Musical Sonification of Avatar Physiologies, Virtual Flight and Gesture

Robert Hamilton[1]*

Center for Computer Research in Music and Acoustics,
Stanford University
rob@ccrma.stanford.edu

Abstract. Virtual actors moving through interactive game-space environments create rich streams of data that serve as drivers for real-time musical sonification. The paradigms of avian flight, biologically-inspired kinesthetic motion and manually-controlled avatar skeletal mesh components through inverse kinematics are used in the musical performance work *ECHO::Canyon* to control real-time synthesis-based instruments within a multi-channel sound engine. This paper discusses gestural and control methodologies as well as specific mapping schemata used to link virtual actors with musical characteristics.

Keywords: musical sonification, procedural music and games, virtual gesture

1 Introduction

From a creative musical standpoint, there have traditionally been necessary synergies between motion and action in space and the production and manipulation of musical sound. And for most pre-digital musical systems, physical gesture was an inherent component of instrumental performance practice. From the sweep of a bow across strings, to the swing of a drumstick, to the arc of a conductor's baton, action and motion in space were directly coupled as physical or intentional drivers to the mechanical production of sound and music [6].

The introduction of computer-based musical systems has removed the necessity of such direct couplings, allowing abstract data-analysis or algorithmic process to both instigate and manipulate parameters driving musical output. However artists seeking to retain some level of human-directed control within the digital context often develop and employ mapping schemata linking control data to musical form and function. Such mappings provide an interface between human intention and digital process and can range from the simple to the complex, from the distinct to the abstract.

* All environment and character modeling, custom animations and art direction for *ECHO::Canyon* were created by artist Chris Platz.

1.1 Reactive Mapping and Gesture.

Choreographies of music and action found in dance and film commonly make use of a reactive association between gesture and sound. Dancers' reactions - spontaneous or choreographed - to a musical event or sequence of events often form physical motions or gestures with direct temporal correspondence to the onset, duration or contour of a sounding event [12]. Similarly, events in static visual media such as film, music video and some computer games are often punctuated by the synchronization of visual elements with unrelated auditory or musical cues, linking the audio and visual in our perception of the event without any causal relationship existing between the two modalities.

1.2 Causal Mapping in Virtual Space.

Interactive virtual environments and the tracking of actor motion and action within those environments affords yet another approach to the mapping of physiological gesture to parameters of sound and music for multimodal presentation. As avatars within three-dimensional space are wholly-digital constructs, there exists a massive amount of data readily available that represents their internal and external state, their ongoing relationship to other objects in the surrounding environment, and the state of the environment itself. This data can drive complex dynamic musical and sound-generating systems while preserving a causal link between the visual gesture and the resultant audio gesture.

1.3 Multimodal Gesture and Motion.

With virtual actors, the contours of motion in virtual space - both macro, such as a three-dimensional Cartesian vector, or micro, such as the relative articulation of individual bones within an avatar skeletal mesh - can be tracked and used as control data for computer-based musical systems. In this manner, the gesture or motion itself drives and controls the sound-generating process, an inversion of a more common reactive model and very much in line with traditional models of instrumental performance.

By pairing macro and micro avatar motions with real-time musical sonification, composers and designers repurpose elements of model physiology and structure, as well as the topographies of virtual space itself, into components of musical gesture. Multiple modalities of interaction can then be combined to create performance works wherein the interactions between virtual actor and virtual environment drive any number of parameters of computer mediated musical sound, structure and space.

2 Musical Sonification in *ECHO::Canyon*

ECHO::Canyon (2013) by Robert Hamilton and Chris Platz is an interactive musical performance piece built within UDKOSC [10], a modified version of

Fig. 1. In *ECHO::Canyon* interactions between player-controlled flying avatars and the environment itself drive procedural sound and music generation using UDKOSC.

the Unreal Development Kit or UDK, a free-to-use version of the commercial Unreal 3 gaming engine [1]. Premiered on April 25, 2013 at Stanford University's Center for Computer Research in Music and Acoustics, *ECHO::Canyon* creates a reactive musical environment within which the idiomatic gestures and motions of flight are mapped to musical sound-producing processes.

