A Flexible Platform for Tangible Graphic Scores

by

Simon Alexander-Adams
Bachelor of Fine Arts, University of Michigan
School of Music Theatre and Dance, 2014

Thesis Submitted in Partial Fulfillment of the Requirements of the Degree of
Master of Arts, Media Arts

School of Music, Theatre & Dance
University of Michigan

April 2015

Approved by: ______________________________

Michael Gurevich, Thesis Director
Assistant Professor, Performing Arts Technology,
School of Music Theatre and Dance

____

Sile O'Modhrain, Thesis Committee Member,
Associate Professor, Performing Arts Technology
School of Music Theatre and Dance

____

Stephen Rush, Thesis Committee Member,
Professor, Performing Arts Technology
School of Music Theatre and Dance

____

Christopher McNamara, Thesis Committee Member,
Lecturer, Screen Arts and Cultures
College of Literature, Science, and the Arts

Date Degree Conferred: May 2015

ABSTRACT

This thesis describes a system for composing and performing *tangible graphic scores*. A tangible score is simultaneously interface and composition in that musical instruction is encoded into the body of the instrument itself. The finished device utilizes modular textured surfaces in conjunction with interactive compositional structures, exploring aspects of indeterminacy, simulated physics systems, and the concept of utilizing traces of previous performances as the basis for future compositions. This document begins by examining pertinent historical work, and related research to the topic, and discusses the conceptual, aesthetic and motivational factors that influenced the compositional decisions. The development of the system is documented, outlining design considerations, and stepping through initial ideas and iterations, up to the final implementation of the system. Then, the compositional process is examined for three separate systems of compositions, followed by a description of a performance given with the system in the Digital Media Commons Video Studio. Lastly, the paper critiques the successes, shortcomings, limitations and possibilities for future development of the system.

ACKNOWLEDGEMENTS

Michael Gurevich for your guidance and advising with this thesis, and for introducing me to the world of interaction design. Working with you has significantly influenced my practice of art making and the way I work with technology.

Thesis committee members, Stephen Rush, Sile O'Modrain, Christopher McNamara for your insightful comments about this project, and support throughout my undergraduate and graduate work at the University of Michigan.

The Digital Media Commons Video Studio staff for providing an excellent facility for the final performance of this thesis. Especially Jeff Alder and Jacques Mersereau, for their deep concern for the quality of the student productions taking place in the Duderstadt Video Studio

Carlos Garcia, not only for enabling this thesis through access to equipment and workspace, but also serving as a source of advice from the technical to artistic.

The Makerworks staff, for providing a wonderful facility for fabrication and making things.

Eric Sheffield and Connor Barry, for being awesome fellow grad students, and providing an open environment for sharing ideas and providing feedback throughout the design process.

The Performing Arts Technology department for being such an amazing community of artists, creators, musicians, thinkers and collaborators. It would be impossible to name everyone who has made my time at the University of Michigan the great experience that it was.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

During my Master's study I have continued to develop an interdisciplinary art practice involving many aspects of music, visual art, and interaction design. This practice ranges from the development of interdisciplinary performances, installation art, and new interfaces for artistic expression. Working within this intersection of art and technology, I frequently find myself switching between the roles of composer, visual artist, designer, engineer and coder, to name a few. In this thesis, I continue developing this practice through the design and implementation of a new musical instrument for the composition and performance of *tangible graphic scores.*

Graphic notation has specific relevance to this project. Over the last four years I have composed a number of traditional graphic scores, often for small ensembles with open instrumentation (Figure 1). It is important to draw a distinction between graphic notation, and visual art. In composing a graphic score, I consider visual forms temporally, varying parameters, exploring thematic material and spatial relationships. It is this intention that separates these scores from my other visual work.

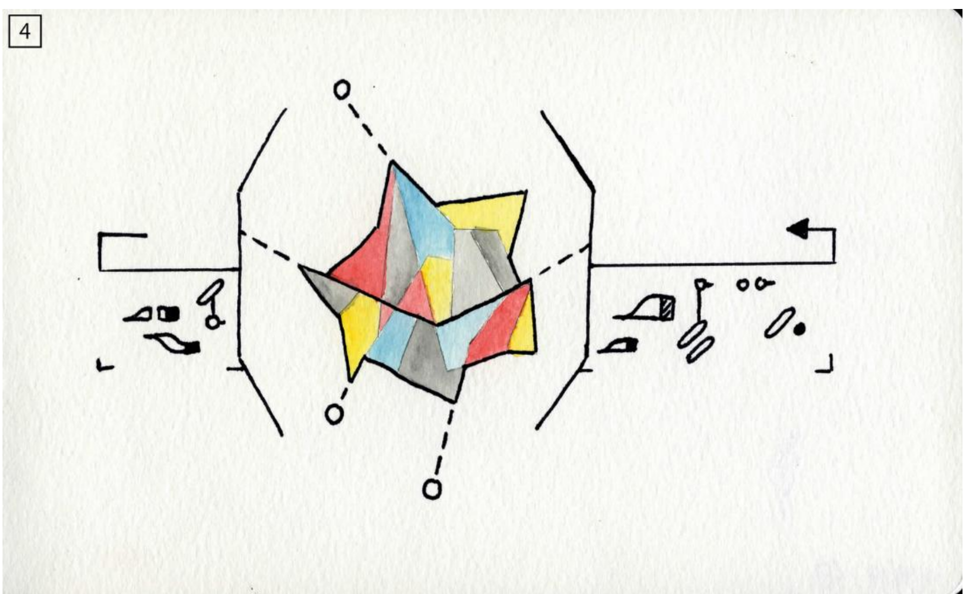


Figure 1. An excerpt from Seven Systems, a set of graphic scores composed by the author

It was through studying with Professor Michael Gurevich that I began to develop a practice in interactive media design, installation art, and interface design. A number of the interfaces and installations I have developed over the last three years constituted what I call *Hybrid Scores* – devices that exist as both score and interface simultaneously. Developing such devices and experiences stem from my parallel interest in graphic notation, and inclination toward abstract representation.

Working with Professor Sile O'Modhrain, I developed further interest in multi-sensory modalities, creating user experiences that explore the relationship between our senses. The sense of touch specifically interested me in how it pertains to developing electronic music interfaces. As a pianist, I have direct experience with the importance of the subtle haptic (vibrational) feedback provided naturally in acoustic instruments, and the way it differs from the feel of most digital keyboard controllers. *Pressure Cube (2013)* was one of my first controller designs to explore passive haptic feedback (Figure 2), sensing the pull on elastic cords suspending a small object in the center of a cube shaped frame. I perused this thread further, implementing a low cost haptic paddle modeled after Stanford University's *Hapkit* (Morimoto 2013).Through building these interfaces, I began to engage with *tangibility* and *tactility*, a thread I continue to explore further in my instrument designs.

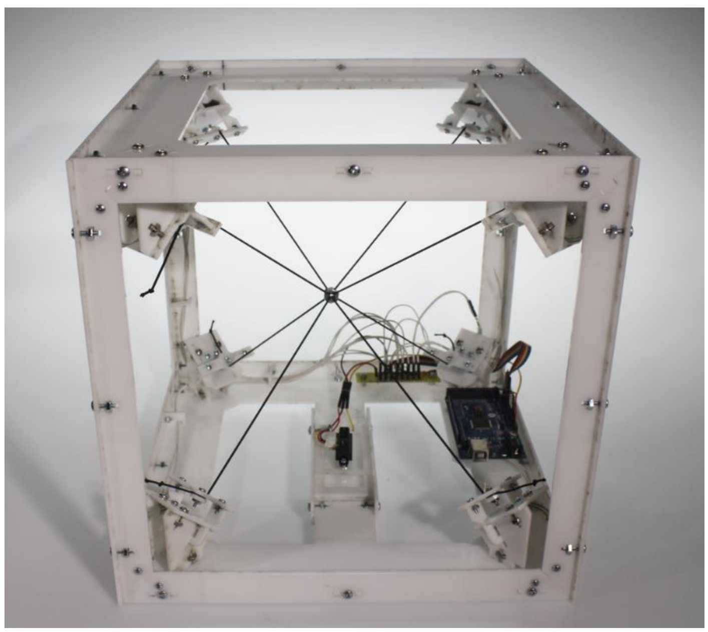


Figure 2. Pressure Cube, a controller exploring passive haptics

I first encountered the concept of *Tangible Scores* in a presentation by Tomás and Kaltenbrunner at the proceedings of the 2014 International Conference on New Interfaces for Musical Expression. They define a tangible score as a "physical layer that is incorporated into the configuration of a digital instrument with the intention of conducting the tactile gestures and movements. A composer could then conceive a musical idea and incorporate it to the instrument body encoding it in a physical manner" (Tomás and Kaltenbrunner 2014). The notion of *Tangible Scores* resonated with many threads already present in my work – such as graphic and hybrid scores, tangibility, and interaction design – providing the inspiration and motivation to develop a flexible system for composing them.

This thesis document begins by discussing related historical work and research to the topic, as well as the conceptual and aesthetic influences which guided compositional decisions. The development of the system is documented, outlining design considerations, initial ideas and iterations, and the final implementation of a system form composing tangible graphic scores. The compositional process is examined for three separate systems of composition, followed by a description of a performance given with the instrument. Lastly, the paper critiques the successes, shortcomings, limitations and possibilities for future development of the system.

2. BACKGROUND

2.1 Graphic Scores and Nontraditional Notation

The limited ability of traditional notation to embody the musical concerns of contemporary composers—timbre, gesture, texture, nonlinear time, indeterminacy—drove several music pioneers of the 1960's to develop and explore a variety of alternative notational systems. Graphic scores can be extremely flexible and open-ended modes of compositional creativity, especially as they are not limited to two-dimensional or static forms, nor constrained to any specific artistic medium. Unified through diversity rather than conformity, graphic notation remains a tradition in its own right. Its influence has expanded far beyond that of static notation, informing many interactive systems and instruments for musical expression.

2.1.1 Conventional Graphic Notation

Earle Brown was one of the first composers to utilize graphic notation, notably in his iconic piece *December 1952* (Earle Brown 2002). Countless composers have since experimented with graphic notation, and there have been significant efforts to archive graphic scores. *Notations 21*, compiled by Theresa Sauer, is a collection featuring graphic notation from composers from over 50 countries (Sauer 2009). Collections such as these illustrate how widespread and influential the practice has become.

Systems for musical notation can be situated on a spectrum of instructional constraints. On one end is traditional notation, in which every symbol has an explicit definition; while on the other end are open scores, such as Brown's *December 1952* (Earle Brown 2002), in which the performer has complete discretion over their interpretation. Cornelius Cardew's massive *Treatise* (1967) is another example of an open graphic score, consisting of 193 pages of abstract symbols, numbers and occasionally musical symbols, with no explicit instruction or indication given as to how to interpret it (Cardew 1967). Between these two extremes, many scores use both explicit and abstract symbolism. John Cage (Cage

1962) and Robert Ashley (Gann 2012, 32) often impose lengthy descriptions as to how their scores are interpreted, while simultaneously leaving the nature of the sound itself up to the performer, only defining durational and rhythmic considerations (Husarik 1983; Weingarten 2008).

One significant difference in graphic notation is the break with the traditional paradigm for music to unfold linearly. Nonlinear forms were part of Stockhausen's scheme for "form-genesis", outlining procedures for developing polyvalent compositions: compositional forms that involve nonlinear trajectories (Coenen 1994, 218). The notion of nonlinearity can easily be applied in graphic notation, through which composers are liberated from the notion of a timeline, and performers have the opportunity to share responsibility for decisions in formal and rhythmic structure.

In some cases, it is easier to express musical ideas through graphic notation than by traditional means. In Iannis Xenakis' *Metastasis,* the graphic component of the score illustrates many continuous glissandi to be performed by strings. To have this performed, Xenakis painstakingly notated sixty-one individual parts, a process which in part inspired his conception of UPIC, a software originally designed in 1977 for composing electronic music graphically (Marino, et al. 1993). As the radical variation of timbre available in electronic music is difficult to notate effectively, graphic notation is often a useful method for abstractly representing its complexity (Krumhansl 1989). Graphic notation has the potential to convey form, gesture, timbre, and other musical properties through non-standardized symbols, making it particularly suited for new interfaces for musical expression.

Graphic notation is a rather flexible medium. It can be used to notate preexisting unnotated compositions, such as Rainer Wehinger's Aural Score for Ligeti's fixed electronic composition *Artikulation,* (Ligeti 1970) which manages to capture rhythmic, frequency, timbre and even spatial (panning) musical parameters through a purely pictographic representation. It can also go beyond two-dimensional forms, such as the three-dimensional scores of Marc Berghaus (Berghaus 2011). Additionally, graphic notation is not limited to music. It has been applied in theatrical settings such as Robert Ashley's opera *in memoriam: Kit Carson* (Austin, Douglas and Nilendra 2011), and there are numerous graphical notation traditions in dance, including Labanotation (Ward, et al. 2008).

2.1.2 Dynamic and Interactive Scores

Dynamic scores move beyond static notation into a space of temporality. Donald Scavarda's *filmSCORE* for Two Pianists (1962), is an early example of a dynamic score, consisting of video which is to be interpreted by performers as it unfolds (Weingarten 2008). Brown's *Calder Piece* (1966) employs

Calder's mobile sculpture *Chef D'Orchestre*, created specifically for the piece, as a the conductor for four percussionists (Earle Brown 2002). *Calder Piece* is significant not only as three-dimensional notation, but also for its kinetic dynamics that inherently behave differently in every performance. Recent examples of dynamic scores include the animated notational interfaces of Justin Yang (Yang 2008) and Homei Miyashita (Miyashita and Nishimoto 2004).

Joel Chadabe describes interactive composing as "a two-stage process that consists of (1) creating an interactive composing system and (2) simultaneously composing and performing by interacting with that system as it functions" (Chadabe 1984). Where dynamic scores possess the capability to animate the spatio-temporal qualities of static graphic scores, *interactive* scores introduce the nonlinear into the dynamic. Interactive scores are most frequently not graphic in nature, though some of the earliest examples were firmly planted in the tradition.

Xenakis' aforementioned UPIC software was originally created in 1977 as a means to compose graphic scores that could be symbolically interpreted by a computer to generate electronic music. Ten years later, as computing power improved, a version of UPIC was created that enabled the "real-time interpretation of a score and, moreover, the real-time control of all the parameters of a sound" (Marino, et al. 1993, 264) The performer can navigate the score, looping sections, jumping in location from page to page, and adjusting tempo in real time. The system can also record segments of the performer's gestures to be edited later, or applied it in real time to different material. Thus, UPIC allows the performer to interact with the score itself (Marino, et al. 1993, 266). This iteration of the UPIC system is one of the first platforms for composing interactive scores, and has continued to inspire those after it. Namely, *IanniX*, a recent software based on the principles of UPIC serves as a "poly-temporal meta-squencer," using symbolic objects to define both macro and micro parameters of interactive graphic scores (Coduys and Ferry 2004).

Interactive scores play an important role in many types of multimedia, and are not only used to control music and audio, but also image, video, and lights (Toro, Desainte-Catherine and Baltazar 2010). Video games in particular make frequent use of interactive music scores, especially when the unfolding of a narrative, or linear progression is determined by the player's real time decisions and actions. Video game scores are often flexible in form, allowing for looping and the addition or subtraction of layers, and sometimes employing aleatoric and algorithmic procedures to produce variation (Sweet 2015). There are instances where the interactive qualities of video game scores have in influenced musical notation practices. David Lieberman leverages the non-linear qualities of video games in what he calls *game-*

scores, “applying game theory concepts to the creation of interactive music manuscript” (Lieberman 2006, 245). The affinity influenced some of the tangible graphic scores for my system, as some of the visual content and interactions bear resemblance to video games.

2.1.3 Tangible Scores

Tomás and Kaltenbrunner define tangible scores as a “the physical layer that is incorporated into the configuration of a digital instrument with the intention of conducting the tactile gestures and movements” (Tomás and Kaltenbrunner 2014, 610). Their own implementation of the concept involves engraving designs into wood, and using the acoustic signal created from scratching the surface of the wood to drive concatenative synthesis. Tomás and Kaltenbrunner cite Alvin Lucier’s concept of the *inherent score* as one of the inspirations for their own work with tangible scores (Tomás and Kaltenbrunner 2014). Lucier observes that in many of Gordon Mumma’s sonic circuits, “the scores were inherent in the circuitry” (Lucier 1998).

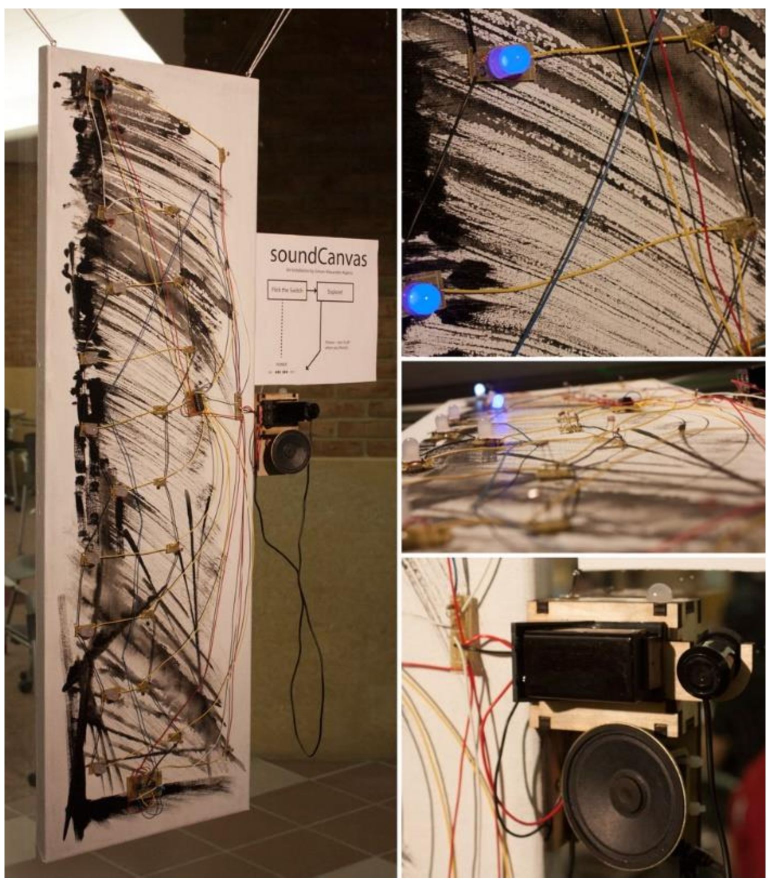


Figure 3. soundCanvas, the author's first interactive tangible score

My recent interactive piece, *soundCanvas*, is my own first inherent score, which nearly constitutes a tangible score (Figure 3). *soundCanvas* was inspired by Nicolas Collins's concept of a "circuit sniffer"—any transducer (often electromagnetic) used to amplify the hidden sounds of everyday circuits (Collins 2006). Additional inspiration came from Nam June Paik's *Random Access*, an installation in which viewers scrub cassette tape arranged on a wall using a disembodied tape head (Paik 2002). In both examples, one traverses a visual structure—a circuit, cassette tape on a wall—and in the processes uncovers a hidden sonic landscape. In *soundCanvas*, the viewer uses an electromagnetic transducer to amplify a circuit embedded into the surface of a painting. Photocells in the circuit create sonic variation as viewers cast shadow on the piece, reacting dramatically when hit with a flashlight or bright source of light. Instead of compressing circuitry into a compact enclosure, wires stretch across the canvas, displacing components and creating isolated islands for sonic exploration. While *soundCanvas* was initially intended as an installation, it can also function as an interface for musical performance, utilizing both the visual structure of the painted canvas, and the unseen networks of the circuit as an interactive tangible score. Like Mumma's circuits, the score is inherent in the circuitry, uncovered by the performer's explorations with the circuit sniffing device.

Vinyl records can also be considered of tangible scores, in the way they are used to perform and organize musical content in the practice of turntablism. In the late 1970s, Christian Marclay was among the first to use turntables as experimental musical interfaces. He regularly covered records with tape and markings to indicate to location of certain sounds, created "lock grooves" to create loops, and intentionally distorted their surface to introduce sonic artifacts (Kahn 2003). Marclay developed his own work as similar practices simultaneously emerged in Hip Hop (Pray 2002). Both were precursors to turntablism's metaphorical significance in many forms of digital electronic music (D'Arcangelo 2004). The notion of vinyl records as tangible scores influenced a number of decisions pertaining to the modularity of my final system for tangible graphic scores.

2.2 Interfaces for Musical Expression

Today, designers and inventors of new interfaces for musical expression have an ever increasing tool box of technological possibilities. With the introduction of open hardware platforms such as Arduino and Raspberry Pi, the increasing power of computers, and the ubiquity of digital fabrication tools such laser cutters and 3D printers, instrument design has become more accessible and immediate than ever before. This section discusses tabletop interfaces and tangible user interfaces, two areas of human computer interaction relating to the ultimate design of my platform for tangible graphic scores.

2.2.1 Tangible User Interfaces

In the mid 1990's a movement towards *Tangible User Interfaces* (TUIs) began, "motivated by the desire to retain the richness and situatedness of physical interaction, and by the attempt to embed computing in existing environments and human practices to enable fluid transitions between 'the digital' and 'the real' " (Shaer 2015, 6). TUIs represent information not with pixels on a screen (in the case of Graphic User Interfaces), but in a physical environment. As Hiroshi Ishii puts it, "TUIs make digital information directly manipulatable with our hands and perceptible through our peripheral senses through its physical embodiment" (Ishii 2008). TUIs attempt to leverage the haptic and dexterous skill that we naturally develop through our interaction with the physical, fitting seamlessly into the user's environment.

Some TUIs are intended to render the involved technology transparent to the interaction. Examples include projects from the MIT Media Lab allow users to manipulate clay (Piper, Ratti and Ishii 2002) and sand (Ishii 2008) in sculpting physical models, featuring top down projection displays for visual feedback. Others incorporate sensing to add interactivity to existing objects, such as the use of swept capacitive touch sensing to let users interact sonically with plants (Honigman, Hochenbaum and Kapur 2014).

More frequently, TUIs utilize a collection of symbolically imbued objects to create interaction based on their relationship with each other, positioned on a globally situated interface, or as "token-constraint" systems (Shaer 2015, 50). An example of a token-constraint system is Durrell Bishop's *Marble Answering Machine*, a telephone answering machine where recorded messages are represented by colored marbles. The user can listen or delete the message, and call back the sender by placing the marble in different locations, defining the marbles as the tokens and the various areas of interactivity as the constraints (Ishii and Ullmer 1997, 236).

