MUSIC IN VIRTUAL SPACE: THEORIES AND TECHNIQUES FOR SOUND SPATIALIZATION AND VIRTUAL REALITY-BASED STAGE PERFORMANCE

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
Zachary Andrew Berkowitz
B.M., New Mexico State University, 2010
M.M., Georgia Southern University, 2013
August 2016

ACKNOWLEDGMENTS

I would like to first and foremost thank my family: Lillia, Alan, Katie, and Jesse. Your love and support throughout my whole graduate education has kept me going. I could not have done this without you.

I would also like to thank the faculty of the Experimental Music and Digital Media program, Dr. Stephen David Beck, Dr. Edgar Berdahl, and Dr. Jesse Allison, whose guidance has been essential to this research and throughout my education at LSU. Also, many other faculty have also informed and supported this research and my education, including Dr. Brett Boutwell, Dr. Robert Kooima, Derick Ostrenko, Dr. Hye Yeon Nam, Dr. Griffin Campbell, and Dr. John Thompson.

Finally, I am very grateful to my cohort, who are an exceptional group of unique individuals that represent the very diverse nature of this field, and are also my friends. I would particularly like to thank Ben Taylor and Andrew Pfalz, who have been constantly supportive friends, colleagues, and sounding boards through all of the ups and downs of grad school life.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

This research explores virtual reality as a medium for live concert performance. I have realized compositions in which the individual performing on stage uses a VR head-mounted display complemented by other performance controllers to explore a composed virtual space. Movements and objects within the space are used to influence and control sound spatialization and diffusion, musical form, and sonic content. Audience members observe this in real-time, watching the performer's journey through the virtual space on a screen while listening to spatialized audio on loudspeakers variable in number and position.

The major artistic challenge I will explore through this activity is the relationship between virtual space and musical form. I will also explore and document the technical challenges of this activity, resulting in a shareable software tool called the Multi-source Ambisonic Spatialization Interface (MASI), which is useful in creating a bridge between VR technologies and associated software, ambisonic spatialization techniques, sound synthesis, and audio playback and effects, and establishes a unique workflow for working with sound in virtual space.

CHAPTER 1: INTRODUCTION

> How does art come into being?
> Out of volumes, motion, spaces carved out within the surrounding space, the universe.
> Out of different masses, tight, heavy, middling—achieved by variations of size or color.
> Out of directional line—vectors representing motion, velocity, acceleration, energy, etc.—lines which form significant angles and directions, making up one, or several, totalities.
> Spaces and volumes, created by the slightest opposition to their mass, or penetrated by vectors, traversed by momentum.
> None of this is fixed. Each element can move, shift, or sway back and forth in a changing relation to each of the other elements in this universe.
> Thus they reveal not only isolated moments, but a physical law of variation among the events of life.
> Not extractions, but abstractions:
> Abstractions which resemble no living thing, except by their manner of reacting.
>
> - Alexander Calder[1]

The relationship between music, sound, and physical space has long been a concern for composers and artists. For example, antiphony, the practice of distributing a composition between multiple choirs or other performance ensembles, is a staple of both Western music and music of cultures around the world. In Western music, examples of antiphony can be found in the polychoral music of Venetian school composers such as Giovanni Gabrieli, Mozart (e.g. *Notturno in D major* for four orchestras), and the works of European modernists such as Karlheinz Stockhausen (e.g. *Gruppen*) and Bruno Maderna (*Quadrivium*).[2]

Using loudspeakers, 20th and 21st century composers have been able to more easily achieve a similar effect. Early notable examples include the works *Williams Mix* by John Cage

---

[1] Alexander Calder, "How to Make Art?," in *Calder: Gravity and Grace*, ed. Carmen Giménez and Alexander S. C. Rower (London: Phaidon Press, 2004), 47.

[2] 'Blue' Gene Tyranny, "Out to the Stars, into the Heart: Spatial Movement in Recent and Earlier Music," *NewMusicBox: The Web Magazine*, January 1, 2003.

and *Octet I* by Earle Brown (1951-53), both of which were composed and realized for eight tracks of tape controlling eight separate loudspeakers,[3] and the multimedia installation *Poème Électronique* (1958) by Edgard Varèse, Le Corbusier, and Iannis Xenakis.

In the same way that these composers were able to explore virtual sonic space using loudspeaker arrays, musicians are now able to incorporate virtual visual space as well. Virtual reality technologies are at a peak of development activity and market popularity, and anyone with an average computer has enough processing power to generate a basic 3D world. There are several free software packages available for creating navigable 3D environments and games. And yet, there is arguably a gap between these technologies primarily meant for digital art, game design, and film, and the work of music performers and composers. The gap is both technological and artistic.

Technologically, there is a workflow problem. Composers wishing to work between 3D world-building technologies and sound creation tools largely develop their own software solutions for doing so. The amount of work and knowledge required can be discouraging. The creators of much of the recent and historic work in this area that is discussed in this study all use some form of custom software and/or hardware solution. For example, composers working in the AlloSphere at the University of California Santa Barbara use a specific custom software package, as do virtual world specialist composer Rob Hamilton and VR pioneer Jaron Lanier (all to be discussed further in Chapter 2 and Chapter 4). Much of this software is highly specific and

[3] Larry Austin, "John Cage's Williams Mix (1951-3): The Restoration and New Realisations of and Variations on the First Octophonic, Surround-Sound Tape Composition," in *A Handbook to Twentieth-Century Musical Sketches*, ed. Patricia Hall and Friedmann Sallis (Cambridge: Cabridge University Press, 2004).

is out-of-date or under-maintained. Therefore, as of the time of writing, composers wishing to experiment with this form of audiovisual composition largely have to rely on custom tools.

Artistically, there is a more complex problem. Most virtual reality technologies seem to be focused on complete immersion, often resulting in isolation. For some composers this might be desirable. However, music is inherently social, and perhaps all art is inherently social. The video game industry, for example, is meant to benefit most from the advancement of immersive virtual reality technologies, yet the very important social elements of gaming should not and cannot be ignored.

An additional artistic problem is the question of musical form in virtual space. What are the implications of incorporating virtual visual space in more traditional compositional techniques? In what ways can composers theorize and justify the act of composing music in virtual space? What compositional methods or considerations can composers turn to when 3D virtual audiovisual spaces?

With this research, practical and particular solutions to both of these problems, artistic and technical, are evaluated and offered through the development of a new software tool and by designing exploratory virtual reality-based compositions as live stage performances. To summarize the contributions of this research:

1) I have realized compositions in which the individual performing on stage uses a VR head-mounted display complemented by other performance controllers to explore a composed virtual space. Audience members observe this in real-time, watching the performer's journey through the virtual space on a screen while listening to spatialized audio on loudspeakers.

2) I have explored and documented the technical challenges of this activity, resulting in a shareable software tool called the Multi-source Ambisonic Spatialization Interface (MASI).

The research, technical, and artistic practice described in this document is organized into four categories: 1) historical background describing past musical works that utilize virtual reality in conjunction with a live audience (see Chapter 2), 2) technical description of the spatialization techniques and workflow tools designed and implemented in the MASI software (see Chapters 3-4), 3) discussion of the correlation between musical form and virtual space in two original compositions by the author (see Chapter 5), and 4) future directions and ongoing projects and research (see Chapter 6).

CHAPTER 2: VIRTUAL REALITY AND THE AUDIENCE

There are innumerable past projects that could be considered as influential in the development of a live virtual reality performance. Therefore, it is appropriate to focus on a few that are the most relevant and form the particular historical narrative that informed the creation of the new technologies and performance practices developed in this research. In other words, this history should not be viewed as exhaustive, but rather as a series of snapshots in time that reveal the practice and promise of virtual reality-based music.

2.1 Jaron Lanier's *The Sound of One Hand*

Virtual reality pioneer Jaron Lanier staged a live performance with virtual reality-based instruments in 1992 entitled *The Sound of One Hand*. The program notes for this performance read:

> A live improvisation on musical instruments that exist only in virtual reality [sic]. The piece is performed by a single hand in a DataGlove. The audience sees a projection of the performer's point of view. The instruments are somewhat autonomous, and occasionally fight back. The music changes dramatically from one performance to the next. The piece also demonstrates a variety of interface designs for handheld virtual tools.[4]

Lanier gave four performances from July 28-30, 1992, during the SIGGRAPH "Electronic Theater" in the Aerie CROWN Theater in Chicago, reportedly to a packed house of 5,000 seats.[5] He has also performed it a few times since the premier in Linz, Toronto, and New

[4] Jaron Lanier, "The Sound of One Hand," *Whole Earth Review*, no. 79 (Summer 1993): 30.

[5] Ibid.

Orleans.[6] The performance involved Lanier playing virtual reality instruments such as the "Cybersax" and the "Rhythm Gimbal."

Lanier's statements on the nature of this type of performance are unique and insightful. He states:

> I was delighted to discover that *The Sound of One Hand* created an unusual status relationship between the performer, the audience, and the technology. The usual use of rare and expensive high technology in performance is to create a spectacle that elevates the status of the performer. The performer is made relatively invulnerable, while the audience is supposed to be awestruck…
>
> *The Sound of One Hand* creates quite a different situation. The audience watches me contort myself as I navigate the space and handle the virtual instruments, but I am wearing EyePhones. Five thousand people watch me, but I can't see them, or know what I look like to them. I was vulnerable, despite the technology. This created a more authentic setting for music…[7]

Other influences are not as direct as Lanier. Most development in the area of virtual environments and spatialized sound focuses on a maximally immersive experience, necessitating VR head-mounted displays for the audience (so it is a solitary experience involving one participant at a time) or a CAVE-like environment with multiple projectors and speakers.

2.2 CAVE Environments

The CAVE system, developed by researchers at the University of Illinois Chicago Electronic Visualization Lab, was also showcased at SIGGRAPH in 1992. CAVE is a recursive

[6] Jaron Lanier, "Virtual Reality and Music," last modified January 10, 2010, accessed May 20, 2016. http://www.jaronlanier.com/vr.html.

[7] Lanier, "The Sound of One Hand," 32-3.

acronym for Audio-Visual Experience Automatic Virtual Environment, and is also a reference to Plato's cave allegory.[8] The authors describe the CAVE as:

> …a cube with display-screen faces surrounding a viewer… Its more recent instance is coupled with a head-tracking device. As the viewer moves within the bounds of the CAVE, the correct perspective and stereo projections of the environment appear on the display screens."[9]

The CAVE is an important development in virtual reality as it is, according the authors, nonintrusive. The authors state: "…in such an environment, the viewer is free to move at will, secure in the awareness of the real, as well as the virtual, aspects of the environment."[10]

This CAVE is also very relevant to this research because of the creators' attitude toward collaboration in virtual reality environments. The authors state in their SIGGRAPH '92 article, "One of the most important aspects of visualization is communication. For virtual reality to become an effective and complete visualization tool, it must permit more than one user in the same environment."[11]

The contemporaneous approaches to collaborative virtual reality by Lanier and the Electronic Visualization Lab researchers represent different philosophies about the potential role of the audience in the virtual reality experience. Lanier seemed to be most interested in a traditional view of performance. He wanted to break down the "suspension of disbelief"[12] required by the creators of the CAVE system, instead focusing on the human element behind the technology. While Lanier was immersed in his virtual reality environment, he wanted the

---

[8] Carolina Cruz-Neira et al., "The Cave: Audio Visual Experience Automatic Virtual Environment," *Communications of the ACM* 35, no. 6 (1992): 67.

[9] Ibid.

[10] Ibid., 68.

[11] Ibid., 70.

[12] Ibid., 65.

audience to not only see the environment, but to see him struggle to control it. He did not request that the audience suspend their disbelief, but rather requested that they focus on the very real human performer in front of them. To him, this human connection was missing from many hi-tech art and virtual reality demonstrations.

The CAVE creators, however, wanted suspension of disbelief. The audience was meant to be immersed in the environment; they were meant to be aware of their surroundings and not completely cutoff from the outside world, but still willing to forget about the outside world in order to have an effective virtual reality experience. The approach described in this paper is the former, as taken by Lanier. It is not required that the audience suspend their disbelief. Instead, the audience should see the human behind the technology. They should feel human connection, not just technical awe.

One prominent modern example of a CAVE-style performance environment is the work done in the AlloSphere at the University of California Santa Barbara. According to the developers, "The AlloSphere space contains a spherical screen that is 10 meters in diameter. The sphere environment integrates several visual, audio, interactive, and immersive components and is one of the largest immersive instruments in the world, capable of accommodating up to 30 people on a bridge suspended across the middle."[13] The developers also state:

> We designed the AlloSphere—a novel environment that allows for synthesis, manipulation, and evaluation of large-scale data sets—to enable research in science and art. Scientifically, the AlloSphere can help provide insight on environments into which the body cannot venture. Artistically, the AlloSphere can serve as an instrument for creating and performing new works and developing new modes of entertainment, fusing art, architecture, science, music, media, games, and cinema.[14]

[13] Xavier Amatriain et al., "The Allosphere: Immersive Multimedia for Scientific Discovery and Artistic Exploration," *IEEE Multimedia* 16, no. 2 (April-June 2009): 64.

[14] Ibid.

The AlloSphere is certainly incredibly sophisticated. It is also a unique space for artist-scientist collaboration, and serves as a potentially unprecedented interdisciplinary space. However, it is limited as an artistic performance environment. As stated, the AlloSphere can accommodate 30 people, which is a lot for an immersive environment (considering the cost and space requirements, it is very large) but does not come close to reaching the packed theater audience that Lanier was able to command.

As an artistic performance space/instrument, the AlloSphere is perhaps guilty of the misguided intention that Lanier describes of leaving the audience "awestruck" and making the performer seem "invulnerable." The audience is meant to perceive the technology as magical and otherworldly, rather than perceive the human effort behind the project. This approach serves a very important place in the development of new technologies and the pushing of artistic boundaries and the artist/scientist relationship, but it can be viewed in contrast to an approach that leaves the technology apparent and the human performer vulnerable in order to gain a more meaningful and urgent audience/artist connection.

2.3 Virtual Poem

Another example of this type of virtual world environment within the field of electroacoustic music was the reconstruction in the mid-2000s of Edgard Varèse, Iannis Xenakis, and Le Corbusier's famed 1958 multimedia experience *Poème Électronique* in virtual reality. According to the authors of this recreation, they used:

> …an integral approach to regain access to *Poème Électronique* through virtual reality (VR) technologies that include a reconstruction of the physical space in computer graphics (CG). We pursued a simulative approach to the reconstruction of the work; that is, we simulate the processing/temporal aspects of the artwork by integrating in a VR

> software environment all the findings of a thorough philological investigation, converted into a digital format. The final work can then be delivered as a VR installation.[15]

Therefore, this installation took a different form than the CAVE. Instead of a physical space in which participants would move, the "Virtual Poem" utilized a VR head-mounted display. This provides a very complete immersion. However, the installation in its initial form can only be experienced by one person at a time.

The choice to reconstruct *Poème Électronique* was natural for the creators of the installation. They perceived *Poème Électronique* to be a proto-VR installation, stating "The integration of music and image inside a space has been said to make *Poème Électronique* (here intended as the entire installation) the first modern multimedia event—and, one could argue, an *ante litteram* virtual-reality installation."[16]

The connections between *Poème Électronique* and virtual reality are indeed strong, as the work was meant to be navigable, which is the goal of most VR works. The main feature of *Poème Électronique* was the space in which it was played. Funded by the Phillips Corporation, Xenakis designed a building at the 1958 Brussels World's Fair with the express purpose of presenting the tape music of Edgard Varèse. According to historian Thom Holmes:

> It was built in the shape of a circus tent with three peaks, a shape that was also likened to that of a sheep's stomach. Inside were 400 loudspeakers to broadcast the sound in sweeping arcs throughout the pavilion. The music was accompanied by visual projections selected by Le Corbusier.[17]

---

[15] Vincenzo Lombardo et al., "A Virtual-Reality Reconstruction of Poeme Electronique Based on Philological Research," *Computer Music Journal* 33, no. 2 (Summer 2009): 26.

[16] Ibid., 30.

