspeaker/room combination. For our tests we have used a DPA 4006 as our reference microphone, which exhibits an exceptionally flat frequency response.

The smoothing parameters act as a means by which to control the tightness of the correction to details of the system response. This is mostly a matter of preference, and can be done by ear. The perceptual impact of using specific values is consistent across different rooms and loudspeaker combinations, and thus it is easy to choose suitable defaults. Regularisation parameters are somewhat harder to optimise, as appropriate values are highly dependent on the amplitude spectrum of the IR in question. In particular, overcorrection at high frequencies can be frustrating for the listener; and care is required, as the rolloff of the different systems measured varied substantially, so a single default set of values cannot be used effectively.

Some discussion of phase issues is relevant, especially as a minimum phase approach has been taken. This effectively rules out problematic ringing, but could arguably create phase issues between channels if the corrections for each loudspeaker were substantially different. This has not been an issue to date. We verified this in a studio setting by switching both channels, and stereo up to use the same mono correction IR rather than two separate correction IRs. Auditioning the two versions would reveal any phase issues in the stereo correction if present.

The problem of compensating for phase issues ‘correctly’ is also a complex one, and arguably one that has no clear definition across a wide listening area. In this case the phase spectrum is likely to vary wildly. One approach is to use a linear phase correction (thus inducing a latency of half the filter length), but at best that only leaves phase relationships between channels the same as with no correction. In the case that a suitable mixed-phase average impulse for correction could be generated and artificial tests suggest that the resultant filter is likely to exhibit an unacceptable and distracting pre-ring. Thus, whilst not theoretically optimal, our approach remains to use minimum phase correction of the amplitude spectrum only.

2.3. Close Acoustic Capture Correction

2.3.1. General Problem

For the presentation of pieces involving processing of a live instrument, it is often desirable for the capture for processing to be taken from close to the instrument, often with a directional microphone. This reduces the pick-up of unwanted reflections and improves clarity.
up of microphone spillage from other instruments, loud-speakers, extraneous noises, and room reverberation, as well as reducing the potential for feedback. Unfortunately, the sound from an instrument is rarely well-balanced at short distances, and the analysis from the microphone tend to exacerbate this problem. The proposed then (as in [3]), is to correct the close pick-up to emulate the results from a tonally optimal microphone type and position. This means a microphone with a frequency response placed such that the radiation patterns of the instrument have been become balanced in a natural sounding manner. The ideal result retains the tonal balance noticeably, better reflecting the expected sound of the instrument from a normal listener position.\textsuperscript{7}

3.2. Proposed Approach

The proposed approach for close acoustic capture correction is very similar to that for loudspeaker and room correction. However, only a single measurement is necessary. This is taken by recording the instrumentalist through two microphones or pick-ups to generate a relative IR (using the \textit{irreference} object). We follow [3] in asking the instrumentalist to play chromatic scales in loud and quiet dynamics across the range of the instrument. Before inversion smoothing is applied, either by setting the \textit{irreference} object to smooth the two recorded signals prior to deconvolution, or by using the \textit{iraverage} object upon retrieval of the IR (post the internal deconvolution of the \textit{irinvert} object), the final result is expected to be minimum phase, with no high level of smoothing as in a reverberant room, and a tendancy towards closer filtering. It is necessary to spend time adjusting the smoothing parameters for the several preferable results. For acoustic bass guitar lower levels of smoothing also resulted in more simulation of the body of the instrument, which was captured by the room reference mic, but not the piano pick-up. The regularisation parameters were also more sensitive to the time-inversion than for loudspeaker room correction, especially in the case of the bass guitar, where poor choice of parameters at low frequencies resulted in either a lack of apparent low-end, or an overly boomy correction. Despite the issues encountered, overall it was possible to reach good results quickly (within minutes), with the corrected signal being both sonically preferable to the uncorrected signal, and audible closer to the sound of the reference signal. The main concern for this instrument, where both microphones are expected to be minimum phase systems there is no compelling reason to use a linear phase filter, unlike in the case of room and loudspeaker correction.\textsuperscript{9}

4. FUTURE DEVELOPMENTS

Currently, correction applications require some manual adjustment of parameters. Whilst this may in future remain the best way to produce optimal results, it is also desirable to have methods of auto-regularisation that can estimate an appropriate regularisation parameter automatically as setup time for concerts is often time-pressed. Preliminary work has been undertaken in this area, but as yet without any firm outcomes.

Further improvements for correction applications might come from improvements to the smoothing algorithm (for instance a perceptually-based algorithm for calculating smoothing widths at a given frequency, using Bark or ERB scales [24]), a common pole modelling approach in which the aim is to analytically identify poles for correction see [10, 9, 19]) or a more relevant approach to phase issues. The addition of time domain deconvolution techniques to the toolbox is another avenue for potential exploration.

In terms of end-user access, plans include an advanced set of tutorials on correction techniques, porting to other environments (\textit{pd} superCollider and \textit{csound}), and the provision of compensation IIRs for common mid-range microphones (such as the AKG 414). The aim of the latter would not be particularly for real-time correction for concert usage, but rather as a means to compensate for the microphone during measurement, thus producing more accurate results when a less coloured microphone is unavailable. AudioUnit and VST plug-ins are under development for applying corrections in a DAW environment.

\textbf{References}

\textsuperscript{1} HISS Website. [Online]. Available: http://www.hiss.org
\textsuperscript{5} A. Farina, “Simulation of the body of the instrument, which was captured by the room reference mic, but not the piano pick-up. The regularisation parameters were also more sensitive to the time-inversion than for loudspeaker room correction, especially in the case of the bass guitar, where poor choice of parameters at low frequencies resulted in either a lack of apparent low-end, or an overly boomy correction.
\textsuperscript{6} Despite the issues encountered, overall it was possible to reach good results quickly (within minutes), with the corrected signal being both sonically preferable to the uncorrected signal, and audible closer to the sound of the reference signal. The main concern for this instrument, where both microphones are expected to be minimum phase systems there is no compelling reason to use a linear phase filter, unlike in the case of room and loudspeaker correction.
\textsuperscript{9} Further improvements for correction applications might come from improvements to the smoothing algorithm (for instance a perceptually-based algorithm for calculating smoothing widths at a given frequency, using Bark or ERB scales [24]), a common pole modelling approach in which the aim is to analytically identify poles for correction see [10, 9, 19]) or a more relevant approach to phase issues. The addition of time domain deconvolution techniques to the toolbox is another avenue for potential exploration.
MAXSCORE – CURRENT STATE OF THE ART

Nick Didkovsky
Rockefeller University
didkovsn@mail.rockefeller.edu,
www.algomusic.com

ABSTRACT

We present recent developments of MaxScore – an
mxj notation object for Max—and the environment
it is embedded in. Since the first presentation of
MaxScore at the 2008 ICMC in Belfast, the
software has gone through some major
development stages making it a prime choice for
music notation within the Max and Live (via Max
for Live) software environments. Besides providing
a growing set of well over 230 messages, which
communicate with the JMSL API, development has
focused on offering convenient tools for real-time
composition and notation in networked music
environments as well as graphical and microtonal
notation. The LiveScore/MaxScore editors are
central to our efforts—implementing different
microtonal notation modes (48TET, 72TET and
Just Intonation) as well as the Max patch-based
Scorepions plugin system.

1. INTRODUCTION

MaxScore is a Java Max object developed by Nick
Didkovsky, which works in conjunction with Max
abstractions created by Georg Hajdu [1][2]. Being
released in 2007 (with a precursor from 2005), it
represents the first integrated notation solution for
the Max programming environment and has
enjoyed a loyal following despite powerful
alternatives such as the Bach Project (http://www.bachproject.net). MaxScore renders to
enjoyed a loyal following despite powerful
represents the first integrated notation solution for
released in 2007 (with a precursor from 2005), it
the Canvas abstraction, which consists of a set of
nested Max patches (Fig. 1). While earlier versions
of MaxScore have used basic drawing shapes such
as line segments, polygons and arcs to represent
musical signs, the versions since 2009 employ
music font and use a more abstract definition of
music glyphs (e.g. note heads and clefs) as well as
music curves (e.g. ties, slurs, tuplets, ottava
alta/bassa). Since rendering is done in Max, music
glyph and curve messages can be intercepted and
reinterpreted by the Canvas, allowing for greater
flexibility and adaptability compared to “boxed”
solutions. The Canvas abstraction now
accommodates various music font maps (New
Aloisen, Jazz, Maestro and Opus) and microtonal
maps (for eighth-tone, twelfth-tone and Extended
Helmholtz-Ellis Jf Pitch notation) and is also
capable of exporting vector graphics in svg format.

Drawing is done with the Max lcd object to which
the processed music glyph and curve messages are
ultimately passed. These come out of MaxScore’s
first outlet while its second outlet is also used in
response to info messages which play an important
role in some scenarios in which note and/or staff
attributes need to be queried before a music glyph
is drawn. The third and fourth outlets pass
instrument output and sequence dumps as well as
echo the indexes of the notes transcribed by the
transcriber (Fig. 2).

MaxScore employs a hierarchical data format
where score, measure, staff, track, note and interval
elements (among others) form a tree. New
attributes (e.g. for the visibility of notes, noteheads
or stems etc.) are continually being added to
accommodate new scenarios. In addition, a user
can define an unlimited number of extra note
dimensions which can be used to control various
processes and drive playback devices.

To complement the functionality of MaxScore,
Ádám Siska has designed a number of Max
externals: sadam.rapidXML, sadam.base64 and
sadam.io among others [3]. As some crucial info
queries return XML code, Siska’s Max wrapper for
the efficient RapidXML parser is an indispensable
ingredient in most scenarios involving real-time
notation, while the other two objects facilitate the
communication of remote MaxScore instances—a
welcome feature in networked music performances.

Recently, a searchable dictionary was added in
order to facilitate programming with MaxScore
with its plethora of different messages.

2. LIVE

When Max for Live extension was released in 2010
by Ableton and Cycling ’74, it became obvious that
MaxScore could provide what numerous Live users
were desiring: a device for the notation of MIDI
clips. To this end, we have developed two
devices for display (LiveScore Viewer) and editing
(LiveScore Editor), and added two more devices (a
sample and a soundfont player) for microtonal
playback, which is not natively supported by
Ableton.

Figure 1. LiveScore Editor with floating palettes.

Table 1. Comparison between Live note lists and
MaxScore sequence dumps

Figure 2. Communication between the main components
of a MaxScore patch.

Figure 3. Screenshot of the LiveScore Editor.

Table 2. Communication between Live and MaxScore during transcription and updating. The flow is
top to bottom.

Figure 4. Communication between Live, Max and
MaxScore on the network.

Table 3. Comparison between Live note lists and
MaxScore sequence dumps.
Still, bi-directional communication between Live and MaxScore is challenged by an important issue which still awaits a solution: Live's note list only consists of 5 parameters (pitch, time, duration, velocity, and a muted flag), not enough to either store or reference additional attributes such as dynamics, articulations, staves etc. Pitch is stored as an integer value, which won't allow for microtonal deviation either [5]. Therefore, the creation of rich scores in LiveScore quickly becomes a one-way street where an embellished clip lives on outside the Live set, separated from its twin clip. We have brought this issue to the attention of Ableton, and are optimistic that a solution will be found in the future.

There are two playback modes: (1) Live’s playback which sends the clips’ note events to standard Live devices via the internal MIDI bus and is synced to MaxScore’s page turns and note flashes as well as (2) MaxScore’s playback engine which passes score events to a capable audio device via a non-standard format which was devised to accommodate microtonal pitches.

In order to synchronize with the Max and Live environments, JMSL’s scheduler is now being driven by an implemented JMSLMaxClock, used as JMSL-clock [6].

While developing the software, special attention was given to allowing several instances of the editor to coexist in one Live set. This particularly concerned potential chutter caused by using several instances of the same floating palettes (Fig 1.). We have implemented a most minimalistic mechanism that prevents a second instance of a palette to be opened and dynamically directs user actions to the most recent, front-most editor window, regardless of whether the palettes are children of this window or not.

2.2. Picster
Since MaxScore’s repertoire of score markings is rather basic, we added a new feature called “rendered messages” making the repertoire of text markings and graphics virtually unlimited. These messages, which can be either attached to notes, measure/staves or measures, consist of an index, position information and a string, which will be passed to the drawing engine (the Canvas abstraction in our case). As long as the string contains messages understood by the engine, they will be rendered and displayed in the score.

Rendered messages will be inserted and saved in JMSL scores marked up by various UserBean tags.

While developing the software, special attention was given to allowing several instances of the editor to coexist in one Live set. This particularly concerned potential chutter caused by using several instances of the same floating palettes (Fig 1.). We have implemented a most minimalistic mechanism that prevents a second instance of a palette to be opened and dynamically directs user actions to the most recent, front-most editor window, regardless of whether the palettes are children of this window or not.

2.3. Scorepions
Scorepions are Max patches consisting of MaxScore messages, which reside in a special folder inside MaxScore Lib and may be dynamically invoked by their parent patch—thus forming a plugin system similar to those available in most commercial music editors. Scorepions can be used to programmatically generate and process all elements of a score. Amongst the 15 plugins that are currently part of the MaxScore release, the Djster-Autobus Scorepion deserves special attention as it represents a revived and further developed version of Clarence Barlow’s legacy software AUTOBUSK—a program devised for algorithmic composition (http://www.musikwissenschaft.uni-mainz.de/Autobusk/). The Djster-Autobus Scorepion takes the non-real-time part of Barlow’s program and uses its algorithms to interactively fill measures with note events based on given parameter presets. His application has also been extended to accommodate the microtonal scales from the Scala archive (an archive with well over 4000 scales; http://www.huygens-fokker.org/docs/scales.zip) as well as complex additive meters. A detailed explanation of the underlying algorithms and their adaptation to Max is outside the scope of this paper.

2.4. Microtonality
JMSL natively supports quarter-tone notation. All types of notation (8th-tone, 12th-tone, Just Intonation) require remapping of music glyph messages in the Canvas. As an example we will now describe the implementation of the Extended Helmholtz-Ellis JI notation which was developed by Marc Sabat and Wolfgang von Schweinitz as a comprehensive approach to the notation of music in Just Intonation (JI) [7]. The aim is to encode harmonic relationships between notes in terms of their accidentals. Sabat and von Schweinitz have created the HE font with accidentals capable of representing a 61-limit harmonic space, but have themselves suggested to cap the system at 23-limit (or possibly even lower). Since JMSL has no built-in concept of harmonic relationships, ratios expressing these relationships need to be calculated each time a page is being rendered. Two attributes are taken into consideration: the key signature of a measure representing the fundamental as well as the pitch of a note in floating point precision. A lookup table is being used to determine the ratio corresponding to the cent interval between the fundamental and the pitch. Lookup tables can be generated according to different principles such as sensory consonance, harmonic entropy or, in our case, harmonic energy, a measure derived from Clarence Barlow’s concept of the harmonic space, but have themselves suggested to cap the system at 23-limit (or possibly even lower). Since JMSL has no built-in concept of harmonic relationships, ratios expressing these relationships need to be calculated each time a page is being rendered. Two attributes are taken into consideration: the key signature of a measure representing the fundamental as well as the pitch of a note in floating point precision. A lookup table is being used to determine the ratio corresponding to the cent interval between the fundamental and the pitch. Lookup tables can be generated according to different principles such as sensory consonance, harmonic entropy or, in our case, harmonic energy, a measure derived from Clarence Barlow’s concept of the harmonic space, but have themselves suggested to cap the system at 23-limit (or possibly even lower). Since JMSL has no built-in concept of harmonic relationships, ratios expressing these relationships need to be calculated each time a page is being rendered. Two attributes are taken into consideration: the key signature of a measure representing the fundamental as well as the pitch of a note in floating point precision. A lookup table is being used to determine the ratio corresponding to the cent interval between the fundamental and the pitch. Lookup tables can be generated according to different principles such as sensory consonance, harmonic entropy or, in our case, harmonic energy, a measure derived from Clarence Barlow’s concept of the harmonic space, but have themselves suggested to cap the system at 23-limit (or possibly even lower). Since JMSL has no built-in concept of harmonic relationships, ratios expressing these relationships need to be calculated each time a page is being rendered. Two attributes are taken into consideration: the key signature of a measure representing the fundamental as well as the pitch of a note in floating point precision. A lookup table is being used to determine the ratio corresponding to the cent interval between the fundamental and the pitch. Lookup tables can be generated according to different principles such as sensory consonance, harmonic entropy or, in our case, harmonic energy, a measure derived from Clarence Barlow’s concept of the harmonic space, but have themselves suggested to cap the system at 23-limit (or possibly even lower). Since JMSL has no built-in concept of harmonic relationships, ratios expressing these relationships need to be calculated each time a page is being rendered. Two attributes are taken into consideration: the key signature of a measure representing the fundamental as well as the pitch of a note in floating point precision. A lookup table is being used to determine the ratio corresponding to the cent interval between the fundamental and the pitch. Lookup tables can be generated according to different principles such as sensory consonance, harmonic entropy or, in our case, harmonic energy, a measure derived from Clarence Barlow’s concept of the harmonic space, but have themselves suggested to cap the system at 23-limit (or possibly even lower).
and denominator determining the direction (clockwise, counterclockwise) and amount (in terms of fifths) to move. Accordingly, up to three accidentals are being chosen in accordance to the primes contained in the ratio.

Ultimately, the growing repertoire of LiveScore’s microtonal models will allow users to create different representations of the same music either by instantaneously switching between views or by juxtaposing different ones. In an unpublished talk at the 2010 Bohlen-Pierce symposium in Boston, Hajdu showed a Max patch, capable of several representations of the same music written in the Bohlen-Pierce tuning. He coined those representations cognitive notation (notation based on the physicality of an instrument, e.g. tabulatures) and logical notation (notation capable of visually representing equal distances) (Fig. 9).

It is possible to create pdfs directly from the Editor. Two Java applications have been integrated via shell scripts (Mac OS X) or DOS commands (Windows): the Batik Rasterizer which turns svg files into bitmapped pdf files and PDF jar which combines separate pdf pages into a single document.

2.6. MaxScore Editor

At some stage in the development of LiveScore it became obvious that it would make sense to offer the editor’s convenience in the Max/MSP patching environment. The MaxScore Editor (a Max bpatcher) has the same features as its sibling but can also be connected with other instances of the Editor, locally or over IP networks, as well as with synthesizers and other playback devices. It also reacts to the same set of messages as the core MaxScore object.

Figure 11. A Max patcher can be used to connect MaxScore components (editor and two playback devices)

Figure 9. Alternate views of the same music (Beyond the Horizon for 2 BP clarinets and synth by Georg Hajdu) in Bohlen-Pierce tuning (top: cognitive notation; middle: performance notation; bottom: logical notation)

2.5. Printing

The MaxScore homepage features a section dedicated to music written by various composers, Nicolas Collins, Arne Eigenfeldt and Peter Votava, among them [9]. In the following paragraphs, we will present the piece Swan Song by Georg Hajdu, which was premiered at the 2011 Shanghai Electronic Music Week. An earlier piece of his, Schwer...unheimlich schwer for bass clarinet, violin, piano and percussion has been analyzed in an issue of the Contemporary Music Review on Network Music [10].

3.1. Swan Song

Like some of Hajdu’s earlier pieces, Swan song - [33] for cello and percussion is based on transcriptions of preexisting sonic materials: speech, music and noises. For this piece Hajdu chose the final scene of a masterpiece of Chinese cinema called Farewell, My Concubine by Chen Kaige, a movie that had a great impact on him when it was released in 1993. The movie revolves around a complicated love story and features scenes from an eponymous Peking Opera. Life and theater blend dramatically in the final scene.

Figure 12. Example patch for using MaxScore to send scores over an IP network.

Figure 13. Section from Swan Song with non-standard glyphs inserted via “rendered messages”.

The rendering of the transcribed materials by the cello and percussion, mimicking the voices and instruments of Peking opera, are accompanied by processed video from the movie as well as electronic and prerecorded sounds. The first two tracks of the master score are being used for real-time part extraction and sent to the players over the network, the third and fourth tracks for the control of audio and video playback and the fifth is a click track, synchronizing the musicians to the audio and video playback (Fig. 13). Like in Schwer...unheimlich schwer the musicians read their music off of computer screens, only in this case, the entire score is being sent at the beginning of the piece with the players using an AirTurn Bluetooth Page Turner to turn their pages.

4. ONLINE PRESENCE AND LICENSING OPTIONS

We are maintaining a WordPress CMS at http://www.computermusicnotation.com to promote MaxScore on the WWW. The website features pages for news, downloads, documentation, support, projects and is connected to a mailing list as well as a discussion forum. We are offering two JMSL license options at different prices: A JMSL license as well as the lower priced LIVE license, which will disable all Java-only features such as the standalone score editor and the JSyn API.

5. OUTLOOK

Since its first release in 2007, MaxScore has developed into an environment for music notation with considerable versatility and adaptability. The recent release of Max 6 opens an opportunity for a review of the existing code base. Two developments are particularly worth mentioning: JavaScript and dict. Cycling ’74 has finally created alternative to the 20 year old lcd object by Mozilla JavaScript engine we will consolidate the nested Canvas and Picter abstractions into large jsui objects. The dict package consists of a set of objects supporting a hierarchical data structure. By switching MaxScore’s XML output to dict’s JSON format we are expecting speed and efficiency gains when exchanging and processing score data. We
are also working on MusicXML import, which will add to MaxScore’s user-friendliness.

6. REFERENCES


THE MOBILE CSOUND PLATFORM

Victor Lazzarini, Steven Yi, Joseph Timoney
Sound and Digital Music Technology Group
National University of Ireland, Maynooth, Ireland
victor.lazzarini@nuim.ie
stevenyi@gmail.com
jtimoney@cs.nuim.ie

Damin Keller
Nucleo Amazonico de Pesquisa Musical (NAP)
Universidade Federal do Acre, Brazil
dkeller@ccrma.stanford.edu

Marcelo Pimenta
LCM, Instituto de Informatica
Universidade Federal do Rio Grande do Sul, Brazil
mpimenta@info.ufrgs.br

ABSTRACT

This article discusses the development of the Mobile Csound Platform (MCP), a group of related projects that aim to provide support for sound synthesis and processing under various new environments. Csound is best an established computer music system, derived from the MUSIC N tradition [1], developed originally at MIT and then adopted as a large community project, with its development base at the University of Bath. A major new version, Csound 5, was released in 2006, offering a completely re-engineered software, as a programming library with its own application programming interface (API). This allowed the system to be embedded and integrated into many applications. Csound can interface with a variety of programming languages and environments (C/C++, Objective C, Python, Java, Lua, Pure Data, Lisp, etc.). Csound compilation and performance is provided by the API, as well software bus access to its control and audio signals, and hooks into various aspects of its internal data representation. Composition systems, signal processing applications and various frontends have been developed to take advantage of these features. The Csound API has been described in a number of articles [4, 6, 7]. New platforms for Computer Music have been brought to the fore by the increasing availability of mobile devices for computing (in the form of mobile phones, tablets and netbooks). With this, we have an ideal scenario for a variety of deployment possibilities for computer music systems. In fact, Csound has already been present as the sound engine for one of the pioneer portable systems, the XO-based computer used in the One Laptop per Child (OLPC) project [5] The possibilities allowed by the re-engineered Csound were partially exploited in this system. Its development sparkled the ideas for a Ubiquitous Csound, which is now steadily coming to fruition with a number of parallel projects, collectively named the Mobile Csound Platform (MCP). In this paper, we would like to introduce these and discuss the implications and possibilities provided by them.

1. INTRODUCTION

Csound originated as a command-line application that parsed text files, setup a signal processing graph, and processed score events to render sound. In this mode, users hand-edit text files to compose, or use a mix of hand-edited text and text generated by external programs. Many applications—whether custom programs for individual use or publicly shared programs—were created that could generate text files for Csound usage. However, the usage scenarios were limited as applications could not communicate with Csound except by what they could put into the text files, prior to starting rendering.

Csound later developed realtime rendering and event input, with the latter primarily coming from MIDI or standard input, as Csound score statements were also able to be sent to realtime rendering Csound via pipes. These features allowed development of Csound-based music systems that could accept events in realtime at the note-level.
such as Cecilia [8]. These developments extended the use cases for Csound to realtime application development. However, it was not until Csound 5 that a full API was developed and supported that could allow finer grain interaction with the core engine. Applications using the API could now directly access memory within Csound, control rendering frame by frame, as well as many other low-level features. It was at this time that desktop development of applications grew within the Csound community. It is also this level of development that Csound has been ported to mobile platforms.

Throughout these developments, the usage of the Csound language as well as exposure to users has changed as well. In the beginning, users were required to understand Csound syntax and coding to operate Csound. Today, applications were developed that expose varying degrees of Csound coding, from full knowledge of Csound required to none at all. Applications such as those created for the XO platform highlight where Csound was leveraged for its audio capabilities, while a task-focused interface was presented to the user. Other applications such as Cecilia show where users are primarily presented with a task-focused interface, but the capability to extend the system is available to those who know Csound coding. The Csound language then has grown as a means to express a musical work, to becoming a domain-specific language for audio engine programming.

Today, these developments have allowed many classes of applications to be created. With the move from desktop platforms to mobile platforms, the range of use cases that Csound can satisfy has achieved a new dimension.

3. Csound for iOS

At the outset of this project, it was clear that some modifications to the core system would be required for a full support of applications on mobile OSs. One of the first issues arising in the development of Csound for iOS was the question of plugin modules. Since the first release of Csound 5, the bulk of its unit generators (opcodes) were provided as dynamically-loaded libraries, which resided in a special location (the OPCODEDIR or OPCODEDIR64 directories) and were loaded by Csound at the orchestra compilation stage. However, due to the uncertain situation regarding dynamic libraries (not only in iOS but also in other mobile platforms), it was decided that all modules without any dependencies or licensing issues could be moved to Csound core library code. This was a major change (in Csound 5.15), which made the majority of opcodes part of the base system, about 1,500 of them, and it is exposed so that developers can use methods not in the code. As a result of this choice, the APIs that are found on iOS devices come pre-wrapped and ready to use with Csound through CsoundObj.

To communicate with Csound, an object-oriented callback system was implemented in the CsoundObj API. Objects that are interested in communicating values, whether control data or audio signals, to and from Csound must implement the CsoundValueCacheable protocol. These CsoundValueCacheables are then added to CsoundObj and values will be then be read from and written to on each control cycle of performance (fig.1). The CsoundObj API comes with a number of CsoundValueCacheables that wrap hardware sensors as well as UI widgets, and examples of creating custom CsoundValueCacheables accompany the Csound for iOS Examples project.

4. Csound for Android

Csound for Android is based on a native shared library (libcsoundandroid.so) built using the Android Native Development Kit (NDK) 1, as well as pure Java code for the Android Dalvik compiler. The native library is composed by the object files that are normally used to make up the main Csound library (libcsound), its interfaces extensions (libcsound), and the external dependency, libsoundfile 2. The Java classes include those commonly found in the csnd.jar library used in standard Java-based Csound development (which wrap libcsound and libsoundfile), as well as unique classes created for easing Csound development on Android.

As a consequence of this, those users who are familiar with Csound and Java can transfer their knowledge when working on Android. Developers who learn Csound on Android can take their experience and work on standard Java desktop applications. The two versions of Java do differ, however, in some areas such as classes for accessing hardware and different user interface libraries. Similarly to iOS, in order to help ease development of CsoundObj class, here written in Java, of course, was developed to provide straightforward solutions for common tasks.

As with iOS, some issues with the Android platform have motivated some internal changes to Csound. One such problem was related to difficulties in handling temporary files by the system. As Csound was dependent on these in the compilation/parsing stage, a modification to use core (memory) files instead of temporary disk files was required.

Two options have been developed for audio IO. The first involves using pure Java code through the AudioTrack API provided by the Android SDK. This is, at present, the standard way of accessing the DAC/ADC, as it appears to provide a slightly better performance on some devices. It employs the blocking mechanism given by AudioTrack to push audio frames to the Csound input buffer (spout) and to retrieve audio frames from the output buffer (spout), sending them to the system sound device. Although low latency is not available in Android, this mechanism works satisfactorily.

As a future low-latency option, we have also developed a native code audio interface. It employs the OpenSL API offered by the Android NDK. It is built as a replacement for the usual Csound IO modules (portaudio,alsa,jack,etc.), using the provided API hooks. It works asynchronously, integrated into the Csound performance cycle. Currently, OpenSL does not offer lower latency than AudioTrack, but this situation might change in the future, so this option has been maintained alongside the pure Java implementation. It is presented as an add-on to the native shared library. Such an implementation will also be used for the future addition of MIDI IO (replacing the portmidi, alsamidi, etc. modules available in the standard platforms), in a similar manner to the present iOS implementation.

At the outset of the development of Csound for Android, a choice was made to port the CsoundObj API from Objective-C to Java. The implementation of audio handling was done in a similar fashion following the general design as implemented on iOS (although, internally, the current implementation differs in that iOS uses an asynchronous mechanism, whereas in Android blocking IO is used). Also, the APIs match each other as much as possible, including class and method names. There were inevitable differences, resulting primarily from what hardware sensors were available and lack of a standard MIDI library on Android. However, the overall similarities in the APIs greatly simplified the porting of example applications from iOS to Android development. Using MCP, the parity in APIs means an easy migration path when moving projects from one platform to the other.
Sound library, which is a standard part of modern Java runtime environments. JAWS Csound has been chosen as the sound engine for the DSP eartraining online course being developed at the Norwegian University of Science and Technology [2].

6. CSOUND 6

In February 2012, the final feature release of Csound 5 was launched (5.16) with the introduction of a new bison/flex-based orchestra parser as default. The development team has now embarked on the development of the next major upgrade of the system, Csound 6. The existence of projects such as the MCP will play an important part in informing these new developments. One of the goals for the new version is to provide more flexibility in the use of Csound as a synthesis engine by various applications. This is certainly going to be influenced by the experience with MCP. Major planned changes for the system will include:

- Separation of parsing and performance
- Loading/unloading of instrument definitions
- Further support for parallelisation

As Csound 6 is developed, it is likely that new versions of the MCP projects will be released, in tandem with changes in the system.

7. CONCLUSIONS

The Mobile Csound Platform has been developed to bring Csound to popular mobile device operating systems. Work was done to build an idiomatic, object-oriented API for both iOS and Android, implemented using their native languages (Objective-C and Java respectively). Work was also done to enable Csound-based applications to be deployed over the internet via Java Web Start. By porting Csound to these platforms, Csound as a whole has moved from embracing usage on the desktop to become pervasively available. The MCP, including all the source code for the SDK, and technical documentation, is available for download from

http://sourceforge.net/projects/csound/files/csound5

For the future, it is expected that current work on Csound 6 will help to open up more possibilities for music application development. Developments such as real-time orchestra modification within Csound should allow for more flexibility in kinds of applications that are possible to develop. As mobile hardware continues to increase in number of cores and multimedia capabilities, Csound will continue to grow and support these developments as first-class platforms.

8. ACKNOWLEDGEMENTS

This research was partly funded by the Program of Research in Third Level Institutions (PRTLI 5) of the Higher Education Authority (HEA) of Ireland, through the Digital Arts and Humanities programme.

9. REFERENCES


5. CSOUND FOR JAVA WEB START

Csound 5 has long included a Java wrapper API that is used by desktop applications such as AVSynthesis and blue. During research for a music-related project that required being deployable over the web, work was done to explore using Java as the technology to handle the requirements of the project, particularly Java Web Start (JAWS). The key difference between ordinary Java desktop and Java Web Start-based applications is that with the former, the Csound library must be installed by the user for the program to function. With the latter, instead, the application will be deployed, downloading the necessary libraries to run Csound without the user having anything installed (besides the Java runtime and plugin).

Regarding security, JAWS allows for certificate-signed Java applications to package and use use native libraries. Typically, JAWS will run an application within a sandbox that limits what the application is allowed to do, including things like where files can be written and what data can be read from the user’s computer. However, to run with native libraries, JAWS requires use of all permissions, which allows full access to the computer. Applications must still be signed, verifying the authenticity of what is downloaded, and users must still allow permission to run. This level of security was deemed practical and effective enough for the purposes of this research.

In order to keep the native library components to a minimum, JAWS Csound only requires the Csound core code (and soundfile access through libsndfile, which is packaged with it). Audio IO is provided by the Java-
EN ABLE THE PRESENCE: DYNAMIC VIDEO COMPRESSION FOR THE TELEMATIC ARTS
Benjamin D. Smith
University of Illinois at Urbana-Champaign
National Center for Supercomputing Applications

ABSTRACT
Telematic performance, connecting performing artists in different physical locations in a single unified ensemble, places extreme demands on the supporting media. High audio and video quality plays a fundamental role in enabling inter-artist communication and collaboration. However, currently available video solutions are either inadequate to the task or pose extreme technical requirements. A new solution is presented, vipr (video-image protocol), which exposes a number of popular, robust video compression methods for real-time use in Jitter and Max. This new software has successfully enabled several inter-continental performances and presents exciting potentials for creative, telematic artists, musicians, and dancers.

1. INTRODUCTION
Distributed, network performance, or telematic performance, is an exciting, growing area of aesthetic exploration. Musicians and dancers are engaging with remotely located counterparts on an increasing basis, testing the possibilities of live performance despite their physical separation. Distances both small and large are being overcome through the use of high-bandwidth networks, even enabling performances between ensembles distributed around the globe [2].

All live performance and inter-performer interactions require a sense of connection and presence in order to frame both verbal and artistic communication. Clear access to the movements, actions and sounds created by their remote partners is a prerequisite to coordinating expressive direction. Many media may be employed in facilitating this sense of presence, however video typically takes a pivotal role by relaying real-time footage between all of the remotely located participants. The quality of the video connection plays a significant role in creating this sense of presence and enabling (or denying) interactions between the performers [8].

While adequate audio technology has been available for some time [1] no satisfactory video software solution has previously been available. Typical performances employ easy-to-use programs such as Skype or iChat which provide poor performance quality, exhibiting large latencies, low color accuracy, frequent compression artifacts, low frame rates, and occasional connection loss. Other solutions use expensive, custom designed systems that require extensive expertise to operate. The setup of any system is further complicated by bandwidth requirements with drastically different capabilities depending on the available infrastructure.

Vipr for Max is a new, easy to use, yet powerful tool to enable live telematics performances, functioning as a dynamic image compression extension for Jitter, and is freely available for non-commercial use. This software provides a simple, robust interface to a variety of popular video codecs integrating seamlessly within the Max development environment. Vipr uniquely provides a dynamic interface to the codec configuration, allowing users to tweak the bit rates, frame sizes, and codec selection in real-time, enabling ready identification of optimal settings as well as new areas for aesthetic expression through dynamic video deconstructing and editing.

2. MOTIVATION
The first musical collaborations over network connections were seen in the 1980s [3], employing satellite connections to bring artists across the United States into communication. Around this time the League of Automatic Music Composers, and its offshoot The Hub, formed with the express intent of exploring the potentials of network-based music and art [7]. Since the turn of the twenty-first century, with widespread institutional access to high-speed networks, artists have begun exploring telematic performance with much greater frequency. This has resulted in a plethora of examples [2, 3, 4, 5, 10], bridging all the performing arts as they intersect or extend into distributed, networked spaces. Musical works set in the networked domain come from one of two different approaches: the first focusing on the computers and network topography, seeking to employ them as instruments for artistic creation [7], and the second focusing on the communicative aspect of networks and their ability to bring people together across large physical (and temporal) distances [10]. Many cases encompass both aspects, such as the work of Weinberg [11] and The Hub, however, the distinction between approaching computers as instruments versus enabling communication is significant.

Practitioners of the later have claimed the term telematic to denote works focusing on distributed, live performance that largely mimic conventional, western concert hall performance practices. These events typically involve ensembles comprising performing artists in two or more physical places connected by high-bandwidth networks relaying real-time audio and video. Thus the performers are able to engage with one another in collaborative performance, ideally transparently enabling co-present ensemble interactions. The technology to facilitate these performances is derived from computer-supported collaborative work systems and has the appearance of typical video conferencing setups.

However, fostering a sense of presence for the musicians requires a high degree of fidelity that extends beyond common telecommunications systems. Presence is the “the perceptual illusion of non-mediation” [8] and is required for the order of telematic performance to be successful. Further, Lombard and Ditton [8] identify a series of characteristics that are desirable for encouraging a sense of presence for the participants (who may be both performers or observers). These characteristics are: image quality, image size and viewing distance, motion and color, perceived dimensionality, and camera techniques.

Perceived image quality is reliant on many elements, including resolution, brightness, contrast, sharpness, color, and the absence of noise or artifacts. Higher resolution images tend to invoke greater presence, as do more photo-realistic images (which is a combination of accurate sharpness, color fidelity, contrast, etc.). Artifacts, typically resulting from image compression techniques, have also been found to decrease a sense of presence by drawing attention to the mediation of the experience.

Image size has similarly been shown to directly impact viewers sense of presence, where images filling a larger field of view typically increase presence. Considering this problem in terms of field of view recognises that small images seem up close (such as in virtual reality goggles) can be equivalent to large images seen at a great distance (cinema screens, for example). In combination with the desire for higher image quality this typically equates to higher resolution images which increases the capability to display large, sharp, high contrast content.

Communication between performing artists requires conveying a sense of physical motion (i.e. halting sequences of still images do not suffice) and human visual perception studies [9] indicate that a sense of motion is most effectively created at video frame rates in excess of 50 frames-per-second (FPS), or 20 milliseconds per frame. The perception of motion is also convolved with image size and field of view to create the illusion of continuity.

For performing artists the amount of delay created by each leg of the network connection further impacts the sensation of presence. Shorter delays are preferred, although longer delays can be used successfully in certain situations [2, 10]. These delays, termed latency, also have far reaching artistic ramifications.

Consideration of the sensation of dimensionality (perceiving a three dimensional space in a two dimensional image) and the use of appropriate camera techniques (such as framing and shot length) are important to creating a sense of presence, however these operate independent of the video transmission system and thus are not treated further here.

Existing video solutions present a myriad of problems for the telematic performer attempting to create a reliable sensation of presence. These revolve partly around tradeoffs between bandwidth availability and computing resources, but also involve the amount of technical expertise required for operation. The easiest and most commonly employed systems (such as Skype) provide a readily accessible solution for the non-technically expert musician, however the image quality, size, and frame rate are all inferior (providing highly compressed, 640x480 video at under 25 FPS).

Figure 1. Screen capture of vipr in Max 6.

Technically advanced systems (such as that employed by [5], Access Grid, and SAGE) require dedicated technicians as well as expensive hardware.
constructions. These solutions typically provide high resolution, high quality images (up to HD standard quality), but rely on access to very high-bandwidth institutional grade networks (intern2 and beyond). Mid-grade solutions (such as ConferenceXP, employed by [14]) still require a high degree of expertise, are operating system dependent, and provide poor hardware support.

3. Vipr

Vipr (fig. 1) was created to address the lack of flexible, accessible video transmission systems by providing access to a selection of popular compression (or codec) techniques within the Max/ vipr environment. This enables strong hardware support and independence, by leveraging Quicktime and DirectX, and takes advantage of the built in networking capabilities of Max. Vipr uses the open-source library ffmpeg, a very robust and highly successful video codec package that is continually incorporating new techniques and refinements, ensuring access to the best available compression models.

Within Max vipr operates as an independent object, compressing or decompressing individual matrices (i.e. images or frames of video) as they are sent to the object. The input matrices can be of any dimensions, in 32-bit color format. Vipr outputs a 1-dimensional matrix of image data that can then be stored or transmitted across a network to another instance of Max. When presented with an already compressed image matrix the external will decompress the image, outputting the original image. Processing takes place in real-time, yielding compression times on the order of milliseconds (depending on hardware and video image characteristics). Decompression is typically very fast (also on the order of milliseconds), and more extensive and precise metrics will be forthcoming. The resulting compression factor, or how many bits are saved by compressing the image, can be anywhere from 30% for lossless codecs (i.e. 70% of the original data space is saved) to 1% or less for lossy codecs (such as mpeg4).

Due to the lightweight nature of vipr it is easily possible to set up a networked installation where each client in a network broadcasts its stream to multiple receivers.

Unique to this implementation, as compared to any other available video streaming solution, vipr exposes all of the parameters of the codecs for real-time control within Max. This not only enables rapid stream optimization capabilities but also presents the potential for the artistic use of the compressors by intentionally pushing the system into unusual states (such as creating artifacts in the image by dropping or repeating frames of the video while changing the bit rate).

The lossless codecs made available by vipr (such as huffyuv, ffmpeg, and fhhuffy) present another unique option for the telematic artist. These codecs provide fully accurate image reproduction on the receiving end without any compression effects in the image. This is a significant advantage when high image quality is desired (in order to facilitate the sense of presence), and their real-time implementation is currently unique to vipr. While the processing load is slightly higher (requiring around 20 milliseconds for a 720p image on typical modern Mac hardware) the data reduction (on the order of 8:1, yielding a stream of 25.6 megabits-per-second for a 720p stream) enables transmission over networks that cannot support full HD video streams (which would typically require a 10 gigabit network or better).

3.1 Codec Parameters

The principle codec parameters exposed by vipr are now described. These are only set on the compressing side of the network, as the decompressing system can automatically detect the settings used for compression.

**Bit-rate**—This sets the target compression amount a lossy codec will attempt to achieve. The rate is not guaranteed, as most codecs (such as the mpeg series) use a variable compression scheme to take advantage of highly compressible scenes, and the final rate may be higher or lower than the target.

**Group Of Pictures (or gop)**—Lossy codecs typically employ a key framing technique in order to minimize compression artifacts in undesirable situations (where frames may be corrupted or dropped). The key frame is a single image with a fully descriptive encoding, allowing the original image to be constructed from the key frame alone. Key frames are typically sent as one out of every 10 frames or more, depending on the reliability of the network connection. The frames sent between the key frames are incremental in nature, requiring the previous image in order to recreate the source image. This typically results in greatly improved compression rates (as compression rates between two images in a sequence do not need to be retransmitted) yet is very susceptible to lost frames which will produce noticeable image quality degradation. The gop parameter sets the number of incremental frames to use between key frames.

**B-Frames**—While the incremental frames resulting from the gop setting only rely on the previous image state (and are termed i-frames), b-frames use a bi-directional model that compresses based on both the preceding and following images. This provides even greater compression rates. However, b-frames also require delaying the stream by the same number of frames, because compression cannot begin until the following image has been processed. While this is perfectly acceptable for prerecorded video sequences it may be undesirable in real-time settings that attempt to minimize latency in order to further the sense of presence.

**Rate:** This dictates how many bits are used for each color component in the image during the internal compression process. Most codecs only operate with a single rate setting, but one of a few specific pixel formats, while fitter uses a 32-bit format that is inefficient and not supported by many codecs. Thus vipr transparently converts the image internally for compression, returning it to a 32-bit format upon decompression. Typical internal formats use fewer than 32 bits, providing additional compression advantages, but shifting the color space slightly (which can be detectable in some situations). Coders that support multiple pixel formats allow the user to change the internal format, which may result in minor image quality improvements.

**Resample method**—In order to perform the pixel format conversion a resampling method is used, which may be selected by the user. vipr implements a variety of methods (including bilinear, fast bilinear, bicubic, point, area, and gaussian), which have measurable performance effects. Typically, fast bilinear provides optimal efficiency however other methods may be preferable depending on the content being converted.

4. DISCUSSION

Vipr provides customizations to enable the best recreation of presence possible for nearly any hardware configuration. Recently it has been used as a component in several inter-continental performances with great success [10] and is being employed as a key technology in a telematic opera production [4].

Finding the optimal parameter configuration is highly dependent on the desired results as well as the available computation resources (i.e. CPU time), network bandwidth, camera quality, and projection capabilities. At the moment general guidelines are not available, however this is a focus of ongoing research. Yet, the immediate response of the system allows even novice users to explore parameter settings in order to maximize quality within a given setup.

Adjusting for the preferred setup requires balancing image size, image quality, and frame rate with the resulting processing load and bandwidth allowances. Identifying a telematic performance situation would involve large-as-life or larger-than-life projections on a stage alongside the ‘live’ musicians, operating at upwards of 50 FPS with highly accurate color reproduction. However this relies on expensive equipment, high grade networks, and precludes many small or more highly distributed events.

Yet studies with live musicians have found that unnoticeably compressed images may not have a significant impact on the fostered sense of presence [10]. Without the appearance of noticeable compression artifacts the loss of sharpness and color smearing with lossy codecs will not significantly affect a viewer’s sense of connection through a tele-present system. Additionally, employing similar codecs opens up the possibility of using consumer grade networks for professional performances. While large projections warrant HD quality video streams, images that can be buffered from high-bandwidth networks personal monitors can easily take advantage of 480P or smaller image sizes.

Perhaps the most valuable aspect for fostering a sense of presence between non-co-located performing artists is the perception of motion [10]. Thus frame rates operating at the theoretically optimal human processing rate are key (with new images every 20-30 milliseconds, or 33-50+ FPS) [9]. Previous solutions (barring a few highly expert systems) have been unable to provide this functionality, especially for lower frame sizes and higher compression rates, vipr makes no frame rate assumptions, being limited solely by the hardware, and easily performs at over 100 FPS for smaller image sizes on contemporary PCs.

Informal evaluations have found that both frame rate (typically described as the fluidity of motion) and color fidelity (i.e. the vivid nature of the images) are the primary aspects of vipr setups that new observers comment on.

While latency is typically considered the primary problem facing telematic performing artists, vipr does little to alleviate this challenge. Latency has many components creating the overall effect, including image capture and digitization, compression, transmission, reassembly and decompression, and projection time. The primary factor in transmission time is the physical distances involved, where a single packet always takes a minimum amount of time, and even the speed of light sets hard limits on potential delays. However, vipr provides a profitable 'headroom' between increased compression times (on the order of milliseconds) and reduced transmission times. While an individual packet still takes the same time to deliver, the amount of time required for the transmission of the whole frame is reduced proportional to the compression amount (anywhere from 1% to 30%), which can lead to significant increases in performance. Noisy situations with high packet loss rates benefit even more, as far fewer packets are required to send the compressed images and thus far fewer are lost and require retransmission.
The network transmission of vjlr images relies on the Max built-in network objects (jit.receive and jit.send). These use the TCP/IP protocol which has both advantages and limitations for video transmission. The primary benefit is the guaranteed in-order delivery of frames, ensuring complete reception of the video on the destination machine. However, this requires the retransmission of dropped packets which can cause increased latency (as the stream must be buffered and wait while the missing packet is requested and retrieved). The alternative, unreliable UDP, does not have implicit quality control and thus does not retransmit lost packets, but packets may arrive out of order or be lost entirely resulting in significantly degraded video quality. At the moment TCP provides the only reliable solution, especially for the non-expert artist.

7. REFERENCES


5. CONCLUSION

Telematic performing artists rely on a sense of presence in order to create the desired collaborative works and this sense of presence can be facilitated by live video connections. A number of characteristics directly impact the formation of presence: image quality, image size, color fidelity, and motion are all key elements of the video system. However, typical solutions rarely provide enough flexibility for the artist, requiring technically complex custom solutions to create a desirable telematic setup.

Vjlr for Max provides a ready solution to this problem and has been successfully employed by musicians and dancers in several trans-Atlantic and trans-pacific concerts. This freely available compression object for Max and Jitter opens up new doors for telematic artists, enabling high quality, highly flexible video transmission across smaller network paths. In a uniquely interactive paradigm, vjlr allows the user to experiment with codec settings in real time, in order to both locate the optimal configuration and to expose new aesthetic styles.

6. ACKNOWLEDGEMENTS

The author would like to thank Dr. Guy E. Garnett, the Illinois-Japan Performing Arts Network, eDream, and the National Center for Supercomputing Applications for supporting this work.

---

SONIFYING ANIMATED GRAPHICAL SCORES AND VISUALISING SOUND IN THE HEARIMPROV PERFORMANCE

Adinda van ’t Klooster
Manchester Metropolitan University School of Computing, Mathematics and Digital Technology

ABSTRACT

HEARIMPROV was a live audiovisual performance project for which nine musicians’ from different musical backgrounds were brought together to improvise acoustically to range of animated graphical scores. These video scores by 4 different artists were projected alongside a live generated spectrogram. The spectrogram software tracked the sound input from three microphones and translated this via Fourier transform into one blue, one green and one red spectrogram that overlapped simultaneously into one image. This literal visualisation of the sound originated from the desire to make the music available to people with a hearing impairment. The HEARIMPROV performance was a playful way of visualising live music through software and simultaneously sonifying graphic scores by a group of musicians. Certain ‘playing rules’ were invented by the musicians, to keep the improvised music varied and semi-structured. The project was inspired by the fluidity of digital media where one form of data can be transformed into another and used a collaborative, participatory method in the process of creation.

1. HISTORICAL CONTEXT

The research domain of computer aided translation from the visual image into sound and from sound into visuals for creative purposes is vast and ever growing. There is also a long history of relating image, and in particular colour, to music that predates the digital era. Possible systems of mapping music to image/colour were being invented since the late seventeenth Century. The most frequently used choice in mapping is by relating hue to frequency. Newton developed one of the earliest systems with this method, in his treatise "Opticks" (1704) where he divided the visible spectrum of light into seven colours (red, orange, yellow, green, blue, indigo and violet) and mapped them onto the notes of an octave. This choice of mapping was based on personal preference rather than logical connection, so not surprisingly Newton kept changing his mind as to which colour should be attached to which note [8], and others after him came up with many different systems [2,4]. One of the first instruments to combine lights with music based on this general principle was built by Castel in 1730. The prototype was a keyboard controlling coloured glass filters and mirrors. How he mapped the colours to the notes changed in the different colour organs he made.[17]

The idea to bring music and colour together in a scientific model is undoubtedly linked to the phenomenon of synaesthesia with which Newton was familiar. Synaesthesia is the "involuntary psychological mechanism by which two sensations are simultaneously triggered by the same stimulus" [5]. One form of synaesthesia is colour hearing, which is often musical. A slightly less familiar branch is the involuntary association of verbal sounds, especially vowel sounds, with colours. The phenomena of synaesthesia are subjective, so each synesthete will have different internal mappings.

From the middle of the nineteenth century psychologists became very interested in synaesthesia. At the same time several colour organs were developed. One example is Bainbridge Bishop's colour organ in 1877, which had a device that sat on top of an organ, allowing light to be projected on a small screen as a piece of music was being performed on the organ [14].

2. OTHER ARTWORKS

Amanda Steggell's emotion organism activates lights when the keys of an adapted organ are pressed. The emotion organ is based on Allen Forte's system for describing chromatic sound in 20th Century atonal music [20].

A performance art piece that visualises sound and voice in particular is 'Menu Di Face' by Golan Levin and Zachary Lieberman [11]. The interface is made for the voices of Jaap Blonk and Joan La Barbara who have a wide repertoire of different vocal sounds and are also skilled actors. Not only their voice but also their facial expressions and movements become part of the performance. Through the use of head tracking technology, the performer can also directly interact with the shapes on the screen. Like in HEARIMPROV the
producing overlapping coloured circles on a screen. They fade out visually as the sound fades out. This interface only represents pitch and volume changes, but not timbre, so it misses one of the main characteristics of sound. Clearly, a representation of variations in timbre should be part of the interface if a deaf person is to get an idea of the quality of the sound.

4. THE HEARIMPROV CONCERT

HEARIMPROV was an experimental audiovisual concert that took place at the Sage Gateshead. Improvised music was visualised through specialised spectrogram software. It targeted people with a hearing loss as well as those interested in improvisation and audiovisual performance. For the concert, I brought together a group of musicians from different backgrounds to improvise a range of graphical scores rather than traditionally notated ones. The music thus created was visualised in a 3-channel live spectrogram realised in Max/MSP by Matt Green. Both the live spectrogram and the graphical scores were presented simultaneously to the audience and the musicians. For a movie, see: <http://www.axxissweb.org/Artwork.aspx?WORKID=70319&VISUALID=88729>.

In the real-time domain there are Winamp and iTunes which are not closely linked to the sound input as the sounds can be very different and still generate similar visuals. Other examples are Virtual Choreographer [9], Synaesthesia by Paul Harrison [7], Phonogramme by Vincent Lesbor in 1993, Metasynth by uK Software in 1997, Visual Audio by Cianan O’Kelly in 2004 [16] and Sonos by Jean-Baptiste Thiebaut [21]. None of these are specifically live spectrograms but they have different approaches to how to visualise sound. Sonos differs from the other examples given here, as it is also an instrument, and allows the user to modify the sound spectrum in real time by manipulating an image [21]. The Visual Audio software is also related to the HEARIMPROV interface as it attempts to visualise music for the deaf. Visual Audio maps the middle C onto red, the D note onto orange, and so forth [16]. The octave determines the brightness of the colour: with each octave the brightness increases by 2 %. The middle C has no added white. Going up the scale, increasingly more white is added, creating pastel tints and going down the scale the colours get darker. As notes are played they

Figure 1) Visual score on the left and spectrogram on the right, behind the musicians at the Sage Gateshead, 2007.

Figure 2) The musicians face both score (left) and spectrogram (right) on the plasma screens positioned behind the audience.

The visualization software created for the HEARIMPROV performance was based on the spectrogram, as one of its aims was to provide people with hearing loss with a visual representation of the sound. Spectrograms are one of the most revealing visual displays of speech. Some people can read voice spectrograms like musicians read notes, and are able to decipher the spoken words from a spectrogram image [1,18] but this translation can be a time consuming task [6]. This could be why it not used widely as a visual hearing aid [3]. However, spectrograms are sometimes used to teach deaf children how to use their voice [15]. The usefulness of visual hearing aids is discussed in more detail in Levitt et al. [12].

In our live spectrogram software frequency is plotted upon the vertical axis and time along the horizontal, whilst amplitude is represented by colour intensity. The analysis of each audio signal is presented within one visualisation, so the three channels (green, blue and red) can be easily compared.

When the three colours overlap they generate all the colours of the RGB gamut which creates visually interesting patterns.

The visualisation of each musician's voice generates a spectrogram. For the performance we had to juggle between keeping similar instruments per microphone with giving each musician enough space, whilst still being close enough to a microphone, and they could only stand in certain positions so as to not obscure the projection. Many different variations could be tried with this setup. It would be interesting to use 3 instruments with a very different spectral distribution.

Although a special effort had been made to invite people with hearing difficulties it turned out to be quite difficult to get people with hearing difficulties to attend the concert. A signer was present at the concert to translate the verbal introductions to the scores and an induction loop was also available, but despite this only a couple of people with hearing difficulties attended the concert. At the end of the concert, the interface was available to try out and those with hearing difficulties were especially encouraged to come forward. However, audience feedback questionnaires handed out at the end of the concert were not returned and therefore further and more controlled study would need to be undertaken to be able to say more about the experience of the performance/the interface from the point of view of a hearing-impaired person.
During the rehearsals the musicians were first simply asked to improvise freely to a range of different visual scores. The visual scores provided ranged from abstract to the most positive response from the musicians. After this only abstract animations were presented to the musicians who, through discussions, developed specific playing rules used for this score were generated at a previous audiovisual live performance piece by the modern composer Arvo Pärt and took place in the Skype of Gloucester Cathedral in 2004. Classical musicians affiliated to Gloucester Cathedral performed Arvo Pärt’s score. This music was captured and tracked by five microphones (one per musician) and visuals were generated through specially commissioned software, written in Max/MSP.

Arvo Pärt composed this music at the height of Soviet power when religion was not allowed. The underlying message compares the conception of the perfect piece of art/music to the miracle of the conception of a human being, and is highly abstracted to hide the religious undertones. The minimalism of the music is sustained in the live generated visuals that respond to the underlying themes with colours, lines, sprouting pinpods (which are protrusions that appear in the womb lining for about two days of the menstrual cycle and are indicative of the implantation window for a human embryo) and red droplets that wax and wane and respond to volume level and attack [13].

The first score used in the HEARIMPROV performance was Translating Nature: B) Nanotextures by Julie Freeman. This eight-minute animated visual score was based on an abstracted representation of a biological process. The process is derived from research into self-assembled nano-textures that could be used for stem cell sorting and the detection of pathogens. The score was created in Processing, a java-based programming environment and through discussions with Jeremy Ramsden, Professor of Nanotechnology at Cranfield University. The playing rules used for this score were simple. There were four different kinds of shapes each interpreted by two musicians each. This provided enough of a structure to keep the music varied.

The second score Hearing the Lines You Drew and the Colours You Spread [10] by the author of this article, was the longest, and needed most ‘playing rules’ to keep the music varied. This graphical score was generated at a previous audiovisual live performance piece by the author of this article. The performance was entitled ‘Sarah Was Ninety Years Old’ after the original score by the modern composer Arvo Pärt and took place in the Skype of Gloucester Cathedral in 2004. Classical musicians affiliated to Gloucester Cathedral performed Arvo Pärt’s score. This music was captured and tracked by five microphones (one per musician) and visuals were generated through specially commissioned software, written in Max/MSP.

Arvo Pärt composed this music at the height of Soviet power when religion was not allowed. The underlying message compares the conception of the perfect piece of art/music to the miracle of the conception of a human being, and is highly abstracted to hide the religious undertones. The minimalism of the music is sustained in the live generated visuals that respond to the underlying themes with colours, lines, sprouting pinpods (which are protrusions that appear in the womb lining for about two days of the menstrual cycle and are indicative of the implantation window for a human embryo) and red droplets that wax and wane and respond to volume level and attack [13].

The Sarah Was Ninety Years Old performance was inspired by research into synaesthesia. In the final scene the organ and the soprano reach climactic heights, which is visualized by a colour field (generated by the organ) that fills the screen and subtly changes colour depending on pitch and harmonic structure. The soprano generates a circle that shimmers in size based on her amplitude and pitch. The graphic score generated during this performance was kept and named Hearing the Lines You Drew and the Colours You Spread. It was passed on to several bands and the musicians who improvised to it and recorded their best result. During the HEARIMPROV concert, the 10th sonic interpretation of this score was created and recorded [10].

The score consists of 5 different scenes. It starts with white drops falling down the screen, creating a slowly rising tide that gradually fills the screen:

**Figure 5** Translating Nature: B) Nanotextures by Julie Freeman

**Figure 6** Scene 1 of the score: Hearing the Lines You Drew and the Colours You Spread

This part had the following playing rules: 1) Play short notes, 2) Do not play when someone else plays, 3) Keep the space dense but allow natural pauses and breaks, and 4) Slowly build up to a climax.

The second scene showed calligraphic lines in green and red. The lines were sonified by alternating pairs of two brass instruments, and the black pauses in between the lines were sonified by different pairs of the piano, the melodia and the accordion.

**Figure 7** Scene 2 of the score: Hearing the Lines You Drew and the Colours You Spread

The third scene is again falling drops, this time in red. The playing rules were: 1) Make quirky sounds and free improvisation inspired by the red drops which represent the cyclical nature of female fertility, and 2) Allow to build to a natural climax.

**Figure 8** Scene 3 of the score: Hearing the Lines You Drew and the Colours You Spread

The fourth scene is a different variation of calligraphic lines drawn across the screen, in blue and ochre yellow. During HEARIMPROV these were ‘sung’ by a male and female voice, and the pauses in between were sonified by the bass, guitar and melodia for the first four screen draws and by the piano for the remainder of this scene:

**Figure 9** Scene 4 of the score: Hearing the Lines You Drew and the Colours You Spread

The fifth scene had, instead of drops, pinpods falling down in a highly reactive fluid that slowly fills the screen:

**Figure 10** Scene 5 of the score: Hearing the Lines You Drew and the Colours You Spread

The final scene of the circle against a changing colour backdrop was a free improv inspired by the moment of conception with the sub-rules that the sax, cornet, clarinet and oboe sonified the circle and the piano, guitar, melodia and bass interpreted the background colour.

**Figure 11** Scene 6 of the score: Hearing the Lines You Drew and the Colours You Spread

The third score was musician Steve Brown’s three-minute score entitled 0, 0, 0. It was built in ‘Second Life’ (the virtual 3-d world in which people interact through 3-d avatars) by creating virtual kinetic sculptures surrounded by black matter. Brown’s avatar, ‘Parthenon Acropolis’, recorded the visuals as a live performance in second life. This piece was so short that it didn’t need any playing rules:

**Figure 12** 0, 0, 0 by Steve Brown

The final score was Happy-NES by Dominic Smith who breathed new life into the Nintendo NES cartridge. Visuals were created by passing live MIDI information to the tiles. This way, he created a dense and minimal
Although the HEARIMPROV project had started out aiming to make music available for deaf people, the aesthetic and conceptual gradually aims became more important than the scientific starting point.

7. ACKNOWLEDGEMENTS

This project received Arts Council Funding and was developed during an AHRC funded, practice based PhD at Sunderland University, completed in 2011. Feedback was given by the supervisory team consisting of professor Beryl Graham, Dr. Lieselotte van Leeuwen and Dr. Lynne Hall. This publication is further supported by the Manchester Metropolitan University, School of Computing, Mathematics and Digital Technology.

All images © reserved by the artists
Text © Adinda van ‘t Klooster
Figure 13) Happy-NES by Dominic Smith

We tried out various ways of playing this score. The opinions on what was the best strategy differed, but I will describe one that I thought worked best: The first musician creates a small looping sound event, it gets added to by the second musician and looped again, then thickened by the third musician and looped again and so forth till all the musicians are playing. Then the last musician changes their contribution to the loop, then the second to last musicians changes their contribution to the loop, and so on all the way back to the beginning again. This was particularly hard to do on the way back, as it required very attentive listening skills. For the performance we took a less structured approach, and allowed the music to end in chaos.

Each time a score was played, the sonic results were slightly different but they developed into pieces with a certain character. The playing rules helped to make the music more interesting.

6. CONCLUSION

To summarize: in HEARIMPROV graphical scores were given as input to musicians who interpreted them sonically in collaborative improvisation with other musicians. The resulting soundscape was visualized by the spectrogram. A tenth musician equalized the sound using a three-channel spectrogram with live graphical feedback.

The spectrogram interface was initially looked at as a tool but became one of the instruments. The live spectrogram was an interesting way of visualising sound but also added the unexpected dynamic of influencing the music of the musicians who started to play to the spectrogram more than to the graphical scores.

Conceptually, the essence of projects like HEARIMPROV and Sarah Was Ninety Years Old lies in the process-based nature of the work where each stage of the work can be seen as an intermediate and temporary state which is part of a bigger whole. The poetry lies in the concept that all of these stages refer back to the same origin. This method is influenced by the Fluxus based approach to musical improvisation reshapes the output.

REFERENCES

[4] [Field, G., Chromatography, or, a treatise on colours and pigments and of their powers in painting, London: Charles Trill, 1835.
ABSTRACT

We introduce Improtek, a system integrating a rhythmic framework and an underlying harmonic structure in a context of musical improvisation. In the filiation of the improvisation software OMax [4, 3, 13], it is built on the factor oracle structure to take advantage of the particularly relevant and rich characteristics of this automaton in a musical environment [5]. Moreover, it can adapt to a regular beat and produce improvisations following a given chord progression. Improtek is conceived as an interactive instrument dedicated to performance: its improvisations are based on the style modeling performed on real playing or on an offline corpus. Combined with pattern reuse techniques, this modeling expands on harmonization and arrangement in a harmonic interaction module.

1. INTRODUCTION

The aim of Improtek is to combine style modeling and interaction to install an original dialogue between musicians and a virtual improviser feeding its inspiration on their playing. Both these paradigms and the automation structure at the heart of its implementation (section 2) make the system a cousin of the improvisation software OMax [4, 3, 13] conceived and developed at Ircam. Following previous works on musical style modeling by G. Assayag et al. [6], OMax is capable of learning the style of a human improviser thanks to a representation based on the oracle structure introduced by C. Allauzen et al. [1] extended to a musical context [5]. The software builds a model of a musician’s playing in real-time, and is then able to navigate through this representation by following different paths from that taken by the musician. This leads to generate original improvisations with a common anchor.

With the same intention, Improtek focuses on measured music supported by an underlying harmonic structure. It provides an enriched interaction by taking the beat into account in the framework of a given chord progression (section 3). This conception of improvisation is made concrete by an architecture enabling its integration in a musical band whose tempo can be extracted and followed into account in the framework of a given chord progression. It provides an enriched interaction by taking the beat and produces improvisations following a given chord progression. Improtek is a system able to expand its modeling on harmonization and arrangement in a harmonic interaction module.

2. MUSICAL STYLE MODELING AND ORACLE STRUCTURE

2.1. Style modeling in improvisation and harmonization

Since M&Jam Factory [22], R. Rowe’s Cypher [19], or G. Lewis’ Voyager [14], often considered as the first real-time interactive systems, many “virtual improvisation partners” have been conceived. Most of them benefited from the development of machine learning techniques with the growing idea to get always closer to the interacting human performer’s discourse.

Among them, the Continuator [16] conceived by F. Pauchet models the musical input using an extended Markovian model to create new phrases from this learning. As in OMax free mode, there is no rhythmic perception to enable a synchronization with the musician. On the other hand, GenJam [7] developed by J. Biles is closer to the previous experiments of OMax with a beat mode: the software provides an accompaniment with a given tempo to support a musician’s improvisation. After listening, it explores some sequences modified through a genetic algorithm. B. Thom’s Band Out of A Box [21] also involves a non-interactive computer accompanist with a fixed tempo in a trading fours interaction scheme where a human improviser and a virtual partner repeatedly call and respond, in four-bar chunks. Each bar of the human improvisation is assigned to a cluster called playing mode, and the computer response is constituted by 4 bars belonging to the same sequence of modes.

In this last category, the corpus can be considered as a training environment to create generative models (among them C. Chuan and E. Chew [11], Microsoft Mysong-Songsmith software [20]) and/or as a musical memory in which fragments are retrieved and combined to create the new material (G. Ramalho’s jazz bass player ImpLuck [18]).

The definition of the segmentation unit in the corpus processing is one of the key differences: the grain can be key notes of a reduced melody in C. Chuan and E. Chew’s system, a single chord in Songsmith and the software Band in a Box (PG Music), or particular chord chunks in G. Ramalho’s system. The idea is to find the right balance between long enough slices to provide plausibility and coherence in the returned accompaniment, and fine enough slices to avoid recreating too long and identifiable segments from the corpus. In the case of Improtek, the chosen unit is the beat. Indeed, the oracle structure prevents from choosing between musical smoothness and innovation by providing continuity by construction, and enables to work with such a fine grain.

2.2. Oracle and pattern recognition

The oracle was initially conceived for optimal string matching, and was extended for computing repeated factors in a word and for data compression. This acyclic automaton represents at least all the factors in a word, and the incremental construction algorithm is time and space linear in its length (for the details of the construction algorithm, see [1]).

The formal properties of the oracle structure are detailed in the following articles, and fully applied to the issue of stylistic reintegration [4] in several OMax papers. We briefly summarize here the tools providing the continuity and coherence of the sequences generated through an oracle: the forward transitions and the suffix links.

The forward transitions (forward links, plain lines) enable to reach every sub-pattern in the original sequence starting from the initial state. In this way, every progression between consecutive states of the original sequence can be generated.

3. IMPROVISING WITHIN A METRIC AND HARMONIC FRAMEWORK

3.1. The beat hypothesis

The philosophy of the current version of OMax is to turn every single musical event into a state in an oracle object. The free real-time navigation [2] through a thus structured memory makes it an interactive instrument dedicated to improvisation in a free musical context. The approach of the Improtek project differs by the integration of two paradigms. First, its improvisations take place in an unstructured harmonic representation by a chord progression we will refer to as grid. In this way, it makes the system able to expand its modeling on harmonization and arrangement. Then, it takes a metric framework into account by setting the beat as the elementary unit in its acquisitions, restitutions, and generations.

The only musical hypothesis is therefore the existence of a regular beat in the material listened and produced by Improtek. The tendency to the synchronization with a periodic beat being a deep universal of the human music perception [17], this sole assumption does not make the system oriented towards a restricted musical field. Starting from this point, we add a notion of labelling some musical slices separated by beats, making them equivalent (see section 5.1.2). This process enriches the set of possible combinations, but does not carry any harmonic hypothesis. Indeed, as discussed in the last section, the references to the jazz idiom through this article come from the context in which Improtek has been used so far.

Figure 1. Oracle for the sequence abcdbcd: generation of a sub-pattern starting from the initial state.

Fig. 1 shows the oracle for the sequence abcdbcd and illustrates this point by displaying a path using forward transitions to generate the sub-sequence bcd (0, 2, 3, 8), originally occurring between the states 5 and 8. The oracle also locates the repeated sub-pattern in the original sequence with the suffix links (backward links, dashed lines). They point at the final state of the previously encountered occurrences of a pattern.

Figure 2. Oracle for the sequence abcdbcd: repeated sub-pattern.
3.2. Mode beat oracles

The improvisation module is built on a previous version of OMax (OMax 2.0, 2004) conceived as a Lisp library under the OpenMusic environment [9]. This version implemented the oracle structure both in a free improvisation context (mode free, direction adopted by the current version of OMax) and with a regular beat (mode beat, starting point of Improtek).

In this view, each state of the oracles represents a musical slice whose duration is given by the current tempo and which contains different types of events happening between two beats. These events can be musical MIDI sequences - melody or accompaniment fragments - or symbolic information such as chord labels. During the generation steps, these different features constituting the states will sometimes be considered as elementary outputs concatenated in a fragment reuse process, and sometimes seen as labels to compare with the given path to follow in the navigation through the oracle.

3.3. Learning and improvising with the live oracle

Improtek improvises by retrieving and combining preexisting elementary units: the new phrases are built by concatenating "beat slices" coming from its musical memory. These fragments are not independently and randomly drawn but continuously collected by following the chosen harmonic grid supporting the musical session as a guideline.

This process involves a first instance of the oracle structure: the live oracle, which carries its learning process out on the phrases played by the musicians (fig. 3). These MIDI inputs are indeed indexed by beat in real-time by the chord labels of the current harmonic grid (see section 4.2) and are therefore formatted for the building of this object.

What we call "improvise" is the generation of new musical phrases using the memory stored in this live oracle. To do so, the chord labels in the oracle are constrained navigations through the live oracle.

A continuity parameter is tested at each step of the calculation. It counts the number of successive forward transitions followed during the navigation, and then gives the length of the duplicated segments from the sequence in the oracle. By trying to follow a minimum continuity parameter, one can therefore quantify the wished balance between fidelity to the original sequence and originality in the generated improvisation: a high value will lead to a rich resemblance whereas a low value will bring more surprises.

The constrained navigation process consists in reading the successive labels of the input grid to look for beats indexed by these same labels in the oracle. If the current continuity parameter does not exceed the imposed maximum continuity, the search is operated following graded modes:

- Continuity mode: If its label matches, follow a forward transition, update the current beat position in the oracle, and output the associated melody fragment. If not, switch to Suffix mode if operating suffix links are found, otherwise switch to Nothing mode.
- Suffix mode: Follow the suffix link pointing on the longest repeated suffix to reach a matching label (see fig. 4), update the current beat position in the oracle, and output the associated melody fragment. Otherwise switch to Nothing mode.
- Nothing mode: The pattern matching is performed independently of the context and the label is searched in the whole oracle. A transposition can be used if necessary.

Fig. 4 shows an example of generation step. At the current stage, the chord labels A, B, and C have been searched and found in the live oracle, and the concatenation of the associated musical fragments D, B', and C' forms the

3.4. Constrained Navigation and continuity

To "improve following a grid" means here to follow a path in the automaton using transitions indexed by labels coming from that grid. At each step, if no matching label is found from the current state, the navigation tries to follow a suffix link pointing on an other state where the required label could be read. The suffix links provide the existence of a conflict in context between both concatenated fragments in the original sequence used to build the oracle. In this way, the navigation first looks for continuity by trying to stick to the previously learned progressions. Then, it searches for the labels independently of the local context if they do not appear, even after transposition, in the chosen oracle.

A continuity parameter is tested at each step of the calculation. It counts the number of successive forward transitions followed during the navigation, and then gives the length of the duplicated segments from the sequence in the oracle. By trying to follow a minimum continuity parameter, one can therefore quantify the wished balance between fidelity to the original sequence and originality in the generated improvisation: a high value will lead to a rich resemblance whereas a low value will bring more surprises.

The constrained navigation process consists in reading the successive labels of the input grid to look for beats indexed by these same labels in the oracle. If the current continuity parameter does not exceed the imposed maximum continuity, the search is operated following graded modes:

- Continuity mode: If its label matches, follow a forward transition, update the current beat position in the oracle, and output the associated melody fragment. If not, switch to Suffix mode if operating suffix links are found, otherwise switch to Nothing mode.
- Suffix mode: Follow the suffix link pointing on the longest repeated suffix to reach a matching label (see fig. 4), update the current beat position in the oracle, and output the associated melody fragment. Otherwise switch to Nothing mode.
- Nothing mode: The pattern matching is performed independently of the context and the label is searched in the whole oracle. A transposition can be used if necessary.

Fig. 4 shows an example of generation step. At the current stage, the chord labels A, B, and C have been searched and found in the live oracle, and the concatenation of the associated musical fragments D, B', and C' forms the
4.3. Using a score follower as a sequencer

Antescofo [12] is a polyphonic score following system and a synchronous programming language for musical composition conceived by A. Cont. This object, developed as an external module for Max/MSP and PureData programming environments, conducts an automatic recognition of music score position from a real-time audio stream. It enables the synchronization of an instrumental performance with the computer realized elements of an electronic score.

The sequences calculated by the Lisp/OpenMusic module are written and saved as Antescofo scores. The phrases thus generated broaden the improvisations collection and are available to be loaded by the performer during the improvisation: in our use of Antescofo, the beats provided by the beat tracker object act as the “notes” in the electronic score, and the contents to be triggered are the beat slices. Finally, an improvisation generated from an acquisition performed at a given tempo can be played with a different one since the time notation under the Antescofo format is relative.

4.4. Segmenting and indexing the inputs

This couple of objects in charge of the beat management is found downstream to play the generated sequences, and equally upstream in the live acquisition process. The improvisation session takes place in the scope of a known harmonic grid, and this last is written as a progression of chord labels, each of them being associated to a numbered melodic track with a symbolic chord progression, respectively. Learning the harmonization and arrangement

The harmonization and arrangement module can be used in an autonomous way as an independent block producing an accompaniment for a melody without interacting in real-time. Yet, it was conceived to be integrated in the wider environment of ImproteK to compose new accompanied improvisations.

It does not use any musical rule and is exclusively based on a corpus which is simultaneously seen as a learning ground to extract empirical harmonization mechanisms, and as a musical memory in the framework of a pattern reuse technique to make the accompaniment concrete once it has been calculated.

5. HARMONIZED AND ARRANGED IMPROVISATIONS

5.1. Learning step

5.1.1. The corpus

The learning of the corpus is performed on three features: the melodic track (theme, solo improvisation, etc.), the accompaniment track, and the associated harmonic grid. It is carried out on live performances by musicians and consists for the moment in jazz standards and pieces by Bernard Lubat.

We refer to “corpus” to designate the set of models (every letter only appears once). In the second example to a character string: an oracle built on the word consonants or repetition in the beat slice. In the same way, two oracles’ structure impacts the number of operating suffix links as illustrates the simplified example in fig 7, applied to a character string: an oracle built on the word vowel.

In the first example, every suffix link points on the initial state because no repeated sub-pattern has been found (every letter only appears once). In the second example using the equivalence classes vowels and consonants, we observe repeated sub-patterns and the structure is much more complex.

5.2. Generation step

5.2.1. Harmonizing and arranging in a cascade

The whole harmonization and arrangement process amounts to generation the mechanism of constrained navigation in a cascade, first with a chosen harmonization oracle then with a chosen arrangement oracle (fig. 8).
The harmonization outputs a chord labels progression, and this symbolic sequence therefore becomes the path to follow for the navigation in the chosen arrangement oracle (filled by sequences of associations between melodic fragments and chord label sliced by beat). The pattern matching is this time performed on chord labels to output a sequence formed by the concatenation of the accomplishment fragments found in every stage of the research.

5.2.2. Formal intermediary
It is important to note that, even if an intermediate step involves symbolic data such as chord labels, no harmonic rule is used to perform harmonization and arrangement. The denomination of this formal intermediary is only user-oriented to make performance more intuitive: the system itself is unaware of the musical meaning of this labeling and only considers two chord labels as two indexes to compare.

The insertion of this formal language at an intermediate level separating the process in two different steps is motivated by three reasons. First, it naturally comes from the usual notation in jazz scores as we found in the Rebooks where a melody is facing a corresponding chords progression. Then, it enables to multiply the possibilities: a phrase can indeed be harmonized with a given instrument and then arranged using an arrangement oracle learned on a completely different corpus. Finally, it will allow to implement in a future development an optional level of chord substitutions based on a grammar [10].

Furthermore, the terms “harmonization” and “arrangement” come from the fact that Improtek has been used so far in tonal jazz sessions. In other musical contexts, its genericity enables an understanding of other forms of vertical associations that can be indexed in an agnostic way with an other grammar.

6. EXPERIMENTS AND RESULTS
Improtek has been used as a virtual partner by professional musicians, in particular French jazz musician Bernard Labar during improvisation sessions conducted in Uzeste in 2011. Video and audio examples can be found at http://ehess.modelisationsavoirs.fr/improtecm/improtek.

Some of them show the real-time control of the software through the Max/MaxMSP interface (see for instance the improvisation based on Erroll Garner’s mambo style transcription). The direct interaction of the computer with the live musician is illustrated by improvisation sessions where Bernard Labar and Improtek alternately play as soloist and accompanist, and trade choruses.

Other series illustrate the wide variety of results for a same input in the harmonization and arrangement module depending on the user’s choice to use different parts of the corpus for both steps, and on the continuity he imposed. It goes from “imitation” by choosing a same part of the corpus for the live, harmonization, and arrangement oracles, to originality or even extravagance when completely unrelated oracles are used.

7. CONCLUSION
We described a music generation system which is able to understand the logic of the horizontal and vertical associations in a live musical improvisation performance to become itself a source of proposals by developing its own aesthetics close to that of its partners. Its prime material is indeed their playing. It is used at the same time as a learning ground for the style modeling, and as a musical memory to develop its own improvisations.

The properties of the oracle structuring this memory enable to get over the dilemma “innovation vs. coherence” by ensuring continuity by construction, and therefore giving the possibility to work with a fine grain: the beat is set as the elementary unit in the calculation, and its restitutions is made possible by following a beat tracker to reach a better interaction.

Improtek is indeed conceived as a proper instrument and requires a full-time performer to manage the learning, the generation, and the playing in real-time. It can alternately be soloist or accompanist and is even capable of creating accompanied improvisations via the harmonization and arrangement module.

Current work is devoted to the evaluation of the compatibility between the harmonic progression of the current session and that of the accompaniment returned by this module. For the moment, the performer is given the comparison between both grids through the interface which displays the respective chord labels. This study will lead to a better integration of the “harmonic interaction” in the instrument, and will make its use more intuitive.

Acknowledgment
This work is realised with the support of the French National Research Agency, in the framework of the project “IMPROTECH”, ANR-09-SSOC-008.

We wish to thank the OMax family Gérard Assayag, Georges Bloch, and Benoît Lévy for the fruitful exchange of experiences and ideas regarding the conception and implementation of Improtek. We thank Laurent Bonnasse-Gahot who made it get rhythm and the beat tracker, Carlos Agon and Jean Bresson for their advice concerning OpenMusic, and Arshia Cont for the custom-made Antecessor. Finally, we want to express special thanks to La Compagnie Labar for the always enriching sessions.

8. REFERENCES


CALDER’S VIOLIN: REAL-TIME NOTATION AND PERFORMANCE THROUGH MUSICALLY EXPRESSIVE ALGORITHMS

Richard Hoadley
Digital Performance Laboratory
Anglia Ruskin University
Cambridge UK

ABSTRACT

Notation is a central issue in modern western music. Composers have often sought ways of expanding and refining the functionality of notation and, in doing so, have re-shaped the music that they were originally aiming to describe. Other musical traditions have very different uses for notation; some have no use for it at all; each approach creates contrasting musical experiences. The role that electronics and computers have played in music has also influenced the nature and function of notation. More traditional ‘live’ notation of note/pitch-based music generated algorithmically has proved particularly problematic: musical notation is itself a very complex subject. Composers and technologists have instead used libraries of images, algorithms for the generation of musical material or simplified notations that can be used as the basis of more improvisatory performances.

This paper presents work involving the live presentation of ‘traditionally precise’ music notation created through algorithmically generated material. This notation can then be performed by a human musician alongside computer-generated diffuse sound or other ‘real’ musicians. Technologies used include the SuperCollider audio programming environment and the INScore notation software used with the Open Sound Control protocol used to communicate between them. As well as providing a fascinating musical experience, the process highlighted a number of issues concerning performance practice, instrumental technique, rehearsal, time and timing, as well as the nature of notation itself and its relationship to improvisation.

1. INTRODUCTION

This paper is three things: a case study of how I tackled the role of real-time notation in a specific piece; an aesthetic analysis of the role of real-time notation in music performance and a description of the system through which algorithmic composition (and subsequent ones) have come about.

In the context of this paper I use the term ‘real-time notation’ (or ‘live notation’) to refer to notation that is generated during the performance progresses and where this process is itself considered to be of central importance in the composition.

Algorithmic composition provides me with an invaluable insight into the creative process; it also enables me to generate novel musical ideas automatically through definable processes. Because the latter are generated by chance as well as choice, the responsible algorithms can be structured to have features, ideas and patterns that are new to me. It is also advantageous that such generated material can be easily auditioned.

Although such a system may be initially developed in association with a specific composition, it inevitably contains functions and processes that can be useful in future work; they are separate from the immediate process of composition. Work on Calder’s Violin has also resulted in my own development of a SuperCollider class to facilitate communication between SuperCollider and certain aspects of INScore. These resources have already been used to create new compositions Flauxus and Flauxus Tree [16]; in the case of the latter including experimental methods for the generation of notation through physical movement.

One of the more divisive issues in electronic music today involves its relationship to live performance: the move from ‘object to dynamic system’, discussed by Chadabe [3], Collins [4,5] and Ariza [2], perhaps reflecting the way in which music notation is a mediating element allowing dynamic interchange between composer and performer. Hudak et al. [18], attempt to understand notation from the perspective of a functional programmer, indentifying many of the ‘limitations’ of ‘common practice notation’: finding it frequently ‘deficient, inconsistent and redundant’ and pointing out perennial issues such as the number and type of tuplet that should fit into a given duration. More obscurely they claim that ‘traditional notation is unable to adequately capture a composer’s intentions’, and that ‘traditional music is biased towards music that is humanly performable’ (their italics). This is however, unsurprisingly, ‘an obstacle when trying to notate music intended for computer performance’. Here might hold that it is not the logicality of the relevant notation system that is important, but its accessibility to the skill with which the number and type of tuplet should fit into a given duration. While being of inherent value, this feature has itself highlighted some interesting possibilities. Cheryl Frances-Hoad, a collaborator in Calder’s Violin, in a talk referred to Fluxus Tree said, that when presented with too much material presented too quickly she found it interesting to intuitively ‘average’ the notation based on where particular phrases appeared on the screen, a creative exploitation of behaviour that might otherwise be considered anomalous [11].

3. IMPLEMENTATION

Some parts of the music for Calder’s Violin are based on interactive music written for the music-dance production of Triggered performed in London in June 2011 [10]. This music was written (in SuperCollider code) as an attempt at creating liveliness and controlled unpredictability when combining scheduled algorithms with the interpretation of data acquired from dancers’ movements, gestures and touch. As a key musical sound used was that of a synthesized piano, the possibility of a conversation between ‘live’ piano, and algorithmically generated piano emerged. Software for generating live notation includes MaxScore [6], the Bach Project [1] (each for Max/MaxP) and INScore [7]. INScore, while still under development is a software environment optimized for external control via OSC. INScore’s abilities in symbolic music notation rendering rely on the GUIDO Engine and the MusicXML library.

4. METHOD

In Calder’s Violin an algorithm is usually a high-level function, and there are about 270 evaluations of these functions scheduled in the composition. The scheduling involves precisely timed events sometimes coloured with unpredictability (evaluate the function in 2.0 plus the random of 1.0 second, for instance). There are also a significant number of notational and layout functions – the latter, for instance, making sure some previously evaluated functions have been terminated. The piece otherwise follows a broadly ABA’ structure, where A is delicate, florid and decorated; B is faster and more rhythmic.

I call these algorithms ‘musically expressive’ because the principal motivation in their design is to emulate my own ideas and gestures: imaginings that are traditionally expressive in musical terms. This iterative process of imaging, implementing, re-imaging, re-implementing, etc., itself plays a very important role in the development of both algorithm (function), musical gesture and indeed the musical context in which these gestures are to occur, as it does in more traditional developmental composition. For me, all of these components work together as musical composition. In order to make the gestures produced by the algorithms fully a part of the composition, elements (arguments) were included in order to increase what I would term their ‘expressivity’: controls on note duration (tempo/ribs), amplitude, note length (articulation), etc. In subsequent projects this can then be extended in a way that reflects both the extension of an algorithm’s functionality in software and the musical development of a melody, a phrase’s shape or the nuance of a harmony.
5. NOTATING ALGORITHMS

An example of a simple algorithm from the piece is the function called ‘chord6’. This process is not so fully available to the performer of Calder’s Violin. In this particular piece, the macroscopic structure remains similar in each performance; in that respect it is not materially different from the performance of a composition with fixed notation. However, the detail is significantly different each time, and sometimes a phrase is not completed before a performer is required to play its opening notes.

Problems do arise: the performer may feel that they have been ‘relegated to being a mere sight-reader (or expert improviser) as one reviewer put it [25]. Alternatively, the data is written to ‘live’ arrays, a ‘finite’ part rather than just being told to ‘improvise’ in a particular way, with maybe a scale provided as an example. Classical performers are used to providing, quickly and efficiently, a confident, fluid performance.

To encourage this I have tried to ensure that the music generated by the software is either not too difficult technically, or is music that can be ‘improvised’ with some ease (swirling chromatic passages, for example). Both Mifune and Cheryl have indicated, in response to the above criticism, that rather than being ‘relegated’ to sight-reading, they find the experience exhilarating [11, 23]. Mifune also confirmed that, as a typical performer of western contemporary music, she found it easier to improvise when provided with a notated basis.

6. ISSUES ARISING FROM THE COMPOSITION AND PERFORMANCE

Calder’s Violin has to date been publicly performed by violinists Mifune Tsuji and Marcus Barcham-Stevens for concerts that were a part of the Cambridge Festival of Ideas in October 2011 [15] and the SuperCollider Symposium in London, 2012 respectively. In May 2012 a related and more experimental work was performed, Fluxus Tree, which utilises the movements and touch of dancers to generate electroacoustic sounds and live notation for the instrumentalist, in this case the composer and (occasional, but excellent) ‘cellist, Cheryl Frances-Hoad Reid. Comments and suggestions made by these excellent musicians have been vital in the development of these compositions.

Although I imagined the display of notation on a laptop screen would involve a fairly simple transposition of notation from paper to screen, the process actually revealed quite a number of issues and challenges for me and for those who have performed using the system. These became even clearer during the subsequent (iterative) processes of rehearsal, further development and performance.

6.1. Rehearsal and preparation

Most performers require rehearsal in order to engage with the music as well as to develop their interpretations of the piece at both macro and microscopic levels. Practice may begin with a review of any background material concerning the piece or composer, before attempting to develop a general feeling for the overall shape - how long is it? Which style does it use? - finally concentrating on the work of getting the notes right.

This is that a delay of about two seconds should be implementable - ‘live’ arrays, a ‘finite’ part rather than just being told to ‘improvise’ in a particular way, with maybe a scale provided as an example. Classical performers are used to providing, quickly and efficiently, a confident, fluid performance.

When I had some time in preparing for my first rehearsal with Mifune, it quickly became clear that many of the presumptions I had made about the display of the generated notation were inaccurate. A fixed, physical score contains much information that is not immediately obvious: an instant overview of the number, size and content of the pages. If you send INScore a ‘note’ message it will default to displaying this in the absolute centre of the page. If you then send a second message with the first note and a new, one, this two-note phrase will be re-centred, and so on. One of the choices, then, is whether to send a phrase so that it appears one note at a time, gradually expanding from the centre of the page, or whether to send the material one phrase, or even one ‘page’ at a time. In the event there are cases of each usage, the choice of which to use can depend very much on the nature of the music and the performance requirements at that point in the piece.

6.3. How soon is now?

Another issue is when to actually display whatever form of notation is required, and in particular how much ‘real’ of an upcoming event should be given to the performer. This isn’t a matter of rehearsal but instead of how much time is required for someone to take in and react to the appearance of a musical phrase. Inevitably it can initially create nervous tension for the performer: often concerning how ‘correctly’ they are going to be able to play the appearing material. This reaction can be somewhat alleviated when the performer becomes aware of the inevitably improvisatory nature of some aspects of the performance. It also influences how the performer is to know when to play the notation in terms of measure and beat. I will consider this further in section 6.4 below.

In general it would appear that a delay in the order of one and a half seconds is sufficient, although sometimes more would be an advantage, particularly when the generated music is more complex. For the software to allow arbitrary lengths of delay would require a modification of the scheduling functionality. At present, algorithms are written such that they maintain their content for the duration of the algorithm. Subsequently, if another set of data is generated, that previous set is over-written and becomes no longer accessible. If a scheduling delay is then introduced that is longer than the timed gap between evaluated instances of the function in question, the re-scheduled function will wrongly use the more recently generated data. In order to arbitarily long delays, all functions would have to write to specifically stored data addresses and these, then, would be used for ‘playback’. While there is nothing wrong with this in essence, it does call into question the nature of the idea of ‘liveness’. In reality a performance using stored data, even if the data is algorithmic, is no different from any other type of fixed media playback. Would the issue be different if data addresses were overwritten immediately they are played back, so retaining their anonymity? But in reality, does over-writing or deleting make any real difference to the situation? The problem highlights a difference in the way in which wet-ware works in comparison to hard-ware and in that case when also combined with the challenged of an un-fixed score. Does it matter when the data is generated?

The delays mentioned above seem to be confirmed experimentally by research undertaken into musical sight-reading, where it is estimated that performers ‘read forward’ between 0.3 and 1.3 seconds of material [12, 21]. This is regardless of the number of notes to be read, although it is affected by their stylistic complexity. It should also be considered that the circumstances presented in Calder’s Violin are not entirely the same: in standard sight-reading the material does not appear and disappear effectively instantaneously. In any case, it indicates that a delay of about two seconds should be ample in most cases.

Why not, then, pre-rend the notation to a fixed-media format and simply play it back? This would certainly make the processes less ‘dangerous’ technically and save me a few nerves! There is one particular reason for this: the programming ‘places’ the ‘printing’ of the music to the screen at particular times and as such, as it is described above, is itself an important aspect in determining when the music is itself performed (usually and depending on context, about a second and a half after display). In my original plans for the piece this would have had a significant impact as it would have affected the influence of real-time sensor data on the music and...
notion. This was not implemented in Calder's Violin but more recently has been in Fluxus Tree [16].

6.4. Other issues and questions

Another issue is whether to include visualisations of the accompanying synthesised piano material in order to help with coordination and prediction, a matter that increases in importance with the addition of further instrumental parts, but for Calder's Violin it proved unnecessary.

A ramification of not including such visualisations is that there cannot be something acting as a ‘pulse’ or ‘tempo’ that might be shared by two people (or a person and a computer). The violin part is notated in accordance with durational algorithms that are designed to display material in clear and expressive ways, but of course duration is only one factor in the perception of tempo. The computer music sometimes sounds as if it has a pulse, but this is an illusion that can be ignored if necessary without the ‘normal’ ramifications that might occur in human performance. In rehearsal the performers seem to respond naturally to these inconsistencies in a way that makes me feel confident about the intuitive methods I have used in describing the notation’s inherent vagaries. Of particular importance are the subjective interpretations of relationships between pulse and tempo; in conversation Mifune and Cheryl each confirmed that getting used to this new environment was indeed a matter of practice. What initially seemed vague and a source of anxiety became, while I’m happy with this, I made a little uncomfortable by the thought that any reliance on an external visual display detracts from the independence and power of the music itself.

How much a part of the creative process could these technologies become? With easy integration of electroacoustic, notation-based, algorithmic and machine-listening realms potentially unified, many new approaches become feasible, even those that arise from behaviours that currently might be considered anomalous.

I am personally interested in integration of physical gesture with live notation: there are roles for this in creative, educational and therapeutic environments.

Finally, for those who are interested, the author is currently preparing a SuperCollider class for use with MaxMSP. ICMC, SARC, Belfast. 2008.

REFERENCES


[3] Chadabe, J. Electronic music and life, Organised Sound 9, p3-6, 2004


CORRECT AUTOMATIC ACCOMPANIMENT DESPITE MACHINE LISTENING OR HUMAN ERRORS IN ANTESCOPO

Arshia Cont, José Echeveste, Florent Jacquemard, Jean-Louis Glavotto
MuSync Team-Project (ircam, inria, CNRS)
Ircam-Centre Pompidou, Paris, France.
nname.lastname@ircam.fr

ABSTRACT

Automatic accompaniment systems are comprised of real-time scoring follows that in reaction to recognition of events in a score from a human performer, launch necessary actions for the accompaniment section. While the realtime detection of score events out of live musicians’ performance has been widely studied in the literature, score accompaniment (or the reactive part of the process) has been rarely addressed. In this paper, we expose the respective literature concerning this missing consideration. We show how its explicit design considerations would allow correct accompaniment despite machine listening or human errors introduced during score following, and furthermore how it enables a more elaborate authoring of time and interaction for mixed live electronic pieces.

1. INTRODUCTION

Automatic accompaniment is the act of delegating the interpretation of one or several musical voices to a computer in interaction with a live solo (or ensemble) musician(s). The paradigm was first put through doubtfully by Danenberg and Vercoe in [4, 9] where a computer would provide automatic accompaniment in the form of symbolic signals (MIDI) from a musician. The most popular form of such systems is the automatic accompaniment of an orchestral recording with that of a soloist in the classical music repertoire (concertos for example) as described in [6]. In a larger context, these systems became popular for interactive computer music repertoire [7], where the association of live musicians with computer generated processes becomes crucial. Figure 1 shows the score of a simple interactive computer music piece where the top staff corresponds to a human musician and the lower staves correspond to automatic accompaniment sections that should be run synthetically to the first staff during live performance. Note here that the accompaniment commands depend exclusively on the nature of the computer music process in question. They can be symbolic commands, continuous curves or sequences written relative to the live instrument section of the score. Within this context, the case of automatic accompaniment for classical music is crucial to a wider context of this larger concept where automatic accompaniment commands are replayed by either symbolic MIDI messages or a live rendering of an

Figure 1. Score example for an interactive piece, the top staff contains the instrumental section whereas lower staves correspond to specific computer music commands.

The best starting point in the design of an accompaniment system is to observe the human counterpart. Musical accompaniment among musicians is a combined act of listening and coordination governed by music scores. While human listening plays a crucial role in accompaniment, it is (computationally speaking) fallible. Moreover, actions produced by musicians in an ensemble can contain perceivable or unperceivable errors. Despite such inconsistencies, the musical output should stay more or less intact and error-free. This is to say that despite the importance of the recognition or listening phase in musical performance, the coherence of the overall musical output is to a great extent covered by the ability of human musicians to coordinate and anticipate their actions and adopt the best synchronization strategies in real-time to achieve the best musical output.

Automatic accompaniment systems in general are comprised of two main components as illustrated in figure 2: A score follower that is capable of decoding the live instrument position as well as necessary musical parameters in real-time given the instrumental score (for the top staff of fig 1); and a second component in charge of launching the accompaniment commands synchronously to the live musician (for lower staves in fig 1). In realtime control theory [8], the score follower is an interactive system with transactions at its own speed with the environment (usually governed by signal processing and machine learning techniques). Execution speed is formally abstracted to be perceptively close to zero for such processes. The accompaniment block is however a reactive system, reacting continuously to the external environment and at the speed imposed by the environment (the human performer). Considering both human and computer accompaniment together, the “healthiness” of the musical output of an accompaniment system requires as much considerations for the coordination/reaction phase as for the recognition phase. The general score following and accompaniment literature has however underestimated the first, focusing on robustness in the listening phase; leaving the action phase to hand-engineered and most often preliminary considerations for the accompaniment actions. This lack of consideration for action coordination would in return degrade the paradigms of interaction between human and computer mediums for interactive pieces, leading to severely simplified programming and authorship for mixed instrumental and live electronic pieces as discussed in [3]. This paper aims at illustrating the importance of the reactive system architecture in automatic accompaniment and showing its usefulness within two contexts: (1) the ability to automatically handle machine listening or human errors with no propagation to the computer output; and (2) to enrich the vocabulary of live electronic accompaniment by providing explicit access to the authoring of time and interaction in the interactive computer music environment.

In this paper, we aim to present our approach to this paper that an interactive listening system is readily available and focus on handling and authoring accompaniment actions. This paper discusses the integration and employment of the proposed paradigms within the Antescofo®[1] software coupling both listening and coordination aspects of such systems. In section 2 we discuss previous works that explicitly handle accompaniment actions integrated within a score following systems. We showcase and motivate problems with existing approaches in section 3 and propose an approach to exposed problems by adopting synchronization strategies during authorship and run-time in section 4. We conclude the paper by illustrating several examples of this approach for automatic accompaniment as well as live electronic pieces using Antescofo®.

2. PREVIOUS WORKS

In this section, we overview two existing and popular systems that handle both machine listening and online accompaniment. For the scope of this paper, we focus on the way automatic accompaniment is transcibed and handled in such systems. Therefore, we do not provide any discussions on the machine listening aspects and do not overview all existing score following systems.

2.1. Music-Plus-One

Music-Plus-One is a musical accompaniment system developed by Christopher Raphael for the classical music repertoire and destined for a soloist in a concerto-like setting. The system is decomposed into three modules: one for a realtime score match using hidden Markov models, a second for coordinating audio playback of the existing accompaniment using a fuzzy classifier, and a third for linking the two using a Kalman filter model for tempo adjustments [6]. Whereas the listening and decoding parts of Music-Plus-One is well documented, there are no explicit documentation on the authoring and handling of the coordination for the accompaniment part (at least at the time of writing this paper). However, demos of the software are available on the web.

2.2. Music-Plus-One’s application is destined for the classical music repertoire and concerto-like settings. It has been however rarely used for contemporary interactive music repertoire. Understanding the synchronization and coordination strategies are important for the creation of new pieces for the system. The discussions hereafter are thus by no means a criticism of this approach.

Music-Plus-One uses the following minimal text data to undertake automatic accompaniment: a musical score of the solo part, an audio recording of the orchestral or accompaniment part, a set of trained parameters for the listening model (trained offline and used during realtime detection), and some timing data in charge of associating the solo score to the accompaniment audio for the rendering phase. Among these, the timing data is of utmost importance since it creates the necessary binding between the soloist score and the accompaniment audio that will be employed during rendering. Figure 3 shows an excerpt of Music-Plus-One text input for illustrating this point. On the left, is a text description of the soloist music score, in terms of relative position in a measure, MIDI pitch numbers and other necessary information for the listening module. On the right (a separate text file), each line contains the accompaniment audio onset time for the specific event in the solo score. This correspondences is (most probably) obtained by a rigorous segmentation of the instrumental audio with regards to the solo score, to provide coherent musical phrases to be used during live accompaniment rendering. Note that the coordination points on the right, do not necessarily correspond to the solo score. This is normal since an optimal segmentation of accompaniment onsets should be based on musical phrasing rather than a one-to-one correspondence.

Within the structure described in the above example, it is not clear how Music-Plus-One handles machine listening or human errors during live performance. Depending
on the musical context, and in presence of machine listen-
ing or human errors (missing events for example), one
might want to skip an accompaniment phrase or launch it with varying speed. Moreover, preparing such accom-
paniment/coordinaton score can be a heavy brain in preparing an accompaniment and not very suitable for
compositional purposes.

2.2. Antescofo

Antescofo [1] is another polyphonic score following sys-
tem capable of handling accompaniment actions and in-
tegrated within the system. The listening machine of Antescofo is documented in [2] and some aspects of its accomp-
iong language in [1, 3]. The main musical paradigm addressed by Antescofo is that of mixed instru-
mental and live electronic (or interactive computer mu-
sic) pieces, where the "accompanying actions" can range
from a simple concertos-like setting to triggering of live electronic processes as common in interactive computer
music. Antescofo requires no training for its listener and accept a single text as score input.

An Antescofo score is composed of both instrumen-
tal (separate) and the accompaniment actions within
one integrated score. Figure 4 shows an example of a simple Antescofo score with both instrumental and accom-
paniment actions. The semantics of the instrumental score allows the construction of complex events such as
trills, glissandi, improvisation boxes and also continuous events. The (accompanying) action semantics is entirely
based on message-passing and provides connections for
grouping of events in order to create polyphonic phrases,
as well as loops and continuous trajectories. The timing of actions can be relative (to the score tempo) in floating point or time numbers, or in absolute time. Graphi-
cal representation and authoring of such scores is possible
thanks to its integration within the Note/Ability score edi-
tor. Figure 1 in fact an illustration of an Antescofo score feature the instrumental score (green boxes), a contin-
uous trajectory and a symbolic action group (lower staff) as accompaniment actions, all living within a single
score framework. In its original text format, composers are also able to create nested hierarchies within electronic phrases (groups inside groups), employ macro expansions and use data flow functions (see [3] for descriptions).

The role of Antescofo in realtime is to decode the posi-
tion and tempo of the performer within a synchronous re-
active system to best interpret the accompanying section. Compared to previous approaches in interactive computer music, the accompanying actions can be described as relative to the performers' tempos and thus dynamically rescheduled in realtime. Each action's starting point
is relative to its backward closest event in the instrumental score. The phrasing schemes available in Antescofo allow the scope of such electronic phrases to go beyond incre-
ments of the instrumental score.

While the ability of authoring parallel phrases (as op-
posed to segmented and chunked actions) makes the act of
authoring more appealing, such timing overlaps create
important issues both for realtime coordination of events with the live performer and also their authoring. For
example, for a simple concertos-like accompanying setting, one might want to specify the accompanying part as one single (and non-chopped and non-segmented) electronic phrase. This appealing tendency creates interesting chal-
 lenges for synchronization strategies as well as error han-
dering of the virtual accompanying interpreter which are
described in the next section.

3. MOTIVATIONS

3.1. Synchronizing electronic with live musicians

For compositional purposes, it seems more natural to be
able to express accompanying actions as phrases as op-
posed to small segments within instrumental note onsets. However, such additions require explicit and dynamic strate-
gies for handling synchrony between accompanying ac-
tions when their scope goes beyond that of its starting in-
strumental event. The need for such strategies become
even more crucial when tempo changes occurs during a
phrase's launch. Figure 5 attempts to illustrate this within
a simple example: Figure 5a shows the ideal perfor-
mance or how actions and instrumental score is specified
to the system. In this example, an accompanying phrase
is launched at the beginning of the first event from the
human performer. The accompanying here is a simple
group of four actions (that are written parallel to
subsequent events of the performer in the original score.
In an ideal setting (i.e. correct listening module) the
action group is launched synchronously to the onset of
the first event. For the rest of the actions however, the syn-
chronization strategy depends on the dynamics of the per-
formance. This is demonstrated in figure 5b (where the
performer hypothetically deaccelerates the consequent events in her score. In this case, the delays between the actions and their corresponding instrumental event will naturally increase. Such asynchrony is despite correct decoding of tempo from the listening machine. We note however, that this behavior ensures a smooth synchronization with
performer tempo changes, but without any guarantee for po-
sition synchronicity. Although this behavior is desired in
some musical configurations, it seems natural to propose
an alternative strategy where the electronic actions would
be synchronous to the events detection.

3.2. The case of machine listening or human errors

The musical output of an automatic accompanying sys-
tem should not solely depend on its listening module or
even to the human performer at specific circumstances.
This means that there is a human (or system) for ensemble performance. A live music performance output should
not be at stake in presence of any error in realtime.

In a live performance situation different errors may be
encountered: the machine listens and could confuse an event with another, miss an event, or produce a false-alarm.
Additionally, musicians might introduce performance errors
can affect the accompanying results. In all cases, we expect not only that the system continues to work, but also that it reacts as musically as possible. Parts but not all of these errors can be handled directly by the lis-
tening module (such as duration or missed events
by the performer). The critical safety of the accompan-
iment part can thus be reduced to handling of missed events (whether missed by the listening module or human
performer). The natural question to ask in this case is why
the system should do in case of a missed event? Should the
associated actions be performed or not? The answer to
this question seems more musical than technical. In some automatic accompanying systems, one might want to
dismiss associated actions to a missed event if the scope of
those actions does not bypass that of the current event at stake. On the contrary, in other situations such actions might be initializations for future actions.

This discussion shows that while such considerations
can be addressed automatically in special cases, error han-
dling attributes should be first-class citizens in any speci-
fication language for automatic accompanying and inter-
active computer music pieces.

4. PROPOSED APPROACH

The musician's performance is subject to many variations
from the score. There are several ways to adapt to this
musical indeterminacy based on specific musical context. The correct synchronization strategies
are at the composer's or performer's discretion. To this end, we propose explicit synchronization and error handling strategies as attributes for the composer to choose, tak-
ing into account performance variations and to manage the errors of the musician and the recognition algorithm.

In this paper, we provide a verbal description of the pro-
posed approach. The formal and semantical definition of
these concepts are described in [5].

Accompaniment phrase and loop constructs are used
to start a sequence of actions from a trigger event in the
instrumental score. Once a sequence of relatively timed
actions is launched, its synchrony by default is governed
by dynamic rescheduling following changes of tempo from
the musician. However, as seen in section 3.1, knowledge of realtime tempo is not sufficient for precise synchroniza-
tion with events played by the musician as shown in Fig-
ure 5. While this loose synchrony is useful in some mus-
cial context (loose phrasing of electronics for example), it is not always desirable. For a finer synchronization, we
provide the composer the ability to assign a tight at-
ttribute to a phrase block. If a block is tight, its inside actions will be dynamically analyzed to be triggered only on note onsets, relative times and not on
n mutant event in the future. This new feature evades the composer from segmenting the actions blocks to smaller segments with regards to synchronization points and provide a high-
degree of freedom during the composing phase. The
scheduling approach is adopted to implement the tight block behavior. During the execution, the system at-
to synchronize the next action to be launched with the corresponding event using a hybrid strategy employing both tempo and future-event positions.

The problem of error handling, as discussed in section 3.2, boils down to the ability of attributing scores and phrases to accomplishment phrases and blocks. Despite their space occupancy in the score, a block is said to be local if dismissable in the absence of its triggering event during live performance; and accordingly it is global if it should be launched in priority and immediately if the system recognizes the absence of its triggering event. Once again, the choice of an entity being local or global is given to the discretion of the composer.

The combination of the synchronization strategy attributes (tight or loose) and error handling attributes (local or global) for a group of accomplishment actions give rise to four distinct situations. Figure 6 attempts to showcase these four situations for a simple hypothetical performance setup. In this example, the score is assumed to demand for four distinct performer events ($e_1$ to $e_4$) with grouped actions whose two actions are initially specified to occur on $e_1$ and $e_2$. The figure demonstrates the simulation of the system behavior in case $e_2$ is missing during live performance for the four configurations discussed above. Naturally, in our runtime setup, $e_2$ is reported as missed once $e_2$ is detected.

Figure 6. Accompaniment behavior in case of missed event for 4-synchronization and error handling strategies.

It is worth to note that each combination corresponds to a specific musical situation encountered in authoring of mixed interactive pieces:

**local and tight:** Strict synchrony of inside actions whenever there’s a spatial correspondence between events and actions in the score. However actions within the strict vicinity of a missing event are dismissed. This case corresponds to an ideal concerto-like accompaniment system.

**global and tight:** Strict synchrony of corresponding actions and events while no actions is to be dismissed in any circumstance. This situation corresponds to a strong musical identity, strictly tied to the performance events.

**global and loose:** An important musical entity with no strict synchrony once launched. Such entity is similar to musical phrases that have strict starting points with random type progressions (free endings, tempo-synchronous).

**local and loose:** The default behavior of Antescofo, and correspond as discussed to integral musical phrases launched in parallel to the instrumental world and influenced by the environment’s tempo.

**5. SAMPLE RESULTS**

The proposed approach has been integrated in Antescofo’s formal language and adopted in various new music productions involving live instrumental and electronic music. The test best for such systems is to see and use them in action. For the sake of completeness we discuss two sample results: one on a simple automatic accompaniment setting and another in case of a contemporary music production. Curious readers are referred to our website for further videos and events.

In the first example, we attempt to reconstruct the performance of a four-voice fugue by J.S. Bach in B-minor where one voice is performed by a musician and others by automatic accompaniment. Figure 7 shows the score for this performance. In this example, and on purpose, the three accompanying voices are written as three distinct groups of actions (for MIDI accomplishment in this case). Each block corresponds to the accompaniment actions for each voice and the top staff represents the instrumental part. We choose to design synchronization strategies to ensure precise synchrony with a performer and also between different accompaniment voices. To ensure musical consistency, local strategies are assigned to the groups. In this way, missed event’s corresponding actions will be dismissed without altering the overall performance. Despite its intuitive nature in run-time, the way the accomplishment voices are written and handled in real-time provide ease of authoring for composers willing to use such systems for their compositions.

The second example is an excerpt score from the beginning of “Tenets” (2010) for String Quartet and live electronic by French composer Philippe Manoury as illustrated in Figure 8. The left side of Figure 8 corresponds to the hand-written manuscript (excepting both instrumental string section, fourth staves in bottom) and live electronic scores (top staves), and the right side of the figure correspond to its Antescofo equivalent used during live performance. The Antescofo score describes three parallel group actions entitled arco, h1-trans and Fiserrat) corresponding to the left column. The two groups arco and Fiserrat contain discrete and atomic sequences.

Figure 7. Antescofo score of Bach’s B-minor fugue in automatic accomplishment mode, in Notethis Pro.

Figure 8. Score manuscript of the first bars of “Tenets” (left), and the Antescofo counterpart (right).

**6. CONCLUSION**

In this paper we attempted to formalize the problem of synchronization and coordination of actions in an automatic accomplishment setting by considering them as reactive synchronous systems on top of classical score following and machine listening. With such considerations we aimed at addressing the error handling and intelligent synchronization strategies despite human or machine listening error imposed to the system during a live performance. This is in contrast to most existing approaches where the integrity of the musical output is highly dependent on the healthiness of the artificial listening modules at work. This becomes possible by studying the language constructs of both music composition and performance within a computer setting.

The solutions proposed in the paper try to cover various musical situations that correspond to concerto-like accomplishment settings as well as compositional and performative aspects of interactive computer music pieces. They are integrated within the Antescofo software and have been employed in various music productions.

The coupling of high-level computer language constructions with that of low-level machine listening, at the core of this paper, is a necessity in musical practices of interactive computer music and requires more research in the computer music community. We believe that research on these lines could address complex problems with simple and elegant solutions useful for both music composition and live performance.

**7. REFERENCES**


IMAGE-BASED SPATIALIZATION

Eric Lyon
School of Creative Arts
Sonic Arts Research Centre
Queen's University Belfast
Belfast BT7 1NN
United Kingdom

ABSTRACT
An approach to spatialization is described in which the pixels of an image determine both spatial and other attributes of individual elements in a multi-channel musical texture. The application of this technique in the author’s composition Spaced Images with Noise and Lines is discussed in detail. The relationship of this technique to existing image-to-sound mappings is discussed. The particular advantage of modifying spatial properties with image filters is considered.

1. INTRODUCTION
Image-based spatialization involves the mapping of an image to the spatial properties of a sound. Whereas existing literature tends to discuss image-based, or video-based, spatialization systems from the standpoint of introducing compositional tools and software systems, the primary contribution of this paper will be to discuss the specific application of image-based spatialization to the creation of a recent computer music composition.

2. IMAGE-BASED SPATIALIZATION
In the work described here, spatialization is achieved by mapping pixel values to sound grains. Each horizontal line is virtually wrapped around the perimeter of a surround-sound speaker array. The technique for Spaced Images with Noise and Lines targets an eight-channel surround sound speaker tray, with four virtual sound source locations calculated using power panning across each pair of adjacent speakers. Thus, this configuration calls for images with 32 pixels along the x-axis, to be mapped to 32 virtual sound locations circling the audience. The vertical dimension maps to time, and thus may be arbitrarily determined for each image. The tallest the image, the more perimeter lines are converted to sonic moments, each with its own spatial profile. Scanning down the lines of the image advances time at a fixed interval. (The time interval could be warped, rather than constant, if desired.) In a given line, the darkness of each pixel represents the amplitude of an event, or grain, at the virtual spatial position location associated with that pixel. An image with a single black line drawn diagonally downward, with a pixel-width of one, would result in a perimeter-panning spatialization effect.

3. IMPLEMENTATION
A Max patch is used to convert an image to a Csound score, which is then compiled into an 8-channel audio file. (See Figure 1.) Each 32-pixel line is transformed into a list of floating point values from 0-1, which drive a JavaScript program that writes its output directly to a Csound score.

4. SPACED IMAGES WITH NOISE AND LINES
Spaced Images with Noise and Lines was composed in 2011 and premiered at a Spatial Music Collective concert at The Joinery in Dublin. All passages in the work were generated as described above, using images that the author created with the free image-processing software, The Gimp [1]. The images are comprised of noise, generated by The Gimp, and freehand, mouse-drawn lines. Prior to audio conversion, the images were subject to various forms of image processing, such as convolution, erasure, and shearing. (See Figure 2.) Most passages were generated from a single image, but in one case, sounds generated from two complementary images were superimposed to create a two-voice spatial trajectory counterpart.

Figure 1. A Max patch to convert an image to a Csound score (based on a design by Shawn Greenlee).

5. MAPPING STRATEGIES
The simplest mapping strategy involves routing a single sound to each virtual location, scaled by the amplitude determined by the darkness of its corresponding pixel. This is the duplication technique described by McGee and Wright [6]. This strategy is applied to the first sound heard in the piece, which is a synthetic noise with a percussive envelope and a resonant peak. The changes in the image with its erasures, noise, and lines, produces a spatial result that fluctuates between a somewhat indeterminate sense of motion, and clear, dramatic spatial trajectories. The use at the outset of a simple, spectrally stable sound focuses the listener’s attention on spatial attributes. Successive mappings introduce greater variety among the sounds. In one instance, the y-axis is mapped to resonant frequency, so that the pitch descends as the corresponding image is scanned downward. Several sections assign fixed resonant frequencies to different virtual locations, while still implementing the darkness-to-amplitude mapping to articulating sounds. The result is a spatialized chord, where the image is noisy, and melodic when the image is line-based. Both visual elements were generally admired in the source images.

In another section, enveloped sine waves are employed. The frequency for each sinusoidal grain is randomly chosen from a fixed, equal-tempered scale. In homage to Karlheinz Stockhausen, the frequency scale is taken from his Studie II [11]. In this section, the pixel amplitude controls two synthesis parameters: the amplitude and duration of a corresponding sinusoidal grain. This section was generated from an image that superimposes two kinds of noise – a low-amplitude, high-density noise, and a high-amplitude, very low-density noise, with just a few pixels activated. The result is a constant, quiet, sort of thudding background articulation (albeit spatially complex) from the low-amplitude pixels, and a sporadic, bell-like surface melody with clear spatial articulation from the high amplitude-pixels. Although sine waves are often difficult to resolve spatially, the single pixel mapping to a fixed virtual location, combined with the percussive attack envelope, locates the individual sinusoids starkly in different regions of the space.

Figure 2. An image from the opening of Spaced Images with Noise and Lines.

6. RELATED WORK

Examples in the digital domain include Christopher Penrose’s HyperUpic [8] and Metasynt by UI Software [12]. Another technique considerably closer to the work described here maps image onto parameters other than space, often to frequency. Here could be mentioned Oramics[7] and Iannis Xenakis’s Upic system [5]. Examples in the digital domain include Christopher Penrose’s HyperUpic [8] and Metasynt by UI Software [12].

7. ADVANTAGES OF IMAGE-BASED SPATIALIZATION
Working from an image provides many of the advantages of a graphical score, including the ability to easily specify gestures and density at multiple time points. The second section of the piece focuses on sampled drum sounds. The initial treatment of drums starts with a hi-hat, which is reminiscent of the opening filtered noise sound. Very shortly, a full ensemble drum texture emerges, and then gradually destabilizes as individual attack times are subject to increasing random deviation. Individual samples are assigned to fixed virtual locations, which are then articulated according to pixel amplitudes. Random delays, ring modulation and filtering are gradually introduced, complicating the timbral space. The texture gradually thins until only the cowbell sample remains.

The work closes with two contrasting sections. First comes the only slow section, in which pixels articulate sinusoids of 10 seconds duration, with frequencies that glide slowly downward. The time interval between line scans is approximately 500 ms. The concluding section uses a line drawing to articulate enveloped, filtered noise, with resonance frequencies locked to virtual locations. With only lines used in this image, the spatial trajectories of this concluding passage are exceedingly clear to the ear.
8. COMPARISONS WITH VIDEO-BASED SPATIALIZATION

Spatialization with video would seem to have certain advantages over image-based spatialization. For example, there is a more direct conceptual mapping between an individual video frame, and a surround speaker array, at least for 2D surround sound arrays. Video lends itself to real-time processing in a way that is unavailable for image-based spatialization. However, the temporal association between the video and audio is a mixed blessing. Images used for spatialization are not to be viewed during performance, and therefore their aesthetic qualities need not be considered. (Of course it is possible to create video-based spatial sound in which the video itself is not presented to the audience. However, every example I have encountered of video-based audio spatialization has been presented as an integrated audio-visual composition.) In audio-visual work, the imagery must be as aesthetically strong as the sound, and the two must work well together. This requirement could put constraints on the content of the video. A situation in which both audio and video are presenting the same information introduces the danger of aesthetic redundancy. Applying filters to modify spatial behavior is much easier and more direct for image processing than for video processing. Nonetheless, video-based spatialization can still be quite effective, especially when drawing on the advantages of live performance. In any case, despite some overlap in the manner of data processing, video-based spatialization and image-based spatialization remain two distinctly different approaches to spatialization.

9. AESTHETIC RESULTS

In “Space-form and the Acousmatic Image,” Denis Smalley defines space-form as “An approach to musical form, and its analysis, which privileges space as the primary articulator. Time acts in the service of space” [10]. In the same article, Smalley makes a distinction between interventionist and naturalist approaches. “With the interventionist approach the composer’s hand is in evidence, and the stamp of the technology and techniques is apparent in the kind of material and the way it is manipulated, whereas in the naturalist work there will be some attempt to hide techniques, and avoid exposing technological signifiers” [10].

Spaced Images with Noise and Lines could accurately be described as an interventionist space-form composition. From the outset, the spatial aspect of the work is kept in the compositional foreground through dramatic panning trajectories. Sonic elements are deliberately kept simple throughout in order to highlight the spatial aspect. Different approaches to texture creation, and the admixture of noise and lines in the images, provide a broad variety of spatial experiences to the audience. Perhaps most importantly, the spatial structures are integral to the music. The complexity of the spatial relations is built into the musical structure; it could not have been imposed externally through performed manual diffusion techniques.

10. FUTURE WORK

Only greyscale image processing was used in Spaced Images with Noise and Lines. It would be interesting to use the separate RGB channels to affect different aspects of synthesis, and/or spatialization. The use of image masks could facilitate the creation of various forms of spatial polyphony. The mapping described here makes the most sense in the context of 2D speaker arrays. A rethink of the spatial image mapping strategy may be appropriate for 3D arrays, such as the Klangdom at ZKM, or the Sonic Lab at SARC.

11. CONCLUSIONS

The technique of image-based spatialization was discussed in the context of the author’s composition Spaced Images with Noise and Lines. The creative potentials of the technique were shown to be somewhat different from more traditional trajectory-based spatialization techniques. Given the growing number of multi-channel performance venues, and opportunities for composers and audiences to explore spatial music, techniques such as image-based spatialization may be of increasing interest to composers of computer music.

12. REFERENCES

MCONDUCT: TRANSCENDING DOMAINS AND DISTRIBUTED PERFORMANCE

Joanne Armitage, Phoebe Bakanas, Joel Balmer, Paul Halpin, Kyle Hudspeth, and Kia Ng
ICSRR – University of Leeds, School of Electronic and Electrical Engineering, School of Music & School of Computing, Leeds LS2 9JT, UK
mconduct@icsrri.org.uk, www.icsrri.org.uk

1. INTRODUCTION

Distributed performance is a compositional technique that facilitates the spatialisation of sound within an ensemble. In its practice, performers can be positioned in such a way that they are in close proximity to each other. In some cases, not all performers are able to see the conductor. The lack of visual communication between conductor and ensemble can cause synchronisation issues within the ensemble. This paper proposes a tool for both composers and conductors to enable the spatialisation of sound in real-time interactive multi-modal feedback.

2. BACKGROUND

2.1. Distributed Performance

The spatial dispersion of performers is apparent in any performance whether deliberate or not, due to the inherent positional relationship between both performers and the audience. The most recorded example of deliberate spatialisation in Western music dates back to the liturgical practice of psalmody (the chanting prayers in a religious scenario) in the 9th century. The technique was employed by classical composers including Mozart, who composed two notable sacred works; Serenade No. 6 and (K. 239) and Serenade No. 8 (K. 286). Serenade No. 6 was composed for two orchestras, with one acting as a distinct solo string quartet. Antiphony was created in through separating the choir into two distinct sections [1]. During the 16th century this developed into polyphonic works that were composed for St Mark’s Church, Venice, to enhance both echo effects and acoustic illusion. It has been suggested that the two groups would have been placed in opposite corners of a room to create a spatial distinction between them [3]. In Serenade No. 8, Mozart wrote for four orchestras each consisting of four strings and two horns. He does not specify the spatial distance between the performers. However, he creates an artificial space through dynamics and development of material.

In a 1964 performance of Mozart’s Serenade No. 8 at Symphony Hall, Boston, one orchestra was positioned on the stage, two on the floor and the final orchestra under a canopy. An additional conductor was required to direct this orchestra through synchronising with video footage [4]. This represents one solution to the communication difficulties in distributed performance. Multiple conductors are also used in Schnebel’s, Ekstasiss, Steurer’s, Stockhausen’s Gruppen Für Drei Orchester and Xenakis’ Duel. Another method, commonly used by conductors, is to view the conductor using a mirror. These methods present restrictions on the movement of and visual perspective the performer.

During the Classical and Romantic periods, spatialisation was generally applied to a standardised symphony orchestra in the form of noted dynamic differences and echo effects through repetition [1]. Through the late 19th century composers began to specify positioning of spatialisation effects, an example of this is Mahler’s, Symphony No. 2 where brass percussion instruments are offset and Symphony No. 8 where offset brass appear at the end of each movement [5].

Stockhausen identifies location as a musical parameter that can affect the tonal characteristics of a sound alongside pitch, duration, loudness and timbre [6]. Many contemporary composers assign instrumentists specific positioning in their composition performance notes. Through specifying sound paths and sound source locations the composer has direct control of the piece’s overall acoustic balance. Spatialisation of sound can be applied in a number of ways in both acoustic and electronic compositions.

With the development of electroacoustic music in the 20th century composers were able to control sound and sound trajectories directly. Arrangements of loudspeakers can be used to form the spatial characteristics of a sound. An example of this is Barret’s recent composition Construction; involving 22 musicians, live electronics and a 16-channel sound installation [7]. Positioning acoustic performers in a professional performance environment eliminates the need for additional equipment such as loudspeakers and microphones. The separation between performers can obscure their view of the conductor leading to logistical difficulties in performance. Barret’s observations on spatiality in music reflects this: “Separated groups are difficult to coordinate - exact rhythmic simultaneous are almost impossible because of the distances between the musicians.”

Brant, 1967 [8: 224]

With the proposed system this difficulty will be addressed. The three orchestras in Stockhausen’s, Gruppen für Drei Orchester, could be controlled by one central conductor [9]. Thus, the separation of the separate conductors could be enhanced without affecting the spatial distinction between them. In this application, the system approaches the coordination difficulties in distributed performance through interactive multi-modal feedback of a conductor’s gestures.

2.2. History of Conducting

The art of conducting has a long and well-established history of using arm and hand gesture to convey musical intent and coordinate a group of players to perform a set piece with precise and delicate synchronisation and timing. The gesture communication is required to control a range of musical parameters; for example, indicating entries, setting tempo, synchronising the various instruments and parts, and shaping the ensemble’s sound. Conducting has not always been so rhythmically focused. Cheironomy, the earliest form of conducting, can be traced back to early Egyptian performances. Before melody was notated in a written score, cheironomy hand signs indicated the melodic shape [10]. Cheironomy was widespread in the ancient world and it endured into medieval times, being used by the choric conductor. However, during the Middle Ages, music became more complex. The melodic line expanded to have more pitches sung upon a single syllable as well as more pitches in the context of the melody. The melodic pitches within the octave also expanded allowing ensembles to have looser tonality. In addition to the development of polyphony, music also became more rhythmically complex. Cheironomy could not keep up and lose its effectiveness and the complexity of the polyphonic music compelled the development of staffed musical notation [11, 12].

During this transitional time, in the Christian church the person giving musical signals held a staff. As music became more complex, the staff was moved up and down to indicate the beat indicating it was an early form of a baton keeping the ensemble together [13]. During the 17th century, rolled up sheets of paper or smaller sticks were used to signal beat and tempo. The first conductor to utilise a baton can be traced back to 1794, when the Belgian composer Guillaume-Alexis Paris used a baton while conducting opera in Hamburg, Germany. As orchestra size increased, conducting became a fixed role in the orchestra. By the early 19th century, it became common practice to have a dedicated conductor, an individual who did not play in the ensemble. Also in the 19th century Wagner developed a theory that shaped the role of conductors today [13]. He believed that conductors should not only keep time but also impose their own interpretation of the piece. Monteverdi’s Baroque singing is built upon the traditional techniques founded by Wagner and other early conductors. However,
conducting is still an evolving art form. Not only does modern music necessitate the use of new conducting techniques but also technology is opening doors for new types of music and new interpretations of conducting.

2.3. Conductor Tracking Systems

A variety of conducting gesture analysis and performance systems have been developed. The first documented mechanism of the conducting baton was during the 1830s in Brussels. The system relayed the conductor's tempo to an offstage chorus through an electromechanical device, similar to a piano key, which would complete an electrical circuit and turn on a light when pressed. Hector Berlioz documented the use of this device in his essay entitled "On Conducting," published in 1843. Although the need for technology to aid distributed performance has existed since the 1800s, today the only common methods for conducting distributed performances are secondary conductors or video monitors. [14]

Since that first effort, there have been many other attempts to automate the process of conducting. However they are primarily motivated by reasons other than distributed performance. The most common objective for these systems is conducting electronic instruments or a virtual orchestra. The term 'virtual orchestra' was first introduced into the musical lexicon in the early 1990s by Bianchi and Smith [15]. Bianchi and Smith developed an interactive computer music system that was used in the Kentucky Opera's 1995 production of "Hansel and Gretel." This marked one of the first use of technology by a major performing arts organisation.

Another example of an early virtual orchestra is the system created in 1991 by Morita, Hashimoto and Ohteru [16]. Their electronic orchestra responded to a conductor's gestures whose movements were tracked through a Charge-Coupled Device (CCD) camera and a sensor glove. Morita et al. categorise the tracked conducting information into two main functions:

i) Basic that includes notes, pitch, frequency, duration.

ii) Musical performance expression (Mpx), such as ritardando, sostenuto, dolce.

The basic information is quantifiable and necessary when performing a piece. The Mpx information is subjective and creates the artistic essence of the performance [16]. Ascertainment beat points to indicate tempo is a minimum requirement for this system. Expanding upon this to measure gestural expression is fundamental in creating an authentic reconstruction.

In 1996, the "Digital Baton" [17, 18] was designed as a multipurpose device to control electronic music through traditional conducting parameters such as tempo, dynamics and duration alongside individual notes and details of particular sounds. Gestures are tracked using accelerometers, infrared LED and piezo-resistive strips. A similar array of sensors will be used in this project.

While many conducting gesture analysis systems focus on controlling a virtual orchestra, for other systems conducting is a research domain that allows the conductor to engage in mapping the conductor's expressive features to a musical score, the aim of analysis in this system will enhance the conductor's control of a distributed ensemble.

Other systems aim at conducting pedagogy. Examples include Peng [20], and Bradshaw & Ng [21, 22] conducting analysis systems. These two projects used a Wii-based tracking system to capture the conducting gesture. Bradshaw and Ng explain how their conducting tracking system can be used in different pedagogical scenarios. Firstly, their system will allow conducting students to express themselves. Secondly, they allow view conductors to view the conductor's conduct as they conduct by comparing the tempo between their own beat points to a set tempo on a metronome. Thirdly, Bradshaw and Ng's system can break down and analyse gestures for their students to analyse. Additionally, an experienced conductor can compare recordings of her/his gestures to highlight different interpretations. Secondly, the system allows conductors to view how consistently they conduct by comparing the tempo between their own beat points to a set tempo on a metronome.

Many other conducting tracking systems have been developed including: Lee et al. [23]; Borchers et al. [24]; Katayose and Okada [25]; Braugge et al. [26]; Nakra et al. [27]; Baba et al. [28].

2.4. Reconstruction of Data

An understanding of feedback technologies is required to successfully reconstruct the inherent musical information and expression of conducting gesture into another domain that allows performers’ visual conductors to be obscured. For this specific application gestural data must be relayed by non-visual and non-cochlear means. Specifically, haptic feedback was researched due to its potential for effective and non-intrusive transmission of data. Data mapping is used widely in multimedia projects and involves translating data across different domains. Eacott’s ‘Flood Tide’ sonifies tidal speed information through mapping it to produce live notation [29]. Bradshaw and Ng [30] analyse the conducting gesture through sonification and visualisation. Robertson et al. [31] designed a beat tracking algorithm for real-time beat visualisation. In this system, data will be mapped to convert gestural conducting data into a different domain, in this instance vibrotactile.

Non-intrusive, trans-domain mapping of data will be required to reconstruct the visual element of conducting. ‘Music via Motion’ (MvM) [32] is a framework that facilitates the trans-domain mapping of movement to another multimedia domain. The modular architecture of the systems using MvM has influenced the design of this system particularly in respect to its potential for multiple applications and multimedia reconstructions.

Raisamo et al. [33] utilised haptic feedback in their research: They performed experiments to understand emotional communication using cutaneous salutation. Similarly, Lemmens et al. [34] designed a vibrotactor jacket with 64 actuators to enhance the emotion and immersion of the cinematic experience. The test subjects wearing the vibrotactor jacket interpreted vibrations as perceived movement that correlated to emotive visual stimuli. The results essentially informed the design of this system, however their focus was to improve the experience for the audience, whereas the proposed system aims to convey expressive information from the conductor to the performer.

3. DESIGN AND DEVELOPMENT

The system needs to be able to obtain and analyse conducted gestural information, particularly direction, intensity and rhythm, and then translate this to a vibrotactor domain in a way that can be easily and accurately interpreted by a performer. In order to achieve this goal, unique features of conducting motion capture and reconstruction are required, with the ability for the data to be received by multiple devices simultaneously. The complete design of the system can be broken down into four distinct sections: data capture, analysis, mapping, and reconstruction.

Whilst designing the system, the weight and dimension of the baton and the receivers were considered to be highly important. The baton device has to be both small and non-intrusive in order for the gestures to be performed comfortably and naturally by the conductor, otherwise the captured gestures would not accurately represent the conductor’s intentions. The receiving device that provides feedback to the performers has to be similarly non-intrusive so as to not distract the performers. The device for gesture capture requires a number of sensors to effectively measure the motion of the conductor’s baton in multiple dimensions. The primary sensors used in the baton are an accelerometer that provides acceleration information in three dimensions, and a gyroscope and magnetometer that provide orientation information. Other sensors, such as flex sensors, and distance sensors are currently being tested within specific use cases.

Figure 1. Final baton prototype with the sensors inside the handle.

While many conducting gesture analysis systems require wired connections for both the baton and receiving units would be detrimental to the non-intrusivity of the system; having cables running from conductors and performers during a performance would be likely impact their movements. Wired connections also contain many more practical problems in terms of cost and aesthetics; long trailing cables are unsightly, and can become expensive, particularly if a number of receivers are used. For these reasons, several wireless protocols were considered. ZigBee was chosen due to its low power consumption and appropriate range (approximately 40m indoors) for this application.

Once captured, gestural information is analysed so that it can be effectively used in a way that will allow accurate reconstruction of conducting movements into a different domain. This analysis is performed through software and involves data smoothing and conditioning.

Figure 2. Overall system architecture.

After analysis, the information is broadcast wirelessly for use in the system and actuator unit for physical translation of the motion with vibration. The data is...
reconstructed in the haptic domain for the performer to interpret. To achieve this, the system uses coin type Eccentric Rotating Mass (ERM) vibration motors. The actuators are 10mm in diameter and thickness of 3.4mm. The strength of their vibration is controlled by a pulse width modulation (PWM) signal, the duty cycle ratio of which is determined by the data captured from the baton device. The directionality of the conducting gesture is conveyed by using 5 actuators that are arranged in a cross layout. The direction of movement is translated into vibrating sequences in the actuator units. See Figure 1 for final baton prototype and Figure 2 for overall system architecture.

4. EXPERIMENTATION AND VALIDATION

Through preliminary experimentation it was determined that both beat points and time signatures can be successfully conveyed via vibrotactile feedback. One experiment determined that people can distinguish between vibrations on multiple actuators: Four actuators were attached to a subjects leg, marking points of a cross, so that subjects could identify which actuators were vibrating. In addition, subjects could tell if the vibrations were passed vertically between the top and bottom actuators or horizontally between the right and left actuators. However, participants had difficulty identifying any vibration patterns that required diagonal movement between the right or left actuators to the top of bottom actuators. It was decided that a redesign was necessary to better haptic recognition. A fifth actuator was introduced in the centre of the cross. An experiment was then designed and carried out to assess how well vibration patterns could be distinguished with five-actuator setup. The experiment used a program that switched on specific vibrating motors in sequence to represent an input gesture understood by the listener differ depending on their position. Audience members sitting equidistant from each group will experience exact temporal similarities and the colour of the sound will be at full intensity. The audience as a whole could be perceived as a matrix of pixels that receive varying intensities of RGB information.

Utilizing new developments in movement sensing and gesture mapping technologies, the co-ordination difficulties have been addressed by the mConduct system as presented in Section 3. The mConduct system’s primary function in the performance is to relay a haptic reconstruction of the conducting gesture to the musicians allowing their view of the conductor to be obstructed. The system reconstructs complex time signature changes through the multiple actuators, allowing the performer to recognise the exact positioning of a beat within the bar. The system is required to enhance the simultaneity of the ensemble. A haptic receiver is placed on the cuffed of a performer in each subgroup, including those in view of the conductor. This ensures that each performer receives the same information. There are different types of modality the system can produce. Continuous vibrations that anticipate the changes in gesture can be used to enhance synchronisation. Directional tracking can be used to convey a performer’s position in the performance and this is discussed in Section 4. Different vibration modalties are applied to the performers depending on the musical content of a section.

5. APPLICATION IN COMPOSITION

Transcending Domains is a piece written for distributed performance that intends to create a perceivable link between an image and sound. This image is spatially representing red, green and blue (RGB) pixels of an image as musical parameters. The image used in this composition is a photograph taken by the photographer Winty Baker [36]. The pixels in its entirety is seen as a collaborative effort. The piece, with the use of the mConduct system, has been submitted and selected for performance at the ICMC 2012.

Pixel information is extracted from the image and used as a data source in this piece. A program was created using Matlab [37] to extract the image's RGB intensity values into their respective two-dimensional matrices, each matrix representing the pixel’s original position. Musical parameters such as pitch, rhythm, dynamic, time signature and tempo are derived from the matrices. A unique synthesis model for audio-colour representation is too subjective to be applied in this context. Instead, an intuitive and personal approach was taken to associate timbral qualities of an instrument with context. Instead, an intuitive and personal approach was taken to associate timbral qualities of an instrument with context. A haptic receiver is placed on the cuffed of a performer in each subgroup, including those in view of the conductor. This ensures that each performer receives the same information. There are different types of modality the system can produce. Continuous vibrations that anticipate the changes in gesture can be used to enhance synchronisation. Directional tracking can be used to convey the performer’s position in the performance and this is discussed in Section 4. Different vibration modalities are applied to the performers depending on the musical content of a section.

In this composition, pixel data from a photograph has been used in a number of ways to define musical parameters such as dynamics, rhythm, pitch and articulation. The composition reflects the shapes, movement and colours in the photograph in an auditory domain and enhances the visual imagery. Performance is further enhanced through application of the mConduct system.

The variable colour intensities are represented through spatialisation as well as the other musical parameters. Audience members are positioned relative to the ensemble subgroups and the temporal simultaneities heard by the listener differ depending on their position. Audience members sitting equidistant from each group will experience exact temporal similarities and the colour of the sound will be at full intensity. The audience as a whole could be perceived as a matrix of pixels that receive varying intensities of RGB information.

Using new developments in movement sensing and gesture mapping technologies, the co-ordination difficulties have been addressed by the mConduct system as presented in Section 3. The mConduct system’s primary function in the performance is to relay a haptic reconstruction of the conducting gesture to the musicians allowing their view of the conductor to be obstructed. The system reconstructs complex time signature changes through the multiple actuators, allowing the performer to recognise the exact positioning of a beat within the bar. The system is required to enhance the simultaneity of the ensemble. A haptic receiver is placed on the cuffed of a performer in each subgroup, including those in view of the conductor. This ensures that each performer receives the same information. There are different types of modality the system can produce. Continuous vibrations that anticipate the changes in gesture can be used to enhance synchronisation. Directional tracking can be used to convey the performer’s position in the performance and this is discussed in Section 4. Different vibration modalities are applied to the performers depending on the musical content of a section.

In this composition, pixel data from a photograph has been used in a number of ways to define musical parameters such as dynamics, rhythm, pitch and articulation. The composition reflects the shapes, movement and colours in the photograph in an auditory domain and enhances the visual imagery. Performance is further enhanced through application of the mConduct system.

The musical material of this composition has been produced through strict processes that have resulted in a number of technical complexities particularly in regards to time signatures, melodic lines and rhythmic structures (see Figure 3). This, combined with the spatialisation of the ensemble subgroups has led to co-ordination difficulties in practice due to the requirement to disseminate specific conducting information to each performer from a distance.

The main objective is to use the baton to enhance distributive performance, it can be used in other applications. It is said that two thirds of communication is through non-verbal channels [38]. Gestures provide information about a person’s emotions and relationships with others, and this research can help further the understanding of gesture communication. The data collected from the sensors can be analysed to explore the
relationship between conductor and musicians and their dependence on each other. Knowledge about the communication between the conductor and musicians can then be applied to gesture communication at large. The application of this system will have an important role in human computer interface. The system also aids the pedagogy of conducting. Virtual orchestra training programs can use the empirical data on how musicians synchronise with the conductor in their mappings of gestural interaction. Recorded conducting gesture can also be preserved for future generation to understand the different interpretations and expression of a specific time period or conductor.

7. ACKNOWLEDGEMENTS

Many thanks to photographer Laura Baker and composer Dr. Michael Spencer for his advice and support through the compositional process. We would also like to thank Dr. Mark Marrington for his support in the development of the mConduct system.

8. REFERENCES


[27] Laura Baker Photography, Internet: laurabakerphotography.co.uk/ [Feb. 19 2012].


SOUND SIMILARITY AS INTERFACE BETWEEN HUMAN AND MACHINE IN ELECTROACOUSTIC COMPOSITION

Hanns Holger Rutz
University of Plymouth
Interdisciplinary Centre for Computer Music Research (ICCMR)

ABSTRACT

We present an investigation into the use of sound similarity and dissimilarity measurements as a compositional strategy and, more generally, as an example for a material process which functions as an interface between the composer and the computer in the electroacoustic arts. It questions the role of generative and automated composition processes, and argues that a domain of engineering—signal processing—can be re-appropriated as an interface in artistic research; an instance not just between the face of the composer and the screen, but rather an oscillating aggregate state between technical rigidity and epistemic vagueness. The argument is led along three complementarily pieces for realtime performance and fixed medium which look at sound similarity from three different angles.

1. INTRODUCTION

Both electroacoustic music as well as many live electronic pieces are defined by their need to create structure from the analysis of actual sounds. While the former is still mostly guided by human manipulation, the latter presents the composer with the situation of requiring analytical “help” from the computer in a realtime situation. Nevertheless, we think that the former can largely benefit from approaches which bring together algorithmic thinking and formalism while maintaining the “prerogative of the ear”. In a simplified view of electroacoustic composition, the material is organised subject to this prerogative, and treated by three basic means: agglomeration, transformation, and division (originality cutting and splicing tape). Systems for computer aided composition on the other hand think of musical material often as symbolically representable. What we seek is thus a method which can algorithmically handle aggregate sound objects.

With the advances in signal processing, the possibilities of unifying electroacoustic and algorithmic work come into reach. We prefer the plain old term ‘Signal Processing’, because we are uncertain of the ontological status of ‘Music Information’ as in Music Information Retrieval, ‘Learning’ as in Machine Learning, or ‘Intelligence’ as in Artificial Intelligence. Instead we take a pragmatic view that sees human composer and computer engaged in a linked effort to produce a piece of art which has a hybrid nature between algorithmic and electroacoustic.

In place of a recourse to the theoretical discussion of this matter, three recent pieces of the author are presented which are situated in three terrains: a live electronic piece, a sound installation, and a fixed media composition. The reason why multiple pieces are considered is to draw a trace of their motivation as well as showing how this motivation has been successively adjusted. By doing so, we hope to support the central idea of applying Hans-Jörg Rheinberger’s “experimental systems” [11] as a model for sound composition, a model of the interplay between human and algorithmic composition. To situate and connect these three compositions on a map is to highlight the diachronic character of the reproduction of ideas surrounding a type of sound organisation that they share—the use of audio feature extraction and (cross-)correlation for sound similarity. We use a bit of a fish eye lens which puts the last piece, presented in the conference, into the greater focus.

2. INTER-PAY / RE-SOUND

The piece to begin with is Inter-Pay / Re-Sound (2011) for amplified dereelicnt piano, human performer and computer. The performer is equipped with a miniature microphone and is asked to focus on highlighting the physical structure of the piano and rather avoid producing “piano music” sounds. The sounds get amplified and are projected back onto the corpus of the piano through a series of transducers, making it a spatially extended and reverberating loudspeaker. The computer registers the sounds produced in this first improvisatory part of the piece, and extracts their temporal and spectral features. An interplay begins with the computer using a selection of algorithms which operate on the recorded and analysed material, or later bring fully in charge of the situation and dismissing the performer who becomes another listener. The feedback circle is now closed by having the computer turn its observation onto itself (running the analysis against its own output signal). Although the piece is now in a generative mode which could go on forever, parameter trajectories chosen within given randomised bounds provide for the piece to finally decay.

During the first part, the live signal is continuously written to an audio buffer as well as a feature buffer holding the evolving Mel Frequency Cepstral Coefficients (MFCC) of the signal. Furthermore markers from an onset detector are stored.

The algorithm comprises eight concurrent processes, iterating through three states, idle, “thinking” (analysing without producing sound), and “playing” (producing sound). Of these, four transform, filter and inject material without a dedicated analysis stage. The other four use extracted features from the material: Process HEARING measures averaged loudness and replays short transients between onset markers when the loudness falls below a time varying threshold. Process EQUILIBRIOCPTION searches for steady portions of flat (noisy) spectra in the developing live audio buffer, and plays them back as short concurrent loops.

The two remaining processes TOUCH and SIGHT are based on sound similarity. TOUCH uses a set of given sound profiles, previously stored MFCC evolutions of short (0.5 . . . 1.0 sec) duration, taken from prototypical sounds such as flageolets on the strings, the empty keys (the key mechanics of the piano have been intentionally disconnected) and objects requested as preparation of the strings (e.g. a piece of paper clamped between adjacent strings). The process then searches through the available live buffer for the best matches according to a similarity measure, cuts out small portions from the buffer at the matching positions, applies a spectral whitening filter to make the sound distinguishable from the live signal and appear more remote. Figure 1 shows an example search result across a fully filled buffer.

On the other hand, SIGHT looks at a subframe of the current DSP graph and picks up its sum signal, measuring its short term spectral evolution and using this as a template here instead of finding best matches, it generates a similarity control signal across the whole available live buffer. Subsequently, a condensed version of this buffer is created by applying a sieve: Only those sounds which satisfy a minimum similarity threshold, and those sounds are moved together to form a new homogenouse gesture, with a duration ranging from a few to dozens of seconds. The overarching thought is to create semi-autonomous structures, for example by amplifying the background and creating space. This space is the space between human and computer, not as alienation, but to allow both to breathe and coexist.

3. WRITING MACHINE

In Writing Machine (2011), the performer is gone, and we are left with a sound installation which runs fully independent of human interaction. Furthermore, the framing of using sound similarity in an algorithmic way changes. This installation bases its notion of “writing” on the broad meaning as formulated by Jacques Derrida: The graphiceme is the manifestation of the process of writing-as-trace, an infinite chain of signification (and an absence of “presence” or re-presence of an original signified) [4]. Signification thus as transition, highlighting the fact that something differs. The chain is reflected by the geometric shape of the circle. A poriferous circle, however, which acts as a system of consumption. The machine “eats” a live signal, preferably one with conventional signification, such as from a television channel. This signal is fed into an uninterested database—ignoring all ideology surrounding its interpretation (what Derrida calls “metaphysics”)—from which all sound gestures are derived. The arrangement is that of a laboratory experiment...
sound’s position, adding to the palimpset’s clarity. On the other hand, a notable characteristic of the Writing Machine is that the database is constantly drained, so each sound chosen has a certain chance of being removed from the database, making space for refilling—and with the live signal, the material is never exhausted.

The construction is much more “process” than “product”, although, as Nick Collins calls to mind, this distinction may conceal the more interesting question of the «loci of human decision making» [3, p. 298]. Indeed if we carefully follow the implications of the idea of grapheme, we see that ultimately Deciding (Forming) ⇔ Writing ⇔ Observing ⇔ Tracing. The human composer only seemingly is the origin of decision making. And so this question is as problematic as speaking of «…the derivative intentional-ity of writing» (Searle 2000: 20), where the code is predefined by a human author who then yields moment to moment autonomy of execution to the machine. ⇔ [2].

