IMPROVISING AVATARS: GAME ENVIRONMENTS AS A NEW MEDIUM FOR INTERACTIVE COMPUTER MUSIC

Steven Lewis

University of California, Irvine
Integrated Composition, Improvisation, and Technology
smlewis2@uci.edu

ABSTRACT

This is a presentation of research outlining the technical and aesthetic considerations in conceptualizing, constructing a virtual game environment as a new medium for procedurally interactive computer music. Extrapolating techniques from a litany of culturally diverse musical practices, including free jazz performance, improvisation, acoustic augmentation, and procedural sound design, a cross-platform, computer-mediated environment between Max/Msp~/Jitter and the Unity Game Engine emerged. The resulting experiment was a virtual musical world featuring a hybridization of an indeterminate Free Jazz Performance Model, with the improvisatory activity occurring between the human performer and a secondary, virtual player controller.

In an effort to derive novel, electro-acoustic improvisational techniques based on non-linear, procedural human-computer interaction, the virtual player's gestural options, as well as their discreet and continuous movements, are converted into a stream of running data. By constructing a musical environment in this manner, there exists an inescapable duality between the participating musicians, where the human performer and virtual player's output greatly influences the decision- making process of the other, and by extension, the processing of the audio and video source material.

1. INTRODUCTION

The initial premise of player interactivity within video games and other immersive environments is based upon the notion of the individual requiring a deep sense of emotional immersion during game interaction. Unlike the non-participative experiences passively observed by audiences in film and television, the user has a direct impact on the events taking place place during the gameplay experience. While procedurally generated graphic displays and visual cues within the gameplay environment often reflect this to the user, game audio technology has found itself significantly lacking in its ability to generate continuously changing audio events that keep responding to the physical stimuli occurring around game participants.

However, the current proliferation of game environments in consumer markets yields innovative ways in which technologists and artists can collaborate to establish new performance expectations of the audience and performer alike; where there was once merely the passive witnessing of an event, audiences would be able to fully participate and interact with their surrounding environment, thereby creating a more enveloped sense of immersion for every participant involved. Immersive realities and 3D game environment are an ideal space for exploring these artistic possibilities, where outcomes could be controlled by game objects programmed to provide meaning data for spatial positioning, shape and morphology.

Furthermore, procedural player-systems communications between multiple programs could be thought of as another iteration of algorithmic composition, one which has substantive performative implications for human-computer interactivity in contexts other than immersive realities. Rather, the concept can be applied in alternatively defined immersive experiences, which includes sound installations, art installations, audio reactive-art, game audio, and live music performance. This was the primary impetus behind this research: establishing a systemic foundation for leveraging techniques from live or gestural performance, procedural sound design, generative composition, and interactive computer music into an entirely different medium, where all of these seemingly disparate elements are integrated into an interdependent system which generates a new musical outcomes upon each iteration of the game experience.

2. HUMAN COMPUTER MUSICAL INTERACTIVITY

3.1 A Layered Approach to Classifying Models of Human Computer Expression

The diversity in systemic paradigms can be partially linked to opting for a linear or non-linear approach to structural organization, but the specific type of musical interactivity each composer vies to emulate is another crucial influence on the idiosyncratic behavior of the program. Thus, there is an interdependent relationship between the score functionality and model of interactivity featured within it; the subjective collaborative process for which each composer prefers their music to made with will programmatically instruct computers to organize and perform information accordingly (Desain, 1992).

The four primary Performance Models of human-computer interactivity - the Conductor, Chamber

Music, Jazz Combo, and Free Jazz Models (2001) - are paradigms attempting to emulate specific live performance traditions primary extrapolated from the styles of Classical and Jazz. The former relies on judiciously constructed musical forms, while the latter accentuates spontaneous improvisation based on a set of collectively agreed upon, highly social, musical guidelines. From the comprehensive user influence found in the Conductor Model, to the completely freedom displaying the Free Jazz Model, the amount of control the designer of each paradigm exerts over the overall sonic output diminishes with each model, respectively.

