Building Interactive Networked Musical Environments Using q3osc

Robert Hamilton[1]

[1]*Center for Computer Research in Music and Acoustics, Stanford University, Stanford, CA, 94305, USA*

Correspondence should be addressed to Robert Hamilton (rob@ccrma.stanford.edu)

ABSTRACT

Interactive networked musical gaming environments designed as control systems for external music and sound programming languages can be built using the *q3osc* Quake III/ioquake3 gaming mod. Bi-directional support for the Open Sound Control (OSC) messaging protocol compiled into the game engine allows for the real-time tracking, sonification, spatialization and third-party control of game entities, clients and environmental parameters. Reactive audio environments ranging from abstract multi-user musical performance spaces to representative acoustic models of physical space can be constructed using either a standard user-centric audio perspective or a potentially more immersive and inclusive space-centric perspective. Issues of space and perspective are discussed as related to the distribution of performance space and sonified environment across both local and wide-area networks.

1. INTRODUCTION

Music and the environments in which music is realized are intrinsically coupled. Sound in physical space is itself shaped by the spectral characteristics of the room or environment in which it is projected, with individual frequencies amplified or attuned by the physical resonant characteristics of that given space. The basic tenets of physics dictate the details of this "call and response" relationship and as such, artists and engineers alike have devised performative and analytic methodologies for developing a comprehensive understanding of and expressive control over sound in space.

One key component of intentional musical sound is often the physical gesture or action, such as a finger plucking a string or a stick striking a tuned membrane. The level of musicality inherent in physical gesture can be seen as linked to the learned performative methodologies understood and applied by the performing musician. In this manner, gesture and physical action can be viewed as events driving the propagation and eventual modification of sound in and by the surrounding physical space.

With the introduction of increasingly realistic computer-based virtual environments and the continuing drive - often focused in the commercial gaming industry - to enhance the immersive and enactive nature of such environments, traditional metholologies for shaping sound in space have been modified and augmented. While physics-based models of sound and sound's interaction with physical construct and environment can be applied in the virtual realm - often with the intent of creating "life-like" virtual sound environments - there also exists the opportunity to create novel interactions and relationships between virtual action and sound without the constraints inherent in a "real-world"-based physical environment.

As abstractions of the physical world, virtual environments offer musicians and researchers canvasses upon which pointedly un-natural sonic and musical characteristics can be made evident. Just as the laws of physics can be set aside within a virtual space, so too can traditional methodologies of coupling gesture to sound within space be set aside. By viewing a virtual environment not as a filter for events occurring within its bounds but instead as an enactive instrument in its own right, able to react to gesture and action with an audible result, traditional relationships between gesture, sound and space can be re-contextualized and up-ended. And by presenting sound events such that their spatial orientation in the virtual environment is audible in physical space, the aural focus of the projected output can be shifted from a common stereo, binaural or surround user-centric presentation to one representative of the physical sounding space itself - in a sense space-centric.

The integration of networking across both local and wide-area networks adds another dimension to the coupling between sound and space, allowing for the expansion of virtual space and its analog corollary sound field over vast distances across highly disparate locations. With multiple local and geographically-removed clients adding motion and gesture into a single virtual environment, from which data is in turn sonified by sound-servers in multiple geographically-removed locations, the notion of an extended audible space can be explored. Musical and sonic events triggered by performers are distributed across the networked sound-field, allowing for models of sonification in which sound events in separate geographies are correlated to aspects or areas of one shared virtual space.

It is these paradigms of interrelated gesture, sound and space that q3osc explores, bringing together virtual space and gesture to form a reactive cross-dimensional sound world correlating gesture and event in virtual space with spatialized sound in physical space. Motions and actions of client avatars and of projectiles generated by each avatar can be tracked in three-dimensional space and sent as control messages from the game-engine to OSC client software capable of sonifying such data. By creating a bridge of control-data from the virtual environment to sound-generating and spatialization processes in the physical world, hybrid sound-spaces can be created, representing characteristics from one world in the next.

2. Q3OSC

q3osc is a modification of the ioquake3 open-source gaming engine [5] which in turn traces its roots to the final 1.32c commercial release of Id Software's 1999 Quake III game [4]. First open-sourced in 2005 under the GPLv2 General Public License following the game developer's move towards a new engine architecture, the ioquake3 code-base has been maintained and augmented by a team of volunteer developers, patching existing issues and adding functionality ranging from IPv6 networking to Ogg Vorbis audio playback.