Intention is only one part of the writing process, perhaps the most shady one. Lars Löfgren points out that in any language, including computer languages, we find two opposite motions. Description as a process of mapping possibly unbounded ideas onto a finite set of symbols, and interpretation as the inexhaustible process of translating a description into meaning [8]. But the fragmentation in the middle, the breaking down of thoughts into discrete steps in an algorithm, introduces its own signification which may relegate “original” intention and meaning.

4. LEERE NULL

The last example, the fixed media composition Leere Null, was actually begun before the sound installation, however it was amended in 2012, and therefore is treated last in this paper. The intermission is responsible for the rather strange form: The first part being a five minute stereo tape, somewhat pedantic and focused on the process, the second part a four channel tape of around eight minutes, a restate-

Figure 2: Left side: photo of the sound installation, size approx. 2 × 2 m. Right side: Diagram of the algorithm.

(see fig. 2)—a tableau of petri dishes, put into vibration by piezo electric elements, and displaying heaps of graphite powder, the disintegrated mineral which borrows its name from graphen (to write). There are two iterations: One describing the algorithmic cycle of departing from the previous situation or phrase, choosing parts of the phrase to be overwritten, then looking for “suitable” materials in the database and overwriting those parts, in other words re-writing the phrase. These steps are shown on the right hand side of fig. 2. Multiple piezo elements are grouped together to form channels, and the phrase, whenever it is restated, advances to an adjacent sector of the physical tableau, performing thus a spatial rotation over the course of a few minutes.

Everything is controlled by motions which are basically ultra-low frequency step wise oscillators or random walks. There is a motion for the duration of the fragments overwritten, for the position of the fragments within the current phrase, for the cross-correlation length when segments overwritten, for the position of the fragments within the database, making space for refilling—and with the live signal, the material is never exhausted.

The point of departure are utopian ideas about composing. Of course, utopian ideas are another form of para-doxes, as we realise them in our imagination, and thus they appear within our grasp, yet they remain in a non-place. Some are virtual as they pervade the collective mind: The composer is the originator—faced with a blank sheet, he creates ex nihilo. This is one way to explain the title which is taken from a novel by the brothers Strugatsky. While the term “Empties” is used in the English translation, the German version is peculiar in that it uses a pleonasm: “Leere Null” means empty (or void) zero. In the book it is used to describe objects allegedly left behind by aliens: Two copper plates spaced apart with “nothing” in between them, and no force can tear the two apart. The Empty fascinates due to a paradox. It is fully permeable for the senses—you can put your hand inside—, yet its meaning is completely opaque. Like the blank sheet, the object could be seen as a container—the contour of the zero—which is susceptible to be filled with our imagination (rewriting the opaque meaning).

The genre of “tape” music imposes another paradox or obstacle: During the writing of the piece, the composer can hear the sounds and gestures it is made of in arbitrary succession, indeed they can be moved around on the “time canvas”, looking down onto them with the comfortable vision of the bird’s eye. However, when the piece is presented, only one path in the decision space is left to be presented. This problem is not unique to electroacoustic music, but the “double listening” greatly emphasises it.

The first part of the piece is a play with these two elements: To create ex nihilo (despite the impossibility), and to reverse the arrow of time. The solution interrelates these two. The approach lies again in the use of sound similarity search and in re-writing of phrases with the help of this search. For this it was necessary to create a multi-tracker timeline editor which could be programmatically controlled, and this tool has been published as an open source project1. It allows the construction of timelines both manually via a graphical user interface, as well as through programs written in the Scala language. Audio regions are shown as sonograms and can be manipulated in the typical manner, i.e. adjusting volume and fade curves, as well as cutting and splicing. The combination with easily accessible programming interface is quite unique, as most similar systems are either tailored towards symbolic manipulation instead of manipulation of electroacoustic material, or their graphical interface is more in the state of a visualisation and not a fully operable editor in its own right. This multitracker is complemented by the similarity algorithms which have been extracted into a separate library also published as an open source project1. It makes use of the feature extraction algorithms found in the SuperCollider system.

While the correlation of MFCC vectors alone proved useful in the live electronic piece, we are seeking better temporal synchronisation between target and match, as both are to be superimposed. For example, if we have a percussive sound, the algorithm should not only find

1http://github.com/Sciss/Kontur
2http://github.com/Sciss/Strugatski

Figure 3: Iteration in constructing the first half of the first part: (1) Initial sound gesture. (2.) Dissected into two smaller sub-regions. (3.) The previous phrase is extended by specifying a minimum and maximum dilation. (4.) The actual spacing is a result of the similarity search. It finds the matches which produce a best correlation both with the left region and the right region within the specified dilation bounds. (5.) Multiple sounds have been layered at time 0‘48”.
a sound similar in terms of the spectral evolution, but also in terms of its temporal envelope. Furthermore, in part 2 which is constructed to great length by the computer alone, we need a way to adjust the volume of the matched sounds to the particles. The program thus includes the loudness contour of the sounds, and for each search we may specify a weight which balances the cost between spectral and temporal.

The use of a sliding MFCC matrix cross- or auto-correlation has been described in previous literature (cf. [1, §3.5.1]), and we wish just to add that we have achieved the most convincing results using normalisation of the coefficients with respect to their occurrence in the corpus. We first created feature files from our database, comprising several gigabytes of sound recordings which had been performed using sliding windows each of which is normalised according to their mean value and standard deviation. Matches are ranked by highest normalised cross-correlation and an Euclidian distance is used to balance spectral and temporal features.

4.1. Part 1

As the Empties are found objects, the first part begins by finding a sound similar in terms of the spectral evolution, but also in terms of its temporal envelope. The algorithm is further refined by gradually introducing dilation they are completely unconstrained in their middle and maximum duration to insert between the left and right part of the sub-region which will be split as part of the iteration, as well as constraints for the database such as limiting the amount of gain that can be applied to the found material. The matches are presented ranked by correlation, and the composer can listen to the effect each choice of sound would have after insertion in the timeline. For this part of the process it was crucial to not stick to the "best" matches as seen by the algorithm, but often the musically surprising ideas were found more down in the list.

This strategy combines coherence and change in an interesting way—while we always have good matches in the beginning and ending of the found sounds, due to the computer which is not characterised by the dominance of sounds based on disjunctive spectra. The results of this strategy are equally interesting as the imitative approach, yet very distinct in their airy transparency. Another crucial aspect for the success of this process was the algorithm which now also needed to be performed automatically. Segmentation is also considered to be one of the most decisive part of concatenative sound synthesis [13], however in the CSS case, the database is already pre-segmented, whereas in our case segmentation is a dynamic process and regards "grains" with often several seconds of duration. We re-use the idea of ecology here, i.e. segmentation is based on maximising dissimilarity between the left and right half of a sliding matrix. A similar approach is described by Foote [6].

In the final construction of part 2, the ECOLOGY material enters the time canvas later than the IMITATION material, creating sensitivity for their respective qualities. The finalisation was guided by constraining human intervention to an almost purely subtractive role, using two stages: First the density in each layer was reduced, and some annoying sounds not connected to the target layer were removed. Then larger chunks of each target sound were cut out of the mix between the eight layers, taking care only to emphasise motives which were already formulated in the generated layers, and to give the piece an overall envelope.

5. INTERFACE

This paper has discussed three individual pieces. Yet they are automatically connected. On the surface, they all make use of algorithms which search for sound material based on similarity measures. But what is more important, by giving focus to this particular linking element we wish to present it rather as a strategy that goes beyond being an application of signal processing. It is a strategy which facilitates a form of relationship between the composer and the computer which is not characterised by the dominance of one over the other. Certainly, the computer would be useless without the implementation of the algorithms. But these algorithms escape the programmers control and exhibit unforeseen properties which thus takes the role of surprising the composer who in turn needs to react to a situation which he or she could not have established without the intervention, without the de-coupling from which he or she meditated this strategy.

In this respect we feel qualified to take a central element of Rheinberger's theory, the dichotomy of technical object and epistemic thing, and apply it to this situation. This situation can be described as the result of a specific way of natural sciences in the last two hundred years, in particular the establishment of experimental systems in bio-

![Figure 4: Cross-correlations included in the matching cost function. + means larger correlations are better (similarity), – means smaller correlations are better (dissimilarity). Channel 1, 2, 3, and 4 are the same as the correlation between channel 2 and 1. 4.2. Part 2

For the second part of the piece, strategies were chosen which we hoped could yield a more organic and acoustically uniform fading of the materials. Re-writing should be used again, but on a different scale and in the form of what could be called "Hidden Strugatsky Chain"—the target sound was already formulated in the generated layers, and to give the piece an overall envelope.
In the course of their creation, the tools we developed appear remarkably adapted to micro scale as well. The substrate presented here goes mainly back to a lecture Rheinberger held in 2008 [10]. The tools we used in the creation of the pieces, all tools that we developed in the course of this creation, somehow oscillate between technical objects and epistemic things, the latter being characterised by an inherent blurriness. Rheinberger calls it the «whole commitment» in a research project, it embodies that which cannot yet be denoted. Technical objects are complementary in that they require rigidity and specificity—a software, an algorithm must be defined, it must be correctly stated in the object language and well formed. This rigidity is responsible for keeping the vagueness of the epistemic things “hypocritical”. In other words, the algorithms provide guidance in the epistemic exploration and artistic research, they act as facilitators, they give coherence so that a previous experiment can be connected to the successive one. The system is productive when in motion, meaning that a formerly experimental algorithm—which perhaps yielded unexpec-ted results—can be used in a new experiment or piece. Its vagueness thus condenses into a new technically controlled object. On the other hand, the focus might now be turned to one particular aspect of the algorithm which had previously been unimportant, promoting it to epistemic rank. The motion in between the two is the “differential reproduction” which keeps the (re-)search going, and we have tried to show how the perspective has shifted in the creation of the three pieces without implying a cause and effect, but rather exemplifying how each piece gained its contours through the act of differentiation from the others.

The vagueness of the epistemic thing, it being un-ambiguous, makes it difficult to present this artistic research in front of the slide of positivism and empiricism. However, Rheinberger suggests two ways in which this problem, which he calls the «problem of reference», can be solved, and we applied both of them here: On the one hand, we can try to transform an epistemic object, to which we could not refer to with a concept, into a technical object— if transitorily—so the reference between the concept and the object gains an accepted validity. We can talk about the concrete algorithm, describe the effects we observed, how we handled it. The second solution is to make the alleged reference “change its position”. If it is used as a new experimental hypothesis, it changes from being the outcome which we cannot foresee to become the point of departure of a new successive search for an epistemic object. We suspect that this second approach is what keeps the artist intrinsically interested in the further pursuing of his or her research.

This argumentation brings us back to the idea uttered in section 3: The fragmentation between conceptualisation and perception is indeed the real thing since it is more effective than these two end points of an alleged communication chain: «The whole weight of this argument lies in the assumption that the primary act of symbolising in the realm of scientific research is itself a material process and not a linguistic process, at least not at its point of depar-true. The epistemic semiosis is one which moves between the material traces which are produced in the experiment and which in one way or another are put in relationships to invisible entities, and hence it is precisely not one between names and things.»

We hope that the works presented here, both with re-spect to their own construction as well as in the discussion about them, contribute to an understanding of artistic research and artistic practice which build on such an epistemic semiosis delineated by material traces. The fact that one of the biologists referred the most to by Rheinberger, François Jacob, and his “jeu des possibles” have been picked up by the recent discourse on artistic research (cf. [9]) is an indicator that the theory of differential re-production may indeed prove as fruitful in the arts as it did in the historiography of natural sciences.

6. REFERENCES

HETEROGENEOUSLY-COUPLED FEEDBACK SYSTEMS. THE ❯ (BAR DOT) PROJECT

Dario Sanfilippo  Andrea Valle
Scuola di Musica Elettronica  CIRMA - StudiUm
Conservatorio di Musica San Pietro a Majella di Napoli  Università degli Studi di Torino

ABSTRACT

Feedback-based systems for audio synthesis and processing have been in use since the ‘60s, resulting both from the theoretical reflection on Cybernetics and System Theory and from the practical experimentation on analog circuits. Taking into account the variety of feedback configurations and typologies, the ❯ (bar dot) project aims at defining and implementing the notion of heterogeneously-coupled feedback system. In order to define heterogeneity, we first briefly introduce historically the use of feedback systems in the audio/music domain and sketch a typology based on 5 categories depending on opposite binary features.

In order to cope with the variety of audio/music feedback systems already implemented, it is possible to sketch a classification based on five oppositions (see [31] for a detailed discussion):

1. FEEDBACK PROPERTIES AND TYPOLOGIES

The notion of feedback is widely used both as a technical feature in the design of audio and music application and as an aesthetic key in their description (see e.g. [14] [23] [7] [18], as well as the musical works based on feedback that we will discuss later). Many feedback-based systems for audio and music production/composition exist and many sound artists/composers consider feedback as a crucial notion at the basis of their work. Rather than being a monolithic category, feedback is a complex notion that presents many different features and aspects that are grouped together because of a family resemblance [26]. A minimal definition of feedback takes into account the configuration of a system, provided with input/output, in which some kinds of transformation are carried out, where the output causes a change in the input, and vice versa. Thus, the response of the system to stimuli is that of compensation, and it will tend to be in equilibrium around a desired target. In a positive feedback, the input-output relation is inverse: an increase in the output causes a decrease in the input, and vice versa. Feedback-based systems are typically unstable and cause exponential variations. Many feedback-based systems in the music domain are indeed of the positive kind, as structural instability is the precondition to the emergence of unpredictable, self-organized patterns [3] [4] [10] [12] [19] [22] [20] [24].

In order to cope with the variety of audio/music feedback systems already implemented, it is possible to sketch a classification based on five oppositions (see [31] for a detailed discussion):

- analog/digital: a first distinction can be traced between systems implemented with analog or digital devices, although some systems can also make use of both the types.
- audio/control (sub-audio): feedback can take place in the audio domain, as it happens in Larsen tones phenomena, or in the control domain, in the case e.g. when information is extracted from sound, and is used to drive processes of sound transformation;
- closed/open: in the first type, systems do not exchange energy/information with the environment, while in the second one there is an external feedback loop that couples the system with its surrounding environment;
- internal/external trigger: in positive feedback systems, the initial conditions are particularly relevant, as some energy is required in order to trigger the amplifying feedback loop. Triggering information can result from the internal activity of the system itself or be provided by some external agent;
- adaptive/non-adaptive: a system can be coupled with its environment but also capable of extracting information from it in order to adapt itself. Other systems, although coupled with the environment, do not exhibit adaptation capabilities.

2. IMPROVISING WITH FEEDBACK SYSTEMS

The previous typology is intended to provide an analytical, even if minimal, framework that allows to include many different works, coming from various traditions and practices, and to specify their mutual relations. Still, a sixth relevant aspect can be taken into account: the presence of a performer that often interacts live with the systems [31]. In particular, improvisation plays a key role in many feedback-based projects as feedback system are, and typically are specifically designed to be, intrinsically dynamical. Being structurally sensitive to minimal variations in their input, these systems tend to prompt an opposite performing situation. In the first case, the performer is absent, and the system is entirely machine based. In the second case, the performer is present: as she is forced to dynamically interact with the dynamical machine system, in the design of the performance an improvisational mood is often preferred to a fixed set of instructions. Also, being improvisation a process where actions are causally related to listening, an aural feedback loop is established between the machine and the performer, the latter becoming an integral part of the overall system. From a technical point of view, an aspect has to be considered, widely studied and explored [17] [30] [5]: feedback systems, and more generally electro-acoustic and computational devices, can operate without external control or actions by performers. Thus, while improvising the performer is in front of an entity which can be autonomous, where the human-machine relation is not necessarily based on subordination, but rather on a non-hierarchical exchange between the performer and the system.

These features have already emerged in some of the very first computational interactive systems from the beginning of the ‘70s. Joel Chadabe’s CIMS and Salvatore Martrino’s SalMar Construction. As noted by Chadabe, in those cases interaction indicated that “performer and instrument were mutually influential […] distinctions fade between instrument and music, composer and performer. The instrument is the music” [8] (291). Chadabe’s conclusion applies indeed more generally to feedback systems.

3. THE BAR DOT PROJECT

The ❯ (bar dot) project aims at defining a specific, feedback-based setting used in live performances. We first describe each component autonomously, and then they generate sound and in the way they extract information can result from the internal activity of the system itself or be provided by some external agent; are grouped together because of a family resemblance [26].

The name is a pun on the association of two elements -“bar” and “dot”- typical of digital, glitch imagery, that sounds like French “barot”, meaning “bouncy”, a mostly— but lack of always— infertile crossbreed. The system consists at its basis of two autonomous subsystems (components, hence the bar and the dot that are coupled, so that they exchange information both in input and output). The system consists at its basis of two autonomous subsystems (components, hence the bar and the dot that are coupled, so that they exchange information both in input and output). The components are purposely very different both in the way they generate sound and in the way they extract information from the environment.

Instead of designing a component autonomously, and then discuss a specific, feedback-based setting used in live performances.

Figure 1. Objectarium: technical framework.

up to now of three different setup. The “Rumentarium” is an ensemble of 24 electro-mechanical devices [33] (see Figure 2). Following an ecological perspective, each piece of the ensemble is built by associating a small DC motor, typically scavenged from discarded electronic devices (CD/DVD players, mobile phones) to a resonator. Resonators are assembled from a mass of reused materials: e.g. pipe tobacco boxes, glass bowls, broken.
cymbals, kitchen pans. Instruments can include Lego parts, in particular when support structures are needed. Motors can be extended by adding parts of various materials (plastic, wood, metal), thus implementing different modes of excitation. Not by chance, the whole ensemble is named “Rumentarium”, as “rumenta” indicates rubbish in North Italy. Four Arduino microcontrollers are used as DAC elements. Such a configuration allows for many different control strategies, from complete automation to instrumental-like usage by means of gestural controllers. Since its creation in 2009 the Rumentarium project has been further extended and generalized and the unstable nature of the instruments has been emphasized in various performances, as instruments have been assembled on a performance basis.

A second setup is inspired by David Tudor’s work on transducers, and in particular by the Rainforest IV project, where transducers where applied to physical objects and used to drive tones at their resonance frequencies (in Tudor’s approach see [9]). In the setup, loudspeakers are used to excite metal boxes filled with various materials; paper and aluminum sheets; goblet drums acting as sort of Helmholtz resonators. A third setup involves the use of relays as controls to switch on/off electro-mechanical devices. An example is the “Machina logettica” installation (presented at the Milano Design Week 2012) that includes hacked radio clocks. Each radio clock’s loudspeaker is connected to a relay. In this way, the speaker can be turned on/off remotely by means of electric signals. With respect to sound, the final result is a sort of real-time, parametrizable, analog granulation of radio contents. Even in this case, in which sounds are generated by loudspeakers, the sonic result shows a typical acoustic mark, as the fast on/off switching of radio clock loudspeakers scattered over a surface emphasizes their presence as physical sources.

3.2. GenES (Generator of Emergent Sound)

GenES is a real-time cybernetic music system implemented on the AudioMulch software by means of digital audio feedback networks. Figure 3 shows a screenshot of the software: on the left, the patcher window displays the interconnected modules, while on the right side various GUIs allows to access the parameters of the modules. A fixed delay of 256 samples is set for feedback loops. The system can be thought as a summation of cascade and parallel recursive processes (comb filters with feedback) with transformations happening inside the loops. More generally, each of the loops can be described by an equation of the type

\[ y(n) = TAz(n) + By(n - N) \]  

(1)

where \( T \) is the digital transformation, \( A \) is the gain of the input signal (if present), \( B \) is the feedback coefficient, and \( N \) is the delay in samples. According to this, the system shows a periodicity of 0.0058 seconds at 44.1kHz, 0.0026 seconds at 96kHz, 0.0013 seconds at 192kHz, respectively 172.4Hz, 384 Hz, 769.2Hz fundamental frequencies and their even harmonics, although the transformations in the feedback loops will result in complex, often non-harmonic spectra. Also, a copy of the signal with inverted phase is calculated in each feedback connection, so that it is possible to gradually crossover from the current to the inverted signal. This results in a shift from positive to negative values of the feedback coefficient B, thus in a comb filter with odd harmonics of the fundamental frequency. Processes in the system include: ring modulation, frequency shifting, granulation, flanging, EQ, waveshaping, bit decimation and reverberation (see Figure 5).

Furthermore, all temporal parameters in the DSP processes are within the 0-30 milliseconds range. This means that the system is in theory unable to produce individual sound events through time, as sounds within that range are perceptually fused [25]. This design has been chosen so that long-term audio events emerge thanks to operations in the micro-structure of sound, and not because of high-level scheduling. Also, no random factors are present in the processes, with the explicit aim of achieving unpredictable behaviors out of a non-stochastic design. As the feedback coefficient can be greater than 1, in order to prevent blow up an attenuation is applied for levels exceeding 0dBFS. As shown in Figure 4, it is also possible to use external analog feedback through microphones and loudspeakers. In this case GenES can generally be defined as a summation of Larsen feedbacks within which a summation of processes described in (1) take place. There are four intercommunicating sub-networks in GenES where two signals from two nodes of each sub-network are taken, and two input signals are provided for each subnetwork, with a total of 8 input/output signals. Figure 5 shows the signal flow of one sub-network in the system, where boxes represent DSP transformations, and arrows the connections from outputs to inputs between the nodes. It is possible to use the system in stereo, quadraphonic or on a more macro- scale, and generally, when two or more sub-networks are operating, the sound of each of them and their interactions will be heard. To conclude, according to the previously proposed classification, GenES is an analog/digital hybrid system (although only microphones and loudspeakers are within the loop, these elements can be crucial in feedback configurations) operating in the audio domain; the system can be made sensitive to perturbations from the environment, so it can shift from closed to open configurations; that triggering can be both external or internal, as in some situations the system operates with the digitally closed inputs; that, although sensitive to the environment, it is non-adaptive. The system is autonomous and "self-sustaining", and noise (numerical garbage within the software or environmental noise if using Larsen tones) plays an important role, as it is the only energy source feeding the networks [16]. Global properties and behaviors come from the low-level interaction between the interconnected DSP processes but are not related to the properties of the single elements. This synergy makes the system an organic sonic whole whose result can be considered as non-conventional sound/form [13] synthesis, where modifications in one of the sonic features (frequency, for example) potentially affects all the others.
(amplitude, spectrum, sustain, etc.). When using microphones and loudspeakers, a coupling with the environment is established, resulting in a system whose behaviors will be mediated by the characteristics of the environment itself (resonances) and perturbed by the sonic events in it (audience, etc.). GenES is a sound generator which so far has been implemented for improvised Human-Computer Interaction (HCI) performances (the LIES project), autonomous interactive sound installations (the SD/OS series) and other collective projects. In HCI sets, the role of the performer consists in dynamically altering the topology of the feedback networks by de/activating connections, and to change the relations between the components by modifying their parameters, thus the transformations they perform. Performance is an exploration of the different identities the system can take by modifying its internal variables, and of the perturbations it is possible to introduce into these identities by feeding the networks with external sounds. In sound installations, the system will be free to “express its own aesthetics”, while the sound it generates becomes a perturbation for its own behavior.

Georges Perec’s book *La boutiquse obscure*, in which the French author collected the transcriptions of 124 dreams [28]. The emergence in the dreaming process of chunks of articulated meaning from the mass of imaginary debris depends on the ranking of letter frequencies in the selected groups, where $n$ is the number of available outs. Grouping is the number of available outs. Grouping

If it is possible to say that in interactive systems “the music is the instrument”, a fortiori in a heterogeneously-coupled component of the Objectarium subsystem while it is running, e.g. modifying mapping ranges. The diagram of Figure 8 shows instead the GenES side of the setup. GenES outputs signals which are sent to Objectarium (black box) and to loudspeakers. The output sound is mediated by the environment and is captured from the microphones, along with possible perturbations coming from the audience or already present in the space. The audience, too, will modify the acoustic characteristics of the room, altering its resonances and the resulting external feedback. The performer, through a MIDI surface mapped to the GenES parameters, can change the system’s variables and explore the range of its behaviors. Three main nested feedback loops are established: GenES-Objectarium; GenES-Environment; GenES-Environment-Performer. Figure 11 shows an Objectarium setup for a Dark Store session: on the foreground, loudspeakers are visible on the floor covered by metal boxes and gobolet drums; a solenoid-armed glockenspiel is on the right; while in the background a tree of bells and metallic percussions is operated through DC motors. Figure 11 shows a second Objectarium setup.

Store audiovisual setup is depicted in Figure 6. The resulting sound emerges from mixing microphones capturing the Objectarium with the audio out of GenES, that is also fed back into Objectarium. Indeed, the direct acoustic output of the Objectarium is added to the sound from the surrounding environment, and contributes to the overall sonic result. As we will see while discussing in detail section 1, the Dark Store aims at investigating a complex configuration including opposite features. Hence the notion of “heterogeneously-coupled feedback system”. First of all, analog and digital sound sources are present, from purely digital synthesis techniques in GenES to mechanical behaviors in the Objectarium. Second, feedback includes both audio and control information, the latter extracted from the audio stream. As a third element, the Dark Store setting results in a coupled system where each component acts like an external environment for the other one, in turn each component being open to its environment. As discussed, GenES can operate both in a closed and open fashion while still acting as the audio environment for the Objectarium. Again, GenES can be triggered both from external sources and from internal digital noise, while Objectarium can only react to audio information in the surrounding acoustic scene.

Figure 7. Dark Store: Objectarium side.

3.3. The Dark Store setting

The Dark Store is a performance setting developed for the | project. The name of the setting originates from the concept of raw auditory event sensitiveness: a new event triggers a change in the state of the sequencer. During the sequence scan, each character (only if alphabetic: i.e. a letter) triggers a notification message to two other dependent modules, DarkTypo and DarkAdapter, forwarding them the analysis parameters with the triggering character added. DarkTypo is a typographical, real-time engine that maps analysis parameters to typographical dimensions (font, gray level, dimension, foreground and background colors). DarkTypo results in a minimalistic visual output (Figure 9): the sentences used for sequencing are composed in real time onto the screen, flickering between black and white. It is explicitly intended not as an autonomous video as in A/V performances: rather, as visual parameters are entirely controlled by the analysis parameters, it provides a visual representation of the information extracted from GenES signal. DarkAdapter is the software module that acts as a generic interface to sound bodies. It can be customized in order to meet different DAC architectures and sound body structures. As an example, in case of DC motors there is only a sound body parameter that can be controlled (voltage) and consequently a strategy must be defined in order to map the three analysis parameter to the single voltage value. A specific mapping function of DarkAdapter is ranking: letters are collected into n groups, where n is the number of available outs. Grouping depends on the ranking of letter frequencies in the selected language: as letter frequencies are distributed over a wide range (hence the famous ETAOIN SHRDLU pattern for English), a heuristic organizes the letter groups so that-as a whole-they have the same frequency. To sum up,
to his now 40 year old SalMar Construction, are enlightening feature of feedback system, their tendency to automatically interact, while performing, we concentrate on the relation of heterogeneously-coupled feedback system. On the one side, we investigate the structure of the system, in relation both to each component and to their mutual interrelation. Following an inclusive approach, our aim is to maximize differences: hence the notion of heterogeneously-coupled feedback system. On the other side, while performing, we concentrate on the relevant feature of feedback system, their tendency to autoorganization and emergence. Martirano’s words, related to his now 40 year old SalMar Construction, are enlightening:

“It was too complex to analyze. But it was possible to predict what sound would result, and this caused me to lightly touch or slam a switch as if this had an effect. Control was an illusion. But I was in the loop. I was trading swaps with the logic. I enabled paths. Or better, I steered. It was like driving a bus.”[8](291)

In our case, a hinn, maybe.

4. CONCLUSIONS

The project aims at exploring feedback and improvisation, starting from the assumption that these two elements are strictly interconnected, and can be seen as two sides of a unique configuration. In fact, feedback concerns the structure of the system and improvisation the way the system runs. Thus, focuses on two intervention points. On the one side, we investigate the structure of the system, in relation both to each component and to their mutual interrelation. Following an inclusive approach, our aim is to maximize differences: hence the notion of heterogeneously-coupled feedback system. On the other side, while performing, we concentrate on the relevant feature of feedback system, their tendency to autoorganization and emergence. Martirano’s words, related to his now 40 year old SalMar Construction, are enlightening:

It was too complex to analyze. But it was possible to predict what sound would result, and this caused me to lightly touch or slam a switch as if this had an effect. Control was an illusion. But I was in the loop. I was trading swaps with the logic. I enabled paths. Or better, I steered. It was like driving a bus.”[8](291)

In our case, a hinn, maybe.

5. REFERENCES


SNAPSHOTS: NEW POSSIBILITIES FOR SOCIAL DIGITAL MUSIC-MAKING ARISING FROM THE STORAGE OF HISTORY

Samuel Aaron
University of Cambridge
Computer Laboratory
Cambridge, England
sja55@cam.ac.uk

Jenny Judge
University of Cambridge
Centre for Music & Science
Cambridge, England
judge.jennifer@gmail.com

ABSTRACT

In this paper, we propose that digital music systems need to be designed with a view to collaborative, social, ‘non-cochlear’ musical engagement. In order for digital music to become more truly social, performers and audiences alike need to be able to see to some extent inside the ‘black box’ of the systems being used. This is particularly important in the case of electronic improvisation, where a lack of understanding of the processes being manipulated and how they are reflected in the sonic result can lead to a sense of alienation for audiences and performers. We propose that the capacity for storage of history represents a powerful tool for the design of interactive musical systems. We introduce ‘Snapshots’, a fully generic OSC-compliant multi-threaded storage and retrieval system, which is capable of storing independent named streams of OSC messages, as well as a novel query language for embodied approaches to digital sound control. The capacity of digital musical systems to store history is of data manipulation: the melodic contour of a past event, for example, could be queried and used to modulate a high-pass filter, turn a specific subset or riff into a generic control signal.

Figure 1. Converting the contour of a previously-played riff into a generic control signal.

1. MUSIC AS SOCIAL BEHAVIOUR: TOWARDS ‘NON-COCHLEAR’ MUSIC

1.1. Snapshots

The capacity of digital musical systems to store history is a potentially powerful tool that could be deployed in an attempt to set the scene for meaningful musical interaction in digital contexts. We introduce ‘Snapshots’, an environment in which the history of a performance is stored as streams of OSC events. Queries result in ‘snapshots’ of immutable sequences of events, which can be manipulated and used as generic input to other systems. Our system allows for greater power in terms of event recall, but it also provides us with a way of adding coherent structure to improvisations by referencing and manipulating past events, thereby increasing the structural intelligibility of improvised digital performances, for both performers and audiences. Since Snapshots operates on streams of OSC events, it is compatible with any physical interface that is OSC-compatible, increasing the possibilities for embodied communication with the audience and with other players. Snapshots could also be used to create semantic visualizations of the data structures in use, which would help the audience (and, indeed, the collaborating performers) to see further into the ‘black box’ of what is happening on stage.

For example, consider the ability to record the sequence of interactions of a pianist during a given performance. If the timing, press, release, velocity and pitch of each individual interaction or event are recorded into a unique stream, then this event can be treated as standard programming data to be processed and manipulated. Using Snapshots, it is possible to specify a query which will return a specific subset or riff embedded in the larger performance. This subset then may be subsequently manipulated via any arbitrary function; the resulting data set may then be used for a variety of purposes: for instance, as a control signal for audio processing modules or seed data for visualisations. Figure 1 shows an example of this kind of data manipulation: the melodic contour of a past event could be queried and used to modulate a high-pass filter, thereby the details of which process could simultaneously be shown via semantic visualisations, in order to enable both the performers and the audience to achieve a more fine-grained understanding of the unfolding structure of the improvisation.

We propose that this is a dynamic new line of inquiry in the attempt to make digital musical performance more truly social, collaborative and meaningful for performers and audiences alike. Before describing in detail the architecture of Snapshots, it is important to discuss the motivations for the design of this system, which stem from concerns raised in ethnomusicology about the expressive and social nature of music-making. Any digital musical system must begin by interrogating what ‘music’ means to begin with; it is to this question that we now turn.

1.2. What do we mean by ‘music’?

Attempts to define music have met with considerable difficulty. The standard ‘dictionary definitions’ of music, which generally take it to be an art form in which sounds or tones are combined in order to express meaning with a view to beauty or form, work well for Western classical music, but run into difficulty when we consider other types of music. In particular, the intimate link between music and culture in most world musics, as studied by the discipline of ethnomusicology, calls into question the notion that music should be seen as an isolated art form at all – perhaps it is best understood in terms of how the cultures that practice it actually use it in social interaction.

1.3. Why should we pay attention to the music of other cultures?

Bruno Nettl [11] notes that ‘standard’ definitions of music lead us to expect a certain level of formal complexity, which may rule out tribal chanting, for instance, from consideration as a type of music. The notion that music is made ‘with a view to beauty’ may lead us to expect that all music must sound pleasant. This is of course problematic, since beauty, too, is notoriously difficult to define, and may well be fundamentally culture-dependent. Moreover, it is difficult to say whether or not other cultures actually have the concept of ‘music’ as we understand it. Until relatively recently, as Nettl points out, ethnomusicologists took it for granted that all cultures had the concept of music. However, the Hausa of Nigeria have no term for music in general. Some native American societies, though they do have concepts for specific instances of what we would call music, have no general concept to tie all instances of it together under one unified category. Perhaps, then, music is not a universal concept at all – perhaps no universal definition of it is possible.

1.4. Music as social life

Thomas Turino takes up this idea, arguing that music is best understood as a type of social life, rather than as an abstract cultural artefact [14]. The main thesis of his book is that practice is not a unitary art form, but rather that this term refers to fundamentally distinct types of activities that fulfill different needs and ways of being human.”

(Turino, p. 1) Turino wants to argue that societies participate in music, not necessarily or exclusively to satisfy some ‘high-level’ aesthetic impulse, but in order to unite with others in daily life. For most non-Western cultures, to divorce ‘music’ from its cultural and social contexts is to make a false move. For example, the massana, or ritual song, dance and music of the Mbendjele Yaka pygmies of Congo-Brazzaville, would be completely misunderstood if it were taken out of the context of the particular world in which the Mbendjele live: the ‘music’ of these people (if we can even call it that) is geared very specifically towards the realities of life in the forest [10], in particular the promotion of group cohesion and the invoking of the forest spirits. In a sense, the practices of Western art music as unusual when one considers the way in which music is practiced socially all over the world. Maybe Western art music is actually unrepresentative: perhaps to make generalisations about music based on consideration of Western art music alone is to make a false step.

For Turino, playing with others allows him to transcend his own experience and feel united with the others in the community: “I think that what happens during a good performance is that the multiple differences among us are forgotten and we are fully focused on an activity that emphasized our sameness – of time sense, of musical sensibility, of musical habits and knowledge, of patterns of thought or action, of spirit, of common goals – as well as our direct interaction and the bounded and concentrated frame of musical performance that sameness is all that matters, and for those moments when the performer is focused and in a zone, that deep identification is felt as total.” (Turino, p. 18).

This unifying aspect of music-making has been discussed by other authors (for example [5]) as a possible reason for music’s being a target of evolutionary selection at the group level, the idea being that a group that practices music together will be more united and hence more successful. This feeling of total synchrony with fellow members of a musical group is quite elusive: in many cases, it is the motivation for musicians seeking new groups to perform with. In addition, Turino points to the possibility that some are more sensitive to this synchrony than others: one member in a band may feel that everyone is well-synchronised, while the other band members may all feel that he is in fact out of time. “Like the good human relationships they index,” says Turino, “good musical relationships are difficult to achieve and require continual work to sustain.” (Turino, p. 20) The pursuit of this feeling of synchrony with others is, for many musicians, a primary motivation to keep playing.

1.5. Different ‘kinds’ of musical performance: Pre-sentational versus participatory music

Turino’s distinction between presential and participatory live music highlights another sense in which traditional dictionary definitions of music leave something out. Most Western art music is, he argues, presentional: the musicians are doing, the audience merely listening. There
is a physical divide between the performers and the listeners. Participatory music, on the other hand, has no such clear divide between performers and listeners; all present are involved in simultaneously doing and listening. This type of music-making is less on music as 'art' as on the social dimension of the act of music-making itself. Non-western music is often participatory rather than pre- sentational. Of course, the distinction is blurred in some cases; for example, a song performed in a non-western musical tradition by individual musicians can play together as part of an ensemble. The motivation behind the development of such ensembles can be different, directed either to a perceived need for a more non-western way of doing music: one that is grounded in real-world interactions and embodied gestures, as opposed to being focused purely on the sonic output of a system. Such initiatives highlight a growing interest in making digital music more social and communicative, both for performers and listeners; to a large extent, it is certainly a step in the right direction.

2.3. Laptop orchestras and black boxes: alienated audiences

However, even in the case of laptop orchestras, the performance of live electronic music can often be an alienating experience for the audience: the composers controlling the sound. This tacit knowledge cannot be got from listening to digital music itself. Of course, symphony orchestras manipulate "tech- nological" parameters of each instrument, and the way in which the actions of the musicians is linked to the pro- duction of the sound. This tacit knowledge cannot be got from listening to the music itself; it arises from en- countered musical instruments, and their use as a social engagement for the audience. The nature of electronic music, in its open-endedness, allows performers and listeners to interact more closely, and vice versa.

2.4. What about electronic music improvisation? Is a truly social creativity possible?

Moving away from purely presentational music, we might ask what the state of affairs is when it comes to participatory music-making — for instance, by considering the nature of the experience of the performers themselves. This becomes particularly relevant when considering the possibility of improvising with digital music, as well as collaboratory composition, which we consider as occupying 'opposite ends' of the same process, following Nettl. In the case of digital music ensembles, much of the same char- acteristics as those outlined above with regard to presen- tational music: that is, because of the presence of 'black boxes' in the ensemble, the audience must be rethought, a per- sonal understanding the others are doing. In traditional ensembles, the technologies are stable: the symphony orchestra, for instance, has maintained the same in- struments for the last hundred years. Even though there have been some changes, the pace of the evolution of the symphony orchestra has been extremely slow. For that reason, it is possible for each player to have a good under- standing of the range of instruments involved in each instrument. With digital ensembles, however, any combination of software and hardware could be in use at any given time. Even if each player happens to have a in-depth knowledge of the combination of software and hardware in use by each of the other players, there is much less of a chance that she will have an in-depth, fine-grained understanding of the parameters of each of those combinations.

2.5. Lack of constraints in electronic improvisation: a barrier to creativity

The nature of electronic music, in its open-endedness, also poses challenges for performers. Jazz improvisers can improve fluently because they understand the pre-given harmonic structures of the piece, the piece is an attempt to combat this individual- ism by creating an environment where electronic musicians can play together as part of an ensemble. The motivation behind the development of such ensembles can be different, directed either to a perceived need for a more non-western way of doing music: one that is grounded in real-world interactions and embodied gestures, as opposed to being focused purely on the sonic output of a system. Such initiatives highlight a growing interest in making digital music more social and communicative, both for performers and listeners; to a large extent, it is certainly a step in the right direction.
constitutes an important step towards creating new, em-  
dabled digital interfaces, which will go towards solving  
at least part of the puzzle faced by electronic instrumen-  
talists. However, the problem of fine-grained understand-  
ing of the parameters of the music itself remains, even if  
we achieve stability of digital interfaces, as well as the  
problem of the excessive freedom of electronic improvi-  
sation. We propose that the capacity of digital ensembles  
for the storage of history suggests a potential solution to  
this problem, paving the way for a more truly social digi-  
tal music.

3. SNAPSHOTS: QUERYABLE HISTORY

3.1. Background: understanding sounds

Insofar as musical listening involves sounds, it is reason-  
able to assume that musical sounds ought to be ‘under-  
stood’ in some way by the listener. An understanding of  
the causal chain implicated in sound production is gener-  
ally present in audience experiences of acoustic musical  
performance, due to familiarity with the instruments used  
and also an embodied understanding of how the sound  
is produced. However, as discussed above, the case of dig-  
ital music listening, in particular the case of improvised  
music more transparent, Snapshots addresses these is-  

sues directly.

3.2. Implementation

Snapshots is implemented entirely within Clojure[7] - an  
expressive JVM-hosted Lisp with an emphasis on concur-  
tency and functional programming. There were two main  
motivations for this choice. Firstly, Clojure’s persistent  
rency and functional programming. Secondly, Snapshots has been developed as part of the suite of libraries supporting Overtone [2]. However, despite being part of the Overtone suite of libraries, Snapshots has been designed to be used entirely independently of Overtone itself, only requiring a JVM implementation, which is freely available on all ma- 

jor platforms. Snapshots is freely available from Github [1] under an MIT X11 license and the architecture for this system is presented in Figure 2.

The snapshots server exposes itself as a generic OSC server sitting on a specific host and listening on a specified port. For example, the following code creates a server on the local machine listening on port 9851:

(def snapshots-server "localhost" 9851)

3.3. Event Storage

In order to store an event, it must be encodable as a stan-  

dard Open Sound Control (OSC) message, which is es-  
tenantly a list of values of a small set of defined types: in-  
tegers, floats, strings and binary blobs (specifics for these types can be found in the OSC specification [15]). A typ-  
ical OSC message contains a path followed by 0 or more arguments. For example, the following encoding of a MIDI-like piano event, which represents note 60 be-  

ginning triggered at volume 10 (within the standard range of 0-127):

/:note/on 60 10

In order to store this message in Snapshots, it is necessary to firstly create an OSC client with the server’s details:

(def c (osc-client "localhost" 9851))

Once the client has been created, the piano event can be sent as arguments to the /store command:

(osc-snd c ""/store" "/my-stream" "/note/on 60 10")

Notice that the /store command also requires an extra argument (i.e. /my-stream) representing the name of the stream in which the event is to be stored. This allows the server to store any number of independent streams.

For each event stored, Snapshots stores an OSC path (i.e. /note/on) and a list of arguments (i.e. 60 (100)). In addition, a timestamp is stored, which records the time Snapshots received the event as well as the sender’s host- 

name and port. Each of these events is represented with an immutable associative map such as the following:

{:path "/piano/on" :args [5 0] 
:ts 132735576447 
:src-port 64546 
:src-host "localhost"}

The store commands may be sent from an arbitrary num- 

ber of external clients and may request the storage of his- 
tory in an arbitrary number of named streams. Snapshots guarantees that these storage commands will never con- 

flict with each other. This guarantee is ultimately met at the CPU level with the instruction Compare and Set (CAS) which ensures the atomic succession of stored val- 

ues.

Once events have been stored, we may then retrieve them, filter them and even generate new named frozen event streams to be further referenced, filtered and re- 


trieved. Retrieved event streams are then available as ba- 

sic native data structures to be manipulated, played back  
or stored to disk from any programming language with an OSC library.

3.4. Event Retrieval

Snapshots supports the retrieval of all the messages stored in any of the named stores as a series of individual OSC messages. Retrieval is achieved using the /fetch com- 

mand, which expects four arguments: a unique identifier for this particular query, the name of the stream to fetch and the hostname and port of an OSC server, to which the stored events are to be sent. For example:

(osc-snd c ""/fetch" 78 "/my-stream" "/note/on 9873")

On receiving a /fetch message, an independent thread on the server first creates an immutable snapshot of the stream’s current state. Due to the combination of the stream’s use of CAS for updates and the persistent nature of Clo- 

jure’s datastructures, this snapshot is guaranteed to repre-

sent a valid stream state (i.e. not in a liminal state such as the partial completion of a storage update). Also, the algorithmic complexity of creating this snapshot is essen- 
tially free, O(1), and does not interfere with the ability for other threads to be concurrently updating the original stream. Once the snapshot has been created, the thread then starts sending each element in the snapshot to the specified location as separate OSC messages. Clearly, the cost of this streaming is O(n) in the size of the stream.

However, as it is operating on a immutable snapshot and not the original stream, new events stored in the original stream between the snapshot creation and the completion of OSC data transmission will not be reflected in the snap- 

shot and therefore not sent as an OSC event as the reply to a /fetch message. Finally, as the snapshot creation and subsequent transmission happen in an isolated thread that holds no locks, it does not block the server’s ability to continue to concurrently handle additional requests.

3.5. SnaQL - Snapshot Query Language

It is not always desirable to retrieve the entire contents of a specific stream. It is therefore important to be able to de- 
scribe interesting subsets of a given stream of events and formulate a query that describes this subset. An example of this through the /query message and supports subset descriptions through its query language SnaQL (Snapshot Query Language). This subset could be as simple as re- 

presenting the event values that were sent to the server, or it might contain more specific filtering. The SnaQL query engine implements this filtering process using a cascading filter-chain gener- 

ated from the SnaQL statements. Note that the cardinality of the original snapshot n is less than or equal to the card- 

inality of the result snapshot m, i.e. m <= n.

SnaQL statements consist of one or more filter express- 

ions, see Table 1. Each filter expression is constructed by using one of the supplied filter types such as drop, take and remove and passing the appropriate param- 

eter. A parameter is either a number or a predicate. A predicate can be seen as a function which returns either true or false and consists of predicate types such as and, 
or and not=, see Table 2. Each predicate takes one or more clauses which may be other predicates or values.

A value is an immutable number or string. Values can either be explicitly entered into the predicate, or may be dynamically retrieved from the current event being fil-

tered. The syntax for referencing values within an event is (event :event-value arg) or (event :arg 1) 40.

In order to avoid stream name clashes, we recommend you use OSC path-like identifiers such as /sam/piano and /jenny/piano.

In order to avoid stream name clashes, we recommend you use OSC path-like identifiers such as /sam/piano and /jenny/piano.

 Galeazzi, datastructures, this snapshot is guaranteed to repre-

sent a valid stream state (i.e. not in a liminal state such as the partial completion of a storage update). Also, the algorithmic complexity of creating this snapshot is essen- 
tially free, O(1), and does not interfere with the ability for other threads to be concurrently updating the original stream. Once the snapshot has been created, the thread then starts sending each element in the snapshot to the specified location as separate OSC messages. Clearly, the cost of this streaming is O(n) in the size of the stream.

However, as it is operating on a immutable snapshot and not the original stream, new events stored in the original stream between the snapshot creation and the completion of OSC data transmission will not be reflected in the snap- 

shot and therefore not sent as an OSC event as the reply to a /fetch message. Finally, as the snapshot creation and subsequent transmission happen in an isolated thread that holds no locks, it does not block the server’s ability to continue to concurrently handle additional requests.

3.5. SnaQL - Snapshot Query Language

It is not always desirable to retrieve the entire contents of a specific stream. It is therefore important to be able to de- 
scribe interesting subsets of a given stream of events and formulate a query that describes this subset. An example of this through the /query message and supports subset descriptions through its query language SnaQL (Snapshot Query Language). This subset could be as simple as rep- 

resenting the event values that were sent to the server, or it might contain more specific filtering. The SnaQL query engine implements this filtering process using a cascading filter-chain gener- 

ated from the SnaQL statements. Note that the cardinality of the original snapshot n is less than or equal to the card- 

inality of the result snapshot m, i.e. m <= n.

SnaQL statements consist of one or more filter express- 

ions, see Table 1. Each filter expression is constructed by using one of the supplied filter types such as drop, take and remove and passing the appropriate param- 

eter. A parameter is either a number or a predicate. A predicate can be seen as a function which returns either true or false and consists of predicate types such as and, 
or and not=, see Table 2. Each predicate takes one or more clauses which may be other predicates or values.

A value is an immutable number or string. Values can either be explicitly entered into the predicate, or may be dynamically retrieved from the current event being fil-

tered. The syntax for referencing values within an event is (event :event-value arg) or (event :arg 1) 40.

In order to avoid stream name clashes, we recommend you use OSC path-like identifiers such as /sam/piano and /jenny/piano.
drop n
Returns the snapshot except for the first n events.
take n
Returns first n events from the snapshot.
drop-while p
Returns the snapshot except for the first n consecutive events for which predicate p returns true.
take-while p
Returns the first n consecutive events for which predicate p returns true.
filter p
Returns only the events for which predicate p returns true.
remove p
Returns only the events for which predicate p returns false.

<table>
<thead>
<tr>
<th>Table 1. Filter types.</th>
</tr>
</thead>
<tbody>
<tr>
<td>and &amp; cs</td>
</tr>
<tr>
<td>or &amp; cs</td>
</tr>
<tr>
<td>not= &amp; cs</td>
</tr>
<tr>
<td>= &amp; cs</td>
</tr>
<tr>
<td>&lt; &amp; cs</td>
</tr>
<tr>
<td>&gt; &amp; cs</td>
</tr>
<tr>
<td>not c</td>
</tr>
</tbody>
</table>

The SnaQL engine parses these statements and generates a four-stage cascading-filter-chain with which to filter the contents of a specific event stream snapshot. The first filter is passed all of the events within an immutable snapshot of a stream. Applying the first filter to this stream results in a new immutable snapshot which removes any events that do not have the path /note/on. This subset snapshot is then passed to the second filter, which only keeps the events for which the first argument is greater than 80. This snapshot is fed, in turn, as input into the third filter, which only keeps the events for which the first argument is less than 80. The final filter then returns only the first 10 events. The snapshot generated as the output of the chain of filters represents the resulting subset of interesting events.

3.7. Query Snapshots

In order to further facilitate the sharing of history between users, it is possible to create named frozen immutable snapshots of a given filter-chain output. This is made possible with the /snapshot command. This command has two required arguments and zero or more SnaQL filter expressions. The required arguments are: unique ID, name of the event stream to query, hostname to reply to; and port to reply to. These are similar to the arguments for the /fetch command. In fact, if no SnaQL filter expressions are passed to /query, it has identical semantics to /query; it snapshots the event stream and then sends each event as a separate OSC message to the specified host and port. However, if filter expressions are specified, the snapshot is filtered appropriately before being sent as individual OSC messages; that is to say, a subset of the original stream is always returned. The above SnaQL example can be expressed as a snapshot

| path n | The path of the event. |
| target n | The target argument of the event. |
| ts | The timestamp of the event. |
| host | The port the event originated from. |
| src-host | The host the event originated from. |
| src-port | The event stream queries using the command

The query as follows:

```
(osc-send c "*/query*" 42 "/jenny/piano" "localhost" 9873 "/note/on")
"filter (> (event :arg 1) 40)"
"filter (< (event :arg 1) 80)"
"take 10"
```

4. CONCLUSIONS & FUTURE WORK

We have described 'Snapshots', a system that facilitates the querying of the history of a digital improvisation session and the subsequent manipulation of the query results. This, we argue, represents an important first step towards making digital improvisation more intelligible, creating as it does the possibility of the use of past material to influence or modulate current and future events. Being implemented within Clojure, Snapshots has built-in concurrency semantics, making it ideal for group scenarios where there are multiple inputs to the system. This, we argue, vital if digital music is to become more truly social and participatory.

We have described a basic system of querying and naming events. Future work could include building functionality for performing computations on these immutable events in order to generate interesting patterns in the improvisation. Another intriguing prospect is the generation of semantic visualizations from the operations of Snapshots, which might represent a further step in crossing the communication barrier, giving the audience visually rich clues as to the processes that are implicated in the sonic results of the improvising group. In other words, by laying the groundwork for interesting manipulation of data on the fly, we are setting the scene for performers and audiences to achieve a more fine-grained understanding of the processes involved in electronic music. We feel that this is an exciting prospect for a more collaborative, outward-looking and expressive digital music.

5. REFERENCES

DESIGNING INTERACTIVE AUDIENCE PARTICIPATION USING SMART PHONES IN A MUSICAL PERFORMANCE

Oliver Hodl, Fares Kayali, Geraldine Fitzpatrick

Vienna University of Technology
Institute for Design & Assessment of Technology, Vienna, Austria
{oliver, fares}@igw.tuwien.ac.at, geraldine.fitzpatrick@tuwien.ac.at

ABSTRACT

In this paper we describe the design and evaluation of an interactive system for audience participation in live performances using smart phones to control the stereo panorama of the lead guitar. The system was developed through feedback from both spectators and artists. The evaluation was conducted during a live concert and builds on interviews and video analysis. Findings include that musicians seem to be cautious about giving up control and that the audience at the same time wants a reasonable amount of control and clear feedback which in turn can be obtusive to other spectators. We outline that balancing constraints with affordances is the key to both the audience’s and musicians’ acceptance of such a system and that a playful participatory design process can lead to better results in this regard. It is also shown that using smart phones opens up a large possibility space but at the same time their use has to be subtle to not distract too much from the music.

1. INTRODUCTION

Interaction within the context of musical performances has been subject to a lot of research over the last decades mostly in combination with new technology which brought a variety of new ways of interactivity into musical performances. Just to mention a few, sensor-based systems allow the perception of bodily interaction (e.g. [5,16]), tangible devices enable intuitive and appropriate interaction with digital devices (e.g. [13,23]), and network-based systems make collaborative performances possible where participants are getting closer to each other even though in some cases they are spatially divided (e.g. [20,26,29]).

However many of these applications rely on bespoke technologies for the interactivity. In this paper, we are particularly interested in exploring if and how we can exploit the ready availability of everyday smart phones, rather than some other bespoke device, to enhance the experience of live music performances. Modern smart phones already combine a range of sensors and network technologies in one off-the-shelf device. The number of people having a smart phone is already high and still increasing and they have been used recently for studies in a music context even though, little has been done using smart phones for letting the audience collaboratively participate in a traditional performance setting.

The broad area this paper addresses is understanding what it means when the audience participates in a musical performance. We talk about interactive audience participation when a spectator can take part or at least make a contribution in a live concert through a technically driven system. Regarding the performance itself it is becoming difficult to make “a neat distinction between interactive/real time performances and participatory installations” [30]. Thus we have narrowed our focus and concentrated on a musical performance of a rock band with the traditional distribution of roles where the musician plays actively in front of a passive spectating audience. Within this context the design of an interactive system is highly dependent on both the musician’s and audience’s desires. Hence this study builds on the following research questions: (1) What do audience and musicians expect from a system for interactive audience participation? (2) Considering both, how can two be balanced out in terms of limitations and capabilities to facilitate meaningful interaction as well as aesthetically pleasant results? (3) Is smart phone technology an option for such a system?

For this purpose we conducted interviews, created a design, developed the technology and finally did an in-situ evaluation in a live concert. We decided to include the artist as well as audience members in the design process to let them “directly inspire and shape the technologies that are developed” [11]. In parallel we followed Kiefer et al. as they underlined the importance of pilot studies in the context of evaluating musical controllers [15]. Thus we did interviews with musicians and audience members first to get a real idea of their requirements. Based on these insights we developed a prototype system for interactive audience participation with smart phones involving artists and spectators in the design process. Finally we conducted an in-situ study at a live concert where the audience collaboratively influenced the sound. For the evaluation we conducted a video-based analysis and used questions referring to the audience’s experience and the system’s performance.

Turino provides a good distinction between participatory and presentational performance in the context of musical styles and different cultures [28]. To be more specific audience participation has been done in previous research in various ways [8,20]. In the study investigating musical play Maynes-Aminzade et al. investigated various techniques for participation with a huge audience [19]. Their paper concluded with “a set of design principles for interactive crowd activities” which finally inspired us to use their approach in the field of musical performances for our in-situ study. In other studies sensor-based [16], wireless [6] and mobile phone technologies [17,26,22] were used for interactive and collaborative musical performances. The “reactable” [13], for example, builds on a collaborative tangible interface which seeks to be “intuitive (zero manual, zero instructions), sonically challenging and interesting, learnable and malleable” [12], all relevant qualities of an instrument designed for adhoc participation. The musician Bjork said she chose to use it on stage because “it also allows the audience to experience and understand electronic music and its performance on a whole new level” [13]. If we are talking about audience participation, it also means making the playing of music available to non-musicians. This opens up the discussion about playing music in a more passive toy-game/like sense [14,25]. In his essay about the “composition-instrument” Herber [10] states that a system designed for this kind of musical play must maintain a delicate balance between “play” (freedom of expression) and “being played” (controlled and musically “safe” results).

The approaches to the field of audience participation are manifold. This paper takes inspiration from approaches towards music participation in media and sound art, sound tools and even games and uses them in the setting of a live concert. The potentially and problems of this design strategy are then assessed taking a classic HCI approach to evaluation. The common denominator of the referenced examples and our research is a playful approach towards designing music and interactivity. The new ground covered is the evaluation of applying playful interaction to a contemporary live performance setting.

In considering this related work we argue that there is still much to learn about using smart phones for interactivity in performances. To explore this, we go on now to the interview study with musicians and audience members followed by the description of the in-situ study which includes study design, technical implementation, our methods and finally evaluation. The results suggest that using smart phones for interactive audience participation during a musical performance is a suitable method for engaging and entertaining the audience, however it is difficult to design an intuitive and easy-to-use system keeping in mind the wishes and needs of both artist and audience.

2. INTERVIEW STUDY

To get an initial idea of the musicians’ and audience’s behavior and habits concerning live concerts we conducted semi structured interviews. Hence our interviewees were able to talk freely at some points telling us about their experiences and desires in more depth than just answering closed survey or more structured questions.

The interview guideline included 38 questions divided into five groups: (1) a general overview about preferred music and live concerts, (2) personal views and general information, (3) details that happen during a live concert including mobile phones and tablets and some examples of particular behavior when thinking of a recently visited live concert, and (5) personal attitude towards technical developments. At the end we showed them examples of already existing systems for interactive and collaborative live musical performances through sensor data [16], mobile phones [26] and the World-Wide-Web [29] and asked for their reactions and any further ideas or examples generated.

We interviewed eight participants between May and June 2011. Participants were recruited through social media, an online magazine for music and art, a university and two music labels. Fans who play concerts regularly, defined as about 20 to 30 concerts a year. The other four persons were spectators who attend live concerts regularly, defined as 5 to 15 concerts a year. Each interview took between 45 and 60 minutes and was audio recorded. All participants were between 20 and 35 years old and three were females.

For the analysis we transcribed. Note that some excerpts quoted below have been translated into English as six out of eight interviews were held in German. We used the open source software Weft QDA² for a thematic and comparative analysis to find out the important aspects as well as the pros and cons in relation to their experience with live concerts. According to the scope of research we focused on statements referring to the musicians’ motivation to play and the audience’s motivation to visit live concerts.

2.1. Results

According to the participant’s taste the different styles of music ranged from acoustic and jazz music to rock and electronic music as well as various hybrid forms. When talking about live concerts both musicians and spectators pointed out the special experience when music is played live and the importance of human elements in live music, most notably when computers are used. However, the spectators’ motivations to go to live concerts varied widely as shown by the following statements: “The show must be powerful and entertaining” (S1), “I want them to play the music like on the record” (S2), “The music should be real, not perfect” (S3) and “I most often ignore the show” (S4). Talking with musicians about their motivation to play concerts revealed a tendency to enjoy the “showmanship” (M1), the “to see ex- cited people” (M3) and even “inspiration of the on-stage situation and the audience” (M4). The analysis of behavior and habits during concerts revealed a strong tendency among all spectators for text messaging, calling, taking pictures and videos or using social media to share the live experience. All of them indicated some action with a mobile phone at least once during a
concert but there was no pattern about when this happened during the performance. The spectators’ opinions about technology used in live concerts in general were far more open-minded than the musicians’ opinions. Interactive mobile phones, and smart phones in particular, were widely used by spectators during a concert. Furthermore understanding and controlling the stereo panorama is very intuitive for the audience since acoustic stereo signals are ubiquitous (e.g. mobile music devices with earphones, computer speakers or TVs). Finally waving hands in the air and forth is a common gesture in the context of a contemporary live concert.

3.1. Technical implementation

The technical implementation of our prototype is shown in figure 1. We refer to the major parts “Audience” and “Musician” as the interaction layer including visual and acoustic elements and refer to “Signal processing” as the technical layer.

![Figure 1. Schematic illustration of the interactive system for audience participation.](image)

Spectators download and install an app (which is described later) that enables them to interact with the stereo panorama. The audience gets visual feedback through a white dot projected on the screen behind the drummer at the back of the stage. The visual and interactive manifestation of sound or its parameters is called a sound object [27, 4]. Sound objects for example have also been used in networked performances by Barabosa and Kaltenbrunner [1]. In figure 2 (left picture) one can see the visualization (white dot) of the left-right-position of one smart phone which is identified by the unique number 8001. In the right image the white dot represents the average left-right-position of all participating smart phones. Thus everybody knows at any time whether his or her device is active or whether all devices are active and therefore all participants control the stereo panorama cooperatively. The acoustic feedback comes directly through the PA speakers situated at the left and right side of the stage facing towards the audience.

![Figure 2. Two still pictures of the stage camera.](image)

To cover the majority of smart phone users we designed our system for Android-based phones as well as iPhones. We used the free apps “TouchOSC” (Android) and “Control” (iPhone). Both are able to send accelerometer data as Open Sound Control (OSC) messages over WiFi. The wireless connection is handled by a WiFi router which transmits the OSC messages over a specified UDP port to a Pure Data (pd) patch running on a laptop.

The pd patch is the core of our system and does most of the work including the processing of OSC messages and sending out MIDI messages to the guitar effect device where the stereo panorama is applied. In pd the accelerometer data is normalized due to the different values ranges of the two apps. Because of limited capabilities in pd for graphic design the idea was also pre-defined for twelve smart phones to connect and send accelerometer data. All data is summarized and divided through the musician at any time to get an average value which is then scaled to standardized MIDI values ranging from 0 to 127 and sent to the guitar effect device.

3.2. Study procedure

The venue for our in-situ study was a free public concert in the club B72 in Vienna, Austria, for standing only. We had a team of four people asking guests at the entrance whether they have an Android-based phone or an iPhone and were willing to participate. The people who agreed received a one-page sheet with a short explanation of the study how to download the smart phone app. Every participant received the free app, inserted the given IP address to connect to the WiFi router and finally a specific UDP port. For testing purposes everyone had to check the correct configuration with the white dot individually as shown in figure 2 (left picture). They were not given any specific instructions about where to stand during the performance. We could only take twelve participants (about an eighth of the audience) due to technical limitations. This is reasonable given that we are a probe system to explore the concept and learn about the characteristics and possibilities [11], not trying to generalize results.

We chose a song that was played twice at the end of the show. Everyone was told that the interaction would take place then and there would be an explanation beforehand. The first time the song was played without audience participation followed by a short explanation of the study and testing the system. Then the actual audience participation was done while the song was played for the second time. We did this to be able to compare the two versions as explained later during the evaluation section. The song lasts about five minutes and is divided in two more or less equal parts. Each participant had the chance to control the stereo panorama individually for 13 seconds by controlling the sound objects of the guitar. With the beginning of the second part the signals of all participants were summarized to control the stereo panorama cooperatively.

3.3. Evaluation methods

According to an approach by Reeves we did a “hybridised form of video analysis” [24] combining video-based analysis and questionnaires. For this purpose we used two cameras to record the audience from two angles and one for the stage. Still pictures of the stage are shown in figure 2. We had a total of 45 minutes of video footage taken over a period of 15 minutes. Immediately after the second time the song was played, twelve non-participating audience members were asked to fill out a short one-page questionnaire handed out by our team according to a similar approach done by Pedersen and Hornbæk [23]. We used our video recordings to analyze non-verbal social interaction among the audience interpreting body movement, gestures, expressions and gaze as done previously by Heath et al. [9] when studying social settings. Following their outlines we did a preliminary review for basic structuring, a substantive review to discover and analyze a total set and finally an evaluation to study specific parts in detail. We divided each of the three videos into parts to analyze them separately: (1) five minutes before the song was played originally, (2) five minutes of explanation and testing with the audience, and (3) five minutes while the song was played with audience participation.

Then each of the twelve participants was analyzed individually during both performances to find out important and constructive contributions and additionally we picked twelve non-participating audience members randomly to analyze their behaviour in the same way. While repeatedly watching certain occurrences in the video we focused on particular aspects of bodily interaction (e.g. synchronous moving of smart phones) and compared the three different camera angles (e.g. movement of the white dot compared with the view of the audience camera). Additionally twenty non-participating audience members filled out the questionnaire. Twelve of them participated with their smart phones and 19 did not. We handed out two different questionnaires: one for study participants and one for the other spectators. The short questionnaires were focused on their experience with smart phones, the procedure and the understanding of the study and their opinion about the audience participation. Finally we asked them which differences they could figure out between the two performances of the song. Combined with the video analysis this led to interesting and unexpected results.

3.4. Results

About half of the twelve participants were standing close to the stage and the others were distributed over the whole venue. By trend, participants concentrated on the performance or rather the screen with the white dot whereas non-participants tended to observe the
study participants regularly and seemed to be a little dis-
traced. This assumption was verified by statements in the
questionnaire: “I focused on the white dot most of the
time” and “I prefer the version without [audience partic-
ipation]”. I could concentrate on the music more.

Participants standing next to each other had short com-
ments and conversations while continuing to watch the
stage or scene. Among the participants, we could identify
different differences concerning smart phone interaction
in relation to speed, range and height when moving the
device. Most obviously, there was a great disparity regard-
ing stance and how the device was held. Audience mem-
bers at the back held their phones up in the air whereas
the girl and the boy in the front row moved their devices
at breast height. Some even tried to “push” the white dot
on the screen by shaking and moving the phone heavily.
Again this behavior can be sub-
stantiated by statements from the questionnaire: “I could
dreadfully move the white dot”. “It was easy to see my influ-
ence when I had exclusive control but I could not really
give it out collaboratively” and “I have tried various dif-
fierent ways to control it”.

Finally some participants tended to dance and syn-
chronized their movement (including the smart phone ges-
tures) to the rhythm of the music throughout the whole
song. Others stood still and seemed to concentrate on the
movement of the smart phone. In general there was no
tendency towards synchronization of movement among
all participants.

The statements about the audience’s experience and
their opinion about the smart phone based participation
were mixed. Some described it as an enjoyable experi-
ence: “People were involved and therefore much more ac-
tive”, “I felt honored to be part of the show”, “Funny”, “I
want to influence other effects that change the sound”.

While our study integrated the perspective of musicians
in the interview study and design process, we did not in-
clude them in this study. Further studies could also eval-
uate a series of concerts with audience participation from
the musicians’ perspective.

Finally our results have shown expressive behavior and
diverging reactions among certain participants in this
context. This indicates that a playful approach to inter-
face design can lead to a better understanding and more
intuitive control regarding the interaction with the device.
In HCI the importance of playful inter-
action [7] has been emphasized especially in connection
with the design of artifacts for social interaction. Social
interaction of course also is an important attribute of our
presented participative performance system. At the same
time the design of the system itself is a playful exploration
of technology [21]. The importance of ludic design pro-
cesses to interactive and participative art is described in
[21]. In a way both the act of playing our performance system
and the act of playfully designing it can be regarded as acts of music making. In his essay on experi-
antial music, Cage [3] describes the purpose of writing
and making music as “purposive purposelessness” and
“purposeless play” which should not imply uselessness
but to make use of the creativity and flow experience [5]
facilitated by a playful approach.

5. CONCLUSION

In this paper we were concerned with new possibilities
to enhance interaction between audience and artists using
everyday smart phone devices. We conducted a prelimi-
nary interview study with musicians and spectators. On
this basis we designed an interactive system with smart
phones for audience participation in a musical performance
using a playful approach to interaction and including spec-
tators and musicians in the design process. Answering
the initially posed research questions the evaluation of
this system showed that (1) musicians seem to be ambigu-
ous and cautious about giving control to the audience and
that spectators want reasonable control and clear feedback
when interacting with sound but that at the same time this
feedback distracts the rest of the audience. (2) A good
balance of constraints and affordances is crucial to both
the audience’s and musicians’ acceptance of such a sys-
tem and that this balance can be achieved by a playful
system design which includes both artists and spectators.
(3) Smart phone technology holds much potential in this
regard because of its versatility and wide spread use but
also has its problems because usage can be obtrusive to
other spectators.

6. REFERENCES

[1] A. Barbosa and M. Kaltenbrunner, “Public Sound

 bile Phone,” Proc NIME, 2011.


fer et la recherche musicale.” Institut national de la
communication audiovisuelle, 1995.

[5] M. Csikszentmihalyi, Flow: the psychology of opti-

Music Environment for Large Groups with Give-
away Wireless Motion Sensors,” Computer Music

ing tangible artefacts for playful interactions and

[8] J. Freeman, “Large audience participation, tech-
nology, and orchestral performance,” pp. 757–760,
2005.

[9] P. Heath, C., Hindmarsh, J., Luff, Video in Qualita-


erson, A. Druin, C. Plaisant, M. Beaudouin-Lafon,
S. Convery, H. Hansen, N. Roussel, B. Ederbäck, S. Lindquist, and Y. Sundblad, “Tech-
nology probes: inspiring design for and with fami-


amic patches for live musical performance,” pp.

Games - Simulation vs. Gameplay in Music-based

dology for evaluating musical controllers: A case


and performing with digital musical systems,” Com-

“Techniques for interactive audience participation,”

[20] G. McAllister and M. Alcorn, “Interactive perfor-

of ludic engagement: Evaluating participation in

niques Based on Social Mobile Computing,” Proc
ICMI, 2011.


[26] M. Robs and G. Essl, “CaMusc: Collaborative Mu-
sic Performance with Mobile Camera Phones,” Proc


[28] T. Turino, Music as social life: the politics of partic-

[29] J. Young, “Using the Web for live interactive music,”

[30] V. Zappi, D. Mazzanti, A. Brogni, and D. Caldwell,
“Design and Evaluation of a Hybrid Reality Perfor-
FOUNDATIONS OF INTERACTIVE SOUND DESIGN FOR TRADITIONAL STORYTELLING

Lonce Wyse
Communications and New Media Department
National University of Singapore
lonce.wyse@nus.edu.sg

Srikumar Subramaniam
Communications and New Media Department
National University of Singapore
srikmarks@gmail.com

ABSTRACT

The traditional practice of oral storytelling has particular characteristics that make it amenable to extending with interactive electroacoustic sound. Recent developments in mobile device and sound generation technologies are also lend themselves to the particular practices of the traditional art form. This paper establishes a context for interactive sound design in a domain that has been little explored in order to create an agenda for future research. The goal is to identify the opportunities and constraints for sound particularly suited to live storytelling, and to identify criteria for evaluating interaction designs. The storytelling domain addressed includes not only particular instances of telling, but also the variability of stories between tellers and tellers, as well as the mechanisms by which stories are passed between tellers. The outcome of the research will be a computer-based platform providing storytellers with the ability to create auditory scenes, sonic elements, and vocal transformations that are controllable in real time in order to support the telling, retelling, and sharing of stories.

1. INTRODUCTION

There is a tremendous variety of storytelling traditions across cultures extending back to (and frequently even explaining) the very beginning of time. For the purposes of this paper, we consider “traditional storytelling” to be characterized by a single teller addressing an audience using speech, physical gestures, props and non-speech sounds. Sounds might be generated vocally, with physical gesture, or with a sound-making device which could be anything from a staff to a percussion or musical instrument. Stories are in general fluid and involve improvisational components, and as part of an oral tradition, are passed from one person to another unencumbered by physical or technological baggage. The objective of the research agenda outlined here is to provide wider access to sound for storytellers than the traditional vocal and instrumental techniques, and to do so without disrupting their long-established modes of practice. This “design research” formulation implies some stringent constraints which will be studied by working with storytellers themselves.

Hand-held, sensor-rich, devices such as the Wii controller, as well as those with touch screen devices such as MP3 players and tablets, have demonstrated their effectiveness for sonic interaction in countless musical works such as those produced by the MoPho Orchestra (Wang, Esdl, & Penttinen, 2008). A hand-held device imposes minimal constraints on, and can be designed to be sensitive to the kinds of physical gestures storytellers typically make with other hand-held props or instruments when their hands are free. The sound design, therefore, is constrained in a way that these devices are no longer critically limited by computational power. For these reasons, this platform seems an ideal candidate to provide the extension of the sound palette for storytelling. However, there remain several outstanding challenges:

1) How to design an interface that works for the possibly very many sounds a story requires.
   a. The physical interaction must be appropriate for all the sounds.
   b. It cannot require physical gestures that detract from the story.
   c. It must be easily learnable.
   d. It cannot require cognitive bandwidth that detracts from the storytelling.
2) How to represent sounds to accommodate the flexibility and improvisation that characterizes the stories they will accompany.
3) How to represent the sounds in such a way that they can be customised by different tellers.
4) How to represent the story with sounds in such a way that the whole package can be passed as fluidly between tellers as stories are in the classical oral tradition.

The term “rig” will be used for the collection and organization of sounds and interfaces that are prepared in advance for a given story.

2. BACKGROUND

In most oral storytelling traditions, stories are generally not told verbatim from memory, nor are they entirely made up on the spot as they are told. They are instead typically passed from teller to teller, over generations, and sometimes over thousands of years (e.g. the Ramayana) undergoing constant mutation. For example, one can find dozens of variations of Little Red Riding Hood (many of which have been written down) – told for different purposes, and frozen at different times and locations in the course of their mutations. Red Riding Hood tales differ in every detail, from the age group of the targeted audience, to who gets eaten in the end. Even stories that are read aloud are delivered differently each time they are presented. Thus one fundamental characteristic of storytelling is its combination of prepared material and structures with variability and improvisation.

A teller is armed with a variety of prepared elements to support an otherwise improvised tale. In addition to plot elements, characters, voices, specific memorized lines, and gestural and theatrical elements, there might also be a variety of props, media, puppets, and musical instruments ready to support a tale. The guitar is one of the most common instruments for self-accompaniment in western traditions due to relatively low physical demands on anything but the hands. The pipa has been similarly used as a storytelling accompaniment in China (Werle-Burger, 1999).

A vast explosion of the sound palette could be supplied by synthetic sound. A further advantage that synthetic sound would offer may be that they could be designed to be flexible for real-time manipulation by the performer, for example, may need to wax and wane through rage calm as the story unfolds. A range of techniques from physical modelling (Cook, 1997; Smith, 1992), and sample based (Brossard, 1997) to acoustic modelling (Arflet, D., 1979; Horner, Beauchamp, & Haken, 1993; Serra, 1997; Wyse, 2004), and sample based techniques can be used to provide flexible, interactive, and when appropriate, realistic sounds under the real-time control of a storyteller.

3. ASPECTS OF SOUND AND STORY

3.1. Auditory Scenes

Sound has many unique qualities that make it an important part of storytelling (no matter what the form). Sound can create atmosphere in ways that neither words nor images can, evoking a deep sense of place (Schafer, 1994). For example, the sound of fog horns, seagulls, water lapping, and an irregular pattern of metallic “dings” of ropes hitting masts can locate us at the waterfront. We can be positioned in historical time by, for example, the kind of ring a telephone makes, or the particular quality of engine sounds.

Specific sound effects are also frequently called upon by the storyteller. They might be made vocally (the “tum-tum-scratch” of the hobbled man who creeps up on unsuspecting campers) or with the help of the hands and feet. Instruments are also sometimes used for extraneous sound effects, such as a bell for knockning. In the Thulu Bommalattam theater of Tamil Nadu, the sometimes sole performer of puppets, music, and sound effects rigs up a pair of wooden planks that can be clapped with his foot when needed (Verle-Burger, 1997). Enriching the palette of sounds available to the storyteller for creating scenes and specific effects or even abstract sounds and music with technology would be a natural extension to the vocabulary and traditional practice of the storyteller.

3.2. Text and sound

One of the most important elements of storytelling is the creation of voices for different characters. The sonic rig would offer the storyteller new possibilities for extending their voice for characters as well as for other sounds.

Of course, manipulating the voice has long been an important part of synthetic and electroacoustic sound arts. Karlheinz Stockhausen in Gesange der Junglinge (1955-56), for example, explored the relationships between voice and synthetic sound with the use of electronic and tape techniques for manipulation. Live electronic processing of voice has also been used in text
sound and sound poetry by artists such as the Swedish Bengt Johnson and Lars-Gunnar Boden, although much of this work is manipulations of sounds to create forms understood more as music than as linguistically meaningful text (Wendt, 1985).

Storytelling is a demanding environment for live performance because the control must be intimately coordinated with the vocal work of the storyteller. Microphones and amplification already commonly support storytelling, but well-designed hand-held interface is all that is needed for the storyteller to have dynamic control of how their voice sounds to the audience.

3.3 Interaction

One of the most challenging aspects of integrating a sound rig into the storytelling performance is the interaction design. Constraints and challenges come from several aspects of the specific performance culture.

The roots of storytelling are grounded in personal, family, and community contexts that do not always bear a resemblance to the concert auditorium. Storytelling has also always been accessible and practiced by communities of every economic stratum. It would be self-defeating to design a system necessitating communities of every economic stratum. It would be self-defeating to design a system necessitating constraints and challenges come from several aspects of the specific performance culture.

One approach to addressing this issue is to use inexpensive unpowered sound systems (Kapur et al., 2005) which have become so widely available. Another is to consider mobile devices such as phones and tablets as a platform. In large parts of the world, mobile phones are far more pervasive and affordable than personal computers. New devices are rich in sensors, and have important implications for the feasibility of meeting the design criteria. For that reason, several specialized programming skills needed to customize sound-capabilities to storytelling. A prototype was developed on a system that embodies many of these first-iteration design goals, notably a platform for transmitting stories in an easy and flexible way between people.

The goal of the research is not to revolutionize storytelling with a “disruptive” technology, but rather to integrate the sound essentials of interactive storytelling into the existing fabric of the storyteller’s art. The motivation is that by providing these capabilities to the storyteller, the medium might or might not appeal to a new generation, and develop in new ways without sacrificing the live and improvisational qualities not possessed by its more popular “big media” rivals such as television, film. This approach is based in the belief that preserving a tradition does not mean freezing it, but rather enabling it to adapt to a new context without its essence getting lost or obliterated.

5. ACKNOWLEDGEMENTS

This work was supported by Singapore MOE grant FY2011-FRC3-003, “Folk Media: Interactive sonic rigs for traditional storytelling.”

6. REFERENCES


A STIGMERGIC MODEL FOR OSCILLATOR SYNCHRONISATION AND ITS APPLICATION IN MUSIC SYSTEMS

Andrew Lambert

ABSTRACT

Non-linear and chaotic dynamics, predominantly used in engineering, have become a pervasive influence in contemporary culture. Artists, philosophers and commentators are increasingly drawing upon the richness of these systems in their work. This paper explores one area of this territory: the synchronisation of a population of non-linear oscillators used for the generation of rhythm as applied in musical systems.

Synchronisation is taken as a basis for complex rhythmic dynamics. Through the self-organisation notion of stigmergy, where entities are indirectly influenced by each other, the notion of local field coupling is introduced as a qualitatively stigmergic alternative to the Kuramoto model and noise, distance, delay and influence are incorporated.

An interactive system of stigmergic synchronised oscillators was developed, that is open to be used across many fields. The user is allowed to become part of the stigmergy through influencing the environment. The system is then applied to the field of music, generating rhythms and sounds by mapping its state.

2. SELF-ORGANISATION AND OSCILLATION

2.1. Stigmergy

A self-organising system is a system that forms a pattern or order without a central control mechanism or external influence. The pattern is formed instead via interactions on a local scale, with each part of the system knowing nothing of the global effect of these interactions. Self-organisation is interlinked with two other related terms, emergence and stigmergy, which seek to encapsulate self-organisation from differing viewpoints.

In emergent behaviour, a set of properties or rules are defined through which a sophisticated pattern not present in the design of these rules is revealed \([2], [16]\). The main criticism of emergence is that an observer must be present. It is only via external observation that emergent behaviour is defined. Agents within the system, by their very nature, cannot intend to produce emergence as that will defeat the point. Furthermore, it is the observer that labels that outcome of the process a 'pattern' prior to being an emergent pattern. This leads to the area being difficult to study with great accuracy.

Stigmergy on the other hand circumvents this problem through its own definition. It is another term that has its roots in the natural sciences, being devised to explain the control of collective behaviour of social insects such as ants and bees \([21]\). It is a notion common today in many agent based simulations, in that the agents remain independent entities. Their interactions with the environment affect the behaviour of the other agents, which in turn affects them. Stigmergy is therefore defined as pattern formation in a collective via an interaction with an environmental mediator.

A common example of Stigmergy is an ant following a pheromone trail to a food source. The ant in turn leaves behind a trail of its own, thus strengthening
the attraction of that path to other ants. The resultant collective behaviour is that of many ants taking the shortest possible route between the food source and the nest.

Theraulaz and Bonabeau identify two current types of stigmergy: quantitative and qualitative. With quantitative stigmergy, the stimuli-response sequence controlling that do not differ qualitatively [21] and only modify the probability of response of the individuals to these stimuli. Qualitative stigmergy differs from quantitative stigmergy in that individuals interact through, and respond to, qualitative stimuli. [21]

The concepts of stigmergy and emergence are so closely related it may seem to be a matter of semantics. However, stigmergy offers a method that acknowledges the notion of the actively observing agent and therefore offers a more stable ground for the experimentation and exploration of self-organising phenomena.

### 2.2. Synchronisation

Oscillations can be observed practically everywhere. In nature, behaviour such as a honey bee’s activity cycle and a fiddler crab’s claw waving are examples of living oscillations. On the microscopic level, every living system is controlled by internal biological oscillators, which reflect the physical, emotional and cognitive behaviour. Furthermore, some systems that appear to be one oscillator, can in fact transpire to be constituted of a whole host of oscillators, collectively exhibiting oscillating behaviour as a basic functional unit, which when combined in some way can output various dynamical behaviours, forming an oscillator population. In this way, the exact behaviour of each oscillator is abstracted away, allowing the same model to be used on several different oscillators. A population of many different oscillators can therefore be synchronised.

\[ \theta_n = \omega_n + \frac{K}{N} \sum_{m=1}^{N} \sin(\theta_m - \theta_n) \]  

(1)

(1) shows the resultant formula that has come to be known as the Kuramoto model for synchronised oscillators. \( \theta \) is the change of phase of the nth oscillator, \( \omega_n, \omega_o \) are the phases of the nth and nth oscillators, \( K \) is the nth natural frequency and \( K \) is the coupling coefficient, \( n \) is the period of the cycle, \( m \) as all each oscillator is inter-connected via a sine interaction function, the output of which is reduced to zero where the phases are identical, or differ by \( \pi \). The interaction is strongest when \( \theta_n, \theta_m \) are in phase, or differ by \( \pi /2 \), as the output of the function is either at one or negative one at this point. This means that the oscillators coupled via this model are attracted to a phase-locked synchronised state [4].

The Kuramoto model has been criticised in terms of its neuroscientific plausibility [4]. The root of the problem is that real neurons and their cortical oscillations are spatially embedded and therefore their coupling should be as well. Thus, Breakspear et al. added a time delay parameter to the model. In (2), \( \tau \) is a mapping function converting a time delay into a corresponding phase offset. Oscillators situated further apart receive the information according to this delay. This delay parameter creates a complex spatio-temporal dynamics in the population. In this circumstance, a stable synchrony becomes difficult, if not impossible, to achieve.

\[ \theta_n = \omega_n + \frac{K}{N} \sum_{m=1}^{N} \sin(\theta_m - \theta_n - \alpha_{mn}) \]  

(2)

#### 2.3. The Kuramoto Model

An artificial evolution study of the firefly, a naturally synchronising agent, hinted that synchronisation was achieved via a method where control signal flow was modified in the agent, resulting in a phase shift [11]. In nature, it is unlikely that the exact phase shifting method evolved by this method is occurring. However, synchronisation through shifting phases is not a new quality and has been explored in great detail in the Kuramoto model.

Kuramoto formulated a mathematical model centred around a generalised oscillator as a basic functional unit, which when combined in some way can output various dynamical behaviours, forming an oscillator population. In this way, the exact behaviour of each oscillator is abstracted away, allowing the same model to be used on several different oscillators. A population of many different oscillators can therefore be synchronised. The model specifies a global coupling where the interaction is still direct and in this sense it is more of an emergent phenomenon than just one oscillator. The inherent problem in the Kuramoto model in a self-organisational context is that stigmergy was not present in the initial design. To incorporate this, a radical rethink is required.

#### 2.5. Local Field Coupling

The Van der Pol oscillator (VDPO) (4) was inspired by biological systems in that it was used to model an extremely common biologically synchronised rhythm; Van der Pol used three oscillations of this form to model the human heartbeat. More recently, Camacho et al. [5] have used VDPOs to study the circadian rhythm of melatonin in the human eye.

\[ x - c (1 - x^2) x + y = 0 \]  

(4)

The VDPO obeys a limit cycle of 2 and is a reflected oscillator, meaning that voltage or activation accrues over time and releases sharply. The non-linearity coefficient \( c \) controls the rate of this action and thus the frequency. Since the dynamics of the system obey a limit cycle, natural frequency can be measured by first measuring the period of the cycle.

These systems can be coupled in a multitude of ways (see [5]), though the one believed to be the most useful here is what is referred to as the bath method. Camacho et al.'s coupling of two VDPOs via a bath is illustrated in (5), (6) and (7). Two VDPOs, \( x \) and \( y \) are coupled via a bath, \( z \). The parameter \( K \) represents a coupling coefficient, \( \varepsilon \) is the same as the Kuramoto model.

\[ \dot{x} = y \]  

(5)

\[ \dot{y} = \varepsilon (y - x) + K (y - z) \]  

(6)

\[ \dot{z} = x + y - K (z - x + c) \]  

(7)

This synchronisation method is of interest in stigmergy since it can be easily altered so that each oscillator has a local bath to which other oscillators contribute to a greater or lesser extent. These local baths can serve as a perception of the environment.

\[ x - c \left( 1 - x^2 \right) x + K (x - z) + p_n \]  

(8)

\[ z = \sum_n K_n J_{mn} \left( x_n - a_{mn} \right) \]  

(9)

\[ o_m = f (\delta_{mn}) \]  

(10)

(8), (9) and (10) show an adaptation of the bath model into a stigmergic form. The model has been extended to become \( n \)-dimensional. In this paper, an \( n \)-dimensional bath is termed a field. Each field is now local to that oscillator, termed local field coupling (LFC). The parameter \( I \) is added as a scale or influence factor between oscillators where:

\[ I \geq 0 \]  

(11)

\( D_{mn} \) is the distance between the nth and the mth oscillators when placed in the field. This influence factor is a further step to that proposed by Breakspear et al. It arises from the notion in Reynolds's Birds [19] that a close neighbour will have greater influence on the agent than neighbours further away. It is a form of signal loss linked to distance.

To make this a viable real world model, the coupling must be resilient to noise, which is accounted for by the noise parameter \( p_n \), a random number for each oscillator. Furthermore, the parameter \( K_n \) satisfies Breakspear et al.'s neurological requirements, a time delay parameter, \( \delta_{mn} \) is added converting a time delay, \( \tau_{mn} \), into a corresponding phase offset.

### 3. STIGMERGIC CREATIVITY

#### 3.1. Chronobiology and Biomusicology

Two scientific disciplines dominate the study of rhythms in biological entities: chronobiology and biomusicology. Chronobiology concerns the study of periodic phenomena in biological systems. Biomusicology, on the other hand, is the study of music and rhythmicity from a biological perspective.

A common feature of chronobiology is the notion of the essential natural nature of biological rhythms, and often the common acceptance that these rhythms are self-organised [1], [14]. The implications of this are substantial. Life cannot function without rhythmicity it seems as all living entities require biological oscillators. Many biological sub-systems are linked to the notion of time, such as respiration and locomotion, growth and death. In addition, Atta [1] states that in the case of certain animals, there is evidence of complex interactions existing with the environment, which in turn affect their biological rhythms. Although the term itself is not used in [1], this is stigmergy.

#### 3.2. Protomusic

Animal calls may be a source of insight into the animal origins of music [15]. Chimpanzees, for instance, engage in pant-hooting, which is a loud, structured, rhythmic hooting used often in chorus to keep in touch with the rest of the band. Early synchronised choral behaviour such as pant-hooting may have arisen from a need to attract mobile females. A synchronised call produces a summed amplitude and is therefore a louder call, meaning passing females are more likely to be attracted to that group of males [17].

However, Marler notes that creativity is a fundamental requirement for the origins of music. The clearly rhythmic, but not necessarily creatively musical behaviour in animals is termed protomusical, meaning that the behaviour exhibits common musical features without being defined as music.

#### 3.3. Rhythmic Behaviour as Self-organised Synchronisation

Not only is a clear operational definition of ‘good’ music hard to come by, a definition of ‘music’ is often arbitrary at best. […] Rhythm essentially refers to timing, both how long events last, and when they are scheduled to occur. […] Music […]
may be beat-oriented or pulsed, but it need not be. [...] Even arhythmic music is rhythmic [3]. Throughout this paper, I have been referring to the notion of rhythmic behaviour, rather than music. This is partly due to the difficulty of defining music and the relative ease of defining rhythm that Biles elucidates above. In the previous section, the term protomusical was defined as a rhythmic behaviour, which is the basis for musical behaviour. Biles' second consideration above notes that the concept of rhythm is inextricable from merely events in time, therefore even random temporal events can be considered rhythmic. A further step is then needed to make rhythmic behaviour more than random noise: organisation. Varese coined a commonly received base definition of music that it is ‘organised sound’ (see [9]). Merker notes that the pulse in a piece of human music is most often constant throughout the piece. We hardly ever encounter music employing discrete, that is, stepwise (from one beat to the next) and frequent tempo changes as a structural device for generating variety. [17] Therefore in protomusical behaviour, as in measured music, the organising principle is pulse. Here, the same distinction is drawn between two loose categories of music as Merker defines; either music is measured, or it is not, which also echoes Biles' sentiment of arhythmic music. This does not mean to say that in measured music the pulse is always fixed; Merker acknowledges that retardations or accents exist, but that these are deviations from a base pulse [17].

In many humans the behaviour of tapping along to music is innate, and it provides evidence for an inherited capacity to perceive a regular pulse. This has an extremely wide range of temps to which we can entrain to, which is not the case in other natural synchronisers such as insects [17]. It is not only the singular individual who has the ability to entrain to a pulse, but many individuals can mutually synchronise their pulses. Hence Merker states, “musical pulse is a cardinal device for coordinating the behaviour of those individuals in a joint, coherent, synchronised activity.” It is a fundamental building block for musical group activity, even individual musical acts such as piano playing require a sufficient level of biologically internal synchronisation to function within the group. According to Merker, there is good evidence to suggest that the synchronised churring of many animal species, or the phase in an oscillatory system, arising out of a competitive strategy the males employ. Each male seize their call in competition to be the first, thus causing a synchronising effect [17]. By definition self-organisation is also an epiphenomenon: it is a secondary occurrence, arising out of the primary behaviour. Merker states that the human notion of music has evolved out of protomusical synchronised rhythm. This suggests that artists may benefit from modular scientific components in their art systems.

4. THE CRICKETS SYSTEM

4.1. System Aims

This paper presents a system for simulating and generating protomusical behaviour. The system, named Crickets, provides an environment in which oscillators are subject to the stigmergic self-organised local field coupling (LFC) algorithm set out above. Traditionally, the phenomenon of oscillator synchronisation has remained within an engineering discipline: chaotic dynamical systems. Hence, there is now a pervasive influence in contemporary culture, with artists, philosophers and commentators increasingly drawing upon the richness of these systems in their work. SymbioticA's Silent Barrage [20] is one such example of an artistic project that integrates many disciplines and has been widely praised in the art world for it. According to Leman, artists are increasingly need the support of scientists in order to explore these areas [13].

Colton [7] hints that in the case where artists have explored these domains, a lack of skill has lead to the work not realising its full potential. This was certainly realised by SymbioticA, since the group was founded by a cell biologist, a neuroscientist and an artist, arguably giving their work more validity. Leman goes on to state that, "Multimedia in art is no longer just a matter of bringing together different art forms on the scene. Instead, the visual forum offers means to synchronise and integrate of microlevels of information processing." [13]. This suggests that artists may benefit from modular scientific components in their art systems.

This separation between scientific model and creative application was a key aim of Crickets. The final system is able to be utilised in a variety of different applications. As discussed above, protomusical behaviour has a variety of different applications in art such as poetry and dance, but it also has applications in science, in areas such as neural oscillation research (see [18]).

Crickets is a simulation tool, providing the microlevel of oscillator synchronisation modelling, and as such it is intended to be a component of a larger system. To date, Crickets has only been applied and tested in music generation systems.

4.2. System Overview

Crickets encapsulates a qualitatively stigmergic oscillations synchronisation environment and interaction interface for that environment. A user interacts with the Crickets interface and the oscillation environment is affected. Crickets then broadcasts its data to other applications, known as patches. Each patch parses the data and produces some output as a result, be that audio or other. Thus the user is reacting to the output of the patch but interacting with the Crickets system, resulting in a closed feedback loop. Crickets sits in the same vein as systems such as Lambda, Sonic Olfage [18] and Silent Barrage [20]. It is a hybrid of these three systems in that it is an environment for exploring a specific self-organisation phenomenon whilst also acting as an interface for controlling and thus becoming a component in a larger system such as an art installation. The majority of the computation in Crickets occurs in an implementation of the LFC model. In the current version of the system, all Van der Pol oscillators. The coupling-coefficient K has been implemented as a global parameter for every oscillator. This is mainly due to simplifying the interaction of the individual oscillators in the field. Allowing for specific coupling values between pairs of oscillators is possible, but creating an intuitive control proved to be a challenge. It seems for the moment that in terms of the usability of the system and so was dropped in favour of the global parameter. Specific values for K between pairs of oscillators can still be adjusted through the user interface, and in terms of the influence variable, I is used to scale K depending on the distance between two oscillators. This acts as a distance dependent weight between the two oscillators.

Crickets uses the Open Sound Control (OSC) protocol to communicate its state with other programs. The OSC protocol was chosen for its flexibility in being able to communicate with many hardware and software systems.

There are two types of OSC messages broadcast: update and field. Update messages are sent for every crickets experience on every simulation step, which currently runs at 60Hz. The message contains the cricket's identification number (ID), output level, selected state, and x and y coordinates. Field messages are broadcast once when the system starts up and subsequently when a cricket's local field has changed due to movement, change of range, or other interaction that affects the cricket's influence level. A field message contains the cricket's ID and information about the IDs and influence level of each cricket in its local field.

4.3. Results

Crickets' ability to be used for protomusical behaviour has been evaluated through user testing. An informal qualitative study was undertaken in which the system was explained to the users and they were given approximately one hour to interact with the system and some example patches implemented in SuperCollider. All the users who took part were computer literate musicians with varying amounts of experience using other music software. Some were also experts in fields such as performance art, others were politicians... The users were encouraged to talk aloud during their experiences and ask questions when they occurred to them.

A key theme in the users feedback was that of exploration. All users mentioned that by interacting with Crickets they felt they were exploring the different sounds and rhythms one could create. Often the results could not be predicted but a sense of order was present in the system. One user even commented on the emergent properties of the system, stating that complexity is hard to design and achieve. The users were encouraged to talk aloud during their experiences and ask questions when they occurred to them.

Some even described it as a potential musical instrument and here it was clear that the point where Crickets ended and a patch began had become blurred. Whilst this is an interesting perception by the user and shows a strong connection between interface and sound, it should be discouraged until patches for Crickets have fully considered the mapping between data and output (see [10]). This is not the same as with GUS since designers of patches for the system can consider it for each patch they create. All users immediately saw the potential for Crickets to be applied in fields other than music and design interest to create their own patch for the system.

Interfacing Crickets with a video system was a common idea among the user group. Robotics, lighting, and musical effects were other areas of interest.

5. CONCLUSIONS

The phenomena of oscillation synchronisation is observable in countless places in the natural world. Living and non-living systems entrain and synchronise microscopic oscillators to form one large oscillator on the macroscopic level, which can in turn be synchronised. This suggests that artists have achieved their synchronisation is largely accepted to be via self-organised processes. However, the neurological
plausibility of Kuramoto’s synchronisation model has been questioned and the self-organisation aspect, in particular in terms of stigmergy, is not present. It has thus been reworked here into a stigmergic model for oscillator synchronisation: local field coupling (LFC). The fields of chronobiology and biomusicology further elucidate oscillator synchronisation in living systems and the phenomena has been used to explain many forms of behaviours in those systems from their activity cycles to their development from birth to death. Even the animal origins of music have been suggested to arise out of synchronisation phenomena: a clearly rhythmic, but not necessarily creatively musical behaviour is achievable through synchronisation. This behaviour is termed protomusical.

This paper proposes that protomusical behaviour can be achieved through self-organised, stigmergic behaviour generated by the system to be reused in many applications across disciplines.

6. REFERENCES
compose an experience for the audience, which they intend to be inspiring on an interactive and musical level. It should be artistically satisfying and user-friendly in its operation. It is a further development in their work, in which they have always combined (live) electronic and acoustic elements, and in which space has always been of central concern.

3. THE ART OF WALKING

In the 1960s the French philosopher and artist Guy Debord organized city walks that he called ‘dérives’: wanderings in which people were guided by their own intuition and by chance, shunning routes that city-planners would have them follow. These strolls were in themselves pieces of art, composed of confrontations with the unexpected and the undetermined.

About one hundred years earlier the French poet Baudelaire came up with the idea of the passionate observer, whose preferred environment is an endless perpetually moving crowd. The observer watches and ruminates; is prepared to be diverted or delayed; doesn’t opt for the shortest route, but leaves time and space for pacing and daydreaming.

Devising the Walk With Me app has been done in the same spirit of aimless walking and pointless daydreaming.

4. WALK WITH ME IN A CONTEXT OF LATTER DAY APPLICATIONS

Developments in hardware and software have made tools for storing and processing data smaller in size, more powerful and easier to use, to the point where hand-held pads and smartphones can perform operations for which institutions and companies needed sizeable rooms filled with noisy computers some forty years ago. The portable and versatile nature of contemporary processing devices, further enhanced by the functionality of their connection to the internet, facilitates their use on the go.

These developments have led to the design of a number of applications.

- ShaMus, as described by Georg Essl and Michael Rohs of the Deutsche Telekom Laboratories [1], “is a sensor-based approach to turning mobile devices into musical instruments. The idea is to have the mobile device be an independent instrument on its own right. The sensors used are accelerometers and magnetometers as can now be found in some commercial mobile devices. The sound generation is also embedded on the phone itself and allows individual and unthethered gestural performance through striking, shaking and sweeping gestures.”

- Mediascape, software developed by HP [2], uses GPS to trigger sound files. At ICMC 2008 David Drury presented his Mediascape composition Piece Lines. Mediascape had its origins in 2002 as Mobile Bristol, a project that explored this earlier in their composition ‘Air Sensible’ – Mediascape, software developed by HP [2], uses GPS to trigger sound files. At ICMC 2008 David Drury presented his Mediascape composition Piece Lines. Mediascape had its origins in 2002 as Mobile Bristol, a project that explored this earlier in their composition ‘Air Sensible’ into one acoustic reality. Strijbos and Van Rijswijk established a symbiotic relationship between the instruments and their electronic counterparts. Walk With Me puts the realtime processing tool in the hands of the audience. Their movements between specified spots in a delimited area trigger sound effects altering environmental sounds to a variable extent, depending on their position in relation to these spots.

Weaving their way through these spots the audience create sequences and patterns. The composers relinquish control, leaving it to the audience to decide on their ‘final’ version of the piece, which is in essence aleatoric. And consequently to undergo the surroundings that are the source of the basic material in a novel way. In this sense Walk With Me is not in itself a finished composition. Analogous to the conceptions of Baudelaire and Debord the art is in the act of walking around an area and listening to the sound events that emerge in it, part of them predetermined, part of them occurring by chance. It is the listeners who complete the composition.

Strolling around in the mind or over the face of the earth may lead to incomparably beautiful conceptions. Devising the Walk With Me app has been done in the same spirit of aimless walking and pointless daydreaming.

What sets Walk With Me apart from the above applications is that it uses GPS to trigger sound files, realtime DSP of audio streams, both live and prerecorded (all of these in combination); that it uses digital musique concrète elements in realtime; and that it uses DSP and GPS with environmental sounds. The app and its server-side infrastructure were developed by Elephantscandy.

5. STROLLING LISTENERS COMPOSE AUGMENTED REALITY

Walk With Me redlines an environment on a sonic level by creating zones around selected spots. Using GPS these hotspots are fixed with a marker. Each location is linked to a certain mode of signal processing, which intensifies closer to the center of the zone around it. The effects used are reverb, tremolo, pitch shifting, delay and equalization. The result is a multi-layered, ever-changing sound piece that covers the entire area: some hotspots trigger prerecorded music or sounds; other hotspots trigger the processing of sounds picked up on the spot by the microphone of the smartphone (see Figure 4 and 5); zones around hotspots overlap (see Figure 6); because of the sonic transparency of the earpieces individuals walking with the app will hear sounds occurring around them.

As a composition Walk With Me plays on the idea of augmented reality, a merger of physical and virtual worlds. In Walk With Me different sonic worlds collapse into one acoustic reality. Strijbos and Van Rijswijk explored this earlier in their composition ‘Air Sensible’ for accordion duo and live electronics, in which they established a symbiotic relationship between the instruments and their electronic counterparts. Walk With Me puts the realtime processing tool in the hands of the audience. Their movements between specified spots in a delimited area trigger sound effects altering environmental sounds to a variable extent, depending on their position in relation to these spots.

Figure 3. manual screen app

Figure 4. multi-layered hotspots

Figure 5. triggering & processing of sounds

Figure 6. zones around hotspots overlap
6. CONCLUSION

Site-specific sound art has been around for several decades, but now developments in technology can have individual listeners walk along a self-chosen path within a composed sound environment. The Walk With Me app [7] has been operational since 2011, and has been devised for numerous places, such as Berlin, London and the Liberation Route. This new combination of GPS, realtime processing of both ambient sound and composed files, and the unique characteristics of smartphones has opened a vast area of new possibilities for contemporary composers. And it does invite new additions using a whole array of parameters, from strength of light to the intermediate distance between users of the app.

7. REFERENCES

2. www.hpl.hp.com/mediascapes/
4. www.davososoundscape.ch
5. www.thefifthandpath.net
6. http://rjdj.me

Pieterjan Gyselinck and Jeroen Jaspers
Department of Music
University of Antwerp
Koningin Astridlaan 23
2610 Antwerpen, Belgium
jeroen@elemenop.com
jeroen@elemenop.com

Figure 6. site-specific hotspots

THE AUGMENTED DRUM KIT: AN INTUITIVE APPROACH TO LIVE ELECTRONIC PERCUSSION PERFORMANCE

Christos Michalakos
University of Edinburgh
Department of Music
cmichalakos@gmail.com

ABSTRACT

This paper aims to outline some aspects of ongoing research towards the development of a computer-mediated electronic augmentation of a traditional four-piece jazz drum kit. The highly customised instrument consists of a traditional drum kit mounted with triggers, contact microphones, speakers, and bespoke software. The acoustic kit becomes part of the control interface of the electronics with the use of various machine listening techniques, and mapping strategies. Firstly, I will present an introduction to the history of the drum-kit as a constantly evolving instrument, supported by examples, and I will also discuss its relationship with the computer. Secondly, I will expand the aims of the research and the technical details of the setup, along with some of the modes of interaction methods for sound transformations through examples. Finally, I will evaluate the success of the system and its use so far, along with possible future directions.

1. INTRODUCTION

“Percussion music is a contemporary transition from keyboard-influenced music to the all-sound music of the future.” - John Cage

The drum kit as an instrument has two very distinct characteristics: firstly, anything can be considered to be percussion. From gongs and cowbells, to a prepared snare drum, any sound making object can be incorporated into the percussionist’s sound palette. Secondly, the evolution of the drum kit tells of a history of inventions and augmentations, in order to make a single percussionist capable of having at their disposal a wider range of sounds simultaneously. The hi-hat, the snares of the snare drum, the bass drum pedal and the cymbal stands are a few obvious examples. Also, within orchestras, the percussion section has been one of the first places for sonic experimentations, with pieces such as Edgard Varèse’s Ionisation [4] incorporating anvils and sirens. In the free improvisation scene, some of the best examples of instrumental expansions come from percussionists, such as Chris Cutler [2] and Tony Oxley [1]. These improvisers were also among the first to use amplification and real-time electronic sound transformations as part of their setup. In this respect, it could be argued that the drum kit has some similarities with the computer as a performance instrument. The two share the fact that they can be highly customised and adapted to the specific needs of the performer. As with the percussionist, the laptop artist builds their instrument by assembling different modules and instruments that fit their aesthetic, programming their own effects or modifying existing ones. Custom-made cymbals and bespoke software environments, paint cans used as drums, or noise-gate modules used with extreme values as real-time sound processing effects, can all be seen as different applications of the same ideas of customisation and repurposing.

2. AIMS OF THE RESEARCH

This research aims to develop a highly personalised electronically augmented drum kit, making use of the computer as the main augmentation device. In contrast to simple sample triggering (as employed by conventional electronic drum kits) the electronic part of the kit is designed to interact with all performance elements and variations, maintaining the responsive qualities of the acoustic instrument. Previous work towards electronically augmenting percussion was taken into consideration, for example An Augmented Snare Drum [4] and The Augmented Djembe Drum [5], as well as recent commercial products such as the Korg Wavedrum®. It was decided, however, that the system would be purely based on an acoustic drum kit, using mostly live audio for the extraction of control data. Although continually a work in progress, this required lengthy periods of time dedicated to practice and improvisation, without changing the system, in order to learn its extended capabilities intuitively. As jazz saxophonist Ronnie Scott put it, there was an effort to “become as close to the instrument, as familiar with it, as possible. The ideal thing would be to be able to play the instrument as one would play a kazoo” [1]. Performing with an augmented instrument, or indeed with any acoustic instrument and live electronics, can be challenging, mostly due to the need to learn new gestures which are often alien to the

1 The Future of Music: Credo (1937)
2 Chris Cutler’s kit description: http://www.cutler.com/cutler/
3 Audio Example: Tony Oxley - Idioms:
http://www.discogs.com/Tony-Oxley-Idioms/release/659887
4 http://www.korg.co.uk/products/wavedrum/whdr
5 http://www.icsrim.org.uk/augdrum
acoustic instrument. Pauline Oliveros describing the rise of complexity of her setup wrote “I experienced a new kind of performance frustration - how could I control in the development of NeViS [2], a networked cueing system for improvisation. It was used most notably for the performance of Socks and Amo at NIME, 2011, a work for hybrid piano and the augmented drum kit.

3.1 Inputs

The signal inputs of the patch can be generally divided into two categories 1) inputs used only for control data; 2) inputs used for sound processing and some control data. Controllers and microphones used include:

- 4 drum triggers mounted on each individual drum (Figure 1), 1 contact microphone attached on a cymbal or metallic spring (Figure 4), 1 drum pad, 1 Korg nanoPad MIDI controller, 1 expression pedal and 1 switch pedal.
- 2 DPA microphones attached on the drummer’s wrists, or up to 4 x AKG clip microphones.

Figure 1. Triggers attached on the drum frames

Each of the control data inputs can affect each of the electronic sound modules in different ways. However, every set of inputs has a specific type of acoustic sound behaviour in mind. The drum triggers are used for onset attack detection on the individual drums, and envelope following with a quick attacks and decays. The contact microphone attached on the cymbal or spring is used for longer amplitude envelopes as the spring can keep vibrating for a longer period of time after its excitation. The same applies to the cymbal. These are used for producing longer amplitude envelopes for certain processing modules, making the spring and cymbal themselves physical amplitude controllers.

A specific example encompassing all of the features described above is the granular synthesis module. The drum triggers provide information on the density of the physical performance, affecting the granular grain density. Also, when hits on the snare drum exceed a certain level, the granular density is maximized for a few milliseconds creating bursts of grain clouds with every hit. Finally the type of drum (bass drum, snare drum, etc.) determines the grain pitch. The piezo microphone acts as the amplitude envelope for the module, so in order for the aforementioned effects to be audible, one needs to keep exciting the cymbal or spring. As soon as the amplitudes and controls are applied on the modules. Thus in combination, even though it is not entirely obvious how the electronic sound is affected, it is clear to the uninstructed observer that there is a strong connection with the acoustic performance.

The drum pad is used to freeze all of the control data of the active modules. This was employed to solve the problem of maintaining constant interaction between the acoustic performance and electronic sound. During improvisations, I often required the electronic sound to stay at the same place while the acoustic performance could go elsewhere, or move around for a while without affecting the electronics. The term freeze here does not refer to spectral freezing, but to unchanged control data, retaining the current character of the electronic sound. A hit on the pad would make the active modules stop responding to the acoustic performance (for example keeping very dense granular synthesis grains regardless of the acoustic performance). After this, if new modules are initiated, they will be responsive until the detection of a new hit on the pad, maintaining the same character.

Any hit on the drums exceeding a certain level will make all modules in this mode go back to listening mode, resuming responsiveness.

Despite the use of triggers for expressive control over the electronic sound, there was a need for specific control over certain parameters where the outcome could not rely on machine listening processes or combinations of gestures. For example, being able to force the volume of the overall sound to zero, and starting or stopping sampling processes at specific points of the performance would have to be controlled more directly. For such reasons, an expression pedal and a foot switch were incorporated into the system. The sustain pedal was used in multiple ways (above simple mapping of its 0-127 expressive range), according to its value and speed of value change: Action A (Boolean), when its value is 0; Action B (Boolean) when its value is 137; Action C (Boolean) when the pedal is idle for more than 300 milliseconds; the actual value of the pedal.

An extensive experimentation with mappings and rehearsals is now possible to control a very significant amount of data intuitively with a single pedal. For example, Action C is used to turn the overall sound volume up or down (with ramps) when there is no new incoming data from the pedal. Whenever I want a very sudden cessation of the electronic sound, I simply have to take my foot off the pedal. This gives a significant sense of control when performing. If I need to access other controls, and have to take my foot off the pedal but do not want the electronic sound to stop, I can hit the pad as described above, and the current control data (which includes the pedal) will freeze, making it possible to maintain the desired amplitude while moving away from the pedal.

The switch pedal is used mostly for sampling, and can be perceived as a functional gesture. Even though it affects the overall electronic sound, this does not happen directly (as in the case of the drum triggers). The effects only become apparent by the impact of the controls, such as the expression pedal, triggers or pizzio. This could be likened to functional gestures of the acoustic performance such as changing drumsticks, turning the drum snares on, or changing the tuning of the floor tom during the performance. The fact that I change drumsticks will not affect the sound unless I hit the drum.

3.2 Sound diffusion

After discussions with Swiss percussionist, composer and improviser using live electronics, Christophe Fellay, in March 2011, I decided to adopt a localised speaker approach, rather than sending the sound to a wider stage PA which isolates the electronic sound from the direct acoustic sound. The idea being that the electronic sound is a part of the instrument, and thus it should be close to the acoustic source. Of course, depending on the venue, the whole electroacoustic sound could be reinforced further by a pair of overhead microphones, but this should be something to be decided according to the needs of each performance. This approach also helps to have a sonic experience closer to that of the audience. Being able to perform comfortably while feeling inside the electronic sound is one of the most important aspects when improvising with an augmented acoustic instrument. Expanding this idea further, I placed a third speaker below the floor tom that would create high reflections and resonate the tom membranes (Figure 2).

Figure 2. Feedback floor tom

By placing objects on top of the vibrating tom, such as small rocks, rice, twigs or chopsticks, it became possible to create slowly evolving soundscapes by impacting the bounces. Also, by pressing the skin with different amounts of force and on different positions, different feedback overtones and amplitudes are generated. Apart from the range of sounds being produced, one of the most important features is the physical control of the electronic sound. Performing on the feedback floor tom could be described as a physical struggle to maintain a balance between complete feedback and complete 1 http://www.nime.org 2 http://www.ddrum.co.uk/ddrum/drumtriggers.html 3 http://www.korg.co.uk 4 found on the back page.
dampness. Placing too many objects or damping the top skin of the tom with an open palm will stop the resonance and thus also the feedback, providing a direct way to route the feedback generated sound without the use of a MIDI controller.

3.3 Graphical User Interface

All mappings and controls were designed to prevent me from having to look at the laptop screen while performing. Theoretically, I should be able to close my eyes and reach the desired electronic “places” with the same ease as hitting a cymbal by remembering the location of the control. Theoretically, I should be able to close my eyes and reach the desired electronic “places” with the same ease as hitting a cymbal by remembering the location of the control system. Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there were no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.

4. CONCLUSIONS

The augmented drum kit (Figure 4) was presented both solo and collaborative settings in numerous festivals, most notably: Sonorities, NIME, BEAM, Dialogues, Soundings, and Network Music Festival. It was also used for the recording of a live solo improvisational album, Etrìction.

Although always a work in progress, the modes of interaction and control have remained successful for a significant period of time and there are no plans to change the framework in the near future. Even though the actual sound processing modules may change (in the same way that a cymbal can be replaced), or be expanded on by the addition of more features, or indeed become more efficient, the control system is not likely to change soon. Having developed the augmented drum kit over several years, the instrument feels extremely intuitive and allows me to perform in a wide variety of situations with the same expressiveness and response as I would have with a purely acoustic instrument.
[2] is a location-based, treasure hunting game, in which players navigate to a specific set of GPS coordinates and attempt to find a container (usually containing a logbook of all previous players to that location). The game makes use of an online community to generate and discover ‘geocaches’, as well as provide feedback and reward players for participation. Geocaches are often hidden in a location, which the player must be able to access by being physically in that location. Sometimes, real rewards can be gained by discovering a geocache that another player has hidden somewhere.

Exploration and audience control are key concepts in many of these previous examples and in Kubisch’s ‘Electrical Walks’ mentioned above. These concepts are also important in location-based oral documentary, ‘[murmur]’ [11]. ‘[murmur]’ takes a very similar form to many of the soundwalks mentioned above; users are given maps that highlight specific locations within an area and they are advised to travel to any point and call a telephone number situated on a sign there. Users are able to listen to accounts and memories about each specific location, whilst being physically in that location. The maps used in ‘[murmur]’ do not predetermine a route between each site, or an order in which to visit the sites, which is left to the decision of the user. In ‘[murmur]’ the emphasis is on using locitive technology to engage an audience with other people. One particular aspect of ‘[murmur]’, that involves simple text into the development of ‘SoundExplore:Leeds’ is the addition of a non-locative, online archive of the project. This is useful for pragmatic reasons, such as the presentation of the project outside of the urban environment.

Of the recent works in this area, ‘noTours’ [8], ‘Sonic City’ [12] and ‘Net_DERIVE’ [13] have the most similar objectives. ‘Sonic City’ project aims to integrate itself into everyday life [12], by allowing users to interact with the urban environment in ways that they may already be doing so. ‘SoundExplore:Leeds’ explores a similar idea; by using technology available in most contemporary mobile phones, it is able to be accessed by a larger number of people at any time. The use of mobile phones does mean, however, that a system of sensors as sophisticated as those at use in ‘Sonic City’ is not yet possible.

3. DESIGN GOALS

Building on the ideas put forward by Schafer, Truax and Westerkamp, as well as some of the recent projects mentioned above, some design goals were conceived. All of the following goals were conceived with the concept of creating a greater engagement with the soundscape through the use of technology.

3.1 Encourage appreciation of the urban soundscape, by highlighting particularly interesting elements in each area

This design goal is approached two ways. The first is the most simple, and involves simple text into the development of ‘[murmur]’. ‘[murmur]’ describes this as augmented aurality, that is, adding a new layer of audible reality to a specific location. The ‘noTours’ application uses a web-based text editor for the creation of each iteration, and an Android application in order to experience it. So far, it has been used for projects ranging from soundwalks to location-based poetry and fiction [8]. ’SoundExplore:Leeds’ differs from ‘noTours’ in that it encourages an engagement with the actual sounds of each location, as well as the sounds generated by the user. Technology in ‘[murmur]’ demonstrates the social capabilities of mobile phones, by creating a virtual network of users [13]. The project asks users to explore the city in their own way, and uses the relationship of each user’s location to create an audiovisual composition.

‘Sonic City’ project that encompasses a series of wearable sensors that enable a user to create electronic music by interacting with their urban environment. The system involved in creating the music uses not only physical location, but also measures light, pollution, speed travelled, proximity to metal objects as a way of interpreting the user’s surroundings. This data is then mapped onto the sound generation parameters to alter the musical accompaniment in real-time. The music created with a ‘Sonic City’ walk uses the live concrete sounds of the city as source material, which is then processed and played back to the user.

The ‘Sonic City’ project aims to integrate itself into everyday life [12], by allowing users to interact with the urban environment in ways that they may already be doing so. ‘SoundExplore:Leeds’ explores a similar idea; by using technology available in most contemporary mobile phones, it is able to be accessed by a larger number of people at any time. The use of mobile phones does mean, however, that a system of sensors as sophisticated as those at use in ‘Sonic City’ is not yet possible.

3.2 Allow and encourage anyone to submit material to the project

One reason that the listener is not informed of any key sounds in each zone beforehand, is to encourage the listener to discover their own interesting sounds. In addition to this, the system has been designed to accept musical material from anyone. The dynamic composition, created by the application, consists of fragments of sounds that anyone can submit. Similar to ‘[murmur]’, this approach allows for listeners to engage with sounds that other people find interesting, and unique to that soundscape.

3.3 Promote and Reward exploration

Similar to some of the soundwalk maps discussed above, listeners are aware of the areas to visit but are able to determine themselves what route to take between them. In this project, the instructions for navigating the city centre are left purposefully vague. The interface displays the current physical zone that the listener is in, and indicates the nearby zones by displaying them on a map. This is important in emphasising the act of exploration.

Listeners are also advised that there are a number of key soundmarks of specific signification to an area [9], in ‘SoundExplore:Leeds’, but are not informed the locations of them. When a listener discovers a soundmark, they are presented with a short recording of it, and are able to listen to it at any point during their walk. A listener can then ‘collect’ their discoveries, as they find others.

3.4 Widen access to project by using ‘end-user’ technology

It was important in the development of ‘SoundExplore:Leeds’ to allow for as many people as possible participate in the project. This led to the decision to develop on widely available technology, specifically smartphone and tablet computers, unlike, for example, ‘Electrical Walks’ or ‘Sonic City’ which both use more specific hardware that must be loaned to users at the start of the walk. In order to simplify the process of participation in the project, a web-based application was developed that listeners could use to download any software to use ‘SoundExplore:Leeds’, and should be able to do so without any modification to the phone.

4. TECHNICAL DEVELOPMENT

The main part of the application makes use of HTML5 technologies: the audio object, and its manipulation via JavaScript, and geolocation, via the ‘navigator’ object. The main part of the interface takes the form of a ‘Google Maps’ map, chosen for its recognisability and its ease of display, with an overlay of a Leeds city centre map segregated into numerous physical zones. The core of the application forms two parts: the geolocation, and the audio generation. The geolocation aspect attempts to determine the users location by using the ‘navigator’ object, an HTML5 standard, to constantly retrieve GPS data. This data is limited to the hardware available on each phone, and all users are advised to turn on Wi-Fi and GPS (if available). Using GPS alone proved in testing to be a slow process, since it could take a while to determine the phone’s location with any great degree of accuracy. Using Wi-Fi proved much faster, but less accurate, so a combination of the two was preferable. The application tracks the accuracy of the geolocation data, and will not pass on that which is too inaccurate. Furthermore, the application will attempt different methods of retrieval if the accuracy is consistently low. Another significant feature is the use of the fact that they may need to turn on some functionality in their phone.

Once the location information is determined, and is accurate enough, it is checked against the database of coordinates corresponding to each zone. The result of this check is then used to update the interface, and inform the audio generation section of any change in state.

On initialising, the audio component will generate a list of all available sounds, and the zones associated with them. It does this by using PHP to examine the files on the server and dynamically generate JavaScript containing all the information. One problem that occurred when developing the system was that each mobile browser had only the capability to read certain file types. This meant that each sound would have to be included in several formats.

The audio generation consists of a number of HTML5 audio objects, into which sounds are dynamically loaded and played back. The web browser’s file system is an audio object that is not currently playing from the list of available sounds in that zone, and load it into a free (i.e. not currently being used to play back audio) audio object. When a new audio object has finished playing back a file, it becomes free for other files to be loaded into. This system for the playing of audio is similar to that used in the web browser, except that downloading and saving files is less accurate, so a combination of the two was used. This system is generating to the users that they may need to turn on some functionality in their phone.

Within each zone, sounds are split into two categories: long and short. This simple classification system implemented so that anyone that intended to submit material to the project, could do so without needing to have a complete understanding of the playback system.

Unfortunately, due to limitations in the Safari for iOS platform, it is not possible to playback audio files without user interaction, nor is it possible to play multiple audio files at once. This means that the iOS version of ‘SoundExplore:Leeds’ would have to be a static audio file. The idea of the project is to allow users to provide variation, and to allow for new material that may have been added by other users.

4.1 ‘SoundRepository:Leeds’

As part of development, the ‘SoundRepository:Leeds’ was created. This online repository allows users to listen

2 See http://diveintomysql15.info/ for more information on these technologies

4.1 ‘SoundRepository:Leeds’

As part of development, the ‘SoundRepository:Leeds’ was created. This online repository allows users to listen
to and download the source material for the project, as well as having access to the source recordings throughout the compositional process. ‘SoundRepository:Leeds’ can also be used to upload and submit sounds to the project, to instantly be incorporated into the audio generated. In order to encourage this activity, the process of submitting material has been simplified as much as possible. The user only has to navigate, via the map or list interface, to the intended zone, and choose ‘long’ or ‘short’, and click ‘Upload’.

5. FUTURE WORK

‘SoundExplore:Leeds’ is an application that uses recent technological developments to further engage an audience with their surroundings. By placing an emphasis on user control and exploration, as in other previously mentioned projects, ‘SoundExplore:Leeds’ facilitates this level of engagement. There are a number of ways that the application can be used to further this engagement.

Currently, the system allows for one set of audio files to be dynamically recombined, and users can submit to that set. Work has begun on expanding this to include multiple layers of sound sets, or augmented auralities, to that set. This would allow for individual users to store and collect multiple layers of sound sets, or augmented auralities, to that set. Work has begun on expanding on this to include multiple layers of sound sets, or augmented auralities, to be played back either simultaneously or independently. This would allow for individual users to store and collect their sounds in a set, and share that with others. Other possibilities involve the invitation of other artists to contribute an original set of sounds.‘SoundExplore:Leeds’ also has the potential to be used as a pedagogical tool, to encourage a wider audience to participate in soundscape listening and recording. With ‘SoundRepository:Leeds’ it is possible to demonstrate the way in which all of the material has been selected and processed to create the final result.

6. REFERENCES


PROCEEDING FROM PERFORMANCE: AN ETHNOGRAPHY OF THE BIRMINGHAM LAPTOP ENSEMBLE

Graham Booth
Sonic Arts Research Centre
Queens University, Belfast
BT7 1NN
graham.r.booth@gmail.com

Michael Gurevich
University of Michigan
School of Music, Theatre & Dance
1100 Baits Dr, MI 48109-4805
mgurev@umich.edu

ABSTRACT

Inspired by recent third-stream research in the field of human computer interaction, we describe a recent ethnomet hodological study of the Birmingham Laptop Ensemble (BiLE) and detail our approach to thick description of the group’s working methods. A core aim of this work was to examine the range of performance modalities that players engaged in during the course of a composition. Initial findings show how traditional notions of composer and performer roles are complicated by the stake holders in the collaborative process, who engage in both infrastructural and instrumental modes of design. We conclude with a series of observations that highlight the socially constituted nature of the group’s performance system and practice.

1. INTRODUCTION

This paper charts our initial efforts to characterise collaborative musical interaction in the Birmingham Laptop Ensemble (BiLE) through ethnographic examination of the group’s working practices. Historically, ethnographic research methods have been valued for their potential to highlight the complex, socially constructed nature of interaction, such as when considering how particular musical communities of practice operate [5]. This contribution has been summed up as “the demonstration that culture can be viewed from inside its arrangements and relationship” [4]. The value of ethnography in the study of ensemble performance is its potential to illuminate the differing perspectives of group members and to place the social dimensions of collaboration at the centre of inquiry. This is of particular relevance in a practice-led field which has largely been documented by its members or directors [11] [2] [10].

2. AN ETHNOGRAPHY OF BILE

The Birmingham Laptop Ensemble was founded in January 2011 by a group of postgraduate composers working within the Music Department of the University of Birmingham’s Sonic Arts Research Centre. At the time of the study, the group consisted of musicians Shelly Knotts, Charles Hutchins, Julien Guillamat, Norah Lorway and Iain Anderson, and visual artist Antonio Roberts.

PROCEEDING FROM PERFORMANCE: AN ETHNOGRAPHY OF THE BIRMINGHAM LAPTOP ENSEMBLE

Graham Booth
Sonic Arts Research Centre
Queens University, Belfast
BT7 1NN
graham.r.booth@gmail.com
3. PERFORMANCE MODALITIES

Performance modalities can be defined as the distinct activities that group members engage in during the course of playing a piece. In Laptopera Act 2, by Charles Hutchins. Our aim was to identify the stakeholders involved in each modality, the systems developed and used in each, and the way in which these systems were adopted in their totality.

In BiLE practice, the composer is charged with the task of proposing an initial idea to the group and guiding the process of collaboration to a satisfactory end point. In the case of Laptopera, the initial idea was presented in the form of a technical document which provides instructions for performers and a text score, which requires players to vocally perform and record lines taken from spam e-mails. A key feature of the piece is that recorded material can be accessed by all members of the group, allowing newly recorded lines to be juxtaposed with processed versions of existing material.

It is worth mentioning here that the score does not allocate specific modalities to specific performers; rather it is the case that all the modalities defined below are employed by all members of the group.

3.1. Hosting and Connecting to the Shared Folder

In Laptopera Act 2, the sharing of sound files between group members is specified by the composer and is fundamental to the character of the piece. This modality is a preparatory technical stage specified by the composer, which requires a single player to host a shared folder on the local network, which others can then connect to as clients. This is accomplished using the native folder sharing features of the Macintosh OS X operating system. Figure 1 provides an overview of this process, where solid lines in the diagram indicate local access by the player hosting the folder and dotted lines represent players accessing the shared folder remotely over the network. It is notable that performers are not stakeholders in this process and must simply follow a predefined set of instructions, with little room for individual differences.

3.2. Voice Recording

As stated, a key requirement of the piece is for players to vocally perform and record lines of dialogue to the previously established shared folder. Figure 1 provides an overview of this recording process. As observed during analysis, players keep a copy of the score open on screen at all times (see Figure 4b), which they refer to when choosing a line and also read aloud from during the act of recording itself. In addition, line choice also depends on other players. As stated in the technical document for the piece, once a line is recorded, other players may record fragments of it again, but cannot backtrack to a previous line.

In early rehearsals, players were able to choose between a Max/MSP or SuperCollider patch to handle recording, depending on their preference. However, the group later experienced technical difficulties reconciling these two patches and the composer took the decision to standardise the SuperCollider patch (see Figure 4a) across the group. Technologically, this emphasises the role of the composer-designer as a stakeholder in standardising interactions that are not seen as contributing valuable individual differences to the piece. From our observations, it can be said that what mattered to the composer here was not the way in which lines were recorded, but rather the way performances differ in terms of content and vocal style.

3.3. Voice Processing

BiLE practice requires members to develop their own approach to performance, where each member “is free to interpret the sound production elements of the piece” [1], representing the tacit acknowledgment that engaging in design serves as a motivation to perform. In the case of Laptopera Act 2, “sound production” refers to the way in which performers playback and process previously recorded lines from the shared folder. The type of design engaged in here is instrumental in nature and differs significantly from the composer-centred and primarily infrastructural form discussed in the previous section. Here, the performer is the key stakeholder, and the process differs from player to player according to their individual abilities and values. This does not mean that performers can design any type of instrument as while they are free to implement the low-level details as they see fit, the patch as a whole must still conform to the basic category of a voice processing instrument, as specified by the composer.

3.4. Sound File Playback

In addition to the vocal material, the score also calls for players to select and play sounds taken from Stockhausen’s Studie II. These are required at the beginning of the piece, but may also be interspersed throughout. Although approaches to sound file playback are not as highly individuated as in the voice processing modality, the process is still approached in a number of different ways, with players accessing sounds from the OS X finder (Les), from the recorder patch (Shelly), or integrating them into their custom designed instrument patches (Julien and Iain).

3.5. Posting to Chat and Starting the Clock

In this modality, players make use of a network tool, written in SuperCollider. The tool itself can be seen in sections e. and f. of Figure 4. In BiLE practice in general, chat is primarily used to announce technical problems or to check when all players are ready to begin playing. The chat function of this tool plays a special coordinating role in Laptopera Act 2, where it is used to announce codes, which give the section and line number of recently recorded lines, in the same way they are referred to in the score. This allows players to track their position within the piece at any given time.

In addition to this chat functionality, players may also use the network tool to start or reset a shared clock. This process has something in common with the above-mentioned modality of hosting the shared folder 3.1, in that a single player must take on the role of starting the clock when the piece begins.

It is notable within this modality that use of the network tool is not specified explicitly by the composer or by performers but is instead emergent in nature, in that it forms part of the existing practice of the ensemble.

4. SOCIALLY CONSTRUCTED SYSTEMS

What becomes apparent from the previous examination of performance modalities in Laptopera Act 2 is that performers systems as a whole are a complex, interlocking mix of instrumental and infrastructural software, which must be adopted in their totality in order for players to be able to successfully perform the piece. We define these here as socially constructed performance systems, where Figure 4 shows the different elements involved in such a system from the perspective of an individual performer.

Such systems represent the socially negotiated needs and requirements of different stakeholders in the group, in this case consisting of standardised infrastructural aspects specified by the composer-designer, and individuated aspects contributed by each performer-designer.

There is no doubt that socially constructed systems pose a number of challenges for interaction, such as the degree to which performers are able to devote attention to each of the constituent elements, the extent to which these elements can be successfully integrated into their practice or adapted to their needs and the ease with which players can switch between different performance modalities that the system as a whole affords. In Les’ case, his instrument was tightly integrated with the sound file recorder by design, allowing newly recorded soundfiles to be selected immediately by using a button on the Wii Remote. In contrast, Iain kept a copy of the shared folder open and monitored the addition of new sounds, before choosing them from a menu. Julien also took this approach, but dragged sound files directly from the shared folder into his instrument window.
In the course of this paper, we have presented an ethnographic study of performance modalities in BiLE’s Laptops Act, identifying the stakeholders involved in each modality and considering how multiple modalities manifest themselves as socially constructed performance systems. Our investigation shows that whilst traditional roles of composer and performer are present within BiLE practice, the consensus driven nature of the group produces a dynamic set of orientations, which do not strictly define players’ sole activities. Instead, interaction is perhaps best understood with reference to hybrid roles such as a) the composer-performer, reflecting the fact that the member who conceives of a piece is also involved in playing b) the performer-designer, reflecting the need for composers to design infrastructures which aid realisation of the piece by simplifying interaction, and c) the performer-designer, reflecting the fact that performers are required to design instruments which stand apart from each other. As these latter two roles show, design inheres within both performance and composition, but in service of different sociotechnical functions, representing the need for standardisation of particular modalities, whilst allowing for indiviualization in others. In addition, other modalities can be described as emergent, due to their reliance on tools or approaches that the group have developed as part of their wider practice.

Taken together, these findings reveal the essentially complex, socially constructed nature of musical interaction in BiLE practice. Here it is precisely the way in which the bounds of collaboration are negotiated anew - rather than their explicit formalisation - that acts as the primary driving force in the creation of new work.

6. REFERENCES


2. RELATED WORK

2.1. Musical Structure representation

There are a few works that evaluate effective methods for memorizing musical scores scientifically. Most of these methods for memorizing musical scores are based on theories derived from the experience of professional players. For example, Bernstein [2] insists that non-conscious memory is weak and may fail to provide the answer to a simple question like “what was the next note?”. Therefore, the backup memory based on conscious memory, when players are conscious of a musical structure as shown in Figure 1 and recognize the modulation point, is important. Accordingly, how to aid learners to commit musical structure to their conscious memory is a key point in the design of our proposed system.

Synder [3] describes the process of memorizing a song, which is investigated from the view points of cognitive psychology and information theory. Specifically, it is necessary for players to analyze the musical structure, and it is especially important for them to be conscious of the different points of two similar phrases. In our research, we develop a system that prevents the learner from memorizing ambiguously by presenting different points among similar phrases on musical scores, thus helping the learner to memorize musical scores correctly.

2.2. Support of Memorizing Musical Score

Players of musical instruments usually memorize musical scores for concerts and live performances. However, memorizing songs requires much effort on a part of the player as they have to play and listen to the song over again. The goal of our study is to construct a system for memorizing musical scores based on the phrase similarity. The proposed system calculates the phrase similarity in the target song, and presents the musical structures and the different points in similar phrases based on the phrase similarity. The learner can understand the musical structure immediately, and can memorize the musical score in a short time because of the reduction of duplicated learning for the similar phrases. Our evaluation results confirmed that our method had advantages compared with conventional musical scores.

1. INTRODUCTION

Players of musical instruments usually memorize musical scores for concerts and live performances. It is important for musicians to memorize musical scores. However, for memorizing musical scores, it needs a great effort on players by playing and listening the song over again. In addition, because it is difficult to memorize musical scores correctly, players sometimes play the same phrase multiple times or forget part of a song due to stress when performing in front of an audience.

On the other hand, a song has musical structures, such as motifs. Figure 1 shows an example of musical structure. There are multiple layers from abstracted layer to detailed layer, as shown in different colors on the figure. The upper area indicates a detailed layer and the lower area indicates an abstracted layer. Additionally, each bracket indicates a phrase belonging to each layer. Most of these musical structures are composed of similar motifs. In addition, information on intersection between motif A and motif A’ can be reduced to minimize the learning time. Players can memorize phrases correctly by being aware of the different points among similar phrases.

Therefore, the goal of our study is to construct a system for memorizing musical scores based on the phrase similarity. The proposed system calculates the phrase similarity in a target song, and presents the musical structures and the annotations, such as similarities and differences among similar phrases. In this way, the learner can understand the musical structure immediately, and can memorize the musical score in the short time because of reducing the shared information of the song.

A SYSTEM FOR MEMORIZING SONGS BY PRESENTING MUSICAL STRUCTURES BASED ON PHRASE SIMILARITY

Yuma Ito
Kobe University
Tsutomu Terada
Kobe University

Yoshinari Takegawa
Future University Hakodate
Masahiko Tsukamoto
Kobe University

ABSTRACT

Players of musical instruments usually memorize musical scores for concerts and live performances. However, memorizing songs requires much effort on a part of the player as they have to play and listen to the song over again. The goal of our study is to construct a system for memorizing musical scores based on the phrase similarity. The proposed system calculates the phrase similarity in the target song, and presents the musical structures and the different points in similar phrases based on the phrase similarity. The learner can understand the musical structure immediately, and can memorize the musical score in a short time because of the reduction of duplicated learning for the similar phrases. Our evaluation results confirmed that our method had advantages compared with conventional musical scores.

5. CONCLUSIONS

In the course of this paper, we have presented an ethnographic study of performance modalities in BiLE’s Laptops Act, identifying the stakeholders involved in each modality and considering how multiple modalities manifest themselves as socially constructed performance systems. Our investigation shows that whilst traditional roles of composer and performer are present within BiLE practice, the consensus driven nature of the group produces a dynamic set of orientations, which do not strictly define players’ sole activities. Instead, interaction is perhaps best understood with reference to hybrid roles such as a) the composer-performer, reflecting the fact that the member who conceives of a piece is also involved in playing b) the performer-designer, reflecting the need for composers to design infrastructures which aid realisation of the piece by simplifying interaction, and c) the performer-designer, reflecting the fact that performers are required to design instruments which stand apart from each other. As these latter two roles show, design inheres within both composition and performance, but in service of different sociotechnical functions, representing the need for standardisation of particular modalities, whilst allowing for individualisation in others. In addition, other modalities can be described as emergent, due to their reliance on tools or approaches that the group have developed as part of their wider practice.

Taken together, these findings reveal the essentially complex, socially constructed nature of musical interaction in BiLE practice. Here it is precisely the way in which the bounds of collaboration are negotiated anew - rather than their explicit formalisation - that acts as the primary driving force in the creation of new work.
3.1. Phrase Similarity

In this research, the degree of similarity is calculated in units of phrase. The phrases and their hierarchical structure are generated using the method proposed by Hamanaka et al.[4] on the basis of GTTM proposed by Erdahl et al.[5]. Additionally, the degree of similarity is calculated based on two types of similarities. One is physical similarity, such as fingering, which is dependent on musical instruments, and the other is musical similarity, such as pitch and duration, which is independent of musical instruments.

3.1.1. Musical similarity

The musical similarity is calculated based on feature values such as pitch and rhythm, which are extracted from each phrase. We employ three feature values; timing (the onset time of each note), pitch (the absolute pitch), and interval of pitch (the difference in pitch between the current note and the one that precedes).

Figure 2 shows an example of timing. We define the onset time of each note on the basis of the first note of a phrase as timing. For example, the two phrases shown in the figure have four sets of different durations, but the onset timing is similar because in both phrases there are many notes which have the same onset timing as shown by the red notes in the figure.

Our system uses DTW (Dynamic Time Warping) [6] to measure the similarity of two phrases. DTW can be used to measure the similarity between two sequences, each of which may be differently stretched or compressed in time.

The value of musical distance $d_m(i, j)$ is defined by the following equation. $N$ is the number of phrases in a musical layer.

$$d_m(i, j) = w_d(i, j) + w_p(i, j) + w_p(i, j) + w_i(i, j) + w_f(i, j)$$

Note that $w_d(i, j)$, $w_p(i, j)$, and $w_f(i, j)$ are the distances in timing, pitch, and interval of pitch, which are calculated as a result of DTW. Additionally, $w_p(i, j)$ and $w_f(i, j)$ are the weights assigned to each element.

Two phrases of which $d_m(i, j)$ is lower than the threshold are defined as musically similar phrases.

3.1.2. Physical similarity

Fingerings in playing a musical instrument is an example of physical similarity. Because positional and physical information, such as fingering, is almost mechanical memory, it is important to back this up with conscious memory by presenting the fingering similarity. Additionally, fingering is different for each musical instrument. In other words, physical information such as fingering is dependent on the musical instrument.

Physical similarity is calculated by DTW in the same way as musical similarity. The physical distance between $i$-th and $j$-th is defined as $d_p(i, j)$. Strings, frets, and musical performance techniques such as arpeggios and hammer on are an element of the DTW of physical similarity. $d_p(i, j)$ is the sum total of the distances of DTW for each element. Two phrases of which the $d_p(i, j)$ is lower than the threshold are defined as physically similar phrases.

3.2. Method of Memorization

We explain a proposed method for memorizing scores with Figure 3 and Figure 4. The proposed system has two modes: All phrases presentation mode and Similar phrases presentation mode. The learner uses the All phrases presentation mode to learn the structure of the song or to select a base phrase that is used in the Similar phrases presentation mode. Additionally, users learn the similarities and differences between the selected base phrase and other phrases with the Similar phrases presentation mode.

3.2.1. All phrases presentation mode

Figure 3 shows an example of the All phrases presentation mode. Each rectangle in the figure denotes a phrase. In our system, the learner can change the presented layer in the layers freely. The learner uses this mode to learn the structure of the current layer or to select a base phrase used in the Similar phrases presentation mode.

3.2.2. Similar phrases presentation mode

This mode presents a base phrase selected by the learner in the All phrases presentation mode and the phrases that are similar to it. We propose two types of content presentation as shown in Figure 4. The left-hand diagram shows a general musical score, and the right-hand diagram shows a summary of the similar phrases. The rightmost scores are guitar tabs. Details of the example in Figure 4 are given below, and the Roman numerals in the black dots in Figure 4 correspond to the following list.

(i) The phrases surrounded by a solid rectangle are base phrases, whereas, similar phrases are surrounded by a dotted rectangle. The numbers next to the rectangles, show the degree of similarity. In the right-hand diagram of Figure 4, the base phrase is placed at the top of the list, and other similar phrases are arranged in order from the highest degree of similarity to the lowest. The learner can easily understand the similar phrases and their location in the musical score.

(ii) The number of identical phrases is indicated with a numerical value, such as “×2”, which appears next to the base phrase. In this way, duplicated information is reduced because the learner does not have to re-remember the phrase when it reappears later in the song.

4. IMPLEMENTATION AND EVALUATION

We implemented a prototype system for memorizing musical scores. The prototype stores musical score including meta-data such as pitch, timing, and fingering, in XML. It calculates the degree of the similarity among phrases based on the musical score data. We implemented the prototype using Microsoft Visual C# 2008 on Windows.

We evaluated the proposed method by comparing with a conventional method.
tional method, we can compare Subject 1 with Subject 2 and Subject 4 who also memorized Song C or Song D using the conventional method. The equation is as follows:

\[
\frac{16}{25} = \frac{19}{27}\%
\]

We formulate the equation for each subject, and define the average number of multiple solutions of \( a_n \) as the final \( a_n \), as shown in the rightmost in the table.

Furthermore, we can calculate the primary memorization time in the case in which the subject did not use the proposed method in contrast to the memorization time with the proposed method. The primary memorization time is determined on the basis that the ratio of the primary memorization time to the time using the conventional method is equal to the ratio of abilities. For example, if we compare Subject 1 with Subject 2, the primary memorization time \( x \) is determined as 15.6 by following equation.

\[
x = \frac{1.50}{1.00} \times 15.6 = 15.6
\]

This value indicates that the primary memorization time of Subject 1 was reduced from 15.6 min to 11 min by using the proposed method. We determine the ratio of the memorization time of the proposed method to primary memorization time, and we refer to this as the ratio of memorization time. The ratio of memorization time is determined by the following equation.

\[
\frac{11}{15.6} = 0.71
\]

This value indicates that the memorization time is reduced by approximately 30%.

Table 2 shows the ratio of memorization time in all the combinations. The three columns on the right-hand side and the bottom three rows of Table 2 show the value of average, standard deviation, and p-value that indicates the statistical significance by t-test for each subject and each song.

The memorization time of Song A and Song D are reduced. The proposed system worked effectively on these songs. The reason is that over half of the phrases are similar to each other. On the other hand, the memorization time of Song B and Song C is increased. These songs have some complex rhythms that take time to memorize.

Subject 4 was able to reduce the memorization time of all the songs he memorized with the proposed method, and it had significance. He stated that the ranking of similarity was useful because he could select the phrases in order of the ease of memorization. In contrast, Subject 2 was not able to reduce the memorization time, and it had a significance. According to his comments, the prototype system had a lot of functions and he could not understand them all. For this reason, he could not get useful information from the prototype system to aid his memorization.

5. CONCLUSIONS

We constructed a system for memorizing songs by presenting musical structures based on phrase similarity. It presents the musical structure of the song, similar phrases, and differences among similar phrases. Form the result of evaluation, although the effectiveness of the proposed system depends on the musical ability of the subject and characteristics of the song, the memorization time was reduced by using the proposed system.

Future work will include experiments using subjects of varying ability and more extensive experiments. Additionally, we intend to apply the proposed system to instruments other than the guitar.

6. ACKNOWLEDGEMENTS

This research was supported in part by a Grant in aid for Precursory Research for Embryonic Science and Technology (PRESTO) from the Japan Science and Technology Agency and by a Grant-in-Aid for Scientific Research (A) (25246009) and Scientific Research for Young Scientists (21770018) from the Japanese Ministry of Education, Culture, Sport, Science, and Technology.

7. REFERENCES


1.2 Input sensors

Due to the nature of the special needs market, there are a large number of standard ‘assistive technology’ switches available for a wide variety of needs. Past system development has focused on wireless adaptors. It is important that there is a clear link between cause (i.e. sensors employed) and effect (sonic output) in special needs work [2]. Non-contact sensors can be problematic in these environments. Also sensors need to be robust and simple to use.

The current range of sensors include:

- Connect – adaptor to allow up to four switches to be linked wirelessly to the Ensemble Hub.
- Dice – each side of the sensor can be used to trigger a different sound or effect.

1.1 The Hub interface

The Ensemble Hub, figure 1, forms the main USB interface for the PC, featuring four sockets for simple on/off switches and two for variable sensors. It also contains a 433MHz transmitter for controlling proprietary sensory equipment and a 2.4GHz module for wireless sensors. A basic portable setup can be achieved by using the Hub together with a laptop or netbook PC.

- Press – sensitive pressure pad that produces a variable signal depending on the amount of pressure applied.
- Tilt – produces a variable pitch and roll output as it is rotated.
- Squeeze – an air pressure bulb whose output varies with the pressure applied.
- Dual – adaptor to allow up to two variable sensors to be linked wirelessly to the Ensemble Hub.
1.3 Designer

The Designer software is a drag-and-drop environment, with blocks on-screen representing the key components of the system shown in figure 3. These are linked together to provide a 'map' of how the system operates.

Each block can have a number of settings or options, which are adjusted in a panel on the right-hand side of the screen. Where a greater number of options are available, a separate editor window can be used, for example when entering MIDI notes or adjusting a colour on a light.

As well as input and output blocks, the software allows for tools, which can alter the behaviour of a signal in the system. These allow the user to add delays, expand the range of a signal or alter how a switch operates.

The Designer software is unaware of exactly what devices it is using, instead relying on rules for connecting and setting-up the blocks. All blocks are defined by an xml file, which can be altered to produce blocks specific to a user’s installation.

The Designer software is a drag-and-drop environment, allowing finished maps to be played without the need for the Designer to be open. This presents a friendly, CD-player style interface for users to interact with. It is particularly important in schools where any hint at complexity can stop people from using the software.

1.4 Player

The Player application, shown in figure 4, separates the actual 'playing' of maps away from the Designer, allowing finished maps to be played without the need for the Designer to be open. This presents a friendly, CD-player style interface for users to interact with. It is particularly important in schools where any hint at complexity can stop people from using the software.

1.5 Devices Monitor

Services are handled by a tray application, which updates dynamically as devices are connected and removed from the system. Separate services interpret messages for the Player software and deal directly with communication with the hardware. In this way, new services can be developed to add additional functionality to the software.

Current services include:

- MIDI Output [8].
- Gamepad Input – allows any device that appears to the PC as a gamepad to be used with the Ensemble.

2. IPHONE/IPAD APPLICATION

2.1. Overview

The mobile control module will primarily aim to satisfy the needs of teachers and special needs specialists. This requires the application to have simple controls affecting a wide range of parameters and for the controls to be intuitive and familiar so that the application can be understood and used easily. The application will also provide features for other user groups such as composers and sound designers who are able to invest more time in exploring the system.

The application controls Ensemble’s surround sound parameters. The interface is created dynamically by requesting the sound outputs and external inputs of the user design ‘map’ from Ensembl. A corresponding design map is recreated on the iPhone/iPad interface.

The design was implemented for the iOS operating system primarily using OpenFrameworks [9]. This is an open source C++ toolkit that was used within the Xcode integrated development environment (IDE).