[17] Thom Holmes, *Electronic and Experimental Music: Technology, Music, and Culture*, 4th ed. (New York: Routledge, 2012), Loc. 10130. Kindle.

Thus, Le Corbusier, Varèse, and Xenakis were trying to achieve a navigable composition with *Poème Électronique*. The content was linear, but it was presented to an audience as something that was experienced not only in linear time, but in 3-dimentional space. The audience would wander through the space, 500 people at a time, experiencing the music and projections in the space. Further, the music was composed with the space in mind. According to Holmes, Varèse's piece, along with the work *Concret PH* by Iannis Xenakis:

> …were composed knowing that they would be projected spatially using a matrix of loudspeakers and three channels of tape inside the Philips Pavilion. The works were played using a 3-track, 35 mm perforated magnetic tape system, the output of which was fed to 325 wall-mounted speakers and 25 sub-woofers around the floor. The projection of the sound and images was controlled by 15-track control tape that automatically switched the audio amplifiers and image projectors. The amplifiers were connected to groups of five speakers and they were switched on and off in a sequence across the space so that the three tracks of sound appeared to be moving in two or three directions at the same time around the audience.[18]

This level of control and sophistication in spatialization was unprecedented at the time. The movement of the sound, and the position of the listener in relation to the sound source, was just as important a compositional element as the sounds themselves. The same can be said of virtual reality. VR artists must not only consider the visual and sonic content of the work, but must consider the ways in which the audience might move around in the virtual space, and how they will perceive the content depending on where they are within the virtual space. Particularly given the current novelty of the VR experience, these considerations are of the utmost importance.

Through Lanier's *The Sound of One Hand*, the CAVE, the AlloSphere, and the VR version of *Poème Électronique*, one can see a range of different modes and methods of audience interaction, ranging from the outside observer (Lanier) to the immersed yet physically in control

[18] Ibid., Loc. 10154.

(CAVE and AlloSphere) to the completely immersed, cutoff from the outside world (VR *Poème Électronique*). Yet another way that an audience can perceive virtual reality is via the concept of virtual reality itself and the notion of voyeurism. The audience watching Lanier can be seen as voyeuristic, observing from the outside while not participating, yet still engaging with and enjoying the work in the same way that a voyeur might enjoy observing the actions of another.

2.4 Additional Works

One work that deals with voyeurism and the conceptual framework of virtual reality (particularly the head-mounted display) is Maurice Benayoun and Jean-Baptiste Barrière's installation "So. So. So (Somebody, Somewhere, Sometime)" (2002). The work is described by Kristine Stiles and Edward A. Shanken as:

> …an interactive media installation that tracks retinal movement to create a palimpsest of memory, again, from which a viewer cannot escape. Looking through binoculars fitted with VR screens, the viewer/voyeur searches for and hones in on a focal point. The darting of his or her eyes is recorded to what the artist calls the collective retinal memory, which registers and projects to the outside audience a visual map of the viewer's interest, thus transforming the viewer into the viewed.[19]

This work, therefore, represents a far end of the spectrum as a work that deals almost exclusively with the conceptual role of the audience in VR performance. Rather than moving into a virtual world that is meant to be immersive, the audience is focused entirely on the human that is in the virtual world. Rather than seeing what the human performer sees, the audience sees a collage of what the human performer focuses on with her eyes, as a "collective retinal

[19] Kristine Stiles and Edward A. Shanken, "Missing in Action: Agency and Meaning in Interactive Art," in *Context Providers: Conditions of Meaning in Media Arts*, ed. Margot Lovejoy, Christiane Paul, and Victoria Vesna (Chicago: Intellect, 2011), Loc. 784. Kindle.

memory." Lanier expressed a similar sentiment of transforming the viewer into the viewed. He was the viewer, yet the audience was also expected to watch him.

The viewer/viewed perspective is far less explored in musical and media art performance than the immersive perspective. However, this perspective is currently being explored in more popular social media. For example, the popular social media site Twitch specializes in enabling users to broadcast themselves playing video games.[20] This trend is in fact so popular that Amazon acquired Twitch for $1.1 billion in 2014.[21] Watching others "perform" in virtual environments has become big business. The numbers are staggering. For example, in October of 2013, 32 million people watched a championship of the game *League of Legends*, and Twitch is the fourth-largest user of internet bandwidth in the United States.[22]

The users who tune in to Twitch are not simply watching users navigate through virtual worlds, however. They are interested in the gamers (performers) themselves. This interest is perhaps the same one that Jaron Lanier was seeking to spark with his 1992 performance. Twitch and Lanier turn what might be a very solitary activity into a public experience in which the human performer seeks contact with the audience, using the technology as medium in which to do so.

Another contemporary example of involving the audience in virtual world musical performance is the work of Rob Hamilton. Hamilton has created modified versions of game engines in order to realize compositions in virtual space (which will be discussed further in

[20] http://twitch.tv/

[21] David Carr, "$1.1 Billion: This Isn't Child's Play," *International New York Times*, September 1, 2014.

[22] Ibid.

Chapter 4.1). An example of these compositions is *ECHO::Canyon* which, according to Hamilton, "…creates a reactive musical environment within which the idiomatic gestures and motions of avian flight and biologically-based creature motion are mapped to musical sound-producing processes."[23] The piece also uses interesting sound spatialization techniques:

> Data generated in the *ECHO::Canyon* environment drives a multi-channel sound server written in Supercollider featuring software-based Ambisonic encoding and decoding to spatialize sound around a multichannel speaker environment. Individual actors and sound generating events in the game environment are spatialized throughout the sound field at locations representative of their position in the rendered environment itself, creating a correlated spatial mapping between virtual action and real world sound. Audiences in a traditional concert setting watch the performance on one or more projector screens showing camera views from a unique camera operator, moving throughout the environment.[24]

The idea of a "unique camera operator" in Hamilton's work provides an interesting contrast. Rather than use a first-person perspective, Hamilton separates the camera so that the audience's listening/viewing perspective is not necessarily first-person, but is dependent on the location of the independently operated camera. This can be interpreted as a decidedly third-person audience perspective.

While Hamilton's work is similar to the original virtual reality performances described in this research in that he uses game engines and performs with virtual world environments for a traditional concert audience, there are some key differences. Hamilton seems to be focused most on the sonification of gestures and virtual choreography. The key element in *ECHO::Canyon* is the bird that is used to explore the space. The bird's movement (wings, etc.) are used to control synthesis processes. The creative work using game engines described here, on the other hand, is

---

[23] Robert Hamilton, "The Procedural Sounds and Music of *Echo::Canyon*," in *Music Technology Meets Philosophy: From Digital Echos to Virtual Ethos*, Proceedings of the International Computer Music Association (San Francisco: International Computer Music Association, 2014), 450.

[24] Ibid.

focused on musical form and 3D space. Using gestures and choreographies of characters to drive sound synthesis is intriguing, but doesn't address certain fundamental artistic issues of musical form in 3D space.

There are, of course, many more examples of performance-oriented VR works utilizing CAVE environments, head-mounted displays, internet streaming, etc. Yet, the early work by Lanier remains the closest example of what this research seeks to achieve, and that work has not been preserved in the form of recording or documentation of the performance. It is for this reason that it is vital to turn the VR experience outward, and to create more performances that can be delivered to the 5,000 person audience that Lanier was able to reach.

CHAPTER 3: TECHNIQUES FOR SOUND SPATIALIZATION

One of the primary technical considerations involved in this project is the realistic spatialization of sound sources. In the envisioned virtual space, sound sources and the first-person user/performer are continuously moving and the sound sources must appear to an audience as though they are realistically located in space.

In order to share an immersive audio experience with a live audience, options besides headphones must be explored. Current technologies for sound spatialization in virtual reality focus mostly or exclusively on the headphone experience. While this may be the most ideal environment for accurate perception of sound location, a binaural or even stereo mix leaves the audience out of the immersive audio world. A solution to this problem lies in the spatialization of sound over larger speaker arrays, or "surround" sound.

5.1 and 7.1 channel speaker systems are common in movie theaters and some consumer systems. These setups can be leveraged to provide a more immersive experience. For an even more accurate experience, many universities and organized festivals of electroacoustic music provide an octophonic ring of speakers or more complex installed arrays. Therefore, it is essential to create a sound spatialization system that is able to be used in multiple environments, from headphones to a speaker array with an arbitrary placement and number of speakers. This adaptability is what is currently missing from available sound spatialization platforms for 3D environments, and is therefore an essential part of this research. This chapter discusses how sound source distance and location can be emulated on a speaker array (or headphones) using virtual acoustics and ambisonic panning.

3.1 Perceptual Cues

Significant prior research has been done in the field of moving sound source simulation, and simulated moving sound sources have been used in creative musical composition for decades. One of the early pioneers in this research is John Chowning, who described methods for the "simulation of moving sound sources" in a 1971 article for the Journal of the Audio Engineering Society. Chowning states, "To locate any real sound source in an enclosed space the listener requires two kinds of information: that which defines the angular location of the source relative to the listener, and that which defines the distance of the source from the listener."[25]

Chowning describes the cues for angular location as "1) the different arrival time or delay of the signal at the two ears when the source is not centered before or behind the listener, and 2) the pressure-level differences of high-frequency energy at the two ears resulting from the shadow effect of the head when the source is not centered."[26] In order to simulate the angular cues, Chowning used a quadrophonic system with a speaker in each corner of a square space, with the listener seated as close to the center as possible. According to Chowning, due to the fact that the exact location of the listener cannot be known, "…any cues to location of a source which are dependent upon delay, phase, and orientation of the listener's head are inappropriate. The cue to angular location must be introduced by a changing energy ratio of the direct signal applied to a loudspeaker pair."[27]

---

[25]John M. Chowning, "The Simulation of Moving Sound Sources," *Journal of the Audio Engineering Society* 19, no. 1 (January 1971): 2.

[26] Ibid.

[27] Ibid., 3.

The cues for perceiving distance from a sound source are described by Chowning as "1) the ratio of the direct energy to the indirect or reverberant energy where the intensity of the direct sound reaching the listener falls off more sharply with distance than does the reverberant sound, and 2) the loss of low-intensity frequency components of a sound with increasing distance from the listener."[28]

These cues can be simulated through means described by Martin Naef et al.[29] The specific components used for simulating the distance cues described by Chowning and Naef are as follows:

1) Sounds are delayed based on distance from the listener, as described by the formula $t_d = \frac{D_s}{v_{sound}}$ where D_s is the distance from the sound source to the listener and v_{sound} is the speed of sound.[30] This variable distance delay produces the Doppler Effect when sound sources or the first-person user are moving through virtual space.

2) High-shelf filtering is applied to simulate the absorption of high frequencies in air, at a rate of -4 dB per kilometer above 1 kHz.[31]

3) The level of the sound source is scaled according to the formula $L_d = \frac{D_{ref}}{D_s}$ where D_{ref} is a reference distance at which the sound is considered to be at the original volume.[32]

4) Reverb is used on a separate bus and sources are sent to that bus with level scaling according to the formula $L_{rev} = 1 - (\frac{D_{ref}}{D_s + D_{ref}})^2$.[33] This produces the effect described by Chowning of the direct sound falling off more sharply than reverberant sound, since the reverb send level and direct sound level are inversely proportional.

---

[28] Ibid.

[29] Martin Naef, Oliver Staadt, and Markus Gross, "Spatialized Audio Rendering for Immersive Virtual Environments," in *Proceedings of the Acm Symposium: Virtual Reality Software & Technology* (New York: Association for Computing Machinery, 2002).

[30] Ibid., 68.

[31] Ibid.

[32] Ibid.

[33] Ibid., 69.

Chowning's cues provide the basis for the spatialization methods implemented in this study. There are other models to consider, however. For example, F. Richard Moore argued that since the positions of listeners and sound sources in a concert hall are unpredictable, "…a practical model for spatial processing must be based on physical characteristics of the real or imaginary space or spaces to be simulated. This suggests modeling the playback situation itself, rather than the details of the listener's perception of it…"[34] Therefore, Moore suggested a method that took two spaces into consideration: an "outer room" and an "inner room." Moore describes:

> The *outer room* represents the illusory acoustic space from which the sounds emanate. The *inner room* represents an actual or intended performance space that holds listeners and loudspeakers. Loudspeakers are modeled as acoustic "windows" communicating with the illusory space outside the perimeter of the listening space…
>
> For example, the inner room might be a living room with loudspeakers placed in each corner, while the outer room's specifications suggest the acoustic properties of a reverberant cathedral. From within the listening space, we listen through "holes" in the walls to sounds that exist in the outer room.[35]

Although Moore's interpretation is an interesting way to deal with the fact of unpredictable listener locations, and deals specifically with the notion of a virtual acoustic space, it does not account for the *visual* aspect of being within a virtual world (not looking through "windows") and is therefore not as directly useful for virtual world spatialization as is Chowning's model.

3.2 Ambisonics

Chowning's method of panning was fixed to the four-speaker implementation. However, several panning methods enable sources to be panned to a specific angular location around the

[34] F. Richard Moore, "A General Model for Spatial Processing of Sounds," *Computer Music Journal* 7, no. 3 (Autumn 1983): 7.

[35] Ibid., 8.

listener while accounting for different or arbitrary speaker setups. One of the most popular of these methods is ambisonics.

Ambisonics is the most useful for a virtual reality live performance because of its flexibility and scalability. However, there are other techniques to consider. The most popular method of localizing sound within virtual reality environments is through the synthesis of "binaural" sound. Binaural refers to a method of recording (or synthesizing) and reproducing sound that reflects accurate localization of sound sources by mimicking the response of the human ear. Humans are able to locate sound in 3-dimentional space through a combination of what are known as inter-aural time differences (ITD), inter-aural intensity differences (IID), and head-related transfer functions (HRTFs).[36] ITDs describe the time difference between the arrival of sound at each ear. IIDs describe the difference in intensity at each ear. HRTFs describe how sound is altered by the head, ears, and torso.[37]

Binaural recordings can be made by using two omnidirectional microphones in the approximate position of ears and placing a baffle in between them, representing the head. Alternatively, one may use a dummy head or wear in-ear microphones.[38] However, it is also possible to do binaural "panning" of sounds, without having recorded them in a binaural fashion. This is possible by applying measured HRTFs to an existing sound, therefore synthesizing sound location. Ideally, the HRTF is obtained from the listener. However, acceptable results can be

[36] Shane Hoose, "Creating Immersive Listening Experiences with Binaural Recording Techniques," *College Music Symposium* 55 (2015), http://dx.doi.org/http://dx.doi.org/10.18177/sym.2015.55.mbi.10863.

[37] Ibid.

[38] Ibid.

obtained using non-individualized HRTFs as well. This is discussed in detail by Wenzel et al. in their 1993 article for the Journal of the Acoustical Society of America.[39]

The method of synthesizing binaural audio using HRTFs is currently applied by the makers of the popular Oculus Rift VR head-mounted display. The company has released a development kit that allows developers to easily incorporate binaural audio into their projects. Therefore, it should be expected that more VR games and other content in the near future will effectively incorporate binaural audio. However, this method is strictly relegated to the headphone experience and Oculus has not provided any means to spatialize over a speaker array, stating in regard to sound localization, "It is possible to mimic this with a speaker array, but it is significantly less reliable, more cumbersome, and more difficult to implement, and thus impractical for most VR applications."[40]

As is evident, the most prominent companies involved in the commercial development of VR technologies are not particularly focused on the issue of disseminating spatial audio to large groups of people. However, there are methods that make this practice more reliable, less cumbersome, and less difficult to implement. Chief among these methods is ambisonics.

Ambisonics was first described by Michael Gerzon in his 1973 paper "Periphony: With-Height Sound Reproduction."[41] Ambisonics is described by David Malham and Anthony Myatt as "essentially a two-part technological solution to the problems of encoding sound directions

---

[39] Elizabeth M. Wenzel et al., "Localization Using Nonindividualized Head-Related Transfer Functions," *Journal of the Acoustical Society of America* 94, no. 1 (1993): 111-23.