However, adhering to four standard classifications become far more difficult to achieve as systems have increased in complexity. The realities of working with these paradigms can reveal nuances in categorization, where observable behaviors within one system run contrary to its overall classification. Similarly to the innumerable music genres that resemble a fusion of multiple and varied cultural and performative traditions, it is evident that interactive systems can in fact show an amalgam of the attributes associated with multiple paradigms in order to successfully achieve the desired musical output (Rowe 1992). Because of the behavioral diversity in these interactive models, expanding the four aforementioned archetypes can be done not only by observing how this communication is organized, but by the way and degree to which the systems respond to a wide variety of user-inputs: to what degree of structural and micro-control is the composer willing to relinquish to the computer? The degree to which the computer's decisions affect the actions of the human performer can be broadly determined by whether the objective is to create a virtual musical instrument which acoustically augments a real-time performance by analyzing human gesture, or if the imperative is to construct a cybernated improviser, insisting of their own performative behaviors nearly independent from the user input (Rowe, 1992). These considerations are at the crux of studying these primary classifications, their hybrid adaptations and in turn, aid in contextualizing new systems by historically cross-referencing modern interactive behavioral functionality and response methods to those paradigms established by their predecessors.

3.3 Examining the Free Jazz Performance Model

The Free Jazz Performance Model is the most apt archetype for this non-linear dynamic , where neither the performer nor the computer user is in complete control, resulting in two distinct improvisatory decision-making processes emerging between that of the human and computer in real-time (Winkler, 2001). This model is emulating the group performance traditions which commenced in the 1960's, where great improvisational liberties were granted to members of experimental jazz ensembles in an effort to establish new parameters for solo and collective musical expression (Pressing, 2002). However, as previously stated, increasingly complex player behavior derives a far more nuanced interpretation of how the performing instrumentalists (user-input) and its computerized player controller leverage techniques from other paradigms in order to model musical typologies. The musical relationships that emerge from such freedoms not only yield different outputs upon each improvisation, but create a dynamic that is also similar to the non-linearity found in video game music; if no one entity - the musical score, specific musician, agreed upon tradition, or temporality - is always guiding nor determining a centralized end result, then an environment has been established that simultaneously increases the unpredictability of the music while also making the temporal linearity of the music less important than the ability of the musicians to adjust to an ever-changing dynamic within each iteration of an improvisational process (Borgo & Goguen, 2005).

Combining characteristics from multiple performances models is a commonality within large-scale interactive systems, (Winkler, 2001), where one of the formalized performance model's functionality must be expanded in order to achieve certain desired sonic outcomes. The flexibility built into the score-following capabilities of a system need to be matched by the allotment of freedom (control) allocated to each of the human and virtual members at any given point in the performance. In order to accurately model this dynamic, the virtual performer must be given as much freedom as possible, so much so that it may not even seem as if the two disparate parts of the system are directly influencing each other in the slightest. Thus the Free Jazz Model includes elements of Rowe's Player-Paradigm (1992), where there exists a second layer of interactivity being controlled by an additional virtual user - the game's main player character. This secondary user is the representation of another performer, as it has been given human-like physiology, a discreet set of gestural options, and the freedom to utilize these options exactly how it is that they choose within the environment (Lewis, 2013). Emulative of the Free Jazz Performance Model, this freedom facilitates a greater degree of independence for all active participants while simultaneously developing a dynamic, non-linear relationship between composer (programmer) and performer.

