Designing and Controlling Virtual Sonic Environments Using a Browser-based 3DUI

Anıl Çamcı*, Paul Murray† and Angus Graeme Forbes‡
Electronic Visualization Laboratory, University of Illinois at Chicago

Abstract

Current authoring interfaces for processing audio in 3D environments are limited by a lack of specialized tools for 3D audio, separate editing and rendering modes, and platform-dependency. To address these limitations, we introduce a novel browser-based user interface that makes it possible to control the binaural projection of a dynamic 3D auditory scene. Specifically, our interface enables a highly detailed bottom-up construction of virtual sonic environments by offering tools to populate navigable sound fields at various scales (i.e. from sound cones to 3D sound objects to sound zones). Using modern web technologies, such as WebGL and Web Audio, and adopting responsive design principles, we developed a cross-platform UI that can operate on both personal computers and tablets. This enables our system to be used for a variety of mixed reality applications, including those where users can simultaneously manipulate and experience 3D sonic environments.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Audio input/output H.5.2 [Information Interfaces and Presentation]: User Interfaces—GUI

1 Introduction

A range of platforms facilitate the design of virtual environments. Most commonly, game engines, such as Unity and Unreal, are used for developing and simulating virtual realities. However, such platforms are primarily oriented towards visual design and provide only limited audio functionality. Making use of existing research into the development of interactive virtual soundscapes [3], we introduce a novel user interface that enables the rapid design of both virtual sonic environments, and the assets (i.e. sound objects and sound zones) contained within them. Specifically our UI offers the following contributions:

- provides a 3D design environment specific to sonic virtual realities, with specialized components such as sound objects and sound zones;
- offers both interactive and parametric control over the attributes of such components, enabling a precise authoring of highly-detailed environments;
- introduces a multi-cone model for creating 3D sound objects with complex propagation characteristics;
- enables adding dynamism to objects via hand drawn motion trajectories that can be edited in 3D;
- makes it possible to design virtual sound fields at various scales using multiple view and attribute windows;
- offers a unified interface for the design and the simulation of such realities, allowing the user to modify a sound field in real-time;
- operates on the web-browser so that it supports mobile devices, which therefore makes it possible for the user to simultaneously explore and edit augmented sonic realities.

*e-mail: acamci@uic.edu; web: http://anilcamci.com
†e-mail: pmurra5@uic.edu
‡e-mail: aforbes@uic.edu; web: http://www.evl.uic.edu/creativecoding/

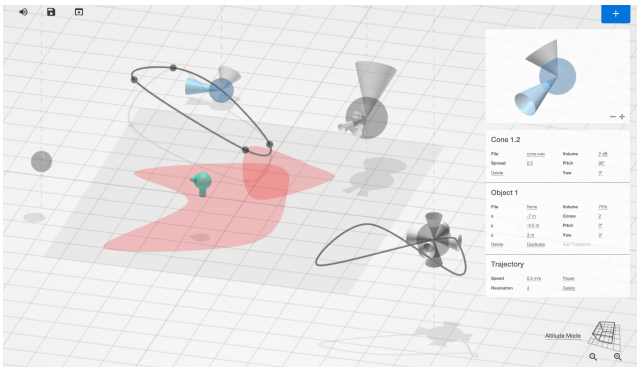

Figure 1: A screenshot of our user interface on a desktop computer displaying an object with two cones and a motion trajectory being edited. On the top right region, a close-up window displays the object with the cone that is currently being interacted with highlighted in blue. The windows below this close-up allow the user to control various attributes of the cone, the parent object, and its trajectory. Two overlapping sound zones are visualized with red polygons. A gray square represents the room overlay. The user is represented with a green dummy head.

2 Related Work

The use of sound in VR dates back to the earliest implementations in this field [4]. Many studies have since emphasized the role of sound in enhancing the immersive capacity of virtual environments [2].

Cross-platform game engines offer basic audio functionality, such as point sources and reverberant zones. These objects are created and manipulated through the same interactions used for visual objects within these environments. Third-party developers design plug-ins to extend the audio capabilities of these engines with such features as occlusion, binaural audio, and Ambisonics. However, these extensions act within the UI framework of the parent engine and force the designer to use object types originally meant to describe graphical objects, which can limit the expressiveness of a sound designer.

Other companies specialize in combined hardware and software VR solutions. *WorldViz*, for instance, offers an "Ambisonic Auralizer" consisting of a 24-channel sound system, which can be controlled with Python scripts using their VR design platform called Vizard[1]. Although their tools have powerful spatializing capabilities, no user interfaces exist for creating sonic environments using them.

[1]http://www.worldviz.com/products/vizard

3 Sound Field

3.1 Interaction

The *sound field* is the sonic canvas onto which the user can place a variety of components, such as sound objects and sound zones. In the default state, the sound field is represented by a 2D overhead-view of an infinite plane. With a *click&drag* action[2], the user can pan the visible area of the sound field. *Zoom* icons found on the bottom right corner allows the user to zoom in and out of the sound field. A UI element on the bottom right corner of the screen allows the user to tilt and rotate the view of the sound field.