[40] Oculus VR, "Introduction to Virtual Reality Audio," *Oculus Rift Developer Center* (2015), accessed November 15, 2015, https://developer.oculus.com/documentation/audiosdk/latest/concepts/book-audio-intro/.

[41] Michael A. Gerzon, "Periphony: With-Height Sound Reproduction," *Journal of the Audio Engineering Society* 21, no. 1 (January/February 1973): 2-10.

(and amplitudes) and reproducing them over practical loudspeaker systems so that listeners can perceive sounds located in three-dimensional space."[42] Sounds are encoded (or recorded) in "B-format," which contains four channels for periphonic "with-height" reproduction, as sound fields. These sound fields can then be decoded and reproduced on a sound system with four or more speakers to surround the listener.[43] A flexible decoder can reproduce the encoded B-format sounds on any arbitrary array of speakers, although a regular array such as a square, cube, or octagon is desirable for accurate reproduction. This flexibility is ideal for a performative virtual reality situation, and was proposed as such by Malham in 1993.[44]

The four channels of B-format represent spherical harmonics. This four-channel encoding method is known as "first-order" ambisonics. Conceptually, sounds are positioned on the surface of a "unit" sphere—a sphere with a radius of 1. One can position sounds using polar coordinates, described as azimuth (θ) and elevation (ϕ). Distance from the sound source is perceptual, and is implemented through the use of the previously described formulas. The four harmonics of B-format are conventionally labelled as W, X, Y, and, Z, where X is the front-back axis, Y is the left-right axis, Z is the up-down axis, and W is the omnidirectional pressure component.

[42] David G. Malham and Anthony Myatt, "3-D Sound Spatialization Using Ambisonic Techniques," *Computer Music Journal* 19, no. 4 (Winter 1995): 62.

[43] Ibid.

[44] D.G. Malham, "3-D Sound for Virtual Reality Using Ambisonic Techniques" (paper presented at the 3rd Annual Conference on Virtual Reality, Westport, April, 1993), http://www.york.ac.uk/inst/mustech/3d_audio/vr93papr.htm. .

These harmonics are visualized in Figure 1 below:

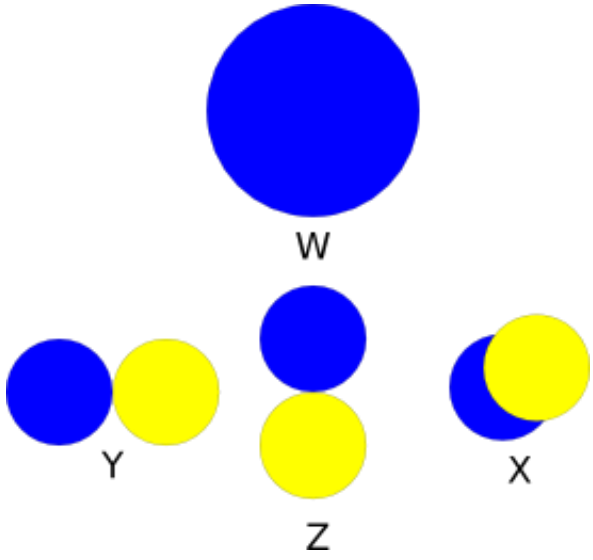


Figure 1: Ambisonic B-format spherical harmonics

The harmonics are reminiscent of the directional response patterns of an omnidirectional microphone and three figure-of-eight microphones. Therefore, with four microphone capsules with the correct directional response patterns placed as closely together as possible and facing the correct directions (the Soundfield microphone, for example), it is possible to record in B-format. It is also possible to encode a monophonic audio stream (i) into B-format, with the sound emanating from a particular azimuth and elevation on the unit sphere, using the following formulas:[45]

$$W = i \times 0.707$$
$$X = i \times \cos\theta \times \cos\varphi$$
$$Y = i \times \sin\theta \times \cos\varphi$$
$$Z = i \times \sin\varphi$$

[45] Malham and Myatt, "3-D Sound Spatialization Using Ambisonic Techniques," 62.

Decoding the B-format to an array of equidistant speakers in a circle, square, sphere, or cube is a somewhat simple and well-known procedure. To decode to an 8-speaker cube, for example, the formulas would be as follows:[46]

$$LFU = W + 0.707(X + Y + Z)$$
$$RFU = W + 0.707(X - Y + Z)$$
$$LBU = W + 0.707(-X + Y + Z)$$
$$RBU = W + 0.707(-X - Y + Z)$$
$$LFD = W + 0.707(X + Y - Z)$$
$$RFD = W + 0.707(X - Y - Z)$$
$$LBD = W + 0.707(-X + Y - Z)$$
$$RBD = W + 0.707(-X - Y - Z)$$

Decoding to irregular arrays, however is more problematic and does not produce optimal results. Nonetheless, it is possible and extremely important to have this functionality available. This functionality is implemented in some ambisonic implementations, including the HOA library of ambisonic tools for Max and other platforms, which solves the problem through a combination of ambisonic decoding and standard panning.[47]

Additionally, it is possible to achieve a much higher spatial resolution through the use of "higher-order" ambisonics. The W, X, Y, and Z harmonics described above are the minimum necessary to achieve 3D ambisonics. However, one may increase the number of spherical harmonics used in the encoding of the sounds. Spherical harmonics up to the third order were described in Gerzon's original 1973 paper,[48] and higher-order ambisonic systems first became practical and began to be explored more fully in the late 1990s, particularly by Malham and other

[46] Ibid., 64.

[47] Anne Sèdes, Pierre Guillot, and Eliott Paris, "The Hoa Library, Review and Prospects," in *Music Technology Meets Philosophy: From Digital Echos to Virtual Ethos*, Proceedings of the International Computer Music Association (San Francisco: International Computer Music Association, 2014), 856.

[48] Gerzon, "Periphony: With-Height Sound Reproduction," 10.

researchers at the University of York.[49] The following formulas describe encoding in the second order of spherical harmonics, conventionally labelled R, S, T, U, and V:[50]

$$R = i \times \sin 2\theta$$
$$S = i \times \cos\theta \times \cos 2\varphi$$
$$T = i \times \sin\theta \times \cos 2\varphi$$
$$U = i \times \cos 2\theta - \cos 2\theta \times \sin 2\varphi$$
$$V = i \times \sin 2\theta - \sin 2\theta \times \sin 2\varphi$$

In recent years, several software packages have been developed to allow for the encoding and decoding of higher-order ambisonics. The aforementioned HOA library, developed by the Centre for research in Computer Science and Musical Creation of the University Paris 8 (CICM), is one such software package.[51] The present implementation of the spatialization system described in this study uses the HOA library to achieve the aforementioned second-order 3D ambisonic encoding and decoding.

[49] D.G. Malham, "Higher Order Ambisonic Systems for the Spatialisation of Sound," in *Proceedings of the 1999 International Computer Music Conference* (San Francisco: International Computer Music Association, 1999), 484-7.

[50] Ibid., 485.

[51] http://www.mshparisnord.fr/hoalibrary/en/

CHAPTER 4: THE MULTI-SOURCE AMBISONIC SPATIALIZATION INTERFACE (MASI)

The Multi-source Ambisonic Spatialization Interface (MASI) was created to connect the spatialization techniques discussed in the previous chapter with graphical interfaces for control, usually in the form of 3D virtual worlds. MASI is an essential component of this research, and serves to establish a workflow for connecting sounds with objects in 3D space and spatializing them on an arbitrary speaker array.

MASI is written for the visual programming language Max.[52] It is primarily a series of "abstractions" written in Max code. The abstractions provide the means for users to load their own Max patches that produce sound and then connect custom graphical interfaces to control the spatialization of the sounds produced. MASI does not provide a graphical panning interface itself, but instead connects to other user-created graphical interfaces through Open Sound Control (OSC)[53] communication.

MASI is primarily intended to be used in conjunction with 3D game-like virtual world environments and interfaces. The primary interface that has been used to control sounds in MASI is the Unity game engine. [54] C# scripts are provided in the MASI package that enable control over the positioning of sounds using a first-person perspective Unity "camera" (which can be controlled by a VR head-mounted display) and "game objects" representing sound sources. Using these scripts, it is trivial to connect a first-person virtual world created in Unity to MASI.

[52] https://cycling74.com/

[53] http://opensoundcontrol.org/introduction-osc

[54] https://unity3d.com/

4.1 Prior Work

There are a few past and ongoing software projects that are influential to the development of MASI. The creators of some of these projects were mentioned previously in Chapter 2.

For example, the technology used by Jaron Lanier for *The Sound of One Hand* bears resemblance to the technology used in the creation of the MASI. Lanier used a visual programming language called Body Electric (later evolving into Bounce[55]) to program the visuals and music, which is very similar in nature to composing for virtual reality using Max.[56] Also, Lanier's company VPL Research had previously developed a device known as the AudioSphere, which was essentially a binaural sound spatializer.[57]

Another example is the Cosm toolkit for Max by researchers at the University of California Santa Barbara, created for use in the aforementioned AlloSphere. The creators state:

> The primary motivation for the *Cosm* toolkit is the creation of immersive, explorable three-dimensional worlds as a unified form of composition. It intimately relates the spatialization of electronic music with virtual environments and computational simulations, and has been used for CAVE™-like systems, audio-visual performance, generative artworks, non-real time composition, speculative research and the prototyping of physical installations.[58]

There are many similarities between Cosm and MASI, and the goals of both are certainly similar. MASI, however, is simpler and more open, providing a simplified GUI for users to load sounds and no attempt at direct connection with Jitter, Max's built-in graphics engine. Whereas

---

[55] http://www.art.net/~hopkins/Don/lang/bounce /bounce.html

[56] Lanier, "The Sound of One Hand," 33.

[57] Virtual Reality Society, "Vpl Research," accessed November 15, 2015 http://www.vrs.org.uk/virtual-reality-profiles/vpl-research.html.

[58] Graham Wakefield and Wesley Smith, "Cosm: A Toolkit for Composing Immersive Audio-Visual Worlds of Agency and Autonomy," in *Innovation, Interaction, Imagination,* Proceedings of the International Computer Music Association (San Francisco: International Computer Music Association, 2011), 13.

Cosm is focused on the possibilities of generative worlds, MASI is designed for stronger ties with more user-friendly video game development systems. Nevertheless, the design of the Cosm toolkit was an influence in the development of MASI for several reasons.

Firstly, the Cosm toolkit is influential because the authors were particularly interested in viewing the 3D landscape as a world that a user would navigate through. The toolkit enables a master user position, adjusting the position and distance characteristics of sounds according to where the user is within the virtual space. This is controlled by means of a star-network topology, utilizing a master node that sends control messages to other nodes responsible for various operations.[59] The authors describe this flow of control signals, which is reminiscent of the methods used in MASI:

> The navigating observer is represented as a mobile oriented point (or local coordinate frame) constructed from a position-vector and orientation-quaternion, accessible via the *cosm.master* object. Both audio and graphical rendering is calculated relative to this coordinate frame…
>
> Data-flows are made spatially local through the *cosm.nav* object, which uses the same coordinate frame implementation and interface as *cosm.master*. However the output of a *cosm.nav* object can be directly attached to any 3D graphical object to co-locate them in the world, and/or to spatialize any MSP audio signal via the *cosm.audio~* object.[60]

As will be described further in the chapter, the MASI design also allows for the specification of the position and orientation of the user by sending messages to the MASI master receiver, similar to the described *cosm.master* object. Each individual sound can also be placed in a specific position within the virtual environment by sending messages to the *masi.encoder~* objects, similar to using the *cosm.nav* objects.

---

[59] John Thompson et al., "The Allobrain: An Interactive, Stereographic, 3d Audio, Immersive Virtual World," *International Journal of Human-Computer Studies* 67 (2009): 938-9.

[60] Wakefield and Smith, "Cosm: A Toolkit for Composing Immersive Audio-Visual Worlds of Agency and Autonomy," in *Innovation, Interaction, Imagination*, 14.

Additionally, the Cosm toolkit is influential in its use of ambisonics as a means of spatialization within a virtual world environment and its automatic incorporation of the previously described distance cues. Similarly to MASI, Cosm utilizes ambisonics to encode angular location and reproduce the sound field on any speaker configuration, and the creators have "…extended [their] implementation to incorporate multiple distance cues for point sources using standard techniques (amplitude attenuation, medium absorption/near-field filtering, Doppler shift and reverberation)."[61]

The Cosm toolkit works to solve many problems associated with composing virtual worlds. However, the toolkit has not been actively maintained publicly for the last several years, and the last release was for Max version 5 (as of the time of writing, the current version is Max 7) in 2011.[62] In response, some Max users have crafted their own methods for moving the position of the user within an ambisonic environment.[63] Additionally, none of the readily available ambisonic spatialization extensions for Max incorporate distance cues for point sources as Cosm did. MASI works to provide a solution to both of these problems. However, as will be discussed further in the chapter, the goals for MASI are different from those of the creators of Cosm, as MASI focuses more on usability and simple connection with game creation software and other graphical interfaces rather than an all-inclusive compositional package that necessarily requires advanced skills with Max and Jitter.

[61] Thompson et al., "The Allobrain: An Interactive, Stereographic, 3d Audio, Immersive Virtual World," 939.

[62] http://www.allosphere.ucsb.edu/cosm/download.html

[63] For example, see https://cycling74.com/toolbox/soundstroll/

Another similar original system for the spatialization of sound within virtual reality environments is the Virtual Audio Reproduction Engine for Spatial Environments (VARESE), which was developed by Thomas Musil et al. for use in the aforementioned virtual reality reproduction of Edgard Varèse's *Poème électronique*.[64] The developers of this project utilized a combination of ambisonic and HRTF binaural spatialization techniques by decoding the ambisonic sound field onto a virtual speaker array which was then re-encoded in binaural, a process they called "virtual ambisonics."[65] Although it appears that the developers originally experimented with large multi-channel systems, they eventually settled on the solitary headphone experience due to the requirements of the project.[66]

VARESE is, similar to Cosm and MASI, a system for connecting 3D visual sources with point-source sounds in a virtual reality environment. It is written for Pure Data (Pd),[67] which is an environment very similar to Max. However, like Cosm, it is also out-of-date and does not have readily available public documentation. A web search reveals only a link to a seemingly early version of the software with no documentation.[68] Nevertheless, VARESE is a noteworthy contribution since: 1) it is designed explicitly with the use of a head-mounted display in mind, and 2) it also incorporates angular location and audio distance cues in one unified system with a graphical user interface.

---

[64] Thomas Musil et al., "Virtual Audio Reproduction Engine for Spatial Environments," in *Free Sound*, Proceedings of the International Computer Music Association (San Francisco: International Computer Music Association, 2005), 790.

[65] Ibid., 791.

[66] Ibid.

[67] https://puredata.info/

[68] http://old.iem.at/projekte/newmedia/VARESE/

One of the most important goals in the original development of MASI was to connect the system with virtual reality head-mounted displays such as the Oculus Rift.[69] The company behind the Oculus Rift has itself developed an audio SDK to deal with audio in an immersive environment on headphones, implementing many of the same methods described in the previous chapter. However, this development kit does not address the possibilities of spatialization on multi-channel audio systems (see the previous chapter), which is the primary focus of MASI (although MASI is also capable of binaural rendering by virtue of the methods for doing so provided in the HOA library).

Despite the headphone-only limitation, Oculus VR has provided its users with a variety of useful binaural spatialization tools intended to be intimately connected with virtual environments. For example, they have developed the Oculus Native Spatializer Plugin (ONSP) for Unity. This allows users to attach audio sources to game objects which are then spatialized according to the aforementioned HRTF methods.[70] Additionally, Oculus VR has created HRTF spatialization plugins for the third-party game audio development systems FMOD and Wwise, as well as VST and AAX plugins for use with Digital Audio Workstations.[71] In all, Oculus VR has developed a more complete solution for point-source HRTF spatialization than has been seen previously. The Oculus VR Audio SDK provides an overall excellent solution for developers who are only concerned with audio playback (the SDK does not provide any methods for audio synthesis, effects, interactivity, etc.) and stereo (and particularly headphone) spatialization.

