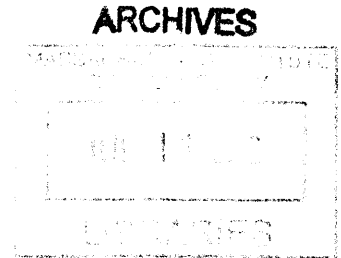


SoundStrand: a Tangible Interface for Composing Music with Limited Degrees of Freedom

by

Eyal Shaha

B.Sc., Tel-Aviv University, Israel (2003)



Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of

Master of Science
in Media Arts and Sciences

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2012

© Massachusetts Institute of Technology 2012. All rights reserved.

Author

Program in Media Arts and Sciences
School of Architecture and Planning

May 11, 2012

Certified by

Tod Machover
Muriel R. Cooper Professor of Music & Media
Thesis Supervisor

Accepted By

Mitchel Resnick
Chair, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

SoundStrand: a Tangible Interface for Composing Music with Limited Degrees of Freedom

By

Eyal Shahar

Submitted to the Program in Media Arts and Sciences
School of Architecture and Planning
on May 11, 2012, in partial fulfillment of the requirements
for the degree of Master of Science in Media Arts and Sciences

Abstract

This thesis presents SoundStrand, a novel tangible interface for composing music. A new paradigm is also presented – one that allows for music composition with limited degrees of freedom, and therefore is well suited for music creation through the use of tangible interfaces. SoundStrand is comprised of a set of building blocks that represent pre-composed musical segments. By sequentially connecting building blocks to one another, the user arranges these segments into a musical theme; and by individually twisting, stretching and bending the blocks, variations of the melodic, harmonic and rhythmic content are introduced. Software tools are made available to program the musical segments and govern SoundStrand's behavior. Additional work, namely the Coda system, is presented in order to put SoundStrand and the described paradigm in a wider context as tools for music sharing and learning.

Thesis Supervisor: Tod Machover

Title: Muriel R. Cooper Professor of Music and Media


SoundStrand: a Tangible Interface for Composing Music with Limited Degrees of Freedom

by

Eyal Shahar

The following people served as thesis readers:


Thesis Supervisor


Tod Machover
Muriel R. Cooper Professor of Music and Media
Program of Media Arts and Sciences


Thesis Reader

Mitchel Resnick
Professor of Media Arts and Sciences
LEGO Papert Professor of Learning Research
Program of Media Arts and Sciences


Thesis Reader

Morwared Mary Farhood
Assistant Professor of Music and Music Education
New York University

Acknowledgments

My thanks go out to:

Tod Machover, for giving me the opportunity and freedom to pursue my passions;

Mitch Resnick, for truly expanding my horizons and for his inspiration;

Mary Farbood, for her support, advice and understanding;

Yan Shen and Kelsey Brigance, for helping me transform ideas into reality;

Catherine Winfield and Stephen Bresnick, for being wonderful people to work with;

Jacob Fainguelernt, for his friendship and faith in me;

Yossi Vardi, for starting a chain reaction in my mind of which this work is a consequence;

Eyal Hershko, my teaching partner, and the rest of the 1577 FRC team, for a most valuable lesson about robots and people;

My colleagues at the Opera of the Future Group at the Media Lab: Peter Torpey, Elly Jessop, Akito Van Troyer and Janice Wang, for their wisdom and knowledge;

Amit Zoran, Yadid Ayzenberg, Roy Shilkrot, Nadav Aharony and Maya Orbach, for being my family away from home;

My friends at the Media Lab and in the chemistry department in MIT;

My family, for believing in me;

Julia Ma, for her love and support. L'aventure commence!

Contents

Abstract	3
Acknowledgments.....	7
Contents	9
List of Figures	13
List of Tables.....	15
1. Introduction.....	17
1.1. Constructing Music.....	17
1.2. Music, Time and Space	18
1.3. Tangible User Interfaces	19
1.4. What is SoundStrand?.....	19
1.5. Outline	20
2. Background and Related Work.....	21
2.1. Opera of the Future Group.....	21
2.1.1. Hyperinstruments	21
2.1.2. The Brain Opera.....	21
2.1.3. Toy Symphony.....	22
2.1.4. The Big Thing.....	24
2.1.5. Hyperscore.....	24
2.2. Towards SoundStrand	26
2.2.1. Chroma District	26
2.2.2. Cicadence	28
2.2.3. SoundStrand's conception	30

2.3.	Musical TUIs.....	31
2.3.1.	AudioCubes	31
2.3.2.	SoundBlocks.....	32
2.3.3.	Music Blocks	32
2.3.4.	The Tangible Sequencer	33
2.3.5.	The Music Table.....	33
2.3.6.	The reactTable.....	34
2.3.7.	Sifteo Cubes.....	35
2.4.	Computer Aided Composition Software.....	36
2.4.1.	Impromptu.....	36
2.4.2.	OpenMusic.....	38
2.4.3.	Musique Lab 2.....	39
2.5.	Contribution and Distinction from Related Work.....	39
3.	Designing SoundStrand	41
3.1.	System Overview	41
3.2.	Musical Behavior.....	42
3.2.1.	Mapping the Physical to the Musical.....	42
3.2.2.	Harmony	42
3.2.3.	Melody.....	44
3.2.4.	Rhythm.....	46
3.3.	Hardware	47
3.3.1.	Skeleton	47
3.3.2.	Skin	48
3.3.3.	Connectors	48

3.3.4.	Electronics	48
3.3.5.	Communication	49
3.4.	Software Design	50
3.4.1.	Programming Language	50
3.4.2.	System overview.....	50
3.4.3.	Updaters.....	51
3.4.4.	Content files	52
3.4.5.	User interaction.....	53
3.5.	Content Generation Tools.....	54
3.5.1.	Motivation.....	54
3.5.2.	Cell Editor	56
3.5.3.	Harmony Transition Table Editor.....	57
3.6.	Example of a SoundStrand session.....	58
4.	Beyond SoundStrand.....	61
4.1.	Introducing Coda.....	61
4.2.	Learning Online.....	61
4.3.	System Description	62
4.4.	Examples.....	63
4.5.	Philosophy of Coda.....	65
4.6.	SoundStrand and Coda.....	65
5.	Conclusion and Future Work.....	69
5.1.	Evaluation.....	69
5.1.1.	Method	69
5.1.1.	Analysis.....	69

5.2. Improving SoundStrand	72
5.3. Composing with Limited Degrees of Freedom.....	73
5.3.1. Improving the Paradigm.....	73
5.3.2. Multitude of Interfaces.....	74
5.3.3. Collaborative Composition.....	74
5.4. Reflections.....	75
Appendix A : Evaluation Survey.....	77
Appendix B : Recruitment Text.....	79
Appendix C : Consent to Participate Form.....	81
Bibliography	83

List of Figures

Figure 2-1: A performance of the Brain Opera [7].....	22
Figure 2-2: Young performers playing Music Shapers alongside a symphonic orchestra in a Toy Symphony [9]	23
Figure 2-3: Five foot model of Big Thing [13].....	24
Figure 2-4: Hyperscore project screenshot [15]	25
Figure 2-5: Chroma District	27
Figure 2-6: (a) Lateral view of a male okanagana vanduzeei cicada. (b) The tymbal. [19].....	29
Figure 2-7: Cicadence	30
Figure 2-8: AudioCubes explored by a museum visitor [20]	31
Figure 2-9: SoundBlocks [21]	32
Figure 2-10: Music Blocks (image from Amazon.com).....	33
Figure 2-11: The Tangible Sequencer [23].....	33
Figure 2-12: The Music Table [24]	34
Figure 2-13: The reacTable [27].....	35
Figure 2-14: Sifteo Cubes [29]	35
Figure 2-15: Impromptu screenshot	37
Figure 2-16: Screenshot of an OpenMusic patch [33].....	38
Figure 2-17: Screenshot of Musique Lab 2 [34].....	39
Figure 3-1: A SoundStrand with five cells.....	41
Figure 3-2: A basic phrase.....	44
Figure 3-3: The phrase transposed to match the preceding motif which is indicated by gray note heads	45
Figure 3-4: The phrase's notes transposed in response to the bending of the cell.....	45
Figure 3-5: The notes of the phrase are quantized to the pitches of the scale and the chord.	45
Figure 3-6: A disassembled cell skeleton.....	47

Figure 3-7: A cell's skin (left) and the skin presented inside-out.....	48
Figure 3-8: A cell's electronic circuit exposed.....	49
Figure 3-9: Packet formation. a) Cell A, the last in the strand, initiates a packet. b) Cell B adds its data to the packet	50
Figure 3-10: SoundStrand software system diagram.....	51
Figure 3-11: XML tree for HTT files	53
Figure 3-12: XML tree for cell content files.....	53
Figure 3-13: A screenshot of the SoundStrand software.....	54
Figure 3-14: A screenshot of the Cell Editor, showing the pitch area on the top and the rhythm area on the bottom. The red patterns are illegal as they have an incorrect number of events.....	56
Figure 3-15: A screenshot of the HTT Editor.....	57
Figure 3-16: Three cell-types programmed in the cell editor	58
Figure 3-17: A simple HTT, programmed in the HTT editor.....	58
Figure 3-18: The theme created by a strand of four cells in their original state.....	59
Figure 3-19: The theme after adjusting the harmony.....	59
Figure 3-20: The theme after modifying rhythmic patterns.....	59
Figure 3-21: The theme after modifying the melodic contour.....	59
Figure 4-1: An example of a music segment analyzed by the Coda system.....	64
Figure 4-2: The kindergarten approach to learning [52].....	67

List of Tables

Table 3-1: An example of a Harmonic Transition Table (HTT)..... 43

Table 3-2: Three versions of a phrase with modified center of rhythmical mass 46

Table 5-1: Evaluation survey results 69

1. Introduction

1.1. Constructing Music

The folk and children songs on which our popular music is based contain recurring short elements of pitch-time relation, which we have learned to seek out and reconstruct as we listen to music. We categorize some of them as *phrases* – sequences of musical events that have temporal balance, a sense of wholeness, and often, exhibit transition to either tension or stability, especially when following or preceding a phrase that exhibit the other [1].

Indeed, composers think in phrases. When composing lyrics, they will often match a musical phrase to a lyrical one. When doing so, they will also draw recognized elements of pitch-time relation from the pool of culturally shared norms and modify them in order to create work that is original on one hand, but also culturally acceptable on the other.

In the eighteenth century, the idea that original music can be created through a process of combining short, pre-existing pieces of music was formalized in the shape of *Musikalisches Würfelspiel* - musical dice games. The creators of these games composed a series of measures that could be played sequentially when arranged in each of many different ways while maintaining a musically plausible result. The player would create a piece by arranging the measures according to a dice throw, following a set of rules dictated by the composer. Versions of the *Musikalisches Würfelspiel* were created by composers such as Haydn, Mozart and C. P. E. Bach; however, the composer who is acknowledged to have created the first one was Johann Philipp Kirnberger (1721-1783) [2].

Other techniques for musical invention developed by Kirnberger demonstrate further formulation of manipulating existing music as a source for inspiration and the derivation of new musical material. For example, one of Kirnberger's techniques suggested that a new sonata can be composed by using a bass part from another sonata, then composing a melody on top of

that bass and finally replacing the bass part with a new part composed to accompany the new melody.

From observing these techniques it is possible to deduce that reuse and modification of previously heard music in the shape of paraphrase and quotation is a common practice in music composition, if not a key ingredient.

Nowadays, when turning to computers to help us automate complex musical procedures, we want to formulate the manner in which paraphrases and quotations are manipulated and put into context, and we aspire to communicate our wills to the computer as to how variations should be made.

1.2. Music, Time and Space

“All art constantly aspires towards the condition of music”, said the English writer Walter Pater, referring to music being the only art in which the subject and the form are one and the same. Music is an outstanding art form in other senses, as well; particularly in its relations with space and time: space is not required for music, only time is. Music cannot exist without time, but it can exist without space.

Time is more stubborn than space. We can move in space as we wish, but we cannot move backwards and forwards in time; we can represent time with space – as we do with our clocks, our graphs and our calendars, but we cannot represent space with time. It is highly possible that the obscurity of music, an art form that resides completely in the reluctant domain of time, has influenced thinkers such as Pater and Arthur Schopenhauer to crown music as the highest form of art. Perhaps it is this very property that empowers music to tell a story that tells itself.

In his momentous paper “Music, Mind and Meaning”, Marvin Minsky suggests that while toys help children learn about spatial properties, music teaches us about temporal properties [3]. Once again, space lends itself to more possibilities – a child can make a fort with blocks, revisit the design, make adjustments and use the contraption as a fort in an iterative process. The child *plays* with the blocks. Is this form of playing ever translated to a child’s engagement with

music? What opportunities and means does a child have to assemble, disassemble and reassemble music, and play *with* the music?

Music notation is a representation in space of art that resides in time. When we design tangible music interfaces, we try to enhance notation from two dimensional drawings to three dimensional objects. Through the addition of a spatial dimension we hope to gain control over new musical dimensions, and furthermore - we want to touch time.

1.3. Tangible User Interfaces

Tangible user interfaces (TUI) have been in the center of a fast growing body of work for the past two decades. They present an alternative to the conventional input and output devices used with computers, namely the keyboard, mouse and screen presenting graphic user interface (GUI). In contrast to the mouse and keyboard, TUIs are not immediately identified as computers. Unlike conventional GUIs, they not only constitute representation of the relevant information but also the means to control it, and by doing so, they offer more resemblance to non-digital objects that surround us [4]. In addition, development of new TUIs appeal to researchers of many fields as it offers fertile ground for innovation in technology, design and human-computer interaction [5].