3.4 Associated Technical and Analytical Considerations for the Free-Jazz Model and Player Paradigm

The analytical or evaluative significance of a free jazz performance is often indomitably linked to "the (musical) judgements made within a shifting cultural and historical context" (Clark, 2012). For example, it is difficult to evaluate a Sun Ra performance in its entirety without fully understanding the cultural and material conditions of the society in which they were created. In a musical context using computer systems, iterative interactions with a software-driven system will "tend to reveal the characteristics of the community of thought and culture that produced them (Lewis, 2000). Unlike in other jazz idioms, Free-Jazz Performance Models are emulating behavioral classification that can be rarely fall within, "the panoply of devices known to music theory" (1987). Even within the practice of generative systems or strictly tonal jazz improvisation, this standardized panoply can be easily categorized into quantifiable micro and macro-interactions for further analysis (Linson, Dobbyn, & Laney, 2012). Free Jazz Performance Models exemplify the difficulties researchers and composers have in determining the best methods for evaluating the correlation between a system's input and overall sonic output, and how the relationship between the two players is perceived by the listener. The challenges are exacerbated when fusing

Rowe's Player-Paradigm with this particular hybrid model.

These systems utilize the virtual player controller with "a musical voice which may be related to, but still in an audible way distinct from, the performance of a human partner" (Rowe, 1996). The data that may reveal pertinent information related to the relationship between the virtual player and its human counterpart may not be able to be defined before the performance. Therefore, the inherent difficultly lies in the measuring for meaning quantifiable data when one may not initially know what relationships may or may not immediately exist between the factions of the system.

Conversely, a purely subjective analysis of a Free-Jazz Performance Model is not ideal either, as the feedback provided will only be relative to the listener's experience, their highly personalized definitions of jazz, and any initial expectations that may arise from these preconceived assumptions (Linson, Dobbyn, & Laney, 2012). Whether or not the output is deemed aesthetically pleasing or satisfactory on a personal level is wholly irrelevant in assessing a system's sonic response to dynamically changing user and virtual player inputs. Similarly to the way in which procedural audio is evaluated based on the physical, perceptual, and psychological relationships a player will establish between themselves and the dynamic behavior of a game object, the Free Jazz Performance Model can be evaluated based upon the ease of which the listener or performers can relate the actions of the virtual player to the musically relevant decisions made by the performer.

4. DESIGNING THE ENVIRONMENT

Considerations must be made that convert existing interactive works from stand-alone programs to new iterations which technologically enable communications between multiple software systems in real-time. In addition to addressing this necessary technical altering, there exists an even more vital consideration: conceptualizing the player's overall effect of the systems musical output, and subsequently constructing a game environment and thoughtful mapping strategies that facilitate an interactive musical process between involuntary automation instruction and user-controlled input. This is highly dependent on not only constructing the game simulation itself, but also judiciously choosing the specific ways in which the player's surrounding environment influences the choices linked to the processing parameters controlling the systems's musical output.

4.1 Conceptualizing the Game Environment

A significant amount of time has to be designated to the initial non-musical stages of the project so that adequate attention can then be allocated for the anticipated technical challenges in connecting the separate programs. While the Max programming component of the project was complex, the Unity game scene was kept relatively simple by comparison[1]. With the exception of adding in enemies and numerous moving shapes (cubes, spheresA), the original scene from a previous gaming project in Unity was used as a template. Considering the manifold of mapping options within the Max patching, the need for a complex game scene was as unnecessary as it was superfluous. A scene need not be overly complex nor ornate for it to contain changing data that could be monitored and effectively used to control a litany of interdependent processing parameters.