For musical applications, there are many approaches for TUI application. "*Instruments* … that are fully controllable sound generators or synthesizers, *sequencer* TUIs that mix and play audio samples, *sound toys* with limited user control, and *controllers* that remotely control an arbitrary synthesizer" (Shaer 2015, 40). Martin Kaltenbrunner has collected many examples of TUIs for musical purposes his website (Kaltenbrunner 2015), including tabletop surfaces such as the *reactable* (Kaltenbranner, et al. 2006) and *Audiopad* (Patten, Recht and Ishii 2002), modular sequencers using a "building block" metaphor, such as *Block Jam* (Newton-Dunn, Nakano and Gibson 2003) and the *Siftables Sequencer*

(Merrill, Kalanithi and Maes 2007), and token-constraint systems such as *BeatBearing* (Bennett and O'Modhrain 2008) and *spinCycle* (Kiser 2006).

2.2.2 Tabletop Interfaces

The origin of direct-touch devices can be traced to Samuel Hurst, who developed the first computer display that incorporated a transparent surface sensitive to touch in 1971 (Müller-Tomfelde 2010, 7). Through the 1980's the technology was still relatively limited, and the role of desktop computers was still coming into focus, as interface designers "turned the physical desktop into a metaphor for the Graphical User Interface (GUI) rendered on the vertical computer screen" (Müller-Tomfelde 2010, 1). Interest in developing touch-based tabletop interaction really started in the 1990's, when reliable methods of touch technology became available. Through the 2000's they became increasingly popular, to the today where commercial touch surfaces are available for musical performance, and frequently employed in interactive exhibits and installations.

Many tabletop interfaces use tangible objects as a means for interaction (Shaer 2015, 15). Examples of tangible tabletop interfaces include the *reactable* (Kaltenbranner, et al. 2006), which has gained notable recognition, in part due to popular musican Björk using one for her 2007 would tour (Shaer 2015, 41), the *Bricktable* (Hochenbaum 2010), *Audiopad* (Patten, Recht and Ishii 2002) and *smallfish* (Fujihata, Furukawa and Münch 2000).

Fiducial tracking is a common method for tracking tangible components, although some tables make use of additional sensing, such as *Instant City,* which uses light sensors to gather information from the vertical arrangement of translucent stacked blocks. *Instant City* is also an example of a collaborative tangible composition system. Compositions are comprised of different sets of tangible building blocks which performers use to build structures on the surface. Each composition involves unique sounds design and techniques for processing input data, creating distinctive aesthetics and modes of interaction in each (Hauert and Reichmuth 2003).

2.3 Conceptual Influence

2.3.1 Composing with Indeterminacy

In composing *dynamic scores*, especially ones that possess nonlinear trajectories, the question of instructional constraint becomes vital to the way the score unfolds temporally. Namely, the role of the performer in the score's dynamic qualities is an important compositional consideration. *Indeterminacy* is one technique that can be leveraged to create nonlinear dynamics in graphic scores. Relinquishing control is part of Cage's notion of indeterminacy, which he regularly employed by leveraging random procedures to determine compositional variables (Husarik 1983), and leaving aspects of the composition open to the performer (Miller 2009). Nonlinear dynamics could leverage pseudorandom numbers, complex physical models, or implicit qualities of performer's actions to determine micro and macroscopic changes in the structure of the score. Another approach is to provide the performer with a set of symbols for explicit control over these changes.

Examples of indeterminacy being used in tangible tabletop interfaces includes the aforementioned *Instant City*, each performance contains elements of indeterminacy in the way performers arrange the blocks (Hauert and Reichmuth 2003), and the *Xenakis* interface, which utilizes probability based musical composition through tangible objects (Bischof 2008).

2.3.2 Lines and Traces

> Wayfaring is the most fundamental mode by which living beings … inhabit the earth … The inhabitant is … one who participates from within the very process of the world's continual coming into being and who, in laying a trail of life, contributes to its weave and texture (Ingold 2007, 81)

In Tim Ingold's *Lines: A Brief History,* he discusses the metaphorical significance of line in the way it relates to human movement, epistemologies and ecologies. *Wayfaring* is likened to that of a *trace,* a continuous gesture consisting of development *along* a surface, be it pen on paper or one walking down a forest path. He contrasts this to destination-oriented *transport*, a point-to-point modality with the goal of carrying *across.* In the process of transport, the destination marks a point of completion, and results in a release of tension, while the opposite is true for wayfaring, in which a pause marks a moment of tension (Ingold 2007, 77)

The process of wayfaring and transport yield two differing epistemologies. *Occupant* knowledge is *upwardly* integrated through the process of transport, and concerned with distinct features without regard to how one arrives. *Inhabitant* knowledge is *alongly* integrated in the process of wayfaring. In other words, “the wayfarer knows as he goes” (Ingold 2007, 89). Ingold argues that “the ability to use one’s knowledge effectively,” requires repeated but *continuous* action. Skill does not emerge from knowing the steps exist, but through the process of repeatedly connecting them. This assertion draws the distinction between *know-how* and knowing *what*, a distinction Ingold poses to make a point about human ecologies, illustrating the way many architectural infrastructures and transport systems in modern society have been designed for occupants, and not the wayfaring inhabitant (Ingold 2005).

This distinction extends to the practice of temporal-based art forms, particularly in relation to the process of notation. A storyline, or melody goes *along;* it does not exist but occurs (Ingold 2007, 89). Ingold points out that as western music notation has evolved it has shifted to notating aspects of the performance, instead of the music itself, becoming an aspect of what is performed, not of how it is performed. “As musicality is transferred from its verbal to melodic aspect, music was detached from the bodily gestures involved in producing it” (Ingold 2007, 33). Graphic notation has the potential to do the opposite, to instead embody *how* the music is performed, by representing a gesture or movement.

2.3.3 Noise and Glitch Aesthetics

“Noise cannot be made a phenomenon; every phenomenon is separated from it, a silhouette on a backdrop, like a beacon against the fog, as every message, every cry, every call, every signal must be separated from the hubbub that occupies silence, in order to be, to be perceived, to be known, to be exchanged.” – Michael Serres (Cox 2009).

The aesthetics of my work, both sonic and visual, are deeply inspired by noise and glitch art. This interest inspired my choice of sonic material and decision to use granular synthesis. The history of noise music dates back at least to Russolo’s 1913 *The Art of Noises* (Russolo 2004), and extends through the 20th century, including musique concrete, industrial and glitch based electronic music, hip hop and sample culture, to the present, when it is common for noise elements to be included in a wide variety of musical spheres, ranging from popular music, to academic pursuits, to underground efforts and those working on the fringe.

In *Sound Art and the Sonic Unconscious* Christoph Cox provides a philosophical grounding for noise art. Discussing the notion of noise in relation to signal, he illustrates how “there is no absolute structural difference between noise and signal. They are of the same nature. The only difference which can be logically established between them is based exclusively on the concept of intent on the part of the transmitter" (Cox 2009, 20). This leads to his notion of the *auditory unconscious,* the idea that noise is continuous and ever-present, forming the background of our perception. It is through our cognition that we filter this noise to recognize distinct sounds. Sound art has the potential to reveal to the observer (both artist and audience) qualities of noise which otherwise might be trivialized or ignored. This concept is encapsulated well in Cage's realization that there is "no such thing as 'silence'—that, as human beings, our sensory perceptions occur against the background noise of our biological systems," up through current practitioners of noise and sound art (Cascone 2000, 4).

Another aesthetic I find particularly interesting is *glitch art*, a practice philosophically related to sound art, involving the intentional introduction of failure into a system. As Nick Briz puts it, "Glitch art is the aestheticization, recognition, and/or instigation of a glitch, a break and/or disruption of the expected flow of a system" (Briz 2010). Essentially, it is willful creation through destruction – a process that has the potential to make one aware of the implicit and political underpinnings of the digital and otherwise electronic tools which we use frequently.

A glitch happens in between spaces; between signal and interpreter. For instance, in a digital setting, the same "glitched" image file might produce significantly different results based on the method used to corrupt it, and the program used to interpret it. This is illustrated in the way the same glitched image file will appear differently when opened in Adobe Photoshop, versus Microsoft paint. From this perspective, the glitch is a process, occurring between the corrupted file and the container or framework which interprets it. Glitch art reveals the unseen structures of the hardware and code defining our tools, structures which are equally peripheral to our perception – if not more so – than the habitual noise floor of our auditory unconscious.

Drawing from these concepts, I gathered glitched sample material, as well as audio samples which traverse the spectrum of signal and noise. The primary glitch technique I used involves reading non-audio files, such as .exe or .dll files, as if they were audio files in Audacity. The result is a myriad of static, beeps, clicks and pops, ready for use as sample material. Another process involves "digitally circuit bending" ATARI 2600 ROMs, emulating the sound of an old video game console crashing. I also use archival samples of various transmissions, such as shortwave number station transmissions (Irdial-Discs 2012), and samples of audified data from NASA satellites. In these samples, the signal often becomes enveloped in noise, producing interesting sonic charateristics.

To manipulate this sample material, I use granular synthesis, or what Curtis Roads calls *microsound.* "Microsonic techniques dissolve the building blocks of music architecture … sounds may coalesce, evaporate, or mutate into other sounds" (Roads 2004). It is a process that facilitates the construction of larger complex sounds through the cacophony of many tiny ones. The concept of microsound also resonates with Cox's philosophy on sound art. Cox describes how the sound of the sea "is derived from an infinity of small perceptions (the sound of all the individual waves), which we unconsciously register but do not consciously perceive" (Cox 2009, 22), an idea which is not unlike the tiny grains of sound that constitute granular synthesis.

3. DESIGNING A FLEXIABLE PLATFORM FOR TANGIBLE GRAPHIC SCORE

3.1 Design Considerations

In designing a platform for tangible graphic scores I took an *iterative* design approach, using a framework by Gould and Lewis (Gould and Lewis 1985). An iterative design process involves a cycle of testing and redesign that begins with initial ideation and extends through the development of prototypes and their refinement. Outcomes are not premeditated; instead they emerge through engagement with the design process.

To enable the instrument's use in a variety of musical contexts, it is important that the performer has control over a range of what Meyer calls *syntactic* and *statistical* parameters. Syntactic elements of music are those that can be cognitively discretized and organized into functional relationships, such as pitch (frequency), rhythm (duration) and harmony, while humans generally do not cognitively organize statistical elements, such as dynamics, sonority, tempo and timbre into discrete scales or "proportional relationships" (Meyer 1998). It is not necessary that an instrument access every syntactic and non-syntactic aspect of music, non-pitched percussion instruments being a common example. That being said, when one or more of these parameters are inaccessible to the player, it becomes more necessary to have precise and immediate control over parameters that are available. For example, non-pitched percussion affords greater rhythmic precision and more pronounced articulation of time than bowed string instruments. In order to facilitate control over both syntactic and statistical parameters, I thought the interface could allow for what Schloss refers to as *microscopic* (timbral), *normal* (notes) and *macroscopic* (formal) control. Microscopic and macroscopic control must both be continuous, while normal control is both continuous and discrete (Schloss 2003).

In the case of tangible graphic scores, the performance interface and musical composition are intertwined. Thus, many design considerations pertain to both developing the functionality of the interface, and composing for the system. Scores composed for the system should be able to convey microscopic and macroscopic ideas, not only suggesting gestures and nuance, but also a larger compositional form. Scores should be *dynamic*, evolving temporally, and potentially *interactive*, reacting to the performers decisions. Additionally, their composition should be flexible—a platform for the composition of new tangible scores needs a way to develop new material. Tomás describes this as *replicability*, suggesting the use rapid prototyping and digital fabrication technologies in the development of the systems modular components (Tomás and Kaltenbrunner 2014).

Traditional music notation is primarily symbolic—for both syntactic and statistical parameters, it encodes desired outcomes as symbols for a performer to interpret. Tangible User Interfaces (TUI) similarly tend to afford symbolic interactions—like notes on a staff, they frequently constitute physical proxies of computational objects. Along these lines, Thor Magnusson argues that computer music interfaces are necessarily *epistemic tools* in that their underlying computational logic is always a symbolic language; unlike acoustic systems, our relationship with digital music interfaces is an intellectual one (Magnusson 2009). Magnusson stresses the importance for the performer to reach a place where they don't have to think to perform with their instrument. As Magnusson puts it, "Such bodily incorporation transforms our tools into ready-at-hand phenomena that are not based on symbolic or theoretical relations any more" (Magnusson 2009, 170). This is similar to Ingold's notion of *Inhabitant* knowledge. Skill is developed not through the knowledge of *what* to do, but knowledge of *how* to do it.

Thus, the combination of tangible interaction and music notation pushes strongly in the direction of a symbolic interaction paradigm, a temptation I sought to resist in my own design. Performance practice of graphic scores, open scores in particular, provide models for alternative, non-symbolic interactions: performers can read graphic scores as moods, as metaphors, as prompts, as recollections of past performances, or in terms of gesture irrespective of sonic result. In the composition of tangible scores, it is important to consider how the information is conveyed through material means, both in the way it is encoded, and received by a performer and audience.

3.2 Initial Ideas and Iterations

In this section, I present early artifacts of the iterative process, which remain valuable for both the ways they informed the current design, as well the reasons they were not pursued further.

3.2.1 Disks and Colored Light

My initial direction was inspired by a performance of Dieter Vandoren and Mariska de Groot's *Shadow Puppet* (performed at the 2014 International Conference on New Interfaces for Musical Expression). In that performance, light projected through spinning discs cast pulsing shadows onto a wall, which were sensed by optical sensors to generate sounds synthesized from oscillations in light intensity (Groot and Vandoren 2013). This performance had significant relevance to Theremin's *Rythmicon*, an instrument designed in collaboration with composer Henry Cowell in the early 1930s. The rythmicon was created to realize precise polyrhythms, which Cowell had been having difficulty getting musicians to accurately reproduce. Pressing a keyboard illuminates bulbs which shine light through regularly spaced holes in a set of rotating wheels (one for pitch and one for tempo). The resulting patterns of light are then picked up using a photodetector and translated into precise polyrhythmic pulses (Schedel 2002, 248).

Recollections of the performance, and inspiration from Theremin's early electronic instrument led to the idea to create graphic scores by projecting light through overlapping discs of colored films. The discs would be motorized, and possess controls to start and stop them through capacitive touch

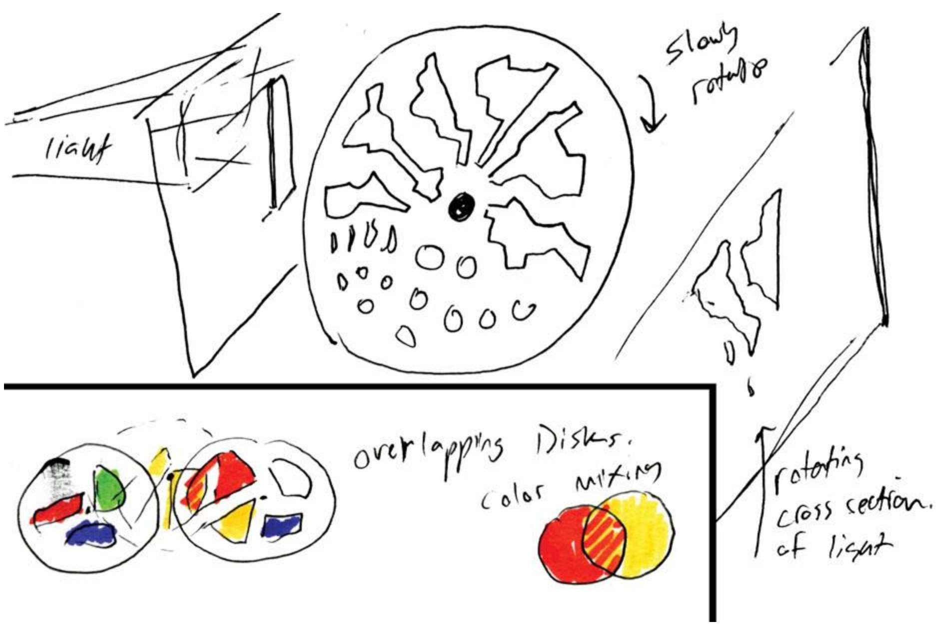


Figure 4. A sketch involving illuminated rotating disks and

sensing. The movement of the discs would be tracked by reading a quadrature pattern with a pair of reflective optical sensors. Information of their position and motion would allow for dynamic relationships between the discs' movements. A performer could either use a sensing device allowing the colored projections to drive computer sound synthesis, or interpret them as performance instructions, in the manner of traditional graphic score (Figure 4). The compositional process would be a combination of both creating the discs as well as programing their interaction. While this concept was compelling, it posed many issues, both in terms of implementation and interaction. Previously I had encountered challenges with the range and sensitivity color sensors. More fundamentally, however, light is by nature intangible, so interacting solely with projections seemed to violate the premise of *tangibility*.

3.2.2 Textured Modular Surfaces

Another concept stemmed from the desire to add a tangible quality to musical notation, while simultaneously leveraging the dynamic qualities of projected visuals. A textured panel would be used as the basis for a multi-touch surface, allowing for both textural and visual compositional tools to be utilized in tandem. This method would be distinct from Tomás and Kaltenbrunner's engraved scores, in

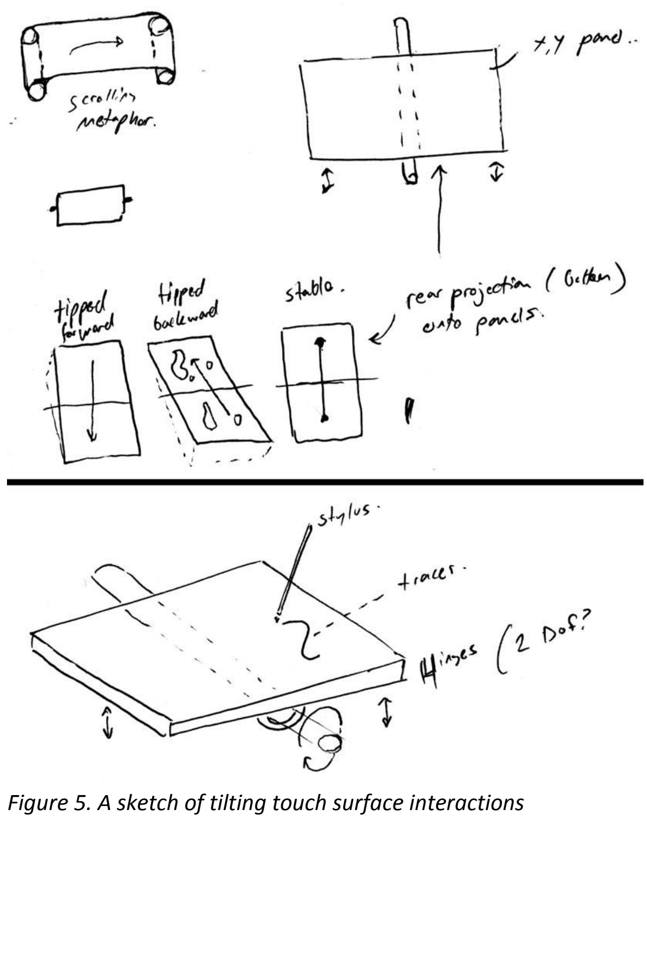


Figure 5. A sketch of tilting touch surface interactions

its use of visual sensing instead of acoustic, and the addition of dynamic visual feedback and stimulus (Tomás and Kaltenbrunner 2014).

Inspired by the way turntablists continually swap out their tangible performance surfaces (records) throughout the course of a performance came the concept of *interchangeable surfaces*. Textured panels would serve as the basis for compositions, or sections of multi-movement pieces. By encoding unique IDs into the plates, the performance system could adapt dynamically to the switching of surfaces, and respond by appropriately changing visual feedback, or completely altering the mode of interaction. Creating a modular system of panels would also facilitate nonlinear compositional forms.

A metaphorically compelling variation of the textured touch surface was to allow the surface to partially rotate around one axis with at least one degree of freedom (Figure 5). Responding to the performer tilting the surface, projected visuals could be animated according to gravity, in a similar fashion to a tilt maze. With the addition of a motorized axis, the temporal unfolding of a score has the potential to be represented visually and kinetically, as graphics slide in and out of the performer's access.

Inspired by Ingold's writing, I began thinking about ways to incorporate *traces* into the concept of textured surfaces. I was particularly interested in the idea of navigating a textured surface via continuous gestures and trajectories. The topography of the texture could either naturally alter incoming sensor data, or exist as a digital representation to create a space for sonic exploration. I also considered different forms for the surfaces, such as an array of rotating cylinders, and how this might lead to continuous paths that become fragmented and diverge as the configuration of the surface shifts (Figure 6).

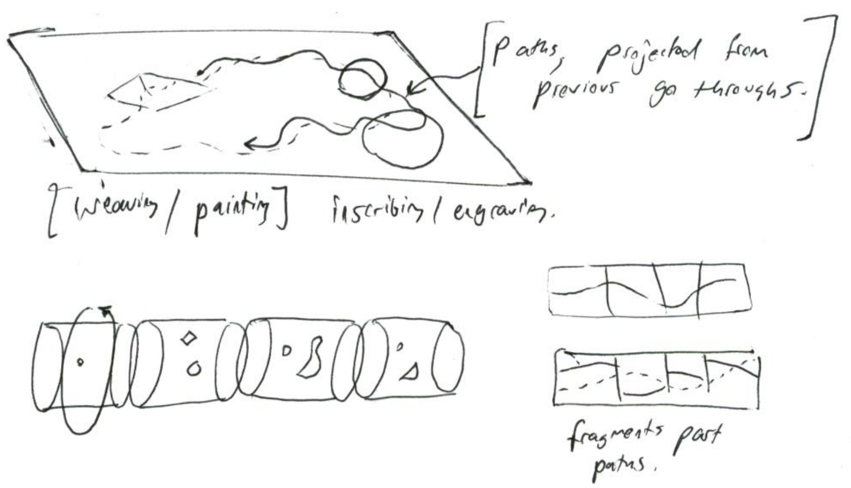


Figure 6. A sketch of traces incorporated with various textured surfaces

An animated component could be projected onto a surface as an interactive component to the score, providing visual cues, feedback and working with texture to create a hybrid score. An early sketch shows an idea for abstract glyphs and textures through which the performer's traces could determine musical material (Figure 7).