To create virtual environments capable of communicating gesture and motion to an external sound-generating system, q3osc makes use of the OSCPack [9] C++ implementation of the OSC protocol [14], linking in-game parameters and events to OSC output calls. Original Quake III functionalities such as virtual weapon systems and the behaviors attributed to weapon projectiles have been customized, repurposing previously violent game-entities into a responsive musical control system.

q3osc acts as a customizable control system for any software or hardware system capable of receiving UDP packets over local or wide-area networks and able to parse OSC messages. q3osc can transmit either individual OSC messages (single "/"-delimited strings containing hierarchical control messages) or OSC bundles (arrays of key:value pairs) to up to 20 individual IP address/Port combinations. At this time, q3osc has been used primarily to drive customized sonification and spatialization applications written in musical programming languages such as ChucK [13], SuperCollider [8], Max/MSP[7] and Pure Data [10]. q3osc can also receive incoming OSC messages, the values of which are mapped to specific environmental parameters such as virtual gravity, entity speed and projectile speed.

Like Quake III and ioquake3, q3osc makes use of a client-server configuration for the game-engine, where an instance of the game server coordinates multiple clients connected over local or wide-area networks. q3osc however can route its OSC output to multiple sound-servers, here defined as machines running instances of OSC receiving software capable of sonification and spatialization. If necessary, a q3osc game-server can be run on the same machine as the sound-server and can currently be compiled and run on Linux or Apple OS X operating systems.

Virtual environments can be designed and rendered using standard Quake III map-building tools such as the GtkRadiant toolkit, a level design program also developed and open-sourced by Id Software. The engine's ability to display customized graphic and model entities give developers and composers a high level of flexibility in designing virtual environments. More complex geometrical client and environment models can be built and imported into q3osc using 3D graphics softwares such as the open-source Blender project [1].

A more complete description of the capabilities and development of q3osc can be found in [3]

3. SONIFICATION

The development of schemes for data mapping and sonification in q3osc are by design left to the composer or designer and as such, each data stream can be used to represent any number of musical or control parameters. q3osc exposes a number of client and environmental parameters within the game engine for sonification. Whether actions within the virtual environment are performed by a single or multiple clients, the game server

outputs OSC streams tagged for each individual client. In this manner, gestures, actions and fired-projectiles can all be attributed to individual performers within a given virtual space.

3.1. Client Motion in Coordinate Space

Environments created for Quake III-based games are built on a three-dimensional coordinate grid and the position of any entity or client that exists within the game server can be defined as a coordinate location within that grid. Motion of game clients in virtual space is captured as X,Y, Z coordinate sets and streamed from the game server to specified sound servers. In this manner, client positioning, directional velocity and distance from specified positions in the game environment itself can be used as drivers for sonification. Additional client parameters relating to physical motion such as view-angle, state flags such as ”jumping” or ”crouching”, and selected ”weapon” can also be used as controls.

3.1.1. *Maps & Legends*

In the immersive multi-channel work *Maps & Legends* (2006) [2], client position is utilized as the primary driver for controlling and modifying audio output. The visual environment is designed such that regions marked with bright yellow circles act as triggers for pre-recorded 8-channel audio files whose relative amplitudes in 8 physical speakers is determined by the triggering-client's three-dimensional Euclidean distance from 8 representative locations in the virtual environment. Between these circular trigger-points lie visible pathways raised slightly from the base environment floor, upon which performers are encouraged to move. By following these paths, performers' vertical coordinates are used to control parameters in a number of signal-processing effects including depth of chorus and a general room-size for reverb. Sonification in *Maps & Legends* is realized using Pure Data.

3.2. Projectile Behaviors

During the development of q3osc, a number of modifications have been made to the underlying game engine including the introduction of new models of projectile behavior.

Projectiles originally intended to be fired as weapons and to subsequently terminate themselves after a specified time threshold were given the ability to persist for indefinite periods of time or until a user-generated keystroke triggers their destruction. To take full advantage of persistent projectiles, reflective or bouncing behaviors were added as well, allowing projectiles to change direction upon a noted collision with visible rendered constructs in the virtual environment. Upon the detection of a projectile collision, an OSC message is generated signifying the collision or bounce which can itself be used as a sonifiable event.

