Visualization, Editing and Spatialization of Sound Representations using the OSE Framework

Amar Chaudhary and Adrian Freed
Center for New Music and Audio Technology
University of California
1750 Arch Street, Berkeley, CA 94709
{amar,adrian}@cnmat.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 by using additional representation-specific controls. Users can also model the 3D location and movement of sounds in a room. 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 examples of several sound representations.

1 Introduction

Audio professionals and researchers manipulate sounds using many different sound representations, including time-domain waveforms, frequency-domain sinusoidal components and resonance models. These representations often include large multi-dimensional datasets. The recently adopted Sound Description Interchange Format provides a standard for storing and sharing such high-level multi-dimensional sound representations [1]. A sound visualization and editing system is needed that provides control over different sound representations. The system should be able to present large multi-dimensional datasets, maintain a meaningful spatial and temporal orientation when multiple sound representations are displayed, provide real-time visual and aural feedback to editing operations and allow the set of supported sound representations and editing capabilities to be easily extended.

We introduce Open Sound Edit (OSE), a framework that provides a three-dimensional user interface to visualize, edit and process complex sounds composed from different sound representations [2]. 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 many of the SDIF standard frame types. It can be easily extended to handle other sound representations as well.

The OSE user interface is illustrated in Figure 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 1), one or more moveable frames (outlined with white bars), and 2D projections of the data set onto the

frames (the black lines). The coordinate system includes time and amplitude dimensions (i.e., the horizontal and vertical directions in Figure 1) and a varying third dimensions for frequency, channel number, etc. All views are presented in the same 3D display window, and aligned along their time and amplitude axes. A ground plane and sky and coordinate axes are included to orient the user in this display.

The moveable frames are used for marking a position on one of the axes. For example, the Resonance Editor view in Figure 1 has frames perpendicular to the time and amplitude axes. The frame parallel to the frequency-amplitude plane is called a *time-selection frame* and 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.

When a sound representation is loaded from an SDIF file, a new view is created and placed at an arbitrary location on the ground plane. Figure 2 illustrates the coordinate axis and ground plane in the display window. Interface controls are supplied to translate and rotate the current viewpoint, or camera position, within the display. Figure 3 shows a view from camera positions perpendicular to the time-amplitude, frequency-amplitude and time-frequency planes. Editing operations are performed via direct manipulation in the 3D display window, or using external controls, as shown at the bottom of Figure 1. The behavior of these edit controls are specific to the type of view selected.

OSE has recently been extended to support room models for sound spatialization. Room models are displayed in the same 3D window as sound representation views and use the same ground plane, sky and coordinate system.

The 3D interface adopted in OSE provides the user a more expressive and intuitive view of sound representations than can be provided on a 2D display. The varying third dimension can be used to express parameters that previously required special mappings onto two dimensions [3]. The unified display allows the user to work with differing representations of sounds without separate 2D windows or the clutter of 2D overlays. Changing the viewpoint and moving representation views on the ground plane facilitates the location components of a sound representation (e.g., resonance partials) that are responsible for a particular characteristic of a sound. Direct manipulation allows efficient editing of sound representations using familiar gestures, such as pulling, stretching and sweeping.

Early tests underscored the need for a familiar 3D user interface model. For example, early versions of OSE displayed sounds on a completely black background and allowed the user unconstrained translation and rotation of the viewpoint. Such an interface presents the user with the metaphor of “flying through space,” which is unfamiliar and disorienting. The addition of the ground and sky metaphors provide a familiar orientation, which is further enhanced by constraining movement to be parallel with the ground plane. Furthermore, properties of familiar 3D environments can be used to provide additional useful information. For example, the black lines projected onto the time and threshold frames use a familiar “shadow” metaphor for projecting the three dimensional representations onto two dimensions.

OSE is a client of a *synthesis server*, an application that accepts sound data and control messages from clients and generates waveforms for audio output in real time. OSE communicates with synthesis servers via SDIF data and the OpenSound Control protocol [4] to provide real-time audio feedback for editing and control operations.

OSE is built using a suite of portable technologies. VTK, the Visualization Toolkit [5], is used for modeling 3D objects. VTK consists of C++ classes commands for visualizing large datasets and complex geometric objects using standard OpenGL graphics [6, 7]. Tcl/Tk [8] is used to bind user input (i.e., from direct manipulation of the 3D objects implemented using VTK or from external Tk widgets) to editing operations and OSC commands for the synthesis server. Tcl/Tk, VTK and OpenGL are available on the major computer platforms used by audio and music professionals.

2 Sound Representations Supported by OSE

OSE currently supports several sound representations used in analysis and synthesis applications. This section describes each supported representation, its visualization and any supported editing capabilities.

2.1 Time-domain Samples

OSE supports viewing of multi-channel, time-domain sampled waveforms. Sampled sounds are displayed on a three-dimensional coordinate system representing time, amplitude and channel number, as illustrated in Figure 4. This view is included to handle time-domain samples that may be bundled with other representations in an SDIF file, and can be useful during analysis. Figure 5 shows an example of a time-domain waveform and a time-varying estimate of pitch superimposed on the same set of coordinate axes.

2.2 Pitch and Energy/Amplitude Estimates

Pitch is a perceptual quality of a sound, often associated with its fundamental frequency or periodicity, depending on the nature of the sound [9]. The *energy* of a sound, measured as either the peak or average amplitude of the samples, is associated with loudness. Many techniques for analyzing music and speech sounds require a good estimate of pitch and energy as functions of time over the duration of the sound. Pitch and energy are also important parameters for synthesizing sound. Traditional music notation instructs performers to produce sounds at specified pitches and amplitudes. Many digital synthesizers can respond to continuous pitch and energy changes as well.

