Indeterminate Sample Sequencing in Virtual Reality

Chase Mitchusson
Louisiana State University and Agricultural and Mechanical College

Follow this and additional works at: https://repository.lsu.edu/gradschool_dissertations

Part of the Composition Commons, Game Design Commons, Interactive Arts Commons, Interdisciplinary Arts and Media Commons, and the Other Music Commons

Recommended Citation
https://repository.lsu.edu/gradschool_dissertations/5381

This Dissertation is brought to you for free and open access by the Graduate School at LSU Scholarly Repository. It has been accepted for inclusion in LSU Doctoral Dissertations by an authorized graduate school editor of LSU Scholarly Repository. For more information, please contact gradetd@lsu.edu.
INDETERMINATE SAMPLE SEQUENCING
IN VIRTUAL REALITY

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The School of Music

by

Chase Mitchusson
B.A., University of Memphis, 2011
M.M., University of Memphis, 2015
December 2020
ACKNOWLEDGEMENTS

I would like to thank my family, Lynn, Terri, Nanny, Maw, Blake, and my lovely partner Bethany for showing unending love and support through my journey. I want to thank my friends Tony, Monica, Eric, Andrew, Tate, Jess, Landon, Matt, Thomas, and Kaci for being such great people who I cherish for being wonderful friends, collaborators, and travel companions over the years. I would also like to thank the LSU faculty, Dr. Jesse Allison, Dr. Edgar Berdahl, Dr. Stephen Beck, Dr. Hye Yeon Nam, Derick Ostrenko, Ken Wesley, Marc Aubanel, and Dr. Joyoni Dey for their guidance, support, and for helping me acquire all I needed to complete my research during my time at LSU. This has been the most challenging and gratifying time of my life and I am so fortunate to know you all.
# CONTENTS

**ACKNOWLEDGEMENTS** ................................................................. iii

**LIST OF FIGURES** ......................................................................... vi

**ABSTRACT** ................................................................................ vii

**INTRODUCTION** ........................................................................... 1

**CHAPTER 1. DIGITAL INSTRUMENTS, MUSIC GAMES** ................. 6
  1.1. Digital Instruments ................................................................. 6
  1.2. Instrument Mapping ............................................................... 9
  1.3. Interactive Music Games ....................................................... 11
  1.4. Interactive Music Performance .............................................. 14

**CHAPTER 2. INDETERMINACY** ................................................... 18
  2.1. A Brief History of Aleatoric Works ....................................... 18
  2.2. Indeterminacy in the VR Sequencer Interface ....................... 20

**CHAPTER 3. DIGITAL MUSIC, INSTRUMENTS IN VIRTUAL REALITY** 24
  3.1. Sequencers ............................................................................. 24
  3.2. A VR Sequencer ..................................................................... 26
  3.3. A Brief Overview of Experimental VR Works ..................... 28

**CHAPTER 4. DEVELOPMENT, INTERFACE, AND COMPONENTS** .... 39
  4.1. Three-Dimensional Assets ................................................... 39
  4.2. Musical Interaction ............................................................... 44
  4.3. Programming Interface ...................................................... 45

**CHAPTER 5. COMPOSING & PERFORMING** ................................. 50

**CHAPTER 6. A PIECE FOR VR SEQUENCER, PRIMITIVES** ........... 54
  6.1. *Touch* ................................................................................. 54
  6.2. *Hold* .................................................................................... 56
  6.3. *Release* .............................................................................. 59

**CONCLUSION** .............................................................................. 62

**APPENDIX A. EXTERNAL LINKS** ................................................. 64

**APPENDIX B. FIGURE PERMISSIONS** .......................................... 65
  B.1. Figure 3.4 ............................................................................. 65
  B.2. Figure 3.5 ............................................................................. 66

**BIBLIOGRAPHY** ......................................................................... 67
# LIST OF FIGURES

1.1 HTC Vive head-mounted display. .................................................. 2

1.2 A view of front and rear of two HTC Vive controllers. ..................... 2

1.1 Piano key action mechanism ..................................................... 8

2.1 Monochromatic red dice in VR Sequencer being thrown in the air during a terrain shift. .................................................. 22

3.1 Ableton Live Session View. ..................................................... 25

3.2 Ableton Push MIDI controller. ................................................... 26

3.3 Green player character inside a white spherical playhead triggering monochromatic red dice in the VR Sequencer. ................................. 27

3.4 Ensemble performance of Rob Hamilton’s Carillon. ........................................... 29

3.5 Zachary Berkowitz’s navigable virtual space generated for Zebra. .......... 34

4.1 A set of dice used to play Dungeons & Dragons. ............................... 40

4.2 Third-person view of several red dice and several transparent white time-and-space spheres. ............................................. 41

4.3 First-person view of a time-sphere having its parameters edited. .......... 44

4.4 A terrain shift sending dice flying into the air. .................................. 45

4.5 Unity Audio Mixer and Snapshot settings for the VR Sequencer. ............ 48

5.1 A photo of VR Sequencer viewed as a graphic notation. This photo or a video clip of this section of music can be used as a graphical score. ............. 53

6.1 A large loop interacting with three smaller loops that make up the foundational material for Touch. ............................................. 55

6.2 Three sets of multiple time-spheres overlapping to trigger a single die. .... 57

6.3 A series of dice being triggered by a single time-sphere. ....................... 60
ABSTRACT

The VR Sequencer is a tool for composing and performing music which takes advantage of spatial configuration of 3-D objects to generate looping sequences. This project combines dice-rolling with omnidirectional playheads and variable terrain to make a sequencer with indeterminate features in Virtual Reality (VR). A comparison of mapping systems for music games and digital instruments demonstrates a middle ground for interactive music making which the VR Sequencer applies. An inspection of current VR works exhibits a disconnect between VR gamers and experimental musicians. In an attempt to give the gaming community access to experimental music making, the VR hardware used in this project is consumer-grade and requires no external peripherals for playing. The VR Sequencer exclusively uses Unity’s audio engine for playback, effects, and mixing to demonstrate its capabilities within the experimental music field.
INTRODUCTION

Many music video games include goal-oriented musical mechanics, but tend not to embrace free play, which leads to fixed musical experiences. Trends in experimental music in Virtual Reality (VR) offer expressive, free play musical instruments, but include combinations of hardware and software that make it difficult to share with owners of VR systems. By using the same hardware and software for which traditional game studios develop music video games, these two fields are bridged to offer non-goal-oriented musical experiences to owners of VR systems.

Game developers and experimental music developers have different goals. Their works, music video game controller mappings, and the roles sound plays in music video games demonstrate the dissonance between their approaches. Additionally, examining experimental VR music reveals a disconnect between experimental music developers and the VR gaming community. The use of traditional video game development methods, the exclusive use of consumer VR systems, the addition of indeterminacy to music mapping systems, and an emphasis on free play, all within an experimental VR sequencer interface, harmonize experimental music development and VR gaming.

The VR Sequencer is a 3-D VR interface that uses spherical playheads (time-spheres) to trigger audio clips associated with six varieties of virtual dice. The stage on which the performer stands in the VR Sequencer is a variable terrain that can jumble the dice and manipulate audio effects. The 3-D nature of VR means rhythmic and melodic content are developed through spatial orientation of virtual musical objects, and is made more elaborate through the use of multiple playheads. Each playhead is capable of triggering different audio clips at different rates and in different areas of the stage.

To preserve access to VR system owners, the VR Sequencer was developed for consumer hardware using Unity, Microsoft Visual Studio, and the HTC Vive Software Development Kit. The HTC Vive standard hardware includes a VR headset (shown in Figure 1.1), two controllers (shown in Figure 1.2), an HDMI adapter to send video to the headset, and two
Figure I.1. HTC Vive head-mounted display.

Figure I.2. A view of front and rear of two HTC Vive controllers.
motion sensors to track the user in the physical world and map them to the virtual environment. The sensors can make sure the video being sent to the headset matches the position of the user’s body and displays the controllers to match their physical positions as well. The purpose of using consumer-grade VR kits is to enable gamers to create art and express themselves musically through a familiar medium.

Three core mechanics, 360-degree range of motion, blurring lines of composer and performer, and indeterminacy, frame the user’s interaction with the VR Sequencer. Understanding each of these components is necessary to understand the VR Sequencer as a whole.

Construction of musical sequences in a 3-D environment with a 360-degree range of motion is a crucial aspect of the VR Sequencer. Full immersion in virtual reality is achieved by replacing the physical world. This replacement involves replication of freedom of motion found in the physical world, which requires a 3-D environment. This philosophy is shared with the creators of Wedge, Alec Moore, Michael Howell, Addison Stiles, Nicolas Herrera, and Ryan McMahan. Wedge allows users to build custom environments for playing back music in VR with access to 360 degrees around the user. This interaction resembles the placement of dice and time-spheres in the VR Sequencer for creating musical sequence structures. The playback methods differ greatly, however. Wedge users rely on hand gestures to playback their constructions, while time-spheres are used for audio clip playback in the VR Sequencer. The gestural controls of Wedge offer the user a conductor-like role to performing their music, which contrasts with the playhead model of the VR Sequencer.

The line between a user’s role as a performer and a composer is blurred with the VR Sequencer. The VR Sequencer does not aim for balance of the two roles, but acknowledges the ambiguity. A composer in the VR Sequencer may have organized a plan to compose a specific structure, but cannot guarantee the results. A VR Sequencer performer may

---


2 Ibid.
embrace the indeterminacy and improvise all interactions. In both cases, the music will be unique to that experience, and will emphasize the musical influence of the tool. Norbert Herber’s Perturb finds itself in a similar situation. Pertub uses a generative musical system that shares music creation with user input, though in this case, it is the system that blurs between composing and performing, not the user. Herber seeks to maintain equal distribution of his composition-instrument’s role in “playing” and “being played.”

Physics-based interaction and indeterminacy are core elements of the VR Sequencer. Toshio Iwai’s Electroplankton shares these behaviors. Electroplankton’s “Hanenbow” level provides a clear demonstration of physics-based interaction and indeterminacy.

The layout of “Hanenbow” consists of a plant emerging from water with leaves that can be repositioned by players and a single leaf that launches tadpole-like creatures, known as Hanenbow, across the stage to bounce along the leaves. Players use a stylus to grab and rotate leaves and use the direction-pad to launch a Hanenbow from the water across the stage. As the Hanenbow contacts the leaves, it bounces off them and a musical note plays. Each time a Hanenbow bounces off a leaf, it may hit another leaf or fall back on the first leaf. Electroplankton uses the physics of the Hanenbow’s trajectory to determine its fall back into the water. The position of the leaves and the trajectory of the Hanenbow determine a musical sequence unknown to the player. The player has control over where to launch the Hanenbow and the arrangement of the leaves, but the note output is determined by the leaf itself. The system is indeterminate because one Hanenbow can hit the same leaf twice but trigger two different notes. Melodic output is not decided by the player, the player only decides the arrangement of the leaves and the trajectory of the Hanenbow.

Arrangement of musical elements in the VR Sequencer determines a musical sequence, but, like in Electroplankton, the content of the sequence is unknown to the user until playback.