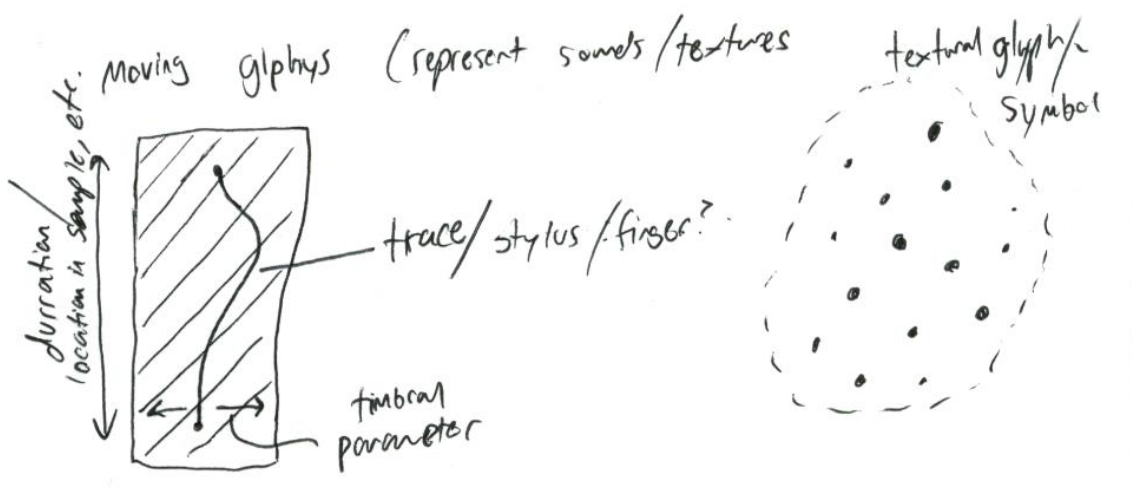


Figure 7. A sketch of traces interacting with visual symbols

3.2.3 Final Direction

The final direction drew upon a number of the aforementioned ideas, namely that of utilizing interchangeable textured surfaces as tangible scores. The concept I chose to pursue involved creating a touch surface in which the user could place different textured plates which would serve as the tangible component of the composition. Using infrared tracking and computer vision the user's movement on these surfaces would be sensed and interpreted by a computer, projecting visuals onto the rear of the texture, and producing the sound of the instrument.

This direction was chosen for its feasibility, replicability of tangible components, and potential for future expansion. Before attempting an interface involving a rotating, projection mapped textured surface, it seemed appropriate to focus on first designing a system that worked with textures and projections. If this system proved successful, it might have the potential for expansion into an interface with further kinetic affordances.

Hardware Design

The design of this system consists of a raised platform, upon which different textured panels can be exchanged. Four infrared (IR) lasers create a Laser Light Plane (LLP) over a transparent surface, and an IR camera detects the performer's interactions with the surface, as they reflect light down onto a diffusive material below the surface. The desire to texture the surface informed the decision to use an LLP approach, as it does not require the touch surface to facilitate internal reflections (Sandler 2015). A projector mounted in the base provides the visual qualities of the score and interaction (Figure 8). Further details of this type of low cost multi-touch interface can be found in *A Low-Cost, Low-Latency Multi-Touch Table with Haptic Feedback for Musical Applications* (Montag, et al. 2011).

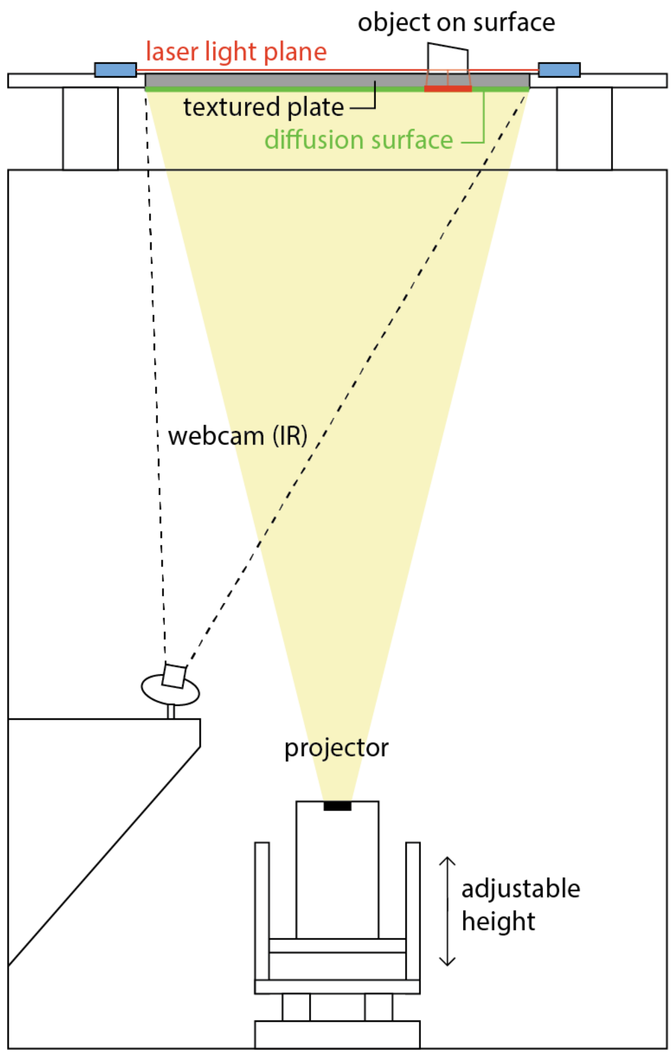


Figure 8. A side view illustrating the system's functioning

In the design of the system, I leveraged digital fabrication procedures such as laser cutting and CNC routing to inexpensively develop new physical components and to modify existing objects. Figure 9 shows the primary designs for said digital fabrication technologies. For the base of the device, I found an existing wooden box at a local reuse center, removed the top, and CNC routed an opening for the projector light and webcam sensing, along with holes for PVC to mount the raised platform (in green).

The raised platform is constructed of laser-cut medium-density fiberboard (MDF). Modular plates fit into a central 8 ½ x 11 hole (in black), laser mounts attach to holes positioned at the corners of this hole (in blue), and raised edges prevent the laser light from extending off the edge of the table (in purple). A set of adjustable rails drop extend down from the raised platform, allowing a diffusion surface to slide in and hold the modular surfaces plates (in purple). This sliding screen was added to improve the previous diffusion surface method of attaching frosted velum sheets to the back of every modular plate.

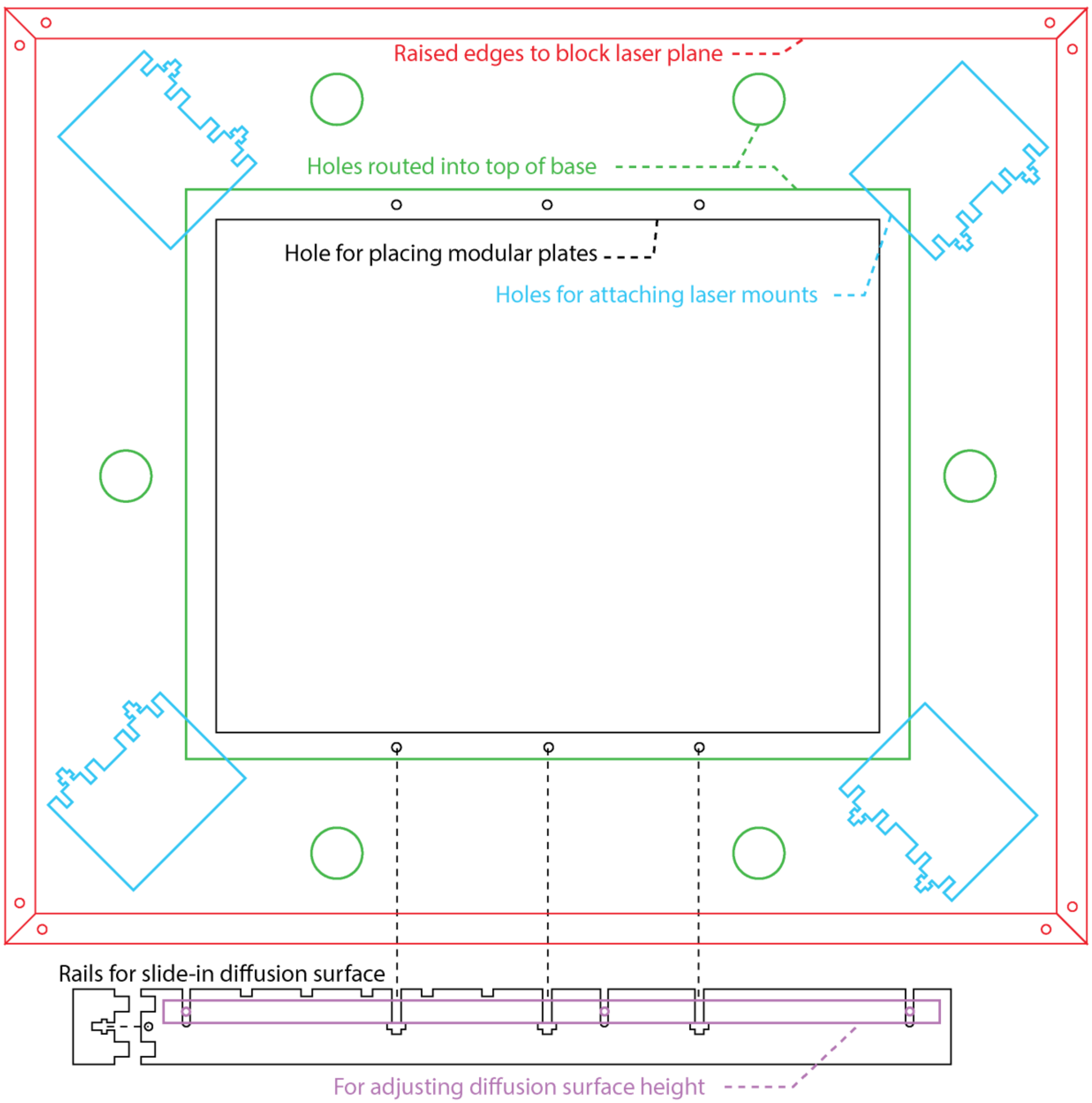


Figure 9. Laser cutter and CNC router designs

850 nm infrared (IR) lasers, along with corresponding laser line generators (for splitting the laser into a plane) were used to create the plane of IR light over the surface. Peau Productions, a company specializing in equipment for building touch surfaces, produces aluminum mounts specifically for these lasers (Peau Productions 2015). I designed a system from laser cut acrylic to attach the laser mounts to the MDF surface, allowing the height of each laser to be adjusted for calibration purposes. An LED was added to each mount to provide visual confirmation of when the lasers are on (Figure 10 and Figure 11). The projector is attached to a laser-cut acrylic mount, providing adjustable height.

The infrared camera was made by hacking an inexpensive computer webcam. After opening the camera, the IR filter was removed, and replaced with a small piece of floppy disk film. Instead of blocking infrared, the floppy disk film passes IR light and blocks the visible spectrum. The projector mount is fixtured in the bottom of the base on top of a PVC structure in a way that it can be removed to adjust the projector (Figure 14 and Figure 12). Figure 13 shows the final system, complete with a textured surface (these will be discussed in the next section).

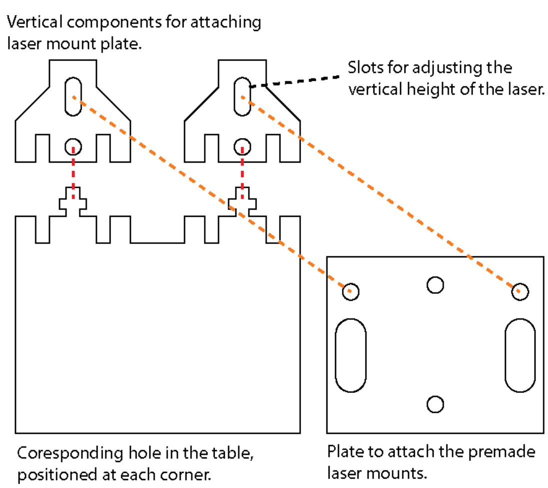


Figure 10. Laser cutter designs for the laser mount

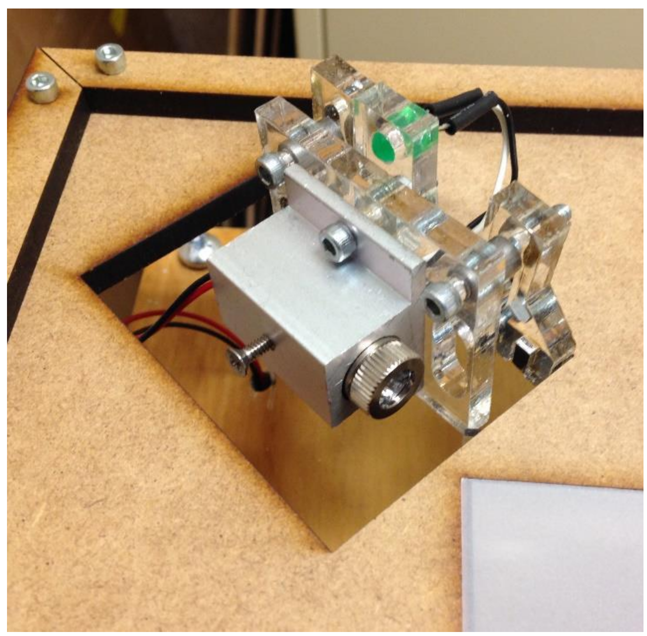


Figure 11. Photo of the finished laser mount

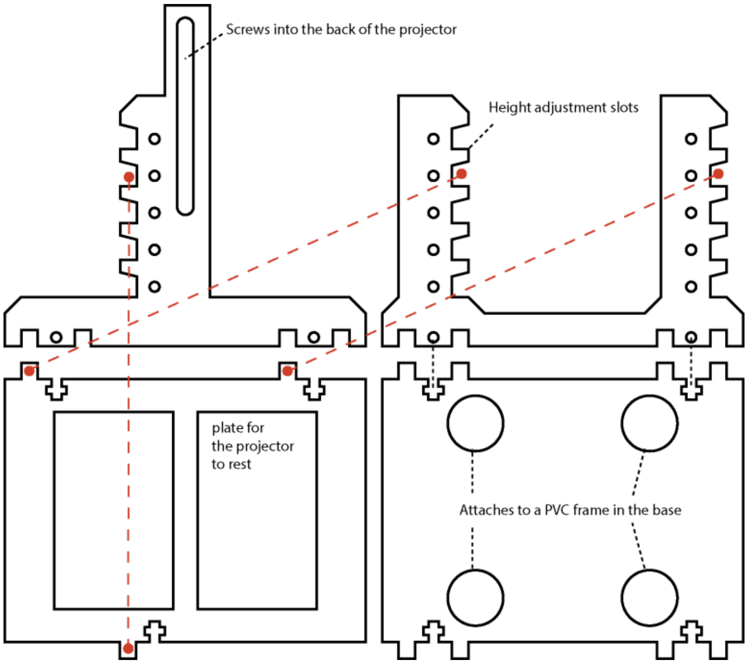


Figure 14. A laser cutter design of the projector mount

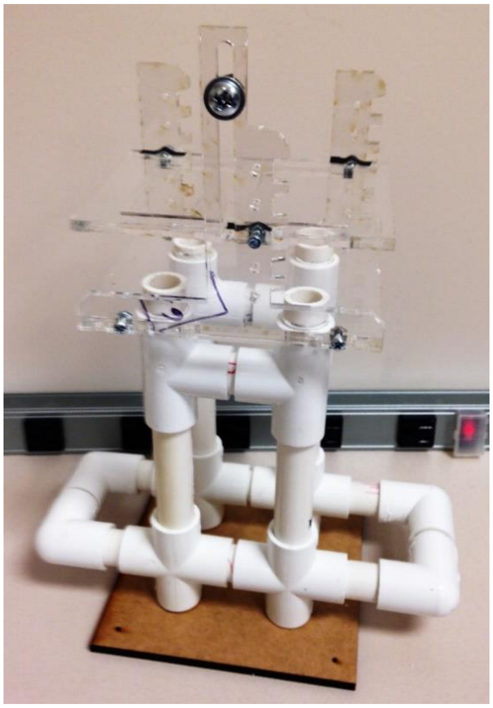


Figure 12. A photo of the projector mount

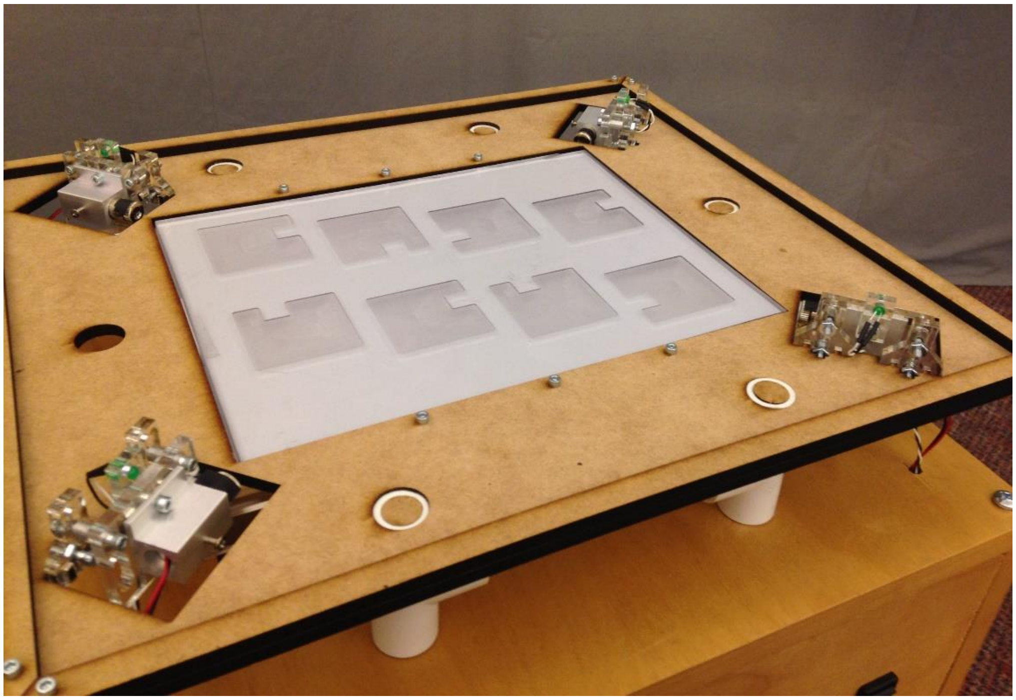


Figure 13. The final system, with textured surface

3.3 Textured Plates

A number of different methods were considered for creating the textured surfaces. The surfaces needed to transparent, to allow projections and infrared light pass through them, but also possess textural variety. They also needed to be fabricated in a way that was inexpensive, easily replicable and that wouldn't inhibit the IR sensing. One thought was to engrave acrylic with a laser, however this process lacked in textural variety, and didn't allow for much depth control. Another approach involved topographically layering thin pieces of acrylic or vellum, a process that would have likely led to fabrication issues resulting from the thickness of material (in the case of acrylic), and difficulty in holding layers together (in the case of vellum).

The final textures were created by CNC routing pieces of ¼" polycarbonate. Acrylic is brittle, and would likely shatter in the case of CNC routing, however polycarbonate is much stronger and has the potential to be machined. To test the process, I used a topographical map of Titan (one of Saturn's moons) as the basis for a score, setting different levels of the map to different depths for the router. I also created a tool path to cut the plate into the appropriate 8 ½ x 11 size. The result was a textured plate of polycarbonate, in the correct size, however, the plate had significantly warped in the fabrication process! Figure 15 shows the design side by side with the test plate, Figure 16 includes photos from the fabrication process.

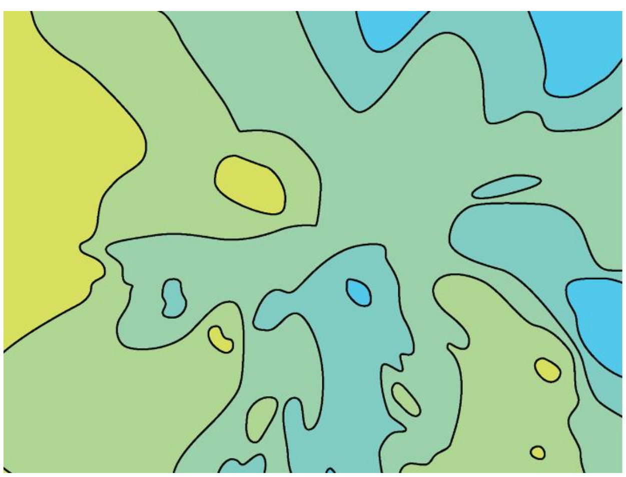
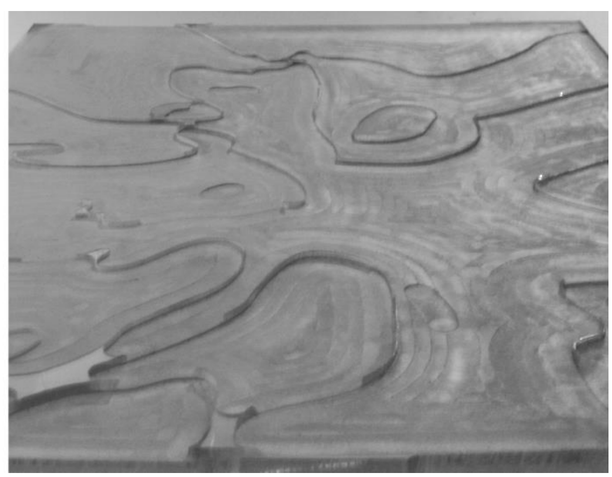



Figure 15. The design and finished test plate

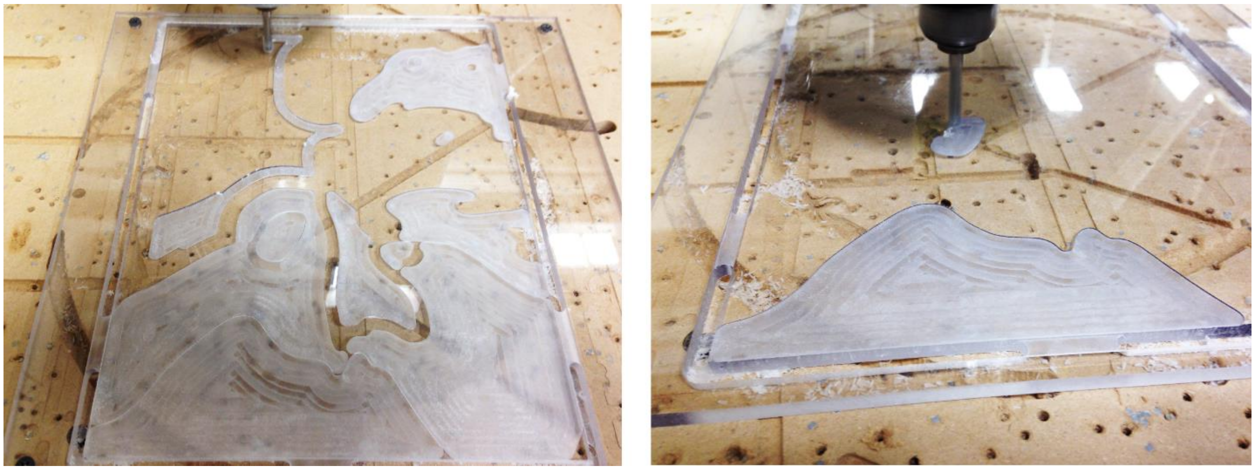


Figure 16. Photos from the fabrication process

The warping likely occurred through disrupting the surface tension of the polycarbonate in the CNC process. Polycarbonate is often used for its ability to reform when heated, suggesting that heating it might aid in flattening it. One attempt at correcting the warping involved heating it for a few in an oven at 200 degrees F. A large rock was placed on the plate, pressing it flat against a cookie sheet. It seemed promising at first, however the plate remained warped after it had cooled. This initial design had machined almost the entire surface of the plate, ranging from .02” to .08” in depth. The most compelling aspect of the plate was not its depth, but instead the contrast between the machined sections, and those left alone, a difference which created a distinct differential in texture. In an attempt to prevent further warping, the second design left the majority of the surface un-machined, and kept the cut depth to 0.02”. Another precaution against warping was to cut the plate out with a table saw equipped with a plastic cutting blade instead of using the router. The result was an un-warped plate.

