PENSATO

A Virtual Reality Framework for Musical Performance

by

Byron Mallett

A Masters Thesis by Composition
submitted to the Victoria University of Wellington
in fulfilment of the requirements for the degree of
Masters of Design Innovation in Computer Graphics

Victoria University of Wellington
2016

B

The final composition video associated with this thesis, Pensato: Fissure can be accessed online through the following link.

http://youtu.be/tz79NmSt_Vo

All code associated with this thesis is available at the following links:

Showtime: **https://github.com/mystfit/showtime**
Pensato: **https://github.com/Mystfit/TeslaRift**

D

I. Abstract

This thesis presents the design for a method of controlling music software for live performance by utilising virtual reality (VR) technologies. By analysing the performance methods of artists that use either physical or gestural methods for controlling music, it is apparent that physical limitations of musical input devices can hamper the creative process involved in authoring an interface for a performance. This thesis proposes the use of VR technologies as a central foundation for authoring a unique workspace where a performance interface can be both constructed and performed with. Through a number of design experiments using a variety of gestural input technologies, the relationship between a musical performer, interface, and audience was analysed. The final proposed design of a VR interface for musical performance focuses on providing the performer with objects that can be directly manipulated with physical gestures performed by touching virtual controls. By utilising the strengths provided by VR, a performer can learn how to effectively operate their performance environment through the use of spatial awareness provided by VR stereoscopic rendering and hand tracking, as well as allowing for the construction of unique interfaces that are not limited by physical hardware constraints. This thesis also presents a software framework for connecting together multiple musical devices within a single performance ecosystem that can all be directly controlled from a single VR space. The final outcome of this research is a shared musical environment that is designed to foster closer connections between an audience, a performer and a performance interface into a coherent and appealing experience for all.

F

II. Acknowlegements

This thesis would not have been possible without the following people:

Thanks go to my supervisor Rhazes Spell, who provided all right questions and insight to help me retain my direction throughout the course of this project.

Xiaodan Gao from Victoria Learning Support who helped me get started with the difficult process of writing, and whom without this thesis would not have reached completion.

Anne Niemetz and Josh Bailey for providing me the opportunity to create a work for the Forks and Sockets concert, the basis of which became the direction for my final project Pensato.

Terry Moore and Michael McKinnon for all the support, encouragement and discussion that helped the refinement of each of my works over time.

Steven Lam for helping me share the experience of completing a masters degree to the end, with all the stress, elation, frustration and satisfaction involved.

Szilárd Ozorák for all his help in filming, cinematography and transportation that made my final video possible.

My parents for their consistent support throughout each of my endeavours.

And to all my fellow masters students who have helped make the long nights bearable, thank you.

Table of Contents

J

1. Introduction

This thesis investigates the creative possibilities afforded by the merging of three separate technological areas: the creation of live music using intangible instruments, the development of meaningful user interfaces viewed and controlled using virtual reality technologies, and the operation of musical software through hardware controllers.

By combining these avenues, this thesis presents a method of controlling musical software with virtual reality technology that explores what possible performance and musical arrangement opportunities are unlocked by the merger of these forms of technology.

1.1 Motivation

My motivation for combining virtual reality technologies with music software stems from my own personal experiences and issues with the complexity of musical software interfaces and the rigidity of hardware audio controller setups. As a musician, there are a number of areas that I believe can be improved upon in the realm of electronic music performance. These areas include the increasing amount of complexity inherit in music software interfaces, finding ways of communicating musical performance actions to an audience, and the lack of flexibility for configuring musical hardware setups on the fly during performances.

1.1.1 Audio software complexity

Electronic music existed well before the use of the computer in synthesized music compositions and performance. Instruments within this genre can be traced back over the last 120 years in various forms, including the following examples.

- The Telharmonium created by Thaddeus Cahill in 1906 was a 200-ton instrument played using a set of four keyboards and broadcast over the New York telephone network (Simon, 2014).
- The Theremin patented by Leon Theremin in 1928 which was an ethereal tone generator played by a performer using only their body and with no physical contact with the instrument (Galeyev, 1991).
- The electronic synthesizers that have been constructed and popularized by Robert Moog (1934-2005) since 1965 which helped to ease the synthesizer into mainstream music.

Even though electronic music production existed long before the age of computers, the progress of both the power of computer systems and the flexibility of music software interfaces has helped to expand the ability of performers to create musical works that would have been difficult or impossible to compose using analog instruments. Modern software provides the ability to synthesize almost any conceivable sound using vast numbers of parameters, leading to increasingly complicated interfaces being required in order to expose all the functionality offered by these systems. But as the number of features and controls within audio software increases, so does the need to have a intuitive and uncomplicated way of interacting with them, so that musical inspiration is not lost through the fight with abstracted input mechanisms.

1.1.2 Audience Communication

One of the primary methods of interaction with audio software is still through the use of the ubiquitous mouse and keyboard. These tools serve well as general purpose input devices, but also limit expression

in musical performance due to their generality. These input devices lack actions dedicated to a performance context and must rely on the host performance software to provide a meaningful context for actions that are then triggered by the input device.

Whilst these tools function well for a wide range of tasks, this can also hide the individual effort of a live performer if they choose to use this as their primary method of performance. This can be attributed to the audience presuming that the use of a mouse and keyboard requires a lower level of skill to operate based on their own day-to-day experiences with using these devices. Performing with these input devices will not offer enough visual cues to the audience to discern whether the performer is carrying out a complex series of musical tasks rather or checking their email (Fintoni, 2013).

1.1.3 Hardware controls and limitations

The extension of musical interfaces using specialized hardware performance controls can help to overcome the limitations of a mouse and keyboard setup for live performance, and provide a central focus for the audience to observe in order to relate the authenticity of the performer's actions to the generated music.

As music interfaces possess a great amount of complexity, custom hardware controls for manipulating parameters are incredibly useful for extending the range of actions a performer can accomplish without switching between multiple contexts. By combining buttons, faders, knobs and multi-dimensional touchpads, a performer has access to a wide array of physical inputs for controlling each individual aspect of a performance. However, most of these hardware controls only have single-use applications. For example, a performer will typically link a control to the audio software to manipulate a single aspect of their performance such as the volume of an instrument. This static mapping offers less options for the performer to create new layout configurations on the fly and limits them to only being able to operate their interface in a fixed number of ways.

Another alternative for controlling audio software is through the operation of instruments that feature no physical components. Through the use of sensors, cameras and micro-controllers, instruments can be created that are solely operated by the movement of the human body, rather than through a physical device. This can open up a wide array of mappings between the movement of a person into the generation or control of sound and music. This quickly introduces problems however, due to the removal of the haptic feedback that is afforded to the performer by having a physical surface to press against. The performer has to memorize precise body positions and motions in order to achieve the sound that they desire, with only their sense of hearing available to gauge if a movement is correct. This can be remedied with controls being visualized in a

graphical manner, but depending on the size of screen used to deliver the visual feedback, this may limit the range of motion that a performer can work with.

This thesis will investigate approaches to solving these issues, by coupling use of an Oculus Rift virtual reality Head Mounted Display (HMD) with sensors for detecting hand and finger positions, in order to create virtual controls that operate similarly to hardware sound controllers, but with the freedom afforded by a virtual environment to organise and remap controls and parameters using spatial interactions between controls.

The final creative output of this thesis is Pensato, a performance tool that allows for the extension of music performance software into a VR space. Pensato utilizes an interface that allows for a performer to control musical software by manipulating virtual objects using a pair of positional and gesture tracking gloves. This thesis also presents Showtime, a software framework for connecting together musical devices together into a shared performance environment.

2. Context Analysis

Before undertaking the task of extending music software with VR technologies, an understanding of how artists create and work with their own live performance systems is required. In order to improve upon these existing implementations, an appreciation of how artists have created their own technology, instruments and conventions for performing live electronic music is needed in order to establish what can be improved through the addition of VR technologies to a musical performance systems.

This chapter will examine how some artists have uniquely approached this area, what interfaces and design methodologies they have constructed in order to create their art, and how their designs can relate to the application of VR technology to a performance system.

It is also important to examine how the artist presents their method of interaction to the audience. Two approaches will be considered: sound synthesis and musical triggering. Sound synthesis will involve the artist creating music controlled and expressed from their direct actions, similarly to how a performer utilizes a traditional music instrument. Musical triggering will cover the performance of arranged music where the performer fills the role of a choreographer and directs the music through the playing of pre-composed sequences.

Figure 1: Barbara Buchholz plays a TVox model theremin (Gregor, 2008).

2.1 Human gestural input in musical performance

Human gestural input in electronic music stretches over 80 years of innovation and encompasses a variety of approaches for involving the human body in a musical performance without the use of a physical instrument. Two instruments that span this timeline include the Theremin, invented by Leon Theremin in 1928, and Imogen Heap's performance gloves from 2013. By analysing these two instruments we can extract positive and negative elements, and detail the advantageous uses of this performance approach as well as possible improvements that can be made using VR input devices.

Figure 2: Leon Theremin performing a trio for theremin, voice and piano (Theremin, 1924).

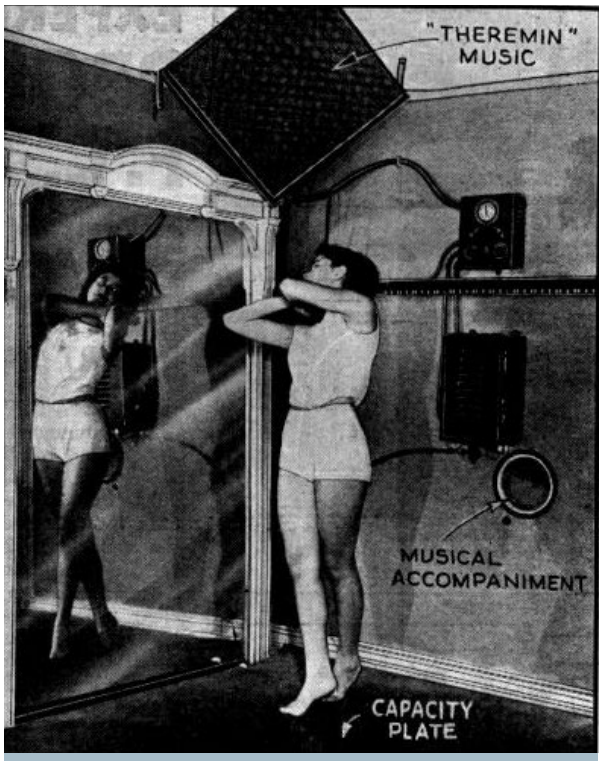

Figure 3: Dancer performing with a Terpsitone (Mason, 1936).

2.1.1 Theremin

Arguably the first person to perform using an intangible electronic instrument was Léon Theremin, using his own invention the Termenvox, more commonly known as the Theremin (Galeyev, 1991). The instrument was operated using the position of the performer's hands relative to two antennas to play a sound that would alter in pitch and volume. The music played by the theremin was usually very legato (without silence between notes), since the performer would constantly be sliding between frequencies as they performed. To remedy this, the theremin performer would also articulate the volume of the instrument with their other hand in order to separate out distinct notes from the constant change in frequency performed with their other hand. Advertisements for theremin performances at the time of its introduction compare it to pulling music from the air itself (Martin, 1995).

Expanding on the concept of controlling music through movement, Theremin went on to also create the Terpsitone (Mason, 1936). This device allowed a dancer to use their height and body motion over a metal floor plate to control musical pitch and their distance from a metal plate on a wall to control the amplitude. The resulting performance resulted in a choreographed dance performance accompanied by a unique musical backing generated by the motion of the dancer.

Figure 4: Me The Machine (Heap, 2014).

Figure 5: Imogen Heap at TEDGlobal 2012 (Heap, 2013).

2.1.2 Imogen Heap's Glove Project

Imogen Heap's demonstrative performance of her glove project involved the use of inertial motion sensors, flex sensors, a Microsoft Kinect, and a series of microphones to remotely create and perform electronic music. The inertial motion sensors and flex sensors were built into a pair of gloves and enabled gestural recognition as a form of interaction for controlling sound parameters and switching performance contexts. The Kinect allowed for the position of Heap's location on the stage to be tracked and could control parameters such as addition of reverb to Heap's voice the further back she would place herself. The combination of these systems enabled her to perform a choreographed routine where her body was used as the control interface for a musical performance.

Imogen stated that one of the primary reasons for creating an interface designed in such a manner, was to foster a closer connection between her movements on stage and the sounds that she was creating (Heap, 2013). This would allow the audience to more closely track the importance of each gesture to an instantaneous musical output. Some examples of gestures included drumming motions to trigger tuned percussion and gesturing at waist height to control the pitch of a bass line.

2.1.3 Analysis

By drawing comparisons between the playing style for both of these instruments we can start to see how such a interaction method would work within a VR space for the benefit of both the performer and audience. Both the Theremin and Imogen Heap's gloves demonstrate the importance of gesture within a performance for fostering a connection between the artist's performance and the audience. By choosing appropriate movements that map to perceptibly audible outputs, the audience can begin to easily associate a particular gesture or type of movement with the sound heard, bringing authenticity to the performer's actions. It will be important to make sure that this type of relationship is preserved within the final VR interface so that an audience will not be confused about the role the performer is playing.

Both the Theremin and Imogen Heap's gloves lack the haptic advantages provided by a physical instrument for playing fast and accurate notes. In the case of Imogen Heap's gloves, small motors are present that provide a vibration feedback response from certain actions. However, these are reactionary at best and don't provide an opposing force for resisting the motion of the performer's limbs.

Physical instruments that have variable pitches (such as a stringed instrument like a violin) require the performer to carefully pay attention to the played sound and alter their pitch to resolve poor intonation. However, since a violin is a physical object that provides feedback to the player through touch, muscle memory is utilized whilst learning how to play series of notes. The performer subconsciously learns where each finger needs to be positioned relative to the other fingers on the neck in order to generate a particular pitch. Both of these discussed instruments do not possess a tangible physical object that is interacted with by the performer, rather the body of the performer is instead the instrument interface. As a result, the performer has to be very precise in knowing where their body is located in relation to a control that cannot be directly felt in the space where the performer is interacting with it.

Both of these instruments also lack strong visual elements that indicates the way body has to move to create a particular audible output. However, Heap's interface does possess some visual elements representing what high-level mode the performer is operating within, such as recording or playing back sound. The addition of a visual response indicating the result of the performer's actions on a screen or projected surface can help resolve this, but using these types of visual media will always maintain a physical separation between the performer's body and the visual output. The addition of a VR interface that synchronizes the performer's body between the real performance space and a virtual one, could make strides towards reducing this separation.

2.2 Hardware controls in Performance

By using custom and off-the-shelf hardware, a musician can create their own unique layouts and methods for interfacing with music software. But musical software has become increasingly complex over time. This means that control layouts have had to follow suit in order to provide artists higher levels of control over their musical systems. This section examines artists such as the French music duo Taylorythm , and Beardyman, who combine various kinds of musical technology to form platforms for performance or musical improvisation. By evaluating the advantages of these artist's unique control layouts, approaches to applying VR technology as an augmentation for a similar system will be considered.