During the piece performers control virtual actors moving through a fully-rendered outdoor landscape using a computer keyboard and mouse or commercial game-pad controller. Each actors' location and rotation in game-space, as well as other parameters describing their interactions with objects within the environment are streamed in real-time to sound-servers using the Open Sound Control (OSC) protocol [20]. The environment itself is sculpted in such a way as to allow performers the freedom to perform musical interactions by moving above, around and through the topography. In this way the process of environment design takes on the role of composition, with sections of virtual hills, canyons and valleys acting as musical pathways through the environment.

While *ECHO::Canyon* is built within a gaming engine, unlike many commercial games where audio and music play a supporting role to displays of rich visual content [21], the role of music and sound within the work are intended to occupy a perceptual role equal to the presented visual modality. Sonifications used in *ECHO::Canyon* are designed to be musical and performative in nature, and are fundamentally presented as foreground constructs, rather than as background or more associative "sound-effect" constructs. To that end traditional approaches for game sound design are replaced instead by sets of composed interactions.

[1] Unreal Development Kit by Epic Software. `http://www.udk.com`

3 Prior Work

The use of video game engines for music and sound generation has become increasingly common as generations of musicians who have grown up with readily accessible home video game systems, internet access and personal computers seek to bring together visually immersive graphical game-worlds, wide-area networks, interactive control methodologies and musical performance systems.

Though its graphical display is rendered in 2-dimensions, *small_fish* by Furukawa, Fujihata and Muench [8] is a game-like musical interface which allows players to create musical tapestries based on the interaction of dynamic components within the environment. Similarly playful in scope, *LUSH* by Choi and Wang uses models of organic interaction and gameplay within an OpenGL framework to represent and control sound generating and organizing processes [5].

Commercial gaming environments have been repurposed as dynamic music-producing systems in *Soundcraft* [4], *q3apd* [16] and *q3osc* [9]. Multi-modal musical performances built within an earlier version of UDKOSC, as well as within a customized implementation of the open-source Sirikata [18] virtual environment produced a series of immersive and interactive musical works [11]. And the mapping of game-play interactions to real-time sound generating process has been pursued as a prototyping methodology by sound designers like Leonard Paul [17], and as an immersive creative interface and display by Florent Berthaut [3].

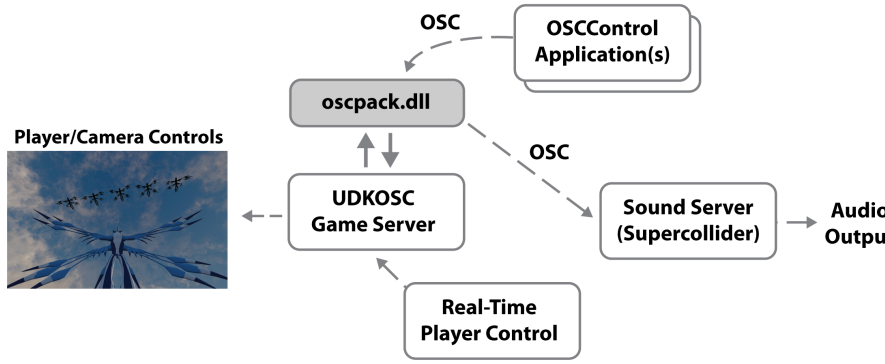

Fig. 2. UDKOSC processes OSC input to control avatar and camera motion while generating OSC output representing avatar and skeletal mesh location, rotation and action/state data.

4 System Overview

To produce musical works such as *ECHO::Canyon*, multiple software and hardware systems must efficiently share large amounts of real-time data with low

latency and a high success-rate of packet delivery. At the same time, the sound generation and three-dimensional graphics rendering are extremely taxing for even higher-end personal computer systems. To optimize both sound and video production, multiple machines are used in any one performance, connected over a local gigabit ethernet network. A sound server running SuperCollider typically runs on one computer (OS X, Linux or Windows) while the UDKOSC game-server and individual game-clients each run on their own Windows machine.

4.1 UDKOSC

UDKOSC was designed to bring together real-time procedural sound synthesis, spatialization and processing techniques from the realm of computer music with the visually immersive networked multi-player environments of commercial-grade gaming engines. Gestures, motions and actions generated by actors in game-space are analyzed and transformed in real-time into control messages for complex audio and musical software systems. UDKOSC was developed to support the creation of immersive mixed-reality performance spaces as well as to serve as a rapid prototyping tool for procedural game audio professionals [19].