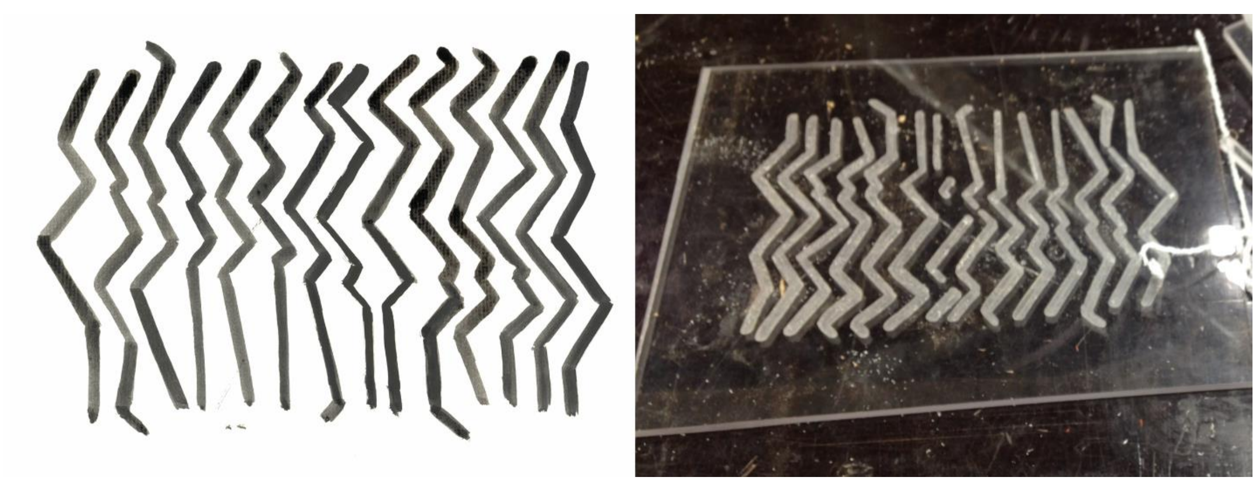


Figure 17. The original design of the second, and successful attempt at creating a textured plate

I based these new plates on ink paintings I made with the intention of translating them to textures (Figure 17). Figure 18 shows some of the ink paintings created to be rendered as textures, while Figure 19 shows the nine different textured surfaces that have been created for the instrument. The first five are employed in existing compositional systems, while the last four were are currently unused. The textured plates used in specific compositions will be discussed throughout section 4.

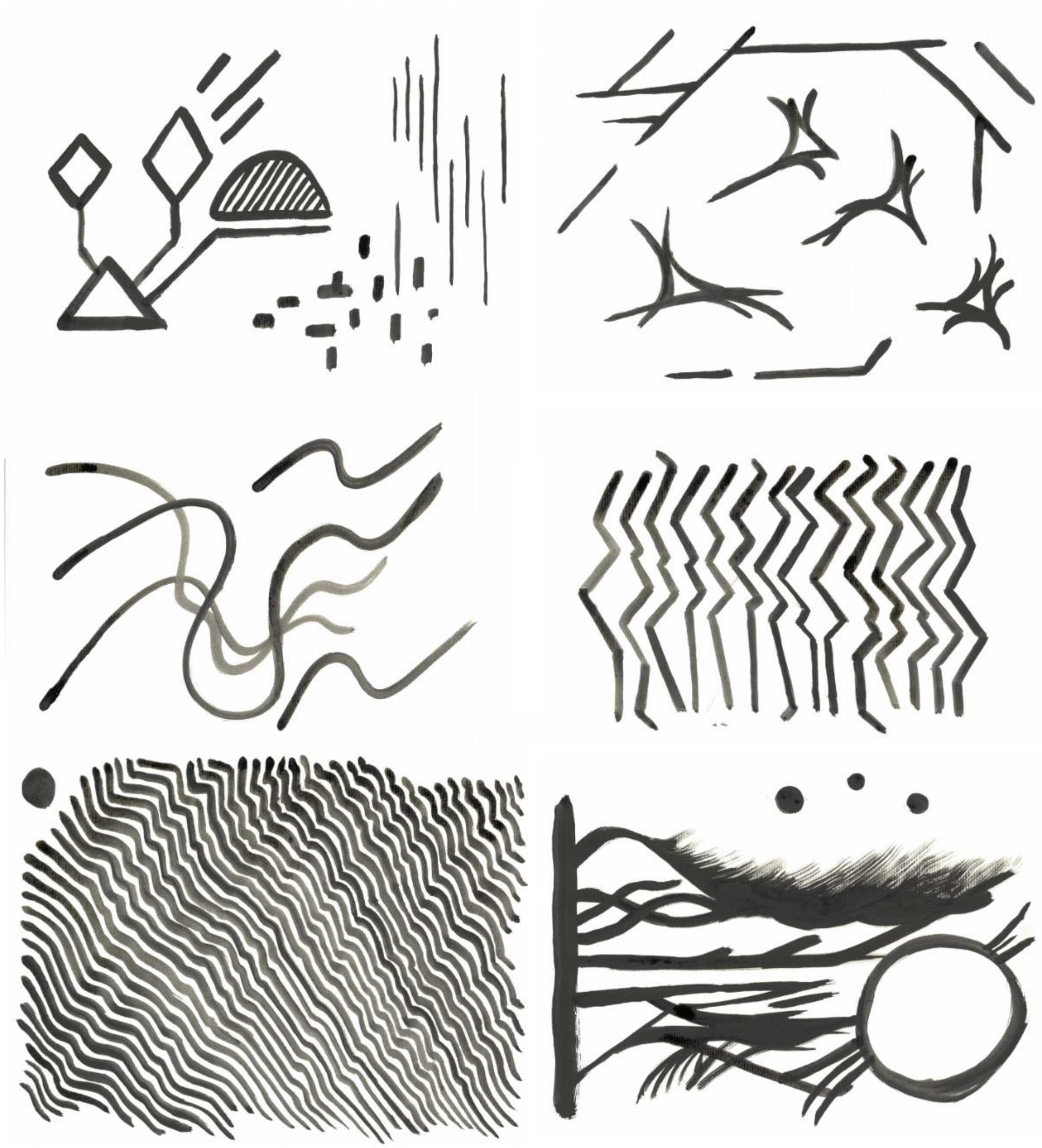


Figure 18. Ink paintings conceived to be rendered as textures

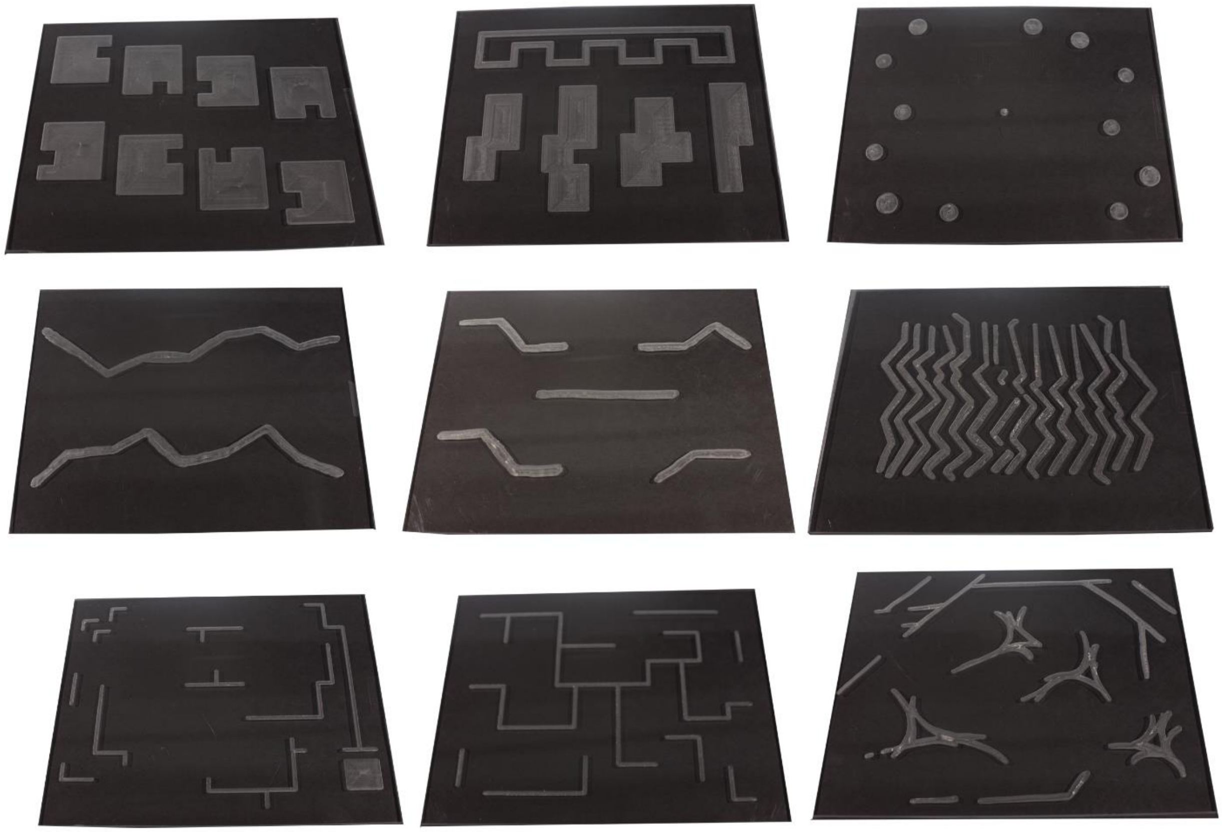


Figure 19. The nine textured surfaces created so far for the instrument

3.4 Analyzing Camera Data

The prototype for the system used X-Y finger positions, determined by taking blob centroids, to control parameters of a polyphonic FM synthesizer in Max. While this is a useful approach for productivity applications with traditional multi-touch surfaces, I wanted to avoid using blob tracking to create a simple XY controller or reciprocating common GUI items such as sliders, faders and buttons, a type of interaction which doesn't suit my desire for non-symbolic interaction (as discussed in section 3.1). Blob detection is an effective system for extracting the X-Y position of finger presses, but in exchange for its precision, it constrains the gesture space to just that: finger touches. Two common uses for multi-touch surfaces—as 2 degree-of-freedom X-Y controllers, and as quantized grids (essentially a low-resolution matrix of 1 degree-of-freedom switches) — are derived from a Graphic User Interface (GUI) paradigm (McGlynn 2012).

Beyond X-Y position, blob tracking can yield a number of other characteristics such as the size and direction of blobs, and their inertial movements. It remains a practical method for the isolation of the user's interaction, and this reason it was used for a number of the compositions using the jit.cv computer vision Jitter externals (Pelletier 2013). After successfully implementing jit.cv to control FM synthesis in the prototype, I shifted to using the built in blob tracking operatior in TouchDesigner, which uses the source from openCV (OpenCV 2015), while utilizing the GPU to improve performance. This provided the benefit of keeping the blob tracking data and visuals in one location. For the final system, blob tracking was switched back to Jitter due to inconsistencies and excessive latency occurring in TouchDesigner. Jit.cv was a more consistent and faster method, and allowed for a more diverse set of features to be extracted. This switch was partly facilitated by recent developments in Spout, a piece of software which sends of real time video streams between software. Open Sound Control (Wright 1997) is used to send blob data from Jitter into TouchDesigner.

While working with blob detection in TouchDesigner, I designed a system for isolating each blob to analyze its visual content individually. Blobs are cropped automatically using bounding box information. Using a replicator operator, the process is dynamically instanced for the current number of blobs being tracked. While this element was not used in any finished compositions, it did serve as a model for the "hover" technique used for adding additional controls to the compositions (discussed throughout section 4).

I had interest in making direct use of the tangible qualities and inconsistencies created by the texturing of the surface, instead of relying solely on blob tracking data. I hoped to involve the camera data directly in the sound synthesis, using raw pixel data in a way similar to McGee's method of image sonification (McGee 2013). While this type of synthesis was not integrated into any of the compositions, it did inspire the image analysis method described in section 4.1.1.

3.5 Sound Generation

The first prototype controlled FM synthesis parameters using raw XY finger positions. This worked as a proof of concept, however I desired a richer timbral palette than FM synthesis allowed. As mentioned in section 2.3.3, I was interested in utilizing granular synthesis as a means for sound generation. Instead of developing a granular synthesizer from the ground up I decided to find an existing example that I could modify to take input from my instrument. Robert Henke's *Granulator* (Henke 2015) was my first choice for a granular synthesis engine, as I frequently use it in my electronic music compositions. I ultimately decided against using it, as it would have required porting a Max for Live over to Max, which didn't seem like it would be a straightforward process compared to finding an existing engine base din Max. I narrowed the selection down to Håkon Nybø's *77_GS* (Nybø 2014). *77_GS* doesn't possess a number of features that made Granulator specifically appealing, such as a spray parameter, LFO and FM synthesis modulation and a wide variety of grain window sizes, however it does have a few features that Granulator doesn't, such the ability to cross fade between two audio buffers, and global preset interpolation to facilitate smoothly transitioning between settings.

77_GS uses the poly~ object in max to manage polyphony, allowing new note values to access unused instances of the synthesis engine. Parameters are routed using messages formatted as "ParameterName value." The existing method for parameter control only allowed for global control, meaning parameters controlled every voice simultaneously. Global parameter control is ideal for a keyboard controller situation, where modifying individual note parameters would be unwieldy for a performer. By slightly modifying *77_GS,* I was able to manipulate the parameters individual voices separately.

3.6 Software Communication

Figure 20 illustrates the final software communication, between Max/MSP/Jitter, TouchDesigner, and the hardware input and outputs. Video from the infrared camera comes into Max and is processed by Jit.cv to extract blob information. This data is sent to TouchDesigner over OSC, along with the raw video over Spout. TouchDesigner processes this input to sort the blob data, and filter noise from the camera data, passing appropriate data streams to the selected composition module. The functioning of individual composition modules will be discussed in section 4. Compositions output OSC data back to Max for the purpose of audio synthesis, and video to the projector mounted in the instrument for visual feedback. Additional video outputs can be sent to external displays and projectors for the sake of an audience. Audio from max is sent to an external audio interface.

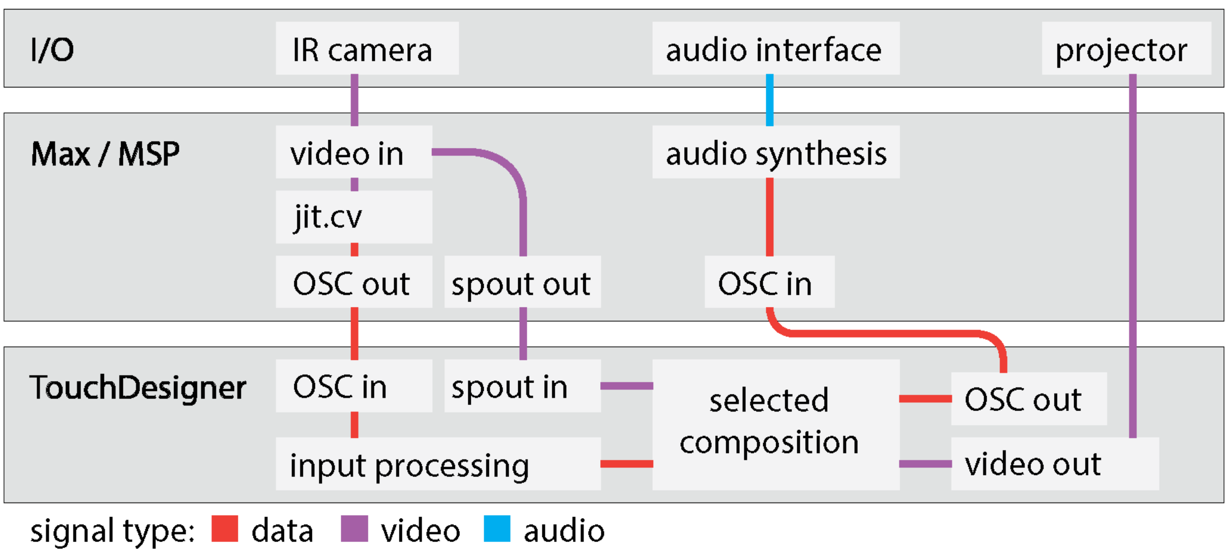


Figure 20. Software communication

4. THE COMPOSITIONAL PROCESS

In traditional composition, melodic, harmonic, rhythmic, timbre, textural or conceptual ideas are all possible points of departure. As the composition unfolds, the relationships between these areas help inform its development into a cohesive piece. Similarly, in the case of tangible scores, the compositional process may begin at many points, with the additional of disciplines unusually unrelated to music composition. The fabrication of textured surfaces, interaction with the temporal projected components and sound synthesis, and basic mode of interaction are all points of departure in addition to those related to the musical aesthetics and form.

The biggest challenge in composing for this system was the initial lack of constraints. Magnusson cites Margaret A. Boden, describing "constraints as one of the fundamental sources for creativity: 'Far from being the antithesis of creativity, constraints on thinking are what make it possible. . . . Constraints map out a territory of structural possibilities which can then be explored'" (Magnusson 2010). Constraints are important considerations in the design of any system, be it the physical interactions with an instrument, or the rules governing a system of a composition.

In the case of my platform for tangible graphic scores, compositions possess both designed constraints and those emergent from the innate capability of the system. Narrowing the scope of the different compositions emerged through engagement with these self and system-imposed constraints. Each new system required different software techniques – some utilized blob tracking, while others analyzed other aspects of the raw video. Some concepts incorporated autonomous agents, while others involved particles systems, or were instead focused on capturing traces created by the performer.

One rather significant system-imposed constraint was the latency of the system. At first, I focused on improving latency. Switching to jit.cv for the blob tracking system was partly due to slowness of the TouchDesigner blob tracking. The Microsoft Kinect's infrared camera has significantly less latency than the hacked IR webcam, however, the new version of the Kinect does not allow control of the infrared emitters through the SDK like its predecessor. Covering up the emitters also proved difficult, as both the camera and emitters are placed behind a transparent pane. Eventually, it became clear that I needed to shift my attention away from making improvements to the system, and begin composing in a way that did not require low latency interaction to create musical expression. In this way, latency became a compositional constraint, and was treated as a trait inherent to the system.

This rest of this section documents compositional concepts, discussing both the elements that made their way into the final design, as well as rationale for excluding those not included and fully developed. This includes the design and fabrication of the textures, the coding and refinement of the interactive, visual and sonic components.

4.1 Composing with History

Drawing from the notion of vinyl records being simultaneously tangible scores and records of prior performance, I developed a concept wherein tangible scores would emerge as a product of repeated performance. To do this, I conceived of an iterative system of scores, in which the aggregate of a performance is encoded texturally onto a new surface. The temporal qualities of the performance are then represented through animated visuals projected onto that same surface. As a performer develops their performance practice with such a score, they would be creating both a record of that performance and a score for future performance.

This concept also addresses the interest in avoiding a purely symbolic interaction paradigm. Instead of symbolically representing *what* is required to achieve an intended sonic outcome like traditional notation, our tangible graphic score illustrates *how* to perform by physically embodying previous performances, and displaying a spatio-temporal map of the qualities of those same performances. In doing so, I also aim to avoid the tendency to rely on tangible proxies for symbolic manipulation. A score iteratively constructed from representations of its own performance would offer the performer the ability to explicitly (not just metaphorically) interact with their prior experience. This compositional method also examines the somewhat difficult question of performance history in the development of new interfaces for musical expression, as they are generally developed in a much shorter span than the amount of time many acoustic instruments have had to evolve.

4.1.1 Capturing the Aggregate

The first method developed to record the aggregate of a performance was inspired by McGee's method of image sonification as a source for synthesis (McGee 2013). I designed a visual representation that slowly changes color based on the duration of the performer's presence. That is – the longer a performer kept their fingers in one position, the brighter that area would become. The process, as illustrated in Figure 21, is named heat map for the color scheme used in the first version. The raw IR video is converted to a lower resolution of 44 x 34 pixels in the interest of operating on every pixel. Then the video is converted to monochrome, and sampled to individual channels. Only the red channel is sampled, as the RGB values will be identical in a monochrome image. Each channel is passed through a threshold which triggers a counter to increment when the pixels are above it. When values are below the threshold the count decreases over time at a slower rate. An additional threshold at the maximum triggers a python script which momentarily increases the decrement rate for the counter, allowing the performer to “flush” the counter periodically by hovering in one place. The counter values are then converted to a single channel, mapped to an RGBA gradient, and converted back to a video stream using a GLSL pixel shader (provided by TouchDesigner staff on their forum). The counter values are mapped to colors, progressing from blue, to purple, red, orange and finally yellow for the highest values. The alpha channel is set to a linear progression from 0 to 0.9, which can be sampled later to determine the counter values from the new image data.