2.2.1 Taylorythm

A good example of the power afforded by combining multiple hardware controls for live performance can be observed through performances created by the french electronic music duo Taylorythm. Analysing the video of their remixed performance of "You are Amazing" by Eleven (Buignet & Tisseraud, 2013) we can see the advantage afforded by the wealth of controllers used in their piece.

In their setup we can see multiple Novation Launchpad controllers (Novation, 2013), one for each performer. Each launchpad is connected to the host performance software Ableton Live (Ableton, 2014) and allows each performer to trigger sections of the song or play samples of sound. Note that there is only a single laptop running Ableton Live for this the piece, necessitating the use of hardware controllers since the logistics required to operate a mouse and keyboard with two performers would be impossible.

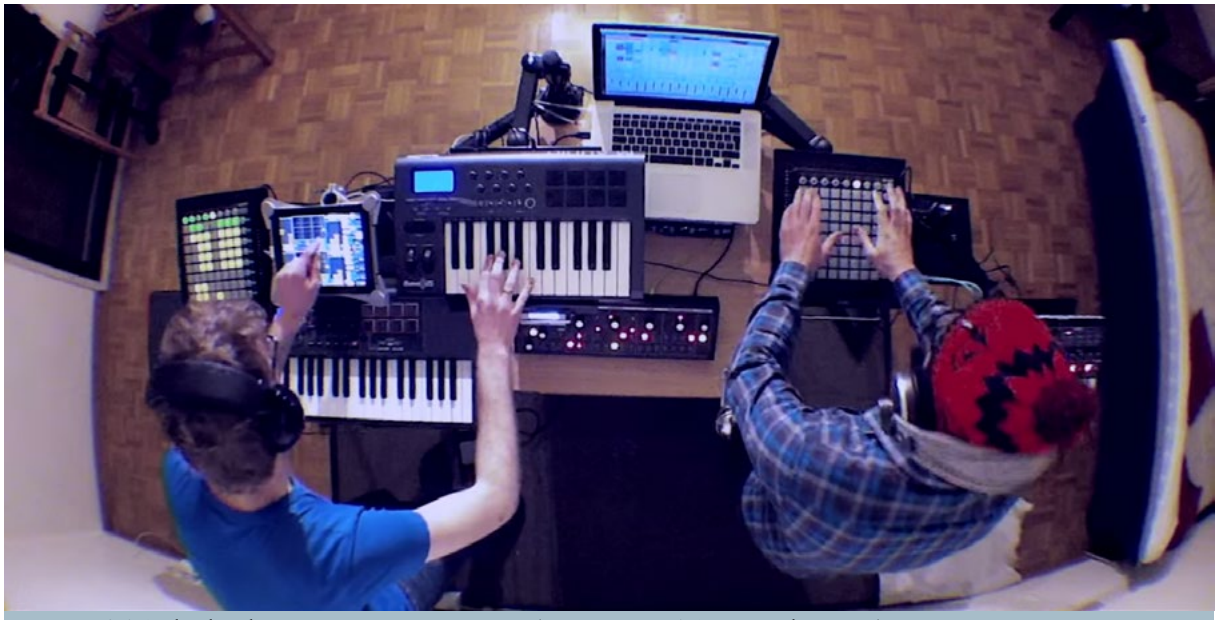

Figure 6: Taylorhythm - You are Amazing (Buignet & Tisseraud, 2013)

The multitude of devices present in this performance demonstrates the necessity for the control layout to be uncluttered and easily accessible. Often one of the musicians has to multi-task between two different devices or parameter controls using both hands simultaneously, demonstrating the need to have a clear arrangement of controls at all times. The clear organisation of control devices in this piece also helps to establish a good division of tasks between each performer and reduce the chance of collision or interference from two musicians competing over one interface.

Several instruments are operated by the variety of keyboard controllers present. In one instance, a keyboard sends note information to a bass synthesizer whilst the rhythm is controlled through a control knob instead of the keys being released and triggered to play notes. This sort of parameter-focussed operation allows the performer to stay in sync with the performance and remove the need for exact timing when playing notes.

An iPad tablet present used in this performance, running the Ableton Live control software touchAble (Zerodebug, 2013), allows the performers to control a large number of mapped parameters on a 2D surface. By using a touchscreen interface in this performance, a wider range of controls can be stacked into one small space and swapped between as required, reducing the total number of hardware controllers required.

2.2.2 Beardyman

Darren Foreman, also known by his stage name Beardyman, is most well known for his use of incredibly complex and sophisticated live-looping techniques using nothing but his voice as a source of musical samples. In his TED demonstration, he demonstrates how his musical workspace allows him to create a vast range of different sounds and instruments through only the processing of his voice through multiple slicing, pitch shifting and modulation techniques. The most recent version of his musical workstation, the Beardytron5000 mkIII, features a bespoke UI design crafted to suit the improvisational style of Beardyman, and demonstrates a variety of novel methods of interacting with sound as it is created in real-time (Foreman, 2014).

Whilst operating the instrument, Beardyman demonstrates a frantic level of interaction with the tablet surfaces and keyboards that comprise most of the input layers of the instrument, but each action appears to be specific to achieving a outcome in the soundscape being created. Beardyman appears to be utilizing gestures that extend past simple button or fader control gestures, and combines dragging, holding and tapping movements to control a single parameter in a number of ways that would be difficult or impossible to achieve using an equivalent physical control.

Figure 7: Beardyman: The Polyphonic Me (Foreman, 2013).

Without the dynamic interface capabilities offered by computer-driven touchscreen technology, such an instrument would have to have a much larger surface area in order to provide enough controls to manipulate every parameter within the system.

2.2.3 Analysis

Both Taylorhythm and Beardyman use complex performance systems that have been hand crafted to facilitate their own kind of musical performance using both unique or commercially available hardware and software combinations. However, there are limits in regards to usability for anyone not familiar with these unique control layouts. Whilst this is appropriate for these two performers, who have built a very personal arrangement of hardware, the complexity and cost of the devices, coupled with the extensive space required and lack of portability, will likely be a deterrent to their adoption by other performers. The addition of touchscreen technology to both performance layouts offers a level of customization that extends beyond simple hardware buttons and sliders, but these controls are limited by the physical constraints imposed by the relatively small surface area offered by the tablet.

2.2.4 Conclusion

I believe that a VR interface could adopt the customization benefit afforded by touchscreen interfaces whilst solving the surface area problem by providing a three dimensional area for touch controls. A disadvantage of this method is that a VR interface will lack the haptic feedback provided by the physical surface of a tablet (as discussed in the previous section). This will be a point for evaluation within the final VR design and alternative solutions will need to be considered.

2.3 Virtual Reality in Musical Performance

The use of VR technology to assist a musical context is not a new idea. There have been a number of attempts made over the last two decades at creating interfaces within virtual spaces for composing or performing music. However, due to the limited availability, high cost, and ineffectiveness of VR technology , these prototypes have remained as proof-of-concepts. These projects discussed provide a number of natural or immersive methods for interacting with objects within a virtual space within a musical context.

2.3.1 DRILE

One VR interface for performing music is DRILE, which is a VR environment that uses head-tracked stereoscopic glasses and rhythmic motion controllers to control objects representing sound effects and parameters. The environment uses hierarchical live-looping to build up sequences of notes and effect parameters using 'worms' that act as nexus points for parameters that affect the music played. The performer can modify these parameters in space by moving 'tunnel' objects over the worms which change the shape, hue and transformations of the worm and thus the musical parameters linked to them. This project promotes musical collaboration by letting performers manipulate multiple musical elements together in the shared space, but is not able to deliver a fully immersive experience for every user simultaneously due to the limitations of head tracking technology when dealing with multiple performers observing the same screen.

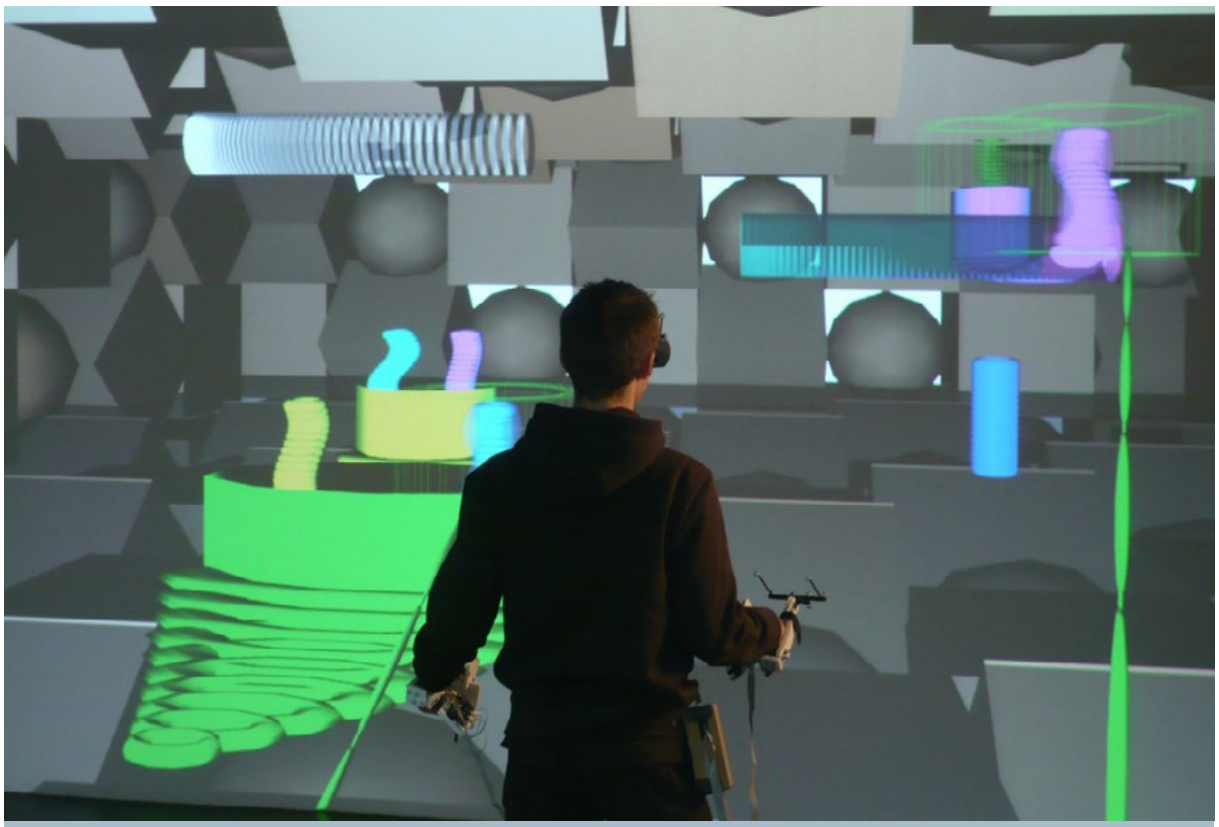

Figure 8: DRILE: An Immersive Environment for Hierarchical Live-Looping (Berthaut, Desainte-Catherine & Hachet, 2010)

2.3.2 Virtual Music Instruments

Another approach for applying VR technology to a musical context is through the creation of Virtual Music Instruments, VMI for short (Mulder, 1998). For this project, The VMIs provided methods of interaction that were inspired by tactile interactivity, though technological limitations stripped the possibility of haptic feedback from the design. Some of the design metaphors used for interaction involved sculpting or interaction with physical objects, such as a rubber sheet or ball. It is significant to note that the final evaluation of the work states that it was difficult to interact with at times, due to the lack of visual depth cues. This can be attributed to the fact that the system was presented to the performer using a traditional two dimensional display, even though the design of the instruments would have been better suited for a 3D VR world. The musical focus was also more strongly orientated towards sound synthesis rather than musical interpretation, due to limitations in expressing tone and rhythm using the input mechanism provided.

2.3.3 Analysis

DRILE demonstrates the playing of notes through the use of physical controllers, which should prove to be a more robust and effective method for playing exact rhythms rather than relying on gesturing in space, due to the presence of a physical object to stimulate performance muscle memory. However, this limits the number of gestures available, as only a series of simple button presses is involved and maintains a level of abstraction between the actual hand of the performer and the virtual space. Removing this input abstraction and using the hand of the performer directly is demonstrated in the VMI project and the positive response that such a natural gesture interaction method can provide is demonstrated.

2.3.4 Conclusions

Whilst both of these VR interfaces demonstrate immersive methods of delivering spatial controls to a performer, their relative isolation from a standard musical performance system means that effects and sound samples need to be tailor-made and chosen for inclusion within the musical environment. My hypothesis is that sticking to a more common musical system through integration with existing music software can expand the creative resources available for a performer or composer, rather than reinventing the wheel in terms of generating music.

2.4 Methods for interacting with VR spaces

A significant subject to discuss in regards to VR interaction design is how gestures and intent can be placed upon a variable scale of realism. In particular, how gestures can fall under the categories of naturalism or "magic" (Bowman, McMahan, & Ragan, 2012).

Natural gestures focus on using actions that we use in our daily lives into VR space on the subconscious level. The actions typically involve types of movements that we do not need to actively think about, such as grasping an object or pressing a button. The user has an intent that they need to communicate to the item in question, and will naturally choose the most appropriate gesture for the interaction.

Gestures that fall in the category of "magic" are ones that exaggerate existing natural gestures in order to let the user perform in ways that would be impossible with natural gestures due to physical constraints, such as reaching for far away objects that lie outside the reach of a normal human. Magic gestures can also encompass novel gestures that don't have a close natural mapping. Many of these gestures can be used to overcome the limitations of a VR environment, such as the attempting to solve the challenge of moving through the VR space whilst being tethered to a workstation generating the feed for your VR headset.

In Bowman's paper, tests were performed on VR tasks such as changing the observable viewpoint, traversing a VR space, manipulating objects, driving, aiming, and performing multiple complex tasks. Most of the experiments provided favourable outcomes for naturalistic gestures, where affording the user the closest possible parallel to a real gestures would increase their precision and effectiveness at manipulating their surroundings. Conversely, magic gestures that enhanced natural gestures provided a greater range of control, at a cost of losing precision. From these results, I believe that utilizing natural gestures in the design of a virtual music interface will prove to be the most effective method for building interaction methods that reinforce learned accuracy over the course of using the interface, due to their closeness to real-world actions.

From this we can conclude that as the level of abstraction increases from a natural gesture to an unnatural or hyper-real gesture, the more actively the user has to put them into action. In a live performance context, naturalistic gestures will theoretically help to foster a closer connection between the performer and their VR tool than hyper-real gestures, though such gestures still have a place for controlling non-time critical user interface elements.

3. Design Experiments

For this thesis, two separate design experiments are presented as explorations of alternative methods of musical and visual control using non-physical methods. Each project takes an opposing approach in regards to interaction fidelity, ranging from abstract motions transformed into meaningful music, to precise and specific motions translated into visual responses. These projects form a basis for the design interactions and decisions evident in the final project outcome of this thesis.