---


4 Toshio Iwai, Electroplankton Instruction Booklet (P.O. Box 957, Redmond, WA 98073: Nintendo, 2005).

5 Ibid.
begins. Dice can be thrown around the VR Sequencer stage and react to terrain movements which ultimately change the melodic and rhythmic content, much like the trajectory of Hanenbow and the position of the leaves. The biggest difference between the VR Sequencer and *Electroplankton* is the passage of time. *Electroplankton* uses speed and trajectory of Hanenbow to mark the passage of time, while the VR Sequencer uses spatial positions, size, and speed of looping playheads to keep time. Time in the VR Sequencer is relative to time-sphere parameters and the position of dice on the stage, not an object’s trajectory.

Merging those behaviors and mechanics found in prior works creates the VR Sequencer. It offers full immersion by replacing the physical world, combines physics-based interactions with indeterminacy, and allows for users to decide their role in writing or performing music.
CHAPTER 1
DIGITAL INSTRUMENTS, MUSIC GAMES

1.1. Digital Instruments

This project examines musical interfaces for digital instruments to discover how interface mapping systems characterize musical instruments in order to help the experimental music community understand how typical musical mapping systems can be applied to non-traditional musical interfaces like video games. The purpose of this research is to expand a musical interface with a simple mapping system by adding variance through indeterminacy to create a composition and performance tool for players of VR video games.

Music video games offer a fun and interactive way to play music in a non-traditional musical setting. However, they give the player limited or no options to improvise and have no real goals in the game because they are designed like toys instead of instruments. In non-goal oriented music video games, wrong notes cannot be played and performers cannot play off-beat.¹ Music video games need to be mapped in a more complex manner, because user actions are strongly tied to emotions in mapping systems and video games. Digital instruments can elicit emotional responses from performers based on complexity of the mapping system.² Music synchronized to game events can control players’ emotions and expectations.³ A complex mapping system in a music video game will produce a complex instrument, because increasing the boundaries of musical limitations allows for more experimentation with


an interface. Digital musical instruments can be easy to learn, yet challenging to master. An interface with complex mapping can have a high degree of control intimacy.  

There is room, however, for adapting a simplistic mapping system to produce a more complex instrument by way of an indeterminate input system. Indeterminacy encourages active listening in performers and affects intimacy of control, because it places the performers in a reactive role. Virtuosity in such a system comes from knowing how to handle a music performance when the outcomes do or do not favor desired musical results.

Interfaces determine how a user interacts with an instrument. A simple example of a musical interface is a piano keyboard. To interact with the piano, a user will depress a key. The interaction alone does not determine the musical output. It is the piano’s mapping system that determines the musical output by separating physical input (depressing a key) to musical output (a note being played). In the case of the piano, the mapping system is a mechanism that causes a hammer to strike the piano strings when a key is depressed as seen in Figure 1.1. When one key is pressed, one note is played. When five keys are pressed, five notes are played. This type of mapping is called one-to-one. If one key depression produced five notes, then the mapping would be one-to-many. One-to-many mappings are commonly found in digital musical instruments to produce complex musical output with simple physical interactions.

A digital musical instrument is an electronic input device that communicates digitally to a sound generator. This is something that produces digital data upon user interaction, commonly with a musical instrument digital interface (MIDI) controller. When an interface produces digital data, the data must be mapped to be heard. One can map digital data in various ways from the simplest one-to-one mapping to mimic a piano keyboard to a complex

---


one-to-many mapping that controls note output, frequency filtering, delay time, and reverberation length. The mapping possibilities are endless, and they define the characteristics of each digital musical instrument.

Music video games are audio-visual digital musical instruments that are sometimes goal-oriented. This means the data from a game controller is mapped to produce audio when certain buttons are pressed and the user additionally gets visual feedback for their interaction. The goals set in music video games like *PaRappa the Rapper*, *Guitar Hero*, or *Beat Saber* typically require players to correctly input a sequence of buttons or actions in the right rhythm to progress through the game. Some video games, like *Electroplankton* and *MTV Music Generator*, lack goals entirely and leave the player to explore music creation at their own discretion.

Given the ability to map digital information across endless numbers of parameters to build a digital musical instrument, there should be some measure of user control for each instrument. Wessel and Wright (2002) define musical control as, “ease of use, potential for development of skill, reactive behavior, and coherence of the cognitive model.” This means an instrument should be easy to pick up and play, but require skill to master, and behave

---

7Pichlmair and Kayali, “Levels of Sound: On the Principles of Interactivity in Music Video Games.”
8Wessel and Wright, “Problems and Prospects for Intimate Musical Control of Computers.”
the way a user expects it to. These characteristics are determined by the mapping system of the interface.

An indeterminate system can behave in expected ways, despite supplying results through random generation. A user can expect a certain level of indeterminacy, but also expect to understand how their actions produce an outcome. VR Sequencer uses dice to determine audio clips, and users should expect to understand behavior of dice, how dice roll and how to roll them. What they will not expect is the musical output. The visual feedback from the playhead leads a user to expect when a clip will play, but not what the clip will play. A user can expect the terrain to shift after pressing a button, but cannot predict the outcome. There is skill required to master the interactions within the VR Sequencer and one can expect certain behaviors while still being unable to predict the outcomes. VR Sequencer is easy to play, requires minimal skill to interact with the environment, has expected physics, but has unexpected musical outcomes.

1.2. Instrument Mapping

Mapping systems will determine what comes out when one plays an instrument. Sometimes there is a need for a simple system, but when a complex system is used the musical results can challenge the performer to be more thoughtful about their actions. \(^9\) Hunt, Wanderley, and Paradis (2003) discuss the effect mapping has on electronic instruments, and that it can elicit different emotional responses from the performers based on complexity. Two studies demonstrate how a simple vs. complex mapping system produce various responses from performers, including that certain complexities resemble playing real instruments. The authors write on designing, testing, and evaluating strategies for mapping electronic instruments and ask how one can measure the performance of an interface. The more real the instrument

feels, the more likely the cognitive-sensorimotor skills used to perform will be challenged.\textsuperscript{10} This shows that skill does not have to be different in the context of digital music systems. The mapping of digital musical instruments determines the skill required to play a digital musical instrument. A reactive mapping system does not require judicious action, but encourages thoughtful reaction. Spawning or rolling dice requires low skill, but the complexity of musical outcomes rewards low skill, encouraging novice musicians to experiment.

An interface with a mapping system that allows anyone to pick up and learn easily, while being skill intensive enough to require practice in order to master will also feel like a “real” instrument. Wessel and Wright (2002) support this claim by finding mapping metaphors that will help create instruments that performers will not outgrow. They mean an instrument that can be explored near infinitely to develop a mastery, unlike many simple mappings that will lead to boredom after discovering all the instrument can do too quickly. The metaphors include flying through space, drag and drop, scrubbing, dipping, and catch and throw. Each metaphor is attached to a physical gesture that Wessel and Wright think is important to digital musical instrument mapping systems. Physical gestures are familiar to users and allow users to understand the mapping better through physical exploration.\textsuperscript{11} The gestural control suggested by Wessel and Wright uses cognitive-sensorimotor skills in a way that is natural and empowering for users. By connecting user control to physicality, an instrument becomes intimate and may improve the speed at which an instrument is learned. VR inherently connects user control to physicality, because it maps physical locations and movements of users to a virtual environment. VR Sequencer embraces the physical nature of VR through its interface. The common gestures used in VR Sequencer are grabbing, throwing, holding, releasing, and rolling. VR Sequencer is not infinite, but the goal is to find a line between solved and infinite using a simplistic mapping system with the aid of indeterminacy. Using one die and one playhead, a VR Sequencer user can find all the possible outcomes without


\textsuperscript{11}Wessel and Wright, “Problems and Prospects for Intimate Musical Control of Computers.”
changing terrain, but adding another die, another playhead, and changing the terrain and audio effects become impossible to solve through trial and error due to the large number of possible outcomes combined with random number generation.

A more specific look into how mapping systems challenge motor and cognitive skills comes from Kabbash, Buxton, and Sellen (1994). They perform a quantitative study on four different bi-manual techniques for a color-based connect the dots task. Their study was designed to understand the motor and cognitive load a two-handed interface required. The two-handed tasks they used are good examples for musical instruments and game controllers, as most require two hands. Measurements include time to complete task, time of each hand used, time between actions, and errors.\(^\text{12}\) This study has measurable results that can be applied to game controller use for those looking to develop new music interactions. VR typically uses two game controllers, but since VR Sequencer’s interface is simplistic, it only needs one controller. Users can use either left or right controller to do the same tasks, such as picking up objects, clicking buttons, teleporting, and changing the terrain. A single-handed approach like this parallels using a mouse, which is a common interface tool for traditional DAWs.

### 1.3. Interactive Music Games

Pichlmair and Kayali (2007) categorize music video games into seven types based on player influence and interaction with the game music and how restricted players are in affecting the output of sound. They also determine two important distinctions in the music game genre. One type of music game, the rhythm game, is scored based on correctly timed input of rhythms. Examples of rhythm games are *Guitar Hero*, *Dance Dance Revolution*, and *PaRappa the Rapper*. The other type, the electronic instrument game, lacks goals and

plays more like a toy than a game. Electronic instrument games include *Electroplankton*, *MTV Music Generator*, and *SimTunes*. Both game types offer an illusion which simulates playing an instrument, but neither game type acts the same as a musical instrument. Pichlmair and Kayali identify the weaknesses of music games when compared to musical instruments by demonstrating the rigidness of rhythm games, and lack of risks in electronic instrument games. Rhythm games require players to follow a strict set of rules through linear gameplay that gets progressively more difficult and complex. Electronic instrument games don’t have rigid sets of rules or goals, which lets the player determine their own goals. While determining one’s own goals is rather instrument like, the restrictive musical output makes it almost impossible to ever play a wrong note. These differ from traditional musical instruments by not allowing players to improvise, only to input commands correctly to play predetermined sounds. The VR Sequencer blends between these two game types by not setting rigid rules or goals, but still allowing users to play ”wrong” notes. In this case, a wrong note is the addition of musical material that does not please the performer. Players determine their own goals, improvise, and input commands, but the results of the commands are indeterminate.

Within Pichlmair’s and Kayali’s definition of music game types, various types of interactions are present. Liebe (2013) categorizes various games based on their musical content and how the player interacts with it. This includes lists of games that feature audio prominently, the characteristics of game audio, and how audio is stored, produced, and written for games. Liebe also discusses what it means for games to be interactive and how rules may affect interactivity, and notes the differences in interfaces based on the player’s physical

---


14 Ibid.

15 Ibid.

16 Ibid.

17 Ibid.
involvement. All of Liebe’s data is put into charts based on Pichlmair’s and Kayali’s seven categories of games as well as the games’ interface types and music types.

The interactive portion of games is also examined by Karen Collins (2013). *Playing with Sound* covers a brief history of sound in games, and further explains the role sound plays in games. Topics include the difference between interacting with and listening to sound, the theory of audio-visual integration and synchrony, how sound in games engages the body and promotes role-playing, and various examples of using games as a medium to produce sound. Sound production in games goes beyond controller input, and Collins provides examples of games that are meant for composition, games that stream live audio into virtual worlds, and using game hardware to create sound in the real world.