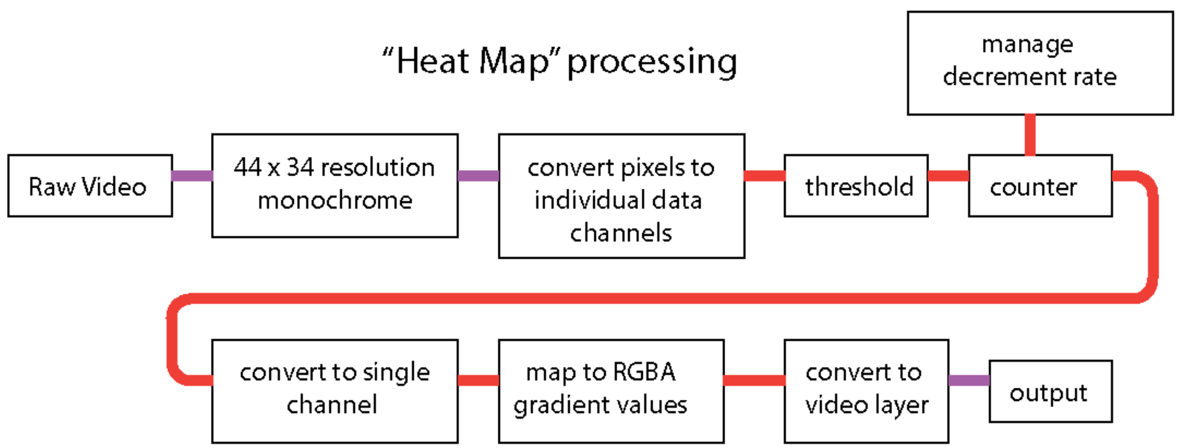


Figure 21. "Heat map" software diagram

The result is a dynamic visualization of the performer's movement – a low resolution aggregate of their positions over time. As the performer repeatedly moves over an area, its color slowly intensifies from blue through the color gradient to a bright yellow (Figure 22). The visuals, while compelling, seemed to capture temporary impressions instead of a macroscopic construction of a performance. Eliminating the decay in the counter values would give an accurate representation of the entire piece, however this method frequently led to a large variance between points with a high and low concentrations of action, eventually reaching a place of stagnation. For this reason I ended up utilizing this process in a different composition, unrelated to concepts dealing with encoding performance history into the surface of textured plates (discussed in section 4.2).

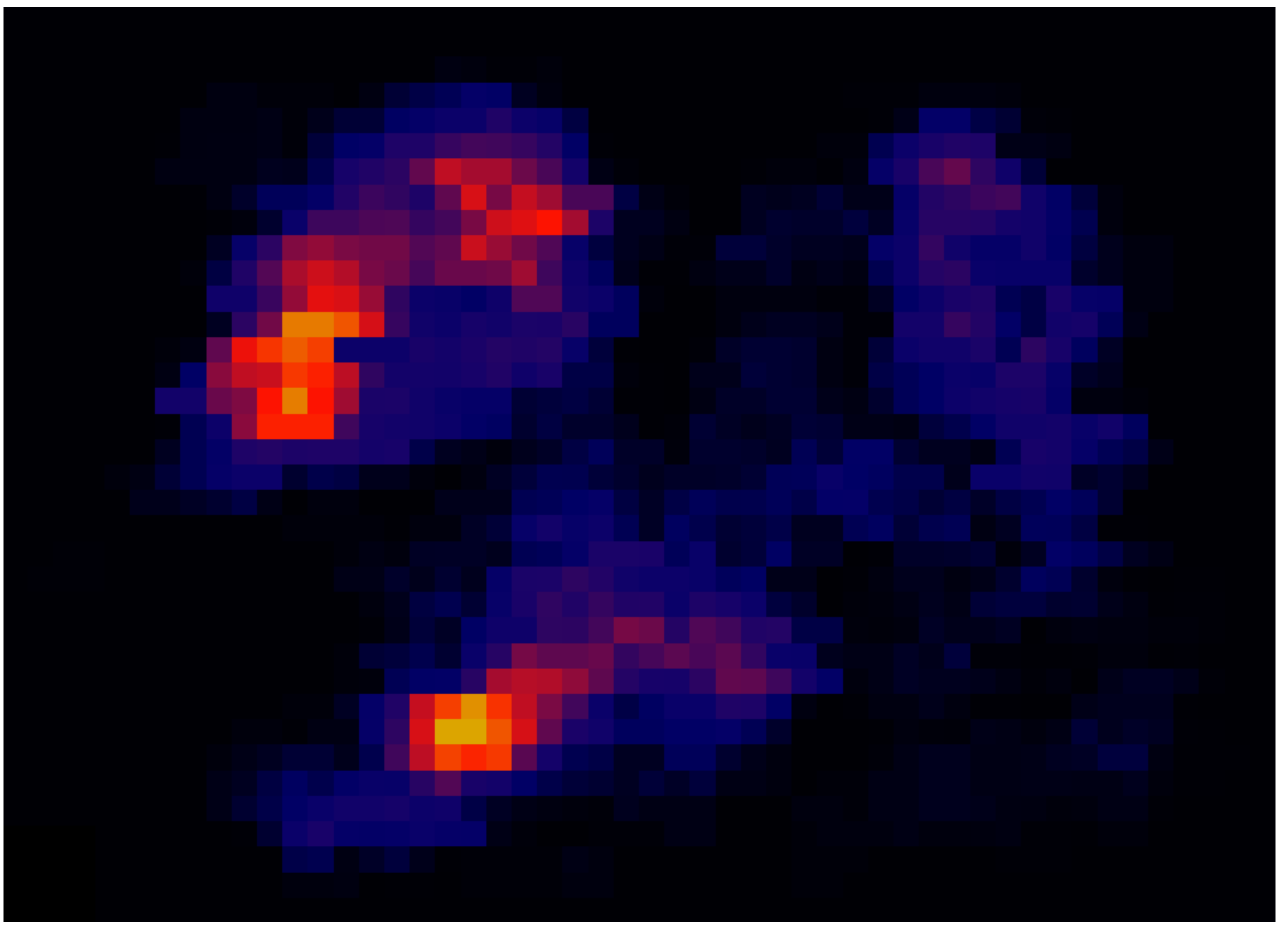


Figure 22. The "heat map" visualization

4.1.2 Recording Traces

The second approach involved recording traces of finger movements. According to Ingold, a *trace* is "any enduring mark left in or on a solid surface by a continuous movement," and is usually either additive or subtractive. The marks on or in a surface are records of the human gestures involved in their inscription; following a trace is not merely a matter of allowing oneself to move along a line, but rather of interpreting its creation (Ingold 2007, 43). The implementation of this idea would involve recording the traces created during a performance, and then using digital fabrication tools to encode this representation physically as a textured surface. This method was closer aligned with the original goal to encode a previous performance into a new tangible score.

To realize this concept, I needed a way to capture X-Y centroid movement, and subsequently format the data for Adobe Illustrator, or another vector software capable of designing CNC router plans. In the initial stages of this process I used the mouse input to quickly generate x and y positions, and captured them with a record operator, creating a sample of X-Y values (Figure 23).

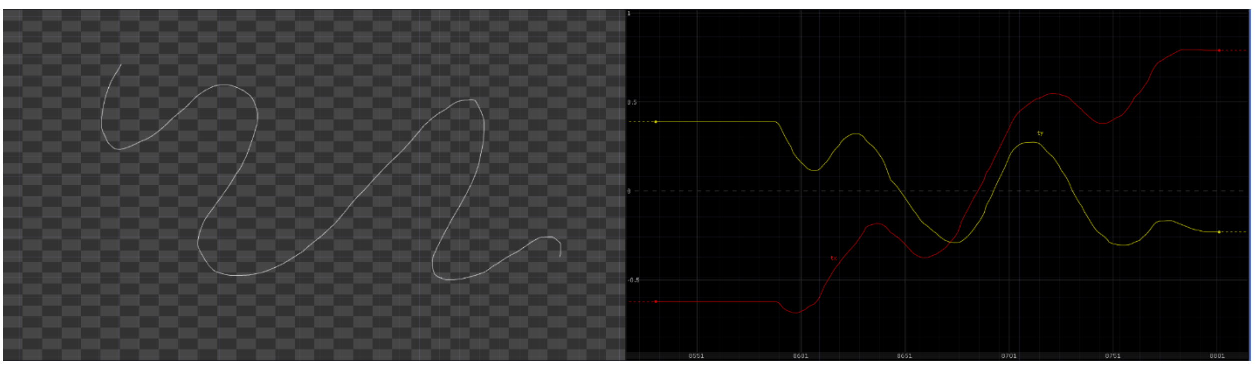


Figure 23. A recorded trace converted to geometry and rendered

After exploring a number of different methods for exporting X-Y data to Adobe Illustrator, I found a comparatively simple way to do it without using additional software to convert between different file formats. The raw X-Y data is exported from TouchDesigner, formatted using a basic find-and-replace text editing tool, and placed inside an existing .svg file, which is then opened in Illustrator. The lines could then be used for the basis of the textural material.

Figure 24 illustrates the line capture system used to record individual line segments extracted from discrete blob centroid trajectories. Data is recorded continuously for a given blob ID, while on and off detection is used to keep track of the start and end points – essentially "punch in" and "punch out" points. Using trim operators the appropriate segments of the sample are individually converted to geometry and rendered, with the possibility of exporting X-Y positions for later fabrication.

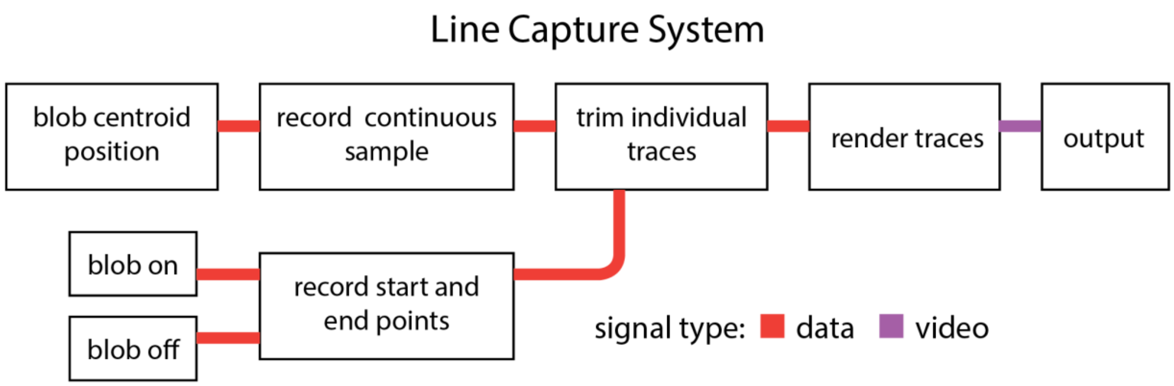


Figure 24. Software diagram for capturing discrete traces from blob centroid positions

I was also interested in potentially using lines as means to let the performer draw the initial state of composition. For example, the performer might draw a number of lines which then become animated autonomous objects for sonic interaction. A system of classification for the lines would allow the user to draw different types of objects, characterized by other parameters of the lines. A few simple methods of classification were implemented. By calculating the distance between the start and end points, I was able to determine when the line was a closed loop. Information about the size, and orientation of the lines was also determined. While this concept did not end up in a finished composition, it did influence the direction for the snakes composition, discussed in section 4.2.

The line capture system was also not used to realize the aforementioned concept of iterative compositions. While the line capture system was fully functional, none of the final compositions seemed appropriate candidates to make use of it. This concept still remains a compelling direction for future work and development of the system.

4.2 Snakes

The *snakes* composition resulted from the desire to use traces drawn by the performer as an initialization state. Traces drawn by the performer could become snakes that would then animate, navigating an environment also created by the performer through tracing closed figures. After this initialization phase, the performer would then interact with the autonomous agents they just created. This concept was intended to leverage the line capture system described in section 4.1.

The first sketche for this composition was based on examples from Daniel Shiffman's *Nature of Code*, (Shiffman, Shannon and Marsh 2012) a book that describes an object oriented approach to creating various autonomous agents in the Processing language. In this prototype, a python script implemented in TouchDesigner generates snakes which move towards a point within a set radius of their current position. These snakes have a number of definable parameters such as acceleration, maximum velocity and turning radius, and bounce of the boundary geometry (Figure 25). An annotated version of the python script can be found in Appendix A.

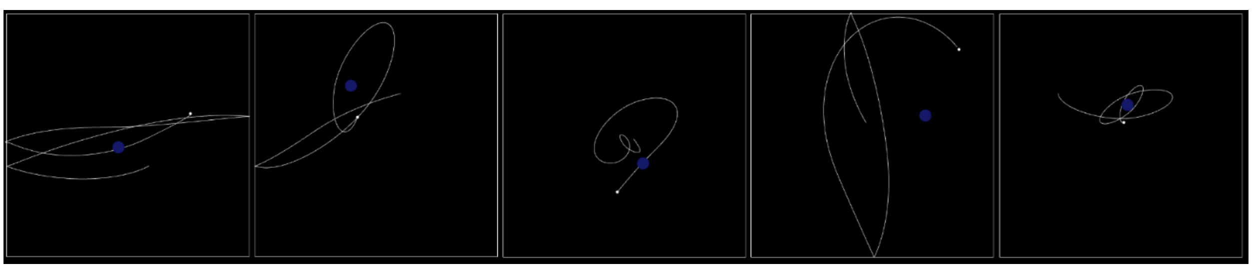


Figure 25. The snake attractors sketch

This first sketch was not used in a finished composition, but did inform the final version of the *snakes* composition, which features virtual snakes moving orthogonally between adjacent cells of a grid. The motivation to switch to a grid based system resulted from the interest to manipulate the movement of autonomous agents using the data generated from the aforementioned "heat map" interaction. Snakes are introduced over time, and move around passageways corresponding to reciprocal patterns engraved into the textured surface. They trigger sounds based on the color of the square they inhabit, their X-Y position, and unique banks of parameter sequences. The colors from the heat map image determine when the snakes are sending sound, the cutoff frequency of the sounds, as well as the speed to the snake, creating fluctuations in tempo for each instance.

The TouchDesigner system for the snake sequencer composition is illustrated in Figure 26. Raw IR camera video is routed to the "hover objects" module, a set of defined areas which the performer can use to manipulate macroscopic changes in the piece. In this composition, hover objects can add and subtract additional passageways for snakes, and increase and decrease the global tempo. The hover objects module contains a set of circles with defined X-Y position and radius. The circles are rendered and multiplied by the raw camera data, functioning as a mask. The resulting video is cropped using the bounding box for each circle, and analyzed to return a value based on its alpha channel. This essentially creates buttons which return higher values as the performer casts more IR light onto them. IR camera video is also sent to a "heatmap" module, as described in section 4.1.1.

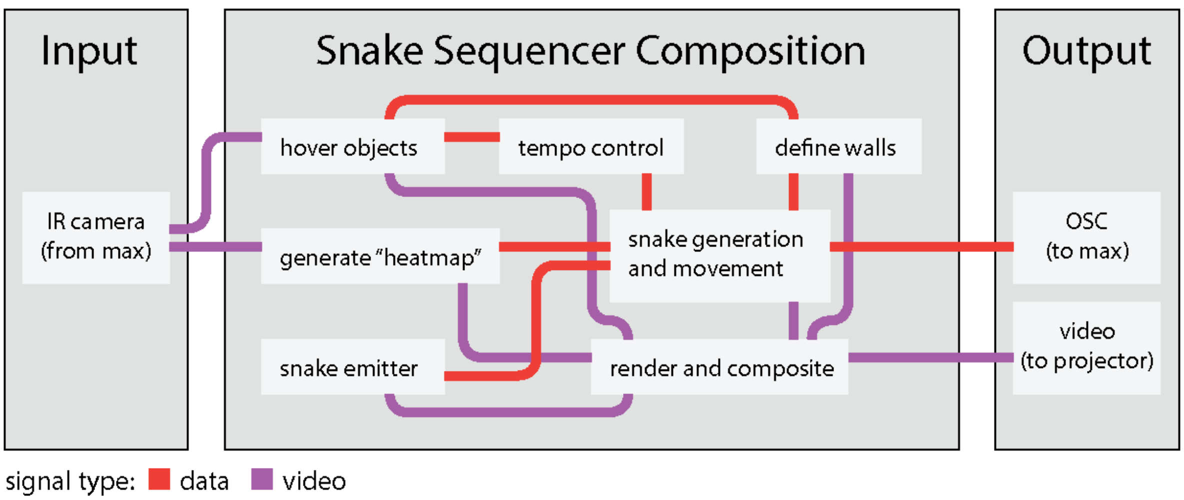


Figure 26. Snake sequencer composition software diagram

The walls are designed in illustrator, exported as raster images, and converted to a 44 x 36 tables of 0s and 1s using TouchDesigner. Snakes reference this table to determine the location of the walls, represented by 0s. For the additional pathways, 2s represent closed pathways, while 1s represent opened. To prevent snakes from getting trapped in the walls, snakes continue moving through values of 2 only if it is currently on a value of 2, otherwise it treats 2 as a wall.

The "snake generation" module contains a table with all of the current positions, last directions, rate of movements, sizes, and on/off values. Each snake instance contains an oscillator which pulses at a rate determined by its relative rate of movement, the color sampled from the heat map visualization, and the global tempo modifier. A pulse from the oscillator triggers a python script which determines the snake's movement, randomizes directions in the case of a choice, samples the heat map and sends corresponding OSC values to max for sound generation. An annotated version of the python script for this module can be found in Appendix B. The “snake emitter” module introduces and removes snakes over time, allowing for either a consistent LFO or a set of animated key-frames to vary rates. Sliders peripherally located on the left and right provide visual feedback for this automated process.

For these compositions, the content for the textured surfaces is determined by the walls for the snakes, creating a simple edge representing the boundaries. The first textures created for the system were machined before the virtual equivalent was fully tested, and used a resolution of 88 x 72 for the snake pathways (Figure 27). It was only after machining the surfaces that I discovered the heat map visualization could not support an 88 x 72 resolution without crashing, as doubling the resolution quadrupled the number of pixels to operate upon every frame.

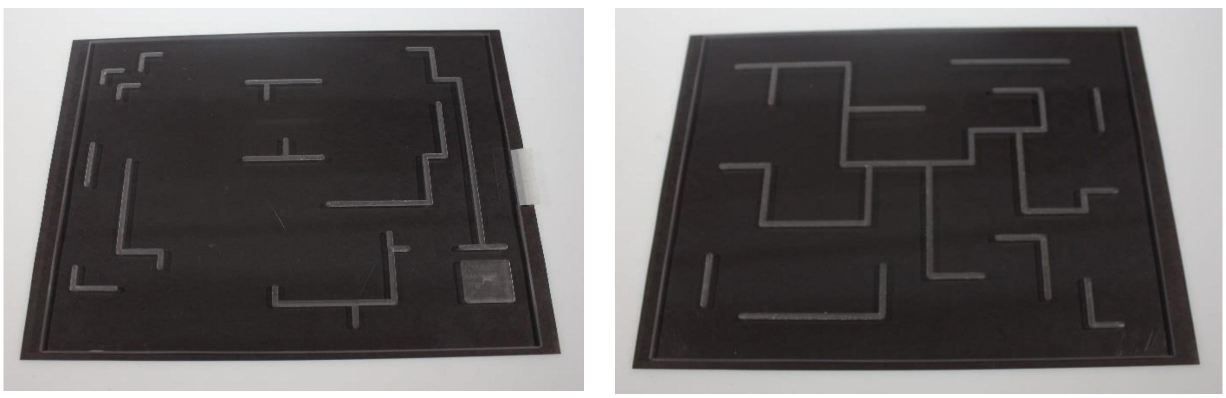


Figure 27. The first two snake pathway designs

Based on both computational constraints, as well as reflections on the previous two designs, new pathways were designed, and fabricated (Figure 28).

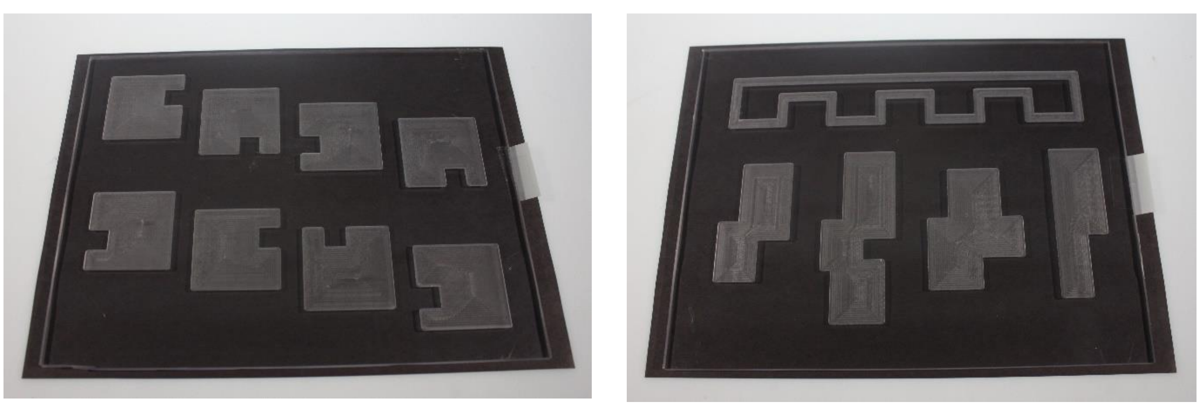
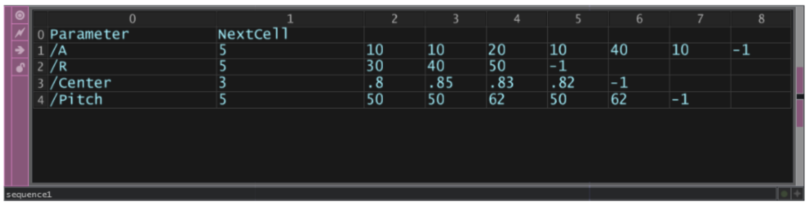


Figure 28. The final two snake pathway designs

Each snake cycles through a bank of parameter sequences used to manipulate its unique instance of the granular synth. These sequences are determined by a table of values defined for each snake instance (Figure 29). New parameters can be addressed by adding rows to the table. The sequences can be different lengths (truncated by a value of -1), allowing for phasing between parameter values. The X-Y position of the snakes also determines sonic parameters, which are mapped globally in a separate table.



	0	1	2	3	4	5	6	7	8
0	Parameter	NextCell							
1	/A	5	10	10	20	10	40	10	-1
2	/R	5	30	40	50	-1			
3	/Center	3	.8	.85	.83	.82	-1		
4	/Pitch	5	50	50	62	50	62	-1	

sequence1

	0	1
0	xPos	GrainSpeed
1	yPos	Crossfade

snakeMapping

Figure 29. An example of the parameter sequence for one snake, and the global position mapping

Figure 30 provides and annotated version of the visuals illustrating the compositions functionality. Snakes move inside of the fixed boundaries, using pseudo-random number generation to choose between left and right upon encountering a wall. The heat map color not only determines fluctuations in the snakes speed, but also influences their size. The hover objects open and close passageways, and provide global tempo control. Animated sliders on the left and right, move at different rates and trigger changes upon arriving at the top and bottom of the frame, visualized with a flash on the corresponding side. The left slider is the snake emitter, creating a new snake each time until the maximum number of snakes have been deployed, and then eliminating snakes until there are none left. The right slider switches between four preset banks defined in the granular synth, allowing for global changes that influence all the snakes simultaneously. Peripheral sliders are also employed in the particles composition (discussed in the next section). Figure 31 shows the design for the 2nd movement of the snakes composition. In this variation, passageways can be used to direct the snakes in and out of various chambers. The mapping of the X-Y position to sonic parameters gives these chambers unique characteristics.