3.1 Experiment One: Sonoromancer

Sonoromancer was a solo experiment undertaken to understand the underlying connection between the body of a performer and the space that they perform within. The project is similar in operation to instruments such as the Terpsitone (see 2.1.1), but instead of sensing magnetic fields a Microsoft Kinect is used as way of tracking the shape and movement of the body of a performer as they move through the performance space.

3.1.1 Background

The concept for this project originated during a research period where various types of optical technologies were examined in order to determine their suitability for controlling a musical environment.

One early work examined was the Dimi-O instrument created by Erkki Kurenniemi which involved a video input being converted to synthesized sound (Kurenniemi, 1971). By having a dancer perform in front of a predefined line, the visual footage of the dancers movement was converted to notes played on a synthesized organ locked to a series of notes within a scale.

More recent derivatives of this optical method have involved technology like the Microsoft Kinect which is used to provide visual as well as depth information for musical performance. This technology has also allowed for tracking of a performer's body which allows for distinctions to be made between specific limbs used within a performance to control different elements. Performances such as the V Motion Project (Kofoed & Trott, 2012) and Chris Vik's demonstration of live looping in Ableton Live using a Kinect (Vik, 2011) demonstrate some novel uses of this technology, where the position of the performer's limbs are used

Fig. 9: Sonoromancer in action

to play or to control effects applied to an instrument, as well as to trigger clips and samples.

As mentioned earlier in the context analysis of non-physical instruments, the lack of haptic feedback and the difficulty involved in playing exact notes is a problem in this method of instrument performance. By extension, any instrument that functions by using external sensors observing the human body, rather than being directly operated upon by a performer will also suffer the same loss in feedback to the user.

Previous instruments constructed with Kinect sensors using the body tracking functionality can usually demonstrate a certain amount of guesswork on the performer's part in terms of for the intended output of a gesture. For example, the most intuitive actions involving the manipulation of parameter values through these systems work most effectively when the value is calculated from a relative position, such as how far one hand may be from the other. This can be attributed to the user engaging their spatial awareness to determine where their hands should be positioned to achieve the desired effect, rather than having to observe their position through a separate display. When observing their actions through external means, the performer adds an extra layer of abstraction between themselves and the instrument, as they need to consciously predict in advance when their actions will result in a visible or audible output. This creates an extra step of conscious thought for the performer, detracting from their ability to make musical choices instinctively.

This experiment approached this problem by removing the requirement for a performer to be very specific in their movements and to adopt fluid methods of movement that were mapped to ensure pleasing audio and visual outputs.

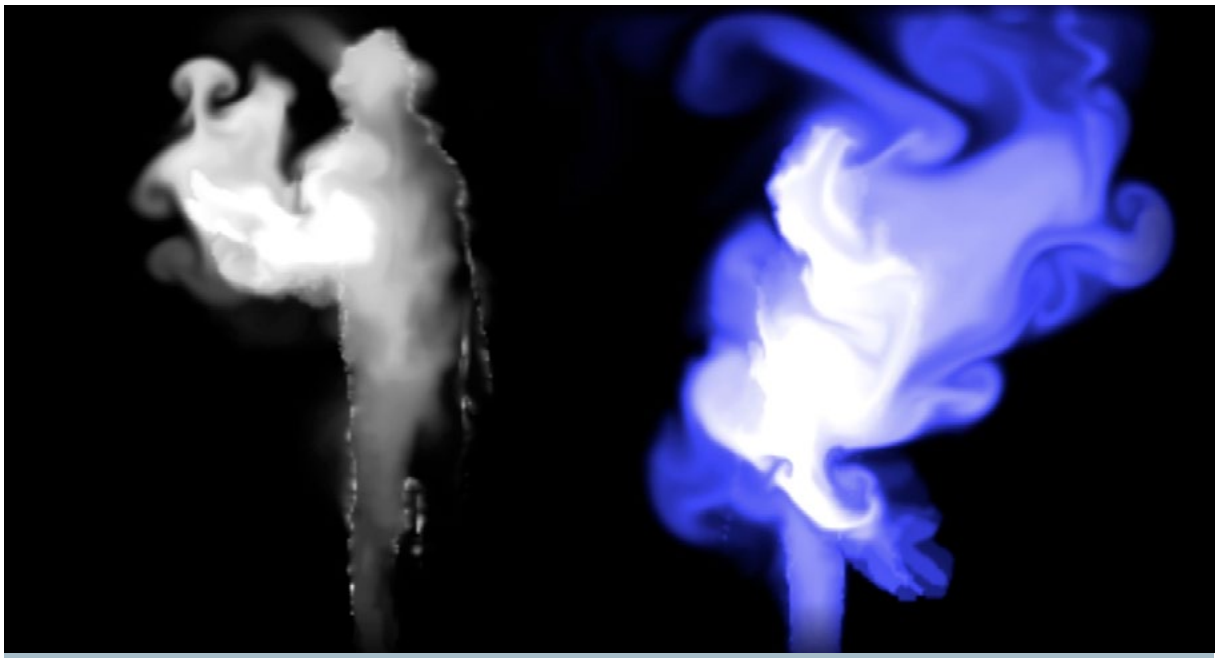

Fig. 10: Sonoromancer projected view

3.1.2 System

For Sonoromancer, only the Kinect depth sensor and colour camera were used. These were used to control the musical system using a more abstract method of body control rather than a per-limb approach. This led to the creation of an instrument that the performer "played" using a water simulation. The instrument analysed how the performer perturbed the water simulation and generated musical notes and textures from the result.

Whilst this may appear to be a counter-intuitive approach, especially regarding the previously stated observations reducing the number of layers of abstraction, this design decision was made in order to discover if there were techniques that could be applied to already abstracted systems in order to increase their intuitiveness. In this case, the simulation input method provided a different variety of gestures and movements a performer could utilize in order to control the instrument, rather than being concerned with triggering individual notes. This removed the requirement for the instrument to react instantaneously to input from the performer, and acted instead as an amalgamation of the performer's intent over time.

Rather than directly track the positions of each limb, the silhouette of the body was used as the main form of interaction, with the Kinect's depth camera used to recognize the silhouette of the performer (see Figure 11: Single update loop of Sonoromancer). This was then used to mask the video feed of the performer and to create motion vectors using an optical flow algorithm. These motion vectors were then used to stimulate the fluid simulation. The depth camera was also used to determine how close the performer was in relation to the front of the performance area in order to inject areas of high brightness into the fluid, which would then dissipate over time. The visual output of the fluid simulation was then passed through a blob detection algorithm in order to determine the shape, size, and position of the brightest areas of the fluid. These blobs were then mapped to a variety of parameters that were converted into MIDI messages and forwarded to Ableton Live to generate the musical output.

3.1.3 Interaction

The final version of Sonoromancer allowed a wide array of options for how a performer would move in order to operate the instrument. The performer could spread themselves out as wide as possible in order to inject a large amount of virtual dye into the system, and as a result, increase the size of the detected blobs. Fast motion would greatly stimulate the fluid system, increasing the rate of dissipation for the different blobs. As each blob would shrink in size and split into multiple small areas, the active musical note would jump around between the remaining areas of brightness and create arpeggiated runs of notes. Somewhat

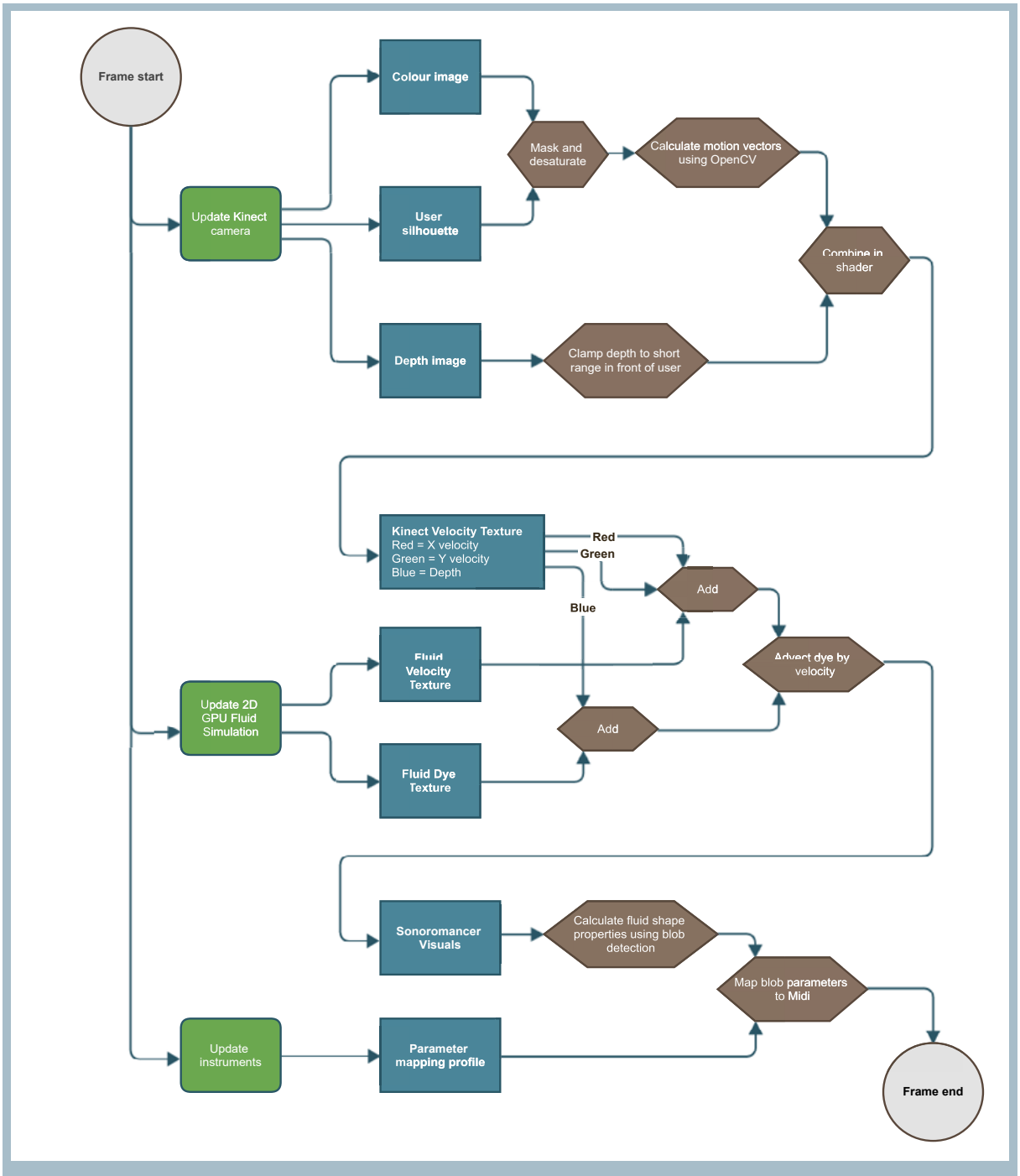

Fig. 11: Single update loop of Sonoromancer.

precise control could be obtained by the performer presenting a small profile to the camera by clasping both hands together and gently pushing them into the depth range that the camera would start to inject dye from. This motion could be extended into quick jabbing motions in different areas to quickly move between multiple pitches.

The audio environment consisted of a number of digital instruments provided by Ableton Live which were mapped to a variety of parameters provided by the fluid system. Each instrument featured different types of mappings for generating a variety of different musical patterns appropriate for the sound of the in-

strument and was locked to a scale of notes. Mapping parameters such as pitch to the horizontal position of the fluid blob would allow the performer to move horizontally to play specific notes, whilst mapping pitch to the size of the blob would create constantly evolving sequences of notes as the performer moved forwards or backwards within the performance area. The performer could also cycle between different instruments and fluid colours by pressing buttons on a discreetly held wireless mouse.

3.1.4 Results

The final version of Sonoromancer worked well as an experimental method for controlling single monophonic instruments, but due to its chaotic behaviour was not always the most reliable of instruments for playing a wide range of music. The most appealing sounding instruments designed for the interface were ethereal sounding synth pads and textures that would evolve gradually over time to create interlocking harmonies, rather than being used to play distinct and controlled melodies. The interface also had only rudimentary options for triggering or controlling elements outside of mapped parameters, so the performer was limited to either playing notes or cycling between instruments. Some test performances were attempted using live-looping, but they quickly became overly chaotic as no interface element was provided to control when loops were being recorded.

The chaos of the fluid simulation created appealing imagery for both the performer and audience, but removed all information about the musical workspace the performer was interacting with. This meant that the performer had to rely on the projected output of the fluid simulation in order to receive visual information about the state of the instrument. In some of the test performances, this meant that the performer often had their back to any observers, as they would need to observe the same visual output as the audience. This could be improved by having the output of the instrument projected onto the floor surrounding the performer in order to more strongly connect them to the visual dimension of the performance.

Fig. 12: Front view of TeslaRift in action at the Forks in Sockets performance event.

3.2 Experiment Two: Tesla Rift

The second experiment functioned as a research platform for designing VR interfaces that could be displayed and controlled using a variety of hardware. The final experiment delivered a number of controls that could be manipulated in a VR space in order to generate live visuals for a performance.

3.2.1 Background

The idea for Tesla Rift originated from my participation in the Tesla music performance *Forks in Sockets* (see Figure 12 on page 26). The performance revolved around musical and visual elements accompanying a 7-foot tall Tesla coil functioning as a giant square-wave synthesizer. The role Tesla Rift played in the performance was to control the generation of visuals that were then passed along to a VJ who mixed and displayed the result with other video feeds. When researching possible interaction methods, the question of using VR technology to display and control the generated visuals was raised which led to an evaluation of the available hardware approaches for both displaying and interacting with a VR space.

3.2.2 System

In order for Tesla Rift to function as an effective input mechanism using VR as a core design element the following components were required.

- Hardware to display the VR space and detect which direction the user was looking in
- Hardware input controls for interacting with the VR space
- Software to generate an real-time interactive VR space

To display the VR world, an Oculus Rift developer kit was chosen. The Oculus Rift is a head mounted display for delivering stereoscopic VR content that provides fast and accurate head tracking across 360° and a wide field of view. Since the Rift arrived for developers in April 2013 after a wildly successful crowd-funding campaign (Oculus, 2012), there has been a resurgence into VR research and content creation due to the Rift's inexpensive price (Kushner, 2014). Whilst the Rift did possess some drawbacks, such as a relatively low display resolution of 640x800 pixels per eye and no positional tracking of the user's head (only orientation) it served remarkably well in delivering a very believable VR environment to the user.

For manipulating the VR interface, a pair of Razer Hydra game controllers were chosen. These motion based input devices were initially developed as game controllers that used magnetic fields to track their exact position and orientation in space. Unfortunately the Hydra was limited by the lack of games that

Fig. 13: A Eurogamer Expo attendant tries an Oculus Rift HD prototype. (Bowles, 2013)

Fig. 14: Razer Hydra controllers (Castle, 2011).

Fig. 15: Razer Hydra usage (Castle, 2011).

properly implemented its motion control functions and was limited to being a novelty game control device for most of its market (Castle, 2011). Since the Rift was released, the Hydra and other types of motion controllers have seen a resurgence of interest from developers interested in experimenting with motion based controls for interacting with VR worlds.