---

[69] https://www.oculus.com/en-us/

[70] Oculus VR, "Introduction to Virtual Reality Audio," 27-34.

[71] Ibid.

Another recent influential body of work in the field of game engine interaction with music and sound is that of Robert Hamilton (whose creative work was discussed in Chapter 2). Sound spatialization is part of Hamilton's work, but his primary technical contribution has been the modification of game engines (Quake 3 and Unreal) to accommodate OSC communication. Hamilton uses these modified game engines to create virtual worlds and sonify various components of the virtual worlds.

An earlier piece of software developed by Hamilton is q3osc, which is:

> …a heavily modified version of the open-sourced ioquake3 gaming engine featuring an integrated Oscpack implementation of Open Sound Control for bi-directional communication between a game server and one or more external audio servers. By combining ioquake3's internal physics engine and robust multiplayer network code with a full-featured OSC packet manipulation library, the virtual actions and motions of game clients and previously one-dimensional in-game weapon projectiles can be repurposed as independent and behavior-driven OSC emitting sound-objects for real-time networked performance and spatialization within a multi-channel audio environment.[72]

Hamilton's software, therefore, is focused on the game engine end of the signal chain. Rather than focus on an audio server that is adaptable to many graphical interfaces, Hamilton has created a means to use a specific graphical engine that is adaptable to any audio server through OSC communication (opposite of the MASI implementation). Hamilton has continued this work for several years, and has since moved to using the Unreal Development Kit (UDK) instead of the older Quake 3 engine. His current software, introduced in 2011, is called UDKOSC, and is similar in nature to q3osc as it adds an OSC communication layer on top of the UDK.[73]

---

[72] Robert Hamilton, "Q3osc Or: How I Learned to Stop Worrying and Love the Game," in *Roots/Routes*, Proceedings of the International Computer Music Association (San Francisco: International Computer Music Association, 2008), 217.

[73] Robert Hamilton, "Udkosc: An Immersive Musical Environment," in *Innovation, Interaction, Imagination*, Proceedings of the International Computer Music Association (San Francisco: International Computer Music Association, 2011), 717-20.

Therefore, while there are some solutions available for the spatialization of sounds within virtual reality environments—and for the integrated composition of works for virtual worlds—it was not possible to find a full and fully-functioning solution for the type of virtual composition undertaken in this research. This search for the proper tool confirmed the need for a new system for virtual reality-based spatial audio composition and provided the impetus for the creation of the MASI system.

4.2 Distance Cue Implementation

One crucial aspect for realistic point-source spatialization in virtual world environments is that the distance cues described by Chowning and others are implemented in an efficient, effective, and convincing manner. To reiterate, the basic acoustic cues relied upon for perception of distance from a source are as follows:

1) The intensity (loudness or volume) of the sound is attenuated in proportion to the distance between the listener and sound source.

2) The delay between sound beginning to emanate from the source and reaching the listener's ear is increased with distance. Fast changes in this delay time caused by either a moving listener or sound source result in a change in pitch known as the Doppler Effect.

3) High-frequency components of a sound (particularly those above 1 kHz) are dampened with greater distance between the listener and the sound source.

4) The relative balance between reflected sound (reverb) and direct sound changes according to distance. The listener perceives more reverberated sound relative to direct sound when the source is further away, and less reverberated sound when the source is closer.

The part of the MASI package that handles the generation of these cues is the *masi.encoder~* abstraction. The signal flow of the distance cue component of this Max patch is

visualized in Figure 2 below:

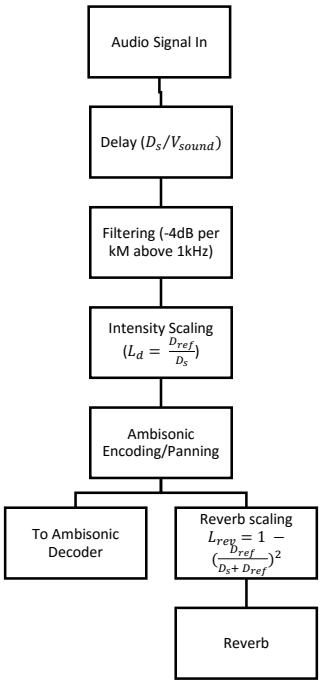


Figure 2: *masi.encoder~* signal flow

As sound enters the *masi.encoder~* patch, it first passes through a delay that is varied based on the first-person user's distance from the sound source.[74] This component is perhaps the most difficult distance cue to implement in software, primarily because delay can be computationally expensive. Implementing a delay in software requires a continuous memory, or "buffer," of sound known as a delay line. Therefore, the system must keep a specified number of digital audio samples in memory. In MASI this number of samples is capped at the amount needed to delay the sound as much as it would be delayed if it were physically emanating from 50 meters away from the listener, or approximately the number of samples needed to represent

[74] A note on distance: it should be noted that the user's actual distance from the source is added to a reference distance, which is the distance at which the sound is perceived to be at full volume. This prevents distances of zero—or any distance below the reference distance—and therefore prevents clipping or extreme loudness.

147 milliseconds of sound, since, if the speed of sound is presumed to be an average 340 meters per second, $50/340 \approx 0.147\ seconds\ = 147\ milliseconds$. In order to retrieve the delayed sound, one needs to "tap in" to this delay line. Since the user and/or source may be constantly moving, however, the tap in point may constantly change (at a rate likely determined by the framerate at which the graphical component is running).

A moving delay time causes an effect that is known in musical synthesis as "modulated delay" and produces a rising and falling of pitch. When implemented subtly, this is the Doppler Effect. However, sudden jumps in delay time (rather than infinitely smooth ramps, as in the physical world) can produce an unpleasant "zipper" noise. Therefore, a pleasant-sounding continuously moving delay line can be difficult to implement. In Max, this is achieved this using the audio-rate *delay~* object (which allows for the delay time to be modulated based on an audio signal) and the *rampsmooth~* object (which takes in a value and ramps to that value at audio rate over a specified number of samples). Each time a new delay value is received, the delay time does not jump to that value instantaneously but instead ramps to the value over 50 milliseconds. A 50 millisecond ramp time was used because this is the minimum time that is needed to resolve the zipper noise. Therefore, a slight latency is introduced in the perceived Doppler shift, which is physically slightly inaccurate but is a necessary compromise.

After delay the sound passes into a filter to imitate air absorption of high frequencies. The filter is, as specified in Naef et al., a bi-quad filter with high-shelving characteristics.[75] The high-shelving filter reduces all frequencies above 1 kHz by approximately 4 dB per kilometer.

[75] Naef, Staadt, and Gross, "Spatialized Audio Rendering for Immersive Virtual Environments," in *Proceedings of the Acm Symposium: Virtual Reality Software & Technology*, 68.

This is achieved by setting the corner frequency of the filter to 1 kHz and the gain of the filter according to the formula $distance\ (m) \times -0.004 = \ filter\ gain\ (dB)$.

After passing through the filter, the overall level of the sound is scaled according to distance. In some literature the formula for distance level scaling is described simply as $1/distance$. Chowning, for example, states "The amplitude of the direct signal is proportional to 1/distance. As an example, assume the distance from the listener to the point midway between two loudspeakers to be L… we wish to simulate a source at a distance of $2L$. The amplitude of the direct signal would be attenuated by $1/2$."[76] Thinking in virtual space, L must be some kind of reference distance, and therefore the formula implemented is $D_{refernce}/D_{source}$. In MASI, the reference distance is configurable by the user. A larger reference distance is perceived as an overall volume increase, since the sounds would be at their full volume at a larger distance away from the listener. Similar to delay time, sudden changes in volume can cause click and zipper noises, and a 10 millisecond ramp is used to alleviate this problem.

After level scaling, the sound passes into the ambisonic encoder and is panned to its proper angular position (this process will be covered in detail in the next section). This encoded signal is split and sent both directly to the ambisonic decoder and to a global reverb. The role of reverb in spatial music is challenging, and there are different more- and less-complex solutions for achieving a desirable result.

Chowning used a more complex method utilizing localized as well as globalized reverb. This is because, according to Chowning, "…if the reverberant signal were to be distributed equally to all channels for all apparent distances of the direct signal, at distances beyond the echo

[76] Chowning, "The Simulation of Moving Sound Sources," 3.

radius the reverberation would tend to mask the direct signal and eliminate the cue for angular location."[77] Thus, the danger in using reverb is that angular precision will be masked. Chowning solves this perceptual problem by sending the point source signal to a global reverb (a reverb distributed equally to all channels) at a level of $(1/distance) \times (1/\sqrt{distance})$ and to a local reverb (a reverb that is panned to the same location as the sound source) at a level of $(1 - 1/distance) \times (1/\sqrt{distance})$. Therefore, as distance from the sound source increases the reverb becomes more localized.[78]

In MASI, a simpler method as described by Naef et al. is used. After all previous processing (delay, air absorption filtering, level scaling, and ambisonic panning) each source signal is sent to a single global reverb bus. The reverb unit used is an ambisonic implementation of the "Freeverb" algorithmic reverb[79] provided as part of the HOA library. The signal (already encoded into ambisonic spherical harmonic format) is scaled according to the formula described by Naef $L_{rev} = 1 - (\frac{D_{ref}}{D_s + D_{ref}})^2$ before being sent to the global reverb.[80] This process is equivalent to adjusting a post-fader auxiliary send level on a mixer. The perceptual effect is that sound sources closer to the listener contribute proportionally less to the overall reverb while sound sources further away from the listener contribute proportionally more. The issue described by Chowning of a loss of angular information when a sound is far away has thus far not presented as a significant concern.

---

[77] Ibid.

[78] Ibid.

[79] For an explanation of the Freeverb algorithm, see https://ccrma.stanford.edu/~jos/pasp/Freeverb.html.

[80] Naef, Staadt, and Gross, "Spatialized Audio Rendering for Immersive Virtual Environments," in *Proceedings of the Acm Symposium: Virtual Reality Software & Technology*, 69.

4.3 Coordinate Systems and Ambisonic Panning

Beyond the aforementioned considerations involved in simulating distance cues, a further challenge in the development of MASI is the interaction between 3D worlds, head-mounted displays with head tracking, and the positioning of sounds on the ambisonic unit-sphere. The desired interaction is that the sounds appear to emanate from the correct angular locations dependent on where the sound source is within the virtual world, where the first-person user is within the virtual world, and the position of the first-person user's head. The most important considerations in developing this functionality are:

1) Sound source positions must be (re)calculated based on user location.

2) Sound source positions must be (re)calculated based on user head rotation and/or the entire ambisonic field must be altered based on head rotation.

3) The standard 3D Cartesian coordinate systems used in game engines and development environments such as Unity must be converted to the spherical coordinates that are standard in ambisonics.

The last of these considerations—the conversion between 3D Cartesian coordinates and spherical coordinates—would seem to be a simple prospect. The formulas are well-known and established. However, the process becomes more complex and convoluted (and is therefore notable) when considering the different coordinate system standards used in the fields of computer graphics and spatial audio. MASI uses a "left-handed" Cartesian coordinate system due to its compatibility with Unity and relative familiarity in the 3D graphics field. At this stage MASI is programmed only for this coordinate system, however it would certainly be possible (and not difficult) to incorporate other coordinate systems as well.

The “left-handed” coordinate system is demonstrated in Figure 3 below:

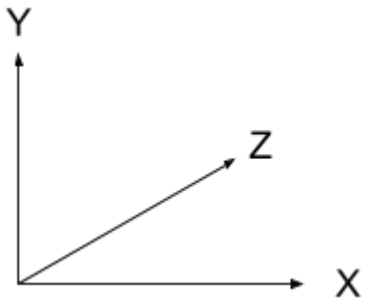


Figure 3: Left-handed coordinate system

In this coordinate system, X is the left-right axis with positive to the right, Y is the up-down axis with positive up, and Z is the front-back axis with positive front. The coordinate (0, 0, 0) is considered to be in the center of the 3D world. The most common alternatives to this coordinate system would be “right-handed”—in which positive Z is facing in the opposite direction—and the system more common in 2D computer graphics in which (0, 0, 0) is in the upper-left corner of the drawing area and positive Y is down. The left-handed system seems to be the most common in 3D video game design and therefore provides maximum potential compatibility with 3D world design tools.

In MASI, the *masi.encoder~* abstraction is responsible for the three positional calculation tasks described above (in addition to the previously described distance cue calculations). The user provides the patch with a user position and a sound source position, either of which can be updated in real-time at any point. The math required to complete the first task—calculating sound source position based on user location—is the simplest of the three. It is important to remember that in fact the user is never really *moving*, rather the scenery is moving around the user. This is akin to filming a driving scene in a film, in which the car is stationary in front of a green screen and images of moving scenery are added later. Therefore, the user position is

simply a conceptual consideration, really only serving as a value by which to offset the sound source position. This is done through simple subtraction:

$$sound\ source\ position - user\ position = panning\ position$$

For example, if the user is at position (0, 0, 0) and the object is at position (-2, 1, 3) then the sound source would seem to emanate from position (-2, 1, 3). However, if the user moves to position (2, 0, -1) then the sound source should seem to emanate from position (-4, 1, 4).

The next step is the process of panning sound sources to their correct perceptual locations is to convert the XYZ Cartesian coordinates to the spherical azimuth, elevation, and distance (θ, ϕ, and ϱ) coordinates required to specify the location to the ambisonic encoder and to the distance cue processes. In order to simplify this process, the first step that is implemented in MASI is to convert the left-handed coordinate system to standard Cartesian coordinates, with X as the front-back axis (positive back), Y as the left-right axis, and Z as the up-down axis. This conversion is performed in Max code as shown in Figure 4 below:

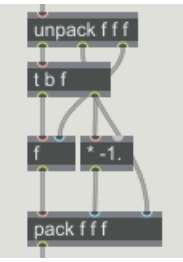


Figure 4: Left-handed coordinate conversion in Max

From this point, the formulas used to convert to spherical coordinates that can be sent to the ambisonic encoder are well-known:

$$azimuth(\theta) = \tan^{-1}(\frac{y}{x})$$
$$elevation(\varphi) = \cos^{-1}(\frac{z}{\sqrt{x^2 + y^2 + z^2}}) = \cos^{-1}(\frac{z}{\varrho})$$
$$distance(\varrho) = \sqrt{x^2 + y^2 + z^2}$$

At this stage the position of the head—as measured by the sensors on the head-mounted display—is taken into consideration. If the head is considered as a sphere, it can be rotated around any of its three axes. These rotations are commonly known as yaw, pitch, and roll. These rotations in the context of the left-handed coordinate system are shown in Figure 5 below:

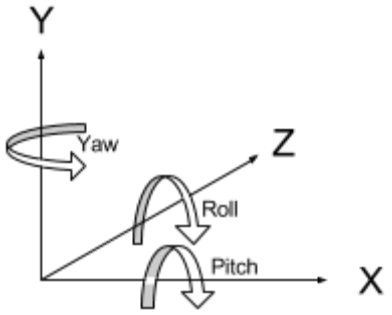


Figure 5: Cartesian rotation

Rotation around the vertical axis is yaw, rotation around the horizontal axis is pitch, and rotation around the front-back axis is roll.

It is also possible to rotate a first-order ambisonic sound field in this manner.[81] This process has been described in detail in the literature.[82] However, it has been noted that the pitch and roll ("tilt" and "tumble" in ambisonics) rotation matrices starting with the third-order and above are not trivial to generate. Solutions for these transformations are not widely known or implemented.[83] The HOA library that is used for ambisonic encoding and decoding in MASI does not provide these algorithms for 3D ambisonics (although it is possible in 2D, ignoring the height component). Therefore, rather than adjust the yaw, pitch, and roll of the entire ambisonic

[81] Note that in ambisonics, the terms yaw, pitch, and roll are conventionally replaced with the terms rotation, tilt, and tumble respectively.

[82] For example, see Malham and Myatt, "3-D Sound Spatialization Using Ambisonic Techniques."

[83] David Malham, *Higher Order Ambisonic Systems* (University of York, 2003), 5.

sound field according to head position, the positions of the individual sound sources within the unit-sphere are adjusted instead.