While the UDK explicitly restricts developers from accessing the Unreal Engine's core C++ engine, it exposes a higher-level scripting language known as UnrealScript. UnrealScript allows developers to bind Windows Win32 .dll's to UnrealScript classes, enabling external blocks of code to interact with the scripting layer. Using this DllBind functionality, UDKOSC binds a customized version of OSCPack [2] to a series of Unrealscript classes and mirrored data structures, passing bidirectional data both into and out from the game engine. In this manner real-time game data can be streamed over UDP to any given ip-address and port combination at the same time control messages and data from external processes can be streamed into the game engine.

Actors within the UDK, or characters moving through and engaging with the environment, can be controlled by human performers - called "Pawns" - or controlled by game artificial-intelligence or pathing algorithms - called "Bots". These actors are separate entities from the "Camera", essentially a projected viewpoint within the environment which is displayed on screen. UDKOSC adds the ability for Pawns, Bots and Cameras to be controlled via commands received from externally-generated OSC messages.

Output Data. Currently, UDKOSC tracks a number of in-game parameters for each actor and exports them using OSC including:

- Pawn and Bot unique identifier within the UDK
- Pawn and Bot Cartesian location (X, Y and Z coordinates) and rotation (pitch, yaw and roll)
- Camera view rotation
- Projectile Cartesian location and collision event (triggered when the projectile touches a solid entity within the environment)
- Interp Actor, Trigger and Static mesh Cartesian location and collision events
- Location and rotation for individual bones from a Pawn's Skeletal Mesh

Input Data. UDKOSC can receive commands over OSC including:

- Pawn and Bot movement speed, direction vector and rotation
- Projectile target location
- Camera cartesian location, rotation and speed
- Camera mode: fixed location, manual rotation, first and third-person modes

4.2 Music and Sound Server.

On the receiving end of the UDKOSC output stream is a music and sound server capable of interpreting OSC messages and mapping game parameters to musical generation and control processes. While any OSC-capable system can be used as the interpreter for UDKOSC output, for most UDKOSC projects, our preference has been to use Supercollider [15] running numerous synthesis processes and spatialized across multiple channels using ambisonics [14].

Within Supercollider, data representing avatar positioning, rotation and action is mapped to specific parameters within instances of synthesized instruments. In *ECHO::Canyon*, the flight of a player-controlled "Valkordia" pawn through the environment is sonified in real-time. At any given moment of the piece each pawn's speed, rotation, absolute Z-location, height relative to the "ground", side proximity to solid environment structures and a Euclidean distance to a series of "crystal" objects in the environment all serve as parameters driving real-time synthesis. Alongside one or multiple human-controlled pawns, flocks of OSC-controlled Valkordia bots, themselves driving separate synthesis processes, are controlled with pre-composed OSC-emitting scripts. During flight as well as during a specially-designed "posing state", the location of bones in the bird-skeleton's wings are tracked and mapped to their own synthesis algorithms.

Musical output for *ECHO::Canyon* is currently spatialized across multichannel speaker systems by a Supercollider-based sound server making use of stereo output, simple 4-channel panning or first-order ambisonics as the performance space allows. When ambisonic output is used events are placed in the soundfield in a mapping schema uncommon in standard video-game audio where coordinate locations in the environment are mapped to static corrolary locations within the listeners' soundfield. When stereo or 4-channel panning is employed, location-based sound events are placed in a more conventional actor-centric perspective, with their amplitudes scaled proportionally to the distance between actor and sound-emitting location.

5 Macro and Micro Scale Gestures

While acknowledging that conversations attempting to define gesture and how that term pertains to music can engender fierce debate, for the scope of this project, gestures can perhaps be initially defined as intended actor motion or action created within game-space for the purpose of instigating or controlling visual or auditory response. Using UDKOSC, this kind of actor motion can be

tracked on a *macro* scale – essentially the gross contours of 3D motion within the environment – as well as on a *micro* scale – the location and rotation of specific bones within and around an actor skeleton. From a compositional standpoint, such an approach allows for great flexibility in the investigation of wholly different types of sounds, mapping schemata and controlled musical interactions.