The intention was to have a large, open area consisting of

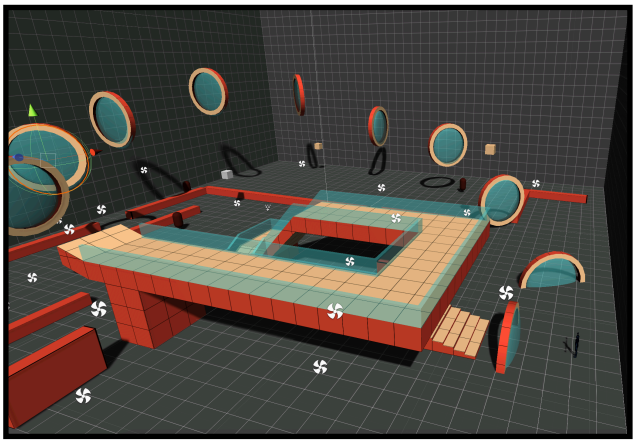

Figure 1. Basic Design Concept

numerous moving parts, each having the ability to change their positional coordinations in three dimensional space. The game level could be thought of as a form of instructions to the performer of this metaphorical, virtual musical score. Similar to a composer deciding vital aspects of a piece of music - tempo, key signature, form, dynamic levels - the game environment itself would inevitably influence the rate and directionality of the virtual player's positional data by exchanging the coordinate information from moving objects in one program to another program. There is an interdependency that exists between the act of designing the game environment and the necessary thoughtfulness invested in mapping the incoming data. In a sense, the role of the game designer is as integral to the musical aspects of the audio-visual compositions as the functionality of the music software or the live performance input.

While it seems like it may lack imagination relative to the ornate game designs that are easily implemented into Unity (and elsewhere in professional global gaming industries), the benign nature of the level serves a vital musical function. If the surface of the level was constantly changing in elevation, or if the walls were too close together, then that would translate to a sound parameters that would be changing only due to the nature of the game environment design itself. The choices of the player would be immaterial in comparison to the overall design of the game world. It is of paramount importance to provide a litany options for how a player could provide meaningful data with limited gestures at their disposal. The area of the game world needs to establish the para-

[1] Original scene was part of a flying tutorial and asset package, which can be found at https://assetstore.unity.com/packages/templates/systems/3rd-person-controller-fly-mode-28647

meters of the choices to the player, but not enforce nor even suggest which options needed to occur at a specific point in time and space. Moreover, it is vital to consider the capabilities and limitations of the virtual player controller when constructing the scene. There was no sense in placing any object in the scene with which the player could not interact, nor is there any musical reason to design a character that could perform an action that was irrelevant or ineffective in respect to the game design;

A prime example of these considerations materializing in the game scene is the use of half and full-circles to initiate the simultaneous exchange of multiple streams of data:

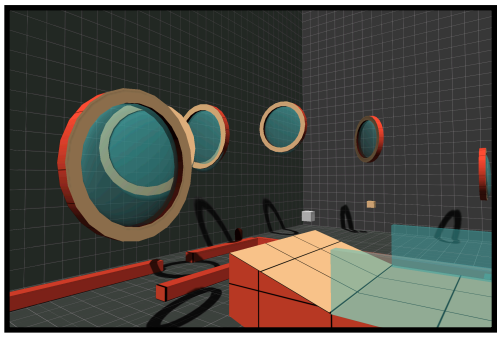

Figure 2. The Launching Point for the Player's Prospective Flight Path

When the player chooses to walk through one, it initiates a trigger being communicated to Max. The same immediate result will take place if one flies though the full cycles above the surface. However, it is the combination of multiple decision in realtime that have the greatest effect, as flying through the circles is a combination of key commands that are complete different than the choices made when interacting with the half circle on the ground. Not only do the distinctive key commands effect the novel processes in Max individually, but the rate at which the player moves is also a continuously changing parameter. This simultaneously effects multiple processes, ultimately yielding musical outputs that could not be anticipated otherwise.

Similarly, if a player wants to avoid enemies so as to not initiate the musical processes that occurs as the result of colliding with these adversaries, the options to do so were to be made clearly evident to the participant. The same options exist for when the player encounters an opportunity to jump over barriers, or break through glass, for instance. This ensures that the balance between the dichotomous elements of structure and improvisation within this metaphorical musical form translates into how the game environment is experienced. Within this context, the structure is the game scene itself, with the improvisation element found in the way in which the player chooses to interact with all the different elements they encounter.