1.4. What is SoundStrand?

SoundStrand is a tangible user interface for music composition. It comprises a set of connectable building blocks called *cells*. Each cell represents a *motif* – a short fragment of music, one measure long. A motif comprises of a *phrase* - a monophonic melody, a chord and a bass. By connecting cells to one another and creating a *strand*, the user concatenates their motifs. The cells demonstrate three degrees of freedom; manipulating the cell in each one of them introduces variations to the motif: bending a cell changes the melodic directionality of the phrase; twisting it changes the harmonic tension; and stretching or compressing the cell changes the rhythmic distribution of the phrase's notes.

Like a *Musikalisches Würfelspiel*, with SoundStrand one composes by combining existing pieces of music. It creates music from high level instructions. Like many non-musical TUIs, SoundStrand allows the representations of both the content and how one can operate on it.

1.5. Outline

Chapter 2 will summarize the relevant work of the Opera of the Future group in the Massachusetts Institute of Technology's (MIT) Media Lab as well as my own work in the lab. SoundStrand will then be put in the context of these. It will introduce the reader to related work in the field of tangible interfaces in the use of music - a review of selected samples from the very large body of work done in the field will be presented; it will include those that present similarities to SoundStrand and help demonstrate SoundStrand's contribution. Selected works in the field of computer assisted composition will also be reviewed.

SoundStrand's mechanical, electrical and interaction design will be described in detail in chapter 3 as well as the design of the accompanying software. Chapter 0 will examine the implications of the musical paradigm that dictates SoundStrand's behavior. Particularly, it will discuss the "Coda" system that focuses on music learning from personally meaningful music. The last chapter will present a conclusion, including the evaluation of SoundStrand, and a discussion on ways to further develop, enhance and build on top of this work.

2. Background and Related Work

In this chapter, I shall review previous work done in the fields of musical TUIs and computer assisted composition. In both domains, the body of previous work is immense; therefore I shall try to examine only a few examples that are outstanding, reside in the same realm of SoundStrand and cover as many of its key aspects. At the end of this chapter, I will note how SoundStrand differs from the previous work reviewed and its unique contribution.

2.1. Opera of the Future Group

Over the years, the Opera of the Future group at the MIT Media Lab has explored innovative uses for technology in music. These range from the effects of musical engagement on physical and mental health to large scale robotic operas. One can surely see an evolutionary process reflected in the projects described here, and to which SoundStrand is a natural perpetuation.

2.1.1. Hyperinstruments

The Opera of the Future group was created early on in the MIT Media Lab's existence with Tod Machover as the principle investigator. It was initially called "Hyperinstruments", and its focus was the creation of augmented instruments. Audio, MIDI (Musical Instruments Digital Interface) and sensor data collected from these novel instruments were channeled to custom software to explore shape and create musical pieces. Initially, these instruments were to be put in the hands of virtuoso performers such as Yo-Yo Ma and Peter Gabriel [6]. Later on, the ideas and methods developed in the group were applied to create hyperinstruments aimed to be used by non-professionals. This path led to the inception of the Brain Opera.

2.1.2. The Brain Opera

The Brain Opera (1996) was an interactive, large scale music experience. It consisted of two sections – in the first section, the audience experimented with a variety of hyperinstruments [7].

Audio segments produced during this exploration were reordered in order to be used in the second section, in which three trained musicians performed the piece using some additional novel, electronic instruments, creating an experience which is unique every time the piece is performed. Largely inspired by Marvin Minsky's book "The Society of Mind", the Brain Opera utilized the audience as a collective of agents working together to create a musical piece as an analogy to the fashion in which collectives of brain processes in the human mind collaborate to form consciousness [8].



Figure 2-1: A performance of the Brain Opera [7]

The instruments featured in the Brain Opera included the *Rhythm Tree*, which allowed the triggering of samples using touch-sensitive pads; the *Gesture Wall*, which produced music and displayed animated graphics in response to audience members' physical movements; the *Singing Tree*, which produced musical accompaniment to audience members' singing; and the *Digital Baton*, which used measurements of an infrared LED placed at its tip, force-sensitive resistor strips mounted along the performer's grip, and a set of accelerometers, to control several musical parameters in various ways during the performance section.

2.1.3. Toy Symphony

The next big project of the Opera of the Future group was the Toy Symphony. Recognizing that these technologies and ideas could be put in the hands of children, the group set out to bring

children and professional musicians closer, as well as to redefine the relationship between the performer and the audience [9]. The project became an endeavor to develop musical software, designed to be used by children. It culminated in a series of concerts, in which the children performed side by side with professional orchestras and soloists using both traditional instruments and novel electronic ones [10]. Preceding each concert the children would engage in a week-long workshop in which they were introduced to the instruments as well as to musical thinking, composition and performance.



Figure 2-2: Young performers playing Music Shapers alongside a symphonic orchestra in a Toy Symphony [9]

The instruments used in the Toy Symphony enabled children to control aspects of the musical piece in an intuitive yet sophisticated way after extremely short training. The *Beatbug* is a hand-held, bug-like instrument that allowed the player to tap a beat on its “back” and modify the recorded beat’s timbre, pitch and phrasing by spatially maneuvering the instrument and manipulating its “tentacles”. A network of Beatbugs could be formed and beats could be transferred from one player’s Beatbug to another’s, where it can be further manipulated, [11]. *Music Shapers* were soft, squeezable instruments that contained pressure and capacity sensors [12]. The data produced by the measurement of these sensors was formatted as MIDI messages that were used to control the musical phrases’ contour, timbre, density and structure.

2.1.4. The Big Thing

Another instrument that was designed for the Toy Symphony was the Big Thing [13]. It was intended to be a gigantic contraption that functions as both a composition environment and a performance instrument. The Big Thing was never completed due to the restriction its physical design imposed on its musical flexibility. Moreover, the dual nature of the instrument's functionality was a source of confusion in the process of envisioning the instrument's interaction model.



Figure 2-3: Five foot model of Big Thing [13]

2.1.5. Hyperscore

With the intention to implement the music composition ideas expressed in the Big Thing while breaking free from the limitations exhibited by its physicality, *Hyperscore* was created.

Hyperscore is a music composition software developed by Mary Farbood and other members of the Opera of the Future group [14]. Three types of editing windows are available: in the *Melody Window*, the user creates short melodies, or *motives*, much in the style of a piano-roll editor present in most digital audio workstations. The user can create many of these melodies and

assign a color to each of them. The *Percussion Window* is very similar to the melody window. However, it is designed to facilitate the creation and editing of short drum beats and percussion rhythmic patterns. Finally, the *Sketch Window* allows the user to combine the motives and drumbeats to a composition by painting lines with their corresponding colors. Different attributes can be assigned to each line such as timbre and volume.

In addition, a *Harmony Line* is present in the Sketch Window, and it is used to describe harmonic tension in the piece. Moderate slopes of the line yield transitions through suspense to relief within the current key, while steep slopes result in a chord progression that leads to a key change. The Sketch window offers three modes of harmonic interpretation to the music material:

- “None”: The harmony algorithm is inactive, and the Harmony Line has no influence.
- “General”: The motives’ notes are altered to fit the current scale to reduce the level of dissonance. The tonal center can be altered using the Harmony Line.
- “Classical”: Adds a higher degree of harmonic control when manipulating the Harmony Line.

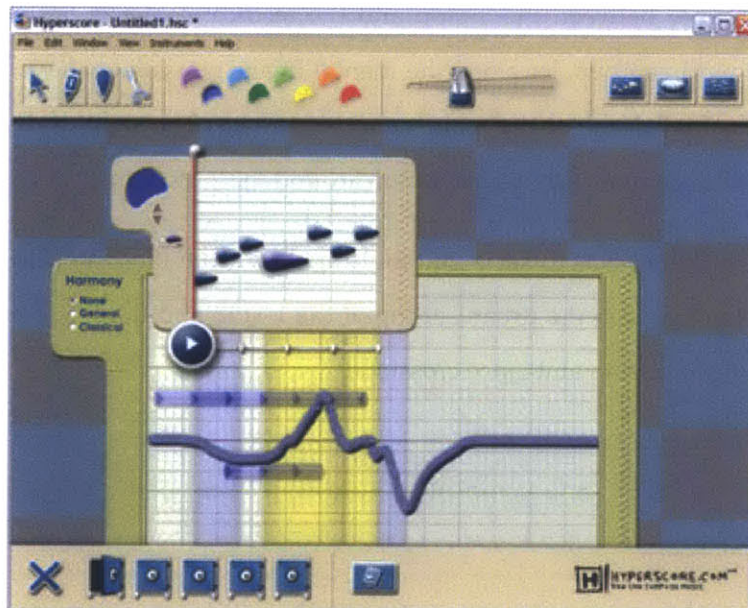


Figure 2-4: Hyperscore project screenshot [15]

2.2. Towards SoundStrand

My work has almost always revolved around music – whether as a professional musician, as a music technology educator or as an engineer developing music information retrieval algorithms. During the period in which I worked in the Opera of the Future group in the MIT Media Lab, I had the opportunity to explore new forms of connections between music, technology, education and physicality, a journey that resulted in the conception of SoundStrand. This section presents projects that signify important milestones in my exploration.

2.2.1. Chroma District

Chroma District was an installation commissioned by MIT for the Festival of Arts, Science and Technology (FAST), held to celebrate MIT's 150th anniversary [16]. Built in collaboration with Akito Van Troyer, it was designed as an interactive experience for people walking from the MIT subway stop towards the MIT campus. It was conceived from the realization that visitors to MIT who arrive on the subway train station, known as “the T-stop”, do not enter campus through its main entrance, but rather through the medical center. The experience is quite surprising – the scenery between the T-stop and the medical center is fairly urban and very different from a typical campus, let alone the mental image newcomers have, which normally includes the famous MIT dome or the main campus entrance. The sight of disoriented visitors asking for directions to campus is not uncommon around the T-stop area.

The objective of the project was to set a guiding trajectory from the T-stop to the campus and smooth the transition between the urban setting of the T-stop and the academic atmosphere of the MIT campus by creating an ambiance crossfade.



Figure 2-5: Chroma District

A set of some 30 lanterns were built, each equipped with the ability to project audio and high luminance light, sense proximity using an ultrasonic sensor, read audio files from a memory card and communicate over radio with neighboring lanterns. Memory cards containing a bank of sounds were mounted in the built-in memory card reader. The sound files were 30 seconds long, and each contained a mixture of digital sounds and processed sound samples recorded at the subway, the subway station and the MIT campus. Each lantern was assigned with a unique lighting pattern and a unique sound file.

The lanterns were to be hung between two rows of trees on the route from the T-stop to the entrance of the MIT medical center. In their idle state, each lantern was to produce its own light pattern and play its own sound file, however the light was to be dim and the sound was to be soft. Once a visitor would approach a lantern to observe and listen more closely, the lantern would reward her by increasing in brightness and playing the sound louder. Moreover, after a few seconds, the entire installation would “celebrate” the interaction between the spectator and the lantern by passing the light pattern and sound segment from one lantern to its neighbor, propagating them towards campus, and by doing so, directing the spectator to MIT.

Unfortunately, only few weeks before installing Chroma District, the trees on which the lanterns were to be hung were cut down, leaving no reasonable alternative in that area. As a last

resort, a small seating area near the David H. Koch Institute for Integrative Cancer Research was selected as an alternative location. However, the narrative of the piece was drastically compromised. Further anguish came from a last-minute electronic malfunction that left the lanterns crippled, having their capabilities restricted to mostly lighting and detecting human proximity – to some degree.

2.2.2. Cicadence

Cicadence, the product of a collaboration with architect Catherine Winfield, is an auditory sculpture presented in FIGMENT Boston 2011, a festival for public art [17]. The motivation for this project was to build a sound sculpture through a biomimetic process. The sound-making organ of the cicada was used as inspiration to this project due to its interesting materiality and mechanical behavior.

A loud, chirping noise is produced by the male cicada as part of the courting ritual. To produce this noise, the cicada uses an organ called the *tymbal*. It is a hollow chamber located on both sides of the cicada's abdomen. The exterior of the tymbal is the *tymbal plate*, which is a thin, flexible membrane. Portions of hardened tissue, called *ribs*, are embedded in the membrane. There are some 2500 species of cicadas, and the number of ribs varies from species to species. As the cicada rapidly contracts and extends a muscle connected to the tymbal, the tymbal moves back and forth, causing the ribs to buckle in and out. The buckling of the ribs that is the source of the sound produced in the cicada's song. A pocket of air, contained between the tymbal and the cicada's abdomens, serves as a resonance chamber [18].

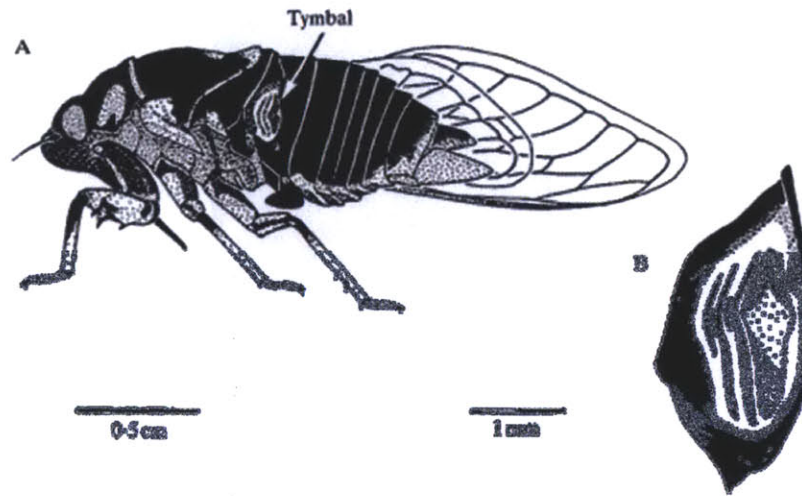


Figure 2-6: (a) Lateral view of a male okanagana vanduzeei cicada. (b) The tymbal. [19]

Cicadence explored this mechanism from several perspectives – buckling of a material as a sound production source; incorporation of a hard, buckling material in a soft flexible material; and geometrical properties derived from the tymbal. The result was a sculpture that is divided into four sections, separated by wooden ribs. Each area produced sound through a different mechanism actuated by the spectator. The first section, the “Accordion”, was a grid of interwoven sheets of metal that could be expanded when the spectator pulled a rope connected to it. As the sheets pulled each other in different directions, various sounds were produced as a result of friction, collision and buckling. The second section, the “Maracas”, was a grid of rattlers, made of polyethylene terephthalate (PETG), and filled with small metal balls. The third section, the “Sheet”, was made of sheets of metal that were folded and cut in a repetitive geometric pattern, producing sound by pushing and hitting the sheet, causing the sheet to buckle and vibrate in various manners. Finally, the fourth segment, the “Clicker”, was made of a sheet of plastic in which small metal clickers, taken from commercially available dog clickers¹, were embedded. As a spectator pushed against the plastic sheet, the clickers would buckle and produce a clicking sound.