The Pitch Editor is used to display pitch and amplitude functions on a 3D coordinate system representing time, amplitude and pitch, as illustrated in Figure 6. The amplitude dimension measures energy in decibels and is plotted on a logarithmic scale. Pitch is measured in Hertz and can be displayed in either a linear or logarithmic scale.

In addition to providing information about the energy or loudness of a sound, the amplitude provides a measure of the confidence of the estimated pitch. Thus, a user can judge the

usefulness of estimated pitch values by their height above the ground plane. Such information would be absent from most two-dimensional plots of time-varying pitch. The Pitch Editor includes an amplitude-threshold frame parallel to the ground plane. Areas of low amplitude can be hidden from view by raising the amplitude-threshold plane.

The Pitch Editor view also includes a time-selection frame parallel to the amplitude-frequency plane. When the time-selection frame is moved along the time axis, the interpolated pitch and amplitude values at the selected time are projected on the frame as a shadow. The time-selection frame can also be used to control the pitch and amplitude of synthesizers that produce sound using these parameters.

Users can scale the pitch and amplitude by a constant factor across the entire function, or in specific time regions around the time-selection frame. Ranges along the time axis can be selected, cut and pasted into other Pitch Editors. When viewed from above, a new pitch contour can be hand-drawn on the threshold frame.

2.3 Time-varying Spectra

A sampled sound can be turned in a frequency spectrum using a discrete frequency-domain transform such as a DFT or FFT:

$$X(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n)\, e^{-2\pi i (kn/N)} \qquad \textbf{(2.1)}$$

where $x(n)$ is a set of N samples from the input sound and $X(k)$ is a set of N complex-valued samples, or *bins*, of the frequency spectrum. Sounds can be represented as a sequence of such spectra over successive time intervals [10].

OSE supports viewing of time-varying spectra, as illustrated in Figure 7. The dimensions of the spectral viewer are time, amplitude and the frequency bins. Amplitude is the magnitude of the complex values in the spectrum, and is measured in on a logarithmic (i.e., decibel) scale. Frequencies can be plotted on either a linear or logarithmic scale. Because both the time-domain and frequency-domain sampling in time-varying spectra are intended to approximate continuous functions, this representation is visualized as a continuous surface. The surface is shaded smoothly with color or grayscale intensities according to amplitude. When viewed from above, the three-dimensional shaded surface looks like a traditional “spectrogram” view.

The spectral viewer includes time-selection and amplitude-threshold frames. The threshold frame is useful for locating areas of high amplitude in the spectrum. When the amplitude-threshold frame is raised, as in Figure 8, areas of high-amplitude appear as “islands.”

The spectral viewer 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 represented by the spectra 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.

2.4 Picked Spectral Peaks

Time-varying spectral representations tend to be extremely large (e.g., many hundreds of frames with 1024 bins in each). Often, we are only interested in the *peaks*, or local maxima, in each spectrum [11].

OSE supports viewing of spectral peaks, as illustrated in Figure 9. Like spectra, peaks are visualized on a coordinate system representing time, amplitude and frequency. Each peak is represented as a vertical line, whose *x* and *z* coordinates in the ground plane represent its time and frequency, respectively, and whose height is its amplitude. The peak viewer includes an amplitude-threshold frame that can be used to hide peaks below a certain amplitude.

2.5 Sinusoidal Tracks

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

$$x(t) = \sum_{i=1}^{N} A_i(t)\cos\left(\omega_i(t)\,t + \phi_i(t)\right) \qquad \textbf{(2.2)}$$

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*. The amplitude, frequency and phase components of each track can be independently scaled or modified using the same operations available for the amplitude function in sampling. Additive synthesis provides a very general model for component-based editing of sounds, but this generalization comes at the expense of large representations. An accurate model of a sound may require hundreds of tracks with several hundred points in each track.

The Track Editor view is used to display sinusoidal track models. Tracks are rendered as connected line segments representing the change in amplitude and frequency over time. Time is measured in sections. Amplitude is measured in decibels and plotted on a logarithmic scale, while frequency is measured in Hertz and plotted on a linear or logarithmic scale as selected by the user. Phase is not shown. An example of an additive synthesis representation used in the Track Editor is illustrated in Figure 10.

The TrackEditor includes a time-selection frame that supports scrubbing along the time axis. When scrubbing is used, the synthesis server holds the values of the tracks constant at the selected time *T*:

$$x(t) = \sum_{i=1}^{N} A_i(T)\cos(\omega_i(T)\,t + \phi_i(T)) \qquad \textbf{(2.3)}$$

Tracks can be selected by clicking on the white bar at the top of the time frame 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 or the amplitude axis. The user can also reshape a track by selecting and scaling individual data points within specified tracks.

Such local scaling operations serve as the basis for more complex editing techniques. For example, during a local amplitude-scaling operation, "neighboring" data points (i.e., points on the same track that are close in time, or on other tracks that are close in frequency) will be appropriately scaled in order to preserve a constant energy over the entire data set (e.g., if the amplitude of a point is lowered, the amplitude of neighboring points will be increased). This "timbre-pillow" function [12] is supported by the TrackEditor.

2.6 Resonance Models

Resonance is the response of an acoustic system to a sound source, called an *excitation*. A wide variety of natural musical sounds can be modeled using excitations and resonances [13]. If the excitation is an impulse, the response is a set of exponentially-decaying sinusoids:

$$x(t) = \sum_{i=1}^{N} 10^{g_i/20} e^{-\pi k_i t} \cos(\omega_i\, t + \phi_i) \qquad \textbf{(2.4)}$$

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 ith sinusoid, 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. The resonance model is the set collection of N quadruples $\{\omega_i, g_i, k_i, \phi_i\}$. Resonance models can be more generally realized using a bank of second-order filters. The filter realization of a resonance model can be applied to any excitation source.