4.2 Conceptualizing the Player Role

The player is analogous to motion-based, gestural improvisation within an interactive music setting. The virtual player character needed to not only be the avatar through which the user experiences the game, but have the most direct effect of the musical processes taking place. Simply stated, the user needed to instantly understand the relationship between their gestural decisions and their effect on the music and the visual output. As the result of this intention, the player needed to been able to utilize numerous gestures and subsequently incorporate them into a control system that would yield musically sensible results.The gestures provided for the player character were as follows:

- Idle on the Ground.
- Walking Forward
- Walking Backward
- Lightly Jogging
- Running Full Speed
- Jumping
- Flying slow
- Flying Fast
- Idle Hovering
- Crouching
- Punching
- Rotating in a Circle

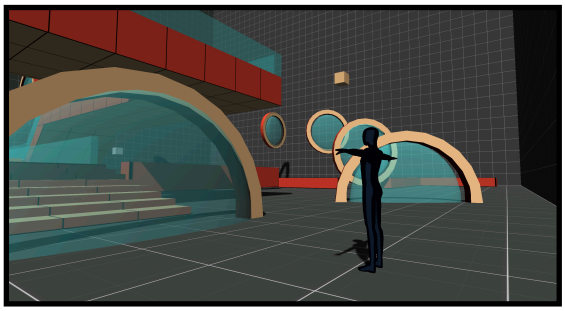

Figure 3.Initial Player Positioning

These gestures could be thought of as to being analogous to the instructions a composer may give to the performer of a graphic or aleatoric score: a musician has been limited to only a subset of what is technologically possibly (whether on a traditional acoustic instrument, an interfaces o controller), but the conditions or instructions presented to them by the composer (in this case the program for designer) aid in them using these restrictions in a creative manner.

There are at least twelve discreet gestures the player can opt to make. However, it is necessary to understand that these controls are being manipulated in real-time, and in most cases, at a rapid pace. For example, there is a transition period between the decision to light jog and sprint at full speed. This period can be thought of a hybrid gesture in and of itself, one in which the user actually has no unilateral control over. Taking this into consideration, the actions available to the player are by no means generating the same type of meaningful data: running around in a circle would indicate changes in positional data, while punching (and connecting) would indicate a discreet action in time. These are completely different actions, and need to be translated consequentially within the Max programming.

A discreet action - jumping, punching, crouching - initiated a process where the events being in triggered in Unity can be translated into a Bang or Toggle in Max. However, continuously running around at various speeds yields three-dimensional positional data relative to the

game world. These disparate gestures yield contrasting data types. Discreet gestures enact a process, such as toggling the individual processing modules, randomizing the value of a control sources, or initiating the playing of an audio file. Positional data used to control the parameters of whatever components the Unity triggers "banged in Max. This means that the positional data needs to change the processing over time while the actions enacted by single key strokes in Unity control the onset of that processing taking place.

A prime example of this is in the way in which the decision to "jump and fly" (space bar keystroke) turns a pitch-shifting algorithm on, while the player flying after this command changes the particular rate and note of the pitch shifter based upon how fast what direction the player chose to fly. This seemed logical, as it was completely impossible to ever fly without jumping first. If the decision to control the pitch shifter (the amount of signal to be shifted, which pitches to use) is directly related to the positional data while in the air, a discreet keystroke from the virtual player was needed in order to turn this process on in the first place. Otherwise, notes are be continuously pitch- shifted. This was a vital process taken for nearly every module in Max, as having every single parameter turned on during the game would yield not only undesired musical results, but would significantly hinder the overall performance efficiency of the system in general.