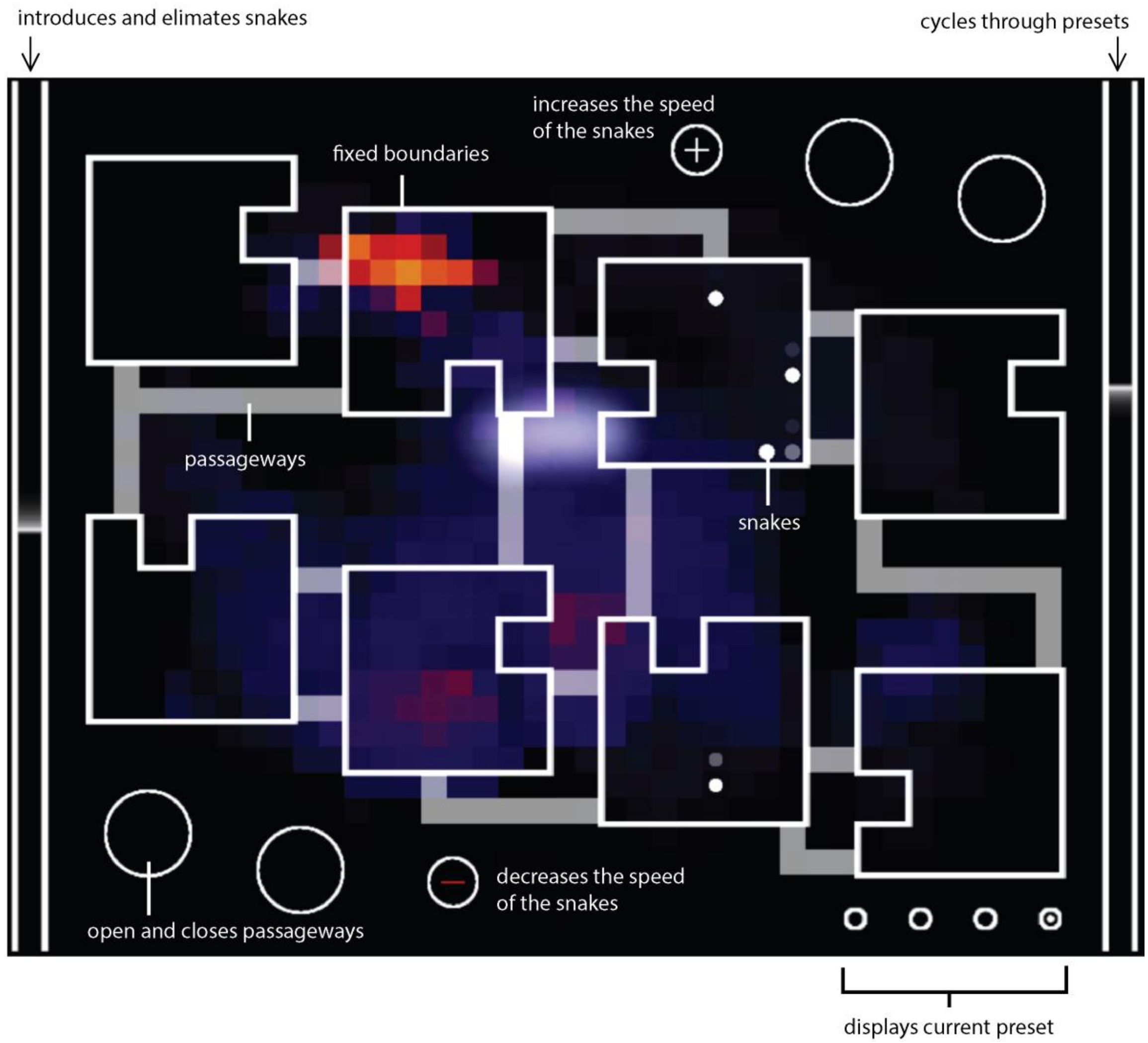


Figure 30. Snakes composition functionality

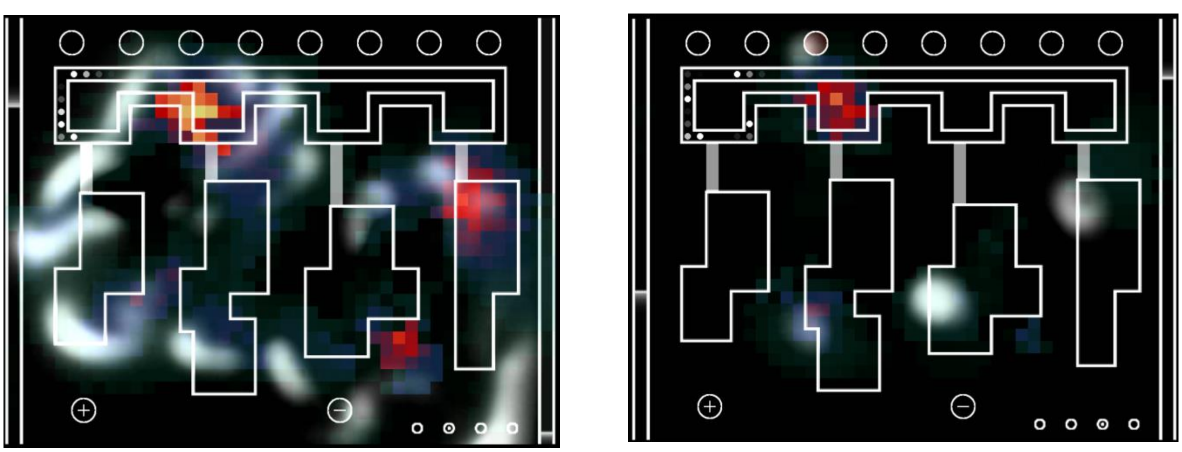


Figure 31. Images of the 2nd snakes composition movement

4.3 Particles

The idea for the *particles* composition emerged from engaging with the capabilities of TouchDesigner, which possesses the capability to create rather compelling particle systems involving various simulated forces, collision objects and surface attractors. I was interested how I might be able to use a particle system a basis for interaction and composition.

Physics models have particular significance to both sound synthesis and interacting with virtual musical instruments. Johnston et al. discusses the possibilities of physical modeling, and its potential for creating meaningful connection between the traces of performed actions, and the resulting sounds. For instance, in synthesis, one can physically model the source of the sound instead of simply mimicking desired timbral qualities - instead of “building a violin sound, the aim is to build a virtual violin” (Johnston, Candy and Edmonds 2008, 558). Johnston et al. argue that using physical models for sound synthesis is not necessary to create strong links to everyday experience, proposing a strategy using physical models to mediate between the musician’s input and audio-visuals generated by a virtual instrument (Johnston, Candy and Edmonds 2008, 559). I thought a natural approach would be to utilize particles collisions to generate sonic impulses, similar to an example by Kuhara and Kobayashi, which allows users to set large particles in motion, and maps linear and angular velocities to pitch and velocity synthesis parameters, creating percussive sounds for particle collisions (Kuhara and Kobayashi 2011).

The TouchDesigner component for the particle system composition is illustrated in Figure 32. IR camera data is sent to a hover object module similar to that of the snake sequencer composition as discussed in section 4.2. In this case, the hover objects are defined by the collision geometry, and manipulate sonic parameters specific to collisions with the object in question. The filter cutoff frequency was used in the current composition, however it could be set to any parameter. Additionally, one hover object is used to increase the rate of particle emission.

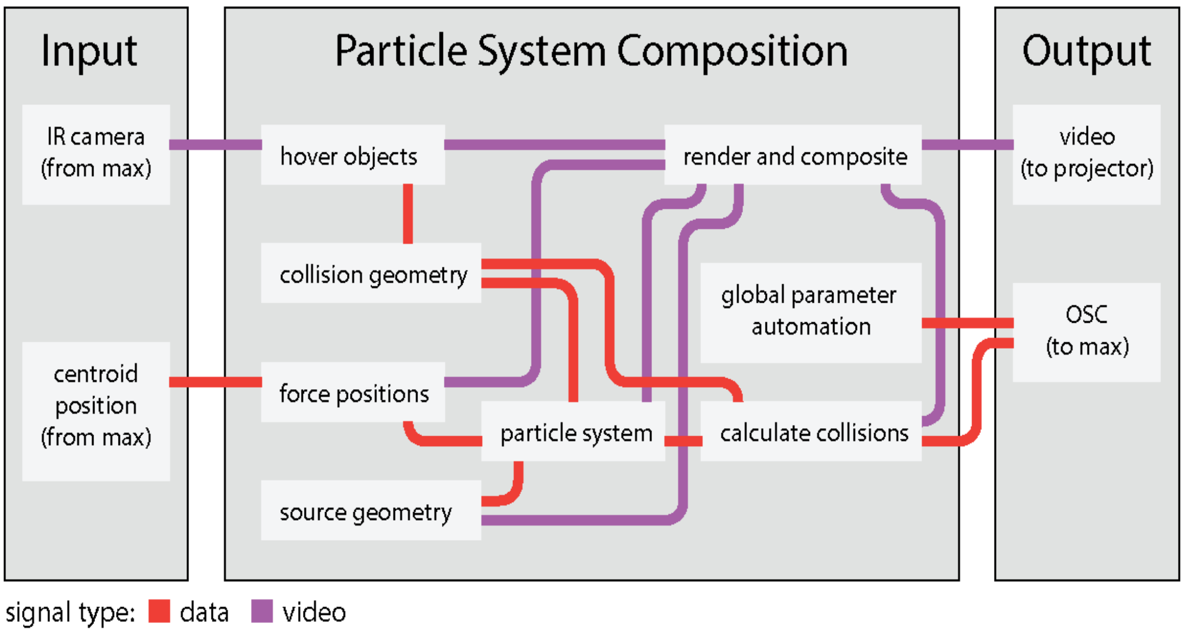


Figure 32. Particle system composition software diagram

The particle system operator in TouchDesigner can be given collision objects, forces defined by "metaballs", and source geometry providing the location of particle emission. Centroid positions are mapped to the location of these metaballs, allowing the performer to manipulate the particle system and navigate swarms of particles into the collision objects.

Although the particle operator certainly calculates collisions for its own simulation purposes, there is no way to retrieve this data – it only exports the X-Y positions of the particles. This is why simple geometry, such as circles, had to be used for collision objects. The "calculate collisions" module determines the collisions by finding the distance between every particle and every collision circle. A threshold determines when a particle reaches the radius of a given collision object, which triggers a python script to send OSC data to max, sending the X-Y position of the particle relative to the circle's center, along with the X-Y position of the collision object and the corresponding “hover” value as determined by the hover objects module. The corresponding texture for this composition was based on the collision objects and location of the particle emitter (Figure 33).

Figure 33. The texture used in the particles composition

To generate the sounds, the particles relative X-Y position upon impact determines the pitch, and provides subtle variation in the amplitude envelope attack. The position of the collision object determines the grain speed and location in the sample. Animated sliders on each side transition between global presents, and cycles through samples, crossfading between them as the slider moves. To make visuals more interesting, a rotating square is used to represent the particle emission, its speed relating to the emission rate. Feedback effects are used on the particle emitter and collision objects to create streams of light expanding outward from the center of the screen. Busts of color are introduced into these streams of light upon collision with the particles. See Figure 34 for a breakdown of the visuals and composition functionality. Figure 35 includes additional images of the particles composition.

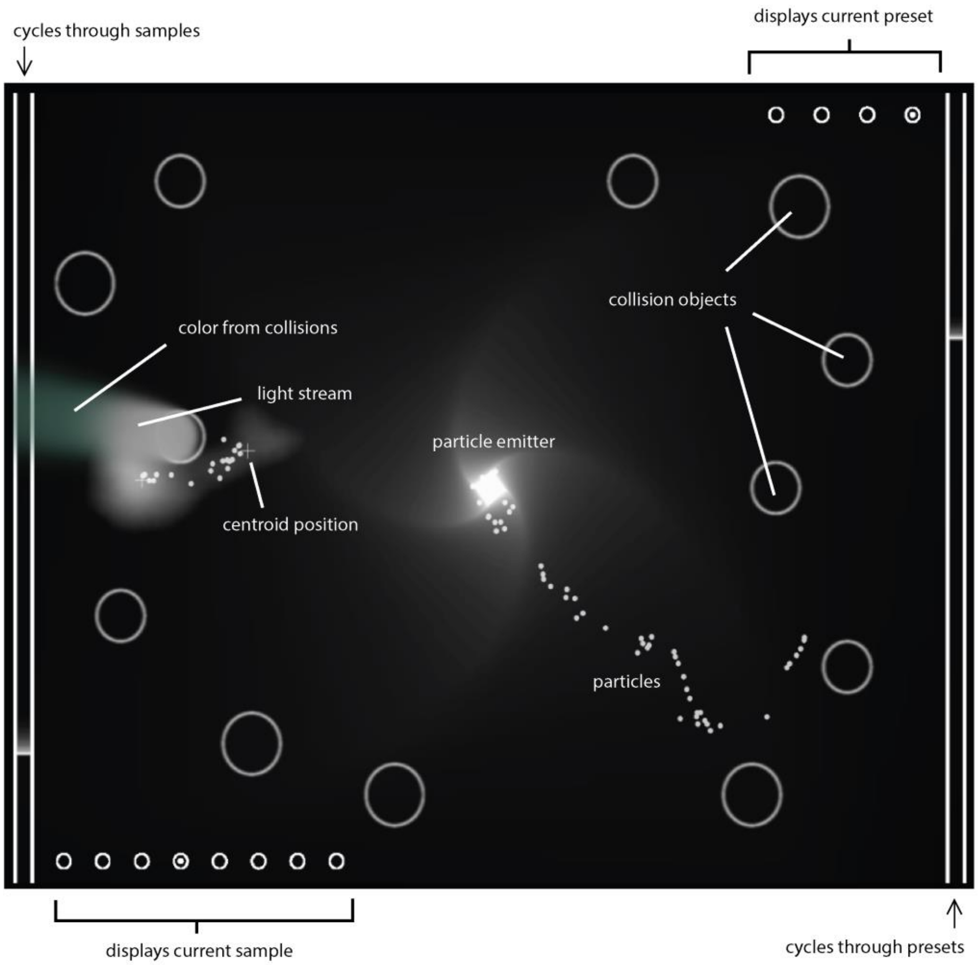
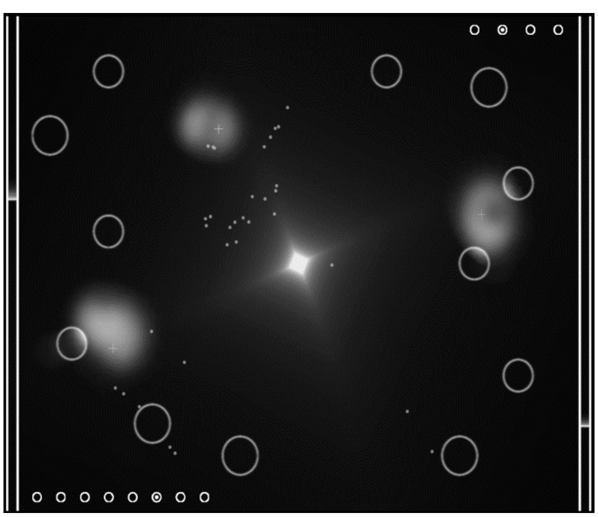


Figure 34. Particles composition functionality



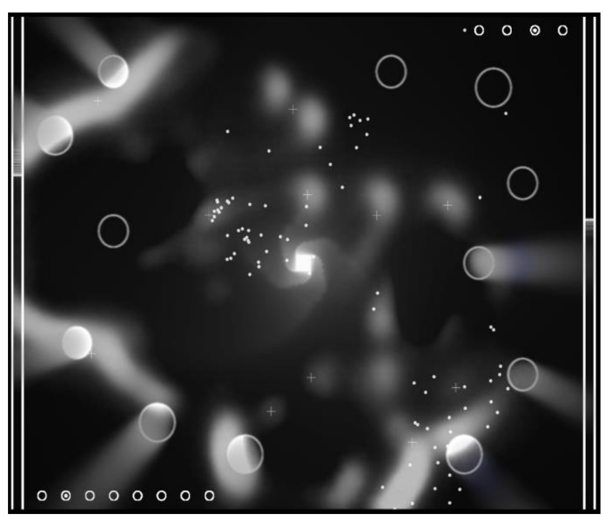


Figure 35. Additional images of the particles composition

4.4 Nodes

Unlike the previous compositions, the modality for the *nodes* composition began with engagement with the textured surfaces. One texture in particular, featuring jagged lines (Figure 36) served as the inspiration for the composition. I was interested capturing displacement from these jagged trajectories, allowing the variation in their path to alter sonic parameters. This concept relates to Ingold's discussion of traces (section 2.3.2), in the way one never creates the same trace twice (Ingold 2007). The differential between the textured trace and performer's trace provided a compelling metaphor for interaction.

Figure 36. The textured plate that in part inspired the nodes composition

The lines on the first plate were too close together for the performer to deviate from a line without immediately infringing on another. New textures were composed for the system, first painted in ink and then converted to vectors in illustrator (Figure 37).

Figure 37. Textures created for the nodes composition

The TouchDesigner component for the nodes composition is illustrated in Figure 38. Camera data from Max is processed using the same "hover objects" module as the previous two compositions, manipulating global sonic parameters such as the filter cutoff and oscillator rates. Blob area is used to control the sound of the sounds associated with each line respectively. A skeleton of points representing the basic structure of these textures is exported from Illustratror and imported into TouchDesigner as geometry for calculations with the blob centroids and for rendering a visual representation of the texture.

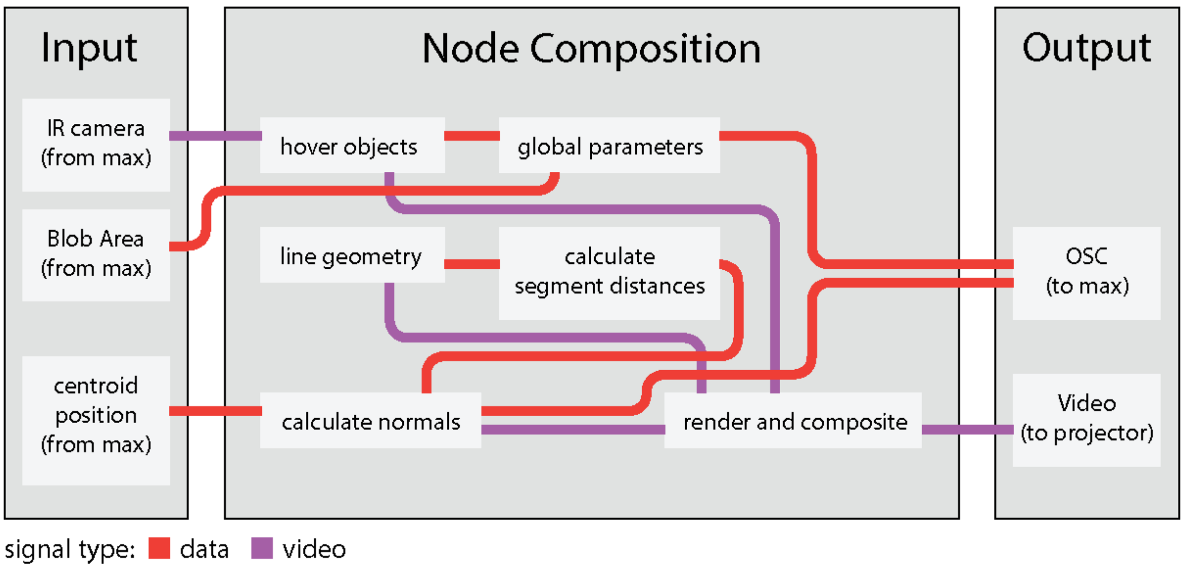


Figure 38. Nodes composition software diagram

To determine the centroid distance from the line geometry, normals are calculated from the position of blob centroids to the closest line segment, illustrated by the triangle in Figure 39. Points A and B define a line segment, and point C a given blob centroid. The length of the normal is equivalent to the height of this triangle. The calculations listed below are implemented in TouchDesigner to find the length of this normal, and subsequently the position of its intersection with the line segment.

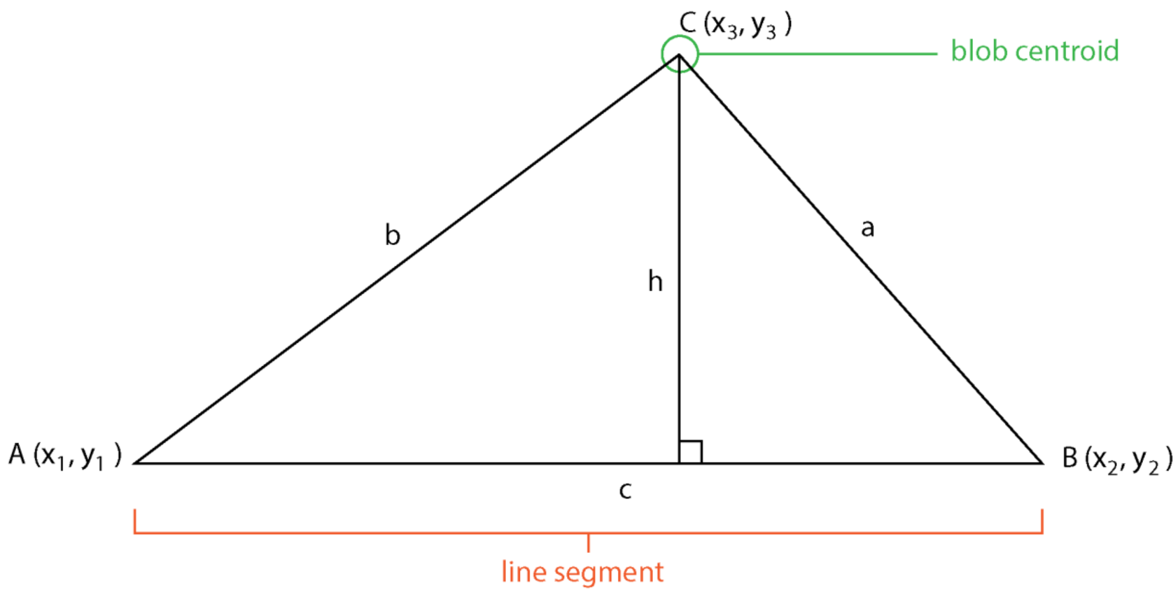


Figure 39. Triangle used to find the height of the normal from a given line segment to a centroid

Calculate the length of side a, b and c (using appropriate x, y values):

$$Distance = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Calculate angle A:

$$\mathrm{A} = \cos^{-1}\left(\frac{b^2 + c^2 - a^2}{2bc}\right)$$

Calculate the area of the triangle ABC:

$$area = \frac{bc \sin A}{2}$$

Calculate the height:

$$h = \frac{2area}{c}$$

To visually represent the normal in TouchDesigner, the point of intersection between the line segment (line AB) and height normal (line CD) also needed to be calculated (Figure 40). The following equations outline the process.