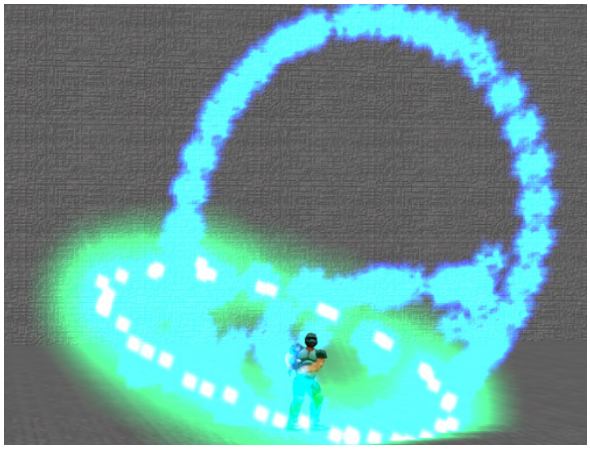

Fig. 1: Homing projectiles surrounding a client avatar.

Projectiles were also given the ability to scan the environment for the closest user avatar and dynamically change their direction to follow that avatar, creating a homing behavior. In this manner, users are given a significant amount of control over the paths of projectile motion after they have been initially fired or generated.

3.2.1. *nous sommes tous Fernando...*

Projectile behaviors are the sole driver of sonification for the interactive improvisational work *nous sommes tous Fernando...* (2008), written for and performed by the Stanford Laptop Orchestra (SLOrk) [12]. Collision events generated by the bouncing of projectiles trigger the firing of sound impulses into a bank of tuned filters across an array of sixteen ChucK sound-servers. The filters themselves are tuned to frequencies based on the X and Y coordinates of the collision event. Resonant characteristics for each filter are then mapped to the Z coordinate of the collision event, creating a responsive percussive tonal environment where sixteen individual users can improvise together simply by firing projectiles at surfaces in the environment. Performers are given the choice of two sets of projectiles, each with a different frequency range and range of filter coefficients.

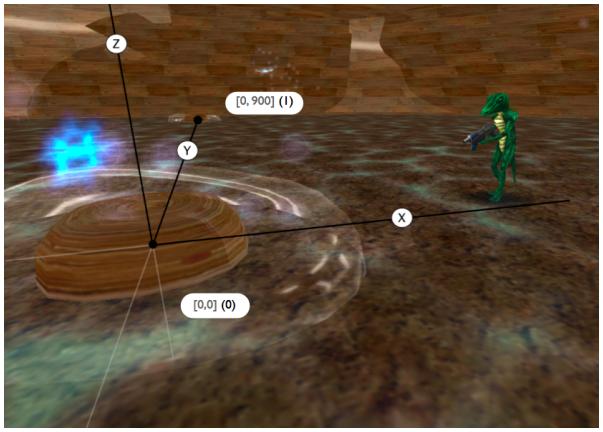

Fig. 2: Client, projectile and speaker construct with coordinate overlay in *nous sommes tous Fernando....*

3.2.2. *...and thus open a window on the world*

In the installation piece *...and thus open a window on the world* (2009), continuous coordinate positions of homing projectiles are tracked to build controllable sound masses, sonified using a sound-server written in the SuperCollider language. By instantiating individual oscillator-based Synth objects for each persisting homing projectile, groups of projectiles essentially move in swarms, tracking a given user across the virtual environment, creating spherical and circular patterns or orbits as performers stand still. As each projectile's location and correspondingly its spatial coordinates change, the frequencies and characteristics of each oscillator are changed in kind, creating a constantly shifting sound mass. As the game engine can currently support hundreds of simultaneous projectile objects, the masses of sound generated through this technique can become quite complex, both visually and aurally.

4. SPATIALIZATION AND PERSPECTIVE

By employing different models of spatialization, designers can create musical environments which emphasize as the central focus point of aural attention either the user or the space. By mapping events in virtual coordinate space to stereo or multi-channel audio output, sound worlds of varying perspective can be linked to q3osc rendered visual environments.