In order to render the virtual interface, a game engine was required as the platform for rendering the environment. The Unity3D game engine was chosen due to its modular programming approach which helped speed the process of prototyping interactions. It was also chosen for its integration with both the Razer Hydra and Oculus Rift peripherals and included demonstration code examples which helped accelerate the development process.

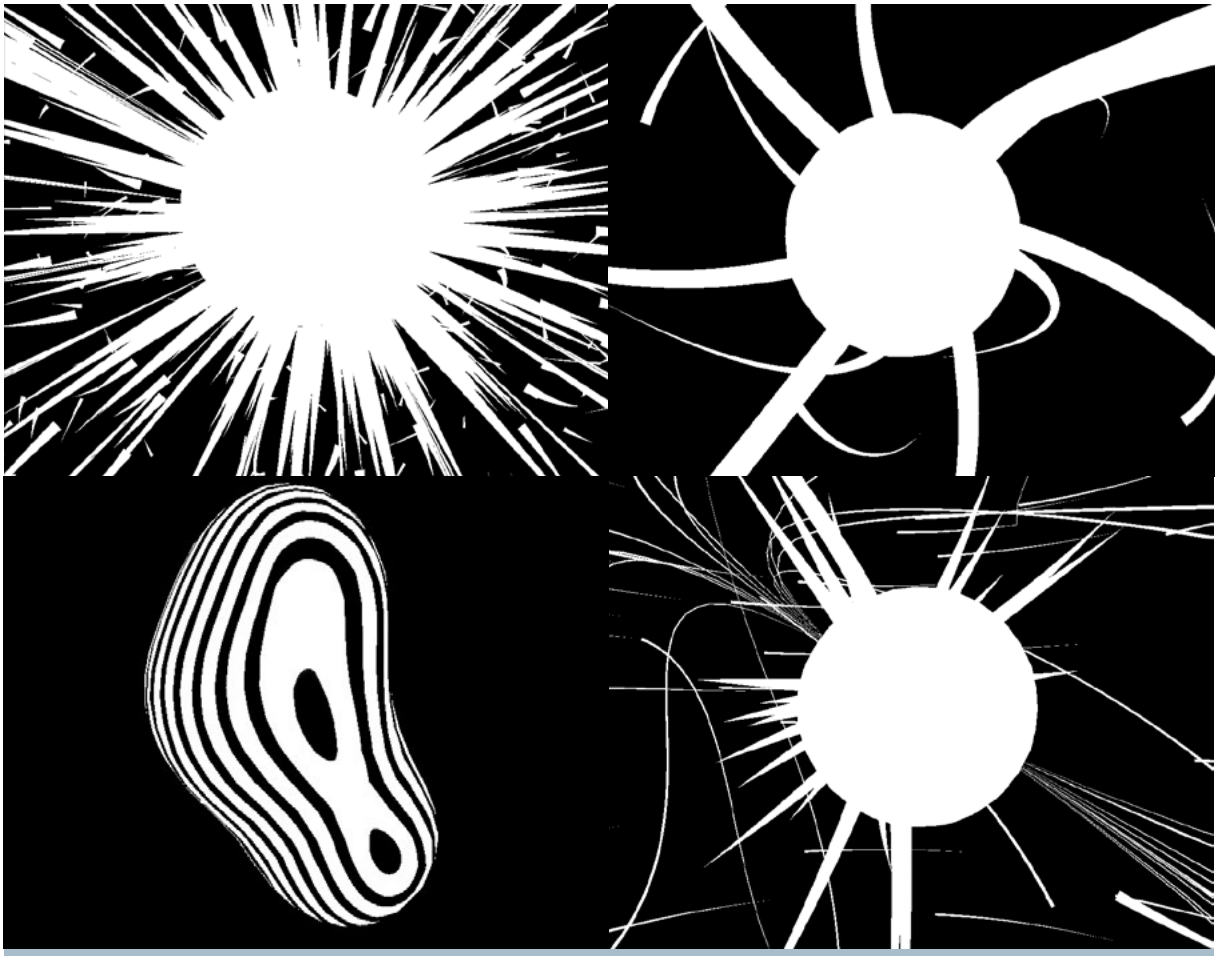

Fig. 16: Example audience-facing visual output of TeslaRift

The final design for the interface component of the performance system involved a number of movable floating orbs that contained a series of panels representing grouped controls for controlling the generated visuals. Each value represented by a panel could be individually scaled by grasping the panel and increasing or decreasing the distance of the hand from the control.

Values could be automated by connecting the panels to a number of floating discs that held different types of repeating patterns, such as a value that would spike upon every beat before quickly dropping away or a variety of different sine waves and values that would rise or fall based on input from external audio input such as a microphone. The performer would draw connections between the parameters surrounding an instrument to these value generators, which would then take over the responsibility of modifying the parameters based on a scale that could be controlled by the performer. Certain parameters functioned as buttons rather than scaling sliders, allowing for a greater variety of control methods for the performer to utilize.

Fig. 17: Preview of the performance interface being viewed through the HMD and the prototype glove input device used for Tesla Rift.

The output visuals revolved around a white sphere as the central element present in a black void (see Fig. 16 on page 29). The interface was used to change parameters of patterns displayed upon the sphere, as well as how its shape morphed over time. Parameters such as speed, angular velocity and pulsation rates could be controlled, as well as triggerable swarms of particles that would fly and orbit through the space. The entire visual experience was rendered using only white or black colours, allowing the visual output to be used by the VJ as live mask for their footage.

The parameters that were being modified in order to control the performance would send their values via the Open Sound Control protocol (OSC) to a copy of the VR environment running on the computer used by the VJ artist. However, the view was presented from a fixed camera angle that did not move as the performer looked around the VR space and also hid the controls that were being manipulated.

3.2.3 Results

Whilst Tesla Rift's interface for controlling visuals provided a novel experience, it contained some flaws in regards to usability. As each control orb could be moved around and placed using physics simulations, it was a time consuming process to organize the controls into arrangements that were close to the performer. The spreading of controls evenly over the surface of a 3D sphere made it difficult to determine what value was located on what control, as the most reliable method of determining this was to hold the orb close enough to read the labels on each panel. This was also due to the low pixel resolution of the Rift of 640x800 per eye, which made small text hard to read.

As the performer, one impressive element I noticed whilst using the system was how I would be affected by the generated visuals in some situations. When the environment was still or slowly changing the size and distance of the central sphere would be hard to establish due to its flat white colour on a black surface, removing my ability to use the shading upon the surface of the sphere to discern its 3D shape. By contrast, when the environment became swamped with swarms of particles my spatial awareness became acute due to the large number of objects present with varying depths.

The input controllers tracked the position of my hands quite well, but a lot of practice was required to memorize the correct button combinations need to perform tasks. The goal of removing this level of abstraction was resolved in later iterations of this project with the replacement of the input controllers with gloves that utilized bend sensors to detect specific gestures rather than relying on button presses.

The setup and configuration process to allow Tesla Rift to talk to external programs was frequently time consuming, due to reliance on using text files to describe parameters within the target of communication. This was later resolved with the invention of the Showtime protocol for Pensato which allowed for parameters to be exposed or controlled across a variety of devices, discussed in "5. Showtime Framework" on page 51.

3.3 Experiment Three: Pensato and Fissure

The final experiment was a VR interface for musical performance (see "4 Final Design: Pensato", 33).

3.3.1 Performance

Pensato was demonstrated by myself in a live performance at Raglan Roast, Wellington on the 19th of September 2014. The event involved a performance of Fissure which is an improvisational piece I arranged to demonstrate Pensato's performance capabilities, a projection screen that displayed to the audience a 2D view of what I was observing as the performer, and was followed by an informal audience QA session.

Audience feedback received from the event featured both positive and negative remarks. The audience found the novelty of watching a performer manipulate an invisible control space in order to create music interesting. However, audience members also reported that they expressed difficulty in connecting some of my movements as a performer to the musical feedback they were hearing. Even with the projection screen displaying what I was seeing as a performer inside the VR headset, actions such as opening and closing menus or reconfiguring interface elements were interpreted as musical gestures but didn't result in the creation or manipulation of sound.

3.3.2 Results

Using the feedback from the previous performance, I created the final video performance of Fissure, a link to which is available on page C. The performance featured three wall-sized projections located directly in front of the performance area, in order to make the connection between my gestures and the changes they would inflict upon Pensato's performance environment easier to observe.

4. Final Design: Pensato

I present Pensato as the final compositional output of this thesis. Pensato in Italian translates to English as the word “thought”, and was arguably established as a musical direction by Anton Webern, described by Perle (1995) as “a note which was to be imagined, not played” (p 149). It is through this definition that I envision and hope how the VR interface will be interpreted by an audience, by appearing as if the performer is skilfully manipulating an imaginary entity.

Pensato is a VR interface that acts as a hub for multiple audio devices. These devices are represented by a variety of VR controls and can be manipulated in a variety of ways using the interface to perform musical works. By using the lessons learnt from my previous VR input experiments, the interface has been designed to leverage the advantages provided by such a VR space and associated technologies.

This chapter will cover the following topics:

- The relationship between Pensato and external music software.
- The tools Pensato provides for authoring configurations of VR controls and how they are used during a performance.
- The method of interaction through the use of natural gestures.
- The organisational principles adopted within the VR environment to reduce confusion.

4.1 Audio Software Relationship

In order for Pensato to satisfy its purpose as central hub for controlling a musical performance, an external program was required for generating the audio output. The software chosen for this purpose was the Digital Audio Workstation (DAW) Ableton Live due to its effectiveness as a real-time musical performance platform. Live's musical controls, organisation and methods of execution were analysed throughout the design process of Pensato in order to create a family of VR controls that would easily map to most of Live's core features. The core features that were targeted for implementation were the organisation of multiple instruments into 'tracks', the manipulation of sound parameters through 'devices' and the arrangement of multiple musical ideas and sections into 'clips'.

Fundamentally, the goal of music software is to produce a mixed waveform that when played through speakers can be interpreted by the human ear as music. Live accomplishes this through allowing the combination of pre-recorded audio samples and synthesized instruments. Samples can range from a single sound clip being shifted in pitch to play multiple notes, to loops and single hits of recorded or synthesized sound that are then layered together to form the musical composition. Synthesized instruments focus on manipulating a variety of parameters to generate a particular waveform and can offer a vast amount of customization to obtain an ideal sound. Live can also be used as an audio editing and mixing environment for creating musical compositions, rather than exclusively as a performance environment.

In Live, samples and synthesized instruments can be played directly using hardware such as MIDI keyboards or through recorded clips which represent recorded sequences of notes or parameter data. One of Live's main strengths that elevates it as a tool for performing music is how it organises these clips in a mode called the "session view" (see Figure 18). The structure of the session view and how it organized its musical sub-objects was adopted as the main way of grouping multiple related VR controls together in a hierarchical manner.

Figure 18: Ableton Live displaying the session view.

The session view comprises a matrix of instrument tracks arranged in columns. Within each cell of the matrix, a clip can be placed which upon triggering will play either a sequence of MIDI notes stored in the clip, or will play an associated sample. Each vertical track represents a grouping of clips that either represents a single synthesized instrument or a defined grouping of samples and is limited to one playing clip at a time. Each horizontal row is called a "scene" and can trigger each clip located upon the row simultaneously, allowing for rows to represent separate musical sections. Each track can contain multiple "devices", which are small modules that transform either the incoming note information or the sound waveform. These devices possess a number of parameters which can be mapped to external MIDI messages allowing for real-time manipulation of the device's properties and behaviour.

The session view was adopted as the inspiration for Pensato's organisational structure as it provides a logical structure for grouping together instruments (tracks), controls (device parameters) and musical ideas (clips). These elements were also chosen as they could be represented using simple graphical elements in the VR space that were easy to move and operate using natural gestures.

4.2 Basic VR Controls

The VR controls comprise the minimum set of objects required to interact with Live, whilst allowing flexibility for improvisation or variance during a performance. In Pensato, device parameters are represented by sliders, clips are represented by buttons, and both parameter sliders and clip buttons are grouped underneath an instrument. Controls map to the Live object they are meant to represent using the Showtime framework (see "5.3 Pensato-Showtime network", 58).

4.2.1 Slider

The slider control is the simplest control available that can manipulate a single parameter value. The value of the parameter can be modified by sliding a finger across the slider in order to smoothly set the value of the parameter to a value between a minimum and maximum amount (see Figure 19). The slider will also update and show the real-time value of the linked parameter if it is modified externally.

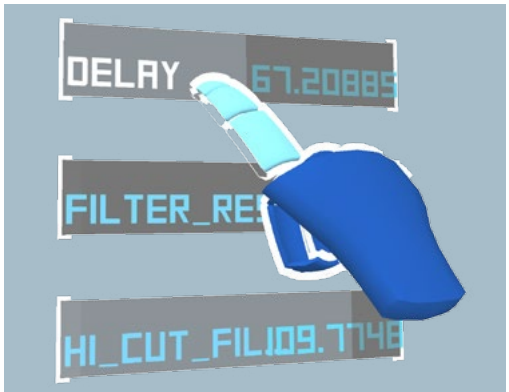

Figure 19: Slider control.

4.2.2 Button

The button control represents a single momentary or toggle-able action that can be triggered by pressing it with a finger (see Figure 20). When used to trigger clips within Ableton Live, the button will light up with a partially transparent colour when the associated clip is queued to play on the next beat, and will light up with an opaque colour when the clip is actively playing.

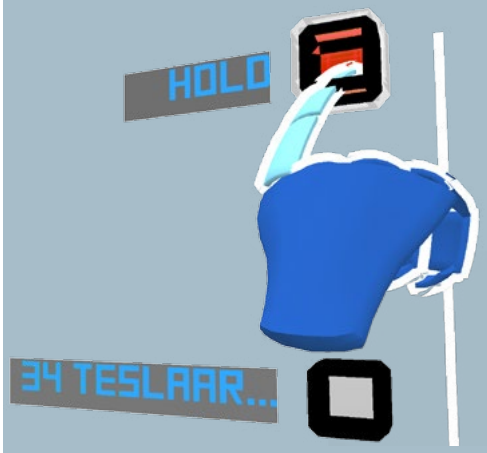

Figure 20: Button control.

4.2.3 Instrument

The instrument control groups all parameter sliders and clip buttons associated with a track into a single object. Tapping the instrument will display all child controls grouped underneath this object, allowing them to be manipulated (see Figure 21). The instrument also functions as a template from which these child controls can be copied and inserted into other widgets to create custom layouts (see "4.3 VR Performance Widgets", 38).

Figure 21: Instrument control.

4.3 VR Performance Widgets

Whilst it is fully possible for a performer to only use the provided slider and button controls for modifying external parameters, each control is only bound to one distinct parameter or action. It can become difficult to accomplish complex sequences of actions without having a vast number of individual controls laid out in front of the performer. To overcome this, Pensato provides a number of more complex controls named widgets which allow for complex groupings of controls to be manipulated simultaneously in a variety of ways.