There are several existing applications that allow users to build dynamic controllers currently available for iOS including TouchOSC [4], Liine Lemur [6] and Fantastick [10]. However these systems do not specialise in surround sound, provide limited automation and path manipulation.

2.2. Design

2.2.1. Routing of Sound

Initially, sound outputs and listener position are displayed on-screen in a grid. These icons can be arranged to reflect the position of speakers and the listener in the physical environment. Figure 5 shows a 5 speaker arrangement and the listener positioned by the user. Speaker and listener icons can be positioned using the multi-touch capabilities of the device. The user is able to navigate and zoom around the display area as well as hide the toolbars for improved design control.

2.2.2. Main Volume Control

The volume of the speakers can be controlled individually as well as globally. The user can make a custom selection of speakers or choose a selection option: ‘Left’, ‘Right’, ‘Centre’ and ‘All’. These options select speakers depending on their position defined by the user.

Volume control has 2 modes: Absolute and Relative. In Absolute mode, the volume of all speakers is changed simultaneously. In Relative mode, as speaker volume is increased or decreased, the relative volume differences between speakers are retained until all the speakers are pushed to minimum or maximum volume.

Figure 6 shows the ‘Right’ speakers selected in Absolute mode. Volume is changed using the fader in the lower menu.

2.2.3. Dynamic Sound Trajectory Control

The prototype application has 3 sound trajectories (A, B, C), each with 2 pan layers, which the user can configure, control and edit. In pan layer 1, a path is drawn directly onto the speaker array display. Once completed, the path is followed at the same speed as it was drawn.

The direction of playback can be reversed and the speed can be changed. In ‘Single Play’ mode, the loop plays once, following the pre-defined path, stopping when it is complete.

Sound sources can be routed to one of the sound trajectories. Multiple sources can be routed to a single trajectory. Figure 7 shows 3 different trajectories displayed on the speaker array. The top menu allows editing of the selected trajectory. The bottom menu shows the 3 trajectories visible and the routing of the input sources. They are colour coded for quick reference during a performance.
2.2.4. User Interface

The User Interface (UI) design is important to appeal to a variety of users. Whereas sound designers, familiar with technical control of audio, may allow for complex control at the expense of the UI, a large proportion of the users (educators) will require a more immediately accessible control interface. Therefore the application will manage workflow and available options depending on the users’ required level of control and proficiency employing a habitation pattern for menu navigation and an incremental construction pattern of the speaker array and sound trajectory design [12].

3. OSC IMPLEMENTATION

OSC messages are sent between the Ensemble system and the mobile device. OSC was developed at CNMAT primarily as a protocol for communication between computers, sound synthesizers, and other multimedia devices optimized for modern networking technology [15].

3.1. Protocol Specification

OSC employs an ‘open-ended’ URL-style symbolic naming scheme. The project aims to carry out most of the data processing within the device and to send out as few different messages as possible.

3.1.1. Configuration Messages

At the setup stage, the application sends a request message (‘/setup/config’) to obtain the quantity of speaker outputs and user inputs. The return messages are in the following form. For the speaker outputs:

\[
/speaker/vol n v1. \quad \text{Where } n = \text{the amount of speakers in the design. For the user inputs:}
\]

\[
/\text{config/soundSources s}. \quad \text{Where s = the amount of advanced time manipulation and display; multi-touch control for additional parameters and advanced path manipulation.}
\]

3.1.2. Control Messages

During operation, messages are sent when the main volume is changed. The message consists of the speaker number and the master volume in the form: ‘/speaker/vol n v1’. Where n = the speaker number; v1 = master volume (0-1). Messages are sent when the input sources are routed to playing sound trajectories or a changing live mode. Messages are sent in the form: ‘/speaker/sound n i2 v2’. Where n = the speaker number; i2 = the sound source number; v2 = the percentage scale of the speaker volume (0-1). The percentage scale value is determined by using distance-dependent mapping principles utilizing the x and y speaker distances.

Additional OSC messages can be sent which specify the x and y distance between sound source and speaker to allow for easier interfacing with the Ensemble environment.

4. TESTING AND FURTHER WORK

The controller application has been tested successfully with a range of surround sound setups using an emulator designed in Max MSP [3]. Apollo Creative has provided user interface and gesture control feedback. Further user testing is to follow. Further features currently being designed include

5. REFERENCES


MODERN TECHNOLOGY AND CREATIVITY: AN APPROACH TO COMPOSING AT 11-14 YEARS

Al McNichol

Department of Music and Drama, University of Huddersfield, Queensgate, Huddersfield, England, HD1 3DH.

a.mcnichol@hud.ac.uk

ABSTRACT

Modern technology offers creative opportunities to music composers within a sound universe of expanded possibilities. Whilst revolutionising music composition in the wider world this also provides pedagogic opportunities for 11-14 year pupils in the classroom for music composition. However, recent evidence suggests that technology is still not being used to its full creative potential for composing at this level in secondary schools. This paper presents recent developments from investigations into this issue in the context of the National Curriculum for England. It considers a theoretical interrelationship of modern technology, creative thinking and 11-14 composing pedagogy. It presents a view of current resources within secondary school music departments and discusses recent action research into the deployment of modern learning resources, including dedicated software, in 11-14 year music classrooms in the North of England. The oral presentation of this paper will include demonstration of the software, digital video of it in use in a classroom and audio examples of musical outcomes from 11-14 year old users.

1. CONTEXT

This paper presents recent developments of research first introduced in its early stages at previous ICMC [8]. Since that time modern technology has continued to develop and become increasingly accessible to many secondary school music departments, providing pedagogic opportunities for 11-14 year pupils in music composition. In England, the use of technology in the 11-14 year music curriculum is statutory. It should be used ‘to create, manipulate and refine sounds. Including the use of music technologies to control and structure sound in performing and composing activities’ [13].

To converge technology, composing and music pedagogy is a study into the development of music in 11-14 years of age in the North of England. The use of technology in the 11-14 year music curriculum is statutory. The study should be used ‘to create, manipulate and refine sounds. Including the use of music technologies to control and structure sound in performing and composing activities’ [13].

Consequently, with technology in the classroom is an opportunity for music pupils to contribute to a more open-ended universe of possibilities within a 11-14 year population. The study’s focus is on the ways in which technology can play a part in providing access to a more open-ended universe of possibilities for music pupils. The study’s focus is on the ways in which technology can play a part in providing access to a more open-ended universe of possibilities for music pupils.

Testing the software in the classroom, the investigation shows that technology is still not being used to its full creative potential for composing at this level in secondary schools. The software was used ‘to create, manipulate and refine sounds. Including the use of music technologies to control and structure sound in performing and composing activities’ [13].

2. THEORETICAL RELATIONSHIP

The development of modern technology has revolutionized how music can be composed. This has influenced theoretical and conceptual thought concerning affordances of an expanded sonic universe. A universe through which a composer can exercise creative thinking and explore musical imagination in the structuring of sound [5,7]. However, the composer must develop knowledge and skills of such technology if any use is to be made of any potential possibilities it might offer.

2.1 Modern technology: an expanded sonic universe

Modern technology affords an expanded sonic universe for music composition. The potential sonic space is an open universe of continua and possibilities ‘where every sound and imaginable process of transformation is available’ [17]. Continuums exist between fundamental sonic properties of pitch, timbre and duration offering the potential for dynamic morphology within sound structures of such a sonic space. Access to such continuums and morphology expand sonic possibilities beyond music based on three-dimensional lattice frameworks of discrete musical distinctions [18]. Such an expanded sonic universe offers alternative possibilities for a composer to explore individual creativity.

2.2 Creativity and composing music

2.2.1 Creativity

Creativity is ‘imaginative activity fashioned so as to produce outcomes that are both original and of value’ [9]. However, creativity can be an intricate concept to define theoretically and practically. Recent definitions are a separation of early research that separated creativity into four stages: preparation; incubation; illumination; and verification [4]. Guilford [6] presents a model identifying two productive abilities of convergent and divergent thinking. Such thinking and abilities are applied in a ‘model of creative thinking in music’ proposed by Webster. This model relates thinking processes with an association to creative products [16].

In music contexts, ‘the distinctiveness of creativity is that pupils use sound as the medium for creative
thinking’ [10] and for many music educators, ‘creativity is at its strongest in the act of composition’ [1].

2.2.2 Composing
Composing is a productive process producing actual outcomes. The very process of composing music involves imagining ideas, which at the point of them being imagined may be worth very little musically, but there comes a point when decision-making becomes involved [5]. It is musical imagination, the minds about to think in sound, to internalize music by ‘hearing’ sounds and manipulating ideas towards meaningful intention that is central to composing. After all, it is the positioning and structuring of sounds together and what might be conveyed that is composing [12;15].

Modern technology, creativity and music composition are interrelated as domains when combined in practice. They become interdependent on each other and ought not to be considered in isolation. The challenge is how might these be integrated practically into an 11-14 year music curriculum to support effective composing pedagogy?

3. TECHNOLOGY SURVEY
Investigation has been undertaken to discover what technology is currently available in regional secondary school 11-14 year music departments. A survey was conducted of 29 secondary schools in two regions in the North of England. The method of survey was by electronic questionnaire to each Head of Music and by follow up visits to participating music departments. What follows is a summary of the findings felt relevant to this project.

3.1 Resources: hardware, software and usage

<table>
<thead>
<tr>
<th>Resource type</th>
<th>% of schools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cubase</td>
<td>40</td>
</tr>
<tr>
<td>Finale</td>
<td>14</td>
</tr>
<tr>
<td>Sibelius</td>
<td>71</td>
</tr>
<tr>
<td>Audacity</td>
<td>86</td>
</tr>
<tr>
<td>Acid</td>
<td>14</td>
</tr>
<tr>
<td>Logic</td>
<td>14</td>
</tr>
<tr>
<td>Reason</td>
<td>14</td>
</tr>
<tr>
<td>Dance Ejay</td>
<td>57</td>
</tr>
<tr>
<td>Other (Pitsools, Garageband)</td>
<td>14</td>
</tr>
</tbody>
</table>

Table 2. Available software resources. The software resources shown in Table 2. show a prevalence of Cubase sequencing package and Sibelius notational scoring package. This is partly due to strong UK market and distribution presence towards education, although other packages do feature in Audacity and Dance Ejay. The latter package often being used in 11-14 year composing activities using pre-recorded audio, largely confined to arranging blocks of pre-composed blocks of sounds.

<table>
<thead>
<tr>
<th>Resource type</th>
<th>% of schools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Composing</td>
<td>100</td>
</tr>
<tr>
<td>MIDI sequencing</td>
<td>71</td>
</tr>
<tr>
<td>Music scoring</td>
<td>38</td>
</tr>
<tr>
<td>Music theory</td>
<td>29</td>
</tr>
<tr>
<td>Sound sampling</td>
<td>30</td>
</tr>
<tr>
<td>Sampling editing</td>
<td>34</td>
</tr>
<tr>
<td>Audio manipulation</td>
<td>57</td>
</tr>
<tr>
<td>Audio sequencing</td>
<td>57</td>
</tr>
<tr>
<td>Performance development</td>
<td>43</td>
</tr>
<tr>
<td>Critical intesting development</td>
<td>29</td>
</tr>
<tr>
<td>Other</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 3. Resource (hardware and software) activity usage.

The use of available resources shows an overwhelming dominance towards composing activities. Much of this composing is often through MIDI sequencing using template sessions or pre-existing notation relying on available software. Little use involved critical listening development. The time given over to such activities varies depending on the school, scheme of work and year group. Predominantly the lesson times are for 1 hour per week although not necessarily occurring every week over a whole term. Quite often lessons are structured in short weekly blocks of time, typically 3 – 6 weeks at 11-14 years. Usage typically increases the higher up the school towards 14-18 years. Quite often use for 11-14 years composing is quite limited if used at all. Resources for composing are rarely accessible outside scheduled class time.

This regional investigation reflected many issues reported at National level and from other large scale surveys [10;11;14]. In response to some findings intervention was planned in the form of action research with the intention of trying to develop the use of technology in 11-14 year composing activities.

4. ACTION RESEARCH
Action research is broadly considered as research undertaken by practitioners into their own practice, in order to improve it. The process is described as a recurring spiral of planning, acting, observing or evaluating and reflecting [2;12].

Action research has been implemented since 2009 in collaboration with four secondary music departments in the North of England. Early materials and results and from two of these were published at a previous ICMP [8]. Since that time three further cycles of action research have been implemented in two other secondary schools. These schools wished to integrate their resources and increase the use of technology in music pedagogy at 11-14 years particularly for composing music. Suitable hardware resources were available in the form of a networked PC lab, although this was mainly used for IT studies and not utilized for music teaching at the time. Other available technology was used in performance, using keyboards and drum performance pads. General IT technical support was available to assist in the deployment of resources across the available hardware systems.

Part of the action research has involved developing learning resources to support the teaching of a composing project using modern technology consisting of three distinct stages: digital sampling of natural sound; appraising and manipulating digital sound and; structuring digital sound. Learning resources included schemes of work, lesson plans, teaching materials, video tutorials and dedicated sound processing software. The dedicated software presented in this paper is intended to encourage and facilitate pedagogy in stage two of the project; sound manipulation. Each section is progressive and incorporates modern practice in the use of music technology for music composition. There are ‘x’ stages within these sections were pupils are able to progress at differing paces achieving a range of outcomes as they do so. Data has been collected during and after each stage through a variety of methods including digital video, digital soundfiles, still photographs, observations, verbal discussion, aural audition and feedback questionnaires. A selection of this material will be available as part of any oral presentation of this paper.

4.1 Revising software for audio manipulation
Software for manipulating digital sound has been recently developed to support this action research. The software has been renamed ‘Soundtools Manipulator’ and is a collection of sound processing tools processing in real-time, presented as a standalone application. Each tool implements a time or frequency domain processing technique designed to be simple to operate yet effective in scope. The processing techniques employed in each tool are real-time and include independent pitch-shifting and time compression/expansion. Also included are delay, filter, reverberation and ring modulation. These particular techniques have been selected for this project to allow exploration of sonic space, whilst introducing the user to simple sound manipulation techniques. The tools are intended to be easily deployed in a classroom environment to encourage creative thinking for the imaginative exploration and manipulation of sound. The tools are programmed using Max/MSP by Cycling74 and are compiled as a standalone application for both Windows or Apple computer platforms.

Since v2.0 of the software was demonstrated at ICMP revision has taken place. This has been informed by reflective practice and results from the action research above. The revisions have been graphically presented and implement automatic configuration with minimal input from the user. Such decisions were in response to user feedback and observations. Audio DSP and routing is now configured automatically although can be overridden and set to Rewire or an advanced routing matrix if required. Digital sound can now be loaded by drag/drop into a central area, giving a graphical waveform for viewing. Minimal transport controls allow playback and looping of the whole, or selection within a soundfile. The primary parameters for initial manipulation are pitch (–100.) and playback speed (=100.–). These are independent of each other and any changes made are applied in real-time.

The GUI also presents recording space and storage controls. A horizontal graph gives the available storage capacity of four minutes shown for a ‘fuel gauge’. Whilst ever the sound level is above a minimal noise floor level the DSP output is automatically downmixed to allow the user to concentrate on sonice manipulation and not worry about saving what sound is produced, whether intentional or accidental. If the user wishes to save what sound has occurred they may save a resampled soundfile to storage. If not, the storage space can be erased and the full four minutes is available once more and the process can proceed. This automatic storage process may be overridden to manual mode to capture decay of time-based effects such as delay and reverberation should these be in use.

4.2 Recent use of ‘Soundtools Manipulator’
‘Soundtools Manipulator’ has recently been used by three separate groups of 11-14 year secondary schools. The pupils are of mixed gender and abilities. It has been used for stage two of the project whilst Cubase SX and Audacity 1.3 where used for stage three on WindowsXP and Macintosh desktop computers. All computers were networked and the application ran from a central server.

The action research projects have all begun with sound capture using portable digital recorders. Pupils are encouraged to explore the sound environment around their school, both indoor and outdoor. A strong emphasis is placed on the quality of sound recording, aural skills development and use of imagination.

The resulting digital samples are used for sound manipulation using ‘Soundtools Manipulator’...
Demonstration and discussion are used to emphasize the close evaluation of sonic characteristics within the terms of pitch, duration, dynamics, timbre and texture. The stimulation of musical imagination is concerned with developing pedagogic resources for 11-14 years. It discusses investigation into this issue within Regional secondary school music departments in the North of England. It presents a view of current resources within these music departments and discusses recent results from action research. This research is primarily concerned with developing pedagogic resources for 11-14 year music classrooms. Such discussion and information may prove constructive to other music educators around the world engaged in developing the use of modern technology for composing pedagogy at 11-14 years. Educators who are interested in fostering links and collaborating in such action research are encouraged to contact the author for discussion.

6. REFERENCES


5. WIDER CONTEXT

This paper is intended to illuminate a theoretical interrelationship between modern technology, creative thinking and composing pedagogy at 11-14 years. It discusses the investigation into this issue within Regional secondary school music departments in the North of England. It presents a view of current resources within these music departments and discusses recent results from action research. This research is primarily concerned with developing pedagogic resources for 11-14 year music classrooms. Such discussion and information may prove constructive to other music educators around the world engaged in developing the use of modern technology for composing pedagogy at 11-14 years. Educators who are interested in fostering links and collaborating in such action research are encouraged to contact the author for discussion.

4.2.1 Pedagogically Speaking

From a pedagogic perspective the data collected shows that the technology could support effective music pedagogy. Pupils became quickly engaged by the technology in exploring sonic material. They enjoy exploring sonic parameters and learn what effect manipulating these has upon sound. This musical learning can become increasingly based on their own imagination, intentions and thematic ideas. However, beyond play and familiarity being able to imagine, manipulate and select sounds towards intentional composing most pupils found much more difficult. Yet some of their work and short compositions do show learning development, musical imagination and real intention in the way they try to structure sounds towards musical intention.

4.2.2 Technologically Speaking

From a technical programming perspective the revisions of SoundTools Manipulator were successful. The software application ran over the school network with no inherent issues within the software, although individual PCs did cause some minor issues with corrupt audio driver issues early in the project. The software has been used without any issues on Apple OSX iMac machines in undergraduate and postgraduate student seminars with teacher trainees. The application requires little storage space (12MB WindowsXP; 30MB OSX) and low CPU usage (8-10%) depending on the specification of the computer. The GUI is reported to be easy to use, functional, less cluttered and more intuitive. There are however more revisions under consideration. These will be implemented and tested in further cycles of action research already being planned for September 2012 and beyond.

GEORGIA STATE UNIVERSITY

MUSIC TECHNOLOGY STUDIO REPORT

Tae Hong Park, Robert Scott Thompson, Alex Marse, Johnathan Turner

School of Music
Georgia State University
Atlanta, GA, USA

park, rsthompson@gsu.edu
amarsel, jtturner30@student.gsu.edu
http://musictech.gsu.edu

ABSTRACT

This study report describes Georgia State University’s (GSU) recent activities in the area of computer music and music technology. A summary of on-going and recent research will be provided as well as an overview of the School of Music, the Music Technology programs, its facilities, key personnel, and future plans in developing the graduate and undergraduate music technology curricula.

1. INTRODUCTION

GSU School of Music enrolls approximately 450 students representing 6 countries and 21 states. Approximately 360 of the students are in bachelor’s degree programs, 70 in master’s degrees, and the remainder in other programs, including a Ph.D. in music education. 40 full-time and 30 part-time faculty serve students in concentrations including performance, music education, music management, music recording and technology, composition, and jazz studies. The music technology program currently includes faculty members Robert Scott Thompson and Tae Hong Park and approximately 30 full-time students.

The music technology program is comprised of an undergraduate concentration in music technology (B.Mus.) and a master’s program in composition with an emphasis in music technology (M.Mus.). Opportunities also exist for student-designed programs focusing on music technology within the Bachelor of Interdisciplinary Studies (B.I.S) degree track. Various courses are offered for graduate and undergraduate students including digital signal processing, computer music I/II, recording techniques I/II/III, music production, electroacoustic music composition, audio post-production and a comprehensive industry-based internship experience.

2. FACILITIES AND CENTERS

The music technology facilities at GSU consist of three recording studios, one computer music composition studio, and a PreTools-based post-production suite designed by Walters-Storlyk Design Group (WSDG) in 1995. The studios are housed in the historic Standard and Haas-Howell buildings in the heart of downtown Atlanta, Georgia.

All three of the recording studios are interconnected via patch bays, allowing users to record from any tracking room into any control room. Each control room contains both digital and analog hardware including Eventide harmonizers and various signal processing modules; Avalon, Universal Audio, and Grace Designs preamps; standard DAWS, a plethora of software plugins, and interactive computer music applications for all of the studio workstations; and Tannoy, Genelec, Adams, and JBL studio monitors. Studios are maintained with the assistance of advanced undergraduate and graduate students. This model serves as a pedagogical mechanism to help students learn the complexities of studio maintenance while fostering a sense of community.

Figure 1. Studio A control room

Other related facilities include the Music Technology Lab and the School of Music Media Center, the latter housing numerous student workstations, a music technology seminar room, a 19-station multimedia teaching lab with a dedicated teaching station, together with a large collection of recordings and reading material. The School of Music also possesses several performance venues including the 400-seat Florence Kopleff Recital Hall, with a built-in recording booth and

2. FACILITIES AND CENTERS

The music technology facilities at GSU consist of three recording studios, one computer music composition studio, and a PreTools-based post-production suite designed by Walters-Storlyk Design Group (WSDG) in 1995. The studios are housed in the historic Standard and Haas-Howell buildings in the heart of downtown Atlanta, Georgia.

All three of the recording studios are interconnected via patch bays, allowing users to record from any tracking room into any control room. Each control room contains both digital and analog hardware including Eventide harmonizers and various signal processing modules; Avalon, Universal Audio, and Grace Designs preamps; standard DAWS, a plethora of software plugins, and interactive computer music applications for all of the studio workstations; and Tannoy, Genelec, Adams, and JBL studio monitors. Studios are maintained with the assistance of advanced undergraduate and graduate students. This model serves as a pedagogical mechanism to help students learn the complexities of studio maintenance while fostering a sense of community.

Figure 1. Studio A control room

Other related facilities include the Music Technology Lab and the School of Music Media Center, the latter housing numerous student workstations, a music technology seminar room, a 19-station multimedia teaching lab with a dedicated teaching station, together with a large collection of recordings and reading material. The School of Music also possesses several performance venues including the 400-seat Florence Kopleff Recital Hall, with a built-in recording booth and
professional sound reinforcement system; the historic 833-seat Rialto Center for the Performing Arts refurbished in 1995; and two large recital spaces in the Aderhold Learning Center, which house a sound reinforcement and recording capabilities.

The music technology program has strong affiliations with GSU’s Digital Arts Entertainment Laboratory (DAEL). DAEL houses a professional video studio, green room, audience response theater, video and audio production suites, and a small performance stage. In 2011, the music technology program has also developed a partnership with the Second Century Initiative New and Emerging Media (2CINEM) research cluster with the recent hiring of Tae Hong Park as part of the core 2CINEM faculty. The goal of the 2CI is to build internationally recognized scholarly strength in research themes with national significance to enhance GSU’s quality, interdisciplinary richness and competitiveness. This initiative which has brought the first group of cluster faculty hires to the GSU campus in 2011 is part of an “ambitious faculty hiring initiative to recruit 100 additional faculty members to the university over the next five years” [1].

3. RESEARCH

A number of students and faculty are involved in ongoing research in the areas of composition, human-computer interaction (HCI), electro-acoustic music (EAM) archiving, EAM analysis and visualization, acoustic ecology, and sound synthesis. Student-based research is closely linked to pedagogical frameworks in the various courses offered emphasizing theoretical and practical mastery of the subject matter. These courses range from techniques of software synthesis to programming for audio signal processing and HCI providing a framework for students to develop elaborate research projects with faculty support and guidance.

Graduate student Alex Marse is developing software tools for interactive exploration of serial compositional techniques exploiting graphic interfaces. The software framework uses a programmable matrix interface to generate rows that can be utilized for formal designs and other various musical parameters including pitch, rhythm, and timbre. The interface, implemented in Max, uses a “drag and drop” paradigm to create modular blocks of musical material that can be organized, ordered, layered, and arranged to facilitate exploration of musical and compositional possibilities. Marse is also engaged in HCI research which focuses on capturing gestures to drive compositional elements and processing algorithms for interactive live percussion performance. The sensors that are currently being investigated include 3-axis accelerometers and Microsoft Kinect.

Graduate student Johnathan Turner is conducting research into algorithmic composition using IRCAM’s Open Music software and Python with the aim of applying sound analysis results to drive and illuminate musical gestures. He is currently focusing on using highly transient portions of acoustic signals to generate pitch and rhythm materials.

Figure 2. EASY Toolbox screenshot

Tae Hong Park is leading a research project entitled Electro-Acoustic Music Mine (EAMMM), which focuses on creating a comprehensive EAM preservation and exploration portal based on: (1) a filtered-crowd-sourced music collection module that is curated according to a credentialed peer-reviewing system, (2) a comprehensive archival module, and (3) an analysis module based on the timbre-centric Electro-Acoustic Music Analysis (EASY) Toolbox [2] providing tools for interactive visualization, navigation, and discovery of musical works. This third module exploits Music Information Retrieval (MIR) and content-based analysis research to extend and enhance traditional text-based indexical discovery and delivery systems. EAMM is one of the projects that has been developed in collaboration with the 2CINEM cluster, the GSU Special Archives and Collections, ICMA, and GSU’s computer science department. Park is working on releasing EASY Toolbox 1.0 – a software system for assisting EAM analysis. The system aims to present perceptually significant features and audio descriptors via GUI-based interfaces and visual designs to help experts and non-experts gain insights into EAM works. The EASY Toolbox utilizes MIR techniques that focus on EAM – music that often emphasizes timbre rather than traditional musical parameters such as pitch, melody, harmony, and rhythm. The EASY Toolbox will be used as a starting point in developing and implementing the visualization and exploration module of EAMM.

Another project led by Park in collaboration with the 2CINEM is called Citrhythm, which is a large-scale project divided into a number of iterations. The current iteration focuses on acoustic ecology, visualization, and sonification. Subsequent investigations will include other sensory data such as heat, brightness, wind speed, and humidity. For the first iteration, we are focusing on a subsection of the city of Atlanta to render time-varying, non-intrusive acoustic maps that are scale accurate, topologically oriented, and dynamic. The main goals are to (1) investigate potential avenues for capturing the flow of crowds, machines, and ambient noise; (2) automatically capture and measure mood/emotion in public spaces through analysis/pattern recognition of noise and text analytics; (3) provide clues to waves of contemplation and response in public spaces, including galleries and museums; (4) expose acoustically relevant traffic patterns; (5) provide hints into the invisible dynamics of conferences and physical spaces; and address a wider issue of urban ontology via quantitative data acquisition of dynamic and non-octave stimuli.

Other on-going research projects include Feature Modulation Synthesis (FMS), a sound synthesis method informed by sound analysis [3]. FMS methods extract salient timbral features directly from a sound object and re-synthesize an altered sound object according to various modulations of timbral features. The FMS system performs salient feature extraction on sound objects, which is followed by feature modulation and synthesis of a new sound object based on the modulated features. The goal of FMS is to synthesize a sound object by modulating only specific timbral dimensions while ideally leaving all other timbral vectors intact.

Robert Scott Thompson is engaged with compositional activities pertaining to electro-acoustic and computer music that encompass several overlapping research areas including software synthesis, compositional algorithms, sound spatialization, musical signal processing and real-time interactivity. Current commissions include works for soloists and electro-acoustics and as well as chamber ensembles combined with fixed media and real-time signal processing and synthesis.

4. CONCERTS, EVENTS, AND OTHER ACTIVITIES

4.1. Concert Production

The GSU music technology program produced and collaborated in a number of concerts during the 2011-2012 academic year including the first computer music concert at DAEL which featured five student musicate concrete works as well as live performance works by guest composers Jon Appleton and Paul Botelho. Several pieces on the concert emphasized social issues concerning capital punishment and the recent execution of Troy Davis as well as public reactions to the controversial book “Cold Mountain” by Charles Frazier. Thompson’s composition Passage (for clarinet and electro-acoustics) was presented at the 2012 SEAMUS National Conference at Lawrence University in Appleton, Wisconsin. The work was also presented in Ljubljana, Slovenia in 2011 by the Society for Slovenian Composers. Thompson presented the fixed media work Out of the Vivaldi Air (for 8-channel
fixed media) during the 2011 ICMM in Huddersfield, UK. Park presented #1 for trumpet and electronics and gave a lecture on his current research at the 2011 EarZoom conference in Ljubljana, Slovenia. His compositional works have also been presented at Sonic Screens in Milan, Italy; Clarke Recital Hall, at the University of Miami; SEAMUS 2011; and Northeastern State University Faculty Trumpet recital in 2011. Park presented his paper at the 2011 ICMM in Huddersfield, UK and gave a workshop on sonification and visualization using Matlab while serving as the Workshops Chair for the 2012 International Conference on Auditory Display (ICAD).

Also, composer Elaine Lillios from Bowling Green State University visited GSU during her 2011 tour of the southeast and gave a workshop on jazz composition to GSU students. Atlanta composer Jason Freeman visited as well and presented his recent work to students during our 2012 spring semester.

4.3. Other Activities

Park recently completed an encyclopedia entry in collaboration with Roger Dannenberg from Carnegie Mellon University. The entry entitled Music and the Machine is part of Music in American Life: An Encyclopedia of the Songs, Styles, Stars, and Stories that Shaped our Culture, which will be published by ABC-CLIO at the end of 2012.

Thompson is also currently writing a book for AR Editions Digital Music Series on the topic of musical signal processing and sonic design utilizing software synthesis techniques within real-time contexts. This book blends a pedagogical approach with a fundamental research orientation.

Thompson’s most recent recordings include Orbital Lullaby (created in collaboration with School of Art and Design colleague Craig Dongoski) which focuses on the integration of fragmentary text and found sounds within a soundscape/acoustic ecological framework. A current project is the follow-up to this work which is made entirely of source recordings from the local environment building upon the collaboration with Ouse Dire (France) during their residency in the School of Music and School of Art and Design in 2003 as part of the Pulse Field International Exhibition of Sound Art.

Two other discs from 2011, Folio Volume One and Folio Volume Two, include works composed using interactive performance systems currently under development in Max and Pd. Several new recordings are scheduled for release in 2012. Slovenian clarinetist, Tadej Ketiq, will record Thompson’s Canto (de Las Sombras) for his own next solo CD release.

5. FUTURE PLANS

The music technology program is currently drafting a proposal for a Ph.D. in computer music and investigating potential partnerships with other departments from the GSU community. The multidisciplinary doctoral curriculum will focus on both musical and technical aspects related to computer music including developing competency in programming languages, HCI, digital signal processing, musicianship, analysis, EAM repertoire, and composition.

In December 2012, the current 2CINEM research lab will be expanded to a 7,000 square foot interdisciplinary space for collaboration and research which will include a professional multi-channel EAM studio, a computer lab, and video lab facilities. The music technology program and 2CINEM are also planning to actively explore project engagement opportunities that are multidisciplinary and interdisciplinary in nature. The Citygram project is such an example where future iterations will focus on other non-ocular energy measurements to visualize, map, sonify, and gain insights about our environment through invisible energies. Other plans include extending, updating, and releasing FMS and EASY Toolbox 1.0, which will be freely available for download in the near future.

6. CONCLUSION

The music technology program at GSU is currently experiencing a period of substantial growth. Many factors, including hard work by faculty and staff have contributed to this positive momentum including the establishment of the Center for Audio Recording Arts (CARA) in 1996, building professional studios and performance spaces, and recruiting new students and faculty in 2011. With the establishment of 2CINEM, the School of Music and music technology programs are further expanding their collaborative efforts and investment in digital arts and new media within a collaborative and interdisciplinary framework.

7. REFERENCES


CENTER FOR COMPUTER MUSIC AT THE COLLEGE-CONSERVATORY OF MUSIC, UNIVERSITY OF CINCINNATI

Mara Helmuth
College-Conservatory of Music
University of Cincinnati
Cincinnati, OH 45221-0003

Joel Matthys
College-Conservatory of Music
University of Cincinnati
Cincinnati, OH 45221-0003

Paul Schuette
College-Conservatory of Music
University of Cincinnati
Cincinnati, OH 45221-0003

Sangbong Nam
College-Conservatory of Music
University of Cincinnati
Cincinnati, OH 45221-0003

Benny Martinson
College-Conservatory of Music
University of Cincinnati
Cincinnati, OH 45221-0003

ABSTRACT

Recent research, music and activities at CCM (the College-Conservatory of Music) Center for Computer Music, or (ccm)2, the computer music studios at the University of Cincinnati will be discussed, and presented in a multimedia format including excerpts of recent works. A new semester-based set of courses in computer music will replace the old quarter-based curriculum in the fall of 2012. In research, collaborative projects in wireless sensor networks for music between the studio and the School of Engineering has resulted in a number of projects and performances using infra-red, accelerometer and other sensor networks. Also, applications have been developed for mobile devices and for interactive performances. A new student laptop ensemble, Ciclop, and participation on concerts by performance faculty, guest composers, and students enhance the Sonic Explorations series. Faculty and students are active in performing new works with inventive technologies at CCM and elsewhere.

1. INTRODUCTION

The most significant recent developments at the studio have been curriculum modifications, an increased interaction between the computer music studio and other departments of CCM, an even more lively level of performance activity by faculty and students, including the new laptop ensemble, and exploration of new areas of experimentation including sensor research, special hardware configurations and mobile device applications. The facilities have been described previously [1, [2]. Mara Helmuth has been the studio director since 1995, and the graduate assistants are Joel Matthys, Paul Schuette and Sangbong Nam.

2. CURRICULUM

The change-over to semesters, and recent developments in the field required re-thinking the courses available in computer music. Because semesters afford more time to dig into topics, the new courses are designed to be more independent modules. Currently the courses are primarily taken by composition students, but occasionally students in performance, electronic media, jazz studies, art, and engineering participate, which facilitates collaboration. Courses currently include Introduction to Electronic Music, Electronic Music Techniques, Computer Music Composition, Interactive Music, Live Electronic Music, Timbre Studies, Laptop Performance, Computer Music Projects, Music Programming Projects, Listening Strategies and Practices, Sound, Music and Science. These courses should both give conservatory students opportunities to master the latest techniques in composition and performance, as well as provide opportunities for collaboration, exchange of ideas and perspectives, and experimentation in related areas for students in CCM and other colleges. The new courses have a greater emphasis on interactive music than in the past.

3. RESEARCH

3.1. Wireless Sensor Networks

The extension of the wireless sensor network and music projects was aided by Mara Helmuth’s sabbatical in...
winter and spring 2011, as well as her UC University Research Council Grant, which provided support for supplies and a research assistant to program and construct a hardware configuration for a new project. Jung Hyun (Peter) Jun, with Talmai Oliveira and Pandit Vaibhav, graduate students of Dr. Dhrama Agrawal in engineering, assisted with this project. A percussionist’s movements are tracked with accelerometer sensors, providing control data for digital signal processing of the percussion sound in MaxMSP.

CCM D.M.A. student Joel Matthys has also been active with research on wireless sensor networks using tinyOS and MSP430 Timex which pass sensor data to Pure Data or MaxMSP. With Jun, Matthys is co-author of the paper “Wireless Sensor Network that Changes the Way of Playing an Ordinary Yo-yo,” presented at the IPSN (Information Processing in Sensor Networks) conference in Beijing, China, 2012.

3.2. Mobile Device Applications

Several (CCM) students have been active in developing applications for mobile devices. Benny Martinson’s free iPhone application Minimalicious is available from the Apple Market. This is a pattern-detecting musical instrument, designed to create minimalist music on the fly. After playing the same pattern twice in a row, Minimalicious keeps the pattern going, allowing the user to set up complex, interlocking rhythms. The user can kill patterns or shift the harmony with the swipe of a finger.

Joel Matthys adapted his composition “The Unicorn,” created with RTcmix, into a free multimedia Android application, available from the Android Market using pd.

```
public void mousePressed(){
    if (mouseX >= 56 && mouseY >= 56 && mouseX <= width-56 && mouseY <=height-56){
        user = true;
        playing = true;
        if (playing) time = null;
        initializeValues();
    }
}
```

**Figure 2.** Code fragment from Matthys’s Unicorn app.

Mara Helmuth has taught iPhone programming and experimented with RTcmix, and created iPhone apps for performance of her composition “Butterfly Mirrors.”

3.3. Hardware Hacking

Paul Schuette specializes in interactive electroacoustic works, audio sculpture, circuit building, hardware hacking and video. A collection of Schuette’s “speaker mobiles” ala Alexander Calder (each accompanied by a homemade synthesizer) will be featured this spring at the art gallery Semantics in Cincinnati. He leads a workshop for a select group of students interested in building analog synthesizers and using Arduino boards for interactive music making and realizing sound installations.

Hemipherical speakers based on Ico Bukvic’s modification of the Stanford and Princeton designs, were constructed by engineering students in a Computer Music Composition course several years ago. These speakers have been used frequently in laptop improvisations.

3.4. Intercultural Exploration

The studio has diverse group of students from the U.S., Korea, Japan and other countries. Mara Helmuth’s works have been influenced by travel in Asia, which resulted in the Hidden Mountain installation, a piece for pipa, ensemble and electronics and a series of interactive improvisations with the qin. Her recent sabbatical travel in Uganda laid the groundwork new installations, after recording wildlife on safari, and traditional Ugandan music. Her blog of the trip is found at http://www.marahelmuth.wordpress.com. She also brought some technological developments to the attention of several groups of musicians in Uganda, which will result in future collaborative work.

Graduate student Sangbong Nam’s two works “Awaken” and “Awaken 2,” were based on an exploration of Buddhist ideas and sounds from a Korean Buddhist Temple.

4. PERFORMANCE

4.1. Sonic Explorations

The Sonic Explorations concert series features works created in the (CCM) studios, as well as those of guest composers and performers. Recent guests include Scott Wyatt, Akira Takasaki, Brad Garton, The Tornado Project and Annea Lockwood. Lockwood’s visit inspired a student's ChucK performance of her piece Burning. Featured performers have been Mary Stucky and Timothy Northcut. The contributions of student composers and performers in works for fixed media, interactive performance with instruments, and multimedia have been noteworthy.

4.2. Conference/Festival Participation

Three (CCM) students, Benny Martinson, Joel Matthys, and Sangbong Nam, presented pieces at the SEAMUS 2011 conference in Appleton, WI, February 9-12, 2012. Benny Martinson’s piece “One,” interacts with a harmonic overtone singer, creating a simple, meditative texture out of the singer’s vocalizations. Joel Matthys’ “Shear Pin” is a fixed-format algorithmic composition created in the ChucK programming language. Sangbong Nam’s “Awaken 2” was commissioned as part of his second place ASCAP award at SEAMUS 2011.

Mara Helmuth’s “Expanding Space” for tuba and computer was performed at the Studio 300 Digital Art and Media Festival at Transylvania University in Lexington, KY. CICLOP, the Cincinnati Composers Laptop Orchestra Project, performed at the festival, performing two works by Joel Matthys.

Sangbong Nam’s piece “Awaken” was featured at the Midwest Composers Symposium, November 11-12, 2011, at the Jacobs School of Music at Indiana University in Bloomington, Indiana. “Awaken” will be released on CD by Ab-Ulre Records in spring 2012.

4.3. Other Concerts and Performances

Mara Helmuth’s composition Levonspace was performed on the Seoul International Computer Music Festival, and her clarinetist Rebecca Dunard’s “Water Birds” was performed at the Beijing Central Conservatory’s MusicAcoustica Festival in 2011. Her “Butterfly Mirrors” was performed during a residency with Esther Lamnec’s New York University New Music Ensemble, using two iPads running iRtcmix, with video and spectral delays of the ensemble performing a structured improvisation. Helmuth’s music was featured on the Ball State New Music Festical in March, 2012, with performances of a new work for choir and electronics, “Heart Sutra,” “Explanding Space,” “Butterfly Within,” and two Helmuth/Allen Ote collaborations, “No. 7 for Gyil and computer” and “Move!” for percussion and “tape.” “Water Birds” was performed at the New Interfaces for Musical Expression Conference 2012 at the University of Michigan, in Ann Arbor, MI, in May, 2012.

Paul Schuette and Kazuaki Shiotani presented works at the Contemporary Dance Theater in Cincinnati, OH, on October 21 and 22, 2011. Paul uses the unpredictability of mobile-making to create unique physical and sonic combinations. Kazuaki returned from Japan with his wife, dancer Karen Wissel Shiotani, to present Kazuaki’s composition “Iroha,” created with his interactive system Human Wind Chimes, which uses computer vision to sonify the motions of a live dancer. This work was also presented at the Ball State New Music Festival. His composition “Zen,” which also uses the Human Wind Chimes system, was performed at the Mississippi Museum of Art on June 26, 2011.

Benny Martinson and Joel Matthys have been commissioned by Vox Musica (Sacramento, CA) to compose new choral works for women’s choir and electronics. The pieces were premiered on March 27, 2012 in California.

The newly-formed ensemble CICLOP, the Cincinnati Composers Laptop Orchestra Project, is led by Joel Matthys and performed on the Sonic Explorations series, Classical Revolutions at Northside Tavern in Cincinnati, OH, on the Studio 300 Digital Art and Media festival at
Transylvania University in Lexington, KY, and at the Contemporary Arts Center in Cincinnati, OH. CicloP undertook a tour on February 6-8, 2012, performing at Butler University (Indianapolis, IN), the University of Illinois at Chicago, and the University of Wisconsin-Milwaukee.

5. CicloP

5.1. History and Membership

The Cincinnati Composers Laptop Orchestra Project (CicloP) was founded at CCM in February 2011. The twelve members of CicloP are composers and computer musicians ranging from the undergraduate level to faculty. Several have played in other laptop ensembles, including MiLO (Milwaukee Laptop Orchestra) and LOl (Laptop Orchestra of Louisville), while others are newcomers to live computer music. The group rehearses weekly under the direction of doctoral student Joel Matthews.

Figure 2 CicloP performing on their Feb., 2012 tour.

5.2. Repertoire and Tour

In February 2012 the ensemble embarked on a performing tour to Butler University (Indianapolis, IN), the University of Illinois at Chicago, and the University of Wisconsin-Milwaukee. The group performed works composed by members of the ensemble, improvisations, multimedia pieces, and existing works written for other ensembles. The group performed a concert-length performance of Gavin Bryar's "The Sinking of the Titanic" several times in spring, 2012.

5.3. Mission Statement

There is an inherent difficulty with laptop performance. The computer screen forms a virtual barrier between the performer and the audience, and the physical act of playing the laptop provides very little of visual interest to the viewers. It's hard to dismiss that sneaking question, are they just checking their emails up there? (I assure you we're not.)

"We have resolved to seek ways of overcoming these shortcomings, through multimedia works, pieces that incorporate physical sensors and networked mobile devices, and novel seating arrangements to break down the barrier between the performers and the audience, while also exploring methods of performing and improvising with musicality and taste. "CicloP's pieces are written in Pd-extended, ChucK, and Processing, powerful software that is also free and open-source. That's free as in free speech, and free jazz."

6. References

The studio website has moved from the old "meowing" site to: http://www.ucm.edu/computermusic.

7. References


Figure 1 Web interface of Probado Music with various search masks.

When first accessing Probado Music, several masks for the formulation of queries are offered to the user (see Figure 1). Besides metadata based search, Probado Music includes content-based search mechanisms. For each modality (lyrics, score, and audio), the system implements according MIR-techniques to search through all documents of that modality. Therefore, the user can also use lyrics to search for a piece of music. Furthermore, a score

lack the capability of directly accessing the match positions within the documents.

The project Probado aims at developing prototypes for enhanced digital library systems for non-textual documents that eliminate the mentioned shortcomings. As two examples of non-textual document types, (architectural) 3D models and music documents are considered. In Probado Music sophisticated user interfaces and content-based retrieval techniques enable online access to large digital music libraries. As a result of our research and development efforts, the Probado Music prototype is now made available to the public at:

http://www-mindb.iai.uni-bonn.de/probado

Furthermore, we collected a large corpus of public domain music documents from various sources and prepared it for presentation with our library system.

The remainder of this paper is organized as follows. In Section 2 we present details on the user interface and in Section 3 we describe the preprocessing workflow for music collections as well as the administration system Macao. In Section 4 the public domain music collection created and managed by our research group is introduced. We conclude the paper with an outlook on future work.

2. Probado Music Frontend

Probado Music: A Multimodal Online Music Library

Verena Thomas, 1 David Damrau, 1, 2 Christian Freymery, 1 Michael Claussen, 1 Frank Kargt, 3 and Meinard Muller 1

1 Computer Science III, University of Bonn, Germany
2 Fraunhofer Institute for Communication, Information Processing and Ergonomics (FKIE), Germany
3 Saarland University and MPI Informatic, Germany

ABSTRACT

After several years of research and development, Probado Music—a multimodal digital music library system—is now made available for the public. To allow access to anyone from anywhere, we have prepared a collection of public domain music material that is accessible through our system. Besides streamlining and presenting digital music documents (scanned sheet music, audio recordings, and lyrics), Probado Music employs current techniques from the field of music information retrieval to offer enhanced browsing, navigation, and search functionalities. We strongly believe that such novel library systems will appeal to music-lovers and can support musicians, musicologists, and music teachers in their work.

1. Introduction

More and more music archives and libraries pursue the digitization of their collections. One reason for these activities is long-term preservation. In addition, the digitization of music collections enables their computer-based remote access. Several institutions already offer online access to their collections (see, e.g., Petrucci Music Library, 1 Chopin Early Editions, 2 or the Neue Mozart Ausgaben). But the document presentation often lacks in user convenience. If for a piece of music several documents are available, the user has no possibility of easily and intuitively accessing them simultaneously. However, being able to listen to a recording while reading the score or quickly comparing two different interpretations of a piece would constitute great benefits. Another shortcoming of many online music libraries is the provided search functionalities. Frequently, only metadata search is available and therefore the user has to know the name of the sought-after piece of music. For digital music documents content-based query techniques can significantly simplify the search process. Some online-collections already support content-based search to some extent, e.g., the melody search of the Petrucci Music Library [9]. However, they

† Is now with Fraunhofer Institute for Communication, Information Processing and Ergonomics (FKIE), Germany.

‡ Is now with Steinberg Media Technologies GmbH, Germany.

1 http://www.nma.at
2 http://www-mmdb.iai.uni-bonn.de/probado
3 http://www-probado.de

http://www-probado.de
editor (Figure 2) allows for the formulation of symbolic queries. Audio matching techniques are available as well. But rather than free query formulation, the user can use extracts from the document collection for search. We will explain this type of query formulation later in this section. As last option, the user is offered a tree-based presentation of all pieces of music contained in the music collection.

Figure 2: Editor for symbolic score queries. The user can choose between a classic score view and a more technical piano roll visualization.

After starting a search (e.g., searching for the string “schöne Müllein” in the metadata), the hit list is presented to the user (see Figure 3a). In PROBADO Music, a piece of music-centered document access is pursued. Therefore, rather than listing all documents matching the current query, pieces of music are returned as hits. After selecting a result, all documents containing the according piece of music are made available for presentation. The current PROBADO Music prototype supports three document types—sheet music, audio, and lyrics—and offers visualizations for each of them (see Figure 3). After selecting a piece for visualization, a document of the according document type is opened in every view. However, the user can easily exchange the document selected for presentation through lists containing all sheet music versions and all recordings of the current piece of music respectively.

A further innovation of PROBADO Music are multimodal navigation functionalities through the inclusion of sheet music-audio synchronization techniques, see Figure 4. As one benefit, these techniques enable score following. While playing the audio, the currently audible measure is highlighted in the score. Another convenience introduced by sheet music-audio synchronization is score-based navigation. The user can freely browse through the currently loaded score book. Upon selecting a measure in the score, the audio recording will automatically jump to the according time position and playback will continue from there. In addition, the employed synchronization allows for keeping the musical position while exchanging the score or audio document selected for visualization. Thus, the user can quickly compare different recordings of a piece of music without repeatedly searching for the specific position he/she is interested in. Similarly, lyrics following and lyrics-based navigation are available.

In addition to the previously described search masks, the user can create content-based queries from within the visualized documents (see Figure 3c). In each view, the user can mark an arbitrary region. Due to the previously described synchronization, the user can then decide whether to use the matching score-, audio-, or lyrics-extract as query. Upon accessing the result of a content-based query, the exact match positions are visualized in the documents, Figure 5. The user can thereby quickly navigate through all matches and compare them.

Figure 3: The PROBADO Music user interface.

Figure 4: Sheet music-audio synchronization for the first measures from the third movement of Beethoven’s Piano Sonata No. 1. Regions in the score image are mapped to corresponding time intervals in an audio interpretation.

Figure 5: Hit visualization for an audio query consisting of the first 15 measures from the third movement of Beethoven’s Piano Sonata No. 17. The matching regions are highlighted both in the music documents and on the timeline below.

3. MACAO

To avoid digital graveyards, digital music collections need to be organized properly. Therefore, an entire process chain for digitizing, processing, organizing, annotating, and linking the data is required. In PROBADO Music such a workflow was defined and implemented through the administration system MACAO (“Music Administration for Content Analysis and Organization”). Given a collection of scanned sheet music pages and digitized CDs, the data is organized and prepared by abiding the following steps:

- Metadata: In cooperation with the Bavarian State Library (BSB), an entity-relationship model based on the FRBR model [5] was developed. Using this model the metadata information of the music collection is created. After this manual step, MACAO provides convenient input masks.
- Dissemination preparation: To enable streaming and presentation of music documents, derived file types need to be created (e.g., textures for the score visualization). In addition, several file types, only required for the subsequent preprocessing steps, are derived from the input data. Upon adding a CD or a score book to the collection these derived file formats are created completely automatically.
- Content extraction: Given scanned sheet music pages, their musical content has to be reconstructed using Optical Music Recognition techniques (OMR). The resulting symbolic score formats contain all music related information available on the scanned images. The lyrics of pieces containing voice parts are usually recognized by the OMR system as well. In PROBADO Music this information is used as the lyrics data presented to the user. Thus, the additional effort of finding and digitizing libretti can be avoided. For the upcoming music synchronization and indexing, score documents and audio files need to become comparable. Therefore, they are converted into a common midlevel feature representation. For the given data types and the intended MIR-tasks, chroma-features are a well suited representation [1, 4]. Their calculation can again be performed fully automatic and no user interaction is required.
- Segmentation and work identification: The content of a new music document has to be split up into individual segments, each associated to a single piece of music. Afterwards, the according metadata entries of the pieces of music have to be mapped to the segments. Automatic segmentation techniques, filters, and input masks support the user in accomplishing this task.
- Synchronization: Music synchronization techniques are employed to enable score-following and score-based navigation. Once the input data was correctly associated to the pieces of music the linking data is calculated without requiring further user interaction. For details on the employed synchronization methods, we refer to the literature [7]. Using the sheet music-audio synchronization results in combination with the lyrics extracted from the score scans, symbolic-score and lyrics-based navigation are quickly realized as well.
- Content-based indexing: The indexes for content-based search are calculated fully automatically. Again, we refer to the literature for information on content-based search techniques [2, 6].
- Revision: The employed synchronization method can produce erroneous linking structures which...
4. MUSIC COLLECTION

In the context of the PR0BADO project, the Bavarian State Library digitized an extract of their music collection. In total approximately 72,000 score pages and 800 commercial CDs were digitized. However, open access to this digitized copyrighted material cannot be granted by the BSB. Instead, a collection of public domain material was setup as proof of concept.8

The Multimedia Signal Processing Group in Bonn is now making an effort of providing a larger, free music collection that is accessible with the PR0BADO system and incorporates several data sources. We used exclusively public domain documents or material that is published under a Creative Commons Attribution License9 or comparable licenses. The documents were collected from the following sources:

- Isabella Stewart Gardner Museum, Boston10
- Mutopia Project11
- Petrucci Music Library
- Piano Society12
- Saurland Music Data (SMD)13
- Wikimedia Commons14

Currently, our music collection contains 2,492 pieces of music from 15 different composers. More details are given in Table 1. The collection can be accessed via the PR0BADO Music prototype.

5. OUTLOOK

The goal of PR0BADO Music is a holistic music experience where all documents related to a piece of music are made available simultaneously. Therefore, the extension of the system to provide access to other document types is architecturally considered and could be realized in the future. In a feasibility study we already showed the potential for adding music videos.8 Other imaginable document collection contains approx. 1,900 score pages and a total of approx. 31 hours of audio material.

Table 1: Content of the free music collection accessible with PR0BADO Music. The collection contains approx. 1,900 score pages and a total of approx. 31 hours of audio material.

<table>
<thead>
<tr>
<th>Composer</th>
<th>Pieces</th>
<th>Score pages</th>
<th>Tracks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bach, J. S.</td>
<td>14</td>
<td>61</td>
<td>4</td>
</tr>
<tr>
<td>Beethoven, L. van</td>
<td>55</td>
<td>511</td>
<td>76</td>
</tr>
<tr>
<td>Brahms, J.</td>
<td>12</td>
<td>97</td>
<td>16</td>
</tr>
<tr>
<td>Busoni, F.</td>
<td>1</td>
<td>34</td>
<td>1</td>
</tr>
<tr>
<td>Buxtehude, D.</td>
<td>1</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Chopin, F.</td>
<td>15</td>
<td>141</td>
<td>22</td>
</tr>
<tr>
<td>Elgar, E.</td>
<td>6</td>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td>Faure, G.</td>
<td>4</td>
<td>83</td>
<td>3</td>
</tr>
<tr>
<td>Franck, C.</td>
<td>4</td>
<td>43</td>
<td>8</td>
</tr>
<tr>
<td>Grieg, E.</td>
<td>3</td>
<td>54</td>
<td>1</td>
</tr>
<tr>
<td>Liszt, F.</td>
<td>11</td>
<td>78</td>
<td>1</td>
</tr>
<tr>
<td>Mozart, W. A.</td>
<td>30</td>
<td>215</td>
<td>29</td>
</tr>
<tr>
<td>Respighi, O.</td>
<td>3</td>
<td>33</td>
<td>3</td>
</tr>
<tr>
<td>Schubert, F.</td>
<td>44</td>
<td>120</td>
<td>6</td>
</tr>
<tr>
<td>Schumann, R.</td>
<td>49</td>
<td>141</td>
<td>89</td>
</tr>
</tbody>
</table>

Total: 249,564,275

REFERENCES


This paper presents how the modern game controller Wii-mote can be used to control Mozart’s dice music. The proposed method allows users to explore and expand their experience with endless music space within where note measures are arranged in a 2-layer structure that can be treated as a playlist. Experimental results show that the proposed organization of measures gives higher melodic similarity between consequent parts according to melodic distance measures such as distribution of pitch classes, distribution of intervals and distribution of note durations. The result is a system that allows user to create different music than in Mozart’s original Dice game and allows exploration of endless music. We also compare the use of different metrics for organizing individual layers in order to achieve melodically more meaningful music.

1. INTRODUCTION

Listening to the music is natural for all human beings. Everyone has a more or less unique musical taste. Not only do we enjoy listening to music, most of us are also unintentionally creating new music on our own, in our minds, usually expressed by humming or whistling. However, we are not composers, since music composition is mainly done by experienced and musically educated experts. New technologies, as well as input from musicians, in a form of collections of short musical pieces, such as measures or other short musical excerpts, make it possible for everyone to explore and expand their experience with music. One does not need an extensive knowledge of music theory or years of experiences in the field, to make music. It is possible to create new music just by playing a simple game with dedicated input devices such as the Wiimote. With such devices, creation of new music can be both easy and fun. This also shows the possibility of relating music to motion and interaction with visualization systems. In this paper we present an example of such an interactive system for music exploitation with modernization of Mozart’s dice music. In section 2 we present the research background and related work, in section 3 we describe our approach, in section 4 we present evaluation results and in section 5 we give conclusions and possible future work.

2. BACKGROUND

2.1. Related work

Music information retrieval

In recent years music and motion in relation with music information retrieval (MIR), is becoming a very active research field. Music and motion connects the fields of MIR, that offers solutions of musical problems, with computer vision, as well as the fields of musicology and choreology. Results of such cooperation and interdisciplinary research are many research publications in recent years.

2.1.1. Music information retrieval

A system for interactive multimedia performances with virtual musical instruments is presented in [7]. The system was used in several performances with virtual music instruments that can be played with gestures extracted from video. A crucial part of the system is a distributed multimedia server for multi-platform, multi-sensor integration. Authors have also presented demo applications that use face tracking for virtual instrument manipulation.

A good overview of gesture based music synthesis is presented in [10]. The paper explains basic terminology and gives comparison of different techniques used in gesture based music synthesis. Gestures are represented as analog input signals that can be analyzed either as a function of gesture (pressing a button, moving an object, etc.), or according to its physical properties (moving a hand to the left, sitting down, etc.). The possibility of converting analog signals to MIDI is presented, as well as some methods for transforming analog to digital signals. The paper also includes a comparison an analysis of different input controllers.

A musical instrument - SoundSaber - that can be used for prototyping motion capture based musical instruments,
by choosing sixteen measures for Minuet and another six
measures for Trio. Even though there are 176 pos-
ible measures for Minuet and 96 for Trio, Mozart’s instruc-
tions allow us to select from eleven measures for Min-
uet and six measures for Trio in each turn only. Matrices in Figure 2 specify the possible measures of each turn, which are selected by throwing dice. The correspondent measures to 1 and 2 in Figure 2 are shown in Figure 1. The matrix was computed so that consequent measures fit together according to melodic, rhythmic, and harmonic properties.

Figure 2. Original organization of measures for produc-
ing Minuet-Trios in Mozart’s Dice game. Rows represent the numbers on dice (in case of Minuets with two dice, in case of Trios one die). Columns represent the position in composition.

Our idea is to modernize the Dice game by using a Wiimote game controller to guide the selection of mea-
sures. The second goal is to automatically organize the measure matrix to best fit different melodic features. We also wanted to eliminate the rule of producing only Minuet-Trios and allow composition of endless music by using the Wiimote game controller. There are some known attempts to modernize the game of creating Minuet-Trios by clicking instead of throwing dice and can be found on-line.

3. OUR APPROACH

Our idea of modernizing Mozart’s Dice game is to use Mozart’s collection of measures for production of endless
music. While the original game has predefined sets of measures to select from at certain position, our idea was to make a space of measures where each pair of neigh-
boring measures have high level of similarity according to selected melodic features. One can move around in
this space by using the Wiimote. Wiimote’s built-in ac-
celerometers allows user to determine the orientation of the controller. The space can be defined as 2-layer torus structure. The first layer consists of measures from the Minuet collection and the second layer consists of mea-
sures from the Trio collection. Both are organized in the
same structure (11 × 16 measures for Minuet and 6 × 16 measures for Trio). The difference from Mozart’s origi-
nal Dice game is that the edges are connected as shown in Figure 3.

Figure 3. The structure of the Minuet layer in space. The original Mozarts organization (2D matrix) is projected onto a torus.

The main problem we needed to solve is how to ar-
range measures in space so that the neighboring parts have low distances as well as maintaining a low overall score (Equation 3). Neighboring parts are also consequent parts in a composition. We have to take into account that each position in such a space has 9 neighbors. One to each side in the same layer (that makes eight neighbors) and another one in the other collection of measures. We used a differ-
ent approach for selection of neighbors between layers, since one layer has 6 and the other has 11 rows. Mapping between layers is described as:

\[ Minuet\_Layer(x,y) \rightarrow Trio\_Layer(x,y) \]

\[ x \in [1,16], y \in [2,12] \]

\[ Trio\_Layer(x,y) \rightarrow Minuet\_Layer(x,y) \]

\[ x \in [1,16], y \in [1,6] \]

Organization of each individual layer was calculated sep-
arrately, since Minuet and Trio measures melodically sig-
ificantly differ. Next we present methods for arranging the measures of each layer.

3.1. Methodology

While the original structure of Mozart’s Dice game was put together manually, that consequent parts melodi-
cally fit together. Manually comparing all possible pairs in a collection of 172 measures is not only time-consuming but also difficult. If measures are randomly organized in space, this would lead to problems with pairs that have very low melodic similarity that is also the reason why such approach was not used in the original Dice game or-
ganization.

To calculate similarity between measures we calcu-
lated several different melodic features on symbolic rep-
resentations (MIDI) of measures. The features were calcu-
lated with the MIDI Toolbox [1] and are as follows: distribution of pitch classes, distribution of intervals and
distribution of note durations. MIDI Toolbox allows sim-
ple usage of these measures as distances between music pieces. Our goal was to achieve the lowest overall score s for each layer (Minuet and Trio), defined as:

\[ s = \sum_{i,j} distance\_measure(i,j) \]

\[ i, j \text{ neighboring measures in layer} \]

We used a simple algorithm to arrange measures in space. First, we read-in all the MIDI files and calculated the distance matrix for all pairs of measures according to distribution of pitch classes using the taxac norm met-
ric for the first 10 samples. The next step is to put-to-
gether the best organization of measures in individual lay-
ers. Because of the number of possible permutations is large, we used an approximate method to find a (subop-
timal) solution for organizing measures by shuffling mea-
sures in each individual layer, calculating the overall score (Equation 3) and selecting the permutation with the low-
est score. As shown by our results such approach resulted in a lower overall distance score than the original organi-
zation.

3.2. Mapping input

In our approach we use accelerometers of the Wiimote
controller as an input source for determining the direction of the next move in space. We chose the Wiimote since it is easy to use for tracking hand orientation. Wiimote can be connected to a computer via Bluetooth wireless connection and allows user to make natural gestures that can be tracked by computer. While the orientation of con-
troller around horizontal axes was used to move between measures in the current layer, the vertical orientation was used for switching between layers of measures (from Min-
uet layer to Trio and back).

The selection of rotation axes is done by a simple thresh-
olding algorithm that checks if acceleration along certain axes is greater than experimentally determined thresholds. In our case threshold values for rotation around horizontal axes are 0.5g (g - standard value of gravitational acceler-
ation), and 0.3g for vertical axes.

4. RESULTS

Our main goal was to allow users to create music with
seamless transitions between subsequent measures, which

and uses the Wiimote controller, is presented in [8]. The system can be used for high-fidelity motion capture.

A digital conductor system for controlling the virtual
orchestra is presented in [4]. The system detects sim-
ple conductor hand gestures that are used for control-
lng tempo and dynamics of a performance. The goal was to build an intuitive and easy to learn system which was later tested and evaluated with trials. The system inte-
grates several features of the art MIR algorithms for tempo adjustment, music stretching as well as video stretching and audio-video synchronization.

In [5], authors present a "FreeDigiter" framework for
mobile devices for fast recognition of finger gestures. Cus-
tom hardware was designed for finger gesture recognition and tested for fast digit input. The framework uses an infrared (IR) sensor in combination with accelerometers along two axes for tracking finger positions. The sys-
tem is very tolerant to different lighting conditions and therefore robust in different working environments. Even though the system was primarily developed for digit input, it could be adapted for other uses as well. A touch inter-
face that can replace classical vinyl plates for popular DJ
sketching is presented in [2]. The interface uses custom
made hardware that can be connected to MIDI input. Pre-
sented inputs can be combined with different visualiza-
tion techniques of music collections. Some techniques for
visualizing music collections are presented in [9], where
authors used different approaches such as disc separation, square separation and tree maps.

2.2. Mozart’s Dice game

By combining ideas of the described approaches, we
decided to make use of new technology and create an ap-
application for real-time composition of meaningful endless
classical music. The idea is in a way similar to musical
games called "Musikalische Würfelspiele", which were popular in 18th century Western Europe. The goal of such games was randomly generated music of the era and games, published in local newspapers, were meant for every day people. Publishers claimed that even amateur people could compose an infinite number of compositions on their own, by throw-
ing dice. The most famous example of such a game is
Mozart’s Dice game from 1787 [6]. A similar game was made by Kirnberger [3].

Figure 1. Two measures from Mozart’s Dice game col-
lection. The first and the second measure are numbered as 1 and 2 respectively in Figure 2.

In Mozart’s Dice game, one Minuet-Trio is composed

\[ \begin{bmatrix}
1 & 2 & \cdots & 11 & 12 \\
13 & 14 & \cdots & 22 & 23 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
21 & 22 & \cdots & 31 & 32 \\
23 & 24 & \cdots & 32 & 33
\end{bmatrix}
\]

\[ (a) \text{Minuet}\]

\[ (b) \text{Trio}\]

\[ s = \sum_{i,j} distance\_measure(i,j) \]

\[ i, j \text{ neighboring measures in layer} \]
we measure by the melodic distance between measures. A comparison of different melodic distance measures can be seen in Table 1. The table shows overall similarity scores (Equation 3) for different organizations of Minuet and Trio measures in each layer (a lower number means a lower overall score and thus better, more seamless result). The overall similarity is calculated by using different distance measures for all pairs of neighboring measures in same layer. For all distance measures we found an organization that has a significantly lower overall score than for average random organization - for distribution of note duration (DND) intervals (DI) and Distribution of note durations (DND).

Table 1. Table shows scores for different distance measures: Distribution of pitch classes (DPC), Distribution of intervals (DI) and Distribution of note durations (DND) for Minuet (M) and Trio (T) organizations for best found distribution (Best), average random distribution (AVG) and Mozart’s original organization.

<table>
<thead>
<tr>
<th>Score</th>
<th>M</th>
<th>T</th>
<th>M</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best</td>
<td>6.83</td>
<td>5.97</td>
<td>6.0</td>
<td>5.9</td>
</tr>
<tr>
<td>AVG</td>
<td>17.02</td>
<td>15.25</td>
<td>17.57</td>
<td>17.62</td>
</tr>
<tr>
<td>Moz1s</td>
<td>17.02</td>
<td>17.62</td>
<td>17.57</td>
<td>17.62</td>
</tr>
</tbody>
</table>

6. ACKNOWLEDGMENT

Authors would like to thank anonymous reviewers for their constructive comments, suggestions and for pointing out errors; they helped make this article better. The work was partly supported by the Slovenian Government-Funded R&D project EthoCatalogue: creating semantic descriptions of Slovene folk song and music.

7. REFERENCES


5. CONCLUSION

In this paper we presented a new approach that uses the collection of measures from the Mozarts Dice game for real-time creation of music with the Wiimote controller. We are no longer limited to production of Minuet-Trios only, but can compose endless music from a well organized collection, where selection of the next possible measure is guided by a similarity measure. This makes the transition between measures more seamless, while also giving the creator a chance of creating more diverse music than with the original game.

There are many possible extensions of our current solution. Instead of using only melodic features of symbolic data we could use audio samples as well, to create a more diverse organization of measures. Instead of using a fixed layer organization of measures, we could introduce a dynamic space for selection of the next measure, as well as

TOOLS AND ABSTRACTIONS FOR SWARM BASED MUSIC AND ART

Daniel Bisig, Philippe Kocher
Institute for Computermusic and Sound Technology
Zurich University of the Arts
philippe.kocher@zhdk.ch

daniel.bisig@zhdk.ch

ABSTRACT

A new version of our swarm simulation environment for musical and artistic applications is presented. The main improvements of this version concern the integration of an OSC based communication protocol and the addition of a graphical user interface. These extensions offer a variety of approaches for the configuration, manipulation and application of simulated swarms in real time. We hope, that these improvements open up the application of swarm simulations to a wider audience of artists and musicians.

1. INTRODUCTION

The swarm simulation tools and their application in musical and artistic creation and education have originated within the context of two research projects entitled “Interactive Swarm Orchestra” (ISO) and “Interactive Swarm Space” (ISS) that were conducted at the Institute for Computer Music and Sound Technology (ICST) of the Zurich University of the Arts (ZHdK). The projects have been motivated by the multitude of artistic and musical applications of swarm simulations such as [7, 8, 10, 11]. Due to the diversity and uniqueness of these approaches, the projects attempt to provide a systematic foundations for the integration of swarm simulations into creative processes in computer music and generative art. As part of this endeavor, we have explored a variety of strategies that emphasize the creative potential of capturing artistic ideas through highly customized swarm simulations rather than using standard swarms as black box mechanisms [2, 9]. Furthermore, we have developed software tools for the creation and control of simulated swarms [1, 3]. These tools try to strike a balance between a simple integration into existing musical and artistic software environments and a high degree of flexibility and customizability that characterizes scientific multi agent simulation software. Finally, we realized several prototypical works that include musical compositions, interactive installations and dance performances [4, 6, 5].

Recently, we have started to integrate the results of the two research projects into our pedagogical activities. For this, we have modified and extended the software tools to help students to quickly gain an appreciation and understanding for the creative capabilities of swarm simulations and to be able to gradually expand the artistic and technical sophistication and uniqueness of their ideas in order to realize entire works. This paper presents the current state and capabilities of the swarm simulations tools and discusses their application both in education and in our own artistic activities.

2. TECHNICAL DESCRIPTION

2.1. Simulation

The swarm simulation environment has started its existence as an open source C++ library that allows to develop a wide range of swarm behaviors and that provides a network communication protocol to control and retrieve simulation parameters in real-time [3]. The simulation library is highly generic in that it does not pre-specify the physical or biological meaning of any of the agents’ properties nor how these properties intertwine with each other via the agents’ behaviors. Neighborhood relationships among agents are handled via their properties, which are assigned positions within corresponding property spaces and its these properties that “perceive” each other depending on their distance and visibility constraints. There is no limit with respect to the number of swarms that can coexist and interact with each other. Simulations can be saved to or restored from XML formatted data files at any point during the running simulation.

2.2. Control and Communication

The swarm simulation environment provides an OSC based communication protocol. This feature permits the usage of any OSC capable software to create arbitrary swarm simulations without having to resort to C++ programming. It also allows to retrieve swarm simulation data to control audio or video generation processes. In order to trigger a simulation event, the OSC message format follows a syntax that employs the address part of the message to identify the type of event and the argument part of the message to supply the necessary parameters for executing that particular event.

Syntax: /Event_Type Parameter1...ParameterN

Example: /AddAgents “Boids” 5
To receive OSC messages from the swarm simulation, the desired agent properties have to be registered for communication. The format of the OSC messages sent by the swarm simulation is as follows: the address part of the OSC message contains the particular agent property whereas the arguments part contains the values.

Syntax: /SwarmName/Agent/Property/Name Values

Example: Boids/1/pos 0.1 0.4 0.3

2.3. GUI based Control

To provide a user friendly environment for the creation and control of customized swarms, we have developed a standalone application for Mac OS X that hides the intricacies of the OSC communication syntax. This software exposes a slightly reduced set of OSC commands via a collection of graphical controls (see Figure 1). The number and types of these controls correspond to the number and types of swarm properties and behaviors. When reseeding a swarm from a serialized state, the GUI modifies itself accordingly. And when manually changing, removing or adding control elements in the GUI, the corresponding properties and behaviors in the simulation are consequently changed, removed or added.

The standalone GUI application does an excellent job at hiding most of the details that are required to create a swarm simulation. In particular, it automates much of the basic swarm setup procedure such as adding frequently used agent properties and behaviors, or solving dependencies between behaviors, properties and neighborhood spaces when new behaviors are added or old ones are removed. On the other hand, this high level of abstraction limits the diversity of swarms that can be created much more than an OSC or C++ based approach.

It is important to stress the fact that these three different approaches are not mutually exclusive. It is very well possible to create a swarm simulation via C++, store the simulation in an XML file and reload this XML file into the GUI application. Or it is equally imaginable to configure a basic swarm simulation via GUI application and subsequently modify it via OSC commands. In order to promote a flexible transition between these different approaches, we have implemented a similar usage and configuration functionality and syntax for each of them. As an illustration, Figure 2 depicts an excerpt of the same steps necessary for configuring a simple swarm simulation. From top to bottom: C++ source code, OSC commands, GUI controls.

4. APPLICATION SCENAROS

The following section presents three different application scenarios in which the swarm simulation tools have been extensively used. These scenarios are a teaching situation, the creation of an audiovisual composition, and the realization of an interactive installation.

5. RESULTS AND CONCLUSIONS

The swarm simulation environment and its OSC based functionality have proven to be sufficiently robust and ver-

![Figure 1: A screenshot of the GUI application together with the visualization that is currently controlled by it.](image1)

![Figure 2: An excerpt of the steps necessary for configuring a simple swarm simulation. From top to bottom: C++ source code, OSC commands, GUI controls.](image2)

![Figure 3: World premiere of Trails at the British Film Institute in London.](image3)
satellite to allow its usage in education and artistic realizations. The standalone GUI application complements this environment in that it not only provides a gentle introduction into the usage of swarm based simulations for users that lack programming skills, but also offers a good starting point for any artistic realizations as it allows to quickly sketch and experiment with customized swarms. Only the realization of rather exotic swarm simulations required an implementation in C++. But as these new behaviors become part of the simulation library, the limitations of a purely OSC based approach gradually decrease. The software and documentation that can be accessed via the project’s website [1] keep up with these improvements.

Our teaching experience proved that the standalone GUI application is very helpful in conveying a practical understanding of the principles and capabilities of swarm simulations as it enables a hands-on approach where the students can immediately experience the effects upon changing parameters. Throughout the course, most of the students kept working solely with the standalone GUI application and did not consider to modify the simulations more thoroughly on a lower level of abstraction. They rather focused on the design of their audiovisual Max/MSP patches and would only return to experiment with the swarm simulation itself when their envisioned result could not be achieved by modifications to those patches alone. It remains to be seen whether a prolonged use of the simulation tools will lead the students to integrate swarm simulation into their works on a more fundamental level.

As for the authors themselves, the swarm simulation tools have proven to be extremely inspiring and useful both for the realization of musical and artistic works. The tools’ flexibility has allowed us to transfer a wide variety of artistic and musical ideas into swarm based approaches. Furthermore, their OSC based real time configuration and control capabilities has allowed us to creatively exploit the swarms’ high level of responsiveness both in the creation process and for the final performance and exhibition situations. In their current state, the simulation tools extensively support the manual design and refinement of swarm simulations and their communication with musical and visual processes. However they do not provide functionality for automated forms of configuration and modification via mechanisms of adaptation such as evolution or learning. Also, the tools fall somewhat short in the creation of simulations that gradually change over extended periods of time and thereby provide the opportunity to experiment with emergent macro scale structures in musical compositions and visual designs. Accordingly, we plan to include adaptive mechanisms in future versions of the simulation tools.

6. REFERENCES


SOURCE NODE: A NETWORK SOURCED APPROACH TO NETWORK MUSIC PERFORMANCE (NMP)

Robin Renwick
Sonic Arts Research Centre (SARC)
Queens University Belfast

ABSTRACT

This paper will seek to outline a Network Sourced Approach (NSA) to Network Music Performance (NMP). The NSA is governed through the software application SourceNode, which can be seen as a working example of a NMP enables. The core focus of the NSA is that a master node will ‘source’ audio content from designated slave nodes, which are synchronised together by the master node. The SourceNode project is a specific type of NMP: a nodal-based performance environment that has been network synchronised through time signature, start/stop, loop point and tempo control. The NSA is assembled in a ‘star’ formation. This paper addresses the characteristics of a star topology within a synchronised environment and describes a standalone software application implemented for this type of nodal-based performance. The study will attempt to outline the implications, advantages and issues faced when implementing such a nodal-based framework and also offer a formal example of this NMP structure in practice, through the SourceNode project.

I. INTRODUCTION

“It is possible to stop seeing music as singular, as a street between point a and point b, and to start seeing music as multiple, as landscape, as atmosphere, as an n-dimensional field of opportunities” [23].

Network Music Performance is a musical practice that has been enabled through, included with, or integrated by computer networking technologies, protocols, systems and topologies that have been incorporated into the design, implementation or architecture of either the musical system or sonic output. A Network Sourced Approach is a distinct form of Network Music Performance; an approach in which the core focus is the sense of ‘sourcing’ or ‘gathering’ sonic content from a delineated network. The sourcing behaviour, and its implications for NMP, may be seen as an extension of previous NMP approaches as outlined by Alain Renaud in the Frequencylator project [15] and Georgi Hadji in the Quintet.net project [10]. The central characteristic of such practice is that nodes within a network perform individual sonic content. The architecture establishes a framework that ensures this individual audio content is no longer mutually exclusive. The nodes are governed with respect to tempo, time signature, loop point and start/stop control. Indeed, they may now be seen as elements of a greater whole [10]. The NSA approach offers a topology that is extremely flexible, allowing for expansion or contraction of the framework without needing to change the network configuration. A master node can implement certain rules; governing the audio content that it receives but the number of streams, or nodes, from which it may source content is unequivocally open.

The SourceNode project is a specific type of NMP, enabled through the intricate use of current technologies derived simultaneously from the fields of music and computer networking technology. The project seeks to reformulate past conceptualisations of NMP models and architectures into a framework that is distinct, unique and evolving in both context and nature.

2. AN INTRODUCTION TO NETWORK MUSIC PERFORMANCE (NMP)

Network Music Performance has always stood at the intersection of technology and art and will continue to do so. Unique and interesting art forms have emerged from these systems; art forms that evolve as technology improves, continuing to ask questions of our modern understanding of the musical art form. An outlook which envisions ‘intermedia’ or ‘hybrid’ spaces being utilised by musicians standing at the cross section of networking, computer music and internet technologies is far from imaginary [19][25]. Similarly, collaborative spaces which harness the powers of synergy will always have networking technologies, architectures and frameworks at their centre [17][20][22][24].

Durin Barnev offers a very simple, yet detailed, explanation of what a network is; what is needed within a structure for it to be termed a networked organisation: “networks are comprised of three main elements: nodes, ties and flows. A
3. THE SOURCE NODE PROJECT

This section of the paper will outline the exact framework for the Source Node application, its motives, and an explanation of the software used in the implementation. The section will also discuss issues faced, such as MIDI-Clock ‘Drift’ as well as latency issues in using the JackTrip audio streaming software. A comparison will be made between the MIDI-Clock Drift apparent when using the Source Node application, with measurements of MIDI-Clock Drift when using the alternative Apple Network MIDI Utility; a software application built into the modern day Apple Operating System (OSX).

3.1. Architecture

The Source Node project is a specific type of NMP, based on a nodal framework; structured in a way that is congruous with a star-shaped network topology. A star shaped network is one that is centralised around a ‘hub’ [15][24]. The nodes in the network are connected to the central node, or ‘hub’. If the central node were to cease to exist, the network would no longer survive; the central node is sometimes called the ‘master’ node. This implementation of a centralised ‘hub’, or master node, draws welcome parallels with the pioneering NMP work of "The Hub" [8].

The Source Node project seeks to connect two nodes, designated ‘slave’ nodes with a third node, which acts as a master [14]. The software applications attempt to manage the ‘flow’ of audio between them, with the goal of enabling a synchronous, structured and musically coherent flow between two or more other points. A tie connects one node to another. Flows are what pass between nodes along ties” [1].

A diagram of a mesh network is shown below, highlighting nodes that are connected together along ties. The diagram displays a complex network system, perhaps one that is more complex than most NMP systems, but it is apt to highlight how a network comprises, fundamentally, of otherwise disparate nodes that are connected together.

Fig.1. Network Diagram [1]

Manuel Castells believes that society, organised into network structures may leverage their degree of influence through increased collaboration, information sharing and organisation [5]. In a musical context, the idea of networks becomes increasingly interesting. A musical system may be created with networked motives; seeking to leverage the contributions are ‘projected’ [14].

The Source Node project is a specific type of NMP, based on a nodal framework; structured in a way that is congruous with a star-shaped network topology. A star shaped network is one that is centralised around a ‘hub’ [15][24]. The nodes in the network are connected to the central node, or ‘hub’. If the central node were to cease to exist, the network would no longer survive; the central node is sometimes called the ‘master’ node. This implementation of a centralised ‘hub’, or master node, draws welcome parallels with the pioneering NMP work of “The Hub” [8].

Fig.2. A Star Shaped Network [15]

In this instance, the star shaped network will consist of three nodes: a ‘master’ node and two ‘slave’ nodes. The master node will share certain structural musical information with the slave nodes, such as tempo (MIDI-Clock), time signature information, loop points and start/stop information. This information, or data, will be created at the master node and then shared, through the network, to each of the slave nodes. This shared information will ensure that the three nodes are ‘synced’ together; allowing a musical performance inherently structured with, and through, the network to be created. The slave nodes perform their individual musical parts, working within the structures defined by the master node. The audio the slave nodes create will be transferred across the network to the master node. The streams of audio that the master node receives will be merged at this point. The master node will be in charge of ‘performing’ the piece of music; implementing an ‘interpretation, reaction and performance’ stage. In effect, this gives the NMP specific two-stage performance and modulation levels.

The first level is performed at the point of the slave nodes, where mutually independent offerings are created and then sent over the network to the master node. The second performance stage takes place at the master node. At the master node, the streams from each of the slave nodes are no longer mutually exclusive. The streams of audio are reinterpreted and combined into a coherent and musical whole.

It is important at this point to recognise a key facet of the architecture of this project: the star network, in this case, will be based in a Local Area Network (LAN) system. LAN’s are quite common within network structures that are based in a star formation [15].

3.2. Software Implementation

The Source Node project, fundamentally, consists of four core software enablers. The first is that each node will run a Digital Audio Workstation (DAW). This software will allow each node to perform their musical piece as well as allowing an interpretation of the networked structural data, such as tempo, time signature, loop points, etc. In this instance the DAW that is being used is Ableton LIVE (www.ableton.com).

The second level of software implementation is the Source Node application. The Source Node application was designed using Cycling 74’s MAX/MSP programming software (www.cycling74.com). The MAX/MSP ‘patch’ controls a number of key enablers within the project. Firstly, the relevant musical information created at the ‘master’ node, such as tempo, MIDI-Clock, time signature information, loop points etc. has to be interpreted and then transferred across the network. MAX/MSP allows a user to capture this information at one point. This data is then packed and transferred across the network. In this instance, the User Datagram Protocol (UDP) network protocol is used. This ensures that the data is transferred as quickly as possible to the slave nodes as is crucially important with respect to information such as MIDI-Clock.

The third level of software implementation is the JackOSX (www.jackosx.com) internal audio router. This piece of software allows for the internal transfer of audio within a computer. At the slave nodes, JackOSX is used to route the audio internally from Ableton LIVE to the JackTrip software. At the master node, JackOSX allows the user to route the separate slave node streams that are received by the JackTrip software to distinguish, user specified audio channels within Ableton LIVE. This feature is a crucial enabler for the successful performance of the Source Node project, as the master node can treat each slave node audio stream independently; allowing different performance processes to be integrated into each.

The fourth software stage is the JackTrip audio streaming software (www kode google com/j/ jacktrip), run in conjunction with the aforementioned JackOSX software.
The JackTrip software is in charge of transferring audio from the slave nodes to the master. The choice to use the JackTrip software was a relatively simple one. As the SourceNode project seeks to implement a type of Realtime Interactive Approach (RIA), it is imperative that the audio is transferred over the network as fast as possible and at least at ‘CD Quality’ audio standard [4]. JackTrip allows for such a transfer.

3.3. Network Sourced Approach

The SourceNode project is not inherently designed to replicate a traditional performance space, but more to utilise the network as a performance enabler. For further delineation of this, we may turn to Alain Renaud and his definition of the ‘Performance in Network’ (PIN) approach. Renaud describes the implications of such an architecture: “the performance becomes network mediated” [15]. The SourceNode project seeks to use the network in a certain way, both as a mediator and enabler. The slave nodes may be seen as being ‘in the network’, whereas the master node is said to be ‘using the network’; mainly as its source for music material. The audio that the master ‘sources’ is structured in a certain way, allowing a network sync to be created.

With this in mind, a better description of this type of framework is a ‘Networked Sourced Approach’ (NSA). A NSA ensures that the topology is configured in a manner which can be musically understood as a conductor, permitting multiple contributions to be synchronised. The network is used as a space in which information, or data, can be shared and utilised at two distinct levels. At one level, data is shared by the master node to the slave nodes who then interpret this data to create, what may be seen as, a ‘traditional’ musical structure. At the second level, information, this time audio, is shared over the network from the slave nodes to the master node. The master node uses this information as a musical ‘source’ from which it creates a coherent musical whole. The slave nodes act as musical objects, the master node as both conductor and performer. Most importantly, the network acts as the enabler.

3.4. Technological Implications - MIDI Drift

The key driver of the SourceNode project is the ability to transfer MIDI-Clock from the master node to the slave nodes as quickly and as accurately as possible. The successful transfer of accurate MIDI-Clock will define whether or not the SourceNode project is possible, and more importantly, if musical coherence is derived from the master node is musically coherent. An unsuccessful transfer of MIDI-Clock data will ensure that the two sources of audio that the master node receives become asynchronous; effecting the musical integrity of the NMP as a whole.

For the purposes of this paper, there exists two distinct ways to transfer MIDI information and data over a network available. The first being the utilisation of the Apple Network MIDI Utility, built into MacOS. The second was the use of the purpose built MAX/MSP software application, SourceNode. To outline a comparison, tests have been implemented to further highlight the merits of each, and this is where the concept of MIDI Drift is introduced. The term MIDI Drift is defined as: the rate at which errors occur in the MIDI-Clock network transfer, with respect to the late, early or asynchronous timing of MIDI-Clock messages, including the extremely disruptive non-arrival of MIDI-Clock messages. By comparing the drift rate, this paper can go some way to outlining the given success or failure of the SourceNode application.

3.4.1. MIDI Drift Measurement

To enable a comparison between the inbuilt Apple Network MIDI Utility and the SourceNode software application, a series of tests were run in order to gather certain measurements for MIDI Drift in both cases of software. It was found that while using the Apple Network MIDI Utility there existed a measurably more accurate and stable connection compared with the SourceNode architecture. However, it must also be stated that the inaccuracy of the SourceNode software was not at destabilising or disruptive levels.

Firstly, it will be beneficial to discuss the Apple Network MIDI Utility. It was found that the average drift over a series of three tests, each of 5 minute duration, was 0.51 milliseconds (ms). An example of the drift between the nodes is found in the illustration below, Fig. 6. The two tracks shown (green and purple) represent the MIDI-Clock at the two slave nodes. The drift measurements are taken to be the fluctuations between the two slave MIDI-Clocks, generated with respect to the MIDI-Clock received from the master node. The best case scenario of the three tests, offered an average of 0.36 milliseconds (ms) which is certainly low enough to be below the levels of human detection; highlighting the accuracy and validity of syncing through this method.

![Fig. 6. MIDI Drift with Apple Network Utility](image)

It would now be beneficial to look at the MIDI Drift measurements while using the SourceNode software application. The expected figures should be higher, due to the inherent and cumulative nature of using MAX/MSP software coupled with UDP and IP networking tools to ensure a network transfer of information. The key question is whether the drift is high enough to cause issues with the successful implementation of the NMP.

![Fig. 7. MIDI Drift with SourceNode Application](image)

Again, three tests, each of 5 minute duration were completed. The average drift figure over the three tests was 8.51 milliseconds (ms). This may at first glance, seem very high, especially when compared to the average while using the Apple Network MIDI Utility. It may be wise to put this drift figure into some sort of context, especially a musical one. Alexander Carot outlines the latency figures which are apparent when a piano player plays his instrument: “the time elapsed between pressing a key and the corresponding note onset is about 100ms for quiet notes and around 30ms for staccato, forte notes” [3]. Piano players have the ability to perform with these types of latencies apparent in their own musical system, so there is no reason to assume that time differentials of around 8.51ms cannot be dealt with by performers in the SourceNode project, even if the instruments that they are using are radically different from a traditional piano. Indeed the best test case of Drift while using the SourceNode system was shown to be 3.62 milliseconds.

A direct, visual comparison can now be made between the best case, as tested with the Apple Network MIDI Utility application, and the best case as tested with the SourceNode application.

![Fig. 8. Apple Network MIDI Utility - MIDI Drift](image)

![Fig. 9. SourceNode Software - MIDI Drift](image)

3.5. Technological Implications - Audio Streaming Latency

The second key driver with regards to the successful implementation of the SourceNode project is the streaming of the audio from the slave nodes to the master node. The Local Area Network (LAN) ensured that Network Latency, theoretically, was kept to a minimum: as delays due to the information having to physically travel long distances were not considered.

3.5.1 Audio Streaming Latency Measurement

The audio streaming within the SourceNode project was implemented through the use of two software enablers. The first software enabler was the JackOSX internal audio routing software system. This software allowed audio to be routed internally at both the master and slave nodes. At the slave nodes, audio had to be routed from the Ableton LIVE software into JackTrip. At the master node site, the audio had to be routed from the JackTrip software into Ableton LIVE, including a distinct separation of the audio streams, so that each slave ‘channel’ could be routed into separate

---

1 The master node created a MIDI-Clock source which controlled a metronome within the master node DAW (Ableton). The MIDI-Clock messages were transferred across the network to the slave nodes, where it generated a metronome signal within the respective DAW’s (Ableton). Both metronome signals were sent via MIDI to a third, recording DAW (PRO TOOLS). The audio streams were compared. Standard settings on all applications were used. Buffer sizes were as follows. Ableton: 512; MAX/MSP: 512; Pro Tools: 512. The MIDI Drift figure represents the compared time differences that exist at each slave node with respect to the synchronisation process. The time differences that exist can be seen as the fluctuations caused by the network and protocols in the transfer of the synchronisation MIDI-Clock messages from the master node each individual slave node.
audio channels within Ableton LIVE. This ensured an integration of a distinct two-stage modulation processes. The first modulation was at the slave nodes, the second at the master node. The second key software enabling is the JackTrip application. The JackTrip software enables the audio to be streamed from each of the slave nodes to the master node.

The tests found that the transfer of audio from slave node 1 to the master node experienced an average latency figure of 8.03 milliseconds (ms). The average latency for slave node 2, was 9.46 millisecond (ms)².

It must be recognised that the latency figures apparent in the streaming of the audio are due to the System Latency at each slave node. The System Latency will be the time taken for the internal software, in this case Ableton LIVE, JackOSX and JackTrip to process the functions that each of them are designated. Envisaging a SourceNode project within a Wide Area Network (WAN), where the streaming configuration covers a much larger geographical distance, the overall latency figures would probably be much greater. This is not to say that the System Latency figures that have been measured are not important, as it has been shown that the overall latency in any NMP system is the Network Latency added to the individual System Latency [2][3][4][5][7][9][11][12][13][15][16][22].

In the case of the SourceNode project the Network Latency is at minimal levels, due to the Local Area Network (LAN) infrastructure. It may be safe to assume that the latency figures, as measured, relate to the System Latency at each node.

4. CONCLUSION

This final section of the paper will discuss the outcomes of the research, focusing mainly on the aesthetic and technological results. There will also be a short discussion on possible future research with regards to both the SourceNode project architecture and the SourceNode software.

4.1. Aesthetic Summary

Aside from the motivation to create an architecture and enabling software technology to permit a Network Sourced Approach, there was a strong musical interest behind the SourceNode project. The project consisted of three participants, one positioned at each of the two slave nodes and one participant at the master. One slave node was designated the role of creating percussive elements and one given the role of melody. The master node decided the tempo throughout the performance, the time signature, if any loop constraints were to be imposed, and the start and end time of the piece. The master node DAW generated the MIDI-Clock messages that were shared through the network. Finally, the master node received the generated audio content from the slave nodes and added the important interpretation, modulation, reaction and performance stage.

The musical outcome was interesting on a number of levels. Firstly, the concept of DAW’s performing within a synchronised network structure was new to all participants. Although the concept was alien to the performers, the sync created by the network ensured that the architecture felt somewhat natural. Secondly, the network sync enabled a degree of freedom within the sonic content of the performance. The imposed time signature, tempo control and MIDI-Clock synchronisation meant that the musical output at the master node remained coherent, without a disconnect of musical ideas. Lastly, the added interpretation, performance and modulation stage at the master node ensured that the musical output of the performance remained fluid, exciting and above all original. Most interestingly, the participants at the slave nodes felt that the modulation at the master node effected and influenced the musical ideas at the slave nodes; creating a musically communicative feedback loop similar to those seen in other NMP performances [10][14].

4.2. Summary of Network Sourced Approach

"a composer can only claim to have created ‘fields of possibility’ whose content will be managed by others" [20].

The SourceNode project architecture has created an original and unique performance space. A space in which audio content is sourced, in synchronous fashion, from a network. The idea of the performer, positioned at the master node, in a multi-faceted orientation of designer, enable, controller and performer is key to any understanding of the architectures core implications.

The master node has democratised the creative musical process through the implementation of a network synchronised musical structure. The systems and technologies utilised may have created an example of ‘Liquid Modernity’ [27]. Bauman sees modern society as having evolved to include both decentralisation and centralisation in a fluid and flexible manner; believing that the decentralisation of certain functions within society is only possible through the centralisation of others. The SourceNode project may be seen in this light: where there exists a distinct decentralisation of creativity, through the centralised control of certain musical structures. The democratisation of both the content management and creativity in the project is principal to an understanding of the projects worth [21][25][26].

4.3. Technological Summary

The SourceNode software application facilitates the synchronisation of remotely located Digital Audio Workstations over a network. The sync created allows a distinct level of influence over disparate sequencers, enabling a star-shaped network formation to emerge. The SourceNode application achieves its desired goals, which is to create a standalone software application that enables a master node to share certain structural musical information, over a network to slave nodes.

The SourceNode application has been compared to the Apple Network MIDI Utility and the measured performance differences with respect to MIDI Drift have been noted. There has also been an investigation into the JackTrip audio streaming software, with measurements made of the Latency apparent within the SourceNode project architecture.

The paper has shown that a unique NMP offering is attainable through the implementation of the SourceNode software application. This architecture combines networking systems and protocols, computer music technology and NMP software audio streaming technology to create a NMP framework that incorporates synchronous performance systems and distinct two-stage modulation and performance levels.

4.4. Future Work

The SourceNode project has outlined a working, stable and unique NMP by offering a standalone application that may assist in the building of an architecture for Network Music Performances of a specific type. That is not to say it is complete. The SourceNode project has shown that it is possible to connect computers within a Local Area Network (LAN) and ensure that the software on these computers, namely the Digital Audio Workstations, remain synchronised. This sync is achieved through the implementation of UDP networking tools and MIDI-Clock synchronisation systems and software.

It must, however, be acknowledged that the software application is far from perfect. The implementation of an inbuilt MIDI-Clock accuracy alignment function would ensure that the SourceNode application could become a standard bearer for synchronous, remote collaboration through the use of DAW’s. With the advent of the computer evolving into the most powerful source of music creation, demand for a such a system: one that is accurate, stable and above all user-friendly, may increase exponentially; musicians who use computers as their primary instrument may seek out unique and novel ways of creating music within collaborative networked spaces.

The SourceNode project has shown that a nodal-based, Network Sourced Approach (NSA) encompassing two stage modulation and performance processes in a collaborative musical architecture, is achievable. As the demand for NMP systems increases, moving away from the scope of just a select few computer music and network technology specialists, we may see the number, form and context of
performances implementing a Network Sourced Approach (NSA) proliferate. These performances may cross geographical borders, musical genre categorisations, cultural boundaries and perhaps even more interestingly, musical and social subdivisions.

5. ACKNOWLEDGEMENTS

The research undertaken for this paper was completed at CIT, Cork School of Music within a thesis submitted by the author in partial requirement for an MSc in Music and Technology. This conference paper was prepared by the author while pursuing a PhD in Sonic Arts at SARC, Queens University Belfast.

6. REFERENCES


OSCTHULHU: APPLYING VIDEO GAME STATE-BASED SYNCHRONIZATION TO NETWORK COMPUTER MUSIC

Caris McKinney
Bournemouth University
Creative Technology Research Group
ccmckinney@bournemouth.ac.uk

ABSTRACT

In this paper we present a new control-data synchronization system for real-time network music performance named OSCthulhu. This system is inspired by the networking mechanisms found in multiplayer video games which represent data as a state that may be synchronized across several clients using a hub-based server. This paper demonstrates how previous musical networking systems predicted upon UDP transmission are unreliable on the open internet. Although UDP is preferable to TCP for transmitting musical gestures, we will show that it is not sufficient for transmitting control data reliably across consumer grade networks.

This paper also exhibits how state-synchronization techniques developed for multiplayer video games are aptly suited for network music environments. To illustrate this, a test was conducted that establishes the difference in divergence between two nodes using OsCroups, a popular networking application, versus two nodes using OSChulhu over a three minute time-span. The test results conclude that OSChulhu is 31% less divergent than OsCroups, with an average of 2% divergence. This paper concludes with a review of future work to be conducted.

1. INTRODUCTION

Computer network music has benefitted from three decades of development, including the experiments of the San Francisco Bay Area network band pioneers, the introduction of the OSC protocol [22], and research into streaming and latency issues. Making an infrastructure suitable for network performance in the face of highly distributed participation presents data as a state that may be synchronized across several clients using a hub-based server. This paper demonstrates how previous musical networking systems predicted upon UDP transmission are unreliable on the open internet. Although UDP is preferable to TCP for transmitting musical gestures, we will show that it is not sufficient for transmitting control data reliably across consumer grade networks.

Computer music programming languages such as SuperCollider and ChucK have led to new and exciting systems focused on code sharing in live coding environments. The League of Automatic Music Composers created interactive programs by directly soldering connections between computers and writing programs which would listen and transmit data on these lines. The network was fragile and error prone. It was also difficult to set up as all the connections had to be re-soldered each time the band rehearsed.

In what can be seen as a natural evolution of the technology, the spiritual successor to the League of Automatic Music Composers, the Hub, utilized a server-based system. This system provided a standardized interface for connections between members with varying computer models, as well as shared memory for the ensemble. Through out this time many other approaches to networking were being developed. Previous efforts mentioned focused on the real-time interaction of performers’ computers, but in the 1990’s several new methods explored non-real-time connections. Systems such as the ResRocket Surfer and Faust Music Online (FMOL) allowed users to collaborate writing music by providing an online repository.

Much research has been done towards investigating the issues that latency presents to instrumentists when streaming audio as well as strategies to cope with this latency in performance [1]. Often solutions favor research for grade connections between sites, providing lower latency, although these types of connections are not widely available to the public [20] [14]. Several alternative approaches have been taken to address network latency such as making the latency a multiplex of a preset tempo, as with NILJAM [4]. Local delay offsets are used in the eJAMMING software to produce the same latency of actual audio output between all users, attempting to bring the synchronicity of the users closer [7].

Recent developments in music programming languages such as SuperCollider and ChucK have led to new and exciting systems focused on code sharing in live coding ensembles. Systems such as Co-Audicle and Rubile create networks in which performers can share code that will be altered, executed and re-entered into the pool [19] [18]. Live coding systems are a fundamentally computer-based performance style and for that reason lends itself well to networking. The information being transferred is small, yet
can produce long-lasting results and the time specific parameters for execution are much less restrictive than in a traditional performance. The practice leads to organic interdependencies and produces layers of uncertainty which afford unexplored modes of performance, composition, and listening.

[6] Currently groups like the Hub, PLOK, Power brokers Unplugged and many others favor the Open Sound Control (OSC) protocol for networking data between nodes. OSC is a flexible protocol that is geared towards communication between musical systems. OSC provides a dynamic and powerful framework with high resolution data which is usable on local networks and across the internet [22]. In surveying the current solutions we feel that there is room for an alternative solution for networking compositions and performances, which implements techniques established by multiplayer video games.

2. OSCthulhu 1.0

Work on a new OSC-based platform for networking collaborative electronic music began in June 2010. The project, named OSCthulhu, was inspired by the program OSCgroups created by Ross Bencina, which enables users to share OSC messages with each other over a network [2]. OSCGroups accomplishes this by creating a central rendezvous server Address Translation (NAT) hole-punching techniques to enable individual users to bypass firewall and router restrictions normally placed on peer-to-peer communications.

2.1. Nat Hole-Punching and UDP multicasting

Normally, a router will block any message that is received unless a previous message has been sent out by the user to that specific IP address and port. This is done to prevent nefarious traffic from reaching the user’s private network and computer. Furthermore, the IP address and port of an application behind a user’s router is obsfuscated by Network Address Translation (NAT), a system utilized by work and computer. Furthermore, the IP address and port of each user that logs into a group. The server individually updates and multicasts that message to everyone of the recorded endpoints. Now when the user receives messages at their endpoint, be it from any of the routers as they have not been met with a matching out- received at either end will be discarded by their respective private IP endpoints of each person within the group. The strain on the server is also minimal, as it only serves as a rendezvous point for users, and none of the actual OSC messages are passed to the server. The system’s multi- casting architecture is quite appropriate for network music systems that require musically significant gestures to be shared over a network with the utmost speed and low overhead granted by UDP messaging systems, but does not require the reliability of a slower Transmission Control Protocol (TCP) based system [16].

2.2. Reality of the Internet: Packet Loss

After extensive usage of this system within the context of the authors’ network-based computer music band and Timothy Lich’s several showdowns of this system became apparent. While OSC’s streams are straightforward and robust, the usage of UDP meant that systems that relied upon expensive synchronization between player and server may suffer from potentially fatal errors during performance due to packet loss. Packet loss (when a packet is sent at one endpoint, but not received at the other) is an unfortunate reality of networking that every networked program must address in some way.

Many programs overcome this by utilizing TCP for reliability, which has huge overheads to handling packet loss, retransmitting lost packets after a certain timeout period [15]. However, for systems that require the utmost speed, such as gaming and musical applications, TCP is deemed inappropriate due to its sluggishness. Furthermore, the nature of TCP’s retransmission mechanism means that critical real-time gestures in a game or piece of music may be transmitted out of order, negatively impacting the quality of play. The TCP protocol is also dead if UDP is too unreliable for pieces which rely upon stringent ac- curacy.

2.3. Looking to Multiplayer Videogames

The similarities between multiplayer gaming and network-based computer music are significant. Both rely heavily on the ability to ensure that multiple peers can react to each other’s ac- tions as realistically as possible. In both, raw speed is considered more important than absolutely receiving every pixel or sound packet of information from the router, al- lowing for full bi-directional peer-to-peer communication between the users [8].

Once external communication has been established, the client application on each user’s computer opens up an internal UDP port that parses any incoming OSC mes- sages it receives and multicasts that message to everyone in that user’s group. Due to the flexible nature of OSC the origin of these messages could be from any application that has OSC capabilities, including programs such as Max/MSP, SuperCollider, or Reaktor [5] [11]. The benefit of this approach is that users can easily and dynamically form groups to share messages be- tween while being completely different places in the world, without having to note the individual public, private IP endpoints of each person within the group. The strain on the server is also minimal, as it only serves as a rendezvous point for users, and none of the actual OSC messages are passed to the server. The system’s multi- casting architecture is quite appropriate for network music systems that require musically significant gestures to be shared over a network with the utmost speed and low overhead granted by UDP messaging systems, but does not require the reliability of a slower Transmission Control Protocol (TCP) based system [16].

Given time for an alternative solution for networking compo-

2.4. A Musical Approach to a Gaming Model

OSCthulhu was created as a musical analogue to the GCSM approach. After testing several implementations of the GCSM as described by Sweeney, some tweaking was re- quired to produce a model that was appropriate for usage in the context of a network music environment. The core of OSCthulhu is the way it represents data, which is very similar in approach as the GCSM. Data is represented in the system as a series of networked entities called Syn- cArguments These Synchronous arguments are accompanied by modifiable values, called SyncArguments. SyncArgu- ments may be Strings, Integers, Floats, or Doubles. While in the original OSCSync that were accompanied by a fixed name, in OSCthulhu they are referred to by in- dex. This change was made to preserve bandwidth.

Another change that was made was the behavior of client actions and ticks. In OSCthulhu, when a client ac- tion is received it is immediately multicast to all of the clients instead of the server waiting for DeltaTime and is- suing a Tick to update the clients. This was done to make the system as fast as possible, though at the expense of more bandwidth. This is considered acceptable for mu- sical purposes, as the average network music server will deal with significantly less traffic than a gaming server, and thus can afford to deal with the overhead granted by UDP messaging systems, but does not require the reliability of a slower Transmission Control Protocol (TCP) based system [16].

One key point to keep in mind when using OSCthulhu is that it is fundamentally a different way of organizing the manner in which a networked composition or soft- ware system is constructed. Oftentimes we as composers think of addressing a network as a series of communication sec- tions, get louder, stop playing, switch timbres, etc. To network with OSCthulhu, a composer must think of his or her composition network in terms of a series of objects. These objects may be manipulated in similar fashion to the components of an object-oriented programming lan- guage. Objects may be created, destroyed, or have their values altered. So for example, instead of having a script that we gave commands to modify music, instead a composer would have an object that represents a synthesis unit gen- erator, including variables that represent that unit genera- tor’s amplitude and timbre. To create an object another instance of that unit generator, perhaps with a different set of argu- ments, one would simply add another instance of that ob- ject to OSCthulhu. This approach may require a bit more focus on the objects, but the current system lends itself well to network- ing musical contexts, especially in remotely rendered syn- thesis configurations.
3. DIVERGENCE TEST

3.1. Methodology

A test was conducted to demonstrate this effect. This test consisted of two nodes, one in London, England and the other in Boulder, Colorado, both using standard consumer level broadband networks, sending messages to each other. Standard broadband was chosen for this experiment as OSCthulhu has been designed specifically to facilitate network music performance in real world environments outside the confines of academic institutions with access to research networks. The results gathered in this experiment may differ on these academic research networks and future experiments are planned to investigate the differences.

These two nodes created and altered various data sets on their own systems, while simultaneously sending messages to each other to coordinate those same changes on the other node. There were three different actions a node could make: create an array (with a random number of indices, each containing a random value), alter an index of an array, or delete an array. These actions were chosen randomly, with index alterations occurring twenty times more often than creating or removing an array, to reflect real world scenarios. The test was conducted with four different send rates at which changes would occur and messages would be sent: every 250, 100, 25, and 12.5 milliseconds. These messages were sent over a period of two minutes, using either OscGroups or OSCthulhu on subsequent run-throughs for comparison.

A value called divergence was collected every 10 milliseconds for each run-through. This test defined divergence as a measurement of the difference between the two nodes’ states at any given moment in time. For example, if at a given moment node one contained four arrays, and node two contained five arrays, but four of those arrays were identical to those contained in node one, then the two systems would be considered 20% divergent. Figure 2 shows the results produced by OscGroups.

3.2. Results

The results show a staggering amount of divergence, with the systems immediately beginning at approximately 20% divergence, and becoming more divergent over time, settling at approximately 50%. This divergence can be accounted for by packet loss, lag time, and the cascading nature of divergence (i.e. if an array is missing on one node, the other node is not aware of this and will continue to attempt to set values in it. They will not realign until the second node serendipitously removes the array). Glitch Lich has personally encountered this divergence in performance, wherein a member at one node is creating sounds with a certain unit generator they have created, but the other nodes do not contain this unit generator, therefore the whole node’s performance is effectively non-existent.

Figure 3 shows the results for OSCthulhu. The results show a stark difference as the amount of divergence is pre-dominantly zero, with spikes up to 5-10%. There are two main reasons for this large difference in divergence between the two systems. Firstly, the effects of packet loss are drastically minimized, as the GCSM server synchronization cycle ensures that every cycle period (1000 milliseconds used for this test) the two nodes locked back in step (unless the synchronization packet itself is lost, which does happen on occasion). This prevents the cascading effects of divergence from taking hold, so differences do not pile upon each other over time. Secondly, differences due to lag time are also minimized, as all the actions are first sent to the central server which then simultaneously broadcasts the effects to both nodes. These nodes then receive the message and act upon it in a very similar time scale.

3.3. Benefits of Convergence

One manner in which networked ensembles may take advantage of this capability is through the usage of what may be called Remotely Rendered Synthesis. In many network music bands, including most of the work conducted by The Hub and PLOrk, network messages are transmitted among multiple participants to influence each other’s behavior. Each member then uses their own computer to output their own sounds. In comparison, in a Remotely Rendered Synthesis configuration, each of the participants share their sound synthesis descriptors with each other member beforehand (in the case of Glitch Lich, the SuperCollider SynthDefs are used), then, a Sound State is constructed that is mirrored on each participant’s own computer. This Sound State is similar to a Game State in Sweeny’s GCSM, except the data being synchronized represents the state of a sonic world instead of a virtual game world. OSCthulhu keeps track of the Sound State present on each user’s computer. Whenever a member makes a change to their particular version of the Sound State, this change is replicated on the server, and shared with the whole group. Then, each member’s computer outputs audio that contains the full sound present in the piece, including audio that is being produced by other members. This is useful for network music performances wherein all the members are not geographically co-located.
any open (non-password protected) servers. This would allow for spontaneous network collaborations and improvisations to occur, similar to how users join video game servers for spontaneous multiplayer games. Finally, convergence classes/libraries are planned to be constructed for SuperCollider, C++, Java, Processing, and Max/MSP that will take care of much of the boiler-plate code required to create an application that utilizes OSCthulhu.

5. CONCLUSION

The OSCthulhu synchronization system offers network music composers a new choice for enabling network-based compositions and performances. As the results show from the tests conducted, OSCthulhu can be more effective than oscGroups in certain scenarios for networked computer music. If a dislocated ensemble wishes to have the fluidity of UDP based networking while maintaining a sufficient level of reliability on the open internet, especially in cases of shared musical resources, then OSCthulhu proves to be a good choice to meet these demands.

6. REFERENCES


[23] The GRid-ENabled Deployment for Laptop orchestras (GRENDL) project [17] aims to eliminate many of the problems associated with the practicalities of the performance and organisation of pieces for laptop orchestrations.

REALTIME WEB TECHNOLOGIES IN THE NETWORKED PERFORMANCE ENVIRONMENT

Rob Canning
Department of Music
Goldsmiths, University of London, UK
rob@kiben.net

ABSTRACT

Current real-time web technologies are capable of providing composers of network based music with a new infrastructure for the distribution, control and synchronisation of the networked score. When these technologies are combined with an adherence to web standards it is possible to remove software and platform specific solutions. Cross-platform web browsers present the composer with a standardised and accessible environment where notational material can be presented to performers.

These technologies provide a useful set of tools and processes to facilitate networked performance. The use of standards compliant systems over proprietary ones promotes interoperability and future proofing and provides a platform for shared research. These systems can also reduce rehearsal/performance set-up time and complexities to a fraction of that currently experienced with more ad-hoc scenarios commonly implemented in this area of performance.

This paper argues for the feasibility of how various Web standards, including the Hypertext Transfer Protocol Revision 5 (HTTP/5), Scalable Vector Graphics (SVG) and the ECMAScript programming language can combine with real-time web server technologies, NodeJS and WebSockets in one possible workflow as implemented in the author’s Node-Score system.

1. BACKGROUND

The pervasiveness of network enabled mobile devices, in the western cultural context, from smart phones to tablets and laptops, alongside widespread high speed network access, in recent years has led to an increase in creative musical experiments using these technologies[13]. There has been an exponential growth of the laptop ensembles, many following on the model established by the Prince- ton Laptop Orchestra (PLOR)[17] followed now by the incorporation of the smart-phone in to the performance environment as with the Stanford Mobile Phone Orchestra (MoPhO). As with their predecessors (groups such as the League of Automatic Music Composers and The HUB)[13] these ensembles often create their own ad hoc strategies and softwares for interaction. L2ORK for example, have created their own fork of the Pure Data software as a platform for both message passing, sound processing/synthesization and networking.

The Laptop Orchestra model of treating the computer as a meta-instrument, utilising its abilities as sound-processor and input for Human Interface Devices (HIDs) has a very different set of demands than for that of the networked score. Common practice amongst composers working with networked score systems involve the use of a generalised audio oriented data-flow programming language, with plug-ins to deal with networking (often Open Sound Control (OSC)) and the presentation of notation (MaxScore, Java Music Specification Language (JMSL), OpenMusic or Lilypond based). Visual programming languages such as Pure Data or MaxMSP are the lingua franca of many computer musicians, and when it comes to finding strategies for the presentation of notational material to instrument musicians, composers will very often use these types of tools as they are familiar and flexible tools (for example in Gerhard Winkler’s Real-time scores[18] or Geoff Hjalt’s Quinetes.net[8]).

These systems can, and do work, but this kind of approach is hampered by problems. Multiple, non-standard installations of expensive platform specific softwares and associated plug-ins (often requiring expensive hardware), difficulties with networking: Network Address Translation (NAT) traversals and the circumvention of firewall restrictions, Network Time Protocol (NTP) synchronisation between clients as well as issues surrounding the incorporation of middleware to enable communications with some HIDs, are examples of some common problems. It suffices to say that many of these configurations are non-trivial to implement and require a technician with considerable understanding of both the specific software and network protocols being used. In this context modularisation is the best way forward and the separation of the score interface from tools designed for the creation of new electronic instruments is an important step.
2. WEB BROWSER AS PLATFORM FOR THE NETWORKED MUSIC SCORE

The ubiquity of web browsers and document formats has led to the development of web based applications (WebApps) that previously would have existed as platform specific, stand-alone applications. The WebApp boom has resulted in the development of a range of associated technologies, providing rich set of software tools that can be re-purposed for the creation of networked "screen scores"[11] or "net scores". The network is the pertinent feature, not the screen; the content pushed through the network is not limited to that which can be displayed, for example haptic or audio driven cueing systems can be imagined. The HTML5 standard presents a rich set of pre- sentational possibilities from plain text and image to high definition video, multi-channel audio, animated SVG and Web Graphics Library (WebGL) 3D environments. By combining these with CSS3 and ECMAScript a new level of control over and interaction with these media assets can be obtained. The standards compliant web browser is becoming a powerful cross platform, cross device, rich web client, and can harness this power without the need of third party plug-ins such as Shockwave, Flash or Java. The use of non-standards compliant technologies can result in a relatively short life span for platforms and compositions built upon them (Duckworth’s shockwave component to the web based Cathe dral composition which no longer functions in modern browsers [5]). The standards compliant browser provides an ideal platform to be distinct by a wide range of rich media HTML5 based net scores. Mike Solomon provided an example of this at ICMC2011 with his browser based score for Norman (age 1) [1] which utilises Javascript to animate an SVG created with the Lilypond engraving software.

A number of uses of the web browser have been explored by composers, for example as a playback/display mechanism in the work of Solomon’s work and as an interface to facilitate crowd sourcing for in Jason Freeman’s Graph Theory, composers have also explored their hyper-textual navigational systems to create browser based open form scores[6][13].

3. THE REAL-TIME WEB

In the mid-nineties Netscape Navigator introduced server-push technologies in two forms, a multi-part response MIME type called multipart/x-mixed-replace which can be used in Common Gateway Interface (CGI) scripts, and Java applets which allow a persistent connection via a handshake Mission Control Protocol (TCP) socket. Today, the most common web real-time features are incorporated in the web is by a collection of technologies working together; at the core are JavaScript and XML working to con- centrate the Document Object Model (DOM), this approach is know as Asynchronous JavaScript and XML (AJAX). AJAX programming is based on the real-time performance of the client, the client makes a requests that is answered when a new server event occurs, this closes the connection and trig- gers a new request and so on. This type of implementa- tion is known by a number of terms most commonly long polling or Comet programming. AJAX based long-polling and multipart responses are not true real-time, syn- chronous technologies, but are workarounds trying to cur- rent problems such as browser timeouts (in the multi- part response method), or trying to emulate a persist- sient server-client connection through asynchronous se- quences; they are also inefficient as each server event ne- cessitates a client-server TCP handshake, slowing down the interaction.

Two current technologies are available which make synchronous connections possible, Bidirectional-streams over Synchronous HTTP (BOSH) and WebSockets, thus to cat- later becoming incorporated in the HTML5 standard [10]. For the demands of the networked score a synchronous, bidirectional connection between the server and client is desirable. Bi-directional XML and HTML5 makes it a non-hierarchical network systems and distributed par- ticipatory style game pieces rather than the traditional conductor- musician hierarchy suggested by the unidirectional model, to encapsulate these types of models is important as they are at the center of research in the field of network perfor- mance.

For NodeScore (the networked score system in con- stant development by the author) the WebSockets protocol was chosen as the most efficient system with the greatest potential future though its incorporation in the HTML5 standard. The WebSockets Application Programming Interface (API) is a fully bidirectional, duplex protocol that communicates over a single TCP socket. It can be browser based and as such can use port 80, bi-passing firewall re- strictions imposed by some servers and clients on non port 80 TCP connections.

4. NETWORKED PERFORMANCE AND BIDIRECTIONAL SERVER INFRASTRUCTURE

NodeScore’s primary goal is the communication of musi- cal score material to, or between, human performers read- ing from displays connected over a network. It is a web based system but unlike much work [2][15][9] with web

http://nodescore.kiben.net

based Networked Music Performance (NMP) it does not attempt any synthesis, either distributed or centralised. NodeScore uses a server written using the NodeJS framework, a server-side environment for the creation of web applications, primarily web servers. It is built on Chrome’s JavaScript runtime (V8), the programs for NodeJS are writ- ten in JavaScript. NodeJS allows the use of a common lan- guage on both server and client sides of the application.

The combination of WebSockets, NodeJS, JavaScript and HTML5 is a flexible, standards compliant base on which to build net scores. The bi-directionality of the connections established between the multiple clients and the server allows inter-browser communications and con- trol. In the context of the net score this opens a number of possibilities; one browser can act as a control mechanism over other connected browsers in a traditional one to many relationship, whereas the server may act as an intermediary. Figure 1:

WebSockets

Figure 1: WebSockets

The ability to exert a level of control over the relation- ship between the material presented and the musicians (improvising or performing within polyvalent structures) gives the composer the opportunity to experiment with the network itself. To compose with network configura- tions makes it possible to explore how musical materials and performers can interact to yield novel results. This process allows reflection on the emergent properties of the network configuration employed and raises questions on how emergent properties may be identified and cate- gorised within these systems.

Most ensemble performance can be considered in the context of the network as an example of communication. In the mid-nineties Netscape Navigator introduced server-push technologies in two forms, a multi-part response MIME type called multipart/x-mixed-replace which can be used in Common Gateway Interface (CGI) scripts, and Java applets which allow a persistent connection via a handshake Mission Control Protocol (TCP) socket. Today, the most common web real-time features are incorporated in the web is by a collection of technologies working together; at the core are JavaScript and XML working to con- centrate the Document Object Model (DOM), this approach is know as Asynchronous JavaScript and XML (AJAX). AJAX programming is based on the real-time performance of the client, the client makes a requests that is answered when a new server event occurs, this closes the connection and trig- gers a new request and so on. This type of implementa- tion is known by a number of terms most commonly long polling or Comet programming. AJAX based long-polling and multipart responses are not true real-time, syn- chronous technologies, but are workarounds trying to cur- rent problems such as browser timeouts (in the multi- part response method), or trying to emulate a persist- sient server-client connection through asynchronous se- quences; they are also inefficient as each server event ne- cessitates a client-server TCP handshake, slowing down the interaction.

Two current technologies are available which make synchronous connections possible, Bidirectional-streams over Synchronous HTTP (BOSH) and WebSockets, thus to cat- later becoming incorporated in the HTML5 standard [10]. For the demands of the networked score a synchronous, bidirectional connection between the server and client is desirable. Bi-directional XML and HTML5 makes it a non-hierarchical network systems and distributed par- ticipatory style game pieces rather than the traditional conductor- musician hierarchy suggested by the unidirectional model, to encapsulate these types of models is important as they are at the center of research in the field of network perfor- mance.

For NodeScore (the networked score system in con- stant development by the author) the WebSockets protocol was chosen as the most efficient system with the greatest potential future though its incorporation in the HTML5 standard. The WebSockets Application Programming Interface (API) is a fully bidirectional, duplex protocol that communicates over a single TCP socket. It can be browser based and as such can use port 80, bi-passing firewall re- strictions imposed by some servers and clients on non port 80 TCP connections.

The ability to exert a level of control over the relation- ship between the material presented and the musicians (improvising or performing within polyvalent structures) gives the composer the opportunity to experiment with the network itself. To compose with network configura- tions makes it possible to explore how musical materials and performers can interact to yield novel results. This process allows reflection on the emergent properties of the network configuration employed and raises questions on how emergent properties may be identified and cate- gorised within these systems.

Most ensemble performance can be considered in the context of the network as an example of communication. In the mid-nineties Netscape Navigator introduced server-push technologies in two forms, a multi-part response MIME type called multipart/x-mixed-replace which can be used in Common Gateway Interface (CGI) scripts, and Java applets which allow a persistent connection via a handshake Mission Control Protocol (TCP) socket. Today, the most common web real-time features are incorporated in the web is by a collection of technologies working together; at the core are JavaScript and XML working to con- centrate the Document Object Model (DOM), this approach is know as Asynchronous JavaScript and XML (AJAX). AJAX programming is based on the real-time performance of the client, the client makes a requests that is answered when a new server event occurs, this closes the connection and trig- gers a new request and so on. This type of implementa- tion is known by a number of terms most commonly long polling or Comet programming. AJAX based long-polling and multipart responses are not true real-time, syn-chronous technologies, but are workarounds trying to cur- current problems such as browser timeouts (in the multi- part response method), or trying to emulate a persist- sient server-client connection through asynchronous se- quences; they are also inefficient as each server event ne- cessitates a client-server TCP handshake, slowing down the interaction.

Two current technologies are available which make synchronous connections possible, Bidirectional-streams over Synchronous HTTP (BOSH) and WebSockets, thus to cat- later becoming incorporated in the HTML5 standard [10]. For the demands of the networked score a synchronous, bidirectional connection between the server and client is desirable. Bi-directional XML and HTML5 makes it a non-hierarchical network systems and distributed par- ticipatory style game pieces rather than the traditional conductor- musician hierarchy suggested by the unidirectional model, to encapsulate these types of models is important as they are at the center of research in the field of network perfor- mance.

For NodeScore (the networked score system in con- stant development by the author) the WebSockets protocol was chosen as the most efficient system with the greatest potential future though its incorporation in the HTML5 standard. The WebSockets Application Programming Interface (API) is a fully bidirectional, duplex protocol that communicates over a single TCP socket. It can be browser based and as such can use port 80, bi-passing firewall re- strictions imposed by some servers and clients on non port 80 TCP connections.
allowing all performers involved to synchronise and send message to one another. The audio and video stream can be hosted on a separate layer, and need not be integrated into the net score’s server infrastructure as it is important to keep the system modular allowing different technologies to be mixed and matched as appropriate.

With real-time web processes a platform can be created to allow for a type of score that can be distributed, synchronised and controlled across a network in new and more democratic ways. Using standards compliant technologies to achieve this creates a shared platform for research and a free and accessible medium for its dissemination.

5. CONCLUSION

Network enabled scores need a unique set of tools and processes for both their creation and distribution. The real-time web combined with standards compliant document formats, provide a framework within which these tools and processes may be developed. These will be of use to both the laptop/mobile ensemble as well as the instrumental ensemble reading from the screen and will provide a flexible layer, working alongside sound processing and streaming technologies for the facilitation of telematic and LAN based networked performance.

The primary concern of NodeScore is to provide a framework for fast, low latency web-based rich media score delivery, control and synchronisation. This is achieved through the use of an infrastructure that aspires towards delivery systems: an exploration of web pull and push technologies, 1999. [Online]. Available: http://aisel.aisnet.org/cais/vol1/iss1/14

6. REFERENCES


SECURITY IN NETWORK CONNECTED PERFORMANCE ENVIRONMENTS

Scott Hewitt, Alexander Harker
CeReNeM
University of Huddersfield
Huddersfield, HD1 3DH, United Kingdom
scott@scotthewitt.co.uk ajharker@gmail.com

ABSTRACT

In this paper we highlight security issues generated by the use of network connectivity in performance. We argue that an awareness of these issues can lead to more secure and stable software, in both a technical and a musical sense. Potential exploits which might compromise performance integrity are illustrated along with suggestions for methods that alleviate such concerns.

1. WHY SECURITY?

1.1. Security: A Relevant Concern

Security as a concern within performance practice is rarely considered a priority, as the perceived threat level from third parties is low.1 However, trust of any apparent user input is crucial to a successful musical performance. It is our contention that, whilst the threat from malicious users is perhaps at present minimal, software for network performance is often needlessly compromised in terms of security, leaving it open to both malicious attack, and (more realistically) failure through unintentional behaviour. In the worst cases these risks can be catastrophic. The underlying problem is implicit trust of all incoming data, which we propose should not be the de facto standpoint.

1.1.1. The Trusted User

We start by considering a non-networked scenario, where all participants are known and in which the software is written by a single composer or creator. This common compositional practice of a composer creating a performance environment/tool for use by a group of performers is inherently a relationship of common intention, if not of complete trust.

It seems logical in this scenario to consider control data from such users to be valid by default, since user interaction within a provided graphical or text-based interface should limit parameters to sane values. This measure mitigates against the performance becoming compromised, either through damage to the musical integrity of the performance from spurious values, or through potential risks to the stability of the application.

1.1.2. The Network Attached User

With the increased availability of computing resources, the emergence of enabling technologies for networking, such as Open Sound Control [9], and the practice of creating distributed environments with network connectivity [3] as carried out by the Milwaukee Laptop Orchestra [7] network performance has become more common. There are also an increasing number of networked performances in which each participant provides their own system, but with the facility for some or all parameters to be accessible to others on the network.

In either of these scenarios the inherent user trust of the isolated device no longer applies (as in non-networked laptop performance). Instead, we argue that the default position should be one of distrust, albeit not always due to suspicion of malicious intent, but rather simply due to the unknown source or reliability of any network data.

Whereas in a non-networked situation as outlined in 1.1.1, the protection of the user interface limitations can be assumed, this is not the case in a networked performance. While the performance intention may be based around the use of a common interface or software, the lack of control over all possible networked attached devices offers the possibility for intervention by unknown users and/or unintended or inappropriate user interfaces.

When a common software interface is not used, there is no guarantee that the interface will offer suitable protection against the insertion of hazardous values, intentional or otherwise. While it is important to consider the malicious user, a more common situation may be the incorrectly configured or inexperienced user creating inappropriate values that expose security problems.

1.2. Scope

Platform stability, crucial to performance, is directly susceptible to risk through possible attack vectors within the operating system and user written code. With the exception of open-source programs, the lack of attack vectors within the underlying operating system and application internals is largely a matter of trust between vendor and client, outside of the scope of this paper.2

Rather, this paper will focus on the security and stability risks created through use of user-written code within performance environments such as ChucK and Max. It should be noted that the illustrated attacks are designed to demonstrate the potential for sabotage, disruption (intentional or otherwise) or prevention of an intended performance only. The compromise of entire systems are not of concern here.

2. EXPLOITS

The proof of concept exploits illustrated here are designed to operate in realistic performance environments. A typical setup is a user system accepting udp packets on a specified port number via OSC over a WIFI network (as required for Scott Smallwood’s PlorK composition ‘On The Floor’ [6], for example).

2.1. Severe

The severe exploits detailed in section 2.1 offer the possibility of disrupting the entire application by exploding language features, rather than bugs. In doing so, the exploits illustrate the requirement for security-conscious user programming. The languages Max and ChucK were chosen due to author familiarity rather than any perceived susceptibility. This should not be viewed as a indictment of the chosen applications but rather that these exploits were developed to illustrate the severity of lack of security.

While these exploits are not universal, in that they require particular code configurations at the receiving end, we contend that these configurations are not unrealistic, and also appear innocuous, with no obvious indications of risk.

2.1.1. Max

Max allows the user to control the application directly by inserting the text: max into a message box, followed by the message to be sent. In Figure 1, the OSC bundle inserts , max crash into a message box and bangs it immediately, causing Max to crash. In the example, the namespace is known, but in fact OSC wildcard matching can be used to send any namesapces allowing this exploit.

2.1.2. ChucK

The ChucK programming language [8] is built around the manipulation of time both at sample rate and also in longer, more musically relevant durations. While ChucK has a primitive data duration, it is typical to communicate

3 Internally Max attempts to dereference a null pointer.

Figure 1. Max Severe Exploit

Figure 2. ChucK Severe Exploit

2.2. Nuisance

Rather than causing complete system crashes, these ‘nuisance’ exploits (see section 2.2) undermine the musical integrity of the performance. Attacks of this type require far less specific coding configurations, rather they rely on the insertion of unexpected parameter values. Thus, they are far easier to achieve, especially due to the human readable nature of OSC namespaces, and popularised use of udp as a transport protocol. While the outcomes may not crash the application, unintended tempo or pitch values would be a significant hindrance to a performance as would altering audio oscillator values to LFO range.3

The methodology for a malicious user trying to identify such nuisance opportunities might be as follows; 1) examine network traffic, 2) identify appropriate range of values, 3) prepare code to transmit nuisance value, 4) send value.

2.2.1. Examples of Nuisances - Rogue values

Figure 3. Max Nuisance Exploit

In the example code (Figure 3 and Figure 4), an oscillator frequency is set directly by an incoming value. The oscillator’s audio output is then sent directly to the audio interface output. As this signal is to be heard, the implicit expectation is that incoming values will set be within the audio range. However, it is entirely possible to set the frequency to an inappropriate value in terms of the ongoing performance or even outside of the audio range, potentially reducing the amplitude range available to valid

3 Should also be noted that at appropriately ranged value transmitted at the wrong time would also be considered a nuisance value.
3. CURRENT HIGH-LEVEL NETWORK SECURITY PRACTICES

The exploits identified require network connectivity. Consequently, we consider typical security counter measures in relation to both the trusted and the malicious user.

3.1. Physical Transit

The physical transit of the network connection must be compromised to allow a malicious attack or even an incorrectly configured client. A fully-wired network offers a significant obstacle not in part due to the difficulty of hiding a physical intrusion in a performance environment; however, the issues surrounding trusted users remain a concern.

3.1.1. WIFI Security (Social Engineering Hacks)

Typical WIFI network security offers only limited protection from malicious users. WEP is now considered compromised [4], while WPA, though cryptographically still secure, is susceptible to guessing or social engineering tricks targeted at pass phrase acquisition. Consequently, data from a wireless network should not be considered secure and the trusted user issues remain.5

3.2. Firewall

While a correctly configured Firewall can be used to prevent network connections, client applications are designed to receive network data and thus appropriate firewall ports must be left open. However, the firewall could be configured to only accept packets from approved IP clients.

5Wireless networks are also susceptible to denial of service style attacks [5] and due to the lack of physical connection, attempted intrusion of this sort is more subtle.

3.3. Multicast

The use of multi-casting (e.g. using mxj net.muxhole object in Max) removes the complexity of network configuration, and is convenient for performance. However, as all network traffic is mirrored to all network connected devices, profiling network traffic is easy. Additionally, as a udp receiver accepts all inbound traffic regardless of source, a device need only send data to inject values.6

3.4. Obscurity

One of the major features of the OSC style namespaces is the human-readable and context-specific nature of the namespaces. A self-identifying namespace offers an opportunity to send appropriate data with an understanding of its context. Whilst namespaces could be obscured in order to increase security, this approach runs counter to the design of the protocol, and is not an appropriate solution for scenarios in which human examination of namespaces is a strong requirement, rather than simply a convenience (e.g. a network jam).7 Regardless of the viability of this approach in a given situation, it is arguable that examination of the network traffic will expose the purpose of specific namespaces, due to the observable range and frequency of change of parameters.

4. SUGGESTED SOLUTIONS

Having identified these security issues we now suggest some methods that offer protection from these behaviours.

4.1. Transport Layer Security

While the ease of use of udp protocol for multi-casting or receiving is attractive, the security issues are significant. Consequently, we would suggest using the TCP protocol as doing so would require an intruder to spoof a connected device.8 Thus the use of an ssh tunnel, or VPN, within an encrypted WIFI or physical network would present a

A conceivable example of this would be a workshop participant leaving the software running through the performance.

The web article ‘Security Through Obscurity. Ain’t They Think It’s ’ by Jay Beale ([1]) offers a discussion regarding the appropriate use of obscurity.

Readers concerned about the latency implications of TCP transports should refer to [2], which indicates that, under normal situations, the latency is comparable to that achieved using udp.

4.2. Data Obfuscation

Obfuscating namespaces offers limited protection; however, analysis of sent values will still identify possible nuisance values. It does, however, make interacting with network control data more difficult. While encrypting OSC traffic from within the application space is an option, an ssh tunnel is likely to be more secure and requires less maintenance by the programmer. Consequently we do not advocate security through obfuscation.

4.3. Data Sanitisation

While severe code exploits are worth identifying and avoiding, nuisance value insertion is always inherently possible. As discussed above, ensuring incorrect values are not inserted into a data stream is a difficult task and as a consequence it should be assumed that not all values within the data stream are suitable for use. Therefore incoming data should be sanitised.

This sanitisation should be considered a two-part process: firstly, filtering out inappropriate data types; secondly, blocking out-of-range values. Obviously, such filtering will be context-dependent, informed by musical intentions. Within the performance paradigm it is also conceivable to vary the criteria for acceptable values in line with score directions.

4.3.1. Max Exploits Sanitised

It is arguable that the .max command should be escaped in all cases due to its severity. In Figure 5, the input is sanitised by disallowing any messages containing a semicolon to be send to a message box. The regexp object matches anything with a semicolon and only unmatched messages are sent on to the message box.

Through the use of zmap in Figure 6 only values within the audio frequency range pass through the patcher.

4.3.2. ChucK Exploits Sanitised

The if control structure within Figure 8 prevents values outside of the approved range from being used.

5. CONCLUSIONS

Within this paper we have sought to illustrate the risks in network connectivity within musical performance and highlight the severity of the issues presented. Having evaluated possible security counter measures we have identified limitations to common approaches and highlighted the need for distrust amongst network environments. Finally, we have demonstrated simple data sanitisation to mitigate the most severe risks and provide greater stability to network performance systems.

References


A PIANO LEARNING SUPPORT SYSTEM CONSIDERING RHYTHM

Yoshinari Takegawa
Future University Hakodate
Hakodate, Japan

Tsutomu Terada
Kobe University / PRESTO, JST
Kobe, Japan

Masahiko Tsukamoto
Kobe University
Kobe, Japan

ABSTRACT
Playing the piano requires various techniques such as correct keying, fingering and rhythm. Our research group developed a piano learning system to support correct keying and fingering for beginners. However, the system did not support the learning of rhythm. Rhythm consists of various kinds of note and rest, and it is difficult for beginners, who are not used to reading a score, to understand the different duration of each note and rest. Alternatively, there are piano roll scores, which describe timing of keying and releasing clearly, but which do not teach players how to read a musical staff. Therefore, the goal of our study is to construct a piano learning support system that considers rhythm. We discuss methods of effectively indicating information for piano performance, such as rhythm information, while teaching how to read a musical staff. We have developed a prototype system, and evaluated its effectiveness through actual use of the system. We found that it had significant advantages over a piano roll method.

1. INTRODUCTION
Piano players need to master various techniques and skills, such as reading a score, correct keying, proper fingering, correct rhythm (the timing of pressing and releasing a key), keeping tempo, and dynamics. Players generally need long-term training. Unfortunately, beginners often give up because of the difficulty of acquiring these techniques.

Our research group developed a piano learning system to support correct keying and fingering for beginners[20]. It uses a projector which is set above the keyboard and can display information along the entire MIDI keyboard. The proposed system has a fingering check function that uses the real-time fingering recognition technique that our research group developed [21]. Additionally, we devised presentation methods to indicate useful information for piano performances effectively. We place emphasis on teaching how to read a musical staff in order to enable learners to be independent from our proposed system after training.

Another important aspect of performance is rhythm because it affects performance quality. When learners play rhythm incorrectly, the performance is awkward even they press the correct keys. There are various kinds of note and rest on a score. It is difficult for beginners, who are not used to reading a score, to understand the different duration of each note and rest, thus they can learn rhythm most effectively by using a mechanism that allows them to intuitively understand the different durations. Additionally, piano performance requires complicated and precise fingering control for each hand in regard to timing. Many beginners give up playing the piano with both hands due to the difficulty of the independent movement of each finger and hand, for example the difference between the timing of releasing a key with a right-hand finger and that of a left-hand finger. It is important to make learners understand their mistakes for example by imposing penalties for errors. The effectiveness of rhythm learning improves through checking mistakes and imposing penalties, such as the system withholding the next piece of learning support information when a learner makes a mistake. Moreover, learners have to acquire proper rhythm as early as possible since it is difficult for them to rectify their mistakes once they are accustomed to playing incorrect rhythm. Furthermore, as the duration of each note and rest depends on tempo, learners have to be conscious of this as well.

Our research group developed a piano learning system to support correct keying and fingering for beginners. However, the system did not support the learning of rhythm. Even if users, who are beginners but practicing playing the piano using the proposed system, press the correct keys with proper fingering in slow tempo with both hands and can foresee the next keys which are to be pressed, the performance is awkward because of the incorrect duration of holding keys and inserting incorrect rests. This is due to the difficulty of paying attention to the notes’ duration while moving each hand in different timing. There are piano roll scores, which describe timing of keying and releasing clearly, but which do not teach players how to read a musical staff. The musical staff is the general medium used in musical performance. If beginners cannot read music, they cannot play pieces of music which are not stored on the system, without using the system.
Therefore, the goal of our study is to construct a piano
learning support system that considers rhythm.

We discuss methods to effectively indicate informa-
tion for piano performance, such as rhythm informa-
tion, while teaching how to read musical staffs. For example,
the proposed system shows the musical staff with colored
bars layered over the notes and rests to indicate their du-
ration. In this way, learners can understand the duration
of each note and rest intuitively even while playing the pi-
ano. Moreover, the system has a rhythm check function
that allows learners to notice rhythm mistakes and rectify
them, using a metronome function. Learners can flexi-
ibly and easily control the speed of the metronome with a
foot pedal.

The remainder of this paper is organized as follows:
Section 2 describes related work, Section 3 explains the
design of the learning support system, Section 4 describes
its implementation, Section 5 explains our evaluation and
discusses the results, and finally Section 6 describes our
conclusions and future work.

2. RELATED WORK

There are many studies of methods to support piano learn-
ers. Piano Tutor[14] is an interactive expert system that
uses with multimedia technology, and has functions such as
automatic page-turning based on score-following tech-
nology, creating performance support information and pre-
senting it with video, music notation, and graphics in re-
sponse to learners’ performance. Piano Tutor does not use
a projector to show performance support information, and
the presentation method of Piano Tutor is typically differ-
ent from that of the proposed system. However, Piano
Tutor is a comprehensive learning system, and there is a
possibility that we can develop a more effective learning
system by utilizing Piano Tutor’s knowledge.

There are keyboards and software[1, 3, 5] that display
keying position, fingering, and sample videos as support
information during performance. However, these have
problems, such as the lack of a rhythm check function,
as described in Section 1.

PianoTouch[11], ConcertHands[2], and MaGeS Y
Trainer Piano[8] are haptic-based instruction systems for
piano learners. They give a player performance informa-
tion through a tactile feedback unit attached to each finger.
Learners are able to learn keying and fingering techniques
easily but they are forced to wear bulky devices on the
fingers.

Additionally, there are systems that automatically de-
tect the weak points of learners including mis-keying and
fluctuation of tempo or dynamics on the basis of a con-
tventional practice log[12, 16, 17, 19]. There are also
piano lesson support systems[18] that show current ar-
ticulation, agogik, and dynamics. Although these systems
do not have rhythm check functions, we derived useful
knowledge from their development and have put it to use in
our learning support system.

Our research also relates to augmented reality research.

Many new types of projector-based augmented reality sys-
tem[6, 7, 9, 10, 13, 15] have also been proposed. These
works attempt to assist a simple movement-based task.
However, our system supports the learning of an intricate
physical task by tracking the movements associated with
the task and augmenting the physical environment with
prompts and other information to aid the task.

3. SYSTEM DESIGN

As described in Section 1, our research group developed
a piano learning system for beginners to teach correct key-
ing and fingering, as well as how to read a musical staff,
to enable learners to play music, which is not stored in
the system, without the support of the system. However,
the system does not support rhythm. Therefore, we con-
structed a rhythm learning system on the basis of improv-
ing upon the previous system. The proposed system has
presentation methods that help to effectively convey piano
learning, including only not only fingering and keying but also
rhythm information (described in Section 3.3 (i)). The
rhythm check function uses a clear presentation method
to allow the learner to recognize and rectify his or her
mistakes (described in Section 3.3 (ii)). Moreover, we
propose a metronome function (described in Section 3.3
(iii)) as well as a function to enhance the usability of the
metronome (described in Section 3.3 (iv) and (v)).

3.1. Previous system

In the previous system, the projector is set above the key-
board and is able to show information along the entire
MIDI keyboard, as shown in Figure 1. Learners find the
piano learning information easy to understand as the pre-
vious system present various kinds of contents, such as col-
ourful figures and characters in an appropriate position to
allow to learners to see the information easily even while
playing the piano. Additionally, the previous system has
a function that recognizes fingering using a camera[21],
and develops methods for presenting learning support in-
formation for users to check their keying and fingering.

In the following section, we explain the information
presented by the previous system. The letters in Figure 5
correspond to the following list:

(a) Each musical note is connected to the corre-
sponding key with a line. This visual support enables
learners to read a score easily, because they can clearly see
the relationship between the musical notes and key
positions.

(b) The numbers and colors of the NextKeys are
the finger- ing information. When the NextKey is pressed
using the correct finger, the key is filled in with
the corresponding finger color. The left NextKey is
yellow colored because the correct finger has been
placed on it. On the other hand, when the NextKey is
pressed with the incorrect finger, the key is col-
ored red. When other keys besides the NextKeys
are pressed, these keys are also colored red. In this
way, learners can understand the positions of the
NextKeys, learn fingering technique intuitively, and
rectify their mistakes.

(c) Each note is represented as a bar. The left
NextKey is outlined in color to provide keying in-
formation. The NextKeys are indicated by the ar-
rows (a) in Figure 1.

(d) The results of evaluative experiments confirmed
that our system significantly enhanced learning effectiveness
in the early stages of practice, when compared with the
lighted keyboard method which turns the NextKeys red.

3.2. System structure

The structure of the system is shown in Figure 2. The sys-
tem has a foot pedal to control the tempo of the metronome,
and a projector to present learning support information.
The projector is set above the keyboard and can display
information along the entire MIDI keyboard. The system
uses MIDI data including pitch data and intensity data
from the MIDI keyboard.

3.3. Presented information

We explain the presented information with Figure 2. This
information is updated in sync with the performance. The
Roman numerals in Figure 2 correspond to the following
list:

i) The duration bar Rhythm consists of various kinds of
note and rest, and it is important for beginners to un-
derstand the different duration of each one. There-
fore, the proposed system enables learners to under-
stand the duration of each note and rest by showing
colored bars, the lengths of which correspond to the
durations of each note and rest as shown in Figure 3.
Additionally, the color of the bar turns from blue
to yellow as the learner holds the key. In this way,
the learner can intuitively understand the remaining
time for which he or she must hold the key. If the
learner holds the key too long the color of the bar
turns from yellow to red and the length of the bar
increases until the learner releases the key.

ii) Rhythm check function The system has a function that
checks the timing of pressing and releasing a key
and whether the key is held for the correct duration.
Moreover, the system checks the timing of press-
ing several keys simultaneously, for example when
Users can turn the metronome on or off. Current tempo and beat are displayed at the distal ends of the keys shown in Figure 2. The tempo and the number of beats of the metronome are controlled by pressing the keys that represent current tempo and beat, respectively.

Control of the metronome using a foot pedal

Different parts of a piece of music have different degrees of difficulty. When a learner is practicing difficult parts, he or she tends to play in a slow tempo at first and then gradually increase the speed. On the other hand, when the learner practices easy parts, he or she plays in the tempo indicated by the score. Therefore, learners can practice a piece of music more effectively if they have flexible control of the tempo. We adapted a foot pedal to control the tempo of the metronome flexibly, and the tempo gets faster when the learner pedals.

Adjustment of the start point of the metronome

There may sometimes be a lot of unexpected pauses because of the difficulty of playing certain parts of a piece of music. Additionally, beginners, who are not used to using one, find it difficult to adjust their own performance to the sound of a metronome. Therefore, we propose a function that automatically adjusts the start point of the metronome to the performance. In this way, beginners do not have to consider the timing of the metronome, and can start playing whenever they like.

Presentation of keying position and fingering

This function was also included in the previous system. When a key is outlined in color this indicates that it is the next key that should be pressed. A number on the key denotes fingering. This function is useful for beginners, who cannot read out keying and fingering information from a piece of music.

Selection of cue points

Users can select cue points which are indicated on the score by numbers in black squares as shown in Figure 4. The cue points enable learners to change the point from which they want to start practicing. This function is useful when learners want to practice part of the score again and again without having to start from the very beginning each time.

Switching of each function

These functions are controlled using the keyboard. Keys can be assigned to commands for operating the system, and an icon which represents the command assigned to a key is displayed on a key.

4. IMPLEMENTATION

We implemented a prototype of the piano practice support system, as described in Section 3.3. We used a SONY VGNS94PS (Intel Core2 Duo 2.60GHz), running Windows 7, a CASIO Privia PX-110 equipped with 88 full-sized keys. We used a Benf MP779 ST as the projector. The projected area was 6 octaves (72 keys) and we painted all the black keys of the MIDI keyboard white. We implemented the system using Microsoft Visual C++ .NET 2010 and Intel OpenCV Library. The prototype is shown in Figure 5.

5. EVALUATION

We conducted an evaluative experiment to investigate the effectiveness of the proposed system in the beginning stage of piano performance, when a piano beginner is practicing the keying, fingering, and rhythm of a new score.

5.1. Experimental Procedure

The evaluation procedure was as follows:

Comparative method

In this evaluation, we compared the proposed method to a piano roll method, and a method without rhythm support, based on the number of keying and rhythm errors. Piano roll scores describe timing of keying and releasing clearly, and are used in KEYBOARD MANIA[4], which enables players who have no formal musical instrument training to enjoy piano performance easily. In the piano roll method, each key has a corresponding vertical bar on the screen as shown in Figure 6. Rectangular icons scroll down the bars to indicate which keys the learner should press. Users can understand the duration of each note and rest because the size of the rectangular icons is based on the duration of the corresponding notes. Timing is also easy to understand as the user simply presses the matching keys when the rectangular icons descend to the bottom of the screen.

Table 1 shows the application of functions for each method.

In the piano roll method, the system displays not only a piano roll score but also a musical staff on the piano roll score. Users are able to see both scores.

The proposed method presented the next learning information when subjects had pressed a correct key with correct rhythm, whereas the piano roll method and the method without rhythm support presented the next information when subjects had pressed only a correct key. The default speed of a metronome is that the duration between clicks is 0.6sec. One sixteenth note is equal to two clicks. The Sixteenth note was the smallest note in the trial score.