The VR Sequencer, by nature of VR, requires physical involvement and audio-visual integration and synchrony. It engages the body via rolling dice, engages listening via indeterminacy, and integrates an audio-visual component via time-spheres and dice. As well, it is a framework to produce music, compose, and perform live, which uses game hardware to create sound in the real world. According to Pichlmair’s and Kayali’s game types, VR Sequencer falls into several categories. Sound agents, which use objects to produce audio; active score music, which involves real-time composing; play as performance, for the physical aspects required to make music; and free-form play, for the lack of goal in VR Sequencer which encourages instrument-like playing.

Karen Collins discusses many other types of interactions including psychological, multi-modal, and sociocultural interactions which she says don’t all share the same direct feedback that physical interactions have. Collins identifies various ways to interact with games and how players create and manipulate sounds in virtual spaces through role-play.

---


sion on what it means to interact with sound in games can blur the lines when trying to define music games, but it is worth considering when designing interactions in new music games.

Interaction and mapping can be integrated within the context of a music game. Stevens and Raybould (2012) discuss how music is used to support players’ actions and decisions when synchronized to game events. They describe the difficulty of aligning “expectation-based musical structures with unpredictable events” in video games, and the necessity of building tools to work toward a two-way interactive system for games in order to strongly connect player actions to player emotions via music. Music made with VR Sequencer does not have “expectation-based musical structures,” and it relies on unpredictable events for musical output. The VR Sequencer is a tool for making music using gaming hardware. It is not a game, nor is it a tool for making games musically interactive, but it does use interactivity to provide users with musical feedback. That feedback serves as an influence and motivation for users to make their next move. The interactions exist between the user and the sequencer. The player is both composer and performer, as the roles are ambiguous within this system.

1.4. Interactive Music Performance

Interactivity in music performance is separate from interactivity in music games. Frengel (2014) covers four models of interactive music and how they are used in live performances. Of note, the conductor model uses a performer to adjust parameters of a musical system and the reflexive model produces predetermined music based on performer input. Somewhere between Frengel’s conductor model and reflexive model lies the VR Sequencer. It is meant to

---

23 Ibid.
24 Ibid.
be a composition and performance tool. The time-spheres run on their own, but have editable parameters, fitting the sequencer into the conductor model. The dice rolling triggers pre-selected audio clips and the terrain adjustments alter audio effects, mimicking the reflexive model.

Frengel also describes attributes of live performance in relation to each of the four models. The merging of performer actions and audience perception are put into a context that tries to define the boundaries of traditional musical performances. It is important to note that the audience is a large part of interactivity in musical performances, which is not necessarily the same with interactivity in music games. Games can be played alone, where the only feedback from the player’s performance comes from the game. This can be a simulated audience or an aural cue that denotes success or failure based on input. Music games played alone are analogous to an instrument practice session in regards to audience feedback. One can perform the VR Sequencer in front of an audience or record a performance and share it through audio or video. This encompasses traditional musical performances, experimental methods of performance, and music game performance. The strong visual connections to the musical output by way of the playheads strengthens the audience perception and comprehension of the VR Sequencer performance. However, since the performer wears a VR headset, their perception of audience feedback is reduced. If the performer chooses to also wear headphones as monitors during a live performance, the audience perception is even further lowered. The lessening of audience feedback while performing in VR is congruent with playing a music game alone.

During a practice session, musicians set goals for themselves to improve their abilities. Gurevich (2014) takes the broad topic of skill into a narrower context of the concept of skill in music performance. After defining traditional thoughts that skill is goal-oriented and a blend of cognitive and sensorimotor skills, the author demonstrates how skill is different in the context of digital music systems. Gurevich argues the inadequacy of the cognitive-

---

26Frengel, “Interactivity and Liveness in Electroacoustic Concert Music”
sensorimotor dichotomy for interactive music.\textsuperscript{27} Even if there is an inadequacy for cognitive-sensorimotor skills in digital music, Collins and Frengel have demonstrated the existence of several other types of interactive models in which other skills may benefit. Users of VR Sequencer can set goals of improving the speed at which they spawn objects and edit settings, like a speedrunner would for a traditional goal-oriented video game, but the sequencer is designed to avoid the need for virtuosic skills.

Music games can make for unimaginative instruments when the player has no options to improvise or has no way to play so poorly that their music sounds bad. The traditional design of music games combines goal-oriented aspects of video games with some level of musical output control. When electronic instrument games are designed with free-form gameplay, they commonly restrict musical outcomes to avoid the risk of failure. Current music game types may reduce the chance of experimentation and expression for the sake of musical consistency, and the result is a game that only offers the illusion of musical control.\textsuperscript{28}

There is a need for digital musical instrument mapping systems to be applied to music video games. Games already use mapping systems, but they are used in a way that holds the players’ hands.\textsuperscript{29} By allowing players to output only correct notes at correct rhythms, the interface becomes too simple. When the mapping system is too easy, the interface becomes toy-like and uninteresting.\textsuperscript{30} By leaving a low skill entry level and introducing a high skill mastery level, an interface transcends the toy and becomes a musical instrument.\textsuperscript{31} Music video games are too much like toys that lose appeal after several hours of use. By adapting traditional digital musical instrument mappings, music video games can become impressive audio-visual instruments that require real skill to master, surpassing their roles as toys. VR Sequencer is a non-goal-oriented system which allows players the chance to

\textsuperscript{27}Gurevich, “Skill in Interactive Digital Music Systems.”
\textsuperscript{28}Pichlmair and Kayali, “Levels of Sound: On the Principles of Interactivity in Music Video Games.”
\textsuperscript{29}Ibid.
\textsuperscript{30}Wessel and Wright, “Problems and Prospects for Intimate Musical Control of Computers.”
\textsuperscript{31}Ibid.
correct musical outcomes through interaction. Performers have musical control, can play wrong notes, and can experiment, because the system uses indeterminacy to augment a simple mapping structure.
CHAPTER 2
INDETERMINACY

2.1. A Brief History of Aleatoric Works

In the mid-to-late 18th century, European composers such as Johann Philipp Kirnberger, Carl Philipp Emanuel Bach, and Maximilian Stadler published works that were to be composed on the fly by musicians rolling dice. There are also works published under Wolfgang Amadeus Mozart’s name, but the level of his involvement in the production of the musical dice games is uncertain. To compose the music, the musician would roll the dice and use the outcome to select a pre-composed measure of music. These pieces generally designated up to 12 variations for each bar of music, and maintained enough bars to conform to a specific musical style, like a minuet or polonaise. This allowed amateur musicians to compose music in their favorite genres without needing theory or composition skills. These musical dice games were composed with functional harmony in mind, therefore always producing ‘correct’ notes in appropriate keys.

The VR Sequencer shares some aspects with these musical dice games (i.e., dice rolling, pre-composed audio, approachability for amateurs), but has key differences that do not confine the outcomes to specific genres and functional harmony. The samples chosen for VR Sequencer are not necessarily tonal and encourage users to interact with the music after hearing initial outcomes. This ability to play outside traditional western tonal harmony introduces amateur musicians to experimental approaches to musical form and genre. Additionally, the VR Sequencer does not start with a score. Instead, it generates a score through composers’ actions, though the choices composers have are limited to the items they can

---


spawn and the parameters they can edit for each item. The aleatoric choices the composers have in VR Sequencer are the audio clips produced by playhead interaction with dice and sound effects presets. However, the VR Sequencer is designed to have such choices deliberately obfuscated.

Early 20th century aleatoric works use the piece’s notation or score to inform the musicians about the indeterminate aspects by offering choices to the performers. By the mid-20th century, John Cage and Iannis Xenakis were composing music through aleatory, albeit differently. Cage used chance to compose, while Xenakis used probability distributions.

John Cage adapted a Chinese text, *I ching* or *Book of Changes*, to determine compositional choices for his piece *Music of Changes* (1951). To incorporate the *I ching* into the composition, Cage followed the text by flipping coins and tossing sticks to determine outcomes, and mapped the outcomes to musical content. This is a form of aleatory known as chance because it uses methods where the outcomes and possibilities are known, but they exist within a controlled system. When flipping a coin, there are two possible outcomes with equal probability of occurring; heads and tails appear 50% of the time. Dice work the same way. A six-sided die gives each number a $\frac{1}{6}$ chance to appear. This equal weighting of probability defines chance-based procedures. When the weight of the probability changes, the chances of an outcome appearing change and a pattern emerges. Once a pattern emerges, the procedures become stochastic.

---


In 1954, Xenakis composed *Pithoprakta* using stochastic functions to determine musical parameters of acoustic instruments. By 1962, Xenakis began using probability functions for musical form, pitch, and duration for electronic music with his Stochastic Music Program. Xenakis began using stochasticism on smaller and smaller musical parameters, leading to his use of stochastic sound synthesis in 1967. This form of synthesis was aimed at altering sound pressure curves through stochastic means, which helped produce *Polytope de Cluny* in 1972. The types of probability distribution Xenakis used determined not only musical parameter values, but were also used as variables for other probability functions within his works.

VR Sequencer focuses more on chance than it does probability distribution functions. There are six shapes of dice in VR Sequencer, and each \(n\)-sided die shares equal weight of probabilities of outcomes with a \(\frac{1}{n}\) chance of happening. From the stochastic point of view, since all dice in VR Sequencer contain the numbers 1 through 4 as outcomes, the probability of hearing the first four audio clips of a given audio bank is higher than hearing the 13th through 20th because only one die has 20 outcomes. Due to the simplification of the audio effects presets mapping, they too are chance-based, because there is a \(\frac{1}{5}\) chance for a preset outcome. These chance-based methods of interfacing with the sequencer offer rather wide ranges of musical outcomes, especially when combined with multiple playheads and terrain changes. Despite knowing what the possible audio clip outcomes are, the spatial orientation of the objects combined with speed of playback and number of playheads contributes to a level of indeterminacy that allows for unique, irreproducible music.

### 2.2. Indeterminacy in the VR Sequencer Interface

The design of the VR sequencer removes synthesis from the project in favor of audio clips. Removing synthesis does not define indeterminacy, as proven with the 3-D reactive widgets developed by Berthaut, Desainte-Catherine, and Hachet which use audio clips triggered by

---

performer gestures\textsuperscript{10} but the method by which the audio clips are selected can define the level of indeterminacy. Indeterminacy in VR Sequencer is achieved through random number generation by way of dice rolling. The looping playhead mechanism in the VR sequencer plays at a regular interval, generating a form of control over random input\textsuperscript{11}

Users roll six varieties of dice to determine an audio clip stored in the sequencer. The dice are not numbered like their physical world counterparts, instead they are a monochromatic red color with no indicators as to what number they have generated (see Figure 2.1). Users must wait for the dice to be triggered by a playhead to find out what audio clip is being called. Furthermore, the playhead has a playback setting that also determines what audio clip to play. The playback setting, known as the 'layer', can be edited by the user and a different set of audio clips will be assigned to play back based on the number generated by the dice. The layer represents a sound bank, and the number generated by the dice represents an audio clip within that sound bank. These two levels of indeterminacy allow for a simple mapping system to represent an unexpected musical output.