4.3 Conceptualizing the Role of Enemies

The enemies assume a completely different function than the player, as they operate with a different logic and control systems. A combination of the A.I. NavMeshGuide and a pathway (known as a waypath in Unity) was chosen because it was one of the most convenient methods in implementing a significant measure of control over the player experience while also automating all of the movement during the real-time experience. The NavMeshGuide allows the non-player characters to find the player by using a simple planar surface that is rendered before the game is ever compiled. After this surface renders, the enemies can then track the player all over the game level. The last step of this procedure is to refine the parameters of the enemies' movement. If left unmodified, any objects connected to the NavMeshGuide will move right toward the player. Too many simultaneous encounters meant that there would be multiple decisions initiating musical processes that may or may not be the intended result from the improvisor.

The waypath functionality can keep the enemies within a range of locations. Instead of instantly locating the player upon initializing the game scene, the enemies use the NavMeshGuide only as it relates to the established points on the waypath. Enemies track the player while constantly moving at one speed, only increasing speed and direction if the player comes into the defined proximity. If the player removes themselves from one of these proximal ranges, then the enemies revert back to their original waypath starting positions and respective speeds.

Similarly to the bangs being triggered when the player crosses the threshold of the circular props placed around the scene, enemies making contact with the player initiate musical processes that can only start after a collision takes place. Furthermore, the enemies oscillating between normalized and faster speeds necessitates the player to choose which gesture to use in response. The player can choose to fly away, run away or maneuver in other ways to avoid colliding with the enemy (unless a collision yields the desired musical result the player is looking to achieve). Providing each of the enemies with their own Speed and route parameters creates distinctive sections to this large level, somewhat metaphorical to different movements of a musical score.

4.4 Modifying Control Systems

The last (and most crucial step) in designing the system is expanding many of the preexisting control systems initially built in Max into a more flexible system which enables two-way musical interaction between the virtual player, the drum performance, and the game environment. The performing musicians establish the initial values of any and all parameters that they wanted, while the smallest hint of user interaction within the game environment would send data that would ultimately affect the processing parameters. Rather than having to repeatedly make the same gesture, the virtual player had to be able to execute the simplest of actions - slowly walking - and yield some sort of change within a contra source specified by the programmer.

The most intuitive type of information to send into Max for the purpose of providing control data would have to be continuous in nature. The data would have to be analogous to an LFO, where a constant source of data could be continuously provided at an inaudible frequency. While the player was under the immediate control of the user, player enemies were the only entity within the game environment that were constantly moving, regardless of the decisions being made by the user. The player could stop moving and this constant source of control data is still provides to Max through the use of four enemies and their respective movement along their individual waypoints. The Non-Player Characters (NPCs), would operate as the metaphorical LFOs for this system - an absolutely essential component of the project to function as it was intended to. These more gradually moving game components could be used to incrementally alter musical processes that not need to be constantly changing in overtly dramatic fashion. Aside from the virtual player, the enemies are the most widely leveraged objects from Unity, as they could easily be to make significant sonic changes to the piece over varying durations.

One of the prime examples of utilizing NPC data was in the tracking and mapping of two of the enemies' Z transforms, both of which ultimately controlling the cross fading and spatialization of the quadraphonic panning in Max. These alterations in data flow translate into continuous streams of numbers for LFOs, mapping, and automation values. As previously indicated, these values have to be scaled in order to be of a relevant range of values within the Max patch. In this particular example, Z- transforms were scaled to values of 0-127 MIDI so that they could serve as the X and Y input to the pictslidr controlling the quadraphonic cross fading. Mapping the incoming data from Unity to Max in this way causes assigned sounds to spatialize at the rate at which these disparate characters move about the scene independently from the player. Furthermore, audio spatialization is an

ideal example of a musical parameter that one may want to only subject to gradual rather than sudden changes.