There are three types of keying error: incorrect keying, when the subject presses an incorrect key, as shown in Figure 7-(a), non-keying, when the subject does not press a key that the musical staff indicates should be pressed, as shown in Figure 7-(b), and extra keying, when the subject presses not only correct keys but also other keys, as shown in Figure 7-(c).

There are two types of rhythm error: extra rest and incorrect holding time. Incorrect holding time is when the subject holds a key over or under the indicated time, taking into account the time allowed for error. In this evaluation, we define the error margin as plus or minus 0.3sec. For example, the duration of the sixteenth note is 1.2sec in the tempo used in the test phase, and the rhythm is deemed correct if the subject holds the sixteenth note from 0.9sec to 1.5sec. Moreover, extra rest is when the time from releasing the current key to pressing the next key exceeds 0.6sec.

Some subjects sometimes held keys while searching for the next keys to be pressed, and released keys ahead of...
errors of the proposed method is small. The difference between the average number of each error, when comparing the proposed method to the piano roll method and the method without rhythm support, was at a level of 5%, calculated from Steel-Dwass’ multiple comparison test. We discuss the results relating to the proposed functions as follows. The behavior of the subjects was observed by the person overseeing the experiment, who consulted with the subjects directly after the evaluation.

The duration bar The reason that the subjects who used the proposed method were able to learn piano performance effectively is that the comprehension of rhythm and the reading of a musical staff were improved. The subjects who used the proposed method or the piano roll method passed on comments such as that the explicit presentation of the rhythm helped them to enhance their comprehension of it.

The subjects using the proposed method were able to acquire not only the rhythm information but also pitch information at the same time, as the duration bar was layered over the notes of the trial piece. In the beginning stage of the evaluation, the subjects concentrated on acquiring the keying information presented on the musical keyboard and rhythm information from the duration bars, and playing with the correct keying and rhythm based on the acquired information. Once they were using the trial piece, they began to understand the connection between the keying and rhythm information and the notations on the musical staff, and they became able to read out the pitch and rhythm information from the musical notes directly. The subjects using the piano roll method, in the beginning stage of the evaluation, did not look at the musical notations on the trial piece, as they practiced the keying and fingering while looking at the information presented on the keyboard. Next, they used the piano roll score to learn the rhythm once they had almost mastered the keying position and fingering. They could not afford to look up at the musical staff above the piano roll score. Finally, they hardly spent any time practicing with only the musical staff score. As a result, when they performed the trial piece in the test phase and unknown or difficult notes directly. The subjects using the piano roll method, sometimes ignored the click of the metronome because of the difficulty parts again and again.

Adjustment of the start point of the metronome In regard to the adjustment function for the start point of the metronome, the subjects passed on comments such as that the function was convenient because they did not have to consider the timing of the metronome before starting to play. We particularly noticed that the subjects needed a lot of rests to check keying position and fingering in the beginning stage of this evaluation. Therefore, subjects using the method without rhythm support sometimes ignored the click of the metronome because of the added difficulty of keeping time with it.

Presentation of keying position and fingering on the keyboard, and selection of cue points All the subjects used the function that presents keying position and fingering from the beginning stage of this evaluation. Furthermore, the cue point function was also used frequently to practice difficult areas again and again. We confirmed the effectiveness of these two functions from the comments of all the subjects as well.

Table 1. The applicable functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Proposed method</th>
<th>Piano roll method</th>
<th>Method without rhythm support</th>
</tr>
</thead>
<tbody>
<tr>
<td>The duration bar</td>
<td>Applicable</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Rhythm check function</td>
<td>Applicable</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Control of the metronome using a foot pedal</td>
<td>Applicable</td>
<td>Applicable</td>
<td>NA</td>
</tr>
<tr>
<td>Adjustment of the start point of the metronome</td>
<td>Applicable</td>
<td>Applicable</td>
<td>NA</td>
</tr>
<tr>
<td>Presentation of keying position</td>
<td>Applicable</td>
<td>Applicable</td>
<td>Applicable</td>
</tr>
<tr>
<td>Selection of cue points</td>
<td>Applicable</td>
<td>Applicable</td>
<td>Applicable</td>
</tr>
<tr>
<td>Displaying a pointer error</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

Table 2. The average number of keying and rhythm errors

<table>
<thead>
<tr>
<th>Function</th>
<th>Keying error SD</th>
<th>Rhythm error SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Piano roll method</td>
<td>3.0 ± 1.4</td>
<td>15.7 ± 3.5</td>
</tr>
<tr>
<td>Method without rhythm support</td>
<td>34.5 ± 0.7</td>
<td>46.0 ± 11.3</td>
</tr>
</tbody>
</table>

6. CONCLUSIONS

We constructed a musical staff-based piano learning support system considering rhythm learning. The learner understands the duration of notes and rests intuitively by using the duration bars layered over the notes and rests on a score. The learner can also understand the remaining time for which they should hold each key by observing the changing color of the bars. The Rhythm check function helps users notice their own mistakes and rectify them. The results of evaluative experiments confirmed that the subjects using our proposed system played the trial piece using correct keying and rhythm during the 30 minute training period, and the system significantly enhanced learning effectiveness in the early stage, when compared with the piano roll method.

As described in Section 1, playing the piano requires various techniques, such as correct keying, fingering and rhythm, which generally need long-term practice. Therefore, conventional piano learning methods make learners practice each technique individually, thus beginners often give up as it takes tremendous time and effort to acquire the skills needed to play a song adequately. We propose a comprehensive learning style, which allows learners to acquire several skills at the same time, by enhancing human processing ability using multimedia technology and information design technology. Future work will involve constructing a more comprehensive piano learning system that includes not only keying and rhythm information but also fingering, dynamics, and articulation, and will include evaluative experiments conducted on beginners of various generations, as well as experiments carried out over a longer period of time.

7. ACKNOWLEDGMENTS

This research was supported in part by a Grant-in-Aid for Scientific Research for Young Scientists(B) (21700198) from the Japanese Ministry of Education, Culture, Sports, Science and Technology.

8. REFERENCES


Figure 7. The measurement of rhythm errors and keying errors


some conducting systems have been developed with educational and research goals in mind. A system designed to analyze and classify hand gestures of conductors is based on Hidden Markov Models (HMM) and developed for MAX/MSP [15]. Similar ideas and goals can be seen in Conga, a gesture analysis system using graph theory [11], analyzing hand poses without real-time constraint [16], video analysis of conductor’s gestures [17], and a baton simulation using a Wii remote [18].

Some projects introduce complex algorithms and systems for conductors’ gesture analysis, and within the constraints and limitations they impose on the gestures they report high accuracy. Most of these related works focused on the activity of conducting in its highest level – developing a digital system that allows an individual to conduct a short excerpt of an entire musical composition. These systems focus primarily on the use of movement to control the speed of prerecorded pieces of music, seeming to aim for the education and entertainment of the general public rather than the learning of kinesthetic skills in order to produce effective conducting gestures to indicate a combination of tempo, duration, articulation, and dynamics.

3. INNOVATIONS

In all the projects described above, there is a missing component which the Maestro system improves upon: previous systems have been developed based on the assumption that conductors’ gestures convey mostly (or only) temporal information; when in practice, a conducting gesture must convey additional aspects of sound generation, such as articulation, volume, and duration. In order to detect and provide feedback for various aspects of gestures, the Maestro system introduces techniques based on an invariant in three main areas: a) gesture anticipation and tracking; b) machine learning for gesture analysis; c) utilization of physical modeling for high-quality audio feedback.

3.1. Gesture anticipation and tracking

Since any delay that occurs between a performed gesture and its audio feedback is undesirable within a music-conducting system, gesture anticipation, allowing precision of a few milliseconds is an essential requirement. The Maestro system uses a high-speed sensing device that provides a data-sampling rate close to 100 Hz in 3D space. Such high-resolution data, combined with pre-trained gestures, allows the anticipation of gestures and achieved accuracy of a few milliseconds.

3.2. Machine Learning for Gesture Analysis

On one hand, Maestro’s machine learning algorithm requires two kinds of analyses a) real-time classification of a performed gesture by comparing it with a set of pre-trained gestures, and b) real-time identification of higher-resolution characteristics of the classified gesture. Both comparisons will be performed with the ultimate goal of mapping any subtle change in a gesture to subtle parameters that will influence the audio feedback.

3.3. Physical Modeling

Physical modeling is a set of audio signal processing and synthesis algorithms that have been developed based on extensive research of the behavior of acoustic instruments. These models allow the synthesis of realistic-sounding audio with relatively low computational and technological resource cost [19], [20].

High-resolution sensing and tracking devices, along with the proposed machine learning-based gesture classification, will allow for an intuitive utilization of physical modeling synthesis, where we will map one gesture to multiple parameters of physical modeling-based musical response. Previous conducting projects have used either MIDI [6], [10], [18] or sampled sounds [11], [12], [13] for audio feedback. Physical modeling is another major step towards a realistic conducting environment.

4. SYSTEM DESIGN

The system consists of four interconnected modules as illustrated in Figure 1. The modules will include a conductor’s baton that transmits to the computer software to analyze the gestures, and an interface for audio and video feedback. The baton will serve as the physical interface for the user. Spatial coordinates of performed gestures are sent through an IR transceiver to the desktop application, where they are recorded and analyzed. Once the analysis algorithm recognizes the completion of a gesture, the system generates audio and visual output that correlates to the performed gesture.

Once a gesture is detected, Maestro’s machine learning algorithm classifies it and the recognized gesture is translated to audio and visual feedback.

4.1. Conductor’s Baton

The baton is a real conductor’s baton, fashioned with an infrared LED (Light Emitting Diode) at its tip to allow movement tracking in a 3D space. Infrared sensors were chosen since they track only the movement of infrared light sources, thus avoiding confusion with other objects in space [21]. The baton is wireless to help simulate a real conducting environment.

In addition to the infrared sensor on the baton, higher-level detection (with a lower sampling rate) of the conductor’s head and torso movements are also sensed and allow the detection of skeletal movement in the 3D space. This analysis, combined with the baton movement, allows the rendering of the visual feedback.

4.2. Anticipation and Tracking

Once data are fed into the system (raw coordinates of baton movement), 2D representations of the baton movements are reconstructed by the software, and are analyzed in two parallel stages. A gesture detection algorithm distinguishes between random movement, system noise, and intentional gestures. The system then searches for specific characteristics of the conducting gestures (e.g. beginning and end of a gesture). A second algorithm anticipates the end of a gesture (i.e. attack – when the baton movement stops) that allows a time-accurate audio feedback without discernable time delay. This algorithm gathers information on a gesture before the parallel algorithms determines that the current movement is indeed a gesture.

Classification of gestures relies on two orthogonal algorithms, providing two layers of detection accuracy. First, gesture statistics pertaining to the current gesture characteristics (e.g. vertical gesture length, acceleration, attack characteristics) are gathered by the anticipation algorithm, and are compared with gathered statistics of the trained gestures. The second layer is a Hidden Markov Model (HMM) algorithm, commonly used for gesture classification and following, specifically for conducting gesture classification [22], [15]. The HMM compares the gesture as a whole once the statistical analysis is complete, and there is a positive match between a performed gesture and a trained one. The two algorithms complement each other to achieve two goals: anticipate the next gesture to provide audio feedback with no discernable time delay, and prevent false positives for cases in which random baton movements might be mistaken to be real gestures.

4.4. Audio and Visual Feedback

Once classification is successful, the musical content is constructed and the recognized gesture is translated to audio and visual feedback.

4.4.1 Audio Feedback

Parameters gathered from the detection algorithm, along with the classified characteristics of the gesture are mapped to produce a tailored sound, correlating in dynamic, articulation, duration, and articulation to the performed gesture. By mapping the rich space of subtle gesture analysis to the rich space of physical modeling sound generation, we are able to provide a sophisticated and intuitive response that would imitate the response of a real orchestra.

The high-resolution sensing and tracking devices, along with the proposed machine learning-based gesture classification, allow the rendering of the visual feedback through synthesis algorithms and models that have been extracted from the gesture in real-time. During tests with the authors conducting, the detection rate (judging if a certain baton movement is a gesture) was 92%, while the classification rate (match between the conductor’s intent and the perceived audio feedback) was 81%. The system displays the back audio feedback comprised of one instrument, and displays the visual feedback in real-time as a mirror image of the gesture.

6. FUTURE WORK

The second iteration of the system will include an expansion of the number of sets of trained gestures and melodic excerpts in order to provide a richer learning environment. These will build on the current discrete gestures to successive gestures and multiple meter patterns. Additionally, future work with audio feedback will move beyond a single instrument sound to allow the user the option to hear full orchestra, band, or vocal sounds in response to their gestures. A second iteration will also include a sophisticated, yet intuitive user interface to allow the user to change sound preferences, move between practice modules, visually and audibly record their session, and change camera viewpoints.
The desired end result of this work is to provide a new, meaningful tool to music conducting pedagogy that enhances conductors’ development of subtle gesture affecting a full range of musical expression. The Maestro system is being developed iteratively and incrementally with input from conductors of various competency levels. An accompanying curriculum is also being developed and will be deployed within the context of a music conducting class of undergraduate music majors. The system will be disseminated and evaluated in an undergraduate introductory conducting course, evaluated by participating students and the course instructors. Following the analysis of the evaluations, further modifications to the Maestro system and collaborative curriculum will be made before another iteration of the study the following year.

Future potential uses of the project include widespread accessibility to conductor training programs and the appropriation of the project for use by individuals at all levels of musical skill and age. System components and techniques that will be developed as part of the project could also be used in medical research such as communicative and movement abilities of disabled persons, sign language technologies for people with visual disabilities, novel gaming interfaces, and music creation software.

ACKNOWLEDGEMENTS

The authors would like to thank Marcelo Ciccone for his help with the initial setup of the hardware interface, and his continuous help with the implementation of the HMM algorithm.

REFERENCES


The desired end result of this work is to provide a new, meaningful tool to music conducting pedagogy that enhances conductors’ development of subtle gesture affecting a full range of musical expression. The Maestro system is being developed iteratively and incrementally with input from conductors of various competency levels. An accompanying curriculum is also being developed and will be deployed within the context of a music conducting class of undergraduate music majors. The system will be disseminated and evaluated in an undergraduate introductory conducting course, evaluated by participating students and the course instructors. Following the analysis of the evaluations, further modifications to the Maestro system and collaborative curriculum will be made before another iteration of the study the following year.

Future potential uses of the project include widespread accessibility to conductor training programs and the appropriation of the project for use by individuals at all levels of musical skill and age. System components and techniques that will be developed as part of the project could also be used in medical research such as communicative and movement abilities of disabled persons, sign language technologies for people with visual disabilities, novel gaming interfaces, and music creation software.

ACKNOWLEDGEMENTS

The authors would like to thank Marcelo Ciccone for his help with the initial setup of the hardware interface, and his continuous help with the implementation of the HMM algorithm.

REFERENCES


3. ARCHITECTURE AND IMPLEMENTATION

Nuance has been designed such that it can synchronize and record data from a variety of inputs and modalities. In this section, we provide an overview of the Nuance recording system and its capabilities.

3.1. Design Overview

As mentioned in section 2, the primary aim of Nuance was to develop a recording application with a traditional DAW-like workflow. The software should be intuitive and support additional output formats (e.g. SDIF/GBDF1) in the future. Other synchronization schemes are possible, and may be required in the future if additional data sources are added. Additionally, Nuance has been written to support additional output formats (e.g. SDIF/GBDF1) in the future.

3.2. System Overview

The general flow of the software system is detailed in Figure 2. A user provides various multimodal input streams, which are recorded as audio files. By default, all streams are recorded as 16-bit uncompressed wav files, at a sample-rate of 44.1kHz. This can be adjusted in the program preferences panel, depending on the requirements and capabilities of the users system, up to 24-bit resolution, and a 192kHz sample-rate.

3.2.1. Synchronization

Nuance implements a synchronization scheme driven by the computer audio cards sample-rate clock (Figure 2). Each sensor or input is responsible for updating itself asynchronously at its own independent rate, and all data-streams are read and recorded within a guaranteed synchronous and thread-safe audio callback system. Whenever a new audio buffer is available, each recorder is simultaneously notified to record its data. For an audio input, this simply means writing its current block of audio. For serial, OSC, and MIDI data, the most recent sample is copied into an array (of equal size as the audio-block) and synchronously written to disk. This sample-and-hold and up-sampling of sensor data happens at a much faster rate than common sensor systems supply new data, and we have found it to be more than sufficient in terms of speed and resolution for MIR applications. Other synchronization schemes are possible, and may be required in the future if additional data sources are added. Additionally, Nuance has been written to support additional output formats (e.g. SDIF/GBDF1) in the future.

3.3. Multimodal Input

A primary concern with Nuance was to support heterogeneous input channels. While the initial four supported input channels are audio, serial, OSC, and MIDI, the Nuance codebase has been written with future extensions in mind. In the following section we describe Nuances multimodal capabilities in greater detail.

3.3.1. Audio

Mono audio recording is achieved in Nuance by adding an Audio Recorder track to a Nuance session. Each Audio Recorder has the following parameters: real-time waveform visualization, input channel selector, a gain slider, and a record arm button.

Figure 3. Audio Recorder Object

As many projects in the community utilize Amel/Arduino/PIC microprocessors, supporting serial communication was a major design consideration. For generalization purposes, Nuance currently supports serial devices outputting data in the following serial format:

Field 0: Sensor/MetaMessage
Field 1: data (32-bit)

A typical use-case using an Arduino microcontroller with two force-sensing resistors connected to analog inputs 0 and 1 might look something like Figure 5.

Figure 5. Example Arduino Serial out messages for two Analog Sensors

In this example, “fsr1” and “fsr2” would be the Sensor/StartMessages, which are immediately appended by the data, and finally followed by a new line character (via println). Nuance uses the new line character to delineate each serial message. Once the serial messages are streaming in the correct format, the user must provide an .xml file (Figure 6) to each sensor recorder object. The .xml file outlines the expected sensor start messages (“start”), paired with a human-readable name (“ID”) to show in the Sensor Recorders input selector. A built in message configuration panel is

1 E.g. Gibson HD.6X digital Les Paul Guitar, YouRock MIDI Electric Guitar, Rock Band 3 Stratocaster Pro, Freightlight Guitar
2 Buffer size is adjustable via the “preferences panel”
3 Sound and Gesture Description Interchange Formats
being considered for a future release, enabling start messages (and paired human-readable names) to be defined and automatically available to all sensor recorders without having to load an .xml file. The serial-protocol currently implemented was designed for simplicity; however, other more optimized protocols are being considered in the future. For serial-based interfaces that cannot conform to the supported protocol format however, it is still possible to capture data via the OSC and MIDI recorder objects.

3.3.3. Open Sound Control

Open Sound Control (OSC) is a versatile communication channel that allows data to be streamed via external sources. The OSC Recorder greatly extends the capabilities of Nuance, making it possible to record data streaming from external applications on the host machine, and from applications and sensor systems connected to networks or remote computers. Additionally, the OSC recorder provides the ability to record sensor-systems or hyperinstruments that do not or cannot follow the generic serial protocol (via a serial-to-osc middleware).

Example external sources can be anything from iPhones and mobile devices, vision tracking and analysis systems, real-time feature extractors, and other derived-data outputs. OSC support allows Nuance to support nearly any input modality and or source data. OSC support allows Nuance to support nearly any input modality and or source data.

4. Recent Applications

Over the course of Nuance’s development, the software has been used in a number of real-world research projects. In this section we present a few examples of the software.

4.1. Performer Recognition

One of the first research projects in which an early version of Nuance was employed was for automatic North Indian sitar performer recognition. Nuance was used to record a multimodal dataset including information from thumb pressure, tilt, and fret sensors, as well as the acoustic output of the instrument. The sitar players (spanning beginner to intermediate levels) played a number of traditional North Indian practice exercises, as well as free improvisations to investigate how well the computer could automatically detect a performer based on characteristics of their performance. More information on the experiment and results can be found in [6].

4.2. Drum Stroke Computing: Left/Right Hand Recognition & Performance Metrics

The most recent use of Nuance has been for automatic drum-stroke identification and drum performance metrics [7]. Nuance recorded the acoustic output of ten performers playing common snare drum exercises, as well as gestural information from two triple axis accelerometers placed on the hands of each performer. In this experiment, the gestural data from the accelerometers were used only in the training phase of machine learning, enabling the computer to recognize left/right hand drum strokes from the mono audio signal alone. Additionally, this research demonstrated new metrics that can be derived from multimodal analysis of drum performance. Other proposed uses of left/right hand recognition include more comprehensive automatic transcription, computer-assisted musical pedagogy (e.g. interactive notation testing and identification), and various uses for control in live-performance contexts.

4.3. Long-term Metrics Tracking

Nuance has been used over the last year for long-term performance metrics tracking. Two musicians (one playing a bowed upright bass and the other a custom electric violin) have been using Nuance on their own as part of their weekly practice routines. Both musicians have been practicing with modified instruments, providing a variety of data streams which include: tilt and gesture of their bow movements, hand-pressure on the bow, bridge pressure on the instrument, and the acoustic output of the instruments. Active research is underway investigating metrics from the player’s performances, such as improvement or trending of rhythm and accuracy over time, improvisatory patterns, and physical and gesture features.

5. Future Applications

While Nuance is already flexible, and is being used in a number of scenarios, we are extremely excited at its nascent potential. There are many areas where Nuance can mature, and in this section we describe some possible future applications in which we hope to see Nuance used.

5.1. Musical Pedagogy

One of our main motivations for Nuance was to create a tool that could lead to what we see as the future of musical pedagogy and performance metrics tracking. We imagine an Adaptive Practice Room, one that is acute to a musicians practice, and offers insightful feedback to help guide and inform training musicians. Already some music education programs utilize various software tools assist in musical training. We have even seen the idea proposed of anthropomorphic robotic music instructors that are capable of responding to human performers [10]. Recently, Percival presented an interesting approach to computer-assisted violin practice, and a good overview of the current state of computer-aided musical learning in [9]. While other researchers in the field have shown clear interest, current tools only tend to analyze audio, and only address specific instruments.

Concurrent work in both the MIR community and industry has proven that robust pitch tracking, beat-detection, and other sort of metrics are obtainable. However, there is no software currently that offers the potential for any practicing musician to play into a system, and get immediate feedback about his/her performance. Nothing yet provides musicians, in any musical context, with the ability to easily track their performance metrics over time. While tools exist for MIR researchers and computer scientists to perform individual experiments, no tool exist which is immediately accessible to practicing musicians, not to mention in the multimodal domain. Although Nuance currently focuses on the acquisition of multimodal data, we hope to support analysis directly in Nuance (or a sister application) in the future. Exporting statistical metrics in a common spreadsheet format would also be a useful first step in this direction. In this way, we hope for Nuance to help bridge this gap between the work happening in the MIR field, the musical classroom, and the way in which future musicians will practice.
5.2. Computational Musicology
We also hope to use Nuance in the future for musicological research. We feel tools like Nuance will prove to be elemental in the preservation of multimodal digital fingerprints of musicians across many musical contexts for future generations. We hope to one day have a digital library not only of musical audio recordings and scores but also transcribed performance subtleties of great performers of the past.

5.3. In The Digital Audio Workstation
Lastly we look forward to a future where multimodal data streams can be integrated into the general workflow of recording and composing as an electronic musician. Working with multimodal musical instruments affords many unique artistic possibilities, from directly manipulating sound parameters, to extracting higher-level features and mapping them to control parameters. Limited ‘general’ tools exist which begin to facilitate these interactions outside of the research laboratory [5], and we hope for Nuance to help guide the way in making this accessible to today’s electronic musicians and composers. With this in mind, we have written the core of Nuance such that it would remain unchanged in the future if we were to author a cross-platform version in VST/Audio Unit/RTAS plug-in formats.

6. CONCLUSION
In this paper we have described Nuance, a software tool for recording synchronous multimodal data streams. Nuance currently supports audio and video input, as well as input from nearly any musical instrument or sensor system via serial, OSC, and MIDI protocols. Nuance differs from other solutions in that it is a no-patching, near-codeless solution. Nuance was designed to be operated by musicians and researchers alike, and has already been used in many real-world scenarios including performer recognition, drum-stroke identification, and performance metrics tracking. Not only can Nuance increase productivity in MIR scenarios, but we hope its tools to and establishes a foundation for other future musical endeavors, both in the MIR-laboratory, the studio, and the musical classroom.

7. REFERENCES

ABSTRACT
There is a gap in Macedonian music. While the classical, pop, underground, and world music scenes are well developed, experimental and electronic music virtually do not exist. The current curricula for music education in the country do not support experimental and/or electronic music education. Because of lack of knowledge and experience, Macedonian musicians cannot keep up with the developments in the international professional field nor compete in the international market. In this article I will describe a model for international collaboration that laid the foundation and put things in motion for the introduction of music technology in Macedonia. The model is based on knowledge exchange and non-formal learning, breaking the traditional way of educating the students as well as the audience. The projects that were realized in the past years according to this model have pushed the boundaries and resulted in a new curriculum for music technology that the School of Music in Macedonia will introduce in September 2012.

1. INTRODUCTION

Breaking the traditional ways of educating, communicating, collaborating, learning, presenting and experiencing in one country is more than a challenge. Once the tradition is so ingrained that it offers safety and comfort, any attempt to change this would fail. Fear of the new, lack of educated staff, lack of resources and facilities as well as conservative educational program within the school system make it almost impossible to even begin introducing changes.

1.1. Historical overview and current situation
Even though considered a region of great musical talents, the formal music education in Macedonia is relatively new. The only School of Music in the country was officially opened in 1978 as part of the University St. Cyril and Methodius in Skopje and took over the previously founded one (1966). Before, musicians were being educated in other cities in former Yugoslavia [3]. It took a great effort to shape the curriculum of the new School of Music as it was and the results were satisfying for the people involved. During the years, any attempts to add contemporary or experimental music were avoided.

In the 1960’s the Macedonian composer Risto Avramovski was among the first to employ sonoristic methods and the use of new technology at the time (the use of electronic instruments and tape) [4]. Even though unique in his artistic language and knowledge that linked the classical composition techniques with the use of electronic means, his many attempts to join the teaching staff at the School of Music failed [3].

With the help of the Royal Dutch Conservatoire in The Hague, in 1998 the School of Music in Skopje set up a studio and a sonology department [8]. The Royal Dutch Conservatoire gave a number of lectures for electronic music including guidance in the use of the studio. The outcome was positive but outstandingly modest. Despite the great interest students showed in learning and making use of the equipment, it remained open for use only to postgraduate composition students, and the school has an average of two or less postgraduate composition students a year.

Occasionally, Macedonian musicians/researchers living abroad organize concerts with international artists, tickling the current musical scene and presenting state of the art pieces from other countries. Unfortunately, offering knowledge that is new can be seen as threatening and therefore rejected. Once the means are there, keeping a locked door doesn’t really stimulate work. On the other hand, only showing how the world has moved on without a concrete idea of how to catch up won’t help either. Because of the economic position of the country, most people cannot afford a computer, let alone the audio equipment necessary to work at home. In the 1990’s a group of composers left the country [5] in search for knowledge. Most of them cannot afford to do so and they should not need to. The knowledge and possibility to learn and grow should be available to them.

1.2. In this text
In this text, I argue that the problem with the previous attempts was the top down approach. The institution was responsible to pass the knowledge on to the students. I decided to take a bottom up approach and start with the students, I will describe a different and creative paradigm based on knowledge exchange and non-formal learning used to introduce music and technology in a traditionally oriented educational system. This paradigm that does not impose itself is presently being modified by the people involved and also immediately includes the much-valued audience.

In the next section, I will discuss the clash between technology and tradition and I will describe how the first step was made possible. Then, I will describe an educat-
2. TECHNOLOGY DOES PENE TRATE

If the richness and authenticity of a culture are measured by the personal signatures of its art [1], than the artists need to be able to constantly explore, search for new languages, re-discover themselves, experiment and reflect with the outside world. We need to create opportunities for continuous information and knowledge exchange along with a space to experiment with the knowledge gained to support the creative drive and place the artist in the professional market. In the past decades, the artistic developments in most European countries rely on technology.

2.1. The challenges technology brings

The fast changing requirements that professionals in the field must satisfy should shape the curricula, making sure they follow the outside world and teach young people how to keep up with the changes while continuously adapting and developing professional education programs [9].

If people from the same generation are shaped by the time period they grew up in [2], then the above mentioned could be challenging for professionals born before technology began to self-evidently become a part of their life. The new generation, born after 1988, has been named Generation Einstein [2]. This generation recognizes that the world consists of an enormous amount of, most of the time, contradictory information; that not everything is what it seems; that a lot is possible and that the world is fragmented and pluralistic [2]. The Generation Einstein perceives and processes information much faster than we are used to. It is a generation of doers and the education in many European countries, not only in Macedonia, is still shaped for doers. So I decided to organize a summer project in the form of an experimental meeting where music technology would be introduced to the technology savvy young generation, full of artistic curiosity but lacking the hands-on experience. In other words, I proposed a blind date between music and technology.

2.2. Challenges in the preparation phase

Being faced with these challenges, the preparation phase before the first project took three years, during which contact was made with every institution in Macedonia important for realizing the task. This includes the University, private schools as well as the appropriate ministries. Even though the idea was generally eagerly accepted, the reactions of the institutions of major importance for this task were modest to say the least. Some institutions, after very enthusiastic encounters were unreachable for further contact, some were interested only after their needs would be met and there were political and individual matters involved.

After three years of contact, there was still no concrete development and no access to the students. The introduction of music technology in the education became a side note of a bigger challenge. In order to fully incorporate this new field the curricula needed to be adapted in a way that would ensure that the education supports the young generations in their development rather than adapting the youth to the existing education. During the preparation phase it became evident that the only way to initiate such changes was to actually avoid these institutions altogether.

3. MAKING THE FIRST STEP

The decision to avoid the institutions brought things back to perspective. The problem with the approach during the preparation phase was that there was no access to the young students.

3.1. Generation Einstein

The new generation, born after 1988, has been named differently by different scientists, for example the copy-paste generation, or the grab-and-mix generation [2]. A more positive title is Generation Einstein [2]. This generation recognizes that the world consists of an enormous amount of, most of the time, contradictory information; that not everything is what it seems; that a lot is possible and that the world is fragmented and pluralistic [2]. The Generation Einstein perceives and processes information much faster than we are used to. It is a generation of doers and the education in many European countries, not only in Macedonia, is still shaped for doers. So I decided to organize a summer project in the form of an experimental meeting where music technology would be introduced to the technology savvy young generation, full of artistic curiosity but lacking the hands-on experience. In other words, I proposed a blind date between music and technology.

3.2. Opening doors

In 2009, the very first step was made possible independently of any music or education institution in Macedonia, with no financial support from the country. For this very first opportunity I decided to organize a summer project with the support of the European Commission, organized by the Zoey Foundation for Arts and Culture, in cooperation with the Utrecht School of Music and Technology and the Macedonian NGO Public Room in the form of an experimental music meeting where, in order to introduce music technology, it would be met with an equally present opponent: tradition.

3.3. Technology vs. Tradition

“Will technology destroy our tradition?” was a question that came up regularly during the project. During this project, tradition and technology had very specific meanings.

In the context of this project, tradition stood for tradi tion in the ways of thinking and perceiving things, as well as traditional ways of education; tradition in idea forming; instrument building and playing; in ways of approaching and dealing with musical problems. Specifically traditional music, folklore or national heritage is not what is meant by tradition in this context.

Technology, in the context of the project, stood primarily for a state of mind. Being open minded, accepting changes, new points of view, problem solving and solution searching, seeking methods of organization in order to achieve a certain goal. Technology also stood for recognizing the potential of a group rather than an individual and working together.

4. FACING THE CHALLENGE

What would happen if we brought together two completely different groups of young people that are the exact opposite of each other? One group of ten music technologists with relatively little experience with acoustic instruments, and the other group of ten music performers, that play classical instruments and have no previous experience with electronics. Just to spice things up a little, we add huge cultural differences to the pot: one group comes from The Netherlands and the other from Macedonia. To prevent things from getting dull, we give them ten days to overcome their cultural differences, get to know each other, make mixed groups that will work together, come up with ideas, compose, prepare and perform the results in front of an international audience.

4.1. Opposites Attract – Profile of participants

Equality among the participants was important to the outcome of the project. Besides the division between music technologists and classical instrumentalists, each group contained a generation gap in a way that brings diversity in knowledge. On this very first meeting, there were no traditionally educated composers invited.

4.2. Dutch participants

The Dutch group of music technologists had Audio Designers, Sound Designers as well as music production students. These young people are used to working very fast; being finished before the deadline if possible; having a carefully structured daily schedule that includes working and resting hours; having results that would impress, but not necessarily by how deep the idea was explored; they are used to improvising. In their working process, the Dutch students are very much used to thinking only within the possibilities (limitations) of the computer program they are using. Except for a few pupils, the professional music education of the Dutch students had lasted no longer than 3-4 years at the time of the realization of the project.

4.2.1. Cultural characteristics relevant to the project

The Dutch group has no experience with authority in education. The youngsters have no problem saying they don’t understand something, as in the context of this group it implies that one is eager to learn, they speak their mind and their teachers are counting on that.

4.3. Macedonian participants

The Macedonian group brought the instruments: violin, viola, cello, flute, goblet drum (tarabuka), classic guitar, bass guitar, kaval (Macedonian traditional instrument) and a self-made didgeridoo. Almost every participant of this group has at least one musical instrument at the time of the project realization. They had a stable education basis and possessed excellence in their instrument. However, as instrumentalists with a strictly classical music education, they had no prior experience in improvising and were not comfortable when asked to. There was no clear distinction between working and resting hours, busy with the piece almost at any time; they appeared to be slower in decision-making; not necessarily concerned by the end result but pointing on details during the working process. These students are not used to thinking from or within limitations of any computer program but are used to working from idea to realization.

4.3.1. Cultural characteristics relevant to the project

The Macedonian group comes from a surrounding in which it is not accepted not to know something, as this implies one hasn’t studied hard enough; it is considered shameful to openly say so. There was also a clear issue with authority, as these students didn’t feel free to articulate their opinions to the Dutch lecturers guiding them.

4.4. Objectives

The general aim was to initiate changes in the curricula of music education in Macedonia in order to create grounds for the introduction and implementation of music technology. In order to achieve this, several objectives were set:

• Initiate cultural cooperation and knowledge exchange between the two countries;
• Create a platform where young people with different musical and cultural backgrounds can meet, experiment, create and learn from each other;
• Support the youth in Macedonia in their quest for knowledge;
• Connect the participants to existing musical networks and promote the project to create possibilities for the participants to promote their work internationally.

Due to difficulties within the institutional cooperation, I started with a pilot project in the form of a summer meeting.

5. WORKING METHODS

In July 2009 the two groups came together for the first experimental meeting in Ohrid, Macedonia.
5.1. The rules of the project
During the project there were a few ground rules that helped guide the development of the project. Choose a partner to work with, get out of the comfort zone, take part in the morning conversations called Distilled Experiences, be open for the work-in-progress presentations and take part in the evening concerts.
The participants were to make mixed groups Dutch/Macedonian and compose together a piece for acoustic instrument(s) and live electronics. The general theme the participants had was to reflect on this cultural encounter in their newly composed musical pieces.

5.2. Guidance
The project introduced the participants to a set of contemporary techniques and concepts behind improvisation, technology-based composition and performance. The participants were free to choose by themselves what tools they were going to use, but mainly they were working with Max/MSP or SuperCollider. There was plenty of guidance for those needing it: one or two lectures experienced in composing with students in various teaching methods. The four professionals were carefully chosen so that their knowledge could cover everything from classical orchestration, instrumentation, to programming and multimedia project based composing.

5.3. Challenges during the realization
The project brought many challenges as both the Dutch and Macedonian group had expectations that were not fulfilled. Most of the challenges occurred in the first few days. There were no lessons, no workshops and no one that would tell the participants what to do. They would have to figure that out on their own.

5.3.1. There were no lessons
The Macedonian group expected to have a passive lesson environment. They had their paper notebooks handy and were ready to learn! Some were disappointed and a few even a bit angry when they realized that we were not going to teach (not like that). The lecturers were not there to make sure they give the best possible interpretation of the software that was being used, thinking within the possibilities of software was not an option. They became better at setting creative borders. They learned that having all the equipment is only good if you know what to do with it.

6. WHAT WAS LEARNED
6.1. Dutch group
The Dutch group influenced by the Macedonian rhythms, modes and melodies as well as the process of making music. By closely working with the musicians, they enriched their theoretical knowledge and learned the instruments, how they are built, different ways to produce sound, different ways to play with the instruments and electronics. They also learned to make scores that would work both for the electronics and the instruments. An important moment was that they were learning to explore their ideas deeper, question choices and decisions and be very critical. Faced with colleagues who do not approach their ideas from the construction of the piece that was being used, thinking within the possibilities of software was not an option. They became better at setting creative borders. They learned that having all the equipment is only good if you know what to do with it.

6.2. Macedonian group
6.2.1. Resonances
During the project, the Macedonian musicians learned to improve their Western-European superiority and get away with their fast-result-seeking way of work, it very much failed. The Macedonian group, on the other hand, insisted on taking the time to experience, feel and live the music they were making. But there simply was no time to do that. Since music was the only asset they could hold on to, and the only thing they had in common, with the deadline racing on them, they could only thing they could do: listen to each other, find middle grounds and create a synergy of the knowledge they had as a team.

6.2.2. What the Macedonians learned
The Macedonian group, on the other hand, insisted on taking the time to experience, feel and live the music they were making. But there simply was no time to do that. Since music was the only asset they could hold on to, and the only thing they had in common, with the deadline racing on them, they could only thing they could do: listen to each other, find middle grounds and create a synergy of the knowledge they had as a team.

6.2.3. Analysing the Macedonian group
Analyzing the Macedonian group, it was noticed that working with students that already had some kind of experience in working with instruments and live electronics was easier in the sense that, we could also talk about esthetics. You can find students who have more basic knowledge in order to develop ideas. The same applies for students that had experience working with any other art field, as the younger students, who have no previous communication problems with the players. The Macedonian group did not have any prior experience with electronics or other art fields, which made the group balanced.

The questions the participants would ask and the kind of guidance they would ask for provided the lecturers with valuable information about their learning and working processes. The way they would use the information showed their level of experience. It also showed how these young people were educated, what kind of program they were following, which subjects had impact and how they implemented the knowledge.

6.4. Educating the Audiences
The project was shaped to ensure maximum exposure, which meant having the activities take place each day in a different location. This made the project highly visible and easily approachable. While working, the young participants were being approached all the time wondering what they were doing with all that equipment on the beach or by the lake. Then they would stay and listen a bit, ask questions. It was the same during the performances in the evenings.

Getting in touch with the local people and international tourists made the youngster think about the way they would explain their ideas to people that are not musicians. They were explaining their methods; inspirations and techniques used as well as answering questions regarding the use of technology in contemporary music to non-musicians of all ages. This motivated people to attend the concerts.

7. A CHAIN REACTION
The way the project was structured resulted in the young participants learning more than they or the professional team could anticipate they would. The objectives of the project were met and the young participants from Macedonia were actively supporting the further developments.
They had experienced another way of learning and touched another field in music that they wanted to explore more and develop in. The reactions of the participants, especially of the young people that did not participate, stressed the need for music technology and for repeating the project. Because of these positive reactions, the doors were open to several schools. This resulted in a new project. In 2010, I organized Urban Resonances, a project that brought several lecturers from the Netherlands to the Netherlands to the Macedonia and five music high schools in Macedonia to give con
certs and workshops. The interest of students was so high that all of the presentations were held in completely crowded spaces.

In 2011 the second meeting between Dutch and Macedonian musicians took place during the summer. This time the project was also open to composers to take part and we had guest lecturers from visual arts and architecture, working in close collaboration with the School of Music and the music high school in Skopje were also at the project. The doors were open.

Between September and December 2011 Shifting Values took place at the Music High School in Skopje. The kick-off presentation was sold out and students were even standing in the hallway.

In September, 2012 the School of music in Skopje in cooperation with the Utrecht School of Music and Technology will be ready to welcome all these interested students with their new curriculum for music and technology.

10. REFERENCES

[1] Abbing, H. Why are artists poor?: the exceptional economy of the arts. Amsterdam University Press, Amsterdam, 2002

8. FOR THE FUTURE

In this work, I presented a bottom-up approach to the introduction of music technology in a traditionally orientated music education system. Previous attempts that took the traditional path failed to reach the students. The paradigm presented here, based on non-formal learning had positive responses. The project brought the Utrecht School of Music and Technology and the School of Music and Technology in Macedonia in close cooperation. These two institutions are now structuring their cooperation for the future while shaping the new curriculum for music and technology. The positive outcome of this project might also trigger other art schools to re-evaluate the upbringing of their young talents.

The summer projects turned into very stimulating labs where talented musicians tested their knowledge, ideas, made music, performed together and experimented with new things. In the years to come, I expect to broaden the cultural sphere by inviting lecturers and participants from other countries as well. There are plans to introduce a new art discipline each year to the live electronic’s dance, visual arts, game design, among others. These projects help equip young people not only with the ability to adapt to change but also with the ability to shape the direction of change.

9. ACKNOWLEDGEMENTS

The author thanks Marcelo F. Caetano for the help while writing this paper, Gerard van Wolferen en Hans Timmermans for their support, help, advice and guidance throughout the years from idea to realization. The Zoey Foundation for Arts and Culture supported the research, organization and realization of the cooperation projects with the help of Utrecht School of Music and Technology.

10. REFERENCES

Michael Edwards
Music
University of Edinburgh

ABSTRACT

This article introduces a new open-source algorithmic composition system, slippery chicken, which enables a top-down approach to musical composition. Specific techniques in slippery chicken are introduced along with examples of their usage in the author’s compositions. The software was originally tailor-made to encapsulate the author’s personal composition techniques, however many general-purpose algorithmic composition tools have been programmed that should be useful to a range of composers. The main goal of the project is to facilitate a melding of electronic and instrumental sound worlds, not just at the sonic but at the structural level. Techniques for the innovative combination of rhythm and pitch data—arguably one of the most difficult aspects of making convincing musical algorithms—are also offered. The software was developed by the author in the Common Lisp Object System and released as open-source software in May 2012; see http://www.michael-edwards.org/sc.

1. INTRODUCTION

"Formally, when one worked alone, at a given point a decision was made, and one went in one direction rather than another; whereas, in the case of working with another person and with computer facilities, the need to work as though decisions were scarce—at though you had to limit yourself to one idea—is no longer pressing. It’s a change from the influences of scarcity or economy to the influences of abundance and—I’d be willing to say—waste.” (John Cage, quoted in [1])

The potential for software algorithms to enrich our musical culture has been established, in the 50+ years since such techniques were first introduced, by personalities as diverse as Hiller, Xenakis, Cage, and Eno. Algorithmic composition usually involves the use of a set of step-by-step procedures, most often encapsulated in software routines, to create music. The power of such systems is, arguably, still not fully understood or deeply investigated by the majority of musicians and composers, whether highly trained or not. Indeed, in the author’s experience, a lot of the prejudice algorithmic composition pioneer

Hiller suffered under [3] is still with us today. But there are clearly many more to be found in algorithmic composition, as the expression of compositional ideas in software often leads to unexpected and surprisingly new, exciting results, and these can seldom be achieved through traditional means. Algorithmic composition techniques can thus play a vital and energising role in the development of modern music and new art disciplines.

slippery chicken is an open-source, specialised algorithmic composition program written in the general programming language Common Lisp and its object-oriented extension, the Common Lisp Object System (CLOS). Work on slippery chicken has been ongoing since 2000. By specialised as opposed to generalised, it is meant that the software was originally tailor-made to encapsulate the author’s personal composition techniques and to suit his own compositional needs and goals. As the software has developed however, many general-purpose algorithmic composition tools have been programmed that should be useful to a range of composers. The system does not produce music of any particular aesthetic strain—for example, although not programmed to generate tonal music the system is quite capable of producing it. But if it is to be used to generate complete pieces it does prescribe a certain specialised approach; this will be described below.

slippery chicken has no graphical user interface and there are no plans to make one. Whilst it is clear that this will be off-putting to some, there are many benefits to interacting with such a system through the programming language it was created in, not least of which is the inflexibility that such an approach implies. As the computer science adage goes, "When using WYSIWYG [What You See Is What You Get] systems, What You See Is All You’ll Ever Get." 3

The algorithmic system in slippery chicken is mainly deterministic but also includes stochastic elements if desired. 4 It has been used to create musical structure for pieces since its inception and for several years now has been at the stage where it can generate, in one pass, complete musical scores for traditional instruments. It can also, with the same data used to generate those scores, write sound files using samples, or MIDI file realisations of the instrumental

1 Preparation of this software for open-source public release was supported by the UK’s Arts & Humanities Research Council [grant number AH/S504321].

2 Though the composer Clarence Barlow would perhaps disagree with this, as he states that in his algorithmic works "he would obtain the same results without the help of a computer" [16, 49].

3 Despite much internet attribution, Donald E. Knuth has confirmed to the author that this quotation did not stem from him.

4 Including, but not limited to, permutations in both a deterministic and random order. slippery chicken utilises the use of fixed-seed randomness, so that repeatable results may be generated. For a discussion of the usefulness of such see [5, 64].
score. The project's main aim is to facilitate a melding of electronic and instrumental sound worlds, not just at the sonic but also at the structural level.\(^1\) Hence certain criteria are imposed on one medium (for instance audio slicing and looping) are transferred to another (the slicing up of notated musical phrases and the instigation of sub-phrase loops, for example). Techniques that can map these one medium through rhythm and pitch data—arguably one of the most difficult aspects of making convincing musical algorithms—have been inferred.

The system includes but is not mainly concerned with the automation of some of the more laborious aspects of composition—transposition, harmonic and rhythmic manipulation for example—and thus facilitates and encourages experimentation with musical data before committing to final forms. By generating music data algorithmically, independent of output format, structures become available for use in the preparation of digital and notated music—in the digital case, particularly for the generation of parameters for the digital synthesis and signal processing language Common Lisp Music (CLM) [13]. The programme in nascent form was first used for the generation of the tape part of a piece by the author for solo violin, ensemble, and stereo tape: slippery when wet\(^2\). Its effectiveness in sonically and structurally integrating the instrumental and digital resources in that piece provided the impetus to pursue the idea further.

What slippery chicken is focused upon then is harnessing the automated management of Common Lisp and CLOS to achieve a top-down approach to musical composition: defining, ordering, combining, and manipulating rhythmic, pitch, sound file, instrumental, and dynamic information, etc., into complete pieces of music or structures ready for further processing within or without the system. The outline of the program is in the form of:

- **MIDI files**, generated with the help of Common Music's (CM 2.6.0 [16]) MIDI routines, and common music notation information that facilitates reading into music notation software such as Sibelius or Finale.
- **Music Score**
- Postscript file generated by interfacing with Common Music Notation (CMN) [14], and thus allowing the algorithmic use of arbitrary symbols, note heads, etc., for the encapsulation of extended instrumental techniques that are difficult or impossible to encode in MIDI
- **Lilypond input text files** [10], with similar advantages to CMN, but more scope for post-generation intervention

\(^1\) Though it can be used purely for instrumental or electronic music also.

2 http://www.southtomp.com/work.php?src=slide5

\(^2\) Sound files, using samples driven by a custom, multi-channel CLM instrument.

The approach to algorithmic composition here is sequence or phrase-based (though this should not be confused with MIDI sequencing). In its most basic form, we define a certain number—a palette, in slippery chicken terms—of rhythmic phrases and pitch sets, that can be combined into a combination of rhythmic and pitch data arguably one of the most difficult aspects of making convincing musical algorithms—have been inferred.

One of the more challenging aspects of algorithmic composition—at least in pieces where there should be a semblance of phrases formed of horizontally connected notes—is the satisfactory combination of rhythms with pitches.\(^3\) For instance, if we were to place a rhythmic phrase without pitch information in front of a trained composer, she could not sing or play back a number of pitch contours that would be likely to work with that idea further.

The implication of this is that not all pitch contours would work with the given rhythms, and that the contours would be influenced by the given rhythms, even if several solutions were available and the general shape of a line were more important than the exact pitches chosen. The reverse is also true: if the composer were offered a pitch contour without rhythm then, the selected rhythms would be influenced by the shape of the line. The process of matching one to the other is complex and idiosyncratic, dependant on culture, musical experience, taste, etc. Thus formalisation of this process is difficult.

slippery chicken's solution is to allow for the provision of an arbitrary number of pitch sequences (perhaps even algorithmically) to each rhythm sequence. The pitch sequences consist of a list of simple integers, one for each attacked rhythm (i.e. not for tied notes), over a user-defined range but where, for example, 2 would indicate a higher pitch than 1.

When rhythm sequences have been mapped to ensemble players, and pitch sets (harmonic material) to rhythm sequences, it is then a matter of slippery chicken selecting pitches from the current pitch set and pitch sequence. The algorithm will of course only choose notes that are within each instrument’s range. A hierarchy to specify which instrument is given priority when the algorithm is assigning notes to instruments can also be defined, as an algorithmic attempt will be made to use as many notes of the set as possible, spreading them out amongst the instruments in the ensemble.\(^4\)

\(^3\) There are of course musical systems which decouple the organisation of pitch and rhythm material: medieval isorhythmic motets, initial sonorities and, and Cope's music compositions like I Ching, for example. But the interdependence of these two parameters continues to exist in a wide range.

\(^4\) Though there is a preference for selecting unused pitches from the set, if this is not possible then previously used pitches will be added until the number of notes available are as close as possible to the pitch sequence's initial number.

\(\text{Figure 1. pitch set example used in Figure 2.}\)

Simple in concept at least then, the basic procedure for using slippery chicken—any part of which may be algorithmically or manually delivered—can be summed up as defining:

1. the instruments’:
   a. ranges
   b. transpositions
   c. chord selection functions (if applicable)
   d. microtonal potential
   e. unplayable notes (e.g. microtones)
2. the instrument changes for individual players (e.g. flue to piccolo)
3. the set palette (harmonic fields) that the piece will use
4. the rhythm sequence palette
5. the set map: sets onto sequences
6. the rhythm sequence map: sequences onto instruments
7. the tempo maps
8. the set limits: for the whole piece and/or instruments

2. 2. GENERAL FEATURES

Although slippery chicken can of course perform many labour-intensive tasks (such as score writing, transposition, and, through its fundamental algorithms, pitch selection and sequence compiling), its main attraction is not, as the general computer myth would have it, as a labour saving system. For composers, arguably the primary benefit of this project and of algorithmic composition in general is that the encapsulation and expression of compositional structure in software often involves a form of practical experimentation which can lead to surprisingly new, rewarding, and exciting results. Randomness is not the issue here: many deterministic and, upon initial examination, seemingly predictable algorithms lead through the combination of a few steps to unimaginable music.

slippery chicken straddles the two poles of compositional process formalisation and what some would consider a relinquishing of compositional autonomy. With one of its main (but not unique) features being a bridging solution—leading composers with little algorithmic experience into the world of music computing, and bringing computer generation techniques to the world of instrumental music—slippery chicken offers a structured method as opposed to a composition software library. Clearly, any algorithmic composition system demands a certain, often idiosyncratic approach: it is a question of to what degree. Some systems are more open than others, and are therefore more akin to a software library: SuperCollider [8], Pure Data [11], Max/MSP [12], and systems made with the latter such as the Real Time Composition Library [6], for example. Others are more specialised: FrasEx [7], David Cope's Experiments in Musical Intelligence [4], and Bernard Bol's Bol Processor [12].

slippery chicken is more akin to the specialised group. This is clearly its greatest advantage: complete pieces of music can be generated with relatively little input from the user. But, individualistic as they most often are, many composers won't find the method to their taste. These may still consider some of the slippery chicken classes, algorithms, and methods attractive. Many of the techniques can be applied without being coerced into the map/palette approach to generating complete pieces, so the package could be employed more as a software tool library also.

Because of its delivery format as an open-source, object-oriented Lisp package, slippery chicken is infinitely extensible. It can be used in its simplest form by entering the necessary musical data in lists and allowing the system to generate a complete piece of music. Or it can generate pieces, sections, phrases, etc., by making more sophisticated use of its internal generative classes and/or user-programmed extensions and subclasses. The generated data structures can also
be altered through a host of included editing functions and methods. Here is where the tension between idealism and pragmatism found at the very beginnings of computer music and algorithmic composition and discussed in [5,62-63] arises. To summarize briefly: Lejaren Hiller believed that if the output of the algorithm is deemed deficient, then the programme should be modified, and the output regenerated; whereas Koenig and Xenakis took a more practical approach, treating the output of their algorithms to transcription, modification, and elaboration. Despite its perhaps idealistic goal of generating complete and coherent musical works, with its collection of internal data, its 출력한 functions, slippery chicken remains solid pragmatically. In its pre-2006 form, before the introduction of the pitch-selection algorithm, slippery chicken was arguably more in the camp of computer-aided composition than that of algorithmic composition, to use Monno’s distinction [9]. It is now more firmly in the algorithmic composition camp, with the potential to act entirely as a digital composition assistant if so desired.

3. SLIPPERY CHICKEN TECHNIQUES AND ALGORITHMS

3.1. With CLM

When writing sound files with a custom, 4-channel CLM instrument, slippery chicken uses essentially the same data as used for generating MIDI and score files. Notation details such as ties, dots, clefs, etc., have no bearing upon sound file generation of course, and are thus ignored. In order to generate sound files, a sound file palette in the slippery chicken object is used. This stores data associated with groups of sound files; they will be cycled through during the algorithmic processing. There can be any number of sound file groups, allowing the musical data to be applied to several different categories of sound. This can create interesting variations of recognizably similar material. Moreover, the use of the same musical data for the generation of scores and sound files in this fashion creates, when they are combined in hybrid musical works, a kind of melding of electronic and instrumental sound worlds mentioned in the overview: striking pitch and rhythm structures will be audibly related whether presented by acoustic instruments or in sound files. The chronological placement and mixing in of sound files with CLM is triggered according to the start times of notes in the score. This may be scaled up or down for a faster or slower rendition of the score. Upward and downward transpositions, from a user-defined zero-transposition point, will also be carried out if the user so wishes, in accordance with the pitches’ deviation from the zero-transposition point. This may also be scaled, as desired. slippery chicken takes advantage here of the high-quality transposition algorithm of CLM; this convolves its input with a sinc function. Duration may also be scaled so as to create a thicker texture through doubling of input sound file lengths allow. Start time within the input sound file can also be automatically incremented upon reuse, in order to avoid repetitions of opening data; if the algorithm is long enough, it will slowly increment the duration of each of the sound files as they are processed in turn.

Engagement with slippery chicken in combination with CLM has become a self-sufficient project, one that has, apart from the author’s main instrument-with-computing works, generated a collection of short pieces in the form of downloadable sound files. Contrary to traditional electroacoustic studio work (which can often be thought of as sound sculptures) the approach here is to generate perhaps hundreds of sound files automatically from a given sound file palette and set of compositional data. Sound file selection then becomes the main activity when using the output of slippery chicken, in keeping with the Cage quotation at the beginning of the overview. The best results of the algorithms (with post-output editing) are to be found on the internet as short but complete pieces.3

3.2. Fibonacci Transitions

Transitions between different musical sections or states have been an important characteristic of Western Classical music for centuries. For instance, in the eighteenth and nineteenth centuries, we have transitions between the first and second subject groups in sonata form; or in the twentieth century, in the textural morphing of Ligeti’s micro-polypitchon structures in, for example, Atmospheres (1961). Transition strategies are basically of two main forms: the procession algorithm4 and Fibonacci Transitions. Transitions in slippery chicken are aimed at the development and variation of musical material at the macrostructural level. For example, this could be used to intersperse a new audio segment into a sample loop, to gradually transform the repetition of one rhythm sequence into another; or to transition between different harmonic fields, etc.

In Fibonacci Transitions, new elements are ‘folded into’ existing repeating elements a number of repetitions determined by Fibonacci numbers5. Such transitions are available in simple loops into the score domain. Looping and chopping of notated rhythms was first applied in the author’s cheat sheet;6 for solo electric guitar and eight-piece overlaps (if input sound file lengths allow). The progression through the 100 loop points per voice is controlled by a modified Fibonacci Transitions call. The modification is a subroutine called remix-in. This inserts earlier segments between adjacent segments so as to avoid a purely binary opposition of rhythmic materials and thus enrich the musical development and associations. It also provides more structural cohesion, investigating previous segments’ rhythmic and metrical effects in the new context. The Fibonacci Transition in cheat sheet creates 277 segments per voice, moving from the beginning of the first bar to the end of the fifth of the original material shown in Figure 5. It returns numbered references ranging from 1 to 100:1-10 refer to the loop points in the first crotchet (quarter note); similarly 11-20 refer to the loop points in the second crotchet, etc. The beginning and end of the transition is shown in Figure 7.

The essentially DSP-inspired looping technique is applied to conventionally notated musical material by dividing the five bars of four-part counterpoint shown in Figure 5 represent all the rhythmic and contrapuntal material available for this 1167 bar piece.

3 http://www.sumtone.com/work.php?workid=131
4 http://sites.ace.ed.ac.uk/algocomp/2011/07/01/you-are-coming-
5 Note that the number of items returned will always correspond to the first argument. Varying repetitions of the lower order alternations will fill any shortfall; these are labeled ‘t/f’ in Figure 3.
6 http://www.sumtone.com/work.php?workid=112

The progression through the 100 loop points per voice is controlled by a modified Fibonacci Transitions call. The modification is a subroutine called remix-in. This inserts earlier segments between adjacent segments so as to avoid a purely binary opposition of rhythmic materials and thus enrich the musical development and associations. It also provides more structural cohesion, investigating previous segments’ rhythmic and metrical effects in the new context. The Fibonacci Transition in cheat sheet creates 277 segments per voice, moving from the beginning of the first bar to the end of the fifth of the original material shown in Figure 5. It returns numbered references ranging from 1 to 100:1-10 refer to the loop points in the first crotchet (quarter note); similarly 11-20 refer to the loop points in the second crotchet, etc. The beginning and end of the transition is shown in Figure 7.

Figure 3. Simple two-datum Fibonacci Transition.

Figure 4. Multi-datum Fibonacci Transition.

Figure 5. Original rhythm sequence palette for the author’s cheat sheet.
Taking the opening flute and clarinet parts, and comparing the score with the original four-part counterpoint (line 1A in Figure 5), we can see how this develops in Figure 8. The rhythms and meter were doubled for ease of reading.

Figure 8. Rhythmic loop slice mapping in the flute and clarinet parts at the beginning of the author’s cheat sheet.

3.4. Transitioning Lindenmayer Systems

Lindenmayer Systems (or L-Systems) are in their simplest form deterministic. For musical composition, this class of algorithms is often preferable to its stochastic counterpart due to the repeatability of results when regenerating material (regeneration typically being necessary in the generate-modify-regenerate iterative process of algorithmic composition). See [5, 64] for further discussion of this issue and for examples of a basic L-System.

One of the attractions of L-Systems is self-similarity; see Figure 10 for an illustration of this. The generated numbers (or any data type) can of course be applied to any musical parameter or material.

A slippery chicken development, Transitioning L-Systems use data returned by an L-System as lookup indices into a substitution table. This table may contain any data, including further references to other data structures (e.g. rhythm sequence palettes). The result of the substitution depends on transitions between an arbitrary number of related but perhaps developing material—such relationships are envisaged though they are not of course enforced. The transitions are created by the L-Systems. Each of the transitions may also contain an arbitrary number of data points in a list; these will be cycled through each time a particular transition is returned.

Returning to cheat sheet, slippery chicken’s L-System implementation is used in three main ways: as a straightforward cycling mechanism; as a simple L-System without transitions; and as a Transitioning L-System. The latter two uses in this musical context will now be discussed to illustrate their properties.

Figure 9. Guitar tuning (scordatura) for the author’s cheat sheet.

A threefold process created the pitch sets:

1) 6 guitar chords were chosen by ear. These had fingerings: (4 2 1 3) (4 1 2 3) (4 4 2 1) (4 3 1 2). The sequencing of these was organised by a simple L-sequence: there are no transitions here but self-similar patterns do emerge (Figure 10).

Figure 10. Self-similarity in L-System results.

2) Whether to play these chords on the four lowest, four middle, or four highest strings (using the first finger fret as a barre on the remaining strings in each case) is determined by a transitioning L-sequence (II.

When this threefold process is combined, the result is a list of chord references each consisting of three elements:

1. which four of the six strings to finger (lower four, middle four, or highest four)
2. the first finger fret (1-15)
3. the fingering, an index (1-6) into the fingering list ((4 2 1 3) (4 1 2 3) (4 3 1 2) (4 4 2 1) (4 3 1 2) (4 1 2 3))

The beginning and end of the results of this process are shown in Figure 13.

The generated numbers (or any data type) can of course be applied to any musical parameter or material.

A threefold process created the pitch sets:

1) 6 guitar chords were chosen by ear. These had fingerings: (4 2 1 3) (4 1 2 3) (4 4 2 1) (4 3 1 2). The sequencing of these was organised by a simple L-sequence: there are no transitions here but self-similar patterns do emerge (Figure 10).

Figure 11. Guitar chords and ensemble extensions for the author’s cheat sheet.

A threefold process created the pitch sets:

1) 6 guitar chords were chosen by ear. These had fingerings: (4 2 1 3) (4 1 2 3) (4 4 2 1) (4 3 1 2). The sequencing of these was organised by a simple L-sequence: there are no transitions here but self-similar patterns do emerge (Figure 10).

Figure 12. First finger fret selection curve for the author’s cheat sheet. The x-axis is scaled automatically to fit the 2177 chords.

When this threefold process is combined, the result is a list of chord references each consisting of three elements:

1. which four of the six strings to finger (lower four, middle four, or highest four)
2. the first finger fret (1-15)
3. the fingering, an index (1-6) into the fingering list ((4 2 1 3) (4 1 2 3) (4 3 1 2) (4 4 2 1) (4 3 1 2) (4 1 2 3))

The beginning and end of the results of this process are shown in Figure 13.

The generated numbers (or any data type) can of course be applied to any musical parameter or material.

A threefold process created the pitch sets:

1) 6 guitar chords were chosen by ear. These had fingerings: (4 2 1 3) (4 1 2 3) (4 4 2 1) (4 3 1 2). The sequencing of these was organised by a simple L-sequence: there are no transitions here but self-similar patterns do emerge (Figure 10).

Figure 13. Results of the Transitioning L-System combined with the simple L-System and the fret curve for the generation of the chord sequence in the author’s cheat sheet.

See Figure 14 for some results of this process using the six guitar fingering patterns discussed.

4. CONCLUSION

Though focussed mainly on the algorithmic production of complete pieces for instruments and computer, the slippery chicken package includes several unique approaches to generating musical structure that may be used in other contexts. Its top-down approach to compositional organisation offers considerable potential for explorative, iterative development, freeing the composer from the commitment to a single labour-intensive path. This can lead, if so desired, to unimagined aesthetic realms with relative ease. Its integration of score and MIDI file writing, along with the use of the author’s own musical data for the generation of sample-driven sound files, strengthens the audible structural links between the often-disparate worlds of acoustic and electronic composition. Its approach to various transition
strategies can be employed towards evolving musical structures out of relatively little, and therefore coherent, musical material. Its release as open-source, object-oriented Common Lisp code encourages further development and extensions on the part of the user.

5. REFERENCES


MANUSCORE: MUSIC NOTATION-BASED COMPUTER ASSISTED COMPOSITION

James B. Maxwell, Arne Eigenfeldt, Philippe Pasquier

Simon Fraser University
SCA/SIAT, Vancouver, Canada

jbmmaxwel@sfu.ca

ABSTRACT

ManuScore is a music notation-based, interactive music composition application, backed by a cognitively-inspired music learning and generation system. In this paper we outline its various functions, describe an applied composition study using the software, and give results from a study of listener evaluation of the music composed during the composition study. The listener study was conducted at a chamber music concert featuring a mixed programme of human-composed, machine-composed, and computer-assisted works.

1. INTRODUCTION

Otto Laske’s notion of “composition theory” [6] focused on three fundamental principles: 1) competence: knowledge of the materials and syntax of music, required for the conception of musical ideas; 2) performance: the practical application of accumulated musical knowledge (competence) to create musical forms; and 3) the task environment: the field of action in which performance draws on competence for the invention of musical works. In the context of Computer-Assisted Composition (CAC), the task environment is embodied by a computer and its hardware/software. When this role is assigned to something as pliable as computer software, composers are suddenly given the capacity to tailor the task environment to their particular needs, in a manner not previously possible. Laske felt that, in a computer-based task environment, composers could access the virtual music of their imaginations in a manner unbounded by musical traditions. He identified the process of conception, design, implementation, and production of the task environment, and its iterative development throughout the compositional process, as the “compositional life cycle” [9]. It was Laske’s feeling that software developed through a compositional life cycle could gradually begin to embody the musical knowledge of the composer in an explicit, analyzable, and extensible way.

Early CAC tools like Koenig’s PROJECT systems [8], or Trux’s POD systems [19], took a top-down approach to CAC, in which high-level concepts were expressed parametrically, or graphically, and the software was responsible for generating numerical representations, or electronically synthesized performances, of musical output. Two important observations can be made about such systems: 1) They are not corpus-based, and thus will not generally maintain an explicit connection with the user’s musical past, and 2) They deal with musical concepts at a high level of abstraction, and thus introduce significant non-linearities into the compositional process—the, by substituting numerical representations for sounds, they separate composition from the act of listening (an idea that Laske very much supported, believing that such a division could lead to “unbounded” musical invention).

The vast majority of commercial CAC packages (conventional Digital-Audio Workstations (DAWs) and MIDI sequencers) are essentially bottom-up systems, which fulfill the basic tasks of recording and manipulating musical performances, or transcribing and electronically ‘performing’ musical scores. Although applications of this type are often equipped with extensive feature-sets, directed toward simplifying and streamlining the workflow for such tasks, their fundamental purpose is to record.

Making up the middle-ground, there are an increasing variety of CAC tools which propose different (and often quite novel) forms of musical representation, and introduce varying degrees of interactivity into the compositional process [23, 20, 13] etc.). Among this class one could also include the increasing variety of music programming languages, graphical or otherwise ([1], [17, 21], etc.), which offer a potentially infinite variety of CAC tools-to-be, and propose potentially infinite mixtures of top-down/bottom-up control.

For the bottom-up tools, competence and performance are essentially unchanged from the traditional requirement of music-theoretical knowledge, instrumental performance ability, skill in instrumental music transcription, and so on. With the top-down tools, the demand placed on competence is shifted (and potentially increased) by the emphasis on abstraction, while performance becomes focused on the interpretation of numerical representations of musical materials [7], and/or on the comprehension of metaphorized descriptions of musical concepts: “density”, “timbral trajectory”, and so on [19].

2. MOTIVATIONS BEHIND MANUSCORE

Our initial goal in designing ManuScore was to create a music notation-based CAC tool for music-literate com-
posers, who might already possess a developed musical language and bottom-up compositional practice, but were interested in exploring a top-down interaction with their musical ideas. In contrast to Laske’s goal of freeing composers from musical tradition, we wanted the system to acknowledge the user’s existing musical practice, so that working in ManuScore need not impose any dramatic change in a composer’s musical language or compositional output. The intention was to create a task environment to augment a composer’s practice, not necessarily to dramatically alter it, and certainly not to completely automate it.

In this sense, the system could be aligned with Cope’s CUE software [5], which draws from a music recontextualization database to offer “continuations” and developments of musical ideas introduced by the user.

Adding to this general conception of a non-interfering, interactive CAC tool with corpus-based generation, we were also interested in the notion of “object-oriented” composition [15]. Our conception of object-orientation focuses on the notational aspects of musical ideas. That is, we are interested in what can be captured in a musical score, what the various structures on the score’s surface represent to the composer (i.e., what is their musical “objecthood”), and how these structures might “inherit” from one another in the developing composition. In this sense, an object in ManuScore is somewhat analogous to a “gestalt” in the music perception and cognition literature [18], i.e., it is an identifiable, holistic item, or concept. To whatever degree possible, we wanted to help composers explore musical ideas as gestures, and to represent them accordingly in their compositional tool environment. This approach connects ManuScore to programs like PatchWork (or PWGL [10]) and OpenMusic [1], which also help composers interact directly with musical concepts, though our focus on notational elements (i.e., leaving aside numerical operations) in the user interface clearly sets ManuScore apart.

The generative capabilities of ManuScore, in its current version, are focused on the notion of ‘continuation’: i.e., of extending musical fragments introduced by the user. Generation is also currently monophonic. However, our goal with ManuScore is to implement real-time, interactive generation, so that musical ideas may also be explored through listening and improvisation, not just through manipulation of scored musical objects.

3. MANUSCORE DESIGN & FEATURES

In designing the Graphical User Interface (GUI) for ManuScore we wanted to emulate a “pencil and paper” workflow, while maintaining a balance between power and flexibility. Wherever possible, we chose to utilize the standard ARROW UP/DOWN/LEFT/RIGHT keys for moving objects, selecting accidnats, toggling articulation markings, and so on. This was because we wanted to remember only a limited set of possible interactions when learning the software.

3.1. An ‘Open’ Musical Space

At launch, the ManuScore GUI presents the user with an empty space—a “blank canvas”, so to speak. The background displays faint vertical guides, which act as a temporal grid for entering musical events. The grid does not strictly follow the conventions of musical time signatures. Rather, it acts as a visual guide to subdivide the musical space, providing a similar function to the grid systems found in graphics and drawing software packages, and provides “snapping” functionality to assist the user in entering rhythmically precise material. Objects can be moved independently of the grid, so that events can be placed at any location in musical time.

At the top of the score window, “Metric Markers” can be inserted, allowing the temporal grid to be subdivided in arbitrary ways. It is worth noting that this division is strictly graphical, and imposes no formal restrictions on the music itself. The numbers in each marker indicate a grouping/subdivision of time. The top two numbers are conceptually analogous to a conventional time signature, while the bottom number indicates the number of “beat divisions” used for object snapping (and thus can create any n-tuplet subdivision). Figure 1 shows a sample score with two Metric Markers added. It will be noted that the markers only alter the grid for the rhythmic space following the marker’s position. This can be seen in Figure 1, where what appears to be a 4/4 time signature is cut short by a 2/4 signature, inserted part-way through the first measure. In conventional notation software, replicating the musical meaning of this structure would require the user to completely redefine the metrical notation of the music (and in most cases, to delete and re-enter the musical passage). However, because the temporal grid in ManuScore is essentially independent from the contents of the staves, this sort of structure can be created at any time, without altering the existing musical material. In this sense, ManuScore’s “rhythmic representation” offers a “window” for composition, as opposed to the highly structured metrical space of conventional notation software.

3.2. Note Entry in ManuScore

Once a staff has been created, single notes can be entered in three ways: 1) Typing and clicking on a staff. 2) Using the Step-Entry Cursor (e key), and 3) Using a MIDI keyboard. It may have already been noted that ManuScore attaches accidentals to all notes, following the practice used by composers like Witold Lutoslawski. In ManuScore, this is a direct result of the inherent lack of conventional bar lines, which traditionally serve to nullify previously written accidentals. Once a note is created, the accidental can be toggled, by holding the CDP/M2512 key and using the ARROW UP/DOWN keys.

Material can also be entered in complete ‘gestures’, using the Gesture Tool (g key). This tool allows the user to draw a free-hand line on the staff, which is subsequently interpreted by the underlying generative system. ManuScore is backed by our “Closure-based Cueing Model” (CbCM), which is used for gesture interpretation. The process operates by using the CbCM to infer the pitch contour of the gesture. For each transition in the inferred “Schema” pattern [11] (i.e., contour), the algorithm selects those pitches which best approximate the position of the gesture line. ManuScore’s interpretation of a drawn gesture is shown in Figure 3. It is worth noting, that although most of the interpreted pitches follow the line quite tightly, the low D represents a ‘best-attempt’ of the CbCM, given its training. Its failure to follow the gesture line indicates that, in the given musical context, the system did not have a learned transition that could better approximate the path of the line. For a detailed discussion of the CbCM, see [11].

3.3. Orchestration in ManuScore

Our general goal of ‘openness’ can also be seen in ManuScore’s flexible approach to orchestration. Rather than following the conventional design, in which instruments are created a priori, on “tracks” similar to those used in analog tape recorders, ManuScore uses an approach inspired by the practice of composing to “short score.” When composers work to short-score, they often apply orchestration notes after the fact, assigning instruments to...
specific musical gestures directly on the staff. Orchestration in ManuScore follows the same process, as shown in Figure 5. With a correctly configured MIDI system, the instrumental switch from Flute to Viola at F♯ will be played back by the instrument assigned to the staff under the release as the “target” staff. The source staff acts as a data source, and the target staff acts as a receiver of some aspect of the source staff’s data. An example of applying the pitch contour from a source staff to the target staff is shown in Figure 7. Link functions currently include the following operations:

- **Pitch contour**: Applies the source staff’s pitch contour to the contents of the target staff. If the target staff has a greater number of events than the source, the source contour is repeated.
- **Rhythmic contour**: Renders the rhythmic values of events on the target staff to match the rhythmic contour of events on the source staff.
- **Pitch grid**: Provides ‘crisp’ locking of target pitches to source pitches.
- **Harmonic grid**: Provides ‘fuzzy’ locking of target pitches to source pitches. The locking algorithm uses a histogram of pitches used in the score up to the time of the target staff, weighted toward the pitches in the source staff.
- **Rhythmic grid**: Imposes the rhythmic pattern of the source staff onto the contents of the target staff.
- **Trigger Staff**: Allows non-linear playback possibilities by causing the source staff to “trigger” the target staff. When playback of the source staff ends, the target staff begins, regardless of its horizontal position on the score.
- **Interrupt Staff**: If the target staff is playing back at the time when the source staff begins, the target staff is muted; i.e., the source staff “interrupts” the target staff.

4. A COMPOSITION STUDY USING MANUSCORE

In spring of 2011, an applied composition study using ManuScore was conducted by composer James B. Maxwell, working under the supervision of composer/Professor Owen Underhill. The objective of the study was to test the functionality of the software in a one-to-one composition study context. During the study, the composer was to create two short works; one using his regular software package (with which he had been working for many years), and the other using ManuScore. Both pieces were to be approximately 5.00 minutes in duration, and both were to be scored for string quartet. As a further limitation on the process, both works would draw source material from Fredrick II’s “Royal Theme” (best known as the subject of Bach’s *Musical Offering*). The two works would be premièred together, in performance, in the fall of 2011, and a listener study conducted at the concert, as described in Section 5.

We do not suggest that the above limitations provide a strong enough framework for quantitative evaluation. However, we do feel that they impose enough commonality on the two compositional processes to isolate, at least to some degree, the software itself as a potential source of difference between the resulting works. Each working process was recorded using video screen capture, in order to provide detailed documentation. An excerpt of play-back from the compositional process of the work composed in ManuScore can be viewed online (audio play-back in the clip is directly from ManuScore, using the “Vienna Instruments” software):

http://rubato-music.com/home/Media/MnS_experiri.mov

For the purposes of this paper we will focus on the two works composed during the composition study described in Section 4. Since the primary design goal of ManuScore is to introduce CAC into the compositional process without disrupting the development of a composer’s musical language, we hypothesize that audience members will not judge the computer-assisted work *expertly* to be implicitly more “human” than the strictly human-composed work, *fundamento*.

6. STUDY RESULTS

In order to avoid the alpha inflation that arises from multiple comparisons, statistical tests were made using post-

---

**Figure 5.** Assigning Instruments in ManuScore.

**Figure 6.** Note-attached staccato, accent, down-bow, and tremolo articulations.

**Figure 7.** Using a Link to apply the pitch contour from one staff to another.
hosc Bonferroni-corrected alpha levels of .005 (.05/10). For part of the analysis, the 46 audience members were divided into novice and expert groups, based on the score they indicated for the "familiarity with contemporary music" question. The novice group consisted of audience members who gave a score of 1-3 out of 5 on the familiarity scale (N = 25). The expert group consisted of the remaining audience members who gave a 4 or 5 (N = 21). Two audience members failed to provide a familiarity score, so their data was excluded from group comparisons.

Table 1 gives the engagement rating for all ten works on the programme. The two works composed during the composition study, fundatio and experiri, are identified in bold type. The score for “Others, Previously”, also written for string quartet, has been italicized to draw attention to the fact that a highly significant difference between the average engagement ratings for all string quartet pieces (M = 4.36, SD = .73) and the “One of the Above” series of solo percussion pieces (M = 3.36, SD = 1.06) was found, t(133) = 8.71, p < .0001. Similarly, a comparison between the string quartet pieces and the “hybrid” string/percussion pieces Dead Slow / Look Left and Gradual (M = 3.69, SD = 1.09) was also highly significant, t(89) = 4.79, p < .0001, suggesting that audience members were more engaged by pieces containing strings than by those containing percussion. A comparison between the percussion and hybrid pieces failed to be significant, t(89) = 1.41, p = .16 ns.

It is worth noting that comparisons between the expert listener engagement ratings for the two works from the composition study, fundatio (M = 4.29, SD = .81) and experiri (M = 4.47, SD = .61) were also significant, t(18) = 1.00, p = .33 ns. Novice ratings for fundatio (M = 4.24, SD = .83) and experiri (M = 4.36, SD = .86) were similarly non-significant, t(24) = .38, p = .71 ns.

In Table 2 we give the results for the “directly human-composed” (c) computer-composed, (h) human-composed, (c-a) computer-assisted (standard deviations in brackets).

It was felt that such discrepancies were primarily arising as an effect of the purely graphical nature of ManuScore’s temporal grid. Since the grid does not impose a specific metrical structure on the music, the cyclical process of writing and listening tends to emphasize aural rather than theoretical principles in the developing composition. With a temporal grid of 5 beat divisions in place, patterns were easily entered into ManuScore in quad-plot patterns. However, through the iterative process of listening, entering material, editing, and listening, the musical forms naturally began to unfold according to perceptual/cognitive principles driven by the musical materials themselves. The phrasing of ideas in the musical foreground gave rise to certain types of groupings, and these naturally gave rise to accomplishments that supported those groupings. And because the temporal grid was easy to adjust to virtually any beat division value, it was simply not a priority to alter the metrical structure of the work-in-progress. In a sense, quintuplets became “the new sixteenths” for the work.

Table 1. Audience evaluation of “engagement”:

<table>
<thead>
<tr>
<th>Work Name</th>
<th>N</th>
<th>Listener Experience</th>
</tr>
</thead>
<tbody>
<tr>
<td>In Equilibrio</td>
<td>1</td>
<td>Expert</td>
</tr>
<tr>
<td>Dead Slow/Look Left</td>
<td>8</td>
<td>Novice</td>
</tr>
<tr>
<td>One of the Above #1</td>
<td>12</td>
<td>Combined</td>
</tr>
<tr>
<td>String Quartet</td>
<td>1</td>
<td>Computer</td>
</tr>
<tr>
<td>Percussion</td>
<td>1</td>
<td>Computer</td>
</tr>
<tr>
<td>Other, Previously</td>
<td>24</td>
<td>Computer</td>
</tr>
<tr>
<td>One of the Above #2</td>
<td>2</td>
<td>Combined</td>
</tr>
<tr>
<td>One of the Above #3</td>
<td>2</td>
<td>Combined</td>
</tr>
<tr>
<td>One of the Above #4</td>
<td>2</td>
<td>Combined</td>
</tr>
<tr>
<td>Gradual</td>
<td>14</td>
<td>Combined</td>
</tr>
</tbody>
</table>

Table 2. Evaluation of “directly human-composed”: (c) computer-composed, (h) human-composed, (c-a) computer-assisted (standard deviations in brackets).

<table>
<thead>
<tr>
<th>Work Name</th>
<th>N</th>
<th>Listener Experience</th>
</tr>
</thead>
<tbody>
<tr>
<td>In Equilibrio</td>
<td>1</td>
<td>Expert</td>
</tr>
<tr>
<td>One of the Above #1</td>
<td>12</td>
<td>Novice</td>
</tr>
<tr>
<td>Dead Slow/Look Left</td>
<td>8</td>
<td>Computer</td>
</tr>
<tr>
<td>One of the Above #2</td>
<td>2</td>
<td>Combined</td>
</tr>
<tr>
<td>numa</td>
<td>30</td>
<td>Computer</td>
</tr>
<tr>
<td>numb</td>
<td>27</td>
<td>Computer</td>
</tr>
<tr>
<td>One of the Above #3</td>
<td>2</td>
<td>Combined</td>
</tr>
<tr>
<td>Other, Previously</td>
<td>24</td>
<td>Computer</td>
</tr>
<tr>
<td>One of the Above #4</td>
<td>2</td>
<td>Combined</td>
</tr>
<tr>
<td>Gradual</td>
<td>14</td>
<td>Combined</td>
</tr>
</tbody>
</table>

Table 1 gives the engagement rating for all ten works on the programme. The two works composed during the composition study, fundatio and experiri, are identified in bold type. The score for “Others, Previously”, also written for string quartet, has been italicized to draw attention to the fact that a highly significant difference between the average engagement ratings for all string quartet pieces (M = 4.36, SD = .73) and the “One of the Above” series of solo percussion pieces (M = 3.36, SD = 1.06) was found, t(133) = 8.71, p < .0001, suggesting that audience members were more engaged by pieces containing strings than by those containing percussion. A comparison between the percussion and hybrid pieces failed to be significant, t(89) = 1.41, p = .16 ns.

It is worth noting that comparisons between the expert listener engagement ratings for the two works from the composition study, fundatio (M = 4.29, SD = .81) and experiri (M = 4.47, SD = .61) were also significant, t(18) = 1.00, p = .33 ns. Novice ratings for fundatio (M = 4.24, SD = .83) and experiri (M = 4.36, SD = .86) were similarly non-significant, t(24) = .38, p = .71 ns.

In Table 2 we give the results for the “directly human-composed” ratings, where it is clear that both fundatio and experiri were estimated to be human-composed works. Again, there is an effect of instrumentation to be considered, as the other string quartet work was also highly rated (score in italics). However, there was once again no significant difference between the work composed in ManuScore and the work composed through the composer’s understanding of the structure of the work.

8. FUTURE WORK

A new version of ManuScore is currently under development. This version is backed by a modular cognitive architecture for music, called MusiCOG, which replaces the CBCM as the generative back-end for the system. An overview of this model, which is a development and extension of the CSMC, can be found in Maxwell et al. [12]. All of the features described in this paper have been included in the new version.

In response to the composer’s experience of frequently altering the pitch content of CBCM continuations to match local key changes, we planned to add a functionality to “quantize” the pitch content of generated material before it is rendered. Since the CBCM often generates continuations based on interval patterns, rather than pitch patterns, deviations from the local key/scale were somewhat expected. However, a pitch quantization method would help reduce the cognitive load on users, and could be implemented using the existing algorithms from ManuScore’s “harmonic grid” Link function.

A useful future development, which came to our minds during the present study, might involve the inclusion of methods for beat induction and metrical inference. Such methods would be useful for MusiCOG’s underlying music perception and cognition functions, and could also be used to periodically re-interpret the temporal grid of the score during creation. Such metrical interpretation could help the user avoid difficult transcription decisions after completing a score, and would also support the composer’s understanding of the structure of the work in progress. An extension of this functionality could allow ManuScore to transcribe standard music notation versions of the composer’s work, for export and printing, thus streamlining the process of moving from ManuScore to concert performance.
9. CONCLUSIONS
It is difficult to evaluate the role that CAC tools play in the compositional process; indeed, the influence of even the most ‘inert’ music notation software on compositional thinking is difficult to deny [2, 3, 4, 22]. Manuscore expands the field of CAC tools by augmenting common notation-based approaches with a more open conceptual design, and with the inclusion of corpus-based, generative capabilities. Although further validation of Manuscore is required, the user and listener studies outlined in this paper suggest that our goal of providing an interactive CAC tool, which enhances the compositional process, without disrupting the composer’s musical language, has been at least provisionally achieved.

10. ACKNOWLEDGEMENTS
This work was made possible in part by the Social Sciences and Humanities Research Council, the Canada Council for the Arts, and the Natural Sciences and Engineering Research Council of Canada.

11. REFERENCES


THE SMUSE: AN EMBODIED COGNITION APPROACH TO INTERACTIVE MUSIC COMPOSITION

ABSTRACT
The evolution of computer-based music systems has gone from computer-aided composition, which transposed the traditional paradigms of music composition to the digital realm, to complex feedback systems that allow for rich multimodal interactions. Yet, a lot of interactive music systems still rely on outdated principles in the light of modern situated cognitive systems design. Moreover, the role of human emotional feedback, arguably an important feature of musical experience, is rarely taken into account into the interaction loop. We propose to address these limitations by introducing a novel situated synthetic interactive composition system called the SMUse (for Situated Music Server). The SMUse is based on the principles of parallelism, situatedness, emergence and emotional feedback and is built on a cognitively plausible architecture. It allows to address questions at the intersection of music perception and cognition while being used as a creative tool for interactive music composition.

1. BACKGROUND
Interactivity has now become a standard feature of many multimedia systems and plays a fundamental role in contemporary art practice. Specifically, real-time human/machine interactive music systems are now omnipresent as both composition and live performance tools. Yet, the term “interactive music system” is often misused. The interaction that takes place between a human and a system is a process that includes both control and feedback, where the real-world actions are interpreted into the virtual domain of the system [4]. If some parts of the interaction loop are missing (for instance the cognitive level in Figure 1), the system becomes only a reactive (vs. interactive) system. As a matter of fact, in most of current human-computer musical systems, the human agent interacts whereas the machine due to a lack of cognitive modeling only reacts. Although the term interactivity is widely used in the new media arts, most systems are simply reactive systems [4]. Furthermore, the cognitive modeling of interactive multimedia systems, when it exists, often relies on a classical cognitive science approach to artificial systems where the different modules (e.g. perception, memory, action) are studied separately. This approach has since been challenged by modern cognitive science, which emphasizes the crucial role of the perception-action loop, the building of cognitive artifacts, as well as the interaction of the system with its environment [37]. In this paper we propose a novel approach to interactive music system design informed by modern cognitive science and present an implementation of such a system called the SMUse.

2. FROM EMBODIED COGNITIVE SCIENCE TO MUSIC SYSTEMS DESIGN
2.1. Classical View
A look at the evolution of our understanding of cognitive systems put in parallel with the evolution of music composition practices, gives a particularly interesting perspective on some limitations of actual interactive music systems. The classical approach to cognitive science assumes that external behavior is mediated by internal representations [6] and that cognition is basically the manipulation of these mental representations by sets of rules. It mainly relies on the sense-think-act framework [27], where future actions are planned according to perceptual information. Interestingly enough, a parallel can be drawn between classical cognitive science and the development of classical music which also heavily relies on the use of formal structures. It puts the emphasis on internal processes.
Disembodiment in classical music composition can be seen at several levels. Firstly, by training, the composer is used to compose in his head and translate his mental representations into an abstract musical representation: the score. Secondly, the score is traditionally interpreted live by the orchestra’s conductor who “controls” the main aspects of the musical interpretation, whereas the orchestra musicians themselves are left with a relatively reduced interpretative freedom. Moreover, the role of the audience as an active actor of a musical performance is mostly neglected.

2.2. Modern View

An alternative to classical cognitive science is the connec-tionist approach that tries to build biologically plausible systems using neural networks. Unlike more traditional digital computation models based on serial processing and explicit manipulation of symbols, connectionist networks allow for fast parallel computation. Moreover, it does not rely on explicit rules but on emergent phenomena stemming from the interaction between simple neural units. Another related approach, called embodied cognitive science, puts the emphasis on the influence of the environment on internal processes. In some sense it replaced the view of cognition as a representation by the view that cognition is an active process involving an agent acting in the environment. Consequently, the complexity of a generated structure is not the result of the complexity of the underlying system only, but partly due to the complexity of its environment [34].

A piece that gives a good illustration of situatedness, distributed processing, and emergence principles is In C by Terry Riley. In this piece, musicians are given a set of pitch sequences composed in advance, but each musician is left in charge of choosing when to start playing and repeating these sequences. The piece is formed by the combination of decisions of each independent musician that makes her decision based on the collective musical output that emerges from all the possible variations. Following recent evolution of our understanding of cognitive systems, we emphasize the crucial role of situatedness in classical music composition: the performer

2.3.1. Explicit Gestural Interfaces

The advent of new sensing technologies has fostered the development of new kind of interfaces for musical expression. Graphical User Interfaces, tangible interfaces, gesture interfaces have now become omnipresent in the design of live music performance or compositions [24]. Most of these interfaces are gesture-based interfaces that require explicit conscious body movements from the user. They can give access to behavioral or self-reported information, but not to implicit emotional states of the user.

2.3.2. Implicit Biosignal Interface

thanks to the development of more robust and accurate biosignal technologies, it is now possible to derive emotion-related information from physiological data and use it as an input to interactive music systems. Although the idea is not new [15, 30], the past few years have witnessed a growing interest from the computer music community in using physiological data such as heart rate, electrodermal activity, electroencephalogram and respiration to generate or transform sound and music. Providing emotion-based physiological interface is highly relevant for a number of applications including music therapy, diagnosis, interactive gaming, and emotion-aware musical instruments.

2.3.3. Emotional Mapping

Music and its effect on the listener has long been a subject of fascination and scientific exploration from the Greeks speculating on the acoustic properties of the voice [14] to Musak researcher designing “soothing” elevator music. It has now become an omnipresent part of our day to day life. Music is well known for affecting human emotional states, and most people enjoy music because of the emotions it evokes. Yet, although emotions seem to be a crucial aspect of music listening and performance, the scientific literature on music and emotion is scarce if compared to music cognition or perception [21, 10, 17, 5]. The relationship between specific musical parameters and time-varying emotional responses is still not clear. Biofeedback interactive music systems appear to be an ideal paradigm to explore the complex relationship between emotion and music.

2.5. Music Processing Modules

Research on the brain substrates underlying music processing has switched in the last twenty years from a classical view emphasizing a dichotomy between language (supposedly processed in left hemisphere) and music (respectively right hemisphere) to a modular view [1]. There is some evidence that music processing modules are organized into two parallel but largely independent submodules that deal with pitch content (“What?”) and temporal content (“When?”) respectively [26, 18]. This evidence suggests that they can be treated separately in a computational framework. Additionally, studies involving music-related deficits in neurologically impaired individuals (e.g. subjects with amusias who can’t recognize melodies anymore) have shown that music faculty is composed of a set of neurally isolable processing components for pitch, loudness and rhythm [25]. The common view is that pitch, rhythm and loudness are first processed separately by the brain to then later form (around 25-50ms) an impression of unified musical object [20] (see [16] for a review of the neural basis of music perception). This modularity as well as the three different levels and time scales of auditory memory (sound, groups, structure) form a set of basic principles for designing our bio-mimetic music system.

3. A COMPUTATIONAL MODEL BASED ON A SOCIETY OF MUSICAL AGENTS

The architecture of SMuSe is inspired by neurological evidence. It follows a hierarchical and modular structure, and has been implemented as a set of agents using dataflow programming.

3.1. The SMuSe’s Architecture

SMuSe is built on a hierarchical, bio-mimetic and modular architecture. The musical material is represented at three different hierarchical levels, namely event fusion, event grouping and structure corresponding to different memory constraints. From the generative point of view, SMuSe modules are divided into time modules (“when”) that generate rhythm pattern of events and content modules (“what”) that for each time event choose musical material such as pitch and dynamics (Figure 3).

These cognitive and perceptual constraints influenced the design of the SMuSe’s architecture. At the low event fusion level, SMuSe provides a set of synthesis techniques validated by psychoacoustic tests [2] that give perceptual control over the generation of timbre as well as the use of MIDI information to define basic musical material such as
pitch, velocity and duration. Inspired by previous works on musical performance modeling [7], the SMuSe also allows to modulate the expressiveness of music generation by varying parameters such as phrasing, articulation and performance noise [2]. At the medium melodic and rhythmic grouping level, the SMuSe implements various state of the art algorithmic composition tools (e.g. generation of tonal, Brownian and serial series of pitches and rhythms, Markov chains, ...). The time scale of this mid-level of processing is in the order of 5s for a single grouping, i.e. the time limit of auditory short-term memory. The form level concerns large groupings of events over a long period of time (longer than the short-term mem- ory). It deals with entire sequences of music and relates to the structure and limits of long-term memory. Influ- enced by experiments in synthetic epistemology and situated robotics, this longer term structure is accomplished via the interaction with the environment [37, 38]. The modularity of the music processing chain is also re- flected in different SMuSe modules that specifically deal with time (“when”) or material (“what”).

3.2. Agency
The agent framework is based on the principle that complex tasks can be accomplished through a society of sim- ple cross-connected self-contained agents [22]. Here, an agent is understood as “anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors” [32]. In the con- text of cognitive science, this paradigm somehow takes a stand against a unified theory of mind where a diversity of phenomena would be explained by a single set of rules. The claim here is that surprising, complex and emergent results can be obtained through the interaction of simple non-linear agents. The agent framework is particu- larly suited to building flexible real-time interactive mus- ical systems based on the principles of modularity, real- time interaction and situatedness.

3.3. Data-flow Programming
We chose to implement this hierarchy of musical agents in SMuSe in a data-flow programming language called Max/MSP [40]. Data flow programming conceptually mod- els a program as a directed graph of dataflow between operations. This kind of model can easily represent par- allel processing which is common in biological systems, and is also convenient to represent an agent-based modu- lar architecture. Interestingly enough, programming in Max/MSP encourages the programmer to think in a way that is close to how a brain might work. Firstly, since Max/MSP is based on a data-flow paradigm, processes can operate in parallel (e.g. pitch and rhythm processes). Secondly, thanks to the concept of patch abstraction (a Max "meta-patch" that ab- stracts or include another Max patch), one is able to easily build several layers of processing units (which is some- how similar to the different layers of the cortex). Finally, each process can be connected to every other in a vari- ety of ways (like neurons). Of course, the comparison to neural processes is limited to higher level, organizational processes.

3.4. Distributed Control
All the musical agents in SMuSe are OSC-compatible [39] which means they can be controlled and processed from anywhere (including over a network) at any time. This gives great flexibility to the system, and allows for shared collaborative compositions where several clients access and modulate the music server. In this collaborative com- position paradigm, every performer builds on what the others have done. The result is a complex sound struc- ture that keeps evolving as long as different performers contribute changes to its current shape. A parallel could be drawn with stigmergic mechanisms of coordination be- tween social insects like ants [34, 3, 11]. In ants colonies, the pheromonal trace left by one ant at a given time is used as a means to communicate and stimulate the action of the others. Hence they manage to collectively build complex networks of trails towards food sources. Simi- larly, in a collective music paradigm powered by an OSC client/server architecture, one performer leaves a musical trace to the shared composition, which in turn stimulate the other co-performers to react and build on top of it.

3.5. Concurrent and On-the-fly Control of Musical Proces- ses
We have proposed a biologically inspired memory and process architecture for SMuSe as well as a computational model based on OSC communication protocol that allows to easily send text-based commands to specific agents in the hierarchy. It allows for flexible and intuitive time-based, concurrent and on-the-fly control of musical processes. The different musical agents in the SMuSe all have a specific ID/address where to receive commands and data. The addresses are divided into /global (affecting the whole hi- erarchy), /voice[n] (affecting specific voices), and /synth[n] (affecting specific sound generators). The OSC syntax supports regular expressions which allows to address sev- eral modules at the same time with a compact syntax. Patterns of perceptually-grounded musical features are sent to the short-term (STM) and long-term memory (LTM) modules at any moment in time via specific commands.

Examples:
```plaintext
/voice/pitch/pattern 0 0 5 7 10 0
/voice/pitch/register 4 5 4 4 4 5
/voice/velocity 12 12 16 16 32
```
4. ARTISTIC REALIZATIONS

We have further explored the purposive construction of interactive installations and performances using the SMuSe system. To name but a few, during the VRobooser installation [2], the sensory inputs (motion, color, distance) of a 3D virtual Khepera1 robot living in a game-like environment modulated musical parameters in real-time, thus creating a never-ending musical soundscape in the spirit of Brian Eno’s “Music for Airports”. In another context the SMuSe generated automatic soundscapes and music which reacted to and influenced the spatial behavior of human and avatars in the mixed-reality space called XIM (for eXperience Induction Machine) [2] thus emphasizing the role of the environment and interaction on the musical composition. Based on similar premises, RePERcursus, an interactive mixed reality performance involving dance, percussion, interactive music, and video was presented at the ArsFutura Festival 07 and Museum of Modern Art in Barcelona in the same year. The performance was composed by several interfaced layers of artistic and technological activities. The music controlled had three components: a predefined soundscape, the percussionist who performed from a score and the interactive composition system synchronized by SMuSe: the physical actors, the percussionist and the dancer were tracked by a video-based active tracking system that in turn controlled an array of moving lights that illuminated the scene. The spatial formation from the stage obtained by the tracking system was also projected onto the virtual world where it modulated the avatar’s behavior allowing it to adjust body position, posture and gaze to the physical world. In 2009, the Brain Orchestra, a multimodal performance using brain computer interfaces, explored the creative potential of a collection of brains directly interfaced to the world. During the performance, four “brain musicians” were controlling a string quartet generated by the SMuSe using their brain activity alone. The orchestra was conducted by an “emotional conductor”, whose emotional reactions were recorded using bio-signal interfaces and fed back to the SMuSe system. The Brain Orchestra was premiered in Prague for the FET (09) meeting organized by the European Commission [2]. Finally, a live performance of a piece inspired by Terry Riley’s “In C” served as an illustration of the principles of parallelism, situatedness and emergence exhibited by the SMuSe at the Ernst Strunngmann Forum on Language, Music and the Brain: a mysterious relationship.

5. CONCLUSIONS

The SMuSe illustrates a novel situated approach to music composition systems. It is built on a cognitively plausible architecture that takes into account the different time frames of music processing, and uses an agent framework to model a society of simple distributed musical processes. It takes advantage of its interaction with the environment to go beyond the classic sense-think-act paradigm [31]. It combines cognitively relevant representations with perceptually grounded sound synthesis techniques and is based on modern data-flow audio programming practices [28, 29]. This provides an intuitive, flexible and distributed control environment that can easily generate complex musical structures in real-time. SMuSe can sense its environment via a variety of sensors, notably physiology-based sensors. The analysis and extraction of relevant information from sensor data allows to re-inject emotion-based feedback to the system based on the responses of the human participant. The SMuSe proposes a set of “pre-wired” emotional mappings from emotions to musical parameters grounded on the literature on music and emotion, as well as a reinforcement learning agent that performs online adaptive mapping. It provides a well-grounded approach towards the development of advanced synthetic aesthetic systems and a further understanding of the fundamental psychological processes on which it relies.

6. REFERENCES


BACH: AN ENVIRONMENT FOR COMPUTER-AIDED COMPOSITION IN MAX

Andrea Agostini
Freelance composer

Daniele Ghisi
Composer - Casa de Velázquez

ABSTRACT

environments include tools for sound synthesis and transformation. It should though be remarked that Max and PureData have very crude native support for sequencing, and essentially none for symbolic musical notation. Another, orthogonal distinction should be made between real-time systems, which ‘immediately’ react to interface actions (such as Finale, MaxMSP, ProTools...) and non-real-time systems, where these actions have no effect until a certain ‘refresh’ operation is performed (such as Lilypond, OpenMusic, PWGL). The latter is the case of typical CAC environments; yet, in some cases this is un-natural, and it might be argued that there is no deep reason why symbolic processing should not be performed in real-time. This does not mean that every compositional process should benefit from a real-time data flow, but some might, as we shall exemplify at the end of the paper. Real-time is a resource, rather than an obligation. Yet, the lack of this resource has pushed, up to now, the development of CAC techniques only in the off-line direction.

In our own experience, the real-time or non-real time nature of an environment for music composition deeply affects the very nature of the compositional process. Composers working with sequencers, plug-ins and electronic instruments need them to immediately react as they change their parameters; likewise, composers working with symbolic data might want the machine to quickly adapt to new parameter configurations. As composers ourselves, we believe that the creation and modification of a musical score is not an out-of-time activity, but it follows the composer’s discovery process and develops accordingly.

This issue has been faced by Miller Puckette in [11]: “While we have good paradigms for describing processes (such as in the Max or Pd programs as they stand today), and while much work has been done on representations of musical data (ranging from searchable databases of sound to Patchwork and OpenMusic, and including Pd’s unfinished data editor), we lack a fluid mechanism for the two worlds to inter-operate.”

Arshia Cont in [5] adds: “The performers of computer music have been faster to grab ideas in real time manipulations and adopting them to their needs. Today, with many exceptions, a wide majority of composed mixed instrumental and electronic pieces
are based on simplistic interactive setups that hinder the notion of interactivity. This fact does not degrade the artistic value of such works in any sense but underlies the lack of momentum therein for serious considerations of interactivity among the second group."

Of course, this dichotomy has already been addressed. Several interesting projects have been developed, linking real-time environments to graphical representations of both classical and non-classical (and potentially non-musical) scores, including OpenTimeLine\textsuperscript{9} and InScore\textsuperscript{9, 7}. In at least one case, namely MaxScore\textsuperscript{6}, this is augmented by a very sophisticated editing interface. A more general approach is FTM's\textsuperscript{14}, which provides a powerful framework for data representation and processing with a focus on musical structures, including some facilities for graphical display of simple scores.

Resuming the ideas of\textsuperscript{2, 3}, with the library bach: automatic composer's helper we have tried to achieve a coherent system explicitly designed for computer-assisted composition. bach takes advantage of Max's facilities for sound processing, real-time interaction and graphical programming, combining interactive writing and algorithmic control of symbolic musical material.

2. PROGRAMMING PARADIGMS

bach complies with the graphical data-flow programming paradigm of Max, in which a musical patch is represented as a vertical, top-down flow. Data, typically coming from some user interaction, enter the program at its top, are acted upon by a chain of specialized operators connected by lines called 'patch cords' and exit the program at its bottom. A simplified model of this mechanism, as seen from a lower-level point of view, might appear as follows: each operator is a function, usually written in C or C++, and the data entering it are the arguments of the function call. After performing its work upon the data it has received, each operator calls the function corresponding to the next operator in the chain, passing the acted-upon data. In this way a call stack is built, in which the operator at the top of the graphical patch corresponds to the function at the base of the stack, and the operator at the bottom of the graphical patch corresponds to the function at the top of the stack. It is crucial to note that all these functions have no return value: the last operator of the chain simply passes the data to an arbitrary output device. In this way, the perception of the user's side is that the program essentially behaves like a musical instrument, in which an action (e.g. pressing a piano key) triggers a sequence of reactions (levers moving, hammers striking) leading in a measurable but usually negligible time, to the production of a sensible result (sound).

The major graphical computer-aided composition environments, that is the Patchwork family\textsuperscript{10, 4} (Patchwork, OpenMusic, PWGL), are based upon the Lisp programming language. Although superficially similar to Max from the point of view of the user interface (data and functions are represented by graphical elements connected by lines representing the flow of elations), the underlying programming paradigm is radically different. Essentially, the graphical program is indeed a representation of a Lisp expression, with elements on the top of the patch corresponding to the deepest elements of the expression. The user requests the evaluation of an operator, which in turn will request evaluation of the operators above it, and so on. From a lower-level point of view, a call stack is built in this scenario as well; the difference is that all the functions in the stack have a return value, and the final return value is returned to the user through a console. Of course, in some cases the side effects of the evaluation (e.g. a change in an user interface widget, or the production of a MIDI stream) are more important then the result itself. This paradigm applies a fortiori to textual Lisp-based environments such as Common Lisp or Impromptu.

The difference between the two paradigms is crucial: if we assume that parameters are handled at the beginning of the process, a bottom-up process (like within the Patchwork paradigm) will ultimately be a non-real-time process, since parameter changes cannot immediately affect anything below them, unless some bottom-up operation is requested on some lower elements. Moreover, the Max paradigm, not having to depend on return values, easily allow for much more completely structured patches: a single action can trigger multiple reactions in different operators (a function can call several other functions, one after another has returned). The Patchwork paradigm, on the other hand, has the advantage of allowing seamless integration with textual coding, which can be an extremely useful resource whenever conceptually complex operations must be implemented. Moreover, representing musical notation (from single notes to an entire score) requires sufficiently powerful and flexible data structures, which the Lisp lists certainly are.

3. THE BACH ENVIRONMENT

As already stated, bach is a library of objects and patches for the software Max, the distinction between objects and patches concerning more the implementation than the actual usage of these modules. At the forefront of the system are the bach:score and bach:roll objects. They both provide graphical interfaces for the representation of musical notation: bach:score expresses time in terms of traditional musical units, and includes notions such as rests, meters, time signature and tempo; bach:roll expresses time in terms of absolute musical units (number of quarter notes, for example). As a consequence there is no notion of traditional temporal concepts: this is useful for representing non-measured music, and also provides a simple way to deal with pitch material whose temporal information is unknown or irrelevant. It should also be noted that the implementation of traditional temporality concepts in bach:score is in fact quite advanced, as it allows multiple simultaneous time signatures, tempi and agogics. Besides this fundamental difference, the two objects offer a large set of common features, among which:

- editing by both mouse and keyboard interface, and by Max messages (see Fig. 1);
- support for microtonal accidents of arbitrary resolution (see Fig. 2);
- wide possibility of intervention over the graphical parameters of musical notation;
- ability to associate to each note various types of meta-data, including text, numbers, files and break-point functions (see Fig. 6);
- variable-speed playback capability: both bach:score and bach:roll can be seen as advanced sequencers, and the whole set of data (such as pitch, velocity and duration information) and meta-data associated to each note is output at the appropriate time during playback, thus making both objects extremely convenient for controlling synthesizers and other physical or virtual devices.

3.1. Data types

bach also provides Max with two new data types: rational numbers and a nested list structure called illi, an acronym for Illi-like linked list. Rational numbers are extremely important in music computation, as they express traditional temporal units such as 1/2, 3/8 or 1/12 (that is, a triplet eight note) as well as harmonic ratios. The nested list has been chosen for both similarity with the Lisp language, in a way to ease communication with the major existing CAC environment, and the need to establish a data structure powerful enough to represent the complexity of a musical score, but flexible enough to be a general data structure whose temporal information is unknown or irrelevant. The structure of a illi representing a bach:score (Fig. 4) might appear quite complex at first sight, but the or

\textsuperscript{9}http://maxj.free.fr
\textsuperscript{9}http://in-score.sourceforge.net
\textsuperscript{10}http://www.computermusicnotation.com

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{Any notation object can be edited by both GUI interaction and Max messages. In this case we're clearing the bach:roll, and then adding two chords.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure2.png}
\caption{Semitonal, quartertonal and eighttonal divisi-}
\textsuperscript{9}nions are supported via the standard accidental symbols (upper example). All other microtonal divisions are supported as well, but will be replaced by labels with the explicit fractions of tone (lower example), or with cents differences from the diatonic note.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.png}
\caption{Tools for working upon illi, performing basic operations such as retrieval of individual elements, iteration, reversal, sorting, splicing, merging and so on (see Fig. 3). Some subsets of the library are applicable to illi satisfying certain given conditions: e.g. it is possible to perform mathematical operations over illi solely composed by numbers; a set of operators for matrix calculations only works with appropriately structured illi; and so on. It is important to stress that all these operators are indeed Max objects, and while the kind of operations performed may have some resemblance with Lisp, the actual implementation and interface are radically different, and as integrated as possible with the Max system. On the other hand, at least one Common Lisp interpreter designed to run on Max objects has been developed, Brad Garton's maxlisp\textsuperscript{8}:

\begin{itemize}
  \item it is extremely easy to exchange data with this object, in order to take advantage of the expressive power of Lisp textual programming within a Max patch.
\end{itemize}

3.2. Music representation

At the intersection between the modules for musical notation and the list operators is a family of objects performing operations upon illi containing musical data.

\begin{itemize}
  \item It is worth noting that different bach objects exchange musical scores in the form of specifically-structured illi, whose contents is entirely readable and editable by the user; this is different from what happens e.g. in OpenMusic, where the exchange of musical data often involves opaque objects. This allows much easier and more transparent manipulation of the musical data themselves. As a consequence, strictly musical operations such as rhythmic quantization are just extremely specialized operations upon illi, of which can be performed only if the illi itself is structured properly, and if its content is consistent from the point of view of musical manipulations through a relatively small set of primitives. In fact, the large majority of the modules of the bach library are

\begin{itemize}
  \item Bach also provides Max with two new data types: rational numbers and a nested list structure called illi, an acronym for Illi-like linked list.
  \item Bach allows Max with two new data types: rational numbers and a nested list structure called illi, an acronym for Illi-like linked list.
\end{itemize}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure4.png}
\caption{A typical example of a bach:score object, with a few musical elements and meta-data.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure5.png}
\caption{A typical example of a bach:score object, with a few musical elements and meta-data.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure6.png}
\caption{A typical example of a bach:score object, with a few musical elements and meta-data.}
\end{figure}
non-destructive operations - as, on the contrary, it is often the case with Lisp.

Figure 5. The structure of a non-measured score in illl form, with branches for voices, chords and notes. Notice the meta-content contained in each note, appearing in the illls starting with the slots symbol. The form (type, range, domain...) of each slot appears in the header, which has not been dumped.

3.3. Data handling mechanism

As the goal of bach is allowing real-time interaction, a great amount of work has been spent to improve the stability and efficiency of the system. All the operations in bach are thread-safe in the context of the Max threading model, and the passing of illls between objects happens by reference, rather than by value, unless the user explicitly requests otherwise, which is the case whenever the contents of a illl need to be passed to a non-bach Max object (that only accepts data passed by value). Thus, illls are copied only when strictly necessary, and in all the other cases a reference counting mechanism is used to ensure that the lifetime of data structures and the usage of memory are correctly managed. On the other hand, all this is transparent to the user, who never needs to cope with the cloning of illls, or the distinction between destructive and non-destructive operations - as, on the contrary, it is often the case with Lisp.

3.4. Practical applications

Taking all this into account, it should be clear that bach is somehow placed at the convergence of several categories of musical software. Its capabilities of graphical representation of musical scores typically belong to music engraving systems - although it should be noted that, in its current state, bach lacks some essential features of this kind of programs, first of all a page view. On the other hand, most of its features are conceived in order to make it a tool for Computer Aided Composition as powerful as the traditional Lisp-based environments, and able to communicate with them. It can be used as the core of an extremely advanced and flexible, stretchable, with the ability to drive virtually any kind of process and playback system. Finally, it can of course lend itself to innovative applications exploiting the unique convergence of these different paradigms and its specific real-time behavior (such as the symbolic granulation example shown in Fig. 7).

4. FUTURE DEVELOPMENTS

At the time of writing, bach is in its alpha development phase: although the system is usable, not all the intended features have already been implemented. Some of the planned additions are:

- Support for rhythmic tree representation, which will allow, for example, nested tuplets to be represented, whereas now a triplet containing a quintuplet is represented as a 'flat' 15-tuplet. This feature is currently under development, together with an intuitive measure linear editing system for the note insertion. The underlying challenge is to keep the tree and linear representations of durations always compatible, so that users should completely deal with the tree representation only when they explicitly ask to (e.g. when they insert as rhythm a nested rhythmic structure), or when they perform hierarchical operations (e.g. when they split a chord). Users will also be able to rebuild a default rhythmic tree from the linear representation at any moment.

- Implementation of hierarchical structures within a score, allowing the user to group elements by name, where an element can be a chord, a note, a marker, or another group.

- Support for import and export of MIDI, MusicXML and SDIF files.

- A solver for constraint satisfaction problems. Notice that the software development situation might have changed at the time of publication, and some or all of the hereby proposed features might already be partly or fully implemented.

5. REFERENCES


In this article, we introduce OSC-NETLOGO, a tool that allows the creation of sonic phenomena by taking advantage of NetLogo's power for designing and building models of complex systems. NetLogo is a multi-agent programming language and modeling environment for simulating natural and social phenomena. It is particularly well suited for modeling complex systems that evolve dynamically over time. We provide two examples taken from NetLogo's library of models. These examples provide evidence for the capabilities and potential of NetLogo as a sound synthesis tool. We hope that this tool could be of aid in future efforts of creating new complex sounds and interesting musical material.

1. INTRODUCTION

NetLogo is a multi-agent programming language and modeling environment for simulating natural and social phenomena. It is particularly well suited for modeling complex systems evolving over time [4]. NetLogo comes from the Logo family of programming languages [3] and has expanded the original Logo concept in a number of ways. NetLogo allows modelers to give instructions to hundreds or thousands of independent agents all operating concurrently, which is something essential for modeling complex systems. This makes it possible to explore connections between micro-level behaviors of individuals and macro-level patterns that emerge from their interactions. NetLogo enables users to open simulations and play with them, exploring their behavior under various conditions by manipulating several graphical objects such as sliders or buttons. NetLogo is also an authoring environment that enables users to create their own models, and try them on the fly.

According to the NetLogo website [8], NetLogo is being used to build an endless variety of simulations. Members of the NetLogo community have turned turtles into molecules, wolves, buyers, sellers, bees, tribespeople, birds, worms, voters, passengers, metals, bacteria, cars, robots, neutrons, magnets, planets, shepherds, lovers, ants, muscles, networkers, and more. Patches have been made into neurons, magnets, planets, shepherds, lovers, ants, muscles, networkers, and more. Patches have been made into molecules, wolves, buyers, sellers, bees, tribespeople, birds, worms, voters, passengers, metals, bacteria, cars, robots, neutrons, magnets, planets, shepherds, lovers, ants, muscles, networkers, and more. Patches have been made into trees, walls, terrain, waterways, housing, plant cells, cancer cells, farmland, sky, desks, fur, sand, etc. [4].

turtle agents and patches are useful to visualize and study mathematical abstractions, specially non-linear dynamical systems, to display behavior in graphical ways, to make art and even play games. NetLogo comes with a very big library of models, which covers topics such as cellular automata, genetic algorithms, positive and negative feedback, evolution and genetic drift, population dynamics, networks, markets, chaos theory, swarming behavior, and molecular physics. All of these models share core concepts such as complex systems, self-organization and emergence.

Although NetLogo is very powerful for modeling and handling complex systems data, and provides some basic audio functionality through MIDI, it lacks more serious digital audio generation and processing capabilities. However, one good thing about NetLogo is that it provides an API for programmers to develop extensions to the program in Java. We took advantage of this feature and developed OSC-NETLOGO, an Open Sound Control (OSC) [9] extension to NetLogo using the JavaOSC library [2]. This extension allows users to directly map any parameter of a NetLogo patch into any sound processing environment such as Max/MSP, Pd or SuperCollider using OSC.

This article is structured as follows. In section 2 we describe the NetLogo application in more detail, including its history, API. In section 3 we describe the netlogo-osc extension, including its installation and usage. Then, in section 4 we provide examples of mappings of two NetLogo models into sound synthesis using Pd. Finally, in section 5 we discuss the main findings and conclusions of our work.

2. NETLOGO

NetLogo is a cross-platform standalone application written in Java. It has been being developed for more than ten years, which assures that NetLogo is a mature product that is stable and fast. It is freeware, anyone can download it for free and build models without restriction. It also comes with extensive documentation and tutorials and a large collection of sample models, created both by NetLogo developers and the general community of users [4].

Abstract

In this article, we introduce OSC-NETLOGO, a tool that allows the creation of sonic phenomena by taking advantage of NetLogo’s power for designing and building models of complex systems. NetLogo is a multi-agent programming language and modeling environment for simulating natural and social phenomena. It is particularly well suited for modeling complex systems that evolve dynamically over time. We provide two examples taken from NetLogo’s library of models. These examples provide evidence for the capabilities and potential of NetLogo as a sound synthesis tool. We hope that this tool could be of aid in future efforts of creating new complex sounds and interesting musical material.

1. INTRODUCTION

NetLogo is a multi-agent programming language and modeling environment for simulating natural and social phenomena. It is particularly well suited for modeling complex systems evolving over time [4]. NetLogo comes from the Logo family of programming languages [3] and has expanded the original Logo concept in a number of ways.

NetLogo allows modelers to give instructions to hundreds or thousands of independent agents all operating concurrently, which is something essential for modeling complex systems. This makes it possible to explore connections between micro-level behaviors of individuals and macro-level patterns that emerge from their interactions. NetLogo enables users to open simulations and play with them, exploring their behavior under various conditions by manipulating several graphical objects such as sliders or buttons. NetLogo is also an authoring environment that enables users to create their own models, and try them on the fly.

According to the NetLogo website [8], NetLogo is being used to build an endless variety of simulations. Members of the NetLogo community have turned turtles into molecules, wolves, buyers, sellers, bees, tribespeople, birds, worms, voters, passengers, metals, bacteria, cars, robots, neutrons, magnets, planets, shepherds, lovers, ants, muscles, networkers, and more. Patches have been made into trees, walls, terrain, waterways, housing, plant cells, cancer cells, farmland, sky, desks, fur, sand, etc. [4].

Turtle agents and patches are useful to visualize and study mathematical abstractions, specially non-linear dynamical systems, to display behavior in graphical ways, to make art and even play games. NetLogo comes with a very big library of models, which covers topics such as cellular automata, genetic algorithms, positive and negative feedback, evolution and genetic drift, population dynamics, networks, markets, chaos theory, swarming behavior, and molecular physics. All of these models share core concepts such as complex systems, self-organization and emergence.

Although NetLogo is very powerful for modeling and handling complex systems data, and provides some basic audio functionality through MIDI, it lacks more serious digital audio generation and processing capabilities. However, one good thing about NetLogo is that it provides an API for programmers to develop extensions to the program in Java. We took advantage of this feature and developed OSC-NETLOGO, an Open Sound Control (OSC) [9] extension to NetLogo using the JavaOSC library [2]. This extension allows users to directly map any parameter of a NetLogo patch into any sound processing environment such as Max/MSP, Pd or SuperCollider using OSC.

This article is structured as follows. In section 2 we describe the NetLogo application in more detail, including its history, API. In section 3 we describe the netlogo-osc extension, including its installation and usage. Then, in section 4 we provide examples of mappings of two NetLogo models into sound synthesis using Pd. Finally, in section 5 we discuss the main findings and conclusions of our work.
allows agents make sounds and music using Java’s MIDI capabilities. However, this extension does not allow to send data in a more flexible way and with a better resolution, being this the most important factor for the need of an OSC extension.

3. OSC-NETLOGO: AN OSC EXTENSION FOR NETLOGO

The OSC-NETLOGO Extension for Netlogo adds primitives to Netlogo that allow users to send data from Netlogo to OSC to third party applications. Turtles, Breeds, Links, Patches and any other variables from a Netlogo patch can be routed to a compatible external hardware or software, via Ethernet.

3.1. Usage

The OSC-NETLOGO Extension including example Pd patches and audio files can be downloaded from http://www.rodrigocadiz.com/osc-netlogo. It can be installed in two different ways:

- Put the osc folder into the Extensions folder in the Netlogo path.
- Put the osc folder into a NetLogo patch.

After installing the OSC extension, in order to use it, it is mandatory to add the following line to the top of the procedures tab: extensions [osc].

3.2. Primitives

 osc:port-out

This primitive should be used in the setup of the model. The user can set the IP (String) and port number (integer) of the receiving host, such as:

```
osc:port-out "169.254.183.129*:10000"
```

If port-out is not defined, the extension will use the default parameters, ip: localhost and port: 57110.

 osc:send-agent

This primitive receives as input a string with the name of the OSC tag, followed by the name of an agentset of the model, and names of default or defined variables for this agentset, and sends them out.

The syntax for this primitive is:

```
osc:send-agent "tagname", agentsetname var1 var2 var3 ...
```

Examples:

```
osc:send-agent "myTurtles" turtles "xcor" "size"
osc:send-agent "breeds" cars "XCOR" "age"
osc:send-agent "links" blue-links "weight"
osc:send-agent "patches" patch 2 2 "pcolor" "xcor"
osc:send-variable variableName
```

This primitive receives as input the name of any variable of the model and sends it out. Example:

```
osc:send-variables "decays" decays
```

4. EXAMPLES

We selected two models from NetLogo model’s library and generated audio mappings of some of the parameters of the models using osc-netlogo. The selected models were: Wolf Sheep Predation and Pursuit.

4.1. Wolf Sheep Predation

This Wolf Sheep predation model [7] explores the stability of predator-prey ecosystems. Such a system is called unstable if it tends to result in extinction for one or more species involved. In contrast, a system is stable if it tends to maintain itself over time, despite fluctuations in population sizes.

There are two main variations to this model. In the first variation, wolves and sheep wander randomly around the landscape, while the wolves look for sheep to prey on. Each step costs the wolves energy, and they must eat sheep in order to replenish their energy - when they run out of energy they die. To allow the population to continue, each wolf or sheep has a fixed probability of reproducing at each time step. This variation produces interesting population dynamics, but is ultimately unstable.

In the Pursuit model [6] there is one leader turtle and a group of follower turtles. It is possible to show the leader’s path or hide it and try to guess the path it’s moving along. The idea of the model is that by watching the followers it is possible to obtain clues to the leader’s path. In the sonic implementation of this model, the idea is to guess the leader’s formula by listening to the followers.

The leader moves along a path according to a preselected formula, such as \( y = x^2 \) and starts at the left edge of the world. The leader always moves from left to right by one unit increments along the x-axis. The leader’s y-coordinate is based on the selected formula and the current x-coordinate. Each follower turns to face the leader, then moves forward by a fixed amount.

...
In this example, shown in figures 4 and 5, it is possible to listen to either the leader or the followers. The y coordinate of each turtle controls the frequency of one sine wave oscillator, while the x coordinate is mapped to stereo panning. It is also possible to change the minimum and maximum frequencies in the osc/dataMapping box and set different frequency ranges for each breed as well.

Figure 5. Pure data patch for the Pursuit model

In this sonogram of this example, shown in figure 6, the different pursuits and different paths of the leader and followers can be clearly observed. Also, the sonic shape that each oscillator produces can be audibly mapped to the equations governing the behavior of the leader.

Figure 6. Sonogram excerpt for the Pursuit model

5. CONCLUSIONS

We have developed OSC-NETLOGO, a NetLogo extension that allows to create very complex sonic phenomena by taking advantage of NetLogo’s power for designing and building models of complex systems. The extension is very simple to install and use and gives the possibility of mapping any variable of a NetLogo model into an OSC-enabled audio synthesis engine.

We have provided two examples taken from the available models at the NetLogo’s library of models. These examples provide evidence for the capabilities and potential of NetLogo as a sound generating and processing tool. The behavior of complex systems is something that is very appealing from a musical standpoint, and we hope that this tool could be of aid in the efforts of creating new complex sounds and interesting musical material.

6. ACKNOWLEDGEMENTS

This research was funded by Fondecy Grant #11090193, Conicyt, Government of Chile.

7. REFERENCES


SCORES LEVEL COMPOSITION BASED ON THE GUIDO MUSIC NOTATION

D. Fober, Y. Orlarey, S. Letz

Grame
Centre national de création musicale, Lyon, France
fober@grame.fr

ABSTRACT

Based on the Guido Music Notation format, we have developed tools for music score “composition” (in the etymological sense), i.e. operators that take scores both as target and arguments of high level transformations, applicable for example to the time domain (e.g. cutting the head or the tail of a score) or to the structural domains (e.g. putting scores in sequence or in parallel). Providing these operations at score level is particularly convenient to express music ideas and to compose these ideas in an homogeneous representation space. However, scores level composition gives raise to a set of issues related to the music notation consistency. This paper introduces the GUIDO Music Notation format, presents the score composition operations, the notation issues and a proposal to solve them.

1. INTRODUCTION

The GUIDO Music Notation format [GMN] [4] has been designed by H. Hoos and K. Hamel more than ten years ago. It is a general purpose formal language for representing score level music in a platform independent plain text and human readable way. It is based on a conceptually simple but powerful formalism: its design concentrates on general musical concepts (as opposed to graphical characteristics). A key feature of the GUIDO design is adequacy which means that simple musical concepts are represented in a simple way and only complex notions require complex representations.

Based on the GMN language, the GUIDO Library [2, 3] provides a powerful score layout engine that differentiates from the compiler solutions for music notation [5, 1] by its ability to be embedded into standalone applications, and by its fast and efficient rendering engine, making the system usable in real-time for simple music scores.

Based on the combination of the GUIDO language and engine, score level composition operators have been designed, providing time or pitch transformations, composition in sequence or in parallel, etc. Developing score level composition operators provides an homogeneous way to write scores and to manipulate them while remaining at a high music description level. Moreover, the design allows to use scores both as target and as arguments of the operations, enforcing the notation level metaphor. However, applied at score level, these operations raise a set of issues related to the music notation consistency.

We propose a simple typology of the music notation elements and a set of rules based on this typology to enforce the music notation coherence.

The next section introduces the GUIDO Music Notation format, followed by a presentation of the score composition operations, the related notation problems and the proposed solutions, including a language extension to handle reversibility issues.

2. THE GUIDO MUSIC NOTATION FORMAT

2.1. Basic concepts

Basic GUIDO notation covers the representation of notes, rests, accidentals, single and multi-voiced music and the most common concepts from conventional music notation such as clefs, meter, key, slurs, ties, beamng, stem directions, etc. Notes are specified by their name (a b c d e f g), optional accidentals (b and # for sharp and flat), an optional octave number and an optional duration. Duration is specified in one of the forms: { duration } or { duration, dotting } or { duration, dotting, duration }.

The next section introduces the GUIDO Music Notation format, presents the score composition operations, the notation issues and a proposal to solve them.
2.3. Notes sequences and segments

A note sequence is of the form \(\text{tagname}(\text{note-series})\) where \(\text{tagname}\) is a series of notes, tags, and tagged ranges separated by spaces. Note sequences represent single-voiced scores. Note segments represent multi-tagged ranges separated by spaces. Note sequences represent line tools, or using a graphic environment named GUIDO.

3.1. Operations

Since GUIDO is a concise textual format, it seems natural to use operations commonly applied to text, like cut, copy, and paste, text concatenation, etc. Thus the first idea with the score level operations was based on textual manipulation, extended to music specific operations.

3.2. Notation issues

Actually, the score level composition functions operate on a memory representation of the music notation. But we’ll illustrate the notation issues with the textual representation which is equivalent to the memory representation.

Let’s take an example with the \(\text{tail}()\) operation applied to the following simple score:

\[
\left\{ \text{ clef}^<\text{f}> \right\} \quad \text{d e c}
\]

A raw cut of the score after 2 notes would give \([a, c]\), removing the clef information and potentially leading to unexpected results (figure 3).

3.3. Structure control issues

Table 2 presents a simple typology of the music notation elements: many of the position tags have an implicit time extent category may also give rise to inconsistent notation:

4.1. Notation elements time extent

The GMN format makes a distinction between position tags (e.g. \(\text{clef}, \text{meter}\)) and range tags (e.g. \(\text{slur}, \text{hair}\)). Position tags are simple notations marks at a given time position while range tags have an explicit time extent: the duration of the enclosed notes. However, this distinction is not sufficient to cover the time status of the elements: many of the position tags have an implicit time duration and generally, they last up to the next similar notation or to the end of the score. For example, a dynamic lasts to the next dynamic or the end of the score.

4.2. Structure control issues

Elements relevant to the others / structure control time extent category may also give rise to inconsistent notation: a repeat begin bar without repeat end, a dal segno without segno, a da capo all fine without fine, etc. We introduce new rules to catch the repeat bar issue. Let’s first define a \(\text{pending}()\) repeat end as the case of a voice with a repeat begin tag without matching repeat end.

5.0) when computing the end of a score, every \(\text{pending}()\) repeat end must be closed with a repeat end tag.

6.0) from successive unmatched repeat begin tags, only the first one must be retained.
Music notation is complex due to the large number of notation elements and to the heterogeneous status of these elements. The typology proposed in table 2 is actually a simplification intended to cover the needs of score level operations but it is not representative of this complexity. However, it reflects the music notation semantic and could be reused with other score level music representation language. Thus apart for the reversibility rule that requires the support of the music representation language, all the other rules are independent from the GMN format and applicable in other contexts.

Score level operations could be very useful in the context of batch processing (e.g. voices separation from a conductor, excerpt extraction, etc.). The operations presented in table 1 support this kind of processing but they also open the door to a new approach of the music creative process.

6. REFERENCES


Table 2. Typology of notation elements.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>explicit</td>
<td>duration is explicit from the notation</td>
<td>slur, cresc.</td>
</tr>
<tr>
<td>implicit</td>
<td>element lasts to the next similar element or to the end of the score</td>
<td>meter, dynamics, key</td>
</tr>
<tr>
<td>others</td>
<td>structure control and formatting instructions</td>
<td>coda, da capo, repeats</td>
</tr>
<tr>
<td>misc.</td>
<td>notes in the range do not separate tags.</td>
<td>new line, new page</td>
</tr>
</tbody>
</table>

7) from successive repeat end tags, only the last one must be retained.

No additional provision is made for the other structure control elements: possible inconsistencies are ignored but this choice preserves the operations reversibility.

4.3. Operations reversibility

The above rules solve most of the notation issues but they do not permit the operations to be reverted: consider a score including a slur, sliced in the middle of the slur and reverted by putting the parts back in sequence. The result will include two slurs (figure 5) due to the rules 1 and 3) that enforce opening opened-begin tags and closing opened-end tags.

Note that Advanced GUIDO allows range tags to be expressed using a Begin and End format (e.g. \l slur\Begin, \l slur\End instead \l slur\ (range)). This format is handled similarly to regular range tags and the open parameter is also implemented for Begin/End tags.

Figure 5. A score sliced and put back in sequence

To solve the problem, we need the support of the GMN language and we introduce a new tag parameter, intended to keep the history of range tags and to denote opened-end and/or opened-begin ancestors. The parameter has the form:

\open\type\n
where type is in [begin, end, begin-end], corresponding to opened-begin, opened-end, and opened-begin-end ancestors.

Next, we introduce a new rule for score level operations. Let’s first define adjacent tags as tags placed on the same voice and that are not separated by any note or chord. Note that range tags are viewed as containers and thus, notes in the range do not separate tags.

8) adjacent similar tags carrying an open parameter are mutually cancelled when the first one is opened-end and the second one opened-begin.

For example, the application of this rule to the following score:

\l anytag\open\end\f g \l anytag\open\begin\f w \n
will give the score below:

\l anytag\f g \f e \n
Note that Advanced GUIDO allows range tags to be expressed using a Begin and End format (e.g. \l slur\Begin, \l slur\End instead \l slur\ (range)). This format is handled similarly to regular range tags and the open parameter is also implemented for Begin/End tags.

ENGRAVING–HAMMERING–CASTING: EXPLORING THE SONIC-ERGOTIC MEDIUM FOR LIVE MUSICAL PERFORMANCE

Edgar Berdahl
Audio Communication Group
TU Berlin, Germany

Alexandros Kontogeorgakopoulos
Cardiff School of Art & Design
Cardiff, United Kingdom

ABSTRACT

Engraving–Hammering–Casting is a live music composition written for two performers, who interact with force-feedback haptic interfaces. This paper describes the philosophy and development of the composition. A virtual physical model of vibrating resonators is designed and employed to generate both the sound and the haptic force feedback. Because the overall system, which includes the physical model and the coupled operators to it, is approximately energy conserving, the model simulates what is known as ergonomic interaction.

It is believed that the presented music composition is the first live composition, in which performers interact with an acoustic physical model that concurrently generates sound and ergotic haptic force feedback. The composition consists of three sections, each of which is motivated by a particular kind of craft process involving manipulation of a tool by hand.

1. BACKGROUND

Physical modeling has been employed for decades to synthesize sound [5, 16, 15]. In real-time applications, the approach is typically to compute difference equations that model the equations of motion of virtual acoustic musical instruments [9]. However, besides merely imitating pre-existing musical instruments, new virtual instruments can be designed with a computer by simulating the acoustics of hypothetical situations [6], creating a “metaphorization of real instruments.” Sounds generated using physical models tend to be physically plausible, enhancing the listener’s percept due to familiarity [7, 14].

Besides synthesizing sound, a physical model can also be employed concurrently for synthesizing visual feedback and haptic force feedback. When these feedback modalities are provided concurrently to a human, the sensory percepts can fuse in the brain of a human and provide a distinctive sense of immersion. The ACROE-ICA laboratory has a long history of working in this area [10], and they have developed extraordinarily high quality hardware for synthesizing haptic force feedback for musical applications [13]. They have also introduced key terminology into the discourse, as outlined in the book “Enaction and Exactive Interfaces: A Handbook of Terms” [12].

In this paper, the term ergonomic interaction will be used. A human interacts ergotically with a system when the human exchanges significant mechanical energy with it and the energy exchange is necessary to perform a task [12]. For example, employing a tool to deform an object or move it is ergotic. Bowing a string or playing a drum is also ergotic. There is a mechanical feedback loop between the human and the environment: the human exerts a force on the environment, and the environment exerts a force on the human. In ergotic interaction, the user not only informs and transforms the world, but the world also informs and transforms the user [12].

As far as the authors know, there has never been a portable musical act that explored the musical applications of simulated ergonomic interaction in live performance. This paper describes the development of a new composition in this area.

2. HUMANS USING TOOLS

The authors are inspired not only by the way people interact with traditional acoustic musical instruments, but also by the way people interact skillfully with tools in general. Indeed, seasoned craftspersons leverage thousands of hours of experience in operating tools. They can almost imagine that a favored tool is an extension of their body, allowing them to focus more on the result than on the tool itself [8]. They use the tool efficiently to preserve energy, while often making graceful gestures to achieve an aesthetically pleasing result.

Interaction with tools for craft was emphasized at the Victoria and Albert Museum in London. The “Power of Making” exhibition presented over 100 crafted objects and provided a glossy outline processes used to make the objects [18]. The following processes were particularly inspiring: “carving, cutting, carving, drawing, forging, glassblowing, grinding, hammering, incising, milling, molding, painting, polishing, striking, tapping, welding, woodturning.” These words provided a strong concept and dictated the form and the sonic qualities of the composition.

3. PORTABLE, DURABLE, AND AFFORDABLE HARDWARE

Prior research has focused on accessible haptic hardware for musicians [3]. In contrast with precise yet expensive and fragile devices designed for simulating surgery,
such as those manufactured by Sensable,[1] it was essential to use devices that are more affordable to musicians and more durable. For this reason, the authors have recently been using the NovInt Falcon device, which is a commercial gaming device with USB interface.

Figure 1 shows a human hand gripping the Falcon device. It does not look as artistic as we would prefer, but it satisfies our requirements for now, and it operates in three dimensions. It measures position in the XYZ Cartesian coordinate space, and it can exert a force in the Cartesian coordinate space. Furthermore, an open-source driver is available for the NovInt Falcon for Mac OS, Linux, and Windows, and this driver has been compiled into both MaxMSP and Pure Data (pd) objects, making it easier to access the device for computer music applications.[1, 2]

![Figure 1. NovInt Falcon haptic force-feedback device.](image)

**Figure 1.** NovInt Falcon haptic force-feedback device.

### 4. MODEL

The authors of this paper designed and implemented a reconfigurable model that allows the performer to experiment with sonic-ergotic "sounds" that could correspond to crafting processes listed in Section 2. The shape is simple so that we could choose to expand upon it someday in future compositions. In the model, the musician reaches inside a virtual shape and can interact with the sides (see Figure 2). The square has been used because it is the only regular polygon with angles of 90°, allowing the hand to quickly move around striking all sides without getting stuck in any corners, while leaving open the possibility to bounce back and forth within one corner at will. Since the model is two-dimensional, the performer is allowed to move freely within the third dimension.

Each of the four sides is modeled as a rigid side moving in and out according to a lumped model. The lumped model is reconfigurable, and the ergotic interaction is simulated using a form of the Cordis-Anima equations for simplicity.[11]

According to the authors’ opinion, the simplest musical model is that of a single mechanical resonator, which vibrates only at a single frequency when vibrating freely.

![Figure 2. Hand reaching inside of a square to interact with it](image)

**Figure 2.** Hand reaching inside of a square to interact with it

It is enjoyable to interact with simple models such as this one, particularly while making early explorations of the sonic-ergotic medium; however, it has been decided eventually to add additional resonances to each side in order to enable a wider range of sounds. Thus, each side’s lumped model corresponds to the mechanical equivalent diagram in Figure 3, in which the blue arrow emphasizes the fact that, at least for the purpose of modeling the sound and ergotic interaction, the movement is assumed to be orthogonal to the surface.

For example, the ith resonance is modeled by the mass $m_i$, which is connected to mechanical ground by a spring $k_i$ and damper $R_i$ in parallel. The performer interacts with the ith resonator by a similar parallel link combination of spring $k_i$ and damper $R_i$, with the exception that $k_i$ and $R_i$ only engage when the position of the haptic force-feedback device is beyond the position of $m_i$. In other words, $k_i$ and $R_i$ allow the performer to push into the mass, but only when the performer is touching the mass. The mass $m_i$ does not “stick” to the performer. This contact spring-damper link $(k_i, R_i)$ element is referred to as the BUT element in the Cordis-Anima formalism.[11] There is a separate link $(k_i, R_i)$ for each resonator so that the tuning of each resonator is independent of the other resonators. It is inspiring to note that many diverse sounds can be obtained with this basic model, simply by employing different physical gestures and by adjusting the parameters of the model.

![Figure 3. Mechanical equivalent diagram for hand/haptic device touching five independent resonators on the right-hand side](image)

**Figure 3.** Mechanical equivalent diagram for hand/haptic device touching five independent resonators on the right-hand side

### 5. COMPOSITION

**Ergotic interaction** is an integral part of our compositional medium. We do not merely synthesize or create the sound; instead, we transform and deform it, while it transforms and deforms us physically and mentally. Ergotic interaction inextricably links the gesture of the musician with the sound. We believe that the audience can comprehend this linkage and appreciate it, as we explore the new possibilities of artistic expression enabled by the sonic-ergotic medium.

#### 5.1. Structure

The composition consists of three sections. In the first section, the performers interact with model parameterizations designed to evoke perceptions of engraving. With high resonance frequencies and low masses for the resonators, the sound is delicate and responds intimately to the small, precise movements made by the performers.

In the second section, the resonators are re-tuned to sound more like pieces of metal or bells. The performers make hammering gestures to make melodic-like passages.

Finally, in the third section, the $k_i$ and $R_i$ parameters of the contact links are varied rhythmically in time. Through this modulation, the virtual instrument seems to gain the ability to exert forces on the performer. It asserts a rhythmic form on the gestures of the performers, as if it were casting the performers’ gestures into a specific form.

#### 5.2. Score

The score for the composition consists of six staves, which are noted in a special manner but also contain traditional marks from Western music notation such as markings, dynamics, etc. The first staff describes which sides the first performer should play and at what time (see Figure 4, top). The "F" note describes the right side, the "a" note indicates the left side, "c" indicates the bottom side, and "s" indicates the top side.

![Figure 4. Top two staves indicating to the two performers when to play which sides](image)

**Figure 4.** Top two staves indicating to the two performers when to play which sides.

**The stiffness** $(k)$ and **damping** $(R)$ interaction parameters are prespecified by the score and not under the control of the performers. The lower four staves of the score specify how $k_i$ and $R_i$ interaction parameters vary during the composition. In the excerpt from the engraving section shown in Figure 5 (left), the interaction stiffness remains low for both performers while the interaction damping gradually increases over five bars for both performers.

Figure 5 (right) shows another example in which the damping remains generally low for both performers. The stiffness for performer one varies periodically to emulate engraving, and after three bars, the stiffness for performer two also begins to vary to emulate casting for performer one (see Figure 5, left). Through the variation of the interaction parameters, the haptic force-feedback device asserts its influence over the performers, in a sense casting their gestures into a form that suits the model’s programming.

6. **CONCLUSIONS**

The form of the composition is shaped by the affordances of the force-feedback device. The NovInt Falcon is designed for simulating interaction with virtual tools, and the composition explores interaction with tools within part of the sonic-ergotic medium. The authors also explore the limitations of the force-feedback device. Because there is a delay in the feedback control loop of the device, it will become unstable for sufficiently large $k$ and $R$. In this case, the device will tend to chatter when coming in contact with the virtual resonators, which produces a sound characteristic of the haptic drum.[4] The chattering interaction is not ergotic, but it is nevertheless interesting because it could not normally occur without the "external" energy source of the force-feedback device’s motors. Indeed, in contrast with other human-input devices, haptic force-feedback devices allow for the possibility of the device to assert partial control of a performer.[2, 17]. In the context of the current composition, the devices only behave assertively for short time frames, in order to augment and accentuate the gestures of the performers, as left up to the volition of the performers.

7. **REFERENCES**

[1] E. Berdahl, A. Kontogeorgakopoulos, and D. Overholt, “HSP v2: Haptic signal processing with ex-

---

INTERRUPTING THE NETWORK: AN EMBEDDED APPROACH TO NETWORK INSTRUMENT CREATION

Tom Davis
University of Bournemouth
Poole House
Talbot Campus
BH12 5BB
tdavis@bournemouth.ac.uk

Alain Renaud
University of Bournemouth
Poole House
Talbot Campus
BH12 5BB
arenaud@bournemouth.ac.uk

ABSTRACT

This paper discusses the design, construction, and development of a multi-site collaborative instrument, The Loop, developed by the JacksOn4 collective during 2009-10 and formally presented in Oslo at the arts.on.wires and NIME conferences in 2011. The development of this instrument is primarily a reaction to historical network performance that either attempted to present traditional acoustic practice in a distributed format or utilises the network as a conduit to shuttle acoustic and performance data amongst participant nodes. In both scenarios the network is an integral and indispensible part of the performance, however, the network is not perceived as an instrument, per se. The Loop is an attempt to create a single, distributed hybrid instrument retaining traditionally acoustic interfaces and resonant bodies that are mediated by the network.

The embedding of the network into the body of the instrument raises many practical and theoretical discussions, which are explored in this paper through a reflection upon the notion of the distributed instrument and the way in which its design impacts the behaviour of the participants (performers and audiences); the mediation of musical expression across networks; the bi-directional relationship between instrument and design; as well as how the instrument assists in the realisation of the creators’ compositional and artistic goals.

1. INTRODUCTION

This introduction is not an attempt to provide a comprehensive review of the field of network performance, rather it outlines some general trends in order to provide a context for the work.

Early examples of distributed performance such as the Telematic Circle [1] sought to recreate traditional concert settings over the network by creating a shared environment, or telepresent performance, via the transmission of high-quality video and audio assets over high-bandwidth networks. Such performances often used traditional acoustic instruments in their attempt to produce a performance in which the boundaries between local and remote spaces dissolved into a single co-located experience or shared environment. This was exemplified in projects such as the Playing Apart study [2], which aimed to promote situated types of musicianship over the network. This study aimed to better understand the conditions of playability over a network, especially when long distances were involved. It also devised ways of introducing new technologies and principles to facilitate the playability and increase interactions between geographically displaced musicians despite high latency values. The study used two contrasting pieces of music (slow/fast) allowing experimentation across several aspects of distanced performance, such as playing with large latencies. The study also investigated the impact of interactive technologies, such as spatialised monitoring, video, and simple display using motion capture technology, upon the musicians’ ability to convey gestures via the network.

As networked performance tradition matured, there was a realisation that the acoustics of the network could be utilised as part of the formal compositional process. Early studies of network acoustics [3] stated that, depending upon the distance between nodes and resulting latency, the network can generate acoustical features ranging from reverberation to echo like effects. This paradigm was exemplified in Renaud’s Renditions [4] a multi-site composition which exploits the delay of the network as a catalyst for musical exchange. A related example is Rebelo’s Networks [5], which utilises the network to extend/blend the natural colours of co-located spaces into a hyper-acoustic. In both scenarios the acoustic properties afforded by the network are exploited for an artistic purpose and the acoustics of the network become an integral part of the performance.
A different approach that has been developing in parallel to the above is the use of the network as a structuring device for improvisation. In this scenario, participants exchange control data in lieu of acoustic information, resulting in a composition mirroring the real-time decision making process of the ensemble. An exponent of this technique is PowerBooks_UndPlugged [6], an ensemble that performs with software called Republic [7], a SuperCollider [8] library that facilitates collaborative live coding and sound synthesis across a collection of laptops. Similar software based approaches to structuring networked performance are incorporated into Renaud’s Frequencilator [9], a real-time system providing spectral and temporal structure to a distributed ensemble through an elaborate cuing system.

2. THE JACKSON4

As a network-based quartet, the JacksOn4 formed out of the desire to maintain an ensemble despite its members moving to separate parts of the world. The members of the group have not shared a common physical space for over six years and, as a consequence, the group’s entire musical practice has been conceived and developed solely over the network.

This is in contrast to “most telematic music projects [which are] simple transformations of traditional acoustic music practices – meaning that the music was conceived in traditional acoustic environments and later reproduced in a telematic environment” [10]. Further, the novelty of connectivity, which drove the majority of past telematic projects, no longer intrigues us. Also, the use of networks as an externalised channel to relay control or acoustic data between sites did not seem to treat the network as an instrument, but rather relegated it to its traditional role as a communications system.

Therefore, the initial aim of the JacksOn4 was to create an instrument that embodied the conceptual and practical aspects of network at its core by designing a tangible acoustic interface. This process is repeated again and again, with sound created at one node transmitted through the network to the acoustic body of the next node. In this way, the instrument can be considered one acoustic entity that is distributed across multiple sites.

A typical performance, such as that presented at NIME 2011 [12], begins with a brief structured improvisation during which the acoustic interfaces are used to excite the network. As material is processed and reintroduced into the feedback loop, the composition mutates from the original concrete sources to abstract invention. This process continues through successive generations until a predetermined time or a point at which the composition naturally concludes. The result is the performance of an integrated meta-instrument that provides the potential for a collaborative, emergent composition, with no one artist being the sole performer or composer. In this sense, we believe we are creating truly authentic compositions of, for, and by the network.

As ensemble members are distributed, remote acoustics and network characteristics also shape the performance. At NIME 2011, three performers were onstage in Oslo while one performer contributed from Norwich, UK. Other performances have seen similar configurations such as two players in Bournemouth, UK, one in Norwich, UK, and one in Tromsø, Norway. In concert, remote participants are represented locally by a single loudspeaker, which is strategically placed on stage en lieu of the remote performer. This allows the local audiences to hear the movement and transformation of audio through the networked instrument, as if all four nodes occupied a single space.