The Resonance Editor is used to view and edit resonance models. Resonances are drawn in three dimensions as lines parallel to the time/log amplitude plane with negative slopes with respect to the time axis, as illustrated in Figure 11. Each line represents the exponential component of the impulse response of a second order resonating filter, i.e., a decaying sinusoid, as described in Equation 2.1 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:

$$t'_i = \frac{g_i - A_{\min}}{(\pi \log_{10} e)k_i} \qquad \textbf{(2.5)}$$

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

The ResonanceEditor includes moveable time-selection and amplitude-threshold. The time-selection frame can be moved along the time axis with black lines representing the amplitude values at the selected time projected on the frame. These lines are the shadows formed by the yellow exponential-decay lines on the time-selection frame. The time-selection frame can also be used for scrubbing when the resonances are interpreted as exponentially-decaying sinusoids. When scrubbing, the amplitudes of the resonance partials are held constant at the selected time *T*:

$$x(t) = \sum_{i=1}^{N} 10^{g_i/20} e^{-\pi k_i T} \cos(\omega_i t + \phi_i) \qquad \textbf{(2.6)}$$

The threshold frame can be moved along the amplitude axis to change the value of A_{min}. Shadow lines indicating how long each resonance lasts until its energy reaches the threshold are drawn on the frame, as shown in Figure 13. The lengths of the shadows change as the threshold frame is raised or lowered. The time is calculated using equation 2.5.

Resonance partials to be edited are specified by selecting a frequency range along the top bar of the time frame, as illustrated in Figure 12. Partials can also be selected by amplitude using the threshold frame. Selected resonance partials can be transposed, gain-scaled or time-scaled. They can also be cut, copied and pasted into other ResonanceEditor views.

OSE includes pre-defined resonance models representing a single decaying sinusoid (i.e., a "pure tone"), a pair of sinusoids separated by a small "beat-frequency," and a harmonic series. These primitive models can serve as source material for building more complex models.

Resonance models can be realized on synthesis servers using either the exponentially-decaying sinusoids or filter-bank interpretations, depending on the needs of the user. The exponentially-decaying sinusoids only model the impulse response, but allow scrubbing along the time axis. Filter banks allow the synthesis server to realize resonance models using any available excitation source, including live audio input.

2.7 Sound Spatialization Models

OSE has recently been extended to embrace sound spatialization models by integrating a specialized sound visualization program, `sview`, developed by Sami Khoury [14]. The user can place and move sound sources and sinks in a spatial representation of a listening room. `Sview` was originally developed to show speakers, microphones, and acoustical instrument sources. Figure 14 illustrates an example listening space in `sview`.

OSE supports views that represent listening rooms, as illustrated in Figure 15. The coordinate axes of the room model view are the length, height and width of the room. In addition to speakers and acoustical instrument sources, the OSE room model accommodates iconized representations of any SDIF data. These representations can be linked to other OSE views to visualize or edit the data.

Real-time processing is performed by communicating spatial position data to synthesis and spatialization servers using OpenSound Control. Spatial positions and trajectories are stored using a new SDIF frame type [1].

3 Discussion and Future Work

OSE currently runs on SGI workstations and Intel-based computers running Windows NT/98/95 or Linux. Both OpenGL and Mesa3D [15] graphics are supported for Linux. OSE is being ported to Macintosh computers as well.

Several recent musical performances used resonance models that were created or modified using OSE [16] [17].

User feedback has been encouraging. Experienced users of sound editing tools particularly enjoy the ResonanceEditor view. It is easier to identify and modify components of resonances, which have only one data point per partial, than with sinusoidal tracks, which may have hundreds of points for each partial. Modifying the frequency, gain and decay rate parameters of a resonance partial has an intuitive effect on the sound output. The compact representation also allows more tightly synchronized graphics and audio output when editing resonances. Additional work will provide high-level editing capabilities for the larger sinusoidal tracks and spectral models, including customizable curve and surface deformations [18].

4 Acknowledgements

We gratefully acknowledge the NSF Graduate Research Fellowship Program and Silicon Graphics, Inc. for their support of this research. We would also like to thank David Wessel and Lawrence A. Rowe, directors of CNMAT and the Berkeley Multimedia Research Center, respectively, for their continuing support and feedback.

References

[1] M. Wright, A. Chaudhary, A. Freed, S. Khoury, and D. Wessel, "Audio Applications of the Sound Description Interchange Format Standard," *107th AES Convention*, New York, 1999.

[2] A. Chaudhary, "OpenSoundEdit: An Interactive Visualization and Editing Framework for Timbral Resources," Masters Thesis, Electrical Engineering and Computer Science, University of California, Berkeley, CA, 1998.

[3] A. Freed, "Improving Graphical User Interfaces For Computer Music Applications," *Computer Music Journal*, vol. 19, 4-5, 1995.

[4] M. Wright, "Implementation and Performance Issues with OpenSound Control," *International Computer Music Conference*, Ann Arbor, MI, 1998.

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

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

[7] `http://www.opengl.org`.

[8] B. B. Welch, *Practical programming in Tcl & Tk*, 2nd ed. Upper Saddle River, NJ: Prentice Hall PTR, 1997.

[9] E. Terhardt, "Psychoacoustic evaluation of musical sounds," *Perception & Psychophysics*, vol. 23, 483-492, 1978.