Each widget can be populated with a number of child controls. Child controls are created by the user copying an existing template control that is located within the menu layer (see "Menu Layer", 48). This is achieved by the user dragging a copy away from the template control and releasing the freshly created clone on top of a widget (see Figure 22). The act of storing one control or widget within another is called "docking", and allows the user to create widgets that represent distinct musical ideas. Each different widget is designed to trigger its child controls in a unique way, and can even contain other widgets as children which contain their own child controls. This allows for complex nested layouts to be formed where a single action performed upon a widget can perform a large number of different actions.

In order of increasing complexity, the widgets that Pensato offers are as follows.

- Value Trigger
- Pump
- Matrix
- Radial Basis Function Sphere

Figure 22: A Selecting control. **B** Tearing off clone. **C** Docking into widget.

4.3.1 Value Trigger

A value trigger widget acts as a storage location for other controls (see Figure 23). All child controls added to the trigger will simultaneously perform their associated actions when the trigger is activated by the performer. The trigger will also store the value of a control that contains a variable parameter when the control is added to the trigger as a child. The trigger will set the parameter associated with the child control to the stored value when the trigger is activated.

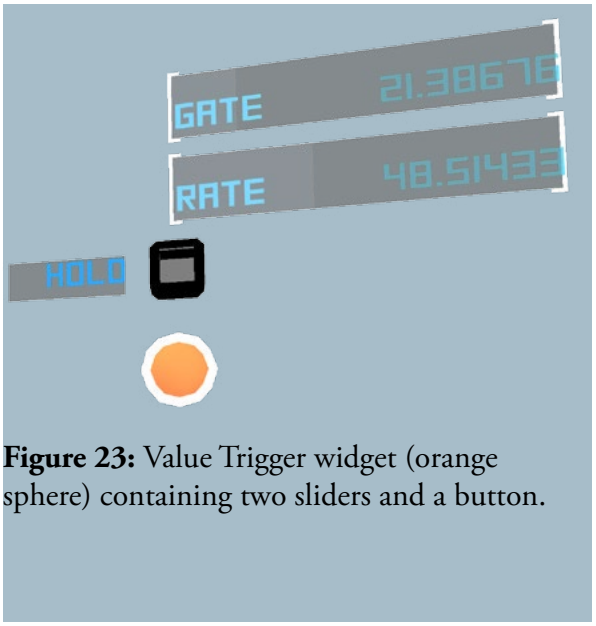

Figure 23: Value Trigger widget (orange sphere) containing two sliders and a button.

This type of behaviour can be used to move between sections of a song. By triggering a value trigger containing a number of clip buttons, all of the children would queue the necessary clips to play at the same time. The performer could also swap an instrument between different patterns of stored values in order to quickly swap between different configurations of sound.

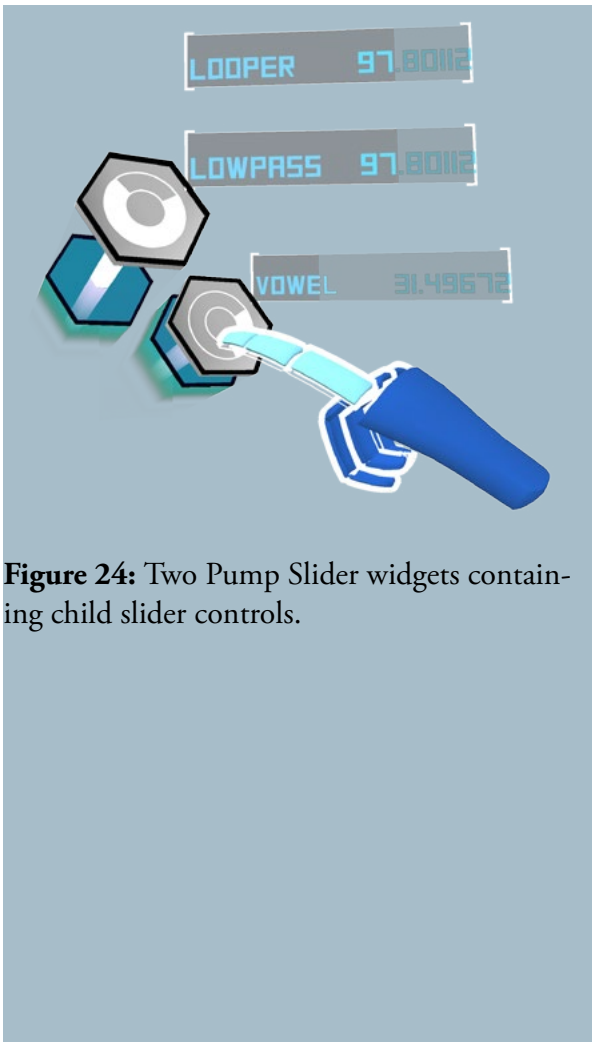

Figure 24: Two Pump Slider widgets containing child slider controls.

4.3.2 Pump Slider

The Pump Slider widget functions as a three-dimensional version of the simple slider control (see Figure 24). The performer can grab the top of the pump and slide it into and out of the base of the widget which changes the value of the pie-percentage circle on the pump's cap. This value will be sent to each of the pump's child slider controls, allowing them to be manipulated in sync with each other.

This widget can be used as a holder any number of slider controls, allowing the performer a way of grouping together multiple variable controls that need to be operated simultaneously.

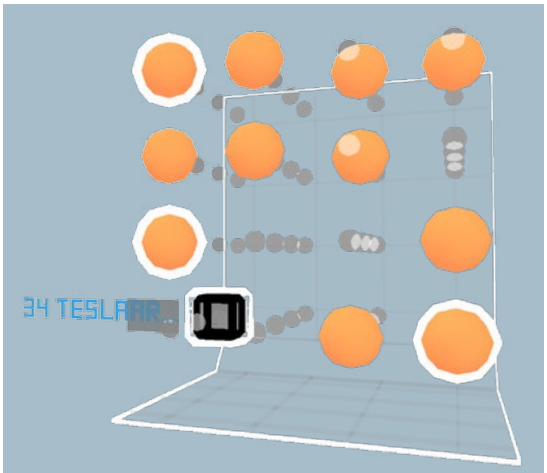

Figure 25: Matrix widget containing multiple value trigger widgets and button controls.

4.3.3 Matrix

The Matrix widget is a 4x4x4 cube that can contain either value trigger widgets or button controls as children (see Figure 25). Whilst each child control can be triggered independently, the matrix widget also allows for controls to be triggered using individual fingers. The position of the performer's hand relative to the cube selects how far back and which row to choose a control from, whilst each finger the performer curls (ranging from index to pinky) will select the column. This action visually appears from the outside as if the performer is "playing" each control on a row as if it were a piano.

The design of the matrix widget was inspired by hardware music controllers such as the Novation Launchpad (see Figure 26). This controller consists of a 2D grid of buttons that can be assigned to trigger clips and controls inside of audio software. The advantage provided by these controllers is similar to that of a MIDI keyboard, but the organization of buttons in a grid rather than a linear arrangement makes it easier to memorise which rows and sections are relevant for the current part of the song the performer is playing.

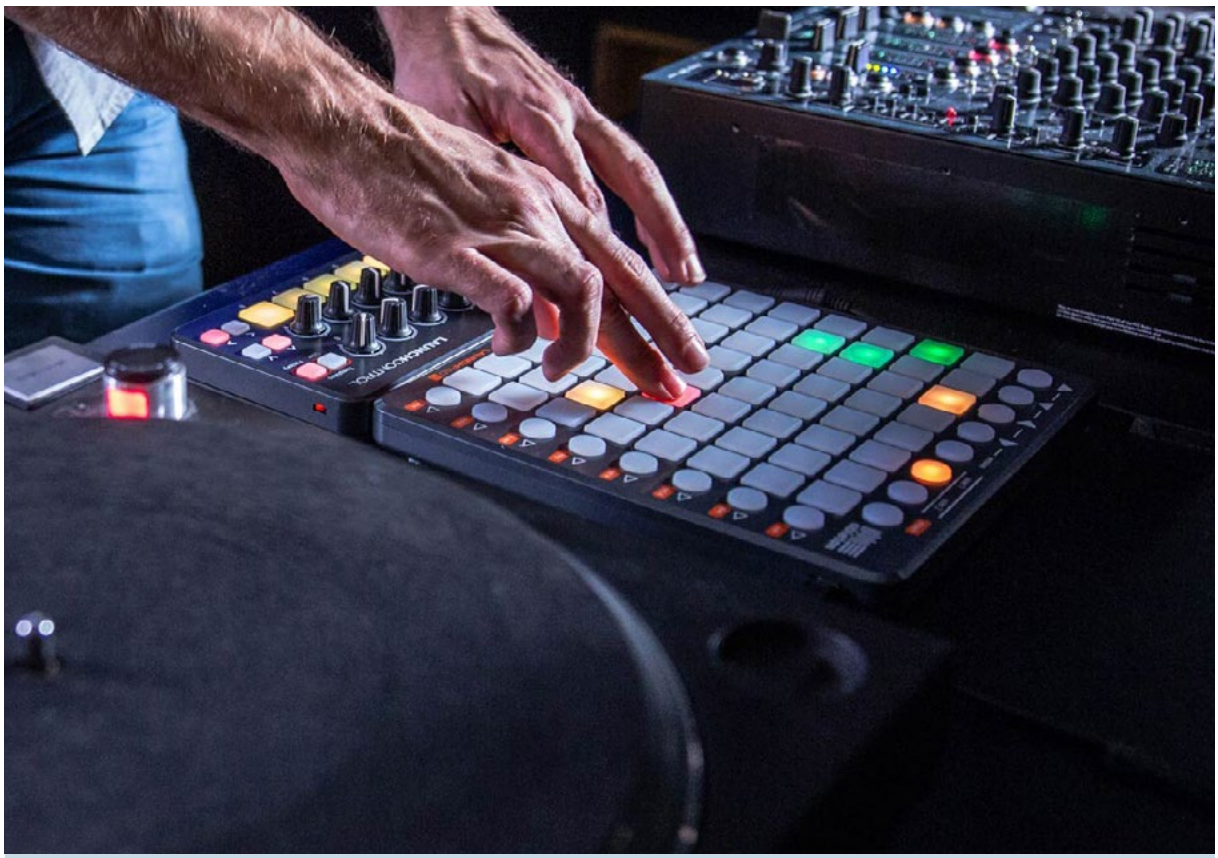

Figure 26: Novation Launchpad controller (Novation, 2014)

Figure 27: RBF Sphere with three docked plug widgets containing a number of slider controls

4.3.4 Radial Basis Function Sphere (RBF)

The RBF sphere is the most complex widget available in Pensato (see Figure 27). It allows the performer to interpolate between values for multiple plug slider widgets using positions in space. This is carried out by a trained Radial Basis Function network which allows for multidimensional interpolations to be performed between a number of inputs and outputs (Broomhead & Lowe, 1988).

In order to train the widget, the following steps are carried out.

- The performer adds a number of pump slider widgets already populated with one or more slider controls into the sphere's region of influence. The pumps will function as the outputs of the RBF network. The pumps will also orient themselves facing outwards from the center of the sphere in order to keep the space visually organised.
- The performer creates anchor points within the sphere's volume that will serve as training points for the RBF network.
- The performer selects one of the anchor points, then alters the value of the pumps. The position of the anchor point is saved as the training point's input value, whilst the values of the pumps are saved as the training point's output values. This is repeated for every anchor point.

When the widget is fully trained, the performer controls the sphere by placing their hand within the volume and forming a pointing gesture. As the hand is moved within the volume, the position of the extended digit is fed into the RBF network which outputs a series of interpolated values for the pumps. The closer the finger is located to one of the anchor points, the more the trained pump values associated with the anchor point's location will influence the active value of the pumps.

4.4 Gesture Fundamentals

The types of gestures used within Pensato are primarily spatial. This means that the user has to 'touch' the virtual controls with their hands directly, rather than using generalized 'magic' gestures (see "2.4 Methods for interacting with VR spaces" on page 17). The motivation behind designing the system this way was due to the control qualities afforded by the gloves for VR immersion. During early tests, methods of control that manipulated objects through proximity rather than direct contact were sometimes awkward to trigger or operate and thus many gestures were rewritten to function only upon touching an VR control.

To detect gestures, I constructed a pair of VR gloves from the Razer Hydra controllers used in "3.2 Experiment Two: Tesla Rift" on page 27. The gloves were constructed by coupling the position and orientation trackers from the Hydra controllers with bend sensors embedded within a pair of dual-layer gloves. Gestures were calibrated using an RBF network (see "4.3.4 Radial Basis Function Sphere (RBF)", 41) which was trained by associating each programmed gesture with values reported from the bend sensors located within each glove.

The different types of gestures provided to the user are generally consistent throughout the operation of the system. This design reflects the common UI scheme afforded by mouse driven systems where a small number of individual actions can be performed using the input device, but expand to a great number of actions depending on the context where an action is used. For example, when using a mouse, a left click signifies the selection of an object, and a right click signifies some sort of secondary action.

Each gesture is linked to a virtual hand which mimics the pose of the performer's hand in the real world. To help the user distinguish when a gesture has been recognized, the appropriate fingers that make up the main form of the gesture, such as the index finger for pointing, or the index and middle fingers for scrolling will light up upon the virtual hand. This is to help reinforce to the performer which gesture they have active at any time, since forming the wrong gesture by accident could cause unexpected reactions from the virtual environment.

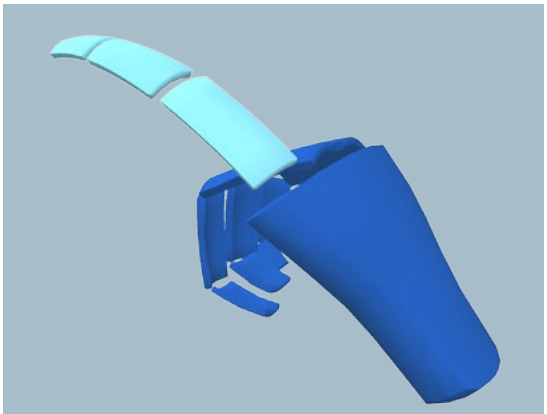

Figure 28: Index Point gesture

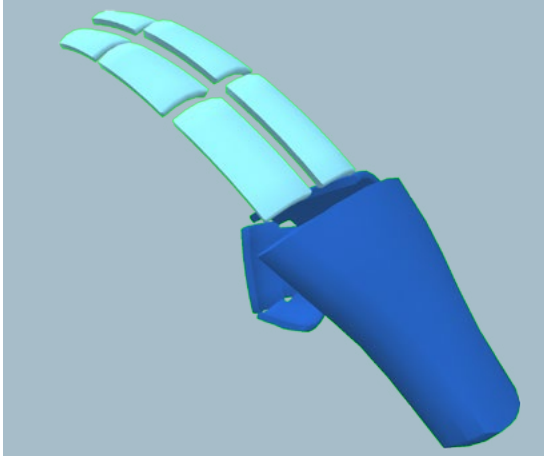

Figure 29: Two-finger point gesture

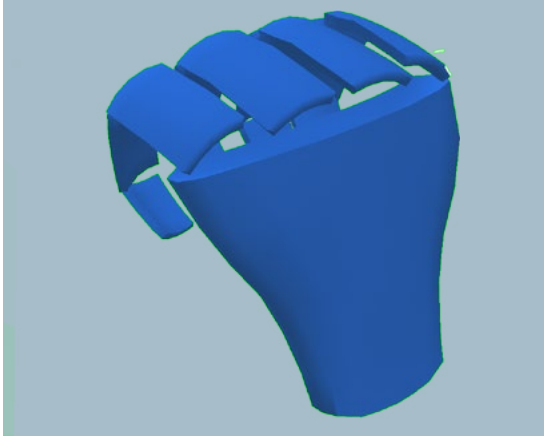

Figure 30: Grasp gesture

4.4.1 Index Point

Pointing is a natural action that someone performs when indicating an object of interest or for controlling something that requires the precision of a single digit. This gesture was chosen as the method of selecting VR controls and triggering objects to activate their primary actions (see Figure 28).