For example, a dance-like series of motions ("actor sprints up a ramp, jumps, twirls and lands") or a direct mapping of human-physical gesture via a Kinect controller ("actor's skeletal mesh mimicks user swinging an arm side-to-side") can each be considered a gesture in this context. The user-directed intention of each motion, controlled or generated, combined with the sound, itself generated in real-time during the creation of the motion, becomes a significant component of the multi-modal gesture.

5.1 Macro-scale Gesture.

Interactions between user-controlled actors and some other entity, either part of the environment or another moving actor, can be categorized as macro-scale gestures. For macro-scale gestures, the scope of a given gesture generally involves the entire actor itself taken as a single-entity, tracking its location in three-dimensional space over a window of time.

Perceptually speaking, macro-scale gestures can occupy the same attentional space as routines or phrases in dance, wherein an actor performs a series of linked motions or actions to convey an intention. Gestures in this vein can encompass multiple actors, interacting with each other either as individual components of a dynamic gesture or as grain-like instances within a singular cloud or mass.

Fig. 3. Valkordia Skeletal Mesh with highlighted right wing-tip bone

Flight-based Gesture. For *ECHO::Canyon*, the theme of avian flight is central to the musical sonification, animation and control schemata created and used for the piece. A character model called a "Valkordia" was created, fusing physical characteristics and idiomatic movements from both bird and insect-like

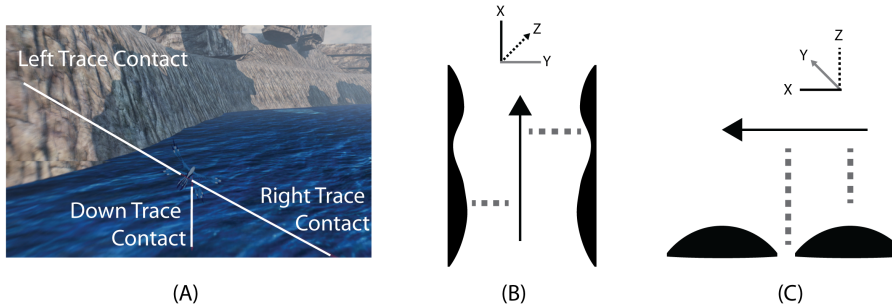

Fig. 4. Ray traces visualized as vertical and horizontal lines in (A) track the distance between the Skeletal Mesh and objects and contours of the environment, both to its right and left sides (B), as well as directly below (C).

creatures. Articulated motion of the Valkordia wings and its torso during flight can be seen in Fig. 3. Bones found in the model's front right wing (visible as a white vertical line through the first pose's middle feather) were tracked to drive a noise-generating process when the avatar was in flight.

The relation of a flying Valkordia actor to the environment was a key gestural component in the shaping of *ECHO::Canyon* both literally and figuratively. Valleys, mountains and caves were sculpted with articulated shapes to accentuate specific features of synthesis processes. Fig. 4 shows vertical and horizontal ray traces tracking the relative distance between a flying actor, the ground, and the walls of a valley. In this example, the ray trace distances were used to control amplitude and filter frequencies of separate synthesis processes, as well as panning for the horizontal traces.

Flocking Pawns. Our cognitive abilities to group and associate like motions of active objects into single cohesive units can bring disparate dynamic elements together into one unified mass gesture [13]. The sonification of such behaviors with simple sound sources can create dynamic musical textures through similar motion and position of each source [7]. *ECHO::Canyon* makes use of flocks of OSC-controlled Valkordia pawns with a relatively simple mapping of their Z-coordinate to a simple oscillator and their distance from the player actor to the oscillator's amplitude. Each bird in the flock tracks a target position which is moved in pre-composed patterns through the game-space by an OSC-generating script. The sonic result is a shifting grain-like cloud of pitched oscillators.

5.2 Micro-scale Gesture.

With the intention of drawing audience and performer attention to the actor itself and away from the environment, micro-scale gestures are comprised of motions and articulations of a given avatar's virtual physiology. By mapping the subtle motions of bones within an actor's skeletal mesh to both dynamic

control systems and evocative musical processes, the micro-scale gestures within *ECHO::Canyon* provide a vastly different viewing and listening experience than the work's macro-scale gestures.