¹ Dog clickers are used for dog training. The clicking sound helps dogs identify desired behaviors.



Figure 2-7: Cicadence

2.2.3. SoundStrand's conception

Arriving to the Media Lab, my vision was to build a physical synthesizer – a collection of mechanical, computer control contraptions that act as modules of an analog synthesizer. When working together – connected to each other, encapsulating one another or acting on one another in space – they would become a configurable musical instrument. Chroma District, in this context, was an abstract exploration of the connections between participant, sound, objects and space. Cicadence was a more concrete effort towards the realization of a physical synthesizer. It was a study of sound production from objects' materiality and formation.

SoundStrand itself started as an endeavor towards a physical synthesizer. In fact, it was an attempt to define the conceptual interface of a digital simulation of that synthesizer. However, ideas and inspiration from previous projects as well as from the Opera of the Future group's legacy changed SoundStrand's designation: from Chroma District, I was motivated to think about creating music with objects that represent musical fragments; building Cicadence and observing spectators interact with it, I was inspired to think about how people can touch music; Hyperscore showed me that given the right technology, one can, and often wants to, engage in music creation and produce meaningful results even without musical training; finally, the Big

Thing, or better yet, the lack of its realization, convinced me that music composition with tangible interfaces is a worthwhile challenge.

2.3. Musical TUIs

Musical TUIs, like their non-musical counterparts, represent data and the means to manipulate it. In the case of musical TUIs, the data is music. This observation is important when making the distinction between musical TUIs and electronic musical controllers and instruments, which do not represent the musical content by their physicality.

The development, research and design of musical TUIs draw a large community of musicians, engineers and designers, and the body of work in the field is vast. This section brings examples of musical TUIs that have certain similarity to SoundStrand. These TUIs are toys, sequencers, a set of similar, attachable objects or some combination of the above.

2.3.1. AudioCubes

AudioCubes uses a collection of cubical tangible objects as a representation of modular synthesis building blocks [20]. Placing a cube on the table in different orientations changes its state. A “drum loop” cube, for instance, selects one of four prerecorded drum loops. Cubes communicate through infrared LEDs and sensors on two dedicated sides.



Figure 2-8: AudioCubes explored by a museum visitor [20]

2.3.2. SoundBlocks

Developed in the Music, Mind and Machine group in the MIT Media Lab, SoundBlocks is a tangible environment for digital sound generation and manipulation [21]. SoundBlocks is comprised of a set of 14 blocks, each representing a basic function in digital sound production, such as a microphone input, a delay and a sample player. Blocks can be connected to each other using rods, to represent audio and data flow between the blocks. Each block has one output, between 0 and 4 inputs and an RGB-LED (red, green and blue light emitting diode). Additional control over the blocks is achieved by interacting with sensors such as buttons and knobs embedded in some of the blocks. In a sense, SoundBlocks can be viewed as a transformation of digital sound production systems such as Max/MSP or Pure Data into the tangible realm².



Figure 2-9: SoundBlocks [21]

2.3.3. Music Blocks

Music Blocks [22] is a music toy available commercially which consists of a base unit and a set of cubical blocks representing musical phrases. The blocks can fit into five positions in the base unit to create a sequence, as the blocks are being played from left to right. Variations on the musical phrases, normally of timbre, are introduced when the orientation of the blocks is changed. Various additional cartridges contain sets of phrases that are used to create simple classical tunes.

² Max/MSP [54] and Pure Data [55] are visual programming language for music and multimedia, developed by Miller Puckette.



Figure 2-10: Music Blocks (image from Amazon.com)

2.3.4. The Tangible Sequencer

The Tangible Sequencer is a set of cubical blocks, each containing a sound [23]. A triangular button is located on top of the block, and when it is pressed, the sound is played. If the triangle is pointing at another block that is placed in proximity, its sound will be played subsequently, and so forth. Complementary computer software allows the user to “drag-and-drop” sound files into the blocks.

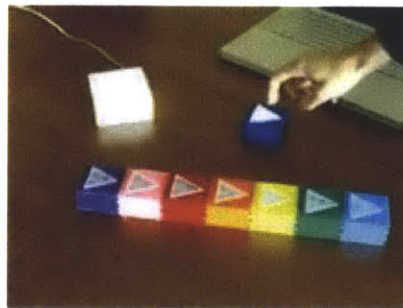


Figure 2-11: The Tangible Sequencer [23]

2.3.5. The Music Table

The Music Table is an augmented reality music composition system [24]. The user arranges cards on a table to create musical phrases. Each card represents a note, with the horizontal and vertical position of the card representing the timing and pitch of the note, respectively. Tracking the positions of the cards is achieved with an overhead camera connected to a computer. The computer is also connected to a large monitor on which the visuals captured by the camera are displayed. The computer software adds computer generated images to the captured image to provide feedback to the user. These take the shape of animated creatures, whose features vary

in relation to the different properties of the notes, such as length and loudness. Other cards allow the user to copy an entire phrase into one card, edit the phrase and change instruments.

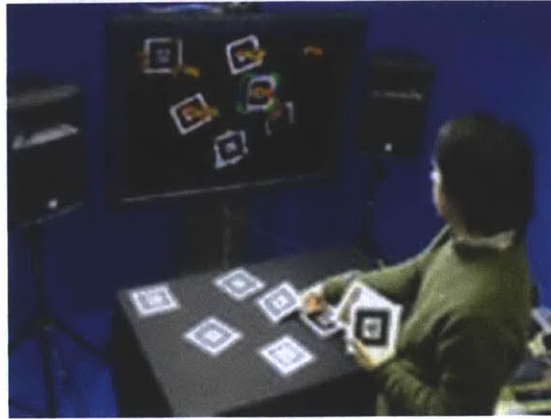


Figure 2-12: The Music Table [24]

2.3.6. The reacTable

The reacTable is a multi-user musical instrument with a tabletop TUI [25]. It is a translucent table on which small objects are placed. A camera, located under the surface of the table and connected to a computer, captures the location of the objects. The system utilizes an analog synthesis paradigm, with the object representing common analog synthesis modules such as oscillators, filters and envelopes. The orientation of the objects and their proximity to each other determine various aspects of interaction between the objects, such as the amount of modulation, cutoff frequency and so forth. A projector, also located underneath the surface of the table, projects computer generated animation that visualizes the connections between the modules.

The technology that lies at the heart of the reacTable is called “reacTIVision”; it enables the computer to analyze, through the image captured by the camera, the location and orientation of every object on the table [26]. Although the reacTable is available as a commercial product, reacTIVision is available as an open-source toolkit and has been used in numerous projects, many of them musical in nature.



Figure 2-13: The reacTable [27]

2.3.7. Sifteo Cubes

A Sifteo Cube is a 1.5" X 1.5" block that features a clickable liquid crystal display (LCD) screen, radio communication hardware and a variety of motion detectors. Developed in the MIT Media Lab under the name "Siftables", the Sifteo Cubes are described as a Sensor Network User Interface (SNUI) – a collection of small physical manipulatives that have sensing, wireless communication and user-directed output capabilities [28].



Figure 2-14: Sifteo Cubes [29]

Sifteo Cubes can be loaded with a variety of applications often offered by a third party. Some of these applications are musical – "Plane of Tune", for example, turns every cube into a small loop sequencer that is responsible for one part of the tune, while all the cubes play in sync.

Notes are inserted into the loop by tiling and shaking the Cube. While most of the commercially available applications, including the music oriented ones, are in fact games, early demo versions of Siftables featured a more professional-looking music sequencer: some cubes acted as *sequences*, capable of holding sequence information. These were aligned together to create a musical timeline. Some cubes acted as *voices*, featuring four possible loops for each Cube that could be inserted into a sequence block by bumping the voice cube against the sequence cube. Other cubes acted as commands, changing the tempo, volume or applying effects.

2.4. Computer Aided Composition Software

Recognizing that the software engine that translates the user's interaction with SoundStrand into music is, in fact, a computer aided composition software, this section brings selected works from the field. It should be noted that the term "computer aided composition" is very general and can include a wide variety of music software, such as digital audio workstations (DAWs), music and audio-oriented programming languages and musical computer games. In selecting the works that are the most closely related to SoundStrand, I searched for several different environments that are directed towards composition, that are music education oriented or that use the manipulation of pre-composed music as a significant tool.

2.4.1. Impromptu

Impromptu is an interactive teaching tool designed by Jean Bamberger. It also provides a fairly conclusive view on Bamberger's work and is an important, early milestone in achieving her vision of a musical LOGO³ programming language [30]

The graphical interface of Impromptu allows the user to create tunes through the assembly of sequences of *blocks*. Three types of blocks are available: *Tuneblocks* are melodic segments, such as phrases or motives, following Bamberger's concept of *Simples* [1]; *Drumblocks* are rhythmic

³ LOGO is a programming language developed by Seymour Papert for the purpose of teaching children mathematics [39].

patterns played on a single percussion timbre; and *Harmonyblocks* consist of chords that can be played to accompany the melody. A sequence of blocks from a certain type can be combined and encapsulated into a new *Superblock*.

Impromptu was designed to accompany Bamberger's book "Developing Musical Intuitions" [31]. The book included activities that were aimed to introduce children and non-musicians to music. However, while conventional music education normally starts with low level properties such as pitches and durations, the activities offered in the book uses higher level music entities, *Simples*, as the entry point to musical understanding. Learners were instructed to assemble known tunes from carefully designed banks of *Tuneblocks* and explore notions such as repetition, tension and balance.

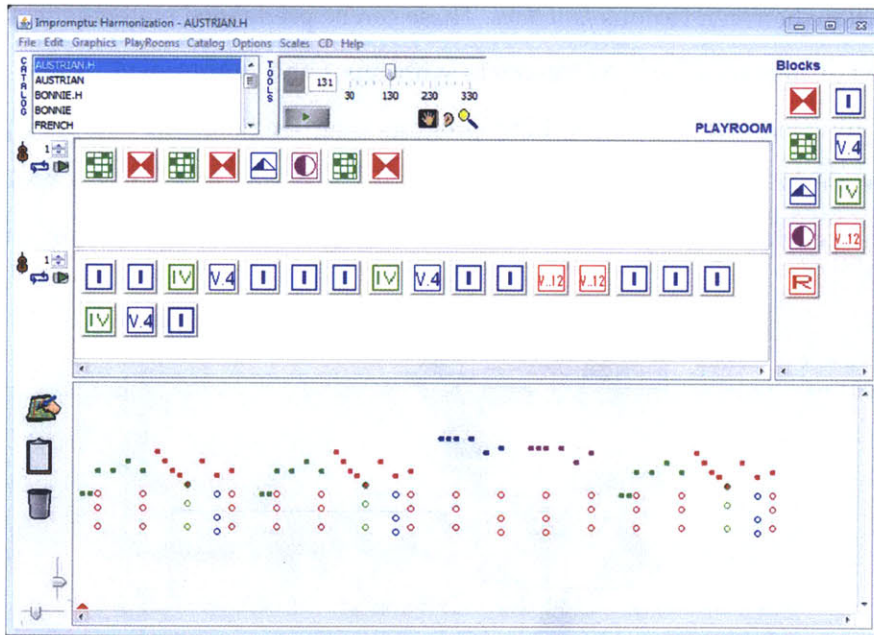


Figure 2-15: Impromptu screenshot

Impromptu also featured lower level functionality as it allowed users to create their own *Tuneblocks*. This was done when the user entered two sequences of integers, with the number of integers corresponding to the number of desired notes in the block – the first sequence represented pitches and the second represented durations. Furthermore, the pitch numbers entered by the user specified the pitch's place in order within a given scale, and the scale itself

could be edited by the user. This is to show that although designed for beginners, Impromptu featured relatively powerful tools for personal creation, tinkering, modifying the system's behavior and applying more advanced knowledge.

2.4.2. OpenMusic

OpenMusic, developed at the *Institut de Recherche et Coordination Acoustique/Musique (IRCAM)*, is an object-oriented visual programming interface for computer assisted composition. To a large extent, it could be viewed as a manifestation of a musical LOGO language as envisioned by Bamberger, even if this was not the designers' intention.

The fundamental building block of OpenMusic is the *patch*, in which the user graphically controls data flow between various *functions*. These can be predefined mathematical operations, manipulations of musical elements, interfaces with input and output devices and so forth, or user-defined functions written in the LISP programming language. In addition, patches can contain other patches. A special purpose patch is the *maquette*, which has a horizontal time dimension and provides high level control over the arrangement of the patches in time [32].