[10] M. R. Portnoff, "Time-frequency representation of digital signals and systems based on short-time Fourier analysis," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-28, 55-69, 1980.

[11] X. Serra and J. Smith, III, "Spectral modeling synthesis: a sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music Journal*, vol. 14, 12-24, 1990.

[12] Personal Communication with David Wessel, 1997.

[13] Y. Potard, P.-F. Baisnée, and J.-B. Barriere, "Experimenting with Models of Resonance Produced by a New Technique for the Analysis of Impulsive Sounds," *International Computer Music Conference*, La Haye, 1986.

[14] S. Khoury, A. Freed, and D. Wessel, "Volumetric Modeling of Acoustic Fields in CNMAT's Sound Spatialization Theatre," *AES 104th Convention*, San Francisco, CA, 1998.

[15] `http://www.mesa3d.org`.

[16] A. Chaudhary, "Two-Tone Bell Fish. " Live performance at CNMAT/CCRMA Spring 1999 Concert Exchange. Stanford University, Palo Alto, CA, April 29 1999.

[17] A. Chaudhary, "Spin Cycle / Control Freak. " Live performance at CNMAT/CCRMA Spring 1999 Concert Exchange. Center for New Music and Audio Technologies, Berkeley, CA, May 15 1999.

[18] Z. Guan, J. Ling, N. Tao, and X. Ping, "Study and Application of Physics-Based Deformable Curves and Surfaces," *Computers & Graphics*, vol. 21, 305-13, 1997.

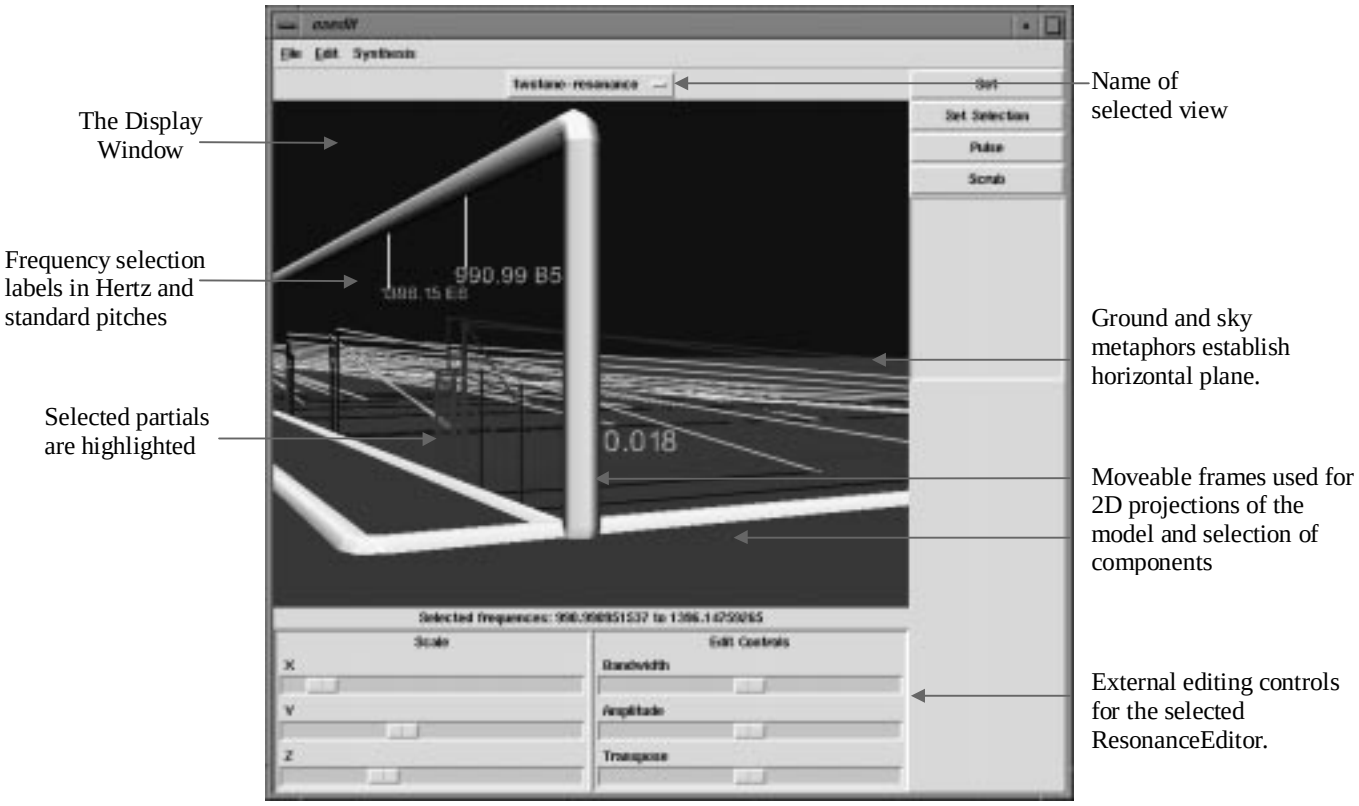


Figure 1. Features of the OSE interface. This example includes a Resonance Editor view.