4.4.2 Two-finger point

This gesture was chosen due to its use in modern multi-touch laptop trackpads for scrolling content in a direction (see Figure 29). This gesture has a real-world equivalent of sliding a piece of paper across a surface, where two fingers are used for stability. In Pensato, it is used to scroll through stacks of VR controls, allowing for elements out of reach to be brought into close enough proximity for the user to manipulate.

4.4.3 Grasp

A grasping gesture is typically used when someone wishes to move a physical object from one location to another. By wrapping their fingers around an object, a person creates a stable attachment between their limb and the object they wish to relocated. This gesture is used within Pensato to move objects around within the VR space (see Figure 30).

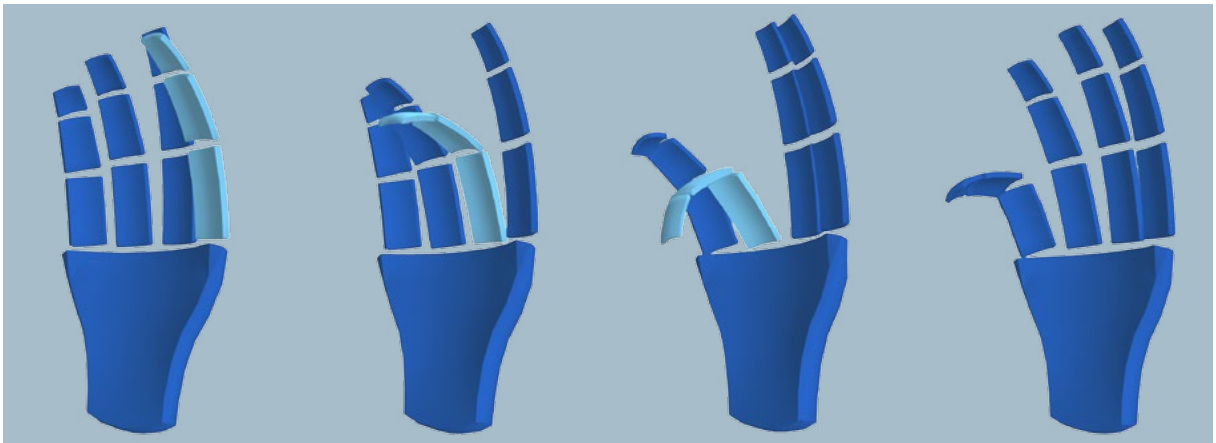

Figure 31: Playing gestures

4.4.4 Individual finger controls

This gesture (see Figure 31) mimics the finger motions of someone interacting with an object that has a large number of controls parallel to each other. This is similar to the action performed when playing on a piano or typing on a keyboard. In Pensato, this gesture is used to control widgets that have intractable sections located in close parallel proximity to each other. It is can also be used to rapidly trigger certain controls where a single digit is not fast enough to perform a repeated action but multiple digits can provide increased speed and less fatigue.

4.5 UI Design

The goal of any user interface is to present to the user a set of objects that either display information, or provide methods of modifying that information. VR provides its own unique set of challenges when designing an effective UI, as well as some of the typical challenges associated with creating 2D interfaces. These include determining the most efficient way of delivering required information to the user, whilst also maintaining a coherent organized interface that delivers information in the most contextually appropriate manner. With VR, obviously these challenges need to be solved in 3D space rather than 2D space. Perspective needs to be taken into account where previously most 2D interfaces would be presented using orthographic methods.

In the context of interface design for music performance, we already have two domains for interface design that we can draw from, hardware and software. Hardware music controllers usually contain buttons or variable controls, such as knobs and faders. More exotic controls such as touchpads also exist, such as in the case of the Korg Kaossilator series of controllers, which can allow parameters to be modified in a two-dimensional space using a single finger (see Figure 32).

Hardware controls provide the closest approximation of a traditional instrument, as the performer will receive appropriate haptic feedback from touching the physical controller. This will help them to learn how to fluidly play their hardware controls through associated muscle memory, just like a traditional acoustic instrument. However, they can be incredibly difficult to modify, especially when using professional production level controllers. Custom built hardware controllers are more flexible in regards to the options provided to the musician, but are still difficult to modify or change on the fly whilst experimenting with musical composition, where the performer needs to stop playing to fabricate or rearrange their controls setup.

In recent years, the introduction of ubiquitous touch screen devices has provided the opportunity for software developers to create custom music controller surfaces that use controls modelled after physical equivalents, but with the flexibility of having multiple configurations that can be swapped between as required. For example, interfaces such as touchAble (Zerodebug, 2013) and Lemur (Liine, 2014) both

Figure 32: Korg KP3 and Kaossilator Pro touchpad audio controllers.

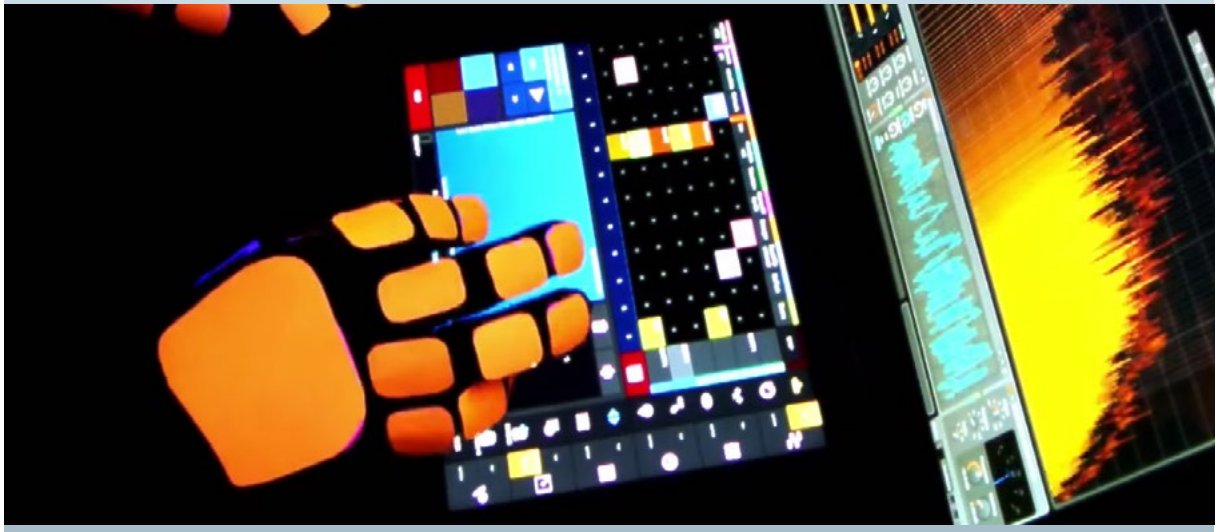

Figure 33: Steo Le Panda performance using touchAble (Panda, 2014).

provide configurable touch screen interfaces for interacting with a variety of media applications. Using VR we can adopt the advantages afforded by natural gestures used in touch screen interfaces, but extends them to the full surrounding 3D space of the performer, rather than being constrained to the physical dimensions of a screen.

An iterative design process was used to construct Pensato's user interface. The earliest version starting in Tesla Rift used a free-form method of organising controls within the VR space. Controls could be placed at any orientation and position in order to create a completely open interface. This ended up being a detrimented quality however, as micro-adjustments needed to be made in order to establish a suitable layout. Since each control existed independently of each other, there was a lack of visual cues providing any sort of structure and this caused confusing visual clutter. This also occurred when multiple menus or controls were overlaid on top of each other, frequently resulting in a confused performer.

To solve this, the final iteration of Pensato uses a number of features to help maximise the legibility and utility of the VR space: using a hovering mechanism to reduce the amount of information displayed except when required, orientating 3D controls and towards the user to increase readability and the organisation of the VR space into layers to compartmentalize functionality (see "Organisation by Layers", 48).

4.5.1 Hovering

One of the disadvantages noted early on in the design of the VR user interface, is that overwhelming the performer with a large number of open menus impedes productivity, due to the 3D nature of the menus making it sometimes hard to determine their placement, especially when cloning and docking controls. To solve this, all controls that involve scrolling functionality, or are used to create copies of template controls, are displayed on a hover or one-at-a-time basis. When the performer places their hand over any of the closest objects that create new controls, the scrolling menus will only open whilst the hand is located over the control in question, and will close if moved too far away. This limits the number of open control menus to one open per hand.

Interacting with instrument menus works on a similar basis, selecting an instrument will display its available clips and parameter sliders, but selectively tearing off a copy of one of the instrument's controls will close the menu, allowing the performer to choose a destination for the freshly cloned object. Interacting with another instrument will also only display one menu per hand.

4.5.2 Orientation

In order to impose some order upon the 3D space surrounding the performer, most controls will perform a degree of self-organisation. Scrolling menus will rotate to face the eyes of the player so that the largest portion of the intractable surface is presented at all times. The rotation happens on a slight delay and only occurs on the yaw axis, so dragging and rotating controls that automatically rotate will still have a sense of 3D depth and motion, particularly in the case of thin controls such as sliders.

Some controls that rely on interaction with the performer using a facing surface, such as the Matrix widget, will use auto-rotation in order to present the most important surface first, whilst free floating widgets like the Pump and Value Trigger widgets can be freely orientated. In the case of controls that function along an axis for interaction, this allows their values to be manipulated using gestures in controllable directions. By orientating a Pump widget upside down, a downwards swiping gesture will scale the value positively, whilst orientated in the opposite direction will scale the value with an upwards motion. Base VR controls such as clips and sliders and instruments will always face the user whilst they are docked within a widget, regardless of the widget's orientation. This is to allow the user to read the value of the control at all times.

4.6 Organisation by Layers

The controls in Pensato are organised by a hierarchy of ring layers. These layers are arranged in a circular formation around waist height and within easy reach of the performer. Previous iterations allowed for controls to exist outside of the reach of the performer requiring the use of either magic gestures to bring controls closer, or have the performer stretch uncomfortably to bring the controls within arms reach. This occasionally caused some safety issues, as the performer would accidentally push their hands into real-world objects that they could not see. By organising controls in a variety of depths, the virtual environment gains cues that help trigger the user's depth perception and help them become more spatially aware of their surroundings. Close objects will trigger a stronger feeling of depth perception than far away objects.

4.6.1 Menu Layer

The closest layer to the performer contains templates for creating widgets or instruments (see Figure 34-A). By hovering their hand over each template spawner, a scrollable menu will appear that provides copyable templates the performer can drag into the workspace, resulting in a new intractable control. The layer also contains buttons that affect the entire workspace on a global level. These include a button that switches between the performance and editing contexts, a scrolling menu that controls what saved layout is active, and a trash widget that will destroy controls dragged onto it.

4.6.2 Instrument Layer

The instrument layer consists of the instrument dock, a widget which provides a place for the performer to store instruments in to make them easier to access (see Figure 34-B). Each docked instrument can be opened with the use of a pointing gesture which displays two menus side by side. These menus are individually scrollable with the two-finger scroll gesture, and provide both a method for controlling the instrument directly, or a source for tearing off copies of parameters and clips for docking into other controls.

4.6.3 Widget Layer

The widget layer lies in the space behind the first two layers (see Figure 34-C). Controls can be cloned from the control template source and placed anywhere in the 3D space surrounding the performer where they can physically reach, but only custom controls can live in this space. Instruments, clips and sliders for controlling parameters can not exist outside of a control and thus must be paired with an associated widget for manipulation.

Figure 34: Menu Layer (A), Instrument Layer (B), Widget Layer (C)

5. Showtime Framework

When working with musical software and hardware devices, a robust method of communication is required to connect together all devices into one cohesive musical environment. In order for Pensato to function as an effective performance platform there were a number of requirements that needed to be satisfied when choosing an appropriate communication method. These requirements were that a communication system had to be fast, descriptive and flexible.

- The messaging system would need to provide a consistently fast throughput of data in order for actions sent to and from devices to instantly create audible responses.
- It would also need to be descriptive and allow for devices to provide information about their capabilities so that the architect of the musical environment can appraise what functions are available across all the devices present.
- Finally, the messaging system would need to provide the ability for multiple devices to talk to each other bi-directionally with a minimum of configuration. It must also work across a variety of programming languages and work with existing musical message protocols.

5.1 Existing messaging systems

Two existing musical communication systems were experimented with during the early design stages of Pensato, MIDI and OSC.

MIDI (Music Instrument Digital Interface) has remained the standard messaging protocol for interfacing multiple digital instruments since it was introduced in 1983 (Loy, 1985). It is a unidirectional messaging format that allows the transmission of note and musical parameter data between either multiple instruments or computers and has achieved ubiquitous status in music software and hardware.

Ableton Live uses MIDI extensively by allowing many mappable parameters within the program to be controlled by or triggered by MIDI messages. Whilst common across almost all musical devices due to its age and small footprint, MIDI does not provide a particularly wide throughput for modern data transmission as the format runs at 31.25 baud (Loy, 1985), a speed appropriate when the format was introduced but lacking for some modern usages. A MIDI connection can be overloaded by sending too many messages simultaneously or in rapid succession. Its unidirectional nature also makes it difficult to set up complex routings between devices.

OSC (Open Sound Control) is a messaging format which allows for descriptive addresses to be used when sending information between devices. The format is networked based and so can leverage network transport mechanisms for connecting devices together. It uses human readable addresses using a tree structure followed by an arbitrary value (e.g.: "/device/parameter 27"). Unlike MIDI, with its single pipe structure, OSC operates using a server/client structure where an OSC server can receive messages from multiple OSC clients using the UDP network transport.

Whilst Ableton Live did not natively support OSC, the LiveGrabber plug-in was used during the initial development of Pensato in order to connect the VR interface to multiple parameters within Live. This approach was eventually scrapped due to the tedious amount of changes that were needed to be performed by hand inside Live to correctly map incoming OSC addresses to individual parameters.