A global mute button on the top left corner allows the user to turn off the entire audio output. This feature makes it possible to make offline editions to the sound field. Furthermore, with dedicated icons found adjacent to the mute button, the user can save and load UI states to restore a previously designed sound field.

3.2 Navigating the Virtual Sonic Environment

The user can explore the virtual sonic environment in via one of two modalities (or a combination of both of them). In *virtual navigation*, a stationary user is equipped with a headphone connected to the device running the UI. Depending on the input device, the user can either use physical or virtual arrow keys to travel within the sound field. In *augmented navigation*, the user moves physically within a room that is equipped with a motion-tracking system. User's gaze direction is broadcasted to the UI via OSC to update the position and the orientation of the Web Audio's *Listener Node*, which effectively controls the binaural rendering of the auditory scene based on the user's movements.

4 Sound objects

4.1 Multi-cone implementation

Directionality in game audio can be achieved using sound cones. A common implementation for this consists of two cones [1]. An inner cone plays back the original sound file, which becomes audible when the user's position falls within the projection field of the cone. An outer cone, which is often larger, defines an extended region in which the user hears a attenuated version of the same file. This avoids unnatural transitions in sound levels, and allows a directional sound object to fade in and out of the audible space.

However, sound producing events in nature are much more complex. Parts of a single resonating body can produce sounds with different directionality, spread, and throw characteristics. With a traditional sound cone implementation, the user can generate multiple cones and affix them to the same point to emulate this behavior, but from a UI perspective this quickly gets cumbersome to design and maintain. In our UI, we have implemented a multi-cone sound object that allows the user to easily attach an arbitrary number of right circular cones to a single object, and manipulate them.

4.2 Interaction

After pressing the "plus" icon on the top right corner of the UI, the user can *click* a point in the sound field to place a new sound object. The default object is an ear-level omnidirectional point source represented by a translucent sphere on the sound field.

Creating a new object, or interacting with an existing object, brings up an attributes window on the top right region of the screen. On tablets, the same interaction zooms the sound field view onto the selected object, and blacks out other components on the field. In this view, the user can interact with the sound object locally and edit its attributes. On desktop computers with sufficient screen size, the same action brings up a secondary window above the attributes window, which displays a similar close-up view of the sound object. The sound field view remains unchanged providing the user contextual control over the object that is being edited as seen in 1.

[2]On mobile devices *click* is replaced by *touch* actions.

In each case, the close-up view allows the user to add or remove sound cones and position them at different pitch and yaw values. The latter is achieved by *click&dragging* a cone using an arcball interface. Interacting with a cone brings up a secondary attributes window for local parameters, where the user can attach a sound file to a cone, as well as control the cone's base radius and lateral height values. The base radius controls the projective spread of a sound file within the sound field, while the height of a cone determines its volume. These attributes effectively determine the spatial reach of a particular sound cone. The secondary attributes window also provides parametric control over pitch and yaw values.

4.3 Trajectories

After clicking the *Add Trajectory* button in the object attributes window, the user can *click&drag* the said object to draw a motion trajectory. Once the action is completed the object will begin to loop this trajectory using either back-and-forth or circular motion depending on whether the trajectory is closed or not. Once a trajectory has been defined, a trajectory attributes window allow the user to pause, play, change motion speed in either direction or delete the trajectory. A resolution attribute allows the user the change the number of control points that define the polynomial segments of a trajectory curve. Once the user clicks on an object or its trajectory, these control points become visible and can be repositioned in 3D.

5 Sound Zones

For ambient or internal (i.e. self-produced) sounds, we have implemented the sound zone component, which demarcates areas of non-directional and omnipresent sounds. Once the user walks into a sound zone, he or she will hear the source file attached to the zone without distance or localization cues.

5.1 Interaction

After clicking the plus icon on the top right corner, the user can draw a zone of arbitrary size and shape within the sound field with a *click&drag* action. Once the action is completed, the UI generates a closed spline curve by interpolating between action start and stop positions. When a new zone is drawn, or after an existing zone is *clicked*, a window appears on the top right region of the screen to display zone attributes, which include audio source, volume, scale, rotation and resolution.

6 Conclusions

In this poster, we introduced a novel user interface to control the 3D projection of sonic virtual realities. Our UI provides an easy-to-use environment to construct highly-detailed scenes with components that are specialized for audio. It offers such features as unified editing and navigation capabilities, web-based cross-platform operation on mobile and desktop devices, ability to design complex sound objects and sound zones with dynamic attributes that can be controlled parametrically using secondary attribute windows, and multiple viewports to simplify 3D navigation. As a result, our UI provides new practical and creative possibilities for designing and experiencing sonic virtual environments.

References

[1] P. Adenot and C. Wilson. Web Audio API, 2015. [Online]. Available: http://webaudio.github.io/web-audio-api/.

[2] D. R. Begault. *3-D Sound for Virtual Reality and Multimedia*. Academic Press Professional, Inc., San Diego, CA, USA, 1994.

[3] A. Çamcı, Z. Özcan, and D. Pehlevan. Interactive virtual soundscapes: a research report. In *Proceedings of the 41st International Computer Music Conference*, pages 163–169, 2015.

[4] M. L. Heilig. Stereoscopic-television apparatus for individual use, October 1960. US Patent #2,955,156.