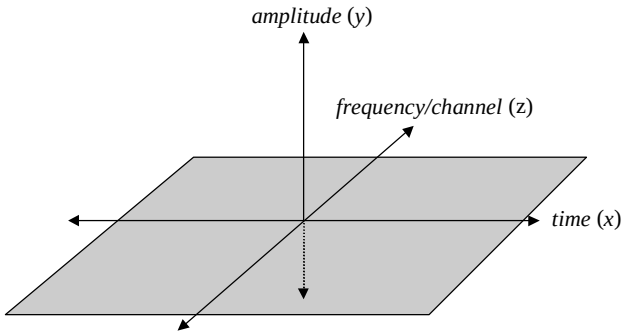
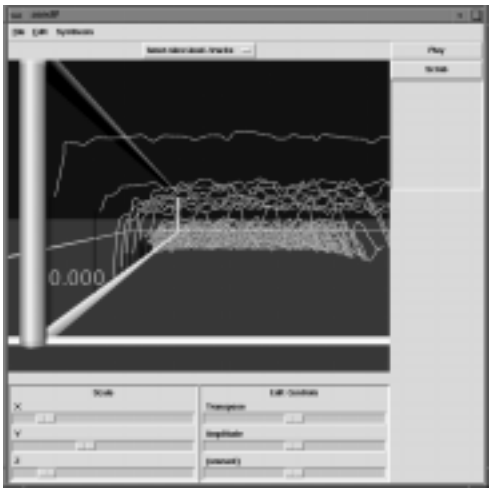
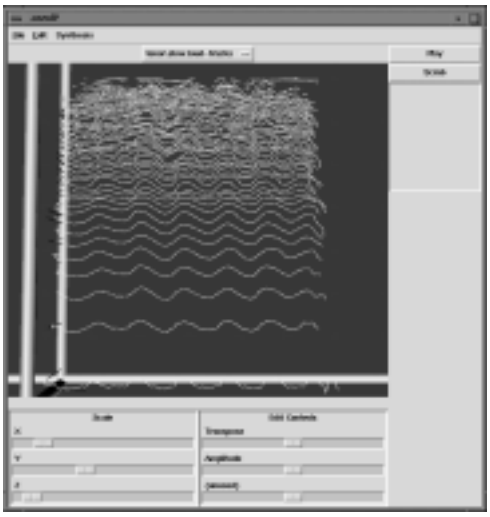


Figure 2. The coordinate system of the 3D display window, with the ground plane shown in solid gray.



a) The time-amplitude plane

b) The frequency-amplitude plane



c) The time-frequency plane

Figure 3. Looking a view from different camera positions. Different perspectives show different characteristics of the sound. In this example, the frequency modulation is confusing in the time-amplitude and frequency-amplitude planes, but can be clearly seen in the time-frequency plane.

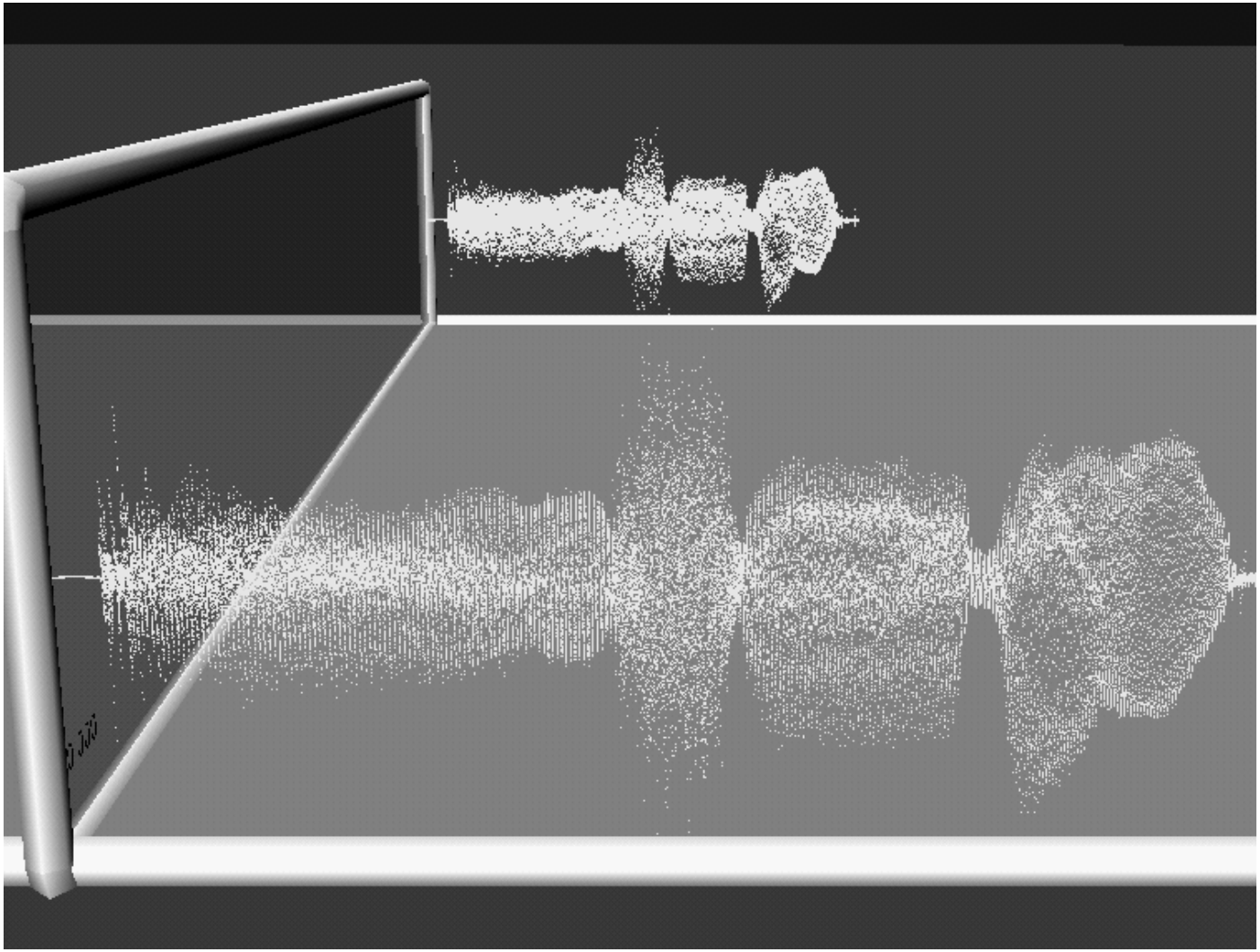


Figure 4. A sampled sound with two channels.