Other gradual changes can be made by scaling the positional transforms of the enemies, including blending the overall dry and wet send values to the reverb objects, and in adjusting the multi-tap delay values as the result of player detection and NavMesh movement. This iterative process, while painstaking and heavily reliant on randomness and an interdependency of a myriad of moving components, creates a vast array of sonic textures that would have otherwise been difficult to conceptualize and materially execute.[2]

The purpose of this version of the system is for the player and the game environment to simultaneously assume a vital, functionary, yet transparent role in an interactive, live music making process. This would confirm two vital priorities: that the performers would feel that the integrity of their performance was not compromised while the virtual player simultaneously became as vital to the music making process as the instrumentalists.

5. CONCLUSIONS AND FUTURE WORK

5.1 Personal Observations:

The project succeeds on many artistic fronts, despite some of the interactive aspects becoming slightly opaque. The artistic output of the system was far more subjectively important than how readily apparent the interactive technology appeared to other users or evaluators. To solely evaluate on a technical basis, or for the technological components to be the most distinguishing or easily apparent component of the experience, seems to undermine the premise of the project. to develop an interactive music system where musicians and virtual controllers would be simultaneously affecting and informing the performance of each other. The purpose of the technology was in facilitate this dynamic, not necessarily be wholly evident while observing the musicians' interactions.

The difficulty in assessing a procedurally interactive system made with the Free Jazz Performance Model in mind is in harnessing the objectivity to parse the significance difference between what each interactive artistically produced, and how it went about doing so. Those are two complimentary, yet opposing types of assessing, with one being vastly more subjective than the other. A system can create a satisfying piece of music (or visuals) yet do so in a way that was unexpected, unintended, or unclear. In this instance, while the system created compelling sonic and visual, the relations between the end result snd the level of dynamic, multi-layered interactivity was not easily recognizable as it was it thought it could be.

It can then be determined that there exists another underlying, far more fundamental issue pertain to the construction of vast frameworks for interactive computer music system in this manner. There is the indubitable presence of a highly complex, extremely variable dynamic co-existing between the aesthetics of the multimedia programmer, the playing preferences of the human musician, and how those factors can come into conflict with holistically interdependent systems using a hybridization of the Player Paradigm-Free Jazz Performance Model.

The perceived success of the system's procedural features, no matter how well designed or programmed, will be determined by the way in which humans decide to interact with the virtual player. The dynamic is no different than an exclusively human improvisation: one freewheeling individual within an ensemble can alter the crucial balance existing between their assumedly malleable roles. These non-linear, complex architectures require far more structural flexible and sensitivity from the human performer than the more Orchestral Performance Models, where the player can gesturally dictates every decision within the system.

There was a clear relationship between the virtual player's effect on the control systems, producing results related to its real-time gestural decisions. However, with the additionally sonic complexity incorporated into each iteration, coupled with the development of more robust and interconnected, it became more challenging to distinguish which sound were as a direct result the human performer, the virtual performer, and the interactions between them. Thus, this system is highly reliant of the judicious musicianship of the human input. Additionally, there must be a musical coherency behind mapping large amounts of interdependent measurements from listening objects to another process within the system. This is best achieved by not combining audio measurements from two disparate inputs into order to create a hybrid control source. While it may produce subjectively interesting audio-visual music, mapping in this manner is not very effective in establishing the relationship between gestural improvisation and its resulting audio-visual output.

5.2 Future Work

Prospective experimentation and prototyping will center around extrapolating the game experience from what is a relatively simple simulation in Unity into a significantly more sophisticated, musically inclusive model in Augmented Reality. This hybrid platform facilitates a more communal experience than, for instance, Virtual Reality. While the former provides the means to provide a visual overlay to an individual's surrounding natural environment, the latter completely engulfs the auditory and visual senses, supplanting the participant into a completely foreign world. In the context of this thesis experiment, the long-term vision is to democratize the parameters that could be used to define musical expressivity in electroacoustic musical performance.The objective is not necessarily to overwhelm by supplanting the user into a visually unrelated world, but for them to realize their direct influence on significantly altering the sonic output of the performance within their communal environment.