In addition to the purely acoustic audio material introduced into the system, the ensemble has taken to augmenting the performance with live electronics generated by applications such as Max/MSP [13], PureData [14], and SuperCollider [8]. As we are using computers to establish the instrument’s connectivity, it seems like a natural affordance. A Max/MSP patch acts as a control panel for each performer, allowing the transmission of processed or unprocessed audio to the network via the JackTrip [11] Audio Server [15] and subsequently JackTrip [11].

3. THE LOOP: A DISTRIBUTED NETWORK INSTRUMENT

In its current configuration The Loop consists of four acoustic nodes connected in series creating a feedback loop. Each node is comprised of three components: an acoustic interface, a contact microphone, and an audio transducer. Intrinsic to the instrument is the networked loop linking one node to the next. This is implemented through the JackTrip [11] audio application. At the local node, the interface is excited (struck/plucked/bowed) and the attached contact microphone picks up the waveform. This signal is sent over the network to the next node in the series, where it is used to drive a transducer affixed to the remote acoustic interface. This process is repeated again and again, with sound created at one node transmitted through the network to the acoustic body of the next node. In this way, the instrument can be considered one acoustic entity that is distributed across multiple sites.

As each node is both an acoustic resonator and a device for creating sound, performers can hear the sound as it passes through their node and choose to modulate it or add to it through direct physical interaction with the local acoustic interface. This reinforces the idea of the distributed instrument as a single instrument and affords many opportunities for tactile/tactile interactions between participants who continuously contribute to the overall resonance/feedback of the system.

4. THE IMPORTANCE OF INTERFACE

Through our explorative process, we have discovered that the design of an instrument for networked musical activity necessitates special consideration. We argue that the musical instrument is the interface to the network, and this interface is what we use to communicate our musical gesture. Thus, instrument design affects how we relate to and explore the network in performance. As Boyle states, “While interactions between participants of network-based works can occur over spatially distributed or localized environments, and the interactions and explorations themselves can be synchronous or asynchronous, the design of the interface through which these explorative behaviours are mediated is of equal importance.” [16] It is our contention that if an instrument inherently embodies the network, then gestures afforded by that instrument will be more suitable for networked performance than any other approach. In the development of The Loop, we came to define the interface as a collection of tangible objects capable of exciting and resonating an acoustic waveform. It is this physical interface with which we interact, and we consider the hardware and software components (e.g. JackTrip), that merely provide connectivity, to be of secondary importance.

In Heidegger’s [17] terminology, in the general act of performance, one can consider the musical instrument to be ‘ready-to-hand’. For the musician performing with the instrument the design controls the creation of music collapses, such that he feels directly connected to the act of music creation. One could argue that the performer’s consciousness in which there is a merging of action and awareness and any perceived division between the performer and the acoustic environment disappears. By incorporating acoustic interfaces and feedback, The Loop allows the network to withdraw into the instrument such that the network itself becomes part of the body of the instrument. In this conception, the performer has no conscious experience of the instrument or the network as an independent entity; rather, the instrument (and the network) becomes phenomenologically transparent [18].

This approach can be contrasted to the use of traditional acoustic instruments in networked performance, where you could argue that rather than acting as part of acoustic body of the instrument itself, the network functions only to enhance the acoustic reach of the instrument. Consequently, when performers accustomed to traditional acoustic scenarios play across the network, the natural performative flow is broken and the relationship between the performers and the instrument changes. In this situation, the musical instrument becomes ‘present-at-hand’. In this scenario the instrument steps becoming transparent and the participants (performers and audiences) become aware of its presence as a tool for mediating the network. Linked and complimentary to this is Bourdieu’s [19]
concept of habitus, the “practical sense” that “inclines agents to act and react in specific situations” that is “the result of a long process of inculturation”. From a performance viewpoint, the habitus of a performer has been established through long hours of instrumental practice during which the relationship between motor action and sensory perception has been built. In performing over the network this relationship between action and perception is challenged, and for a while at least, this difficulty in performing produces a breakdown of this compositional relationship. We are not suggesting that performers cannot adapt to these new scenarios, or even that this breakdown of relationship is always undesirable, but we believe The Loop provides the sort of instrument that inherently and transparently sits and works within the network. In addition, it is an instrument designed and created by musicians who have only known this network performance tradition. In a way, they have become virtuosos of their own creation and, consequently, The Loop provides the opportunity for developing a unique style of composition.

5. THE INSTRUMENT AS COMPOSITION

From a compositional standpoint we were interested in making an instrument that embodied some of the musical challenges of performing networked pieces into a design, such that the construction of the instrument itself could be interpreted as the starting point of the compositional process. Historically, instruments have been developed both practical and musical problems encountered by the composer: Cage’s wish for a portable percussion ensemble resulted in the prepared piano, while Wagner's desire to blend the timbre of the horn created the eponymous Wagner tuba. As a distributed improvising ensemble, our compositional challenges were to coordinate individual musical effort across the network and ensure our compositional challenges were to coordinate individual musical effort across the network and ensure that the musical decisions made by a group of performers and audiences) are distributed across of contrasting improvisational styles. As musical material is transferred from one acoustic interface to the next, via electroacoustic transduction and digital processes, the temporal and spectral qualities of the original performed acoustic signal are modified. The extent of this change in quality is both a factor of the types of materials the acoustic signal passes through (metal plate/woden block), as well as the number of signals a signal might encounter before being auditioned. Thus, musical material created at the local node will receive a unique treatment at each successive remote node down the line. This progressive shaping of timbre and gesture provides the composition with an orchestration and arrangement that could not be replicated by (or transposed to) any other instrument. The instrument is one of a kind, and so is the resultant composition. Further, as we are sharing the control of one single distributed instrument, we become more sensitive to each other's contribution to the overall composition. We have to be, because it is relatively easy to overdrive an instrument that is in a state of perpetual feedback. We share a communal responsibility of regulating the progression of the signal through the loop, by either dampening the remote resonant body or intentionally driving more signal through our node by boosting the amplification of the transducer driving our interface. This action of collectively modulating another performer’s sound is an activity typically not afforded to performers and is unique to this type of instrument design.

Initially a laptop ensemble, the JacksOn4 have progressed away from computational processes, which theoretically can provide any gesture or timbre possible, and embraced natural acoustic parameters that greatly limit the scope of what is performable. We recognise that The Loop, through its incorporation of a feedback process, along with providing the musicians with an interface to modulate the composition in real-time at the local node, imprints a unique compositional structure upon our improvisations, unifying our collective performances.

6. CONCLUSION

In taking an embedded approach to creating a distributed networked instrument we have highlighted the tight relationship between instrument, performer, and composition that The Loop provides. We feel this tight integration has afforded us the opportunity to go beyond merely performing across the network. By design, the instrument affords the chance to utilise the network in a much more idiomatic way, allowing the network to take on a definitive musical role in our creative practice.

7. REFERENCES

REAL-TIME AUDIO SYNTHESIS IN A WIRELESS INTERACTIVE SENSOR PLATFORM

Da-Lei Fang, Yi Qin, Guang-Bin Chen
Shanghai Conservatory of Music
Dept. of Music Engineering
Shanghai, China

Jia-Liang Lu
Shanghai Jiao Tong University
Dept. of Computer Science & Engineering
Shanghai, China

ABSTRACT

In this paper, we introduce the real-time audio processing for live performances in Wireless Interactive Sensor Platform (WIS platform) that is built by our team. WIS platform is designed based on on-body motion sensors and processing computers to support dynamic and interactive applications. It is composed of a capture system, a processing unit and an audio-visual display control unit. We embed several general real-time sound synthesis modules including additive synthesis, FM synthesis, scene sound synthesis and effect processing functions for audio synthesis to satisfy the distinct needs of the live performances. The wireless communication between multiple on-body sensor nodes and display control units has been investigated to enable smooth gesture recognitions and offer real-time audio and visual experiences to users. We also illustrate the audio synthesis through a recent interactive installation Air-ergy.

1. INTRODUCTION

Recently, audio synthesis through a real-time interactive system has become a popular research direction in computer music. Meanwhile, advances in wireless communication technologies and the reducing size of sensors have empowered the use of wearable sensors in interactive music performances. Many sensor systems [7, 10, 6, 11, 12] were developed for interactive music performances. Each of these system achieve a specific application. In [7], authors had an interesting discussion concerning the integration of the dancing body and the musical body in the Performance of Electronic Music. In [6], authors proposed a wireless system for gesture following for music pedagogy. In [10], the authors investigated to enable smooth gesture recognitions and offer real-time audio and visual experiences to users. We also illustrate the audio synthesis through a recent interactive installation Air-ergy.

2. WIS PLATFORM

WIS platform consists of three main parts: capture system, processing unit and display unit. However the primary purpose of WIS platform is to implement an open architecture to combine these three parts. One can easily adapt either capture or display part regarding to application’s requirements and circumstances. The system architecture of WIS platform is given in figure 1.

2.1. Capture System

The capture system includes wireless on-body sensors and receivers. The sensing part is a SparkFun 9 Degrees of Freedom (9DoF) Razor IMU [2] that incorporates a triple-axis gyro, a triple-axis accelerometer and a triple-axis magnetometer. In addition, a modified XBee adapter with an Xbee 2 RF module [5] is used as the wireless communication module whereby it sends and receives data via XBee 2 RF module [5]. The processing unit is composed of three modules. Initialization module is responsible for parameters and state initializations as well as for loading DSP Scene control module loads pre-defined scenes with regards to the environment definition and the results of sensor data processing module. It sends OSC packets through LAN to control audio and visual display unit. Parameters verification module serves as a tool to visualize the parameters and to test them on-the-fly.

2.2. Processing Unit

The processing unit is composed of three modules. Initialization module is responsible for parameters and state initializations as well as for loading DSP Scene control module loads pre-defined scenes with regards to the environment definition and the results of sensor data processing module. It sends OSC packets through LAN to control audio and visual display unit. Parameters verification module serves as a tool to visualize the parameters and to test them on-the-fly.

2.3. Audio-Visual Display Unit

The audio display unit consists of mixer, effect processing and multi-channel panner modules. It interacts with processing unit to play different scenes as well as to generate sonic elements related to captured data. It is worth noting that processing unit and audio display unit are realized on the same computer to minimize the influence of delay on the audio system. Therefore, the control signal is directly sent through MAX/MSP program, but does not go through LAN. In section 4, the real-time audio synthesis in WIS platform will be more detailedly presented. The visual display unit is implemented on the second computer for particle generation and control, as well as texture mapping and render. If multi-projector system is necessary for the application, then the morph should also be activated in this part.

3. REAL-TIME SYNTHESIS CONTROL

The OSC packets sent from capture system such as the yaw, pitch, roll and 3-D acceleration are processed, then the processing unit will send control packets to audio and visual units to interact.

3.1. Acceleration Data Processing

The acceleration data is processed and output by two ways: continuous output and state output. Continuous output represents any movements, such as turn around and foot-lifting. Firstly, the processing unit gets the differences of acceleration in a small period. Then the absolute acceleration value will be computed. Finally, an envelope follower is used to smooth out the change. To make adjustment easier, all the parameters above can be stored into presets. So the same structure can be used in many sub-patches with different parameters presets. The continuous output is sent to many sub-patches, such as step recognition and wind synthesis. State output can represent any movement states, such as fast movements or sudden stop. The continuous output passes through a series of comparators. Then four state triggers are output according the value of continuous output: slow, mid, high and fastest. Wind 1 and 2 use these data to change the timbre.
3.2. Step Recognition and Position

The step recognition is essential for in-door tracking and is widely used in Dead Reckoning solutions [8, 9]. In our realization, the step recognition is based on yaw, pitch and roll and 3-axis acceleration. Firstly, all the raw data go through the AIO with offset and adjustment. By setting time switch, the unit is able to synchronize data from different on-body sensors for processing. Secondly, differential and envelope operations are performed on the data series to output a continuous acceleration variation.

The 3D acceleration is enabled to get direction and a series of values. A threshold value should be defined to trigger the step. A number of tests are executed to get this threshold. It is also possible to implement a self-learning algorithm to get this threshold. In order to eliminate repetitive trigger, a switch is used with time interval for the possible following trigger. During the exhibition, it turns out that the threshold range is applicable for 90% of utilization.

4. AUDIO SYNTHESIS IN WIS PLATFORM

4.1. Wind Scene Sound Synthesis

Wind Scene is a scene triggered by the rotation data from the sensor on the user’s shoulder. This scene synthesis a breeze/wind sound effect. "Pink Noise" is used as the raw sound material for Wind Scene. In the first version, a low pass filter with resonance was used. Although this was enough to simulate a wind sound, we implement a second version, and try to bring more musical elements into the wind sound. 12 band pass resonators are used. The cutoff frequency of each filter is a multiple of the cutoff frequency of the current filter. In this way a series of harmonics are obtained. The cutoff frequency is between 100 and 400Hz, and the multiples vary from 1.05 to 1.30. The values of these parameters are triggered by the 3-level acceleration state output (low, mid and high) from sensing data processing part. The volume of each resonator is controlled by the continuous acceleration data. Each carrier has a little detuned to create chorus effect. The output of the FM synthesis is fixed to 4 channels. FM synthesis is used as a pad sound. We also implement a sample record/playback system. The user can record his/her voice before the experience, and these samples are played in a granular synthesizer during the experience.

Figure 3. Wind scene diagram.

Wind Scene is a scene triggered by the rotation data from the sensor on the user’s shoulder. This scene synthesis a breeze/wind sound effect. "Pink Noise" is used as the raw sound material for Wind Scene. In the first version, a low pass filter with resonance was used. Although this was enough to simulate a wind sound, we implement a second version, and try to bring more musical elements into the wind sound. 12 band pass resonators are used. The cutoff frequency of each filter is a multiple of the cutoff frequency of the current filter. In this way a series of harmonics are obtained. The cutoff frequency is between 100 and 400Hz, and the multiples vary from 1.05 to 1.30. The values of these parameters are triggered by the 3-level acceleration state output (low, mid and high) from sensing data processing part. The volume of each resonator is controlled by the continuous acceleration data. Each carrier has a little detuned to create chorus effect. The output of the FM synthesis is fixed to 4 channels. FM synthesis is used as a pad sound. We also implement a sample record/playback system. The user can record his/her voice before the experience, and these samples are played in a granular synthesizer during the experience.

5. AIR-ERGY: AN INTERACTIVE AUDIO-VISUAL EXPERIENCE

In this part, we present Air-ergy as a work of interactive installation (videos are available on YouTube). It is exhibited in the 3rd Shanghai International Electronic Music Week [3] in October 2011. Figure 4 shows a scene of Air-ergy. This work emphasizes the interactions between human body and its environment. By making use of WIS platform in a particular space, the demonstration of this work is triggered by the user’s body action. The idea of this installation is to show the kinetic energy of a man in real world through interactive sound effects and videos in a particular space. This is a closed loop system that embodies a real physical world transitioning to a virtual cyber world. Table I gives the audio synthesis of a set of gestures in Air-ergy.

Table 1. Interactions implemented in Air-ergy

<table>
<thead>
<tr>
<th>User gesture</th>
<th>Audio-visual experience</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slow body rotation</td>
<td>drifting breeze</td>
</tr>
<tr>
<td>Rapid body rotation</td>
<td>swift and wide oscillation of sound wave</td>
</tr>
<tr>
<td>Raise foot</td>
<td>particle floating up, virtual footstep sound</td>
</tr>
<tr>
<td>Leave from the area</td>
<td>the particles settle down, deformed user voice looming</td>
</tr>
</tbody>
</table>

LAN using OSC. The vvvv adds a fast up ramp to the height of the particles. If no more steps are received, the height will fall slowly. If more steps are received, the height will be added cumulatively. The height is also sent back to Max/MSP through LAN using OSC. This parameter is used to determine the volume of each partials in wind 2 and additive synthesis.

6. CONCLUSIONS

WIS platform is an open platform designed for interactive music performance. We set the basic sound synthesis modules in real-time sound synthesis unit. Users are free to add other sound synthesis modules into WIS platform as well as redo the mapping relationship according to the specific requirements. Moreover, the wireless sensor systems or motion capture systems can be easily interfaced with WIS platform. Air-ergy is a good example that employ of WIS platform. We are currently working on the WIS platform to include more general audio synthesis modules and map the general parameters.

7. ACKNOWLEDGMENTS

We would like to thank Christophe Lefebvre at GRAMIE and for a number of suggestions and discussions on this project as well as Qinghua Gu at CAM for his contribution to this project. Chao-Lun Ni at SITU also has helped for the installation of Air-ergy.

This work is supported by NSF of China under grant No.61102010, the Key Science and Technology Program of STCSM under grant No.10511500900, Shanghai MEC Innovation Project No.12Y5080 and Shanghai Key Laboratory for Art of Musical Acoustics. The authors from SHCM are supported by Construction Program for Innovative Research Team, Shanghai Institutions of Higher Education (2nd Phase) and Talents Training Mode Innovation Experimental Zone Construction Project by MOE and MOF of China.

8. REFERENCES

PERFORMING ARTICULATION AND EXPRESSION THROUGH A HAPTIC INTERFACE

Lauren Hayes
University of Edinburgh
Department of Music, Edinburgh, UK
laurensarahlhayes@gmail.com

1. INTRODUCTION

A violin, played at professional standard, can be likened more to a localised instrument of torture (with its complimentary disciplinary rewards), than a harmonious continuation with human agency. Why is there no impetus to develop a violin that blends ergonomically with the player? [10]

When playing acoustic instruments, the haptic perception involving both tactile sensors in our skin (especially in the fingertips, hands or lips), as well as our kinesthetic perception of the position and movement of our muscles and joints, is pivotal within the complex relationships between performer, instrument, and the resultant sounds produced. It is precisely these ongoing negotiations and reassertions through multimodal feedback loops that lead to the diversity of achievable musical expression. However, the audio-tactile link has been largely missing in systems of digital music. Instruments can no longer be described to be of those resonating bodies of physical constructs; indeed the sound sources of DMIs cannot usually be linked to real-world vibrating objects. Emmerson writes that, as performers, we “sense this loss of tactile location” [4]. MIDI controllers are often modelled on the studio paradigm, consisting of knobs and faders, and tend not to offer the types of interaction that make use of physical forces such as pressure, friction, collision, resistance, and so on. Even instrument-based MIDI controllers not only fail to communicate the whole picture in terms of the performer’s gestural information, but do not go far enough in allowing the vital unvarnished types of engagement to manifest between performer and instrument, as hinted at by Rebello and Coyne’s quotations. As research into tangible computing develops, the importance of how we engage with the digital world through our physical interactions is being increasingly highlighted. The incorporation of haptic feedback into mobile phones, and more recently tablet computers, is one such example. Developments in the field of haptic technology directly related to music are steadily increasing. These include ways of using our tacit knowledge of how we interact with physical objects as a means of mapping input gesture to digital audio parameters [8]. Other work specifically uses the already acquired motor-skills of a musician as the basis for haptic instrument design [2]. In related research, I have attempted to reintroduce the tactile sensation of sound to the performer by incorporating vibrotactile feedback within the performer-instrument coupling [6].

In what follows I will attempt to argue the case for employing tangible forces as a means to enhance the potential for sophisticated expression and articulation of performed digital music. I will propose that:

1. Diverse musical expression is achieved through constant multimodal negotiation between performer and instrument.
2. Exploring the resistance of an instrument, either physically or virtually, can help to craft a performance.
3. Haptic technology can help to create shared participation between performer, instrument and performance space.

2. SOUND SCULPTING WITH A HAPTIC CONTROLLER

During his keynote address at the International Computer Music Conference, 2011, University of Huddersfield, Simon Emmerson suggested that the idea of sculpting sound was now “a metaphor that could be made ‘real’ through suitable haptic interfaces and three dimensional representation” [1]. What follow will offer a modest step towards a realisation of this idea, where digital audio can indeed be moulded and articulated through three-dimensional (3D) tangible interactions. In this way, not only do we rely on auditory feedback to navigate between sounds, but we are also guided by ongoing tangible exchanges with the instrument.

2.1. Repurposing a Commercial Game Controller

The Novit Falcon is an affordable, commercial 3D game controller, providing up to 9 novemens of resistant force-feedback to the user [1]. While this is of a much lower bandwidth than, for example, Claude Cadoz’s Modular Feedback Keyboard [3], one of the first haptic musical instruments, or “gestural force-feedback transducers”, it is nonetheless receiving attention as a low-cost solution to designing haptic DMIs. The Haptics Lab 2, at CCARM, Stanford University, is an audio-performance workshop specifically based around the Falcon.

Open source software is available from Nonpoly-nomial Labs 3, enabling the Falcon to operate within musical programming environments such as Max/MSP or Pure Data 4. Edgar Berdahl, at CCRMA, developed HSP, a simple platform for implementing haptic musical instruments [1], aimed at making the incorporation of haptic technology into DMIs both inexpensive and easy to programme. The toolkit offers a series of basic force-feedback profiles for the Falcon, including virtual walls and springs. Used and adapted in combination with various DSP techniques, new instrument prototypes start to emerge.

The Falcon ball grip can be moved within a 3D space, approximating a cube of side around 11 centimetres. However, depending on design choices, a virtual space of infinite size could be traversed. The haptic technology can provide the experience of palpable (but virtual) objects, surfaces, and indeed textures. Thus possibilities for designing interactions based on a host of different gestures arise, where we can imagine not only bumping into virtual surfaces, but also being able to move through different atmospheres, at various speeds, depending on the viscosity employed 5. Furthermore, interactions that defy usual physical relationships can be created through the HSP’s interface. For example, a simple wall model could disintegrate after being hit with a particular velocity, and reform once the user has passed through to the other side. The meaningful relationships between such interactions and their effect on the musical outcomes must be considered.

2.2. Performance Potential

On trying out the device for the first time, it feels, peculiarly, both new, yet familiar. After initial experimentation within this environment, the possibilities for prototyping new haptic-based musical instruments become obviously clear. The three degrees of freedom give potential not only for macro-movements of the hand, but for further engagement from more of the body. Thus larger gestural movements can be transduced into parametric information, while still allowing for finely focussed articulations. Rebello refers to performances as “multimodal participatory spaces,” somewhere where the performer employs “negotiation of subtlety and the recognition of threshold conditions” [9]. I would argue that by introducing haptic devices, we also increase the potential for this type of engagement.

* Citing around $20 at the time of writing, January 2012. http://sourceforge.net/projects/libnifalcon/

* http://sourceforge.net/projects/libnifalcon/

* http://sourceforge.net/projects/libnifalcon/

* As well as providing cross-platform functionality to include OS X, which is, at a time of writing, unsupported by Novint.

* Viscous damping is implemented in the HSP toolkit through a series of band filters.

Emmerson’s idea of sound sculpting suggests that 3D manipulations would enable us to mould multiple aspects of the sound simultaneously. The very nature of the Falcon affords this type of interaction, given that any movement will produce a minimum of three data streams 6. Thus, the potential for multi-mapping is inherent. The expressive potential of acoustic instruments can, in part, be attributed to the multiple and ongoing negotiations between performer and instrument. For example, playing fast staccato scales on a piano requires continuous assessment by the performer about kinaesthetic and haptic information, such as the bounce of the fingers against the keys, and the speed or acceleration of the hand. Thus, simply by nature

6On the most basic level, x, y and z position can be extracted, before even considering force and resistance.
of its design, the Falcon already goes some way in offering the qualities necessary to make a convincing instrument.

2.3. Development of an Instrument

After testing the various physical models given within the HSP examples, the virtual wall was used most extensively as a starting point, as the action of striking a surface with different objects and forces could be considered to be one of the most primitive sound-producing gestures.

I implemented a visual rendering of a ball bouncing on the wall using JavaScript within Max/MSP, which was initially useful for providing additional visual feedback while developing the system. However, since much of my practice within computer music aims to remove the need to look at a laptop screen while performing (through engagement either with haptic interfaces, or augmented instruments) [7][8], this quickly became redundant.

One of the most interesting aspects of the instrument was that depending on the different force profiles used, it could rapidly change between allowing wild gestures, to a very resistant, even secure, environment where moving through detailed nuances of a sound could be explored. The four buttons on the small ball grip of the Falcon allow rapid switching between different forces profiles and parameter mappings, within a single performance.

3. RUNNING BACKWARDS, UPHILL

This section describes the implementation of the Falcon as a performance instrument within a compositional framework.

Running Backwards, Uphill (2011) [9] for violin, cello, piano and live electronics, attempts to explore the relationships between touch, gesture and timbre by examining the sonic qualities of the acoustic instruments, and furthering these through the use of electronics. The performers are directed to lurch and fall off the keys, or to create the most delicate airy bowed sound. Extended techniques are combined with sound analysis methods, resulting in an informed integration between the two sonic worlds.

3.1. Articulation and Expression

One of the challenges with this composition was to develop a method for articulating the electronic part that could evoke the same expressive qualities within the music as would be expected of the professional ensemble. The musical language of the piece itself is very gestural (see Figures 2 and 3), and thus it was important that the electronic part should also be performed in such a way as to reflect this.

The boundaries of any instrument can be where the most expressive moments emerge, from an unstable clarinet multiphonics, to the point at which a digitally generated pulse turns into pitch. But for instruments within the class of resonating bodies, these boundaries are intrinsically coupled with physical resistance. As Aden Evans explains:

“For his part, the musician resists the resistance, which it is to say, he employs technique... technique is designed to place the instrument's resistance in contact with the musician, to allow him to feel the many dynamics it offers of force and sound.”[5]

Thus, by working with the haptic device, I could successfully couple physical resistance with these virtual boundary areas, in turn allowed for a greater range of potential expression in which the performer could employ technique.

In the first part of the score (see Figure 2), the Falcon is used to play short segments of samples of a prepared piano. With the added resistance of the controller, I was able to make micro-movements along the domains of both start and end points of the sample, as well as the speed of playback of the samples themselves. Without the force to play off, the result would have sounded jerky and ill crafted, whereas pushing through the resistance allowed smooth transitions through the various parameters, producing a legato effect.

In this section, the string instruments play long tremolo lines in which both the speed of the tremolo, and pitch transition continuously. The dynamics also ebb and flow gradually until the entrance of the piano. I wanted the electronic part to reflect these characteristics by transitioning gradually both in timbre, dynamic and pitch. The haptic device gave the performer a sensation, or feeling of moving through the sound, increasing the amount of information available for negotiating the performance.

3.2. Interaction and Engagement

In the middle section of the piece (see Figure 3), the Falcon was used to transduce fast gestural sweeping movements to process various effects (including bit-crushing, feedback and filtering), which were applied to a second set of samples. Here, the piano part leaps through descending cluster chords, in a syncopated manner. The pianist is instructed to play “clumsily, with hands almost falling off keys.” The laptop performer uses a different, non-uniform force profile in this section, which facilitates the “jerky” expression marking noted in the score, but which also provides more resistance at the boundaries of the DSP. In this way, the performer is less likely to, for example, hit the more piercing parts of the audible feedback, as the added resistance would prevent them from reaching that part of the instrument’s range easily.

4. RESULTS AND DEVELOPMENT

The purpose of this paper was not to detail the technical capabilities of the Novint Falcon, as these are thoroughly described elsewhere [1], but rather to describe how employing a haptic device can manifest imagined musical ideas and expressive nuances. Not only through practice, but also by considering the crucial negotiations and feedback loops that take place between performer and instrument, it becomes evident how employing haptic technology can solve some of the problems associated with the performance of live electronic music.

Implementations of different force profiles, combined with visous damping, allow the performer to feel their way through different aspects of digital audio, and sculpt the sound accordingly. In the work described, using resistant forces designed with very simple physical manipulations allowed vastly different types of engagement to be employed within the piece. Through the use of haptics, as a performer, I could more easily position myself within the shared participatory performance space. I will conclude by suggesting that this is a place that we, as composers of digital music, need to more deeply explore.

5. REFERENCES

PLAYING WITH LIGHTS: MUSIC GENERATION USING THE LED CUBE

Dariusz Jackowski, Jakub Stepniewicz

University of Wroclaw

Institute of the Computer Science, Wroclaw, Poland
dariusz.jackowski@ii.uni.wroc.pl  jakub.stepniewicz@gmail.com

ABSTRACT

In this work we present new instrument/sound installation based on the exchange of the information between the components through the light intensity. The system is composed of the 8x8x8 LED cube and simple programmable instruments. Additionally to the description of used software and hardware, we present some ideas for performances and extensions of the system.

1. INTRODUCTION

One of the most popular DIY electronic projects is the LED cube, which in the musical context is used mostly as a visualisation device. Our approach is different - we had chosen to use measured intensity of the light generated by the cube as parameter in music generation algorithms (this approach is inspired by works on image-to-sound mapping by Lesbios [13] and by the Metamath [1]). Additionally we prepared some light-based instruments without internal light sources to use with the cube (the instruments are inspired by products of IXI software [2], theremin, Beamz[3], AudioCubes[4] and the light harp[5]). In this work we describe the hardware (sections 2 and 3), the software(4), possible usage(5) and possible extensions (6) of the system. The audio and video examples can be found on the project website [12].

2. THE INSTRUMENTS

Our main goals, while designing the instruments, were to ensure the easiness of use and replication, flexibility and support for cooperation between players. Each of our instruments consists of the same building blocks - the Arduino board for main processing, one or more light sensors and speakers or buzzers. Arduino[6] is an open-source single-board microcontroller based on the ATmega168 chip. It allows us to build fully programmable instruments. Additionally we are given option to program onboard USB controller, thus changing the way (i.e. hardware type) in which the instrument is perceived by the computer. Due to the lack of the internal light source the instruments are dependent on the external one. Said source can be controlled by ather performer (either as simple as flashlight or as complicated as our LED cube) or it could be uncontrolled ambient light. As sensors we used simple LDRs (light dependent resistors).

3. THE CUBE

3.1. Overview

The heart of the system is the 8x8x8 LED cube built with the integrated ATmega32 microcontroller (the project comes from [7]). The detailed list of parts can be found in [12].

3.2. LED

The base cube construction element is Light Emitting Diode (LED) [1]. To light a diode you need to apply positive voltage to the cathode and negative to the anode.

3.3. How does it work?

With the smart approach to the connection problem the number of needed wires could be reduced to 72 or even to 0.

The software and possible usage of the instruments are described in sections 4 and 5. The detailed list of needed components and support for cooperation between players. Each of the system performances and extensions of the system.

4. SOFTWARE

4.1. Instruments

For the instruments we proposed the following software:

- The most basic usage is as the “optical theremins”. The frequency of the generated tone is directly connected with the reading of the light intensity (or more specifically of the resistance of the LDR). Movement of hand in front of the sensor affects the sound in the similar way as with the theremin (only the aural effect is similar, the mechanism is completely different). We use two potentiometers for calibration and volume. The buttons are used for turning off the sound, recording and playing recorded loop (the time of recording is limited to a few seconds). Loops are useful for organizing group improvisation.

- The finite state machines (FSMs) with light intensity as input and frequency of sound as output. In our experiments it proves to be most useful for generating long sounds for drones or harmony instead of producing one sound as output the FSM can be used to choose one of the prepared loops (this is useful both for the baseline and the internal voices of the harmony).

- Melody generated using Markov chains. Here we use two approaches - in the first one the intensity of light is used instead of random numbers generator for selecting the next note and in the second one the light intensity affects the speed (tempo) of the generated melody. As a source for the data used in the matrices for the algorithm we use tunes in the ABC format from [9].

- As a controller for the software running on the computer. Either without reprogramming the USB controller (for example the Processing language [10]) has libraries for cooperating with Arduino) or reprogram to be visible for the computer as the MIDI controller or keyboard.

Figure 1. LED, a – anode, c –cathode

The cube consists of 512 blue leds. Everyone of them can be separately controlled. Due to the number of leds it would be difficult to connect the separate cables to each of them. Each diode would need 2 connections which would make 1024 wires. In the following subsection we present the solution to this problem.

Figure 2. The 4x4x4 cube, a,b,c,d – anode layers, e,f,g,h –cathode columns

We illustrated it on the fig. 2 which shows smaller version of the cube. First consider the elements labelled anode layers. In each anode layer anodes of every diode are connected to each other. That decreases number of anode connections from 512 to 8. The Leds' cathodes in each column are also connected, which allows us to use only 64x8 wires.

Unfortunately, although the number of needed wires is reduced in this way, it makes impossible to light all diodes at the same time. This isn’t much of a problem, because (due to the perception of vision [8]) human eye perceives lights blinking with rate greater than approximately 25 Hz. As our cube achieves the refresh rate of circa 1000 Hz it can take full advantage of the mentioned phenomenon.

To improve the appearance of the cube we do not use any wires to connect the diodes, but we use their anodes for horizontal connections and cathodes for the vertices ones.

3.4. The Microcontroller

Another very important element of the cube is its controller. It could be divided into three parts, as shown in fig. 3.

- Controller - made using an array of transistors. It allows to select which layer is currently rendered. At the same time only one layer can be selected.

- Column selector - set of Type-D Latches. Selects which leds in current layer will be turned on.

It is also possible to control the cube directly from PC using RS-232 interface. This allows to visualize much more complicated things, that are impossible to generate using integrated microcontroller, due to limitations in both memory and processing power.

Figure 3. General schematics

PC input

Controller

Column selector

LED

- Column selector

- LED

- Controller

- ABC format from [9].

- As a controller for the software running on the computer. Either without reprogramming the USB controller (for example the Processing language [10]) has libraries for cooperating with Arduino) or reprogram to be visible for the computer as the MIDI controller or keyboard.

- Column selector

- LED

- Controller

- ABC format from [9].

- As a controller for the software running on the computer. Either without reprogramming the USB controller (for example the Processing language [10]) has libraries for cooperating with Arduino) or reprogram to be visible for the computer as the MIDI controller or keyboard.
4.2. Cube

For the LED cube we propose the following:

- Non-interactive animations. They are useful both for installations and for live performance. In installation their usage allows predicting the effects or replaying installation the same way every time (it depends on the configurations of the instruments).

- In life performance it introduced structure in improvisation of participants. Another possible variant is to introduce human control for choosing previously prepared animations.

- 3D version of Conway’s Game of Life [11]. It has similar advantages to the animations save the fact that it tends to be chaotic at the beginning and periodic afterwards.

- Human-controlled (we tend to call the controlling person the conductor) movable cursor for toggling the state of diodes (we have named it the XOR Snake). The conductor is also able to set whole cube into one of predefined configurations. If used with live players there can be set some additional rules for players actions in accordance to the state of cube.

5. EXAMPLES OF USE

In this section we present two examples of use of our system which proved to have satisfying results.

5.1. Installation

On the cube we display some predefined animations separated by segments of 3D game of life. As some of animations involves words we are able to show additional information to the audience. The instruments use as an input differences between measurements of the light intensity taken from different points in the neighbourhood of the cube, thus making the resulting sound independent form the variations of the room ambient lighting. We use mix of all the software for the instruments described in 4 except the optical theremins.

5.2. Improvisational game

The conductor is able to play animations from predefined set and at any time freeze the animation and use the XOR Snake instead. Other players use optical theremins and also allowed to change their location, use loop recording and playing abilities and hand gestures for the manipulation of sound. Additionally we introduced the following rule - the number and speed of actions of player should be related to the intensity of light on his side of the cube. For the best effect the game should take place in the dark room, which, in addition to make visual aspect of performance more striking for the audience, allows for additional special achieved by the usage of another light source - we sometimes end the performance by switching on the room lights, which causes raising of the pitch of the generated sounds by all the instruments.

6. EXTENSIONS

Our system is easily to extend. The simplest extensions is usage of alternative light sources in addition or instead of the cube. One of the experiments was to use few stationary optical theremins and give participants some flash lights with the regulation of light intensity. Another way to extend our system lies in programmability of all the devices - both the cube and the instruments can be used in various ways. Currently we are experimenting with some sort of feedback - the cube displays images based on the sounds which in turn are generated by the algorithm dependable on the state of the cube. Last but not least the human factor proves to be very important - most players, after the initial fascination with the novelty of the instruments and their sound, tend to explore the possibilities of the system in interesting ways (for the in-depth analysis of performance on sensor-based instruments see [14]).

7. CONCLUSION

The described system allows for musically satisfying performances with visually impressive effects. The main advantage of it is simplicity in the construction (it can be assembled with ease even by an inexperienced person with low financial cost), programming and usage. Our approach differs from the most common light-based instruments similar to the Beamz[3]) which uses lasers and sensors with only two states (on and off). Instead we created system more similar to the gestural or motions sensors interfaces, but much more affordable and easy to expand (due to flexibility in the programming and possibility of use of different light sources). Hopefully the presented examples of use and possibilities of extension show the potential which lies in the presented system.

8. REFERENCES

SABRe: AFFORDANCES, REALIZATIONS AND PERSPECTIVES

Sebastien Schiesser, Jan C. Schacher
Zurich University of the Arts
Institute for Computer Music and Sound Technology
Zürich, Switzerland
{sebastien.schiesser}{jan.schacher}@zhdk.ch

ABSTRACT
This paper focuses on the Sensor Augmented Bass clarinet Research (SABRe) in terms of musical practice, instrumentalist skills and compositional approaches. After a short overview of the concept of embodiment and instrument, the new possibilities of the SABRe instrument are exposed. Then, the first realized compositions present the strategies they deploy in view of the new control modalities and the exploration of affordances this augmented bass clarinet offers. Finally, some possible improvements and perspectives are outlined.

1. INTRODUCTION

The SABRe objective is to develop and evaluate a conventional bass clarinet augmented with various sensors. These sensors are divided into four different modalities, each providing different affordances1 and demanding varying degrees of instrumental awareness. Technical details of the SABRe project have been described in a previous publication [11].

After a first project phase concerned with the construction and implementation of the instrument itself, the second phase focuses on an in-depth generation of the instrument and a number of musical realizations. These occur through collaborations with different clarinetists, as well as in several composer residencies, with the goal of obtaining a clearer picture of the musical usefulness and the technical improvements necessary to produce a robust musical instrument.

The project is carried out at the Institute for Computer Music and Sound Technology (ICST), part of the Music Department of the Zurich University of the Arts (ZHdK) with a long tradition in education of music practitioners: instrumentalists, composers, teachers, conductors. Thus, it remains self-evident that any technological innovation of the institute has to be firmly positioned within the field of the musical praxis.

Funded by the Swiss National Science Foundation (SNF) for a duration of two years, the SABRe project is now in a transition phase towards a small series production.

1. Traditional Instrument

Traditional instruments are built around basic physical modes of sound generation and have been refined and optimized for specific desired qualities in the instrumental sound. The playing modes on the instruments were developed in relation to the physical possibilities that the instrument and the player’s body offer.

2. Digital Musical Instrument

Technological instruments dissociate the sound generating algorithms from the sound controlling gesture. The basic physical techniques of producing the sound have been replaced by signal processing algorithms and their metaphorical representation — usually through a real-world descriptive approximation. In today’s new digital music instrument (DMI) designs, the traditional fusion of control interface, mapping structure and sound generation is split up. Many technical innovations have focused either on one of these three aspects, but only few provide an overview on the whole discipline. Some background work on new concepts related to new media art, as well as an overview of the implication of engineering would help to better understand this quickly evolving field.

3. Augmented Instrument

The augmented instrument sits at the junction between the traditional and digital musical instrument. Even though acoustic instruments have long been modified physically and amplified electrically, a radical hybridization of instruments has only been made possible with the advent of digital signal processing and new sensor technologies. The remarkable quality that these mixtures present lies in the juxtaposition of the physical sound production modes and control interfaces with the abstract and symbolic representations inherent to digital signal processing algorithms.

In order that this additional layer of agency do not simply serve the dualistic “idea of enhanced human expressiv-

4. SABRe

After about 250 years of development and practice, the bass clarinet offers a set of affordances and constraints which requires from the player the acquisition of extremely differentiated body techniques. These body tech-

3.1. Traditional Instrument

Traditional instruments are built around basic physical modes of sound generation and have been refined and optimized for specific desired qualities in the instrumental sound. The playing modes on the instruments were developed in relation to the physical possibilities that the instrument and the player’s body offer.

3.2. Digital Musical Instrument

Technological instruments dissociate the sound generating algorithms from the sound controlling gesture. The basic physical techniques of producing the sound have been replaced by signal processing algorithms and their metaphorical representation — usually through a real-world descriptive approximation. In today’s new digital music instrument (DMI) designs, the traditional fusion of control interface, mapping structure and sound generation is split up. Many technical innovations have focused either on one of these three aspects, but only few provide an overview on the whole discipline. Some background work on new concepts related to new media art, as well as an overview of the implication of engineering would help to better understand this quickly evolving field.

3.3. Augmented Instrument

The augmented instrument sits at the junction between the traditional and digital musical instrument. Even though acoustic instruments have long been modified physically and amplified electrically, a radical hybridization of instruments has only been made possible with the advent of digital signal processing and new sensor technologies. The remarkable quality that these mixtures present lies in the juxtaposition of the physical sound production modes and control interfaces with the abstract and symbolic representations inherent to digital signal processing algorithms. Contrary to purely digital instruments, the augmented instruments usually maintain their traditional playing and sonic capabilities and superpose an additional layer of agency related to digital sound processes.

In order that this additional layer of agency do not simply serve the dualistic “idea of enhanced human expressiv-

4. SABRe

After about 250 years of development and practice, the bass clarinet offers a set of affordances and constraints which requires from the player the acquisition of extremely differentiated body techniques. These body tech-

3.1. Traditional Instrument

Traditional instruments are built around basic physical modes of sound generation and have been refined and optimized for specific desired qualities in the instrumental sound. The playing modes on the instruments were developed in relation to the physical possibilities that the instrument and the player’s body offer.

3.2. Digital Musical Instrument

Technological instruments dissociate the sound generating algorithms from the sound controlling gesture. The basic physical techniques of producing the sound have been replaced by signal processing algorithms and their metaphorical representation — usually through a real-world descriptive approximation. In today’s new digital music instrument (DMI) designs, the traditional fusion of control interface, mapping structure and sound generation is split up. Many technical innovations have focused either on one of these three aspects, but only few provide an overview on the whole discipline. Some background work on new concepts related to new media art, as well as an overview of the implication of engineering would help to better understand this quickly evolving field.

3.3. Augmented Instrument

The augmented instrument sits at the junction between the traditional and digital musical instrument. Even though acoustic instruments have long been modified physically and amplified electrically, a radical hybridization of instruments has only been made possible with the advent of digital signal processing and new sensor technologies. The remarkable quality that these mixtures present lies in the juxtaposition of the physical sound production modes and control interfaces with the abstract and symbolic representations inherent to digital signal processing algorithms. Contrary to purely digital instruments, the augmented instruments usually maintain their traditional playing and sonic capabilities and superpose an additional layer of agency related to digital sound processes.

In order that this additional layer of agency do not simply serve the dualistic “idea of enhanced human expressiv-

4.1. Affordances

In order to take advantage of the improvements of the traditional instrument which have been optimized over the years, SABRe makes a use of all the instrument’s properties: neither the control interface nor the sound generating
part have been altered. Furthermore, it is an assumed ob-
jective not to move too far from the existing playing tech-
niques, so that they may serve as a foundation on which a
new practice may be developed. In that manner, the entry
threshold remains affordable for trained clarinetists and
their years of experience can be leveraged for the acquisi-
tion of a new body knowledge.

4.1. Key Sensors

The key sensors (Figure 1(a)) provide real-time infor-
amation about continuous key positions, about the press-
ing and releasing speed as well as additional meta-
information like clarinet fingering or MIDI note values.
The most direct use of these sensors is probably fin-
gering retrieval, whereas the value of single-press informa-
tion (press/hold) of each key can be captured in real time.
This mode requires the lowest level of awareness from the
player, since it makes use of fingering information which is already
provided by conventional playing and where complex se-
quences can be written on a score and executed with a very
high accuracy. This allows e.g. to control a DMI without
having to learn new fingerings and to retrieve pitch infor-
mation without microphone and pitch recognition system.
Key sensors also provide continuous position values, as
well as pressing and releasing speed calculation, which
offers control values in the time domain. It has to be
pointed out, that these measurements take place in key
displacement ranges of less than 1 millimeters and re-
quire yet undertrained skills: players mostly focus on the
binary key positions. Even though this de facto limits the
accuracy and reproducibility of the output values, it does
offer data of a different quality, directly related to the tra-
ditional instrument ‘selection gestures’ [2].

4.1.2. Air Pressure

Control of mouth air pressure (Figure 1(c)) is a well
trained skill of wind instrument players, primarily used
for regulated sound intensity, but also for subtle timbre
changes or sound control in different instrument registers.
Since SABRe allows ‘soundless playing’ — by completely
closing the reed at the mouthpiece aperture while provid-
ing mouth pressure — it decouples the air pressure param-
eter from the conventional direct acoustic feedback. This
opens up the possibilities of using pressure as an accu-
rate controllably expressible value.

Since the sound production is not impeded by the air-
pressure sensor, digital sound processing can still be used
in complement to pressure measurement, as well as for
pitch recognition and does not interfere with the SABRe
sensors in any way.

4.1.3. Thumb Switches

The thumb switches (Figure 1(b)) afford similar gestures
as the clarinet keys, but are intended to be used in a
different manner and offer other feedback qualities. First,
while keys are often pressed over a relatively long time
(from one to several note periods), the thumb switches are
used as toggles to trigger events and are quickly pressed
and released. Second, the switches differ from the clarinet
keys in shape (about 5 mm diameter), displacement less
than 1 mm) and haptic feedback (‘click’ sensation).

Notwithstanding, the left hand thumb is less in de-
mand on the clarinet than other fingers and the switches
remain (at least partially) reachable while having the
thumb for octave key pressed. With new coupling and na-
bituation processes, this feature should afford a high level
of virtuosity.

4.1.4. Inertial Sensor

In traditional music performances, musical gestures
mainly serve direct sound production or transformation.
Other movements — ‘accompanist’ or ‘ancillary gestures’
[2] — are often not conscious or serve different goals such as
expressivity support.

Since the 1970s, a number of music works have been
deliberately staged by the pioneers of experimental mu-
tical theater like a.o. Kagel [5], Stockhausen [12] or
Aperghis [1], assigning a new role to the former ‘accom-
panist gestures’. These gestures however work on a visual
channel and are not meant to affect — in the worst case
they act as an impediment to — sound production. The fact
of capturing movements with the inertial sensor connects
both the visual and acoustic channels by delivering infor-
ation about position, orientation and acceleration of
the instrument. This can be used as control parameters
and at the same serve staging necessities.

This new paradigm of transforming formerly ‘accom-
panist into ‘instrumental gestures’ could easily become
the most challenging aspect of the SABRe affordances.
Since many musicians are neither trained nor particularly
comfortable with movement, this presents a field to be ex-
plored systematically in order to obtain experience, iden-
tify exploitable patterns and build a strong proficiency.

5. REALIZATIONS

The first phase of evaluation for the SABRe project is still
ongoing and composers willing to experiment have started
working with this new interface. The very first piece writ-
ten for SABRe [10] makes use of three different features of
the instrument: finger patterns, instrument spatial
attitude and rotational acceleration.

Finger patterns serve synchronization purposes and
trigger sound events through the use of key combinations,
which do not correspond to any notes that would normally
be played and which cannot occur accidentally.

The inertial sensor is utilized in a sound in a sur-
round environment by ‘drawing’ i. e. pointing towards the
desired displacement on the floor with the bottom of the
clarinet. For this purpose the instrument’s tilt and head-
ing angles are derived from the gravity vector values of
the accelerometer. In addition quick rotations of the
instrument around its vertical axis give rise to unambigu-
ous values of the gyroscope, which are also used to trigger
events, but this time in a more visually obvious manner.

For a different composition, which is currently being
written, an alternate approach using MIDI notes extracted
from the fingering patterns is chosen. These notes play
a virtual instrument in parallel to the acoustic one and
propose a complex counterpart based on real-time sound
transformations.

Finally, a proposition is made with the capture of syn-
chronized audio and sensor data sequences. These se-
quences can be used on the one hand to analyze a player’s
performance in qualitative and quantitative terms (e.g. in
order to analyze timing and tempo) or can be used for
playback at a later time in a different setting such as the
composer’s studio.

6. PERSPECTIVES

SABRe extends the bass clarinet at its control interface,
thus providing a tactile/kinesthetic feedback through the
conventional instrument physical properties4. However,
the augmented part of the instrument does not provide the
same vibrational coupling through ‘resonance’ as does the
base instrument. A feedback channel might be added in a
future implementation through tactile actuators. This
warrants further investigations which exceed the current
scope of the project.

The four data capture modalities opened up by SABRe are by
far not completely covered by the application examples and
first musical realizations mentioned in the sections 4.1 and
5. The intention of the research project is to spread the
word in the interested communities, make several copies
of the instrument available to clarinetists and composers
and to collect composition and performance experience.
Thus, when a certain playing proficiency has emerged, it will
be feasible to evaluate the new instrument in the light of
its embodied potential.

7. CONCLUSION

By building upon a mature musical instrument and its op-
timized control interface, the SABRe project exploits a
solid framework in order to extend the instrument control
possibilities. The four data capture modalities provided
offer new affordances to the interface, while preserving
those offered by the conventional instrument, thus allow-
ing the player to take advantage of already acquired in-
nstrumental skills.

The collaboration with an interested community of
musicians will serve as much for evaluation purposes as
for the generation of new musical ideas. These will further
open up the possibilities the instrument has to offer. Once
this knowledge has successfully been acquired by several
clarinetists, it will be more easily transmitted. With this
ultimate step the SABRe project will have succeeded in
extending a traditional music instrument. The long-term

4With the exception of the thumb switches, which have their own
tactile/kinesthetic properties.

goal is to eventually establish a benchmark for augmented
bass clarinets (and other woodwinds) and to bring forth a
repertoire of new pieces of music for this instrument.

8. REFERENCES

Control of Music. L. M. Wanderley and M. Batter,
94.
[3] S. Gallagher, How the body shapes the mind. Oxf-
ford University Press, 2006.
ning, Acting, and Knowing. Shaw, Robert and Brans-
ford, John, 1977.
sal Editions, 1969/70.
und Medialitat des Musikinstrumentes unter besonderer Betrachtung digitaler interaktiver
Musikperformanzen,” in Klang (ohne) K¨orper,
Spuren und Potenziale des K¨orpers in der elektron-
ischen Musik. Harenberg und D. Weisberg, Eds.
algorithmic sound generation,” Contemporary Mu-
sic Review, vol. 25, no. 1/2, pp. 139 – 149, Febru-
ary/April 2006.
[8] ——, “Embodiment and agency: Towards an aes-
thetics of interactive performativity,” in Proceedings
of the 4th Sound and Music Computing (SMC) Con-
Psychology of Music, D. Deutsch, Ed. San Diego:
mented Bass Clarinet,” in Proceedings of the 12th
International Conference on New Interfaces for Mu-

tical Expression International Conference on New
Interfaces for Musical Expression. University of
Michigan, May 2012.
[12] K. Stockhausen, Der kleine Harlekin, for clarinet.
K¨urten: Stockhausen Verlag, 1975.
VARIANCE IN REPETITIVE GAMES MUSIC

Axel Berndt
Institute of Software and Multimedia-Technology
Faculty of Computer Science, Technische Universität Dresden
Dresden, Germany
Axel.Berndt@tu-dresden.de

ABSTRACT

The musical scoring of video and computer games is faced with the unpredictability of interaction. Music has to follow the process of the interactive scene but this leads to two basic problematic situations: (i) the player is too fast and the music has to react before the current piece is finished, and (ii) the player is too slow and music has to bridge a longer period than the current piece does. While earlier papers were mainly treating the first problem this paper focusses the second.

As the length of an interactive scene is usually impossible to predict it is likewise impossible to say how much musical material is needed. The common way to bypass this problem is to loop the music for as long as the scene lasts. This approach involves an existential danger: Sooner or later the player becomes aware of the repetition; the game scenario emerges as a mere mechanical arrangement and loses much of its integrity. Variance is needed that renews the music each time it repeats. This paper presents several views to this problem and introduces a variety of possible approaches.

1. INTRODUCTION

An important basic principle of audio-visual media scoring is that musical change indicates (and therefore necessities) a corresponding change in the scene, the narration, or the dramaturgy, even if not visible [12]. In interactive media such changes are highly dependent on user interaction. It is generally impossible to predict the amount of time a player will spend in a scene or how long a certain situation lasts until the player makes that triggering interaction. Thus, it is equally impossible to plan the length of the corresponding music in advance. The endless loop is the most common means today to musically stay at a situation for an uncertain period.

But the exact repetition of a complete musical piece is a very specific means too that can become very conspicuous again. More variance and a less binding memory of recombinations with big diversity become available [1]. But still, after a while, when all the precomposed musical material in enough permutations and combinations was introduced to the listener, its recurrence can become conspicuous again. More variance and a less binding memorable structure can be approached by very short musical snippets that play single figures or even just single sound events (tones and chords) that fade in and out and may overlap with others. This cue collage may sound very diffuse but it features the possibility to be combined with (or triggered by) interactive events, creating a very reactive musical score.

2. THE COMPOSER’S APPROACH

Memorable features of the compositional structure are the primary hints that listeners recognize and remember. Derived from this insight, games music composers try to conceal repetition by a more diffuse structural layout that implies the recognition of specific features [14].

A leading melodic part is one of the strongest catchers of attention. It is very often built of motivic figures that recur and vary over time. Due to its inherent formal principles, the melody mediates a strong feeling of structure and form [8]. A common way to conceal this is to abdicate motivic work and to split the melody into multiple preferably overlapping figures, diversified into a polyphonic formation. This technique is also well known from composers of the romantic era.

Furthermore, clear structural borders are easily memorable features. Fluent structural borders can be achieved by polyophonically overlapping structural layers and seamless connections of consecutive form elements.

Most effective is a cleverly timed disposition. Preliminary user studies can determine the average playing behavior. This gives clues to dispose the appropriate length of the accompanying pieces of music. Structural diffusion is a cheap and easy way to conceal musical repetition. But it works only for the average player. Repetition itself is not eliminated and recognition is just delayed, in the best case for long enough.

3. THE ARRANGER’S APPROACH

Variety can be achieved by varying the sequence of musical pieces or segments. A very coarse musicbox like approach can be found in the The Elder Scrolls: Morrowind and Oblivion [1]. For each state of the gameplay there is a set of musical compositions with alike characters of expression from which one is randomly chosen. When the piece is over, another one is selected.

A more fine-grained approach is to arrange on the level of inner-musical phrase and section structure as proposed within several research prototypes (e.g. [6, 22]). They introduce the so-called loop segments One Shots and Loop segments. So called One Shot segments are played back only during the first iteration of the musical loop and skipped later on. The Loop segments remain. Thus, the first repetition appears to be a rearrangement instead of a repetition. Furthermore, by providing multiple alternatives for each segment (just as in musical dice games [17]) a huge number of recombinations with big diversity become available [1].

But still, after a while, when all the precomposed musical material in enough permutations and combinations was introduced to the listener, its recurrence can become conspicuous again. More variance and a less binding memorizable structure can be approached by very short musical snippets that play single figures or even just single sound events (tones and chords) that fade in and out and may overlap with others. This cue collage may sound very diffuse but it features the possibility to be combined with (or triggered by) interactive events, creating a very reactive musical score.

4. THE ORCHESTRATOR’S APPROACH

Orchestration deals with the different timbres of instrumental material. Orchestration deals with the different timbres of instrumental material.

5. THE GENERATIVE APPROACH

Variation and improvisation are probably the oldest concepts of music. We can find printed evidence for sophisticated variation techniques already in medieval music. Later, variation became a kind of a compositional aspect. Performers had to learn how to vary a musical material correctly. The baroque was the era where variation and improvisation became a high art and necessary ability for the performers [19]. Up to the extensive variation works of the classic era the development of a multitude of variation techniques can be postulated. Today’s musical morphosis distinguishes variations by two aspects [3]:

Subject of Variation: The direct variation is applied to the theme/motif, whereas the indirect variation retains the theme/motif unchanged and varies its accompaniment.

Type of Variation: The strict variation saves the harmonic and architectural characteristic of the theme/motif. Its shape and gestalt quality remain unchanged. On the other hand, the free variation changes not just melodic and rhythmic aspects, but also harmonic and formal ones. Each one of such variation can afford new gestalt and quality.

The variation, adaptation, and improvisation over a given musical material is also a classical subject in computer music research. We have chosen representative prominent approaches from the last decade for discussion.

5.1. Embellishment

A strict and direct type of variation is the melody embellishment. A given plain melody is enriched by various ornaments. In this respect the systems MeloNet and JazzNet are very interesting. They utilize a neural network to learn melody ornamentations, i.e., ornamentation figures/patterns, including the melodic and harmonic context where they were applied [11]. This is demonstrated with melody variations in the style of J. Pachelbel and Jazz improvisations imitating Charlie Parker. The learning set directly influences the stylistic imprint of the network.

A generative music approach that utilizes genetic algorithms is described by Garland-Jones. MusicBlox [10]. It combines several (predefined) input patterns to create variants. The fitness function can measure the relational distance to the input patterns. Thus, it is possible to apply mutation and recombination operations and vary the result within the domain spanned by its input patterns. This is meant to be used as a creative tool or toy for music composition. But it can also be used to combine, for instance, a plain melody and several embellished versions to create new embellished versions.

5.2. Improvisation

The improvisation can be seen as a free variation. It can change all aspects of the original, its structure, melodic, rhythmic, and harmonic properties.
A genetic algorithm based approach is John Al Biles’ GenJam system [16]. The musical input of n co-performing human musicians is varied by mutation and crossover operations to generate an improvisational response. It is melody based: the chord progression, the tempo, the rhythmic pace, and overall arrangement are predefined. GenJam’s stylistic repertoire ranges from Jazz over Latin to New Age. The musical quality and variety of its improvisations strongly depends on the quality and variety of the human performer’s input. In the games scenario, where we have static precomposed music to be varied, this may lead to over-fitting problems over time.

A very popular means in computer music, Markov models, was used by François Pachet in his system Continuator [18]. It builds a Markov model based structure from real-time musical input of a human co-performer. New patterns are generated not just for the melody but also for its accompaniment. Since it directly analyzes the real-time input the system is stylistically independent to a certain amount. The system can run in stand-alone mode like a music generator, as a collaborative improviser and composer that creates continuations to the musician’s input.

5.3. Reharmonization

The harmonization of a melody determines a sequence of chords and creates a polyphonic counterpoint. Reharmonization changes one or more of these chords and adapts the music to seamlessly transition and combine different performances and orchestrations.

Yoon and Lee describe a planning approach for affective reharmonizations [24]. A system that implements reharmonization is described by Livingstone [13]. It implements the relatively unproblematic minor-major changes. Changes to completely different chords are implemented in the system by Stenzel [21]. However, it also proves that great changes do harm to the musical coherence and conclusiveness quite quickly. Naive changes in voice leading may be legato in one performance and portato in another, this creates either a tight or a brittle sound.

Most subtle differences can be achieved by random variations of note onset times and velocities, aspects that we know as random variation in human performance. Furthermore, the synchrony of the parts can be changed. A leading part may be ahead to mediate an active progressive mood or it can be behind creating a laid-back kind of feeling. However, these means may already be too subtle and not sufficient to give enough variety to the music. Hence, they should be used in combination with the other features discussed so far.

Approaches to create such expressive performances are knowledge-based [9], machine learning-based [23], and derived from a mathematical music theory [15]. The performance engine described in [5] implements a technique to seamlessly transition and combine different performances and orchestrations.

7. CONCLUSION

We have introduced a variety of approaches to tackle the problem of repetitive music in games. The compositional and arrangement approaches can already be found in today’s video and computer games. But they are restricted to the variety of its limited precomposed material. The same applies to the polyphonic and orchestral approaches which are based on static precomposed material as well. This restriction can be overcome by introducing generative aspects to the music processing. Embellishment and reharmonization edit the precomposed music on the compositional level. Improvisation contributes new material. The expressive performance renders the music in different shades. These approaches are not mutually exclusive but can be combined and thereby open up a much wider range of possibilities.

8. REFERENCES

[17] W. A. Mozart, “Musikalisches Würfelspiel: Anleitung so viel Walzer oder Schleifer mit zwei Würfeln zu composerohemunsmischal zu seyn noch von der Composition etwas zu verstehen,” Köchel Catalog of Mozart’s Work KV1 Appendix B64d or KV1616e, 1787.
MUSIC AS A MEDIATOR OF EMOTION

Adinda van ’t Klooster
Manchester Metropolitan University
School of Computing, Mathematics
and Digital Technology

ABSTRACT

This paper describes the development of the Emotion Light, an interactive biofeedback artwork where the user listens to electronic music that is composed to calm the user down. The viewer can see a visual reflection of their bodily response by holding a sculptural light that tracks the user’s galvanic skin response and heart rate and translates these into changing light patterns that emerge from the sculpture. After an introduction into the principle of biofeedback, the hardware and software developed for this project are described and the work is briefly evaluated through a review of informal interviews held with users of the Emotion Light.

1. INTRODUCTION

Music is often composed not only to express certain emotions but also to generate these in the listener [2, 6]. Music is often composed not only to express certain electrical biosignals are the Electroencephalogram (EEG), the Magnetoencephalogram (MEG), Galvanic Skin Response (GSR), the Electrocardiogram (ECG/EKG), the Electromyogram (EMG) and Heart Rate Variability (HRV). The field of research that tries to create Human Computer Interfaces and technological systems, one related to arousal and the other to valence [17]. The term valence in this context relates to being attracted (positive valence) or repulsed (negative valence) by a stimulus, which in physiological systems, one related to arousal and the other to valence [17].

The colour of the light visualises the users physiological response to the music. To achieve this, changes in physiological data like GSR (galvanic skin response), heart rate and movement are tracked and translated via code into changes in light patterns. A high arousal level is mapped to red and a low arousal level is mapped to blue, with a continuous scale through green, yellow and orange in between. The heart rate is directly reflected in the speed of the pulse of the light. In the Emotion Light the inducive input for the user is the pre-composed electronic music. The negative feedback is the colour of the light as the user can quickly assess how far away she/is still from the blue colour.

Emotion research is usually carried out in laboratory settings and results from there cannot necessarily be reproduced in a less controllable setting such as a gallery or conference hall. However, varying arousal levels can quite accurately be obtained from biosignals even in non-laboratory settings. It should be noted though, that this is not the same as ‘reading ones emotions’.

Originally it was thought that the autonomic nervous system was fully automatic and beyond human control. The arrival of biofeedback technology in the sixties changed this. One can speak of a biofeedback system when it is designed not only to reflect back one’s physiological signals but also to help the user reach a specific target state. Biofeedback was experimented with as a treatment for various medical conditions like hypertension [16, 4] and asthma [20, 9]. Although the medical claims in terms of what can be achieved through biofeedback have become more humble than they were in the seventies, the technology has shown that it can help humans to learn to control physiological processes like heart rate, body temperature and blood pressure and it is still used in therapy [17].

The musician and pioneer David Rosenboom made it his life’s work to investigate the aesthetic potential of biofeedback. He suggested that the biofeedback interface has to provide both an inducive input and a negative feedback [14]. The latter allows the user to see how far off the goal she/is and the inducive input helps the user to reach the desired state. Rosenboom made many works that used biofeedback and some also used light or projection [13]. Contemporary artists who have combined biofeedback with projection are George Khut [12], Mariko Mori, Brigitta Zics and Char Davis. Apart from Mignonneau and Sommerer in their work Mobile Feelings [18] not many artists have combined sculpture with biofeedback.

2. THE EMOTION LIGHT

The wireless Emotion Light artwork combines music, sculpture and light with the technology of biofeedback. It aimed to provide an aesthetic experience where artwork and technology merge on all levels. The technology is hidden from view to allow the user to focus on the experience of listening to the music and seeing how their body responds to this. This way, it aims to reveal the connectedness of mind and body and how these are influenced by one’s immediate environment.

This project also explored how biosignals can be best used in an arts/exhibition context, which is much less controllable than the lab environments where emotion research is normally carried out and has aesthetic criteria that inform how the technology ought to be used. [8] Conceptually this artwork is also about the body. The shape resembles a stylised arousal and movement and is mapped to red and a low arousal level is mapped to blue, with a continuous scale through green, yellow and orange in between. The heart rate is directly reflected in the speed of the pulse of the light. In the Emotion Light the inducive input for the user is the pre-composed electronic music. The negative feedback is the colour of the light as the user can quickly assess how far away she/is still from the blue colour.

In order for machines to detect emotion, they need to be able to detect emotional cues. Humans get these from looking at people’s faces and body language and from listening to the intonation of other people’s speech. The computer can use these cues as well but can also record and analyse biosignals. Biosignals are indicative of emotional variation [1,6,2,3] but there is no one-to-one relationship between particular emotions and values per biosignal and analysing the signals in an intelligent way is not trivial. There are large interpersonal differences per biosignal and even one person’s data can vary from day to day (van den Broek et al., 2009). The best-known electrical biosignals are the Electroencephalogram (EEG), the Magnetoencephalogram (MEG), Galvanic Skin Response (GSR), the Electrocardiogram (ECG/EKG), the Electromyogram (EMG) and Heart Rate Variability (HRV).

The field of research that tries to create Human Computer Interfaces and technological systems, one related to arousal and the other to valence [17]. The term valence in this context relates to being attracted (positive valence) or repulsed (negative valence) by a stimulus, which in simplified terms means positive or negative emotion.

The colour of the light visualises the users physiological response to the music. To achieve this, changes in physiological data like GSR (galvanic skin response), heart rate and movement are tracked and translated via code into changes in light patterns. A high arousal level is mapped to red and a low arousal level is mapped to blue, with a continuous scale through green, yellow and orange in between.

The heart rate is directly reflected in the speed of the pulse of the light. In the Emotion Light the inducive input for the user is the pre-composed electronic music. The negative feedback is the colour of the light as the user can quickly assess how far away she/is still from the blue colour.

The sensors were embedded into the sculpture and sent their data directly to an Arduino BlueTooth microcontroller. From there the data was sent wirelessly to a Max/MSP patch on a nearby laptop. Interpreting and mapping of the data took place in Max/MSP from where the final colour output was sent back to the microcontroller inside the sculpture to change the colour of the LEDs in real time. The electronics were fixed on a bespoke PCB that fitted directly onto the microcontroller.

2.2 The mapping in the Emotion Light

How to transfer physiological data into aesthetic interfaces remains debatable. In earlier version of the Emotion Light (version 2b) I tried to create a classifier that could detect eight different emotions [7]. This turned out to be too ambitious for this hardware setup combined with the uncontrollable gallery conditions. For the current version of the Emotion Light (version 3), the mapping was based on the circumplex model of affect that suggests that all emotions derive from two neurophysiological systems, one related to arousal and the other to valence [17]. The term valence in this context relates to being attracted (positive valence) or repulsed (negative valence) by a stimulus, which in simplified terms means positive or negative emotion.

The arrival of biofeedback technology in the sixties changed this. One can speak of a biofeedback system when it is designed not only to reflect back one’s physiological signals but also to help the user reach a specific target state. Biofeedback was experimented with as a treatment for various medical conditions like hypertension [16, 4] and asthma [20, 9]. Although the medical claims in terms of what can be achieved through biofeedback have become more humble than they were in the seventies, the technology has shown that it can help humans to learn to control physiological processes like heart rate, body temperature and blood pressure and it is still used in therapy [17].
Arousal has to do with intensity and can be read directly from the bio signals. Valence is harder to read from the bio signals, as it is a higher-level information: i.e. it has to do more with pro-interaction. The benefit of using this model to design the system is that it allows for a sliding scale of emotional/physiological variation, which makes it point.

Arousal is obtained from the GSR sensor. The GSR component is divided into a phasic and tonic component [10]. The phasic changes quickly, and shows peaks in the area which happen every couple of seconds; these are called phasic events. The tonic component changes slowly over a longer period of time. One could therefore say that the tonic component reflects the mood or general arousal level and the phasic reflects more immediate responses to the external environment: temporary increases in the sympathetic nervous response. The tonic component was mapped to the colour of the light, with a low tonic value (so more sweat, and thus more arousal) tends to go red very quickly and then it calmed down whereas ladies it seemed to hover around the green and the blue a lot longer. With ladies it changed more often but with men, when they settled down after the initial red, theirs kind of stayed a constant colour, whichever colour it was going to, regardless of whether they were listening to the music or not. The music definitely affected the ladies more. Because as women were listening to it, it did have a lot more of a calming effect, seemed, than on the gentlemen. Why that is, I don’t know.

It would be interesting to do further tests to investigate whether the perceived gender difference can be sustained with evidence. During ISEA09 most visitors where listening to the music whilst holding the light, so it is impossible to say whether the initial ‘red surges’ observed in men were in response to the meaning of the shape, their response to the music or their initial state of arousal. One obvious function of the headphones themselves is that they block out external noise. A female visitor said:

“When I was playing with the Emotion Light I had headphones on as well and I was listening to the soundtrack that was made with it. And I think taking away the environmental sound and allowing me to really focus on the music, or whatever I was doing through those headphones, made me really focus on the experience I was getting from the work. So I think it was adding a lot. It made it a more immersive experience because my eyes were watching the light and holding the light and something and my ears were blocked from the general chitchat. I think the sound really does help the experience. When I took the headphones off, my anxiety levels were higher because I was watching what was going on around me and I could hear stuff so I think yeah, it probably plays a role in calming you down which is all part of the experience.”

For some visitors however, the headphones with the music were not enough because of the external environment and they felt their emotions were on display for others to view, which added to their anxiety. A male visitor said:

“I kind of felt, working with the Emotion Light, that I was quite anxious that it didn’t go red because I was so anxious and nobody wants to be portrayed as someone who’s in that state of mind. But it went red a lot. Ha-ha-ha!”

People did seem to get better at it when they came back a second time, which was also observed by the invigilator: “I think from the people that revisited it, because they came back an second time, and it was a lot easier for them. They got used to doing it.”

3. CONCLUSION

As an artwork, the Emotion Light worked well. There were a few different layers to the work, which gave an ambiguity that kept it interesting for people and made them revisit the piece. When they came back a second time they were not just reaching the ‘blue zone’. This is likely to be due to them getting better at controlling their own bio signals, due to the visual feedback provided by the system.

How much the music helps the user to calm down is highly individual and more scientific research would need to be done to determine the exact role of the music. The Emotion Light itself is the shape of a uterus so for testing the influence of the music it would be necessary to use a neutral shape.

In technical terms, this system could detect arousal but not valence. Future research could focus on making the system able to detect valence. For this, more biosensors would need to be added to the system [1]. A temperature sensor could be added and a more intensive analysis of heart rate variance could also help to obtain more information on valence [19]. Further research is necessary to determine whether valence could then be accurately obtained.

4. ACKNOWLEDGMENTS

This project was developed during my PhD at Sunderland University completed in 2011. Advisor on biosensors was given by Ben Knapp and the programming was done by Vincent Akkermans, Robin Price and Ken Brown. Hardware development was by Marc Bow and Ken Brown.

Dave Knapton and Neil Milburn from AMAP helped to turn the sculptural porcelain model into a rapid prototyped ABS plastic version.

Funding for this project and exhibition of it came from the Arts Council England, the AHRC, the Mondian Fund, and ISEA09, the University of Sunderland, CRUMB, the European Regional Development Fund, One North East the Futures Fund and Manchester Metropolitan University.

5. REFERENCES


AIRduino: An Inexpensive DIY MIDI Wind Controller

Timothy Anderson  
University of Montana  
32 Campus Drive  
Missoula, MT 59812  
TimAnderson.UM@Gmail.com

ABSTRACT
The AIRduino is a hardware MIDI controller featuring a bi-directional breath pressure sensor that has the ability to control different parameters by inhaling and exhaling through the sensor. The controller also features four videogame-operator thumbsticks for extra versatility, along with eight trimpots for extra versatility. Through these two unique types of sensors, the AIRduino allows the performer a versatile and unique method of controlling his or her music. This MIDI controller is easy to assemble from inexpensive parts, and is open source in both code and circuitry schematics.

Keywords
Arduino, MIDI, Electronic Instrument, DIY, Breath Pressure Sensor, Thumbsticks

1. INTRODUCTION
As both a musician and an avid computer programmer, I often see other musicians express confusion about their electronic instruments. While they can easily accomplish tasks with their instruments, they seldom understand how their hardware works. My goal through this project is to use the Arduino microcontroller, some simple, cheap, and easy to use sensors, and one unconventional breath pressure sensor to enable musicians to easily build their own MIDI controllers, and learn about the controller's structure in the process.

2. AIRDUINO: ABOUT THE CONTROLLER

2.1 Arduino at the Heart of the System
As the name AIRduino implies, the MIDI controller is built around the Arduino Uno Microcontroller board. Because of the relatively small price of the board, ease of hardware implementation, and wealth of online resources, the Arduino infrastructure was an obvious choice. At a price of thirty USD at time of writing, the board is an affordable microcontroller.

Additionally, Arduino is easy to use in development, as no soldering is required. Instead, the Arduino is equipped with pin slots that wires can be inserted into. Because knowledge of complex circuitry isn't a prerequisite to working with the Arduino, it becomes much more accessible to musicians interested in building a MIDI controller. The tutorials and examples on Arduino.cc work to further bridge the gap of technical understanding required to construct a MIDI instrument. By having access to a plethora of tutorials on the hardware and software involved, even those with little circuitry experience can easily find answers to any problem they might have while constructing their own AIRduino.

2.2 Bi-Directional Breath Pressure Sensor
When I started working on this project, I was determined to create an interface that was simple and utilitarian, yet had some unique function that couldn’t be found elsewhere in MIDI controllers. However, since my main instrument is Saxophone, the first few ideas for this project looked similar to the Electronic Wind Instrument (EWI) by AKAFL. In order to distinguish the AIRduino from this well-established counterpart, the AIRduino was equipped with a specialized breath pressure sensor that is able to read both positive and negative pressures. What this means to performers is that the AIRduino MIDI controller be an expressive, versatile instrument. However, this couldn't counteract the need to keep the individual components as inexpensive as possible. After the eight trimpots were implemented, the instrument still seemed lacking. It was then that the thumbsticks were added. Since they have the unique feature of resetting to their centered position when not being manipulated, they added a unique performance option to the MIDI controller. As an additional layer of control, a switch on the back of the AIRduino allows the performer to switch between two preset controls. When the switch is up, the thumbstick reads its X and Y position and sends out the two states as individual MIDI Continuous Controllers. However, if the switch is flipped, the data are broken into four individual controllers, corresponding to up, down, left, and right. In this way, the thumbsticks can provide more nuanced control over a MIDI instrument, and reset all the controls to zero when not being manipulated.

2.3 Why Thumbsticks?
As the creator, I was concerned that the AIRduino MIDI controller be an expressive, versatile instrument. However, this couldn't counteract the need to keep the individual components as inexpensive as possible. After the eight trimpots were implemented, the instrument still seemed lacking. It was then that the thumbsticks were added. Since they have the unique feature of resetting to their centered position when not being manipulated, they added a unique performance option to the MIDI controller. As an additional layer of control, a switch on the back of the AIRduino allows the performer to switch between two preset controls. When the switch is up, the thumbstick reads its X and Y position and sends out the two states as individual MIDI Continuous Controllers. However, if the switch is flipped, the data are broken into four individual controllers, corresponding to up, down, left, and right. In this way, the thumbsticks can provide more nuanced control over a MIDI instrument, and reset all the controls to zero when not being manipulated.

2.4 The Advantages of the MIDI Port
The AIRduino connects to the computer through a standard MIDI connector. While sending MIDI information out the USB port on the Arduino board seems to be popular with other projects, it has inherent downsides. In many cases, special drivers have to be written to read the send data. In other situations, multiple pieces of software are needed to convert to usable MIDI data. However, by implementing a MIDI port, the AIRduino requires no drivers or software. Instead, the AIRduino may be connected through an audio interface's MIDI port, or into a USB port from a MIDI to USB converter cable. This change allows the AIRduino to be plug-and-play compatible with any Windows or Mac system.

3. SUMMARY
At around ninety USD for all parts and materials, the AIRduino MIDI Controller is an affordable tool for any performer wanting distinct, versatile instrument. By using the Arduino infrastructure and simple, inexpensive parts, it is my honest wish as its creator that the AIRduino allow performers easy access to new methods of expression.

4. ACKNOWLEDGEMENTS
I would like to acknowledge my professor, Charles Nichols, for supporting me on this project. I would also like to thank the people at Arduino Software for their excellent microcontroller. All photography for this article was taken by Louis Habeck. I would also like to thank Nathaniel Shifmore at Shifmore.Blogspot.Com for the idea to use the MIDI to USB cable. And finally, I would like to thank my father, for all the support he has given me, and for the idea of the bi-directional pressure gauge.

8. REFERENCES
[6] USB to MIDI  connector. While sending MIDI information out the USB port on the Arduino board seems to be popular with other projects, it has inherent downsides. In many cases, special drivers have to be written to read the send data. In other situations, multiple pieces of software are needed to convert to usable MIDI data. However, by implementing a MIDI port, the AIRduino requires no drivers or software. Instead, the AIRduino may be connected through an audio interface's MIDI port, or into a USB port from a MIDI to USB converter cable. This change allows the AIRduino to be plug-and-play compatible with any Windows or Mac system.

3. SUMMARY
At around ninety USD for all parts and materials, the AIRduino MIDI Controller is an affordable tool for any performer wanting distinct, versatile instrument. By using the Arduino infrastructure and simple, inexpensive parts, it is my honest wish as its creator that the AIRduino allow performers easy access to new methods of expression.

4. ACKNOWLEDGEMENTS
I would like to acknowledge my professor, Charles Nichols, for supporting me on this project. I would also like to thank the people at Arduino Software for their excellent microcontroller. All photography for this article was taken by Louis Habeck. I would also like to thank Nathaniel Shifmore at Shifmore.Blogspot.Com for the idea to use the MIDI to USB cable. And finally, I would like to thank my father, for all the support he has given me, and for the idea of the bi-directional pressure gauge.

8. REFERENCES
MÓRIMO. TACTILE SOUND AESTHETIC EXPERIENCE

Justyna Zabrycka
Paweł Cyrta

Inter-University Multimedia Program
The Fryderyk Chopin University of Music

00-368 Warsaw
Poland

ABSTRACT
The project, called Mórimo, aims to provide a platform for musical expression, with an emphasis on tactile properties of sound. It regards music in a very sensuous way – as a tactile composition perceived physically trough the body. In addition to being heard, the Human body may experience sounds also as haptic sensations, from subtle vibrations to shaking. This phenomena is produced by natural as well as artificial sources; and so it may also be experienced when listening to music. The idea of a tactile sound aesthetic experience brings an artistic view to this phenomena.

Among existing familiar projects, this work investigates the human body-music interaction, within the performative context in particular. This experience is shared between listener and performer. Additionally, both roles (listener and performer) can easily be taken as deaf or hearing-impaired individuals, as to reveal a broader and deeper sense of listening.

1. INTRODUCTION
Tactile sensations, especially those from the sound waves, very rarely play a role in artistic expression. There are only few known examples of touch based art works. Performers don't tend to explore this modality from many possible reasons. Provided audible and visual messages can be received by a number of people at the same time from a distance. It seems there is much more limitations with engaging the sense of touch, which is accociated with direct skin contact. We don't expect to feel the sound, we don't think about it as a tactile energy, we don't even use such expression in our language.

The human perception is, however, an active process and it is able to being shaped by experience and expectations. Sense of touch can be developed with the conscious awareness of sound frequency perception. Evelyn Glennie (b. 1965) is a Scottish virtuoso percussionist, who aims to raise the consciousness about sound perception. Since she has been profoundly deaf at age 11, she learned to recognize the audible qualities trough the haptic system. Glennie feels various frequencies with different parts of her body used as a resonance chamber. In fact, she teaches people, how to listen [1]. This example might be an inspiration for several artists/researchers, who would investigate 'bodily listening' during the last decade. We explore this subject, creating another medium for tactile sound aesthetic experience.

2. RELATED WORKS
Satoshi Morita produced an object inspired by his experience during his travel in Switzerland. He was lying inside a tent in sleeping bag, while it was raining outside. Although I didn't even touch or see the raindrops, the feel of the tiny drops was intimately tangible. The membrane of the roof of the tent was synchronized in my imagination with my feel of skin and became a sort of second skin. Satoshi developed the piece Sound capsule (2010), that is intented for one person lying inside this cocoon-like object and perceiving recorded sound via vibrotactile stimuli [3]. Few years earlier similar project called Sonic Bed, was conceived by Kaffe Matthews. This object seems to allow the listener to interact in more dynamic way, since it is not closed. The listener is invited to lie in foam mattress framed with the wood, that covers a series of loudspeakers and subwoofers. Person is immersed in low frequencies sounds, penetrating the body [2].

2.1.1. Mórimo
Our issue is the ability of immersion in the sound matter in different context, that afford the listener to perceive a music performed alive and to receive the performers expression. It is not only about exploring body-music interaction. We investigate in new form of artistic expression and performer-listener relation.

3. DESIGN SOLUTIONS
The platform consists of a sound transmitter, intended for one listener. It acts as musical instrument controlled live by a performer through a haptic interface. The listener togheter with the performer are enclosed within one environment, where the studied interaction takes place. The system is designed in a way that this experience include tactile, sonic and visual sensations and it is based on the direct involvement of its participants. Both of them stand vis a vis; the listener is resting his back on the membrane, immersing himself in the sound and looking at the performer, that manipulate this sound with a gesture. One can name this performance as playing the music on anothers body in a distance.
3.1. The receiver

Construction is made of wooden frame, plywood base and wings. An elastic textile is placed on it in a way, that it can be easily taken off and back on. Underneath 5 loudspeakers are attached to build spatial auditory display. On the upper wings one for each side, mid-high frequency tweeters are mounted. At the centre, 12 inch subwoofer spread low omnidirectional vibrotactile sensation. On both side at the lower wings mid range squawker are attached.

3.2. The controller

We develope gesture controller consisting of ultrasonic sensors matrix, that gives a tangible feedback. Performer uses two-dimensional space, manipulating data that are send to the receiver - Mórímo, in order to shape spatial rendering. Direct manipulation in z-plane aims to control the synthesis or to transform sound parameters. The mapping of z-plane range is non-linear. It allows to shape the sound's intensity and to transform the function of synthesized textures indirectly.

3.2.1. Data manipulation

Music is based on soundscape synthesis with an emphasis on texture. The interface gives possibility to loop a sound object, cloud of sonic atoms, producing layer aka tracks. Mixture of non-tonal sound's cues is randomized. Performer is provided with foot controller to change timbre and processing presets. There are eight different sound textures with three modulation for each.

4. FIRST FEEDBACKS

Firstly the receiver was builded and tested with simple samples of various low frequencies. We customized certain speakers on each level for different parts of the body, that has different sensitivity on sound frequencies [4]. Then we made some tests of sound control with bio-signals mapping. We used galvanic skin response sensor to track the emotional input of performer. We tried to combine it with gesture control, to allow dynamic expression, and then we tried also the gesture control alone. The best result was just a gesture control, that afford an easy, natural expression together with easier sound control.

Currently the Mórímo receiver prototype is fully compound and tested together with the first trial version of the controller. We get an interesting feedback. At the first moment the listener is trying carefully the membrane with his back, moving gently. When he feel the tactile sensation, he's focusing on it for few moments and then starting to observe the performer's gestures. Those gestures are visibly synchronised with the signals received from the sense of touch. That gives this feeling, like one of the listener called, of being touched by the performer from a distance. At the same time the performer get a tactile feedback on his palms, manipulating with the sound, that is send to the receiver. This create extraordinary expression, nearly intimate relation between both participators. After this experience testers claim to be much more sensitive for the everyday phenomena of tactile sensations coming from the sound. Few of testing listeners find even some mystical emotion derived from the form of this medium.

5. FUTURE WORK

The project Mórímo won an artistic scholarship of Bia dystok city's president to realize the workshops integrating hearing-impaired and normal hearing children. This workshop, in collaboration with specialist pedagogue, aims to develop the sensitivity for tactile sound and various ways of music experience with participants. It is one of possible applications of this medium, that features an exploration in music education beyond certain limitation.

6. REFERENCES

Digital adaptations of the scores for Cage Variations I, II and III
Lindsay Vickery, Cai Hoare, and Stuart James
Western Australian Academy of Performing Arts, Edith Cowan University

ABSTRACT
Western Australian new music ensemble Decibel have devised a software-based tool for creating realisations of the score for John Cage’s Variations I and II. In these works Cage had used multiple transparent plastic sheets with various forms of graphical notation, that were capable of independent positioning in respect to one another, to create specifications for the multiple unique instantiations of these works. The digital versions allow for real-time generation of the specifications of each work, quasi-instantaneous exploration of diverse realisations of the works and transcription of the data created using Cage’s methodologies into proportionately scrolled graphical scores.

1. INTRODUCTION
John Cage’s eight Variations (1958-67) occupy a unique position in the composer’s output. By the late 1950s, Cage had made significant progress in exploring the use of indeterminate sound sources (such as radio and LP recordings), a range of chance procedures for generating notation and indeterminacy of notation. His attention now turned towards the indeterminacy and “flexibility” of formal structure itself “a way to further the diversity and flexibility of his compositions by removing the toxicity of the score itself” [26]. The eight Variations were the principal work for the exploration of this idea, constituting nearly a quarter of his compositional output during this period. Following the completion of Variations VIII, the most open of the works in every respect, Cage returned, for the most part, to more traditional compositional outcomes marked by his exploration of the “recomposition” of pre-existing works.

<table>
<thead>
<tr>
<th>Score specification</th>
<th>sound sources</th>
<th>performance space</th>
</tr>
</thead>
<tbody>
<tr>
<td>I (1955)</td>
<td>quasi-determinate</td>
<td>instruments</td>
</tr>
<tr>
<td>II (1961)</td>
<td>sound producing means</td>
<td>unspecified</td>
</tr>
<tr>
<td>III (1963)</td>
<td>indeterminate score</td>
<td>actions</td>
</tr>
<tr>
<td>IV (1963)</td>
<td>topographical map</td>
<td>sound producing means</td>
</tr>
<tr>
<td>V (1965)</td>
<td>astronomical chart</td>
<td>electronic sound systems</td>
</tr>
<tr>
<td>VI (1966)</td>
<td>sound system component diagram</td>
<td>integrated</td>
</tr>
<tr>
<td>VII (1966)</td>
<td>real-time sounds</td>
<td></td>
</tr>
<tr>
<td>VIII (1967)</td>
<td>“silence” (ambient sounds)</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: A summary of Variations I to VIII.

Although Decibel created digital versions of Variations I-IV, this paper focuses upon the digital realization of Variations I, II and III, works that employ multiple transparent plastic sheets inscribed with either point lines or circles, for the purpose of creating a unique score for a performer to read.

There is relatively strong documentation of the evolving non-digital performance practice of the Variations as performed by David Tudor (Variations II-1961 [30], [31]). In this study, Cage (Variations II-1963 [28]), John Cage, Merce Cunningham et al (Variations V-1965-26), [19], David Miller (Variations I and II-2003 [24]), [30]. One of these adaptations was “installation” based, in that they generated both the score and a sonification of the score for viewers to manipulate in an art gallery, rather than scored materials for live performance.

The impetus behind Decibel’s realisation of these works has been principally performative: to create practical tools for the realisation of these works that retain both the indeterminacy and the precision of the Cage’s specification.

2. VARIATIONS I AND II
In Variations I and II, Cage’s materials generate what might best be described as a blueprint for the creation of a determinate score. (Miller describes them as “blueprint” [23 p. 21].) Although Cage states that the score resulting from the application of “rules” of this work may be “simply observed” by the performer, there are significant challenges involved in actualising Variations I and II in this way (as will be discussed below).

At first glance these works appear to be a deconstruction of traditional score, with only the five stave lines and the noteheads remaining and left to float freely in two dimensions. The lines and points are in fact used by the performer to generate a unique score, in which the distance of each point from each line determines one of five musical parameters: frequency, duration, amplitude, timbre and point of occurrence.

James Pritchett identifies the “BV” notation from Cage’s Concert for Piano (1958), illustrated in Figure 1 as the origin of this approach [29]. The connections between the “paper imperfection technique” works such as Music for Piano (1952-6), in which points representing events were spatially located on the page at knots in the surface of the paper and to and the “folded paper templates” of Music for Carillon No. 1 (1952), in which points were notated at intersections between creases in folded paper, are also significant. In Variations I the notation is, more mobile, as the lines and points are printed on transparent sheets, however the “fixes the number and structure of events” is still fixed [289 p. 136].

Earl Brown’s concept of proportional notation [18], developed some years earlier, is taken to its logical endpoint: here everything is measured. The ability to “read” the score in any orientation also draws on Brown’s December 1952 (1954) which may be read in any direction (Left to Right, Top to Bottom, Right to Left, Bottom to Top).

The precisely defined multi-parametric nature of Variations I also suggests the influence of the integral serial methods of the European Avant Garde, which had dominated Cage’s compositions [29 p. 78-90]. But most importantly, in these works Cage demarcates a new end point for the act of composition, leaving not only the interpretation, but also the final realisation of the works to the performer.

The materials for Variations I comprise six square transparencies: the first printed with points and the other five printed with lines. Square I consists of 27 points of four sizes corresponding to the number of sounds they represent as illustrated in Table 2.

<table>
<thead>
<tr>
<th>Square</th>
<th>27 Points</th>
<th>No. of Sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Very Small</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Small</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Greater size</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Largest</td>
<td>4+</td>
</tr>
</tbody>
</table>

Table 2: The contents of Variations I square 1

Each of the five additional squares is printed with five lines corresponding to the five parameters shown in Table 3. The performer may freely choose which parameter to apply to each line.

Squares 2-6

<table>
<thead>
<tr>
<th>5 Lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

Table 3: Variations I Squares 2-6 showing the parameters to be assigned to each line.
A reading of the work is created by measuring the distance from each point to each of the five lines to generate a composite of parameters that define each event with the following attributes: number of sounds (1-4+), frequency, duration, amplitude, timbre and point of occurrence. These attributes are relative with the continuum upon which the parameter is measured defined by the performer. For example: the point of occurrence of each event is relative to the total duration of the work (which is not defined by Cage). Figure 2 illustrates the measurement process required to define one event [16].

<table>
<thead>
<tr>
<th>Indeterminate</th>
<th>Frequency/Overtone/Amplitude Range</th>
<th>open</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Instrumentation</td>
<td>open</td>
</tr>
<tr>
<td></td>
<td>Total Duration/Event occurrence</td>
<td>open</td>
</tr>
</tbody>
</table>

Table 4: Determinate and indeterminate qualities of Variations I

Variations II uses a similar system of dots and points, with some small but significant differences. There are six transparencies each with a single line and five transparencies each with a single point. The sixth line determines the structure of the musical event, whether it is a single sound, an aggregate or a constellation of sounds, the function that had been determined by the size of the points in Variations I. The orientation of the lines and points is therefore completely open, meaning that there are an infinite set of potential configurations of the score. A performance consists of any combination of configurations and therefore in theory Variations II may describe any possible musical work [24 p. 42]. In this sense it "represents the most flexible composition tool that Cage ever invented" [29 p.136].

![Figure 2: For each event, five parameters (A-E) are defined by the measurement of the perpendicular distance from each point to each line.](image)

This procedure results in a mixture of determinate, permutable and indeterminate variables in Variations I. The number and position of the points and lines is fixed and there is a finite number of possible combinations and orientations of the transparencies, however the range of the continuum upon which each parameter is plotted is indeterminate. Table 4 illustrates the determinate, permutable and indeterminate factors involved in the generation of an instantiation of the work.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points/Sounds</td>
<td>27</td>
</tr>
<tr>
<td>Lines/Parameters</td>
<td>5</td>
</tr>
<tr>
<td>Min. no. of parameters</td>
<td>135 (27x5)</td>
</tr>
<tr>
<td>Permutations</td>
<td></td>
</tr>
<tr>
<td>Orientation of Points Square</td>
<td>8</td>
</tr>
<tr>
<td>Function of Lines</td>
<td>51 (120)</td>
</tr>
<tr>
<td>Orientation of Lined Squares</td>
<td>8</td>
</tr>
<tr>
<td>No. of Lined Squares</td>
<td>5</td>
</tr>
<tr>
<td>Max. No. of Permutations</td>
<td>38400</td>
</tr>
</tbody>
</table>

![Figure 3: Annotated score for Variations I by Kopatchinskaja [7].](image)

Performance of Variations I and II has traditionally involved one of three methods: "simply observing" [5] the resulting score, annotating an instantiation of the score [5][21] or transcribing the detailed measurements of an instantiation into a "performance score" [24 p. 22]. Figure 3 shows violinst Patricia Kopatchinskaja’s annotation of the score of Variations I [21].

![Figure 4: Reading the score for Variations I in two-dimensional arcs.](image)

The principal issue associated with “simply observing” or annotating the score, as can be observed in Figure 3, is that the notation on Cage’s transparencies is two-dimensional as opposed to traditional one-dimensional linear musical notation. To preserve the order of note occurrence, the transparencies must be read "two-dimensionally" in arcs emanating from the line that determines “point of occurrence” as shown in Figure 4. The distances to the other four lines and calculation of their parametrical value must occur simultaneously.

![Figure 5: David Tudor’s transcription of Variations II [30].](image)

Although David Tudor’s realisation of Variations II relied on “careful definition of measurement scales and a precise performance score” [30 p. 2], James Pritchett shows that Tudor’s version of Variations II reduced Cage’s prescribed measurements to binary values: simple and complex. Figure 5 shows Tudor’s transcription of two events from the work. Tudor’s score overcomes the issue of reading multiple axes (the 50 events he used were aligned in rows), however its transformation of the multi-parametrical notation into single- or double-bordered squares with intersecting lines and circled or plain points is nearly as enigmatic looking as the original.

3. THE SCORE-READER FOR VARIATIONS I AND II

The imperative of generating performance materials that are easily and intuitively read led Decibel to a decision to transcribe the data created in Variations I and II, into proportionally notated graphical scores. In Decibel’s realisations of Variations I and II the parametrical data derived from measuring perpendicular distances is evaluated and then used to generate a scrolling, proportionally notated screen-score. The score moves from right to left with the point of occurrence of each event, rendered as a horizontal rectangle, indicated by its point of contact with a vertical line or “play-head” on the left of the screen. In this way the score moves "towards" the performer from the right in the same direction as a traditional paper score.

![Figure 6: Decibel’s scrolling, proportionally notated screen-score for Variations I. The arrow indicates the direction of the scrolling score.](image)

Duration is represented proportionally by the length of the rectangle. The vertical position of the rectangle indicates its frequency, thickness indicates volume and shade indicates timbre. The number of sounds in each event is specified by a number attached to each rectangle. A portion of such a realisation is shown in Figure 6. The notation draws on conventions established in works by Cage and his colleagues Earle Brown and Christian Wolff, as illustrated in Figure 7.

![Figure 7: Graphical Notation Conventions drawn from a) Cage Arius (1958) [8] - Timbre-Shade equivalence; b) Wolff Duo for Pianists I (1957) [35] - numbers representing the number of sounds in an event; c) Brown Felix and 4 systems (1954) [5] - Proportional Notation: length-duration and thickness-amplitude equivalence [2], [17], [31], [32].](image)
For this purpose there were advantages for implementation in a procedural language like Java by making use of recursive function calls. This proved to be significantly faster to process than using a scheduled message environment like Max. The calculations were made in the following way:

```java
for (int j = 1; j < circles.length; j++)
    if (circles[j].getGroup() == group && distance <= (circles[j].getRadius() * distanceSquared))
        circles[j].setGroup(circles[i].getGroup());
```

Recursion was used repeatedly throughout all implementations of the Variations. It also proved advantageous to declare all coordinate values in the Java code, rather than in the usual data structures used in Max/MSP such as the cell object.

**4. VARIATIONS III**

In Variations III, Cage moved to a significantly different score paradigm. Here the composer's focus was on aural actions rather than sounds. The score is created by distributing 40 circles (defined as a single large group) onto a surface and then removing all but the largest group of circles that are in direct contact with one another. According to Friedott, Cage's aim was to "make line and direct actions in the performance - one would simply do things and cause the actions and variables to perform themselves." [29 p. 148]

The digital score for the Variations III mimics this procedure first randomly distributing circles on the screen, then eliminating the distances between them and fading out all but the largest group of overlapping circles.

**5. CONCLUSION**

Rendering these works digitally annuls two diametrically opposed arguments often raised against nonlinear indeterminate works such as the Variations I, II and III. On one hand, since the audience always hears the works in a linear fashion sequentially in time there is always the question of which the indeterminacy is somehow "fake" that the performers arranged it before hand. In addition such works sometimes provoke in the audience the notion that the performers are themselves "making it up" because there is no way to determine whether they are accurately reading the score.

The precision provided by the score-players for Variations I and II, arguably lends legitimacy to the performance, because the score that is created is both "accurate" to a reasonable degree and easily read by the performers in a verifiable manner.

On the other hand, such works are sometimes criticized on the grounds that potential existence of other versions implies that the particular one that is being performed might not be the best exemplar of the work. The ability to almost instantaneously generate multiple versions of the work, as demonstrated in Variations III provides the opportunity to choose interesting and promising instantiations of the work.

We have attempted to be as authentic as possible to the specifications Cage prescribed in these three works, using technology to provide a platform that is precise and accurate in its realization while still leaving open the human element of interaction with the score. As Miller expresses in regard to "authenticity" in the performance of these works:

"Cage's formal statements (...) should be taken as points of departure and of periodic return in the course of developing realizations. They are the documents that express, however emotively at times, the works' potentials and particularities." [25 p. 64]
ABSTRACT

The number of solutions involved in many algorithmic composition problems is too large to be tractable without simplification. Given this, it is critical that composition algorithms be able to move through different levels of abstraction while maintaining a well-organized solution space. In this paper we present the following contributions: (1) extended formalizations and proofs needed to implement the chord spaces defined by Tymoczko [11] and Callender et al. [2], (2) a generalized framework for moving between levels of abstraction using quotient spaces that can easily be integrated with existing algorithmic composition algorithms, and (3) an application of both to voice-leading assignment.

1. INTRODUCTION

A major problem in the area of algorithmic composition is the need for organized and easily traversable sets of solutions, also referred to as solution spaces, which are tractable in terms of both runtime and memory requirements. Many music-theoretic ideas are also not formalized to the degree necessary to ensure correct implementation of algorithms and accompanying data structures. In this paper we address both of these problems by presenting a general framework for organizing and traversing harmony-related solution spaces. Our work builds on that of Tymoczko [11] and Callender et al. [2], a generalized framework for moving between levels of abstraction using quotient spaces which can be easily integrated with existing algorithmic composition algorithms, and an application of both to voice-leading assignment.

Our approach to voice-leading assignment uses a type of quotient space called a chord space [2, 11]. Chord spaces are a way to organize chords in musically meaningful ways and provide a convenient, intermediate level of organization between abstract and concrete chords. For example, one such chord space groups chords based on pitch class content, providing a useful level of abstraction for voice-leading assignment. We use this space to turn a sequence of abstract chords represented in terms of pitch classes into a sequence of concrete chords. When finished, each pitch class in each chord is assigned an octave and a particular voice.

There are many other chord spaces that relate chords in different ways. These can also be used with our algorithm to perform variations on the voice-leading assignment task, allowing the algorithm a greater degree of control over what musical features are generated. By simply changing the chord space, our voice-leading assignment algorithm can be generalized to make choices about pitch classes and octaves.

Data-driven algorithms such as Markov chains have been used to learn voice-leading behavior from collections of examples [3, 12]. Markov chains suffer from state explosion when addressing low-level features in music while still capturing structure. Variable-length Markov models [1] and probabilistic suffix trees [10] attempt to address this problem, but are still prone to the same problem with the large alphabets involved in musical problems. Chord spaces [2, 11] can help with this, since they allow generative problems to be broken into multiple steps, each at a different level of abstraction.

Chord spaces, however, present a number of repre-
to our discrete pitch space, \(Z^2\), as well. We make use of three of these equivalence relations:

- **Octave equivalence**, \(O\): Chords belong to the same equivalence class if they have the same vectors of pitch classes: \(\bar{v} \sim_{O} \bar{v'} \iff \forall i \in \mathbb{Z} \circ \bar{x}_i \in \mathbb{Z} (c(x_i) \in \mathbb{R})\). For example, \((0, 4, 7) \sim_{O} \langle 5, 1, 8 \rangle\).

- **Octave, Transposition, and Permutation equivalence**, \(OPT\): \(\bar{v} \sim_{OPT} \langle \sigma(\bar{v} + 12\bar{c}) \rangle \in \mathbb{Z}^2, \sigma \in \mathbb{S}_\mathbb{Z} \) for all \(c \in \mathbb{Z} \). Chords in the same equivalence class have the same intervallic structure of their pitch classes. For example, \((0, 4, 7) \sim_{OPT} \langle 5, 1, 8 \rangle\).

- **Transposition equivalence**, \(T\): Chords with the same intervallic content belong to the same equivalence class. For example, \((0, 4, 7)\) and \((5, 1, 8)\) are \(T\)-equivalent.

The relation is defined in [2] as \(\bar{v} \sim_{T} \bar{v'} \iff \bar{v} \sim_{O} \bar{v'} \in \mathbb{R}\) but we further constrain the definition by requiring \(c \in \mathbb{Z}\) for the remainder of this paper to be consistent with our discrete interpretation of pitches.

The \(O\), \(P\), and \(T\) relations can be used individually or combined to produce additional equivalence relations. Two equivalence relations, \(R_1\) and \(R_2\), can be combined to make a new equivalence relation using the *join operation* \(R_1 \sqcup \delta R_2\) [8]. We will use the term join to denote the transitive closure of relation \(R\). For two equivalence relations, \(R_1\) and \(R_2\), where \(R_1 \cap R_2\) is the composition of the two relations:

\[
R_1 \sqcup \delta R_2 = \left( R_1 \cap R_2 \right) \cup \left( R_1 \cap R_2 \right)^2, \\
\text{for all } \bar{v}, \bar{v'} \in \mathbb{Z}^2.
\]

The join operation is commutative, such that \(R_1 \sqcup \delta R_2 = R_2 \sqcup \delta R_1\). For simplicity, we will abbreviate \(R_1 \sqcup \delta R_2\) as simply \(R_1.R_2\). For two points \(x, y \in \mathbb{S}\), an equivalence space \(\mathbb{S}\) for applying relation \(R\) is set \(S/R\). A chord space is a quotient space formed by applying relation \(R\) and an equivalence relation \(\sim\). Chords form an equivalence relation \(\sim\) on \(\mathbb{S}\) with the same elements:

\[
\bar{v} \sim_{OPT} \bar{v'} \iff \exists \sigma \in \mathbb{S}_\mathbb{Z} \text{ such that } \bar{v} \sim_{O} \bar{v'} \iff \forall c \in \mathbb{Z}\circ \bar{x}_i \in \mathbb{Z}, c(x_i) \in \mathbb{R}, \sigma(c(x)) \in \mathbb{Z}, \sigma(c(x)) \neq c(x). \\
\bar{v} \sim_{OPT} \bar{v'} \iff \exists \sigma \in \mathbb{S}_\mathbb{Z} \text{ such that } \bar{v} \sim_{O} \bar{v'} \iff \forall c \in \mathbb{Z}\circ \bar{x}_i \in \mathbb{Z}, c(x_i) \in \mathbb{R}, \sigma(c(x)) \in \mathbb{Z}, \sigma(c(x)) \neq c(x).
\]

The join operation can be used to combine the OPT operations to produce four other equivalence relations described in [2]: \(OPT, OT, PT, and OPT\) are defined:

1. **Octave and Transposition equivalence**, \(OT\): \(\bar{v} \sim_{OPT} \bar{v'} \iff \exists \bar{v} \in \mathbb{Z}^2, \bar{v}' \in \mathbb{Z}^2, \forall c \in \mathbb{Z}\circ \bar{x}_i \in \mathbb{Z}\). Chords in the same equivalence class have the same intervallic structure when represented as vectors of pitch classes. For example, \((0, 4, 7) \sim_{OPT} \langle 5, 1, 8 \rangle\).

2. **Octave, Permutation equivalence**, \(OP\): \(\bar{v} \sim_{OP} \bar{v'} \iff \exists \sigma \in \mathbb{S}_\mathbb{Z}\circ \bar{x}_i \in \mathbb{Z}\). Chords in the same equivalence class have the same multisets of pitch classes and the same addition modulo \(12\). For example, \((0, 4, 7) \sim_{OP} \langle 5, 1, 8 \rangle\).

3. **Permutation equivalence**, \(P\): Chords with the same set of pitch classes. A chord is represented as a vector of its pitch classes. For example, \((0, 4, 7) \sim_{P} \langle 5, 1, 8 \rangle\).
Recall that \( R \) and \( F \) requirements for equivalent to each other. Therefore, every point outside for \( \text{OP-space} \).

Algorithm 3, \( \text{normOP} \), is a normalization for \( \text{OP-space} \).

If \( \sigma \) is equivalent to \( \tau \) under the conventional sorting of \( \text{OP-space} \), then \( \sigma \) and \( \tau \) are representative subsets of the same equivalence class. The reason for this is illustrated by the points \((0.5, 7)\) and \((0.5, 7)\), which are related by:

\[
(0.5 \cdot 7) \sim (12.5 \cdot 7) \sim (5.7, 12) \sim (0.2, 7)
\]

The point \((0.5, 7)\) should, therefore, be normalized to \((0.2, 7)\) under the conventions of our representative subset. However, we cannot use any of the normalizations discussed so far to accomplish this. \((0.5, 7)\) will be mapped to itself with \( \text{normP} \), \( \text{normT} \), and \( \text{normO} \). The same thing happens with \((0.2, 7)\) as well. Therefore, we have two choices: create one or more new normalizations, or use another algorithm to test whether two chords are \( \text{OP-equiv} \)alent.

We have chosen to use another algorithm that, although makes use of the \( O, P, T \) normalizations, does not define a normalization for the \( \text{OP-equiv} \) relation itself. This algorithm returns true if and only if two chords are \( \text{OP-equiv} \)lent.

Algorithm 5. \( \text{optEq}(\vec{x}, \vec{y}) \) =

1. Let \( \vec{x} = \text{normOP}(\vec{x}) \), \( \vec{y} = \text{normOP}(\vec{y}) \).
2. Let \( \vec{S} = (\text{normOPT}(\vec{S}) \setminus \{0, 1\}) \). \( \vec{S} \) includes all possible sorted intervallic structures of \( \vec{S} \)’s pitch classes. We have that \( \vec{x} \in \{0, 1\} \), \( \vec{y} \in \vec{S} \), and \( \vec{S} \) includes all sorted, \( \text{OP-equiv} \)lent vectors to \( \vec{y} \) within the range \([0, 11] \). Therefore \( \vec{x} \in \vec{S} \) if and only \( \vec{x} \) is \( \text{OP-equiv} \)lent.

3. \( \vec{S} = \{\vec{x} \in \vec{S} \mid \vec{x} \sim \vec{y}\} \).

3. GENERATING CHORD PROGRESSIONS

Finding a normalization for a chord space is important because it is one way to determine which chords belong to the same equivalence class. Many progressions will share the same pattern of representative points when their chords are normalized, so the process of rewriting harmony for n voices using chord spaces can be modeled as a three-step process:

1. Normalizing the chords in the original progression to find a path through the representative subset of \( \text{OP-space} \).
2. Generating all solutions that share the same normalized path.
3. Choosing a solution with the desired characteristics.

In the case of our example for three tenors, \( R \) would be \( \text{OP} \) and the desired characteristics would include specific ranges for each voice.

A sequence of equivalence classes represents many possible chord progressions, each with a unique path through \( X \). In \( \text{OP-space} \), choosing the specific shape of the path through a series of equivalence classes is analogous to choosing a voice-leading behavior. For a chord space, \( \text{OP-space} \) and a sequence of concrete chords, \( X = [x_0, \ldots, x_n], \vec{x} \in X \), the set of chord progressions sharing the same equivalence class of \( X \) is:

\[
\{[y_0, \ldots, y_n] \mid \vec{y} \in \text{E}(X, \text{OP-space})\}
\]

We can further constrain this to some subset of \( \text{OP-space} \), limiting the range of each voice based on instrumental or performer constraints.

Finally, we want solutions that exhibit desirable behavior, such as those having certain voice-leading characteristics (e.g. smooth voice-leading, no voice crossings, etc.). Giving a musical predicate, \( H \), that defines what desirable behavior is for a chord progression, we can further narrow the set of candidate solutions:

\[
\{[y_0, \ldots, y_n] \mid \vec{y} \in \text{E}(X, \text{OP-space}) \text{ and } \vec{y} \in X\} \subseteq \text{OP-space}
\]

3. MUSICAL PREDICATES

We define two types of musical predicates: pairwise and progression predicates. Pairwise predicates apply to pairs of chords, and progression predicates apply to a sequence, or progression of chords.

First, we define a pairwise predicate that rejects cases where voice crossing occurs. When voices cross, a permutation is caused, and the chord’s voices will not correctly sort the other chord’s voices.

\[
\text{hCross}(\vec{x}, \vec{y}) = \exists \vec{z} \in \vec{S}_r \wedge (\vec{x} = \text{sort}(\vec{z}) \land \vec{y} \neq \text{sort}(\vec{z}))
\]

3.2. Algorithms for Applying Predicates

Finding a chord progression satisfying certain predicates is analogous to the satisfiability problem in computer science, which is NP-complete for arbitrary formulas on Boolean variables [7]. Because of this, there is a tractability issue involved in finding candidate solutions that satisfy one or more potentially arbitrary predicates. If there are \( k \) possible choices for each of \( m \) chords in a progression, there are \( 4^m \) total possibilities. For a given quotient space \( S/R \) and predicate \( H \) we clearly need a more efficient method for finding solutions than generating all \( R \)-equivariant solutions, storing them, and then looking for cases satisfying \( H \). An algorithm generating \( H \)-acceptable solutions must perform more aggressive pruning of the solution space.

One way to prune the solution space is through the use of predicates that operate on sub-progressions, such as pairwise predicates. An algorithm can apply the predicates while generating partial solutions. We present one such algorithm below for pairwise predicates. While this general strategy does not change the complexity class of the problem, it avoids computing and storing unnecessary progressions.

Algorithm 6. \( \text{pairProg}(R, S, \text{hpair}, [\vec{y}_0, \cdots, \vec{y}_n]) \) =

1. \( \vec{y}_0 = \text{return } E(\vec{x}, S, R) \mid \{\vec{x} \in \vec{S} \land \vec{x} \equiv \vec{y}_0\} \), otherwise continue.
2. \( \vec{y}_0 = \text{return } \text{pairProg}(R, S, \text{hpair}, [\vec{y}_0, \vec{y}_1, \vec{y}_2]) \).

Even when solutions are filtered using predicates, the work involved in traversing the entire set of solutions to locate desirable ones and even the number of desirable solutions can be intractable in situations involving many chords, many voices, and/or large ranges for the voices. Fortunately, in music, rules are not always strict, and so it may be sufficient to find a solution that mostly satisfies a set of predicates, even if some parts of the solution violate the predicates. We present an alternative, greedy algorithm for generating chord progressions, that while not guaranteed to find a solution satisfying a progression predicate, will attempt to satisfy a pairwise predicate when choosing each chord.

Algorithm 7. Let \( S \) be the chord for which we wish to find a new \( R \)-equivalent member of \( S \subset \text{OP-space} \). Let \( \vec{y}_0 \) be the previously chosen chord, choose\( \vec{y}_0 \) as a function to stochastically select an element from a set, and \( (\vec{y}_0, \vec{y}_1, \cdots, E(\vec{x}, S, R) \) as a fallback method for choosing \( \vec{y}_1 \).

1. \( \vec{y}_0 = \text{return } \vec{y}_0 \in E(\vec{x}, S, R) \mid \text{pairProg}(R, \text{hpair}, [\vec{y}_0, \vec{y}_0]) \).
2. If \( \vec{y}_0 = \text{then return } \vec{y}_0 \in E(\vec{x}, S, R) \).

Otherwise, return \( \vec{y}_0 \) to choose \( \vec{y}_0 \).
Algorithm 8. greedyProg\((x_1, \ldots, x_n)\), \(S \subseteq \mathbb{R}, R_{H_{opf}}, f = (y_1, \ldots, y_m)\), where

\[
W_i = \begin{cases} 
\text{choose}(E(x_i, S/R)) & \text{for } i = 1 \\
greedyChord(x_1, \ldots, x_i, R_{H_{opf}}, f) & \text{otherwise}
\end{cases}
\]

The main advantage to this approach is that the solution space is maximally pruned at each step. This allows the algorithm to operate on inputs that would cause tractability problems for pairProg. The downside is that greedyProg is not guaranteed to find predicate-satisfying solutions. It is possible to find a partial solution with no subsequent choices that satisfy the supplied predicate. In such a situation, there are three options: fail and return an error message, backtrack to try to find a better solution (analogous to lazy evaluation of pairProg), or try another predicate. For our implementation, we chose the latter: a fall-back method for choosing a next chord is therefore required if we wish to ensure that greedyProg produces a solution. In practice, it may be sufficient to have a result that mostly satisfies a predicate even if some chord transitions do not. Since this greedy approach does not require examining all possible solutions, it presents a more tractable option for larger scale composition problems.

3.3. Moving Between Levels of Abstraction

Problems of solution space size mentioned so far are more prominent for chord spaces with very large equivalence classes. For example, a given equivalence class in OP-space can be larger than in O-space. Similarly, OP2-space will have some larger classes than OP-space. However, before generating a final solution, we can move between levels of abstraction using only representative subsets at intermediate steps. For example, it would be possible to find a path through a representative subset of OP2 space using predicates on chord quality and then, in a second step, transform that abstract path into one through OP with predicates to shape the voice-leading. If we were to try to find solutions in OPT2-space directly using a predicate for both chord quality and voice-leading behavior, the number of initial possibilities to explore would be much larger.

4. APPLICATION TO VOICE-LEADING

We implemented the algorithms described in the previous sections using Haskell, a lazy programming language that allows for a concise and elegant implementation.\(^1\) We present examples relevant to our originally described problem: re-writing existing chord progressions. Durations of chords shown in the examples do not change, since our algorithms only make decisions about the assignment of pitches to voices (not duration).

\(^1\)Our implementation is available at the Yale Haskell Group website, http://haskell.cs.yale.edu/.

Our results show potential as a framework for some tasks in automated composition. We control specific solution choice with the use of predicates and we use chord spaces as a way to organize the solution space at different levels of abstraction. Our results also highlight several important issues in the representation of chord spaces for compositional tasks as well as problems with the tractability of searching a particular solution space under a set of predicates.

We present sample results using a three-voice input progression created by the authors, shown in Figure 1. This chord progression is then re-written using our algorithms to find a different chord spaces: OP-space and OPT-space. The examples make use of the following predicates:

- \(h_{cross}(a, b)\) := true iff no voice crossing occurs between \(a\) and \(b\).
- \(h_{noPar}(a, b)\) := true iff no parallel motion occurs between any pair of voices in \(a\) and \(b\).
- \(h_{max}(a)\) := true iff no voice moves more than 7 halfsteps from \(a\) to \(b\).
- \(h_{all}(p)\) := \(h_{cross}(p) \land h_{noPar}(p)\).

For the three-voice progression in Figure 1, two OP-equivalent and two OPT-equivalent progressions are shown. Figures 2, 3, and 5 utilize the range of \([36, 57]\) for each voice, for example, would increase the odds that a solution generated with \(\text{pairProg}\) of Figure 1’s progression as input, and \(\text{hall}\) of Figure 1’s progression as input, and \(\text{hall}\) of Figure 4’s progression as input, and \(h_{all}\).

Figure 1. A simple chord progression for three voices.

Figure 2. A solution generated with greedyProg using \([36, 57]\) / OP, figure 1’s progression as input, and \(h_{all}\).

Figure 3. A solution generated with pairProg using \([36, 57]\) / OP, figure 1’s progression as input, and \(h_{all}\).

Figure 4. A solution generated with greedyProg with an always-true predicate and a transposed version of \(S_{opt}/\text{OPT}\) (4 octaves higher for readability).

Figure 5. A solution generated with pairProg using \([36, 57]\) / OP, figure 4’s progression as input, and \(h_{all}\).

When using OP-space to choose voice-leading for pitch classes, it may be also useful to learn a particular style or composer’s method of choosing voice-leading from a data set. Similarly, equivalence relations for concepts such as chord substitutions and short progressions could be learned. Machine learning algorithms such as probabilistic suffix trees \(^1\) may aid in extracting relevant sub-progressions to form concepts such as chord substitution for use in more complex quotients of spaces of short chord progressions.

4.3. Representing Equivalence Relations

One shortcoming of our implementation is the need for single, Boolean tests for equivalence under multiple relations. Partitioning a set under multiple relations creates representational problems: either the space must be large enough that it contains all intermediate points needed to preserve transitivity, or functions must be developed to test for equivalence under multiple relations in one Boolean
test. In other words, if we wish to compute \( S(R_1) / R_2 \), we must first find a suitable test, \( t \), for \( R = R_1 \land R_2 \) and then compute \( S / R \) directly instead of first finding \( Q = S(R) \) and then computing \( Q / R_2 \). As we have shown, this is possible for the \( OPT \) relations. It is not always possible to find simple and concise formulas to combine multiple equivalence relations into single Boolean tests as we have done for all combinations of the \( O, P \), and \( T \) relations. In the case of \( OPT \) equivalence, we relied on understanding of the music theoretic aspects of the relations to create a simple Boolean test. Equivalence relations learned from data sets could require storing and traversing the entire learned model to perform equivalence tests.

Storing an entire chord space is also problematic, since the number of chords grows exponentially with the number of voices. For large numbers of voices, it may be necessary to generate the quotient space on an as-needed basis. These issues must be addressed in order to pursue applications that involve large quotient spaces.

4.4. Algorithms for Satisfying Predicates

The size of the solution spaces demonstrated by our simple examples shows the need for intelligent methods of traversing these spaces without generating and storing each possible solution. With lazy evaluation in a language like Haskell, it is relatively easy to calculate the first solution possible solution. With recent advanced audio manipulation technologies and widespread use of video sites on the Internet, it became commonplace to create original songs and broadcast theses to the world. These situations can motivate “potential users” who are not skilled at composing or mixing music, but have a desire to create their original music. Many difficulties of creating music have been overcome by the introduction of technologies for music content manipulation. Digital audio workstation (DAW) has enabled users to obtain tracks with high quality instrument sounds and edit music through cut and paste and adding audio effects. Furthermore, focusing on the more detailed aspects of composition, our research investigates how computers can assist users in overcoming the difficulties of composing melodies, especially when they do not have much proficiency or pertaining to composition.

Since early research on automatic composition with computers \[1\], discussions have been made on how computers can assist the user’s creation of musical compositions \[2\]. Graphical interfaces have been exploited for composition systems as well \[3\], which make it possible for users to handle more abstract commands for music generation. New computer languages and data structures could be used as well, such as quotient spaces formed from sets of short progressions rather than sets of individual chords. In our ongoing research, we are investigating the construction of equivalence relations to form more complex quotient spaces, as well as more efficient ways to store and traverse them.

Acknowledgements

This research was supported in part by NSF grant CCF-0811665.

6. REFERENCES


ASSISTANCE FOR NOVICE USERS ON CREATING SONGS FROM JAPANESE LYRICS

Satoru Fukayama, Daisuke Saito, Shigeki Sagayama

The University of Tokyo

Graduate School of Information Science and Technology

7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan

{fukayama,dsaito,sagayama}@hil.u-tokyo.ac.jp

ABSTRACT

This paper describes a system designed to assist users in creating original songs from Japanese lyrics with ease. Although software which helps in accomplishing this task has advanced recently, assisting users in going through the difficulties of composition is still a challenging task. We discuss a possible solution for assisting composers through three approaches; to design a system with direction functionality in generating songs, to formulate composition as an optimization problem, and to integrate a synthesis and analysis engine of vocals and lyrics. After 54 days of operation of our implemented web-based system, 15,199 songs were automatically generated by 5.908 distinct users. On average, 2.33 songs were generated per access to the website per user and a wide variety of composition parameters were chosen for song generation. The results indicate that our method is able to greatly assist users in generating original songs from Japanese lyrics.

1. INTRODUCTION

With recent advanced audio manipulation technologies and widespread use of video sites on the Internet, it became commonplace to create original songs and broadcast these to the world. These situations can motivate “potential users” who are not skilled at composing or mixing music, but have a desire to create their original music. Many difficulties of creating music have been overcome by the introduction of technologies for music content manipulation. Digital audio workstation (DAW) has enabled users to obtain tracks with high quality instrument sounds and edit music through cut and paste and adding audio effects. Furthermore, focusing on the more detailed aspects of composition, our research investigates how computers can assist users in overcoming the difficulties of composing melodies, especially when they do not have much proficiency or pertaining to composition.

Since early research on automatic composition with computers \[1\], discussions have been made on how computers can assist the user’s creation of musical compositions \[2\]. Graphical interfaces have been exploited for composition systems as well \[3\], which make it possible for users to handle more abstract commands for music generation. New computer languages and data structures have also been proposed for composition \[4\]. These languages provided ease of use that enabled users to generate music with algorithmic procedures. Computer systems for editing acoustic events and music scores also have been proposed \[5, 6\]. These attempts raise questions about what kind of interface is user-friendly in such systems. Interpretation of musical theoretical knowledge and conventions of compositional methods in a form that computers could handle is also looked into several approaches; Expert-knowledge based systems \[7\], a system based on constraints satisfaction \[8\], systems with genetic algorithms \[9\], imitating musical styles with example based programming \[10\], probabilistic modeling of music \[11, 13\] and machine learning \[14\]. These approaches support users when they do not have enough technical background on music or composition.

The aim of this research is to design a system which assists novice users in creating original songs easily. By reviewing the previous automatic composition methods from the viewpoint of composer’s assistance for novice users, three problems arise: (1) how to give directions on generating songs, (2) how to maintain consistency regarding musical theories, and (3) what the most easy-to-use interface will be. In the following sections, we discuss the approaches taken to deal with these problems. They are (1) to design a system which combines the direction functionality on generating songs, (2) to formulate composition as an optimization problem, and (3) to integrate a synthesis and analysis engine of vocals and lyrics.

2. SYSTEM DESIGN FOR ASSISTANCE ON CREATING SONGS FROM JAPANESE LYRICS

2.1. Design for giving directions on generating music

2.1.1. Direction based on decomposed components

In order to give directions on generating melodies of songs, providing examples of existing songs can be effective. Since music can be broken down into components such as melody, harmony and rhythm, songs can be also decomposed into rhythm of melody, chord sequence, accompaniment and others. In addition, these musical components correlate with one another. For instance, when the chord sequence is in a sad mood, the melody with that chord sequence tends to also be in a sad mood. Hence, posing direc-
tions on generating melodies are possible by referring to the musical components which can be found in existing songs.

Let us define \( f \) as a song with melody \( m \) that the user would like to generate, with a direction that reflects the mood of the chord sequence and a rhythm of the melody appearing in \( s_1 \), and an accompaniment in another song \( s_2 \). By representing the composition of song as \( f \), the composition can be formulated as follows:

\[
\begin{align*}
  \hat{s} &= (\hat{b}_1, \hat{r}_1, \hat{a}_2) \\
  \hat{f} &= (\hat{c}_1, \hat{r}_1, \hat{a}_2),
\end{align*}
\]

where \( c_1, r_1 \) and \( a_2 \) are the chord sequence in \( s_1 \), rhythm of melody in \( s_1 \) and pattern in \( s_2 \), respectively. Those are obtained with:

\[
\begin{align*}
  (c_1, r_1, a_2) &= f^{-1}(s_1) \\
  (c_2, r_2, a_2) &= f^{-1}(s_2),
\end{align*}
\]

where \( f^{-1} \) represents the decomposition of music. Methods for designing \( f \) will be discussed in Section 3. Since decomposition of songs is a difficult task, the preparation of libraries which contain typical patterns of chord progression, rhythm of the melody and the accompaniment is proposed.

Variety can be expected in the generated results by taking advantage of the vast number of possible combinations of patterns in the libraries. For instance, if 20 patterns were prepared for chord sequence patterns, melody rhythm patterns and accompaniment patterns respectively, 20\(^3\) types of melodies will be possible. Although duplication might happen in music styles of the melodies due to poor variety in the libraries, a careful design of the libraries should be able to handle this. In addition, in order not to confuse users by having them choose a lot of parameters, preset parameters can be prepared for each of the musical styles.

### 2.1.2. Editing functionality of composition parameters

Although prepared libraries provide users with the ability to impose directions easily for generating melodies, it may cause limitations for creating songs, as their composition will be subject to the available patterns in the libraries. A possible solution for this problem is to install a pattern editing functionality with capabilities such as editing chord sequence patterns, rhythm patterns for the melodies, accompaniment patterns, and analysis results of accent and phrasing of the lyrics. These interfaces can assist users in composing songs with more specific directions, nims the difficulties of writing totally new chord sequences, rhythm patterns and so forth.

### 2.1.3. Editing tentative generated results

After the songs are generated by the system, a user may want to change the details of the generated songs. Here, it is hypothesized that users will feel editing an existing result less cumbersome than creating a song from scratch. This concept of user assistance can be implemented by enabling reference to the composition parameters when the songs are generated, and setting a resume button in the interface for setting the parameters.

### 2.2. Design of a user-friendly interface

Lyrics are an easy-to-use input for novice users on music since they require little or no musical knowledge in writing them. In case the user cannot find appropriate lyrics to attach melody, an automatic lyric generator with natural language processing techniques can be employed, such as techniques for interpolating between specified keywords using N-gram models. In general, a lot of information included in the lyrics to be reflected in songs are difficult to extract (e.g. semantic information in the lyrics). However, prosody for the lyrics is relatively easy to estimate from the lyrics input with the language processing front-end of the text-to-speech engine. Direction of structure often appears to be related to how the lyrics are structured. Therefore, the linefeed code in lyrics input can be used for generating the structure segment for the music.

It is possible to display the score in musical notation for the generated results. Generated results are able to represent in score with music notation language. Assuming there are users who are not necessarily capable of reading musical scores, the results should be in practice. Accompaniment audio track can be generated from MIDI data with a MIDI synthesizer. Singing voices can be generated with the singing voice synthesizer. A variety of voice qualities can be obtained by varying the training data set or the vocal tract length parameter which can be specified in the model.

### 2.3. Maintenance of consistency between musical components

Since there are dependencies between the musical components and the melody, maintaining consistency between musical components during melody generation is necessary. Directions given on the melody are: (1) lyrics, (2) chord progression, (3) accompaniment, (4) rhythm patterns for the melody, (5) musical theory such as contrapuntal conventions. Maintenance of consistency beyond the given direction can be handled by using a probabilistic modeling and optimization as described in Section 3.

### 3. ALGORITHM FOR MELODY COMPOSITION FROM JAPANESE LYRICS

In this section, we briefly review the method to compose melody from Japanese lyrics with chord sequence pattern, rhythm pattern of the melody and accompanying pattern. This method is mainly based on the previous research [13].

3.1. Japanese Prosody and its Role in Composition

Japanese is said to “have a fixed shape consisting of a sharp decline around the accentuated syllable, a decline that is usually analysed as a drop from a H² tone to a L³” [15]. Furthermore, as shown in Fig.1, “the place of the accent is lexically contrastive, as in ka’mi ‘god’ vs. kami ‘paper’” [15]. A melody attached to the lyrics cause an effect similar to the accent. Therefore, we can assume that the prosody of Japanese lyrics imposes constraints on pitch motions of the melody.

### 3.2. Composition of Rhythm

#### 3.2.1. Allocation of Lyrics on Melody

We assume that melody consists of segments each of which correspond to a phrase structure and that the lyrics should be divided into segments. For instance, 2 bars can be treated as a segment for a song with a length of 8 bars. Furthermore, in most of classical Japanese songs, one syllable (mor) corresponds to one note in a melody. Thus the number of notes in each segment is determined by the number of syllables. When we consider the constraints on dividing the lyrics, the following 3 criteria can be assumed: (1) the similar number of syllables in each segment is preferred, (2) the border of the segments should not be crossed over within a word, (3) overly short lyrics should be iterated prior to allocation. Under these constraints, we can solve the syllable allocation problem by using dynamic programming.

#### 3.2.2. Keeping Unity of Rhythm in Melody

Even though the numbers of notes in each of the segments are decided, there still exists a large degree of freedom in rhythm. One possible way to put constraints on rhythm is to make it so that the generated rhythm belongs to the same “family” of rhythms. To cope with this matter, a “rhythm tree” can be used, that is one rhythm has similar features, when one can be derived by uniting or dividing the node on the tree. In practice, tree structured templates of rhythm as shown in Fig. 2 are prepared beforehand by hand.

\[ \hat{x} = \text{argmax } P \left( \hat{X} | \hat{r} \right) \]

\[ P \left( x_k | \hat{X}_{k-1}, \hat{r} \right) \propto P \left( x_k | \hat{x}_{k-1}, \hat{r} \right) \]

\[ \hat{x}_k = \text{argmax } \left[ P \left( x_k | \hat{x}_{k-1}, \hat{r} \right) \right], \]
where $P(x_i|y_i^0) = P(x_i|y_i^N)$. Since there are $128^N$ possible sequences of pitch, it is computationally unfeasible to search all of the possible sequences for the optimal one. However, obtaining the optimal pitch sequence becomes $O(N)$ by using dynamic programming [16].

4. ORPHEUS: AUTOMATIC COMPOSITION SYSTEM FROM JAPANESE LYRICS

4.1. Overview of “Orpheus version 3”

“Orpheus version 3” is a web-based system for automatic composition where users can create songs from Japanese lyrics with choice of composition parameters. System design and the composition algorithm discussed in Sections 2 and 3 have been implemented. Figure 3 shows the flow of the system. This is our third version of “Orpheus” automatic composition system series.

4.2. Lyrics input and choosing preset

The lyric input interface appears when the user accesses the web site. The interface is shown in Fig. 4. Here, users can input their lyrics in the text field. Linefeed code is used for setting the structure. Users are also provided with reserved symbols for instrumental segments mark-up, used for the generation of intro or endings to the song.

In case the user could not find out what lyrics to input, the system provides the user with an automatic lyrics generator, which can generate lyrics from input of 1 to 5 keywords and then interpolate between keywords with an N-gram model trained with lyrics database. Radio buttons for choosing the preset parameters set for composition are available at the bottom of the interface in order to avoid irritating users with having to set a lot of composition parameters.

4.3. Giving directions on composition

When the user proceeds past the lyric input interface, the interface for giving directions on composing songs will appear. This second interface is shown in Fig. 5. Each segment of the song is represented with a box. Users can choose composition parameters for each segment. In the latest version (version 3) of our system, around 30 chord progressions, 65 rhythm patterns for melody (including 10 patterns which are able to generate melody with large, and 37 accompaniment patterns are included for the user to give directions on generating songs. Prosody of the lyrics is analyzed with the text-to-speech engine of RealTalk [17], and shown in the text fields located in the boxes. Users can manually correct the prosody by editing the string in the text field. Composition parameters such as key settings and the upper and lower bounds of the melody pitches can be organized with the pull-down menu options. Parameters are also prepared in order to add variety in generating songs such as: the choice of adding the user’s name and the title of the song in the score (two text field on the top of the interface), tempo change (10 choices from 40 to 180 beats per minute), choice of voices (11 choices which are obtained by varying vocal tract length parameter), number of accompaniment tracks (2 tracks in maximum), choice of musical instruments for each accompaniment track, and choice of drums (24 patterns).

4.4. Composition result and results dissemination

As a result of the composition algorithm, users will obtain songs satisfying constraints given by the parameters, musical theories and the up-and-downward pitch motion of the lyrics. This interface is shown in Fig. 6. Vocals are generated with a vocal synthesizer based on a hidden Markov model [18]. Scores are generated with music notation language “Hyphen”. Users can download the score and the audio file of generated songs. Dissemination of results with Twitter is also possible. If the user is not satisfied with the result, it is always possible to return to the previous webpage to change parameters and execute the composition again.

5. DISCUSSIONS ON OPERATED RESULTS

During 54 days of operation, 5,098 distinct users tried “Orpheus version 3” and 15,139 songs were generated with 11,578 access to the composition server (Table 1). Daily comparison of the numbers of generated songs, server access and distinct users is shown in Fig. 7. The results show 280 songs on average were generated daily during operation. The number of distinct users was counted by detecting and removing counts for the same IP addresses in the access log. In order to analyze how well our system assisted the users to compose songs, we calculated the average number of generated songs per access: $R$ with following equation:

$$R = \frac{1}{N} \sum_{i=1}^{N} \frac{A_i}{S_i}$$

where $N$ is the total number of users, $i$ is for the index of each user, $S_i$ is the number of songs which user $i$ generated and $A_i$ is the access count of user $i$. In addition, we excluded the counts for generated songs and accesses which were related with our research team members by checking the IP addresses. Also, the number of accesses where users did not compose is not counted. Results indicate that 2.33 songs in average were generated per access to the system. These statistics indicate that our system provides adequate solutions for the novice users to compose their original songs. Statistics for the chosen composition parameters are shown in Table 2. Results show that a variety of parameters were chosen for composition. This may indicate that users were proficient in giving directions on composing songs with our prepared parameter sets or preset styles. However, more investigations are needed for finding whether the generated results were satisfying the users’ intention or not.

In further versions and future work, in order to remove limitations on user creation using the pre-installed composition parameters, preparing functionality for uploading patterns by users through the web can be suggested. Furthermore, a function of rank those uploaded parameters or generated results may promote the users to compose more original songs. This may bring about a social network of musical composition on the web.
1. INTRODUCTION

Pitch and tone scale organization is one of the many interesting musical parameters, but it is foremost the oldest one that has been studied. In ancient Greece, several scientists/philosophers searched for the intimacies of sound and its sympathetic vibrations. Using a monochord, Pythagoras revealed the intervallic ratios for his tuning system that was used for ages. It was the beginning of an ad
venture between mathematical and physical theories versus sounding realities and musicality, where people empirically spoke about commas, wolf tones, theory of affects and temperaments. In Western music, music theory developed gradually towards an equal temperament, with some exceptions by experimental composers. In non-Western classical music however, many alternative tuning systems that use specific intervals such as quartetones were described. In oral music, one encounters even more different scales, that were developed in a master-student relationship, tangled in a functional and societal context and less depending on theoretical framework.

We described a method to assist novice users in the creation of original songs from Japanese lyrics, and introduced our system “Orpheus version 3”. In order to help navigate the user through the difficult process of composition, we proposed the following three system designs for the solutions: (1) to design a system with direction functionality in generating songs, (2) to formulate composition solutions; (1) to design a system with direction functionality in generating songs, and (3) to design a system synthesis and analysis engine of vocal and lyrics.

Evaluation of our method took place through the web service of our automatic composition system. The average number of generated songs per user in a access to the webserver was 2.33 songs, and in total 15.139 songs were generated automatically during 54 days of operation. Various presets for composition parameters were chosen for giving directions on generating songs. These results indicated that our method was a possible solution for encouraging novice users to compose their original songs. For future work, we plan to add functionality for uploading music components and recommending generated songs.

2. RESULTS

This paper elaborates on a setup for microtonal exploration, experimentation and composition. Where the initial design of the software Taros aimed for the scale analysis of ethnic music recordings, it turned out to deliver a flexible platform for pitch exploration of any kind of music. Scales from ethnic music, but also theoretically designed scales and scales from musical practice, can be analyzed in great detail and can be adapted by a flexible interface with auditory feedback. The output, the scales, are written into the standardized Scala format, and can be used in a MIDI-to-WAV converter that renders a MIDI file into audio tuned in a particular scale. This setup creates an environment for scale tune exploration that can be used for microtonal composition.

3. REFERENCES

<table>
<thead>
<tr>
<th>Reference</th>
<th>Source</th>
</tr>
</thead>
</table>

We started to look at the differences between the tones that were written into the standardized Scala format, and can be used in a MIDI-to-WAV converter that renders a MIDI file into audio tuned in a particular scale. This setup creates an environment for scale tune exploration that can be used for microtonal composition.

This paper elaborates on a setup for microtonal exploration, experimentation and composition. Where the initial design of the software Taros aimed for the scale analysis of ethnic music recordings, it turned out to deliver a flexible platform for pitch exploration of any kind of music. Scales from ethnic music, but also theoretically designed scales and scales from musical practice, can be analyzed in great detail and can be adapted by a flexible interface with auditory feedback. The output, the scales, are written into the standardized Scala format, and can be used in a MIDI-to-WAV converter that renders a MIDI file into audio tuned in a particular scale. This setup creates an environment for scale tune exploration that can be used for microtonal composition.

The final chapter states several case studies. The final chapter states several case studies. The final chapter states several case studies. The final chapter states several case studies. The final chapter states several case studies. The final chapter states several case studies.
Figure 1. Circular triad between input and output. Tarsos analyzes audio. Scala organizes the output of the analyses, which can be uploaded again in Tarsos so any scale can be rebuilt and sonified.

Figure 2 shows a conceptual visualization of different musical concepts and transformations. As can be seen, it is e.g. possible to start with an audio file in Tarsos, export a scala file with a detected tone scale or play a midi keyboard.

Figure 3. This song uses an unequally divided pentatonic tone scale with mirrored intervals 168-318-168, indicated on the pitch class histogram (horizontal axis visualizes the octave reduced view). Also indicated is a near perfect fifth consisting of a pure minor and pure major third.

an interesting mirrored set of intervals is present: 168-318-168. This phenomena has been encountered several times in the RMCA archive: a scale that is constructed around a set of mirrored intervals. It could be a (unconscious) cognitive process to build a scale around such set of mirrored intervals, but makes it also more convenient to perform for human voice.

2. CASE STUDIES

The aim of this research is microtonal composition based on exploration of musical pieces that contain microtonal pitch classes. Therefore some case studies have been chosen to raise a corner of the veil.

2.1. Ethnic scales

Ethnic music offers a unique environment of characteristic timbres, rhythms and textures that need adapted or completely new, innovative tools. The potential of computational research within the context of ethnic music has been stressed by the introduction of the term Computational Ethnomusicology[7]. Hopefully this new interdisciplinary (sub)field can give some impetus to the study and dissemination of a rich heritage of music that is now hidden in archives and aid or even stimulate new musicological field work [6]. As an example for computational pitch analysis, an interesting song is found in the archives of the Royal Museum for Central-Africa (RMCA, Belgium). This song, recorded in Burundi in 1954 by missionary Scothy-Stroobants, is performed by a singing soloist, Leonard Ndengabaganizi. The detected intervals, visualized in Figure 3, are respectively 168, 318, 168, 210, and 336 cents; a pentatonic division that comprises small and large intervals, rather than an equal tempered or meanante divisi. A capella singing does give some variation in pitch classes, but still some particularities can be described: although diverse interval sizes, three nearly fifths are present in the scale. One of these fifths is built by two thirds that resemble a pure minor third and a pure major third (that lies between the intervals 168 + 210 = 378 cents). Thirdly,

3. MICROTONAL COMPOSITION

Harry Partch, Ivar Darreg, and composers from spectral music really devoted their oeuvre to aspects of microtonality. As a tribute, a composition from Darreg is analyzed here as an example. Tarsos has analyzed and rebuilt the scale that was used in Partch's composition 'From beyond the Xenharmonic Frontier'. This composition uses an equal temperament of 9 tones per octave as

The first one uses the interactive histogram components. One can listen to any histogram in Tarsos by clicking it. A click sends a MIDI-message with a pitch bend to a synthesizer. Therefore some case studies have been chosen to raise a corner of the veil.

3.1. Ethnic scales

Ethnic music offers a unique environment of characteristic timbres, rhythms and textures that need adapted or completely new, innovative tools. The potential of computational research within the context of ethnic music has been stressed by the introduction of the term Computational Ethnomusicology[7]. Hopefully this new interdisciplinary (sub)field can give some impetus to the study and dissemination of a rich heritage of music that is now hidden in archives and aid or even stimulate new musicological field work [6]. As an example for computational pitch analysis, an interesting song is found in the archives of the Royal Museum for Central-Africa (RMCA, Belgium). This song, recorded in Burundi in 1954 by missionary Scothy-Stroobants, is performed by a singing soloist, Leonard Ndengabaganizi. The detected intervals, visualized in Figure 3, are respectively 168, 318, 168, 210, and 336 cents; a pentatonic division that comprises small and large intervals, rather than an equal tempered or meanante divisi. A capella singing does give some variation in pitch classes, but still some particularities can be described: although diverse interval sizes, three nearly fifths are present in the scale. One of these fifths is built by two thirds that resemble a pure minor third and a pure major third (that lies between the intervals 168 + 210 = 378 cents). Thirdly,

3.2. Microtonal composition

Harry Partch, Ivar Darreg, and composers from spectral music really devoted their oeuvre to aspects of microtonality. As a tribute, a composition from Darreg is analyzed here as an example. Tarsos has analyzed and rebuilt the scale that was used in track five from Dettovulatite 'From beyond the Xenharmonic Frontier'. This composition uses an equal temperament of 9 tones per octave as

Figure 2. Detailed circular triad as a block diagram for microtonal exploration and composition.
can be seen in Figure 4. Each interval counts 133 cents, which entails the occurrence of 9 major thirds and 9 augmented fifths in the scale as well. It provides the piece a scale that is built on a mixture of unknown and more familiar intervals. However Tarsos did retrieve all nine pitch classes, as three small deviations of pitch classes were noticed. Each of these three pitch classes measure consequently 38 cents or 59 cents above the three tones from the intended scale (namely 231 633 and 897 cents). They occurred in a specific octave, and not over the entire ambitus. Tarsos will be applied on the entire RMCA archive intending a better insight in African tone scales.

6. REFERENCES


THE XYOLIN, A 10-OCTAVE CONTINUOUS-PITCH XYLOPHONE, AND OTHER EXISTEMOLOGICAL INSTRUMENTS

Steve Mann and Ryan Jansen
University of Toronto, Faculties of Engineering, Arts&Sci., and Forestry

ABSTRACT

A class of truly acoustic yet computational musical instruments is presented. The instruments are based on phones (instruments where the initial sound-production is physical rather than virtual), which have been outfitted with computation and tactaction, such that the final sound delivery is also physical.

In one example, a single plank of wood is turned into a continuous-pitch xylophone in which the initial sound production originates xylophonically (i.e. as vibrations in wood), as input to a computational user-interface. But rather than using a loudspeaker to reproduce the computer-processed sound, the final sound delivery is also xylophonically (i.e. the same wood itself is set into mechanical vibration, driven by the computer output). This xylophone, which we call the “XYolin”, produces continuously variable pitch like a violin. It also covers more than 10 octaves, and includes the entire range of human hearing, over its 122 centimeter length, logarithmically (1 semitone per centimeter).

Other examples include pagophones in which initial sound generation occurs in ice, and final sound output also occurs in the ice. More generally, we propose an existemological (existential epistemology, i.e. “learn-by-being”) framework where any found material or object can be turned into a highly expressive musical instrument in which sound both originates and is output idiophonically in the same material or object, which may include some or all of the player’s own body as part of the instrument.

1. NON-COCHLEAR SOUND

The theme of this year’s ICME conference is Non-Cochlear sound. The notion of non-cochlear sound is suggestive of two things:

1. sound that is perceived by other than the cochlea, e.g. tactile sound (sound that can be felt through the whole body rather than only heard);

2. a metaphor likened to Marcel Duchamp’s “non-retinal” visual art, broadening our perception of what is meant by art, through “Readymades” (ordinary found objects as art, for example). Likewise Non-Cochlear Sonic Art can be thought of as broadening our understanding of sonic art in the Seth Kim-Cohen sense of “Non-Cochlear” [Kim-Cohen, 2009].

This paper presents a methodology and philosophy of instrument-building that embraces non-cochlear in both these senses, i.e. instruments that are tactile (and can thus, for example, be played and enjoyed without the ear—they can even be enjoyed by the deaf!), and instruments that are “Readymades” in the Duchamp/Seth Kim-Cohen sense (with the existential self-determination of the DIY “makes” culture).

2. BACKGROUND AND PRIOR WORK

The work presented in this paper can be thought of as an extension of the concept of physiphones [Mann, 2007] (using the natural acoustic sound production in physical material and objects for computer input devices), which itself may be regarded as an extension of hyperinstruments [Machover, 1991].

2.1. Computer music and user-interfaces

Traditional computer-music is generated by using various kinds of Human-Computer Interfaces (input devices), connected to a computer system, which synthesizes the sound we hear through a loudspeaker system. See Fig. 1, in relation to Fig. 2.3.4., to be described in what follows. Some of the input devices used for computer music are very creative. For example, Hiroshi Ishii of the MIT (Massachusetts Institute of Technology) Media Lab has worked extensively to develop TUIs (Tangible User Interfaces) [Ishii and Ullmer, 1997].

TUIs have been extensively used as user-interfaces [Vergoal and Ungvary, 2001] [Alonso and Keyson, 2005]. Many of these user-interfaces are extensive and creative, and use real-world objects as input devices. For example, Luc Geurts and Vero Vanden Abeele have used a bowl of water with electrical contacts in the water as a computer input device so that splashing the water triggers the playback of a pre-recorded sound sample [Geurts and Abeele, 2012]. Others have created systems that allow anyone to easily turn any objects such as fruit, plants, human skin, water, paintbrushes, or other objects into musical instruments [Silver et al., 2012]. Thus the piano keyboard symbol of Fig. 1 is meant to stand for any of the wide variety of Human-Computer input device in common usage, which can include real world physical objects, such as a bowl of water, as input devices.

2.2. Machover’s Hyperinstruments

In 1986, Tod Machover, from the MIT Media Lab developed the concept of hyperinstruments, in which real physical objects such as a violin, cello, or piano, are fitted with sensors as input devices to a computer which
In this paper, we use the term “Natural User Interface” in a narrower sense to denote tabletop interfaces. In this term, the original sound, not a synthesized sound, is used. It should be noted that such instruments are not merely acoustical instruments but also computer-actuated musical instruments.

2.3. Mann’s Hyperacoustic instruments

Throughout the 1980s and 1990s Steve Mann created a variety of input devices that use the real world itself as the user-interface, for which he coined the terms “Real-ity User Interface” and “Natural User Interface” (NUI) [Mann, 1999]. Before Microsoft Corporation began using this term in a narrower sense to denote tabletop interfaces, Mann et al., 2007]

Individual parts of the tree can then be labeled with chalk, e.g. A, B-Flat, B, C, C-sharp, etc.