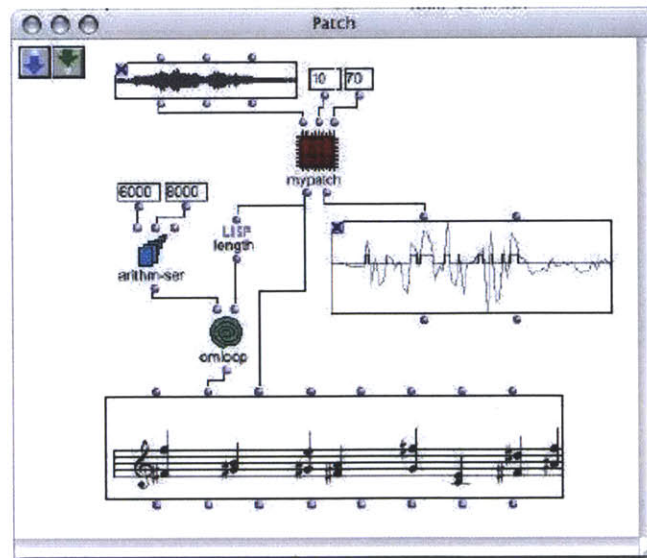


Figure 2-16: Screenshot of an OpenMusic patch [33]

2.4.3. Musique Lab 2

Developed on top of OpenMusic, Musique Lab 2 (ML2) is an educational application designed to empower teachers to develop new pedagogical approaches towards music education. One of the three modules that constitute ML2 is “ML-Maquette”, in which the user composes music through high level manipulation of musical material [34]. *Objects* are musical entities such as chords, note sequences and rhythmic sequences. These can be manipulated by a set of *operators*, which perform transformations on objects to create new objects. Operators can perform rhythmic quantification, create arpeggios, generate a melody with contrary movement, etc. Finally, the *maquette* area is a sequencer for objects that not only allows the user to arrange the objects in time, but also dictate a dynamic behavior that parametrically arranges the objects in time.

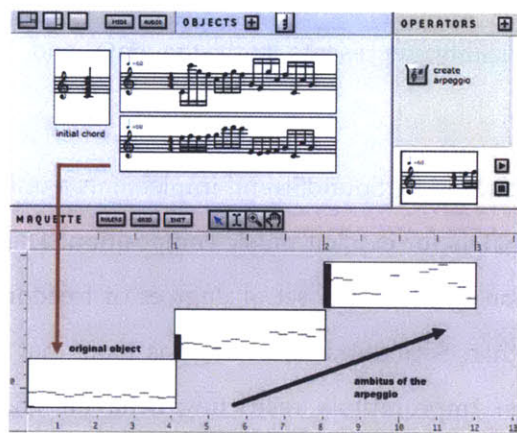


Figure 2-17: Screenshot of Musique Lab 2 [34]

2.5. Contribution and Distinction from Related Work

Examining related work in the field of TUIs for music creation suggests that explicit composition of music through the use of TUIs is a particularly hard task. Most of the previous work with TUIs in the domain of musical composition is confined to sequencing of samples, such as with Music Blocks and the Tangible Sequencer, or the creation and manipulation of a soundscape, such as with AudioCubes and the reacTable. I assert that a conflict rises between the great freedom and fine resolution that exist when one composes music and the coarse

resolution and limited number of degrees of freedom that one is presented with when interacting with a TUI. SoundStrand attempts to resolve that conflict by introducing a large number of musical variations accessed by scanning very few degrees of freedom.

An exception to most existing musical TUIs is the Music Table, in which explicit composition is indeed achieved. However, in contrast to the Music Table and, in fact, to all other TUIs described above, a musical piece emerges in SoundStrand when objects representing musical fragments are attached to each other, forming a new, singular and unique object that is a physical representation of the piece. This aspect alone presents a new, intriguing perspective on the objectification of music. Moreover, in the Music Table, manipulation of the musical material is not done by physically manipulating the representing object, but rather by introducing a dedicated mediating object. In my view, this misses the point of using a TUI, as it loosens the mental coupling of the data with the interface. With SoundStrand, I introduce a musical interface that by its physicality represents both the data and the ways in which it is manipulated.

In addition to its physical conception, SoundStrand implements a unique musical approach that strives to enable the use of TUIs for explicit music composition. The paradigm of composition through phrase manipulation by a limited set of degrees of freedom is general enough to be applicable for TUIs other than SoundStrand, as well as tools that feature software alone. A notable gap exists between Impromptu's restricting behavior and ML2's complexity. The musical paradigm that governs SoundStrand attempts to fill this gap with a set of simple and immediately available operations that provide substantial musical freedom.

Finally, I would like to stress that composition with tangible interfaces can enable paradigm shifts in collaborative composition, performance and music education.

3. Designing SoundStrand



Figure 3-1: A SoundStrand with five cells

3.1. System Overview

The SoundStrand system is divided into three main parts – the SoundStrand interface; the SoundStrand software running on a computer; and the content generation software tools. The interface, shown in Figure 3-1, is a set of cells, which are cylindrical objects, roughly the size of a soda can. The cell features a 3D-printed skeleton that allows the cell to be manipulated in three degrees of freedom; electronic circuitry, that handles the measurement of the cell’s configuration and data exchange with the computer and the other cells; visual feedback in the form of a light emitting diode (LED); and a skin that covers the cell and diffracts the light emitted from within the cell. Cells connect to each other sequentially with connectors built into their circular surface. Electronic connectors are also located on the circular surface to allow communication between the cells.

The SoundStrand software designates musical material to each cell based on stored Extensible Markup Language (XML) content files. It receives the data transmitted by the SoundStrand interface and determines the strand’s configuration. Based on that configuration, the software applies a series of algorithms to the music material in order to modify the music in accordance to the user’s intentions.

Finally, the content generation software tools are independent computer programs that assist the user in creating content files to be read by the SoundStrand software and determine its behavior.

3.2. Musical Behavior

Each SoundStrand cell carries an ID which defines its *cell-type*. Every cell-type is coupled with a basic phrase from which variations are derived. This allows users to think about their music in a structural, figural way, recognizing that repetition, structural symmetry and motif development are central to our perception of music [1] [2] [3].

3.2.1. Mapping the Physical to the Musical

Assignment of the various degrees of freedom of the SoundStrand cell to the different types of phrase variation attempts to be as intuitive as possible. As cells are connected sequentially to one another, such assignment suggests that time moves along the trajectory connecting the cells, i.e. along the cells' lengths. Therefore, the variation concerning the timing of the notes – their rhythmic distribution – is mapped to physical changes along that axis, which is the cell's elongation. Pitch is commonly referred to as the axis perpendicular to time; therefore, the cell's bend is mapped to the phrase's melodic directionality. Finally, the mapping of the cell's twist to harmonic tension seems natural as the act of twisting is often paired with physical tension, such as springs or the lids of glass jars.

3.2.2. Harmony

The chord played in each SoundStrand motif is determined by the amount of its cell's twist. A cell can be twisted to five different positions, and these loosely represent harmonic tension degrees, inspired by David Cope's SPEAC system of music analysis [35]. The SPEAC system provides a level of abstraction for describing the motions of notes, harmonies and motives. These musical events are classified to one of five identifiers, to which "SPEAC" is an acronym:

- *Statement (S)*: typically occurring near the beginning of musical passages, is not result of a pre-occurring event, but rather an independent event.
- *Preparation (P)*: modifies the meaning of another event by standing ahead of it.
- *Extension (E)*: serves to lengthen a previous event
- *Antecedent (A)*: causes a significant implication and requires a resolution, typically followed by a Consequent identifier.
- *Consequent (C)*: appears as a resolution to an antecedent event, often the same chord found in a Statement identifier.

Table 3-1: An example of a Harmonic Transition Table (HTT)

Previous function	S	P	E	A	C
I	iii	IV	vi	V	I
ii	vi	IV	ii	V	I
iii	I	IV	ii	V	vi
IV	I	ii	IV	V	vi
V	iii	ii	IV	V	I
vi	I	IV	vi	V	ii

SoundStrand’s harmonic transitions system is only loosely based on the SPEAC system. In SoundStrand, for example, there are no limitations as to which two identifiers can immediately follow one another. This allows the users to implement their own high-level description of musical tension when programming the harmonic behavior themselves. The implementation of SoundStrand’s harmonic behavior lays in a Harmonic Transition Table (HTT), which determines which chord a motif will take based on its degree of tension, as expressed in the cell’s twist and on the chord of the motif preceding it. Every line in an HTT is called a *rule* and it relates to a specific chord played by the previous motif. Every column corresponds to a certain amount of cell twist.

Consider the set of rules in Table 3-1. The first thing to notice is that the naming of the tension degrees is taken directly from the SPEAC system, but the connection to the actual chord

progression is intentionally very loose. The built-in HTTs do attempt to maintain the following principles:

- *Intuitiveness*: Statements and Conclusions are usually stable chords – the tonic or a substitution of the tonic such as the third degree or the sixth degree. Antecedents are usually the dominant chord.
- *Broadness*: the same chord does not usually appear in the same rule more than once to allow as many options as possible.

However, these are merely guidelines. We shall see in 3.5.3 how new HTTs can be programmed by users to accommodate their style and subjective perception of harmonic tension, test musical ideas and apply newly gained theoretical knowledge.

3.2.3. Melody

The basic phrase associated with cell-type determines merely the intervals to be played, with middle C being a point of reference. The final phrase to be played is affected by the ending note of the preceding motif; the adjusted melodic directionality as conveyed by the cell's bend, the harmony of the motif, and the chosen melodic interpolation algorithm. As I describe this process in detail, I shall demonstrate each step using the basic phrase transcribed in Figure 3-2.



Figure 3-2: A basic phrase

First, the notes of the phrase are transposed in order to create a natural melodic flow from the preceding motif to the current one. The interval between the last note of the preceding motif and middle C is used as a base for this transposition. In our example, the basic phrase starts with a middle C; therefore, the phrase will be transposed exactly to the note ending the previous phrase. In Figure 3-3, the phrase was concatenated to a motif ending with an E and therefore transposed to begin in that same note.

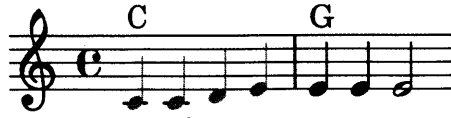


Figure 3-3: The phrase transposed to match the preceding motif which is indicated by gray note heads.

The notes are then further transposed up or down, based on the direction to which the cell is bent. Naturally, the more bend that is introduced to the cell, the more the notes are transposed. Furthermore, the amount of transposition introduced to every note is a function of its onset time in the phrase – notes that are further towards the end of the phrase are transposed more than notes that are closer to its beginning. This reinforces the sense of an adjusted melodic directionality. Figure 3-4 shows how the phrase in our example is modified by an upwards bending of the cell.



Figure 3-4: The phrase's notes transposed in response to the bending of the cell

Finally, the notes are shifted to match the harmony of the motif based on the melodic interpolation algorithm: the *Naïve* algorithm quantizes all notes to pitches that belong to the scale of the motif's chord, and then shift the first and the last notes to the closest pitches that belong to the motif's chord; the *Diatonic* algorithm acts in a similar manner, but instead of quantizing to the motif's scale, the notes are quantized to the song's scale; the *Chord Notes* algorithm quantizes all the notes of the phrase to the closest pitches that belong to the motif's chord. In Figure 3-5, the notes of the phrase were quantized according to the diatonic algorithm – all the notes of the phrase were transposed to the closest pitches that belong to the C-Major scale. The first and last notes were further transposed to pitches that belong to the G-Major chord.



Figure 3-5: The notes of the phrase are quantized to the pitches of the scale and the chord.

3.2.4. Rhythm




Every basic phrase can appear in any number of rhythmic permutations. These are selected by compressing or stretching a cell and can be programmed by using the content generation tools. For the sense of consistency and as a guideline, the built-in content files implement a principle named “rhythmical center of mass”.

Let us consider one $\frac{4}{4}$ measure and annotate its timeline as starting at time 0 and ending at time $L = 4$. The measure has n notes, with onset time t_i for every note i . The center of rhythmical mass R is:

$$R = \frac{1}{nL} \sum_{i=1}^n t_i$$

By compressing a cell, the user decreases the phrase’s center of rhythmical mass, i.e. shifts the notes towards the beginning of the measure. By stretching a cell, the notes of the phrase are shifted towards the end of the measure. Table 3-2 shows a phrase in three versions, each time with a different center of rhythmical mass.

Table 3-2: Three versions of a phrase with modified center of rhythmical mass

phrase	Center of rhythmical mass
	$\frac{1}{4 \cdot 4} (0 + 0.5 + 1 + 2) = 0.21875$
	$\frac{1}{4 \cdot 4} (0 + 1 + 2 + 3) = 0.375$
	$\frac{1}{4 \cdot 4} (1 + 2 + 2.5 + 3) = 0.53125$

3.3. Hardware

3.3.1. Skeleton

The skeleton of a SoundStrand cell is fabricated by 3D-printing. It enables the cell to be bent, twisted and elongated, and to be connected to a neighboring cell.

The centerpiece of the skeleton is the *frame*. It has a groove in which slides the *rack*. The rack can be moved back and forth to change the elongation. The frame has a niche to which the *pinion* fits. When the rack slides back and forth, the pinion turns, and by measuring a potentiometer attached to the pinion, the amount of elongation can be determined.

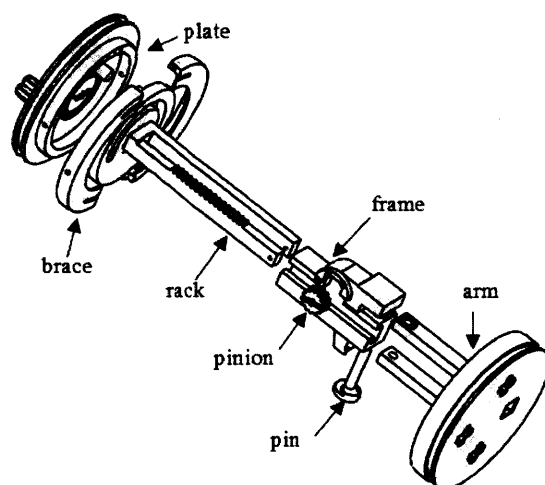


Figure 3-6: A disassembled cell skeleton

The *arm* serves as the bending mechanism of the cell. It extends from the *frame*, to which it is connected with the *pin*, allowing it to turn around the *pin*'s axis. The *pin* itself has an extrusion that fits into a niche in the *arm*, forcing them to move together. A potentiometer attached to the *pin* is used to measure the angle between the *arm* and the *frame*, determining the cell's bend.