Using the dice in a linear spatial configuration when composing produces a predictable result in musical timing of clips, but adding more playheads and editing each playhead’s parameters leads to a more intricate indeterminacy. This is made more complex by terrain changes, which cause the dice to roll, bounce, raise, and lower. Terrain changes are not always drastic. Sometimes the change is insignificant, barely affecting the dice configuration. Due to this, performers cannot always rely on formal changes when using the terrain generator. The audio effect presets, known as 'snapshots', are chosen randomly, but there are fewer possibilities of outcomes in this category. However, a drastic change in effects can make a drastic change in terrain seem even more impactful.

The stage in the VR Sequencer can be shifted with the click of a button. A shift means the terrain will rise or fall, creating hills and valleys which interact with the dice. The


\textsuperscript{11}Jensen, “John Cage, Chance Operations, and the Chaos Game: Cage and the "I Ching".”
settings of the terrain shift are limited to ranges that were selected during play tests in development. Random numbers from the ranges of the terrain settings are generated upon clicking a button, and the terrain shifts to the new settings over a random amount of time between four and twenty seconds. Shifts in terrain are under considerably less control to the performer than the dice rolling, and may even cause unintentional dice rolling as the hills and valleys are generated.

Within the terrain changes, a system of audio effects settings are changed as well. When a terrain begins to shift, an audio effect preset also shifts to a random selection of other audio presets, interpolating between the settings for as long as it takes the terrain to adjust. The audio effects presets are set differently for three different stages in the VR Sequencer. The audio effects chains are different based on which stage the performer chooses before starting the piece. These audio effects shifts, like the terrain shifts, are wholly out of control of the performer.

A performer using the VR Sequencer is making music with an aleatoric tool. That is, the sequencer is designed to have a “deliberate withdrawal of control”\textsuperscript{12} of the musical output.

\textsuperscript{12}Griffiths, “Aleatory.”
VR Sequencer can be truly left up to the random results of the program or changed based on the performer’s judgement.
CHAPTER 3
DIGITAL MUSIC, INSTRUMENTS IN VIRTUAL REALITY

3.1. Sequencers

Sequencers are commonly software applications that read musical sequences of audio clips or MIDI in a linear manner using a playhead. They are also known as digital audio workstations (DAW) when they include audio production tools like sound effects, looping mechanisms, and multiple tracks.

Popular DAWs have features far beyond the sequencing aspects, and allow for users to produce an entire piece of music from recording to mixing to mastering within one program. Each DAW may cater to a different user based on workflow and features. Each DAW shares the sequencing feature as a method of musical playback. A playhead moves across the screen from left to right, reading and playing audio based on the musical data it passes. The musical data can be audio recordings, MIDI, or sound effects automation. Loops are a common feature found in DAWs that send the playhead back in time after reaching a certain point in the music to loop back again, repeating the audio. Ableton Live (see Figure 3.1), takes advantage of the looping feature by encouraging users to write music while the audio is looping. Live even has a session feature where a user can collect a series of clips to trigger playback on the fly for live DJ sessions.

Ableton Live and other DAWs allow users to attach samplers to MIDI tracks. Users can use a sampler instrument for a single sample played back at various pitches based on MIDI input, or a sample rack which has a sample mapped to each different MIDI key input. Users

---


choose the samples they want, and trim them to their liking for playback. The Ableton Live playhead will play back recorded MIDI events in a linear fashion, looping if desired. These MIDI events are mapped to the samples chosen by the users. The playback controls include tempo, time signature, and loop length. Samples can be played in reverse by users selecting the reverse setting on the sampler before playback begins. The MIDI events can be edited to change start points, end points, and velocity, and can be deleted as well. Since Ableton Live is meant to be used a performance software, Ableton have produced a hardware controller called Push (see Figure 3.2), to interact with the DAW. Push allows users to input MIDI events which can playback virtual instruments, trigger loops, trigger samples, change audio effects, and generally control the form of the music being performed. There is much more beyond these features that modern DAWs do, but this covers the basics of what to expect from a consumer-grade sequencer.
3.2. A VR Sequencer

This project is an effort to design a way to perform and write music in virtual reality using an atypical musical interface. The prominent interaction for music making in VR Sequencer is dice rolling. Though dice are capable of providing numerical data which can be mapped in a wide range of ways, including determining synthesis parameters, effects parameters, dynamics, and tempo, VR Sequencer specifically uses the data to determine sample selection. Selecting a sample is not enough to make music, there must be a way to play the selected sample. In initial development, the dice were made to play samples when they were rolled, but this led to a single playback of a sample, which meant the user would need to constantly roll dice to make music. A familiar method of playback was employed to consistently trigger the samples. Like tape machines, DAWs use a playhead to trigger samples but DAWs

\[\text{Case, “Digital Audio Workstation”}\]
exist in a 2-D space, and a VR playhead must exist in a 3-D space. The first design of the VR playhead was a 3-D analog of a 2-D playhead in the form of a tall vertical cylinder in the center of the stage that grew in diameter. The cylinder would pass over dice and loop back to the center once it reached the edge of the stage. While this worked well initially, it became apparent that the playhead’s height needed to change as well to support the changes in terrain height. Ultimately, a spherical playhead that expands upward and outward across the stage was implemented to accommodate changes in the 3-D environment (see Figure 3.3).

This work defines the VR project as a sequencer that is able to write music and be performed and shared with audiences who may be unfamiliar with experimental music, virtual reality, and musical performance. A traditional sequencer can select sections of music to loop and the same expectations exist for the VR Sequencer as well. Furthermore, multiple playheads are able to be generated to create multiple looping processes. This increases the
complexity of the loops, generates interesting rhythms, and expands the capabilities of typical sequencers. The 3-D nature of VR Sequencer demonstrates how spatial orientation of the dice determine the sequences. This goes for playheads as well. Two playheads in different spots play the same orientation of dice differently. Since there are six varieties of dice, they can all play separate sonic material.

The VR sequencer replaces the 2-D interface of traditional sequencers with a 3-D virtual environment. The idea of a sequencer is expanded by changing the shape of the playhead, by including the ability to generate multiple playheads, and by determining sequences based on spatial configurations of objects in a 3-D environment.

### 3.3. A Brief Overview of Experimental VR Works

There are many ways to make music in VR currently. Some use the styles of rhythm games, like *Beat Saber* by Beat Games, while others offer experimental approaches to DAW interfaces, like *ControlRoom* by Persp3ctive. The approaches of experimental designers align with methods commonly found in digital media experimentation, such as finding a way to connect multiple pieces of software or hardware to create a new product. The approach of game developers, like those who made *Beat Saber*, align with the current VR gaming formats, such as access on several consoles or PC options using consumer VR hardware. This method works for the VR Sequencer because it provides access to experimental music making to the gaming community.

An experimental developer does not mind using an external application as an audio engine for a VR environment, or taking advantage of hand and finger tracking with a Leap Motion device or haptic gloves. Such a developer may also strive for virtual emulation of physical instruments. Projects which incorporate virtual spaces that can adapt to a user’s physical restrictions, tools that are influenced by spatial orientation of virtual objects, and some layer of abstraction between a user’s actions and the resulting outcomes are all preferable.
Figure 3.4. Ensemble performance of Rob Hamilton’s *Carillon*.

for designing music tools in VR, and all can occur within the realms of consumer-grade VR kits.

### 3.3.1. *Carillon*

Rob Hamilton’s *Carillon* is a virtual reality instrument meant for collaborative performances. *Carillon* was built with Unreal Engine 4 and Pure Data. Other than collaborative performances, *Carillon* can also be played as a solo performance or exhibited in a gallery as an interactive sound installation. *Carillon* performers use Leap Motion to control a virtual actor. Performer’s gestures control speed and rotation of a set of rings that produce procedural sounds based on Jean-Clause Risset’s additive synthesis bell. The collaborative version of *Carillon* was performed by Stanford Laptop Orchestra in 2015 (see Figure 3.4), having part of the ensemble wear head mounted displays (HMDs) and controlling rings in the virtual space with Leap Motion while the other part of the ensemble used granulators to further manipulate the audio produced by *Carillon*.5

---

This is a great example of a video game engine used in tandem with synthesis software like Pure Data in an ensemble setting and in an interactive setting. Carillon shows the flexibility of virtual reality instruments by being used in ensembles, solo works, and interactive installations. The core audio components are built in Pure Data and use Leap Motion to perform the rotating rings. The rotating rings offer an opportunity to use velocity and rotation as parameters for synthesis. This project is inspiring for its adaptability and interactivity.

3.3.2. Finnish VR Instrument Experiments

Mäki-Patola et al. developed software for creating virtual reality musical instruments. Their virtual reality system is a “cave-like room” setup for users to see the virtual world on a massive wall. Users interact with their system by way of 3-D glasses, haptic gloves, and sensors. Four instruments were designed using their platform: a xylophone, a theremin-like FM synthesizer, a virtual membrane, and a virtual air guitar. Each instrument is compared to its physical counterpart to determine efficiency and learning curve. The xylophone, for instance can be arranged in any bizarre manner, can be played by striking either side of each plate, and allows for users to pass through plates. The gestural FM instrument is compared to a theremin. Unlike a theremin, the FM instrument has visual feedback, which helps users learn the instrument quicker. The virtual membrane is capable of being transformed into a massive scale object which produces thunderous sounds not common with physical membranes. The air guitar uses a Karplus-Strong model and follows the users hands to determine articulation, pitch, and volume.⁶

Mäki-Patola’s software is capable of experimental design of instruments that mimic real-world instruments. There is still room for design of instruments that do not share physical counterparts. The authors’ interest in analysis of use helps with the merit of their experiments. The xylophone in particular would be interesting to experiment with in the context of

this project. Each plate could represent a sound and could be read by a playhead scrolling in various directions, creating different playback orders without moving the plates. The plates could also be rotating and have different sizes and shapes. This, however, would take away from the analogous musical counterpart for which the authors were striving, and possibly make the xylophone less like an instrument and more like a sequencer.

The nearest analogous counterparts in VR Sequencer are the dice which are meant to be rolled. The big difference here is that dice are not musical instruments in their own right. VR Sequencer still uses a familiar medium with which users can interact comfortably like the developers of these virtual musical instruments have done. Using a physical analog in a virtual medium can aid users ability to learn how the virtual medium functions.

### 3.3.3. Crosscale

Marcio Cabral et al. developed a musical interface for VR using Unity 3-D, an Oculus Rift HMD, and Razer Hydra controllers. The system allows performers to use swiping gestures across a plane of spheres which represent notes. Each sphere the performer moves across while engaging a button will play back the note written on the sphere. This system was designed this way to be like a modular piano keyboard. Cabral and team programmed the Crosscale to be able to show only notes within specific scales to aid performers. The distance between a C and a B would be 11 spheres, but shrinking the interface to a C major scale brings C and B physically closer at a distance of 7 spheres. This system only uses VR game hardware and takes advantage of virtual space by changing physical distances between notes so performers can use less physical motion to achieve difficult musical leaps. The swipe gesture is reminiscent of touch screen typing on smart phones. Users can also make chords by reaching away or toward their bodies and turning the controls to select a chord preset based on the note they are hovering over.