3. Proposed Instruments

In this paper, we propose “acoustic physiphones” which are natural user-instruments in which:

• the initial sound production (sound generation) is natural, i.e. acoustic, as with physiphones;
• the final sound delivery (sound reproduction) is by way of the natural material. Thus if the sound originates xylophonically (from vibrations in wood), the processed sound is also reproduced xylophonically (from vibrations in wood), the processed sound is also reproduced xylophonically (by way of vibrations in wood). Preferably the same wood that is used to generate the original sound is used to deliver the processed sound (e.g. pitch-transposed) sound.

The software used for the work done in this paper was written in the “C” programming language, on specialized embedded computers that we designed and built to be completely waterproof and environmentally sealed, so as to operate in a natural environment. We used GNU Linux and wrote our own device drivers to extend the operating system to adapt to the new hardware we built.

3.1. The "Xyolin"

We now present an example of an acoustic physiphone, which we call the "Xyolin", named and invented by author S. Mann. It is a xylophone, but it has infinitely continuous pitch like a violin. It can be played by striking, or by rubbing or bowing (thus giving it the capability to be played either percussively or with infinite sustain for notes of whatever duration are desired).

Various single-plank xylophones were built from high quality Sitka Spruce soundboards. But one of these instruments was made from a piece of rough plywood found in a garbage dumpster. It was fitted with four transducers, one in each corner, which could sense and effect vibrations in the wood. Originally these were used as both listening devices and excitatory devices, but later a much larger transducer was put in the center of the board. See Fig 5. In addition to position tracking by listening (time-of-arrival differences in the various receive transducers, etc.), various other position sensing technologies were used in this work. These included a 24,360 GHz home-made radar set adapted for close range, an ultrasonic range sensor, and an overhead camera to improve the position-sensing (especially while rubbing, when the beat of sound was less discernible), and to recognize various mallets, sticks, gestures, etc.

Annually, fine granules of brightly colored sand were often placed on the board, face-up, so as to form a grating, then be washed away to the overhead camera. In this way the camera can "see" the nodal patterns in the vibrating wood, and this information can be used as part of the feedback loop in driving the transducer(s) to affect the vibrations in the wood. Other variations used ruffle tanks as, or on, the vibrating medium of the instrument.

The board becomes both the input device as well as the soundboard for the instrument, delivering a variety of public performances without the need to use a PA (public address) system.

See Fig 6. When hitting the board with one or more mallets or sticks, the surface texture had little effect on the sound production or sound delivery. But when rubbing the surface with a mallet or stick, the surface texture of the board was found to be very important. It was found that rough plywood, covered in violin rosin, worked best for generating long sustained violin-like notes, through rubbing with a stick also coated in violin rosin.
The xylophone pictured in Fig 6 covers just over 10 octaves, with a resolution of exactly one centimeter per semitone (i.e. 12 centimeters per octave). The centimeters are marked with lines, as is every octave (in bolder lines) but the user can hit the plank between markings to get quarter tones or any other microtonal intervals.

The frequency range of the instrument is from E-flat 0 (19.45 Hz) to E10 (21,096.16 Hz). Thus it spans the entire range of human hearing from less than 20Hz to greater than 20kHz, over its 122 cm (122 semitone) length.

Position is determined by an array of listening devices on the underside of the plank (using initial time-of-flight estimation in the wood, corrected for the differences in the speed of sound going along the grain versus going cross-grain, etc.). Additionally, a side-looking K-band complex (in-phase and quadrature) radar set and an overhead camera run a machine vision algorithm with background subtraction [Yao and Odobez, 2007]. This provides improved tracking accuracy and distinguishes between various malicious feedback loops, and the transducers have to be lifted off the acoustic material before damage occurs.

A simple feedback system is shown in Fig 7(a), with $g_1$ representing an amplified transmit transducer (turns an electrical signal into acoustic vibrations), $P$ representing the physical material through which the sound is fed back, and $g_2$ representing a receive transducer with amplifier (turns acoustic vibrations into an electrical signal). In control theory $P$ is often used to represent a “plant” (e.g. a joint in a robot), and here $P$ literally is a plank when we are using a tree branch. The system in Fig 7(a) is typically unstable and difficult to operate. That is, if we turn up the gains $g_1$ and $g_2$, high enough such that a vibration occurs, the vibration can suddenly grow out of control, in the positive feedback loop, and the transducers have to be lifted off the acoustic material before damage occurs!

Compressors typically act on a signal in the manner shown in Fig 7(b), acting on the amplitude of a signal (determined over several periods of the waveform) rather than being applied at each point in time through the waveform (which would add harmonics due to a nonlinear effect on the shape of the waveform itself). Therefore the natural sound of the acoustic process is preserved, and feedback is controlled and maintained.

In this paper we present controlled feedback in idiophonic media, using adaptive computational processing (e.g. compression, filtering, etc.) to control and sustain feedback with a pitch, timbre, and amplitude that can be accurately and reliably controlled by the player. See Fig 8.

Even though the compressor is nonlinear, we can take a small segment of time over which the compressor’s gain $C_{\star}$ is static, to first order (it gradually varies over the course of many waveform cycles), thus creating a linear feedback system. Over the course of a single waveform, then, the input-output transfer function simply becomes:

$$\frac{Y(s)}{X(s)} = \frac{\frac{g_1}{1+g_1 C_{\star} R}}{\frac{g_1}{1+g_1 C_{\star} R} + g_2 R}.$$ 

This mathematically describes the acoustic response of a feedback system derived from the player, represented by input $x$ in Fig 8. The compressor adjusts $C_{\star}$ to ensure the feedback is sustained.

3.3. Reshaping the “Xyolin”

The embodiment of the one-plank xylophone pictured in Fig 6 works quite well, but we wished to improve both its sound, and its aesthetic form. There is something nice about the aesthetics of a standard xylophone, as the higher notes have shorter bars. We wish to mimic this exponential shape, both for appearance and for improved sound.

Conceptually, we can imagine a xylophone that has 12 wooden bars per octave. A two-octave xylophone will have 25 bars ($12^2 + 1$ to complete the octave), as shown in Fig 9(leftmost). Notice that the rightmost bar is half the length of the leftmost bar, since the fundamental frequency of vibration varies inversely with the square of the length, i.e. half the length results in four times the frequency [Lapp, 2010]. Thus length $= \sqrt{f}$ (length is inversely proportional to the square root of the frequency).

Now we suppose we can make a microtonal xylophone, with quartertones, thus having 51 bars for the same two octave ranges (the rightmost bar still being half the length of the leftmost bar). Therefore, we generate a continuously exponential shape that runs over the entire 10 octave range, as shown in Fig 9(rightmost). The left side of this shape is 32 times taller than the right side.

3.4. A single-plank exponentially shaped xylophone

Cutting out the plank in this shape, gives our instrument a nice new shape, although the number of receive transducers was reduced from 4 down to 3 (and the transmit transducer was moved to a new location closer to the fat end of the plank). The new artistic aesthetic serves a practical purpose. For example, it is now obvious which end is the end for low notes and which end is the end for high notes. The extreme differences between the two ends also helps to make apparent the extreme range of pitches that the instrument is capable of producing. See Fig 10.

However, the shape goes beyond mere aesthetics. Now the lower modes of vibration tend to occur more strongly at the larger end, and the higher modes of vibration tend to occur more strongly at the smaller end. Thus we hear low notes emanate mainly from the large end, high notes mainly from the small end, while midtones emanate mainly from the middle of the plank.

Moreover, when using a stick or mallet with a pickup in it, the infinite sustain actually works better with this new tapered shape. For example, the very narrow end can vibrate easily at very high frequencies, up to and beyond the range of human hearing. The large end works better at low frequencies, especially as it can move more of the surrounding air in the room, in order to better reproduce low pitches. We also preferred the timbral changes to the sound arising from the tapered shape, especially the improved clarity of long sustained high notes.

3.5. Natural User Interfaces

A walk in the forest with a rubber mallet will often reveal fallen tree branches that are sonorous. Accordingly, a fallen branch of Sitka Spruce was found, which sounded quite nicely on its own.

This piece of fallen tree was made into an acoustic xylophone, by fitting it with a transmit transducer and a number of receive transducers. See Fig 11. The result is a highly expressive and sonorous instrument that can be used to play highly intricate recognizable songs and classical or jazz repertoire (including intricate Bach fugues, etc.) as well as new experimental music, owing to the microtonal character and high degree of timbral variability.
Finally, a forest concert was prepared, in which numerous trees were turned into xylophonic ensembles of musical instruments. Special mounting brackets were developed to attach transmit and receive transducers to tree branches, to softly “grasp” the green branches without damaging them. See Fig. 12.

4. OTHER ACOUSTIC PHYSIPHONES

The same principles that apply to our Xyolin, in all its Readymade embodiments, from office desks, to wooden planks, to branches, to forests, etc., can also be applied to other materials. This work was the opening keynote for ACM (Association of Computing Machinery) TEI conference, by way of a performance using ice as an interactive musical medium.

In this performance, we used four transmit transducers, and 12 receive transducers, arranged on and in blocks of ice. Some of the transducers were frozen right into the ice blocks, and others were coupled acoustically to the ice. See Fig. 13.

We also invited audience members to bring forward any object that they wished to turn into a musical instrument. We took requests, e.g. “Can you play Pachelbel’s Canon on this rubber boot?” or “Can you play Gershwin’s Rhapsody in Blue on this soft-cover book?”, which we did.

In this figure, the stick is a “magic wand” containing an active illumination source (LED) as well as a wide-angle projector to spread the light into the surroundings. We attached a camera and projector to this stick. The stick is an interactive touch sensor and musical instrument.

5. READYMADE FOUNTAINS

The proposed method of creating acoustic physiphones from nearly any found objects is not limited to idiophonic sound creation.

As an example of another form of sound creation, a musical instrument was made from a bathtub found in a dumpster. After cleaning out the tub it was fitted with various hydrosounds (12 receive hydrosounds and two transmit hydrosounds), and some waterproof computer equipment. Four wheels were installed, under the tub, one in each corner, to create a kind of “batmobile”. A propane heater was fitted to the tub, so that it could be rolled around while being played. A circular sytem was created from electric pumps running from a car battery and power inverter installed in the underside of the tub, together with the various computational and sensory equipment.

The resulting readymade bathtub is an instrument in which sound:
• originates as vibrations in water, by playing any of the 12 water jets installed on the tub;
• is delivered to the audience by vibrations in the same water.

See Figs. 15 and 16. Sound production and sound delivery are thus hydrophilonic, with computational capabilities and a wide range of acoustic timbres and capabilities. Moreover, the sound is truly tactile, in the sense that participants can feel the sound in their fingertips, and also see the sound vibrations in the water. See Fig. 17.

As a result, hearing impaired musicians can also enjoy the instrument. For example, hearing impaired percussionist Evelyn Glennie played on the instrument, and was able to play and feel melodies and harmonies on it. Thus, like the idiophones presented in this paper, the bathtub instrument is non-cochlear in both senses of the word: it can be experienced without the cochlea, and it also truly references the work of Marcel Duchamp, in many ways!

6. SCIENCE OUTREACH

STEM is an acronym for Science, Technology, Engineering, and Mathematics, and an agenda of public education is integrating these disciplines.

Other interdisciplinary efforts like MIT’s Media Lab ory focus on Art + Science + Technology. Design is also an important discipline, so we might consider DAST = Design + Art + Science + Technology. DAST could put a “heart and soul” into STEM, e.g. going beyond “multidisciplinary” to something we call “multipassionary” or “interpassionary” or “transpassionary”, i.e. passion is a better master than discipline (Albert Einstein said that “love is a better master than duty”).

Consider, for example, DAST = STEM + Design + Art + Science + Technology + Engineering + Mathematics (“dastemology”), or perhaps DASI = Design + Art + Science + Engineering + Innovation. Perhaps we want to nurture the “inventor” (inventor+philosopher), through existemology (existential epistemology, i.e. “learn-by-being”). This goes beyond the “learn by doing” (the constructionist education of Minsky and Papert at MIT).

A simple example of putting existemology into practice is when we teach our children how to measure something, using anthropomorphic units (measurements based on the human body) (wikipedia.org/wiki/Anthropic_units) like inches (width of the thumb) or feet. The human body itself becomes the ruler. We learn about rulers and measurement by becoming the measurement instrument.

Consider a four-year-old learning about water pressure in pounds per square inch or Christinas (her own body weight) per square Stephanie (her sister’s area). The very inaccurate anthropomorphic public understanding is when used across various age groups, is why the concept is so powerful as a teaching tool: it is OK to make mistakes, to take guesses, and to get a rough imprecise understanding of the world around us.

Another example of existemology is wearable computing: we learn about computers by “becoming” the technology in the “cyborg” sense, Learning by Being: Thirty Years of Cyborg Existemology, INTERNATIONAL HANDBOOK OF VIRTUAL LEARNING ENVIRONMENTS, 2006, Part IV, 1571-1592.

Much like the Suzuki method for teaching music, the “Mann method” (author S. Mann) of teaching is based on existemology. The human body itself becomes a musical instrument that teaches physics, states-of-matter, mathematics, and the like.

An example around this idea is Pipe Dreams, a series of performances and demonstrations in 2011, in which author Stuart Manning played instruments while sleeping. A skull cap with 64 brainwave electrodes was connected to a computer that played four instruments, one in each state-of-matter: chimes made of pipes (solid matter); a hydraulophone (liquid matter); a pipe organ (gaseous matter); and a plasmaphone (sound from the fourth state-of-matter).

When the solid, liquid, and gas pipes are arrayed together around the sleeping subject, they form an interesting sculptural form as well. The tubular glockenspiel has pipes that vary in length inversely as the square root of the frequency, whereas the pipe organ has pipes that vary inversely with linear frequency, and the hydraulophone pipes vary inversely with the square of the frequency.

Moreover, the chimes (glockenspiel) are velocity-sensing, whereas the pipe organ is displacement-sensing, and the hydraulophone is absent-sensing. Absence is the time-integral of displacement. More generally, hydraulophones give rise to a new kinematics (“sciamatics”) that includes negative derivatives—velocity: \( \ldots \text{acceleration, abseck, abseber, abselebration, absey, abseble, displacement, velocity, acceleration, jerk, jounce, } \ldots \).

These simple and fundamental aspects like state-of-matter and kinematics allow it to be useful in new ways, beyond music. For example, others have recognized the didactic value of this new kinematics philosophy.
6.1. Water, Forestry, and First Nations Instruments

The water instruments allow a natural element — water — to itself become a musical instrument. We are working to combine water and forestry in a series of musical performances in various forests. One such performance contextualizes the forest canopy as a “cathedral” of sorts, where native flutes are played high in the forest canopy, along a canopy walkway. Additionally, various water instruments are played on and in natural bodies of water in the forest.

In one of the compositions there are three elements:
- Earth: Native Drums, forest, and tree instruments, including the Xyolin. These instruments are played on the ground;
- Water: Hydraulophones, which are played on and in natural bodies of water. Some of these instruments are actually played underwater;
- Air: Native Flutes played high in a forest canopy walkway.

Thus we have Earth on the ground, Water on and in the water, and Air in the air.

The use of the five Elements (Earth, Water, Air, Fire, Idea) is part of our work at the nexus of art, science, technology (engineering) and design (Art, Science, and Technology) outreach.

Lateral thinking within this new “states-of-matter” musical instrument ontology (physical organology) can lead to the prototyping of many new musical instruments in a DIY readymade context well-suited to exotemological outreach.

7. CONCLUSION

We have created several instances of a new kind of computer-based musical instrument in which the sound (a) originates acoustically, and (b) is conveyed to the audience acoustically, i.e. by acoustic vibrations in the physical body of the instrument.

Examples include the “Xyolin”, a xylophone that has infinitely many notes and covers the entire audio range of human hearing, where sound originates as vibrations in wood, and is conveyed to the audience by vibrations in wood, as well as the pagophone, in which sound originates in vibrations in ice, and is conveyed to the audience by waves of vibrations in ice.

The instruments can play any jazz or classical repertoire, intricate Bach fugues, etc., but they can also play a wide range of original works not possible on any other instrument.

Moreover, these new instruments give rise to a new way of thinking about and learning about science, such as states-of-matter, and a new perspective on kinematics that includes negative derivatives of displacement.

8. ACKNOWLEDGEMENTS

The authors wish to thank Andrew Kmic, Jason Huang, Valmir Rampersad, Raymond Lo, Queen’s University, NSERC, and AMD.

References


The Investment of Play: Expression and Affordances in Digital Musical Instrument Design

Joanne Cannon

Interaction Design Lab
Computing & Information Systems
Melbourne School of Engineering
The University of Melbourne
joanne_cannon@bigpond.com

ABSTRACT

This paper introduces the investment of play, its role and significance in the design and development of digital musical instruments (DMIs). Dimension map analyses are used to create a qualitative numerical estimate of DMI expression. Expression is then longitudinal compared to data sets spanning a lifetime study epoch of the Bent Leather Band. This study identifies multiplicity of control and other parameters, as significant affordances for DMI musical expression and skill development. The paper argues that Expression is proportional to the sum of invested play and the processional affordances latent within the DMI system.

1. INTRODUCTION

Many computer music practitioners strive to build expressive digital musical instruments (DMIs) for virtuosic performance. An often-used quote “low entry fee with no ceiling on virtuosity” [17] typifies what many consider to be the optimal qualities of a DMI, i.e. an expressive instrument that can be played immediately, encouraging the development of virtuosic skill in the years to come. DMI virtuosity, however, is yet to be clearly understood. Question: How is virtuosity attained with a DMI? Can it be attributed to the expressive potential of the DMI or the artist(s)?

Playing music can be categorized into a number of activities including but not limited to performing, practicing, improving, exploring, exploring, and selfexpressing.

It can be pursued for recreation, self-development, or as a career, music can be played alone or in an ensemble. The importance of playing music is well understood by traditional musicians. A significant investment of play is considered necessary to develop the requisite psychomotor, timing and aural perception skills for music. Many DMIs promote ease of use, requiring little to no investment of play or the development of skill.

Additionally, the appropriation of game controllers and mobile phones brings highly specialized usability based design features with them. Are these features compatible with longer-term artistic endeavour and the development of virtuosity or expression? To paraphrase: “ease of use may offer a low entry fee along with a low ceiling on skill development”.

Recent studies [9], [13] have identified the evaluation and understanding of computer interaction to be a highly subjective practice, lacking a coherent overview and theoretical framework. DMIs are often dynamic evolving systems, undergoing continual modification to their physical or software components. This makes them difficult to study. Attempts to define the expressive potential of one DMI over another are hotly contested due to DMI practitioners’ diversity encompassing repertoire players, improvisers (best artists), and installation sound artists. As a result, the field of DMIs has remained nascent, adapting and appropriating the latest forms of technology to novel and often short-term musical ends.

A culture of disposable instruments now reigns, where instruments are made and discarded before any long-term play is invested. Although this is an interesting development in the history of musical instruments, it constitutes a profound disconnection with the art of instrument playing, and its highly evolved and formalized practice. Disposable instruments do not promote the facilitation of skill nor do they encourage skilled musicians to want to play them. Disposable instruments confine DMIs practitioners to a technological ghetto, focused solely on technological innovation.

A number of unique and specialized digital instrument musicians argue that practice on one instrument system over extended periods. Andrew Schloss and the Radio Drum, Michel Waiswiz and his instrument the Hands, Serge Lauber and his Meta Instrument, Mark Applebaum and the Meuskeetree, the Hyperstring Instruments by Jon Rose, are each examples of instrument systems performed over the extended period of the DMIs artist(s).

This list is not exhaustive yet it is generally accepted within the field that these artists display a highly developed skill and sense of expression, i.e. digital musical instrument virtuosity.

Dobrian and Koppelman [2] define virtuosity as “complete mastery of an instrument”. They argue that although an instrument affords expression (i.e. Gibson’s affordances of Ecological Psychology) [5] it is the musician’s virtuosity that facilitates expression.

Technically speaking, virtuosity is the attribute of the musician and not the instrument. Expression they argue, is generated from the player and not the controller interface, “control 1 expression” [2]. Also enumerated in Dobrian and Koppelman’s [2] compares actions between an animal (subject) and its environment (or object) based on the animal’s capabilities and the environment’s qualities. “The affordances of the environment are what it offers the animal, what it provides or furnishes, either for good or ill…. It implies the complementarity of the animal and the environment….”

Affordances are best thought of as measurable properties. “For instance we perceive stairways in terms of their climbability, a measurable
property of the relationship between people and stairs” [4]. Atau Tanaka has applied affordance theory to the use of mobile phones as DMIs [12]. The potential of sound to afford human movement has been investigated by Nymoen et al [12]. Other studies examining the properties of music controller interfaces, sensors and their potential to afford performance skill and expression, include [16], and dimension map analyses of controllers and DMIs [1] and [8]. The properties and theory of playable DMIs have been documented and theorised by the Bent Leather Band [3].

Kilborn and Isaksson [7] investigate movement, visual and haptic perception (tactile and kinematic), and the development of bodily skill over time within a controlled hospital environment. Their study demonstrates that the role of haptic affordances change when a person becomes skilled in the execution of a specified task. This can be observed when a person’s movements change from exploratory to performatory i.e. fluent with a “focus on timing” (ibid, 2007). They argue, “the perception of an affordance is crucially connected with the understanding of how to use it”. Central to this understanding is the processional nature of information discovery regulated through nested and sequential affordances [4]. This is observed in instrumental music pedagogy when a musician no longer requires visual instrument feedback and instead feels what is being played through the haptic senses. This moment usually marks a significant stage in skill development. Therefore the musician develops skill and expression over time as a result of the investment of play.

DMI play has yet to be comprehensively investigated. There are very few long-term studies, none longer than a year [10], [14], and [6]. Oore [14] compared experiences learning a number of interfaces with up to one hundred hours of playing time invested. Controllers included a Glove Talk II, a digital Marienot (puppet controller) and piano keyboards. Oore’s study demonstrates processional techniques of skill development including: identifying basic actions, isolating and repeating actions, exaggerating and reducing of actions and performing multiple actions and refinement. Gurevich et al [6] conducted a longitudinal study with a group of nine volunteer musicians using a very rudimentary or “constrained controller”, in fact a hand-held push button device that played a simple monophonic sound. After playing the controller for a number of weeks the group demonstrated highly divergent and personalized styles of performance. Marshall & Newton [10] have conducted a 10week long tailoring study with musicians developing new augmented instrument systems. Although this study has potential to contribute, so far the authors have yet to publish any substantial results. Meyer [11] demonstrates how beginner pianists conceive passages of music as physical movements, visual keyboard information and abstract sound structural information. Advanced pianists in contrast, conceptually transfer physical motor, and visual information to sound or abstract conceptions of the musical phrase. Practiced actions therefore become automatic over time “conceptual meaning dominating articulatory movements” (ibid). This in turn allows for the musician to focus their cognitive attention on other areas such as expression and performance monitoring.

Together these studies identify progressive stages of play investment. A summary and comparison between authors is presented in the table below (see table.1). Working together as a duo, a conservative estimate of our investment of play is in excess of four thousand hours. Our DMIs, including the Light Harp synthesizer controller and Contra-monster augmented bassoon (see fig.1 & 2) have evolved from simple prototypes into elaborate aesthetic pieces. Archived and readily accessible data following these DMIs evolution exists in the form of photographs, software, audio and video recordings.

Both instruments demonstrated a capability to create a wide variety of sound and expression. Of interest were many sections of musical phrases where the performers closely interacted each other’s sound. Controller data revealed each instrument utilized very different approaches to sound creation and control. Both instrument systems however, made use of a many direct 1-to-1 parameter mappings. The data also revealed an extensive amount of controller multiplicity with up to six simultaneous channels of control. Many of these highly articulated musical gestures were consistently reproduced and developed.

From this analysis, 8 dimensions (parameters) describing the expression of the Bent Leather Band’s musical language were identified. These dimensions follow general music descriptors such as pitch, dynamics and timbre etc. however they also include stylistic elements particular to the Bent Leather Band’s own musical aesthetics. These dimensions were subjectively or empirically graded for dimension map analysis of musical expression. The aim of developing the dimension maps was to create a numerical estimate for performance expression. This estimate would then be used to compare separate performances.
1. **Pitch Range:** (quantitative) Measured in Audio Analysis software from lowest to highest pitch, this value was checked against aural transcription. Six classes included: up to 2 octaves, 4 octaves, 6 octaves etc up to 12 octaves, graded one - six respectively.

2. **Pitch Style and Tuning:** (qualitative) This parameter was transcribed aurally and comprised the following six classes: conventional tuning, pitch bending up or down by a tone, vibrato expression, pitch modulation greater than a minor third, free pitch, and microtonal tunings.

3. **Dynamic Range:** (quantitative) Measured in audio analysis software and calculated as a range from the lowest to highest over an entire track of audio. Classes one - six included: 20, 40, 60, 70, 80, 90dB or near equivalent to the maximum dynamic range of CD audio format (96dB).

4. **Timbre Style & Process:** (qualitative) Consisting of six classes in increasing order of timbre technique including: synthesis/live sampling, filtering techniques, modulaton filtering control, comb/harmonic filtering, noise treatments, granular or resynthesis treatments.

5. **Articulation:** (qualitative) This parameter describes the clarity of sonic execution, and definition of phrasing. The scale progresses from **blunted** through to **virtuoso** in six steps.

6. **Ornamentation:** (qualitative) This dimension begins with **trills** and progresses through **melisma**, **gama**kku, **hallicatic**, **multi dimensional** and **complex** approaches to ornamentation.

7. **Number of Parts or Streams:** (quantitative) This parameter reflected the total independent musical parts, each part developing in itself and contributing to the musical flow.

8. **Spatial:** (quantitative) Grades progressed from **mono** through to **multi speaker projection** of sound using spatial motion.

2.2. **Qualifying Expression Evolution**

A number of CD recordings across two decades were then selected for expression map analysis (see figures 3-7, & 8-12). Each date was represented by several contrasting samples/pieces. Dimension maps were then created from the recorded material using transcription and audio analysis. Creation dates for digital files, instruments, sensor hardware systems and key performances were then tabled to create a spreadsheet timeline for longitudinal study.

**Figure 3. Dimension Map for Expression**

From 1992-95, the Light Harp’s sound engine was a Roland JD800 wave shaper synthesizer. From 1995 through to 2012 the Light Harp has continued to be used with a Roland JV-1080 module. The Augmented Bassoons sound engines have continued to migrate first across various commercial hardware signal-processing units [1992-2003], then to software VST plugins [2003-2006] and finally to MaxMSP programs. Of greatest significance was the shift to VST software but since 2005, the signal processing train has remained almost unchanged. Max programming was undertaken when favourite VST plugins became outdated and system incompatible. The duo has mapped sensors with Max since 1993.

**Figure 4. Example of Contra-monster Sonogram and Controller data Analysis**

Overleaf figures 3-12: Dimension Map dates refer to recording of performance. DMI is pictured to the right with the date built.
3. LONGITUDINAL STUDIES

These dimension maps show the Bent Leather Band’s DMI iterations maturing after several prototype stages. The Light Harp remained purely a synthesiser controller while the Bassoon, zurna, serpents and monster, utilized the same approach of live signal processing, using audio delays. It was estimated that during the study period, each musician invested roughly four thousand hours in play, practice, performance, recording and rehearsal. Across the study timeline, expression values increased following corresponding periods of musical play (from a month upwards to almost 6 years between builds). A cross section of DMI system characteristics, describing the controller interfaces, mapping and sound engine parameters were then compared. The majority of these remained constant throughout the study epoch: including total number of sensors, sensor components, characteristics, sensor scanning rates (20kHz) and resolution (7bit, 0-127). Mappings remained simple 1 -to-1 throughout. As mentioned previously the Light Harp has used almost exclusively the same synthesiser, patch and waveform sample throughout. The Max programming language was exclusively used, as was the communication protocol [MIDI].

Interestingly, a number of DMI system characteristics increased proportionally to expression. These parameters were of special interest since they are increased proportionally to expression. These parameters are presented in (charts 1&2) and consist of:

1. Control multiplicity (blue): Corresponding to the total number of sensors (or actuators) that can be played both together and independently (i.e. degrees of freedom) or the number of simultaneous channels of control available. Points on this line represent new DMI builds.

2. Expressiveness (red): A qualitative numerical estimate of expression, the square root of the dimension map’s total sector area, a comparable value falling roughly between 1-10. Points on this line correspond to the dates of music performances sampled.

3. Number of Mapping Layers (green): Mapping layers intercede the sensor/actuator data and their sonic parameters. Layers serve; to rectify sensor errors, exploit an expressive data range, and re-map. The number of mapping layers indicates the depth and level of DMI system refinement. Points on this line represent important changes in mappings.

4. Signal Routings (purple): This parameter refers to the number of audio signal processing (live-mixing) pathways permissible by the DMI. Two pathways may indicate a reverber level control i.e. a dry signal routing with no reverb and a reverberant (wet) signal routing. More signal routings allow for more processing to be combined (mixed) through gestural control. Points on this line also represent mappings.

Both instrument charts display proportionality between expression, control multiplicity, the number of mapping layers and signal routings. Control-multiplicity correlates strongest with expression (0.96 for both instruments). Correlation and a latent proportionality indicate control-multiplicity to be a key affordance of expression in the Bent Leather Band. We can be reasonably confident, that increasing the number of mapping layers and signal routings, will also afford an increase in expression. It is also possible that these characteristics come into play when more channels of simultaneous control are available. Control multiplicity allows for more options to map signal routings to controllers and for more parameter combinations to be exploited simultaneously. For the Bent Leather Band, multiplicity of control plateaus at 6-10 control channels over 16 years, possibly suggesting a threshold for control redundancy.

The trend data also showed a dramatic change in the number of mappings each musician used in any given performance. Each mapping allowed the DMI to play with a different sound, or reconfigure the controller or DMI system to behave in a piece-specific way. The potential for DMIs to be complex and flexible regarding the organization of sonic capabilities is considered a huge advantage over traditional acoustic instruments. Interestingly, the Bent Leather Band trend data shows that by 2006, entire concerts were played on just one single mapping per instrument. This refinement of mapping identifies a shift from exploratory to performative play. This also suggests that the activities of mapping and building DMIs both afford each other in procession (See Chart 3. & 4. below).

With five study epochs and so few sample points, we cannot be completely confident of the trend data. Both authors agree that as more duo play was invested the more each musician’s improvised language adapted to the others’. The evidence presented here will require follow up study with a larger group and more sample points.

4. DISCUSSION

If, as these Bent Leather Band studies suggest, that expression is proportional to the amount of invested play, what does this mean for disposable instruments? Indeed what does it mean for a configurable multi mapping instrument. This paper examines one isolated case where the definition of expression is idiocyncratic and personalized. However, the expression value defined from the dimension maps can also be interpreted as a measure of the interaction between the player and the DMI system. In this sense, the findings of this paper demonstrate, that cyclic and processional affordances can significantly contribute to the design of expressive DMIs. By neglecting the investment of play, the digital musician’s potential to develop their expressive art may be severely diminished. Additionally, if the DMI design...
stance is one that favours ease of use, it is possible that this may significantly impoverish the quality of long term evolving interaction.

Results also suggest there may be an advantage for DMI musicians to work together in duos or possibly ensembles. Although it is not known whether the benefits are there for larger groups, perhaps playing, practicing and performing in a duo may have the potential to double expressive capability.

5. CONCLUSION

The investment of play constitutes a music-centered interaction between a musician and DMI system. An investment of play yields:

- Conceptual transfer
- Perforatory i.e. adroit, fluent action
- Refined musical expression
- Elements of personal style
- Reproducible interaction

The Bent Leather Band studies suggest that expression is proportional to the amount of invested play. In turn this interaction is an artistic process governed by the DMI system’s affordances. These affordances are cyclic and processional, their contribution cumulative over time. They are manifest and latent within the DMI’s controller interface, software mapping and music. These affordances can be identified and harnessed for future DMI system iterations, modifications and establish new generations of instruments.

This paper’s findings suggest that existing DMI frameworks may need to address the investment of play into the process of DMI design, its correlation to musical expression and software mapping. These affordances can be investigated and harnessed for future DMI system iterations, modifications and establish new generations of instruments.

6. REFERENCES


ABSTRACT

In this paper we develop adaptive techniques for mapping generic user interfaces to synthesis engines. Upon selecting a subset of synthesis parameters, the system automatically finds the parameters-to-sound deterministic relationship in a multidimensional space. We analyze this sonic space using two different unsupervised dimensionality reduction techniques and we build the mapping using statistical information on a lower, but maximally representative, number of dimensions. The result is an adaptation of any general purpose interface to a specific synthesis engine, providing control directly over the perceptual features with greatest variance. This approach guarantees a linear relationship between control signals and perceptual features, and at the same time, reduces the control space dimensionality maintaining the maximum explorability of the sonic space.

1. INTRODUCTION

Synthesis engines often expose a large set of parameters to users. Runtime variation of the parameters produces modification in the sound generated as well as in its perception. The physical separation of control from synthesis has promoted the proliferation of a variety of generic control interfaces, enabling reusability of the same controller with different synthesizers and vice versa. Since controllers and synthesizers are not “co-designed” [1], some kind of manual intervention is generally required to establish the “mapping” between them.

In modern music genres the flexibility of the synthesis engine is widely exploited in such a way that notes and chords are looped or generated algorithmically rather than played with individual input gestures. The parameters that the musician modulates, usually represented by a real-valued numbers, result in timbral variation. This trend can be seen in a recurrent interface design pattern where sensors capable of capturing real-valued and time-continuous gesture are usually represented by a real-valued numbers, result in timbral variation. This trend can be seen in a recurrent interface design pattern where sensors capable of capturing real-valued and time-continuous gesture are used as an input to control the synthesizer. The only available options. Therefore, even assuming that it is possible to garner a heuristic understanding of the parameters-to-sound relationship, the desired mapping implementation might be impossible without the introduction of an intermediate processing layer.

In this work we propose a technique to adapt a general-purpose interface to a synthesis engine through:

- automatic analysis of the synthesis engine parameter-to-sound relationship based on perceptually related audio features;
- generation of an adaptive mapping based on the application of unsupervised dimensionality reduction techniques on the multidimensional perceptual sonic space.

Similar one-to-one, one-to-many or many-to-many mappings [2] have been developed through the introduction of an intermediate layer in the perceptual space [3]. Our work is focused on reducing the burden on the user who needs only to provide the system with information about the variable synthesis parameters. The dimensionality of the control space (number of independent signal from the control interface) can be set or modified a posteriori. Other than enabling direct control over perceptual aspects of the synthesized sound, which are user defined, this technique introduces two additional benefits:

- a linear relationship is created between the control signal and the variation in the generated sound, avoiding situations where the controllers’ range leads to drastic sound variation or to null sound variation;
- an optimal mapping is created from a control signal space C, with dimensionality c, to
synthesis engine control parameters space $P$, with dimensionality $p_j$ with $c < p_j$.

The optimal mapping is defined as the one allowing the user to access a sonic exploration when projecting the control space $C$ into the perceptual features in the sonic space $D$, directly related to the synthesizer parameter space $P$. The number of concurrent signals contributed through human gesture is constrained by human physical and cognitive limitations. The consequence is that of dimensionality of $C$ is generally much smaller than $P$. Figure 1 shows a generic control interface driving a synthesis engine through the perceptual sonic space. In our approach this space is retrieved automatically and analysed with unsupervised dimensionality reduction techniques in order to compute the adapted mapping between control interface and synthesized sound.

## 2. SYNTHESES ENGINE PARAMETERS-TO-SOUND ANALYSIS

Within this context we define a synthesis engine as any chain of algorithmic processes that produce audio. We consider this chain of processes as a black box that converts vectors $p$ of synthesis parameters into sound. Moreover we assume a deterministic behaviour, excluding the presence of any stochastic component within the chain. Hence it is possible to state that given a vector $p$, there is one and only one associated sound generated by the synthesis engine. The opposite of this statement may not be always true since, depending on the synthesis engine, different control $p$ may lead to identical or very similar sounds. This will be taken into consideration in the adaptive mapping strategy in Section 3 to avoid potential noisy or discontinuous output.

The set of unique combinations of synthesis parameters $p_{u}$ is defined upon selecting the variable $p$ of the synthesizer parameters, their respective maximum value, minimum value and sampling resolution. Here we assume that each parameter is in the range $[0,1]$ (it can be extended to any range if the scaling operation is applied). Choosing $j$ parameters, the cardinality of $p_{u}$ is given by the equations below:

$$|p_u| = \prod_{j=1}^{p} |p_j|$$

$$\min(p_j) = \min_{\text{resolution}(p_j)}$$

$$p_{u} \subseteq \{p_1, p_2, \ldots, p_j \}$$

Equation (1) shows the cardinality of $p_{u}$ computed through the product of the cardinality of each $p_j$ which in turn depends on the individual maximum, minimum, and sampling resolution (2). The set $p_u$ can be represented with a matrix $P$, where each column is a vector $p$ (unique combination of synthesis parameters).

The cardinality of $p_u$ and the size of $P$ grow exponentially with the number of parameters $j$, and linearly with the sampling resolution values. The selection of these values is an extension of the trade-off between the level of detail in the synthesis engine analysis and the size of $P$. The size of this matrix affects not only memory but computational load as well, as discussed in Section 3.

The sound is generated for each $p_j$ and analyzed to produce a corresponding vector $d_j$ using a fixed note on the chromatic scale. For each unique combination of synthesis parameters we compute not one but a sequence of vectors containing perceptually related features. For timbre that is static over time, the $d_j$ corresponding to the fixed $p_j$ is set to the mean of the sequence of computed feature vectors. This helps to minimize the noise caused by the random position of the analysis window in relation to the generated sound.

For dynamic timbres, such as those due, for example, to the presence of low frequency oscillations (LFOs) in the synthesis algorithm, the sequence of feature vectors is used also to capture extra information about the dynamic aspect of the sound. The vector $d_j$ corresponding to the fixed $p_j$ is set to the mean of the sequence of vectors, adding an extra scalar, which represents the timbre periodicity. Autocorrelation is used to compute the periodicity of each computed feature. If different periods are detected, their mean is used instead. The size and number of the analysis windows define the minimum detectable periodicity, while window overlap affects the maximum. For a better consideration of the dynamic aspect, the vector $d_j$ can be further extended adding a periodicity value and oscillation range for each perceptually related feature, tripling its size.

Vectors $d_j$ are stored in a matrix $D$ and together with $P$ fully characterize the parameters-to-sound relationship of the synthesis engine in the perceptual sonic space. Through column indexing it is possible to associate the $d_j$ unique combinations of synthesis parameters with the relative perceptual features vector and vice versa. The adaptive mapping is based on the information embedded in these two matrices.

## 3. ADAPTIVE MAPPING

Here we assume that the general-purpose control interface generates a set of independent signals with range $[0,1]$ and with uniform distribution. Therefore the space $D$, with dimensionality $c$, can be approximated with a hypercube. To obtain an adaptive mapping we further analyze the matrix $D$ to generate another hypercube in a projected perceptually related parameter space, then the mapping is simply obtained by linking the two hypercubes. The number of control signals does not affect the $D$ post-processing stages to define the mapping; its posterior definition simply restricts the navigation in the sonic space to a certain number of dimensions. However this method guarantees that even with a limited control space dimensionality $c$, the perceptual feature space is explored along the directions corresponding to the maximum variability. This method may not always correspond to the maximum variability in the pure human perception. We describe and apply two different unsupervised dimension selection of these reduction techniques: PCA (Principal Component Analysis) and ISOMAP, both followed by a statistical analysis from which the mapping is derived.

### 3.1. Principal Component Approach

PCA is an unsupervised technique that uses an orthogonal transformation to convert a set of multivariate observations of potentially correlated variables into a set of uncorrelated variables called Principal Components (PCs). Since the matrix $D$ can have high dimensionality, we apply a stage of PCA to project the data into a lower dimensional space. The multivariate data in $D$ is subjected to a prior whitening which scales each dimension to zero mean and unitary variance. The orthogonal and uncorrelated set of PCs is ranked by variance, representing the quantity of information carried by each. Mapping the hypercube $C$ on to the PCs of $D_{PC}$, guarantees control within the subspace where the perceptual features change the most. Compared to PCA, the number of perceptually related features can be relatively high here. It is not necessary to have prior knowledge about variations of features with synthesis engine parameter alteration.

Perceptually meaningful features that are constant are automatically discarded. However, the user can compose and weight individual features in order to customize the adaptive result if desired. In this way it is possible to obtain a control focused on specific perceptual features that are not necessarily the dominant in terms of absolute variability.

To provide a response that is as linear as possible, it is necessary to analyze the data across the PCs. For each dimension the density is estimated through a histogram with a fixed number of buckets, which represent the inverse of the sampling resolutions. Since PC ranges with low density should be explored with a finer step compared to those with high density, we use the complement of the histogram, represented in (5) $\text{hist}^{\text{one}}$. For each PC the mapping function is based on its normalized integral, implemented through the cumulative sum in the discrete domain. Two examples of $c_i$ (vertical axis) mapping over the $PC_i$ (horizontal axis) are showed in Figure 2, where the black continuous lines represent the mapping function. Equation (4) shows how the control signal $c_i$ is transformed into a $PC_i$ value through the inverse of the mapping function $m_i$.

$$PC_i = m_i^{-1}(c_i)$$

$$m_i(p_j) = \int_{PC_i}^{\text{hist}^{\text{one}}(PC_j)} dp_j$$

Figure 2. An example of histograms (scaled 10x) and mapping functions $m_i$ (solid line) for the first two $PC_i$.
The interface signals €c are used to generate a value of the first $e PC$ of $Dc$ with linear interpenetration, obtaining a $d$, in the principal components space. The number of components considered in the system is limited to the number carrying 90% of the total energy. If $c$ is smaller than the number of $PC$, the control signal mapped on the lower rank component is optionally mapped at the same position and all the remaining ones.

3.2. ISOMAP Approach
ISOMAP is a low-dimensional embedding method [8], where geodesic distances on a weighted graph are incorporated with classical scaling. It is exploited to compute a quasi-isometric, low-dimensional embedding of a set of high dimensional data points. At the same time this algorithm provides a simple method for estimating the intrinsic geometry of a data manifold. The main difference with other multi dimensional scaling methods is in the choice of the geodesic distance metric, rather than the Euclidean one. In ISOMAP, the geodesic distance is the sum of edge weights along the shortest path between two nodes, computed using Djikstra's algorithm. The top $n$ eigenvectors of the geodesic distance matrix represent the projection of the data points in the same multidimensional space, ISOMAP implements a transformation of the space, while PCA projects the data into a new coordinates system in the same space.

Dimensionality reduction with ISOMAP is applied to $D$ with the same method described in the previous subsection for PCA. The mapping of the $c$ control signals on the new coordinates system, named $ISO$, is based on an estimate of densities and distributions as before. ISOMAP has a higher computational cost compared to PCA, but it detects and exploits the embedded manifold, achieving a more effective dimensional reduction. ISOMAP is preferred when the control space $C$ has a very low dimensionality. This difference is evident when comparing the energy in each dimension of the residual variance: Figure 3 shows an example of an energy distribution, measured in terms of variance, across the $PC$ and $ISO$ for the same data set. The number of vectors $Dc$ in $D_{ISO}$ can be lower than $Dc$, because the ISOMAP algorithm includes an outlier removal stage. To guarantee coherence, the number of elements in $D_{ISO}$ and $PC$ must be the same. Hence the vectors $Dc$ relative to the outliers are removed from $PC$.

3.3. Synthesis engine parameters retrieval
For both approaches, after the generation of the vector $D_{ISO}$ (or $D_{PC}$) we search the nearest neighbour vector in the matrix $D_{ISO}$ (or $D_{PC}$). Through column indexing we retrieve from $PC$ the vector $p$ used to drive the synthesis engine instantaneously. This simple approach leads to potential discontinuity in the synthesis parameters generation, because different combinations of synthesis parameters that might be far apart in the control space may lead to identical or near points in the perceptually related feature space. To guarantee continuity we propose two solutions. In the first one we retrieve $K NN$ (nearest neighbours) in $D_{ISO}$ rather than one and drive the synthesis engine with the mean of the $K$ corresponding vectors $p$. In the second one, before searching for the nearest neighbor, we append $p$ to the $D_{ISO}$ (or $D_{PC}$) and we append the matrix $Pc$ in $D_{ISO}$ (or $D_{PC}$). The first solution shows a limitation when the $K p$ are very far apart, while the second can be debatable because perceptual features and synthesis parameters are merged in the same space, hence the search is performed in a heterogeneous space. However, these methods improve a shortcoming in [5] where occasionally the system gets trapped in local minima. As mentioned before, the sampling resolutions affect the size of $PC$ and $D$. The computational load required for the $K NN$ search is thus proportional to the size of the matrix and it affects the system minimum response time.

Figure 3: PCA (left) and ISOMAP (right) energy distribution across the reduced dimensions accounting for 90% of the total energy for the same dataset (note the different y axis scale).

4. PROTOTYPE AND APPLICATION
A prototype has been developed and implemented in Max/MSP and MATLAB. The prototype uses the FTM [9] and MaxM [10] toolbox for vector and matrix processing in Max/MSP. The perceptually related feature set is based on Tristan Jehan’s “analyzer~” (includes “tiddle~” by Miller Puckette) max external. The feature vector hence includes: loudness, pitch, brightness, noisiness, and the energies in the 25 Bark bands. Each feature can be enabled/disabled by the user and a weight vector can be defined as well to provide better customization. The adaptive approach is independent of the dimensionality and content of the feature vector, therefore a different selection is possible.

The prototype is integrated with Ableton Live using the Max For Live framework for the interfacing capabilities with state-of-the-art synthesis engines. Two Max For Live patches cooperate to analyze the generated sound. The front-end generates the $p$ set and drive the synthesizer with up to 8 parameters, and the back-end analyses the audio signal, stores $p$, and the relative multiple $d$, in the matrices $P$ and $D$. The post processing of $D$ described in Section 2, and the adaptive mapping described in Section 3, are computed within MATLAB using the author’s ISOMAP implementation. Another two Max For Live patches implement the runtime adaptive control for PCA and ISOMAP respectively, exposing up to 4 $PC/ISO$ mapped control parameters.

Through the prototype’s Max For Live patches it is possible to set and modify system settings allowing exploration of different configurations. In the analysis patches it is possible to set the sampling resolutions, the parameters range, the note and the timing (in terms of delays) of the automatic analysis. Moreover, the number of analysis windows and the hop size are flexible, while the window size is fixed at 4096 samples. The control patches allow further reduction of the dimensionality of the PCA projection and ISOMAP transformation, modification of the $K NN$ number, and the dimensionality of the control space $C$. The prototype allows also for inverting the polarity of every $PC$, or $ISO$, in order to flip the synthesis engine response.

4.1. Single Parameter Application
In this first application we chose a simple scenario to demonstrate the adaptation capability. The synthesis engine is the Ableton Live Operator synth, implementing a simple FM synthesis using just two oscillators. The only variable parameter is the cut-off frequency of the low pass filter. We run the analysis over the full range of the parameter and a reduced set of features, using the energy of Bark bands only. Figure 4 shows the analysis windows per state $p$, computed with a hop size of 2048 samples, using C2 as fixed note. For the mapping we use only the principal dimension from the PCA and ISOMAP methods in order to have a 1D comparison metric. Both provide an identical result in terms of adapted control: most of the energy is concentrated on the first component since there is high correlation in $D$. Figure 4 shows three-dimensional scatter plot of the first three $PC$, or $ISO$ (note the different axis ranges), where is possible to appreciate the capability of ISOMAP to detect the manifold and organize the data points. Figure 5 shows a line $PC2$ (top right), $ISO1$ (bottom right). How the adapted control provides a linear response over the feature with the greatest variance, while the control signal applied directly to the cut-off frequency presents a non-linear response and a range with almost no effect over the generated sound.

Figure 4: 3D scatters of the lower dimensional perceptually related features after PCA (left) and ISOMAP (right).

4.2. Two Parameters Application
In a second example, we run the analysis computing the complete features set on a preset of the Ableton Live Analog synth, modifying the two “oscillator detune” parameters with a coarse sampling resolution. Ten analysis windows per state $p$ are computed with a hop size of 1024 samples, using C3 as fixed note. Figure 6 shows how the two principal PCA and ISOMAP projected perceptual features are very noisy over the control parameter space, but in Figure 7 it is evident that there are linear and stable due to the adaptive control. The wider range obtained with the ISOMAP is due to its capacity to embed energy in a lower number of dimensions.
4.3. Partikkel Hadron Application

In the last example we use the Partikkel Hadron1 granular synthesizer with one of the provided pre-set. Through granular synthesis it is possible to obtain large timbre variation due to the nature of the synthesis, but often the control parameter set is large and challenging to design an interface for. This device exposes just 6 parameters for timbre manipulation thanks to the exploitation of the Modulation Matrix [7]. We analyze the generated audio with 25 Bark bands energies over the whole control space given by all possible combinations of the 6 parameters. Sixteen analysis windows per state p are computed with a hope size of 512 samples, using C2 as fixed note. In this more complex scenario the performance of ISOMAP is sensibly better than PCA. The data presented in Figure 8 shows how ISOMAP, when compared with PCA, allows the reduction of at least one dimension in the control space C without losses in the overall descriptors space energy. With the ISOMAP adaptation, we obtain a further reduction of the control space. This enables the use of a simple 2D controller, while still permitting the navigation of the majority of the granular sonic space spanned by the original 6 parameters.

![Figure 8: PCA (left) and ISOMAP (right) energy distribution across the reduced dimensions for the same Hadron granular synthesizer dataset.](image)

5. CONCLUSION AND FUTURE WORK

We presented a generic method to adapt general-purpose interfaces to synthesis engines through unsupervised dimensionality reduction techniques and statistical analysis of the perceptually related features computed over the synthetic sound. The application of the prototype demonstrates the benefits introduced by this adaptive technique, including the linearization of the relationship controller-to-sound and the dimensionality reduction of the control space. However some aspects can be further explored for improvements.

The exploitation of dynamic features in synthetic timbres must be explored more extensively. The computation of the dynamic aspect of the timbre has been tested, but embedding static and dynamic features in the same vector d may not be appropriate for all cases. Storing this information in two separate matrices and running dimensionality reduction separately on each may result in an adaptive mapping that is easier to use. The current MATLAB implementation of the ISOMAP algorithm is computationally expensive in terms of time and memory, thus we had to limit the dimensionality of D and P to 4000. This is too small to handle large numbers of parameters sampled with a high resolution. This limitation of the resolution is reflected in the usability experience. An optimization of the algorithm implementation is thus desirable.

In Section 3 we make some assumptions about the control interface output signals. These are generally true for most of the commercial general-purpose interfaces (e.g. sets of sliders, knobs, touch surfaces, touch screen devices). For other interfaces built with large numbers of sensors, or devices capturing human gesture through image or sound, the assumptions may not hold. Through a statistical study of the interface signals it should be possible to apply a pre-processing stage that produces independent components within the desired range.

6. REFERENCES


---

1 http://www.partikkelaudio.com/

---

A MICROPHONE ARRAY INTERFACE FOR REAL-TIME INTERACTIVE MUSIC PERFORMANCE

Daniele Salvati
AViRES lab
Dep. of Mathematics and Computer Science, University of Udine, Italy
daniele.salvati@uniud.it

Serugo Canazza
Sound and Music Computing Group
Dep. of Information Engineering, University of Padova, Italy
canazza@dei.unipd.it

Gian Luca Foresti
AViRES lab
Dep. of Mathematics and Computer Science, University of Udine, Italy
gianluca.foresti@uniud.it

ABSTRACT

A novel digital musical interface based on sound source localization using a microphone array is presented. It allows a performer to plan and conduct the expressivity of a performance by controlling an audio processing module in real-time through the spatial movement of a sound source. Musical interfaces are often used to allow novice users to enhance the expressive control over the sounds generated by their acoustic instruments in a live electronics context. E.g., in the works by Adriano Guarnieri – Medico (2002) and Fili bianco-velati (2005) – produced at the “Centro di Sonologia Computazionale” of Padova, the movement of a musician is followed by a motion capture system based on infrared cameras to control a live electronics patch [4], and using the robust very expensive, PhaseSpace optical motion capture system. It is composed by led systems, video cameras, and calibration procedure. In general, those kind of systems have considerable complexity and in some situations there could be problems with the low and/or not always controllable lighting of the concert hall, even when using infrared camera. It has been shown in [14] that there is some potentiality in using the source sound localization to directly control the position of a sound played back through a spatialization system by moving the sound produced by its own musical instrument. This work has been improved in [13] introducing an adaptive parameterized Generalized Cross-Correlation (GCC) PHAT filter to localize musical sounds that are mainly harmonics. Both interfaces [14] [13] are been tested in a controlled real environment without verifying if the system works with interfering sources from a sound reinforcement system and other instruments. Thus, in this paper a validation in multi-source scenario is presented, introducing the adaptive parameterized SRP-PHAT with a ZCR threshold (Section 3) that has a better performance than the parameterized GCC-PHAT proposed in [13] as shown in Section 4.
2. SYSTEM ARCHITECTURE

The proposed interface has the advantage of being completely non-invasive (no need for markers, sensors or wires on the performance), and requires no dedicated hardware. The architecture consists of a microphone array and digital signal processing algorithms for robust sound localization. Figure 1 summarizes the system architecture of the interface. The array system is composed of three half-supercardioid polar pattern microphones, which reduce ambient noise and pickup of room reverberation, arranged in an uniform linear placement. In this way, we can localize a sound source in a plane (three microphones are the bare minimum). Signal processing algorithms estimate the sound source position in an horizontal plane by providing its Cartesian coordinates. A SRP-PHAT method is used to compute the acoustic map analysis. To improve the performance in the case of harmonic sounds, or generally pseudo-periodic sounds, a parameterized SRP-PHAT is proposed. The ZCR function is used to determine if a sound is pseudo-periodic, and to adapt with a threshold value the parametrically control of PHAT filter. A Kalman filter [8] is applied to smooth the time series of observed position to obtain a more robust and accurate xy values. The last component implements the mapping strategy [16] to associate the x-y coordinates with audio processing parameters.

3. ADAPTIVE PARAMETERIZED SRP-PHAT WITH ZERO-CROSSING RATE THRESHOLD

The SRP-PHAT [5] is based on the concept of adding several time delay estimation functions from the microphone pairs. It consists of calculating the GCC-PHAT function between pairs of microphones and using the Global Coherence Field (GCF) [11] to construct a spatial analysis between pairs of microphones and using the Global Coherence Field (GCF) 

\[ \text{GCC}(t) = \sum_{f} \left| \text{STFT}(x_1(t) \cdot x_2^*(t)) \right|^2 \]

where the GCC(t) is the GCC-PHAT of the rth pair. The position of the source is estimated by picking the maximum peak

\[ \hat{s}_m = \text{argmax}_s \langle s, f \rangle \left| \text{GCC}(t) \right| \]

The GCC-PHAT [9] is the classic method to estimate the relative time delay associated with acoustic signals received by a pair of microphones in a moderately reverberant and noisy environment. The GCC-PHAT basically consists of a cross-correlation followed by a filter that aims to reduce the performance degradation caused by additive noise and multi-path channel effects. The GCC-PHAT in the frequency domain is

\[ \hat{S}_{g_{cc}}(f) = \frac{1}{L} \sum_{t=1}^{L} \Psi(f) S_{x_1 x_2}(f)e^{j2\pi ft} \]

where \( f \) is the Discrete Fourier Transform (DFT) integer frequency index, \( L \) is the number of samples of the observation time, and \( \Psi(f) \) is the frequency domain PHAT weighting function, and the cross-spectrum of the two signals is defined as

\[ S_{x_1 x_2}(f) = E[x_1(t) \cdot x_2^*(t)] \]

where \( X_1(f) \) and \( X_2(f) \) are the DFT of the signals \( x_1(t) \) and \( x_2(t) \) respectively, and \( \ast \) denotes the complex conjugate. GCC is used for minimizing the influence of moderate uncorrelated noise and moderate multipath interference, maximizing the peak in correspondence of the time delay. The PHAT weighting function places equal importance on each frequency by dividing the spectrum by its magnitude. The PHAT normalizes the amplitude of the spectral density of the two signals and uses only the phase information to compute the GCC

\[ \Psi_{\text{PHAT}}(f) = \frac{1}{\left| \hat{S}_{g_{cc}}(f) \right|} \]

We note that SRP-PHAT, which uses the sum of the GCCs of the microphone pairs, is equivalent to using a steered response filter and sum beamforming with PHAT weighting. In fact, the SRP of a 2-element array is equivalent to the GCC of those two microphones.

The SRP-PHAT algorithm has been shown to be one of the most robust sound source localization approaches operating in noisy and reverberant environments [15]. This algorithm enhances the performance of localization with a network of large arrays. However, the computational cost of the method is very high. To reduce the processing time of search algorithms, improvements have been suggested [2][3].

It is important to note that the PHAT performance is dramatically reduced in the case of harmonic sounds, or generally pseudo-periodic sounds. In fact, the PHAT has less capability to reduce the deleterious effects of noise and reverberation when it is applied to a pseudo-periodic sound. An accurate analysis of the PHAT performance for a broadband and narrowband signal can be found in [6]. The results of this work highlight the ability of the PHAT to enhance the detection performance for single or multiple targets in noisy and reverberant environments, when the signal covers most of the spectral range. Thus, to work with pseudo-periodic sounds the proposal is to use a parameterized SRP-PHAT that weights the contribution of the PHAT filtering, depending on the threshold of the ZCR parameters. The PHAT weighting can be generalized to parametrically control the level of influence from the magnitude spectrum [6]. This transformation will be referred to as the PHAT-\( \beta \) and defined as

\[ \Psi_{\text{PHAT}-\beta}(f) = \frac{\beta}{\left| \hat{S}_{g_{cc}}(f) \right|} \]

where \( \beta \) varies between 0 and 1. When \( \beta = 1 \), equation (6) becomes the conventional PHAT and the modulus of the frequency transform becomes 1 for all frequencies; when \( \beta = 0 \), the PHAT has no effect on the original signal, and we have the cross-correlation function. Therefore, in the case of harmonic sounds, we can use an intermediate value of \( \beta \) so that we can detect the peak to estimate the time delay between signals, and can have a system, at least in part, that exploits the benefits of PHAT filtering to improve performance in moderately reverberant and noisy environments. The results of localization improvement in case of pseudo-periodic sounds using the PHAT-\( \beta \) are reported in [13]. To adapt the value of \( \beta \), we use the ZCR to determine if the sound source is periodic. ZCR is a very useful audio feature and is defined as the number of times that the audio waveform crosses the zero axis

\[ \text{ZCR}(t) = \sum_{i=1}^{N} \text{sgn}(x(t + i)) - \text{sgn}(x(t + i - 1)) \]

where \( \text{sgn}(x) \) is the sign function.

4. EXPERIMENTAL RESULTS

The experiments to verify the performance of interface in a multi-source scenario were conducted in a rectangular room of 3.8 x 4.4 m, in a moderately reverberant (RT60=0.35 s) and noisy environment. The experiments were made with a real-time interface developing a Max external objects for interactive performance.

The interface works with sampling rate of 96 KHz, a Ham window analysis of 42 ms, and a distance between microphones of 25 cm. The sampling rate and the microphone distance determine the number of samples to estimate the TDOA. In fact, by increasing these parameters it is possible to obtain a higher resolution on the TDOA, and therefore a higher resolution in the sample plane position. Figure 2 compares the sample space position with different sampling rate, considering a square with 100 cm sides, three microphones with distance of 25 cm, and the TDOAs between microphones \( m_1 m_2 \) and \( m_2 m_3 \).

The working area is located in a square with 1 meter sides. The axis origin coincides with the position of microphone 2, the x axis can vary between -50 cm and 50 cm, and the y axis can vary between 0 and 100 cm. It is important to note that when we use a SRP-PHAT in multisource case, we have to consider a largest area of analysis, otherwise competing sources can cause a peak that would be seen by the system as a false source in SRP-PHAT map inside the one meter square area. Therefore, for the experiments we consider a square area of analysis with 2 meter
Figure 4. The x-y position estimation by the system of a flute sound with SIR=35 dB. The Kalman filter data is the black line and raw data are the dots. a) GCC-PHAT-β b) SRP-PHAT-β

Table 1. The values of RMSE(y), maximum and minimum of x related to the experiments.

<table>
<thead>
<tr>
<th>SIR</th>
<th>RMSE(y)</th>
<th>Max(x)</th>
<th>Min(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 dB</td>
<td>2.5</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>20 dB</td>
<td>4.5</td>
<td>80</td>
<td>0</td>
</tr>
<tr>
<td>30 dB</td>
<td>6.5</td>
<td>120</td>
<td>0</td>
</tr>
<tr>
<td>40 dB</td>
<td>8.5</td>
<td>160</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 5. The x-y Kalman position estimation by the system with different SIR. a) GCC-PHAT-β b) SRP-PHAT-β

Besides, it is shown how the angular resolution degrades the localization performance, in particular for the GCC-PHAT-β, in correspondence with the increase of competitive sources; locate is impossible when the SIR is less than 20 dB. An increase in the number of microphones could improve the system performance, especially with an algorithm based on SRP, which tends to have a more robust performance with a large array.

The interface has considerably less complexity than systems based on electric field, optical and video camera sensors. The latter are widely used, especially for great success of Kinect, but in general it is possible to have problems for the low or variable lighting condition during live electronics. One of the most robust way is to use a motion capture system (e.g., PhaseSpace), which is very complex and expensive. Hence, we believe that our approach based on microphone array is presented as a viable alternative. However, in the future, we plan to use and test the microphone array interface in live electronic performances.

6. REFERENCES


STATISTICAL PARSING FOR HARMONIC ANALYSIS OF JAZZ CHORD SEQUENCES

Mark Granroth-Wilding, Mark Steedman
School of Informatics
University of Edinburgh, Edinburgh, UK
mark.granroth-wilding@ed.ac.uk, steedman@inf.ed.ac.uk

ABSTRACT

Analysing music resembles natural language parsing in requiring the derivation of structure from an unstructured and highly ambiguous sequence of elements, whether they are notes or words. Such analysis is fundamental to many music processing tasks, such as key identification and score transcription.

The focus of the present work is on harmonic analysis. We use the three-dimensional tonal harmonic space developed by [4, 13, 14, 15] to define a theory of tonal harmonic progression, which plays a role analogous to semantics in language. Our parser applies techniques from natural language processing (NLP) to the problem of analysing harmonic progression. It uses a formal grammar of jazz chord sequences of a kind that is widely used for NLP, together with the statistically based modelling techniques standardly used in wide-coverage parsing, to map music onto underlying harmonic progressions in the tonal space.

Using supervised learning over a small corpus of jazz chord sequences annotated with harmonic analyses, we show that grammar-based musical parsing using simple statistical parsing models is more accurate than a baseline model that also produces an analysis in the tonal space.

We focus here on harmonic analysis. We use a three-dimensional tonal harmonic space ([4, 13, 14, 15]). This representation provides the basis for a theory of tonal harmonic progression. The framework allows us to analyse the relationships between the chords underlying a passage of music and the relationship of the notes to their underlying chords. We treat the analysis of the tonal relations between chords analogous to the logical semantics of a sentence. By defining a representation of movements in the tonal space in a form similar to logical representations of natural language semantics, we are able to apply techniques from NLP directly to the problem of harmonic analysis.

We use a formal grammar of jazz chord sequences in a formalism based closely on one used for NLP. We then use modelling techniques commonly applied to the task of statistical parsing of natural language sentences with such grammars to map music, in the form of chord sequences, onto its underlying harmonic progressions in the tonal space. In the present paper, we omit the details of the representation of tonal space movements as formalized in the grammar (the semantics), and the syntactic component of the grammar. Instead we introduce the structures that the grammar is designed to analyse and focus on the statistical parsing techniques and their performance on the harmonic analysis task.

We use supervised learning over a small corpus of chord sequences (76 songs, ~3k chords) of jazz standards from lead sheets used by performers, annotated by hand with harmonic analyses that we treat as a gold standard. We describe some experiments comparing the use of grammar-based musical parsing models from NLP to a baseline Markovian model that also produces an analysis in the tonal space. We show that the grammar-based model performs better than the baseline model at producing a tonal space analysis matching the hand-annotated gold standard.

We presented a small, context-free syntactic grammar of jazz chord sequences designed to capture twelve-bar blues chord sequences. [19] further developed the blues grammar, using a syntactic formalism and language of harmonic analysis that form the basis for those used in the present work to construct a wider-coverage grammar of tonal jazz chord sequences. The present paper uses statistical models to apply the grammar to an analysis task. [5] and [16] have proposed a syntactic model of harmonic structure closely related to that we employ. The experiments present provide further support for structured approaches to musical analysis and the use of techniques adapted from NLP.

2. MUSICAL SYNTAX

The syntactic of tonal harmony and that of natural language can both be analysed using tree structures, and both have been claimed to feature formally unbounded embedding of structural elements ([10, 12, 18, 16]).

2.1. Cadences

Cadences are notes or words that form the basis of a sentence. By defining a representation of movements in the tonal space in a form similar to logical representations of natural language semantics, we are able to apply techniques from NLP directly to the problem of harmonic analysis.

We use a formal grammar of jazz chord sequences in a formalism based closely on one used for NLP. We then use modelling techniques commonly applied to the task of statistical parsing of natural language sentences with such grammars to map music, in the form of chord sequences, onto its underlying harmonic progressions in the tonal space. In the present paper, we omit the details of the representation of tonal space movements as formalized in the grammar (the semantics), and the syntactic component of the grammar. Instead we introduce the structures that the grammar is designed to analyse and focus on the statistical parsing techniques and their performance on the harmonic analysis task.

We use supervised learning over a small corpus of chord sequences (76 songs, ~3k chords) of jazz standards from lead sheets used by performers, annotated by hand with harmonic analyses that we treat as a gold standard. We describe some experiments comparing the use of grammar-based musical parsing models from NLP to a baseline Markovian model that also produces an analysis in the tonal space. We show that the grammar-based model performs better than the baseline model at producing a tonal space analysis matching the hand-annotated gold standard.

[18] presented a small, context-free syntactic grammar of jazz chord sequences designed to capture twelve-bar blues chord sequences. [19] further developed the blues grammar, using a syntactic formalism and language of harmonic analysis that form the basis for those used in the present work to construct a wider-coverage grammar of tonal jazz chord sequences. The present paper uses statistical models to apply the grammar to an analysis task. [5] and [16] have proposed a syntactic model of harmonic structure closely related to that we employ. The experiments present provide further support for structured approaches to musical analysis and the use of techniques adapted from NLP.

2.2. The Jazz Sublanguage

We term this operation coordination by virtue of its similarity to right-node raising coordination in natural language. For example, in Keats bought and will eat beans, bees satisfies the expectations of both bought and will eat. Coordinated cadences may themselves be embedded in coordinated cadences, as in the example taken from Call Me Irresponsible shown in figure 2. Once again, a similar form of embedding occurs in natural language examples like Keats (certainly eats) but (may or may not) cook beans.

Dominant function chords are often partially, though never unambiguously, distinguished by the addition of notes outside the basic triad. In particular, the dominant seventh, realized by the note two semitones below the octave, enhances the cadential function of a dominant chord and heightens expectation of the resolution. However, a dominant may omit this note and the same note (or rather, one indistinguishable from it in equal temperament) may even appear in chords not functioning as dominants.

The typical size and complexity of the cadence structures discussed above varies with musical period and genre. Tonal jazz standards are of particular interest for this form
of analysis for several reasons.

First, they tend to feature large extended cadences, often with complex embedding. Second, they contain many well-known counterfactuals, harmonic variations of a familiar chord, created using a well-established system of harmonic substitutions, embellishments and simplifications.

Finally, jazz standards are rarely transcribed as full scores, but are more analytically notated as a melody with accompanying chord sequence. Analysing the harmonic structures underlying chord sequences, rather than streams of notes, avoids some difficult practical issues such as voice leading and performance styles, but still permits discovery of the kind of higher-level structures we are concerned with. They therefore provide a convenient starting point for our investigation.

Our study focusses on the analysis of harmonic structure in chord sequences of jazz standards. This is not to say that the approach to analysis is not applicable beyond this domain or even that it depends on analysing chord sequences. Our grammar’s lexicon, introduced in brief in section 4, is specific to the genre, though a lexicon suitable for another tonal harmonic genre would have much in common.

3. A MODEL OF TONALITY

In analysing the roles of pitch in music, it is important to distinguish between consonance, the sweetness or harshness of one sound that results from playing two or more notes at the same time, and harmony, which is the dimension relevant to the phenomenon that we have already alluded to as tension (and the creation of expectation) and resolution (or its satisfaction). The intervals over pitches are determined by small whole-number ratios, and are often confounded. However, they arise in quite different ways.

3.1. Consonance

The modern understanding of consonance originates with Helmholtz ([6]), who explained the phenomenon in terms of the coincidence and proximity of the secondary overtones and difference tones that arise when simultaneously sounded notes excite real non-linear physical resonators, including the human ear itself, inducing harmonics or secondary tones. To the extent that any of its secondaries are separated in frequency by a small enough difference to beat at a certain rate, it is perceived as consonant, or sweet-sounding. To the extent that any of its secondaries are exactly coincident, it is perceived as dissonant, or harsh.

The modern understanding of consonance and dissonance in chords, and the effects of chord inversion. We ignore the issue of consonance, unlike [9, 11], and are interested instead in the somewhat orthogonal issue of harmony.

3.2. Harmony

The harmonic system also derives from combinations of small integer pitch ratios. However, the harmonic relation is solely on the first three prime ratios in the harmonic series: ratios of 2, 3 and 5 (the octave, perfect fifth and major third). The tuning based on these intervals is known as just intonation.

3.2.1. Just Intonation

In just intonation, an interval can be represented as aFrequency ratio defined as the product 2^a 3^b 5^c, where a, b, c are positive or negative integers. It has been observed since [4] that the harmonic relation can therefore be visualized as an infinitely extending discrete three-dimensional space with these three prime factors as generators. Since notes separated by octaves are essentially equivalent for tonal purposes, it is convenient to project the space onto the 3, 5 plane. We adopt this form in the two-dimensional projection.

[15] observed that all diatonic scales are convex sets of positions, and defined a Manhattan distance metric over this space. According to this metric, it will be observed that the major and minor triads, such as CEG and CE#G, when plotted in this space are two of the closest possible clusters of three notes. The triad with added major seventh is the single tight cluster of four notes. The triads and the major seventh chord are stable unambiguous chords that raise no strong expectations and are of the kind that typically end a piece. Chords like the diminished chord and the dominant seventh are more spread out, a difference vital to the induction of harmonic expectation and its satisfaction.

3.2.2 Equal Temperament

Over several centuries, an approximation of the tonal harmonic space was gradually adopted, first by slightly musing the fifths to equate all the positions with the same label in figure 3, and then by further distorting the major thirds, to equate enharmonic equivalents (C with B#, D#, etc.). The 12 tones of the diatonic octave are spaced evenly, so that all the semitones are (mis)tuned to the same ratio of √2.

This system of equal temperament has the advantage that all keys and modes can be played on the same instrument without retuning. In the tonal space, the result is a distortion of the pitches so that the infinite space is projected onto a finite space of just 12 points, looping in both dimensions. Each point is (potentially infinitely) tonally ambiguous as to which point in the infinite justly intoned space of figure 3 it denotes. Thus, equal temperament makes the interpretation of tonal relations ambiguous. Its advantage, however, is that it allows the hearer to resolve this tonal ambiguity.

It is important to realize that ambiguous equally tempered music is unconsciously interpreted in terms of the full tonal space of harmonic distinctions, just as a theoretically infinitely ambiguous two-dimensional photograph is interpreted as a three-dimensional scene. We perform this disambiguation explicitly in our analyses by mapping equal-temperament chord sequences onto paths through the justly intoned tonal space.

3.3. Domain for Analysis

In our grammar for jazz chord sequences, we take the full two-dimensional tonal space as the semantic domain of harmonic analysis. A harmonic interpretation of a piece is the path through the tonal space traced by the roots of the chords.

If we establish that there is a dominant-tonic relationship between two chords, we know that the underlying relationship between the roots is a perfect fifth, a single step to the left in the space. Likewise, a subdominant-tonic relationship dictates a perfect fourth, a rightward step. Where no tension-resolution relationship exists, as when a chord is assigned to a chord, it assumes the chord’s root and thenceforth applies its constraints relative to that root.

For example, a schema Dom is used to interpret a chord as having a dominant function, including recursive dominant sevenths, as described above. It constrains its subsequent resolution to be rooted a perfect fifth below it. Its semantics represents a levend step in the tonal space. Another schema Ton interprets a chord as a tonic chord and may serve as the resolution to a Dom category. Further categories are included to handle subdominant chords, substitutions (such as the tritone substitution), and passing chords, and so on.

Figure 5 shows a full CCG derivation of the cadence from Call Me Irresponsible, the structure of whose harmonic semantics was shown in figure 2, this time with a full theonic resolution (as it was not reached until another, similar cadence structure.) We do not describe the grammar’s lexicon and combina-

Figure 3. Part of the space of note-names (adapted from [13, 14])

This theory successfully explains the experience of consonance and dissonance in chords, and the effects of chord inversion. We ignore the issue of consonance, unlike [9, 11], and are interested instead in the somewhat orthogonal issue of harmony.

Figure 4. A tonal space path for the extended cadence: C7 D7 Gm7 C.

4. A GRAMMAR FOR JAZZ

Combinatory Categorial Grammar (CCG) is a lexicalized grammar formalism. A CCG lexicon contains combinator categories that are assigned to the words of a sentence which specify constraints on the structures in which the word’s semantics may combine with that of surrounding words. Once a category has been chosen for each word, a small set of combinators may be used to produce a semantics for the whole sentence from that of the individual words. An adaptation of CCG to the parsing of harmony was introduced by [19]. We use here a further development of that formalism and introduce a statistical parsing model and an implementation that were missing there.

We have hand-crafted a lexicon containing categories suitable for assigning harmonic interpretations to chords. Our musical CCG grammar contains several combinators, similar to those used for parsing natural language. Each item in the lexicon is a schema that generalizes over chord roots. When it is assigned to a chord, it assumes the chord’s root and thenceforth applies its constraints relative to that root.

Figure 4. A tonal space path for the extended cadence: C7 D7 Gm7 C.

4. A GRAMMAR FOR JAZZ

Combinatory Categorial Grammar (CCG) is a lexicalized grammar formalism. A CCG lexicon contains categories that are assigned to the words of a sentence which specify constraints on the structures in which the word’s semantics may combine with that of surrounding words. Once a category has been chosen for each word, a small set of combinators may be used to produce a semantics for the whole sentence from that of the individual words. An adaptation of CCG to the parsing of harmony was introduced by [19]. We use here a further development of that formalism and introduce a statistical parsing model and an implementation that were missing there.

We have hand-crafted a lexicon containing categories suitable for assigning harmonic interpretations to chords. Our musical CCG grammar contains several combinators, similar to those used for parsing natural language. Each item in the lexicon is a schema that generalizes over chord roots. When it is assigned to a chord, it assumes the chord’s root and thenceforth applies its constraints relative to that root.

Figure 4. A tonal space path for the extended cadence: C7 D7 Gm7 C.

4. A GRAMMAR FOR JAZZ

Combinatory Categorial Grammar (CCG) is a lexicalized grammar formalism. A CCG lexicon contains categories that are assigned to the words of a sentence which specify constraints on the structures in which the word’s semantics may combine with that of surrounding words. Once a category has been chosen for each word, a small set of combinators may be used to produce a semantics for the whole sentence from that of the individual words. An adaptation of CCG to the parsing of harmony was introduced by [19]. We use here a further development of that formalism and introduce a statistical parsing model and an implementation that were missing there.

We have hand-crafted a lexicon containing categories suitable for assigning harmonic interpretations to chords. Our musical CCG grammar contains several combinators, similar to those used for parsing natural language. Each item in the lexicon is a schema that generalizes over chord roots. When it is assigned to a chord, it assumes the chord’s root and thenceforth applies its constraints relative to that root.

Figure 4. A tonal space path for the extended cadence: C7 D7 Gm7 C.

4. A GRAMMAR FOR JAZZ

Combinatory Categorial Grammar (CCG) is a lexicalized grammar formalism. A CCG lexicon contains categories that are assigned to the words of a sentence which specify constraints on the structures in which the word’s semantics may combine with that of surrounding words. Once a category has been chosen for each word, a small set of combinators may be used to produce a semantics for the whole sentence from that of the individual words. An adaptation of CCG to the parsing of harmony was introduced by [19]. We use here a further development of that formalism and introduce a statistical parsing model and an implementation that were missing there.

We have hand-crafted a lexicon containing categories suitable for assigning harmonic interpretations to chords. Our musical CCG grammar contains several combinators, similar to those used for parsing natural language. Each item in the lexicon is a schema that generalizes over chord roots. When it is assigned to a chord, it assumes the chord’s root and thenceforth applies its constraints relative to that root.

Figure 4. A tonal space path for the extended cadence: C7 D7 Gm7 C.

4. A GRAMMAR FOR JAZZ

Combinatory Categorial Grammar (CCG) is a lexicalized grammar formalism. A CCG lexicon contains categories that are assigned to the words of a sentence which specify constraints on the structures in which the word’s semantics may combine with that of surrounding words. Once a category has been chosen for each word, a small set of combinators may be used to produce a semantics for the whole sentence from that of the individual words. An adaptation of CCG to the parsing of harmony was introduced by [19]. We use here a further development of that formalism and introduce a statistical parsing model and an implementation that were missing there.

We have hand-crafted a lexicon containing categories suitable for assigning harmonic interpretations to chords. Our musical CCG grammar contains several combinators, similar to those used for parsing natural language. Each item in the lexicon is a schema that generalizes over chord roots. When it is assigned to a chord, it assumes the chord’s root and thenceforth applies its constraints relative to that root.

Figure 4. A tonal space path for the extended cadence: C7 D7 Gm7 C.
5.1. Jazz Corpus

tors any further here, but include this derivation merely to give a flavour of how an interpretation is produced from lexical categories.

The lexicon is deliberately specific to the genre we wish to analyse. Another lexicon could be constructed with which to interpret the harmonic relations of another tonal harmonic genre and would have a number of categories in common with our European baroque music, for example, would not use all of the substitution categories included in the jazz lexicon and would require some additional categories to reflect different connotational expressions of the perfect cadence.

5. STATISTICAL PARSE MODELS

Just as with natural language parsing, the lexical ambiguity of interpretation of chord sequences prohibits exhaustsive parsing to deliver every syntactically well-formed interpretation of a sequence. Moreover, we require some additional categories to reflect different connotational expressions of the perfect cadence.

5.2. Adaptive Supertagging

Supertagging is a technique, related to part of speech (POS) tagging, useful as a first step in parsing with lexicalized grammars like CCG (117). Probabilistic sequence models, using statistics about short windows of sequences, are employed to choose CCG categories from the lexicon for each word. In music, as in natural language, the choice of a category representing a plausible interpretation of a chord depends on the analysis of potentially distant parts of the sequence (long-distance dependencies). In practice, short-distance statistics can often reliably rule out at least the most improbable interpretations.

A bad choice of categories could make it impossible to parse the sequence. The adaptive supertagging algorithm (13) allows categories considered less probable by the supertagger to be used in such cases. First, the supertagger assigns to each word (or chord) a small set of what its model dictates are the most probable categories and the parser attempts to find a full parse with these categories. If it fails, the supertagger supplies some more, slightly less probable categories and the parser tries again. This is repeated until the parser succeeds or we give up (for example, after a set number of iterations). If multiple full parses are found in one iteration, the single most probable one is chosen.

Many types of probabilistic sequence model can be used as a supertagger. One is a hidden Markov model (HMM) with states representing categories. The state emissions of the model are not the chords themselves, but a pair of the chord type and the interval between this chord’s and the previous chord’s root. This has the effect of making the model account only for relative pitch. We trained the model by maximum likelihood estimation over the annotated categories from the corpus described above.

We performed some initial experiments with higher-order Markov models (n-gram models) which suggested that they do not perform any better than the HMM we use here when trained on this small corpus. We expect that the model would benefit from the use of higher-order statistics given a larger training set.

5.3. Parsing Models

[7] adapted the generative probabilistic parsing models of probabilistic context-free grammars (PCFG) to CCG. Using a corpus of gold-standard parsed sentences, probabilities are estimated for expansions at internal nodes in the derivation tree. These probabilities are used to estimate a probability for every subtree produced during the derivation.

In our experiments, we use a model like that of [7] to parse chord sequences, which we refer to as PCCG. During parsing, the model is used to assign a probability to internal nodes in the derivation: that is, every combination of categories by a combinator. A beam is applied to internal nodes: all but the most probable derivations, according to the parsing model’s probabilities, are removed. A second model uses the supertagger with the adaptive supertagging algorithm described above to narrow down the choice of lexical categories available to the parser. The parser then proceeds just as in PCCG. We call this model ST+PCCG.

Using both models, we allow the parser a fixed amount of time to parse a particular sequence before giving up, chosen such that most parses complete within the time.

5.4. Baseline Model

In an attempt to quantify the contribution made by restricting interpretations to those that are syntactically well formed under the jazz grammar, we have constructed a model which assigns tonal space interpretations without using the grammar. We use an HMM very similar to that described above as a supertagger model, which directly assigns a tonal space point to each chord, instead of assigning categories to chords and parsing to derive a tonal space path. The representation of the chord sequence is identified by the path of the supertagger’s.

We can define a naive, deterministic procedure to construct a tonal space interpretation for a chord sequence as follows: for each chord, choose from the infinite set of points mapped by equal temperament to the chord’s root the point that is closest to the previous point on the path. The states of the model are constructed to represent deviations from this naive path.

There are two reasons why such deviations from the naive path are required. First, valid analyses must be so because the resolution of this G7 and that at the end of the second cadence are constrained to be the same.

The naive procedure identifies the correct tonal space point for most cases and deviates usually small. The HMM’s state labels are of the form (Tsub, Xsub). The pair (Tsub, Xsub) identifies the relationship between the equal-temperament projection of the chord root in the analysis and model representation, thus modelling chord substitution. (0, 0) is most common; the tritone substitution would result in (2, 1). (Tsub, Xsub) accounts for cases where the tonal relation between this and the last root is not clearly measured by a distance. It represents the distance from this initial estimate to the root in terms of a number of horizontal and vertical cycles of the equal temperament 4 ± 3 space. The states of the HMM include those tonal relations observed in the training data.

The model is trained in the same way as the supertagger, only this time the training data is chord sequences paired with their annotated tonal space paths. We refer to this model as HMMPATH.

Unlike the supertagger, this model’s results are not filtered by the parser for grammaticality. PCCG and ST+PCCG will completely fail to assign a path in cases where a full parse cannot be found. This may be because the beam removes all derivations that permit a grammatical interpretation of the full sequence, or, in the case of ST+PCCG, because the supertagger fails to suggest a set of lexical categories from which a full interpretation can be derived. HMMPATH will assign some path to any se-
The evaluation of the tonal space path is performed in every case only on the path returned by the model with highest probability.

7. RESULTS

The results of the four experiments are reported in table 1. Although PCCG has the full set of lexical categories available to it, its results are all lower than ST+PCCG. This is because we needed to apply a more aggressive beam during parsing in order to handle the wider choice of interpretations at the lexical level. It seems, then, that the supertagger is a necessity for practical parsing and is doing a good job of cutting down the parser’s search space. ST+PCCG produces high precision results, because, unlike HMM PATH, it can only produce results that are permitted by the grammar and fails when it can find no such result. As we would expect, the addition of the backoff reduces the model’s precision, but improves its recall. Since ST+PCCG rarely fails to produce a result on this dataset, the backoff has little impact on the overall result. However, ST+PCCG+HMM PATH is robust in that it is guaranteed always to produce some result.

As described in section 5.1, we included in our corpus only sequences to which it is possible to assign a valid harmonic interpretation using our grammar. The results we report here for the models that use the grammar are therefore higher than we would expect if applying the technique to chord sequences sighted in the wild. In this case, we would expect the benefit of the backoff to become clearer, since the PCCG model would more often fail to find an analysis.

We draw two key conclusions from the results. First, they show that HMM PATH is a reasonable model to back off to when no grammatical result can be found. Second, they show that the use of a grammar to constrain the paths predicted by an HMM supertagger substantially improves over the purely-algorithmic information captured by a pure HMM-based model.

8. CONCLUSION

We have described a parser that uses a formal grammar of a kind employed in NLP, and statistically based modelling techniques of a kind standardly used in wide-coverage natural language parsers, to map music onto their harmonic interpretation, represented as harmonic progressions in the two-dimensional tonal space. The jazz harmony corpus we used to train our models is small, but experience with CCG parsing for NLP shows that these chord symbols: a human has divided the music into time segments of constant harmony, selected the most prominent notes, and narrowed down the range of possible chord roots somewhat. We intend to continue this work by constructing a model that incorporates these tasks into the analysis process, accepting note-level input (in MIDI form, for example) and suggesting possible interpretations in the way the supertagger component of our parsing model does.

9. REFERENCES


[7] J. Hockenmaier and M. Steedman, “Generative models for statistical parsing with Combinatory Cat-


A NEW APPROACH FOR CONSTRAINT PROGRAMMING IN MUSIC USING RELATION DOMAINS

Sascha Van Cauwelaert, Gustavo Gutiérrez, Peter Van Roy

Université catholique de Louvain
ICTEAM Institute, Louvain-la-Neuve, Belgium
sascha.vancauwelaert@gmail.com
gustavo.gutierrez@gmail.com
peter.vanroy@uclouvain.be

ABSTRACT

Constraint programming (CP) has been used for several decades in music composition and analysis. It has served as the underlying technology of different tools that allow composers to compute with musical abstractions (e.g., notes, scores). However, the traditional domains used in musical CP, namely finite domains (integers) and finite sets (integer sets), are not well suited to represent and express properties on structured information such as a score in a compact and efficient way. This paper introduces a new domain, the relation domain, where relations are a set of integer n-tuples. It proposes new constraints on relations and shows how to use them for musical composition. A single relation variable can represent a score of any size and the transformation between scores. The result is a system that directly supports computing with musical abstractions at a high abstraction level more pleasant to composers. The relation domain and its constraints are implemented using Binary Decision Diagrams and are provided as a library in the Gecode platform.

Keywords: Constraint programming, relation decision variables, music, computer-aided composition, binary decision diagrams, Gecode, OpenMusic, relational algebra.

1. INTRODUCTION

Constraint programming (CP) is a common approach to tackle problems in music ([5, 3]). To avoid musicians the effort of going to the low level of using integer and set decision variables directly, modeling frameworks such as Strasheela [4] have emerged. These frameworks provide musical abstractions for modeling the problems and then translate that model into an equivalent one that can be tackled by a constraint solver.

During the translation process the framework expresses the problem as a CSP. That is, in terms of decision variables, their possible values, and constraints on them. The efficiency of the solution process depends on the kind of variables and constraints supported by the solver, usually integer and integer set decision variables. This paper presents relation decision variables, a new domain in which the represented information consists of relations rather than just integers or sets. The proposed domain brings several advantages. First, it provides a new level of abstraction for modeling which can lead to succinct and more expressive models. Second, it offers a new set of constraints that can be used to express high level properties on the variables. And third, it provides a new solver that works directly at this high level, thus making it possible to achieve higher efficiency.

1.1. Motivating example for relation domains

Consider the problem of finding a result (also known as simple canon), with a leader voice V0 and one follower voice V1. V0 has to be played 0 onsets after V0. Additionally, when played together some given harmony rules must be respected. The value α is called the offset between the onsets of V0 and V1.

The solution of this problem is a score. From the problem, we need to consider at least the pitch and onset parameters for every note in it. We represent by Rs the resulting score. Rs is a binary relation (e.g. set of pairs) in which every element has the form (pitch, onset). The same representation is used for V0 and V1.

The offset is represented by a unary relation Offset. Any element in this relation will represent the offset performed by a voice. So, for the particular case of one follower voice, Offset contains only one element. For the harmony let us assume an input binary relation Consonant with elements of the form (pitch, pitch). This relation groups pitches that are accepted to be heard at the same time.

Expression (1) states the round property between the two voices and (2) ensures that both voices get in the score we need to find. Expression (3) links pitches of Rs that are played at the same onset. Those links are then forced to belong to PP. The harmony constraint (4) is ensured by stating that all the pitch links are consonant.

\[
\begin{align*}
(V_0 \times \text{Offset}) & \Rightarrow \text{Plus} \Rightarrow V_1 \\
(R_0 = V_0 \cup V_1) & \Rightarrow V_0 \subseteq PP \\
(R_0 \cap R_1 = PP) & \Rightarrow \text{Consonant}
\end{align*}
\]

The way of expressing the offset between the voices can look intimidating at first sight. By doing V0 × Offset we compute a ternary relation with elements of the form (pitch, onset, offset) that basically computes any note in V0 concatenated with any offset. By using the operation \( \cup \) (introduced further in Section 3) we compute new notes of the form \( \langle \text{pitch, onset} \rangle \) where \( \text{onset} \) is the new onset at which the note has to be played. Expression (1) contains the relation Plus. This is a ternary relation in which every element \((x, y, z)\) has the property \( x + y = z \). In order to link pitches played at a same offset, we use the operation \( \cap \) on \( R_0 \) with itself, but with permuted components. That is, we compose (see Section 3) tuples of the form \( \langle \text{pitch, onset} \rangle \) with tuples of the form \( \langle \text{onset, pitch} \rangle \).

Some strengths of using the introduced domain can be appreciated in this model. The constraints are stated in a high level way and over natural music concepts like scores. A score as a set of notes is represented directly as a single relation variable. The transformation used to represent each note is kept together as a tuple inside a score. To show these strengths clearly, we provide a second model of the same problem using integer and integer sets which are traditional domains in CP.

1.2. Traditional approach using integer domains

We now model the same problem using only decision variables over integers and sets. To model a score V we use two arrays \( V_0 \) and \( V_1 \), one that represents the pitches (5) and the other one the onsets (6). The arrays must be big enough to store all the notes in the largest possible score we consider. The notes themselves are referenced by the indices of the arrays. A note \( n \) in the voice V is then represented by its pitch \( \text{Pitch}(n) \) and its onset \( \text{Onset}(n) \). As there is no sense in having the same note (i.e. same pitch and onset) inside a given voice we have to add the constraint (7).

\[
\text{Pitch}(n) = \{p_1, \ldots, p_l\} \\
\text{Onset}(n) = \{o_1, \ldots, o_l\}
\]

\[
\text{V}_0(0, -n) = \oplus \text{Pitch}(n) \Rightarrow \text{Onset}(n) = \oplus \text{Pitch}(n) \\
\Rightarrow \text{Onset}(n) \neq \text{Onset}(n)
\]

We represent the two voices of the problem by the arrays \( \text{Pitch}(V_0) \), \( \text{Onset}(V_0) \), \( \text{Pitch}(V_1) \), and \( \text{Onset}(V_1) \). Expressions (8) and (9) state the round property between the scores.

\[
\begin{align*}
(V_0 \times \text{Offset}) & \Rightarrow \text{Plus} \Rightarrow V_1 \\
\text{Onset}(n) & = \text{Onset}(n) = o_0 \\
\text{Pitch}(n) & = \text{Pitch}(n) \Rightarrow \text{Pitch}(n) = \text{Pitch}(n)
\end{align*}
\]

To enforce the harmony constraint, we use an array of sets Consonant. A pitch \( p_i \) is consonant with a pitch \( p_j \) if \( p_i \in \text{Consonant}(p_j) \). This is stated by expression (10). Notice that similar constraints must be imposed for pitches of notes of a given voice to be consonant.

\[
\begin{align*}
\forall j \in \text{V}_0: \text{Pitch}(n) & = \text{Pitch}(n) \Rightarrow \text{Pitch}(n) \in \text{Consonant}(\text{Pitch}(n))
\end{align*}
\]

1.3. Comparison of the two approaches

Let us compare the relation-oriented modeling of the problem with the integer-oriented approach. The relational model is simpler and it keeps together the notions of pitch and onset that characterize a note. The model works for scores of any length with no ad hoc size parameters. On the other hand, the integer model uses two arrays of ad hoc size and a constraint to represent the notion of a note. Moreover, that constraint is translated to \( O \times (1 \times (1 \times 1) \times \ldots) \) binary constraints on the solver. The properties represented by (8) and (9) are translated into \( l \) constraints each. The disadvantage of this translation is that it makes constraint inference limited to two properties of the same note at a time while ignoring useful information when the score itself is considered. This reduces the efficiency of the CP solver in the integer model.

There are several additional advantages to use relation domains. First, it can be seen that generalising the problem to n voices is easily done in the first model but not in the second one. The only thing to do is to allow \( O \) set to contain several elements. Notice that in that case \( V_1 \) will actually contain several follower voices. Moreover, if Offset is not fixed at modeling time, the number of follower voices are not fixed either. Offset could also be constrained in some way, e.g. all offsets are multiple of a given value, for instance 4. Even more, as it will be shown in Subsection 5.4, it is possible to offset several musical parameters of a score, and then to constrain that general “offst structure”. If we were working with integer sets, those generalisations would not be straightforward at all (e.g. we have to use one array per parameter per voice and use a special value for silences).

1.4. Contributions

This paper defines the relation constraint domain and explores ways to use it in the field of music. The notions of relation and tuple allow to represent musical entities such as score and note in a more concise and natural way than traditional domains. General purpose constraints provide the building blocks for representing new abstractions that benefit final users such as composers. These constraints complement other domains by allowing constraint inference at higher levels of abstraction.
The constraint domain is implemented on top of the Gecode [12] constraint library, its sources are freely available [19] and an interface for integrating it with OpenMusic [1] is in development [19].

1.5. Document structure
Section 2 shows how relations can express key concepts in music. Section 3 introduces CP with relations and defines the new constraints of this domain. Section 4 explains how CP with relations can be used to model musical CSPs. Section 5 gives modeling examples that target musical composition and Section 6 shows how to solve a larger example. Section 7 briefly presents our implementation in Gecode. Conclusions and future work are presented in Section 8.

2. MUSICAL CONCEPTS AS RELATIONS
Current music constraint systems1 all share the same problem internally2, highlighted by the composer J. Kretz in [10]: a lack of structure in the music representation that forces to handle musical parameters independently, making them difficult to interact. J. Kretz proposed then to use an (hypothetical) "organized structure for notes (a bundle of information containing many parameters like pitch, duration, ...)" in order to define "more 'intelligent' rules" [10]. In a sense, this is what we are proposing in this work. This section defines the concepts needed to link this general idea with constraint programming on relations. We define the two concepts of Musical Bundle (MB) and Musical Bundle Sets (MBS) as a way of representing musical concepts in terms of tuples and relations, which are the basic concepts of our new constraint system.

Musical bundle. Is a set of pairs where each pair combines a musical parameter with an integer value. An example of a MB is a note defined by several musical parameters, such as the following tuple:

\[
\text{tuplet} \leftarrow \{\text{pitch}, \text{duration}, \text{onset}, \text{instrument}, \ldots\} \tag{11}
\]

Musical bundle set. Is a set of musical bundles. Examples of MBSs are a chord, a bar, or a score. But an MBS can express more abstract musical concepts as well, for example a transformation between two scores can also be represented as an MBS. When modeling musical problems, we will use the terms MBS and relation variable interchangeably.

![Figure 1. Score represented by a MB relation. Every note is represented by a MB tuple.](https://example.com/figure1.png)

In figure 1, the note C played on the first beat and the note F played on the second beat can be respectively represented by the tuple (60, 1) and the tuple (65, 2) (if we use MIDI values for the pitch parameter). The score represented by figure 1 is then represented by the set of all the tuples (i.e., a relation) that represents the different notes played on the score.

3. CONSTRAINTS ON RELATIONS
Constraint programming on relations is about using relation decision variables along with constraints that enforce properties on these variables. It also provides search abstractions and predefined generic heuristic strategies for solving models with this kind of variables. A relation decision variable (relation variable for short) represents a relation out of a set of possible relations. A relation is a set of tuples of the same arity. A tuple is an element of the cartesian product of the finite set of integers. \[\mathbb{Z} = \{0, \ldots, k\}\] The arity of a tuple is the number of elements that belongs to it. For instance, tuples of arity \(n\) are elements of the set \(\mathbb{Z}^n\). This set is also represented as \(\mathbb{Z}^n\).

The domain of a relation variable \(\text{X}\) denoted \(\text{DX}\), is the set of possible relations that \(\text{X}\) can be assigned to. A constraint \(C\) on a set of relation variables \(\{\text{X}_1, \ldots, \text{X}_n\}\) will ensure that the domain of every variable \(\text{X}_i\) only contains relations that satisfy it. This is also called constraint inference. A variable is considered determined or assigned when its domain contains only one relation. The constraints that can be used on relation variables come from two fields: set theory and relational algebra [7]. As a relation variable represents a relation it makes sense to express properties like: \(\text{X} \supseteq \text{Y}\) and \(\text{X} \subseteq \text{Y}\), where \(\text{X}\) and \(\text{Y}\) are relation variables. Properties from the relational algebra include for instance: \(\text{X} \bowtie \text{Y} = \text{Z}\), which states that the join of the two relations \(\text{X}\) and \(\text{Y}\) on the components in \(\text{C}\) must be the relation \(\text{Z}\).

The notion of tuple concatenation is used for the definitions of the constraints. Given two tuples \(r = (r_1, \ldots, r_m)\) and \(s = (s_1, \ldots, s_n)\), we represent their concatenation by \(r \cdot s = (r_1, \ldots, r_m, s_1, \ldots, s_n)\). The arity of the resulting tuple \(r \cdot s\) is \(m + n\).

Projection. This is a binary constraint on variables \(X\) and \(Y\) of different arities. It takes a constant \(i\) as an extra parameter, the number of components on the right of \(X\) that is projected. Its semantics is:

\[
\begin{align*}
\sum_{i} X & = \sum_{i} X_i Y_i, \quad \text{for } i = 1 \ldots k \tag{12} \\
\{1, \ldots, i-1, (i+1), \ldots, k\} & \subseteq \text{proj}(\text{Y}) \quad \text{if } X \supseteq Y \end{align*}
\]

As a special case, this constraint creates a channel between a relation and a set decision variable, if \(i = 1\).

Join. This is a ternary constraint on relations. It takes also a constant \(i\), which represents the number of components on the right of \(X\) and on the left of \(Y\) that are considered by the constraint. Its semantics is: \(X \bowtie Y = Z \bowtie \forall v : |a| = i \tag{13}\)

\[
((r \cdot u \subseteq X \land u \cdot s \subseteq Y) \iff (r \cdot u \cdot s \subseteq Z)) \tag{14}
\]

Compose. This is a special form of the join constraint where the joined components are removed from the result. It matches the semantics of relation composition from the relational algebra [7]. Its semantics is: \(X \bowtie Y = Z \bowtie \forall v : |a| = i \tag{15}\)

\[
((r \cdot u \subseteq X \land u \cdot s \subseteq Y) \iff (r \cdot u \cdot s \subseteq Z)) \tag{16}
\]

Confluent join. This is a special case of \(\bowtie\) with an additional confluence condition. The additional condition ensures that it collects only those combined steps in \(Z\) for which all possible first steps in \(X\) can always find a next step in \(Y\) that continues to the same result. \(X \bowtie Y = Z \bowtie \forall v : |a| = i \tag{17}\)

\[
((r \cdot u \subseteq X \land u \cdot s \subseteq Y) \iff (r \cdot u \cdot s \subseteq Z)) \tag{18}
\]

Permutation. This allows to impose that a relation is equal to another one but with some components permuted. Using it, we can apply projection, join, compose, confluent join and confluent compose constraints on components of relations without regarding their position. We do not define it here because we will not use it explicitly in the presented models, in order to keep them more readable.

4. MODELING WITH MUSICAL BUNDLE SETS
This section presents some of the new possibilities that CP on relations offers in the musical field. When we define a new musical problem in terms of relation variables, we need first to declare these variables. We declare them by giving the minimum and maximum MBS that the variable can assume. We then impose constraints on these relation variables. The constraint solver then uses a combination of propagation and heuristic search to find values of the relation variables that satisfy the constraints. For example, suppose that we are interested in finding a musical piece \(M\) with some specific characteristics. The piece itself is represented by the relation variable \(M\) with the following minimum and maximum MBS:

\[
M \in [0, \ldots, \{\{0, \ldots, 72\} \times \{0, \ldots, 7\}]\tag{17}
\]

This notation says that the minimum MBS of \(M\) is empty (i.e. 8 beats of silence) and the maximum MBS of \(M\) is the musical piece that contains all the notes from middle C to C one octave higher, occurring on the first 8 beats. The minimum and maximum MBS of \(M\) are represented by the following scores:

Additionally we want our score to respect some composition rules. This is done by imposing constraints such as those given in Section 5. In some sense, this corresponds to what a composer does when he wants to compose a new musical piece. He starts from nothing and can potentially add and delete. From this provided "musical material", he decides what he will use to construct his piece by imposing some "properties" on it.

4.1. MBS as a transformation of MBs
An MBS can be used to define a transformation between two other MBSs. For instance, each note (pitch, onset) in the original MBS can be transformed into (pitch‘, onset‘) in the new MBS. How the new values for the attributes are computed depends on an MBS that represents the transformation itself. As a particular example, let us consider a score \(\text{score}\) like the one used in the introduction and a relation \(T\) to represent the intended musical transformation. Every element of \(T\) has the form: \(\{\text{pitch}_{\text{trans}}, \text{onset}_{\text{trans}}, \text{pitch}_\text{on}, \text{onset}_\text{on}\}\)

To obtain the transformed score \(\text{score}_{\text{trans}}\) (represented by the binary relation \(\text{score}_{\text{trans}}\)) resulting from the transformation of \(\text{score}\) by \(T\) we impose the constraint

\[
\text{score}_{\text{trans}} = \text{score} \cdot T \tag{19}
\]

In this way we ensure that any MB of \(\text{score}\) that can be transformed by \(T\) has one (or several respective) transformed MB in \(\text{score}_{\text{trans}}\). More generally, it is actually possible to link any two relation variables with a third relation variable that represents the transformation, if we code the transformation in terms of integers. All three relation variables in this link will participate in the solution process. That is, the CP solver can use the third variable to calculate the transformed relation, but it can also use the two relation variables to calculate the third variable.

\footnote{We assume here a signature 4/4 and quarter notes. For this example we only consider pitch and onset as the relevant parameters of the MBs.}
4.2. MBS as an aggregation of MBSs

One way of linking a set of MBSs together is to aggregate them into another MBS. This can be done easily by adding a parameter to each MBS and assigning it a different value for each MBS, i.e., all the MBSs of a given MBS have the same value for this new parameter. Formally, if $S$ is the set of MBSs to aggregate, we have

$$\forall s_i \in S, \forall m_b \in s_i; \quad m_b = i \in A$$

where $A$ is the MBS that aggregates the set of MBSs $S$, and $m_b = i$ represents an MB constructed from the MB $m_b$ in which a new parameter with value $i$ has been added. All this can be achieved using CP on relations. If $R_A$ and $R_m$ are relations that represent respectively $A$ and the $m^8$ MBS of $S$, we have

$$R_A = R_m \cup \{i\}$$

(21)

In this way, we can for instance aggregate two scores (e.g., the left and right hand parts of a piano score) to form a new score. We can then compute the resulting score in some way, and this will be reflected in the parts of the aggregation.

5. EXAMPLES OF MUSICAL CONSTRAINTS

This section presents several examples to show how to use relation constraints to solve musical problems. Notice that, as said in Section 3, we will ignore here the use of permutation constraints, in order to clarify the text. Also, we will use an MBS named score that contains MBSs of the form $\langle \text{pitch, onset, duration} \rangle$ to represent a score and an MBS named Chords in which every MB has the form $\langle \text{pitch, ChordIndex} \rangle$ in order to represent a set of chords. In each MB, ChordIndex is an index identifying a chord (i.e. a set of pitches) and pitch is one of the pitches of the chord indexed by ChordIndex. By chord, we mean here any set of pitches (e.g. C major, beginning from middle C, in its fundamental position).

5.1. Forbid simultaneous chords

We propose here a constraint that is hard to express without relation variables. The constraint is the following: in a given score, we want to forbid to hear chords of some sets of complete chords altogether, even if the different notes constituting the chords do not begin to play at the same time. The constraint takes three musical parameters into account: pitch, onset and duration. We need an MBS FSChordSets that will be the set of forbidden-simultaneous-chords sets. Every MB in FSChordSets has the form $\langle \text{SetIndex, ChordIndex} \rangle$, where ChordIndex is an index identifying a chord and SetIndex is an index identifying a set of chords that cannot all be heard completely simultaneously. The first thing to do is to apply the constraint to transform the original score score into a score for which all the notes with a duration $n$ become a $n$ consecutive notes of duration 1. To do this, we need the transformation MBS represented by the relation

$$T_{\text{transformation}} = \{ \langle X, Y, Z \rangle; \quad Z \in [X, X + Y - 1], Y > 0 \}$$

(22)

To get the transformed score (represented by the relation variable $\text{score}_{\text{transformation}}$), we only apply the constraint

$$\text{score}_{\text{transformation}} = \text{score} - T_{\text{transformation}}$$

(23)

After that, we need to get all the complete chords heard at a given time in score, that is, all the complete chords played at a given onset in $\text{score}_{\text{transformation}}$. This can be done using the constraint:

$$\text{IO} = \text{Chords} \cup \text{score}_{\text{transformation}}$$

(24)

where IO is a binary relation linking a given onset with chords completely played at that last part. The next step is to express the constraint to impose that on a given onset (of $\text{score}_{\text{transformation}}$), we cannot have all the chord indexes of a given chord set index.

$$\theta = \text{FSChordSets} \subset \text{IO}$$

(25)

Notice that if the constraint is not respected, we know because of which chords, by using the operation confluent compose. This information can help to lead the search.

5.2. Score harmonization

Another constraint that can be easily expressed is the harmonization of a score during the search to determine that score. So, this information can be used to lead the search in some way. One more time, we use here several musical parameters together in order to express what we want. Here, we will only use a part of a score (for instance, a bar), and look for one chord harmonizing that part. Moreover, in some cases, we can propose several chord alternatives. The first step to express the constraint is to get all the notes of the given part and to keep the expected constituent notes of the chord. In our case, we simply keep the notes with a minimal duration value. To get the relevant notes, we create a new MBS (represented by the relation variable score) that contains only those notes. To do this, we impose the constraint

$$\text{score} = \text{score}_{\text{transformation}} \times \text{Onsets} \times \text{Duration}_{\text{transformation}}$$

(26)

where Onsets and Duration$_{\text{transformation}}$ are respectively the set (unary relation) of onsets of the part$^{19}$ and the set (unary relation) of all possible durations, with a minimum lower bound. After that, we just need to identify the chord index of all chords that can be used to harmonize the given part of the score. To do this, we use the constraint

$$\text{harmscore} = \text{score} \cup \text{Chords}$$

(27)

where harmscore is a relation linking onsets (and possibly other musical parameters different from the pitch) and chord indexes. The harmonization is then done. The interesting thing with this constraint is that it is possible to use partial information inferred from that constraint. For instance, if at some time during the search we know that a majority of chords of a given tonality has been used to harmonize the score, we will first try to impose the use of pitches of that tonality in the score.

5.3. Simplified orchestration

Suppose we want to distribute a piece of music score on two instruments $i_1$ and $i_2$ (it could be generalized to n instruments easily). Scores of these instruments can also be represented by two relation variables $\text{score}_{i_1}$ and $\text{score}_{i_2}$, whose components have the same semantics as those of score. The only thing we have to impose are:

$$\text{score} = \text{score}_{i_1} \cup \text{score}_{i_2}$$

(28)

and

$$\text{score}_{i_1} \cap \text{score}_{i_2} = \emptyset$$

(29)

There are of course lots of solutions to this problem. But it becomes interesting when we begin to add other constraints on $\text{score}_{i_1}$ and $\text{score}_{i_2}$, for instance one is the transformation of the other by some transformation.

5.4. Score made of musical patterns

We present here a constraint that is in some way an extension of a constraint used for Michael Jarrell’s “Con-”

Heurges” (presented by Serge Lemouton in [11]). The constraint that will be expressed here states that a score is made only of a given set of musical patterns, i.e. sub-scores in which notes are represented by several parameters. Let this set be represented by the relation Patterns. Every tuple of Patterns has the form

$$(\text{indexPattern}, \text{pitch}, \text{onset}, \text{duration}, \ldots), \quad \text{score}_{\text{transformation}}, \ldots, \text{score}_{\text{transformation}}$$

Patterns is a relation of arity $n + 1$ to be used to create a relation that represents a score with $n$ parameters. The component indexPattern is used to identify one pattern in the set of patterns.

We would like to impose that a given score is made from some patterns of this set of patterns, and nothing else. In order to do this, we will simply get the used patterns from the current score (found during the search), and recreate a new score from those used patterns. If the recreated score is equal to the initial one, only patterns of Patterns have been used, that is, the constraint is satisfied$^{19}$.

The possibility to shift the patterns in some components should exist. To allow this, we use a relation variable Shift that will represent all the shifts applied to patterns. Every tuple of Shift has the form

$$(\text{indexPattern}, \text{offset}_{\text{param}}[1], \ldots, \text{offset}_{\text{param}}[n])$$

Using Shift, we can get a relation variable that represents the set of all possible shifted patterns. We explain in the following how, but we must first introduce the relation Plus$^n$:

$$t \in \text{Plus}^n \iff \forall i \in [0, n - 1]: \quad (t[i] + t[i + n] = t[i] + 2n)$$

(30)

Plus$^n$ can be defined in terms of Plus (defined in Section 1):

$$\text{Plus}^n = \text{Permute}_{\text{plus}}(\text{plus} + \ldots + \text{plus})$$

(31)

With Patterns$\text{shifted}$, we are able to retrieve all the patterns used in the score score, even if they have been shifted in the score.

$$\text{usedPatternsInScore} = \text{score} \cap \text{Patterns}$$

(33)

Having used confluent join instead of confluent compose allows us to keep the information about the score. The relation variable usedPatternsInScore gives us the information about how and which patterns have been used in score. The only thing that remains to be done is to check if the used (shifted) patterns construct exactly the score score. We only need to remove the pattern indexes from usedPatternsInScore and check (impose) equality with score:

$$\text{score} = \bigcup n \text{usedPatternsInScore}$$

(34)

Adding some constraints on Shift can help to have more control on the solution. For instance, we can avoid to have the exact same shifts for different patterns. Moreover, we can notice that this constraint can be used incrementally. Indeed, we can first use some patterns in a given search, and afterwards use patterns that contain some of those patterns as subpatterns, and so on. Eventually, we are here more general than specified because the definition of confluent join contains only an implication and not a double implication: this allows to use only parts of the patterns, i.e. subpatterns. In order to force the use of complete patterns, we simply need to modify the definition of confluent join with a double implication instead of the implication.
6. A COMPLETE EXAMPLE

This section presents a practical problem solved in the OpenMusic environment using constraint programming on relations. To work with Gecode inside OpenMusic we use GeLSo ("Gecode in common Lisp using Sockets").

The musical CSP is the one presented in Section 1 with some additional constraints:

\[ V_1 (V_1) = \emptyset \]  
(35)

\[ V_{\text{OnsetPart}} = V_0 \lor \text{OnsetPart} \]  
(36)

\[ V_{\text{OnsetPart}} = V_0 \lor \text{OnsetPart} \]  
(37)

\[ (V_{\text{OnsetPart}} \times \text{OffsetPart}) \setminus \text{Part} = V_{\text{EarlierPart}} \]  
(38)

\[ V_{\text{EarlierPart}} \land V_{\text{LaterPart}} = \emptyset \]  
(39)

Expression (35) states that the two voices do not have any note in common. Expressions (36) and (37) allow to get some note subsets of the voice \( V_0 \). With expression (38), we are able to shift the MBS \( V_{\text{OnsetPart}} \) on 8 onsets later. Eventually, thanks to expression (39), we impose that the shifted part of \( V_0 \) does not have any common note with \( V_{\text{EarlierPart}} \).

Here are two solutions to the musical constraint satisfaction problem presented above. In both cases, we only allow to use the MIDI pitch values 60, 67, and 72 and they can all be heard together. In the first case, we impose that the parts of \( V_0 \) that cannot be the same are the first 4 beats and the beats from 8 to 12. In the second case, the constraint is applied on the first 5 beats and on the beats from 8 to 13. Experimentally, we noticed that from 6 beats on, no solution can be found in a reasonable time (in this case less than 2 minutes). But no special heuristic is used for now.

7. IMPLEMENTATION

We now explain how the new relation constraints are implemented. The implementation is available as an extension of the Gecode constraint library [12]. A modified version of the BDD library CUDD [13] is used internally for the domain representation. The source code of the complete extension is available [9] under the terms of the MIT license.

7.1. Domain approximation

As the domain \( D_X \) of a variable \( X \) can be a considerably large set of relations we require an approximation of it that can be practically stored. The goal of this approximation is to provide a good trade-off between the complexities of representing the data of the domain and the basic operation on this data that supports the operations in the domain. \( D_X \) is approximated by the pair of relations \( \{\text{glb, lub}\} \). The set of represented relations is \( \{R : \text{glb} \subseteq \text{lub} \} \). \text{glb} stands for greatest lower bound and \text{lub} for least upper bound. In the following we use \( \text{glb}(X) \) (resp. \( \text{lub}(X) \)) when referring to the relation \( \text{glb} \) (resp. \( \text{lub} \)) of \( D_X \). This representation of the domain was proposed in [8] to represent the domain of integer sets.

7.2. Domain and bound consistency

With the formalization of the relation domain as presented in Section 3 it is possible to have constraints that enforce domain consistency [12]. Let us consider a constraint \( C \) with scope \( X = \{X_1, \ldots, X_n\} \). \( C \) is domain consistent if for every \( X_i \in \{\text{glb}(D_{X_i}), \text{lub}(D_{X_i})\} \), \( \forall \chi \neq \emptyset \subset D_{X_i} \cdot C(\chi) \land \chi \not\in \text{lub}(D_{X_j}) \cup \text{glb}(D_{X_j}) \land X_i \neq X_j \) holds. Domain consistency is stronger than bound consistency. Bound consistency is more general because it allows to represent the domain of integer sets.

An implementation of the domain on top of the Gecode [12] library is available [9]. This allows to start using the following 5 new constraint types in musical applications using C++. An interface to use Gecode (including the relational domain) from OpenMusic [1] is currently work in progress [14].

8.1. Future work

The work we have presented is just the beginning of a wider set of applications of CP on relations in the musical field. We recognize that we can target other areas like orchestration where we are already working. New abstractions, terms of constraints that address particular properties on music are also under consideration. For instance, using musical transformations, we could model theme and variation problems. Moreover, using the subset constraint we can consider modeling leitmotiv constraints.

The MBBS we used are generally scores but we could consider to work with other types of MBBS, such as set of patches and some constraints are represented in terms of physical parameters. In the future, we also plan to develop a visual framework inside OpenMusic [1] for composers. This framework should contain built-in constructs such as scores, bars, and other MBBS, some predefined constraints that can be applied on them and some search heuristics. Moreover, it should allow the possibility to be extended with new constraints, constructs and search heuristics. All those concepts should be represented visually, using visual components such as patches.

9. REFERENCES


A METHOD FOR COMPUTER CHARACTERIZATION OF “GESTURE” IN MUSICAL IMPROVISATION

Christopher Dobrian
Professor of Music
University of California, Irvine
dobrian@uci.edu

ABSTRACT

In the design of interactive computer music systems and the composition of interactive computer music, the tracking and analysis of "gestures" characteristic motions discerned within musical attributes — provides a promising challenge. There are in fact ways that one can clearly and empirically define and identify "gesture" in musical content, often with conceptual models and tools similar to those used for tracking and identifying physical gestures. The analysis of musical gesture as "significant motion" can be applied to many aspects of music: melodic contour, note speed and density, loudness, level of dissonance, etc. Gestures can be characterized by the shapes produced by measuring changes in these aspects over time, and the derivation of data about change, rate of change, etc. within a particular feature or set of features.

Computer evaluation of gesture may be divided into the tasks of measurement, segmentation, identification, and taxonomy. What are the elements of musical gesture and how can a computer best discern them? How can a computer know when a gesture begins and ends? How can different, unforeseen gestures be compared and classified? Perhaps most significantly, how can a computer, once it has identified and characterized a gesture, attribute musical meaning to it? This research proposes criteria and groundwork for the tracking, measurement, and analysis of "gesture" in the musical content of sound structure, and the use of that analysis in interactive computer music.

1. INTRODUCTION

Gesture is any motion that, by certain imminent characteristics of its conveyance meaning. As it is used in musico-linguistic discourse — by composers as well as theorists — "gesture" refers not so much to the physical action of a performer as to ways of characterizing musical content; the content itself implies motion, and that motion conveys characteristic meanings.

The discussion of "gesture" in musical content has taken place mostly in musico-linguistic studies of classical music [7, 8, 9, 10], and the definition of gesture as used in this context remains much more poetical or concrete and empirical. Yet there are in fact ways that one can clearly define and identify "gesture" in musical content, often with conceptual models and tools very similar to those used for tracking and identifying physical gestures. The analysis of musical gesture as "significant motion" can be applied to many parameters of music, the shapes (functions) produced by measuring changes in these aspects over time, and the derivation of data about the morphology of change within those parameters. The existing techniques for tracking and analyzing the physical gestures of a performer can therefore be applied similarly for tracking and analyzing the gestural nature of the music itself. The new insights thus gained into the nature of musical gesture can be applied in interactive music systems to enhance the expressivity of computer music.

The successful capture, tracking, and analysis of the physical gestures of musicians has been a central topic of research in interactive computer music for years [12], and notably has been the subject of much important recent research by the Realtime Musical Interaction (RMIT) team at IRCAM [1, 2, 10]. This research is important because the design of responsive computer instruments depends on the successful translation of physical gestures into expressive control of a sound generator. In the design of expressive instruments, the syntonic relationship between physical gesture and sound is a vital part of music performance and musical understanding by the listener. Thus, in addition to the tracking and analysis of physical performance gestures, this other aspect of the word "gesture" — the kinetic quality evoked by the nature of the sound itself — is equally important in developing expressive, intelligent interactive music systems.

As I have pointed out in articles on the topic of "interactivity" [4, 5], for a computer system to be considered truly interactive, the computer must be capable of cognition of unforeseen musical events in the environment, and it must have the power to respond to them autonomously. The use of an interactive system therefore inherently demands improvisation on the part of both the computer and the live performer. A test of the computer's analysis of gesture must include the computer's response to unforeseen — i.e. improvised — musical input. The computer can demonstrate the success of its analysis by contributing appropriate responses to perceived musical change.

This article describes a method for measurement and analysis of "gesture" in musical content, and employing that gesture recognition in new interactive music software for improvisation between live instrumentalist and computer.

2. APPROACH

Crucial to this effort is the fact that music can be metaphorically mapped into spatial dimensionality. For example, in the West we talk about pitch "height".

referring to pitches as "low" or "high". The basis of metaphor is a mental technique that cognitive scientists refer to as "cross-domain mapping". Cross-domain mapping refers to drawing direct correspondences between an incompletely understood source domain (such as musical pitch, in this example) and a useful target domain, such as spatial height. In his book Conceptualizing Music [14], Lawrence B. Zikowski points out that such cross-domain mappings are largely culture-dependent.

When we measure and graph any musical parameter over time, we are in effect employing cross-domain mapping to visualize the morphology of that parameter's evolution over time as a shape in two-dimensional space (a graph of the parameter as a function of time). Thus, if the computer can detect and characterize the shape of that parameter over time, and compare and categorize different shapes, it can establish a lexicon of shapes (or shape descriptions) that refer to particular types of motion in a musical parameter.

Starting from a working definition of gesture as "significant motion", how should the computer analyze this motion, and how should it detect significant motion? For a computer to "learn" something useful about a set of gestures, it must be able to measure (quantify aspects of), segment (seek beginnings and endings of), characterize (reduce the information of), and categorize (compare and contrast) the motional quality in musical sound structure.

My initial research and experimentation on this topic has yielded the following few insights so far.

1) In many cases "gesture" can be found by directly examining implicit motion (i.e. significant change over time) in environment sound and musical data.

2) Larger phrases or data sets can be segmented into individual "gestures" by evaluating the data for "remarkable events" — unusual occurrences in the data.

3) The so-called remarkable events are almost invariably indicated by data that would be termed outliers in the statistical sense.

4) Dixon's simple algorithm for detecting a single outlier in a small set of numeric data [3, 11] can be implemented in real time and can be used successfully to detect outliers in an ongoing stream of realtime data.

5) Gestures thus segmented can be characterized using obvious traits in the "motion" of the data, such as its duration, its overall slope from beginning to end, its jaggedness or smoothness, and its linearity or curve.

6) For detecting a gesture's linear/sparse, evaluation of its autocorrelation via linear prediction RMS error appears to be more "musically meaningful" than measurement of standard deviation statistics.

7) These listed traits can be used to make a multi-dimensional categorization of all the gestures in a given input data stream.

8) Statistical evaluations of these categories can give useful information about the prevalence of certain kinds of gesture.

9) Slight modifications of any one of these traits can result in new but musically recognizable variants of the gestures evaluated in the input data.

10) Such variants are comparable to some musical responses utilized by human improvisers, and can give a sense of intelligence and expressivity in a computer-generated improvisation in real time.

3. METHODOLOGY

3.1. Measurement

In the initial stages of experimentation, I have elected to focus on those aspects of the musical sound structure that are relatively easy to measure with some degree of reliability and that are directly related to common theoretical understandings of important elements of musical discourse. I thus chose to focus on pitch (specifically pitch interval), dynamics (differences of loudness in decibels), and rhythm (changes in inter-onset time interval between note attacks). Undoubtedly, measuring a wider range of sonic attributes, including useful features such as timbre (changes in spectral centroid), will be useful in the future for obtaining a fuller evaluation of the sound structure. But as a first step, a combined consideration of pitch interval, changes in loudness, and changes in inter-onset intervals (IOIs) provides what seems to be a sufficient body of musically useful information for analysis.

3.2. Segmentation

In order to divide a continuous stream of musical input information into distinct entities that the computer can refer to as "gestures", the incoming data of pitch intervals, loudnesses, and IOIs are continually analyzed in search of outliers — significantly distinct data points that might signal the beginning of something new. Using the statistical outlier detection algorithm known as the Dixon Q test, each new data point is evaluated to see if it should be considered significantly different from what has come before. When an outlier is detected in any of the parameters under consideration, a determination is made that a new gesture has begun. The gesture preceding that data point is considered an entity to be characterized and remembered, the program resets itself, and the process of measurement and segmentation begins anew.

The Dixon Q test was selected for its extreme simplicity and its apparent effectiveness in detecting single outliers. However, there are many more complex and sophisticated segmentation methods that should be explored in future research, in an effort to find segmentation criteria that can adequately designate gesture beginnings and endings that correspond to musicians' intuitive understanding of musical gesture.
3.3. Characterization

Once the gesture bounds have been determined, the three musical attributes under consideration in this methodology — pitch intervals, decibel changes, and IOI changes — are each characterized in that gesture, according to a few key measures. Each attribute is assigned a slope, based on its change from beginning to end over the length of the gesture, a ‘jaggedness’, defined as the number of times it changes direction within the gesture, a “dispersion” based on how widely it deviates from a linear path from beginning to end, and a “centroid” based on its mean value. These attribute characterizations are stored — along with some global characteristics of the gesture such as its length (number of events), its starting time index (amongst all the detected note events), and its ordinal index of occurrence (which gesture it was) — as a single gesture description vector (a one-dimensional array) in a matrix of gesture descriptions.

A single gesture description is thus an array consisting of ordinal index, note index, length, slopes, jaggednesses, centroids, and dispersion values. This is not a record of the exact contents of the gesture — although the note index and length can, if desired, be used to look up the exact recording of the gesture in question — but rather a reduced-information descriptor of salient characteristics of the gesture.

3.4. Categorization

Because the gesture descriptions are, with this method, stored as ordered arrays of specific known characteristics, the list of gesture descriptions (i.e., the array of arrays) can be sorted according to any trait. For example the gestures can be sorted by order of occurrence, length of gesture, steepness in slope of any attribute (pitch height, loudness, or IOI), jaggedness, etc. Such sorting leads to gesture descriptions that are in some way similar being stored adjacent to each other in memory. This makes it easy for an improvising program to access related gestures simply by choosing other gestures that are located nearby. Thus, without the computer program needing to be imbued with any artificial musical intelligence about relationships between musical gestures, it can be made capable of choosing similar or dissimilar gestures based on their proximity after various sorting operations.

4. IMPROVISATION

One useful test of the effectiveness of this way of modelling the “gestural” nature of musical sound is to employ these gesture descriptions as input for a higher-level generative improvising algorithm. Indeed, this was the motivation for this research in the first place. The goal of this research is to work toward making an improvising computer algorithm seem more gesturally dramatic (and thus, one hopes, seemingly more improvising computer algorithm seem more gesturally generative improvising algorithm. Indeed, this was the employment of these gesture descriptions as input for a higher-level generative improvising computer algorithm to provide a more lively, ‘embodied’, and “gestural” computer-generated performance in concert with a live improviser.

5. CONCLUSION

The term “gesture” as used to describe the evocation of musical sound is not generally sufficient to function as a satisfactory improvising partner. Therefore, improvisers actually employ a great many higher-level methodologies to shape the larger formal structure of a performance. Improvisers also develop and employ a personal repertoire of modes of decision making and modes of response. Furthermore, a good improviser learns by observing the modes of response employed by her/his musical interlocutor(s). Thus, while gesture characterization and categorization is demonstrably useful as a way of giving a certain gestural evocation to computer-generated musical phrases at the local formal level, this technique must be employed in the context of other algorithms for formal structuring, decision making, and higher-level learning. Such techniques are beyond the scope of this article but they are the subject of ongoing research by this author, research that will be described in future writings.

6. REFERENCES


DEVELOPING AND COMPOSING FOR A ROBOTIC MUSICIAN USING DIFFERENT MODES OF INTERACTION

Mason Bretan, Marcelo Cicconet, Ryan Nikolaidis, Gil Weinberg
Georgia Institute of Technology
Center for Music Technology, Atlanta, GA
masonbretan,cicconet,ryannikolaidis@gmail.com, gilw@gatech.edu

ABSTRACT

This paper describes three compositions written for the improvisational robotic marimba player, Shimon. The three compositions demonstrate multiple modes of interaction between Shimon and a human performer. Some of these modes are created using techniques from pattern recognition, tempo detection, and computer vision. We illustrate the process of composing and developing for Shimon while attempting to maintain creativity and musicality considering the limitations and opportunities that emerge when working with a mechanical apparatus.

1. INTRODUCTION

Research in robotic musicianship opens the door for new and interesting methods of creating and performing computer music. The combination of software and mechanics provides unique opportunity and limitation for developers, composers, and performers. Robotic musicians greatly differ from the pure software based computer music applications which bestow an unlimited library of timbre, floating point precision, and extremely low latency. Some of the most distinctive differences prevalent in robotic musicianship are the ability to create unique, acoustically sound using traditional instruments, their mechanical limitations (encompassing everything from the rate of movement to the mere fact that their extremities occupy physical space), and perhaps most importantly their ability to provide visual feedback through a much more natural and engaging experience for an interacting human musician.

Robotic musicianship research has been a major focus at the Georgia Tech Center for Music Technology (GTCMT), most notably with the development of Haile [8] (a robotic percussionist) and Shimon [9] (a robotic marimba player). In the last year GTCMT researched multiple approaches to musical human-robotic interaction. This work culminated in compositions each building from these different approaches. This paper describes three interactive compositions written for Shimon and a human performer. With each piece we address underlying facets of robotic musicianship including compensating for mechanical latencies, machine listening and generating note algorithms, computer vision, robotic head movement, and visual cues. Each composition uniquely demonstrates different approaches to creating music in the context of human-robotic interaction. These approaches involve various aspects of machine listening and computer vision methods, including methods such as pattern recognition, tempo detection, improvisation, and gesture recognition. It is important to note that when referring to Shimon we refer to the amalgamation of software, physical mechanism, and any additional vision system.

2. COMPOSING FOR SHIMON

There are a few mechanical idiosyncrasies inherent to Shimon that make composing and developing both interesting and challenging. These characteristics influence both the creative and scientific processes of working with Shimon.

2.1. Shimon’s Arms

In addition to path planning, programming the movement of Shimon’s head is another feature unique to robotic musicianship. Hoffman and Weinberg [4] demonstrated the advantage of visual cues such as arm movement and mallet strikes. Through empirical study, we found that well programmed head movements can provide even more valuable visual cues to an accompanying musician. These head related visual cues include gestures varying from Shimon looking towards a person versus its own arms, positioning its head high above the marimba versus very close, and nodding to the beat versus more generic breathing gestures. These cues provide an environment more similar to human-human interaction as opposed to human-robotic interaction. Shimon develops a defined personality and any interaction becomes social and naturally engaging. In our compositions a combination of previously developed higher-level head gestures (such as head banging to the beat) are used in coalescence with lower level commands (such as head and neck rotation).

2.2. Head Movement

In the first section Shimon responds to human arm movements analyzed by the Microsoft Kinect. The Kinect is able to provide positional values describing the depth, height, and width of each arm in relation to the camera. Each arm triggers one of two motifs (a total of four motifs). The horizontal position determines the particular motif to be played. Depth and height influence the tempo and velocity, respectively. Though Shimon may take a more deterministic role, the importance of detecting physical gestures to influence the music still resides. Humans use a plethora of physical movements to convey emotional or descriptive pieces of information. In performance humans often cue downbeats or transitions through exaggerated gestures. In Bafana, physical communication with Shimon parallels the performance of a conductor, in which the conductor makes movements to elicit musical responses from the musicians. Figure 3 shows a human performer in this state of interaction with Shimon.

3. BAFANA

Bafana was inspired by street performances of African marimba bands. The piece moves through several phases that use a number of repetitive interlocking rhythmic motifs, played interdependently on two marimbas by Shimon and a human performer. Shimon listens to the human improvisation, detects specific motifs, and responds in a manner that is times predictable and at times surprising. Each phase presents a different situation for the human player as Shimon is programmed to respond differently. The piece begins with a playful vision-based interaction between the human and the robot that introduces some of the musical motifs in a social non-rhythmic context. The human player then chooses any of nine possible motifs to start the piece, plays it in a loop, and awaits Shimon’s response. In the current implementation of the piece, Shimon’s personality develops from predictable consonant responses at the beginning, to more surprising chromatic reactions towards the end, challenging the human player to constantly listen and carefully adapt to the evolving nature of the music.

3.1. Part 1 - Vision Based Interaction

In this first section Shimon responds to human arm movements analyzed by the Microsoft Kinect. The Kinect is able to provide positional values describing the depth, height, and width of each arm in relation to the camera. Each arm triggers one of two motifs (a total of four motifs). The horizontal position determines the particular motif to be played. Depth and height influence the tempo and velocity, respectively. Though Shimon may take a more deterministic role, the importance of detecting physical gestures to influence the music still resides. Humans use a plethora of physical movements to convey emotional or descriptive pieces of information. In performance humans often cue downbeats or transitions through exaggerated gestures. In Bafana, physical communication with Shimon parallels the performance of a conductor, in which the conductor makes movements to elicit musical responses from the musicians. Figure 3 shows a human performer in this state of interaction with Shimon.

3.2. Part 2 - Melodic Interaction

The vision-based interaction serves as a precursor to the more involved melodic interaction, which serves as the principal mode of interaction for the piece. As previously
mentioned Shimon and the human performer influence each other by choosing from a library of nine motifs. The human performer initiates the transition from vision to melodic based interaction by playing any single motif. Because the motifs are known a priori, Shimon’s motif detection consists of a pattern matching algorithm which overlays the input midi values over the hardcoded values of each motif. A string edit distance threshold describes the number of differences between the actual motif and the input allows for errors. At the time of detection of this first motif the tempo is determined through a measurement of the inter-onset-intervals (IOI) and set for the whole piece. At this point Shimon begins nodding its head at a rate described by the tempo signaling to the human that it is listening and has matched tempo. During this portion of the piece the human leads while Shimon responds to each motif by imitating, repeating, and synchronizing with the human player. After a predefined number of motif changes has occurred Shimon no longer leads performance to playing in sync with the human. In fact, Shimon is programmed to not play in sync, but rather in canon. The repetitive nature of the beginning becomes less apparent as the patterns interlock, creating new and interesting sounds.

The music created in the first part of the piece is predominantly driven by the human’s decisions with Shimon’s melodic responses coming directly from the motif motifs chosen by the human. In the remainder of the piece the robot begins to make stochastic decisions and influence the progression of the piece. The algorithm continues to detect the specific motif of the human, but responds based on predetermined weights applied to each of the motifs. This weighting describes the probability of one motif being played with another. For each motif the human plays, Shimon answers with a degree of chance and surprise, possibly playing multiple motifs simultaneously. The progression of the piece is obvious as the interlocking of several motifs becomes more complex.

4. INTERPRETATIONS

Interpretations is a piece for Shimon and drumset. In this piece Shimon makes musical decisions based on a stylistic analysis of the drummer’s playing. Shimon runs a program which intelligently determines whether the drummer is playing in time or out of time or improvising more freely. Then based on the decision it responds in different ways. The pattern matching and response theme of Bafana largely inspired the writing of this piece. Interpretations similarly explores pattern recognition and matching algorithms, but in the context of real-time improvised performance. Many of the algorithms in this piece are based on attempting to extract significant patterns in the drummer’s play worth responding to.

4.1 Feature Extraction and Style Estimation

An algorithm using rule-based artificial intelligence techniques was developed to estimate tempo and style on a measure to measure basis. This method helped to calculate several parameters depicting the drummer’s play. The first consisted of a boolean decision deciding whether the drummer’s play was considered to be “in time” or “out of time.”

4.1.1. Tempo Detection

A piezo-electric sensor was put on each drum (bass drum, snare, small tom, and floor tom) with each signal being fed into Max/MSP. Using Pucke’s spectral difference onset detection object [6] an accurate symbolic representation of each onset (representing the specific drum) was established. Then, using this symbolic representation with the IOIs a tempo was able to be measured.

In order to estimate the tempo the algorithm was trained with a model describing a simple, typical pattern of one measure played by the drummer. As shown in Figure 4 this model describes a general pattern in which the bass drum is most likely to hit on beats 1 and 3 and the snare drum on beats 2 and 4. If this exact pattern is played Shimon calculates the tempo by simply divided the temporal length of the pattern by four (based on a fixed four beats per measure). The model describes an additional pattern of useful information, which is in general the bass and snare alternate a total of four times from one measure to the next. So another method to extract the tempo is to measure the length of time it takes for four alternations to occur.

Further analysis is computed using the tempo calculated from the previous measure to create a sixteenth note valued tatum grid. Each onset must be within a 1.5% threshold of the tatum to fit the grid. A weighted ratio of the onsets that fit the grid versus the onsets that did not describe the probability of the tempo. Additional weight is given to the bass drum and snare onsets if they fit both the grid and the model. When the ratio falls below a threshold a new tempo is estimated by using the alternations between the bass and snare and a new tatum grid is created with the value.

Through this process of determining the tempo it is possible to concurrently utilize the degree of deviation from the original model to simultaneously decide whether or not the drummer is playing in time or out of time. Because Shimon always attempts to calculate a tempo this deviation can be considered a confidence value describing how likely the calculated tempo is to the actual tempo. When the confidence falls below a threshold a decision is made that the drummer is now freely improvising without a set tempo. The number of onsets, the percent of onsets that fit as a sixteenth tatum of the calculated tempo, and the number of different drums hit are the features which determine this confidence value.

4.1.2. Pattern Recognition and Discovery

Unlike Bafana which uses pattern recognition to find specific, precomposed patterns, a primary focus of this piece is to discover improvised patterns. In interactive free improvisation one musician may repeat a motif or short musical pattern (rhythmically or melodically) which the other musician can respond to. These spontaneous patterns create interesting interactions between the improvising musicians and are the influence for developing such an algorithm. Two methods are used to find patterns of significance.

This approach is compared to a concatenation of StringA and the next contiguous string of length n, which we can call StringB. If StringA is equal to StringB then a contiguously repeated pattern exists. Here, n must be greater than three and at least two different drums must be represented.

4.2. Musical Response

Shimon’s response to the drummer’s play is dependent on the results of the style estimation.

4.2.1. In Time

Shimon uses notes from a precomposed melody, but how Shimon chooses to use the notes varies depending on the drummer’s play. The closer the drummer’s pattern follows the model the more likely Shimon is to play directly from the precomposed melody. As the drummer begins to ornament the pattern with additional onsets Shimon also begins to ornament the melody. This is done by adding notes to the melody based on a transition matrix from a first order Markov chain of the melody. When the drummer completely moves away from the model but is still in tempo (meaning each onset fits a sixteenth tatum of the previously calculated tempo) Shimon ignores the precomposed melody and generates notes based purely on the transition matrix. For each sixteenth note in a measure the likelihood that Shimon play a note is directly correlated with the note density of the drummer.

4.2.2. Out of Time and Improvisation

The primary features Shimon is programmed to respond to during the free improvisation is note density, recurring patterns, dynamic, and specific drum hits. The note density increases Shimon begins to play more dissonant notes. This is done by transforming the note chosen by the Markov chain such as shifting the note a half step or augmented 4th. A ratio describing the percentages of each different drum hit is continuously measured for the last 40 onsets. This ratio describes the variety of drums being hit and as this variety increases Shimon will continue to make note transformations by transforming the octave the note is played so that at the drummer’s peak variety each note Shimon plays is at a distance of at least 12 half steps from the previous note. The drummer’s dynamic is mapped to the velocity of Shimon’s notes in unison.

Using the pattern discovery algorithm Shimon continuously looks for drum patterns of significance. When one
is found Shimon creates another competing pattern derived from notes calculated from the most recent feature values. Similarly to the stochastic processes in Bafana Shimon may create and repeat a pattern of his own and the drummer can choose to respond how he sees musically fit.

4.3. Head Movement

Shimon is programmed to visually communicate software processes through the use of head movements. These visual cues help the human performer to better understand how Shimon is interpreting the drum playing. Shimon looks toward the drummer when either a new tempo is detected or it has a low confidence for the current calculated tempo. This not only gives the perception of Shimon "listening," but also informs the drummer of Shimon’s confidence level in the calculated tempo. During sections when Shimon is improvising the neck moves on its vertical axis so that Shimon looks closer towards the marimba to simulate concentration. Depending on the level of improvisation (creating patterns during the out of time section versus merely coloring the melody during the in time section) Shimon will adjust the neck level informing the drummer of the current state Shimon has detected.

5. GESTURE DETECTION

The percussionist trains the system before the performance with the three gestures he or she would like to use to send messages to Shimon. To train the system the musician simply creates a gesture (in this case moving a mallet to send messages to Shimon). Though we only used three gestures for this composition, the number of recognizable gestures supported by Shimon is virtually unlimited.

5.1. Training the System

The percussionist trains the system by the performance with the three gestures he or she would like to use to send messages to Shimon. To train the system the musician simply creates a gesture (in this case moving a mallet to send messages to Shimon). Though we only used three gestures for this composition, the number of recognizable gestures supported by Shimon is virtually unlimited. The first gesture triggers Shimon to play a section. Before the beginning of the song, the motif is played as soon as the gesture is recognized. After the beginning of the song, once the gesture is detected, the motif is scheduled to play in the next section of 8 beats. A second gesture controls how Shimon improvises during the section. Shimon is using a simple improvisational method in which it only uses notes from the major or minor scale degrees of the current chord in the progression. The gesture will trigger either a low or high note density. The third gesture controls when to end the piece. Once this gesture is recognized, the final section of 8 beats is scheduled to start after the current session of 8 beats.

5.2. Detecting Gestures and Performing

The gestures are detected in real-time during the performance, so the percussionist is able to communicate with the system while playing. Three gestures for this composition, the number of recognizable gestures supported by Shimon is virtually unlimited.

6. DISCUSSION AND CONCLUSION

Using Shimon we attempt to demonstrate the capability of a robotic system to adequately perform written scores, improvise, and respond to human performers all while enriching the musical experience through levels of sophistication (such as Shimon playing complex musical lines while concurrently analyzing a human’s play). Several other robotic systems such as the Logos Foundation robots, LEMUR bots, and the KarmitK Machine Orchestra have already demonstrated this. Aside from the musical response and unlike the other robotic systems much of the interactive focus is spent programming Shimon to visually communicate musical or software processes in a manner much more rich than purely solenoids striking a drum or piano keys moving up and down. Through our own interaction with Shimon and working with other musicians who have performed our compositions we have learned that visual cues and naturally engaging social interaction is essential to successful human-robotic performance and as shown by [5] is important to audience appreciation as well. Visual cues play a significant role in human-human interactive performance and through our compositions we attempt to demonstrate that some of those interactions can be successfully translated to human-robotic performance.

Each composition tackles a unique mode of interaction including aspects of auditory and visual communication. Though musicians enjoy the experience of working with Shimon in these specific interactive scenarios they often long for additional ways to communicate. For example, performers of Bafana often improvise Shimon will respond to tempo changes or a more improvisatory style of play. Though Bafana was written to be performed in a certain manner, requests of these nature indicate that a truly successful robotic-musician should be able to simultaneously make use of all of these interactive modes such as tempo detection or gesture recognition much like a human does. As we continue to compose and develop for Shimon we will work towards a system that simultaneously embodies many of these interactive features.

7. REFERENCES


A FRAMEWORK FOR THE CHOREOGRAPHY OF SOUND
Gerhard Eckel, Martin Ramori, David Pirró, Ramón González-Arroyo
University of Music and Performing Arts
Institute of Electronic Music and Acoustics
Graz, Austria
{eckel,ramori,pirro,arroyo}@iem.at

ABSTRACT
A framework developed in the context of the artistic research project The Choreography of Sound is presented. The project aims at furthering the practice of electroacoustic composition, especially with respect to the spatial in this genre. The general approach of the project is introduced as a basis for presenting the design of the framework. Its main components are described and the motivations for their design are explained. The framework comprises various tools to formulate spatial aspects of compositional ideas in heterogeneous ways (geometrically, as dynamic systems, and as objects composed from shared properties). Existing spatialisation techniques can be integrated in the framework and exposed to compositional control, such as to allow for unorthodox combinations of them as well as articulating them with other levels of composition. Visualisation and auralisation components allow for sensory off-site in the otherwise very site-specific project. The CoS software framework is conceived as extensions to the SuperCollider audio programming environment.

1. INTRODUCTION
The Choreography of Sound [3] is an artistic research project, which aims at furthering the practice of electroacoustic music composition, especially with respect to the treatment of space in computer music. The inquiry into new possibilities of composing spatial sonic entities is carried out through artistic practice, i.e. the composition of case studies, which explore alternative ways to thinking sonic space through existing spatialisation approaches. As opposed to perceiving spatialisation as solving a problem of representation, CoS understands composing sonic space as a sculptural problem and sound as a plastic object [5] (or even body) to be composed and choreographed.

Spatialisation is often understood as providing a window into a virtual sound space, which is represented by rendering processes trying to emulate the effects of physical sound propagation. In [1] we can read for instance: “The generality of this formulation is motivated by the openness of the research in this project. We cannot know yet which kinds of sonic entities will emerge from our practice-led inquiry. We just know that we do not want to deal only with the notions of sound source objects typical for traditional approaches in sound spatialisation.”

“Spatial audio technologies attempt to find a technical solution to create a spatial sound image.” The notion of the image suggests that there are objects which are depicted or rendered audible through spatialisation. An alternative way of thinking about the spatial in music is to compose sound such that spatial sonic entities emerge in the experience of the listener. This does not only involve what typically is understood by rendering, which mainly addresses the sensory aspect of auditory experience. For spatial sonic entities such as plastic sound objects or space-filling textures (to name only two of those imaginary entities we are interested in) to be experienced, also phenomena on the level of Gestalt perception and cognition (cf. “auditory spatial schemata”, [6]) need to be addressed, i.e. composed.

In CoS, we understand composition as extending onto these levels of sonic organisation. i.e. composition also takes place on the level of the spatialisation technique. Therefore it is not enough to include traditional spatialisation parameters in a compositional framework, such as described in [12] in “spatial sound synthesis”. The spatialisation algorithm itself has to be made subject of composition, opened up in order to be linked to any other level of the compositional work. In CoS, we are composing possibilities for spatial sound entities to be constituted in and through the enactive perception (cf. [8]) of our listeners. The classical distinction between sound space description and rendering is not productive in this context, as the entities populating the sound space cannot be explicitly represented. They are illusions and allusions (cf. [2]) in the body and mind of the listener, which are provoked by compositional decisions on all levels, and not only on the level typically understood as spatialisation.

In our work we are not concerned with rendering sound images for various concert halls using different reproduction systems but with inscribing sound into a specific hall and setup in such a way, that it allows for taking up many different perspectives that expose musical significance. Rather than rendering sound for a sweet area, CoS attempts to create music which can be perceived from multiple angles or listening positions, each of them provoking a different experience, and all of them forming an integral part of the composition.

As a consequence of this approach, the promiscuous type of music presentation may turn out as inadequate: the audience may want to change the listening perspective during the piece. CoS is therefore also exploring different presentation formats mixing concert and installation practices, but these will not be discussed in the context of this paper. This paper rather focuses on introducing the general approach of the project, which forms the necessary background for describing the tools used in the research and for reporting about the challenges of the design and the development of the required software. The general approach of the project is informed by a large number of heterogeneous concepts about the spatial in music and the use of technology for its conception and production, all of which cannot be developed in depth in the context of this paper. Nevertheless, we found it important to introduce at least the most relevant ones briefly.

2. APPROACH
One of the main research activities in the CoS project, the composition of case studies, may be compared to the laboratory practice of molecular biologists, as described by Hans-Jörg Rheinberger in his theory of experimental systems and epistemic things [9]. Rheinberger’s experimental systems are “machines to produce future”. They generate the unknown and allow for the unexpected to emerge. Research questions materialise in the experimental system, they are revealed by research practice. They do not necessarily exist at the beginning of the process. The role of Rheinberger’s epistemic things, which paradoxically represent what we do not know yet, can be compared to the role of the case study in artistic research.

While composing a case study focusing on a particular set of compositional problems, the need for certain kinds of formulations emerges, informing the parallel process of tool design and development. Furthermore, the metamorphosis of epistemic things, once they are understood, into technical objects, which in turn, become new building blocks in the experimental system, can also be observed with case studies – when the knowledge and understanding they embody forms the basis for new works to be created. Grasping this kind of knowledge is the goal of CoS.