Finally, the twisting mechanism consists of the *plate* rotating against the *rack*'s end. Two *braces* locked into each other and to the *plate* encapsulate the *rack*'s end and hold the *plate* in contact

with it. A potentiometer attached to the plate measures the amount of rotation between the plate and the rack's end in order to determine the cell's twist [36].

3.3.2. Skin

The cells are covered with an elastic fabric skin shown in Figure 3-7. The skin is sewed into a cylinder, and it keeps its shape with three plastic rings fastened to its interior. Two rubber rings are sewed to the ends of the skin and fitted to grooves on the edge of both the skeleton's ends.



Figure 3-7: A cell's skin (left) and the skin presented inside-out

3.3.3. Connectors

Three cross-shaped extrusions are located on the face of the plate. These fit into corresponding holes in the connected cell's arm's face. In addition, the electrical connectors, which are a 2x2, 2.54mm pitch header-connector pair, not only provide power and communication but also support the mechanical connection between cells.

3.3.4. Electronics

In the center of every SoundStrand cell is an Atmel ATmega168 microcontroller. It measures the value of three potentiometers fixed to the cell's skeletal structure to determine the cell's elongation, bend and rotation. The electronic circuit is shown in Figure 3-8.

Cells connect to each other with a four-pin male-female connector pair. The first cell in a strand is connected to the computer with a USB FTDI cable. This connection allows the computer to

provide 5V power and ground lines for the entire strand and receive the strand's configuration, encoded as described in 3.3.5. Two of the pins are used as a shared 5V power supply. A third pin is used to transfer data over a serial bus from a cell to the one preceding it. The fourth pin is reserved for future use.

The circuit features an RGB-LED that serves to indicate that the cell is working properly as its color is determined by the state of the potentiometers. The light is clearly visible to the user as it is diffracted by the cell's skin.

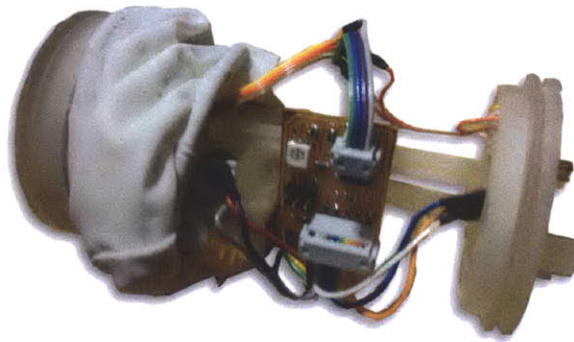


Figure 3-8: A cell's electronic circuit exposed

3.3.5. Communication

By design, data can flow between the cells in only one direction, from the end of the strand towards the computer. The last cell in the strand periodically initiates the data transfer with a packet that contains the cell's type and potentiometer values, followed by an "End of Transmission" (ETX) byte. The packet is passed to the preceding cell which adds its own type and potentiometer values to the beginning of the packet before passing it on. This process is illustrated in Figure 3-9. Working in *tail* mode, a cell assumes it is the last on in the strand unless receiving a packet in its input port. While operating in this mode it will initiate a data transfer every 250ms. Once a packet is received, the cell will no longer consider itself last and will enter *body* mode. Under this mode of operation, packets received will be promptly modified and passed on. If no packet is received for a period of 500ms, the cell will assume that

its subsequent cells have been removed and return to tail mode. The 500ms interval assures cells do not leave body mode prematurely.

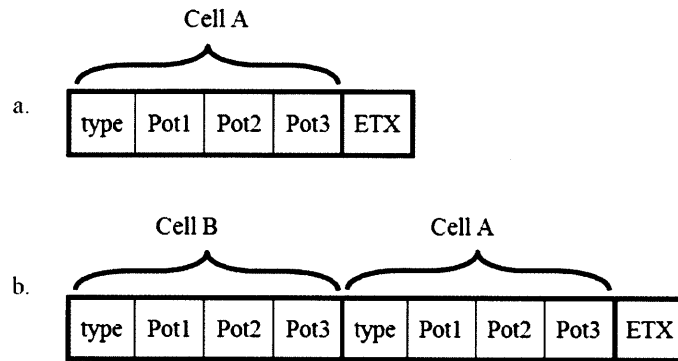


Figure 3-9: Packet formation. a) Cell A, the last in the strand, initiates a packet. b) Cell B adds its data to the packet

3.4. Software Design

3.4.1. Programming Language

The software used to translate the strand’s configuration to a musical theme is written in Java. Leading arguments for choosing Java were its object-oriented approach; its environment being platform-independent; its relatively easy interfacing with peripherals such as the serial port; and the low effort required to create GUIs.

Most popular computer operating systems, including Microsoft’s Windows 7 and Apple’s OS X, include a built-in, general MIDI software synthesizer as well as a built-in MIDI sequencer. The SoundStrand software utilizes the sequencer and synthesizer for its sound production and real-time sequencing needs. Interfacing the synthesizer and sequencer proved to be extremely easy when using Java, a fact that considerably reduced the effort needed to provide SoundStrand with sound production and sequencing capabilities.

3.4.2. System overview

Figure 3-10 shows a block diagram of the SoundStrand software. Two input methods are available in the system - manipulating a SoundStrand tangible interface, in which case the *serial*

port adapter decodes the data received from the serial port; and using the GUI, in which case the *GUI simulator* replaces the role of the serial port adapter. The *strand manager* is a module that manages a set of *motif* modules and sets their parameters based on the data received by one of the input modules. A motif module is aware of its cell's configuration and holds all the musical material associated with its cell.

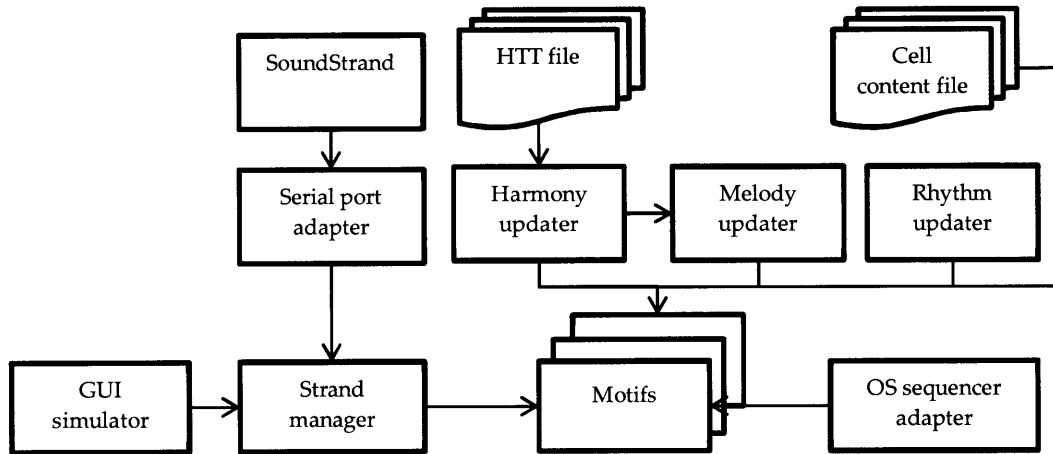


Figure 3-10: SoundStrand software system diagram

Melody, harmony and rhythm *updaters*, discussed in depth in 3.4.3, act on the motifs whenever the configuration of the strand changes. They use the physical configuration and musical data stored in the motifs to create the MIDI data of the theme and store it back into the motifs. *Cell content files* are loaded into the cells upon the user's request to replace the phrase and rhythmic permutation of the cells, and *HTT files* are loaded to alter the strand's harmonic behavior. Finally, the *OS (operating system) sequencer adapter* manages the transfer of the MIDI data stored in the cells to the operation system's built-in sequencer.

3.4.3. Updaters

The software mechanisms that interpret SoundStrand's temporal physical configuration into variations of musical motifs are called *updaters*. There are three types of updaters: melody, harmony and rhythm. However, as explained in 3.2, there are different ways to interpret operations, such as a shift of the rhythmic center of mass or an alteration in the melodic direction. Therefore, each type of updater can be implemented as a family of updaters.

It was required that different updaters would be interchangeable at runtime, allowing the user to decide, for example, whether the melody should be quantized in *Diatonic* mode or in *Naïve* mode by selecting the appropriate melody updater. The importance of this requirement comes from three different standpoints: from a program developer's standpoint, updaters should be easily tested, compared and configured; from a composer's point of view, different updaters present different aesthetic options and different levels of musical complexity; finally, from a learner's point of view, different updaters present different constraints on the musical permutations that arise from manipulating SoundStrand's cells, therefore allowing interaction with graduated levels of musical skill.

The manner in which the updaters were implemented is known in the software engineering jargon as the *Visitor* design pattern [37]. It allows the motif object to interact with a certain type of updater, while being aware of the exact implementation of the updater only in runtime. It also allows the updater to maintain its inner state as it sequentially operates on different motifs.

3.4.4. Content files

HTT files and cell content files are stored as XML documents. This allows:

- Use of existing libraries to read and write content files
- Readability of the files and ease of editing in any text editor
- Ease of creating tools for content generation.

The XML tree structures for the HTT files and the cell content files are depicted in Figure 3-11 and Figure 3-12 respectively.

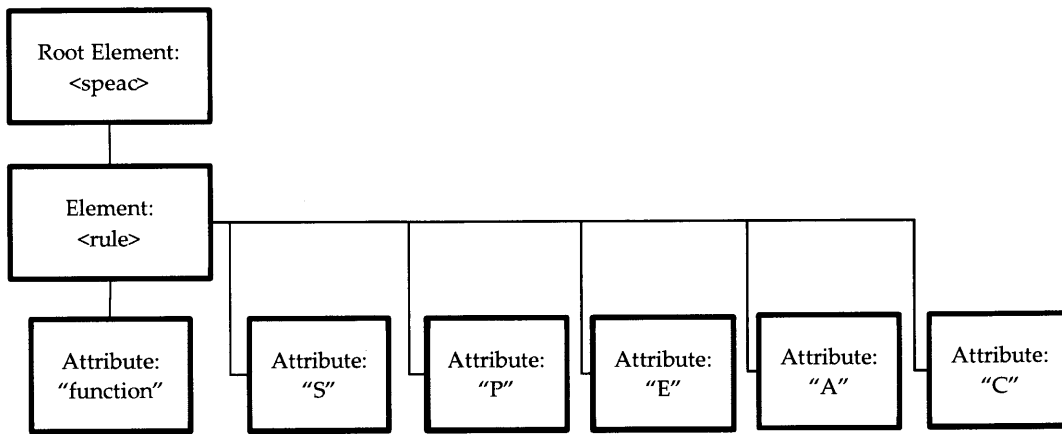


Figure 3-11: XML tree for HTT files

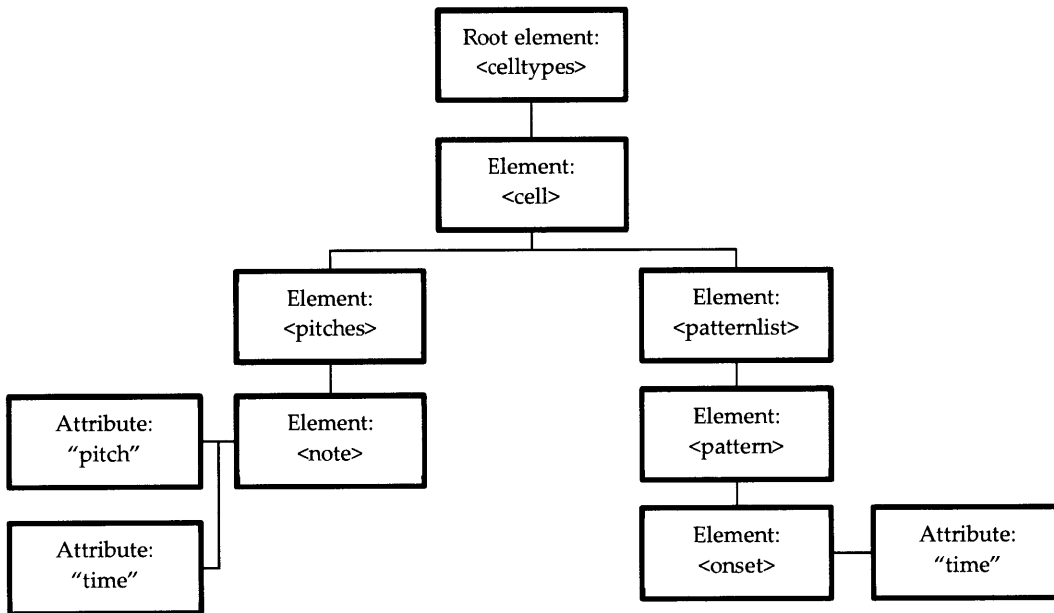


Figure 3-12: XML tree for cell content files

3.4.5. User interaction

The SoundStrand software’s GUI, shown in Figure 3-13, allows the users to load cell content files and HTT files, and choose between the various melodic, harmonic and rhythmic updaters.