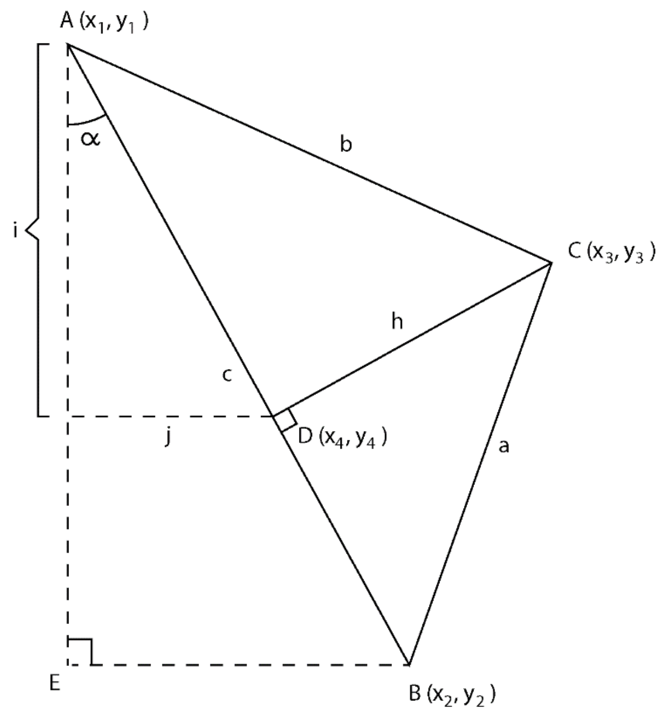


Figure 40. Triangle used to find point D

Calculate the length of line AD:

$$|AD| = \sqrt{b^2 - h^2}$$

Calculate angle *alpha*:

$$|AE| = |x_1 - x_2|$$

$$\alpha = \sin^{-1}\frac{|AE|}{c}$$

Using line AD as the hypotenuse, legs *i* and *j* are calculated to determine the displacement of point D from point A.

$$i = |AD|\sin\alpha$$

$$j = |AD|\cos\alpha$$

Before calculating the position of point D, the signs of *i* and *j* are corrected based on the location of point B to compensate for the variety of orientations that line AB can take.

$$x_4 = x_1 + j\left(\frac{x_1 - x_2}{|x_1 - x_2|}\right)$$

$$y_4 = y_1 + i\left(\frac{y_1 - y_2}{|y_1 - y_2|}\right)$$

Using the X-Y position of points C and D the normal from a blob centroid to a line segment is drawn. The height of the normal, and location along the line segment are used to manipulate sonic parameters. To implement this logic in a system with any number of blobs and line segments, the two closest points are calculated for each centroid, to determine which line segment to use. A maximum height determines if the normal is drawn, and subsequently sends OSC data for points within the height range. The normal calculation is limited to acute triangles, as otherwise normals are drawn to locations beyond the visible portion of line segments. Figure 41 shows an early implementation of this process.

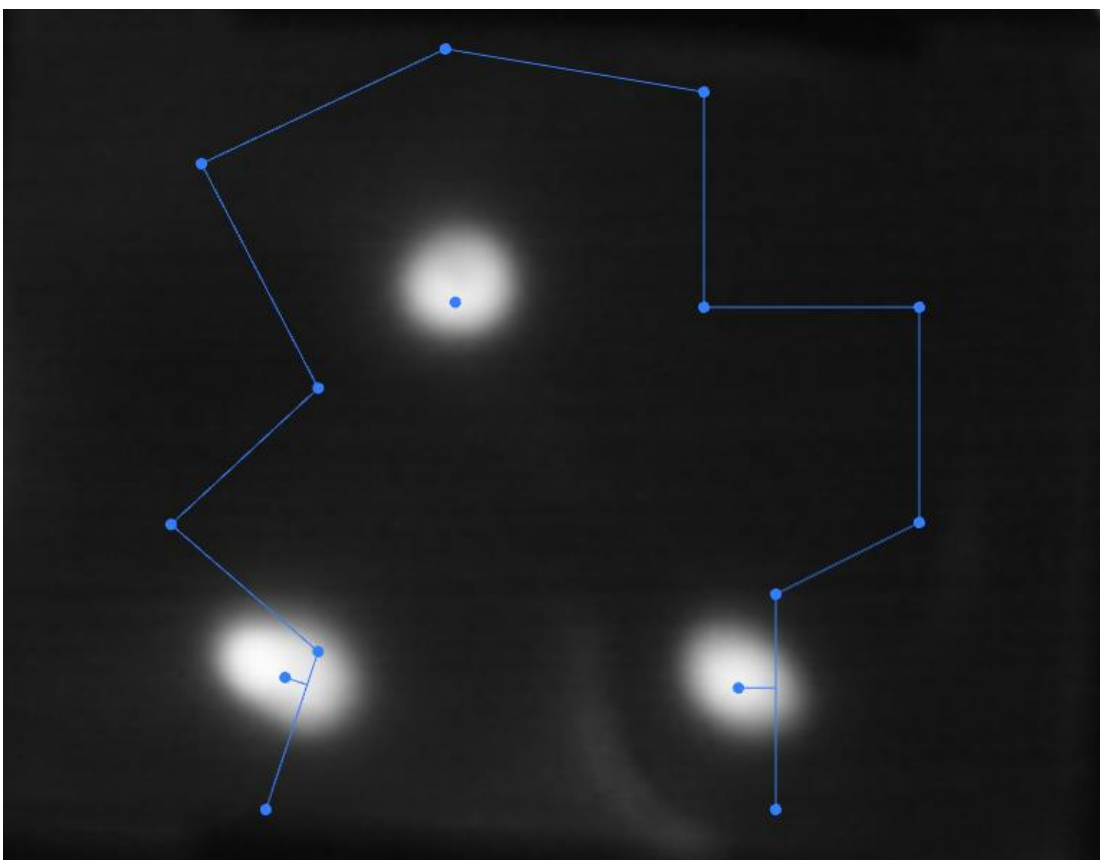


Figure 41. An implementation of calculating normals drawn from blob centroids to lines segments

One limitation of this system in its current state, is the introduction of discontinuities as the centroids move between adjacent line segments. Instead of redesigning the calculations, I treated these discontinuities as system-imposed constraints. At first, this system was used to drive the same granular synthesis engine as the *nodes* and *snakes* compositions. The height of normal each normal determined the rate of rapidly triggering note on-off values, speeding up as the centroid moved closer to the line segment. The location along the overall line determined sample position, and the area of the blob the cutoff frequency.

This method didn't create a very large space for the performer to explore new sounds and ways of interacting. It also functioned similarly to the two previous compositions, which both triggered short sounds from the granular synth, limiting the sounds to discrete sonic events. In the interest of composing a piece with contrast to the other two compositions, I created a system with the capacity for continuous control. Instead of using granular synthesis, the new sound design uses a synth based on a Max / MSP example involving a sawtooth oscillator that modulated sine and triangle oscillators, which are in turn used to reset a separate sawtooth oscillator using a sample and hold object. Centroid positions along the line control different oscillator rates, mapped based on the order of their conception. The hover objects manipulate filter cutoff frequencies and the oscillator frequencies that have the most dramatic effect on the sound. In rapidly prototyping the synthesis engine compromises were made in the systems flexibility, and as a result it can only be used with the first of the two textures (pictured on the left in Figure 37). Each of the two lines in the texture controls the parameters of a separate instance of the synthesizer, with a small amount of cross talk added so influencing one slightly modifies the sound of the other. This new synthesis method facilitates long sustained drones, which can be manipulated by moving along the textured lines.

The visual aspect of the nodes composition includes visual feedback effects on the two lines. The angle of the feedback is determined by the hover object parameters, providing a visual representation of their values. Figure 42 provides an annotated image of the compositions visuals. Additional images from the composition are provided in Figure 43.

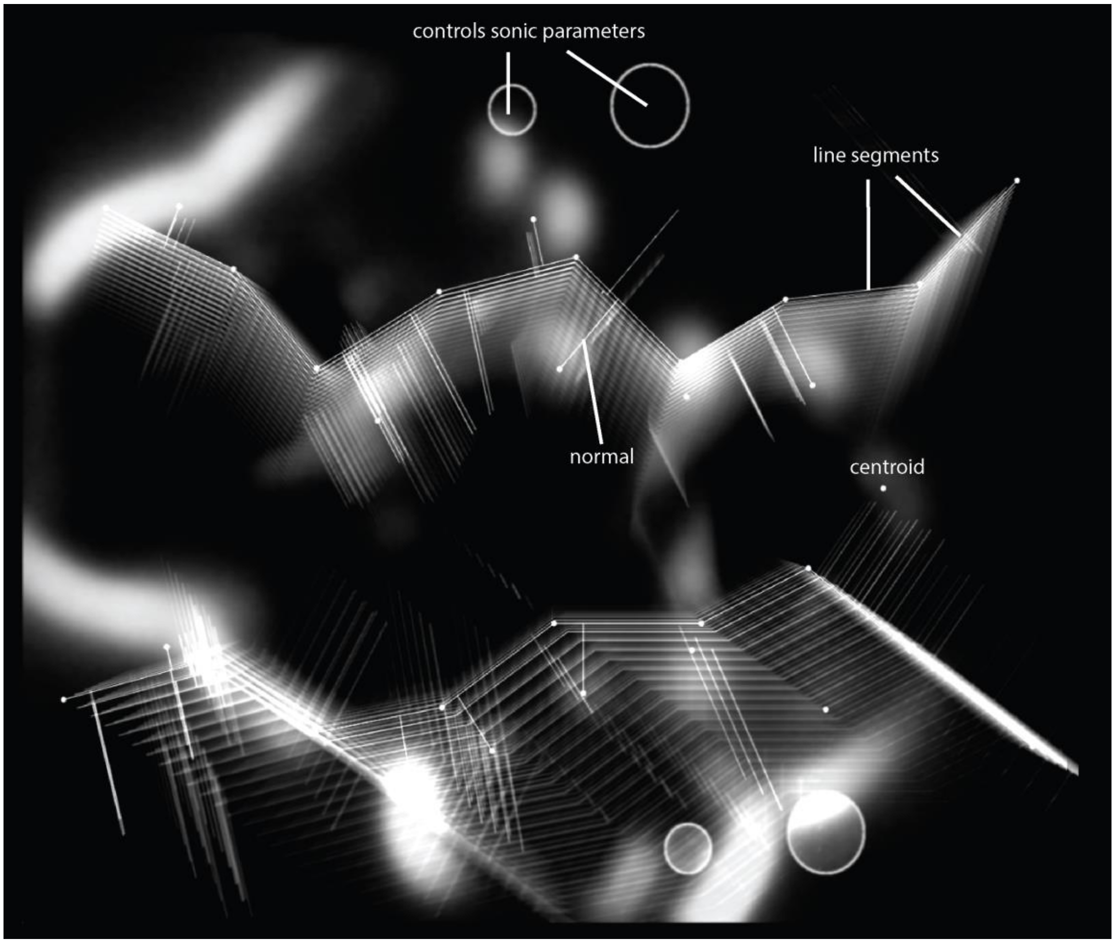


Figure 42. Nodes composition functionality

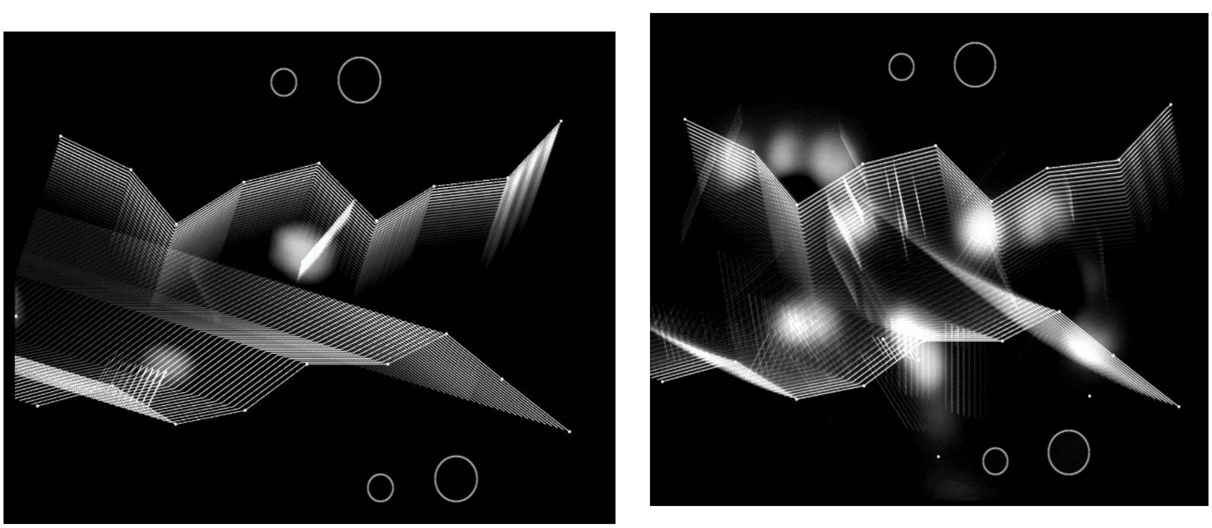


Figure 43. Additional images from the nodes composition

5. VIDEO STUDIO PERFORMANCE

The design and refinement of the compositions was followed by a performance with the system that took place April 24th, 2015 in the Digital Media Commons Video Studio, located in the Duderstadt Center and the University of Michigan, Ann Arbor. Video from this performance can be found in the accompanying DVD (Appendix C). Figure 44 shows the performance set up for the final performance in the video studio. The external computer monitor used to start the system was laid flat and covered, and the system was navigated with the keyboard in the instances when textured plates were changed. The system was used in tandem with a mixer (with onboard reverb), two delay pedals and a looper station to facilitate a solo performance. This is along the same lines that turntablists utilize mixers and other audio effects to expand their performance.

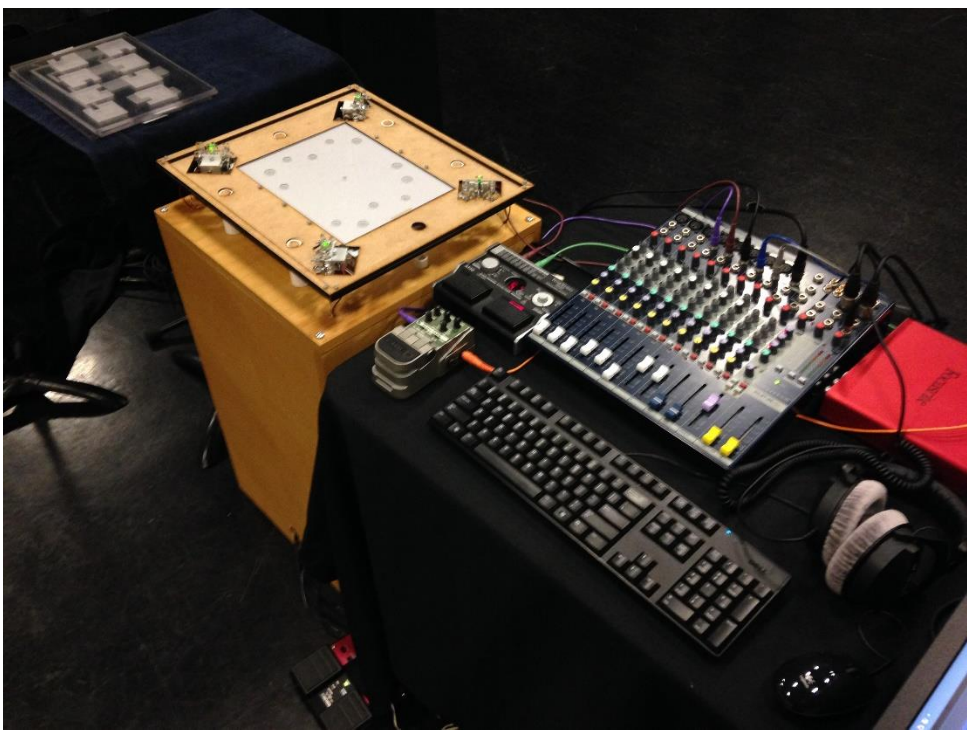


Figure 44. The performance setup for the video studio performance

Two large projection screens reciprocated the visuals projected onto the rear of the textured surfaces. In between these was a third projection screen, featuring a live camera feed from directly above the setup. This allowed the audience to watch my finger movement from above as I performed. Figure 45 includes photos from the performance.

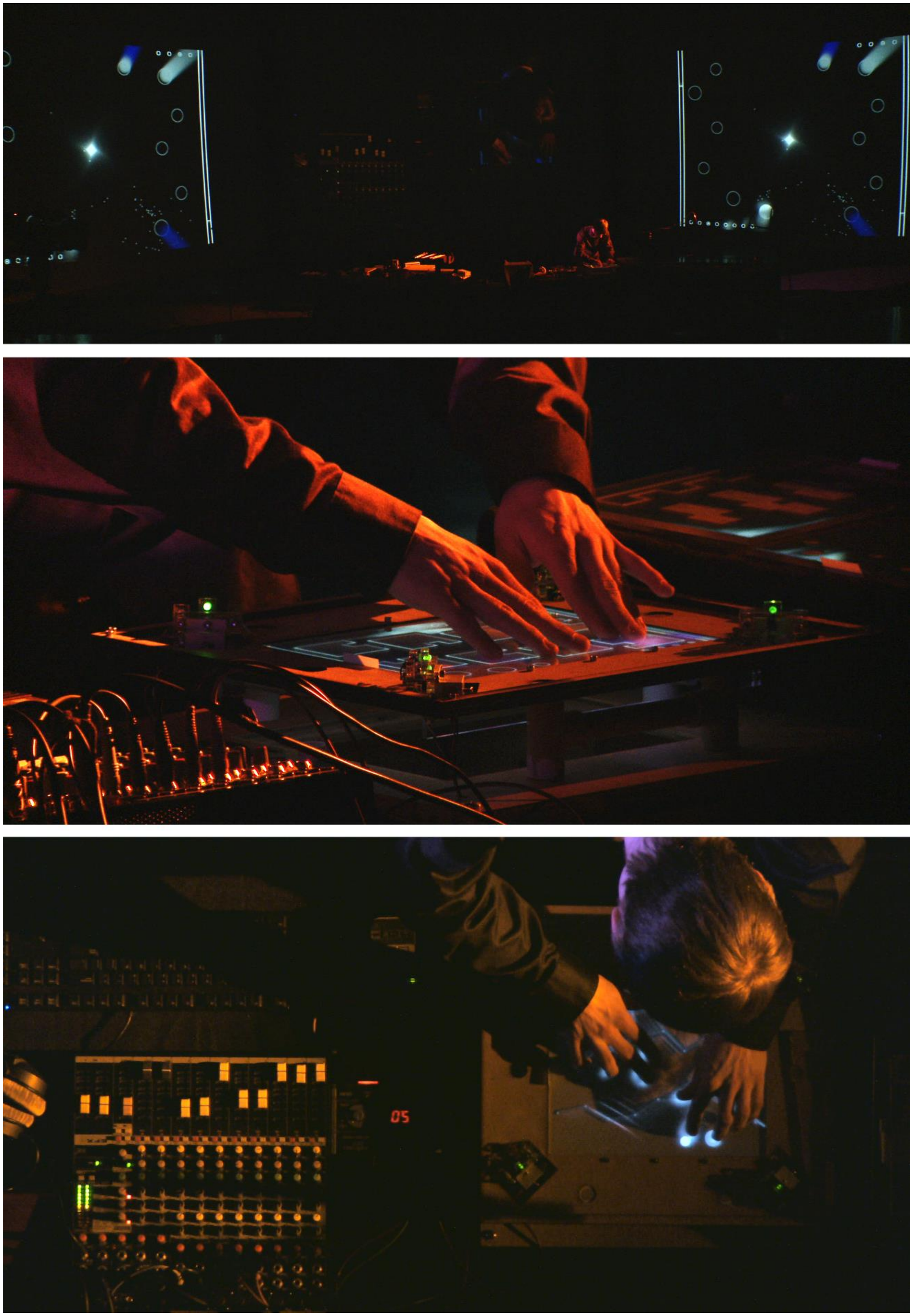


Figure 45. Images from the video studio performance

6. DISCUSSION AND CRITICISM

6.1 Reflections on the Performance

Overall the performance was successful. The projected visuals and live camera feed helped the audience understand the nature of the interaction, and the performances on the compositions went smoothly. As the system was being refined up to the point of the performance, there was little time to develop a performance practice with them. That being said, I did develop techniques I liked which I made use of during the performance: methods of distributing and focusing the particles, ways to route the snakes which allowed for the creation of different densities and textures, and a variety of gestures to produce different drones with the nodes composition. The addition of outboard effects greatly aided the performance in the times when the scores, especially in the particle and snake compositions, which occasionally felt stagnant in their sonic output.

That being said, the compositions are not fully fleshed out in their current states. Continued performance and engagement with the system would have benefited aspects of their interaction design and performance practice. I focused intently on making the interaction work in specific ways, and in doing so limited the territory of possible interaction. The snake and particle compositions were both relatively constrained as to what actions could be performed, and what sounds they could produce, making them difficult to perform musically. Of the three, I find the node composition the most successful tangible score, partly because it isn't as predictable. Its interaction is less symbolic than the other two systems allowing me to develop a bodily relationship to the score which transcended purely symbolic interaction.

6.2 Technical and Practical Issues

There are many technical issues that could improve the system. The current projector is 25 lumens, making it almost impossible to view when light shines on it. Adding a brighter projector in the base would improve its functionality in illuminated spaces. This improvement will require redesigning the projector mount and adding a mirror in the base.

A more pressing improvement would be to replace the infrared sensing. The current system uses floppy disk film to filter visible light, which is far from perfect, and caused significant issues during the performance due to interference from the stage lights. Any amount of light on the surface from the stage lights washed out the IR camera, rendering the system dysfunctional. I plan on switching to a PS3

Eye camera equipped with a filter attachment that only passes 830 nm light – the same wavelength as the IR lasers.