The greatest advantage of this method is that it is simple, adaptable, and accurate. There is no need to generate complex transformation matrices for higher-orders, and therefore it is simple to adapt MASI to operate at the highest order computationally possible. There is also no concern for the validity and accuracy of the transformations. The major disadvantage of this method is that it limits the use of the software to the spatialization of point sources. In other words, pre-generated sound fields and ambisonic sound field recordings cannot be "explored" using MASI. This is not an issue for most users since the most common intended use of the software is in conjunction with game-like and totally synthetic environments. However, a possible next step in the development of the software could include greater options for exploring recorded environments, particularly as the availability and effectiveness of 360° video recording technology improves.

Thus, the process that is used for adapting the positions of the point sources to the movements of the head is simple addition. After the azimuth and elevation angles of the sound source are calculated based on XYZ user and sound source position, the yaw and pitch of the head are simply added to the azimuth and elevation. At the time of writing, the Z-axis roll of the head is not incorporated into this process. Personal observation has shown that the effect of this type of rotation is not very striking—particularly in a live environment—and also that the calculation is more complex. Additionally, it is not common for a person to move their head along the front-back axis and is an uncomfortable motion for most.

4.4 UI Design and Workflow

In the design of MASI, the workflow is intended to be as open as possible. In other words, creators working with MASI could develop their work according to their own thought process. A creator could choose to design sounds and then design a graphical method to control these sounds. Or, a creator could design a virtual world and then attach sounds to the objects within that virtual world. Therefore, the user interface for MASI does not suggest any specific flow or metaphor, but is intended to be purely functional. The main MASI window is shown in Figure 6 below:

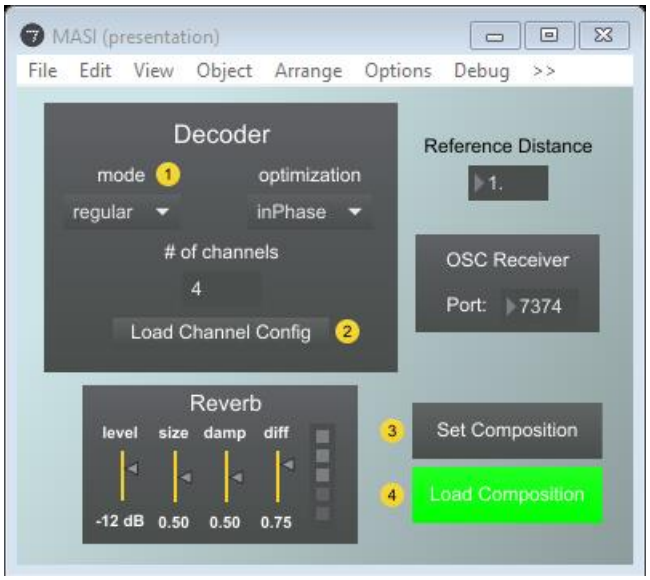


Figure 6: MASI main window UI

The MASI window is itself a Max patch. MASI is distributed as a Max "package," and once installed the main window can be opened through the "Extras" menu. In fact, the MASI window is a collection of Max patches that are embedded with visible UIs through the use of the *bpatcher* object in Max. Besides the main patch, there are four other patches embedded in the main window: a patch containing the ambisonic decoder (*masi.decoder~*), a patch containing the reverb (*masi.reverb~*), and a patch containing the Open Sound Control receiver

(*masi.oscreceiver*). In order to use the MASI interface, a user must be aware of three basic components, which are titled “channel configurations,” “sound sources,” and “compositions.”

A "channel configuration" is a user-supplied list of azimuth/elevation coordinates specifying the speaker setup (this is unnecessary if using binaural rendering). This should be created as a single-line *.txt* text file. For example, a typical 4-channel setup (Lf, Rf, Ls, Rs) would read as follows (in degrees, 0° is front, direction of rotation is counterclockwise): *45 0 315 0 135 0 225 0*. Azimuth is between 0° and 360°, while elevation is between 90° (directly above listener) and -90° (directly below listener).

A "sound source" is a stream of audio in Max. Sound sources are made available to MASI through *outlets* in Max abstractions. A very simple example of a MASI sound source is the patch shown in Figure 7 below, placed somewhere in the Max search path:

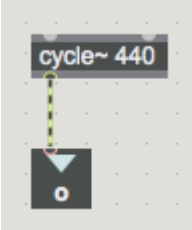


Figure 7: A simple sound source in Max

A "composition" is a user-supplied JSON (JavaScript Object Notation) file that contains key-value pairs denoting an abstraction with one or more sound sources (*outlets*) and a unique name for each source. For example, if the simple patch shown in Figure 7 was saved as *source.maxpat* somewhere in the Max search path, and the single sound source it contained should be called *uniqueName*, then the JSON should read as follows:

```
{
        "source" : "uniqueName"
}
```

It is also possible to have a single abstraction with multiple sources (outlets):

```
{
   "source": ["uniqueName1", "uniqueName2", "uniqueName3"]
}
```

Or multiple abstractions with any combination of single or multiple sources:

```
{
   "source1": "uniqueName",
   "source2": ["uniqueName1", "uniqueName2"]
}
```

Using MASI involves coordinating these three components. An example workflow for a simple experiment might be as follows:

1) Use Unity to create a simple environment such as a room. Add a few objects to the room, such as boxes, or models of a sound-producing object such as a speaker. Note the names of the objects. Add the provided Unity scripts to the objects and camera (to be discussed further).

2) Design a Max patch that plays back a sound for each object. These are sounds that will seem to emanate from the objects in the virtual room. Instead of connecting the sounds directly to the Max audio output, connect them to *outlet* objects (so they will be used as sound sources).

3) Create the JSON composition file listing the name of the Max patch created in the previous step as a key and unique names for each of the outlets (should be the same as the Unity object names) as an array of values.

4) Load the correct speaker configuration and composition files, and begin the Unity game.

The final step in this process—loading the composition file—instantiates the aforementioned *masi.encoder~* objects. When the "Load Composition" button is pressed, JavaScript code embedded within the MASI main patch iterates over the JSON file (loaded in a *dict* object) and calls the user-created abstraction, instantiates *masi.encoder~* objects for each sound source, and connects the sound sources to the *masi.encoder~* objects.

4.5 OSC Communication and Internal Messaging

Due to its open design, the spatialization of sound in MASI is not accomplished through a built-in graphical interface. Instead, MASI is more akin to middleware. Users can use their own graphical interfaces or virtual world engines to make graphic environments, can define their own methods of sound generation using the full capabilities of Max, and then can connect the two via Open Sound Control.

Open Sound Control (OSC) is a communications protocol developed by researchers at the University of California Berkeley. In recent years, it has become a commonly used standard in the field of computer music. It is particularly useful because it uses standard UDP and TCP network protocols. Therefore, it is possible to add OSC functionality to a diverse range of software and hardware. The computer music community has created OSC libraries for most major programming languages and OSC extensions for many popular programs. For this reason, the spatialization of sounds in MASI can be controlled in a variety of ways, including (but not limited to): web-based interfaces, other Max patches, standalone programs written in almost any language, OSC enabled hardware devices, and game engines. A virtual world developer could potentially connect MASI to WebGL, HTML5, a game developed as a standalone in C, Java, Python, etc., or a game developed using the popular 3D game editors Unity and Unreal Engine (users have contributed open source OSC extensions for both). The interface with which MASI has been primarily used is Unity (for which specialized scripts have been developed and included in the MASI package) but the choice of using OSC as a bridge rather than some method of connecting directly to Unity was deliberate. Through OSC, MASI is useful not only for work with Unity, but for the work of others who may wish to use other tools.

The OSC messaging system within MASI is intentionally simple by default, but with some knowledge of OSC can be easily extended. There are a few basic messages that one can send to MASI to invoke the built-in spatialization functionality:

1) The first-person perspective camera's position can be changed with the message */position x y z*

2) The first-person perspective camera's rotation can be changed with the messages */rotation x y z*

3) An individual sound source's position can be changed with message */uniqueName/position x y z*, where *uniqueName* is the unique name assigned to the object in the composition JSON file.

4) The message */uniqueName/enable 1* or */uniqueName/enable 0* sends the 1 or 0 value specified to the object enable receivers (in a process to be discussed further below).

To reiterate, these OSC formatted messages can be sent from any OSC capable device or program to MASI at the port specified by the user.

Within MASI, a system of specially named Max *send* and *receive* (non-local connection) objects handle the distribution of this data to the correct locations within the software and also provide the user with access to the incoming data if he or she desires to use the data for other parameters beyond the spatialization handled by MASI. The first part of this functionality is handled in the *masi.oscreceiver* patch. The patch takes incoming OSC messages and distributes them according to Figure 8.

The first place that *masi.oscreceiver* routes the OSC messages is to a Max *send* named *OSCDump*. All incoming OSC messages, completely unsorted, are passed to this *send*. There is no matching *receive* for this particular *send*. Instead, it is intended that should a user want to access all incoming OSC messages within her own patch, she can simply include the object *receive OSCDump*.

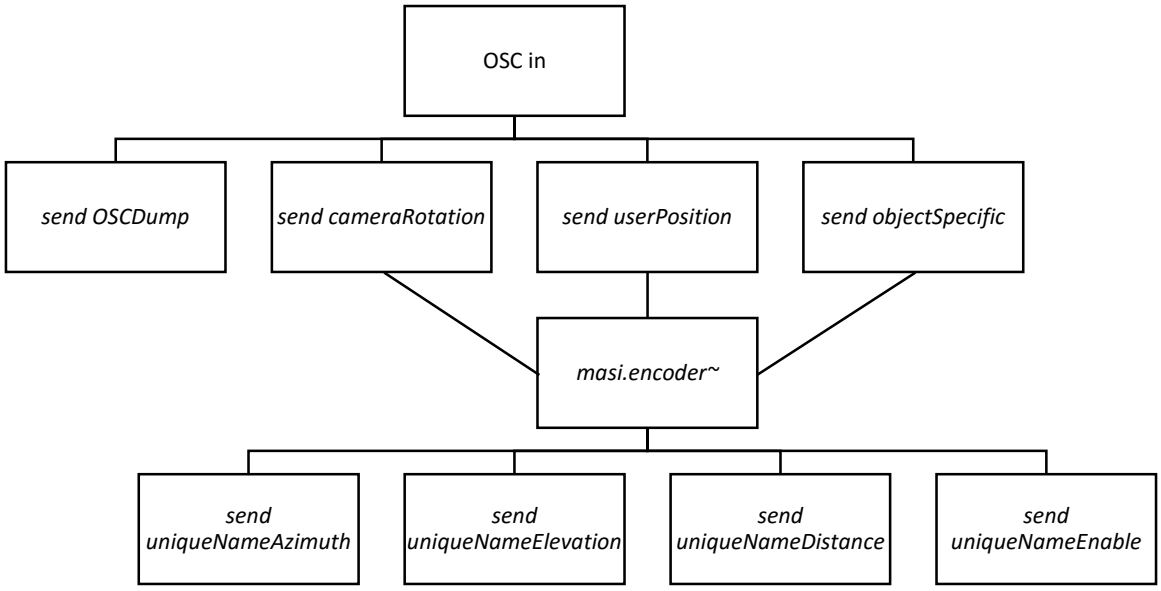


Figure 8: Send distribution in MASI

At this point, the incoming OSC messages are sent to a *route* object that sorts through the messages in order to send them to the correct *receive* objects within *masi.encoder~*. The messages are sent to different places depending on whether they are */position* messages (*send userPosition*), */rotation* messages (*send cameraRotation*), or other messages (which are passed through *send objectSpecific* and sorted in *masi.encoder~*).

The receives *userPosition* and *cameraRotation* are used within *masi.encoder~* to calculate the angles for ambisonic position and distance for distance cue encoding as previously discussed. The *objectSpecific* receive, however, is passed through additional routing. In a MASI composition there are likely many difference sound sources, and therefore many different instances of *masi.encoder~*. Each instance of *masi.encoder~* must receive the user position and camera rotation, but the "object specific" messages should only be interpreted by the correct instance. Further routing within *masi.encoder~* ensures that only the correct object position

messages are being used (so long as the unique names of objects are properly associated with their respective outlets in the composition JSON).

Further text processing within *masi.encoder~* creates more data points that users can access within their own patches. As stated earlier, more advanced users may wish to use the *receive OSCDump* object to do their own OSC routings. However, *masi.encoder~* automatically incorporates some basic functions that a user may want to explore. First, when the */uniqueName/enable* message is sent from the OSC interface, the 1 or 0 received is passed to a send called *uniqueNameEnable*. Then, if a user adds the object *receive uniqueNameEnable* to her Max patch, that *receive* will be passed the 1 or 0 received through OSC. For example, in the Unity scripts included with MASI, the message */uniqueName/enable 1* is sent for all game objects with the script attached as soon as the object becomes active. This is useful because the *uniqueNameEnable* receive can be used to start audio for an object as soon as it enters the scene, eliminating the need for separate controls to instantiate a graphical game object and start audio.

Additional functionality is added through the use of sends for each component of the spherical coordinates used in the ambisonic and distance encoding of the sounds. When *masi.encoder~* updates the spherical coordinates describing a sound source's position in response to the movements of either the source or the first-person user, it passes this information on to sends named *uniqueNameAzimuth*, *uniqueNameElevation*, and *uniqueNameDistance*. The user can use the corresponding *receive* objects in her Max patch in order to gather information about the object's location. As an example, perhaps a user wants a sound to play faster when the first-person user is closer to the sound and slower when further away. This could be achieved by simply inserting the object *receive uniqueNameDistance*, scaling and offsetting the incoming

value appropriately, and passing it to a playback speed control developed either by the user or through built-in functionality within Max (using the *playlist~* object, for example).

4.6 Using the Unity Game Engine

MASI is designed not to be tied to any particular game engine or graphical component. The only requirements of a graphical control interface for MASI is that it is able to send custom OSC messages. As the name suggests, the Multi-source Ambisonic Spatialization Interface is software designed solely for the manipulation of audio. However, the envisioned primary usage of MASI is along with software that is used to build navigable 3D virtual worlds and environments. Among the easiest options for building virtual worlds are game creation software packages such as Unity and the Unreal Development Kit. Unity is a popular choice, is exceptionally well-documented, has a full-featured free version, is extensible, and is commonly used in classroom and educational environments. It also has built-in support for popular VR head-mounted displays such as the Oculus Rift and a few available OSC implementations. Therefore, it is a good choice for the task of building a navigable virtual world with minimal setup effort.

In Unity, an empty virtual space is called a "Scene." Within that Scene are multiple "Game Objects"; these include cameras, models, lights, etc. Users can attach scripts to Game Objects to control the object's behavior in the Scene. The Unity scripting engine is robust, and provides simplified control over many characteristics of the Game Object. Scripts can be written in C#, JavaScript, or Boo. Using the scripting features of Unity, a few users have created implementations of OSC. Of these implementations, Jorge Garcia's UnityOSC[84] is used for the

[84] https://github.com/jorgegarcia/UnityOSC

scripts, since it is low-level and does not assume a specific usage case (it is open and adaptable to a variety of situations).

Once UnityOSC is installed in the Unity project as described in the UnityOSC documentation, the C# Unity scripts provided in the MASI package can be used. There are two scripts provided with MASI: *CameraOSC.cs* and *ObjectOSC.cs*. These two scripts can be attached to the first-person camera and to other scenery objects respectively.

The first step in creating a navigable virtual world in Unity is to create a first-person camera. The desired setup is that of a "first-person shooter," or FPS, since the most common instance of this kind of control and view setup in gaming is in the context of such a game. Unity has an included a "prefab" (a combination of several game objects with set parameters) called *FPSController* within the Unity Standard Assets that makes this setup easy for any user. Using this prefab sets up a camera, a virtual body-like object to represent the player, and "mouse look" style control. Additionally, if using a VR head-mounted display, simply enabling VR support within the Unity project in combination with the *FPSController* prefab produces the desired result.