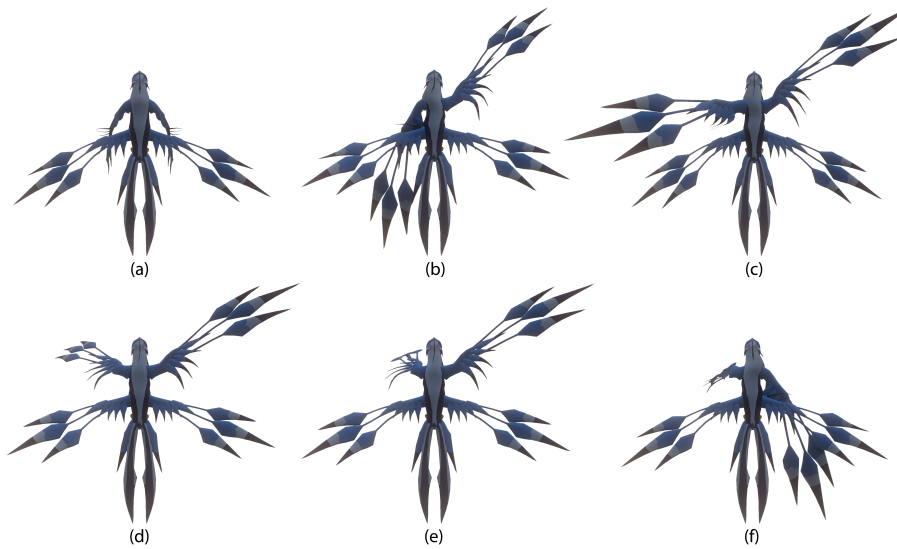

Fig. 5. Valkordia model with manual wing positioning during "Posing" state

Posing State. Performers in *ECHO::Canyon* enter the posing state by toggling a key on the game-pad. Upon entering the state, actors no longer fly through the environment; instead each avatar interpolates into a nearly vertical pose and control over the actor's front two wings are directly mapped to each of the gamepad's two two-dimensional analog joystick controls. Users control the forward, side and back rotation of each wing independently by rolling the analog joysticks around in circular patterns, mimicking the rotation of arms or wings in shoulder sockets. A series of wing poses can be seen in Fig. 5, examples (A) through (F).

Rather than mapping pre-composed wing animations to output from the joystick controllers, each wing instead tracks an end effector, using inverse kinematics [1]. The location of each effector is itself controlled in 3D space by the joystick output, scaled and acting upon a Cyclic-Coordinate Descent or CCD Skeletal Controller, itself a component of the UDK. In Fig. 6, the effectors for each wing are visualized as globes towards which the chain of bones from the tip of each wing to the shoulder socket are reaching.

Tracking Bone Location. The tracking of individual bone locations relative to a central point on the actor's skeletal mesh changes the focus and scale of

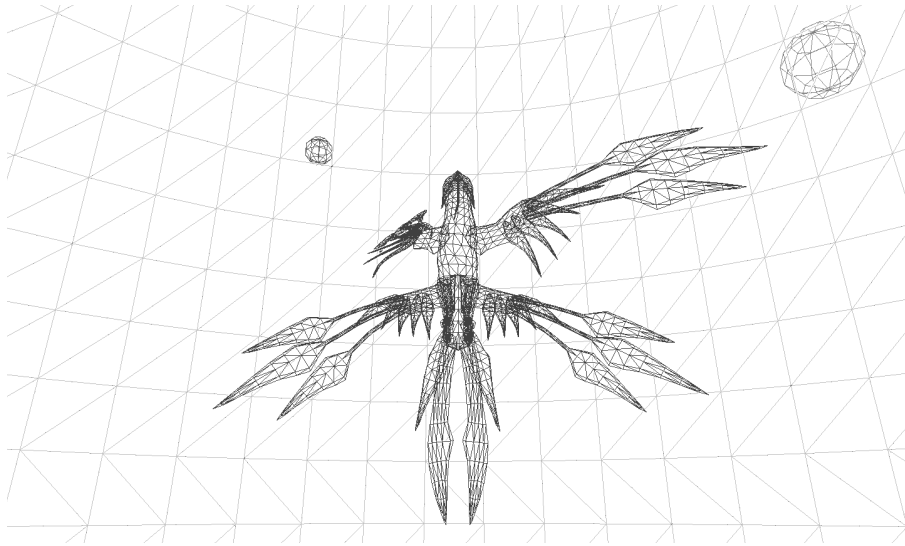

Fig. 6. Using inverse kinematics controlled by a dual-joystick gamepad, players can dynamically control Valkordia wing poses.

gestures to reside firmly in the micro-scale. In this manner, the extension of a wing to its full length can be mapped to a "larger" sounding sonic response than a "smaller" gesture, closer to the central point. Each bone that comprises a model's skeletal mesh can be tracked in UDKOSC, though due to the high number of individual bones used in many well-articulated skeletal meshes, it is generally a good idea to track a few key bones to reduce the amount of data tracked and output in real-time.