6. REFERENCES

1. Beauchemin, S. "Chapter 2: A Rare Breed: The Audio Programmer." *Game Audio Programming 2:* Principles and Practices (pp. 35-51). Boca Raton, FL: CRC Press.

2. Borgo, D., & Goguen, J. (2005). Rivers of consciousness: The nonlinear dynamics of free jazz. In *Jazz research proceedings yearbook* (Vol. 25).

3. Böttcher, N., & Serafin, S. (2009). Design and evaluation of physically inspired models of sound effects in computer games. In *Audio Engineering Society Conference: 35th International Conference: Audio for Games*. Audio Engineering Society.

4. Böttcher, N. (2013). Current problems and future possibilities of procedural audio in computer games. Journal of Gaming & Virtual Worlds, 5(3), 215-234.

5. Clarke, E. F. (2012). Creativity in performance. Musical imaginations: Multidisciplinary perspectives on creativity, performance, and perception, 17-30.

6. Desain, P. (1992). Can computer music benefit from cognitive models of rhythm perception?. In *PROCEEDINGS OF THE INTERNATIONAL COMPUTER MUSIC CONFERENCE* (pp. 42-42). INTERNATIONAL COMPUTER MUSIC ASSOCIATION

7. Farnell, A. (2007, September). An Introduction to Procedural Audio and its Application in Computer Games. In *Audio Mostly Conference* (Vol. 23).

8. Farnell, A. (2011). Behaviour, Structure and Causality in Procedural Audio. In *Game Sound Technology and Player Interaction: Concepts and Developments* (pp. 313-339). IGI Global.

10. Hug, D. (2011). New Wine in New Skins: Sketching the Future of Game Sound Design. In *Game Sound Technology and Player Interaction: Concepts and Developments* (pp. 384-415). IGI Global.

11. Hsu, W. (2005). Using Timbre in a Computer-Based Improvisation System. In *ICMC*.

12. Collins, K. (2008). Game sound: an introduction to the history, theory, and practice of video game music and sound design. Mit Press.

13. Lewis, G. E. (2000). Too many notes: Computers, complexity and culture in voyager. *Leonardo Music Journal*, 33-39.

14. Lewis, G. E. (2013). Critical Responses to 'Theorizing Improvisation (Musically)'. *Music Theory Online*, *19*(2).

15. Lewis, G., & Piekut, B. (2016). Introduction: On Critical Improvisation Studies. In *The Oxford Handbook of Critical Improvisation Studies*. Oxford University Press.

16. Lewis, G. E. (2018). Why Do We Want Our Computers to Improvise?. *The Oxford Handbook of Algorithmic Music*, 123.

17. Liebe, M. (2013). Interactivity and music in computer games. In *Music and Game* (pp. 41-62). Springer VS, Wiesbaden.

18. Linson, A., Dobbyn, C., & Laney, R. C. (2012, May). Critical issues in evaluating freely improvising interactive music systems. In *ICCC* (pp. 145-149).

19. Phillips, W. (2014). *A Composer's Guide to Game Music*. Cambridge, MA: MIT Press.

20. Pressing, J. 1987 "The Micro- and Macrostructural Design of Improvised Music". *Music Perception* 5(2): 133-172.

21. Rowe, R. (1992). Interactive Music Systems: Machine Listening and Composing. MIT press.

22. Rowe, R. (2001). Machine musicianship. MIT press.

23. Somberg, Guy.(2016). " Sound Engine State Machine." *Game Audio Programming (pp. 35-52)* Boca Raton, FL: CRC Press.

24. Sweet, M. (2015). *Writing interactive music for video games: A composer's guide*. Upper Saddle River, NJ: Pearson Education.

25. Walder, C. "Chapter 21: Synchronizing Action-Based Gameplay to Music." *Game AudioProgramming 2: Principles and Practices* (pp. 280-296). Boca Raton, FL: CRC Press.

26. Winkler, T. (2001). Composing interactive music: techniques and ideas using Max. MIT press.