Once this task is accomplished, adding the *CameraOSC.cs* script to the *FirstPersonCharacter* prefab (which is a child component of *FPSController*) will begin sending the */position* and */rotation* OSC messages to the specified OSC client (the OSC client is setup in the installation of UnityOSC), which is, in this case, MASI. Then, the user may add other objects to the scene and attach the *ObjectOSC.cs* script to each of those objects. This script will send the */uniqueName/enable* OSC message to the client specified in the camera script when the object is enabled and will send the */uniqueName/position* OSC message whenever the object moves (the unique name can either be specified as an attribute that will show in the object editor

window or will automatically be set to the name of the object in Unity if no alternate name is supplied). Additionally, the *ObjectOSC* script sends collision information when using the Unity physics capabilities. When an object with the *ObejectOSC* script attached collides with another object, it sends the name of the object it collided with along with the absolute magnitude of the collision in the message */uniqueName/collision objectCollidedWith collisionMagnitude*. This entire process is visualized in Figure 9 below:

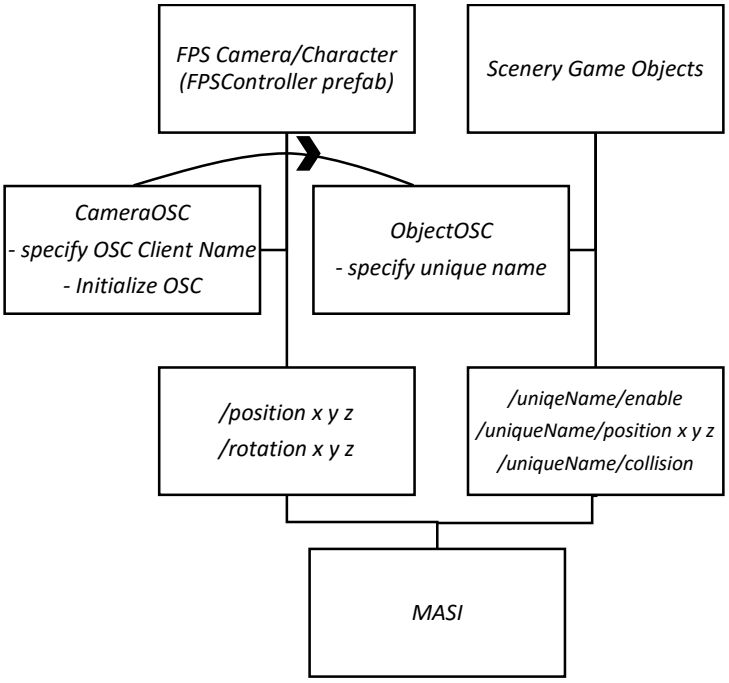


Figure 9: OSC messages in Unity

The primary goal in creating the scripts to connect MASI and Unity is to provide users with a "drag-and-drop" visual composition solution. With some basic practice with Unity (by going through some of the excellent beginner tutorials[85]) it is possible to create a simple project very quickly. For composers, this is essential. MASI is meant for composers seeking ways to simplify and expedite the process of 3D virtual world composition with interactive spatial audio.

[85] https://unity3d.com/learn/tutorials

Prior to MASI and the included Unity scripts, the process of doing this was completely experimental and not straightforward. With MASI, a compositional flow has been established.

Additionally, the Unity scripts can be very educationally useful. Students with little or no prior experience in coding can actually use MASI and Unity to create virtual worlds with spatial audio relatively quickly. This software can be incorporated into classrooms and/or workshops, and by using MASI and the Unity scripts it would be possible for a young student with no prior experience to create a basic virtual world with spatial audio in just a few hours.

CHAPTER 5: COMPOSITIONAL APPROACHES FOR VIRTUAL SPACE: FORM AND THE MUSIC OF EARLE BROWN

Thus far, this study has dealt exclusively with the historical context and technical considerations involved in using virtual reality as a live music performance medium. However, one of the most crucial aspects of the success of this endeavor is the quality of the content presented using the new medium and the relationship between the content and the virtual reality medium. Therefore, when considering a musical performance in virtual visual space, it is important to consider environment not only as a means for creating a more novel or immersive experience, but also as an essential component of the musical form. Among the ways for incorporating a virtual visual space into musical performance, two conceptual approaches are defined:

1) A "static" virtual space, in which the space does not change but is instead explored and interpreted by the performer, and
2) a "dynamic" virtual space, in which movement of the space or objects within the space directly influences or controls the performance of the music.

5.1 Static Approaches

Poème Électronique can be considered a static approach to virtual world creation. Sounds and images move around the space, but neither the space nor the objects within the space are moving or changing. The space affects the compositional methodology and the perception of the sounds and images, but the space itself does not serve to create or compose the content. Instead, the space adds a navigable component to pre-composed content.

Alternatively, the environment may serve as a visual cue for performance or improvisation. Examples of this can be found in the works of Earle Brown. In his work

December 1952, Brown used an algorithmic process to create a series of lines on a page which would serve as a score. This work can be seen as a type of navigable composition. Rather than compose linearly, Brown chose to compose spatially. Brown states:

> …on *December*'s original score there's a scale of numbers on the left margin and a scale on the bottom of the page, forming in geometry an abscissa and an ordinate. And then, working out a program which would allow the piece to generate itself as I wished it to, I would find a number on the left vertical scale and a second number on the horizontal scale along the bottom of the page. At the intersection of these two numbers, drawing the left one horizontal to the right, drawing the number on the bottom vertically up—at the intersection of these (which are called indices)—at the center would be a point in this total space. Once I achieved this point, my cybernetic program would then give me a number which would indicate whether from this point a line would move to the right or to the left on a horizontal plane, or up or down on the vertical plane. Once I found this out, I would get the duration—that is, the length of that line, horizontal or vertical. Then another number would give me the thickness of that line.[86]

In this way, Brown develops a compositional method in which the space on the page literally determines the form of the music. Rather than consider the sound specifically, Brown only considered the visual/spatial elements of position, thickness, and direction of lines in composing the work. One reason for this decision is that Brown did not intend the piece to be performed from left-to-right, and therefore, it did not need to be composed in that manner. Instead, the score of can be read in any direction, adding a further formal spatial component to the piece. Brown states:

> It seemed clear to me that a piece that was not going to be performed from left to right did not need to be composed from left or right. In other words, I could not predict the movements of a performer from one point to any other point, and rather than compose it just by taste or some kind of imaginary continuity structure which would then not exist in the performance, I chose to consider the entire area a field of activity and within this field, by this coordinate technique, the various elements were placed.[87]

[86] Earle Brown, "On *December 1952*," *American Music* 26, no. 1 (Spring 2008): 5.

[87] Ibid.

Other works of Brown employ a similar method of "spatial composition," including the octophonic tape work *Octet I* (1953). In this work, Brown used essentially the same method for composition as *December 1952*, using random numbers to determine the physical location of an event on the score, but instead of drawing lines on a page of a fixed length he placed splices of tape along a time continuum, using algorithmic processes to determine the timing of the event (the horizontal axis of the score) and the playback channel (the vertical axis of the score). Brown did not know nor consider the content of the tape he was placing.[88] Therefore, the only parameters that Brown considered in the composition of *Octet I* were the density of sound clusters and duration of sounds, although even duration was treated as a spatial parameter in the work as it was measured in inches of tape rather than time.

A further example of a slightly different approach by Brown is *Wikiup* (1979), an installation in which several tape players playing endless loop cassettes are strung from the ceiling. The physical locations of the cassette players are re-configurable through a pulley system that is operated by the audience.[89] In his sketch for the piece, Brown writes "I like the idea of walking around in sound—and sound which changes its relationship to itself and to the listener and even to the performers. Most all of my 'regular' music does this."[90] The piece is further established as a work utilizing the static approach to virtual space by Susan Sollins who, in the catalog for the exhibition "Supershow" (for which the work was commissioned), describes

[88] Volker Straebel, "Interdependence of Composition and Technology in Earle Brown's Tape Compositions *Octet I/Ii* (1953/54)" (paper presented at the Beyond Notation: An Earle Brown Symposium, Northeastern University, Boston, January 18-19, 2013).

[89] Earle Brown Music Foundation, "Works: Wikiup," accessed March 20, 2016. http://www.earle-brown.org/works/view/41.

[90] Ibid.

"…the constant change inherent in *Wikiup* which encourages spontaneous invention in relation to a prescribed sound world."[91]

5.1.1 Static Approach to Virtual Space in *Zebra*

The composition *Zebra* (2015) serves as an example of a musical composition for a static virtual space. A screenshot of this virtual space is shown in Figure 10 below, and a screencast of a full performance can be viewed at https://vimeo.com/144070139.

Figure 10: Screenshot from *Zebra*

Zebra primarily consists of an arrangement/realization of a MIDI file released by composer Daniel Lopatin (a.k.a. Oneohtrix Point Never). In a similar approach to Brown's *Octet I* or *Wikiup*, sounds are placed in within a composed space. The sound itself is linear and pre-composed, but the virtual-physical environment in which the sound exists is generative and navigable.

[91] Susan Sollins and Eike Solomon, *Supershow: A Traveling Exhibition Organized by Independent Curators Incorporated, New York* (New York: Independent Curators Inc., 1979).

The MIDI file (somewhat altered) is played back in musical time, driving a polyphonic synthesizer. The MIDI score represents a series of chords, and the individual notes of each chord are distributed so as to emanate from different objects within the virtual space. In this case, the virtual space is an environment created using the Unity game engine and the objects are simple spheres with lights. These spheres are positioned randomly for each performance, so the layout is always different, and the notes are distributed to the spheres based on voice number in the polyphonic synthesizer (using Max's *poly~* object). Therefore, the distribution of notes to spheres is generative but repeatable.

During performance, the performer navigates the virtual space wearing a virtual reality head-mounted display, while the audience watches on a screen from the first-person perspective of the performer (similar to Lanier's *The Sound of One Hand*). Additionally, the sounds are spatialized so as to seem to emanate from their respective locations in virtual space.

This static approach provides an interesting way to realize and explore musical content in new ways. It also gives the performer a certain agency to shape the formal content of the work. *Zebra* was also the first full composition developed using the MASI system and Unity. In this way, it serves as an experiment to test how well this style of performance might work with an audience. The first performance of *Zebra* took place at the 2015 Electric LaTex festival at Rice University in Houston, Texas. Although no formal data was gathered about the audience experience at this particular performance, audience members generally expressed that the first-person perspective virtual world performance worked and, importantly, did not cause nausea or uneasiness. The interaction was clear and effective from an audience perspective.

While this approach can maximize the performer's sense of agency, it does not fully engage the potential of the virtual space. In other words, the virtual world itself lacks agency.

As in Brown's *December 1952* and *Wikiup*, the performer can navigate a world and perhaps even change the positions of objects within that world, but the objects have no agency or ability to change themselves. This issue is revisited below through a more dynamic approach to virtual world composition.

5.2 Dynamic Approaches

In contrast to using a static environment to navigate pre-determined sonic content, one might consider instead using a dynamically changing environment to determine, change, and affect the sonic content. This approach can perhaps be more engaging in some contexts, and it can result in stronger conceptual ties between audio and visual components.

An earlier work of Earle Brown serves as a compelling and historic example of dynamic composition using space. Brown was fascinated and inspired by the work of Alexander Calder. Calder was a sculptor known for his kinetic, hanging mobiles that helped to redefine modern sculpture. Brown desired to create music that was, like Calder's sculptures, re-configurable and therefore "mobile."

Similarly to what has been outlined in this chapter, Brown considered two kinds of mobility: "…one the physical mobility of the score itself, and the other the conceptual mobility—which is to say the performer's mental approach to the piece—holding in mind the considerable number of different ways of moving, moving the mind around a fixed kind of graphic suggestion, or actually physically moving the score itself."[92] "Conceptual" mobility can be considered similar to the "static" approach defined here. The second kind of mobility—that in which the score itself is moving—can be considered the "dynamic" approach.

[92] Brown, "On *December 1952*," 3.

Originally, Brown desired a more dynamic moving score approach for *December 1952*. As Brown describes:

> In my notebooks at this time I have a sketch for a physical object, a three-dimensional box in which there would be motorized elements—horizontal and vertical, as the elements in *December* are on the paper. But the original conception was that it would be a box which would sit on top of the piano and these things would be motorized, in different gearings and different speeds, and so forth, so that the vertical and horizontal elements would actually physically be moving in front of the pianist.[93]

In other words, Brown desired to construct a Calder-esque mobile that would be used as a cue for an improvising performer to perform music "very closely connected to [the] physical movement of [the] objects in [the] three-dimensional box."[94] It appears that the primary reason that Brown never realized *December 1952* as such a score is that he was not "able to get motors" nor "all that interested in constructing it."[95]

Later in his career, Brown would work together with Calder on *Calder Piece* (1966), for percussion quartet along with a mobile sculpture by Calder entitled "Chef d'orchestre" (conductor of the orchestra). The work, which was recently performed at the Tate Modern in November 2015, incorporates the Calder mobile in two ways. Firstly, at times the percussionists approach the mobile and use it as an instrument (the sculpture consists of metal plates that produce a gong-like sound when struck). Secondly, at other times the percussionists watch the movement of the mobile while playing other percussion instruments. In this configuration, the hanging parts of the mobile determine which parts of the score the percussionists should be playing. As Richard Bernas described the recent Tate Modern performance:

---

[93] Ibid.

[94] Ibid.

[95] Ibid.

> …movements of the sculpture are paralleled by the performers' trajectories; …improvised passages played on the sculpture italicize the more notated percussion solos; …the integrity of the concept on a multiplicity of material and sonic levels creates continuity despite some surprises along the way. Though unfixed in some of its detail, the concept is clear and far from arbitrary. Brown and Calder demonstrate that flux, movement and uncertainty can indeed be positives.[96]

3D virtual space can be used to extend the concepts put forward in *Calder Piece* and related works. In addition, 3D virtual space can be used to resolve some of the practicalities involving a score that is itself a moving object. Finally, 3D virtual space can enable generation of complex spatial textures of sound, an idea explored below by another original composition.

5.2.1 Dynamic Approach to Virtual Space in *Calder Song*

Calder Song is an example composition that utilizes a dynamic virtual space. The work is a variation on the idea of Brown's *Calder Piece* but with a different aesthetic approach.

Like *Zebra*, *Calder Song* employs Unity, Max, and MASI to create a 3D audiovisual space with realistic sound source locations that the performer can navigate among from a first-person perspective. However, *Calder Song* has moving parts in the form of Calder-esque virtual sculptures. Each of these sculptures demonstrates a different musical interaction. These interactions are more simple and direct than those in Brown's work, valuing a less improvisational aesthetic than Brown. A screencast performance of the piece can be viewed at https://vimeo.com/163116373. Figure 11 below shows an example of one of these virtual sculptures.

[96] Richard Bernas, "Flux, Movement, and Uncertainty," *Journal of the Institute of Composing*, no. 7 (February 2016), http://www.instituteofcomposing.org/journal/issue-7/flux-movement-and-uncertainty/.

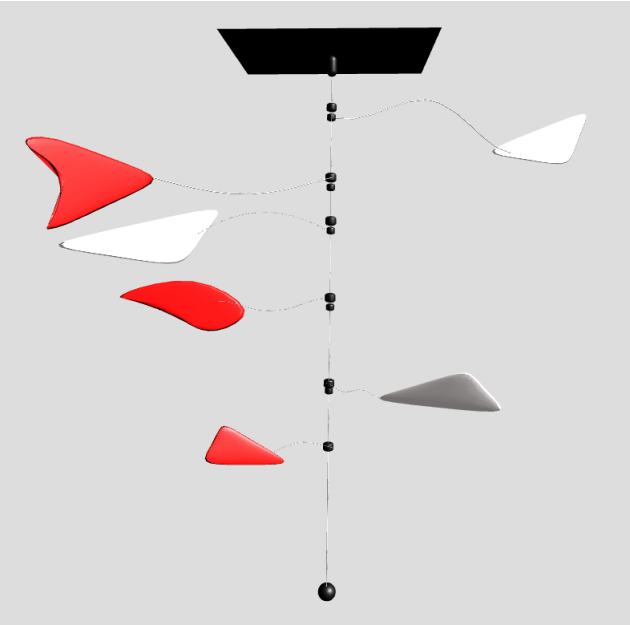


Figure 11: A Calder-like mobile from *Calder Song*

The triangular hanging pieces in this sculpture move as though their connecting wires were attached to motors. Using the physics available in the Unity game engine, it is possible to build sculptures such as this that may be used to affect musical and artistic form through physics, as in Calder's mobiles and Brown's musical interpretations of them. Through the use of virtual space, it is much easier to realize Brown's idea for a moving, sculptural score.