4.1. User-Centric Perspective

Video-games designed to present motion through space from a "first-person" vantage point traditionally align sounds in virtual space to correspond with a real-world based model of hearing; the game is projected as a parallel lifelike reality, mimicking standard alignments of human visual and aural perception. To a game-player wearing headphones or within a stereo, 5.1 or similar sound-field, this would be made manifest as an aural model where the amplitude, filtering and placement of sounds in the space are designed to make the user feel as if they themselves are physically within the presented virtual environment.

As an example, a user moving forward through such an environment would always hear the sound of their own footsteps as a near-field sound source. If another user moved from a point on the user's perceived "right" side, across their field of vision to stand on their "left" side, the sound of the second user's footsteps would perceptually appear to move in a similar direction to a similar location; the user's visual and aural environments are matched. An immersive audio environment of this kind augments the user's view of their place in the virtual world and works with the video projection to create a realistic experiential model.

Such a user-centric perspective is relatively easy to sonify by tracking the coordinate distance between event locations and the user's in-game avatar. For these purposes, in stereo versions of *nous sommes tous fernando...* a simple three-dimensional Euclidean distance measurement has been used as the base calculation of amplitude for two-channel audio streams in ChucK.

4.2. Space-Centric Perspective

By shifting the focus of sound objects away from a single user's perspective towards a focus on the placement of sound in a correlated model of virtual and physical space, designers can create a space-centric aural representation of virtual space. In a physical space such as a concert hall or a sound studio, by configuring an immersive audio environment such as an 8, 16 or 32 channel listening space, specific speaker locations in physical space can be linked to matching virtual speaker locations within the virtual environment. Control events occurring in the virtual environment are then sonified and spatialized across the multi-channel sound field, creating a spatial mapping between coordinate locations within the virtual space and the corresponding coordinate locations in physical space.

In such a model, sounds are no longer contextualized based on their proximity and relationship to a given user. Instead sounds are scaled and processed to match

their implied position in both virtual and physical space. Such presentations allow for multiple users as well as audiences to share in a communal sound world.

4.2.1. Immersive Sound Fields

In *Maps & Legends*, speaker locations in physical space were roughly matched to virtual speaker locations in the game environment. As part of the visual aesthetic of the work, large in-game constructs texture-mapped with images of the same Alesis monitor speakers used in the physical performance space were used as points of reference for performers. Euclidean distance measurements were made from performer avatars to each of the eight speaker constructs to drive each channel's respective gain.

A similar strategy was employed for a demonstration environment modeled after the Stanford University Center for Computer Research in Music and Acoustics (CCRMA) heptagonal multi-channel Listening Room. Users of the system sitting in the center of an eight-channel sound field controlled their game avatar through a virtual model designed to replicate the dimensions and structure of the Listening Room. Sounds triggered from positions within the virtual space were represented in the physical sound space by tracking user coordinate position and using the previously mentioned distance function to control gain on each speaker channel.

In the previously mentioned examples, a crude mapping between Euclidean distance and speaker amplitude was sufficient to create an identifiable if not acoustically realistic correlation between locations in virtual and physical space. Current efforts are underway to make use of Ambisonic encoding and decoding [6], Vector-based Amplitude Panning (VBAP) [11], and more realistic models of reverberation and filtering, with a hope of creating more controllable and acoustically realistic sound-fields.

4.2.2. Point-source Speaker Arrays

During collaborative work using q3osc with the Stanford Laptop Orchestra, an entirely different sound-field model was put to use, generated by sixteen six-channel hemispherical speaker arrays with each speaker array connected to one of twenty networked laptops running both q3osc client software and an instance of a ChucK sound-server. In this model, each laptop's ChucK sound-server was launched with parameters setting it's three-dimensional coordinate location in virtual space as well as an effective radius value. Events occurring within