---

There is excellent use of space in Crosscale. Providing the user with the ability to change the interface is a powerful tool for performance. This project focuses on exploiting virtual reality’s ability to manipulate space in strong ways. Providing users with the ability to play multiples spheres on the top layer would be an interesting addition. The spheres are static unless the user rearranges the interface. There is, however, no way to examine and alter a single sphere for altering playback.

Cabral et al. visually identify the contents of each sphere in their work. This could be adapted into VR Sequencer by labelling each die with the name of the audio file it has selected, but would remove some of the indeterminacy tied so strongly to the project. Cabral’s use of virtual space adjustment is congruent with the teleport traversal system common in VR projects. It allows the user to move through virtual space without moving through physical space.

3.3.4. ChromaChord

John Fillwalk developed the ChromaChord for Oculus Rift Headset and Leap Motion Controller to track users’ heads and hands as they interact with a musical interface built in Unity. The Unity interface was designed to simulate the tangible properties of a physical musical instrument. The Unity interface has three views. The main view displays two columns of four cubes that play a note when touched. The second view displays a grid of root notes that change the main view tonality when touched. The third view is an orb that affects the frequency and Q of a filter. Interaction with the orb generates and/or removes particles around the orb as a form of visual feedback of the state of the filter. Fillwalk concludes that the ChromaChord simulates tangible user experience.

ChromaChord aims to replicate familiar interactions in Unity while blending in an abstract control mechanism, the orb. This project approaches concepts in VR Sequencer be-

---

cause the orb acts as an editable musical object that transforms as users engage with it, which is how the playheads in VR Sequencer function. The orb has characteristics of a button, in that it does something when pressed. The particle generation as a map to the filter parameters is an interesting step away from the button pressing interactions of the rest of the ChromaChord. Particle generation takes some control away from the user’s ability to filter a precise frequency and requires the user to listen carefully to the audio as they generate or remove particles. Interacting with particles on their own is a nice touch which further abstracts how a user is filtering the audio. Providing abstract control to users pushes ChromaChord toward instrument territory. Forcing a user to listen carefully to how they are changing the sound is representative of a theremin, or even a bowed instrument, and how performers should work within the VR Sequencer.

3.3.5. **Wedge**

Alec Moore, et al. designed *Wedge*, which uses hand gestures to build and play music in an immersive environment. *Wedge*’s design is based on three features: utilize 360 degrees around the user, allow users to customize the instrument and interface, and allow users to play chords with a single gesture. The team coined the term Composition-Appropriate Immersive Environment to define how the complexity of a virtual environment should reflect the complexity of a given piece of music. If the piece is simple, so should be the interface. This means the interface changes for each piece of music. Much like many other VR projects, *Wedge* uses Oculus Rift Headset with Leap Motion Controller to interface with a Unity project. To build a composition in *Wedge*, users take any of the twelve chromatic scale-based cubes and place them in a preferred order around the environment. To move cubes, users do specific horizontal hand motions. A set of cubes can be struck and played as chords.
To play back as a composition, users do specific vertical hand gestures and strike the cubes, which use velocity to determine loudness.\footnote{Moore et al., “Wedge: A Musical Interface for Building and Playing Composition-Appropriate Immersive Environments.”}

A unique aspect of \textit{Wedge} is the way it places sequencing software into a three-dimensional interface. Users can build a sequence in a three-dimensional virtual space in any order they choose, then play back the sequence in any order they choose. \textit{Wedge} offers the familiarity of a sequencer with the freedom of playing stochastically. This blurs the line of compositional tool and instrument. \textit{Wedge} is also seemingly related to \textit{Crosscale}. Both tools allow great changes in space to benefit the performer. The largest difference is that \textit{Wedge} is 360 degrees, and \textit{Crosscale} is a single plane with a limited field of view.

\textbf{3.3.6. Zebra}

For \textit{Zebra}, Figure 3.5, Zachary Berkowitz created a navigable virtual space for pre-composed music using the Multi-source Ambisonic Spatialization Interface (MASI) he developed which uses Open Sound Control (OSC) messages to connect audio engines to graphics engines. The pre-composed material used in \textit{Zebra} is tied to visible objects in the virtual environment. Each object emanates MIDI notes and their positions are generative. To perform the piece,
the user walks through the environment at their discretion, hearing the virtual objects play music based on proximity to the user and panned based on the orientation of the user’s headset.\[10\]

This project is so very near VR Sequencer in terms of spatial orientation, but diverges in technique. Both projects use a navigable virtual space, and include spatialization of musical elements within the virtual space. However, Zebra uses an external audio engine, Max/MSP, connected to MASI, which connects to Unity. VR Sequencer uses Unity’s audio engine, which may lack the ambisonic qualities MASI is capable of, but allows ease in distribution of the project by not requiring the use of Max/MSP. Zebra is an interactive game because it produces music using game hardware, engages the body, and streams audio into a virtual world as described by Collins.\[11\] As an installation and performance Zebra is an excellent example of taking full advantage of a virtual environment with audio at the forefront.

### 3.3.7. Perturb

Norbert Herber (2006) created a composition-instrument called Perturb, that composes and performs music via generative systems and user interaction. The article mostly discusses how to define a composition-instrument approach to create interactive musical systems, and demonstrates similar systems in place through a look at the history of experimental music. This source provides an example of an artist who has made a work that celebrates the blurring of playing and performing a game. Herber focused on interaction for musical creation and determined a mapping system for the user to perform and compose in the virtual world.\[12\]

“The concept strives to find a balance; neither the ability to ‘play’ nor ‘be played’ should dominate a user’s experience. If interactions are too direct (‘be played’ is too apparent), the piece becomes too much like an instrument and


\[11\] Collins, *Playing with Sound: A Theory of Interacting with Sound and Music in Video Games*

\[12\] Herber, “The Composition-Instrument: Musical Emergence and Interaction”
the significance of other aspects of the artwork can be diminished. Similarly, if an unresponsive musical environment obscures interactions and ‘play’ dominates the experience, the work loses its novelty in being tied to the course of a user’s interaction. The composition-instrument approach permits equilibrium between these two and as a result, acknowledges user interactions as perturbations in the overall musical system."

Direct interaction in VR Sequencer is mildly obscured by a random outcome. Users may interact directly by rolling dice or editing playhead parameters, but still receive an unknown outcome. This method exposes the responsive nature of the sequencer and emphasizes interaction, which Herber may deem unbalanced. However, the indeterminate nature of the sequencer and the reactive role of the user encourages performers to find their own balance between “playing” and “being played.”

3.3.8. Virtual Pebble Box

Sinclair and Wanderley developed a virtual environment for physical modeling and haptic feedback called Virtual Pebble Box. They used a combination of C++ for haptics and Pure Data for audio, connecting them with OSC. Virtual Pebble Box uses collision-based playback of audio and gestural control of spatialization, which is close to the dice-rolling mechanics of VR Sequencer. VR Sequencer triggers audio clips to play on collision, while Virtual Pebble Box combines physical synthesis and modal synthesis on collision. Furthermore, the Virtual Pebble Box uses more granular aspects of the primitive objects, such as acceleration, velocity, position, and rotation, as well as relative movements called constraints which act as hinges and joints. These properties offer unique and extensive options for music making, strongly benefiting the choice to use synthesis over sampling, and exceptional for audio effects parameters. The VR Sequencer terrain properties mimic these continuous properties found in Virtual Pebble Box.

---


3.3.9. Discussion

All of the aforementioned projects take space into consideration for developing virtual reality music tools, but not all take manipulation of time into consideration. In fact, the spatial considerations are mostly surface level, offering the user the ability to change the space between objects or allowing them to use 360 degrees of freedom to place objects wherever they like. There is potential for objects’ coordinates, rotation, and scale to factor in how the objects are heard, like in the Virtual Pebble Box. There is also potential for time to pass through objects at different rates and directions.

Some of these projects rely on Leap Motion for hand tracking, and others use either VR controllers or other camera tracking software. This demonstrates a disconnect with VR development and VR users. Granted, this is experimental research, but the tools developed by those who use specialized tracking systems are likely less accessible than those with standard VR systems. In order to share VR works with owners of VR systems, it is important to develop for VR systems while requiring no extraneous hardware, such as Leap Motion or an external camera, and stick to the controllers sold with the VR systems.

Many musical VR projects choose to use synthesis, rather than audio clips, which impart the feeling of playing an instrument and allow a performer’s virtuosity to strongly impact the resulting sounds. Rob Hamilton’s Carillon does this by incorporating additive synthesis via Pure Data combined with a performer’s skill in hand-eye coordination and musical timing to play bell-like instruments. VR instruments developed by Mäki-Patola, Laitinen, Kanerva, and Takala use custom gloves and sensors to mimic instruments like xylophones, guitars, and membranes while utilizing Karplus-Strong physical modeling techniques to produce audio. Given that their goal was to make virtual versions of physical instruments, the performers generally have some level of familiarity with how to play the instruments, and their virtuosity may depend on how the virtual version differs from the physical.

16 Mäki-Patola et al., “Experiments with Virtual Reality Instruments.”
There is a theme of connecting various digital media to design these virtual instruments, which the VR sequencer does not embrace. Use of leap motion devices, tracking gloves, and software communicating from outside the 3-D engine lends itself to high levels of experimentation and customization, but is not particularly well suited for commercial use or access. The VR sequencer is designed to be enjoyed by owners of consumer-grade virtual reality hardware. This choice opens access to experimental music making to the gaming community, who may not be familiar with writing and performing experimental music. Getting a copy of the VR sequencer should be as easy as downloading it from an online game store and playing it with the hardware VR gamers have at home. Custom hardware and extraneous software inhibit the ability to share these VR projects with non-experimental musicians.

---


CHAPTER 4
DEVELOPMENT, INTERFACE, AND COMPONENTS

The HTC Vive Input Utility (VIU) is being used for developing the VR Sequencer with Unity game engine. The VIU includes code available for quickly setting up interactions like grabbing, teleporting, and raycasting using the HTC Vive hardware as an input. There are several ways to set up grabbing objects. The two forms of grabbing used in the VR Sequencer are grabbing and throwing an object, and grabbing an object that stays at the transform position when released (unaffected by gravity). Teleportation is used to move the user across the stage. This type of locomotion allows a user to traverse a large amount of virtual space without physically moving themselves.

Raycasting is used for interacting with the user interface by means of pointing and clicking. The ray starts at the user’s controller and moves outward, away from the player to indicate where a user is pointing when selecting a button, toggle, or slider in a menu. The VIU also includes code for setting up the camera and player character in the VR Sequencer so the user can see the virtual environment with the VR headset at the user’s height. Changing the height of the player character helps make the virtual environment feel comfortable.

4.1. Three-Dimensional Assets

Sequences in VR Sequencer are created by interacting with two types of 3-D assets: dice and spheres. To make music, a user must roll a die and have it come in contact with a time-sphere. The dice decide what audio clip to play back and the time-sphere decides when to play it back. The time-spheres have editable parameters which can change the outcome, adding variety to an already varied musical outcome.