One practical issue is the portability of the system. The table itself is heavy and difficult to transport. Wheels on the bottom would be a great improvement, along with redesigning a smaller, lightweight base. The textured plates are heavy to be carrying around a significant number. Exploring thinner polycarbonate, or other lighter methods for creating textures would be excellent improvements.

Calibration a time consuming process, as the lasers need to be adjusted individually to create a consistent laser plane, and the projection keystoned to line up with the surface whenever the system is moved significantly. Finalizing their positions, and creating a permanent fixture would reduce calibration time.

6.3 Future Directions

None of the current compositions apply the textures directly for the purposes of sound synthesis. This was partly because the textures did not make a dramatic difference in the IR camera feed. Utilizing textures, or physical qualities of the plates directly in the sound synthesis would help create a system which moves away from symbolic interaction to an interface allowing the performer to develop *know-how* through exploring an abstract interaction. The iterative process for deriving new scores based on the aggregate of performance history (described earlier in section 4.1) also remains a compelling direction for this interface, one that would likely be improved by developing the aforementioned direct synthesis techniques.

Currently, when the performer changes textured plates there is a sequence of keyboard commands that must be triggered, to load the correct preset, initialize the system, and refresh the IR video background subtraction. An elegant way to improve this would be to encode binary numbers into the textured plates using magnets, which line up with a corresponding hall sensor array. These would serve as unique identifiers for the plates, as well as a trigger for initializing the system. It would still be important for the user to have control over the IR calibration, which could take the form of a physical button on the system itself, removing the computer keyboard from the equation completely.

There are many other conceptual ideas that remain threads of possibility. The same textured plate could be recontextualized, and more textural variations that make use of Z-depth explored. Scores could take the form of smaller, modular components that fit together like puzzle pieces, or extend beyond the laser plane. Scores that make significant use of nonlinear dynamics and generative procedures would be additional avenues to explore further.

Another important consideration is the context for performances with the device. Incorporating it into a band of other musicians, both electronic and acoustic would help inform future interaction. This was not attempted for the video studio performance in the interest of first getting completing the systems core functionality. Developing scores which can be both performed as tangible graphic scores, and simultaneously serve as a dynamic graphic score for collaborating performers is one compelling option for its use in an ensemble.

The system for tangible graphic scores I developed possess a great deal of potential for development and refinement. New compositions, ways of interpreting the data, and visual are all possible avenues for expansion, not to mention the prospect of integrating the instrument into an ensemble of other live instruments. The topic of tangible scores remains ripe with possibility, and will likely be a thread I pursue in my future work, be it with this device, or through different realizations entirely.

7. BIBLIOGRAPHY

Austin, Larry, Kahn Douglas, and Gurusinghe Nilendra. 2011. *Source: Music of the Avant-garde, 1966–1973.* Univ of California Press.

Bennett, Peter, and Sile O'Modhrain. 2008. "The BeatBearing: A Tangible Rhythm Sequencer." *Proceedings of NordiCHI.*

Berghaus, Marc. 2011. "Some Ideas for Three-Dimensional Musical Scores." *Leonardo Music Journal* 21: 7–8.

Bischof, Markus. 2008. "Xenakis: Combining Tangible Interaction with Probability-Based Musical Composition." *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction.* ACM. 121-124.

Briz, Nick. 2010. "Glitch Codec Tutorial." Accessed April 26, 2015. http://nickbriz.com/glitchcodectutorial/.

Cage, John. 1962. *Cartridge Music.* Mainstream Records.

Cardew, Cornelius. 1967. *Treatise.* Gallery Upstairs Press.

Cascone, Kim. 2000. "The Aesthetics of failure:'Post-Digital' Tendencies in Contemporary Computer Music." *Computer Music Journal* 24 (4): 12-18.

Chadabe, Joel. 1984. "Interactive Composing: An Overview." *Computer Music Journal* 8 (1): 22-27.

Coduys, Thierry, and Guillaume Ferry. 2004. "Iannix-Aesthetical/symbolic Visualisations for Hypermedia Composition." *Sound and Music Computing Conference.*

Coenen, Alcedo. 1994. "Stockhausen's Paradigm: A Survey of His Theories." *Perspectives of New Music* 32 (2): 200-225.

Collins, Nicolas. 2006. *Handmade Electronic Music: The Art of Hardware Hacking.* Taylor & Francis.

Cox, Christoph. 2009. "Sound Art and the Sonic Unconscious." *Organised Sound* 14 (1): 19-26.

D'Arcangelo, Gideon. 2004. "Recycling Music, Answering Back: Toward an Oral Tradition of Electronic Music." *Proceedings of NIME.* National University of Singapore. 55–58.

2002. "Earle Brown." *The Musical Times* 143 (1880): 5-6.

Fujihata, Masaki, Kiyoshi Furukawa, and Wolfgang Münch. 2000. "Notes on Small Fish." *Ars Electronica 2000 Festival Catalog* 306-309.

Gann, Kyle. 2012. *Robert Ashley.* University of Illinois Press.

Gould, John D., and Clayton Lewis. 1985. "Designing for usability: key principles and what designers think." *Communications of the ACM* 28 (3): 300–311.

Groot, Mariska de, and Dieter Vandoren. 2013. "Shadow Puppet?" 3 8. https://vimeo.com/63393966.

Hauert, Sibylle, and Daniel Reichmuth. 2003. "Instant City." *ein elektronischer musik bau spiel automat.* Rosen and Spademan Enterprise.

Henke, Robert. 2015. *Granulator II.* Accessed April 9, 2015. http://www.monolake.de/technology/granulator.html.

Hochenbaum, Jordan. 2010. "Designing Expressive Musical Interfaces for Tabletop Surfaces." *Proceedings of the Conference on New Interfaces for Musical Expression.* 315-318.

Honigman, Colin, Jordan Hochenbaum, and Ajay Kapur. 2014. "Techniques in Swept Frequency Capacitive Sensing: An Open Source Approach." *Proceedings in NIME.*

Husarik, Stephen. 1983. "John Cage and LeJaren Hiller: HPSCHD, 1969." *American Music* 1 (2): 1-21.

Ingold, Tim. 2005. "Beyond Art and Technology: The Anthropology of Skill." In *Anthropological Perspectives on Technology*, by Michael B. Schiffer, 17-30. UNM Press.

—. 2007. *Lines: a brief history.* Routledge.

Irdial-Discs. 2012. *The Conet Project: Recordings of Shortwave Numbers Stations.* Accessed April 10, 2015. http://www.irdial.com/conet.htm.

Ishii, Hiroshi. 2008. "The Tangible User Interface and Its Evolution." *Communications of the ACM* 51 (6): 32–36.

Ishii, Hiroshi, and Brygg Ullmer. 1997. "Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms." *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems.* ACM. 234–241.

Johnston, Andrew, Linda Candy, and Ernest Edmonds. 2008. "Designing and Evaluating Virtual Musical Instruments: Facilitating Conversational User Interaction." *Design Studies* 29 (6): 556-571.

Kahn, Douglas. 2003. "Christian Marclay's Early Years: An Interview." *Leonardo Music Journal* 17-21.

Kaltenbranner, Martin, Serji Jorda, Gunter Geiger, and Marcos Alonso. 2006. "The Reactable*: A Collaborative Musical Instrument." *Enabling Technologies: Infrastructure for Collaborative Enterprises* 406–411.

Kaltenbrunner, Martin. 2015. *Tangible Music.* Accessed April 9, 2015. http://modin.yuri.at/tangibles/.

Kiser, Spencer. 2006. "spinCycle: A Color-Tracking Turntable Sequencer." *Proceedings of the 2006 Conference on New Interfaces for Musical Expression.* IRCAM—Centre Pompidou. 75-76.

Krumhansl, Carol L. 1989. "Why is musical timbre so hard to understand." *Structure and perception of electroacoustic sound and music* 9: 43–53.

Kuhara, Yasuo, and Daiki Kobayashi. 2011. "Kinetic Particles Synthesizer Using Multi-Touch Screen Interface of Mobile Devices." *Proceedings of NIME.* 136-137.

Lieberman, David. 2006. "Game Enhanced Music Manuscript." *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia. ACM.* 245–498.

Ligeti, György. 1970. *Artikulation. An Aural Score by Rainer Wehinger.* Mainz: Schott.

Lucier, Alvin. 1998. "Origins of a Form: Acoustical Exploration, Science and Incessancy." *Leonardo Music Journal* 8: 5 - 11.

Magnusson, Thor. 2010. "Designing Constraints: Composing and Performing with Digital Musical Systems." *Computer Music Journal* 34 (4): 62-73.

—. 2009. "Of epistemic tools: Musical instruments as cognitive extensions." *Organised Sound* 14 (2): 168–176.

Marino, Gérard, Marie-Hélène Serra, Jean-Michel Raczinski, Gerard Marino, and Marie-Helene Serra. 1993. "The UPIC System: Origins and Innovations." *Perspectives of New Music* 31 (1): 258-269.

McGee, Ryan. 2013. "VOSIS: a Multi-touch Image Sonification Interface." *Proceedings of NIME.* 460-463.

McGlynn, Patrick, Victor Lazzarini, Gordon Delap, and Xiaoyu Chen. 2012. "Recontextualizing the Multi-Touch Surface." *Proceedings of NIME.*

Merrill, David, Jeevan Kalanithi, and Pattie Maes. 2007. "Siftables: Towards Sensor Network User Interfaces." *Proceedings of the 1st International Conference on Tangible and Embedded Interaction.* ACM. 75-78.

Meyer, Leonard B. 1998. "A Universe of Universals." *The Journal of Musicology* 16 (1): The Journal of Musicology.

Miller, David P. 2009. "Indeterminacy and Performance Practice in Cage's "Variations"." *American Music* 27 (1): 60–86.

Miyashita, Homei, and Kazushi Nishimoto. 2004. "Thermoscore: a new-type musical score with temperature sensation." Edited by National University of Singapore. *Proceedings of NIME.* 104–107.

Montag, Matthew, Stefan Sullivan, Scott Dickey, and Colby Leider. 2011. "A Low-Cost, Low-Latency Multi-Touch Table with Haptic Feedback for Musical Applications." *Proceedings of NIME.* 8–13.

Morimoto, Tania K., Paulo Blikstein, and Allison M. Okamura. 2013. *History of the Haptic Paddle.* September. Accessed April 8, 2015. http://hapkit.stanford.edu/about.html.

Müller-Tomfelde, Christian. 2010. *Tabletops - Horizontal Interactive Displays.* London: Springer London.

Newton-Dunn, Henry, Hiroaki Nakano, and James Gibson. 2003. "Block Jam: A Tangible Interface for Interactive Music." *Proceedings of the 2003 Conference on New Interfaces for Musical Expression.* National University of Singapore. 170–177.

Nybø, Håkon. 2014. *Music and sound design projects of Håkon Nybø.* Accessed January 20, 2015. http://hakonnybo.com/.

2015. *OpenCV.* Accessed April 9, 2015. http://opencv.org/.

Paik, Nam June. 2002. "Random Access." *Leonardo* 35 (5): 489–489.

Patten, James, Ben Recht, and Hiroshi Ishii. 2002. "Audiopad: A Tag-Based Interface for Musical Performance." *Proceedings of the 2002 Conference on New Interfaces for Musical Expression.* National University of Singapore. 1–6.

2015. *Peau Productions.* Accessed April 9, 2015. http://www.peauproductions.com.

Pelletier, Jean-Marc. 2013. "cv. jit–." *Computer Vision for Jitter* 15. http://jmpelletier. com/cvjit.

Piper, Ben, Carlo Ratti, and Hiroshi Ishii. 2002. "Illuminating Clay: A 3-D Tangible Interface for Landscape Analysis." *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* New York, NY: ACM. 355–362.

2002. *Scratch.* Film. Directed by Doug Pray. Produced by Momentum Pictures / Palm Pictures.

Roads, Curtis. 2004. *Microsound.* MIT press.

Russolo, Luigi. 2004. "The Art of Noises: Futurist Manifesto." *Audio culture: Readings in modern music* 10-14.

Sandler, Seth. 2015. *Multitouch How To.* Accessed January 23, 2015. http://sethsandler.com/multitouch/.

Sauer, Theresa. 2009. *Notations 21.* Mark Batty Pub.

Schedel, Margaret. 2002. "Anticipating Interactivity: Henry Cowell and the Rhythmicon." *Organised Sound* 7 (3): 247-254.

Schloss, W. Andrew. 2003. "Using Contemporary Technology in Live Performance: The Dilemma of the Performer." *Journal of New Music Research* 32 (3): 239-242.

Shaer, Orit. 2015. "Tangible User Interfaces: Past, Present, and Future Directions." *Foundations and Trends in Human–Computer Interaction* 3 (1-2): 1–137.

Shiffman, Daniel, Fry Shannon, and Zannah Marsh. 2012. *The nature of code.* D. Shiffman.

Sweet, Michael. 2015. *Writing Interactive Music fo Video Games.* Pearson Education Inc.

Tomás, Enrique, and Martin Kaltenbrunner. 2014. "Tangible Scores: Shaping the Inherent Instrument Score." *Proceedings in NIME.* 609–614.

Toro, Mauricio, Myriam Desainte-Catherine, and Pascal Baltazar. 2010. "A Model for Interactive Scores with Temporal Constraints and Conditional Branching." *Journées d'Informatique Musicale*.

Ward, Nicholas, Kedzie Penfield, Sile O'Modhrain, and Benjamin Knapp. 2008. "A Study of Two Thereminists : Towards Movement Informed Instrument Design." 117–121.

Weingarten, Emily. 2008. *The Music of ONCE: Perpetual Innovation*.

Wright, Matthew. 1997. "Open sound control-a new protocol for communicating with sound synthesizers." *Proceedings of the International Computer Music Conference.*

Yang, Justin Wen-Lo. 2008. *Sometimes I feel the space between people (voices) in terms of tempos—A work for percussion duo with computer animated notational interface.* Standford University.

APPENDIX A – *snake attractors python script*

Python script for creating snake attractors in TouchDesigner.

```
def whileOff(channel, sampleIndex, val, prev):
	currentPos = op('currentPos')
	targetPos = op('targetPos')
	currentVel = op('currentVel')

	#get current position, target position and current velocity from external TouchDesigner operators
	x = currentPos[0,0]
	y = currentPos[0,1]
	targetX = targetPos[0,0]
	targetY = targetPos[0,1]
	velX = currentVel[0,0]
	velY = currentVel[0,1]

	#set the bounds
	xMax = 1.0
	xMin = -1.0
	yMax = 1.0
	yMin = -1.0

	#define vectors and forces
	location = tdu.Vector(x, y, 0)
	target = tdu.Vector(targetX, targetY, 0)
	velocity = tdu.Vector(velX, velY, 0)
	acceleration = tdu.Vector(0,0,0)
	maxSpeed = 0.1
	maxForce = 0.01

	#check for wall collisions, flip velocities
	if location[0] > xMax:
		velocity[0] = velocity[0] * -1.0
		location[0] = xMax * 0.99
	elif location[0] < xMin:
		velocity[0] = velocity[0] * -1.0
		location[0] = xMin * 0.99

	if location[1] > yMax:
		velocity[1] = velocity[1] * -1.0
		location[1] = yMax
	elif location[1] < yMin:
		velocity[1] = velocity[1] * -1.0
		location[1] = yMin

	#set desired vector normalize and scale to maximum speed
	desired = target - location
	desired.normalize()
	desired = desired * maxSpeed

	#steering = desired minus velocity
	steer = desired - velocity
```

```
        #limit the force
        if steer.length() > maxForce:
                steer.normalize
                steer = steer * maxForce

                acceleration = acceleration + steer

        #update velocity
        velocity = velocity + acceleration

        #limit the velocity
        if velocity.length() > maxSpeed:
                velocity.normalize
                velocity = velocity * maxSpeed

        #update location
        location = location + velocity

        #update current postition and velocity values
        currentPos[0,0] = location[0]
        currentPos[0,1] = location[1]
        currentVel[0,0] = velocity[0]
        currentVel[0,1] = velocity[1]
        return
```

APPENDIX B – *snake sequencer python script*

Python script for moving snakes, and triggering sounds in the snakes composition.

```
#define walls, snake positions, project resolution, the current snake ID (row), and OSC output
walls = op('../walls')
pos = op('../snakePosition')
res = op('/project1/lowRes')
row = me.parent().digits + 1
oscNote = op('/project1/oscout1')
oscParam = op('/project1/oscParam')

#define global variables
currentPixel = 0.0
pixelThresh = 0.05

#the main function, which runs whenever an external pulse LFO fires (depends on the rate of the snake)
def offToOn(channel, sampleIndex, val, prev):
        res = op('/project1/lowRes')
        xPos = int(pos[row,0])
        yPos = int(pos[row,1])
        direction = pos[row,2]

        #is the snake on:
        if int(pos[row, 5]) == 1:
                pos[row, 6] = 1 #set size to 1
                #set Random Direction
                randDir = int(tdu.rand(absTime.frame + .1 * int(me.parent().digits))*2

                #Move snake (if above pixel thresh)
```

```
        grid = op('/project1/snakeComp/heatMapOut')
        global currentPixel
        currentPixel = grid.sample(x=xPos, y=yPos)[3]
        if currentPixel > pixelThresh:
            if int(walls[yPos, xPos]) == 1:
                snakeMove(direction, xPos, yPos, randDir, 1)
            else:
                #Move snake if midway in passage
                snakeMove(direction, xPos, yPos, randDir, 2)
    else:
        #if the snake is off set size to 0
        pos[row, 6] = 0
    return

#function for trigger sounds and sending OSC parameters
def sound(xPos, yPos):
    #Map pixel Intensity to Cutoff
    cutoff = int(translate(currentPixel,pixelThresh,0.9,500,2000))
    vals = [cutoff, row]
    oscParam.sendOSC('/FilterCutoff', vals)
    #Map X Position
    xRange = op('/project1/lowRes')[0]
    xScaled = (xPos / xRange)
    vals = [xScaled, row]
    oscParam.sendOSC('/' + op('../snakeMapping')[0,1], vals)
    #Map Y position
    yRange = op('/project1/lowRes')[1]
    yScaled = yPos / yRange
    vals = [yScaled, row]
    oscParam.sendOSC('/' + op('../snakeMapping')[1,1], vals)
    # sequence parameters
    sequence()
    return

# function to sequence the sonic parameters
def sequence():
    out = op('output')
    seq = op('../sequence' + str(row))
    numParam = seq.numRows
    # sequence parameters
    for i in range(1, numParam):
        cell = seq[i,1]
        if seq[i,cell] > 0:
            # output current cell, increment cell
            if currentPixel > pixelThresh:
                vals = [seq[i,cell], row]
                oscParam.sendOSC(seq[i,0], vals)
            seq[i,1] = cell + 1
        else:
            # reset sequence to beginning for next time
            if currentPixel > pixelThresh:
                vals = [seq[i,2], row]
                oscParam.sendOSC(seq[i,0], vals)
            seq[i,1] = 3
    return

# functions that check for walls. If there is no wall it moves the snake in that direction and calls the sound function
def up(xPos, yPos, x):
```

```
	if yPos + 1 <= res[1] and (int(walls[yPos + 1, xPos]) == 1 or int(walls[yPos + 1, xPos]) == x):
		pos[row,1] = yPos + 1
		pos[row,2] = 0
		sound(xPos, yPos + 1)
		return 0
	else:
		return 1

def down(xPos, yPos, x):
	if yPos - 1 >= 0 and (int(walls[yPos - 1, xPos]) == 1 or int(walls[yPos - 1, xPos]) == x):
		pos[row,1] = yPos - 1
		pos[row,2] = 1
		sound(xPos, yPos - 1)
		return 0
	else:
		return 1

def left(xPos, yPos, x):
	if xPos - 1 >= 0 and (int(walls[yPos, xPos - 1]) == 1 or int(walls[yPos, xPos - 1]) == x):
		pos[row,0] = xPos - 1
		pos[row,2] = 2
		sound(xPos - 1, yPos)
		return 0
	else:
		return 1

def right(xPos, yPos, x):
	if xPos + 1 <= res[0] and (int(walls[yPos, xPos + 1]) == 1 or int(walls[yPos, xPos + 1]) == x):
		pos[row,0] = xPos + 1
		pos[row,2] = 3
		sound(xPos + 1, yPos)
		return 0
	else:
		return 1

# Function for scaling values
def translate(value, leftMin, leftMax, rightMin, rightMax):
# Figure out how 'wide' each range is
   leftSpan = leftMax - leftMin
   rightSpan = rightMax - rightMin
# Convert the left range into a 0-1 range
   valueScaled = float(value - leftMin) / float(leftSpan)
# Convert the 0-1 range into a value in the right range.
   return rightMin + (valueScaled * rightSpan)
# function for determining the direction of the snake movment
def snakeMove(direction, xPos, yPos, randDir, x):
# up: y + 1
	if direction == 0:

		if up(xPos,yPos,x) == 1:
			if randDir == 0:
				if left(xPos,yPos,x) == 1:
					if right(xPos,yPos,x) == 1:
						down(xPos,yPos,x)
			else:
				if right(xPos,yPos,x) == 1:
					if left(xPos,yPos,x) == 1:
						down(xPos,yPos,x)
```

```
# down: y - 1
    elif direction == 1:
        if down(xPos,yPos,x) == 1:
            if randDir == 0:
                if left(xPos,yPos,x) == 1:
                    if right(xPos,yPos,x) == 1:
                        up(xPos,yPos,x)
            else:
                if right(xPos,yPos,x) == 1:
                    if left(xPos,yPos,x) == 1:
                        up(xPos,yPos,x)
# left: x - 1
    elif direction == 2:
        if left(xPos,yPos,x) == 1:
            if randDir == 0:
                if down(xPos,yPos,x) == 1:
                    if up(xPos,yPos,x) == 1:
                        right(xPos,yPos,x)
            else:
                if up(xPos,yPos,x) == 1:
                    if down(xPos,yPos,x) == 1:
                        right(xPos,yPos,x)
# right: x + 1
    elif direction == 3:
        if right(xPos,yPos,x) == 1:
            if randDir == 0:
                if down(xPos,yPos,x) == 1:
                    if up(xPos,yPos,x) == 1:
                        left(xPos,yPos,x)
            else:
                if up(xPos,yPos,x) == 1:
                    if down(xPos,yPos,x) == 1:
                        left(xPos,yPos,x)
    return
```

APPENDIX C – *DVD guide*

This DVD included with this thesis contains a video of the performance in the Digital Media Commons Video Studio (April, 24th 2015).

The media from this DVD can also be found at:

http://www.simonaa.media/content/tangible-graphic-scores