In the posing state, the relative X, Y and Z positions of one single bone located at the tip of each wing is tracked and output with OSC. In Fig. 7 data from a right wing bone is shown, first in a flying state, and then during manual control over the wing position. The peaks showing manual arm gestures are clearly visible on the right half of each plot. By comparison, the oscillating wing motion during flight shows clearly as a fairly continuous signal on the left half of each plot. It should be noted that coordinates in the Unreal Engine are measured in "Unreal Units", a unit of virtual measure where one UU corresponds roughly to 0.75 inches, or one foot = 16 UU and 1 meter = 52.5 UU.

6 Musical Sonification in *ECHO::Canyon*

The following list defines an example set of control events and actions that have been explored within *ECHO::Canyon* and a description of their musical analogues:

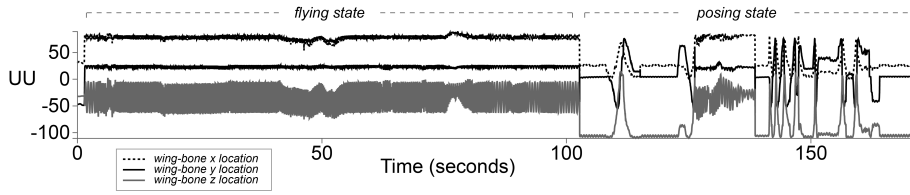

Fig. 7. Wing X,Y,Z Coordinate data in a flying state and during manually controlled wing gestures

Actor Proximity. An actor's relative distance to objects in the environment is determined through the use of horizontal and downward ray traces. The distance between the center of an actor's bounding-box and an object with which the ray trace collides is output over OSC. From a design standpoint, traces are used to drive musical processes when an actor moves through a space such as a tunnel, cave or chasm, or simply swoops down above some part of the terrain.

- In SuperCollider, horizontal ray trace distance and global location is used to modulate the amplitude, central frequency and grain count of a cloud of granulated SinOsc bursts.
- Amplitude is scaled inversely to horizontal trace distance, while grain count and central frequency are both modulated by the actor's current height, or Z-location.
- Vertical trace distance shapes both the amplitude and the chaotic oscillations of a "screetch"-like sine feedback FM oscillator with phase modulation feedback using the SinOscFB UGen.

Actor Speed and Motion. As user avatars move through three-dimensional coordinate space, each avatar's X, Y and Z location data is streamed over OSC to a sound-server (see Fig. 8). The speed of motion is calculated and used to scale the speed of the flight animation, itself driving parameters of a noise-based synthesis instrument.

- Actor speed is indirectly sonified as the speed of oscillation of the right and left wing bones drives each bone's position in the Z-plane (relative to the actor's central coordinate location).
- Location data controls simple amplitude and ambisonic spatialization of continuous sound-sources for each osc-controlled Valkordia flocking pawn.
- Actor speed also modulates the frequency of a filter shaping the output from each actor's downward trace SinOscFB process.

Actor Bone Motion. The structural core of each actor's character model is a skeletal mesh comprised of numerous connected-yet-independent bones, each one with a coordinate location and rotation accessible via OSC. By tracking motion of each bone within the skeletal mesh, complex control signals can be generated through the use of simple avatar motions.

- During flight, the relative z-location of each wing bone is sonified with a simple sine oscillator, with subtle beating frequencies made audible through a slight freqency offset between each wing's synth.
- During the manual posing state, the same mapping continues, however the manual extension of each wing causes the pitch of each oscillator to modulate within a range of approximately four-semitones.
- The frequency of an actor's manually-triggered "call" sound is mapped to the combined distance between right and left wing tip bones.