4.1.1. Dice

The 3-D assets which play back audio are digital manifestations of dice that come in six shapes. Five of the shapes are polyhedral with 4, 6, 8, 12, and 20 sides, while one is a 10-sided
Figure 4.1. A set of dice used to play Dungeons & Dragons.

trapezohedron kite shape, similar to those in Figure 4.1. The dice shapes in VR Sequencer, shown in Figure 4.2, were chosen because the sequencer has indeterminate qualities, and dice work well in chance-based operations. These digital dice are designed to be grabbed by a user when the user contacts the Vive controller with a die and pulls the Vive controller trigger to pick up the object. Once held, the die can be thrown across the stage to land on an indeterminate face. The face of a die that is touching the ground in the VR sequencer determines which sample is triggered by the time-sphere. Music is composed with the VR sequencer by rolling dice. The chance-based sample selection and spatial-location-defined rhythms lead to indeterminacy in performance and replication of a piece of music.

Audio Clip Assignment

The Unity project’s Assets folder contains all assets in the VR sequencer. Within the Assets folder is an Audio sub-folder which contains its own sub-folders full of audio clips. The audio clips from each folder are assigned to a face of each die using Unity Web Request. Unity Web Request reads a file path and provides the file to Unity as needed. In this case, Unity Web Request reads the Audio folder to find the number of sub-folders, then reads each sub-folder to find the WAV files at the end of the path. It assigns the WAV files of every sub-folder to each new instance of a die until it reaches the number of faces on the given die. So a four-sided die (D4) will contain the first four WAV files of each Audio sub-folder, and a ten-sided die (D10) will contain the first ten WAV files of each Audio sub-folder, and so on.
The decision to give them audio clips from every folder is due to the time-spheres doing the actual playback triggering based on a layer number, which is a Unity attribute that, in this case, acts as a sound bank. If there are five Audio sub-folders, then there are five layers for the time-sphere to choose. Time-spheres on layer 3 only trigger samples from the third Audio sub-folder, and time-spheres on layer 5 only trigger samples from sub-folder five. That way, a single die can have \(5 \times n\)-number of sides worth of samples accessible by users.

This method shows favoritism toward the first audio clips in a folder, so by altering the C# scripts of each die, one can arrange a D4 to always play the first four clips and a D10 to play the last ten clips. This behavior can be set to different layers as well. A D4 on layer 1 will read the first four clips, but will read clips five through eight on layer 2. Leaving the programming modular in this manner creates opportunities to avoid tedious repetition of audio clips.

4.1.2. Time-And-Space Spheres

Spheres which grow from a variable minimum size to a variable maximum size at a variable speed, constantly looping, represent the passage of time in the VR Sequencer. As 3-D assets come in contact with a sphere, their samples are triggered to play. This behavior mimics DAW playheads reading MIDI left-to-right in popular professional and consumer software sequencers.
Each sphere has a visible, grabbable portion called the space-sphere, and an intangible portion which loops and interacts with the audio objects called the time-sphere. The space-spheres are the tiny, opaque white dots and the time-spheres are the large transparent white portions of the spheres in Figure 4.2. The space sphere is movable when grabbed, but is unaffected by gravity. Therefore, users can grab a space sphere, move it to a new location, and let the sphere hang in the air where they release it. Time-spheres are attached to the space sphere’s location transform, wherever the space sphere goes, the time-sphere will follow. However, the user cannot collide with or grab a time-sphere, they must interact with the space sphere. The space sphere is like a glass orb and the time-sphere is like light pulsing out of the orb.

The two spheres together are referred to as time-and-space spheres. To spawn a time-and-space sphere, users click the menu button on the Vive controller, which brings up a menu where they can click a button to instantiate a new time-and-space sphere. The sphere will spawn in front of the user, and from there users can interact with it. To interact with the sphere, a user must contact the space sphere with the Vive controller and pull the Vive controller trigger to grab it. Once grabbed, the space sphere can be placed anywhere in the scene. While holding the space sphere, the user can click the menu button to bring up the time-sphere’s editable parameters, as shown in Figure 4.3. These parameters are layer, minimum size, maximum size, speed, reverse, and delete.

**Layers**

In Unity, a layer is an attribute of an object in the scene. The layer attribute is used in the VR Sequencer to selectively recognizing collisions between dice and time-spheres. A time-sphere’s layer can be set to a number between 1 and 15, each representing a folder containing audio files (sound bank). The layer chosen for the time-sphere determines what

---

sound bank is used when it contacts the dice. For instance, layer 1 relates to sound bank 1 which may contain guitar samples, and layer 2 relates to sound bank 2 which may contain drum samples. This is done so that multiple time-and-space spheres in a scene can add variety to the sample playback.

**Minimum and Maximum Size**

Minimum size determines the smallest size the time-sphere can be and maximum size determines the largest size the time-sphere can be. The default minimum size is a diameter of 0 meters, and the maximum default is a diameter of 150 meters. Changing the size alters which dice come in contact with the time-sphere, therefore altering the melody of the music. A time-sphere with a minimum of 80 meters and a maximum of 90 meters only comes in contact with a narrow portion of objects, while a minimum of 0 and a maximum of 125 will contact nearly every object in the scene. Users can take advantage of size to be selective of which dice they want to trigger.

**Speed**

Speed refers to the time it takes for the time-sphere to go from minimum size to maximum size. Speed is listed as integers on a slider, the default being 1. The higher the number, the faster the time-sphere moves. Users can change the speed parameter when they want to affect the tempo of a given sphere. There is a hidden unit as well called duration which can change the scale of speed units. A speed of 1 and duration of 1 means one second of time, but a speed of one and a duration of 0.5 means two seconds of time.

**Reverse**

Reverse is a check box that determines which direction the time-sphere moves. When unchecked, the time-sphere grows from minimum size to maximum size, but when reverse is checked it shrinks from maximum size to minimum size. This feature reverses the playback
order of the dice it contacts, which adds variety to the music. It is the equivalent of reading a MIDI file in retrograde motion in a traditional DAW.

**Delete**

The delete button does exactly as one may suspect, it deletes the currently selected time-and-space sphere. When composing or performing with the VR sequencer, deleting spheres breaks up the monotony in musical patterns and loops that have been playing too long.

**4.2. Musical Interaction**

Users need a way to quickly make large musical changes. To incorporate a mass music mix-up, a terrain generator has been developed which jumbles the dice at the click of a button. When users press the grip button on the controller, a perlin noise map is generated and applied to the terrain mesh. To smooth out the transition from an old terrain to a new one, the noise and height values interpolate from old to new over a span of a random time between four and twenty seconds. The changes can be extremely subtle, barely changing over long periods of time and doing little to jumble the dice, or the changes can be drastic, changing quickly over a few seconds, sending dice flying into the air, as shown in Figure 4.4.

Figure 4.3. First-person view of a time-sphere having its parameters edited.
Connected to the terrain changes are audio effects changes. There are three different stages with three different audio effects chains connected to the terrain. In Unity, audio effects can be made into presets called snapshots, and snapshots can be called at run-time. When a user clicks the grip button to change the terrain, a different snapshot is called and the values interpolate to the new preset over the course of the terrain change. This adds another level of indeterminacy to the music. The effects, combined with terrain values and dice being re-rolled create what feels like a new section in a piece of music, and can have potent impact on the form of the music.

4.3. Programming Interface

4.3.1. Audio Clip Playback

The \texttt{Toll(int b, int faceVal)} method created for the VR Sequencer is responsible for playing audio clips when time-spheres contact dice. The method takes two arguments: an integer representing a sound bank, \texttt{int b}, and an integer representing an audio clip within the sound bank, \texttt{int faceVal}. In this context, a sound bank is an audio clip array. To have
many sound banks, a list of audio clip arrays was created. For each folder containing audio 
clips in the Unity Assets folder, a clip array was added to the list. For each clip array in the 
list, the audio clips within each asset folder were added to the clip array using Unity Web 
Request.

Unity Web Request retrieves files from a given URL. In this case, asset directory paths 
were provided to retrieve WAV files and convert them into AudioClip types inside Unity. 
Each WAV file converted to AudioClip type was then added to the audio clip array associated 
with that asset folder. There are eight folders, or sound banks, of ten to twenty audio clips 
each stored in the Unity Assets folder. This means there are eight clip arrays in the list, 
and each clip array contains between ten and twenty AudioClip types converted from WAV 
files.

In the VR Sequencer, the time-sphere represents a sound bank and the dice represent au-
dio clips. To call the Toll(int b, int faceVal) method, the time-sphere layer parameter 
provides the int b argument to determine the audio clip array to use, and the dice provide 
the int faceVal argument to determine the AudioClip within the clip array to play. When 
the time-sphere collider is triggered by dice, the dice send an integer based on which of their 
faces is touching the floor, faceVal, and the time-sphere sends its layer number, b, to the 
Toll() method to playback a specific audio clip in a specific folder.

4.3.2. Random Number Ranges

Noise Map Data

The terrain shifting effects are caused by random number generation. In order to keep the 
terrain from getting too steep, ranges were created to limit the random numbers. The values 
fected by the random numbers impact a perlin noise map which is used to create the hills 
and valleys on the stage. The noise map parameters are noise scale, persistence, lacunarity, 
offset X, offset Y, and the stage’s mesh parameter called mesh height multiplier which affects
the mesh associated with the perlin noise map. The last value to change is the time it takes to interpolate between the current values and the new values.

The ranges used to limit the noise map data which provided the most desirable results are as follows. A noise scale between 5 and 10, persistence between 0.15 and 0.65, lacunarity between 1 and 3, offset X between -10 and 10, offset Y between -10 and 10, and mesh height multiplier between -1.7m and 1.7m. Given the ambiguity of size units in Unity, the noise map data values are based on what seemed appropriate during development, the height multiplier, however, is in meters. In the context of Unity, the player character is about 1 to 1.2 meters tall. Multiplying the mesh height up to 1.7 meters generates hills shallow enough that the player character can easily traverse and that encourage some dice to settle without constantly rolling down hill. The time values are a range of 4 to 20 seconds, and set at the same time as the aforementioned values.

Users must click the grip button on the Vive controller to shift the terrain. Once clicked, the `MakeRandomVals()` method is called which sets new random values within the ranges listed above to each of the noise map and mesh parameters. Afterward, a time value resets to zero, and a boolean declares the terrain shift is true. Once the time is set to zero, it begins to count up during each frame to the new randomly generated time value. As time is counting up, the C# `Mathf.Lerp()` method is called for each parameter to set the old value to the new value over the duration of the new time value. As the new time value is reached, the new parameter values are set to the current parameter values, the terrain shift boolean is set to false, and the process is ready to run again.

**Snapshot Effects**

Similar to the noise map data and mesh data of the terrain, the audio effect snapshot data is selected at random after the user clicks the grip button on the Vive controller. The audio effects chain in Unity looks similar to that of a traditional DAW, and the audio effects have various sliders for value changes which one expects to see in a music production environment.
During development, the desired value changes for each audio effect were saved as snapshots in the Unity Audio Mixer, as shown in Figure 4.5. A snapshot works like a preset, when a user wants to recall effects settings, they can do so through pre-programmed snapshots.

The audio effects included in the VR Sequencer are reverb, chorus, echo, pitch shift, and a resonator. There are three stages in VR Sequencer, each with a different audio mixer setup, and each mixer has different snapshots. For stage one, five chorus and echo snapshots are available, stage two has five reverb and pitch shift snapshots available, and stage three has five resonator snapshots available. The value settings for each snapshot were chosen through trial and error and listening during development.

