OpenSoundEdit: An Interactive Visualization and Editing Framework for Timbral Resources

Amar Chaudhary[1] Adrian Freed[1] Lawrence A. Rowe[2]

[1] Center for New Music and Audio Technology, 1750 Arch Street
[2] Department of Electrical Engineering and Computer Science, Soda Hall
University of California, Berkeley, CA 94720
{amar,adrian}@cnmat.berkeley.edu, rowe@cs.berkeley.edu

OpenSoundEdit is a sound-editing system that provides a unified three-dimensional user interface to edit complex sounds composed from different representations. Sounds are mapped onto a single three-dimensional space representing time, amplitude and a view-dependent third dimension (e.g., frequency, channel number, etc.) Sounds can be edited by direct manipulation on the 3D display window or using additional representation-specific controls. As the user edits sounds, changes can be heard through a synthesis server, an application that generates sound in response to commands supplied by OSE. OSE is implemented using extensible, multi-platform technologies. This paper illustrates the features of OSE with sinusoidal-track and resonance models.

1. Introduction

The emergence of real-time audio synthesis on desktop computer systems is providing musicians and sound designers with richer and more complex control over sounds. *OpenSoundEdit*, or OSE, is a sound-editing system that provides a unified three-dimensional user interface to edit complex sounds composed from different representations. In OSE, sounds are mapped onto a three-dimensional space representing time, amplitude and a view-dependent third dimension (e.g., frequency, channel number, etc.). Sounds are edited by direct manipulation on the 3D display window, or using additional representation-specific controls. As the user edits sounds, changes can be heard through a *synthesis server*, an application that generates sound in response to data and control messages sent by OSE during user operations. Servers can run on the same machine as OSE, or on another host in a network.

This paper describes the novel sound-editing features of OSE. Section 2 illustrates the use of OSE with sinusoidal-track and resonance models, and section 3 describes the implementation. A more detailed report on the design and implementation of OSE is available elsewhere [C98].

2. Features of OpenSoundEdit

In OSE, each sound representation is associated with a *view*, or a mapping of the sound onto the three dimensional space. OSE currently supports views for time-varying sinusoids and resonances. It can be easily extended to handle other sound representations as well. All views are presented in the same 3D display window, and aligned along their time and amplitude axes. A ground plane, horizon and coordinate axes are included to orient the user in this display.

The OSE user interface is illustrated in Figure 2.1. Each view contains a coordinate system drawn on the ground plane, a visualization of the data set (i.e., the yellow lines in Figure 2.1), one or more moveable frames (outlined with white bars), and 2D projections of the data set onto the frames (the black lines). The data set visualizations will be described in detail in the following subsections. The moveable frames are used for marking a position on one of the axes. For example, the ResonanceEditor view in Figure 2.1 has frames perpendicular to the time and amplitude axes. The frame in the frequency-amplitude plane can be moved along the time axis. The projected black lines will change to reflect the amplitude of the partials in the resonance model at the selected time.

OSE supports editing of sounds by direct manipulation on the 3D display window, or using additional representation-specific controls, as shown at the bottom of Figure 2.1. The behavior of these edit controls are specific to the type of model being viewed.