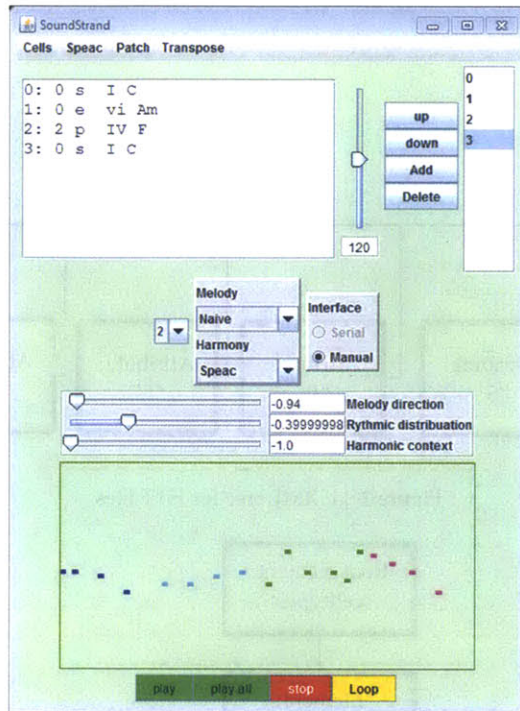


Figure 3-13: A screenshot of the SoundStrand software

The software also features a transport bar that allows playing, stopping and looping the entire strand or a particular cell, as well as a piano-roll style visual representation of the theme.

Finally, the software allows the user to enter a simulation mode, in which connection to a physical interface is not necessary, and cells are added, deleted and manipulated virtually through various controls in the GUI.

3.5. Content Generation Tools

To achieve even greater flexibility using SoundStrand, advanced users can generate new content using two content generation tools. These define the melodic content, the rhythmic permutations and the harmonic behavior of SoundStrand.

3.5.1. Motivation

The ability to create one's own content files is a key feature in the SoundStrand system. In the most simplistic perspective, it allows one to create new possibilities whenever the system is

limited. For example, if a user is trying to produce a specific phrase and is unable to do so, he can program a new cell-type using the cell editor. In that sense, user-generated cells reflect the user's personal style. The user's skills are also reflected in his cell-types and HTTs. More complex cell-types and bold harmonic transitions are an expression of knowledge, confidence and a sense of self-critique.

For a musician experimenting with musical concepts such as motifs and variations, harmonic progressions and figural structure, whether it is for learning or for creating, SoundStrand acts as "an object to think with" [38]: it embodies ideas, supports strong connections between the user and her ideas and evokes creativity and learning through their objectification and exploration through use of the object.

From a learner's point of view, moving from using SoundStrand with pre-made content to creating personalized content is the step from concrete, combinatorial thinking about one's music to formal, self-referential thinking about music. On the other hand, applying newly programmed content to musical pieces with SoundStrand serves as a mechanism to concretize formal music knowledge. In this sense, SoundStrand is an example of digital systems' ability to support dialogue between the formal and the concrete, as well as dialogue between the general and the personal as a means for the internalization of knowledge [39].

The observation between a learner and a composer is done here in a very artificial manner and only to stress different uses and benefits of a content generation toolkit. In practice, composers never cease to be learners, and allowing learners to be composers is at the very heart of the SoundStrand project.

It is intended that the content files themselves become a meaningful creation, signifying the user's musical knowledge, style and skill; a creation to take pride in and share, in hopes that it will be used by peers in the community for their processes of creation and learning.

3.5.2. Cell Editor

The Cell Editor allows the user to program cell-specific properties. The user can create new cell-types, edit them and save a collection of cells-types as a *set*. Saved sets can be loaded for further editing.

Once a cell-type is selected or created, the user modifies its properties in two areas of the window. The *pitch area* defines the number of notes present in the cell-type and their pitches at stable state. This area is a piano-roll-like editor, and although the timing of the events is also expressed in this editor, it is merely for reference and for the initial setting of a new rhythmic pattern. The actual timing is specified in the *rhythm area*.

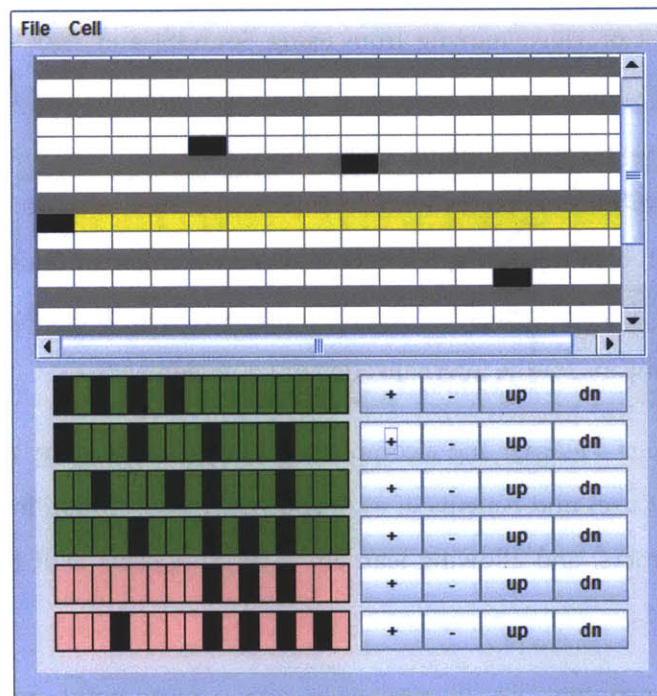


Figure 3-14: A screenshot of the Cell Editor, showing the pitch area on the top and the rhythm area on the bottom. The red patterns are illegal as they have an incorrect number of events.

The rhythm area allows the user to create the various rhythmic patterns accessible through modifying the elongation property. The pattern editing interface is a row of boxes representing the 1/16th note intervals in the measure. The user edits a pattern by marking the desired onset

times. When a pattern has the same number of onsets as the number of events in the pitch area, the pattern is considered legal and it is painted green. Otherwise, it is painted red and the user will not be able to save the collection.

The user can add, delete and change the order of the patterns. This will be very significant when later using these cell-types with SoundStrand – when changing the cell’s elongation property from fully compressed to fully stretched, this is the order in which patterns will be selected.

3.5.3. Harmony Transition Table Editor

The HTT Editor is a software tool that allows the user to create a set of rules that determine SoundStrand’s harmonic behavior. A *rule*, as a software entity, has two fields: *state*, which is a string expressing the harmonic function of the preceding cell; and *transitions*, which is an array of strings, each expressing the next function based on the degree of tensions as conveyed by the cell’s twist. All strings are Roman numeral representations of the harmonic function.

Previous	Statement	Preperation	Extension	Antecedent	Consequence
I	I	IV	vi	V	I
IV	I	ii	ii	V	I
V	vi	IV	ii	V	I
vi	I	IV	vi	V	ii
ii	I	IV	ii	V	V

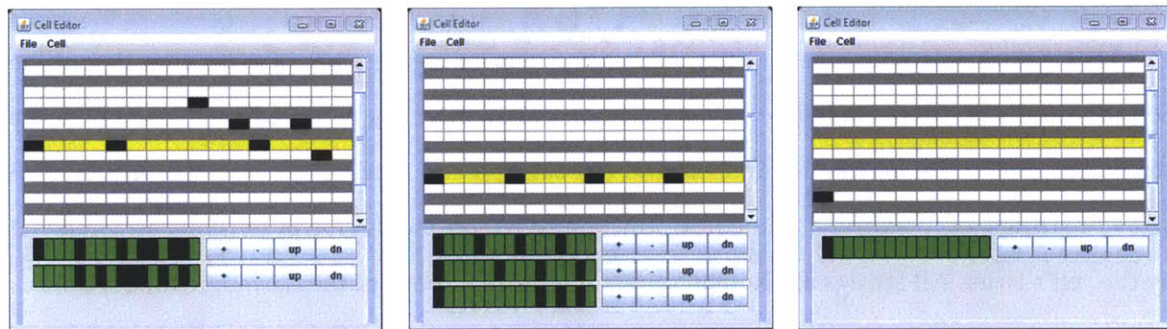
Figure 3-15: A screenshot of the HTT Editor

When the user begins creating a new HTT, she starts with an empty table. When adding a new rule, the software prompts the user to determine the *state* of that rule. The User can choose any degree on the scale, i.e. I to VII, and has a choice between major or minor chord, expressed in upper and lower case respectively. The user can then determine the different *transitions* of that rule. However, these can be only harmonic functions that already exist as *states* of a defined rule. This scheme preempts SoundStrand from reaching a harmonic function from which harmonic transitions are not defined.

Naturally, the HTT Editor supports deleting of rules, saving the HTT and loading existing ones for further editing.

3.6. Example of a SoundStrand session

Let us describe a typical, yet short, SoundStrand session. We start by programming three cell-types in the Cell Editor. These are shown in Figure 3-16. We shall also program a simple HTT in the HTT editor, as shown in Figure 3-17.



cell A

cell B

cell C

Figure 3-16: Three cell-types programmed in the cell editor

Previous	Statement	Preperation	Extension	Antecedent	Consequence
I	vi	IV	vi	V	I
IV	I	vi	IV	V	vi
V	vi	IV	V	V	I
vi	I	IV	vi	V	I

Figure 3-17: A simple HTT, programmed in the HTT editor

Cell A is intended to be used as a “question”, cell B will serve as an “answer” and cell C will serve as a closure to the theme. First, we select the *diatonic* mode. We continue by assembling four cells, in the order {A, B, A, C}, starting with the cells at their original state. This means that for every cell, the rhythmic pattern is the one closest to the middle of the permutations list, the melodic directionality is unaltered and the SPEAC identifier the *Statement*. The outcome is shown in Figure 3-18.



Figure 3-18: The theme created by a strand of four cells in their original state

Let us continue by adjusting the desired harmonic tension. We twist the second cell to the *Antecedent* position, the third cell to the *Preparation* and the last cell to the *Consequence* position. The result is shown in Figure 3-19.



Figure 3-19: The theme after adjusting the harmony

Now, we shall make some rhythmic variation. As we completely compress the first cell and fully stretch the second one, we select the first rhythmic pattern of cell-type A for the first cell and the third rhythmic pattern of cell-type B for the second cell. The outcome is shown in Figure 3-20.



Figure 3-20: The theme after modifying rhythmic patterns

Finally, let us shape the melodic contour. We shall bend the second cell upwards and the third cell downwards. After experimenting with different amounts of bending, we reach a plausible result which will be our final theme, as shown in Figure 3-21.

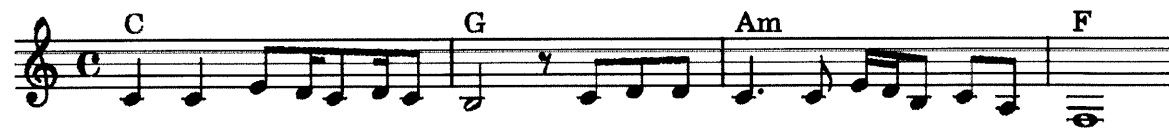


Figure 3-21: The theme after modifying the melodic contour

4. Beyond SoundStrand

4.1. Introducing Coda

Coda is a web service prototype developed by Stephen Bresnick and myself [40]. It is a collaborative musical encyclopedia, very much in the spirit of Wikipedia [41]. In Coda, however, the articles are accessed by selecting graphic entities in a musical score. Coda is meant to serve as the center for a community of learners who share music theory knowledge and musical ideas through musical pieces that bear personal meaning.

4.2. Learning Online

The internet redefined the meaning of a community and created opportunities for the formation of online communities of learners. A few notable examples for online learning communities are:

- MOOSE Crossing – A text-based world, or MUD – multi-user dungeon, that is aimed to teach object-oriented computer programming and creative writing to children between the ages of 8 to 13. Members of the community not only experience and interact with the MOOSE Crossing world with through a text console – they create it [42]. With a specially designed programming language called MOOSE, members of the community create objects and locations and define their behavior. Since the inhabitants of the world are both those who experience it and those who create it, community members are constantly surrounded by inspiring elements. These take the shape of projects made by their peers from which they can learn; more experienced peers who can assist with technical and moral support; and role models, specifically experienced members who enjoy computer programming in spite of gender or socio-economic stereotypes [43].
- FIRST – an organization that strives to spread passion for science and technology among children between the ages of 6 to 18 through robotics competitions. Every competing team is a small community of learners on its own that involves students, mentors,

parents and sponsors working closely and mostly in person. Moreover, the entire FIRST community acts as an online community of learners with a system of forums that supports knowledge exchange between mentors and students from all around the world. Exchange of knowledge is strongly encouraged by different means, following the principles of “gracious professionalism” [44]

- Scratch – Developed at the Lifelong Kindergarten group at the MIT Media Lab, Scratch was designed to teach children computer programming through storytelling and game design [45]. Users can create graphical, interactive environments by assembling blocks that correspond to common, simple software design mechanism such as loops, statements, input and output modules and conditional constructs. Scratch allows users to share their creations online, a feature which gave birth to a vibrant community in which users teach each other not only programming but also artistic skills. It is not uncommon to find Scratch projects created by members of the community that are, in fact, tutorials for creating projects in Scratch⁴ [46].
- The Knowledge Forum software – a general knowledge building environment for communities [47]. Members of the community share information, ideas and theories in *notes* that contain different kinds of media – text, graphics, movies etc. – and place them in *views* which are organized backgrounds for notes. Views can take the shape of concept maps, diagrams and so forth. From any of these entities, users can create yet higher levels, such as views of views, or lower levels, such as sub-notes within notes. The Knowledge Forum software pays particular attention to the discourse taking place in knowledge building communities, and provides tools to shape it [48].

4.3. System Description

Coda is built around the *NoteFlight* website. NoteFlight is an online music writing application that allows users to create, view, hear and share music notation [49]. It also supports an API that

⁴ Most fascinating are the Scratch projects demonstrating how to hack Scratch.

allows the embedding of NoteFlight scores and retrieval of information about the score and the user's interaction with the embedded score object.