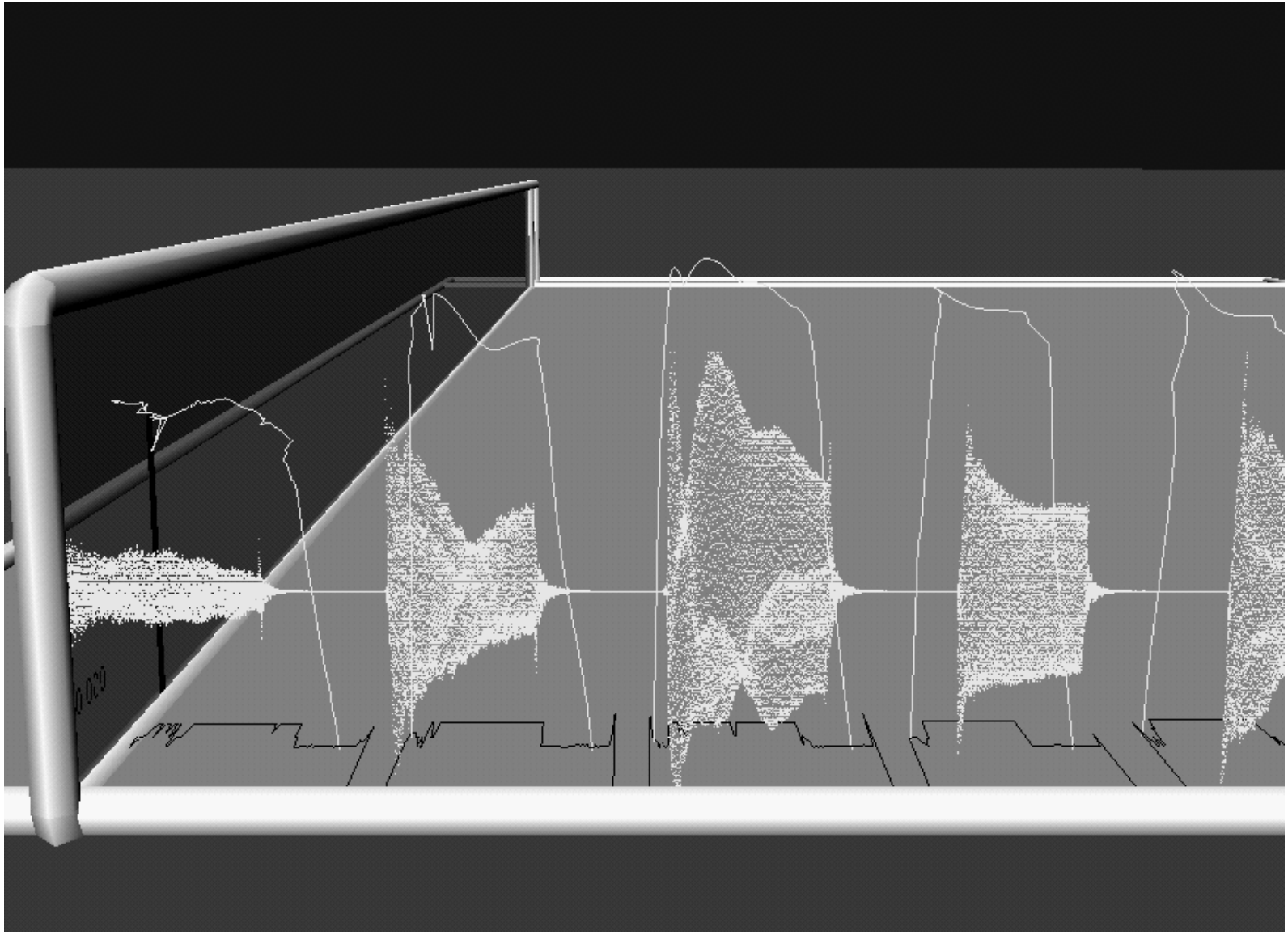


Figure 5. A sound represented as time-domain samples with pitch and amplitude estimates. The pitch estimates can be seen as shadows on the ground plane, and the amplitude estimates "envelop" regions of high amplitude in the sampled representation.