As the mobile in Figure 11 turns, the hanging triangles generate notes when they become vertically aligned with other hanging triangles. The notes each triangle plays are determined by which other triangles they are vertically aligned with, and the sound emanates from the location of the triangle. In this way, the sculpture generates a tapestry of sounds that continually vary their rhythms and reconfigure themselves spatially. The balance and speed of the virtual mobile determine the musical trajectory of the "part" to which the sculpture is assigned, as part of the greater "song."

Other mobiles within the virtual space demonstrate different musical interactions. Figure 12 shows one of the other mobiles:

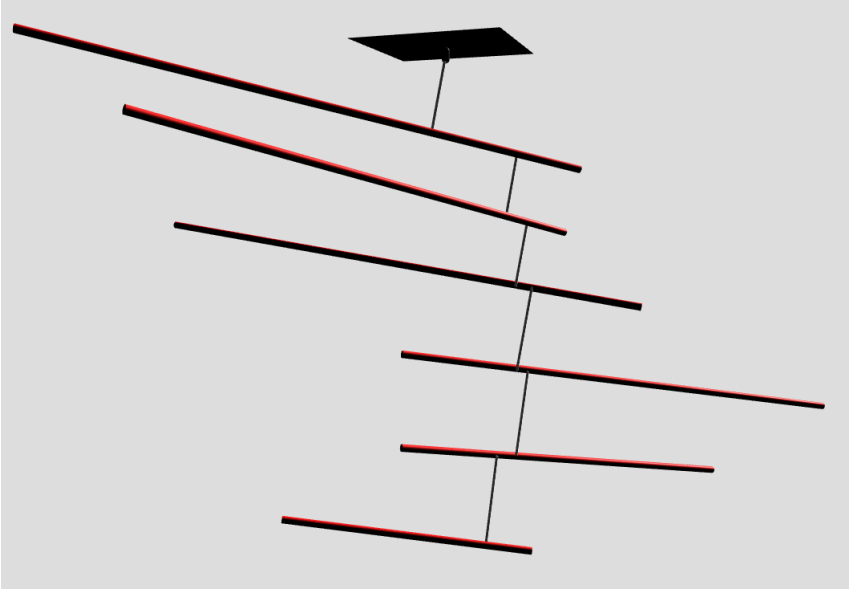


Figure 12: Calder-like mobile from *Calder Song*, mimicking *Untitled*, ca. 1942, wood and wire by Calder.

This particular mobile is fashioned directly after one of Calder's works (*Untitled*, ca. 1942, wood and wire). As is evident, it demonstrates a very unusual sense of balance. Similarly to the mobile shown in Figure 11, this mobile also generates notes as the moving parts cross certain triggers. In the case of this mobile, however, notes are produced only when the ends of the sticks reach a particular azimuth in relation to the upper base of the sculpture. Each stick is assigned a particular note, so notes are not reassigned as in the Figure 11 mobile. The effect is somewhat akin to a music box, in that the sticks are like the pins of a cylinder plucking the teeth of a virtual comb.

One further mobile in the scene serves a more rhythmic role, simply spatializing a composed rhythmic figure. Further non-mobile (potentially moving, but not sculpturally mobile) sound producing objects are added to the scene to fill out the sound and provide guideposts for

the musical structure. The intentionality of arranging the environment into various instruments and parts, and the fact that composer control is maintained over the notes produced and they are arranged according to principles of traditional tonal music theory, demonstrates a marked aesthetic difference between this work and that of Earle Brown.

In fact, several aesthetic choices differentiate *Zebra* and *Calder Song* from some of the other works mentioned in this study. One point of differentiation is the choice of sound materials. Rather than choose to use the environment to control experimental sound processes (like the works by Rob Hamilton, for example) subtractive synthesis is chosen as the method of sound generation. In both *Zebra* and *Calder Song*, all of the sonic content is generated by a two-oscillator synthesizer built in Max. Part of the inspiration to go back to this sound generation technique for these pieces was the work of Daniel Lopatin (composer of the MIDI sequence used in *Zebra*), who utilizes traditional synthesizers to powerful effect in his music.

Another aesthetic differentiation between these pieces and others (e.g. *Poème Électronique*, some works conceived for the AlloSphere, and the works of Rob Hamilton) is the use of rather simple tonal harmony. Using understandable tonal harmonies and more approachable synthesizer sounds in these works may be important for immersion. Stage performance using VR and immersive virtual worlds will be quite new and different for the majority of the audience. By using recognizable musical features, it is possible to more fully engage the audience and to increase their understanding of the process they are witnessing and engaging in themselves. The intention in making these aesthetic choices is to communicate the power and effectiveness of virtual worlds as live performance environments to a broad mass audience.

CHAPTER 6: FUTURE WORK AND CONCLUSIONS

The first area of study that needs to be expanded in this research is musical. Composers must create a more extensive body of work for this type of performance, and this work must be musically engaging and compositionally relevant. Performing for larger and more diverse audiences in different venues, incorporating more live performers, and exploring larger sets and transitions between different virtual worlds are all areas that will be explored.

In addition to this obvious need for more content, there are other possible uses for the MASI system that have significant potential. Two of these areas can be discussed: using MASI to connect 3D graphics and physical modeling synthesis, and using MASI in conjunction with mobile device apps and interfaces.

6.1 Using MASI with Physical Models

Physical modeling synthesis refers to a method of sound synthesis in which the waveform is generated by a mathematical model. A model can be defined as "any form of computation that predicts the behavior of a physical object or phenomenon based on its initial state and any 'input' forces."[97] Therefore, physical modeling synthesis is used to mimic the physical properties of real acoustic instruments and/or to create hypothetical instruments.

New tools are being created to help composers more easily work with physical models. For example, *Synth-A-Modeler* by Edgar Berdahl and Julius O. Smith "enables artists to

[97] Julius O. Smith, "What Is a Model?," in *Physical Audio Signal Processing* (2015), http://ccrma.stanford.edu/~jos/pasp/What_Model.html.

synthesize binary DSP modules according to mechanical analog model specifications."[98] Using this software, it is possible to create physical models using a graphical interface and export the generated code for use in many applications, including Max.

It is easy to imagine many use cases for connecting physical models with 3D environments, particularly those 3D environments that incorporate realistic physics. The Unity game engine, for example, has built-in physical capabilities. Using the Unity scripts provided with MASI, it is possible to pass certain physics information between Unity and MASI.

As mentioned in section 4.6, the Unity scripts send basic collision information to MASI, particularly the name of the object collided with and the absolute magnitude of the collision. An example of using this information along with physical modeling synthesis is can be found at https://vimeo.com/162861547.

The physical model used in this example, generated using *Synth-A-Modeler*, is of a rectangular membrane. Among others, the model has parameters for the size of the membrane and X and Y excitation position. The sizes of the membranes of the three physical models triggered in the linked example are relative to the sizes of the 3D modeled rectangles struck by the spheres. The positions of the spheres as they strike the rectangles determine the X and Y excitation position of the physical model. The spheres are dropped from a virtual height of 20 meters and allowed to bounce as they hit the rectangles. This demonstrates the spheres striking the rectangles with different magnitudes, and the magnitude of the impact is mapped to the pulse width of an impulse entering the physical model.

[98] Edgar Berdahl and Julius O. Smith, "An Introduction to the Synth-a-Modeler Compiler: Modular and Open-Source Sound Synthesis Using Physical Models," in *Proceedings of the Linux Audio Conference* (2012).

This example is a basic demonstration of how the MASI system and physical modeling might be used together. However, it leaves much room for improvement. Pieces utilizing these technologies with greater musicality, more graphical interest, and—perhaps most importantly—more interesting interaction with the 3D models (and therefore the physical sound models), may have stunning potential.

6.2 Using MASI with Mobile Device Apps and Interfaces

As has been mentioned throughout the document, MASI is not tied to any particular graphical interface. Sounds are spatialized using OSC messages, which can be sent from almost any application. Mobile apps can be made to send OSC and, in the case of the forthcoming example, web applications are capable of it as well. The possibilities of this capability are exciting, as there is the potential for spatialization capabilities to be distributed to an audience, or any other situation that might benefit from distributed control. Also, it is relatively easy to generate cross-platform interfaces using web technologies.

Figure 13 shows an example mobile touch interface for spatialization using MASI. In this example, the user can enter the number of sound sources they have loaded into MASI and the real-world size that the web browser space should represent. The sources may be moved around through touch-drag interaction, and multiple sources can be moved at the same time. In some ways, this is a relatively traditional spatialization/diffusion interface. There are some interesting differences, however.

First of all, as sources are moved they do not merely move to different angular locations. MASI's virtual acoustics properties are applied, creating a sense of distance via gain attention, filtering, delay/Doppler shift, etc. Secondly, the headphone icon in the center of the screen is

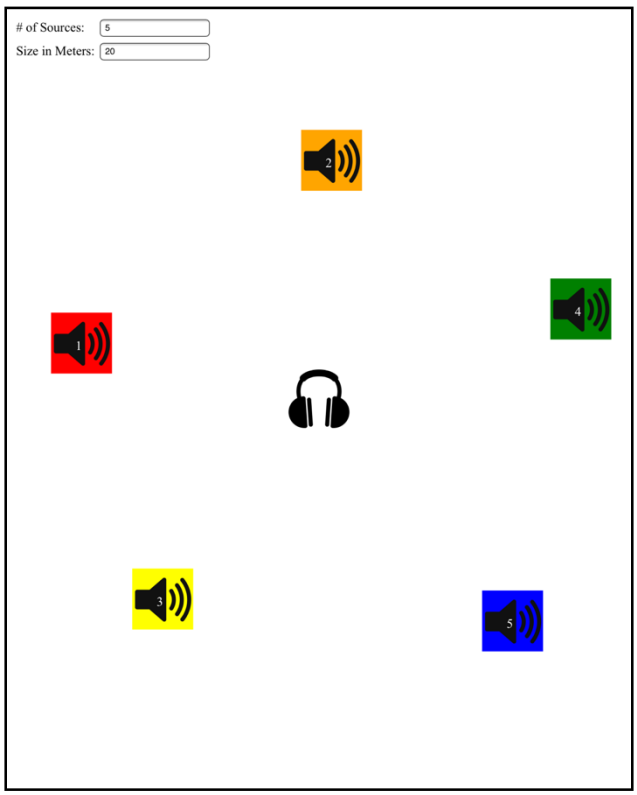


Figure 13: Screenshot from an example MASI control app

also draggable. This icon represents the listener position. Since changing the listener location is an essential built-in function of MASI, it is natural to use this function as well. Sounds can move, but the center of the audience can also change.

As stated previously, another possible application for such an interface is distributed control of spatialization among an audience. The example shown in Figure 13 is a cross-platform web interface. It only uses JavaScript/JQuery and HTML elements with CSS styling. The interaction is through cross-browser standard touch methods. Therefore, it is possible to use this interface on almost any smartphone, tablet, or touchscreen computer by simply navigating to a web page on which it is hosted.

Furthermore, creating distributed applications is now fairly trivial by using a combination of Node.js, which is a "JavaScript runtime built on Chrome's V8 JavaScript engine…" that

"…uses an event-driven, non-blocking I/O model that makes it lightweight and efficient," [99] and WebSockets, which is "…an advanced technology that makes it possible to open an interactive communication session between the user's browser and a server."[100] Using these two technologies, one can start a web server to which users can connect, send individualized messages between the users and the server, and—very importantly for this application—open multiple very quick and responsive communication sessions in which the positions of the sources as the users move them can be relayed back to the server. This type of application was demonstrated at the author's doctoral recital on April 12, 2016, when several users connected to a local server hosting an application very similar to the one shown in Figure 13. In this case, however, the application was designed so that each user would only see one source on his or her screen, and different users were responsible for moving different sources around the space.

6.3 Final Thoughts

Virtual 3D space remains a vital and rapidly developing medium for musicians and composers. While VR technologies such as the Oculus Rift have become increasingly popular and are being hailed as the future of media, almost all of the focus of these technologies in both development and popular consciousness has been on the visual aspect and the solitary, immersive perspective. Yet, these technologies have great potential for artistic expression, performance, and sound, and the industry is beginning to capitalize on this potential.

[99] Node.js Foundation, "Node.Js," accessed April 14, 2016. https://nodejs.org/en/.

[100] Mozilla Developer Network, "Websockets," accessed April 14, 2016. https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API.

Major companies are working toward standardizing the style of audio generation described in this paper. Dolby's Atmos® audio platform, for example, is exploring this "object-based" method. As the company describes on its website, "Rather than being constrained to channels, sounds exist as individual entities that are precisely placed and moved in three-dimensional space."[101] This may well become the standard method of delivering sound in the near future. As in MASI, all sounds will be individual sources, or "objects," as opposed to being mixed down to a fixed channel format. Rather than being the standard only for games, this may be the standard for film, television, and music as well.

Musicians have the potential to lead in this new style of audio. I hope that MASI can serve as an example platform by which musicians might consider the potentials of adapting to these most popular forms of media, and might consider integrating traditional musical practice with technologies normally reserved for video games and now film. I hope that the compositions I have realized thus far using this platform—and the compositions I have yet to realize—can serve as inspiration for the ideas of other like-minded individuals that may want to explore similar territory.

After much time with this study, I am left with many questions and am very excited about the future of this practice. My way of bringing music and virtual reality together as stage performance is but one option among countless potentials. How else will it be done? In what ways will virtual reality become a part of the musical landscape, as it has become a part of the landscape of other media? How can virtual reality enhance or achieve the kind of real-world, physical collaboration that is the essential element of music? I hope to explore, witness, and

[101] Dolby, "Dolby Atmos Audio Technology," accessed April 14, 2016. http://www.dolby.com/us/en/brands/dolby-atmos.html.

achieve a future performance practice that incorporates this new medium, adding a new dimension to music as we know it.

BIBLIOGRAPHY

Amatriain, Xavier, JoAnn Kuchera-Morin, Tobias Hollerer, and Stephen Travis Pope. "The Allosphere: Immersive Multimedia for Scientific Discovery and Artistic Exploration." *IEEE Multimedia* 16, no. 2 (April-June 2009): 64-75.

Austin, Larry. "John Cage's Williams Mix (1951-3): The Restoration and New Realisations of and Variations on the First Octophonic, Surround-Sound Tape Composition." In *A Handbook to Twentieth-Century Musical Sketches*, edited by Patricia Hall and Friedmann Sallis, 189-213. Cambridge: Cabridge University Press, 2004.

Berdahl, Edgar and Julius O. Smith. "An Introduction to the Synth-a-Modeler Compiler: Modular and Open-Source Sound Synthesis Using Physical Models." In *Proceedings of the Linux Audio Conference*, 2012.

Bernas, Richard. "Flux, Movement, and Uncertainty." *Journal of the Institute of Composing*, no. 7 (February 2016). http://www.instituteofcomposing.org/journal/issue-7/flux-movement-and-uncertainty/.

Brown, Earle. "On *December 1952*." *American Music* 26, no. 1 (Spring 2008): 1-12.

Calder, Alexander. "How to Make Art?" In *Calder: Gravity and Grace*, edited by Carmen Giménez and Alexander S. C. Rower, 47. London: Phaidon Press, 2004.

Carr, David. "$1.1 Billion: This Isn't Child's Play." *International New York Times*. September 1, 2014.

Chowning, John M. "The Simulation of Moving Sound Sources." *Journal of the Audio Engineering Society* 19, no. 1 (January 1971): 2-6.