Actor-Group Motion and Density. While individual actor avatars each communicate their positions through individual OSC streams, actors moving in concert together – in flocks, swarms or herds - can be tracked and sonified as a group. For fast moving particle-based objects, like projectiles generated by an actor or actors, granular synthesis-based instruments have proven an interesting mapping. Similarly, flocks of flying avatars tracked as simple sine-waves have been used to create a shifting field of additive signals.

- Flocks of OSC or AI controlled Valkordia pawns are represented with simple sine oscillators which can be spatialized in an ambisonic soundfield.
- Projectiles "fired" by an actor generate an inexpensive pitched collision sound – a burst of noise passed through a tuned filter – when the projectile bounces off a surface in the environment. The filter frequency is mapped to the distance of the collision point from the coordinate center of the environment. As high numbers of projectiles can be quickly generated by performers, all synthesis processes used with projectiles in UDKOSC are relatively computationally cheap.
- Projectile location can also be spatialized in an ambisonic soundfield. In this case each projectile is represented by a simple sine oscillator with its freqency mapped to the calculated distance from the environment center.

Spatio-centric spatialization. In contrast to traditional gaming concepts of user-centric audio, a spatio-centric presentation superimposes a virtual space onto a physical multi-channel listening space, spatializing sound events around a physical space to correlated coordinates in the virtual space. The goal of such presentations are to immerse an audience in an imposed sound world, creating a perceptual superimposition of virtual and physical environments.

Fig. 8 traces a simple Valkordia flight path in three-dimensions. If this were presented in a rectangular concert space, the position of the displayed sound source would move along the shown trajectory.

- When ambisonic spatialization is used, each sound generated is positioned in the soundfield according to its position in game-space. Unlike traditional gaming presentations, where sounds are generally positioned relative to the player's head location, such a presentation can represent the location of multiple users and objects to an audience watching without a decided "first-person" viewpoint.

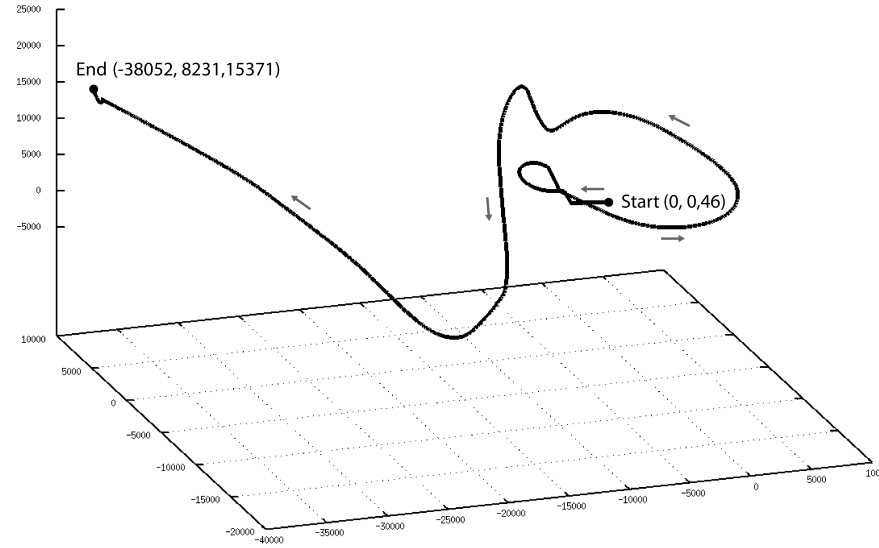

Fig. 8. Actor flight path in 3D coordinate space.

Location-Based Triggers. Specific locations in coordinate space are stored on the sound server and users' relative Cartesian distance from each location can be used as a synthesis parameter. For instance, by mapping amplitude of a sound source to such a distance measurement, an instrument's volume will fluctuate as a factor of a user's distance from a particular location.

- In Fig. 1, a number of large crystal-like structures are visible, each of which serves as a location-based sound source.
- Each crystal location drives a separate instance of the Gendy1 dynamic stochastic synthesis UGen.
- An actor's distance from each of these crystals controls both the Gendy1's amplitude as well as the frequency of a ResonZ two pole filter.

Active Triggers. For sound events or processes that require simple mode/context switching control or triggering, users can actively interact with a trigger in the form of a button or key-press. When a user fires a "use" event the trigger sends an OSC message with its name, location and trigger state.