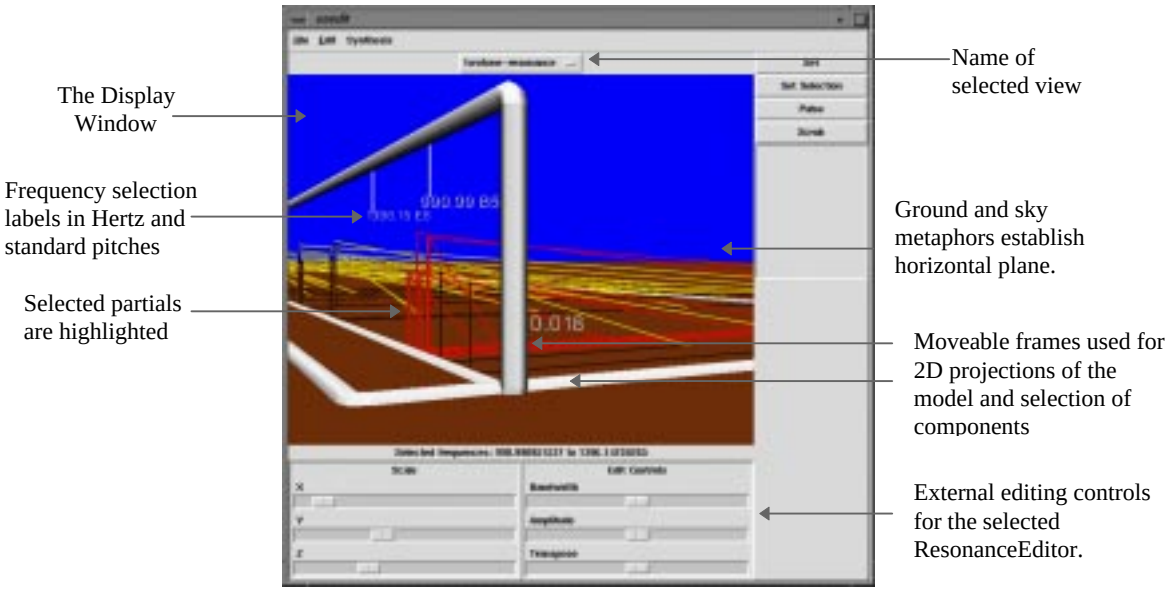


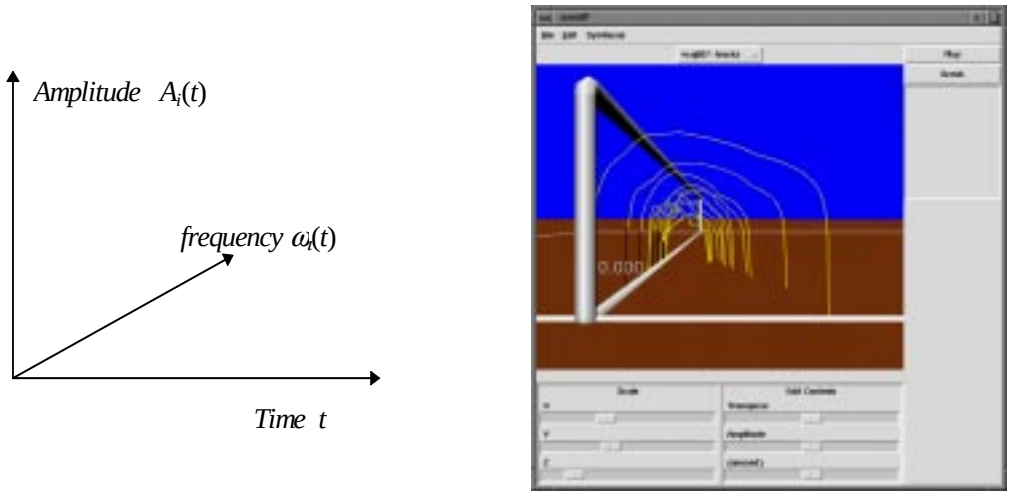
Figure 2.1. Features of the OpenSoundEdit interface. This example features a ResonanceEditor view.

2.1 TrackEditor View

In additive synthesis, sounds are modeled as a sum of *N* sinusoid functions whose amplitude, frequency and phase change over time:

$$x(t) = \sum_{i=1}^{N} A_i(t)\cos\left(\omega_i(t)\,t + \phi_i(t)\right) \qquad (2.1)$$

The functions $A_i(t)$, $\omega_i(t)$ and $\phi_i(t)$ represent the time-varying amplitude, frequency and phase of the *i*th sinusoidal function, respectively, and together form a *sinusoidal track*. Each component of a track can be independently scaled or modified. The TrackEditor view is used to display and edit sinusoidal track models, as illustrated in figure 2.2. Tracks are rendered as connected line segments representing the change in amplitude (in decibels) and frequency (in Hertz) over time (in seconds). Amplitude and frequency are displayed on logarithmic scales. Phase is not shown.



a) *Coordinate system* *b) An example TrackEditor,*

Figure 2.2 The structure of TrackEditor view.

The TrackEditor includes a time-selection frame parallel to the frequency-amplitude plane. The interpolated frequency and amplitude values of the tracks at the selected time are projected onto the frame as black lines. The time frame can also be used to control time manually when the sound is being realized on a synthesis server. This technique is similar to what is known as *scrubbing* in the professional audio community, but without any pitch change associated with the time-domain motion.

Tracks can be selected by clicking on the white bar at the top of the time window and dragging the pointer along the range of frequency values to be selected. Selected tracks can be copied into other TrackEditors using the traditional cut, copy and paste operations. Tracks can also be scaled by a constant factor along the frequency axis (i.e., transposed) or the amplitude axis. The user can also reshape a track by selecting and scaling individual data points within specified tracks.

2.2 ResonanceEditor View

Sounds in which the frequencies of the partials remain constant and amplitudes are exponentially decaying functions are referred to as *resonances*. Sounds based on resonance models are represented more efficiently using a special form of the additive synthesis model:

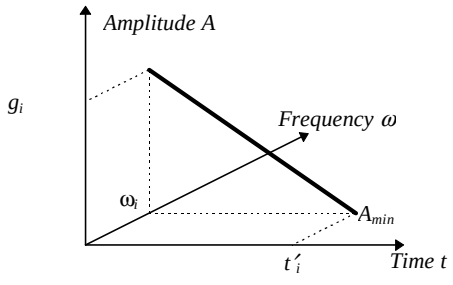
$$x(t) = \sum_{i=1}^{N} 10^{g_i/20} e^{-\pi k_i t} \cos(\omega_i t + \phi_i) \qquad (2.2)$$

where g_i, ω_i, k_i and ϕ_i are the gain (i.e., initial amplitude, expressed in decibels), frequency, bandwidth and phase of the *i*th resonance, respectively. The gain determines initial energy of the resonance, and the bandwidth determines the rate of decay. A smaller bandwidth means a longer decay, and a bandwidth of zero means the resonance stays at constant energy. A resonance model is described by a collection of N quadruples $\{\omega_i, g_i, k_i, \phi_i\}$.