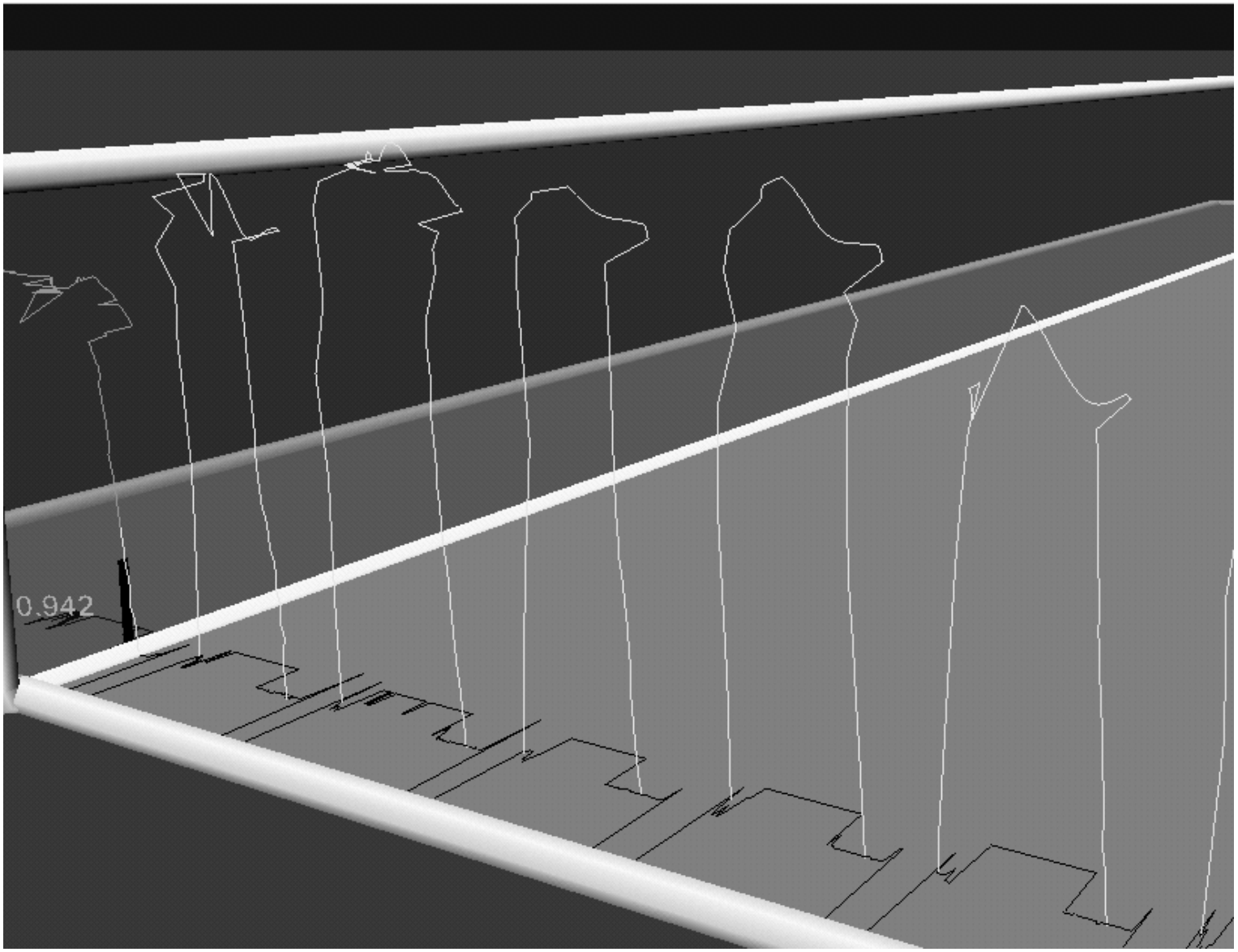


Figure 6. Pitch and amplitude estimates. Amplitude is visualized as distance above the ground plane and pitch is displayed as shadows on the ground plane. The shadow on the time-selection frame indicates the estimated pitch and amplitude at 0.942 seconds.