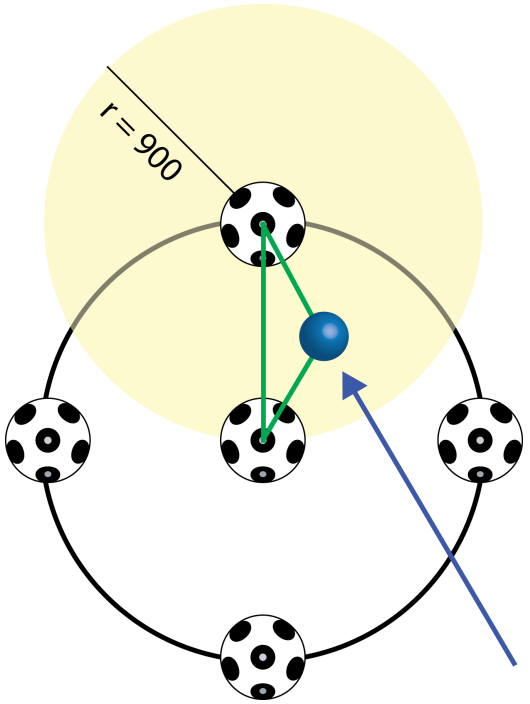

Fig. 3: Euclidean distance measured from projectile to speaker, with radius value.

the given radius of a given speaker array would cause a distance calculation to be made, scaling the gain on the specified speaker-array for the sounding event. Hemispherical constructs were built in the virtual space in the same pattern as the physical speaker arrays were laid out in physical space, created a direct correlation between the generation of sound events in the physical space and the generation of events (in this case projectile collisions) in the virtual space.

4.3. Networked Models of Spatialization

Building on the Quake III engine's inherent network capabilities, q3osc allows performers in multiple physical locations to connect and share OSC control data. As q3osc performers each connect their standard ioquake3 game client to a q3osc-enabled game server, each client's individual ip address and OSC listening port can be entered into the server, causing q3osc to stream control data back to the client's machine for local sonification.

In this same manner, not only can game clients from multiple locations connect to a central performance sound server (such as during a concert presentation in a multi-channel hall), but multiple sound servers in multiple lo-

cations can also each sonify part of a virtual environment, creating a de-centralized and extended audio realization of the virtual space. If each of those sound-servers is driving a multi-channel space-centric environment, then geographically separated concert spaces can each sonify the same sets of data in either the same or widely different ways, creating a multi-dimensional and constantly polyvalent presentation of each performer's musical intent.

5. ACKNOWLEDGEMENTS

The author is grateful for the assistance of the Stanford Laptop Orchestra, Chris Chafe, Ge Wang, and Chryssie Nanou.

6. REFERENCES

[1] Blender, http://www.blender.org, *as viewed 11/2008.*

[2] Hamilton, R. "maps and legends: FPS-Based Interfaces for Composition and Immersive Performance" In *Proceedings of the International Computer Music Conference.*, Copenhagen, Denmark, 2007.

[3] Hamilton, R. "q3osc: Or How I Learned To Stop Worrying And Love The ~~Bomb~~ Game" In *Proceedings of the International Computer Music Conference.*, Belfast, Ireland, August 24-29, 2008.

[4] id Software, http://www.idsoftware.com, *as viewed 11/2008.*

[5] ioquake3 Project Page, http://www.ioquake3.org, *as viewed 11/2008.*

[6] Mahlam, D. and A, Myatt, "3-D Sound Spatialization using Ambisonic Techniques" Computer Music Journal, 19;4, pp 58-70, Winter 1995.

[7] "Max/MSP", Cycling '74, http://www.cycling74.com, *as viewed 11/2008*.

[8] McCarthy, J. "SuperCollider", http://supercollider.sourceforge.net, *as viewed 11/2008*

[9] "OscPack–a simple C++ OSC Packet Manipulation Library", Ross Bencina, http://www.audiomulch.com/ rossb/code/oscpack/, *as viewed 11/2008*

[10] Puckette, M. 1996. "Pure Data." In *Proceedings of the International Computer Music Conference.* San Francisco, 1996, pp. 269-272.

[11] Pulkki.V. "Virtual sound source positioning using vector based amplitude panning." Journal of the Audio Engineering Society, 45(6), June 1997, 456-466.

[12] "SLOrk: Stanford Laptop Orchestra", http://slork.stanford.edu, *as viewed 11/2008*.

[13] Wang, G. and P. R. Cook. "ChucK: A Concurrant and On-the-fly Audio Programming Language" In *Proceedings of the International Computer Music Conference.*, Singapore, 2003.

[14] Wright, M. and A. Freed. "Open Sound Control: A New Protocol for Communicating with Sound Synthesizers" In *Proceedings of the International Computer Music Conference.*, Thessaloniki, Greece, 1997.