Another main research activity of the CoS project lies in the construction and maintenance of the experimental system itself. Shortly after the beginning of the project it has been decided to close it to a particular concert hall and its infrastructure, the Gnörgy Ligeti hall of the MUMUTH in Graz/Austria (cf. section 3). This decision has been motivated by the need to immerse our practice into one situation and its possibilities, being convinced that only such a concentration may lead to the depth of experience and understanding required to advance our research. We accept the fact that our case studies will probably only be strictly valid in this very context, valuing specificity over generality – an approach frequently taken in artistic research.

The Ligeti hall as a central part of the CoS experimental system appears, at first sight, as a technical object in Rheinberger’s sense. Also the existing spatialisation algorithms CoS uses, aiming at combining them in new ways (cf. section 4.4), are technical objects – objects embodying existing knowledge. However, the special infrastructure of Ligeti hall, the large number of available loudspeakers along with an advanced speaker positioning system (described in more detail in section 3) also invites to treat approaches to sound projection as epistemic things.

The representation or display components (cf. section 4.5) are particularly developed for the CoS experimental system. They have to be seen mainly as technical objects as they are based on existing know-how about three-dimensionvalisation and binaural auralisation, tailored towards the particular requirements of the project. But there are also aspects of these components that are typical for epistemic things, representing what we do not know yet. The display tools allow for certain ways of thinking, imagining and manipulating the main epistemic things of our research – spatial sonic entities. How the display tools will eventually condition the research process cannot be foreseen.

Clearly categorised as epistemic in nature can be the formulation tools developed by CoS, which are understood as scaffolds for modelling compositional objects. Among these we find class libraries for representing objects geometrically (such as 3.4.1), typically loudspeakers and reflecting surfaces, but also compositional objects with geometric properties, such as sound sources or clouds. Besides relating objects geometrically, there is a need in CoS for defining the temporal behaviour of objects organised in dynamical systems (section 4.3.2). For other, more general relations to be modelled, CoS pro-
vides an advanced mechanism for dynamically defining and sharing object properties (section 4.3.3).

3. LIGETI HALL

As introduced in section 2, the target place for most of the CoS research is the LIGETI concert hall of the MUMUTH building in Graz. The LIGETI hall is equipped with a large multichannel loudspeaker setup. 33 speakers are mounted on independent, custom-made motorised suspensions, which allow for adjusting the mutual angle and the tilt of the speakers. Their 2D position with respect to the ground floor area, however, is fixed. Four speakers are mounted in the corners of the room, while the other 29, if put to the correct locations and angles, can form a hemisphere suitable for optimised Ambisonics soundfield reproduction.

Furthermore, the LIGETI hall is furnished with a 64-channel artificial room acoustic system (section 3) mounted on the hall’s ceiling and on upper parts of the walls, completed by eight subwoofer channels. All channels can be individually fed, bypassing the acoustical system, which is the primary use case in the context of CoS.

Additionally, there are several speakers of different types mountable on stands at arbitrary positions in the hall, plus two big subwoofers. For some research studies, also the 20-channel IEM Ikosaeder loudspeaker is used [13]. In CoS, these speakers are referred to as “flying speakers”.

All speaker channels are connected to a versatile mixing and routing matrix incorporating a Lawo mc2 66 mixing desk. Two computers, one Macintosh and one Linux PC, are equipped with two MADI interfaces each. Thus, each of them can control 128 channels, either individually, in parallel, or in a chain configuration. Typical CoS case studies may deal with 30–120 independent speaker channels in total.

The ability of flexible speaker positioning very quickly shifts the roles and understandings of designing and using certain loudspeaker setups in general. Rather than achieving “the ideal” setup mainly based on experience and intuition, the sensual experience and evaluation gains much more importance. It also implicitly devalues and estimation, the sensual experience and evaluation using certain loudspeaker setups in general. Rather than ly shifts the roles and understandings of designing and

4.1. Framework

4.1. Role of software development in CoS

The software framework being developed in CoS is conceived as extensions to the SuperCollider audio programming environment. Main reasons for choosing SuperCollider were the program’s openness towards different programming paradigms, its performance and multichannel capabilities, its free availability for different platforms and, not of less importance, the fact that everybody in the team already was familiar with this software.

During the research process in CoS, it became clear quite quickly that the software we want to develop cannot be understood in a one-way functional relationship as a pure “facilitator”. This understanding of software as a tool, which serves as a means for realising a priori conceived compositional ideas, is quite widespread in the community, even if the influence of these “tools” on artistic processes is widely acknowledged in the same context (as an example, see [1]). As an implication, also software development processes (even those taking place in an artistic context) are rarely considered an intrinsic part of artistic or research activities but rather a separate engineering task of implementing assumed pre-existing, though often implicit, service specifications. This is il- lustrated by the ever recurring figure of “developing tools for composers in order to make technology accessible” (as e.g. in aforementioned [1]).

In CoS, we assume two things which contradict this understanding. Firstly, at least in an artistic context, there is almost never a pre-existing catalogue of “service specifications” which could be implemented e. g. by software. Rather, those “specifications” are aspects of an ongoing negotiation between the artists and their means. Therefore, the software development process in CoS is regarded as an intrinsic part of and contributing to the artistic research taking place.

Secondly, the artist is not merely the “user”, but in a certain sense the “expert” who investigates technology as a facet of his very subject. This includes applying and incorporating prior knowledge, existing solutions or standard tools, as done by scientists in their respective areas. This understanding is supported (but not coerced) by the fact that in media art, including computer music, the “artist” and the “engineer” often act in a personal union.

Picking up on Rheinberger, this process might be un- derstood in terms of the abovementioned dichotomy between technical object, be it a “sound source” in the traditional sense or of a more complex form, it may mean an abstract com- positional thing or an object in terms of programming. However, for the scope of this section we stick to the gen- eral term “object”, as its differing meanings in the various contexts still often refer to the same entity. Likewise, the term “space” may denote various things. A space may

be a purely abstract construct, which organises composi- tional objects such as musical layers, but also a model of a real space where objects are organised geometrically. De- pending on the compositional approach, these two might be partially congruent or turn back from the latter, a univer- sal scene graph concept has been adopted from computer graphics, which is described in section 4.3.1.

Apart from describing object relations in terms of dependencies (hierarchies) and observers using the geo- metric representation, dynamics modelling allows to con- struct and evaluate networks of forces between objects, which is described in 4.3.2.

Reflecting the universal understanding of the various roles an “object” might play, its software representation is extended towards a system for the runtime definition, mu- tation and composition of objects from dynamic property sets. This facility is described in section 4.3.3.

4.3.1. Geometric Representation

As a starting point for CoS research, a means for repre- senting the geometric relations of objects was sought. In the first place, the idea was to construct a model of the MUMUTH LIGETI hall (cf. section 3), its playback capa- bilities (loudspeakers) and their layout, and compositional entities such as sound sources. This should serve as a generic basis for the application of common spatialisation approaches. For many standard or non-standard spatialisation algorithms, different parameters of those geometric relations need to be computed, such as the distance of a sound source to each of the speakers for DBAP (Distance Based Amplitude Panning) [7].

On the other hand, it might be desirable to express geometric object relations in hierarchic structures. This may include the grouping of certain speaker arrangements to a single entity, as in the case of the 20-channel IEM Ikosaeder speaker (cf. section 3), but also modelling spa- tial dependencies of sound objects, such as multi-source objects or swarms. Based on these findings, as a generic representation for geometric relations a scene graph model of three-di- mensional hierarchial coordinate systems has been imple- mented. As opposed to computer graphics, the scene graph approach is not primarily a means of optimising for rendering but rather of structuring conceptual relations. Furthermore, it provides the basis for a largely uniform interface to different spatialisation techniques.

An initial attempt of including a fourth (e.g. temporal) dimension with the geometric representation was dropped so far. It turned out that the various modes of dealing with time in music composition cannot be easily subsumed under a single extended geometry. This implies a substantially narrowing the practical access to temporal pro- cesses.

The adopted scene graph approach within the CoS framework is one of including a fourth (e.g. technical ob- ject, adopted from computer graphics, and subsequently turned into an epistemic thing. This object is characterised
by the confrontation of well-known computer graphics technology with a possible way of formulating computer music, combined with an expectation that this might reveal knowledge on spatial composition which has been only implicitly before.

4.3.2. Dynamics Modelling

From the experience gained in the first CoS case studies, the necessity emerged to extend the static, geometric links between objects by relations that could alter the objects’ properties or their state over time, i.e. dynamically.

In our thinking, these alterations are caused by effects or “forces” objects exert on each other. In the dynamics modelling subsystem every object contains a property describing how its effect applies to others and a record of all objects affecting it. Objects that are interconnected by these forces form a system. Systems evolve by triggering all the effects and applying the changes to all objects at once, setting them into a new state. Furthermore, each system advances at its own pace, allowing to have parallel systems developing at different rates.

For instance, with these extensions, particle-based physical models can be implemented and simulated. In this case, the properties on which forces act are the objects’ position, velocity and spatialisation of the sound source along its path. The dynamical system generates motion, a spatial and temporal behaviour, eventually choreographing the appearance of sound in the Ligeti hall.

By these two perspectives on dynamics modelling can be made evident. From one point of view, interactions take effect on the sound source, pushing it into motion. From the other, the forces fields form and shape the space in which the sound source lives. Here, movement is the result of the source “exploring” valleys dug and mountains raised into its space by the forces, trying to reach its equilibrium. In practice, both points of view are equivalent.

4.3.3. Generic Objects with Hierarchic Properties

Finding a representation of more generic objects is not as straightforward as in the case of geometric objects or those designed for dynamics modelling. As different kinds, uses and functionalities of such objects are expected to arise during the research process, it became clear that a flexible object framework is needed that can be extended and adapted according to the developing needs and applications.

The current attempt of such a framework are objects that are characterised by a set of properties. These are basically dictionaries of key/value-pairs, which may contain both primitive (e.g. numbers) or complex objects as well as functions. Property dictionaries can be used locally for a single generic object, but they may also be shared among multiple objects. As property dictionaries involve a notification mechanism to the objects referred to, this can be used to update multiple objects upon changes of shared property sets.

Property dictionaries may in turn contain property dictionaries, such that hierarchically structured property sets can be constructed. On any level of such a hierarchy, a property dictionary may be shared or private. Even subtrees of shared dictionaries can become private again, which is made possible by a sophisticated yet entirely transparent per-object-handling of subtrees.

The CoS framework is completed by convenience functionalities such as read-only property sets or optional copy-on-write semantics for an automatic transition of shared dictionaries to private ones.

The open design of the CoS property system allows for specifying objects using different design patterns in a fully dynamic way. Template property subtrees may be referenced and further specialised in order to furnish an object with certain functionalities. This is similar to object oriented concepts like multiple inheritance, interface/protocol definitions or object composition whose application happens only at runtime.

In CoS, several building blocks of the software framework are interfaced to the objects by such sets of properties. For example for simple filters, how an object is graphically represented is conceived as templates for property subtrees (cf. section 4.5.1). Certain spatialisation algorithms that evaluate further data beyond geometric coordinates are fed by property dictionaries (section 4.4), as well as objects’ characteristics in the context of dynamics modelling (section 4.3.2).

4.4. Spatialisation Algorithms

In CoS, most spatialisation algorithms operate on the level of the geometric representation. Usually implemented as a geometric object itself, a spatialisation object is associated with a set of other geometric objects. For example, in the case of the DBAP, the associated objects represent the speakers whose distances are used to calculate the amplitude weights, while the DBAP object itself represents the points of the geometric source. Other algorithms may take into account further objects such as listeners, absorbers or alike.

As geometric objects, subtrees of generic CoS objects, they also provide hierarchic properties (section 4.3.3). Spatialisation algorithms may evaluate subtrees of this property sets in order to allow for specific functionality that cannot be controlled by geometric coordinates only, such as radiation patterns. Even entirely abstract (non-geometric) objects may be used, e.g. for controlling larger groups of spatialisation processes.

The integration of spatialisation algorithms with the objects they are related to allows for easily replacing or manipulating spatialisation processes. Also, many of those may happen at the same time, using independent or other kinds of spatialisation sources or abstract compositional objects. The combination of a three-dimensional geometric representation and the versatile hierarchic property system facilitates the efficient evaluation of standard spatialisation algorithms while advanced techniques operating on more complex data may still be conceived within the same framework.

However, this uniform interface to different spatialisation methods allows for a more flexible decision towards a separation of an auditory scene description (sources, trajectories etc.) and the spatial rendering, as claimed by many spatialisation projects on standardisation attempts (for an example, cf. [12]). Although it is possible to use the software framework following this paradigm, and although in some cases of spatialisation this might be preferable, in the CoS framework spatialisation is regarded as an intrinsic part of musical composition which generally cannot be separated from other kinds of sound synthesis or processing (cf. section 1).

This is underlined by the fact that spatialisation algorithms in the CoS framework do not operate directly on the digital signal processing level but are rather used as parameter generators for a separate signal processing stage. In other words, how and at which level these parameters are applied. For many spatialisation techniques, however, the signal processing actually taking place boils down to amplitude weighting and possibly the application of spatial filters. The spatialisation algorithms in the CoS software so far are implementations of well established sound diffusion techniques such as DBAP and a variant (section 4.4.1), a simplified WFS approach (section 4.4.2) and prospective interfaces to Ambisonics spatialisation (section 4.4.3).

It is, however, not the goal in CoS to have a complete suite of spatialisation algorithms. On one hand, those algorithms can be easily implemented the “soft way” without appearing as an object class of its own, e.g. using properties and notifications of SuperCollider functions. This is actually the case for evaluating several non-standard algorithms. On the other hand, extending the library of geometric spatialisation objects is very easy, such that they can be added on demand when they are needed.

4.4.1. DBAP and ADBAP

As the DBAP amplitude panning algorithm makes no a priori assumption of the effective loudspeaker setup, this spatialisation method was a sort of natural choice in the CoS case studies as it best fits the project’s general approach.

The DBAP algorithm implemented in the CoS framework is a three dimensional extension of the technique formulated in [7]. Having specified the rolloff and blurring coefficients, the distances of the source to the associated loudspeaker objects are computed and used to determine the relative amplitudes of the projected sound.

Based on the principle of constant intensity panning, the original method requires that the squares of the amplitudes sum up to one. The resulting overall intensity is constant, regardless of the position of the source. As a consequence, positions far away from the loudspeakers cannot be clearly rendered: in this region the relative amplitude differences tend to zero with increasing distance while the overall intensity is still kept constant, resulting in a spatially undifferentiated sound output.

However, in the course of our case studies, it turned out necessary to spatialise sources that could also travel out of the loudspeaker field and completely disappear. To achieve this, we modified the DBAP algorithm removing the constant intensity condition: sound spatialisation is achieved defining a distribution of absolute rather than relative amplitudes. This causes sources that move sufficiently far away from the loudspeaker array to fade out.

Furthermore, we discovered that especially the trajectory of moving sounds (as in the example described in section 4.3.2) appears more clearly shaped or “sharper”, compared to the unmodified DBAP algorithm. Lacking a more explanatory name, we call the simplified version of the DBAP algorithm Absolute Distance-Based Amplitude Panning (ADBAP).

4.4.2. WFS

At this point the CoS framework offers a simplified implementation of the WFS approach (section 4.3) for experimental purposes to Ambisonics spatialisation (section 4.4.3).
function of the plane wave orientation and the speaker positions. It is noteworthy that the implementation of the algorithm based on the underlying geometrical representation takes only four lines of code. This illustrates that existing spatialisation tools can be added easily and whenever demanded from a compositional point of view. The structure of the framework supports the conception and implementation (i.e. composition) of new techniques and the experimentation with them—especially if they exploit geometrical relationships between loudspeakers and spatial or compositional (i.e. conceptual) entities. In the process of developing CoS case studies, this WFS implementation was used to perceptually estimate the effects of irregularly spaced loudspeaker arrays exhibiting spatial aliasing due to their sparsity (cf. [4]).

4.4.3. Ambisonics So far, spatialisation with Ambisonics does not play a major role in CoS. This may be derived from the principal research approach as described above, since Ambisonics is a holophonic soundfield reproduction technique, which is restricted to a sweet spot or area. However, it is still desirable to have Ambisonics available for approximating it in an experimental way using subgroups of closely related speakers, under-determined or non-ideal speaker layouts, or making various, often hidden parameters of Ambisonics encoding and decoding available for composing.

A basic interface to existing Ambisonics implementations such as AmbiHEM can be easily realised within CoS: geometric parameters like relative orientation of objects (i.e. azimuth and elevation) are available from the geometric representation (cf. section 4.3.1) while there are no assumptions on the actual signal processing.

For further experimentation it is desirable to have access to advanced case study tools as mixed-order systems, distance coding or multi-band decoding. Prior research has been undertaken to design such a framework for SuperCollider, called Girafe [10], that benefits from both the powerful features and from the multi-channel routing capabilities of the server. Promoting a separation of signal processing and control data generation itself, it nicely integrates with the design principles of the CoS framework, but its integration is not yet finished.

4.5. Display Tools 4.5.1. Visualisation Subsystem For taking up different perspectives, a visualisation subsystem with interaction capabilities is being conceived. It consists of a precise three-dimensional model of the Liger hall including the speaker lifting mechanism and models of the other speakers used in the project. For displaying and navigating the model, the Blenderootnote{http://www.blender.org, last retrieved 2012-02-20} game engine is used, which is controlled via OSCootnote{http://oscsoundercontrol.org, last retrieved 2012-02-20} from the CoS software system in SuperCollider.

The visualisation system of CoS attempts to fulfill several functions. Firstly, it serves as a kind of display and debugging tool by visualising the geometric layout of certain constellations defined on the textual side of the CoS system. It is also used for creating documentary screen shots or videos.

Moreover, the access to Liger hall is limited, the visualisation system also serves as a means for working on case studies outside the actual space. Due to its common acceptance, there is an established understanding on how a three-dimensional visualisation relates to the pictured reality. In CoS, we assume that this relation can be exploited to support the composer’s abstract imagination of a certain space, which works on the level of experience and retrospection rather than that of a simulation.

In another way, the visualisation system is also used as a construction tool, e.g. for designing new loudspeaker setups (cf. figure 1). Some of the setups used in CoS case studies are designed by composing first-order wall and floor reflections of imagined on-axis speaker radiation at the same level as direct sound from the speakers. The model here serves for visualising and estimating these propagation paths in a simple way. This must not be mixed up with fully fledged auralisation software or room acoustic simulations tools, which would be far beyond the needed degree of detail of CoS.

4.5.2. Virtual MUMUTH – Binaural Auralisation Another display in the CoS framework is provided using binaural projection of the measured Liger hall acoustics. As for the visualisation subsystem, the motivation behind the binaural auralisation is not a simulation allowing for a perfect immersion, instead and again, it is meant to take up a different perspective on the acoustic and audiovisual processes taking place in the real space. It is therefore also an investigation on the viability and validity of using such simulation tools for the process of composing music.

Virtual MUMUTH is realised using sets of measured binaural room impulse responses (BRIR) in a multichannel convolution matrix [11]. Every set of those impulse responses corresponds to only one loudspeaker setup and only one listening position and orientation. Therefore, no advanced techniques from binaural simulation systems can be applied, such as rotation compensation using head tracking, and the auralised configuration is mostly static. On the other hand, using BRIRs preserves the actual acoustic properties of the Liger hall as a result of communications engineering to a much larger extent than common room simulations. The direct comparison of the binaural auralisation and the real acoustics in informal on-site listening tests showed that the binaural auralisation successfully captures the loudspeaker setup and other relevant objects in the hall (e.g. reflectors). The binaural auralisation possibility, together with the feedback of a loudspeaker setup allows for a sensible work outside of the Liger hall, given that one has experienced the respective case study in the hall at least once before listening to it through the auralisation.

The CoS software framework will be made freely available as an extension library to SuperCollider on the CoS website.ootnote{http://cos.kug.ac.at, last retrieved 2012-05-23} Additionally, the three-dimensional model of the Liger hall, several sets of binaural room impulse responses and case studies will be available at the same place.

6. ACKNOWLEDGEMENTS The CoS project was conceived by Gerhard Eickel and Ramón González-Arroyo. Together with Martin Rumori and David Pirró they form the core team of the project at IEM in Graz, maintaining links to researchers at BEK in Berlin, IRCAM in Paris, and also to international tone-meister conventions, such as the 9th Int. Conference on Digital Audio Effects (DAFx-05), 2006.

6. REFERENCES


SpatOSC: PROVIDING ABSTRACTION FOR THE AUTHORING OF INTERACTIVE SPATIAL AUDIO EXPERIENCES

Mike Wozniowski, Zack Settel†, Alexandre Queossy, Tristan Matthews, Luc Courchesne
La Société des arts technologiques [SAT]
Montréal, Québec, Canada
†Université de Montréal, Faculté de Musique
Montréal, Québec, Canada

ABSTRACT
Creating an interactive project that deals with 3-D sound becomes difficult when it needs to run in multiple venues, due to a diversity of potential loudspeaker arrangements and spatialization systems. For effective exchange, there needs to be a standardized approach to audio spatialization with a unified, abstract way of representing 3-D audio scenes. Formats from the gaming and multimedia communities are appealing, but generally lack the features needed by artists working with interactive new media. Non-standard features need to be supported, including support for multiple listeners, complex sound directivities, integration with networked show control and sound synthesis applications, as well as support for the tuning spatial effects such as Doppler shift, distance attenuation and filtering. Our proposed solution is an open source library called SpatOSC, which can be included in existing visualization engines, audio development environments, and plugins for digital audio workstations. Instead of expecting that everyone adopt a new spatial audio format, our library provides an immediate solution, with “translators” that handle conversion between representations. A number of translators are already supported, with an extensible architecture that allows others to be developed as needed.

1. INTRODUCTION
In the domain of interactive new media, there are no truly standardized ways of representing the spatial characteristics of sound. As a result, there exist a multitude of different audio formats, editing tools, and panning/rendering environments for 3-D audio, and it is quite difficult for a performance or installation to be adapted to different venues. The traditional approach has been to use a channel-based representation, where sounds are rendered down to a number of digital audio channels that can be directly reproduced on a known loudspeaker arrangement. The most common format for this approach is 5.1 channel surround sound, which is a pantality arrangement (all loudspeakers are on the same plane) only capable of reproducing horizontal spatial audio effects. Although formats exist for periphrastic (3-D) arrangements, like the 22.2 channel system for ultrahigh-definition TV [4], the general drawback of channel-based representations is that there is no flexibility for experimenting with loudspeaker placement.

The other major drawback is the difficulty of interactive control required by artists, since sounds that are already encoded in raw audio channels cannot be separated. Ambisonic formats [3] are perhaps an exception to this rule, offering a channel-based representation that can be decoded and interactively panned. However, a majority of surround formats are only effective for fixed media production (i.e., non-realtime playback systems) with an authoring stage that is separate from the presentation of the work. For interactive art performances and installations, where sound movements are controlled in real time, an object-based representation is more effective [2]. Instead of raw audio signals, sounds are described as virtual sound sources in 3-D space, and their positions can be manipulated in real time, providing artists with new ways to think about, and work with sound.

While object-based representations offer more flexibility in terms of interactivity, they too come with certain challenges and drawbacks. An application known as a “spatializer” must be able to render the virtual audio scene into corresponding audio signals for a particular loudspeaker arrangement. This of course must be done in real time, which can take significant computational resources for large systems.

There are several computation techniques for such rendering [1]. Binaural systems can render a scene for headphones using filters that are ideally tuned to a listener’s head shape. Vector base amplitude panning (VBAP) can produce fairly accurate rendering in the middle (“sweet spot”) of equidistantly spaced loudspeakers. Conversely, wave field synthesis (WFS) can provide volumetric simulation of a sound field with no sweet spot but requires a huge number of loudspeakers and thus significant computational resources.

Given this variety and the fact that there is also no standard way of describing a spatial audio scene, a variety of spatializers exist, each with different representations and constraints. Some spatializers only perform basic sound source directivity, acoustic spaces, distance attenuation, and constraints. Some spatializers only perform basic sound source directivity, acoustic spaces, distance attenuation, and constraints. Some spatializers only perform basic sound source directivity, acoustic spaces, distance attenuation, and constraints. Some spatializers only perform basic sound source directivity, acoustic spaces, distance attenuation, and constraints. Some spatializers only perform basic sound source directivity, acoustic spaces, distance attenuation, and constraints. Some spatializers only perform basic

However, SpatDIF is just a format, requiring that control applications and spatializers implement a “SpatDIF Interpreter” to read/write or send/receive appropriate messages. While there are a few implementations (e.g., Jamoma, ICST’s Ambisonics Tools [6], all other spatializers will need to adopt the format in order to become a successful standard. What makes this difficult is that many high-end spatializers already have an OSC control system that allows external control of parameters. Examples like D-Miri from Meyer Sound, Spatialisateur from IRCAM, and Zirkonium from ZKM already have many users and existing tools that work with these systems, creating substantial inertia to change.

We believe that a better solution is to introduce the translation on the send side. We assume that spatializers have their own custom formats that are resistant to change, so we provide a library that can learn those formats. The controller or editor application can link with this library and use a single API for communicating with any spatializer that is known by the library.

2. BACKGROUND & RELATED WORK
At the International Computer Music Conference (ICMC) in 2008, researchers from around the world met to discuss exchange formats for spatial audio scenes [9]. The main development of this panel was the presentation of SpatDIF, a spatial sound description interchange format [5]. The format uses OSC messages to communicate both the general properties of the scene (coordinate system, units, etc.) and control messages for dynamically changing sound source positions. The assumption is that there is a lower common denominator for describing an audio scene that is common to all spatializers. Any SpatDIF-compliant renderer should be able to understand these “Core Descriptors” and produce spatial sound to match the desired effect.

A controller or editor application thus only ever needs to know one format or language, and sends messages with the only content to know nothing about the output system in advance. For more complex behaviours, SpatDIF has a number of “extensions,” to specify this functionality in 2D or 3D. The controller or editor application thus only ever needs to know one format or language, and sends messages with the only content to know nothing about the output system in advance. For more complex behaviours, SpatDIF has a number of “extensions,” to specify this functionality in 2D or 3D.
tensile format and extensions could be written to support these features.

In fact, the AudioBIFS format from MPEG-4 [8] is an extension of X3D with a focus on describing audio-visual scenes in a compact and object-oriented fashion that ultimately leads to streamable interactive content. A scene graph hierarchy is used to organize nodes, and specialized nodes for audio processing are defined, such as ‘Sound,’ ‘AudioSource,’ ‘AudioFX,’ and ‘ListeningPoint.’ Recently, ‘DirectivSound’ has been added, allowing for directional sound sources, as well as ‘WideSound’ which provides a description for a sound source that occupies a larger volume. Likewise, parameters for modelling acoustics and acoustic materials have been added [10]. BIFS (which stands for Binary Format for Scene Description) is binary, and must be authored separately and then compiled for distribution. Interaction is accomplished by allowing certain fields to be exposed and updated by the content server, but this control is not implicit. Rather, the developer must pre-define all possible interactions during the authoring phase, which limits the possibility for live experimentation.

In contrast to the previously mentioned formats, ADSF [2] is primarily intended for musical purposes. It is an XML-based extension to SML (Synchronized Multimedia Integration Language), which focuses on temporal control and synchronization of audiovisual content. ADSF adds spatial functionality and although it still lacks a lot of features, the attention to timing and synchronization is important for musical composition. In contrast, formats like X3D provide little or no means to reorganize sequences of events, since timing is abstracted through routings between scripts and processes.

### 2.3. Existing Libraries

For software developers working in the fields of 3-D games and virtual reality, there exist several libraries that manage virtual sound source simulation. Examples like FMOD, Wwise, irrKlang, DirectX and OpenAL provide realistic real-time rendering of sound sources, and are easily integrated into existing development environments. Unfortunately, these libraries are not really geared towards musical or highly interactive control of sound synthesis, and it is difficult to combine these libraries with external synthesizers or programming environments like Max/MSP, SuperCollider, or Pure Data. Many APIs also have no method for describing sound sources and consider all sound sources as omni-directional. In the cases where directional sounds are supported, these are typically implemented with linear attenuation applied as a function of angle. There is usually no method for the complex radiation patterns of traditional musical instruments, or the cardoids that are commonly found in audio equipment. Furthermore, there is often only support for a single listener and standard audio formats (e.g., stereo or 5.1 channel surround). Arbitrary speaker configurations are rarely supported, and the listener is usually assumed to be wearing headphones or sitting in a centralized sweet spot that is equidistant from all speakers.

In most cases, these APIs also provide hardware abstraction and directly send computed sound signals to the sound card, which may be desirable for a game programmer, but limiting to a sound artist. Our approach instead assumes that the audio hardware is controlled by a separate process, often on a separate machine, and that OSC is used to send spatialization parameters.

### 3. THE SPATOSC LIBRARY

A system that uses SpatOSC is composed of a host application (typically written in C++ or Objective-C), and an audio renderer (that provides physical output to loudspeakers). These are independent components, which may or may not be located on separate machines, and which communicate via OSC in order to update spatialization parameters.

Figure 1 gives an overview of how SpatOSC integrates within an interactive system. This example is typical of most virtual reality systems or 3-D audiovisually installations, where the host might be a plugin for a DAW, an external for Pa or Max/MSP or any engine or virtualization software built using libraries like Ogre, OpenSceneGraph, or Unity 3D. The host application maintains the state of a virtual scene, including spatial information about audio-related items. The SpatOSC library provides an API to the application that allows for the creation and updating of these nodes. Whenever a 3-D sound source is moved, a function call is made to SpatOSC and the state within the library is updated.

One of the most important aspects of SpatOSC is the translator system (see Section 3.6). One can think of a translator as a plugin for the host application, which allows for communication with a particular third-party audio spatializer. A user chooses which translator to use depending on the system configuration, and typically only needs to specify the remote host name and port to which OSC messages are sent.

#### 3.1. SpatOSC Internal Representation

Internally, SpatOSC maintains its own representation of the scene, with a simplified audio scene description containing elements common to all spatializers: sound sources (which emit sound into the scene) and listeners (which capture sound at a particular location). A listener may be thought of as the virtual representation of a loudspeaker or group of loudspeakers. In the case of something like a 5.1 channel surround system, all loudspeakers combine to simulate the sound field at one location in the scene, so they are grouped together and thought of as one listener. However, there may be cases where loudspeakers actually move during a performance, like when multiple sets of headphones are tracked with a motion tracking system. In such a case, multiple listeners need to be defined. Another example is an installation without a centralized sweet spot, such as a long sound wall, where each loudspeaker provides a localized rendering of a certain portion of the wall. In such a case, a listener would need to be defined for each loudspeaker.

Sound sources on the other hand are generative, and emit sound energy into the virtual scene at particular locations. Their signals may derive from pre-recorded sound files, remote streams, realtime sound synthesis, or live input signals from audio hardware or software busses. The following properties may be associated with sound sources and listeners:

- **Spatial Pose:** Both sources and listeners have 6 degrees of freedom: position in cartesian (x, y, z) space, and they have an orientation which can be described either by a direction vector, Euler angles (pitch, roll, yaw), or quaternions. Positions are always defined relative to a global coordinate system.

- **Radius:** A radius parameter allows a sound to occupy a volumetric region instead of acting as an infinitesimally small point source. If a listener node enters within this radius, all spatial effects are disabled and the sound plays naturally, with unity gain and no filtering. A similar parameter is often found in sound party systems under names like spread, diffusion, blur, etc. By default, the radius for all nodes is set to zero.

- **URI:** For sound sources, the URI describes the media type and location emitted at that location. This could be a simple sound file reference (file://loop.wav), a live input channel on the soundcard (adc://1), a stream (http://stream.mp3), or a plugin (plug://looper-pool).
- **Extension:** Each node can also be extended with arbitrary key-value parameter pairs that describe additional features that may be required by specialized systems.

#### 3.2. Connections

One of the biggest differences between SpatOSC and other spatialization systems is that sound sources are not necessarily connected to every listener, and there is an explicit connection class that is used to describe both the logical transfer of sound from one location to another, and also the physical modelling effects involved in the transfer. The CONNECTION contains information about distance, attenuation of gain resulting from sound field propagation, the time delay incurred for sound to travel that distance, and other physically-modelled properties.

A benefit of maintaining explicit connections includes the ability to temporarily disable sections of the scene, or to provide unique sounds to listeners without hearing them (even if they are close by). However, the main benefit is the ability to customize spatial effects for some connections differently than others.

#### 3.3. Manipulation of connection effects

In order to provide artists with flexibility for experimentation and non-standard audio scene interaction, the CONNECTION object provides the ability to tune (enable, disable, scale) several spatial effects, and provides fine-grained controls for spatialization that is not typically possible with other libraries. The DISTANCEFACTOR specifies the extent to which distance attenuation and absorption effects should be applied. The DIRECTIVITYFACTOR provides a parameter to scale the effect of sound directivity. And the DOPPLERFACTOR allows for Doppler shift to be minimized or emphasized, which is particularly useful in order to preserve timing and intonation in musical contexts.

#### 3.4. Internal Conversions & Computations

Given that SpatOSC needs to be able to supply a number of different parameters to different types of spatializers, we often need to convert, scale, or apply transformations to the internal data. Consider for example, the gain attenuation of a signal as a result of distance. Amplitude panning systems (VBAP, etc.) may only do panning and
do not support simulation of distance effects. For convenience, SpatOSC computes a gain coefficient for each connection using the following formula (where A and B are the positions of the source and listener):

\[ g = \frac{1}{(1 + |B - A|)^2} \]  

(1)

The result, in the range of [0,1], represents the amount by which the amplitude of the signal should be scaled to approximate the decay of sound due to distance. The DISTANCE FACTOR, \( \beta \), helps to control the steepness of the exponential decay. With this parameter, one may gradually transition from zero distance attenuation (\( \beta = 0 \)) to the inverse square law that sound intensity obeys in nature (\( \beta = 1 \)), and even beyond in order to create hyper-localized sounds (\( \beta > 1 \)).

There are also helper functions available to convert between units. For instance, one might want to know the gain value in decibels rather than amplitude gain, or the position of a sound source in radial units (AED: azimuth, elevation, distance) instead of Cartesian units (x, y, z). One might also require sound source positions to be relative to a listener instead of the global coordinate system. In such a case, the CONNECTION object can be queried and the relative radial or Cartesian coordinates can be retrieved.

Conversions between coordinate systems are also supported. SpatOSC provides transformations for the entire scene that allows conversion between right- and left-handed coordinate systems, flipping axes, and rotating the global coordinate system.

3.5. Directivity

The representation of sound source directivity is potentially complex, and not surprisingly, one of the least standardized aspects of spatial audio scene description. In fact, most spatializers only support omni-directional sound sources, while others represent directivity parameters with parametric functions or simplified models such as cones or ellipsoids. The goal of SpatOSC is to be able to store directivity in such a form that it is convertible into any of these other formats.

For SpatOSC, we use axisymmetric attenuation tables that specify sound intensity at different angles from the source sound source’s vector. These tables are used for both the horizontal and vertical directions, meaning there are two attenuation tables representing orthogonal angles from 0 to 180 degrees. We use variable sampling instead. There are two attenuation tables representing orthogonal angles from the horizontal and vertical directions, meaning there is no meaningful 3-D situation the effect of fully open azimuth span and no zenith span creates a “ring” of sound on a horizontal plane, meaning that sound is emitted uniformly from all speakers at a certain height. This situation has no meaningful 3-D representation in SpatOSC, so it is useful to use extension properties to store the azimuth and zenith span values for every node. The Zirkonium translator thus knows how to deal with those specifically-named properties, while other translators simply ignore these parameters. It should also be noted that multiple translators can be assigned at once, making it possible to simultaneously represent scenes on different spatializers. This is useful in situations where multiple listeners are exploring the environment.

3.7. The SpatOSC Audio Unit plug-in

Given that the SpatOSC library is written in C++, it is generally quite easy to include the library within a plugin for a DAW. As of writing, only an Audio Unit (for OSX) has been developed, but the intention is to also develop VST (Windows) and LADSPA (Linux) plugins.

The Audio Unit can be added to a track in a composition, and provides parameters and a graphical user interface (GUI) for moving sounds. In the most common setup, a track’s signal output is sent directly to an input channel on the spatializer while OSC messages send corresponding control signals as to how that input should be processed. SpatOSC has no methods for storing trajectories or sequencing spatial parameters, so one of the best benefits of using SpatOSC within a DAW is the built-in automation system used to record and playback trajectories for moving sounds.

4. CONCLUSION & DISCUSSION

We have discussed the need for an abstract audio scene description mechanism, geared towards musicians and artists. By exploring the diversity and complexity of various spatialization techniques and implementations, we note that it is a difficult task to create a standard spatial audio format. Instead, we have developed an open source software library with an extensible architecture based on translator plugins, which can accommodate a range of technologies. Networking via OpenSoundControl allows for a separation of tasks, letting spatializers do what they do best, while providing flexibility for the integration of spatial audio content into a number of different host applications. SpatOSC currently has no way to model the environmental effects of a 3-D scene; further work in this area is needed in order to achieve realistic spatial audio effects. Further attention also needs to be paid to sequencing or timed playback of material. However, the utility of SpatOSC for real-time interactive work is clear. We have already created audio installations that easily play on multiple types of spatialization hardware. As more translators are written, interchange between systems will increase, hopefully leading to more experimentation with 3-D sound throughout the artistic community.

5. REFERENCES

CYCLICAL FLOW: SPATIAL SYNTHESIS SOUND TOY AS MULTICHANNEL COMPOSITION TOOL

Andrew Dolphin
Leeds Metropolitan University
Leeds, UK
a.dolphin@leedsmet.ac.uk

ABSTRACT

This paper outlines and discusses an interactive system designed as a playful ‘sound toy’ for spatial composition. Proposed models of composition and design in this context are discussed. The design, functionality and application of the software system is then outlined and summarised. The paper concludes with observations from use, and discussion of future developments.

1. INTRODUCTION

Cyclical Flow is a multichannel spatial composition tool and sound toy. The system was originally realised to create spatial sound materials for composition. The project is a real-time spatial composition or spatial performance system, in which spatial, textual and spectral parameters are interlinked. The player (composer) creates and modifies varied forms of spatial and spectral patterns or trajectories in real-time. Spatial, spectral and granular processing techniques are implemented and codependent. The system is intended for use as an exploratory and playful tool for spatial composition and performance in multichannel concert or studio spaces.

2. CONTEXT

The project is part of a series of sound toy systems [1] designed and implemented that explore compositional system development and design incorporating audio-visual interfaces [2][3][4], with symbolic interactions determining sonic output. These works explore sound toys as open work, composition medium and compositional tool, and are a collection of interactive sonic centric audiovisual systems, designed and influenced by aesthetics, processes and techniques familiar to the field of electroacoustic music, and informed through the realization of fixed media and multichannel electroacoustic compositional works.

These sound toy systems are designed according to a proposed model for composition (Figure 1). Compositional input is considered to be multidimensional, and the importance or significance of each input form as an element of composition is somewhat open to interpretation. Compositional input may be attributed to three primary forces or agents, each dictating or influencing characteristics of the output.

Composer/Designer - Offline
User/Player (Composer) - Real-time
Simulated Physics - Real-time

The composer/designer is responsible for designing and creating the framework for composition, making compositional decisions during the construction of the system. Modes of interaction, sound materials, transformation processes, compositional options and constraints, and modes of presentation and representation are all determined by the composer/designer. The user/player engages with the system in real-time, responding to both visual and aural feedback from the system. There is often a codependence between the human player and a simulated physics system. This acts as a third compositional agent, adding an algorithmic component to the system.

The algorithmic component is accompanied by symbolic representations of the algorithmic processes in the virtual visual space. These provide the user/player with some insight into this aspect of the control system, which is enhanced through play, exploration and learning. Symbolic representation of a simulated physics system allows real-time interaction between the user/player and the system in both visual and aural domains, also allowing anticipatory responses that enable the user/player to react to forthcoming events. These two compositional forces (simulated physics system & user/player) influence each other throughout play.

3. OVERVIEW

In Cyclical Flow, dynamic movement of sound through space is the central theme. Symbolic kinetic approaches in the visual domain represent and control spatial motion in the sound domain. The movement of symbolic objects within a virtual representation of a multichannel performance space directly control patterns of spatial motion and spectrum of sound. The visual component of the project attempts to provide a playful interface for spatial composition or “musicking” [5] [6]. The project is to some degree influenced by phase music process. The player may take a monophonic single line approach if they choose, but the system is designed to allow the creation of layered cyclical patterns of motion, where the relationships of each line/part shifts over time due to the use of different cyclical motion rates. This results in a form of spatial counterpoint [7] and spatial interactions between multiple spatialized sound materials.

4. SPATIAL MOTION

Ideas presented by Wishart (1996) on spatial motion are considered relevant to Cyclical Flow, and can be related to specific features of the system. These features are not necessarily direct realizations of his ideas. However, techniques he describes can be aligned with some of the spatial motion and spatial approaches adopted. Wishart’s writing on spatial motion [7] includes a number of diagrammatic depictions of spatial trajectories, and a number of his representations of spatial contours and trajectories are achievable with the Cyclical Flow system. Theories presented by Wishart that are of particular significance include “time contours” [7]. Here he describes how “motion is characterized not only by its path in space but also by its behavior in time”. He goes on to define “first order time properties (different speeds of motion) and second order properties (the way in which the speed changes through time, the acceleration or deceleration of the motion)”. This is relevant to the twenty-seven easing types implemented, allowing the user to determine acceleration and deceleration of spatial trajectories.

In Cyclical Flow, symbolic systems for setting and adjusting trajectories and spatial pathway nodes (which the spatialized sound objects move through) are particularly relevant to Wishart’s definitions of “frame”, and “frame motions”. The software system developed also facilitates a number of the frame transformations he describes. Simple quasi-generative systems implemented in Cyclical Flow for modulating trajectories and pathways are also relevant to Wishart’s definition of “frame motions”. See “frame rotate” for example, which represents one-dimensional frame motions. Frame rotations are implemented as an optional automated system, allowing the pathway of each sound object to be modulated at different speeds and in different directions, creating shifting spatial trajectories. Whilst there are distinct interrelationships between the features implemented in Cyclical Flow and the spatial motion techniques described by Wishart, it is important to note that the system is not a direct implementation of all the spatial motion behaviors and theories he presents.
However, a number of spatial techniques he describes can be related to specific features and intended spatial applications of the Cyclical Flow system.

5. SPATIAL & SOUND ENGINE STRUCTURE

The sound engine uses three primary sound sources for each Spatial Object. These may be used independently, or combined (mixed) to create richer textures. All three primary sound sources for each Spatial Object follow identical spatial trajectories. The three sound sources are an interpolating phase vocoder, [9], granular engine and simple sample player.

The developed game engine [10] application is a tool for creating real-time coordinate and control data, which is used to control an external spatial sound and synthesis engine constructed in Max/MSP/Jitter [8]. Coordinate data is mapped to a number of parameters. These include:

- Read position and interpolation of spectral data re-synthesized by phase vocoder.
- Read position of the granular engine, creating a granular time-stretch effect.
- Spatialisation of sound sources.
- Spatialisation of a reverb (two channel), using delayed coordinate data to create spatially reverbent sound effect.
- Speed of spatial motion determines pitch modulation, simulating Doppler effect.

Simultaneous mapping of coordinate data to both spatial and sound generation parameters creates a direct link between the sonic characteristics of the output, and its spatial behavior, as timbral development and spatial motion are interlinked when using the granular engine and phase vocoder sources. This technique, when combined with the simulated Doppler effect, provides a range of creative possibilities for spatial composition. ICST Ambisonic Tools [11] are at the foundation of the spatialisation system, using Neukom & Schacher’s Ambisonic Equivalent Panning [12] technique to reduce computational load. Mapping of Cartesian coordinates from the spatial composition play space to the spatial sound engine is made simple due to ICST’s ambisonic GUI object [11] accepting both Cartesian and spherical data input, so Cartesian coordinates output from the game engine only require smoothing and scaling before being mapped. It should be noted that the reverber trail system implemented is not intended to simulate real or imagined acoustic spaces, and is designed as a dynamic spatial trailing effect for moving sound objects.

6. 8 CHANNEL VERSION

The user is presented with a top down perspective of a 2D virtual space that represents the physical performance space. The player selects a sound type, a cyclical spatialisation sound object is then selected and introduced into the virtual space, this is known as a Spatial Object. The user adjusts the positions of Path Maker Nodes that determine the path, or trajectory of the sound generating object, which is termed the Sound Projectile. The coordinates of the Sound Projectile directly control spatial motion and two key parameters of sound generation. These are: spectral frame and grain read position (playhead) in the phase vocoder and granular synthesis engines. Coordinates of the Sound Projectile thereby control timbre, sound evolution and spatial motion. The x and y axis both control spatial parameters, whilst only the x axis is used to control spectral frame and grain read position. This mapping decision was considered appropriate for a series of pieces being developed, but may be quickly modified to use alternative axes, or decoupled completed if required.

A simulated Doppler effect is also integrated to create more dramatic and quasi-realistic spatial motion effects. The Doppler effect becomes a more prominent feature when using faster spatial trajectory motion speeds. As a result, spatial motion can be perceptually exaggerated. High frequency absorption filtering is also implemented to simulate real-world spatial behaviors.

Each Path Maker Node has an associated Projectile Rate Node that is used to control the speed of the Sound Projectile as it travels towards the respective Path Maker Node. When Path Maker Nodes are positioned a greater distance apart, with a high Sound Projectile rate, fast moving spatial effects occur, with quicker transitions in timbre and more exaggerated pitch modulation effects. When the nodes are positioned closer to each other with a slower projectile rate, gradual timbral shifts occur within a more limited spatial area. As the y axis is not mapped to control spectral frame or grain read position, it is possible to create more gradual timbral shifts, with greater spatial motion by placing the Path Maker Nodes in roughly the same position along the x axis, but at very different positions along the y axis.

Figure 4 represents two Spatial Objects and their associated nodes. Each cluster of Path Maker Nodes corresponds with a selected sound type, with motion of each Sound Projectile limited to the interconnected nodes within its cluster. Each of the Path Maker Nodes can be moved freely around the virtual space, changing the trajectory, or ambisonic of the Sound Projectile as it travels between the different nodes. The smaller sphere attached to each of the Path Maker Nodes is the Projectile Rate Node, this allows the player to control the speed of the projectile as it travels towards its corresponding Path Maker Node. The time contour of the Sound Projectile is determined by the mouse, through selection of the available easing curves, further shaping spatial modulations. As the Sound Projectile moves between the nodes, spatial and spectral parameters of sound develop.

7. AUTOMATIC TRAJECTORY MODULATIONS

A generative feature is implemented which enables the user to activate automated motion of all nodes within a Spatial Object, altering spatial targets and trajectories, resulting in dynamic shifting spatial, spectral, and timbral effects. Further user input is possible when the generative rotation nodes are active, as the Path Maker Nodes and Bezier Nodes in a Spatial Object may still be freely moved and repositioned as normal.

A central rotation point is calculated for each Spatial Object. The rotation point is averaged from the positions of all Path Maker Nodes within the cluster. If activated by the player, these nodes rotate around the central rotation point in either a clockwise or anti-clockwise direction, according to the selection of the player. The Bezier Nodes may also be rotated, and these rotate independently to the Path Maker Nodes. If the Path Maker Nodes are kept static and the Bezier Nodes are rotated, then dynamic shifting trajectories occur which move between the same static points in space. The rate
of rotation can also be adjusted for both node types. The rotation modes result in dynamically changing trajectories of the Sound Projectile, creating a form of generative effect that is dependent upon the position of each node relative to each other node. The player can continue to adjust node position relationships whilst the rotation mode is active.

The user is presented with two work areas and several groups of button controls. Spatial Objects are instantiated using the icons in the center of the interface, (labeled O1, O2, O3 etc. in Figure 8). The instantiations of Spatial Objects appear in the Spatial Trajectory Prep Area where they can be freely positioned and speed settings adjusted. No sound is generated by the Spatial Objects when in the Prep Area. To move the Spatial Objects to the Performance Area where they become active and output sound, the corresponding Spatial Object icon is again selected. Once in the Performance Area, the Spatial Object is activated and animated, generating dynamic coordinate data for the spatial sound engine. When the Spatial Object is removed, or moved back to the Prep Area, sound output for the object ceases. When in the Performance Area, all nodes, modes and speed settings of the Spatial Object may still be modified by the user, allowing real-time modification of spatial and spectral trajectories during sound generation/spatialisation.

8.1. Easing Palette

The Easing Palette contains 27 different easing options, or interpolation curves that determine changes in speed, and sometimes direction of the Sound Projectile as it travels between each of the Path Maker Nodes. These curves expand the range of potential spatial and spectral effects, and allow time contours to be varied. Different time contours may also be combined.

8.2. Speed Display & Master Speed

The Speed Display section updates dynamically whenever a node speed parameter is changed, providing visual feedback of the speed settings for each node when adjusted by the user. The values represent the time it takes for the Sound Projectile to move from the previous node to the destination Path Maker Node. The overall speed of motion can be scaled for all Spatial Objects in the Performance Area, allowing all Sound Projectile nodes to move at speeds to be increased or decreased, whilst still retaining relative rate relationships.

8.3. Snapshots & Recall

Once the user has created an active cyclical pattern, using any number or combinations of Spatial Objects, the position of every Spatial Object and its associated nodes can be stored as a Performance Area Snapshot. Speed settings and automated rotation mode states are also stored. These snapshots can then be later quickly recalled, allowing dramatic structural shifts in spectral and spatial features. The Performance Area Snapshots allow for thematic repetition, as previously created patterns can be reinitiated. Cyclical patterns may also then be developed with further adjustments of the Spatial Object nodes from the stored snapshot positions. The snapshot feature enhances the performance potential of

9. 24 CHANNEL VERSION

The primary functions of the eight channel version of Cyclical Flow are present in the twenty-four channel version, but in this version sound is spatialized in three dimensions. Additional player editable windows are included, facilitating the control of height. The new lower windows represent the same performance space, but provide a front facing perspective to accompany the top down or inter-window. These allow the player to adjust the elevation of each Path Maker Node and Bezier Node, allowing spatial movement of sound throughout a three dimensional performance space. A 2D representation of a 3D space is implemented to assist in accuracy of object placement.

Figure 9. The graphical user interface (24-channel)

This version includes identical groups of button controls as are found in the eight channel version, except these are presented in a slightly different layout. The two additional lower windows represent height and width, the upper windows width and depth. Combined these windows represent all three dimensions (x, y, z) of the performance space. The upper and lower left windows represent the Spatial Trajectory Prep Areas for the upper and lower right windows showing the active Performance Areas. All features of the eight channel system are available in the twenty-four channel system. Spatial Object behaviors, trajectory modulation, continual time contours, and snapshot storage and recall systems are fundamentally identical, except that in this version these systems function using coordinates in three dimensions.

10. OBSERVATIONS FROM USE

The system was explored during the realization of two twenty-four channel fixed media works. The opportunity to experiment, explore and improvise with interfaced sound generation, textural, spectral and spatial parameters provided an alternative perspective on working on and developing sound materials, with spatiality in three dimensions an inherent part of the sound creation and sound design process at an early stage of composition. It was found that shifting complex spatial counterpoint and interwoven trajectories can be quickly created and interchanged. The system was found to be best suited to real-time creation and performance of spatially complex compositions in large multichannel environments. The spatial components of Spatial Objects allowed for structuring and organizing spatial materials, which may also be recorded, for future use as spatial trajectories in larger multichannel environments. It therefore provides a personal solution for the composition of multichannel works. Throughout the testing and development phase additional features were added to extend the performance potential of the system. Whilst these features have been used when capturing spatial performances for fixed media composition, the system has not been presented within a concert situation, so has currently only been tested and applied as a compositional tool. There is certainly scope for presenting the project as an audiovisual performance in a concert situation, in which the audience may view the performed and generative trajectories, and meaningful feedback on audience experience in this context is to be sought.
11. FINAL COMMENTS & FUTURE DIRECTIONS
The system was originally designed and conceived as a solution, tool and sound toy for real-time transformation and spatialisation of sound materials for personal compositions in larger scale multichannel environments. However, the system may be easily adapted for alternative specialist performance spaces with differing speaker configurations, and further opportunities for testing, development and creative exploration are to be sought, with the software made freely available for other composers to investigate and explore. There is potential for exploring and applying the tool in the composition and realization of multichannel pieces and performances on various scales. Initial testing has also been conducted using binaural / HRTF techniques, predominantly for documentation, demonstration and system development purposes, with the system being flexible enough for this approach to be easily and quickly implemented. The results of which were encouraging, although not deemed as successful as when working with a multichannel loudspeaker array. More thorough testing in this area is yet to be completed. The system could also translate effectively to current multi-touch technologies, with the visual interface application running on a multi-touch device, communicating coordinate and control data over a wireless network to the sound engine running on a host machine.

12. REFERENCES
QED - SIMULATED PHYSICS

2.1. ‘The tertium quid’

“Without the computer-based simulation, the material culture of late twentieth-century microphysics is not merely inconvenient – it does not exist” [4, p. 689]. Numerical computation, and especially the Monte Carlo (MC) simulation method (developed in 1949, [7]) have changed the face of scientific research, starting with physics. Crucial elements of these simulations are ‘pseudo-randomness’ and the ‘generation of data’: in theory, random numbers are needed for MC simulation, but numerically only pseudo-random numbers can be generated, which work – depending on their repetition cycle – quite as well as real random numbers. Second, the ‘generation of data’ is a term scientists often use when they run a simulation for a certain time, generating data.

Physics is classically divided into theoretical and experimental physics, but some researchers argue, that simulations are the tertium quid, a categorically different research approach. Some even adhere to the so-called stochastic view, pointing out that the ways of thinking about physical systems have changed due to computers: “Asking the question, ‘How can I formulate the problem on a computer?’ has led to new formulations of physical laws and to the realization that it is both practical and natural to express scientific laws as rules for a computer rather than in terms of differential equations” [5, p. 4].

2.2. Perception of simulation data

Any results of a simulation are ultimately presented as visualizations – otherwise they ‘live’ only inside computers. This can be a direct plot of data with often numerous 2d slices of the data, or the graphics of an aggregated outcome, as for example the histograms of particle detection at CERN. Physicists are very experienced in generating and manipulating graphics for finding evidence to confirm or refute hypotheses, or to gain an idea of the data. In visualization, many conventions exist that help to objectively interpret the perceptualization process. Still, no visualization is objective, but implies human pattern recognition with the data. The spatial coordinates of a lattice site are allocated to a fixed position in the tracking region, as depicted in Fig. 2. The height, as the third dimension z, is compressed, thus all lattice sites (x, y, z) can be reached within the range of a person coughing down or stretching and walking around. The fourth dimension, the time, is inherently used for the sonification as described in Sec. 3.1.

The localization of the lattice sites is presented via a binaural rendering, using direction dependent transfer functions of the human outer ear (head related transfer functions, HRTF). If the listener was exactly ‘on’ a lattice site, the according sound was played in mono, leading to in-head localization. Otherwise, the sound was located virtually in space. The described setting allowed for one person at a time exploring the data listening space, while others could follow the sound outside the tracked area with additional headphones.

2.4. The generated data

The data were generated with an MC simulation of lattice QED that was integrated in the SC source code. Due to large computation times, single configurations of the model simulated at different coupling values (within both phases) were stored beforehand and could be chosen by the listener. The data are given on a 4-dimensional lattice, which can be interpreted as three spatial dimensions, \( x, y, z \), and time, \( t \). Our lattice was of the size of ten sites \( \times 10 \times 10 \times 10 \) = 1,000,000, thus it had \( 10^6 \times 10^6 \times 10^6 \times 10^6 \) sites. At each of these space-time-sites of the lattice, a data value is given (see Sec. 3.1).

3. DATA LISTENING SPACE

The goal of the installation was to create an aesthetically interesting listening experience, that would allow a general public to assess this abstract data. The main challenge with the data structure was to display a four-dimensional space in a way that still permitted orientation. Furthermore, the physical observables, the loops, are very complicated entities to communicate.

In the data listening space a person moves freely about, while her/his position and rotation are captured by a motion-tracking system over a target attached to headphones. The spatial coordinates of a lattice site are allocated to a fixed position in the tracking region, as depicted in Fig. 2. The height, as the third dimension z, is compressed, thus all lattice sites \((x, y, z)\) can be reached within the range of a person coughing down or stretching and walking around. The fourth dimension, the time, is inherently used for the sonification as described in Sec. 3.1.

The localization of the lattice sites is presented via a binaural rendering, using direction dependent transfer functions of the human outer ear (head related transfer functions, HRTF). If the listener was exactly ‘on’ a lattice site, the according sound was played in mono, leading to in-head localization. Otherwise, the sound was located virtually in space. The described setting allowed for one person at a time exploring the data listening space, while others could follow the sound outside the tracked area with additional headphones.

Figure 2. Schematic plot of the data listening space. Three space-like dimensions of the lattice are placed virtually in a real physical space. A listener can move freely about with headphones that are tracked by infrared cameras. Each dot represents the virtual position of a sound/lattice site in the data listening space. The positions in the height dimension are compressed, in order to allow the listener to reach all lattice sites.

3.1. Sonification operator

Sonifications are of interdisciplinary nature, and the communication between, e.g., physicists and sound designers is often difficult. Therefore we developed notation moduli [11] that allow an explicit formulation to clarify the dependencies between the data and sound parameters, and the sound synthesis. This results in a formal sonification operator as suggested by J. Rohrhuber [9]. The advantage is, that the mathematical formulas of the neighborhood can be interpreted more or less universally within the scientific community, and does not imply notations based on programming languages that are, e.g., focusing on sound synthesis, or other, domain-related indices.

The sonification starts from the data values \( p, p, p \), given for each lattice site \((x, y, z)\) and containing numbers within \([-\pi, \pi]\). Obviously, not all of the 10,000 data sites can be perceived at the same time. To avoid ‘auditory overload’, only 125 sites of the neighborhood were played simultaneously, with the nearest sites being dominant in amplitude. This range was also chosen due to machine performance. The choice of the sites, whose values are sonified, depends directly on the position of the listener in the data listening space. This relation is indicated in the notation by the first ‘interaction index’, \( C_{ij} \).

The single numbers \( p, p, p \) are collected into sequences of numbers, following closed loops on the lattice (see Fig. 3). The simplest loop is the ‘0-loop’, where one reads out all values of the time dimension and loops this sequence. More complicated loops spin around the time dimension and one other dimension (\( x, y, \) or \( z \)). Depending on the range \( \lambda \) in this other dimension, we call it a ‘1-loop’, ‘2-loop’ etc. The three possible 1-loops at each \((x, y, z)\)-site are shown in Fig. 3 next to the (one possible) 0-loop.
Formally, the data $d_0$ are given for the '0-loop' as:

$$d_0(x,y,z,t) = \rho(x,y,z,C_1;t;r_2=0) = d_0$$  \hspace{1cm} (1)

A second interaction index, $r_2$, is controlled by a second tracking target, held in the hand. It defines the range $r_2$ and lies within $1$ and $6$, $r_2 \in [1;6]$. Thus, depending on the distance of the second tracking target top the first, the listener hears only the 0-loops, or a superposition of the 0-loop with higher-ranging ones.

The whole data set can thus be written as:

$$d_0(x,y,z,t) = \{\rho(x,y,z,C_1);t;r_2(x,y,z,C_2)\}$$

A second interaction index, $r_2$, is controlled by a second tracking target, held in the hand. It defines the range $r_2$ and lies within $1$ and $6$, $r_2 \in [1;6]$. Thus, depending on the distance of the second tracking target top the first, the listener hears only the 0-loops, or a superposition of the 0-loop with higher-ranging ones.

The whole data set can thus be written as:

$$d_0(x,y,z,t) = \{\rho(x,y,z,C_1);t;r_2(x,y,z,C_2)\} = d_0$$

(2)

This choice of the data guarantees in a first step to receive physically meaningful entities, i.e. closed loops in the lattice.

Now the loops are conditioned as input data for the sound synthesis. The 0-loop is distorted, by dividing by the maximum value and raising to the power of $8$, thus normalizing to:

$$d_0^{\text{dist}} = \frac{d_0}{\max(d_0)}$$  \hspace{1cm} (3)

The focus of interest thus lies on large changes in the 0-loop that remain after the distortion, while very uniform loops that remain after the distortion, while very uniform changes are found in the 0-loop, the quicker the loop range.

Even if many loops are played at the same time, single bands, and the higher-ranged ones in higher octave bands. Where one hears the lower-ranged loops in lower octave bands, the torted 0-loop determines the looping time.

The focus of interest thus lies on large changes in the 0-loop that remain after the distortion, while very uniform changes are found in the 0-loop, the quicker the loop range.

Thus results a polyrhythmic series of overlapping sounds, whose length and amplitude depend on the 'discontinuity' of the basic 0-loop, and whose frequencies depend on the farther loops, running over $r$ and one space dimension each.

3.2. Resulting sound

The resulting sound gives the following information:

- **Silent sites** are not interesting in our choice as there are hardly any changes in the 0-loop's time. Loud sites, on the contrary, indicate large fluctuations that can be interpreted as high energy.
- The pitch and the rhythm encode the energy: the more fluctuations in the lattice, the higher the pitchs and the quicker the rhythm.
- Different loop sizes are distinguished by different frequency bands. Therefore, bigger loops can be compared to smaller ones. The overall impression gives an acoustic average of the different loop sizes.

4. TECHNICAL SETUP

The data listening space was physically installed in the Lajeri hall in the Mumuth building in Graz. In this space the VICON5 motion capturing system has been installed in the hall: the 16 infrared cameras (see Figure 4), which are part of the system, have been mounted on an eight by eight meters rig suspended at 3.5 meters height.

The chosen frequency ranges (Eq.6) result in a mapping, where one hears the lower-ranged loops in lower octave bands, and the higher-ranged ones in higher octave bands. Even if many loops are played at the same time, single ones can be identified by their specific frequency band.

The distorted 0-loop signal $d_0^{\text{dist}}$ is looped, i.e. repeated infinitely. In a formal way, a looping operator $L_0$ is defined, that loops over 10 sites, i.e. the whole time range, infinitely often. This is a sound wave, which is taken as input for a bank of resonant filters, $\mathcal{F}^\text{reson}$, at the frequencies $f_\text{reson}$. The whole sound grain is modulated by envelope $\mathcal{F}^\text{env}$, which gives the 'heterodyne' of $\mathcal{F}^\text{reson}$ the smallest perceived meaningful sound entity. In this case, the perceivable entities of the sonification operator are indicated by an ring, e.g. $\gamma$, depending on time $t$.

$$\gamma(x,y,z,t) = \sum_{r_2 \in \mathcal{C}_2} \mathcal{F}^\text{env}_{\text{reson}}[t] = \mathcal{F}^\text{reson}_{\text{env}}[t]$$

(7)

The signal is rendered binaurally depending on the position and orientation of the listener. In the formalization, the 'headphone operator', $\mathcal{F}^\text{headphone}$ transforms the signal $\gamma$ into a pair of binaural signals, $\gamma^L$ and $\gamma^R$:

$$\Omega \gamma^L[x,y,z,C_1] \rightarrow \gamma^L \gamma^R$$

(8)

This sound series is looped for each location of the listener, until s/he moves on (formalized as interaction 1, $\mathcal{C}_1$). This gives the overall sonification operator, $\mathcal{F}^\text{sonification}$:

$$\mathcal{F}^\text{sonification} = \gamma = \sum_{r_2 \in \mathcal{C}_2} \mathcal{F}^\text{env}_{\text{reson}}[t] = \mathcal{F}^\text{reson}_{\text{env}}[t]$$

(9)

To experience the installation visitors had to wear headphones. On these we fixed a tracking target consisting of a set of five reflecting markers (see Figure 5). The position and orientation data of this object is then streamed to a different machine by the Vicon200C program developed at IEM Graz. The OSC messages are received by a SC client. According to the listener's position and orientation in space, the entire auditory scene is generated by the sonification algorithm as described in Sec. 3.

The analysis lead to 'foci', along which the subjects could be compared. This analysis lead to 'foci', along which the subjects could be compared. This analysis lead to 'foci', along which the subjects could be compared. This analysis lead to 'foci', along which the subjects could be compared. This analysis lead to 'foci', along which the subjects could be compared. This analysis lead to 'foci', along which the subjects could be compared. This analysis lead to 'foci', along which the subjects could be compared. This analysis led to 'foci', along which the subjects could be compared. This analysis led to 'foci', along which the subjects could be compared. This analysis led to 'foci', along which the subjects could be compared. This analysis led to 'foci', along which the subjects could be compared. This analysis led to 'foci', along which the subjects could be compared. This analysis led to 'foci', along which the subjects could be compared. This analysis led to 'foci', along which the subjects could be compared. This analysis led to 'foci', along which the subjects could be compared. This analysis led to 'foci', along which the subjects could be compared.
4. ACKNOWLEDGEMENTS

We would like to thank Christof Gattringer and his team at the Institute of Physics of the University of Graz for providing their expertise in the QED simulation and for supporting our research project (QCD-audio, Music and Performing Arts Graz, 2010).

7. ABSTRACT

In film or video production, the impact of score music is frequently considered pivotal to a movie’s conveying of semantic and emotional content. However, filmmakers (directors or editors) often lack the musical expertise to communicate their dramaturgic requirements to the composer, resulting in the regular use of temp tracks, which are generally disliked by composers. We try to ameliorate this situation by developing optimized ways of describing the desired musical outcome using a crowd-sourced scheme of semantic and affective categories, wrapped in the definition of an XML format.

1. INTRODUCTION

1.1. Problem Definition & Relevance

This work presents a part of the research project GeMMA [8], which tries to develop novel ways to enhance communication flows between film directors and composers. Score music composition or selection is acceptedly considered a vital part of a movie’s reception and its ability to convey moods, metaphors and meanings. Since directors or editors often lack the musical expertise to communicate their requirements to a film composer, so-called temp tracks are used as a fallback. Temp tracks are, however, often disapproved of among film composers, as this pre-produced music material and/or sound-a-like ideas from the director or editor often limit the opportunities and quality of the composition process.

Within the research project, we try to improve this situation by

- semi-automating the generation of temp tracks in an algorithmic composition based scenario, and
- enhancing the communication between the parties involved in the film production process by developing optimized ways of describing or selecting the desired output.

In this paper, we describe the development of methods, and a technical format incorporating these methods to describe the semantic and emotional content of a movie scene. Section 2 covers the framework of semantic annotation that we have developed by clustering an annotated database of movie clips. In section 3, we show why we regard instrumentation as a pivotal parameter in the conveying of a certain symbolic significance. Section 4 shows the development of instrumentation clusters for use in an algorithmic composition scenario, while in section 5 we finally present an XML format for semantic movie scene annotation.
tering on these criteria, with the goal of reducing the depth of categorization to a manageable scope of super-categories.

In order to achieve this, unsupervised Agglomerative Hierarchical Clustering in complete linkage mode was performed [3]. The distance measure used therein was a Hamming distance based on three binary vectors associated to every clip (e.g. Clip A is annotated with Event 1 = 1, else 0, etc.) (cf [5]).

2.2. Results

This resulted in 8 event, 6 symbol and 8 material clusters. These super-categories produced further questions: Which correlations between events / symbols and acoustic materials exist? Are there correlations between events / symbols and specific musical timbres? To answer these questions, the musical timbres used frequently within each cluster, as well as key timbres, i.e. timbres that are used solely within one cluster and thus clearly flag a clip with a super-category, were analyzed. Tables 1 and 2 delineate instrument profiles pertaining to each event or symbol super-category.

### Table 1. Instrument profiles used within event super-categories.

<table>
<thead>
<tr>
<th>Event Cluster</th>
<th>Instruments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movement</td>
<td>Drama, Percussion, Brass, Trumpet</td>
</tr>
<tr>
<td>Drama</td>
<td>Violin, Oboe, Double Bass, Flute,</td>
</tr>
<tr>
<td>Accident</td>
<td>Electronic, Violin, Trumpet</td>
</tr>
<tr>
<td>Shock</td>
<td>Electronic, Vocals</td>
</tr>
<tr>
<td>Violence</td>
<td>Violin, Horns</td>
</tr>
<tr>
<td>Surprise</td>
<td>not specifiable</td>
</tr>
<tr>
<td>Death</td>
<td>not specifiable</td>
</tr>
<tr>
<td>Celebration</td>
<td>Guitar, Vocals, Tambaourine</td>
</tr>
</tbody>
</table>

### Table 2. Instrument profiles used within symbol super-categories.

<table>
<thead>
<tr>
<th>Symbol Cluster</th>
<th>Instruments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action/Violence</td>
<td>Brass</td>
</tr>
<tr>
<td>Fear/Tension</td>
<td>Violin, Dudergino, Piano,</td>
</tr>
<tr>
<td>Freedom</td>
<td>Vocals, Mallets</td>
</tr>
<tr>
<td>Romance</td>
<td>Violin, Chroms, Piano,</td>
</tr>
<tr>
<td>Tragedy</td>
<td>Violin, Piano</td>
</tr>
<tr>
<td>War</td>
<td>Vocals, Horns</td>
</tr>
</tbody>
</table>

After reviewing clip instances not clearly assignable to a single cluster, we decided to consider two additional symbol super-categories to be included in the description format: Joy/Comedy and Desolation.

3. SEMANTIC LISTENING SURVEY

3.1. Methodology

After evaluating these results, the decision to interlink semantic designations of a movie scene with a certain type of instrumentation was taken. For example, if a user of a semi-automated temp-track generator specified a scene to be a war scene, this would trigger a specific orchestration template. To fulfill this, further investigation into the perception and interpretation of instrumentation in connection with semantic properties was necessary.

The aim of this research task was to find out what musical trait (instrumentation, melody or rhythmic features) listeners attribute the highest influence regarding symbolic significance, as well as whether there are specific correlations between perceived affect and recognized symbol, and to what degree listeners agree on a certain symbol without being presented the visual context (i.e. the image).

120 short movie clips from the clip library were taken and pre-assessed by a group of experts prior to the survey to obtain a ground-truth set. This pre-assessment consisted of tagging the clips with symbol, tonality, primary and secondary instrument groups. Selection criteria were further restricted in order to obtain analyzable results:

- Clips may only contain classifiable instruments
- Clips are restricted to a length of 10 seconds and should contain a concluded motif or melody
- Clips may not contain vocals, as the lyrics are prone to influence participants perception
- A homogenous mixture of symbols (from the ground-truth set) should be present in the test set.

To acquire a reasonable amount of assessments, 20 tests with 6 samples each were assembled. The test was conducted anonymously with 8 groups of students, resulting in >30 assessments for each symbol. The test included the following questions/ratings:

- Position the perceived mood of the clip in a Circumplex
- Select one symbol out of 8 (see section 2.2)
- Select the reason(s) for your symbol selection (multiple answers possible): a) Instrumentation, b) Rhythm, c) Melody and d) Other (free category)
- Recognition of the movie, if possible

3.2. Results and Observations

The following general observations were made; the complete data evaluation and will be made available under [8].

3.2.1. Instrumentation is a Key Factor for Symbolic Functions of Score Music

Almost every symbol showed a clear tendency towards instrumentation as a key factor for symbolic impact. Where it did not, it was mentioned in combination with another factor (mostly melody) throughout the survey.

3.2.2. Interaction between Melody and Instrumentation

Whenever a scene’s symbolic significance appears to be dependent on melody and there is no clear primary instrument group, the score is often perceived as passive (thus rather relaxed and calm). This is especially true for scenes expressing tragedy or freedom.

Whenever a scene displays no clear primary/secondary instrument group or no clear harmonic structure, it often has an activating effect but is difficult to categorize in terms of symbolic significance. This can, again, very likely be attributed to the absence of visual context. Again it can be pointed out that some semantically clearly distinguishable symbols, such as joy and desolation, can exhibit very similar musical structures.

3.2.3. Fear/Tension’s Special Position in the Analysis

The so labeled compositions can be regarded as being isolated because of their atonal harmonics and the very clearly delimited playing styles of the instruments (e.g. staccato violins). Thus, the symbol had a very high recognition rate as well as a very high consensus regarding mood perception (activating negative).

3.2.4. Rhythm has Little Influence on Symbol Perception

Eventually, rhythm was called a decisive factor for symbol attribution only in isolated cases (e.g. fear/tension). This again fosters the interpretation that in motion pictures, rhythm perception depends highly on the style of editing (fast vs. slow cutting; hard cuts vs. dissolves etc.)

4. CLUSTERING OF INSTRUMENTATION PROFILES

To arrive at a model of symbolic instrumentation suitable for use in a computational music generation engine, the final step lay in finding clusters of instrumentation pertaining to each symbol. To achieve this, every clip included in the test was tagged with the instruments it contained (1 = instrument is present in the mix, 0 = instrument is not present).

Again, a Hamming distance was used as a distance measure, and clips were then clustered using the WEKA data mining software. For clustering, the expectation maximization algorithm was selected, letting the software decide on the optimal amount of clusters. Thus, it was possible to obtain an appropriate amount of instrument profiles to let a music generation engine select from, once the user specifies a certain symbol.

This process resulted in a varying amount of instrument clusters for each symbol, from which a music generation engine or a user can select the most appropriate.

5. SEMANTIC MOVIE SCENE ANNOTATION XML FORMAT

The aforementioned studies were conducted to produce a firm conceptual basis to develop a machine readable XML (extensible markup language) formal.

This XML schema was devised to be used to flexibly interchange semantically annotated metadata for the description of movie scenes; listing 1 displays an exemplary implementation.

Listing 1. Exemplary movieannotated.xml

```xml
<movieannotation>
<config>
<track name="symbol" type="discrete">
<symbol>
<pos>60</pos>
<valence>0.9</valence>
</symbol>
</track>
<track name="valence" type="continuous">
<valence value="-0.7 -0.5...
</track>
<track name="affect" type="discrete">
<affect>
<pos>60</pos>
<valence>-0.6 -0.8...
</affect>
</track>
<track name="sentiment" type="continuous">
<sentiment value="0.4 0.5..."
</track>
</config>
<tracks>
<event>
<symbol value="0">
</symbol>
<event value="0">
</event>
<event value="0">
</event>
<event value="0">
</event>
</tracks>
</movieannotation>
```

The schema is conceived to be modular, in order to be able to insert additional information (e.g. more tracks) at a later stage. To generate annotation files, a prototypical video annotation tool (see figure 1) was created in MaxMSP. Users can play movie scenes and add semantic information on a timeline, as well as provide affect values as valence/ arousal pairs on a X-Y-slider.

This tool is comparable to ANVIL, [6], however, in this context a customized tool for continuously annotating media content in a 2-dimensional manner was necessary, as well as a close binding to specific audio- and video-related soft- and hardware. The annotation format is furthermore easily expandable to support multiple media contexts (MIDI, OSC, common media exchange formats, etc.).
We have presented a flexible XML format for semantic movie scene annotation based on thorough literature research and crowd sourcing of metadata-annotated movie clips. We believe this structure to be a valuable addition to the field of score music studies and semi-automated, generative composition of temp tracks, because it provides the basis for an optimized vocabulary and grammar for film and music professionals to build their communication upon.

While it can be argued that the found super-categories are considerably arbitrary concerning integration in a well-defined ontological hierarchy, their selection appears to be nonetheless valid for their crowd-sourcing provenance (by media professionals), as they are also backed by relevant literature on the topic. Moreover, concept-based ontologies designed for use in video retrieval contexts (e.g. [7]) did not seem applicable to the task at hand, because they predominantly feature concrete rather than abstract semantic concepts. Notwithstanding, the found semantic clusters will have to be tested for completeness and usability in annotation tasks, and possibly improved thereafter.

Moreover, we have found significant correlations between these super-categories and musical/timbral traits, as outlined in section 3.2. Exemplary musical content that has been produced using clustered instrumentation profiles (section 4) yielded subjectively convincing results that will be evaluated in a listening survey (using the same layout as in section 3) for its suitability as temp tracks for a specific semantic context.

7. FUTURE DIRECTIONS

At a later stage of the research project, it will be possible to upload annotated movie scenes to a web service which will process the information and provide score music prototypes in MIDI format with a pre-defined channel scheme.

Moreover, the framework used to generate optimized temp tracks will also be published and made available under an open source license.

8. REFERENCES


6. CONCLUSION

We have presented a flexible XML format for semantic movie scene annotation based on thorough literature research and crowd sourcing of metadata-annotated movie clips. We believe this structure to be a valuable addition to the field of score music studies and semi-automated, generative composition of temp tracks, because it provides the basis for an optimized vocabulary and grammar for film and music professionals to build their communication upon.

While it can be argued that the found super-categories are considerably arbitrary concerning integration in a well-defined ontological hierarchy, their selection appears to be nonetheless valid for their crowd-sourcing provenance (by media professionals), as they are also backed by relevant literature on the topic. Moreover, concept-based ontologies designed for use in video retrieval contexts (e.g. [7]) did not seem applicable to the task at hand, because they predominantly feature concrete rather than abstract semantic concepts. Notwithstanding, the found semantic clusters will have to be tested for completeness and usability in annotation tasks, and possibly improved thereafter.

Moreover, we have found significant correlations between these super-categories and musical/timbral traits, as outlined in section 3.2. Exemplary musical content that has been produced using clustered instrumentation profiles (section 4) yielded subjectively convincing results that will be evaluated in a listening survey (using the same layout as in section 3) for its suitability as temp tracks for a specific semantic context.

7. FUTURE DIRECTIONS

At a later stage of the research project, it will be possible to upload annotated movie scenes to a web service which will process the information and provide score music prototypes in MIDI format with a pre-defined channel scheme.

Moreover, the framework used to generate optimized temp tracks will also be published and made available under an open source license.

8. REFERENCES


6. CONCLUSION

We have presented a flexible XML format for semantic movie scene annotation based on thorough literature research and crowd sourcing of metadata-annotated movie clips. We believe this structure to be a valuable addition to the field of score music studies and semi-automated, generative composition of temp tracks, because it provides the basis for an optimized vocabulary and grammar for film and music professionals to build their communication upon.

While it can be argued that the found super-categories are considerably arbitrary concerning integration in a well-defined ontological hierarchy, their selection appears to be nonetheless valid for their crowd-sourcing provenance (by media professionals), as they are also backed by relevant literature on the topic. Moreover, concept-based ontologies designed for use in video retrieval contexts (e.g. [7]) did not seem applicable to the task at hand, because they predominantly feature concrete rather than abstract semantic concepts. Notwithstanding, the found semantic clusters will have to be tested for completeness and usabil-
consider each note as part of musical phrase, and they express this idea by controlling expressive parameters, such as tempo, dynamics or timbre. We can therefore assume that there is a finite number of latent abstract phrase states having transitions on every single note. Furthermore, we assumed that observable score and performance features have different probability distributions under each abstract phrase state. Based on these assumptions, musical phrasing can be modeled as a simple HMM. Let \( I_n \) be an abstract state of the \( n \)-th note, whose value is one of the \( K \) number of states \( S_k \). We assume that \( S_0 \) and \( S_K \) represent the start and end of a phrase, respectively, and the other states are in between. Then, we can model state transitions such as in Figure 1. Since a phrase can have a different number of notes, self-transition should be allowed. Assuming that there is no overlaps between two consecutive phrases, the transition \( S_k \rightarrow S_k \) then represent a phrase boundary. The number of states \( K \) defines the minimum length of a phrase, so this should be carefully selected. The observation vector of the \( n \)-th note \( X_n \) can include both score and performance features. Several studies on performance modeling indicate that pitch interval and norminal duration ratio are two of the most fundamental features describing melodic structures, so we have select them as score features. We define those features by

\[
\text{PitchInterval}_n = \text{Pitch}_n - \text{Pitch}_{n-1}, \quad (1) \\
\text{DurRatio}_n = \frac{\text{Duration}_n}{\text{Duration}_{n-1}}, \quad (2)
\]

Tempo changes are certainly one of the most important parameters for the expressive shaping of a phrase. Since every performance has a different average tempo, relative tempo changes are more informative than the absolute tempo of each note. Consequently, we define relative tempo changes by

\[
\text{TempoFactor}_n = \frac{\text{Tempo}_n}{\sum_{i=1}^{n} \text{Tempo}_i}, \quad (3)
\]

and quantize them into a finite number of discrete observations. Let \( Y \) be a sequence of abstract phrase states \( Y_n \), and \( X \) a sequence of observed score and performance feature vectors \( X_n \). Assuming that the current state is conditioned only by its predecessor state, the joint probability of \( X \) and \( Y \) can be written as

\[
P(X|Y) = P(Y_1|X_1)P(Y_2|X_1, Y_1)P(Y_3|X_1, Y_1, Y_2)P(Y_4|X_1, Y_1, Y_2, Y_3), \quad (4)
\]

This means that we can obtain a complete HMM-based phrasing model by estimating the initial state probability vector \( \pi \), the state transition matrix \( A \) and the observation probability matrix \( B \).

3.2. Boundary Estimation with Unsupervised Learning

The model parameters \( \pi \), \( A \) and \( B \) have to be estimated from input performance data. In order to define the dimension of \( B \), we need to enumerate observations \( X_n \). The number of possible observations \( X_n \) can, however, be extremely large, so we have enumerated only score and performance feature vectors observed in the input performance. Since the abstract phrase state for each note is unknown, a direct calculation of the model parameters is not possible. However, the Baum-Welch Algorithm provides an iterative solution for a Maximum Likelihood Estimation of the model parameters. While the algorithm is based on the Expectation-Maximization method, this unsupervised learning process can be understood as a clustering of observed score and performance pairs into abstract states while considering state transitions. Once we have trained all model parameters, we can estimate a sequence of abstract states \( Y \) given score and performance feature vectors \( X \) by computing

\[
Y^* = \arg \max_{Y} P(Y|X, \pi, A, B). \quad (5)
\]

This is an optimization problem, which can be efficiently computed with a Dynamic Programming Technique known as Viterbi Decoding. Since \( Y^* \) is a sequence of detected abstract states, we can estimate phrase boundaries by finding where the state transition \( S_k \rightarrow S_k \) occurs.

3.3. Evaluation Method

To evaluate the performance of the statistical model, we compared the estimated boundaries with annotations made by a performer and a listener. To obtain those annotations, we recorded different performances of F. Chopin, Mazurka Op. 30, No. 2 and Prelude Op. 28, No. 7 with a student pianist. The pieces were selected, because expressive tempo modulations were expected to play an important role in their performance. Prior to the recording, the pianist annotated phrase boundaries he intended to perform. Since the performer’s intentions and the listener’s perception can be different, another student pianist annotated phrase boundaries while listening the recorded performances. We assumed that the note sequence with the highest pitch as a melody of a phrase, and score and performance features were extracted manually. The number of abstract phrase states \( K \) is selected on the basis of F-measures with different \( K \) values, and tempo changes were quantized into 16 discrete observations. In addition, we compared our method with one of the state-of-the-art methods, Temporerley’s Grouper system, by applying both methods to professional performances selected from CrestMuse PEDB1. Grouper estimates phrase boundaries by applying rules solely based on score features, while we can test the efficiency of the proposed statistical method considering both score and performance features.

4. RESULTS

4.1. Comparison with Human Annotations

Figure 2 shows the results of a comparison with human annotations. The performer’s intended phrasing was following the composed rhythmic and melodic patterns, shaping them with salient tempo modulations, and the listener’s annotations largely match those of the performer. The proposed statistical method could estimate most of the boundaries annotated by performer and listener. Some boundaries look mismatched in the score with the distance of a single note.

![Figure 1. State transition diagram. ST and TR represent the start and end of a performance, respectively. Sj represents an abstract phrase state.](image)

We also recorded a single piece with two different phrase structure interpretations. As shown in Figure 2b and 2c, the performer shaped phrase boundaries with different tempo modulations, establishing a phrase end either before or after the third beat in measure 2, 4 and 6. The listener’s annotations followed these intentions, and so do the estimations of the statistical model. There were also estimated boundaries whose musical relevance are difficult to explain, such as measure 7 in Figure 2c. Still, the results indicate that the proposed method could succeed in taking into account different phrasings of the performer.

4.2. Comparison with Rule-based Methods

Figure 3 shows the results of a comparison with Grouper’s estimations. While Grouper estimates phrase boundaries by applying rules solely based on score features, the proposed statistical method additionally includes performance data in the analysis. This leads to slightly different boundaries, such as in Chopin’s op. 30 (Figure 3a), where Grouper estimated boundaries following the rhythmic pattern repetitions in the score, while the boundaries estimated with our method follow the tempo fluctuation of the performance in measures 10, 12 and 14. Where the performer slowed down the tempo after the first beat, indicating a phrase end different from measures 2, 4, 6 and 8, where the tempo arch ends with the bar line. This result illustrates that a boundary estimation considering only note patterns can be misleading if the performer’s phrasing (in line with or contrary to the composer’s intention) doesn’t follow the “obvious” solution given by the score.

The performance of Mozart KV 545 (Figure 3b) has a flat tempo fluctuation, so note patterns would be the main factor in perceiving phrase boundaries. In the first four measures, there are clear repetitions of a rhythmic pattern, and both Grouper and our method estimated boundaries matching those repetitions. After measure 5, the boundaries estimated with the statistical method, however, seem more plausible in a musical sense, better matching the rhythm structure of the piece. This result indicates that the statistical machine learning approach might be superior in certain cases, even if only score information is processed as an input.

5. CONCLUSION

We presented a new method for estimating phrase boundaries in music performance with HMM-based unsupervised learning. It takes into account score information on pitch and duration as well as performance features related to tempo changes. The results indicate that the model could identify score-related phrase boundaries as well as boundaries related to individual performances, which are often different from those suggested by the notation alone. Since the proposed method is tested only on some of selected music performances, we will conduct a quantitative evaluation on a larger corpus, such as newly published CrestMusePEDB-STR database [4].

6. REFERENCES

Comparison of proposed method and Temperley’s Grouper system with the number of states $K = 5$. All performances were taken from CrestMuse PEDB. Note that a) is a different performance from Figure 2a.
states. Particle filters are also widely used in other computer research fields such as computer vision and robotics. Most of these techniques make an implicit assumption that the observation is aligned with the reference audio globally in terms of the musical structure. However, this assumption does not always hold as explained above. This problem has received less attention in the literature. Fremeley et al. [2] have developed a modified version of DTW, called JumpDTW, to address the jump and repeat problem in classic music at bar levels. Teken et al. [8] presented a score coincident injection to tolerate wrong notes and jumps in the audio. In the remainder of the paper we present a particle filter based method that can address the above problem at the note, as well as measure levels.

3. METHOD

Given the audio of a complete performance we would like to track a different and unfolding performance of the same piece with respect to this reference performance. Let \( F = \{ f_0, f_1, \ldots, f_n \} \) denote the frame-based features calculated from the reference performance and similarly \( F = \{ y_0, y_1, \ldots, y_n \} \) for the new performance. The purpose of the online alignment is to find a mapping between \( F \) and \( Y \) in a causal manner. Alongside the feature frames for the new performance we use a two dimensional vector to represent latent variables of music position \((s, t)\) and tempo \((\tau)\), that is, \( x_n = (s_n, t_n, \tau) \). Here, music position serves a similar role to score position in audio-to-score alignment. We use this term because the method cannot assume the availability of an annotated score and therefore a score reference cannot be used. In the task of audio-to-audio alignment, music position is interpreted as the position in the reference audio and tempo as the relative tempo with respect to the reference audio. Our goal is to track the performance by inferring the music position and the tempo relative to the observations on frames \( y_0, \ldots, y_n \) up to the current time.

We use particles to represent distributions of the latent variables and spread a small number of particles in a wide range in order to catch potential jumps while keeping most particles in a relatively small range in order to infer the current music position. The idea is to infer the local score position using the dominant group of particles while maintaining some awareness of other possible positions that are farther away. Since this accounts for the possibility of jump in the audio, if there is evidence of such a condition, then more particles will iteratively move to the position suggested by the evidence eventually affecting the decision to jump.

3.1. Observation and state transition model

Observation likelihood \( p(y_i | x_i) \) is the probability of observing an audio frame given the latent state. This likelihood is calculated based on a chromagram feature. The chromagram is a commonly used feature in music alignment models because it can capture the harmonic structure from the audio in a relatively compact form. We use a representation similar to the one used in [1] for the observation model.

First, we calculate the angle \( \alpha \) between the chroma vector \( y_i \) from the observed frame and chroma vector \( f_{\alpha} \) from the reference audio frame given by \( x_{\alpha} \):

\[
\alpha = \arccos \left( \frac{f_{\alpha} \cdot y_i}{\|f_{\alpha}\| \cdot \|y_i\|} \right)
\]

We define \( p(y_i | x_i) \) as:

\[
p(y_i | x_i) = \sigma \left( \frac{p(y_i | f_{\alpha})}{\alpha} \right)
\]

where \( \sigma \) is the normalization factor to make \( p(y_i | x_i) \) a probability distribution and \( \alpha \) is a scalar.

Tempo \( \tau_i \) is updated from a normal distribution and the parameter \( \epsilon \) controls the variance of tempo change:

\[
\tau_{i+1} = N(\tau_i, \epsilon_j)
\]

Then one of the following two equations is chosen stochastically to update the current music position from the previous music position. The first equation corresponds to normal tracking without any jump:

\[
x_{i+1} = x_i + \epsilon
\]

The second equation corresponds to a jump that might have occurred in the audio:

\[
x_{i+1} = x_i + \epsilon + \tau
\]

where \( \epsilon \) is drawn from a uniform distribution from \(-C\) to \( C\) and \( \tau \) is the maximum length of frame jump in the audio that we can expect.

3.2. The tracking algorithm

We start with \( N \) particles in a music position at 0, tempo uniformly distributed over a tempo range, and all labelled as \( A \). We initially set the minimum tempo to be half, and maximum tempo to be twice the reference tempo. Tempo is always updated with Equation (3). With chance \( a \), we update the music position with Equation (4) otherwise we use Equation (5) (in our case we set \( a = 0.97 \)). If the music position of a particle is updated from Equation (4) which represents normal tracking without jumping, then the particle will remain with the original label. Otherwise the particle will be labelled as \( B \) regardless of its previous label. In order to maintain the tracking on the regular tracking we need to limit the number of particles exploring jumps. If the number of particles labelled as \( B \) exceeds a certain percentage threshold (10 percent of the total number of particles in our case), then we will update music position with Equation (4) only. We use Equation (5) to keep the particles spread in a large range around the current music position in order to catch potential jumps. However, if we already have a fair amount of particles labelled \( B \), then we do not want to manually add more particles, so we limit the resampling step. Particles with higher weight will be reproduced to create duplicate copies. Higher number of particles labelled \( B \) will indicate stronger possibility of a jump.

After updating the music position, tempo, and label, a weight is calculated for each particle. For particle \( i \) we use:

\[
w_i = p(y_i | x_i)
\]

and the weight will be normalized such that the sum of the weights for all particles adds up to one.

In addition, we penalize the weight of every particle labelled as \( B \) by a factor \( b \) (1.3 in our case). Also, if the particle is drawn from Equation (5) and has a music position within a certain range of the previously inferred music position, then we set the weight of the particle to \( B \) because the possibility of that music position is already represented by particles labelled \( A \). We then check whether the number of particles labelled \( B \) is greater than a threshold (75% of the total number in our case) because of the fact that more particles labelled \( B \) would imply a jump has occurred in the audio. In this case, we swap the labels \( A \) and \( B \) since we always want to infer the music position with the dominant particles. For inference, we first calculate the average weight of the particles labelled \( A \) then infer the music position based on the weighted average of only those particles that have weights higher than the average.

After inference, we use a standard bootstrap filter to resample. That is, each sample has probability proportional to its current weight to be reproduced in the next state. New particles are sampled from this distribution and then some small Gaussian noise is added to the new particle. Note that the new sample keeps the original label. This ensures that a particle with higher weight is more likely to be reproduced, possibly creating multiple copies. After resampling, all particles are set to have equal weights. As a result, if particles labelled \( B \) have high weights, the resampling steps will produce more particles labelled \( B \). The resampling step will remove the particles with lower weights while keeping the diversity of the particles and making it possible to detect potential missing notes and sections.

4. EVALUATION

We evaluate our approach on a collection of real music performance recordings of Chopin’s mazurkas. This set of recordings, which consists of 8 mazurkas, is sampled from 3 different performances for each of the recordings. In total, we have 4 different performances for each one listed in tables 1 and 2, and is also used in [4] for evaluation. The annotations provided by Craig Sapp (The Mazurka Project, http://www.mazurka.org.uk) list the audio onset times for all score beat events. This data is only used for evaluation purposes since our method uses audio but not score information. We report both the average and maximum alignment errors for each piece. The group of performances constitute all combinations of pairs within each piece. The errors are measured with a Euclidian distance between labelled \( B \) to the reference (See Figure 1 for an example) in terms of seconds between the ground truth labels from the closest alignment point. We first run our algorithm on the original music recordings without any alteration. Then, we manually cut different number of beats in the audio and test the algorithm again. In our experiments, we used 1000 particles and the thresholds mentioned in Section 3.2. Results are compared to a normal implementation of a particle filter that only uses equations (3) and (4).

Both methods perform very well on the audio recordings of Chopin’s Mazurkas. Results are compared to a particle filter that only uses Equation (4) to update the music position performed slightly better over our method which we believe can be attributed to the consideration of potential jumps by our model in which in fact no actual jump occurs.

<table>
<thead>
<tr>
<th>Errors</th>
<th>Proposed Method</th>
<th>Regular PF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Op. 5</td>
<td>0.061 1.267</td>
<td>0.080 1.796</td>
</tr>
<tr>
<td>Op. 7</td>
<td>0.116 1.267</td>
<td>0.100 1.717</td>
</tr>
<tr>
<td>Op. 21</td>
<td>0.081 0.660</td>
<td>0.081 0.595</td>
</tr>
<tr>
<td>Op. 30</td>
<td>0.069 0.316</td>
<td>0.059 0.314</td>
</tr>
<tr>
<td>Op. 63</td>
<td>0.108 1.175</td>
<td>0.101 1.272</td>
</tr>
<tr>
<td>Op. 67</td>
<td>0.094 0.564</td>
<td>0.074 0.581</td>
</tr>
<tr>
<td>Op. 88</td>
<td>0.135 0.941</td>
<td>0.103 0.619</td>
</tr>
</tbody>
</table>

Table 1: Average and maximum errors on the given list of Chopin’s Mazurkas. The proposed method in this paper is compared to our implementation of a regular particle filter. * only half of the pieces because in one piece we picked which was shorter than the rest.

<table>
<thead>
<tr>
<th>Errors</th>
<th>Proposed Method</th>
<th>Regular PF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Op. 5</td>
<td>0.248 2.270</td>
<td>0.032 2.870</td>
</tr>
<tr>
<td>Op. 7</td>
<td>0.065 0.839</td>
<td>4.099 8.664</td>
</tr>
<tr>
<td>Op. 21</td>
<td>0.116 2.731</td>
<td>0.100 2.787</td>
</tr>
<tr>
<td>Op. 24</td>
<td>0.093 1.168</td>
<td>0.101 1.110</td>
</tr>
<tr>
<td>Op. 30</td>
<td>0.126 2.764</td>
<td>0.426 2.142</td>
</tr>
<tr>
<td>Op. 63</td>
<td>0.175 1.755</td>
<td>0.392 2.724</td>
</tr>
<tr>
<td>Op. 67</td>
<td>0.104 0.797</td>
<td>0.192 0.797</td>
</tr>
<tr>
<td>Op. 88</td>
<td>0.298 2.180</td>
<td>1.281 3.974</td>
</tr>
</tbody>
</table>

Table 2: Average and maximum errors on the selected Mazurkas with 5 beats missing.
5. CONCLUSION

In this paper, we present a novel approach to improve the audio-to-audio alignment performance of causal alignment systems in the presence of either note-level or sectional performance differences. We have shown that our system is able to cater to alignment tasks in which there are some performance discrepancies between the reference and the one to be aligned. The proposed method addresses this problem without incurring significant performance degradation with the alignment of complete performances – those that have the same musical event sequences. The extension of this method to score-to-audio alignment is straightforward. This can be done through an annotated score or using MIDI audio synthesis. Our future work will concentrate on improving responsiveness of the system in order to track the missing notes. One possible way is to dynamically tune the parameters in the observation model by decreasing the number of particles labeled B when we are confident about the inferred music position and use more particles if we are less confident.

REFERENCES


BROWSING MUSIC AND SOUND USING GESTURES IN A SELF-ORGANIZED 3D SPACE

Gabrielle Odowichuk
University of Victoria
Department of Electrical and Computer Engineering

George Tzanetakis
University of Victoria
Department of Computer Science

ABSTRACT

As digital music and sound collections increase in size there has been a lot of work in developing novel interfaces for browsing them. Many of these interfaces rely on automatic content analysis techniques to create representations that reflect similarities between the music pieces or sounds in the collection. Representations in 3D have the potential to convey more information but can be difficult to navigate using the traditional ways of providing input to a computer such as a keyboard and mouse. Utilizing sensors capable of sensing motion in 3-dimensions, we propose a new system for browsing music in augmented reality. Our system places audio files in a virtual cube. The placement of the files into the cube is realized through the use of audio feature extraction and self-organizing maps (SOMs). The system is controlled using gestures, and sound spatialization is utilized to provide auditory cues about the topography of the music or sound collection.

1. INTRODUCTION

Advances in technology have drastically changed how we interact with music. The increasing capabilities of personal computers have allowed listeners access to digital music collections of significant size, and the number of available songs increases, searching and browsing through this music becomes difficult. The conventional hierarchy of “Artist-Album-Track” and the spreadsheet interface of music software such as iTunes is no longer the dominant way of organizing and navigating digital music collections. While this method is effective for finding a specific song when one knows exactly what song they are looking for, it does not allow for effective browsing through music collections when there is no specific target song. To address this issue, browsing interfaces that are based on organizing music tracks spatially based on their automatically computed similarity have been proposed. Content-based browsing has some advantages over traditional systems, many of which stem from the fact that users can browse music aurally, and no longer require a pictorial or textual representation. By removing the need for a key word representation, we can possibly access music that has no associated text, or text available only in a different language. This type of audio browsing can also be useful for music creators or videogame audio designers who need to sort through large collections of sound clips or sound effects. Accessing music information aurally makes sense intuitively, and even allows people with vision or motion disabilities improved access to the world of music [19]. We describe a novel interface for browsing music and sound collections based on automatically computed similarity, spatially arranging the audio files in 3D using self-organizing maps (SOMs), and browsing the sonified space using 3D gestural controllers.

2. RELATED WORK

Novel interfaces for browsing music began to appear about ten years ago with SOMs being one of the first algorithms to be used for music clustering and visualization [4]. The early development of applications demonstrating these concepts such as the Sonic Browser [9], Marsyas 3D [18] and Musescap [16] was fueled by advances in the field of Music Information Retrieval (MIR). Each system uses direct sonification rather than button triggered playback as a means of music browsing to create a continuous stream of sound while navigating the music space. In Pampalk, Dixon and Widmer [13] and Knees et al [5], a visualization of the organized music collection is proposed in which the clustered songs are represented as islands, where the height of each island is relative to the number of songs in each cluster, and the terrain itself is based on a 2D SOM. In each of these applications, navigation is achieved using a mouse or joystick. In Ness et al. [11], the authors explored the use of various controllers for interfacing with self-organized music collections. These controllers include multi-touch smartphones, motion trackers like the wii-mote, and web-based applications. While advances in self-organized browsing progressed, the use of augmented reality in musical applications was being developed [14]. Often, augmented reality (AR) is understood to be related to display technologies. However, AR can be applied to any sense, including hearing. In Azuma et al. [1], a mixed-reality continuum is presented, with Augmented reality defined as virtual objects added to a real space. Another good example of early combinations of self-organized music collections and augmented virtual spaces is the “Search Inside the Music” program [7]. This application allows users to browse through a virtual 3D space of songs and also showed the songs on each album visualized with the cover art. The main contribution of this work is the utilization of gestural 3D control for interacting with a 3D self-organized map of music.
3. ORGANIZING MUSIC IN A 3D SPACE

One of the main goals of Music Information Retrieval is to approximately model the concept of "similarity" in music. Similarity can be determined by using manually assigned metadata, however MIR often also focuses on extracting features directly from the audio signal. A variety of methods have been proposed in self-organized music browsers to project high-dimensional feature data onto a lower dimensionality for visualization, such as Principal Component Analysis. Although there are many dimensionality reduction methods, the most common approach to organizing music collections is that of the Self-Organizing Map [6]. In this case, a set of features is extracted from an audio file, producing a single high-dimensional feature vector representing each song. The feature vector corresponding to a piece of music or a sound is then mapped to a corresponding set of coordinates in a discrete grid. Feature vectors from similar audio files will be mapped either to the same grid location or neighbouring ones. The resulting map reflects both an organization of the data into clusters as well as a mapping that preserves the topology of the original feature space.

The goal of feature extraction is to produce a vector of numbers known as features that represent a piece of audio. By choosing how the vectors are computed, we are able to come up with numbers that are similar when they correspond to perceptually similar sounds or music tracks. As described in [17], we extract features such as Flux, Roloff, MFFCs (Mel-Frequency Cepstral Coefficients), pitch histograms andhythm-based features. These audio features are extracted for very short periods of audio (usually under 25ms). An entire song would therefore have an array of numbers for each feature, depicting how these features change over time. To model large collections of songs, this sequence of feature vectors representing each song needs to be summarized into a single feature vector characterizing the music at the song level. To shorten the length of these feature vectors and simplify the calculations each sequence of a particular feature is summarized down to two single values: the mean and standard deviations each sequence of a particular feature is summarized as the euclidian distance between the song features and node weights, as shown in Equation 3. The smallest distance corresponds to best-matching node or best-matching unit (BMU). Now each node in the vicinity of the BMU is updated with a new set of weights, adjusted to become more like our BMU. Equation 4 described how this adjustment is made. V(t) is the feature vector, W(t) is the weights vector, and L(t) is a learning function, which decays over time and allows the organizing algorithm to converge.

\[ d = \sum (V_i - W_k)^2 \]

\[ W(t+1) = W(t) + \eta L(t)(V(t) - W(t)) \]

The training process involves selecting a song to train the map with and determining which node represents that song best. Similarity between songs and nodes is calculated as the euclidian distance between the song feature and node weights, as shown in Equation 3. The smallest distance corresponds to best-matching node or best-matching unit (BMU). Now each node in the vicinity of the BMU is updated with a new set of weights, adjusted to become more like our BMU. Equation 4 described how this adjustment is made. V(t) is the feature vector, W(t) is the weights vector, and L(t) is a learning function, which decays over time and allows the organizing algorithm to converge.

The self-organizing map is a type of artificial neural network, meaning that it is inspired by interactions between biological neurons. Our neural network begins with a set of objects referred to as nodes. Each node has an associated weight vector, W, as shown in equation 2, and spatial placement \( P = [x, y, z] \). Although the nodes in figure 1 have been spaced evenly within a cube, these nodes could hypothetically be placed in other, more arbitrary formations. Initially, the weights of each node are set randomly. As the organization process progresses, the weights of each node will begin to align more closely with their neighbours and also more closely with our song features. This process is depicted in Figure 1, where each node has weight vector visualized as a colour. Initially, the weights shown in this figure are random (Figure 1a). As the SOM is trained with 8 distinct colours, the weights of each node become organized (Figure 1b).

The SOM is trained with 8 distinct colours, the weights of each node become organized (Figure 1b).

The training process involves selecting a song to train the map with and determining which node represents that song best. Similarity between songs and nodes is calculated as the euclidian distance between the song feature and node weights, as shown in Equation 3. The smallest distance corresponds to best-matching node or best-matching unit (BMU). Now each node in the vicinity of the BMU is updated with a new set of weights, adjusted to become more like our BMU. Equation 4 described how this adjustment is made. V(t) is the feature vector, W(t) is the weights vector, and L(t) is a learning function, which decays over time and allows the organizing algorithm to converge.

\[ d = \sum (V_i - W_k)^2 \]

\[ W(t+1) = W(t) + \eta L(t)(V(t) - W(t)) \]

By iteratively training our SOM, our resulting nodes reside in a space where nearby nodes have similar weight vectors. Each song is mapped to the most similar node, resulting in a set of songs residing in a space where nearby songs have similar feature vectors. In Figure 2, you can see that songs from similar genres will tend to be near one another. Note that the self-organizing map algorithm has no knowledge of the genre labels and their spatial organization is an emergent property of the mapping and the underlying audio features.

4. NAVIGATING THROUGH THE COLLECTION

Once our songs have been organized into a virtual 3D space, user interaction becomes a significant consideration. Since the use of 3D sensors was one of the primary motivations behind this work, our focus has been on using sensors capable of reporting gesturally-produced position data for two or more points. How we go about using that captured motion is another point of discussion, and we present here only a few of the many possible ideas for expanding and refining user-interaction. Previous work has been done into user interaction with 2D visualizations for music browsing [8], and similar concepts can be applied to the 3D scenario. We utilize two controllers: the radiodrum and the Kinect.

The radiodrum [2] is a controller with a long history known mostly in the computer music community. It is a music controller that rapidly tracks the position of the tips of two drumsticks in 3D space, and has been used to navigate and fade between two pieces of music in self-organizing maps [10]. The recent popularity of the XBox Kinect, an infrared 3D motion sensing device, has been a catalyst into further researching intuitive intelligent uses of gestural control. The Kinect provides a type of sensing in some ways similar to that of the radiodrum. It enables tracking of the hands and body of a user and does not require any hardware to be touched by the user. The radiodrum has much higher temporal precision and therefore feels more interactive. However it has the disadvantage that it is not widely available, is expensive and does not have as much spatial resolution as the Kinect. We believe that for this application scenario the Kinect is the better choice as it is mass produced, cheap, and enables very natural interaction. On the other hand the radiodrum has helped us understand better the timing requirements for such an interface. Using our sensors for control data we want to sonify the organized sounds as we move our 3D cursor about. The simplest way to do this is to simply play back songs from whichever node is currently closest to one cursor, and only one song plays back at a time. The other hand could then be free to perform other types of control gestures. Another content-aware browser [15] presented a different method for playing back songs. In this case, the user can navigate the space using a scrubber bar and the proximity of an encompassing circle, and any songs within the circle will play simultaneously. To modify this method for our purposes, the two cursors were made to act as the bounding points for a variable-size sphere. Node weights positions within the user-controlled sphere are sonified, with a gain relative to their nearest to the center of the sphere. Once the cursor data from the sensors is mapped to playback in the auditory representation of our sound collection we need a richer gestural language to enhance the user control. For example, once music exploration is complete and the user has found a song they would like to listen to, they will want to listen to that song and stop searching. Our simple way of implementing this functionality is to use timers, so that if we preview a song for longer than a set duration it will trigger song playback. Each node is sonified with a loudness based on its position relative to the cursor. By creating listening points that surround our cursor, we are able to perform multi-channel panning. As shown in figure 3, one controller is centered on either side of the user’s current position, representing the two listening points required for a stereo reproduction. This spatialization gives an aural sense of space and direction for navigation of our music collection.

5. IMPLEMENTATION

The hardware required for this music browser is simple: a controller, a computer, and a sound system. Figure 3 demonstrates the application design and interactions between the devices and software libraries. The C++ project uses a creative toolbox called openFrameworks1, which allows easy access to other libraries like openGL to create visualizations, Marsyas2 for audio feature extraction, and

---

1http://www.openframeworks.cc
2http://marsyas.info
6. FUTURE WORK AND SUMMARY

Future work will involve performing user evaluations that could help to answer questions about browsing music with this system. Three-dimensional SOMs have the possibility to represent richer topological spaces, reflecting more accurately the relationship between songs in our music collection. Furthermore, using 3D gesture-based controllers to navigate a 3D space probably offer advantages over using a joystick or other 2D controllers. However, without the proper evaluation provided by a user study any claims we can make are purely speculative. Further evaluation of this system is required, in which the time it takes to complete tasks of browsing for certain music will be measured. Quantitative comparisons between 3D and 2D SOMs can also be performed, where the distance between similar songs are compared for the same set of songs.

The self-organized map has become a popular method for organizing songs based on similarity. This type of music browser not only reflects the way that how we interact with music is changing, it also reflects how our interaction with technology and computers is changing. By expanding previous work with self-organized music collections and adding a third dimension, it is possible to convey additional information and browse extra songs. Additionally, navigating this type of map is a good example of the advantages 3D gestural sensors like the radiodrum and the Kinect have in specific control contexts and the more naturally, navigating this type of map is a good example of the possibilities for sound creation and musical expression.

7. REFERENCES


ADVANCING EXPERT HUMAN-COMPUTER INTERACTION THROUGH MUSIC

Benjamin D. Smith
Gay E. Garnett
University of Illinois at Urbana-Champaign
Illinois Informatics Institute, National Center for Supercomputing Applications

ABSTRACT

One of the most important challenges for computing over the next decade is discovering ways to augment and extend human control over ever more powerful, complex, and numerous devices and software systems. New high-dimensional input devices and control systems provide these affordances, but require extensive practice and learning on the part of the user. This paper describes a system created to leverage existing human expertise with a complex, highly dimensional interface, in the form of a trained violinist and violin. A machine listening model is employed to provide the musician and user with direct control over a complex simulation running on a high-performance computing system.

1. INTRODUCTION

As software systems and cloud computing provide more and more powerful, complex computing tools it becomes necessary to discover new ways of augmenting and extending human control while retaining human judgment and analysis of complex situations. Popular computer-human interfaces for personal computing, such as WIMP elements and hardware, are typically designed generically to support as many different users and uses as possible. Today, physical computer interactions are becoming increasingly multidimensional, as advances in technology and understanding encourage movement away from conventional human interface devices towards hands-free gestural controllers. Additionally, developments in physical computer interaction systems, such as contemporary game console controllers and the newest 3D imaging cameras, are enabling new, specialized modes of interaction. Although these systems may not be as general, versatile, or approachable, the rewards in terms of capabilities in specific domains can outweigh the requirements of expertise and loss of generality. Given a user’s willingness to practice and learn the new interface, such a system can give the user new levels of control and power.

Physical devices with extreme learning curves have long been in use, and we take the musical instrument, the violin, as a first class example. While being notoriously difficult to learn, the violin presents vast possibilities for sound creation and musical expression in the hands of a practiced master, audibly and visibly demonstrating the human capacity for using complex interfaces successfully. The typical professional musician today will spend upwards of one thousand hours per annum developing and maintaining their proficiency on the instrument. We then ask: what could this level of dedication lead to if the target was a computational control interface?

Another dominant trend in computing is increasing power and capabilities for complex data processing, data mining, simulations, and visualizations. This is especially apparent in high performance computing (HPC), with the recent completion and activation of several petaFLOP (10^18 floating-point-operations-per-second) capable systems. In such a setting a user with extensive practice and skill could become a virtuoso of the computer, capable of transforming and manipulating vast data sets in novel ways, enabling new modalities of operation and knowledge discovery.

To explore this possibility we describe a novel interface to enable high-dimensional, continuous input and control, leveraging the very precise gestural manipulations of professional musicians. This new interface builds on machine listening techniques, applying unsupervised machine learning algorithms to develop a machine model of expert human music listening. This setup presupposes expert knowledge of a musical instrument on the part of the user, effectively transforming the acoustic instrument into a digital control device. The result is the concurrent, precise, and direct manipulation of many independent parameters of a complex data simulation running on a supercomputer, employing the violin as a sophisticated, multimodal, tangible interface.

2. MOTIVATION

As available computing power and network bandwidth increase it gradually becomes possible to transform practices of high-performance computing and scientific data processing from offline, batch oriented modes to real-time interactive data mining and simulation. Interactive high performance computing is a growing area of research, demanding new systems and solutions [6]. The extreme level of sophistication that these interactive supercomputer jobs can achieve suggests the need for highly dimensional, tangible or gestural control interfaces in order to fully exploit the available computing power. To date, little work has been published in this direction.
Interfaces based on expert use of existing physical devices, and which are worthy of extreme practice must add as little additional cognitive load as possible. Towards this end it is desirable to develop an automated understanding of the current practice (i.e. making music) identifying the preexisting repertoire of gestures. Machine learning algorithms have proven successful in this domain, categorizing musical and physical gestures with sufficient accuracy and flexibility [5, 6]. Additionally, some approaches enable tracking of gestural qualities and styles [4] built of more or less subtle nuances and variations in execution. These elements of expressivity contain additional information that can be leveraged as a control source, potentially giving the computer access to a type of emotional understanding of the input.

3. MUSICAL STRUCTURE

Despite the many parallels that exist between spoken language and music, each domain presents distinctly different problems for machine listening. While language is highly semantic and denotative, music is more referential and connotative. Composers and theorists of classical musics typically describe structures and developments in music through relational patterns in the musical material (as the result of a melody or a modulation to a new tonal area). These relationships are understood to give meaning, especially when realized as the initial state for a new category. In this way the known category incorporates the new input (a smoothed simple distance measure is employed to find the best fit. Concurrent dependencies in the update functions. As more entities are added the complexity of the algorithm proceeds as \(O(n^2)\), but the update calculations are shared uniformly across all available CPUs. [8] found unoptimized performance on the order of 50 updates per second with 10,000 entities on 128 CPUs, which was sufficient for proof of concept. We refer the reader to [8] for further details of the simulation implementation.

The data set from the simulation is rendered as a projected 3D visualization, displaying the flocking entities as uniform abstract shapes, rotating and moving dynamically in a virtual environment. Displaying the modeled behavior of the entities in a fashion analogous to flocking creatures observed in the physical world is a natural mapping given the nature of the simulation algorithm.

4. ARCHITECTURE

The platform employed for the prototype is a distributed system comprising a supercomputer in combination with consumer grade desktop computers (fig. 2). The Abe supercomputer, housed at the National Center for Supercomputing Applications (NCSA), was used for the simulation core. A Mac Pro desktop computer was used to process and analyze the sound input and display the state of the simulation.

The system being controlled by the musician in this prototype was an agent based flocking simulation, modeling behaviors observed in nature as exhibited by herding and schooling creatures [11]. The non-centralized, particle nature of the algorithm is a distinct computational advantage, making it readily parallelizable and scalable. Each simulated entity maintains its own state and behavioral coefficients and parameters, updating its properties based on its immediately proximal neighboring entities with no concurrent dependencies in the update functions. As more entities are added the complexity of the algorithm proceeds as \(O(n^2)\), but the update calculations are shared uniformly across all available CPUs. [8] found unoptimized performance on the order of 50 updates per second with 10,000 entities on 128 CPUs, which was sufficient for proof of concept. We refer the reader to [8] for further details of the simulation implementation.

The data set from the simulation is rendered as a projected 3D visualization, displaying the flocking entities as uniform abstract shapes, rotating and moving dynamically in a virtual environment. Displaying the modeled behavior of the entities in a fashion analogous to flocking creatures observed in the physical world is a natural mapping given the nature of the simulation algorithm.

5. DISCUSSION

Testing of this system was conducted with one of the authors as a participatory designer and target expert user. Evaluation was carried out continuously during design and implementation, ensuring that the system met targets of usability, functionality, and accessibility to a trained musician. The resulting prototype enables clear demonstration of the musical control scheme, allowing the musician to make easily perceptible manipulations of the sound data.

While the ultimate goal is a transparent leveraging of musical playing, the current system places a not insignificant level of cognitive load on the user. Musicians, especially when improvising, privilege their intuitions and a sense of emotional communication [1]. However, operating our system requires a different mental orientation that forces the musical user to form a very conscious memory of what they played during the session. Unlike most user interfaces designed today our system does not have a preconception of what input commands will be employed (as long as they are sonic in nature). This requires that the user define and learn their own control scheme. Since the ART network is learning on-the-fly, returning to the same musical place (in the sonic feature space) is necessary to reproduce the
same outcome. On many occasions this proved difficult for the user, who struggled to recall their earlier improvisation.

Most of the observed difficulties in the use of the system centered around issues of intent and conceptual mapping. The complexity and adaptivity of the machine learning requires significant cognitive resources on the part of the user, who typically desires to exert precise control over the simulation. However, this is dependent on a precise understanding of the analyzed inputs (pitch, brightness, noisiness, etc.) which often are not heard focally by the player. This is an inherent challenge of the design, as the system is intended to respond intelligently and naturally, without requiring a strong cognitive model on the player’s part. When the musician was able to ‘let go’ and focus on the music, creating a compelling sequence of sonic events, the control of the system become much more facile and transparent. This typically lead to the most rewarding experiences.

The primary user found our system promising and exciting overall. The above difficulties not withstanding, we were able to afford control of a complex, dynamic simulation for an expertly trained musician. Even failure in the control was mostly undetectable by any except the user, who can recognize a mismatch between intent and result. Additionally, this system leverages the intrinsic value and reward found in the actions employed in this control (i.e. making music), which could encourage sustained use of such designs.

While the information extraction provided by the ART network is very promising, employing this data in an appropriate and meaningful fashion remains a challenge. Ideally this mapping will be constructed dynamically and intelligently, serving to reduce the aforementioned cognitive load by making apparently natural or intuitive choices. This might be accomplished by associating some sense of affect with a given input mapping, such as by tying bold, heroic melodies to strong, quick changes in the simulation space, and mapping gentle, melancholic melodies to subtle, continuous movements (of course these mappings would be crafted uniquely for each given user, as descriptions of musical affect can vary dramatically from individual to individual).

We have shown that a complex data simulation, based on scientific observations, running on a supercomputer, can be dynamically controlled in real-time by a trained musician, demonstrating the potential for highly specialized, multidimensional expert interfaces. The prototype evaluates well in terms of control precision and rate of interaction, but demands a high level of extra-musical awareness and focus. Someday systems affording rich expert control will be prevalent and anticipating these situations can lead to the design of the best natural interactions possible, allowing users to learn new interfaces in the same way a violinist learns to play the violin.

6. REFERENCES


[6] Kun-Ting Tsai National Chiao Tung University Master Program of sound and music Innovative Technologies

ABSTRACT

“SurrSound” is an interactive spatial sound installation, this title is choosen to represent both the meaning of surround and sound. The appearance of this installation is very similar to a big fruit hanging upside down on a tree in the jungle. It allows user to control the directions of sound moving around the space surrounded by 16 loudspeakers; The sound could either be the pre-recorded samples or live recorded segments by users. To record the voice in real-time, users need to squeeze the ball to trigger the bend sensor to turn on the mini microphone. “SurrSound” consists of 16 balls, each of them made by four parts: The joystick for the control of the sound directions, the mini microphone is for recording, the bend sensor switch, and the ADXL335 three axis accelerometer for modulating the voice. In this installation, sixteen loudspeakers are used to fill the space in order to simulate the natural situation of sound moving in our environment or to offer a brilliant sound spatialization system for acoustical music performance. The “SurrSound” enables user to manipulate their voice interactively by user spatialization. The presented installation also aims to give users the experience of being surrounded by the fascinating aural illusions.

Keyword: Interactive spatial sound installation, Max/ MSP, Arduino, sixteen speaker, sphere, sequence two ULTRAGAIN PRO-8 DIGITAL ADA8000, and the computer and speakers were connected via the cable hubs. “SurrSound” was expected to be intergrated into the space, therefore, sixteen loudspeakers were set up in the room to make the surrounding sound field. To further increase the interactivity of “SurrSound”, sixteen balls average distributed around the space were also constructed, giving people the feeling of staying in the nature environment. [4]
Max/MSP will receive the signals from sensors, and transmit sounds to 16 loudspeakers via 16 channel interface. In interactive mode, people can toss, throw and press the ball. Moreover, “SurrSound” can also be played by multi-users. It allows one user to record his voice while the other people is modulating the recorded sound at the same time by tossing the ball. This gives users the free reins to create brilliant music together. (Figure 5.)

3.2. Appearance and Spatial structure

We make a small acrylic box that allows us to put all of our circuits inside the box. Those circuits include the ADXL335 three axis accelerometer (Figure 7.), LED and microphone. We use the soft rubber ball to encase the box and iron plates and hold the box in the middle of the ball. (Figure 8.) As being a innovative spatial sound installation, SurrSound is constructed to be both aural and visual pleasing by creating a “Fruit-like” appearance for the components. (Figure 9.)

3.3. Software Design

The joystick signals were sent to Arduino and then to computer. Max/MSP receives one signal per fifty micro second from Arduino. (Figure 13.)

The direction of signals are transmitted to Max/MSP. The patch we use is “Ambisonics” [5]. This patch can control the sound surrounding in the space. This object: encodes signalsto setup the all position in the space, and decodes the signalsto set the position of loudspeakers. (Figure 17.)

3.4. Hardware Design

We put the bend sensor into the ball and stick it onto the inner wall of the ball. Three methods were used to test the position of bend sensor where we stick. As shown in Figure 12, both the first one and the second one are not significant for bending the bend sensor in all angles. For the third method, we stickly the bend sensor close to the bottom of the gap edge. (Figure 22.) The approach makes the bend sensor to be averagely pressed from all angles. (Figure 23.)

The ADXL335 three axis accelerometer were installed inside the sphere. Sounds will be produced when the sphere is being shaken. By throwing the ball, the sound can be modulated. In addition, we plug in a mini microphone (Figure 26.) to record the sound. This enables the sound around the space to be recorded as the ball moving through the environment. For the direction control, we use joystick. The joystick can locate 8 direction. (Figure 27.) The is the stationary status, and is the status of up, down, left, right. If right and up be trigger, it means the direction is the upper right corner. And so on. (Figure 28.)
"SurSound" is expected to have wireless via bluetooth or network in the coming upgrades.

**Figure 30.** The microphone.

**Figure 31.** Up, Down, Left, Right.

**Figure 32.** The interactive situation. The user is Ping-Jui Huang.

**5. CITATIONS & REFERENCES**


1 Thanks Tzu-Heng Chi and Ping-Jui Huang for their contributions to the part of project.

**ABSTRACT**

In this paper, we describe an interactive graphical user interface for online motivic analysis. We employ a string-based pattern detection approach based on 8-grain similarity matrices. Thus, substrings of various length as well as their variations are extracted from a given MIDI file. The detected types of variations range from melodic repetitions, pitch retrogrades and/or inversions to rhythmic repetitions and their retrograde. To allow for a fast visual analysis of the extracted patterns, all patterns as well as their specific occurrence positions are displayed in a piano roll based score representation. Furthermore, hierarchical subpattern relationships between the motive candidates are calculated and made available in the user interface.

**1. INTRODUCTION**

A complete motivic analysis involves three steps [11]. First, motives in a given piece of music have to be identified. Second, the repetitions, variations, and developments of these motives are detected and described. Finally, the function and meaning of the motives in the context of the piece are determined. Several approaches for the computer-based treatment of the first two steps of motivic analysis (i.e., motif identification and detection of variations) have been proposed. A common approach in motif identification is to search for musical patterns that are repeated nearly unvaried. As not all of the detected patterns may actually constitute motives, this task is commonly referred to as pattern detection. Most of the existing approaches to pattern detection are either string-based [6, 7, 8] or geometric [10, 15].

While string-based approaches are constrained to monophonic pieces, geometric approaches are applicable to arbitrary polyphonic pieces and are capable of detecting non-consecutive patterns extending to several voices. Usually, geometric approaches only consider patterns that are repeated unmodified—except for time translation and pitch transposition. In contrast, string-based approaches can easily account for some types of pattern variations. For instance, Adiloglu et al. [1] proposed an approach where repetitions as well as pattern occurrences in inversion are detected. Furthermore, small melodic variations of the original pattern are considered, while rhythm is disregarded altogether. Additionally, they extended their framework and computed substring relationships between musical patterns [2]. In [14, 16], similarity matrices are employed to detect exact melodic repetitions of patterns as well as patterns appearing in retrograde and/or inversion. To allow for small interval variations often required in tonal compositions a generic interval division was chosen.

Several different approaches for the detection of variations of a given pattern were proposed, e.g., [4, 13]. As this paper mainly focuses on pattern detection, we will not give a detailed account of existing work in this area. Although a lot of approaches for computer-aided motivic analysis have been developed, only a few graphical user interfaces have been proposed. In [3, 5] an interactive visualization is presented. The authors implemented three representations: weight functions, motivic evolution trees and melodic clustering. All of them provide information on the detected variations of given patterns. Furthermore, the score of the motif candidates can be accessed. However, no full score visualization highlighting the motives and their variations is available. Collins published a video of a tool using his automatic pattern discovery method [9]. With this tool, arbitrary symbolic score data encoded in the Humdrum file format can be opened for analysis. The detected patterns are ranked and made available for visualization in a rendered score visualization. However, the patterns and their occurrences can only be browsed successively and are not made available simultaneously.

In this paper, we present the MOTIF VIEWER, a prototypical tool for the visualization of automatically detected repeating patterns in a musical score. Using similarity matrices exact repetitions and repetitions in retrograde and/or inversion are made available. In contrast to most approaches, both pitch-intervals and durations are considered.
ered individually and in combination to account for different types of variation. Following the idea of subsequence relations between musical patterns presented by Adiloglu and Obermayer [2], a pattern hierarchy is calculated. The proposed system provides visual access to the detected patterns, their occurrences as well as the subpattern relationships.

The remainder of this paper is organized as follows. First, we give an algebraic formulation of the problem of motivic analysis (Section 2). In Section 3 we present the employed pattern detection method in more detail. The graphical user interface for hierarchical motivic analysis is described in Section 4. Section 5 concludes the paper with an outlook on future work.

2. ALGEBRAIC FORMALIZATION

The terminology we use largely follows [4].

The most important parameters describing a musical note are onset time, pitch, and duration. Using those three parameters, a piece of music M can roughly be represented by a finite set of notes \( \{ p, d \} \). We are going to study repeated patterns within M. In general, a pattern in M is just a non-empty subset of \( M \) of \( M \). To specify repetitions, we let the counter point group CP act on the (idealized) universe \( N = \mathbb{R} \times \mathbb{R} \times \mathbb{R} \) of all notes. This group is generated by all time and pitch shifts together with time and pitch inversion. In this group, every element \( g \) is specified by \( g = (x, y, \varepsilon, p) \) with \( \varepsilon \in \{ \pm 1 \} \) and \( t, x \in \mathbb{R} \). Such a group element \( g \) shifts the note \( [p, d, q] \) to \( g \cdot [p, d, q] = [x + p, y + d, \varepsilon \cdot p] \). Thus, \( \varepsilon = -1 \) induces a time inversion (retrograde), whereas \( \varepsilon = 1 \) corresponds to a pitch inversion. By setting both \( \varepsilon \) and \( \varepsilon \) to 1 retrograde inversions of musical patterns are described. The action of CP on N induces an action of CP on subsets of N via \( g \cdot X = \{ x \in X \} \). If \( \mathbb{P} \) is a subgroup of CP and \( M' \) is a pattern in M, then \( \mathbb{P} \cdot M = \{ g \cdot M' \mid g \in \mathbb{P} \} \) is the set of all \( \mathbb{P} \)-transformations of \( M' \).\footnote{By definition, a note \([ p, d, q] \) is active in the time interval \([ y, y+d) \).}

In this paper, we are concerned with repetitive patterns in M, i.e., we consider subsequences \( M' : = \{ [p_1, d_1], [p_2, d_2], \ldots, [p_n, d_n] \mid 1 \leq i \leq n \} \). The positions of patterns in \( M' \) are called \( M \)-invariant subsequences of \( M \). From this definition, the two patterns and their occurrences are visualized, see (c).

3. PATTERN DETECTION

Motivic analysis is concerned with monophonic note sets \( M \). In such sets, no two notes are active\footnote{A note is said to be active if it is present in an occurrence, otherwise it is inactive.} at the same time. Thus a monophonic note set \( M \) can be viewed as a sequence of notes ordered by onset times: \( \{ [p_1, d_1], \ldots, [p_n, d_n] \} \) with \( \varepsilon \leq d_1 < \cdots < d_n \) and \( t + d \leq \varepsilon \).

Figure 1. Hierarchical motifs in the first measure of the Invention No. 1 in C major by J. S. Bach. Motives \( a \) and \( b \) are submotives of \( M \). As they are also repeated outside the context of \( S \) in the remainder of the score, they are independent.

Figure 2. Upper half of the 3-gram similarity matrix for the melody of the first movement of J. S. Bach (BWV 772) (left) and enlargement of the marked region (right). Matrix entries \( S_{ijkl} \) are depicted in black, while zero-entries are white.

Furthermore, subpattern relationships are easily deduced from \( S_{ijkl} \) as well. Given a diagonal \( S_{ii} \) we take the position interval \([ i : i+k] \) along the first dimension of \( S_{ijkl} \). The pattern represented by the diagonal element is a subpattern of \( S_{ijkl} \) if its position interval constitutes a proper subset of the position interval of another diagonal in \( S_{ijkl} \). By calculating the self-similarity matrix \( S_{SIM} \), \( D \) of \( S_{ijkl} \) with positions \( l \) and \( k \) and \( \ell \) and \( \ell + r \) with \( \ell \leq \ell + r \) are considered by calculating the \( D_{ijkl} \) matrices and \( S_{ijkl} \) are compared. Finally, retrogrades are detected by combining the two. As last variation type, rhythmic pattern retrogrades are considered by calculating the \( \ell_{MIN} \) or \( \ell_{MAX} \) or \( \ell_{MIN} \) matrices.

Obviously, a pattern might be detected in more than one of these similarity matrices. Equally, an occurrence might match with respect to several variation types. All information on pattern occurrence positions as well as their variation types are stored. Based on this information, the detected patterns are ranked using a ranking strategy that considers pattern length, number of pattern repetitions as well as their respective variation types. Given a pattern \( w \) and its occurrences \( \mathbf{a}_{1}, \ldots, \mathbf{a}_{k} \), we calculate its ranking value \( r(w) = \max \{ S_{ij} \} \), where \( a_{i} = \mathbf{a}_{i} \), \( b_{i} = \mathbf{a}_{i} \) or \( b_{i} = \mathbf{a}_{i} \) describes the importance of occurrence \( o \). Here, \( b_{i} = \mathbf{a}_{i} \) or \( b_{i} = \mathbf{a}_{i} \) represent the binary information whether \( o \) is a rhythmic or melodic variation. Moreover, the variable weights \( \alpha_{i} \) and \( \beta_{i} \) influence the individual impact of the two types of variations.

4. MOTIFVIEWER

With the MOTIFVIEWER, we present a graphical user interface for computer-aided motivic analysis. The system performs online pattern detection as outlined in Section 3 and offers an interactive visualization of the detected patterns and their variations. By including a melody extraction algorithm, the MOTIFVIEWER can process both monophonic and polyphonic pieces.
The MOTIF VIEWER interface is shown in Figure 3a. On the left side, the user can manipulate various parameters (e.g., minimal and maximal pattern length and the ranking weights \( w_a \) and \( w_f \)). Furthermore, several controls allow for an interactive access to the detection results. Upon opening a MIDI file for analysis, the musical data is visualized as piano roll. In addition, the detected patterns are highlighted in the music data. A piece of music usually contains a vast amount of patterns (of different length) and each pattern can in turn reoccur multiple times. To maintain usability, the MOTIF VIEWER offers two view modes for patterns of a given length \( f \). In an overview, the beginnings for all patterns of length \( f \) are marked in the score (Figure 3a). Upon selecting a pattern beginning the MOTIF VIEWER switches to a detailed view showing the full pattern as well as all its occurrences (see Figure 3b). In this view, the matching types of each occurrence are color-coded. The user can easily switch back and forth between these two views. Furthermore, a different pattern length can be selected to explore the extracted patterns (of that length).

In addition, the MOTIF VIEWER provides a list of all detected patterns (including the calculated pattern ranking) as well as a table describing the pattern hierarchy of the given piece of music (Figures 3d and 3e). These lists can be used to easily access specific patterns or to explore the exact occurrences of a pattern and its subpatterns (Figure 3c). Finally, the interface offers the possibility of selecting the types of pattern variations to be visualized. Thus, the user can, e.g., decide to concentrate on melodic and rhythmic repetitions of a pattern while ignoring retrogrades and/or inversions.

5. OUTLOOK

The MOTIF VIEWER is designed as an interactive graphical user interface for computer-aided motivic analysis. Currently, a pattern detection algorithm extracts motif candidates as well as their occurrence positions from a given MIDI file. For the first prototype, we decided on a piano-roll based visualization. This geometric type of music visualization is easily graspable for non-musicians and therefore allows a universal application of the system. However, to allow for a more musico-logical approach to motivic analysis, we currently work on supporting other symbolic music formats, like MusicXML or the Humdrum file format, as well as a conventional score visualization. In addition, for the interactive motif and motive hierarchy visualizations that support computer-aided motivic analysis will be explored.

Besides exact repetitions of a pattern, variations and developments thereof should be calculated as well. The presented approach already considers some motivic variations (e.g., tonal adaptation, inversion, retrograde, and rhythmic patterns) while several others are ignored (e.g., embellishment or augmentation). In the future, we plan on expanding the functional range of the MOTIF VIEWER by supporting further types of motivic variations.

6. REFERENCES


PARTIAL TRACKING IN TWO STEPS

Adám Siska

sales@sadam.hu

ABSTRACT

This paper proposes a new algorithm for the creation of spectral envelopes in the context of additive analysis of recorded sounds. The key idea behind this method is the recognition, that the behaviour of a partial is different on the microscopic and the macroscopic level. Thus, our suggested process divides the envelope matching procedure into a micro-level and a macro-level analysis. Our current results are promising, however, further investigation is needed to understand every detail of this method.

1. INTRODUCTION

Johannes Kretz’s KLANGPILOT 3 environment [2] is a real-time Computer Aided Composition tool that addresses the issue of including spectral qualities (amongst them, synthesis parameters) in a composition – particularly, in its score. This software has an extensive synthesiser module allowing an arbitrary combination of additive, subtractive, and FDF (fonction d’onde formantique) synthesis methods.

As the synthesis possibilities of the system started to grow, efficient ways had to be developed to ease the composers’ task of virtual instrument creation. It was a plausible idea to build an additive analysis module which would create additive synthesis-based instruments that the users could modify accordingly to their own purposes. During the implementation of this module, we realized that even partial tracking principle accessible for us was based on the same method of taking an initial number of tonal peaks and trying to extend them as long as possible – which is, on the contrary, that partials won’t share the same aspects on the macroscopic scale of musically perceivable durations than on the microscopic level of a few milliseconds. This led us to the development of a new method for partial tracking, which we present in this article.

The main difference between these aspects lies in predictability. On very short-term it is not hard to predict the future (or past) behaviour of a partial. However, due to the complexity of most arbitrary sounds, one would always find such sudden changes in the spectrum of a real sound which are totally impossible to predict using only past information. Following this idea, we split the envelope following procedure into two separate tasks. On the micro-level, we create short envelope chunks that follow some clear trend. Then, on the macro-level we merge these segments into bigger partials considering only the similarity of these chunks.

2. ADDITIVE ANALYSIS

Additive synthesis approximates a target sound by adding together (a given number of) sinusoids

\[ s(t) = \sum A_i(t) \sin(2\pi f_i(t)) \]

where \( A_i(t) \) and \( f_i(t) \) are the amplitude and frequency trajectories of each partial, respectively [10]. An additive analysis method must, therefore, extract these partials from any arbitrary sound. The most convenient way of doing this consists normally of three main steps. First, one has to decompose the sound into a dataset whose elements carry meaningful information in terms of amplitude and frequency. Shortly, Fourier transforms (STFT) are particularly popular for this task, although other methods – e.g., Discrete Wavelet Transform – could be adopted as well [6]. As a next step, one would filter this dataset to extract the most important data points that contribute to the sound, which we would call tonal peaks. Finally, one would deduce the desired trajectories using these peaks. 2.1. Preparatory Steps

For the first step, our analysis tool uses STFT technique with arbitrary-size analysis windows, using zero-padding to reach an adequate (being a power of 2) FFT-window size. To find the tonal peaks, we followed the parabolic interpolation strategy described in [11]. Any alteration – e.g., due to psychoacoustical reasons – to the obtained peaks should be applied at this point! At the end of this step, all tonal peaks that we found are collected into a set that we called peak pool.

Before continuing, let us define the absolute and relative differences of the tonal peaks \( P_i(t_1, p_1) \) and \( P_j(t_2, p_2) \) (here, \( t_i \) is the level and \( p_i \) is the pitch of \( P_i \)). Now we can define the points

\[ P_1 = P_2 \Leftrightarrow P_1 - P_2 := \begin{cases} \hat{t}_1 - \hat{t}_2 & \text{if } 1 \text{ vs. } 2 \\ \hat{t}_2 - \hat{t}_1 & \text{if } 2 \text{ vs. } 1 \end{cases} \]

and the relative

\[ P_1 = P_2 \Leftrightarrow P_1 - P_2 := \begin{cases} \exp \left( \frac{\hat{t}_1 - \hat{t}_2}{\hat{t}_2} \right) & \text{if } 1 \text{ vs. } 2 \\ \exp \left( \frac{\hat{t}_2 - \hat{t}_1}{\hat{t}_1} \right) & \text{if } 2 \text{ vs. } 1 \end{cases} \]

These considerations will be added in a future release of KLANGEPILOT.
as the absolute difference of two peaks. Following these definitions, we can define the absolute and relative distance of the two peaks as

$$\Delta \rho (p_1, p_2) := \left| \frac{p_1 - p_2}{\rho(p_1 - p_2)} \right|$$

where $\rho \in \{\text{abs, rel}\}$, standing for ‘absolute’ and ‘relative’. It is easy to prove that both ‘coordinates’ of these distances are metrics.

We introduce at this point the term \textit{virtual tonal peak} as well, being the mean of the set $(p_1, p_2)$ of simultaneous tonal peaks according to the formula

$$\Pi = \left( \frac{1}{\tau} \sum_i (p_i - \Pi)^2 \right)^{\frac{1}{2}}$$

(5)

To represent the original tonal peaks that were merged into the virtual tonal peak $\Pi$ we define the variance of a virtual tonal peak as

$$\text{Var}(\Pi) = \left( \frac{1}{\tau} \sum_i (p_i - \Pi)^2 \right)^{\frac{1}{2}}$$

(6)

Note that a normal tonal peak can actually be considered as a virtual tonal peak with zero variance, hence we don’t really need to distinguish between real and virtual tonal peaks.

The most difficult part of an additive analysis method is usually the third step: that is, to build a partial tracker using the extracted tonal peaks. Our main assumption was that if a particular partial plays an important role in the sound, then it should contain at least a few loud tonal peaks. Therefore, instead of the usual way of starting the envelope matching algorithm at zero-time, our method would first search for the loudest available tonal peak and start an envelope following process based on that peak. This procedure consists of two major steps.

2.2. Finding Envelope Prototypes

Partial tracking is, by its nature, a paradox activity. On one hand, peaks that deviate ‘too much’ from a given envelope must not be taken into account. On the other hand, the process can’t be too rigorous – unexpected alterations of the partials must be captured as well. To fit with these ambiguous needs, our algorithm breaks the envelope following into a micro- and a macro-level part. First, we create short-time envelope segments – called envelope prototypes – which follow some simple and straightforward pattern; this is the micro-level analysis. The macro-level analysis would then merge these prototypes into real enve.

For the microscopic analysis, we introduce a set of user-defined parameters: the maximum allowed errors for level and pitch (\(\Delta\text{abs}\)) as well as for amplitude and frequency (\(\Delta\text{corr}\)) and the so-called time extension factor (\(T_{\text{scale}}\)), whose purpose will be explained in a moment. To create a new envelope prototype, we do the following sequence:

1. We take the loudest tonal peak from our peak pool. We also get every simultaneous tonal peak (taking into account also those which are already included in other envelope prototypes) whose distance from this loudest peak is either relative or absolute – is smaller than the user-defined errors. We merge then all these peaks into a single virtual peak as described in Equations (5) and (6). This will be the first member of our new envelope prototype.

2. We extrapolate the envelope prototype by one step forwards in time.\(^1\) If the envelope prototype consists of a single peak, we simply duplicate that peak. Otherwise, we use cubic spline extrapolation [7].

3. We create a new virtual tonal peak using all tonal peaks whose distance from the extrapolated peak is smaller than the defined errors. If we found at least one such tonal peak, we add the result to the envelope prototype.

4. We repeat steps 2–3 in backwards-time direction as well.

5. If we found at least one new peak during steps 2–4, we repeat the process from step 2. If we didn’t find any new peak, but the next time step forwards (backwards) is closer in time than the extended ending (starting) time of the envelope prototype – which is defined by the actual duration of the prototype multiplied by \(t_{\Delta}\) and then added to both ends of the envelope, – we skip this time-point and jump back to step 2.

6. When the new envelope prototype is ready, we remove every tonal peak contained by the newly created prototype from the peak pool and jump back to the first step. We repeat this process until the peak pool is not empty.

There are several reasons for using cubic spline extrapolation to guess the new peaks:

- The extrapolation favours ‘horizontal’ envelopes to start with. On the other hand, even with two points it gives a linear extrapolation, taking already into account the main deviations of the partial from the very beginning of the whole process.

- Splines are smooth enough to be good candidates for partials, yet free enough to conform with unexpected changes in the trajectories.

- The model has minimal assumptions on the shape of partials compared to linear prediction systems or the classical step-by-step envelope follower processes.

2.3. Merging Envelopes

After the micro-level analysis, we would usually end up having a huge number of quite short prototypes – envelopes that normally contain no more than a few virtual or real tonal peaks. The macro-level part of the analysis consists of the creation of the final envelopes by merging the prototypes into larger blocks. This process relies on the pseudo-cross-correlations of the envelope prototypes.

Let \(\delta_1\) and \(\delta_2\) be two envelope prototypes. Let \(\rho_{\delta_1}\) and \(\rho_{\delta_2}\) denote the starting and ending times of these envelope prototypes, respectively (here the starting and ending times mean the extended times, as explained in the previous section). To get the pseudo-cross-correlation of \(\delta_1\) and \(\delta_2\) we would first find the appropriate \(\rho_{\delta_1}\) and \(\rho_{\delta_2}\) values so that \(\rho_{\delta_1} = \max(\rho_{\delta_1}, \rho_{\delta_2})\) and \(\rho_{\delta_2} = \min(\rho_{\delta_1}, \rho_{\delta_2})\). As a next step, we compute the following ‘two-dimensional’ values:

$$\gamma(\tilde{\rho}_{\delta_1}, \tilde{\rho}_{\delta_2}) \leq \rho_{\delta_1} \leq \rho_{\delta_2} \Rightarrow \bar{\delta}_{\delta_1} := \delta_{\rho_{\delta_1}} - \delta_{\rho_{\delta_2}}$$

(7)

where \(\bar{\delta}_{\delta_1}\) is the spline-interpolated virtual tonal peak of the envelope prototype \(\delta_1\) at time \(t\) (if \(\delta_1\) contains a tonal peak at the given time, \(\delta_{\rho_{\delta_1}}\) would give us that tonal peak as a result, of course). Let \(\Lambda\) denote the total number of \(\delta\)-s defined by Equation (7) and let \(\Delta\) denote those \(\delta\)-s whose lower boundaries are non-negatives. Now we can define the pseudo-cross-correlation of \(\delta_1\) and \(\delta_2\) as

$$\gamma(\delta_1, \delta_2) := \gamma(\bar{\delta}_{\rho_{\delta_1}}, \bar{\delta}_{\rho_{\delta_2}}) = \sum_{i,j} \frac{\bar{\delta}_{\rho_{\delta_1}}(i) \bar{\delta}_{\rho_{\delta_2}}(j)}{\bar{\delta}_{\rho_{\delta_1}}(i) \bar{\delta}_{\rho_{\delta_2}}(j)}$$

(8)

where the upper indices \(I\) and \(P\) denote the level and pitch ‘coordinates’, respectively.

It is easy to prove that \(\gamma(\delta_1, \delta_2)\) is commutative and has the following properties (hence the name ‘pseudo-cross-correlation’):

- \(0 \leq \gamma(\delta_1, \delta_2) \leq 1\)
- \(\gamma(\delta_1, \delta_2) = 1 + \gamma(\rho_{\delta_1}, \rho_{\delta_2}) \leq \gamma(\delta_1, \delta_2) = \delta_{\rho_{\delta_1}}(i) \delta_{\rho_{\delta_2}}(j)\)
- \(\gamma(\delta_1, \delta_2) = 0\) if and only if there is no peak in \(\delta_1\) whose relative distance from \(\delta_2\) would be smaller than \(\Delta\text{corr}\) (and vice versa).

Based on this definition, we can get the final envelopes by computing the pseudo-cross-correlations of each pair of envelope prototypes and merging each pair where this value is above a certain user-defined limit (denoted \(E_{\text{corr}}\)) into a bigger envelope chunk. To merge the envelope prototypes \(\delta_1\) and \(\delta_2\), we simply take the union of all tonal peaks contained in \(\delta_1\) and \(\delta_2\). During this, we should not forget to create additional virtual tonal peaks if needed – that is, if simultaneous tonal peaks exist in the set. Note that the newly created envelope chunk is an envelope prototype as well so therefore we can go on with the envelope merging process recursively, as long as we find envelopes pairs with cross-correlations exceeding \(E_{\text{corr}}\).

It is important to note that we didn’t assume anything about the global shape of our partials during the macro-level analysis, the only thing we used was the correlation between envelope prototypes (which is a clear measure of the level of their similarity).

3. POTENTIALS AND DRAWBACKS

The final set of partials produced by the method described above is adequate for the additive synthesis described by Equation (1). Nevertheless, one might notice that the variances of the virtual peaks defined by Equation (6) were not used by the analysis at all. Keeping track of these variances is, however, a promising way to extend our method. These values show us the error levels of our approximation. One could, for instance, add randomness to the obtained frequency and amplitude envelopes, where the level of the randomness would be defined by the instantaneous variance of the partials\(^2\). Further analysis of the variance could also lead us to find more details and patterns in the partials, like high-frequency ring modulations etc.

Another important aspect of the model is that the main analysis block won’t rely on the time structure of the tonal peaks – they don’t need to be arranged into the usual ‘grid’ that we obtain after our STFT. Thus, even if we change drastically the process that extracts the tonal peaks (for instance, by replacing the STFT with a Wavelet-like transform), our envelope follower wouldn’t need to be altered.

We shall also briefly explain why we allowed during the micro-level analysis described in Section 2.2 that tonal peaks could appear in more than one envelope prototype simultaneously: in real-life situations adjacent partials usually cross each other from time to time. Envelope following methods that won’t let a tonal peak to belong to several partials can have great difficulties with the treatment of such scenarios [4]. On the other hand, the complexity of the whole process won’t increase too much by this ‘tolerance’. To illustrate the importance of this question, we show a successfully recognised envelope cross in Figure 1.

\(1\)This is the method that we actually implemented in KLANGPILOT.