Coda lets a user view a score from the NoteFlight database and select any sort of musical material – a set of notes, a collection of bars, etc. Once a selection is made, a data structure containing the score's content and information about the user's selection is transferred to the Coda server. The main back-end program, written in Python and utilizing the "mingus" music package [50], passes, in turn, the score and selection data to a bank of *analyzers*. These are algorithms that perform various kinds of musical analysis and produce text strings to be used as *entries*, which serve as keys to *articles*. Users are encouraged to write articles for entries that do not yet have one. Users can also edit existing articles. The analyzers repository itself is open-source and users are encouraged to add analyzers to the repository.

Let us summarize the terminology we use to describe Coda:

- *Article*: text describing a musical concept
- *Title*: a short, meaningful name given to an article by its author
- *Entry*: a text string that is used by the software to identify and direct to an article, whether written or not
- *Analyzer*: an algorithm that generates an entry from a selection of music notation symbols

4.4. Examples

Let us assume that the user selects the entire bar depicted in Figure 4-1. Here are some examples of how different analyzers might process the selection:

- The *Intervals* analyzer will ignore the bass staff, since it has polyphonic events. However, observing that the treble staff has only two notes, it will calculate the interval they form and will output the entry "interval:major_second".
- The *Duration* analyzer will observe that all of the notes have the same duration and output the entry "duration:half_note".

- The *Harmonic Progression* analyzer will observe that the treble clef contains a monophonic phrase and therefore ignore it. In the bass staff, it will find the two chords – an F-major followed by a C-Major. Noticing that piece is in a C-Major key, it will produce the entry “`progression:IV_I`”.
- The *Parallel Voices* analyzer will observe that the segment qualifies as a four part-harmony and will detect the presence of parallel octaves. It will therefore produce the entry “`parallel_voices:octave`”.



Figure 4-1: An example of a music segment analyzed by the Coda system

Let us continue through the example and examine a user’s interaction with these results, and for the sake of clarity, assume that they are read by the user in the same order described above. The first entry in the results list is “Major Second”. It seems another user has already wrote an article for this tag and also added a title that is a more readable string than the entry “`interval:major_second`”. Our user might read in this article that a major second is an interval of two semitones, and it is the first interval in the major scale.

Our user sees that the second result is “`duration:half_note`”. An article for this entry has not yet been written. Our user might choose to write this article, stating that a half note is twice the duration of a quarter note, half the duration of a whole note and in Italy is called “Minima”. The user might also give the article a cleaner title and call it “Half Note”.

The third result is called “Plagal Cadence”. The author of this article gave this title to the entry “`progression:IV_I`”, giving it a more profound meaning rather than stating the chord progression itself.

Finally, the user sees the entry “Parallel Octaves”, finding the article, written by another user, a refreshing reminder of the rules of four-part harmony writing.

4.5. Philosophy of Coda

Coda is designed to harness the internet's advantages in supporting effective learning and apply this to the learning of music theory [43]. Constructionist learning, in which learners construct their knowledge by sharing knowledge and building on community members' ideas [51], is achieved when users discover musical principals, if not in their own pieces then in a peer's , and learn about it from yet a third user. The context in which knowledge is acquired is always personally meaningful. It is accessed from engagement with musical pieces that are meaningful to the users. There are no lessons or tutorials – instead, musical concepts and principals are demonstrated through musical pieces that were uploaded by users and their peers.

Pluralism of participation is achieved since users are not required to upload pieces, write articles or contribute code. They participate by engaging in any combination of these activities with any level of musical expertise [39].

In a knowledge building community, the knowledge exists in the discourse of the community [48]. This is expressed in Coda in the collaborative wiki article system. In addition, built on top of, and being envisioned as a social network, Coda supports a personal, open discussion between users.

4.6. SoundStrand and Coda

SoundStrand, being an example of an interface for composition that does not require theoretical knowledge, provides an entry point to music theory learning through the Coda system. By creating a piece with SoundStrand and uploading it to Coda, users can explore theoretical principles and concepts demonstrated in their own piece, thus learning in a personally meaningful context. It is important to understand how different this approach is from traditional music education, in which theoretical principals must be learned first from pieces created by others before learners can implement them in their own creation. With the approach presented here, a learner first “plays” with the interface to create a piece and only then engages

a learning experience, or rather a theoretical discussion, which revolves around the piece. This is not to underrate, of course, the importance of analyzing and learning from pieces written by professional musicians, as they demonstrate good practices, interesting technique and, in the case of great composers, dictate the aesthetics in which we listen, evaluate and create.

Using the tools provided by SoundStrand and Coda, we can expect a learning process, much in the spirit of the natural learning experience in kindergarten as described by Mitchel Resnick [52]. According to Resnick, kindergarten learning is an iterative process, with every cycle going through the steps of imagining, playing, sharing and reflecting. Projecting SoundStrand and Coda's tools on this process, *imagining* takes place as the user first turns to SoundStrand to create a piece. She selects cell content files and HTT files based on the piece she "hears" in her head, whether it is a melody or a desired genre. She then *creates* the piece using the SoundStrand interface. As the user *plays* with the toy, an iterative process occurs of testing the piece and refining it to satisfy the user's artistic taste. Once the piece is finished, it is *shared* through Coda. This is an opportunity to receive feedback from a community, but also a step towards the user's ability to *reflect* on her creation. Once in the Coda system, the piece can now be reviewed by the user in theoretical terms. She can find musical entities such as harmonic progressions, rhythmical patterns or melodic tensions that she finds interesting and retrieve theoretical information regarding them. She will also be pointed to common examples in which these principals are used, which will broaden her music understanding and taste, and direct her imagination in new directions.

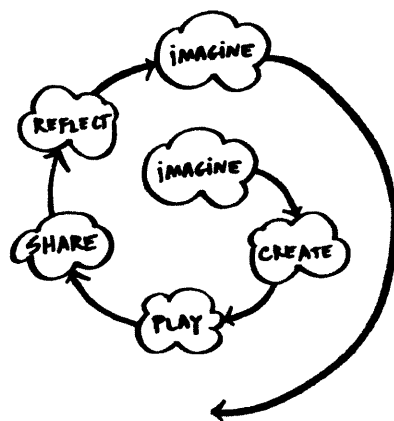


Figure 4-2: The kindergarten approach to learning [52]

This approach is not limited to beginners. Even more musically educated composers can benefit from a similar learning process. As SoundStrand content files that implement various styles and levels of musical complexity will be made available on the web by a community, advanced learners will first retrieve SoundStrand content files that implement those styles or principles they wish to learn, and so their first activities with these skills are *imagining* and *creating*. Once a personally meaningful context has been created, they are now ready to learn from their own doing.

An important aspect in both SoundStrand's and Coda's design is their capacity to enable users to affect the system's behavior through programming – content files in SoundStrand, analyzers in Coda. This is not merely a mechanism for producing more content; it is a tool for articulating formal knowledge. By programming, the user “teaches” the system meanings of different musical terms and therefore needs to think about them in a higher level of abstraction [39]. The concept of a system that has a “high ceiling” is applied here for both the programmers' peers who have access to new concepts that have been embedded in the system's intelligence and also to the programmers themselves, pushing the boundaries of the way in which they think about music.

In conclusion, from the standpoint of music learning, SoundStrand and Coda complement each other. Although they are both designed as self-contained systems, working in conjunction they

form a system for music learning that addresses learners from their very early stages but can accompany them in their advanced stages as they learn, formalize and create with deeper and more advanced musical concepts.

5. Conclusion and Future Work

5.1. Evaluation

5.1.1. Method

Over the course of six months, SoundStrand was demonstrated and put in the hands of many users from diverse backgrounds, receiving generally positive responses.

During the semi-annual member meeting in the Media Lab that took place between April 23rd and April 25th, 2012, Media Lab students and guests of the meeting were asked to spend some time interacting with SoundStrand and answering a short survey. The survey comprised of five statements, to which the participants were asked to rate their agreement on a scale of 1 to 5. Participants were also encouraged to write comments and suggestions (for the full survey, see Appendix A). Due to the fast paced nature of the event, only 12 participants chose to complete the survey. The results are presented in Table 5-1.

5.1.1. Analysis

To the statement “The interaction is intuitive”, participants responded positively with low variance. This suggests that almost all of the users felt immediately that they understand how SoundStrand should be operated.

Table 5-1: Evaluation survey results

Statement	Average	Variance
The interaction is intuitive	4.00	0.33
I understand how my actions are translated to music	4.08	1.25
The interface is responsive	4.29	0.94
I can easily get the musical result that I want	3.21	1.10
This method for composition is musically versatile	4.08	0.58

While participants also agreed with the statement that they understand how their actions are translated to music, the variance of the results was quite high. A possible factor to this is the difference in participants' musical background. Those who are musically trained might better articulate what they hear in musical terms and therefore correlate their actions to the results. From reviewing the comments, it appears that users are often able to better internalize the mapping of bending to pitch and elongation to rhythm than twisting to harmony ("*I don't understand what happens if I twist the SoundStrand*"). I postulate there are several factors causing this. Again, for participants with no musical background, harmony is often a vague concept. Second, the HTT normally used in demonstrations and the survey (see Table 3-1) was one that was clear to *me* as its designer, but perhaps a simpler one should have been chosen for new users. Finally, the mapping of the SPEAC identifiers themselves to the different twist positions could have had a negative impact on the intuitiveness of the harmony control. Instead of scrolling through the identifiers by order (Statement, Preparation, Extension, Antecedent, Conclusion), perhaps a different order should have been used, with the Statement identifier in the middle, for instance. Another alternative will be to abandon the SPEAC model as a source of inspiration and think of chord changes directly as a function of tension as expressed through the cell's twist.

The statement "The interface is responsive" was rated with an exceptionally positive response. This suggests that most users are able to detect changes in the music in response to their action. The variance, although somewhat high, still suggests a positive response, considering the high average.

Participants were generally neutral in regards to the ease in which they can produce desired musical themes. It seems users find it hard to *control* SoundStrand, even when reporting that the interface's response to their actions is coherent to them. Participants were only able to play with SoundStrand once, so it would be interesting to see whether this notion changes as users practice and gain experience with the interface over the course of multiple sessions. It is also important to remember that users were offered with a limited set of cell-types (4 cells, 2 of them of the same cell-type) and did not use the content generation toolkit.

Finally, there was a positive response to the statement “this method for composition is musically versatile”. The high average and low variance suggest that participants can feel the versatility of the musical paradigm even when operating SoundStrand for a short time and with a limited selection of cells.

While observing users interact with SoundStrand, a few important insights were gained. As these are reviewed, participants’ comments which support these insights are quoted:

- New users who received cells one at a time had a better understanding of the interface: this is not trivial, as most users interact with SoundStrand after viewing a demo and attempt to continue with that same, assembled strand. When encouraged to “start over”, users generally have a better understanding of the different cells, recognize the coupling between the cell and its corresponding motif and better internalize the mapping of their physical actions to the musical results.
- The lack of temporal, built-in visual feedback is crucial (*“some sort of visual feedback would be helpful to know where you are in the segment – like segmented lights, etc.”*). Some users look to the screen for feedback, although once aware of their screen dependence, they usually attempt to look away, having the impression that by looking to the screen they are “cheating” (*“I find myself using the visual interface to interpret the results of the tactile manipulation – not sure if that’s desired behavior”*).
- Many users tend to continuously twist, stretch and bend the cells, without waiting to hear the result. This is probably due to the lack of visual feedback, but also the form factor. The cells are compelling for users to hold in their hands rather than placing them on the table as intended (*“it really feels like something you can hold on to and ‘play’”*). While holding it, the users are often more occupied with physical play than listening.
- Users who also received a demonstration of the Coda system demonstrated greater understanding of the motivation behind tools, such as SoundStrand, that empower musically non-educated people to engage in musical composition.

Generally, most users reported the SoundStrand as being a “*fun interface*” and wished they had more time to experiment with SoundStrand and develop the intuitions that will allow them to act on the interface with intent.

5.2. Improving SoundStrand

There is wealth of possibilities to adjust, change and improve SoundStrand in its existing form. The notion of being able to physically hold a representation of a musical piece in one’s hands inspires many directions for continued work. Most exciting is the possibility to perform or conduct the finished piece by handling “it” in space. For that purpose, SoundStrand should be enhanced with location, movement and pressure sensors and appropriate methods for mapping should also be devised.

Other possible improvements for the SoundStrand interface may include:

- **Materiality:** the materials used to construct SoundStrand’s skeleton, skin and electrical connections were selected due to their availability in the Media Lab. This compromises SoundStrand’s robustness. A thorough research for more adequate materials and components will surely benefit the performance of the interface.
- **Mechanical design:** although performing very well in its current design, the skeleton can benefit from some improvements. These should mostly address the ease in which the user can adjust the elongation property while the cell is considerably bent and the skeleton’s capacity to maintain its shape under the weight of connected cells.
- **Visual feedback:** as expressed by many users, this is a crucial element that is missing in SoundStrand’s current design. An improved visual feedback system might illuminate the currently playing cell or even the current playing position within the cell.
- **Transport and meta-data control:** in the current design, in order to play or stop SoundStrand’s playing, audition a cell, change timbres or change tempo, the user must turn to the computer. Ideally, the interaction with the computer would be minimal; therefore, additional sensing capabilities, such as pressure or proximity, may be introduced to the cells to change cell-specific operation and transport functions. Meta-

data cells may also be introduced to control global parameters such as tempo and timbre.

- Zooming out: in order to allow longer pieces to be composed, a zooming out option can be offered, where themes composed by the user are represented by cells rather than the shorter, pre-programmed motifs. The strand then becomes the structure of the song, and the physical manipulations of the cells can represent variations in the dynamics, length or arrangement of the structural elements.
- Zooming in: further using the idea of working with the same interface in a different scope of the musical piece, the SoundStrand interface might also be used to program the phrases rather than using the content generation toolkit, with the physical manipulation of the cell changing the notes' pitch and duration.