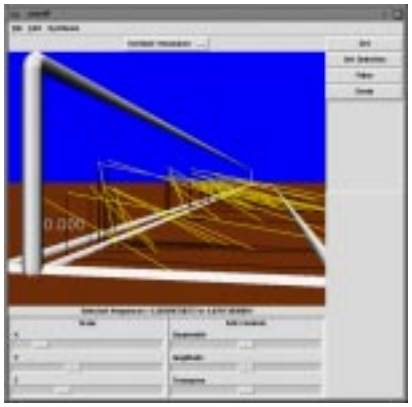
The ResonanceEditor is used to view and edit resonance models. Resonances are displayed in a coordinate system where the *x*-axis is time in seconds, the *y*-axis is the amplitude in decibels, and the *z*-axis is frequency in Hertz. As with tracks, amplitude and frequency are plotted on logarithmic scales, and phase is not shown. Each component $\{\omega_i, g_i, k_i, \phi_i\}$ is represented by a line as shown in Figure 2.3a The endpoints of this line are $(0, g_i, \omega)$ and (t'_i, A_{min}, ω) where A_{min} is the user-definable threshold of hearing, and t'_i is the time at which the energy of the resonance reaches A_{min}, calculated as follows:

$$t'_i = \frac{g_i - A_{\min}}{(\pi \log_{10} e) k_i} \qquad (2.3)$$

If the bandwidth k_i is zero, the resonance is represented by a horizontal line starting at $(0, g_i, \omega_i)$.



a) Visualization model



b) A ResonanceEditor

Figure 2.3 Resonance representation

The ResonanceEditor view includes moveable time-selection and amplitude-threshold frames parallel to the frequency-amplitude and time-frequency (i.e., ground) planes, respectively. As in TrackEditor views, the time-selection frame can be moved along the time axis and lines representing the amplitude values at the selected time are projected on the frame. The threshold frame can be moved along the amplitude axis, and changes the value of A_{min}. Lines indicating how long each resonance lasts until its energy reaches the threshold are drawn on the frame. The time is calculated using equation 2.4.

Resonances to be edited are specified by selecting a frequency range along the top bar of the time frame. Selected resonances can be transposed, gain-scaled or time-scaled. They can also be cut, copied and pasted into other ResonanceEditor views.

A demonstration of the ResonanceEditor view is presented separately [CFKW98].

3. Implementation Issues

OSE is a client process that connects to a real-time synthesis server. It communicates with the server using the Open Sound Control (OSC) protocol [WF97] and uses the Sound Description Interchange Format (SDIF) [W98] to read and write sound representations to files.

OSE is implemented using portable graphics and user-interface technologies. OpenGL [WND97] provides a standardized graphics library across different platforms. VTK, the Visualization Toolkit [SML96], is used for modeling 3D objects in the display window. Tcl/Tk [W97] and MIT Otcl [WL95] are used to bind user input (i.e., from direct manipulation of the 3D objects or from external Tk widgets) to editing operations and OSC commands for the synthesis server. The OSE implementation has been run successfully on an SGI O2 running Irix 6.3 and an Intel Pentium Pro 200Mhz running Windows NT 4.0.

Acknowledgments

We gratefully acknowledge the NSF Graduate Research Fellowship Program and Silicon Graphics, Inc. for their support of this research.

References

[C98] A. Chaudhary. "OpenSoundEdit: An Interactive Visualization and Editing Framework for Timbral Resources." Masters Degree Report, University of California at Berkeley, 1998. Available at `http://bmrc.berkeley.edu/papers`.

[CFKW98] A. Chaudhary, A. Freed, S. Khoury, D. Wessel. "A 3-D Graphical User Interface for Resonance Modeling." *1998 International Computer Music Conference*, Ann Arbor, MI.

[SML96] W. Schroeder, K. Martin, B. Lorensen. *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*. Prentice Hall, Upper Saddle River, NJ, 1996.

[W97] B. B. Welch. *Practical Programming in Tcl & Tk, 2nd Edition*. Prentice Hall, 1997.

[WL95] D. Wetherall, C. J. Lindblad. "Extending Tcl for dynamic object-oriented programming." *Proceedings of the Tcl/Tk Workshop 1995*, Toronto, Ontario, 1995.

[WND97] M. Woo, J. Neider, T. Davis. *OpenGL Programming Guide, 2nd ed.* Addison-Wesley, Reading, MA, 1997.

[WF97] M. Wright, A. Freed. "OpenSound Control: A New Protocol for Communicating with Sound Synthesizers." *Proceedings of the 23rd International Computer Music Conference*, Thessoloniki, Greece, 1997. `http://www.cnmat.berkeley.edu/Research`

[W98] M. Wright et al. "New Applications of the Sound Description Interchange Format." *1998 International Computer Music Conference*, Ann Arbor, MI. `http://www.cnmat.berkeley.edu/SDIF`