Cruz-Neira, Carolina, Daniel J. Sandin, Thomas A. DeFantl, Robet V. Kenyon, and John C. Hart. "The Cave: Audio Visual Experience Automatic Virtual Environment." *Communications of the ACM* 35, no. 6 (1992): 64-72.

Dolby. "Dolby Atmos Audio Technology." Last modified 2016. Accessed April 14, 2016. http://www.dolby.com/us/en/brands/dolby-atmos.html.

Earle Brown Music Foundation. "Works: Wikiup." Last modified 2013. Accessed March 20, 2016. http://www.earle-brown.org/works/view/41.

Gerzon, Michael A. "Periphony: With-Height Sound Reproduction." *Journal of the Audio Engineering Society* 21, no. 1 (January/February 1973): 2-10.

Hamilton, Robert. "Q3osc Or: How I Learned to Stop Worrying and Love the Game." In *Roots/Routes*, Proceedings of the International Computer Music Association, 217-24. San Francisco: International Computer Music Association, 2008.

———. "Udkosc: An Immersive Musical Environment." In *Innovation, Interaction, Imagination*, Proceedings of the International Computer Music Association, 717-20. San Francisco: International Computer Music Association, 2011.

———. "The Procedural Sounds and Music of *Echo::Canyon*." In *Music Technology Meets Philosophy: From Digital Echos to Virtual Ethos*, vol 1, Proceedings of the International Computer Music Association, 449-55. San Francisco: International Computer Music Association, 2014.

Holmes, Thom. *Electronic and Experimental Music: Technology, Music, and Culture*. 4th ed. New York: Routledge, 2012. Kindle.

Hoose, Shane. "Creating Immersive Listening Experiences with Binaural Recording Techniques." *College Music Symposium* 55 (2015). http://dx.doi.org/http://dx.doi.org/10.18177/sym.2015.55.mbi.10863.

Lanier, Jaron. "Virtual Reality and Music." Last modified January 10, 2010, Accessed May 20, 2016. http://www.jaronlanier.com/vr.html.

———. "The Sound of One Hand." *Whole Earth Review*, no. 79 (Summer 1993): 30-4.

Lombardo, Vincenzo, Andrea Valle, John Fitch, Kees Tazelaar, Stefan Wenzierl, and Wojciech Borczyk. "A Virtual-Reality Reconstruction of Poeme Electronique Based on Philological Research." *Computer Music Journal* 33, no. 2 (Summer 2009): 24-47.

Malham, D.G., "3-D Sound for Virtual Reality Using Ambisonic Techniques." 3rd Annual Conference on Virtual Reality, Westport, April, 1993, http://www.york.ac.uk/inst/mustech/3d_audio/vr93papr.htm. .

———. "Higher Order Ambisonic Systems for the Spatialisation of Sound." In *Proceedings of the 1999 International Computer Music Conference*, 484-7. San Francisco: International Computer Music Association, 1999.

———. *Higher Order Ambisonic Systems*. University of York, 2003.

——— and Anthony Myatt. "3-D Sound Spatialization Using Ambisonic Techniques." *Computer Music Journal* 19, no. 4 (Winter 1995): 58-70.

Moore, F. Richard. "A General Model for Spatial Processing of Sounds." *Computer Music Journal* 7, no. 3 (Autumn 1983): 6-15.

Mozilla Developer Network. "Websockets." Last modified 2016. Accessed April 14, 2016. https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API.

Musil, Thomas, Johannes M. Zmölnig, Vit Zouhar, and Robert Höldrich. "Virtual Audio Reproduction Engine for Spatial Environments." In *Free Sound*, Proceedings of the International Computer Music Association, 790-3. San Francisco: International Computer Music Association, 2005.

Naef, Martin, Oliver Staadt, and Markus Gross. "Spatialized Audio Rendering for Immersive Virtual Environments." In *Proceedings of the Acm Symposium: Virtual Reality Software & Technology*, 65-72. New York: Association for Computing Machinery, 2002.

Node.js Foundation. "Node.Js." Last modified 2016. Accessed April 14, 2016. https://nodejs.org/en/.

Oculus VR. "Introduction to Virtual Reality Audio." *Oculus Rift Developer Center* (2015). Accessed November 15, 2015. https://developer.oculus.com/documentation/audiosdk/latest/concepts/book-audio-intro/.

Sèdes, Anne, Pierre Guillot, and Eliott Paris. "The Hoa Library, Review and Prospects." In *Music Technology Meets Philosophy: From Digital Echos to Virtual Ethos*, vol 1, Proceedings of the International Computer Music Association, 855-60. San Francisco: International Computer Music Association, 2014.

Smith, Julius O. "What Is a Model?" In *Physical Audio Signal Processing*, 2015. http://ccrma.stanford.edu/~jos/pasp/What_Model.html.

Sollins, Susan and Eike Solomon. *Supershow: A Traveling Exhibition Organized by Independent Curators Incorporated, New York*. New York: Independent Curators Inc., 1979.

Stiles, Kristine and Edward A. Shanken. "Missing in Action: Agency and Meaning in Interactive Art." In *Context Providers: Conditions of Meaning in Media Arts*, edited by Margot Lovejoy, Christiane Paul, and Victoria Vesna, Loc. 574-1008. Chicago: Intellect, 2011. Kindle.

Straebel, Volker, "Interdependence of Composition and Technology in Earle Brown's Tape Compositions *Octet I/Ii* (1953/54)." Beyond Notation: An Earle Brown Symposium, Northeastern University, Boston, January 18-19, 2013.

Thompson, John, JoAnn Kuchera-Morin, Marcos Novak, Dan Overholt, Lance Putnam, Graham Wakefield, and Wesley Smith. "The Allobrain: An Interactive, Stereographic, 3d Audio, Immersive Virtual World." *International Journal of Human-Computer Studies* 67 (2009): 934-46.

Tyranny, 'Blue' Gene. "Out to the Stars, into the Heart: Spatial Movement in Recent and Earlier Music." *NewMusicBox: The Web Magazine*, January 1, 2003.

Virtual Reality Society. "Vpl Research." Last modified Accessed November 15, 2015 http://www.vrs.org.uk/virtual-reality-profiles/vpl-research.html.

Wakefield, Graham and Wesley Smith. "Cosm: A Toolkit for Composing Immersive Audio-Visual Worlds of Agency and Autonomy." In *Innovation, Interaction, Imagination*, Proceedings of the International Computer Music Association, 13-20. San Francisco: International Computer Music Association, 2011.

Wenzel, Elizabeth M., Marianne Arruda, Doris J. Kistler, and Frederic L. Wightman. "Localization Using Nonindividualized Head-Related Transfer Functions." *Journal of the Acoustical Society of America* 94, no. 1 (1993): 111-23.

APPENDIX: MASI SOFTWARE MANUAL

Multi-source Ambisonic Spatialization Interface (MASI)

What is MASI?

MASI is a series of patchers for Cycling '74's Max (https://cycling74.com/) that provide a simplified interface for the realistic spatial positioning of sound sources in a virtual 3D environment through ambisonic (https://en.wikipedia.org/wiki/Ambisonics) panning and virtual acoustics. Because MASI uses ambisonic panning, the sounds can be decoded for playback on multichannel speaker arrays (e.g. 5.1, 7.1, octophonic, or any arbitrary array) or for binaural playback on headphones and standard stereo systems.

MASI does not provide a graphical panning interface itself, but instead connects to other user-created graphical interfaces through Open Sound Control (OSC) (http://opensoundcontrol.org/introduction-osc) communication. MASI is primarily intended to be used in conjunction with 3D game-like virtual world environments/interfaces. Scripts are provided to connect MASI with the Unity (https://unity3d.com/) game engine.

MASI is licensed is under the terms of the GNU Public License (http://www.gnu.org/copyleft/gpl.html) .

Instructions

Installation

Dependencies

- MASI uses CICM's HOA Library (http://www.mshparisnord.fr/hoalibrary/en/) (source (https://github.com/CICM/HoaLibrary-Max/)) (v. 2.2 or higher) for various ambisonic encoding and decoding tasks. Download the HOA Library for Max here (http://www.mshparisnord.fr/hoalibrary/en/downloads/max/) and place it in your Max Packages folder.

Installation Instructions

- MASI is structured as a Max Package. To install, download the repository as a .zip (https://github.com/zberkowitz/MASI/archive/master.zip) , de-compress, and place the folder in your Max Packages folder (in Max 7 this is located at `/Documents/Max 7/Packages` on both Mac and Windows).

How to Use

Components

There are 3 basic components needed to use MASI: channel configurations, sound sources, and compositions.

1. A "channel configuration" is a user-supplied list of azimuth/elevation coordinates specifying the speaker setup (this is unnecessary if using binaural rendering). This should be created as a single-line `.txt` text file. For example, a typical 4-channel setup (Lf, Rf, Ls, Rs) would read as follows (in degrees, 0° is front, direction of rotation is counterclockwise): `45 0 315 0 135 0 225 0`. Azimuth is between 0° and 360°, while elevation is between 90° (directly above listener) and -90° (directly below listener). See the provided example channel configurations (under `misc/channelconfigs`) for more example configurations.

2. A "sound source" is a stream of audio in Max. Sound sources are made available to MASI through outlets in Max abstractions. A very simple example of a MASI sound source is the following patch, placed somewhere in the Max search path:

3. A "composition" is a user-supplied JSON file that contains key-value pairs denoting an abstraction with one or more sound sources (outlets) and a unique name for each source. For example, if the simple patch shown above was saved as `source.maxpat` somewhere in the Max search path, and the single sound source it contained should be called `uniqueName`, then the JSON should read as follows:

```
{
 "source" : "uniqueName"
}
```

It is also possible to have a single abstraction with multiple sources (outlets):

```
{
 "source": ["uniqueName1", "uniqueName2", "uniqueName3"]
}
```

Or multiple abstractions with any combination of single or multiple sources:

```
{
 "source1": "uniqueName",
 "source2": ["uniqueName1", "uniqueName2"]
}
```

With these three components in mind, Open the MASI main patch, found in the Max `Extras` menu, and follow these steps:

1. Set the decoding mode to `regular` (default) or `binaural`.
2. Load the channel configuration (single-line `.txt` file). This step is unnecessary if using binaural mode.
3. Set the composition (JSON file).
4. Load the composition (this final step loads the abstractions specified in the composition JSON file and connects their outlets to MASI).

Spatializing Sound Sources

OSC Communication

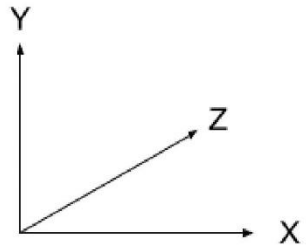
Moving a Sound Source

Once a composition has been loaded, the sound sources can be moved through space using OSC messages to address each source individually. When open, MASI is receiving OSC at the local IP address on the port specified in the `OSC Receiver` portion of the main patch. The address of each source is determined by the unique name assigned in the composition JSON file. For example, given the following simple composition:

```
{
    "source" : "uniqueName"
}
```

the single sound source can be moved by sending the OSC message `/uniqueName/position X Y Z` to the receive port, where X Y and Z are coordinates in 3D space.

MASI uses a left-handed coordinate system with vertical Y, as shown below:



MASI treats a unit as 1 meter. For example, a sound source at position `-5 0 0` will sound as though it is 5 meters to the left of a centered listener (achieved using a combination of ambisonic panning and acoustic principles such as amplitude scaling, delay, filtering, and reverb scaling).

Moving the Listener

Additionally, the position and rotation of the listener or "camera" in a first-person virtual world environment can be changed by sending the OSC messages `/position X Y Z` and `/rotation X Y Z`

Named Receives

An additional feature of MASI is the ability to access information within user-created patches via a system of uniquely named sends and receives. When a composition is loaded, the following sends are created for each sound source (where `uniqueName` is the name assigned to the source in the composition JSON):

`uniqueNameAzimuth` `uniqueNameElevation` `uniqueNameDistance` `userPosition` `cameraRotation` `uniqueNameEnable` `uniqueNameCollision` `OSCDump`

For example, the object `receive uniqueNameDistance` in any patch will provide the distance between the user and the object. `uniqueNameAzimuth` and `uniqueNameElevation` will provide the azimuth and elevation of the object as provided to the ambisonic decoder. `userPosition` and `cameraRotation` will provide the position of the user in the scene and the rotation of the camera (head) respectively.

When using the Unity scripts (described in the next section) a few additional receives are provided. `uniqueNameEnable` receives a 1 when an object with the `ObjectOSC` script is enabled in the scene. `uniqueNameCollision` will receive the name of the object that an object with the `ObjectOSC` script collided with as well as the initial magnitude of the collision.

Additionally, `OSCDump` will receive all incoming OSC messages, which can be parsed using standard methods within Max such as the `route` object.

Using Unity

One of the best use-cases for MASI is in conjunction with the Unity game engine and editor. Two C# scripts for Unity are provided in the `misc/Unity` folder. `ObjectOSC.cs` can be attached to any Unity game object and reports the position of the game object, while `CameraOSC.cs` can

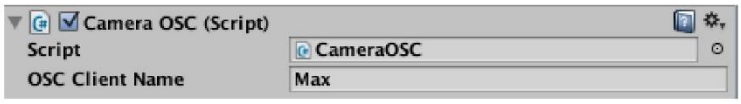
be attached to the main camera in a first-person setup and reports the position and rotation of the camera. Unity games can be compiled for a variety of platforms, and Unity 5 enables easy integration with virtual reality HMDs such as the Oculus Rift.

Dependencies

The Unity C# scripts use Jorge Garcia's UnityOSC (https://github.com/jorgegarcia/UnityOSC) . Download and follow the instructions for incorporating UnityOSC into your Unity project before attempting to use the MASI Unity scripts. You will also need to follow the instructions for adding a new OSC Client to the `OSCHandler.cs` script.

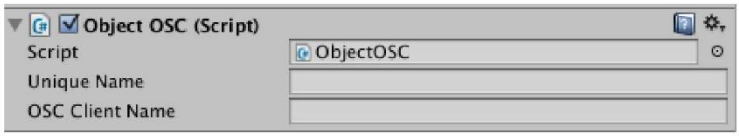
Camera OSC

The `CameraOSC.cs` script should be attached to the main camera in a first-person environment. The easiest way to set this up is to use the `FPSController` prefab found in the Unity Standard Assets. `CameraOSC.cs` should be added to the `FirstPersonCharacter` prefab (which is a child of `FPSController`). You will need to set the public variable `OSC Client Name` to the name of the OSC Client created in `OSCHandler.cs` (`Max` in the following example). The camera will now initialize the `OSC Handler` and report its position and rotation (**Note**: since this script handles initialization, if you are **not** using the `CameraOSC.cs` script then the `OSCHandler.Instance.Init()` method will need to be called elsewhere):



Object OSC

The `ObjectOSC.cs` script can be attached to any game object to report that game object's position when moved. This script has two public variables, `Unique Name` and `OSC Client Name`. If left blank, these variables will default to the name of the game object and the OSC Client specified by the Camera OSC script, respectively, but can be set differently if desired:



The `Unique Name` corresponds to the unique name specified in the composition JSON. Therefore, a Unity game object named `Cube` and a composition JSON such as:

```
{
    "abstractionName" : "Cube"
}
```

will interact properly.

VITA

Zak Berkowitz is a composer/media artist/sound engineer/percussionist/coder currently pursuing his PhD in Experimental Music & Digital Media at Louisiana State University. He received his Master of Music degree in Music Technology from Georgia Southern University, where he studied primarily with Dr. John Thompson. He was awarded a Bachelor of Music in Instrumental Performance from New Mexico State University, where he studied percussion with Dr. Fred Bugbee and Dr. Ed Pias.

Zak's current research and creative work explores the virtual worlds made so familiar by modern 3d video games and highly lauded new virtual reality technologies. He is seeking ways to turn the VR experience outward by performing on stage within his own virtual world using the Oculus Rift and massively multichannel spatial audio.

Zak’s work has been presented at recent festivals and conferences including the Root Signals Electronic Music Festival, Linux Audio Conference, ICMC, Channel Noise at Georgia Southern University, the New Mexico State University Contemporary Arts Festival, and SEAMUS.