Even though both OSC and MIDI offered a number of advantages due to their pedigree as established music communication protocols, their limitations were inhibiting the types of features that could be implemented within the VR space. To answer Pensato's needs for a fast, descriptive and flexible method of communication between itself and multiple musical devices, the Showtime messaging system was developed.

5.2 Showtime

The Showtime framework is a cross-platform software library designed for connecting together any number of performance devices or software programs in a manner that allows for easy configuration and inter-communication. The framework was designed in mind for performances both figuratively and practically. “Nodes” (representing multiple performers) exist on top of a “stage”, which provides the context for the performance. The framework accomplishes this by forming a structure of connectable nodes that can broadcast messages to the entire performance, or instruct other devices to respond to certain inputs.

5.2.1 Node

A node represents a single device within the performance that possesses a number of controllable or broadcastable actions. The types of actions that can be performed by nodes will be explained using an orchestral performance as a metaphor.

- Nodes contribute to the performance using broadcast actions, akin to how a instrument in an orchestra contributes to the overall piece. In this case, the sheet music and performer ‘trigger’ the instrument to ‘broadcast’ a sound, and the sound is ‘observed’ by an audience
- Nodes expose controllable actions that can be operated by other remote nodes. This is comparable to how a section in the orchestra will react to a visual command from a conductor, such as a visual command to raise or lower their volume. In this case, the conductor is a control node who is requesting the performer nodes to run their control actions related to volume manipulation.

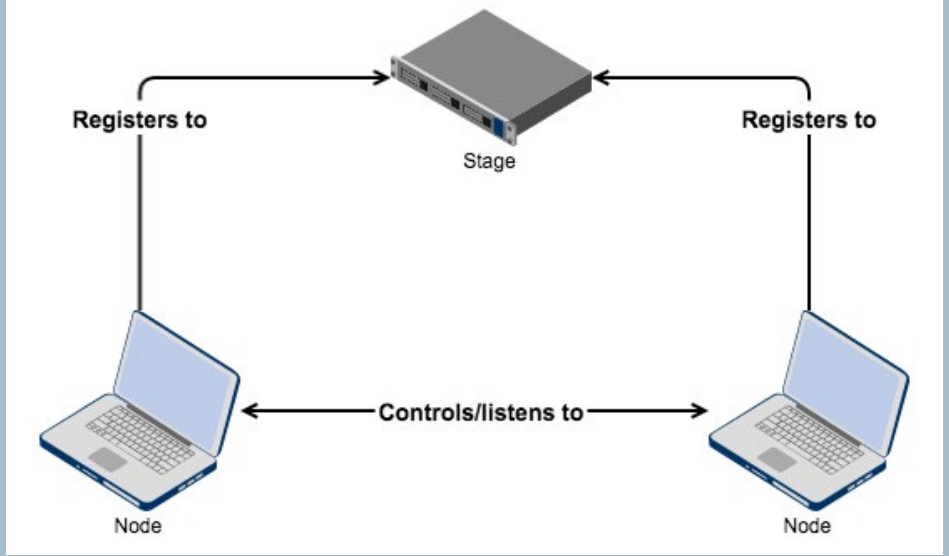

Figure 35: A simple Showtime network.

The actions that a node possesses are remotely triggered by other nodes using Remote Procedure Calls (RPC), which is defined as "a procedure call in which the actual execution of the body of the procedure takes place on a physically distinct processor from that on which the procedure call takes place" (Daintith & Wright, 2014). However, unlike standard RPC methods, a node's broadcast actions are closer to an event model, where messages are announced globally to all interested parties. This hybrid approach of both RPC and broadcast events allows for complex arrangements of nodes to be created that are not hindered by solely adopting either approach.

In order to integrate devices into a Showtime network, nodes can be added to a host application using the host's specific scripting language. Nodes can be written in a number of languages including Python, C#, Java and Processing. The hosting application can then trigger the node to broadcast actions when certain events occur within the host, or the host can register an local function to be ran when a specific control action is triggered by a remote node.

5.2.2 Stage

The stage is a dedicated node which acts as a directory service for the entire performance. When started, each node will connect to the stage and register both the network address and port of the node, as well as list of all of the RPC actions it will either broadcast or respond to. This allows nodes to enter the performance through a single fixed IP address rather than having to connect individually to other node's addresses. Nodes can then query the stage for the addresses of other nodes and obtain lists of what actions they will broadcast or respond to.

5.2.3 Creating a Showtime performance network

The process involved when creating a Showtime network passes through a few stages ranging from nodes registering to the stage directory to connecting to nodes creating and responding to connections made between each other. An example follows where two nodes are used to form a small performance network.

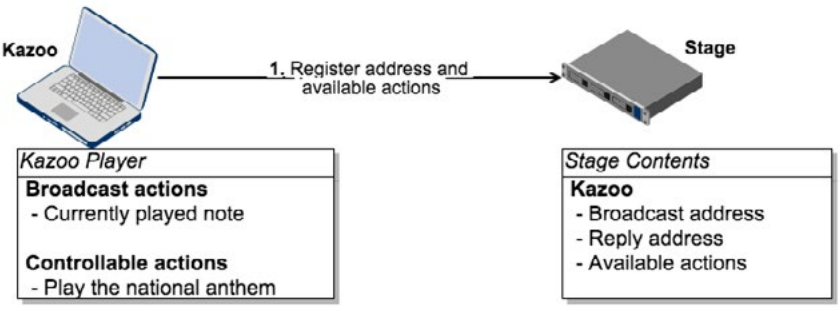

Figure 36: Example of a node registering itself on the stage.

To join the network, a node first needs to register itself to the stage. In this example, a Kazoo player node wishes to join the performance. The Kazoo player registers three IP address and port combinations to the stage (Figure 36) including: a socket that other nodes can connect to in order to receive outgoing broadcast actions from the Kazoo, a socket that will accept messages from other nodes to trigger control actions, and a reply socket that allows the Kazoo to receive connection requests from other nodes and reply with a confirmation. The stage also receives the node's available broadcast and control actions.

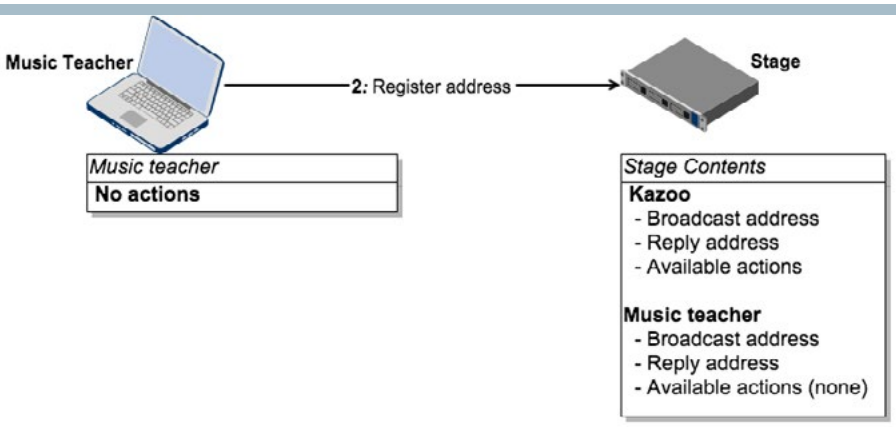

Figure 37: A node registering to the stage without any actions.

A music teacher wishes to join the performance (Figure 37). This node will be used to issue control actions to the Kazoo player and will also listen to the Kazoo player's broadcast actions. Whilst the music teacher does not possess any actions, it has broadcast action socket that can publish control action requests to other nodes, and a control action socket that receives broadcast actions from nodes that it subscribes to.

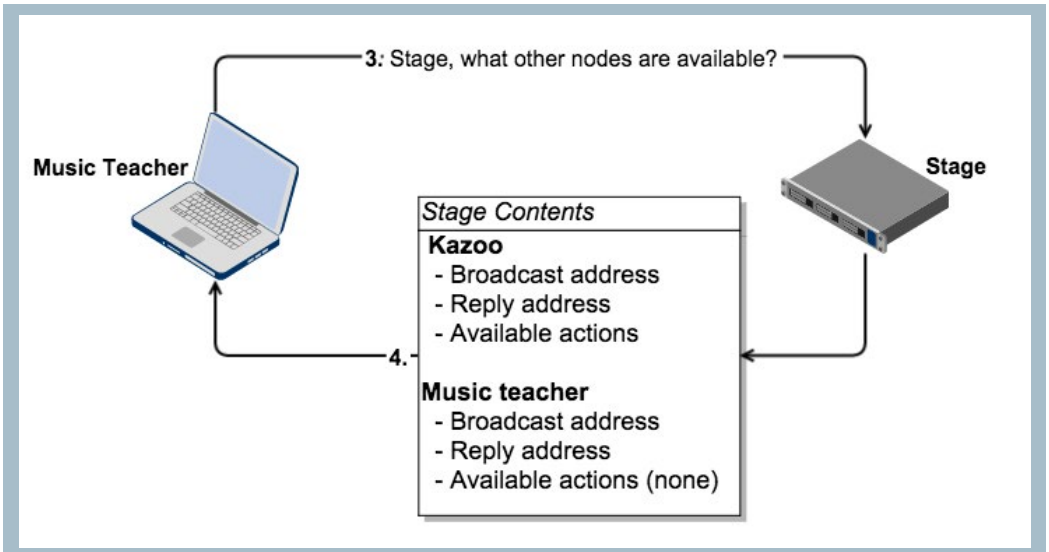

Figure 38: A node querying the stage for information about the performance.

In order for the music teacher node to connect to the Kazoo player node, the music teacher needs to know what the kazoo player's network addresses are. It queries the stage (Figure 38) for a list of all available nodes and receives the addresses of all nodes within the performance. It also downloads the list of actions the kazoo can broadcast or respond to and can use this list in the future to craft control actions.

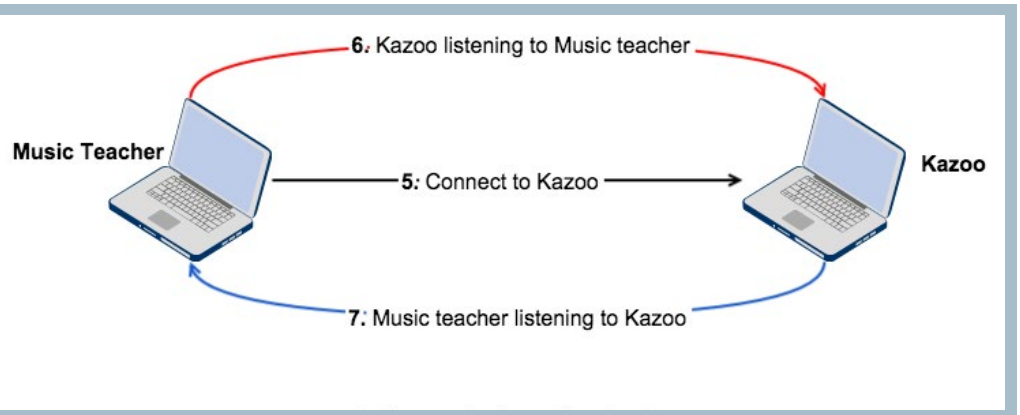

Figure 39: One node initiating a direct connection to another.

Once the music teacher has the addresses required to connect to the Kazoo, it initiates a connection to the kazoo's reply socket (Figure 39) and requests that the Kazoo's control action socket subscribes to the music teacher's broadcast action socket. Now the Kazoo will listen to all control actions sent from the music teacher's broadcast action socket and will act upon any action sent that it understands. The Music Teacher's control action socket will also subscribe to the Kazoo's broadcast action socket to in order complete the two-way relationship between the two nodes. At this point, both nodes are now fully connected and can now communicate with each other.

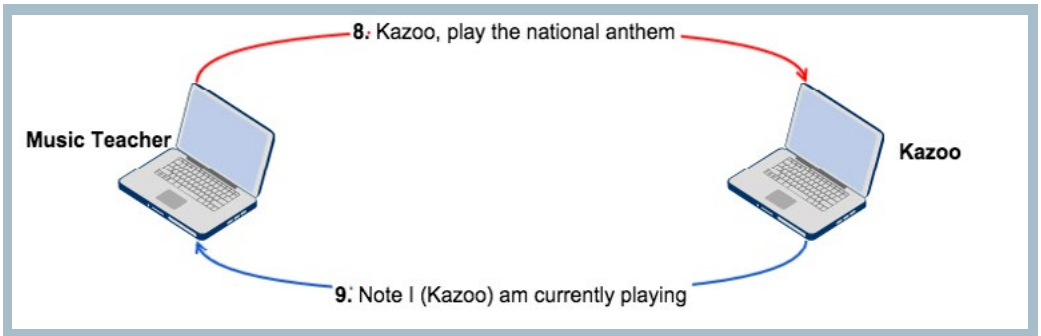

Figure 40: A node sending a control action to a listening node, followed by broadcast actions being generated from the node and sent to all listening nodes.

The Music Teacher can now create a control action from the Kazoo's list of actions downloaded from the stage. When the music teacher runs the control action locally, the node broadcasts out the control action over its broadcast socket (Figure 40). Since the Kazoo is now listening to the music teacher, it receives the control action and compares it against it's own list of control actions. When it finds a match, it runs whatever function is associated with the control action, in this case the Kazoo starts playing the national anthem.

Now that the Kazoo has started to play notes, it has the opportunity to run its broadcast action "currently played note". As the Kazoo moves over each new note, it will triggers the broadcast action which is pushed out over the Kazoo's broadcast action socket. This is picked up by the Music Teacher on its control action socket, where it can decide how to respond to the action.

5.3 Pensato-Showtime network

Showtime was designed foremost to be a replacement for OSC in connecting Ableton Live to Pensato's VR environment. Its design was driven by the need for a way to interact with Ableton Live's underlying application programming interface (API), rather than by binding specific MIDI or OSC messages to individual parameters. By switching to Showtime, which could directly trigger Live's API functions, the need to create manual mappings between both Live and Pensato's parameters was removed.

In order to map the correct VR controls to the correct parameters, Pensato will request Live to provide it with an XML formatted text file describing the layout of tracks, devices and parameters that make up the currently open song within Live. From this layout file, Pensato will construct proxy VR controls to represent tracks, clips and parameters (see "4.2 Basic VR Controls", 37). Each VR control stores enough identifying information about the object so that calls to Live's API know how to map a particular request to the correct object. For example, a slider in Pensato contains a track, device and parameter ID which is passed through Showtime to Live's API in order to manipulate the correct device parameter. Each proxy VR control can also catch broadcast actions from Live that indicate how the original linked object changes. This allows the proxy VR control to visually represent the state of the linked Live object whenever it is updated. For example, a slider would listen for broadcast actions from Live that communicate that a value has changed. The slider would then visually update to reflect the current value of the linked object.

The final Showtime network that was constructed to help serve Pensato's performances can be seen in Figure 41. Of note is that Ableton Live does not directly possess its own Showtime node. In order to bypass some technical limitations within Live's Python API, a program was written to bridge a Showtime node to Ableton Live using a Python specific messaging system called Pyro (Jong, 2012). Some other nodes of note are the glove nodes. These were used to send glove position and gesture data from a master instance of Pensato that ran the VR environment to another instance that displayed the VR environment in a projectable format. By synchronising the glove data between the two instances, the performer's actions became visible to both the performer and the audience.