5.3. Composing with Limited Degrees of Freedom

5.3.1. Improving the Paradigm

The implementation of the musical paradigm governing SoundStrand is far from allowing fully explicit composition, and making such an attempt is beyond the scope of this project. Trying to maintain a reasonable scope resulted in the selection of the constraints that still provide a sense of musical flexibility. The main tradeoff as a result of these choices was between automated versus manually controlled variations of the musical material.

Further developments of the composition with limited degrees of freedom paradigm may remove some of the constraints forced on the paradigm's implementation and allow:

- Support for multiple harmonic transitions within a single measure
- Variable length motifs
- "Soft" motif borders – the ability for a phrase to bleed into another motif's bar.
- Automatic generation of rhythmic permutations

- Individual parametric control over note transposition in response to changes in the melodic directionality – the ability to have individual notes more (or less) responsive to cell bending
- Automated harmonic transitions

5.3.2. Multitude of Interfaces

The design of SoundStrand’s physicality was largely dictated by the desired mapping of physical manipulations to musical variations. As explained in 3.2.1, this mapping was devised with the intention to be as intuitive as possible to the user. SoundStrand is, of course, only one of an infinite number of possible tangible interfaces that can use any sort of mapping. Naturally, different form factors and scale suggest different usages and exhibit various advantages and disadvantages, as well as new possibilities altogether.

5.3.3. Collaborative Composition

Revisiting the various components of a piece composed with SoundStrand, we observe the following layers:

- Motivic – the pitch properties of the events and the possible rhythmic permutations of every phrase
- Harmonic – the harmonic transitions table
- Contextual – the specific collection of musical phrases and how they are assigned to cell-types
- Structural – the ordering of the different cell-types and their manipulations along the three degrees of freedom

These layers describe the piece along orthogonal axes. They operate on different scopes of the piece, and they require different levels of musical skill to engage. This suggests that a musical composition may be the product of the effort of a group of people, each operating on the piece at a different layer and in a different scope. Collaborative composition using limited degrees of freedom can take many forms, such as a set of large-scale objects, filling the function of

SoundStrand's cells, manipulated simultaneously by participants in a public space. Another example is an online project in which users contribute to any of the layers, either while relating to musical material already contributed by other collaborators or independently of their work.

5.4. Reflections

In my view, more than being a tool for composition, SoundStrand is a tool to demonstrate ideas. On the surface level, it shows that tangible musical composition is a feasible idea, and it demonstrates a method to achieve it. On a deeper level, it is a tool that helps explain new ideals of music engagement, such as creating, collaborating and mostly – learning. It is my deepest hope that the work I have shown here will impact the design of new interfaces for music composition, whether they are physical or virtual, and the way people learn, share and create music.

Appendix A: Evaluation Survey

Thank you for taking the time to help us evaluate SoundStrand. After you have played with SoundStrand for a while, please ring the response that you think is most appropriate to each statement:

The interaction is intuitive

Strongly disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

I understand how my actions are translated to music

Strongly disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

The interface is responsive

Strongly disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

I can easily get the musical result that I want

Strongly disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

This method for composition is musically versatile

Strongly disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

Please use this space for any comments you might have:

Appendix B : Recruitment Text

“Thank you for playing with SoundStrand. To help us evaluate and improve SoundStrand, we would be thankful if you took the time to fill a short survey. The survey is completely anonymous (*we will here present the subject with the survey*). We might quote comments in various publications. Please feel free not to write comments if this makes you uncomfortable”.

Appendix C : Consent to Participate Form

CONSENT TO PARTICIPATE IN INTERVIEW

SoundStrand: a Tangible Interface for Composing Music with Limited Degrees of Freedom

You have been asked to participate in a research study conducted by Eyal Shahar from the Media Lab at the Massachusetts Institute of Technology (M.I.T.). The purpose of the study the use of tangible interfaces for music composition. The results of this study will be included in Eyal Shahar's Masters thesis. You should read the information below, and ask questions about anything you do not understand, before deciding whether or not to participate.

- This interview is voluntary. You have the right not to answer any question, and to stop the interview at any time or for any reason. We expect that the interview will take about 5 minutes.
- You will not be compensated for this interview.
- Unless you give us permission to use your name, title, and / or quote you in any publications that may result from this research, the information you tell us will be confidential.

This project will be completed by 4/25/2012. All survey forms will be stored in a secure work space until 1 week after that date. The forms will then be destroyed.

I understand the procedures described above. My questions have been answered to my satisfaction, and I agree to participate in this study. I have been given a copy of this form.

I give permission to use direct quotes from this survey in publications resulting from this study

Name of Subject _____

Signature of Subject _____ Date _____

Signature of Investigator _____ Date _____

Please contact Eyal Shahar with any questions or concerns:

Email: persones@mit.edu

Phone: 617-386-3368

If you feel you have been treated unfairly, or you have questions regarding your rights as a research subject, you may contact the Chairman of the Committee on the Use of Humans as Experimental Subjects, M.I.T., Room E25-143b, 77 Massachusetts Ave, Cambridge, MA 02139, phone 1-617-253-6787.

Bibliography

- [1] J Bamberger, *The mind behind the ear: How children develop musical intelligence*. Cambridge, MA: Harvard University Press, 1991.
- [2] D. Cope, *Experiments in Musical Intelligence*. Madison, Wisconsin: A-R Editions, 1996.
- [3] M. Minsky, "Music, mind, and meaning," in *The Music Machine: Selected Readings*, C. Roads, Ed., 1989.
- [4] B. Ullmer and H Ishii, "Emerging frameworks for tangible user interfaces," *Human-Computer Interaction in the New Millenium*, pp. 579-601, 2001.
- [5] H. Ishii and B. Ullmer, "Tangible bits: Towards seamless interfaces between peoplebits and atoms," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, New York, NY, 1997.
- [6] T. Machover, "Hyperinstruments: A Progress Report," Cambridge, MA, 1992.
- [7] J. Paradiso, "The Brain Opera Technology: New Instruments and Gestural Sensors for Musical Interaction and Performance," *Journal of New Music Research*, vol. 28, no. 2, pp. 130-149, 1999.
- [8] M. Minsky, *The society of mind*. New York: Simon & Schuster, 1985.
- [9] Machover, T. et al. (2004) Toy Symphony. [Online]. <http://www.toysymphony.net>
- [10] K. Jennings, "Toy Symphony: An International Music Technology Project for Children.," *Music Education International*, vol. 2, pp. 3-21, 2003.

- [11] G., Aimi, R. and Jennings, K. Weinberg, "The Beatbug Network - A Rhythmic System for Interdependent Group Collaboration," in *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, Dublin, 2002.
- [12] G., Orth, M., and Russo, P. Weinberg, "The Embroidered Musical Ball: A Squeezable Instrument," in *Extended abstracts of the CHI 2000 conference on human factors in computing systems*, The Hague, Netherlands, 2000, pp. 283-284.
- [13] Margaret A. Orth, "Sculpted Computational Objects with Smart and Active Computing Materials," Massachusetts Institute of Technology, Cambridge, MA, Doctoral dissertation 2001.
- [14] M. Farbood, H. Kaufman, and K. Jennings, "Composing with Hyperscore: An Intuitive Interface for Visualizing Musical Structure," in *Proceedings of the 2007 International Computer Music Conference*, San Francisco, 2007, pp. 471-474.
- [15] Hyperscore. [Online]. <http://www.hyperscore.com/>
- [16] (2011) Festival of Arts, Science and Technology at MIT. [Online]. <http://arts.mit.edu/fast/>
- [17] J. C. Fox. (2011) The Boston Globe website. [Online].
http://www.boston.com/yourtown/boston/downtown/gallery/figment_boston/
- [18] D. Young and H. C. Bennet-Clark, "The role of the tymbal in cicada sound production.," *Journal of Experimental Biology*, no. 198, pp. 1001-1019, 1995.
- [19] R. K. Josephson and D. Young, "A synchronous insect muscle with an operating frequency greater than 500 Hz," *Journal of Experimental Biology*, no. 118, pp. 185-208, 1985.
- [20] B. Schiettecatte and J. Vanderdonck, "AudioCubes: a Distributed Cube Tangible Interface based," in *Proceedings of the Second International Conference on Tangible and Embedded Interaction*, Bonn, Germany, 2008, pp. 3-10.

- [21] J. Harrison, "SoundBlocks and SoundScratch: Tangible and Virtual Digital Sound Programming and Manipulation for Children," Massachusetts Institute of Technology, M.S. Thesis 2005.
- [22] J. Sosoka, B. Abercrombie, B. Emerson, and A. Gerstein, "Educational Music Instrument for Children," 6,353,168, March 5, 2002.
- [23] J. T. Bernstein. (2005) The tangible sequencer - a simple musical instrument. [Online]. <http://murderandcreate.com/tangiblesequencer>
- [24] R. Berry, M. Makino, N. Hikawa, and M. Suzuki, "The Augmented Composer Project: The Music Table," in *Proceedings of the 2003 International Symposium on Mixed and Augmented Reality*, Tokyo, Japan, 2003, pp. 338–339.
- [25] S. Jorda, M. Kaltenbrunner, G. Geiger, and R. Becina, "The Reactable*," in *Proceedings of the International Computer Music Conference (ICMC 2005)*, Barcelona, Spain, 2005.
- [26] M. Kaltenbrunner and R. Bencina, "reactIVision: a computer-vision framework for table-based tangibleinteraction," in *Proceedings of the 1st international conference on Tangible and embedded interaction*, 2007, pp. 69-74.
- [27] the reacTable. [Online]. www.reactable.com
- [28] D. Merrill, J. Kalanithi, and P. Maes, "Siftables: Towards sensor network user interfaces," in *Proceedings of the 1st international conference on Tangible and Embedded Interaction*, Baton Rouge, Louisiana, 2007, pp. 75-78.
- [29] Sifteo. [Online]. www.sifteo.com
- [30] J. Bamberger, "Developing a Musical Ear: a New Experiment," Massachusetts Institute of Technology, Cambridge, MA, Memo 264, 1972.

- [31] J. Bamberger, *Developing Musical Intuitions*. New York: Oxford University, 2000.
- [32] G. Assayag, C. Rueda, M. Laurson, C. Agan, and O. Delerue, "Computer-Assisted Composition at IRCAM: From PatchWork to OpenMusic," *Computer Music Journal*, vol. 23, no. 3, pp. 59-72, Autumn 1999.
- [33] OpenMusic. [Online]. <http://repmus.ircam.fr/openmusic/home>
- [34] J. Bresson, "ML-Maquette / Musique Lab 2," in *Proceedings of the International Computer Music Conference*, New York City / Stony Brook, NY, 2010.
- [35] D. Cope, *Computer and Musical Style*. Madison, WI: A-R Editions, 1991.
- [36] Y. Shen, "SoundStrand Design: Designing Mechanical Joints to Facilitate User Interaction within a Physical Representation of Digital Music," Massachusetts Institute of Technology, Cambridge, MA, B.S. Thesis 2011.
- [37] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*.: Addison-Wesley, 1995.
- [38] S. Turkle, *Evocative Objects: Things We Think With*. Cambridge, MA: MIT Press, 2007.
- [39] S. Papert, *Mindstorms: Children, Computers, and Powerful Ideas*. New York: Basic Books, 1980.
- [40] S. Bresnick and E. Shahar, "Coda," Massachusetts Institute of Technology, Cambridge, MA, 2011. [Online]. <http://web.media.mit.edu/~persones/coda.pdf>
- [41] Wikipedia. [Online]. <http://en.wikipedia.org/wiki/Wikipedia>
- [42] A. Bruckman, "The Future of E-Learning Communities," *Communications of the ACM*, vol. 45, no. 4, pp. 60-63, April 2002.

- [43] A. Bruckman, *The Cambridge Handbook of the Learning Sciences* NY:, R. K. Sawyer, Ed. New York: Cambridge University Press, 2006, ch. 27, pp. 461-472.
- [44] A. Melchoir, C. Cohen, T. Cutter, and T. Leavitt, "More than Robots: An Evaluation of the FIRST Robotics Competition Participant and Institutional Impacts," , Waltham, MA, 2004. [Online]. http://origin-www.usfirst.org/uploadedFiles/Who/Impact/Brandeis_Studies/FRC_eval_finalrpt.pdf
- [45] M. Resnick et al., "Scratch: programming for all," *Communications of the ACM*, vol. 52, no. 11, pp. 60-67, November 2009.
- [46] Scratch. [Online]. <http://scratch.mit.edu/>
- [47] The Knowledge Forum Software. [Online]. <http://www.knowledgetforum.com/>
- [48] M. Scardamalia and C. Bereiter, "Knowledge Building: Theory, Pedagogy, and Technology," in *Cambridge Handbook of the Learning Sciences*, R. K. Sawyer, Ed. New York: Cambridge University Press, 2006, ch. 7, pp. 97-118.
- [49] NoteFlight. [Online]. <http://www.noteflight.com>
- [50] mingus. [Online]. <http://code.google.com/p/mingus/>
- [51] Y. B. Kafai, "Constructionism," in *The Cambridge Handbook of the Learning Sciences*, R. K. Sawyer, Ed. New York, NY: Cambridge University Press, 2006, pp. 35-46.
- [52] M. Resnick, "All I really need to know (about creative thinking) I learned (by studying how children learn) in kindergarten," in *Proceedings of the 6th ACM SIGCHI conference on Creativity & cognition*, Washington, DC, USA, 2007, pp. 1-6.

[53] M. Farbood, "Hyperscore: A New Approach to Interactive Computer-Generated Music,"
Massachusetts Institute of Technology, Cambridge, MA, M.S. Thesis 2001.

[54] M. Puckette, "Max at 17," *Computer Music Journal*, vol. 26, no. 4, pp. 31-43, 2002.

[55] M. Pucketter, "Pure Data," in *Proceedings of the International Computer Music Conference*, San
Francisco, 1996, pp. 269-272.