When the Vive controller grip button is clicked, a random snapshot from the given mixer is picked, the new time value from the terrain shift is received, and a transition from the current snapshot to the new snapshot over the duration of the new time value begins. Once the transition is completed, a break in the coroutine occurs, and the script is ready to make a new transition at the click of the grip button. The Unity API expects
transitions between snapshots, which makes the scripting relatively straightforward with a
TransitionTo(duration) method available for each audio mixer snapshot.
Performing and writing music with the VR sequencer is exciting. When one starts creating objects and throwing them around and editing parameters and changing the terrain, the music just happens naturally. It is an incredible exercise in listening when performing with the VR sequencer. One’s instinct takes over when they hear something they like or don’t like and they know when to change what is happening or when to wallow in it. Something not present in the Primitives music videos is the sense of scale. The dice are huge! They are all close to 2 meters tall, so users feel like they have been shrunk down to a tiny size. But they can also throw the dice very far, which makes users feel very powerful.

While starting a performance, the music can be irritating and tedious. This is due to the natural amount of time it takes to set up an ideal musical pattern. The time between spawning and throwing a few dice and spawning and editing a new time-sphere feels short while making the actions, but feels longer when listening. This is a byproduct of the VR interface. In a traditional DAW, the music does not have to play back during the entire production of a piece of music. In the VR sequencer, and similarly in live coding music, the audio is not silenced while composing, which does not allow one’s ears to rest during the construction of the piece.

Once the piece starts, however, the music is engrossing. The multiple looping playheads add interest in terms of rhythm, melody, and harmony. The initial interest can wear off, though, which is why the terrain changes are essential to the VR sequencer. Changing the terrain generates a breath of fresh air into the music, and creates formal structure to the composition.

The music made with the VR sequencer can produce many genres and arrangements. The repetition of loops remains a constant across the music written with the VR sequencer. Even through terrain changes, the music will constantly be looping. However, the loops can feel less repetitive when edited in certain ways. Using one to three dice and only one to three
time-spheres at the slowest settings can create ambient music that rhythmically feels closer to wind chimes than music. The loop still exists in that context, but with clever spatial orientations of the objects, the loop can act like a polyrhythm changing the melodic order of dice playback every few loops.

Quick, repetitive, percussive music can be made as well by separating items so that there are as many time-spheres as there are dice, and each one follows its own tempo and rhythm to create a textural soundscape. A melodic style of music made with most sequencers can also be achieved by making one time-sphere at a maximum size and moderate speed playing back a series of dice in a specific order spaced equally apart. This method of organization gives a sense of real time signatures.

The audio clips used in the current configuration of the VR Sequencer include samples of Chase Mitchusson’s work from the last ten years and samples from masters of experimental composition including Pierre Schaeffer, Luc Ferrari, and Bernard Parmegiani. These samples were procured from Ableton, who were giving them away for a celebration of Institut national de l’audiovisuel (INA) and Groupe de Recherches Musicales (GRM). Using exceptional source material, like that found in the INA GRM sample pack from Ableton, lends to exceptional composition with the VR Sequencer. Less exceptional source material, like the original placeholders during development, tended to make less than compelling compositions. Those original placeholders were field recordings with found objects that tended to be short, percussive sounds. They did not fill the soundscape with interesting timbres or textures like the samples from master composers’ music have. Having a large virtual space to explore, and empty sounding samples makes the music sound small and un compelling. Noisy audio sources work well for the VR Sequencer. Mixing several noisy clips together to find a melody or rhythm makes for more interesting music than thin percussive source material can.

The VR Sequencer can produce music in different ways. The first way is a performance that viewers see from start to finish as a user spawns the initial dice and time-spheres and eventually deletes the time-spheres or removes the dice to end the piece. This style of
performance leans in to the indeterminacy of the sequencer. The performer can have some plan of action, but the actual musical outcome will be indeterminate. By embracing the indeterminate aspects of the VR Sequencer this way, the terrain changes and audio effects will have a strong impact on the music. If one does not wish to perform an entire piece this way, another option is to record a performance and edit together the best parts of the performance. This allows indeterminacy to affect the music, but leaves the composer in control of the final outcome. To take this idea further, a composer may want to record several performances then edit and layer them together in a DAW. The music is still a product of the VR Sequencer and chance, but the composer can make a collage of the best ideas to add depth to the musical potential of the VR Sequencer. Possible approaches that have not yet been explored include using the stage as graphic notation for live performers, allowing multiple users to contribute to the music as an installation, as well as a networked multiplayer version of the sequencer.

The first work made with the VR Sequencer, *Primitives*, uses a mixture of the first two methods described above. It is part performance and part composition. Each movement was recorded and edited to have a beginning and end. The way the movements end differs, but they all start after several objects have been spawned and music is being made. These starting point edits were chosen based on the medium of the release of *Primitives*, which is a YouTube video. Choosing a starting point of each movement is an attempt to interest YouTube viewers quickly. The medium of release impacts the decisions to edit the recordings. If there were no video, edits to the music would seem natural. In a video format, edits may confuse the viewer since the actions of the performer are so strongly tied to the musical output. Seeing the stage constantly change would not help first time viewers understand how the VR Sequencer works. Layering edits in a video format would also lend to confusion and seem deceptive. Editing together video clips from various performances may weaken the correlation of the arrangements of objects on stage if the aural portion is too different from the visual portion. Supplementing the music with additional layers of recordings from other
performances disconnects the correlation of performer actions and musical output, better lending itself to an audio release medium.

VR Sequencer generates a graphical score during performance, as shown in Figure 5.1. A top-down view of the stage will show a 2-D view of the spatial arrangement of various shapes of dice and time-spheres. Further, the score can be a still image or a video. A single shot of the stage, which will not demonstrate the size and speed of the time-spheres, can produce a score for musicians, but a video or a series of short video clips are the preferred formats for the graphical scores created by VR Sequencer.

The non-traditional methods of music generation found in the VR Sequencer provide an outlet for experimentation within the sequencer. New methods of musical performance are encouraged through placing importance on spatial orientation of objects and randomized musical output. The role of the performer is a reactive one, and that changes how musicians play with the VR Sequencer.
Primitives was written with three improvised performances using the VR Sequencer. Each movement is a clip of an improvisation that has been trimmed at the start to remove the initial setup time from the performance. Similarly, movements one and three are trimmed at the end of the movements to create a conclusion without having to delete every time-sphere that was created. Movement two, however, displays the technique of throwing dice outside of the ranges of the spheres to end the piece.

6.1. Touch

Movement I, Touch, focuses on multiple loops being affected by a larger, all-encompassing loop. The goal is to discover the outcome of loops being played by more than one playhead. In a DAW, there is only one playhead, but in the VR Sequencer there can be many playheads reading the same musical material simultaneously from different positions in space. VR Sequencer can playback the same content in many different ways, creating a new musical loop technique, similar to Alexandernaut’s Fugue Machine which allows playheads to playback content at different speeds and in different keys.1

At the start of Touch, 2-3 dice make up a looping pattern for a single small time-sphere. Similar loops are created with new dice and time-spheres. For each loop, the time-spheres are using a different layer to play different sound banks. This collection of small looping time-spheres with 2-3 dice each create a rhythmic foundation for the piece. A wave of change comes over this foundation when a time-sphere large enough to trigger all the dice on the stage is created, as depicted in Figure 6.1. This wave carries over the rhythmic foundation to create a melodic idea. The dice that are generating the rhythmic foundation are now the same dice generating the melody. The spatial orientation of each initial time-sphere

Figure 6.1. A large loop interacting with three smaller loops that make up the foundational material for *Touch*.

eventually determined the organization of the melody. To add to the melody, more dice are spawned inside the largest time-sphere. These new dice are only triggered by the largest time-sphere, unlike the initial dice, which are being triggered by two time-spheres each.

The importance of spatial orientation for making music in the VR Sequencer is present in this arrangement of the dice and time-spheres. To add variety and move to the next section of the piece, the terrain and audio effects are altered. The first terrain adjustment happened to be very subtle, but managed to slightly warp the pitch of the music. After a few more dice are spawned inside the largest time-sphere, the terrain and effects are altered again. This time, the terrain rises in height, which is apparent by observing the shallowness of the time-spheres in the scene. This terrain change has effectively shifted time as well as added some audio effects that make the music quieter and echo quickly.

Changes in the music are at the whim of the performer or composer. For the next musical changes, instead of shifting the terrain, the dice are re-rolled. In just two re-rolls, the music takes on a completely new sound. To explore the new sound, the smaller time-spheres are edited to match the size of the largest time-sphere on stage, then the terrain is shifted once again. The terrain moves just enough to flip a few dice around and the audio effects become greatly exaggerated. As a reaction to the unwanted effects, the terrain is changed
again, leading to a mellow, drone-like sound with a phrase that ends in a rhythmic idea that sounds like a printer error. There is an interesting shape to the musical phrase, and a nice contrast of ideas between how the phrase starts and ends. It starts with soft whole notes and ends with rigid and sharp-edged ticking.

After another terrain shift, the music transitions into a jittery sound that slowly gets washed out. The melodic and rhythmic ideas become less noticeable. Then, a warbling effect fades in, followed by the affected samples sounding like rocks scratching against the earth. This wash out effect brings a poignant end to the piece.

6.2. **Hold**

Movement II, *Hold*, is an experiment in multiple looping time-spheres which uses dice sparingly. Little pockets of loops are created that feature multiple time-spheres with different settings, overlapping each other, triggering a single die as shown in Figure 6.2. This demonstrates the power that the time-spheres hold in the VR Sequencer. With each time-sphere using different settings, a wide range of samples and melodic content can be created.

By replicating the loop pockets several times, the complexity of the musical content increases and the textural aspects of the music becomes more apparent. Similarly, adding a large time-sphere that interacts with each of the few dice on the stage breaks up the potential for monotony by obscuring the start and end points of the loops, while also emphasizing the musical content. After generating a large time-sphere that interacts with everything, the speed of the large time-sphere is increased to create a pulsing rhythm. Between each pulse are the contents of the pocket loops that started the piece, which creates a textural counter-rhythm. After the counter-rhythm is established, more dice are spawned in hopes to find new interesting material and interactions. These dice are thrown around to be triggered by the large pulse sphere, or by some smaller pocket loop spheres and the large pulse sphere.

When exploring new interactions in the VR Sequencer, it is important to listen. A performer’s listening skills become integral to the composition of the piece, especially with
Figure 6.2. Three sets of multiple time-spheres overlapping to trigger a single die.

so much content that produces indeterminate music. Scrolling through the sphere layers to find a new musical arrangement is an excellent exercise in listening while performing and composing. During layer experiments, a siren-like honking began to replace the large pulsing beats. This honking became the central idea for the next section, while other time-sphere layers were adjusted. Constant adjustment of sphere properties is a large part of the compositional process in the VR Sequencer. Parameter editing results in very quick musical feedback, allowing users the chance to rely on their ears before continuing, and not just rely on the interface’s sliders and buttons.