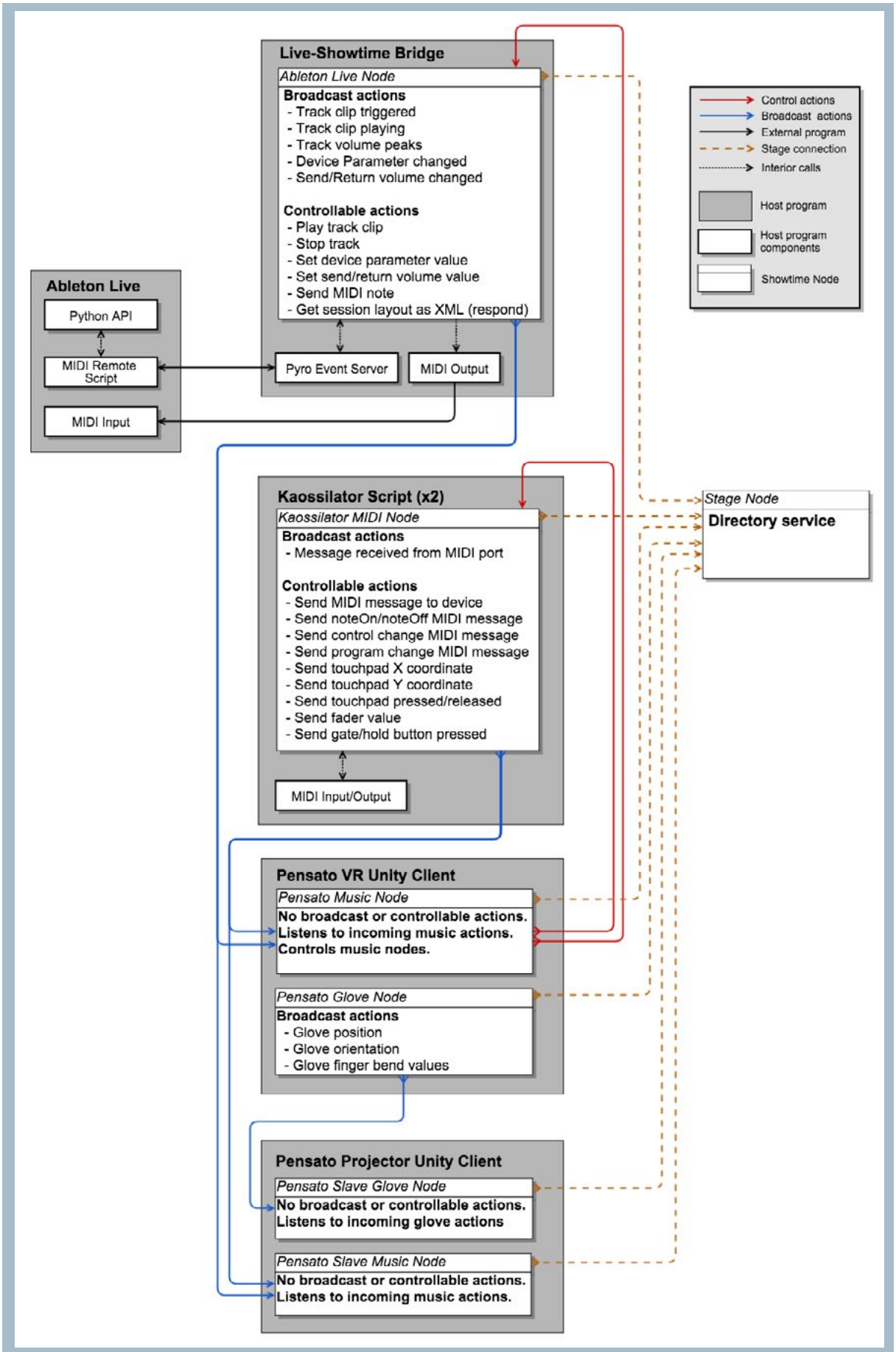

Figure 41: Pensato's Showtime network layout

5.3.1 Conclusion

The initial goal when creating the Showtime framework was to create a musical messaging system that was fast, flexible and descriptive. Showtime removed the requirement for having to create manual connections between individual Ableton Live parameters and their Pensato VR control equivalents, speeding up the creative process. The framework also operated at a consistently high speed, due to its foundation being built upon on the high-speed ZeroMQ library and the ability to take advantage of high-speed network hardware.

The framework provided a flexible environment to work within, allowing for the creation of multiple specialized nodes across a variety of programming languages. This helped to spread the high computing requirements of Pensato and Ableton Live between multiple machines. These machines were configured to specialise in the handling of audio, VR visual rendering or audience visual rendering tasks.

Showtime has provided me with a way to quickly prototype new types of connections between Ableton Live that helped to spur the design process of new features within Pensato. It is also my hope that the Showtime framework can also be used by other artists in the future as a way for sharing multiple musical devices within one shared digital environment.

6. Conclusion

This thesis has explored a variety of methods for combining virtual reality technologies with an electronic music system, in order to create a platform for arranging and performing live music. By analysing past and present musicians who use either off-the-shelf hardware or unique control setups for performance, the relationship between performer and audience was established as a crucial element. VR display technology was identified as a possible solution for closing the control abstraction gap between full body musical controllers and the visuals reinforcing the actions of a performer.

A number of design experiments were undertaken in order to research methods of musical control using the movement of the human body as a musical interface. The first experiment was Sonoromancer, an interface that used the silhouette and position of a performer's body to create and perturb simulated musical fluid. This final iteration of this experiment created mesmerizing visuals, but was difficult to precisely control in order to create specific musical notes or patterns.

The second experiment was a parameterized visual controller using a VR user interface named Tesla Rift. The experiment focused on exploring possible methods for interacting with controls in a 3D space using hand tracking technology, primarily using floating groups of parameters that could be moved around the environment using physics. Whilst fun to interact with, the performer would frequently become confused due to the non-permanent placement options for some of the UI controls and UI text labels being arranged in 3D orientations without a common point of reference.

The final compositional output of this thesis was Pensato, a VR interface for controlling the music software Ableton Live using gesture recognition gloves. Building upon the previous two experiments, Pensato delivers an interface that encourages a performer to interact with its VR controls as if they were real physical objects. By using gestures that would map to similar actions in the real world, a performer is able to quickly learn how to interact with the VR interface and can apply their tacit knowledge for interfacing with physical objects to the performance space. Pensato also allows the performer to create custom mappings between multiple musical parameters using a library of VR widgets, which can be built and saved into layouts that can be switched between during a live performance. The final software embraces an auto-organization philosophy for sorting VR controls that allows for unique layouts to be created by the performer whilst reducing visual clutter and confusion by sorting and displaying controls as needed in groups.

The Showtime library was also created during the development of Pensato, which allows for the connection of multiple pieces of media software, code, or hardware together in a shared network performance space. This let Pensato control a number of external MIDI devices, talk to and listen to the innards of the Ableton Live API, and distribute data from the glove input system to an external Pensato client allowing for a second computer to visualize the performer's VR space using an external projected surface.

6.1 Future directions

VR technology in recent years has improved both in quality and affordability, which is a trend that hopefully will continue for the foreseeable future. This will allow for easier to use and more accurate input

hardware to be adopted for performances using VR spaces. The quality of the input hardware used in Pensato for tracking the position and gesture of the performer's hand could be drastically improved from its current level in order to provide a completely immersive input solution for manipulating VR controls. The possibility of using inertial motion based tracking solutions to accurately track the entire upper body of the performer down to the individual finger joint level would definitely increase the overall accuracy and immersiveness of the interface to allow for more efficient and fluent performances.

Whilst outside of the scope of this thesis, tactile feedback is a very important element missing from Pensato that should be investigated in the future. The lack of haptic feedback between VR controls and the performer's limbs means that many rhythmic controls possess a degree of uncertainty for when they will be triggered from the performer. By reinforcing the visual stimulus of hitting a virtual control with an associated haptic response, it is hoped that this will help synchronize the performer's action with the intended outcome.

Another natural step beyond using VR spaces for solo performance, is to introduce collaborative components. By combining multiple performers with their own individual input hardware into a shared VR space, performers could share instruments, create custom controls and parameter mappings or even record new sequences and patterns. These personal controls could then be passed around like physical objects, possibly by passing or throwing controls to another performer. It is hoped that Showtime will provide functionality for this sort of collaboration in the future and will further improve how performers dynamically connect into and control aspects of the performance space. As VR headsets also become cheaper, more mobile and ubiquitous, it is hoped that members of the audience will be able to join and observe the same VR space that the performers exist within, even if they are located in a room located halfway across the world.

Pensato harnesses the growing usefulness of virtual reality technology as a way of extending a performer's ability for creating musical performances. It is my hope that this approach can be further explored in the future to not only improve the breadth of tools available for live music performers, but to help further the evolution of appropriate interface and interaction designs for virtual reality.

7. Bibliography

Ableton. (2014). What is Live? Retrieved November 13, 2014, from https://www.ableton.com/en/live/

Berthaut, F., Desainte-Catherine, M., & Hachet, M. (2010). DRILE : An Immersive Environment for Hierarchical Live-Looping. In Proceedings of the International Conference on New Interfaces for Musical Expression (pp. 192–197). Retrieved from http://www.nime.org/proceedings/2010/nime2010_192.pdf

Blaszczyk, R. L. (2003). Review of Analog Days: The Invention and Impact of the Moog Synthesizer by Trevor Pinch; Frank Trocco. Journal of Design History, 16(4), 357–359.

Bowman, D. A., McMahan, R. P., & Ragan, E. D. (2012). Questioning Naturalism in 3D User Interfaces. Commun. ACM, 55(9), 78–88. http://doi.org/10.1145/2330667.2330687

Broomhead, D. S., & Lowe, D. (1988). Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Networks (pp. 5–6).

Buhmann, M. D. (2003). Radial basis functions: theory and implementations (Vol. 5). Cambridge university press Cambridge.

Castle, A. (2011, September 12). Razer Hydra Review. Retrieved June 9, 2014, from http://www.maximumpc.com/article/reviews/razer_hydra_review

Daintith, J., & Wright, E. (2014). Remote Procedure Call. A Dictionary of Computing (6th Edition). Oxford University Press. Retrieved from http://www.oxfordreference.com/view/10.1093/acref/9780199234004.001.0001/acref-9780199234004-e-4448?rskey=KuFMVN&result=4778

Foreman, D. (2013, August 2) Beardyman: The polyphonic me. (2013). [Video file]. Retrieved from http://youtu.be/dIh8KBOOkYU

Foreman, D. (2014, August 24). Beardyman - Leeds Festival 2014 (end segment). Retrieved October 6, 2014, from http://soundcloud.com/beardyman/leeds-festival-2014-end-segment

Freed, A., & Schmeder, A. (2009). Features and Future of Open Sound Control version 1.1 for NIME. Retrieved from http://cnmat.berkeley.edu/node/7002

Galeyev, B. M. (1991). L. S. Termen: Faustus of the Twentieth Century. Leonardo, 24(5), 573. http://doi.org/10.2307/1575663

Heap I. (2013). Heap Performance with Musical Gloves Demo: Full Wired Talk 2012. [Video file]. Retrieved from http://youtu.be/6btFObRRD9k

Heap, I. (2014). Me The Machine (Official Video). [Video file]. Retrieved from http://youtu.be/N0lCL2hpRPM

Introducing the Midi Fighter Spectra. (2013). Retrieved from https://www.youtube.com/watch?v=BF9Pp2qFBzI&feature=youtube_gdata_player

Jong, I. de. (2012). Pyro: Python Remote Objects (Version 3.16). Retrieved from http://pythonhosted.org//Pyro/

Kneppers, M. (2013, November 18). Livegrabber. Retrieved October 14, 2014, from http://showsync.info/index.php/tools/livegrabber/

Kofoed, J., & Trott, M. von. (2012). The V Motion Project -- Can't Help Myself [Official Music Video]‼ . Assembly. Retrieved from http://youtu.be/YERtJ-5wlhM

Kurenniemi, E. (1971). Erkki Kurenniemi - DIMI Ballet [DVD]. Retrieved from https://www.youtube.com/watch?v=d-yHULQ2V5c

Kushner, D. (2014). Virtual reality's moment. IEEE Spectrum, 51(1), 34–37. http://doi.org/10.1109/MSPEC.2014.6701429

Liine. (2014, January 21). Lemur. Retrieved October 13, 2014, from http://liine.net/en/products/lemur/

Louis Buignet, & Adrien Tisseraud. (2013). Taylorythm - You Are Amazing (Remix) [Music Video]. Retrieved from http://youtu.be/YHGj9JF73-w

Loy, G. (1985). Musicians Make a Standard: The MIDI Phenomenon. Computer Music Journal, 9(4), 8–26. http://doi.org/10.2307/3679619

Martin, S. M. (1995). Theremin: An Electronic Odyssey. Documentary, Biography, History.

Mattis, O., & Moog, R. (1992, February 1). Pulling music out of thin air: An interview with Leon Theremin. Keyboard Magazine. Retrieved from http://www.moogmusic.com/legacy/pulling-music-out-thin-air-interview-leon-theremin

Mulder, A. G. E. (1998). Design of Virtual Three-dimensional Instruments for Sound Control.

Nortje, J. (2010). Persistence of Illusion: Using Spatial Illusion as a Visual Performance Mechanism. Retrieved from http://researcharchive.vuw.ac.nz/handle/10063/1257

Novation. (2013). Launchpad S [Product page]. Retrieved October 21, 2014, from http://global.novationmusic.com/midi-controllers-digital-dj/launchpad-s

Oculus. (2012, January 9). Oculus Rift: Step Into the Game. Retrieved December 5, 2014, from https://www.kickstarter.com/projects/1523379957/oculus-rift-step-into-the-game

O'Modhrain, S. (2000). Playing by Feel: Incorporating Haptic Feedback Into Computer-based Musical Instruments. Department of Music, Stanford University.

Ostertag, B. (2002). Human Bodies, Computer Music. Leonardo Music Journal, 12, 11–14.

Perle, G. (1995). The Right Notes: Twenty-three Selected Essays by George Perle on Twentieth-century Music. Pendragon Press.

Simon, C. (2014, December 1). The "Telharmonium" or "Dynamophone" Thaddeus Cahill, USA 1897. Retrieved from http://120years.net/wordpress/the-telharmonium-thaddeus-cahill-usa-1897/

Vallis, O. S. (2013). Contemporary Approaches to Live Computer Music: The Evolution of the Performer Composer. Retrieved from http://researcharchive.vuw.ac.nz/handle/10063/2831

Vik, C. (2011). Live Looping with Ableton and Xbox Kinect. Retrieved from http://youtu.be/xP-coM7BIDZ4

Zerodebug. (2013, November 23). touchAble 2. Retrieved October 13, 2014, from http://www.touch-able.com/touchable-2/

8. Figures