\(2\)This is the method described by Equation (1).
Here, we analysed the following, simple sound:
\[
s(t) := \sin(\omega t) = \begin{cases} 
\sin(2\pi f_1 t), & 1 \leq t < 2s \\
\sin(2\pi f_2 t), & 2s \leq t < 3s \\
0, & \text{otherwise}
\end{cases}
\]
where \( \omega = 440 \ Hz \) and \( 0 \leq t < 4s \).

After examining our results, we can see that the algorithm detects the crossing, but is very confused at the moment where these crossings happen (we can observe this on both the loudness and frequency envelopes). We assume that this is mainly caused by the use of spline extrapolation. Future research needs to address this problem, probably by refining the extrapolation method.

As a comparison, we analysed the same sound using two different methods, linear prediction [3] and the bias corrected estimation technique [9] (see Figure 2).

![Figure 2](image.png)

**Figure 2.** Linear prediction (upper) and bias corrected estimation (lower) analysis of \( s(t) \).

As we may see, the linear prediction system failed to recognize the crossing of the envelopes, instead, it detected three different partials. On the other hand, the bias corrected estimation – after some fine-tuning of the parameters – was able to detect the crossing in a quite precise way.

### 4. CONCLUSION

Our first results with the new partial tracking process proposed in this article seem to be promising. However, there are still many unanswered questions. Particularly, a better understanding of the way how the parameters affect the final results would be essential. Also, more research should be carried out in order to determine the best practices for both the micro- and macro-level analyses. We might experiment with alternative extrapolation methods instead of splines; also the definition given in Equation (8) for the positional technique was used, what kind of themes, motifs and phrases were composed, how many times and on which degree the theme or a motif was imitated, what variations were introduced and similar. Analysing the compositional techniques used in the score is one of the most awkward and time consuming tasks, but to our knowledge there are no widely used methods or computer software packages for musicologists. In this paper we design a method to detect and store significant melody patterns found in the analysed score.

Tempferly [9] used prediction to analyse and predict the melody contour in German folk songs. He showed that probability is not useful only for analysing or fixing the score but also for composing new melodies. Weyde [11] proposed the segmentation of the motifs. He showed that the motifs candidates can be split into multiple segments based on the internal melodic and rhythmic similarities. His approach is important because he combined submelodies from different melodies in a similar way as the method presented in this paper does. Takasu et al. [8] proposed a complex approach to automatically extract the theme phrase from the score stored in MIDI format. They used the submelodic similarities using the Longest Common subsequence, a decision tree to classify the candidates based on their pitch and frequency and improved the results by using a finite state machine built on subsequent melodies of three. We used similar melody attributes in our research to build a simpler version of the melody evaluation function. Hou et al. [3] introduced an RP-tree based on the covariance matrix of the repeating substrings in the score. They argued that the suffix tree itself was too small and time consuming for their use case. In this paper we show that using the path compression technique and correctly choosing the characters representing the score, suffix trees can be efficiently used to store and manage string suffixes.

There are not many end user applications focused on music analysis from the musicologists point of view. JRing [5] allows importing MIDI files, annotating musical features and exporting results to a text file for further processing. Analysis [1] was developed during the harmony statistics research. It supports importing the MIDI file and determining the key, chord type and a tonal degree of the chord.

In this paper we propose an innovative approach for computer aided melodic analysis based on suffix tree. The approach was initially described in the bachelor’s thesis [4]. In Section 2 we present the suffix tree data structure commonly used for manipulating strings. Section 3 describes a procedure to construct a suffix tree for storing the note pairs and then transforming it into the compressed suffix tree for storing the melodies. A separate Section 4is dedicated to the evaluation of the melodies. We integrated the proposed methods into a computer program Harmony and we briefly present the solution and the empirical results in Section 5. We conclude in Section 6.

### 2. SUFFIX TREE

The suffix tree data structure [10] is a trie [2] in which we insert all suffixes of a given text. Although we use this data structure to find melodic patterns, we will present it first on an example of its original use for alphabetic strings.

Suffix tree supports efficient operations on a text \( w = c_1 c_2 ... c_n \) where \( n = \alpha \) and characters are from a finite alphabet \( \Sigma \). Suffix tree, as any other data structure, can be augmented to also contain the number of leaves in each subtree, the reference to the linked list of leaves in a subtree etc., and make the operations like searching easier to implement and run faster.

The straightforward way to build a suffix tree from a text is to insert into it all possible suffixes of \( w \) from
\((c_1 - c_3), (c_2 - c_4)\) and all the way to \((c_9)\).

Moreover, since the suffix tree is just a trie, we can merge several suffix trees into a single, combined suffix tree in a natural way. So, if we have a set of texts \(\{w_1, \ldots, w_n\}\), where \(w_i = n\), we insert each text \(w_i\) into all its suffixes into a combined suffix tree in the same way as the first text obtaining a final, combined suffix tree.

Finally, we do path compression on the tree by merging the nodes with only a single child (cf. Patricia, [6]). This results in a tree in which edges represent more than one character. The size of a path-compressed suffix tree becomes \(\sum n_i\) and therefore it is minimal. Consequently, this makes our further operations described in the following sections time and space optimal with regard to the size of the input text, i.e. the analysed composition. Further details of rigorous time and space complexity of our solution are here omitted. Figure 1 shows an example of a path-compressed suffix tree built from the text BANANA.

![Figure 1. Compressed suffix tree for the text BANANA.](image)

Each node represents a suffix from the root to that node. The number in the rectangle denotes the position of the suffix in the text.

3. SUFFIX TREE FOR MELODIC ANALYSIS

In this section we describe how to store all the melodies in the score into suffix tree. Initially we need a music score represented in a notation format (e.g. MusicXML, MIDI). The first step is to linearise the input data to obtain voices singing at most one note at a time. We will not describe this step in this paper (details in [4]) and assume the input data is already organized in such a way. Then we re-express the melody as a sequence of note pairs. The note pairs represent our alphabet \(\Sigma\) and their sequence represents text \(w\) from which we construct a suffix tree. Afterwards we do path compression on the note pairs suffix tree to obtain the melody suffix tree.

3.1. Note pairs generation

We have a list of voices \(v_1, \ldots, v_{\nu}\). Each voice consists of a sequence of notes \(\langle n_1, \ldots, n_{\nu} \rangle\), where \(n_i\) denotes the note’s pitch and \(r_i\) denotes the note’s length. We define the note’s length as the difference between the note’s start and the end time of its event. This way we treat the empty text \(\langle \rangle\) as the one character. The size of a path-compressed suffix tree becomes \(\sum n_i\) and therefore it is minimal. Consequently, this makes our further operations described in the following sections time and space optimal with regard to the size of the input text, i.e. the analysed composition. Further details of rigorous time and space complexity of our solution are here omitted. Figure 1 shows an example of a path-compressed suffix tree built from the text BANANA.

![Figure 2. Melody of initial two bars of the J. S. Bach’s Menuet No. 1 in G-major.](image)

For each voice we generate note pairs. The \(i\)th note pair is defined as \(\langle d_{\nu}(p_i, p_{i+1}), d_{\nu}(r_{i}, r_{i+1}) \rangle\), where \(i = 1, m - 1\). Functions \(d_{\nu}\) and \(d_{\nu}\) describe note’s pitch and rhythmic distance respectively. We define the following strategies for the function \(d_{\nu}\): a musical interval based distance (quality and quantity or quantity-only), pitch difference in semitones and relative note names.

The first strategy consists of the interval quality, quantity and direction. This strategy is used to determine which melody occurs in a specific gender (eg. major, minor). It requires the note accidentals to be correctly determined. The value of the first note pair in the example is \(7\).

The second strategy consists of an integer describing the number of semitones between the notes. This method works well if the note accidentals are not correctly determined. This method is common when importing music from the MIDI format because the format does not contain any information on enharmonic tones. The value of the first note pair in the example is \(7\).

The third strategy consists of the note names including accidentals and the direction. This is useful for finding occurrences of the melody in exactly the same pitch regardless of the octave, for example when analysing the main theme in sonata form where we wish to separate same melodies in the exposition (Tonic) and the development part (Dominant). In the example, the value of the first note pair is \(D-G\).

For the rhythmic distance function \(d_{\nu}\), the notes’ rhythm ratio or absolute rhythm are used. The first distance is defined as \(r_{\nu}(1)\). This method is robust because of a rhythmic invariance to the same melodies written for a factor of shorter (diminution) or longer notes (augmentation). The value of the first note pair in the example is \(2\). The absolute rhythm distance is defined as \(r_{\nu}(1)\) and is used when we need exact information on which melody is represented using which note lengths. The value of the first note pair in the example is \(1/4, 1/8\).

3.2. Suffix tree of music patterns

First we build a suffix tree containing all voices \(\nu\). The alphabet of characters \(\Sigma\) consists of the note pairs as transformed by \(d_{\nu}\) and \(d_{\nu}\). Note, that our one character extraction from the same piece of music, although this need not to be the case. Indeed, we could use our technique to analyse several pieces of the same or of different composers.

The second step is to prune the suffix tree based on maximum suffix length and minimum number of occurrences. The former pruning is implemented by stopping building the suffix tree at a node \(v\) of a specified depth, but still counting the number of patterns ending in a subtree rooted at \(v\). This makes the latter optimization, where we prune the nodes corresponding to patterns with less than the prescribed number of occurrences in the sequence feasible. We call such a suffix tree a \(v\)-suffix tree because it contains pairs of notes.

Finally, we do path compression the \(v\)-suffix tree as described in section 2 obtaining an \(v\)-suffix tree – because it contains melodic patterns or simply melodies. Figure 3 shows the \(v\)-suffix tree built for the example shown in Figure 2. We visualize it by showing the whole suffixes (melodies) in vertices from the root to the corresponding node.

![Figure 3. The \(v\)-suffix tree for the first two bars of J. S. Bach’s Menuet No. 1 in G-major as seen on Figure 2.](image)

The first component in the braces contains the interval quantity and direction, the second component contains the rhythm ratio. The number in rectangle denotes the index of the of the pattern in the score. No pruning is used.

5. EVALUATION

Harmonia (http://harmonia-music.sourceforge.net/) is a free cross-platform application for music analysis. It is written in Python, uses Canorus score editor for score manipulation and LilyPond for rendering the melodies. It allows the user to import scores in MIDI and MusicXML file formats and then perform various analysis operations.

We added support for the melody analysis described in sections 3 and 4. The most important views show the \(p\)- and the \(v\)-suffix trees. The former allows the user to observe the frequency of different intervals in the score. For instance, we analysed 48 fugues from J. S. Bach’s Well-Tempered-Clavier and observed the most common sequence of intervals. These were the major and the minor seconds for the works written in the major and in the minor keys, respectively.

The \(v\)-suffix tree shown in Figure 4 allows the user to look at the melodies which are related to submelodies including the information on their locations in the score, the frequency, the melody length and the diversity. We analysed the Bach’s fugues and observed the melody rank calculated by the melody evaluation function as described in section 4. Fugues have a strict form and the main theme is always present at the beginning of the
interesting patterns. This greatly helps the user as there was an average of 483 melodies per fugue. By empirically tuning all the parameters the following combination turned out to work best: pitch difference in semitones, notes’ rhythm ratio, maximum theme length: 23, required min. number of occurrences: 2, Breq = 4, Blow = 60, C1 = 128, C2 = 23, C3 = 91.

Application also offers other functionalities. It allows user to listen to the whole score, selected voices or only specific melodies. If the analyst is interested in a specific melodic or rhythmic pattern he can search for it using the LilyPond syntax. The p- and m-suffix tree views allow user to flatten the tree for example to sort the results by specific attribute like the melody rank. Tight integration with Canorus allows selecting a melody and showing all its occurrences on all degrees in the score. Harmonia also provides score statistics like the note count per voice or instrument and the number of nodes in both p- and m-suffix trees. It can also visualise score statistics, for instance the melody distribution according to their length and frequency.

We believe that the proposed technique can (and needs to) be extended even further. Perhaps the most obvious direction is to improve the evaluation heuristic by incorporating some aspects of aesthetics and, even more importantly, to let the tool recognize variations of a theme as the same theme.

7. REFERENCES


Figure 4. Screenshot of Harmonia showing the m-suffix tree for J. S. Bach’s Fugue no. 2. WTK 1. Upon selecting the melody, all its occurrences were also selected in Canorus visible on top.

6. CONCLUSION

Determining the “true” theme of a musical piece is a difficult and, in general, subjective process, so completely automated analysis of melodic patterns is impossible. The best we can hope for is a computer tool to assist the analyst in the process of sorting out the frequently occurring interesting patterns.

We presented a data structure that can be used to discover, store and explore interesting melodic patterns in music, and a heuristic function for ordering the patterns by their supposed interestiness. Although we empirically showed that proposed heuristics work well, the true contribution of our work is the working application for interactive analysis of patterns, implemented as a part of Harmonia suite.

1. ABSTRACT

This paper traces the development of a series of works for solo trumpet with laptop electronics, pedals and guitar amplifier. Through a reflection on the interactive and intersubjective account of the creative process it develops a cultural context for this activity based on the author’s experience as a trumpeter/composer/improviser living and working in Australia while also taking account of broad movements in contemporary music particularly in improvised forms.

Following a practice-based research model it investigates how composition and improvisation intersect in the author’s practice. It also describes and reflects upon the use of electronic processing, sampling, and layering in the author’s improvisational and compositional language as he outlines the broadening of his practice through the integration of this media. The study and practice of extended techniques for the trumpet and their application in this electroacoustic setting are also briefly documented here.

2. INTRODUCTION

“Allotrope: Any of two or more physical forms in which an element can exist - diamond and graphite are allotropes of carbon” (Macquarie Dictionary 1991, p. 46). “The solo project creates the possibility for opening, expanding, and focussing technique, for deve lopment as aspects of ensemble playing that can be constricting. It is also a setting wherein the comfort zones offered by ensemble playing are stripped away. It is a difficult, vulnerable, but rewarding place” (Peter Knight Journal entry 2/4/08).

I am a trumpet player and composer with a background in jazz. I began to expand my practice through the incorporation of electronic media around six years ago. The solo project, Allotrope, which I describe during the course of this presentation, involved the development and refinement of a language of sounds created using the trumpet and real time laptop processing along with a range of other pedals and a guitar amplifier.

This project led me to examine my practice in new ways, to learn new skills, and new approaches to performance. In it I bring together contemporary approaches to the trumpet and technology to develop an idiosyncratic and highly personal sound world focussed on texture and timbre. Minimal and abstract, it also incorporates melodic material. Rhythmic elements are often obscured – pulse or ‘beat’ gives way to breath and gesture.

3. FREEDOM THROUGH LIMITATION

Early in the development of this work I decided to limit myself to using only sounds generated by the trumpet as the raw material for digital processing. This removes an array of possibilities offered in the digital realm such as synthesised sounds and samples from other sources. This limitation was an important step for me as it gave the project focus and enabled me to move forward – in a sense it gave me freedom.

Stravinsky talks about the apparent contradiction of freedom created through self-imposed constraint in The Poetics of Music (1942). He describes “a kind of terror” at the “abyss of freedom” that confronts him in the moment of setting to work on a composition. He writes: “Let me have something finite, definite matter that can lend itself to my operation only insofar as it is commensurate with my possibilities” (1942, p. 64). Perhaps the range of possibilities one confronts with digital media is even more terrifying than that which Stravinsky describes because literally anything is possible without even the limitations of technique, as material from any source can be ‘flown in’ at the mere push of a button. This ‘democratisation’ of music-making, which has accompanied sampling and processing of sound using computers has in a sense made musicality even more of a premium – while, with a laptop and the appropriate software, anyone can put sounds together to create an ambience or a ‘soundscape’, fewer can use the same materials to delve deeper into the craft of composition to explore structure and other formal elements.

I hasten to add that, for the purposes of this discussion, my definition of ‘soundscape’ relates to the creation of ambient sound works rather than to the usage of the word pioneered by Murray Schafer (1994) and others in the 1970s, which relates more generally to the acoustic
environment and environmental sound. The definition of works such as ‘soundscapes’ as well as ‘composition’ are slippery so for this discussion I define music composition as the creation of a piece of music that requires a complete listening to be properly ‘read.’ In my view, this is distinct from ‘soundscapes,’ which I define as a piece of music or a sound work that can be listened to at any point in its duration for any length of time and be understood. In addition it is worth noting that the creation of soundscapes is an entirely valid form of composition. I would argue it is a different form of expression from that of music composition.

From the outset my aim has been to create compositions rather than soundscapes. And for me it doesn’t really matter whether you are writing a pop song, a symphony, creating musique concrete, or indeed a spontaneous composition (improvisation), what we do when we compose music is balance consistency and variation to create interest, and to create a sense of ‘unfolding in time.’ This is something one of my teachers, Mark Pollard, impressed upon me and it is one of the most valuable lessons I have had. It makes sense of the various strands of my diverse practice, connecting what I do when I play jazz to what I do when I create in more abstract settings. The most valuable lessons I have had. It makes sense of the various strands of my diverse practice, connecting what I do when I play jazz to what I do when I create in more abstract settings.

MODES OF SOLO PERFORMANCE

‘I am now thinking of the technology – problem solving, learning about the parameters of plug-ins, finding more efficient approaches to familiar tasks. And it seems like each time I discover something new about how the software can be used it recontextualises the creative work. In this way the boundaries of my project appear to keep shifting’ (Peter Knight Journal, internal entry 2/4/08).

I have worked for some time on refining digital environments within which I can improvise, and identifying plug-ins that produce processing effects fitting the aesthetic I want to produce. I have been using Pro-Tools for recording and editing and have some experience with Max MSP but the primary software I have employed for this project is Ableton Live. This program has some incredible features and some specific limitations of its own. I would argue it is a difficult, in an improvised setting, to use the computer in an intuitive, sensory, manner to create textural variation. It is particularly difficult in a ‘real time’ performance environment.

This is a problem musicians and technicians have been grappling with since people started making music on computers. There are some some sound processing solutions available for the Ableton Live interface that consist of sophisticated hand operated MIDI-controllers, but these work best when the musician is not also using their hands to hold a wind instrument. There are wind players like Jim Denley (saxophone) and Natasha Anderson (bass recorder) who have countered this by using custom key triggers and mid-enabled sensors to manipulate the processing environment, creating ‘meta-instruments.’ However, this is not the way I decided to proceed because I am also interested in the time-based (rhythmic) processing of sampled material that is possible in Ableton, and to work using these parameters the musician needs to be designing his improvisation on the screen.‘ The solution I have developed involves using a mid-foot pedal unit and a basic MIDI mid-controller.

To allow me to change the processing parameters using the expression foot pedals and ‘play’ the computer in tandem with the trumpet in a gestural manner that augments my trumpet palette then the path to take may have been clearer, but what I realised I was actually doing here was developing a new practice in the digital realm and hybridising that with my instrumental practice. It’s perhaps a subtle distinction but one which seems relevant to this discussion of this project.

6. PLUG-INS AND PROCESSING

Crucial to this development was the Ableton Live plug-ins. Ableton is designed for improvisation in its own right with some attempt to mimic the sound of a room. At times Frozen Build Up can sound almost like feedback, leaving notes reverberating in the software. I used this plug-in to produce long sounds to enhance the sense of the movement of the breath in space.

When working with breath sounds I also, at times, employed the Rezator plug-in, which alters the timbre of a sound and gives it a tone or pitch even if the original signal is unpitched. I used this plug-in with the pre-set Berlin often to create a drone often in conjunction with a reverb pre-set called Frozen Build Up. This reverb has an extremely long decay and a shimmering sound quite unlike that of conventional reverbs that attempt to mimic the sound of a room. At times Frozen Build Up can sound almost like feedback, leaving notes reverberating in the software.

Ableton is a platform on which you can derive an opportunity for further processing of the sound or the creation of dissonant or consonant harmony through the layering of other signals. The plug-ins mentioned above process the sound in a ‘linear’ manner, which means, in simple terms, the signal moves forward in time, but a quite different type of processing called granular synthesis, which Opie describes as “a method by which sounds are broken into tiny grains which are then redistributed and reorganised to form other sounds” (2009). Granular synthesis has a rich history in contemporary music since it was first pioneered by Grusin in the ‘80s and later used by Xenakis and other composers who created the effects by cutting up tapes. There are a range of plug-ins I used during this project that process

1 Behringer FCB 1010 which can be programmed to trigger any parameters in the software.
2 FCB controllers plug into USB port and can be programmed similarly to FCB 1010
3 MAX MSP is one of the most popular sound processing programs currently in use and is favoured by many electronic musicians.
4 A pre-set is a set of pre-programmed parameters that can be chosen when using a plug-in.
amplifier. The second clip-on bell microphone (a
by a RAT distortion pedal and then into a guitar
power5, which I route to a volume pedal followed
of the microphones is a dynamic microphone,
using two clip-on bell microphones, volume
laptop I expand my processing environment by
As well as processing my trumpet through the
HIPNO suite of plug-ins (now
the world of extended techniques in
acoustic extended trumpet techniques. These form
with the 'clean' trumpet. As I have worked more and more with
addition I can modulate the feedback by the
movement of my hand over the mouthpiece of the
trumpet. If I have worked more and more with
I make extensive use of un-pitched
breath sounds, which form some of the
fundamental timbral and textural effects of this
work along with other sounds that are traditionally considered 'impurities' to be avoided in the
production of trumpet 'tone.' I relish these
'undesirable' sounds: the hissing of air, the
crunching of valves and the squeak of a
'misplaced' note. An essay by Henry Cowell,
etitled 'The Joys of Noise' is relevant in which he
states, "Since the 'disease' of noise permeates
all music, the only hopeful course is to consider
the noise-germ, like the bacteria of cheese, is a
good microbe, which may provide hidden delights
to the listener, instead of producing musical

Drunken Sailor can be a hard instrument to
to control or predict but I enjoy the randomness and
have stumbled across sounds I would never
otherwise have found" (Peter Knight Journal entry 06/05/2009).

One of the most satisfying aspects of this set-
up is that it relies on the physical gesture. The
note produced by the feedback of the amplifier
changes depends on the proximity and position of the
tromp and on which valves are depressed. In
addition I can modulate the feedback by the
movement of my hand over the mouthpiece of the
trumpet. As I have worked more and more with
this I have developed a physical language that
incorporates the movements of my feet across the
pedals and my trumpet in space.

8. PROCESSING OF EXTENDED TRUMPET TECHNIQUES

In addition to the time spent developing my digital
processing environment, during the course of this
project I also focussed on expanding my range of
acoustic extended trumpet techniques. These form
much of the raw material that I use for the feedback.
With the same microphone set-up and
processing parameters.

One of the techniques I employ to
explore this 'noise-germ' in involves using my two
clip-on bell microphones to carefull effect. I take
out the second valve slide slide of the trumpet
( refer Fig. 2) and clip one of the microphones to
the side of the trumpet to pick up the breath sound
escaping from the valve slide then I clip the
other onto the bell as another signal.

The Hipno suite of plug-ins (now
discontinued), made by Cycling 74 the creators of
MAX MSP, also employs granular synthesis as a
primary mode of processing. I focussed on one of
these plug-ins called Drunken Sailor throughout
the course of this project. Drunken Sailor is
particularly interesting and effective because of the
way its 'buffer' works. It samples a small amount of material and then processes it using
numerous parameters, which change depending on
where the puck is located in the coloured circle
(see Figure 1).

The position of the puck can be
controlled using the expression pedals on my foot
board enabling a very intuitive approach to the
processing parameters.

The second clip-on bell microphone (a
condenser microphone) is routed straight into my
laptop via my MOTU soundcard and is in-my laptop so I can process the
amplifier signal.

This set-up enables me to generate and
control feedback from the amplifier through the
use of the volume pedal. I can set the feedback off
using the clicking of valves or with breath blown
through the trumpet and also then also process the
sound of the feedback. I used these sounds in
juxtaposition with the sounds of the 'clean'
trumpet to create a dialogue between the two.

One of the most satisfying aspects of this set-
up is that it relies on the physical gesture. The
note produced by the feedback of the amplifier
changes depends on the proximity and position of the
tromp and on which valves are depressed. In
addition I can modulate the feedback by the
movement of my hand over the mouthpiece of the
trumpet. If I have worked more and more with
this I have developed a physical language that
incorporates the movements of my feet across the
pedals and my trumpet in space.

9. REFLECTIONS

With Allotrope, as in my music practice in
general, it is important for me to somehow create
a balance between the conceptual and the direct –
the sensual and the cerebral. The contrasting
approaches of trumpeters such as German based,
Axel Dorner (2004) and Norwegian, Arve
Henriksen, who have both exerted influence on
my practice and in particular in the context of this
solo work, are relevant.

Dorner uses a trumpet modified with
integrated contact microphones and processes the
sounds created through various electronic means.
He challenges our notions of what trumpet
the sound via granular synthesis including
Transverb and Scrubby, which are both made by a
company called Destroy FX.

"Drunken Sailor can be a hard instrument to
control or predict but I enjoy the randomness and
have stumbled across sounds I would never
otherwise have found" (Peter Knight Journal entry 06/05/2009).

With the same microphone set-up and
processing environment I also create popping and
thudding sounds with the valves. This is a
technique I discovered for myself as a boy pulling
my trumpet apart. When the first or third valve
slide is pulled, pressure builds up. It can be
released by popping the slide all the way off or by
depressing the valve corresponding to the slide
being pulled. When the valve is depressed, the
air pressure is released creating a popping sound.
This turns the trumpet into a kind of a percussion
instrument except that the sounds do not resemble
sounds that can be made using any instrument
other than a brass instrument. And this is important to me: it is important that the sound
world I create using these techniques remains,
however abstract, connected to the trumpet.

Similarly I aim to maintain this connection in the
processing techniques I apply. I try to focus on techniques that will enhance and
explore the properties of the trumpet rather than
render them unrecognisable. In truth though, I am
sure most listeners would not be able to discern
the sources of many of the sounds (either raw or
processed) I use in Allotrope if they listened
cosmatically, but nonetheless the sounds remain
recognisable and linked for me and this is crucial.
Again this is about focus and creative direction
achieved through, to some extent, limiting one’s
palette.
ITERATIVE SYNAESTHETIC COMPOSING WITH MULTIMEDIA SIGNALS

Angus Forbes
University of California, Santa Barbara
Media Arts and Technology Program
angus.forbes@mat.ucsb.edu

Kiyomitsu Odai
University of California, Santa Barbara
Department of Music
kiyomitsu.odai@gmail.com

This paper explores the use of orchestrated feedback as an inspirational theme in an interactive multimedia composition entitled Annular Genealogy. The composition is performed by two players, each of whom use a separate digital interface to create and interact with the parallel iterative processing of compositional data in both the aural and visual domains. In the aural domain, music is generated using a stochastic process that sequences tones mapped to a psycho-acoustically linear Bark scale. The timbre of these tones and the parameters determining their sequencing are determined from various inputs, including especially the 16-channel output of the previous pass fed back into the system via a set of microphones. In the visual domain, animated, real-time graphics are generated using custom software to create an iterative visual feedback loop. This software runs on the iPad tablet and uses a custom fluid dynamics system, a vector visualization technique, and custom image processing filters to generate complex, evolving visual structures. Additionally, information from each of the domains is transferred into the other domain in real-time over a wireless network: sonic data is used to control the image processing parameters and visual information influences the generative parameters of the audio component. Interactive control of the composition is available through multi-touch interaction via the iPad tablet (for the visuals) and via the use of SuperCollider as a live coding environment (for the audio).

1. INTRODUCTION

Annular Genealogy is an interactive multimedia composition for two performers using multi-channel speakers, a projector, and a tablet computer. The performance is organized around a generative music composition and its visual analog. Both the audio and visual components are explorations of feedback processes that encourage the performers to interactively shape aleatory elements and transmute them into appealing, transient structures. The composition engine works with a stochastic sequencer that uses Brownian motion as a guiding metaphor. Similarly, the visualization engine depicts colored fluid energy as a representation of dynamic, ephemeral structures. In addition to exploring these feedback processes independently of each other, each engine also directly influences the other via networked communication: both the visual and audio processes broadcast data via OSC messages which then influence various parameters of the composition and/or visualization. Finally, even the physical interactions are fed into the generative system as contact microphones are used to pick up the tapping and other ambient sounds made during the interaction. The ultimate goal of the performance is to bring various layers of feedback into a performative experience. These feedback layers are interconnected, but can be broadly categorized as physical feedback, internal or digital feedback, interconnected or networked feedback, and performative feedback. Specifically, these layers include: the generation of new musical motifs being created from the processing of the output sound; the generation of visual forms from the processing of the output graphics; the vector positions that govern the displacement of the visual forms used as inputs to control music parameters; and the sequencing parameters controlling the generation of the composition used as inputs to control image processing parameters.

In addition to having cybernetic properties of interacted feedback systems, we underscore the piece as being fundamentally synaesthetic. That is, the mixing of the mutual generative processes configures the aural with the visual and vice versa [4]. Through the continuous interlinking of the two engines (via the performers and via the data sent over the network) a single interconnected multimodal signal is created. The output of this signal is represented simultaneously in multiple domains. Figure 1 shows a high-level chart of the relationship between the performers, the audience, and the visualization and composition engines. The performers input information to software in parallel using live coding and the multi-touch capabilities of the iPad. The outputs of each of the software engines then are fed back into themselves and to each other in various ways.
2. INTERACTIVE PERFORMANCE

By supplying a multi-touch and live coding environment as an interface to and influencer of the generative processes we add another layer of feedback where the performer is able to sense and shape the multimedia output. That is, we conceive of the performers as participants in a compositional process rather than as on-the-fly creators of audio-visual output. The generative software serves to create some structures independently of the performers; it is the role of the performers to guide the generational processes toward more compositionally interesting output and away from output that is overly repetitive, monochromatic, garish, or otherwise less satisfactory. Likewise, the audio and visual engines, via the various feedback processes, continually push against the explicit control of the performers. Overall, the composition is defined by a network of nested feedback loops that link the performer and the algorithm to create an inherent aesthetic tension between the generative and the interactive, the performed and the composed, the random and the intended. Figure 2 shows a photo of a performance of the piece.

Our composition refers directly or indirectly to a number of previous installations. Compositonally, we were inspired by David Tudor’s Rainforest IV, an early sound-art installation featuring an entirely analog feedback system. Rainforest IV is “a collaborative environmental work, spatially mixing the live sounds of suspended sculptures and found objects, with their transformed reflections in an audio system” [12]. In particular, Annullar Genealogy extends the concept of using loudspeakers as “an instrument unto itself” (rather than a tool of replication) and of using feedback as a compositional source.

Another overt influence on Annullar Genealogy is Iannis Xenakis’s concept of “Stochastic Music.” As described below, our stochastic sequencer is an integral part of the generative composition. In particular, Xenakis draws a parallel between his compositional methodology and such natural phenomena as “the collision of hail or rain with hard surfaces, or the song of cicadas in a summer field” [15]. Similarly, we use a stochastic, time-based timeline that is elastizied by Brownian randomness to create foreign sounds that nonetheless have the feel of natural phenomena. Xenakis, in describing his landmark orchestral work, Metastasis, hypothesizes that by constructing acoustic spaces of constant expansion out of long passages of Webern glissandi “one can produce rules by drawing glissandi as straight lines” [15]. However Xenakis’s straightforward mapping of these glissandi sweeps to a chromatic scale is somewhat problematic as it imposes a non-linear relationship between source and output frequency. Our approach to this linearity problem, described below, involves replacing the chromatic scale with the Bark scale, which is at least psycho-acoustically linear, and thus more effectively captures the intended naturalistic feel.

Other recent multimedia installations have also featured generative compositions that made use of feedback mechanisms between the audio and visual components. For instance, Karen Curley’s Licht und Klang is an audio-visual installation that generates sounds via optical sensors that use the refractions of light through oil and water as inputs into sound generation software [3]. Various electro-acoustic ensembles have explored the use of networked feedback as a tool for improvised performance. Most famously, The Hub creates multimedia performances based on sets of rules that transform signals passed between performers and that are purely audio and visual domains [6]. A wide range of works have explicitly explored the notion of synaesthesia in installations. For instance, Jack Ox and David Britton’s 21st Century Virtual Reality Color Organ uses visual representations of sound waves as an element in creating an interactive landscape [10]. More recently, Daniela Veto’s Multisensory Interactive Installation explores the interactive sonification of Kandinsky paintings [14]. Other works invoke Michel Chion’s concept of synaesthesia to describe the “welding together” of auditory and visual phenomena [2]. For instance, Niall Moody’s audiovisual instrument, Ashikita, generates simultaneous multimodal output from single gestures that similarly creates sonorous output based on a synthetic fusion of a mixed audio and visual feedback loop.

3. COMPOSITIONAL DETAILS

Annullar Genealogy is made up of two distinct software engines, one governing the aural domain and the other the visual. The visual and aural components are related by the structural mechanisms of generating and processing feedback and by the ways in which the engines focus on generating organic structures that continually devolve and transform into new structures. Both the aural and the visual engines represent the movement of energy through a system. These software engines are completely decoupled, but influence each other via the output of different multimedia data transmitted via OSC. For instance, in this way, each channel is part of a component of each other’s feedback loop. In this section, we describe the individual iterative feedback processing for the composition and the visual engines, and also indicate where output is sent to and received by the engines.

3.1. Composition Engine

The generative composition is largely generated through the receiving and processing of feedback from various sources: from the performer, from parameters received from the visualization engine, from the audience interaction, and via the piping of the composition data back into the composition engine itself. The external inputs are directly provided by the output of the visualization engine (i.e., via the interaction via live coding of a SuperCollider script, and contact microphone inputs that capture ambient sound. Moreover, the composition process is based on the continual recycling of audio data that is iteratively fed back into the microphones. Figure 3 shows a more detailed diagram outlining the main components of the composition engine.

![Figure 3. Overview of the interconnected components that comprise the composition engine. The darker line indicates the main feedback loop where the output audio signals are recaptured by microphones to be used in the generation of sound.](image-url)

The composition engine is written entirely in SuperCollider 3, and consists of various interrelated components, described below: the interactive timbre generator, the stochastic sequencer unit, and the Bark scaling unit. The interactive timbre generator controls the overall quality of sounds by convolving the output of a compressor unit generator with a sine tone oscillator. In addition, a parametric equalizer and a ring modulator further control the signals before and after the convolution. The stochastic sequencer unit controls the timing and frequencies following a series of compositional heuristics. These frequency values are then piped into the Bark scaling unit, which defines the mapping of the frequency values according to the perceptual linearity of human ears (described).

3.1.1. Interactive Timbre

The main input signal is captured by a number of contact microphones, passing through a 6-band parametric equalizer and a Bark equalization, to order to make the output sounds controllable. The feedback loop further functions as a distortion box by convolving sine tones with the output signal from the compressor. At the final stage of the system, the signal is ring-modulated with the low-frequency oscillator (LFO). The process is then iteratively repeated in real-time. The output through the speakers is again picked up by the contact microphones and becomes the main input of the feedback loop. Interactive control of this feedback loop is available through the live coding environment and, additionally, via the performer’s making sounds that will be captured by the microphones. In our original performance, for instance, contact microphones were attached to the iPAD controller to use the performer’s tapping of the visual performer as another input into the composition.

3.1.2. Stochastic Sequencer

The sequencer unit triggers the data for the following elements: the frequencies and durations for the envelopes of sine tones, the bandwidth and their bandwidth as Q values for the parametric equalization, LFO frequencies for the ring modulation, and parameters for the composition of the output sounds. All the values for those sequenced parameters are generated through stochastic processes that are based on four modes of increasing randomness. In mode 0, a sequence of values is created through a simple rising motion (which is not random). In mode 1, we use Brownian motion, where each following number is either incremented or decremented slightly from the current number. In mode 2, we use interpolated randomness, where a random number is averaged with the current number, and thus more closely related to the current number. Finally, in mode 3, we use a non-interpolated, completely random number that is not related to the current value. Each parameter is then manipulated slightly from the current mode. Both mode 0 and mode 1 are utilized to update the panning amongst the 16 speakers. We use mode 1 to update the amplitude, band-pass processing...
3.2. Visualization Engine

The dynamic visualization is also created through a series of simple processes, using an algorithm called Fluid Automata (initially created by one of the authors as the visualization engine for the original performance) [5]. Inputs into the visualization engine are provided by the compositional engine via OSC messages, from performer interaction, and via the iterative real-time processing of the output image. The engine handles the main processing layers: the interface layer, which translates multi-touch gestures into fluid vectors; the fluid dynamics layer, which translates fluid energy across a discretized grid of cells; and the image processing layer, which interprets the fluid energy as distortions of a texture map, and blends this distorted texture map with a noise field made up of randomly colored pixels.

The engine was coded in Objective-C using the OpenGL ES 2.0 graphics library and the GLSL shading language and runs entirely on the iPad tablet device. Figure 4 shows a more detailed diagram outlining the main components of the visualization engine.

3.2.1. Interactive Fluid Dynamics

In keeping with the goal of creating artificial, yet naturalistic elements, we used a fluid system as a primary metaphor for creating evocative, transient image structures; a visual analog to the stochastic elements created by the composition engine. A number of interactive art projects use fluid simulation as a component of the work. A method invented by Jos Stam to create stable fluid systems first made it possible to represent realistic looking fluids at real-time frame rates [11]. Many interactive artworks have made use of this technique. For instance, Memo Aten has created a series of demos based upon Stam’s method, showcasing the use of mobile devices for interaction with the fluid system [1]. Other similar fluid simulation methods use shader programs that are optimized for real-time interaction in video games [7]. Since the goal of the project was not to represent reality, but rather to generate new creative possibilities, we chose to create our own fluid engine. By so doing, we had a much greater control over the types of visual structures that were created. Moreover, we were able to ensure that our system was robust and also to create visual structures not based on simulation equations that aim to empirically describe the physical world.

The fluid system is discretized into a grid made up of a 15 by 15 cells. The fluid system is defined by directional energy vectors that move energy from each of the cells into their adjacent neighbor cells. New energy is added into the system via the iPad’s multi-touch interface as the performer drags fingers across the screen. The energy is divided into three streams of directional momentum: forward momentum, and left and right orthogonal momentum. Energy flows from the current cell into the neighboring cells at each iteration of the screen refresh rate, (approximately 60 times a second). A small amount of energy is removed at each step, and after some time (if no new energy is added), the fluid system will have no energy. A more detailed description of the system can be found in [5]. Despite the straightforwardness of this algorithm, particular ratios of forward to orthogonal momentum create pleasing patterns of vortices and waves. Moreover, other parameters controlling the rate of movement between cells create effects that look somewhat like crashing ice, or drifting sand, or are extremely turbulent, or imply some other unfamiliar, yet naturalistic effect. Figure 5 is a close-up photo of a performer using the iPad interface to manipulate the fluid energy.

3.2.2. Iterative Image Processing

The main image processing scheme is based on a feedback loop whereby a high-resolution background image is perpetually blended together with a distorted version of itself. The characteristics of the distortion are based directly on the current state of the fluid system. Through the feedback loop of blending the previous output with a colored noise field, the individual colored pixels in the noise field are “smeared” in the direction of the current fluid vectors. This system is similar to a vector visualization technique first described by Jake van Wijk, called Image Base Flow Visualization (IBVF). IBVF has been extended for use in a variety of scientific visualization applications, including animated and 3D flows [13].

Figure 5. Photo of multi-touch interaction during a performance of Annular Genealogy.

The image is further processed by a number of image processing kernels that control the brightness, contrast, saturation of the image, as well as the blending factor weighting the averaging of the distorted image with the original noise field of random colors. The parameters governing these simple kernels are updated in real-time by the output data from the composition engine (via OSC messages). An enormous amount of color variation is possible through the adjustment of these parameters. Because the performer has no control over these parameters during the performance, care was taken so that combinations of parameters did not lead to unfortunate cases, such as the screen turning completely black when brightness and saturation were set too low. Figure 6 shows a detail of the projected output image of the visualization engine.
find an aesthetic balance between the human and the computational. The iterative generative system can quickly fall into patterns that become either repetitive or overly chaotic. Finding the creative “cusps” teetering between these two extremes was the most rewarding aspect of the performance.

While some of the results of interconnecting multiple feedback layers are unpredictable, the performers nonetheless begin to have an intuition as to how their actions will update the overall composition. For example, while there is no direct mapping of how the visualization data will update the compositional structures, after some experience using the iPad interface, it becomes clear that certain gestures during certain kinds of passages generate a particular shaping of the composition. We also found it interesting to re-conceive the performers role as a “guide” of aesthetics, rather than as a creator. A possible direction for future versions of the piece would be to more explicitly highlight the effect that an interaction has as it is transmitted from one medium to the other. Although the focus of the piece has been on interconnecting different feedback loops, one obvious result of the piece has been the discovery of new ways to perceive the piece’s structure and could encourage more intuitive musical ideas.

5. REFERENCES

VAZIK: A PAINTING GRAPHIC SCORE INTERFACE FOR COMPOSITION AND CONTROL OF SOUND GENERATION

Aura Pon
Interactions Lab
Department of Music
University of Calgary
calgary. aapon@ucalgary.ca

Junko Ichino
Graduate School of Information Systems
University of Electro-communications
Tokyo, Japan
ichino@is.ucc.ac.jp

David Eagle
Department of Music
University of Calgary
calgary. eagle@ucalgary.ca

Nicolos’Alessandro
Institute for New Media Art Technology
University of Mons
Mons, Belgium
nicolas@dalessandro.be

Sheelagh Carpendale
Interactions Lab
Dept. of Computer Science
University of Calgary
Calgary, Canada
sheelagh@cpsc.ucalgary.ca

ABSTRACT

Vazik is an interface for creating and visualizing music through painting gestures on a large interactive surface. It can subsequently act as a graphical score embedded with control messages for facilitating its sonic realization via any networked sound generator or instrument, and serve as a visual reference of the music to possible performers and audience. We present an overview of our motivations, design and implementation of the current prototype of Vazik and discuss its various application in music education settings and computer music performance. We aspire for Vazik to offer the creator, performer and audience an alternative insight into music's composition through graphic score visualization, and open up new ways to create and realize musical ideas.

1. INTRODUCTION

Throughout history, music’s intelligibility has benefited from the tangibility offered by multimodal renderings perceivable by sight and other senses beyond hearing. Visual representations of music such as traditional music notation, graphical scores [8], or musical inspired art [7] give this ephemeral medium a more permanent form through which to study, preserve, and recreate it [2]. The temporal nature of music is sometimes a barrier to visualizing, analyzing, and approaching the composition of music even for those who are musically educated. We believe that if one could not only visually see or touch music, but also freeze it in time and hold its representation in stasis for more prolonged examination and contemplation, then one could gain greater understanding of its structure. The design of musical interfaces that leverage a person’s existing understanding of basic concepts about the physical world would help build a usable understanding of music’s structure and could encourage more intuitive music exploration and creation, in similar reasoning to that posed as a case for Reality-Based Interaction [6]. Mapping features of music to physical properties of objects that can also be experienced through a person’s non-auditory senses like sight and touch may create certain abstract aspects of music more concrete and therefore more intuitive. Likewise, relating musical features to certain kinaesthetic actions may further embody musical understanding within existing familiar motor skills.

Towards a goal of fostering music understanding and creativity through multisensory tangibility, we created the Vazik interface (Figure 1) which is inspired by these past classic visualizations. Vazik, is foremost, an interface designed to empower people to make electronic music using painting gestures and visual representations of music on a large vertical interactive surface [11]. By giving music a lasting form by way of a visualization that people can see in addition to hear that effectively transcends its temporal nature, music can become more accessible and tangible, and therefore more intelligible. Beyond this graphical score facility, Vazik extends its capabilities one step further to allow its score to control and trigger aspects of its sonic realization through network control messages, which enables versatile sound generation, whether through an automated sound engine or performer-controlled instruments. The latter scenario has been found to offer unique capabilities in expanding the live performance capabilities of these instruments, as will be discussed.

We designed Vazik as simple and playable enough for a child to use, yet also to have capabilities to afford meaningful, complex musical experiences for more experienced musicians. We hope that the Vazik composing interface could open up new creative possibilities for composers and artists that would be engaging for the audience as well. This paper outlines our implementation efforts and describes the current prototype. We also present several evaluation efforts and applications of Vazik, and outline our coming future efforts.
2. RELATED WORK

Although other interfaces exist that support composition of music graphically, such as the Making Music software [13] or Hyperscore [4], Vuzik approaches composition differently in terms of its use of freeform painting gestures and physicality, and its focus on the micro elements of music construction, such as timbre, layers of sound, and dynamics. Some aspects of Vuzik’s appearance may be mistaken with a sequencer, or with interfaces such as The Music Animation Machine [10]. However, Vuzik is fundamentally different from a sequencer by its freeform painting-style interaction, importance of stroke direction, and greater integration of visual and sound elements. And unlike The Music Animation Machine, Vuzik is interactive.

Perhaps the most classic example of a drawing based composition tool was Xenakis’s UPIC system from 1979 [1], which consisted of a digitizing tablet linked to a computer that allowed the user to draw waveforms and volume envelopes. The system also allowed for real time performance as well by moving the stylus across the tablet. Another UPIC inspired piece of software is LightLP [2] which is a graphical music sketching tool consisting of a built-in synthesizer, sequencer and sound mixer, though does not allow gestural sketching beyond what can be achieved with a mouse or artist tablet. Vuzik takes inspiration from the UPIC system while striving for increased interactivity in the relationship of visuals to sound, and in its open-platform nature that allows for use with a variety of sound generation methods through control messages.

A number of other music visualization tools exist that employ the use of control messages sent via network to trigger or control sound from within the music visualization display. IanniX [5] inspired by the UPIC system, is a graphical open-source sequencer which runs via Open Sound Control messages and curves to a real-time sound environment like Max/MSP or PD. Although the graphic music visualizations that trigger the control messages are visually engaging, their complexity may render them unintelligible for use as a score from which live performers can play instruments, and the visualizations are created with a typical mouse controller, not gesturally as Vuzik promotes. The music notation software Notation Pro [9] also allows OSC messages to be embedded in a notated score, for the purposes of triggering sound events in real-time environments like Max/MSP when the score is scrolled to those points, either through “score-following” responding to input from a live acoustic musician, or manually. Vuzik takes inspiration from these particular interfaces while remaining distinct in its simple graphic notation and gestural painting interaction style.

3. VUZIK INTERFACE

Named with reference to “viewable” music (and pronounced similarly to music), Vuzik allows a person to compose electronic music graphically by full-scale painting gestures through a mapping of sound to visuals that effectively allows people to “see” their music as they hear it, using a vertical interactive surface, paintbrush, and icon palette (Figure 1). Use of an interactive surface provides direct freehand painting input of the sound with a tangible paintbrush (as well as alternate tools or possibly fingers), which additionally nurtures the connection of physical gestures to the resultant sound and visuals. The brushstroke is the performance and creation gesture that is both seen (as an action and in creating a visual representation) and heard.

Vuzik’s two modes of operation, creation mode and playback mode, offer both immediate, spontaneous play with homophonc sound, and reflective, creative construction of more complex polyphonic music. Creation mode is the initial mode of operation where a person can paint in input and hear corresponding sound feedback in real time that is related to the visual features of what is painted. At any time, at the tempo of their choosing, a person can use playback mode to hear a single selected stroke, or they can play back the entire canvas and hear all the strokes sounding polyphonically as the composition scrolls and plays from left to right, with highlighting circles following each stroke as it is being played.

In this mode that Vuzik is most effective in live performance, Vuzik will stream Open Sound Control protocol messages corresponding to components of the music’s graphic score wirelessly over the network to listening sound generators, whether it be a sound synthesis engine on a computer, or devices or instruments being played by live performers. The scrolling motion of the score as it generates sound in real-time can serve both as a visualization for audience members, and as a score for visual instructions to any live performers participating in the realization of the piece.

3.3. Implementation

The graphical user interface (GUI) is currently implemented using Microsoft Windows Presentation Foundation (WPF) Visual Studio 2010 C# and .NET Framework 3.6, and utilizes the Open Sound Control protocol for messaging to the listening sound generators.

Through these interactive elements of combined tangibles, visual-music metaphors, and open-platform nature of its control messaging aspects, Vuzik attempts to offer people informative and engaging mechanisms for composing music and flexibly performing or realizing it with a variety of sound generation possibilities.

4. APPLICATIONS AND EVALUATIONS

Vuzik was originally envisioned as a tool for children to explore music and sound, while still offering the capability for more advanced music composition. We propose that Vuzik’s audio-visual integration and gestural input style invites child interaction and promotes understanding of some more abstract aspects of music.

Towards evaluating this aspect, we are currently undergoing a user study with 16 or more school children of grades 4-6 for investigating Vuzik’s ability to facilitate music composition for children. The evaluation incorporates a between-subjects comparison of Vuzik and Hyperscore with the goal of understanding the strengths and weaknesses of the former and how it compares or contrasts to similarly motivated interfaces. Over two sessions, subjects were to learn the interface, explore example compositions, complete a composition which has beginning and ending provided, receive brief training in composition on the interface, complete a composition by adding melody to a provided accompaniment, and finally compose their own music.

In addition to a formal user study motivated for music education applications, we also explored its use in more advanced music composition and performance. Through the generative and creative process of composing music using Vuzik and realizing it live using an ensemble of musicians to rehearse and perform with it in a live concert setting, we were able to gain valuable insights into the application, success of design choices, and best performance practices for this interface.

We utilized Vuzik to extend the capabilities and complexity of music performance possible for an ensemble of artificial voice ChoirMob mobile device-based instruments [3] in several essential ways. Seven
of the ten participants in the ensemble had some formal musical training, while three would be considered music novices. Its role in this ensemble was a medium for communicating a composer's composition, amounting in a score for reading and understanding the music, and a "conducting application" for coordinating ensemble participants and their instruments. All participants regardless of musical background were able to use the rudimentary stroke-based graphic notation of Vuzik to inform them of the rhythms and timings of the phrases they were to play, and it fostered a level of awareness of what other ensemble parts were doing and what the pitch contour of their phrase was. A 13-minute, 3 movement composition for 4 independent voices, entitled Inertwine, was composed for this ensemble using the Vuzik interface. Vuzik's control messaging served the integral function of shifting the central reference pitch and recentering the range of semitones available on each instrument's small playing surface based on the needs of the composed music, thus extending the instrument's range and making difficult pitch changes for the performer. It did this independently for each vocally inspired iteration of the instrument (soprano, alto, tenor, bass). This piece incorporated strictly notated material as well as controlled improvisation sections, each conveyed graphically through the score. Four members of the ChoirMob ensemble performed this piece in a live concert setting at the East Vancouver Cultural Centre on November 7, 2011, to an audience of over one hundred people, following personal practice time and several ensemble rehearsals. The Vuzik visual score was projected for audience viewing, while displayed for the performers for reading and ensemble coordination on a smaller display.

Although this performance was the most thoroughly documented, others followed using the same system and setup, including a repeat performance of the same piece by different performers, a performance of a 2 minute work with Vuzik controlling a mixed-phase synthesis sound engine, and a performance of a 3 minute work using one solo ChoirMob instrument accompanied by the mixed-phase sound engine controlled by Vuzik. Finally, this combination system of Vuzik and the ChoirMob instruments was successful in being chosen as a finalist in the 2012 Guthman New Musical Instrument Competition held at Georgia Institute of Technology, for which the quartet performed Inertwine again.

In these instances, Vuzik enabled the gestural painting creation of an original graphic score that, through its simple graphical notation and control messages broadcasting parameters of the composed music, enabled performers with networked digital instruments to realize the music to a high level. The experience was anecdotally reported to be satisfying for the composer, performers, and audience.

In augmentation of the current ongoing user evaluation with children, we will endeavour to further inform our design of an effective music education tool by consulting elementary school music teachers about the unique challenges and processes in their music classrooms and how Vuzik could support them. Such discussions could contribute towards the design of curriculum that incorporates Vuzik as an illustrative tool for music educators in support of their curriculums. In addition to further development and understanding of Vuzik as a music education tool, we intend to further validate Vuzik as a composition tool for musicians by continuing to create and perform musical works with it. Such exploration will aim at discovering what type of composing and musical style is possible and idiomatic for the interface, what varieties of sound generation methods could be used with it, and different approaches for the interface to be used in performance either for real-time composition, in collaboration with other instruments, or as a standalone system for presenting complete compositions using an automatic sound engine. We would like to continue to explore how the creation of a dual-modality artwork of sound and visuals, and the gestural interaction style, could be engaging for both audiences, performers, and composer in unique ways. In summary, our ongoing work will aim to determine the interface's unique capabilities, shortcomings, and strengths as applied to composition, performance, and music education.

6. REFERENCES

PRECISE PITCH CONTROL IN REAL TIME CORPUS-BASED CONCATENATIVE SYNTHESIS
Aaron Einbond
Center for Research in New Music (CeReNeM)
University of Huddersfield
A.M.Einbond@hud.ac.uk
Christopher Tranopi
Computer Music Center (CMC)
Columbia University, New York
cmt2150@columbia.edu

2. PREVIOUS AND RELATED WORK

This approach draws on existing techniques for corpus-based concatenative synthesis and feature modulation synthesis.

2.1. Corpus-Based Concatenative Synthesis

The recent technique of corpus-based concatenative sound synthesis [12] builds upon a database of pre-recorded or live-recorded sound by segmenting it into units, usually of the size of a note, grain, phoneme, or beat, and analysing them for a number of sound descriptors, which describe their sonic characteristics. These descriptors are typically pitch, loudness, brilliance, nasiness, roughness, spectral shape, etc., or meta-data, like instrument class, phoneme label, etc., that are attributed to the units. These sound units are then stored in a database (the corpus). For synthesis, units are selected from the database that are closest to a desired target pitch and loudness of a number of sound descriptors, which describe the sonic characteristics of a sound. It consists in finding the precise sound transformation and its parameters, that have to be applied to a given sound, in order to change its descriptor values to match a given target descriptors. The difficulty is here that a transformation usually modifies several descriptors at once, e.g. pitch shifting by resampling changes the pitch and the spectral centroid and other descriptors. Recent approaches [8, 9] therefore try to find transformation algorithms that only change one descriptor at a time.

A data-driven search-based approach to FMS using CBCS has been introduced in [13]. A related approach to “descriptor-driven transformation” in an audio mosaic-context is introduced in [1], but without the real-time capability that is one of the key goals of CBCS.

The present pilot study does not implement a full FMS approach, but limits itself to parameters of pitch and intensity awaiting a more general FMS implementation in the future, or its integration with a hybrid concatenative synthesis approach, i.e. concatenation of segments in a parametric sound representation such as additive or source–filter models, as employed by SYNFUL [7].

2.2. Feature Modulation Synthesis

The generalized technique of Feature Modulation Synthesis (FMS) [5, 6, 8] can be applied to change timbral features beyond pitch and loudness, the features traditionally treated by a sampler. FMS in general is an analysis/synthesis approach that can be regarded as a meta-synthesis technique, borrowing from various other synthesis techniques in order to modulate a certain feature of a sound. It consists in finding the precise sound transformation, and its parameters, that have to be applied to a given sound, in order to change its descriptor values to match a given target descriptors. The difficulty is here that a transformation usually modifies several descriptors at once, e.g. pitch shifting by resampling changes the pitch and the spectral centroid and other descriptors. Recent approaches [8, 9] therefore try to find transformation algorithms that only change one descriptor at a time.

A data-driven search-based approach to FMS using CBCS has been introduced in [13]. A related approach to “descriptor-driven transformation” in an audio mosaic-context is introduced in [1].

3. ACCURATE TARGET PITCH AND LOUDNESS TRANSFORMATION

One advantage afforded by corpus-based transcription is that the larger the sample corpus, the more likely it is to contain a unit with a descriptor value within a given threshold of a target value. In order to modulate this unit to precisely match the target value, a relatively small alteration in the original unit can be made. Therefore other descriptors, for which modulation is not desired, remain relatively undisturbed.

Starting with the pitch for a given corpus unit $f_r$ in Hz or $m_r$ in MIDI note number that has already been estimated during the descriptor analysis phase, and given a desired target pitch of $f_t$ or note number $m_t$, we can determine the necessary transposition in semi-tones $t$, or directly the needed resampling factor $r$ as

$$m = m_r \frac{12}{\log_2 f_t / f_r}$$

$$f = f_r (2^{(m-m_r)/12})$$

Analogously, for a corpus unit of mean loudness level $L_r$ in dB, and given a desired target level of $L_t$, we can calculate the necessary gain factor $g$ in dB and the resulting amplitude multiplication factor $a$ as

$$g = L_t - L_r$$

$$a = e^{g/20}$$

Note that with loudness, we sloppily denote the mean logarithmic energy of a corpus unit, not the psychoacoustic percept of sound pressure. Neither do we take the sonie into account, for the moment, i.e. the time-dependent perceptual integration of loudness of the unit.

4. IMPLEMENTATION

A pitch- and loudness-modulation module has been implemented combining the latest full releases of CATA-RT 1.2.2, FTM 2.5.0.BETA.22, and bach v0.6.7 alpha for MAX/MSP. The CATA-RT software system for MAX/MSP realises corpus-based concatenative synthesis in real-time. It is a modular system based on the freely available FTM&MCP extensions [10], providing optimised data structures and operations in a real-time object system. CATA-RT is released as free open source software at http://imtr.ircam.fr.

4.1. Bach

Using the bach library developed by Andrea Agostini and Daniele Ghisi for MAX/MSP a “bach” interface has been implemented to combine with existing CATA-RT modules. The first major advantage of bach is its visual interface, capable of representing several pre-processed sound descriptors of musical instruments. A second feature is the sequencer playback of the bach.sequ (for metered music) or bach.roll object which can store the entire score or pitch content of a piece in musical notation. All of these features can be harnessed and interfaced with CATA-RT.

4.2. Corpus-Based Transposition

Our current model of targeted transposition works best with a corpus whose grains are clearly segmented into units of definable and constant pitch, for example by using segmentation based on change of pitch on a harmonic sound, or by loading banks of samples. One or more target pitches are defined before playback. As grains are selected from the corpus by proximity to target descriptors, which may include pitch itself and/or other descriptors,
their note number content is examined, and a transposition value equivalent to the difference between the estimated note number and the target pitch is sent to CATART before playback as defined in equation 1. If more than one target pitch is defined, for example a harmonic field of possible pitches, the pitch of each sample can be either drawn at random (with or without replacement) or chosen based on the shortest distance to the original pitch of the unit. Using an interface combining modules from CATART and the bank library, a dense field of microtonal pitches can be easily edited by the user (see Figure 1).

4.3. Loudness Modulation

The current version of CATART includes a loudness descriptor in units of decibels, therefore a simple subtraction is sufficient to adjust the gain on playback to a target value, as given in equation 5.

4.3.1. Sound sets

A finer control of loudness can be constrained by other descriptor values, for example SoundSet, a user-specific index associated with each unit. This is particularly useful when using different directories of samples, for example, associated with different instruments or playing techniques. For this purpose a SoundSet-control abstraction was created, which allows various subsets of the corpus grouped by SoundSet (indexed by directory by default) to be enabled and disabled in real time. Integrated into this module is a prototype mixer for loudness modulation, allowing a specific decibel level to be sent according to SoundSet classification before playback (see Figure 2).

4.4. Corpus-Based Transcription

In addition to its capabilities for real-time synthesis, CATART has been used effectively for real- and deferred-time audio mosaicing and computer-assisted composition [4]. In both cases, a live or recorded audio input target is analyzed and compared to a preloaded corpus according to descriptors chosen and weighted by the user. This process may be termed “Corpus-Based Transcription” and the goal is to create a mosaic of samples form the corpus that best approximates one or more audio features of the target.

Taking a feature modulation approach, corpus units can be altered to match better the descriptor values of the target. In the simplest case, the feature used to match target to corpus is the same one modulated. For example when the catart-analyzer module is used to retrieve NoteNumber estimates from the audio input and these values are treated as targets for pitch modulation, the resulting transpositions are relatively small resulting in a re-synthesis relatively faithful to the original corpus samples. However as other descriptors are added, values for transposition become higher, resulting in more significant changes in sample playback speed.

5. RESULTS AND DISCUSSION

The musical results of this approach can already be heard in new compositions by the authors, and in sound examples accessible online. While these examples answer the musical motivations presented in the introduction, they point the way for several future directions.

5.1. Ensemble Musiques Interactives

Targeted transposition with CATART made its public performance debut in a new piece Five Out of Six for the Ensemble Musiques Interactives by Christopher Trapani at the Festival of Interactive Music at Columbia University in March 2012. A large bank of instrumental samples is distributed over two CATART modules, each carrying up to 27 SoundSets per corpus. An interface permitting a high degree of control over all playback parameters (gain length, envelope, gain) creates a constantly evolving web of textures in real time. A second component of the work involves live video, in collaboration with the Madrid-based duo Things Happen. Using MIDI controllers to manipulate up to three layers of live images, musical and visual data are freely exchanged and interact. For instance, the degree of luminosity of an image corresponds to a given descriptor continuum of a selected grain, or the movement of a projected image across a screen is broken down into x- and y-coordinates that correspond to two descriptors on the axes of the catart-led, so that the position of the image triggers grain selection according to a predetermined descriptor space.

5.2. Voice and Electronics

In a new work Without Words by Aaron Einbond for voice, ensemble, and live electronics commissioned by the Fromm Music Foundation for Ensemble Dal Niente and premiered in June 2012, the approach of Corpus-Based Transcription is used to create a mosaic of vocal samples based on targets from live input and pre-recorded field recordings. Due to listeners’ perceptual sensitivity to playback of recorded voice, only small alterations in the corpus of vocal samples could be tolerated. Therefore NoteNumber alone was used as a probe-feature of the target to search the corpus for matching units, and the chosen unit descriptor values differ from those of the corpus by a maximum of two semitones, as summarized below. In this case corpus units were modulated with relatively small changes in the unit playback speed. Nevertheless, the added nuance and variability in unit playback produces a noticeably more rich and dynamic synthesized texture.

5.3. Statistical Evaluation

These two recent compositions present case studies with which to quantify and evaluate the effectiveness of the Corpus-based transposition approach. For Five Out of Six two CATART modules are used, each with its own preloaded corpus containing respectively 1897 units in 682 sound files of 76.2 min. using 769.2 MB and 1518 units in 636 sound files of 80.5 min. using 812.8 MB. The former is divided in to 27 SoundSets ranging in size from 6 to 338 units and corresponding to a collection of standard orchestral solo instruments, one instrument per set, including those shown in Figure 2. The SoundSet-control abstraction provides a fast and flexible way to navigate the instrumental timbres of such a large corpus.

Without Words also employs two corpora containing, respectively, vocal samples and instrumental samples. The former contains 3457 units in 293 sound files of 42.0 min. using 424.1 MB and is divided into 10 SoundSets sorted by vocal performance technique. The latter contains 8443 units in 1343 sound files of 180.2 min. using 1819.0 MB, divided into 15 SoundSets corresponding to the live instruments of the ensemble as recorded by the composer to generate the source-material for the score.

For a typical Corpus-based transcription task in which a target field recording is analyzed and the single descriptor MIDI NoteNumber is used to probe the corpus of vocal samples, the median transposition (absolute value) required during 3039 unit selections is 0.02 semitones and the maximum is 1.81 semitones. This is consistent with the assumption that transposition values for a large corpus are small enough to preserve sound quality and minimally affect other descriptors. However if a second descriptor Periodicity is introduced into the selection along with NoteNumber, the median transposition rises to 0.45 semitones and the maximum to 6.49. Adding more than two descriptors to the selection raises these values further.
data, and implementation of a more comprehensive FMS framework.

6.1. Pitch Estimate
The averaged gbr.yin-pitch detection employed by CATA-
RT to calculate unit NetTntumber descriptors is prob-
lematic especially for percussive sounds with a noisy at-
tack transient, for example pizzicato strings or vibraphone
played with knitting needles. The detection could be
improved upon, for example, by removing attacks or other
noisy frames before averaging the pitch. Alternatively,
one could calculate the median pitch on the whole seg-
ment, which should be undisturbed by the attack. That is
possible to implement in CATA-R-1.5’s modular descrip-
tor analysis architecture [14]. However the use of pitch
meta-data from the filename will always be another effec-
tive method for difficult-to-detect pitches. Not merely an
ad hoc solution, this alternative is necessary to accommo-
date users’ subjective judgements of the pitch of instru-
mental samples of noisy or extended playing techniques
that may leave only a faintly-perceptible pitch.

6.2. Meta-Data
Beyond pitch meta-data, other meta-data could be useful
for generalized corpus-based feature modulation synthe-
sis to include features not easily calculated on import. For
example, “spatial location” descriptors could be defined
in Cartesian or spherical coordinates and associated with
each soundfile. These could then be manipulated and in-
terpolated like other existing descriptors, with potential
uses for spatialized CBCS as described in [3].

6.3. Feature Modulation Synthesis
Finally existing literature on feature modulation synthesis
will be adapted for a more comprehensive corpus-based
feature modulation synthesis framework. The expanded
and expandable descriptor list in CATA-R-1.5 will be ad-
vantageous in developing a list of modulatable timbral
features, for instance spectral centroid, spectral flatness,
and further features that exist in current or upcoming ver-
cions of CATA-R may be processed based on a compari-
son of pre-calculated descriptor values and target descrip-
tor values.

7. ACKNOWLEDGMENTS
The authors thank the anonymous reviewers for their helpful comments. The work presented here is partially
funded by the Agence Nationale de la Recherche within
the project Topophonie, ANR-09-CORD-022, see http://
topophonie.fr.

8. REFERENCES
ing sound mosaicing with descriptor-driven trans-
formation,” in Digital Audio Effects (DAFx), Graz,
Austria, 2010.
[2] A. de Cheveigné and H. Kawahara, “YIN, a Funda-
mental Frequency Estimator for Speech and Music,”
Journal of the Acoustical Society of America (JASA),
with corpus-based concatenative synthesis,” in Proc.
based transcription as an approach to the composi-
tional control of timbre,” in Proc. ICMC, Montréal,
Canada, 2009.
sis: mapping acoustic and perceptual features onto
synthesis parameters,” in Proc. ICMC, Copenhagen,
Denmark, 2006.
phrase modeling?,” IEEE Signal Processing Maga-
(FMS),” in Proc. ICMC, Copenhagen, 2007.
FMS: Taking it to the next level,” in Proc. ICMC,
Belfast, 2008.
[10] N. Schnell, R. Borghesi, D. Schwarz, F. Bevilacqua,
and R. Müller, “FTM—Complex Data Structures for
Max,” in Proc. ICMC, Barcelona, 2005.
early years,” Journal of New Music Research,
vol. 35, no. 1, pp. 3–22, Mar. 2006, special Issue
on Audio Mosaicing.
[12] ———, “Corpus-based concatenative synthesis,”
IEEE Signal Processing Magazine, vol. 24, no. 2,
pp. 92–104, Mar. 2007, special Section: Signal
Processing for Sound Synthesis.
sound texture sampling,” in Sound and Music Com-
puting (SMC), Barcelona, Spain, JulIet 2010, pp.
510–515.
[14] ———, “A modular sound descriptor analysis frame-
work for related-real-time applications,” in Proc.

VISIONALIZATION OF PERCEPTUAL QUALITIES IN TEXTURAL
SOUNDS
Thomas Grill, Arthur Flexer
Austrian Research Institute for Artificial Intelligence (OFAI)
Vienna, Austria
{thomas.grill,arthur.flexer}@ofai.at

ABSTRACT
We describe a visualization strategy that is capable of effi-
ciently representing relevant perceptual qualities of textu-
ral sounds. The general aim is to develop intuitive screen-
based interfaces representing large collections of sounds,
where sound retrieval shall be much facilitated by the ex-
ploration of cross-modal mechanisms of human percep-
tion. We propose the use of metaphoric sensory properties
that are shared between sounds and graphics, constructing
a meaningful mapping of auditory to visual dimensions.
For this purpose, we have implemented a visualization
using tiled maps, essentially combining low-dimensional
projection and iconic representation. To prove the suit-
ability we show detailed results of experiments having
been conducted in the form of an online survey. Potential
future use in music creation is illustrated by a prototype
sound browser application.

1. INTRODUCTION
For music-making in the digital age, techniques for effi-
cient navigation in the vast universe of digitally stored
sounds have become indispensable. These imply appro-
priate characterization, organization and visual represen-
tation of entire sound libraries and their individual ele-
ments. Widely used strategies of sound library organiza-
tion include semantic tagging or various techniques from
the field of Music Information Retrieval (MIR) to auto-
matically classify and cluster sounds according to certain
audio descriptors which characterize the signal content.
Moreover, there is a need for appropriate user interfaces
in order to browse through such libraries. Common in-
terest lies especially in graphical, screen-based interfaces
that are capable of representing both the attributes of in-
dividual sounds as well as the structure of an entire col-
lection of sounds. Such interfaces should allow users
to efficiently pinpoint a sound given some specifications
and also to learn about the context of a sound element,
e.g. which other sounds of the collection exhibit related
properties (see e.g. [20]). Sensory (as opposed to arbi-
trary, cf. [25]) properties that are aligned with human per-
ception are most expedient, since they enable access with-
out the necessity of learning.

In this paper, we outline and evaluate an implemen-
tation of a screen-based interface capable of representing
major perceptual qualities of sounds. We restrict our focus
to textural sounds, that is, sounds that appear as stationary
(in a statistical sense), as opposed to evolving over time.
This broad class of sounds of diverse natural or techni-
cal origin (cf. [23]) is interesting for electroacoustic com-
posers, sound designers or electronic music performers
because of its neutrality and malleability, functioning as
source material for many forms of structural processing.
The structure of the paper is as follows: In Section 2 we
genralize our research endeavor and describe our approach
and the experimental setup. This is followed by a detailed
evaluation of our experimental results and a prototype
application example in Section 3. Finally, Section
4 concludes with a summary of the findings and pos-
sible implications for the future.

2. METHOD AND CONTEXT
2.1. Perceptual qualities of textural sounds
For the following, we refer to recent research of our
group in [12]. We have elicited a number of so called
personal constructs that are relevant to human listen-
ers for the description and distinction of textural sounds.
More precisely, those constructs are group norms that are
shared by the range of persons – all trained listeners in
that case – who participated in the experiments. The
most significant constructs are listed in Table 1, sorted
by the degree of agreement among listeners. As can be
seen, each of the constructs is bipolar, spanning a con-
tinuous range from one extreme to the other. The con-
struct natural–artificial is somewhat special as it does
not refer to an objective, potentially measurable qual-
ity of sound, but rather to the source of its production.
Since in parallel research we are mainly interested in au-
tomatically computable quantities we have not consid-
ered this construct for the present paper. Furthermore,
the obvious quality of loudness has been explicitly ex-
cluded since its perception is much more related to the
sound reproduction than to an inherent sound property.
The listed qualities describe spectral/timbral (high-low,
tonal\textsuperscript{1}–noisy) and structural/temporal (ordered–chaotic,
smooth–coarse, homogeneous–heterogeneous) aspects of
sound. Apart from the perceptual qualities proper, we can

\textsuperscript{1} Tonal, as in tonal language is synonymous to pitched
build on quantitative ratings in those perceptual dimensions obtained in [12] for a broad variety of 100 textual sounds. Notably, those ratings are manually annotated values (normalized to [0,1] range for our purposes), as opposed to computed MIR-style audio descriptors. Figure 1 shows a correlation matrix of those perceptual constructs which reveals that the qualities are not fully independent. Substantial off-diagonal values for the constructs

<table>
<thead>
<tr>
<th>Construct</th>
<th>Agreement $\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>high-low</td>
<td>0.588</td>
</tr>
<tr>
<td>high-dull</td>
<td>0.586</td>
</tr>
<tr>
<td>ordered-chaotic</td>
<td>0.550</td>
</tr>
<tr>
<td>coherent-erratic</td>
<td>0.551</td>
</tr>
<tr>
<td>natural-artificial</td>
<td>0.590</td>
</tr>
<tr>
<td>analog-digital</td>
<td>0.527</td>
</tr>
<tr>
<td>soft-rapy</td>
<td>0.523</td>
</tr>
<tr>
<td>bright-dull</td>
<td>0.519</td>
</tr>
</tbody>
</table>

Figure 1. Pearson correlations between perceptual qualities. The smallest significant correlation value (at $\alpha = 0.05$, two-tailed) is $\pm 0.049$. Adapted from [12].

Table 1. Perceptual qualities (bipolar personal constructs) with their synonymous alternatives. Constructs are ordered by decreasing agreement (Krippendorf’s $\alpha$) – top ones have been rated more consistently by the subjects than those at the bottom. Adapted from [12].

2.2. Visualization of sound collections

As indicated above we are interested in visualizations that represent both the structure of an entire collection of sounds and the properties of its individual elements. The task of visualizing the structure of a multi-dimensional data set, such as a collection of sounds each characterized by five values, is a problem of dimensionality reduction. Various algorithms have been developed to achieve a mapping from a higher dimensionality to 2D as is needed for screen-based interfaces, e.g. Principal Component Analysis (PCA), Self-Organizing Feature Maps (SOM), etc. (see [7] for a survey) and, as of late, ‘Distribution Stochastic Neighbor Embedding’ (t-SNE)[24] which performs particularly well in preserving both the local and global structure, revealing the presence of clusters within the data. All of these techniques can perform a projection of set element positions to a two-dimensional map. In the work by Pampalk[20] it can be seen how this is applied for ‘maps of music’ where related properties are clustered at ‘islands’, with several of them spanning entire sound collection. Other examples of map-like visualizations for screen-based interfaces would be [4, 13, 22, 5].

In order to represent the individual elements we can use iconic visualizations of their properties, as shown with the ‘Sound Mosaics’ in [8]. The combination of a map-like 2D projection and iconic representation of individual elements leads us to titled maps. Sound properties of the elements are encoded into the graphical appearance of the individual tiles, and the arrangement of tiles represents the collection. Figure 2 shows first takes on such an approach, developed in cooperation with audiovisual artist Ulla Raanter. These tiles allow a more or less continuous illustration of properties on a 2D map. The rightmost example using a honeycomb-like pattern of hexagonal elements served as a starting-point for our implementation. It is intuitive that for the grid density (resp. element size) of the tiles there is a tradeoff between the number of individually representable elements (resolution of the map) and the precision of representation of an individual element’s qualities. Without having conducted a more rigorous evaluation, a tile size of 20 pixels has turned out to be a feasible minimum size. Although smaller values might be desirable for larger databases displayed on small maps, the reproduction of some graphical properties (e.g. element shape or stroke width) would degrade considerably.

2.3. Visualization of texture properties

The prevailing visual representations of sound are based on low-level physical parameters, as e.g. instantaneous amplitude or spectral coefficients in the omnipresent waveform or spectrogram displays [1]. However, such visualizations are highly abstract, lacking a direct relationship to perceptual attributes of sound. Only with considerable expertise in interpreting such representations is it possible to deduce basic qualities like volume, high-frequency content or pitchiness. The goal of this research is to immediately link high-level perceptual qualities (as listed in Section 2.1) to visual correspondents, thereby potentially enabling a much more intuitive access. Our approach is easily motivated by Köhler’s [16] classic experiment from 1929 depicted in Figure 3. When sub-
We looked at two main aspects of the survey results: First, the percentage of correct associations, that is, how often the icon corresponding to the reference sound was chosen. This hit rate is biased by the fact that two sounds can be very alike in respect to their perceptual qualities, and also the respective visual representations very similar. If a sound very similar to the reference sound is among the choices for a vote, chances are high that the visual representations are confused. Therefore the more appropriate measure is the RMS$^2$ distance of the five perceptual qualities (as in Table 2) characterizing two sounds. In the following, we will speak of the RMS error when we refer to the perceptual distance between the sound belonging to the chosen icon and the reference sound.

3.1. Expertise and listening conditions

The expertise level and the listening situation are expected to influence the accuracy of the voting and therefore also the results. It was found that e.g. for survey B, the RMS error for level 0 differs from level 1 with $t = 2.20$ for $t_{0.05,0.04,46} = 1.68^*$ (for 19 + 29 voters), and level 1 from level 2 with $t = 3.32^*$ for $t_{0.01,0.03,73} = 1.67$ (for 29 + 48 votes). The influence of the listening condition is less significant, depending on the survey version. For survey B, mean RMS errors differ rather insignificantly with $t = 1.46^*$ for $t_{0.05,0.04,41} = 1.66$, while for survey A, significant differences can be seen with $t = 5.66^*$ for $t_{0.01,0.03,41} = 1.68$. Overall, it can be stated that the expertise level and the listening condition have a significant impact on the voting results.

For the detailed analysis of results we focus on two subject groups: First, the general group of all subjects having contributed at least 20 votes, and second, the sub-group of above subjects with 'practice in electronic/experimental music' (expertise level 2) and availability of good listening conditions. This restricted group of expert listeners represents the actual target users of a textual interface.

3.2. Evaluation of survey version B

Table 3 shows an overview of the results for different subject groups, having contributed to survey B. We see that the results of expert listeners are generally better than those of the entirety of all subjects.

<table>
<thead>
<tr>
<th>Expertise level</th>
<th>Listening cond.</th>
<th>Voters</th>
<th>Mean RMS error</th>
<th>Mean hit rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>674</td>
<td>0.175</td>
<td>35.0%</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>674</td>
<td>0.175</td>
<td>35.0%</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1555</td>
<td>0.165</td>
<td>39.2%</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1155</td>
<td>0.165</td>
<td>39.2%</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>202</td>
<td>0.187</td>
<td>30.2%</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>202</td>
<td>0.187</td>
<td>30.2%</td>
</tr>
</tbody>
</table>

The baseline for the mean RMS error (for random choices) is 0.234, and for the mean hit rate 0.2. We find that for the entirety of all subjects the mean RMS error ($t = 0.152, \sigma = 0.154$ for $n = 5257$) is very significantly lower than the baseline, with $t = -24.4^*$ for $t_{0.001,0.000,3.29} = 3.29$. Naturally, the same goes for expert listeners: Again, the mean RMS error ($t = 0.133, \sigma = 0.145$ for $n = 2019$) is significantly lower than the baseline, with $t = -34.3^*$ for $t_{0.001,0.000,3.30}$.

Figure 5 plots the RMS errors for both the baseline of random choices (dashed curve) and the actual data of expert listeners (solid curve), ordered by ascending magnitude. At the positions on the x-axis where those error curves rise from the zero level, the vertical dashed lines mark the fraction of correct matches (with RMS error $t = 0$), both for random selection and for the survey data. All in all, we see that the area under the curves, representing the error distribution, is considerably smaller for the tests conducted more than 20 votes, which amounts to 3–5 minutes of interaction. This restriction reduces the total number of subjects substantially from 220 to 134, but the vote count by only about 8%, from 8130 to 7452.

Another aspect, even more fundamental, is the motivation of the contributing persons, whether they took the test seriously, and took time for it, or whether they were just curious or wanted to try out the technical implementation. It is straightforward to assume that a person was interested and willing to contribute meaningful data if she or he spent more than just a few moments with the survey. Hence, we have only considered data of persons having added input fields for meta-information.

2.4. Online survey

In order to evaluate our mapping from auditory constructs to visual parameters we conducted an online survey. Our implementation uses HTML5/Javascript canvas vector graphics, generated dynamically by any modern web browser from the coefficients of the five perceptual qualities. Figure 4 shows a screenshot of the survey, presenting an audio player element for a reference sound and five graphical renderings representing different sets of perceptual qualities. The reference sound is a random selection out of the collection of 100 texture sounds used in [12]. One of the graphical renderings is a representation of this reference sound, while the others are either generated by random perceptual qualities (each of the five dimensions uniformly distributed over full range [0, 1], survey version A), or by randomly chosen other sounds of the collection (survey version B). The two survey versions present quite different situations to a subject: In version B, the coefficients used to generate all five graphics come from the same distribution of sounds in the texture collection. Depending on the variety within the collection the distribution might or might not be representative for all possible texture sounds. Survey version A tries to account for that by fully exploiting the ranges for the perceptual coefficients. This may, however, result in coefficient combinations that would never appear in actual sounds, constituting a somewhat artificial situation. The results from both versions are qualitatively very similar, so, for the sake of compactness and clarity, in the following we restrict ourselves to the discussion of survey version B. We have chosen B rather than A because the former seems to represent a more ‘natural’ situation, also carrying interdependencies between the individual constructs.

In the basic configuration our survey design does not provide any explicit feedback for the subjects, whether their vote would match the reference sound or not. With this deliberate choice we try to inhibit possible learning of the systematics, thereby shifting focus to the potential of intuitiveness.

We decided to conduct an online survey rather than on-site interviews in order to make it more accessible and to cover a larger audience. However, this comes with considerable loss of control on the individual voting situation, e.g. sub-optimal listening conditions such as poor loudspeakers, or bad quality of the graphical rendering due to browser and screen variations. We tried to account for that lack of immediate control by suggesting modern standard-compliant web browser software (Firefox, Chrome or Safari) and asking the subjects to convey both their expertise level (‘non-musician’, ‘classical musical training (also jazz, pop, rock etc.)’ or ‘practice in electronic/experimental music’, in the following coded as level 0, 1 or 2, respectively) and the individual listening conditions (‘laptop loudspeakers or similar’ or ‘good loudspeakers/headphones’, coded as condition 0 or 1). As we can see in Section 3.1 both latter aspects have a significant impact on the results.

Table 3. Texture visualization survey B, results for different subject groups, all with min. 20 votes

<table>
<thead>
<tr>
<th>Expertise / Voters / Mean RMS error</th>
<th>Expertise / Voters / Mean hit rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 / 12 / 36 2019 0.133</td>
<td>46.4</td>
</tr>
<tr>
<td>1 / 202 0.187 30.2</td>
<td></td>
</tr>
<tr>
<td>0 / 202 0.187 30.2</td>
<td></td>
</tr>
<tr>
<td>2 / 15 / 674 0.175 35.0%</td>
<td></td>
</tr>
<tr>
<td>1 / 1415 0.165 39.2%</td>
<td></td>
</tr>
<tr>
<td>0 / 1555 0.165 39.2%</td>
<td></td>
</tr>
<tr>
<td>2 / 22 / 1155 0.148 42.2%</td>
<td></td>
</tr>
<tr>
<td>1 / 12 / 792 0.148 42.2%</td>
<td></td>
</tr>
<tr>
<td>0 / 4 / 202 0.187 30.2%</td>
<td></td>
</tr>
</tbody>
</table>

Footnote: mean RMS distances among all 100 texture sounds used in the survey.

Footnote: mean RMS distances among all 100 texture sounds used in the survey.
Figure 6 Pearson correlation between qualities of the selected sounds (represented by the visual icon) and the reference sounds, evaluated on the data of expert listeners having rated least 20 sounds. Survey B with 36 users, 2019 votes. The smallest significant correlation value (at $\alpha = 0.05$, two-tailed) is $r = 0.644$.

Construe high-low, which implies that this quality is most unambiguously interpreted by the visual representation as understood by the subjects. Tonal-noisy and smooth-course also show comparably high correlations, while ordered-chaotic and homogenous-heterogeneous lag behind a bit. For all those values the significance of correlation would be all one, while absence of correlation would result in zeros. Negative numbers correspond to inverse correlation. Along the diagonal, we find the highest correlation for the

3.3. Learning effects

It is intuitive that persons rating a substantial number of survey examples will implicitly try to discover the underlying systematics. A successful match of the visualization methodology to the one implied by the user should lead to a learning effect, resulting in decreasing error rates over time. In order to evaluate such learning effects, linear regression models for the measured RMS errors were calculated per subject, not taking into account expertise levels or listening situations. The average number of votes per individual was about 55 where a linear model seems still justified. For larger training sets a saturation effect resulting in non-linear characteristics – much like exponential decays – would be expected.

For survey B, 96 subjects having rated at least 20 sounds contributed 5257 votes in total. The mean of per-user regression slopes for RMS errors is $r = 1.033 \times 10^{-3}$ (RMS error change per vote) with a propagated standard deviation of $7.78 \times 10^{-3}$. The mean slope is smaller than 0 with $t = 12.6 > t_{2019.005, \bar{g}_H} = 3.18$. Hence, a significant learning effect can be observed even in the absence of explicit feedback.

An additional experiment was carried out, using a variation of online survey B with feedback after each vote. For each of the graphical icons grades were shown, ranging from 0 (very distant) to 10 (perfect hit), thus indicating the proximity to the ‘correct’ answer. 17 subjects who have not taken the original survey were contributing a total number of 960 votes. Per-user regression slopes of RMS errors are $-9.82 \times 10^{-4}$ with a standard deviation of $4.09 \times 10^{-4}$. Again, this is significantly smaller than 0, with $t = 9.89 > t_{2019.005, \bar{g}_H} = 3.6$. Counterintuitively, the learning effects of the survey with feedback are to be a tad smaller than without feedback. However, these differences are not significant with $r = 0.168 \leq t_{2019.005, \bar{g}_H} = 2.02$.

3.4. Decision time and perceived difficulty

Our goal is to achieve a sensory mapping between auditory and visual dimensions. This should open up the possibility for purely intuitive association from one domain to the other, without the necessity of reasoning what the systematics of such a mapping could be.

Although the guidelines of the online survey advised to 'proceed simply in an intuitive manner', the mean time per one vote was found to be almost 14 seconds which seems quite long for purely intuitive action. For each subject having rated at least 20 sounds with survey B we calculated the Pearson correlation between the data points of decision time per vote and the mean RMS errors, amounting to a value of 0.13. The significance test amounts to $t = 1.26 < t_{2019.040.05} = 1.99$, which means that the dimensions of decision time and RMS error are not significantly correlated. This finding implies that it does not help in general to ponder extensively what the best choice could be, but it is equally successful to quickly select the icon suggested by intuition.

Persons were also asked to rate the perceived difficulty of the auditory-visual association at each vote, by selecting one of the options: ‘straightforward/unambiguous’, ‘difficult/ambiguous’, and ‘impossible’, as seen in Figure 4. This input field has been introduced to obtain a measure which of the sounds are more difficult to associate than others, in order to improve the mapping later on. We calculated the Pearson correlation between the perceived difficulty and the mean RMS error for each of the 100 sounds used with survey B, amounting to 0.484, with a significance test yielding $t = 5.31 > t_{2019.005, \bar{g}_H} = 1.98$. Obviously, the contributors were able to correctly predict when an association would be close to the correct answer, on the other hand, when an association would not likely be successful.

3.5. Application

As a proof of concept, a prototype sound browser application was developed which combines the two aspects of mapping the structure of a sound collection and visualizing the qualities of its elements. Figure 7 shows a screenshot of the browser interface, consisting of a dynamically drawn map with white dots marking the positions of the individual sounds and a continuous tilting of the 2D space corresponding to the perceptual qualities in the map. The dimensional reduction was performed in a pre-processing step using t-SNE [24], yielding clusters of resembling perceptual qualities. These areas of similar qualities are perfectly reflected by the graphical representation – note the dark and light regions, or areas with more colorful or irregularly spaced elements. Inverse Distance Weighting is used for interpolation in between the projected positions of the individual sounds for a smoother appearance.

A k-d-tree [3] allows for efficient retrieval of sounds to play them interactively by mouse hovering. This browser interface also utilizes methods from parallel research of our group published in [11], which allow the computation of novel audio descriptors mimicking the perceptual qualities found in [12].

4. CONCLUSIONS

By relying on ‘synesthesia-like’ metaphorical correspondences between the auditory and visual modalities, we have developed mappings for the visual representation of relevant perceptual qualities of textual sounds. We conducted an online survey to evaluate our approach and found that subjects were able to successfully associate sounds with graphical representations, with RMS errors and rates of correct assignment better than random with a very high significance. We have also found that the indicated expertise of subjects as well as listening conditions for the test have a significant impact on the experimental results, with increasing expertise and better listening conditions, respectively, yielding better results. For ‘expert listeners’ the rate of correct assignment amounts to 46.4% in comparison to the baseline of 20% for random selection. We also found significant learning effects, independent of the presence or absence of explicit feedback, which led to the conclusion that there is a correct implicit understanding of the employed mapping. We attribute the presence of any learning effects at all to the necessary accommodation of subjects to the online survey, the range of sounds presented, and the characteristics of the visualization in particular. The test setup cannot be set up in a fully intuitive way without a significant correspondence to the perceptual difficulty and error rate.

5. ACKNOWLEDGMENTS

This work was supported by the Vienna Science, Research and Technology Fund (WWTF) project AudioMiner (MA09-024). Many thanks to the reviewers for their constructive comments!

\[\text{http://gpe02.org/kest/teavia/map.html},\text{ retrieved 2012-04-19}\]
REFERENCES


[7] M. Ferreira de Oliveira and H. Levkowitz, “From visual data exploration to visual data mining: A sur-
vey;” IEEE Transactions on Visualization and Com-


tober 2008.


[17] L. E. Marks, “On cross-modal similarity: The percep-
tual structure of pitch, loudness, and brightness;” Journal of Experimental Psychology: Human Percep-
tion and Performance, vol. 15, no. 3, pp. 586–
602, August 1989.


[20] E. Pampalk, A. Rauber, and D. Merkl, “Content-


[23] G. Strobl, G. Eckel, and D. Rocchesso, “Sound text-
ture modeling: A survey;” in Proceedings of the 2006 Sound and Music Computing (SMC) Interna-
tional Conference, Marseille, France, 2006, pp. 61–
65.


[25] C. Ware, Information Visualization: Perception for design. San Francisco, CA, USA: Morgan Kauf-

THE YIN YANG THEORY IN SOUND AND MUSIC: A FIRST EXPLORATION

Leonardo Gabrielli
Dept. Information Engineering, Università Politecnica delle Marche, Ancona, Italy
l.gabrielli@univpm.it

ABSTRACT

Today the Chinese theory of the Yin and Yang principles is well known in western countries, often in its philosophical aspects. However, in the far east it has been used for mil-
lenia to explain and solve pragmatic problems. Norwith-
standing its funny and holistic nature, the Yin Yang theory has already been successfully applied to the analysis of physical and biological systems. In this paper, the the-
ory is extended to the field of acoustic signals and their time-frequency representation, allowing for a simple yet functional way to analyze, discuss and formalize various aspects of sound, accessible to experts and non-experts. The framework can also have application on synthesis al-
gorithms, generative music and music therapy. Two high-
level features for automatic analysis are proposed based on MPEG-7 low-level descriptors and future scenarios to assess sound properties and their effect on human subjects are discussed.

1. INTRODUCTION

For millennia, the Yin Yang theory has been one of the pil-
lars of ancient Asian thoughts, used practically to describe all aspects of life and properties of objects and living be-
ings. The theory is the pillar of several eastern philos-
ophies and beliefs, such as Taoism and Zen. The Bagua (eight trigrams) and Five Element theory, on which Feng Shui and I King are based, stem from Yin and Yang. Western thinkers have been highly fascinated by such eastern concepts. A remarkable example in contemporary music is John Cage’s extensive use of the I King divina-
tion book for composition by chance [12]. It must be noted, however, that the Yin Yang theory must not be re-
garded “as a religious belief or a principle of lifestyle” [18] but a rather more practical knowledge. Not surpris-
ingly some western scientists have found connections be-
 tween the Chinese theory and the results of contemporary physics. An extensive review of the funding principle to ancient eastern theories and the evidences of their validity in modern physics is contained in F. Capra’s work “The Tao of Physics” [3]. Holistic thinking is also highly re-
garded in contemporary theory of dynamical systems or chaos theory, which exploits new theories drawn from bi-
ology and mathematics [4].

Although the seemingly dualist approach at the basis of this theory may seem to have an excessively reductivist, thus appar-
ently making the object of highly specialized techni-
cal knowledge a trivial matter, it is not. On the contrary, by incorporating many aspects of the object under analy-
sis, it achieves a holistic gaze, enabling to better deal with complex phenomena such as those related to human perception, emotion and health. Objections to holistic ap-
proaches are often made by scientists, especially in fields like medicine and medical care where the health of people is at stake. It must be stated, however, that modern medi-
cal science because of the drastically sectorial knowledge (which helped fighting e.g. communicable diseases in the past century), is now failing to provide effective therapies and prevention to noncommunicable diseases such as car-
diovascular diseases, cancers, chronic respiratory diseases and diabetes [2], which are found in both high-income and low-income countries and can be prevented with a change in lifestyle and dietary habits [31]. In fact, there have been documented cases of successful applications of holistic approaches to medicine. An excellent example is the mac-
robotic diet [24, 20, 17], a well-defined dietary regimen based on the Yin Yang principle and on the Five Elements theory, which is proving capable of reducing risks and greatly improve health conditions in individuals affected by several chronic diseases [25, 13, 11].

The Yin Yang theory has also been applied in comput-
utional sciences and logic, where it inspired the for-
malization of a bipolar fuzzy set [35], machine learning techniques [34, 33, 16] and of a pattern classification system for mental disorders [36]. We can probably say that the ancient Yin Yang theory can be applied to most, if not all, fields of human knowledge and practice, with good chances of achieving benefits by gathering a broader un-
derstanding.

It is our belief, thus, that the Yin Yang theory can be applied to music and sound by direct extension of the physical properties of Yin and Yang, as has been done in other fields of physics and biology. Henceforth, if this theory is properly applied and it is correlated to emotional states and human activities (the ancient theory does cate-
gorize these in terms of Yin and Yang), it can prove prof-
table for different scenarios, e.g.:

- for composers or performers to drive high-level au-
tomatic composition or synthesis techniques in a
more holistic way, thus delegating all the problems related to low-level control and the large dimension of the parameters space to the machine;
• for artists of different backgrounds to communicate in mixed media performances and installations without the need for a highly specific common language to be shared;
• for any kind of performance where there is a human-computer feedback and a high-level control variable needs to be controlled and balanced;
• for individuals with no formal music training to ass- sess and describe sound and music;
• for psychotherapist to understand the Yin or Yang state of mind (and hence his general mental condi- tion) of a person from his speech, observing general properties such as prosody, pitch, and so on. Gener- ally speaking, Yin is linked to relaxation and Yang to strain;
• for patients under music therapy to share with their therapist, and for the latter to deduct the effect of music and sound on their patients avoiding troubles in a more rigorous description;
• to possibly lead to a better understanding of the ef- fect of acoustic pollution and environment acoustics to psychological and physiological effects in human subjects.

As the research is ongoing, the aim of this article is to lay the foundations for the extension of the Yin Yang theory to the fields of sound analysis and synthesis, music clas- sification and composition, and hence to any possible application in the field of musicology, music therapy, com- position, arts and performance. This article focuses on the basic physical properties of sound, acoustical signals and time-frequency analysis and how they can be analyzed in the context of the Yin Yang theory.

Section 2 introduces briefly the Yin Yang theory and the foundations for the extension of the Yin Yang theory to acoustics, the spectrogram in Figure 1 depicts two different (orthogonal) signals such as a long sine wave (extremely compact in frequency) and a Dirac pulse (extremely compact in time) and a possible transition between the two.

3. EXTENSION OF THE YIN YANG THEORY TO SOUND

In this paper we demonstrate how the rules on the inter- play between Yin and Yang can be applied also to sound. Sound is a continuous time-domain signal, physically rep- resenting the pressure variation from its local atmospheric value

\[ \rho_0 \]

with \( p(\tau) \) being the instantaneous pressure level. Inci- dentally, in the fluctuation generating any audible phe- nomenon, the pressure \( p(\tau) \) expands and contracts, thus varying between two opposite polarities. The more the energetic this variation is, the more the sound a sound is perceived. Highly energetic sounds are regarded as Yang (Yang is generally associated with Energy), as opposed to weak sounds (silence is regarded as Yin).

As explained before, Yin is here regarded as the prin- ciple of expansion and Yang as the principle of contrac- tion. Therefore, it can be assumed that a signal \( s(t) \) that is compact in time can be seen as a Yang sound, while one that is extended in time is more Yang. In the field of sig- nal processing and analysis, the time localization[6] of a signal is usually a measure of a compact yet optimal [27] basis function used for the analysis is a classical topic.2 Following what has been stated, it can be said that a perfect sound cannot exist in reality, as it would extend indef- initely in time, spanning the whole universe history. The same stands for a perfect Yang sound, which would be a perfect Dirac impulse, which is impossible to have due to the natural low pass nature of physical systems, whose convolution brings to a "smear" version of any perfect pulse. Although mathematical abstractions such as the in- definitively extended in time or frequency are conceivable and useful to gather a discrimination of Yin and Yang, they are of no use in reality just as the Fourier transform as a mathematical operator \(^3\) as will be discussed in more detailed below, the perfect Yin or perfect Yang can not be experienced.

To further clarify this first step in extending the Yin Yang theory to acoustics, the spectrogram in Figure 1 depicts two different (orthogonal) signals such as a long sine wave (extremely compact in frequency) and a Dirac pulse (extremely compact in time) and a possible transition between the two.

2 It must be noted that the first work in this field is probably by Dennis Gabor, well known to the computer music community for the inspira- tional effect of Gabor atoms described in his Nature article [10]. These are said to have inspired early formulation of granular synthesis, but they are not the same objects. In the signal processing community, some trans- formations of the Gaussian basis functions that makes their use impractical. In practical signal processing context, such as basis functions (or more simply, windows) are needed, as the ones used in the Wavelet Transform [6].

3 Of course, we can consider the discrete set of the Fourier operator over real-life signals. "Through mathematically this theorem is beyond re- proach, even experts could not at times conceal an uneasy feeling when it came to the physical interpretation of results obtained by the Fourier method" [9].

2Electronic Dance Music

Figure 1: Spectrogram of two opposite signals: a sine wave and a dirac pulse and the transition between the two.
and carry similar polarity. By combining all the different properties it is possible to better understand the nature of a sound, compared to other sounds taken as reference. It must be noted however that a reference sound, i.e. a perfectly balanced sound does not exist: can an intermediate time compactness or an intermediate frequency localization between 0 and infinite be found? Can the bounds given by human hearing (in both time resolution and frequency response thresholds) or by digital sampling (in both time resolution and the Nyquist frequency limit) be used to find an intermediate sound? This is questionable as other variables also appear related to human perception (should the intermediate frequency be found as the arithmetic mean of the 20 Hz - 20 kHz auditory range, in a linear scale or in an octave-spaced scale?). Although finding a “standard reference sound” in scientific terms is difficult, perhaps impossible, the human mind seems capable of assessing the absolute value of a sound (its pitch or loudness, for example) with ease. Any non-impaired listener, when presented with a pitched sound, will be able to define it as either “high” or “low”, according to a personal scale. “To some extent tonal relations are computed even when only a single tone is higher or lower, or longer or shorter, than a conventional tone encountered across the course of a lifetime” [14]. The same can be said about a listener who is familiar with the Yin Yang theory, assessing a presented sound in terms of Yin and Yang. We clearly relates to what stated in Section 2, (3). When dealing with acoustic signals, several properties can be analyzed in terms of compactness, energy and transformation. Table 1 reports some of these properties and their assessment in terms of Yin and Yang following the criteria introduced hitherto for analysis of acoustic signals.

Table 1: Description of sound properties in terms of Yin (⊙) and Yang (△).

| Envelope shape (attack, decay) | △ | sharp | △ | smooth |
| Envelope | △ | compact | △ | wide |
| Periodicity | △ | aperiodic | △ | periodic |
| Spectral content | △ | rich | △ | poor |
| Pitch | △ | high | △ | low |
| Sound Pressure Level | △ | high | △ | low |

After having briefly discussed general properties of sound, the four basic principles (1)-(4) reported at Section 2 will be discussed again but directly applied to sound.

(1) Yin and Yang stem from the same source: a fluctuation of a physical quantity, which bears both the two opposites at the same time in a specific ratio.

(2) A sound is always generated by transformation and a fluctuation between opposite polarities, in the same way as Yin and Yang continuously change one into the other. In Chinese theories this turnover is depicted similarly to what we know as a harmonic system, as illustrated in Figure 2.

Figure 2: The turnover between Yin and Yang as an oscillating harmonic system.

Concepts like chaos and randomness were not known at the time. Fluctuations between the two opposites are found also when looking at the evolution of a sound at a higher level. Any sound will raise (birth is regarded as Yang in the theory) and fade (aging and death of living beings is regarded as Yin), its spectral content may change with time, etc.

(3) No sound can be measured in terms of Yin or Yang as it would be with a physical quantity, as the true nature of Yin and Yang cannot be known by pure rational thought alone, while the second provides the localization of a signal (the closer to the attack or decay time, the more Yang). The resulting feature sums the two LLDs with weight w to be evaluated empirically, providing sounds that are more Yang in their nature with higher indexes:

\[ F_1 = w \cdot LLD_1 + (1 - w) \cdot \frac{1}{t_d} \cdot (TC - t_\text{dec}) \]

where \(t_d\) is the decay time and the attack time is supposed to be at \(t = 0\).

(4) Yin and Yang cannot be tore apart or exist as separate entities: a sound that is perfectly Yin or Yang cannot exist in reality as it would have infinite extension in time or in frequency. This is also reflected in the analysis of any signal: analyzing the two opposite aspects of sound, i.e. time and frequency, cannot be arbitrarily perfect. The resolution of one dimension in a space-phase representation of the signal, such as the spectrogram, cannot be arbitrarily high without affecting the resolution of the other dimension. This is formalized in the indeterminacy principle exposed by Gabor in [9], which in turn relies on the similar principle by Heisenberg in quantum physics.

4. ASSESSING YIN AND YANG

4.1. By audio analysis techniques

Discrimination of a sound’s character can be done by ear, provided that the listener knows well enough the Yin Yang theory. However, to enable human-machine interaction, automatic analysis and classification, some algorithms can be designed in order to extract information in terms of Yin and Yang. To do so, several low-level descriptors (LLDs) are combined to obtain two high-level features that analyze the sound in time and frequency. In this work, the proposed LLDs are taken from the well documented MPEG-7 standard [1].

Figure 1: this feature highlights the temporal characteristics of signals by adopting the Log(AttackTime) (LAT) and TemporalCentroid (TC) features. The former provides a measure of the rise time of a signal attack time (the faster the better), while the second provides the localization of a signal (the closer to the attack or decay time, the more Yang). The resulting feature sums the two LLDs with weight w to be evaluated empirically, providing sounds that are more Yang in their nature with higher indexes:

\[ F_1 = w \cdot LLD_1 + (1 - w) \cdot \frac{1}{t_d} \cdot (TC - t_\text{dec}) \]

where \(t_d\) is the decay time and the attack time is supposed to be at \(t = 0\).

(2) This accounts for the change or the difference between different sounds that are of interest, e.g., is the detection of a specific event in a pleasing way to gather the attention of the audience? Is a noisy glitch better for an alarm clock tone than an ascending sound texture? By collecting assessment in the time domain (at different scales) and in the frequency domain, these measures can provide a relative assessment. Unfortunately, a perfect intermediate sound to use as a reference cannot be found a priori. Reference sounds can, however, be chosen depending on the context. Also, the evolution of a sound or a group of sounds can be evaluated, as generally speaking it is of higher interest to know how possible evolutions rather than how it is in a specific instant.

4.2. By assessment of human psychological and physiological parameters

Another concept arising from direct comparison with macrorobotics and similar disciplines is that the nature of a sound can be determined by its effect on human subjects. For instance a Yin sound (e.g. a prolonged pad texture with slow attack) is expected to have a Yin effect on the mood or the physiology of a listening subject that can be qualified by questionnaires or quantified by measuring its heart rate, blood pressure, respiratory rate, EEG and other parameters. Similar test involving music have been used in a wide variety of works spanning from evaluation of chord progressions [8] and variation of physiological parameters under exposure to musical stimuli [29], or involving other acoustic stimuli in psychology [21] and pedagogy [19] to human and animal environmental health [22]. The methodology employed in these works can be extended to the evaluation of sounds and music of Yin or Yang character. Once a scenario is designed, including the description of the desired subject, methodology of the trial and the expected effects to be monitored, tests can be conducted and sounds can be classified by the effect they induce on psychological or physiological parameters. A pattern must be then extracted to find acoustic or musical features that are correlated with the obtained effect. The preparation of the test procedure and the administration of the stimuli must be carefully planned. The Yin Yang theory does not rely on absolute concepts, e.g. the effect of a certain stimulus on an individual, conditioned on the type of subject, the length of the stimulus administration (which on the long term may cause the opposite effect), etc. Our research on this topic is very promising though at its early stages. In future works, case studies will be reported in order to show the accuracy of the ancient Yin Yang theory in predicting effects of auditory stimuli on subjects.

5. CONCLUSIONS AND FUTURE WORK

This work provides a first extension of the ancient Yin Yang theory to the case of acoustic signals and sound perception. Motivation for such a task is provided for several fields of application. Extension of the theory to musical structure is still to be documented as more complex aspects are involved. However, it can be regarded as a particular case of the time-frequency conceptualization discussed in this paper, in which (i) the temporal scale is more relaxed (the shortest events are never explicit), (ii) events are (ideally) quantized and (iii) pitched events are (ideally) locked to discrete steps in frequency depending on the tonal system in use. Also, interplay between events affects the quality of a musical verse (clustering vs. sparseness, dissonance vs. consonance, etc.). Characterizing music will require more complex analysis techniques than those introduced in the present paper as cognitive aspects are as important as the perception of music. A framework involving information theory, entropy and their relations to the Yin and Yang principles.
will be proposed. The use of the Five Elements theory can prove helpful to connect this knowledge to moods and emotional states.

The concepts adopted for acoustic signals can be further extended to other kind of signals by using the same principles. Electromagnetism can easily relate to this dualistic theory as it has a innate dualism also from a physics standpoint. Physiological signals are of particular interest as they can be correlated with other aspects of human health considered in the Yin Yang theory and the Five Elements theory.

As Yin or Yang do not represent any physical quantity, their assessment is no easy task. Two simple algorithms are proposed as a high-level features for a Yin Yang sound analysis. Their accuracy in describing sounds is still to be quantified. Assessing the effects of sounds on subjects’ physiological parameters is also regarded as another way of understanding the nature of their properties and their relation to human health, psyche and physiology. Studying the effect of sounds on physical and mental states will gather further insight on their nature and will help providing the best analytical tools for their selection and manipulation. The subtle, relative and fuzzy nature of this theory makes its use, however, delicate. The mastering of these concept is not easy nor fast to obtain and surely always perfectable. Much work must be conducted in all the areas reported above; the effort, however, is motivated by the ability to construct a new framework enabling for a more holistic approach to the use of music and sound, and, ultimately, to the realization of Fritjof Capra’s dream: that is the renewal of western sciences by their harmonization with ancient wisdom.

6. REFERENCES


NAVIGATING VARIATION: COMPOSING FOR AUDIO MOSAICING

Diemo Schwarz
Ircam-CNRS-STM3
Centre Pompidou, Paris
diemo.schwarz@ircam.fr

Benjamin Hackbart
CRCA
University of California, San Diego
hackbart@ucsd.edu

ABSTRACT

We present a method, applicable to corpus-based concatenative synthesis and specifically to audio mosaicing, that assists the composer in exploring the relationship between the parameterization of a concatenative algorithm and the resulting similarity between the output sound and the original target soundfile. Rather than focus solely on straightforward imitation, our work is predicated upon the notion that similarity can be manifest in a variety of perceptually meaningful ways and that both semblance and dissemblance have compositional utility. Our method consists of visualizing a collection of concatenated outputs, each of which is a unique solution to the problem of matching the same target soundfile with the same sound database but using a different combination of descriptor weights. We create a solution space where the location of each output is modeled by its similarity to the target as well as by its similarity to each other solution. Visualization and navigation of this space is made possible through a multi-dimensional scaling algorithm, permitting 2D browsing, aural feedback, and the composition of paths through the solution space. This meta-control framework helps to give the composer a more comprehensive understanding of concatenative potential. By arranging concatenated outputs into different regions of similarity and dissimilarity, the solution space provides a rich and expansive terrain for compositional exploration and discovery.

I. INTRODUCTION

The heightened ability to design and manipulate sonic morphology is an alluring aspect of electronic music composition. Much music in the fixed media tradition relies on ordering sonic chunks intuitively, by hand, in order to create temporal structures which evoke perceptually meaningful ways and that both semblance and dissemblance have compositional utility. Our method consists of visualizing a collection of concatenated outputs, each of which is a unique solution to the problem of matching the same target soundfile with the same sound database but using a different combination of descriptor weights. We create a solution space where the location of each output is modeled by its similarity to the target as well as by its similarity to each other solution. Visualization and navigation of this space is made possible through a multi-dimensional scaling algorithm, permitting 2D browsing, aural feedback, and the composition of paths through the solution space. This meta-control framework helps to give the composer a more comprehensive understanding of concatenative potential. By arranging concatenated outputs into different regions of similarity and dissimilarity, the solution space provides a rich and expansive terrain for compositional exploration and discovery.

2. MODELING THE VARIATION SPACE

Unit selection in AUDIOGUIDE is made according to Euclidean distance calculations using continuously valued amplitude measurements, spectral features, and morphological pitch estimates. Evaluating time varying features permits the preservation of the morphological profile of target and corpus units. Corpus units which overlap in time are selected according to an algorithm which approximates the composite of descriptors, thus enabling the selection of polyphonic mixtures which match the target’s morphological profile.

When parameterizing a concatenation, the user may prescribe any number of features with different weights in order to influence each feature’s saliency during unit selection. In addition to varying the weights of features, the user may specify different normalization and transformation strategies for each feature, permitting more expressive control over the sonic results.

Despite decreasing the fidelity of imitative similarity, these normalization and data transformation strategies are significant in that they permit the user to shape and sculpt the target’s representation in order to alter the concatenated output. Thus, rather than considering the target soundfile as a fixed object for imitation, these normalization and transformation tools allow the user to deploy the gestural profile of the target with an enhanced degree of creative freedom. This encourages the composer to treat the target soundfile as a set of correlated features, the composite of descriptors, which can be obtained with a single target and a single database. Consequently, the resulting parameter space which affects a resynthesis has a rather high number of dimensions, which would not otherwise try to make it amenable for efficient musical exploitation.

The space of solutions the AUDIOGUIDE algorithm can be represented as a (not necessarily Euclidean) space populated by the set of possible solutions of a resynthesis of one target sound. We use the composition distance, which we define as the distance between two solutions and in equation (1) is questionable, since it compares the feature weights used during the selection of one solution, and not the timbral characteristics of the solutions themselves. Thus, this distance is situated on a conceptual level, possibly removed from the perceptual implications of the parameterisation of the selection.

Consequently, the parameter space which affects a resynthesis has a rather high number of dimensions, which would not otherwise try to make it amenable for efficient musical exploitation.

The space of solutions the AUDIOGUIDE algorithm can be represented as a (not necessarily Euclidean) space populated by the set of possible solutions of a resynthesis of one target sound. We use the composition distance, which we define as the distance between two solutions and in equation (1) is questionable, since it compares the feature weights used during the selection of one solution, and not the timbral characteristics of the solutions themselves. Thus, this distance is situated on a conceptual level, possibly removed from the perceptual implications of the parameterisation of the selection.

Consequently, the parameter space which affects a resynthesis has a rather high number of dimensions, which would not otherwise try to make it amenable for efficient musical exploitation.

The space of solutions the AUDIOGUIDE algorithm can be represented as a (not necessarily Euclidean) space populated by the set of possible solutions of a resynthesis of one target sound. We use the composition distance, which we define as the distance between two solutions and in equation (1) is questionable, since it compares the feature weights used during the selection of one solution, and not the timbral characteristics of the solutions themselves. Thus, this distance is situated on a conceptual level, possibly removed from the perceptual implications of the parameterisation of the selection.

Consequently, the parameter space which affects a resynthesis has a rather high number of dimensions, which would not otherwise try to make it amenable for efficient musical exploitation.

The space of solutions the AUDIOGUIDE algorithm can be represented as a (not necessarily Euclidean) space populated by the set of possible solutions of a resynthesis of one target sound. We use the composition distance, which we define as the distance between two solutions and in equation (1) is questionable, since it compares the feature weights used during the selection of one solution, and not the timbral characteristics of the solutions themselves. Thus, this distance is situated on a conceptual level, possibly removed from the perceptual implications of the parameterisation of the selection.

Consequently, the parameter space which affects a resynthesis has a rather high number of dimensions, which would not otherwise try to make it amenable for efficient musical exploitation.

The space of solutions the AUDIOGUIDE algorithm can be represented as a (not necessarily Euclidean) space populated by the set of possible solutions of a resynthesis of one target sound. We use the composition distance, which we define as the distance between two solutions and in equation (1) is questionable, since it compares the feature weights used during the selection of one solution, and not the timbral characteristics of the solutions themselves. Thus, this distance is situated on a conceptual level, possibly removed from the perceptual implications of the parameterisation of the selection.

Consequently, the parameter space which affects a resynthesis has a rather high number of dimensions, which would not otherwise try to make it amenable for efficient musical exploitation.

The space of solutions the AUDIOGUIDE algorithm can be represented as a (not necessarily Euclidean) space populated by the set of possible solutions of a resynthesis of one target sound. We use the composition distance, which we define as the distance between two solutions and in equation (1) is questionable, since it compares the feature weights used during the selection of one solution, and not the timbral characteristics of the solutions themselves. Thus, this distance is situated on a conceptual level, possibly removed from the perceptual implications of the parameterisation of the selection.

Consequently, the parameter space which affects a resynthesis has a rather high number of dimensions, which would not otherwise try to make it amenable for efficient musical exploitation.

The space of solutions the AUDIOGUIDE algorithm can be represented as a (not necessarily Euclidean) space populated by the set of possible solutions of a resynthesis of one target sound. We use the composition distance, which we define as the distance between two solutions and in equation (1) is questionable, since it compares the feature weights used during the selection of one solution, and not the timbral characteristics of the solutions themselves. Thus, this distance is situated on a conceptual level, possibly removed from the perceptual implications of the parameterisation of the selection.

Consequently, the parameter space which affects a resynthesis has a rather high number of dimensions, which would not otherwise try to make it amenable for efficient musical exploitation.

The space of solutions the AUDIOGUIDE algorithm can be represented as a (not necessarily Euclidean) space populated by the set of possible solutions of a resynthesis of one target sound. We use the composition distance, which we define as the distance between two solutions and in equation (1) is questionable, since it compares the feature weights used during the selection of one solution, and not the timbral characteristics of the solutions themselves. Thus, this distance is situated on a conceptual level, possibly removed from the perceptual implications of the parameterisation of the selection.

Consequently, the parameter space which affects a resynthesis has a rather high number of dimensions, which would not otherwise try to make it amenable for efficient musical exploitation.
In this approach, we map the high-dimensional dissimilarities between two elements to the nominal lengths of simulated springs linking two masses. The model is then run iteratively in 2D, updating force, speed, friction, viscosity at each step, until it converges (or until the user stops the iterations). Here, the target is mapped to a fixed mass, and the target distances $d$ are mapped to a link to each of the $n$ solutions. For initialisation, we place the solutions as movable masses in circles around the target, with a radius corresponding to the target distance and random angle. See figure 3 for an example of this placement. Then, we create links between each solution and all others (729 in our example), with lengths given by the inter-solution variation distance matrix $v$.

Running the mass–spring–damper model, the similar-sounding solutions will try to get close to each other, and push back dissimilar solutions, thus laying out the solution space along a path from one cluster of very dissimilar solutions, passing by the closest solutions, and reparting into a very different realm, also shown in figure 6. In an actual resulting composition, the target would of course vary over time, and the path might be stretched out over the duration of a part of the piece.

### 4. IMPLEMENTATION

The variation explorer within which the navigation of the variation space takes place is implemented in MAX/MSP based on CATART [6] and the FTM [3], MNM [1], and GABOR [4] extension libraries, taking advantage of FTM&Co’s advanced data structures such as matrices and dictionaries, and the real-time optimised operators that work on these. It allows to load the result data and distance matrices from a run of AUDIOGUIDE, to interactively control the layout process by manipulating the parameters of the mass-spring-damper model while it is running, and to browse the corpus of solutions by moving over the points with the mouse, using CATART’s fence or click mode. These browsing movements can be recorded and the AUDIOGUIDE parameters corresponding to each closest solution written to a multi-break-point-function file. For smoother transitions, an interpolation between the parameters of the three nearest neighbor points could also be output.

This way, a path through the 6-dimensional parameter space of AUDIOGUIDE can be composed based on audition of example solutions, and navigating in terms of resemblance to the example target. The accompanying webpage gives an example of browsing through the solution space along a path from one cluster of very dissimilar solutions, passing by the closest solutions, and reparting into a very different realm, also shown in figure 6. In an actual resulting composition, the target would of course vary over time, and the path might be stretched out over the duration of a part of the piece.

5. CONCLUSION

The work presented here is an encouraging first realization of a meta-control framework for better integrating audio mosaicing into a compositional workflow. By organizing a collection of concatenated outputs according to their sonic character as well as their relation to the target soundlike, composers may explore different categories of similarity as well as the relationship between search criteria and the degree of likeness to the target. While current research in corpus-based concatenative synthesis routinely emphasizes the importance of optimizing the closest match, our work seeks to help the composer explore the concept of similarity more deliberately and with a heightened degree of nuance.

6. REFERENCES


http://imtr.ircam.fr
http://ftm.ircam.fr
This paper investigates the performance and suitability of evolutionary algorithms for music composition by enhancing representation schemes. First, we argue that genetic programming (GP) is well suited to capture higher order musical structures due to its hierarchical representation. Representational enhancements are proposed on the standard GP tree: considering different branches for different musical dimensions (pitch, duration, etc.), and making use of “Automatically Defined Functions” to define reusable patterns in the generated music. Each representation scheme is described, along with the role of genetic operators in evolving compact representations. Representations are compared for their ability to evolve a population over a range of different target melodies. The results illustrate the benefits of improvements that result from the enhancements to the basic GP tree representation.

1. INTRODUCTION

Evolutionary algorithms (EAs) are a group of methods inspired by processes from biological evolution. They have been successfully applied to many problems in search, optimisation and learning, including in the field of algorithmic music composition (see, e.g. [13,3,16]). Music composition can be considered a process of creative exploration and search of a musical space [7]. Any non-trivial musical space is potentially vast and its structure often unknown, so it comes as no surprise that EAs have been a popular choice for algorithmic composition. But even though the workspace of EAs in searching such vast spaces is of limited success in many cases [11]. In this paper, we examine a number of different representations of musical structure and test them for suitability in evolutionary music composition. A good representation will compactly and succinctly represent the most common musical structures across a variety of genres, allowing fast and efficient evolvability of musical compositions.

When designing an EA-based system, there are three main issues to consider: genotype representation, evolutionary operators, and phenotype fitness evaluation. Representation – the scheme genotypes use to build a phenotype (compositions) – effectively determine the type of compositions evolution can produce. For example, a linear representation of individual notes cannot produce a chord. Operators determine how genotypes change during the evolutionary process, affecting whether phenotypes change slowly and gradually, or can perform large “jumps” in the space of possible compositions. Finally, fitness provides a driving force by influencing the probabilities of survival and reproduction of individuals.

We have designed a series of representation schemes and genetic operators for music composition systems based on standard genetic programming (GP) techniques [8]. In order to compare each scheme’s performance, we used a simple experiment: searching the musical space defined by the representation for a variety of pre-defined target melodies and comparing each representation’s ability to evolve melodies similar to the target. The edit distance of a candidate melody to the target melody was used as a fitness measure. While not as accurate as perceptual-based musical distance measures, edit distance provides a reasonable and easily computable measure of similarity between two melodies.

The next section (2) briefly examines existing musical representations for evolutionary algorithmic composition. This is followed by a discussion on music characteristics and representation considerations in Section 3. Section 4 describes the representation schemes we have developed and summarises their performance, which is followed by a brief discussion of results and conclusions in the final section.

2. REPRESENTATIONS FOR EVOLUTIONARY ALGORITHMIC COMPOSITION

An important consideration for any algorithmic representation is that each musical note has many different attributes, including pitch, duration, timbre, volume, and articulation. This raises a critical design decision: should these different attributes be grouped together as a single unit of representation, or should they be kept separate, coming together only when the representation is converted to actual music?

Research in human music perception supports a distinction between pitch- and time-based relationships. Pitch intervals and melodic contours are processed in different regions of the brain than articulatory units [15]. In contrast, many evolutionary composition systems tend to tie different attributes of the note together, probably because they are traditionally considered as a whole (e.g. as in traditional Western notation). Dahlstedt, for instance, uses a representation based on graphs (with custom edges controlling the type of traversal) in which each leaf node contains a note and composition [4]. The information stored for each note includes onset time, pitch, duration, articulation, and pitch, amplitude, duration, and articulation. Fru et al. employ a genetic algorithm to compose musical phrases consisting of notes represented as (pitch, duration, intensity) triples [6]. Povel’s Melody Generator [14] generates hierarchically organised temporal sequences of notes. Once again, all the attributes (duration, timbre, etc.) have been tied together in each note.

Somewhat differently, Biles utilises a string representation in his GA-based GenJam, describing it as “a cooperating, two-level, position-based, binary representation scheme”. Each chromosome represents a series of eight events, which could be a new note, a rest, or a hold; one for each eighth note duration of a 4/4 measure [2]. In other words, there is no explicit value indicating the duration of each note.

3. TOWARDS A MORE DEVELOPED REPRESENTATION SCHEME

Dahlstedt divides genetic representations into three categories: basic, structural, and generative [5]. In a basic representation, such as a list of pitches and durations, the genotype essentially is the phenotype. An improvement is to incorporate structural information into the representation. For instance, musical structures can be grouped into a hierarchy (e.g. each phrase consists of motifs, each motif consists of notes, and so forth). Generative representations draw on the ability of certain processes to generate complexity far greater than their specification, a principle known in computing as database amplification [12].

A number of authors have described the organisation of tonal music as a hierarchy (e.g. [1,9]). Experimental work in psychology, neuroscience, and electrophysiology supports the hypothesis of a hierarchical and modular organisation of music perception in brain [15]. The context in which a musical note is set could be much further than just a few previous notes. Distant notes or phrases are often more related to a particular note semantically than to immediate neighbours. In contrast, traditional music notation, event-based sound control codes (e.g. MIDI), and many algorithmic composition representations structure music chronologically.

In contrast, structural (e.g. Melody Generator [14]) and generative representations (e.g. NeV/Muse [10]) capture information about the hierarchical structure of music, potentially making them more aligned with human music perception and composition [11]. A generative representation provides an efficient framework for encoding complex, repetitive patterns [12].

4. EXPERIMENTAL RESULTS

We begin our experiments with a generative representation based on standard GP techniques. Notes (as a collection of attributes such as pitch, duration, etc.) and a set of musical functions form the nodes of a tree which is then traversed to generate a melody. We denote this the “standard GP-Rep”. Further developing this representation, attributes of notes are separated and stored on different branches under a common root of the GP tree. In this case, which we refer to as “extended GP-Rep”, each branch under the root node represents a different dimension of the composition (e.g. one branch for encoding the pitch sequence, one for the duration sequence, or rhythm, and so on). In a further modification, we utilised automatically defined functions (ADFs) [8] as a means of further compressing the information captured by this representation. This design, which we refer to as “extended GP-Rep with ADFs”, permits definition of reusable musical patterns, which in turn results in a compressed form of information encoding. The evolutionary algorithm for each representation is based on standard GP [8], but is different in terms of how crossover is performed, in addition to some other details, which we describe shortly.

For all representations tested, each individual encodes a melody, i.e. a sequence of notes; and each note is considered to be a (pitch, duration) tuple, for the sake of simplicity. The convergence of populations to pre-defined target melodies was compared for each of the three representations. The goal was to minimise the fitness value, defined as:

$$fitness(m) = \sum_{i=1}^{\text{dim}(m)} \frac{\text{dim}(m) - \text{dim}(\text{target})}{\text{length}(\text{target})}$$

where the function $\text{dim}$ returns Levendrin distance between two arguments, $\text{dim}$ returns the dimension of the melody (i.e. pitch or rhythm), $\text{target}$ denotes the target melody, $m$ is the phenotype melody to be evaluated, and $\text{length}(\text{target})$ is the number of the notes in the target melody. The parameters used for each experiment are shown in Table 1. In order to assess the performance of each representation over a variety of styles, two melodies from each of five different genres (classical, jazz, pop, folk, and nursery rhymes) were used as target melodies (see Table). All results for each representation show the average of 100 independent runs, i.e. 10 runs for each target melody.

Table 1. Parameters used for running the programs

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size</td>
<td>500</td>
</tr>
<tr>
<td>The number of generations</td>
<td>700</td>
</tr>
<tr>
<td>The maximum depth of tree</td>
<td>10</td>
</tr>
<tr>
<td>Crossover rate</td>
<td>0.65</td>
</tr>
<tr>
<td>Darwinian reproduction rate</td>
<td>0.20</td>
</tr>
<tr>
<td>Mutation rate</td>
<td>0.15</td>
</tr>
</tbody>
</table>
**Table 2. Primitives for the standard GP-Rep.** The argument \( S \) is a string of notes i.e. \((\text{pitch},\text{duration})\) tuples.

<table>
<thead>
<tr>
<th>Functions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concat(S1,S2)</td>
<td>concatenate ( S1 ) with ( S2 )</td>
</tr>
<tr>
<td>Repeat(S)</td>
<td>repeat ( S )</td>
</tr>
<tr>
<td>ShiftUp(S)</td>
<td>transpose pitches up by one semitone</td>
</tr>
<tr>
<td>ShiftDown(S)</td>
<td>transpose pitches down by one semitone</td>
</tr>
<tr>
<td>Double(S)</td>
<td>double durations</td>
</tr>
<tr>
<td>Half(S)</td>
<td>half durations</td>
</tr>
<tr>
<td>RetroPitch(S)</td>
<td>Retrograde only pitches of ( S )</td>
</tr>
<tr>
<td>RetroDuration(S)</td>
<td>Retrograde only durations of ( S )</td>
</tr>
</tbody>
</table>

**Table 3. Primitives for the extended GP-Rep.** The argument \( S \) is a string of integer values, which could be either a pitch or duration sequence.

<table>
<thead>
<tr>
<th>Functions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concat(S1,S2)</td>
<td>concatenate ( S1 ) with ( S2 )</td>
</tr>
<tr>
<td>Repeat(S)</td>
<td>repeat ( S )</td>
</tr>
<tr>
<td>ShiftUp(S)</td>
<td>transpose pitches up by one semitone</td>
</tr>
<tr>
<td>ShiftDown(S)</td>
<td>transpose pitches down by one semitone</td>
</tr>
<tr>
<td>Double(S)</td>
<td>double durations</td>
</tr>
<tr>
<td>Half(S)</td>
<td>half durations</td>
</tr>
<tr>
<td>RetroPitch(S)</td>
<td>Retrograde only pitches of ( S )</td>
</tr>
<tr>
<td>RetroDuration(S)</td>
<td>Retrograde only durations of ( S )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Terminals</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pitch,duration)</td>
<td>( \text{pitch} ) an integer from 0 (C0) to 127 (G10), ( \text{duration} ) an integer from 0 (dotted whole) to 15 (128\textsuperscript{th} note)</td>
</tr>
</tbody>
</table>

**4.1. First representation: inspired by standard GP**

As a first experiment, we used a representation from standard genetic programming \([8]\), with a set of eight functions (Table 2) and notes as terminals. Figure 1 illustrates an example individual for this implementation. The three reproduction operators “Darwinian reproduction”, “crossover”, and “mutation” follow the standard definition from Koza \([8]\). This algorithm was run with the parameters in Table 1 to find melodies similar to the target melodies. Figure 4 illustrates the results averaged over 100 independent runs.

**4.2. Second representation: multidimensional tree**

In the second representation (extended GP-Rep), individuals have one branch under the root entry for each attribute dimension (here pitch and duration). This separation also enables us to have common functions for branches by abstracting their behaviour (e.g. Retrograde subsumes RetroPitch and RetroDuration, see Table 3). Figure 2 illustrates an example of an individual in extended GP-Rep.

In this representation, each dimension has one or more function-defining branches (ADFs), and one main result-producing branch (the RPB). An ADF is defined as a function which gets a terminal as its input and returns a sequence of terminals as output. Once defined, an ADF can be called repeatedly by the RPB with different arguments. Since one function can represent many musical phrases (which are not necessarily the same, though follow the same pattern) the representation is efficient in compressing repeated information. Figure 3 shows an example of an individual in the extended GP-Rep with ADFs. The maximum number of ADF branches on each dimension and the maximum depth of ADFs are set as program parameters. When initialising the population, ADFs are randomly generated independently (i.e. they do not call each other) for each individual from the primitives shown in Table 3. Next, the RPB is randomly generated from the union of the primitive and ADF sets. During evolution, the crossover operator is applied only to the RPB branch. If a crossover results in moving a branch which has a call to an ADF, then the ADF needs to be copied to the appropriate destination tree. The ADF is removed from the original tree if it is no longer used. Mutation operators may be applied to either of the branches, including ADFs. A mutation in a frequently invoked ADF usually results in an explorative change, whereas mutations on the RPB could be either explorative or exploitative depending on the branch undergoing the mutation.

**4.3. Third representation: using ADFs**

Musical information often contains repetitive patterns. As repetitions are not necessarily consecutive, the function \texttt{Repeat} cannot fully capture this feature efficiently. Furthermore, repeated patterns can be found at different levels of abstraction (e.g. intervals at a higher level abstraction from absolute pitches). Koza’s automatically defined functions (ADFs) are used to evolve reusable components, which can be invoked repeatedly, typically with different inputs. In the third representation, we augmented the extended GP-Rep with ADFs as a means of capturing repetitive patterns and storing them as reusable components. In this representation, each dimension has one or more function-defining branches (ADFs), and one main result-producing branch (the RPB). An ADF is defined as a function which gets a terminal as its input and returns a sequence of terminals as output. Once defined, an ADF can be called repeatedly by the RPB with different arguments. Since one function can represent many musical phrases which are not necessarily the same, though follow the same pattern the representation is efficient in compressing repeated information. Figure 3 shows an example of an individual in the extended GP-Rep with ADFs.

**Figure 2.** An example of an individual in the extended GP-Rep.

**Figure 3.** An example of an individual in the extended GP-Rep with ADFs which shows only the pitch branch.

**Figure 4.** A comparison between “standard GP-Rep”, “extended GP-Rep”, and “extended GP-Rep with ADFs” on converging to the target melodies. This figure shows an improvement for extended GP-Rep over standard GP-Rep as a result of separating out different dimensions into different branches on the representation tree. Further improvement using ADFs shows up after generation 100. The results are the average of 100 independent runs for each representation (i.e. 10 runs for each target melody).

**Table 4.** Best individual found for each target melody. S-GP, E-GP, and E-GP* stand for standard GP-Rep, Extended GP-Rep, and Extended GP-Rep with ADFs respectively.

<table>
<thead>
<tr>
<th>Melody Name</th>
<th>S-GP</th>
<th>E-GP</th>
<th>E-GP*</th>
</tr>
</thead>
<tbody>
<tr>
<td>For Elise (Beethoven)</td>
<td>0.65</td>
<td>0.62</td>
<td>0.53</td>
</tr>
<tr>
<td>Symphony No. 40 (Mozart)</td>
<td>0.68</td>
<td>0.60</td>
<td>0.56</td>
</tr>
<tr>
<td>West End Blues (Armstrong)</td>
<td>0.89</td>
<td>0.78</td>
<td>0.71</td>
</tr>
<tr>
<td>Wonderful World (Armstrong)</td>
<td>0.30</td>
<td>0.29</td>
<td>0.26</td>
</tr>
<tr>
<td>Hey Jude (Beatles)</td>
<td>0.91</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td>A Man After Midnight (ABBA)</td>
<td>0.15</td>
<td>0.13</td>
<td>0.23</td>
</tr>
<tr>
<td>Turkish Folk</td>
<td>0.84</td>
<td>0.70</td>
<td>0.56</td>
</tr>
<tr>
<td>Persia Folk</td>
<td>0.55</td>
<td>0.44</td>
<td>0.48</td>
</tr>
<tr>
<td>Twinkle Little Star</td>
<td>0.47</td>
<td>0.45</td>
<td>0.36</td>
</tr>
<tr>
<td>The Farmer in the Dell</td>
<td>0.50</td>
<td>0.37</td>
<td>0.35</td>
</tr>
</tbody>
</table>
5. DISCUSSION AND CONCLUSIONS

The standard representation suffers from the problem of its pitch and duration being tied together in one structure, the "Note". This implies there may be cases where one aspect of the music, say the duration sequence, gets close to the duration sequence of the target melody, while leaving much room for improvement in another aspect (i.e. pitch). For the target melody shown in Figure 5 for example, the individual in Figure 6 gets a relatively good fitness value because its rhythm closely matches the target melody, although it does not sound similar because the pitches are quite distant. In this situation, most attempts to fix the pitches would result in worse fitness, due to a single mutation or crossover effecting pitch and duration simultaneously. This situation is akin to being trapped in a local optimum. We can avoid increases in rhythm distance if we can modify pitches independently of the durations. We attempted to minimise this problem by separating out duration-related functions from pitch-related functions (e.g. RetroPitch and RetroDuration), but we found the problem inevitable because of the function "Repeat", which could not be split into separate functions.

Next, we evolved pitches and durations on different branches for each individual. Providing functions that can be applied to different aspects of music independently, allowed the extended GP-Rep to find melodies closer to the target. Figure 4 shows how the standard GP-Rep fails to get close to the target melody, whereas the population in the extended GP-Rep converged, on average, closer to the target. The problem of standard GP-Rep sometimes takes an extended GP-Rep to find melodies closer to the target. The figure shows how effectivly the extended GP-Rep with ADFs can represent melodies with repetitive patterns. The results are the average of 10 independent runs for each algorithm.

Figure 6. An example of an individual in the standard GP-Rep which gets a high fitness because of its rhythm closeness to the target melody shown in Figure 5, although it does not sound similar due to pitch differences.

Figure 5. The beginning of the Turkish March by Mozart

Figure 7. A comparison between Standard GP-Rep, extended GP-Rep, and extended GP-Rep with ADFs using a section of "Symphony No. 40" (Mozart) as the target melody. The figure shows how effectivly the extended GP-Rep with ADFs can represent melodies with repetitive patterns. The results are the average of 10 independent runs for each algorithm.

As would be expected, extended GP-Rep with ADFs demonstrated the best performance for target melodies with a large number of repetitive patterns. As Figure 7 shows, when a target melody with a large number of repetitive patterns (such as "Symphony No. 40") is used, the addition of ADFs improves performance. The best individuals for this genre of music sounded very similar to the target melodies, being easily recognisable to a human listener. Conversely, the compositions without or with fewer repeated patterns remained hard to find. Using a part of the pop song "Hey Jude" (The Beatles) as the target melody. This figure shows that ADFs do not significantly improve the efficiency of the extended GP-Rep for melodies without repetitive patterns. The results are the average of 10 independent runs for each algorithm.

6. FUTURE WORK

The current design of the extended GP-Rep with ADFs suffers from a lack of domain knowledge. For instance, a pattern extracted from a perfectly valid phrase in a tonal composition, can generate a phrase some of the pitches of which fall outside the scale. In this case, this problem could be avoided if the representation took care of tonality. So, one possible improvement is to encode the basics of domain knowledge in the representation, such as key and time signatures. Evaluation will allow these features to change as required.

Further improvements could be made to the fitness measure. In this article, we proposed a simple fitness function based on the edit distance for examining the performance of representations in finding target melodies. A more sophisticated measure based on perceptual similarity was considered too difficult to use for these experiments. This edit-distance fitness function, in its current form, can not be used extensively for music composition. A more sophisticated fitness measure would focus on subjective evaluations of compositions according to preferences, or be capable of measuring a set of well-defined, musically important features.

7. REFERENCES

FLEXIBILITY, SUBLTETY AND SPONTANEITY IN NEW INSTRUMENT DESIGN: THE FEEDBACK JOYPAD

Tom Mudd
Goldsmiths, University of London, New Cross, London, SE14 6NW
UK

ABSTRACT
This paper examines important issues around the instrumental approach to live electronic music, particularly design factors in the mapping stage affecting flexibility, spontaneity and subtlety in the control of new instruments. These elements are discussed in the context of the mapping layer in the author’s own Feedback Joypad instrument, highlighting ways in which design decisions can - intentionally and unintentionally - bias the instrument towards certain musical elements.

1. INTRODUCTION
The predominant approach to creating and performing electronic music has generally been through an engagement with individual sonic parameters. This can be traced from early instruments such as the theremin, through to synthesizers and on to digital systems such as Ableton Live and Pro Tools. This parametric approach is often at odds with established modes of musical performance, as noted by Chadabe [2], musicians generally prefer to be thinking about the music rather than the instrument (though arguably less so in some forms of composition and improvisation). Attempting to juggle individual sound parameters in a live performance can be difficult without sacrificing either spontaneity or subtlety, making it difficult to play alongside acoustic performers - particularly in improvised situations. This paper is concerned with approaches to electronic instrument design that attempt to address this problem and allow electronic sound worlds to be controlled with a higher level of flexibility, subtlety and spontaneity.

These three elements play a key role in determining how much depth an instrument has: the potential for mastery and range of expression available to be included here (see the archives of the New Interfaces for Musical Expression conference for research on this).

2. REMOVING A LAYER OF THOUGHT THROUGH COMPLEXITY
Acoustic instruments rarely have straightforward relationships between input and output parameters. A reed instrument, for example, is not simply a set of individual controls for pitch, volume and timbre. The various aspects of the resultant sound are interconnected with the various input parameters in a complex web-like interweaving of parameters (see [1] and [6]) that is less intelligible in terms of individual settings, more intuitive in terms of performing complex musical gestures, and can provide a richer experience over time than a basic mapping could achieve [4]. Although the mapping is complex and the instrument can be confusing initially, it allows the performer to think directly about the sonic result of their physical gesture without having to concentrate on which control affects which parameter, enabling a more direct link between the bodily actions and sonic results. This relationship allows the sound world to be explored in an intuitive manner: an instrumental manner.

3. THE FEEDBACK JOYPAD
3.1. Sound Engine and Interface
The instrument is based around a feedback loop within the software, triggered by a short fragment of noise and then sustained as long as is required by keeping the feedback level above a certain threshold. The loop passes through filters of various kinds as shown in fig. 1, which help to control the pitch, volume and timbre of the feedback.

The multi-rate feedback filter bank and the comb filter bank each consist of three separate filters that can be given a specific pitch from the controller. In the case of the band-pass filters there are in fact up to 24 filters, as each of the three pitches can have up to 8 partials. The interaction between these filter banks and the 2-pole filter that is placed after the delay unit is a key element as the manipulation of the cutoff frequencies can cause different partials to come through at different strengths. Since the audio is in a constant feedback loop, the louder a particular frequency is, the more it will increase in volume over time, so the system generally tends towards a clear tone. The user can effect this progression by altering the pitches of the band-pass and comb filters, by cutting off different frequencies with the 2-pole filter or by having two frequencies interfere with each other (e.g. two very similar frequencies).

A dual-analog joystick is used as a physical controller to handle the various parameters within the patch. This was deemed appropriate due to the mixture of discrete and continuous controls, and for various other reasons, but the issues around interface design are too numerous to be included here (see the archives of the New Interfaces for Musical Expression conference for research on this).

3.2. Mapping Examples
Two examples of different kinds of mappings utilised in the Feedback Joypad are described below. The first is an example of a one-to-many mapping where multiple parameters of the sound engine are controlled by a single input parameter. The second is the converse: a many-to-one mapping.

3.2.1 One-to-many mapping
Although all of the joystick’s controls are utilised, some are more powerful than others. For example, a single input parameter is used in the path to alter multiple parameters within the feedback loop as shown in fig. 2. This links several aspects of the feedback loop together: the filter cutoff, the level (and method) of feedback, and the balance between the band-pass filter output and the comb filter output. Together they form a complex entity which can affect the pitch, volume and timbre of the sound being produced. It also becomes a relatively impenetrable mapping from a performer’s perspective, forcing them to focus on the sonic result of the action.

3.2.2 Many-to-one mapping
Although pitch is ostensibly controlled by setting the band-pass and comb filter frequencies, there are many other factors that contribute towards the pitch of the final sound. For example, if the 2-pole frequency is filtering out the fundamental, then a higher harmonic may be heard instead; or, if the three filter pitches as described in 3.1 are set to differing values, the frequencies may compete for dominance in the feedback loop. The list of parameters that affect pitch is lengthy: band-pass and comb filter pitch controls, pitch bend controls, the 2-pole cutoff frequency and filter type, the resonance of the band-pass filters, the number of overtones present in each filter bank, the level of feedback b) (see fig. 1); in short, almost any parameter affecting the feedback loop can potentially have a large-scale effect on the pitch of the resultant sound.

4. MUSICAL PROCLIVITIES IN THE FEEDBACK JOYPAD
The interlinking of parameters as described above affects flexibility, subtlety and spontaneity in different ways. Multiple methods of controlling pitch provides a level of nuance by allowing for not only the fine tuning of the pitch, but also for many different approaches to slipping, sliding, shifting, fading or jumping between pitches. This also provides a flexibility of sorts by providing a range of approaches to articulation, phrasing and polyphony. Flexibility is limited in other ways by this approach however. The one-to-many mapping described in fig. 2 prevents the possibility of individual manipulation of feedback parameters which narrows the range of sonic possibilities available to the performer.

This has been sacrificed in favour of subtlety, spontaneity and nuance of control over more limited sonic terrain.

4.1. Built-in Bias
Many new instruments are developed for specific performances or specific compositions. In these situations an intimate level of control is perhaps not so important and elements such as flexibility and...
instruments more flexible than others? Few would argue that the music in some way. So are some aspects of musical performance. It cannot provide equal control of all possibilities on the instrument. In the glissandi example, the pitch will only glide up and down and cannot jump to a particular harmonic approach. Rudimentary polyphony is possible, but it favours either leaving certain pitches static (drones), or keeping fixed intervals between notes. It is difficult to isolate a particular part of the design that is responsible for these predilections. With the exception of the sixth item, they all seem to be a result of the combined mappings of different instrument models. The pitch and the way they interact with the 2-pole filter that makes it easy to emphasise different harmonics. From a control perspective, it is difficult to separate the mapping from the feedback engine, and this is perhaps a false distinction. When the score is a deterministic model the separation is much clearer, but when something more akin to a physical model is used, it brings its own cross mappings into play. A particular strength of the Feedback Joypad is that it has a complex and chaotic system at the heart of it which dictates many of the mappings itself. A similar manner to the reed instruments described by Backus [1].

4.2. Bias in the Feedback Joypad

Identifying the bias in a particular instrument is not a straightforward task either. Below is a (by no means exhaustive) list of objections to the hypothesis that feedback Joypad can influence - if not impose - the music it produces:

1. Long sustained phrases are very easy to produce, whereas more fragmented phrases are relatively difficult.
2. Slowly evolving sounds are often easier to produce than more static, constant sounds.
3. Staccato sounds are possible, but it is comparatively difficult to play quick, precise sequences of short notes.
4. Beat patterns can be pronounced and are easy to control.
5. It very easy to shift sounds up and down the harmonic series which can end up imposing a particular harmonic approach.
6. Rudimentary polyphony is possible, but it favours either leaving certain pitches static (drones), or keeping fixed intervals between notes.

Biasing the control towards certain musical areas in this way will tend to bias the musical output. This is perhaps desirable in the situations described above for specific compositions, but will limit the range of possibilities on the instrument. In the glissandi example, the pitch will only glide up and down and cannot jump (as is the case in the theremin for example). This is not an issue that is unique to electronic instruments either. Zithers such as the gusheh or the koto tend to emphasise a particular scale and a reliance on pitch bending. Bagpipes lend themselves well to music based around a drone note, but are not able to produce complex counterpoint harmonics. The piano is unable to deviate from equal tempered pitches or to interfere with the development of a note that has been sounded other than by stopping it or by combining it with others (discounting for the moment the possibility of touching the piano strings). These musical biases are often so embedded that they go undetected. Sergi Jordà states that "A good instrument should not impose its music to its player" [3], but it is clear that however an instrument is designed, it cannot provide equal control of all possible musical dimensions, and will therefore influence the music in some way. So are some instruments more flexible than others? Few would argue that a violin has more scope for varied expression than a triangle, but it is a difficult thing to quantify and beyond the scope of this paper to do so.

4.3. Constraints and Coherence

The musical predilections inherent in instruments can be seen as the embedding of the compositional process within the design process, and many performers embrace this. David Wessel states that in his work, "the most important act of composition is the design of the computer model - the nature of feedback Joypad can influence - if not impose - the music it produces."

1. Long sustained phrases are very easy to produce, whereas more fragmented phrases are relatively difficult.
2. Slowly evolving sounds are often easier to produce than more static, constant sounds.
3. Staccato sounds are possible, but it is comparatively difficult to play quick, precise sequences of short notes.
4. Beat patterns can be pronounced and are easy to control.
5. It very easy to shift sounds up and down the harmonic series which can end up imposing a particular harmonic approach.
6. Rudimentary polyphony is possible, but it favours either leaving certain pitches static (drones), or keeping fixed intervals between notes.

It is difficult to isolate a particular part of the design that is responsible for these predilections. With the exception of the sixth item, they all seem to be a result of the combined mappings of different instrument models. The pitch and the way they interact with the 2-pole filter that makes it easy to emphasise different harmonics. From a control perspective, it is difficult to separate the mapping from the feedback engine, and this is perhaps a false distinction. When the score is a deterministic model the separation is much clearer, but when something more akin to a physical model is used, it brings its own cross mappings into play. A particular strength of the Feedback Joypad is that it has a complex and chaotic system at the heart of it which dictates many of the mappings itself. A similar manner to the reed instruments described by Backus [1].

5. CONCLUSION

The instrumental approach to live electronics can potentially offer a way to explore electronic sound worlds away from the abstractions of numbers and parameters. The Feedback Joypad utilises creative cross-mappings and conditional controls based on the characteristics of a computer model to help to provide the instrument with sonic depth, the potential for spontaneity, and control over the finer nuances of the sound. Various limitations and musical predilections are put into place through the combination of sound engine and mapping strategies. The degree to which flexibility is an important factor will depend on the purpose of the instrument - whether it is intended as a tool for realising a fixed composition, whether room needs to be left for interpretation or whether it needs to be able to adapt to varied situations.

6. REFERENCES

Music Journals
from Routledge

We publish a wide range of music journals to help you with your research including:

Contemporary Music Review
Journal of New Music Research
Popular Music and Society

Would you like to know more?
Each of our journals has a dedicated website with information about:
- the journal’s aims and scope
- free access to articles and other special offers
- recent and forthcoming themed issues
- calls for papers for general and themed issues
- subscribing and library recommendations

Visit the website to find a journal to match your area of interest today!