The layers and sizes of the pocket loops which started the piece were adjusted in search of new chaotic material. The idea was to make larger spheres which will trigger more dice on stage. Larger spheres in this context can build a wall of sound and create musical tension. Tension was eventually achieved with resizing loops, generating new large loops, and spawning new dice.
Composing music involves building and releasing tension. With an indeterminate system, one can’t know exactly how their actions will produce or relieve musical tension. Without a drastic change to the entire stage, the VR Sequencer interface would require a slow, determined method of relieving tension by editing every time-sphere. It can be done methodically, but the music, and audience for that matter, will suffer as the performer slowly troubleshoots each time-sphere until the tension is gone. The terrain changer is the VR Sequencer’s answer to this problem. At the click of a button, audio effects, height adjustments, and dice locations can be shifted. Once enough tension was produced in *Hold*, the terrain shift was utilized. An eighteen second long shift pushed the music up in pitch and increased reverb. The tension increased, but there was a sudden change in the loops and a new musical point was reached. Some of the dice rolled over, causing new samples to ring out, in particular, a metallic ding accents each “bar” of the loop. This whole section sounds more metallic than the previous, and has qualities reminiscent of pitched percussion. Again, the terrain button was clicked, and a ten second shift maxes out the reverb and washes away any noticeable musical attacks and transients from the prior section, with the exception of a bird call that has increased in speed and pitch so much that it sounds like a sound effect for a magical spell in a video game. This electric magic peeks out of what sounds like reverberating metallic walls. For now, the tension is displaced.

Another click of the terrain button makes a nineteen second long transition to a still reverb-heavy, yet lower pitched section which acts as a tonal change to the piece. The descent in pitch gives the feeling of a relaxation of the music, which finally alleviates the tension built several minutes prior. Noticeably, the electronic-magic bird sample is missing due to the dice having rolled over. When the terrain changes a final time, the reverb lessens, the bird calls return, and the percussive nature from the beginning returns. This is a cadential section which denotes the final part of the piece.

One way to stop the audio in the VR Sequencer is to manually remove the dice outside the boundaries of the time-spheres. It is a slow process, but can have a useful purpose if
the performer’s desire is to strip away each element of the piece clip-by-clip until there is nothing left. In the first movement, *Touch*, audio and video editing was used to place the ending, but in *Hold*, removing the dice one-by-one was more thematic to the core idea as a return to sparing dice use. Some of the time-spheres are so large, dice cannot be thrown far enough away to stop their audio clips from triggering. As the dice are thrown towards the outskirts of the time-spheres, new material is consequently being written. The spatial aspects of the sequencer really shine in situations like these. As the orientation of the dice is reorganized and they are thrown as far away as possible, the terrain shifter is utilized to take advantage of the new orientation of the sequences. As fewer time-spheres trigger the dice, the music becomes much thinner and quieter until there is a drone-like audio clip humming. The tension is eased, and as the last die goes flying off into the distance, the piece ends.

### 6.3. Release

The third movement, *Release*, initially focuses on a straight-forward use of the VR Sequencer, where one time-sphere is triggering a series of dice as shown in Figure 6.3. It begins with an example of music in the style of a traditional sequencer. This is not easy to accomplish in the VR Sequencer because there is no grid to accurately measure note lengths. This lack of rhythmic quantization means meter and rhythm are imprecise. Instead, a looser idea of meter can be explored, because the looping mechanism ensures playback a guaranteed speed and provides a rigid expectation of the musical loop.

Once the foundational loop is created, a time-sphere that is moving much slower and is much smaller than the primary time-sphere is created. This new small, slow sphere is only contacting a single die, and is being used to add some rhythmic emphasis to the current loop. One strategy to disrupt the current sequence is to re-roll dice, but a performer can expand on this idea by throwing dice at other dice to get more interaction when performing. Throwing dice at each other adds another element of surprise in the music. If the two dice hit each other, it is likely at least one of them rolls over and triggers new material to start
playing. After throwing dice, a new time-sphere is made that is small, but quick and triggers a percussive shaker sound from a single die. This shaker sound gets cut off by the largest time-sphere every few loops and creates an oddly balanced meter for the loop.

Once the foundation of the piece is expanded, the terrain shift is triggered, which is subtle and quick. The shift raises the pitch of the music and emphasizes some resonant frequencies. To explore the new audio effects, a few dice are re-rolled. A few elements that sound like saw waves at various pitches create an interesting descending melodic phrase. A lone die lies out of reach of the time-spheres. A new time-sphere is created for the lone die, then adjusted in size to have it just barely touch another nearby die. These begin to reinforce the recently introduced harmonic content.

The terrain is shifted which turns the sequence into a gorgeous descent in pitch and increases the reverb and delay. The harmonic content is really putting in work here to help that descending melody, which is now hauntingly sweet. The indeterminacy of the sequencer can lead to something magical happening in which the performer can wallow. A single die was created to mildly impact the music, and another die nearby was re-rolled to add some variety. The terrain is shifted once again, only to create even more beautiful music. The new effects are pitched even lower, causing the playback speeds to slow down as well, the reverb
has increased again, and the air is filled with a wall of harmony. What initially seemed like too much reverb, revealed a heart-melting melody. The music is breathing and swelling in and out carefully. The combination of effects, pitch, and speed led to more experimentation with more dice. Spawning dice tests how new material sounds in this environment, and in this case it managed to bring a glittery shimmer to the loop which contrasted so carefully against the full rich whole note harmony. The terrain is shifted one last time. As the terrain shoots the dice in the air and the audio effects distort the clips, the audio cuts out and the piece ends.

This is such a great example of the indeterminacy of the sequencer, how important listening skills are, and the ephemeral nature of music in general. Writing and performing with the VR Sequencers is like a fun game, and ending a piece is like watching Buddhist monks destroy their sand mandalas. A performer can be careful to create the art they want, but at the click of a button it is gone. The major difference between the monks and users of the VR Sequencer is recording your work for posterity. Of course, one could also forgo saving and recording by just sharing the moment in time with others.
CONCLUSION

Standard interactions such as spawning and grabbing dice and time-and-space spheres are working as designed. The large size and shape of the dice promotes throwing them sidearm style so they will roll smoother and farther than using an overhand throw. The playback of audio based on time-sphere values is also working as designed.

The size of the dice is novel, but for more comfortable interactions, they may need to be resized down to around 1 or 1.25 meters in height. That would hopefully lead to ease of rolling and throwing overhand, sidearm, and underhand. It is also worth developing a better method of menu selection than the current raycasting point-and-click style. A UI that takes advantage of the controller’s trackpad may be better suited for this application.

The most useful features of the current version are the terrain changes and the time-sphere parameters. When starting a piece, the composer may want to add several time-spheres to play back one die, and they can carefully edit the spheres to achieve an interesting rhythmic and melodic idea. The layer parameter is an obvious first change to make the die play different samples, but changing the size and speed can enrich the music with rhythmic intricacies. Given that the time-spheres are constantly looping, the music can become tedious, especially when first starting a piece. This is because of the time it takes to spawn several dice and make any changes to the time-sphere. That tediousness is eliminated with the terrain changes. Once the composer deems the music as stale, they can click a button to change the terrain and create something new almost instantly. If what the terrain generates is not worthy of the piece, again, the composer can click a button and move on to a new section. Since the audio effects are tied to the terrain changes, the outcomes of the changes can be fascinating and worth meditating on for a while. Each massive change can feel like a fresh start and lead to creative ideas on how to continue the piece.

While a way to change everything at once has been incorporated, what is lacking is the ability for a performer to command specific mass interactions such as stopping four time-spheres from triggering audio at one time, or the ability to roll only all D4 at once. When it
comes to performing, a constant loop can become grating to listeners, so allowing a performer to conduct the time-spheres becomes a necessity. The spatial music production interface from Wozniewski, Settel, and Cooperstock tackles this issue well by allowing performers to dip the volume of source nodes, like using a sampler or mixing board to bring audio in and out. This is a powerful tool that helps develop form and structure while writing music, and cleverly allows for mixing audio using 3-D space.

Music making in virtual reality should reach audiences at a consumer level. By stepping away from developing musical VR experiences with hardware and software outside the virtual reality headsets and controllers, people who lack access to special VR events, concerts, and conferences can now enjoy experimental music making in their home. Rather than focus on how acoustic instruments can be replicated in VR, the VR sequencer strives to relate more to musical samplers and sequencers. Beyond looping audio clips, the VR sequencer adds indeterminacy in melody making and allows for multiple layers of loops at once to aid in creating interesting rhythmic ideas. Basing interactions around how objects relate to each other in space creates an unorthodox method for writing and performing musical material.

---

APPENDIX A
EXTERNAL LINKS


Github Repository: https://github.com/cmtchssn/Dissertation
APPENDIX B
FIGURE PERMISSIONS

B.1. Figure 3.4

Figure 3.4 provided by Dr. Rob Hamilton via e-mail.

chase m <cmtchssn@gmail.com>
Thu, Jul 23, 2020 at 3:00 PM
To: hamilr4@rpi.edu

Hi Rob,

I am citing your work, Carillon, in my dissertation and was hoping you would allow me to use a photo from your paper. I have attached the photo to this email for reference. Would it be alright if I included it?

Best,
Chase Mitchusson
PhD student
Experimental Music & Digital Media
Louisiana State University

Hamilton, Rob <hamilr4@rpi.edu>
Thu, Jul 23, 2020 at 3:01 PM
To: chase m <cmtchssn@gmail.com>

Hi Chase,

Sure. Do you want a better quality image? I probably have one here somewhere.

-----------
rob hamilton, phd
assistant professor of music and media
rensselaer polytechnic institute
west hall, 307
e: hamilr4@rpi.edu
w: http://robhamilton.io
p: (518) 276-8083
t: @robertkhamilton
B.2. Figure 3.5

Figure 3.5 provided by Dr. Zachary Berkowitz via e-mail.

From: Chase Mitchusson <cmitc79@lsu.edu>
Sent: Wednesday, July 22, 2020 3:56 PM
To: Zachary A Berkowitz <zberko1@lsu.edu>
Subject: VR Work

Hi Zak,

I’m wrapping up my dissertation and citing your work, Zebra. Would you mind if I used a screenshot from the vimeo video in my dissertation? I’ve attached the photo to this email.

Best,

Chase Mitchusson
PhD Candidate
Experimental Music & Digital Media
Louisiana State University

Re: VR Work
Zachary A Berkowitz <zberko1@lsu.edu>
Thu 7/23/2020 9:24 AM
To: Chase Mitchusson <cmitc79@lsu.edu>

Hi Chase,

Yeah, absolutely! Let me know if you need any more info from me or anything!

Zak Berkowitz
IT Manager
College of Art + Design
Louisiana State University
108 Design Building, Baton Rouge, LA 70803
office 225-578-5409
zberko1@lsu.edu | lsu.edu | design.lsu.edu
BIBLIOGRAPHY


VITA

Chase Mitchusson is a composer, VR developer, and computer music instructor pursuing a Ph.D. at Louisiana State University. He is currently researching emerging technologies for the purpose of connecting music to digital media in order to create new interfaces which help people express themselves. As a part of teams in production classes and Global Game Jams, Chase has helped develop several games and experiences in Unity and Unreal Engine. His work with VR has been an effort to supplement creators’ toolkits and augment live musical performances. Other interests of his include coding web audio, field recording, and live multi-channel sound.