- The most commonly used trigger of this sort in *ECHO::Canyon* is the Valkordia's "call": a layered cloud of granular pitches with frequencies randomized within a narrow range.
- The central frequency of each call is set by the actor's Z-coordinate which is itself mapped into the note range of a semitone/chromatic scale.

References

1. Aristidou, A., Lasenby, J.: Inverse Kinematics: a review of existing techniques and introduction of a new fast iterative solver. Technical Report. Cambridge (2009)
2. Bencina, R.: Oscpack. http://code.google.com/p/oscpack (2006)
3. Berthaut, F., Hachet, M., Desainte-Catherine, M.: Interacting with the 3D reactive widgets for musical performance. In: Journal of New Music Research, vol 40, no. 3, pp. 253–263 (2011)
4. Cerqueira, M., Salazar, S., Wang, G.: Soundcraft: Transducing Starcraft. In: Proceedings of the New Interfaces for Musical Expression Conference, pp. 243–247 Daiejon (2013)
5. Choi, H., Wang, G.: LUSH : An Organic Eco + Music System. In: Proceedings of the New Interfaces for Musical Expression Conference. pp. 112–115. Sydney (2010)
6. Dahl, S., Bevilacqua, F., Bresin, R., Clayton, M., Leante, L., Poggi, I., Rasamimanana, N.: Gestures In Performance. In: Musical Gestures: Sound, Movement, and Meaning. Godøy, R., Leman, M. (eds.) pp. 36–68. Routledge, New York (2010)
7. Davis, T., Karamanlis, O.: Gestural Control of Sonic Swarms: Composing with Grouped Sound Objects. In: The Proceedings of the 4th Sound and Music Computing Conference. pp. 192–195. Lefkada, Greece (2007)
8. Furukawa, K., Fujihata, M., Muench, W.: small_fish. http://hosting.zkm.de/wmuench/small_fish (2000)
9. Hamilton, R.: Q3OSC: or How I Learned to Stop Worrying and Love the Game. In: Proceedings of the International Computer Music Conference, Copenhagen (2008)
10. Hamilton, R.: Sonifying Game-Space Choreographies with UDKOSC. In: Proceedings of the New Interfaces for Musical Expression Conference. pp. 446–449, Daiejon (2013)
11. Hamilton, R., Caceres, J., Nanou, C., Platz. C: Multi-modal musical environments for mixed-reality performance. pp. 147–156. JMUI, Vol. 4, Issue 3–4, Springer-Verlang, Heidelberg (2011)
12. Jensenius, A., Wanderley, M., Godøy, R., Leman, M.: Musical Gestures: Concepts and Methods of Research. In: Musical Gestures: Sound, Movement, and Meaning. Godøy, R., Leman, M. (eds.) p. 13. Routledge, New York (2010)
13. Lehar, S.: Gestalt Isomorphism and the Primacy of Subjective Conscious Experience: A Gestalt Bubble Model, Behavioral & Brain Sciences, vol. 26;4, pp 375–444, March, Cambridge University Press (2004)
14. Malham, D., Myatt, A.: 3-D Sound Spatialization using Ambisonic Techniques. Computer Music Journal, 19;4, pp 58–70, Winter (1995)
15. McCarthy, J.: SuperCollider, http://supercollider.sourceforge.net
16. Oliver, J.: q3apd. http://www.selectparks.net/archive/q3apd.htm (2008)
17. Paul, L.: Video Game Audio Prototyping with Half Life 2. In: Adams, R., Gibson, S., Mller Arisona, S. (eds.) pp. 187–198. Transdisciplinary Digital Art, CCIS, Vol. 7, Springer, Heidelberg (2008)
18. Sirikata. http://www.sirikata.com (2009)
19. Verron, C., Drettakis, G.: Procedural audio modeling for particle-based environmental effects. In: Proceedings of the 133rd AES Convention. San Francisco (2012)
20. Wright, M., Freed, A.: Open Sound Control: A New Protocol for Communicating with Sound Synthesizers. In: Proceedings of the International Computer Music Conference. Thessaloniki (1997)
21. Zehnder, S., Lipscomb, S.: The Role of Music In Video Games. In: Playing Video Games: Motives, Responses, and Consequences. Vorderer, P., Bryant, J. (eds.) pp. 282–303. Lawrence Erlbaum Associates, Routledge, New York (2009)