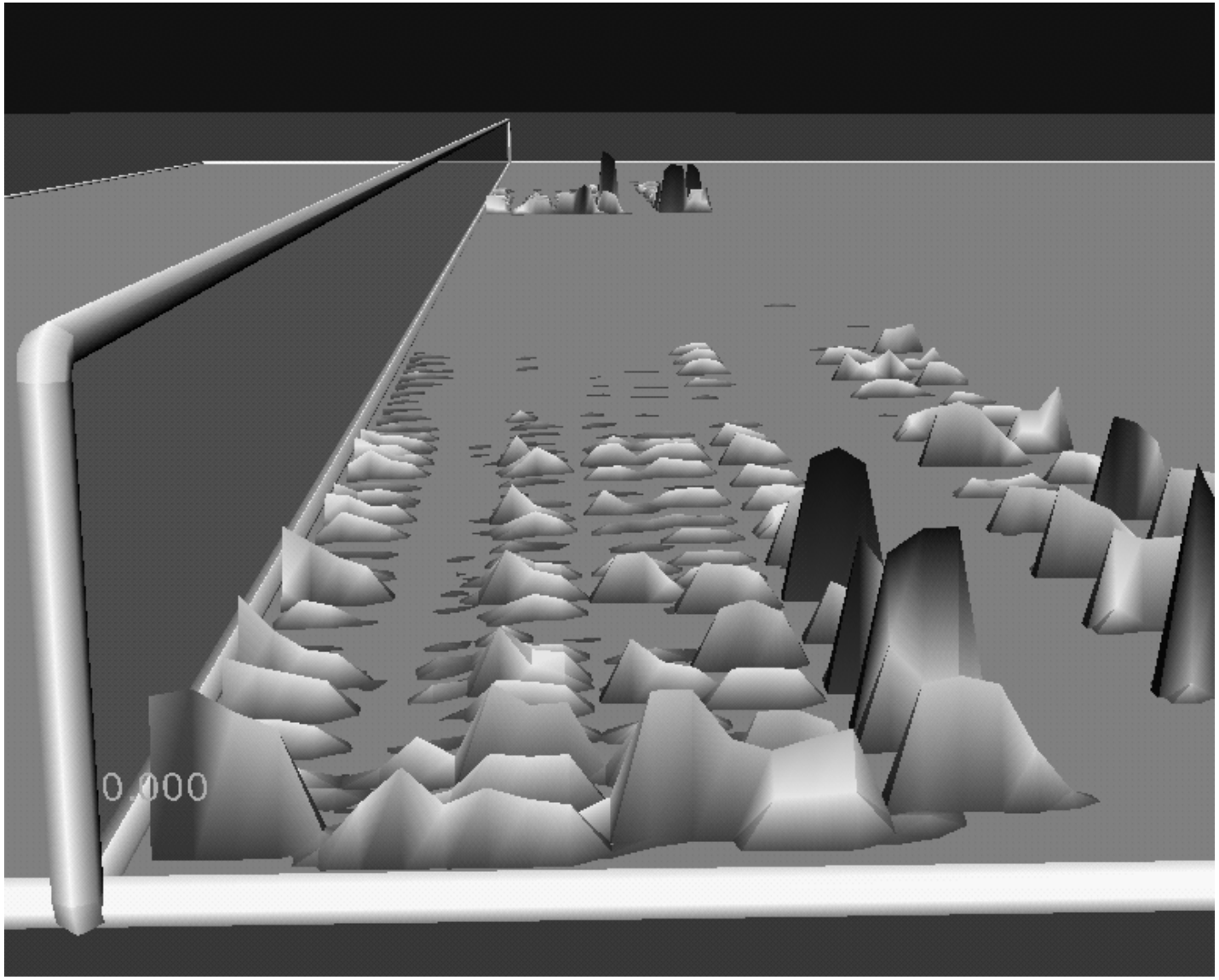


Figure 7. Time-varying spectra are visualized as shaded surfaces.

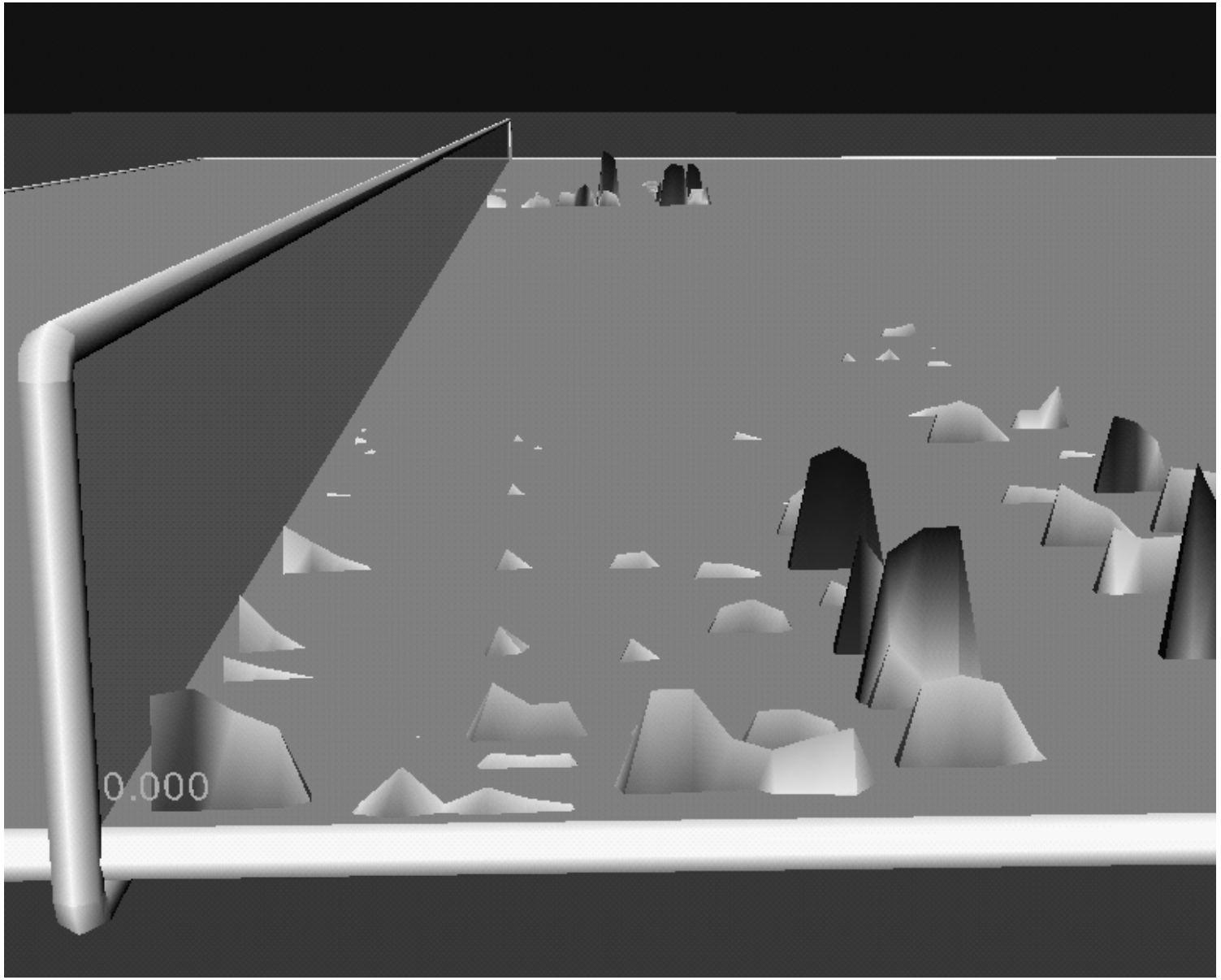


Figure 8. When the amplitude-threshold frame is raised, only regions of high magnitude are visible.

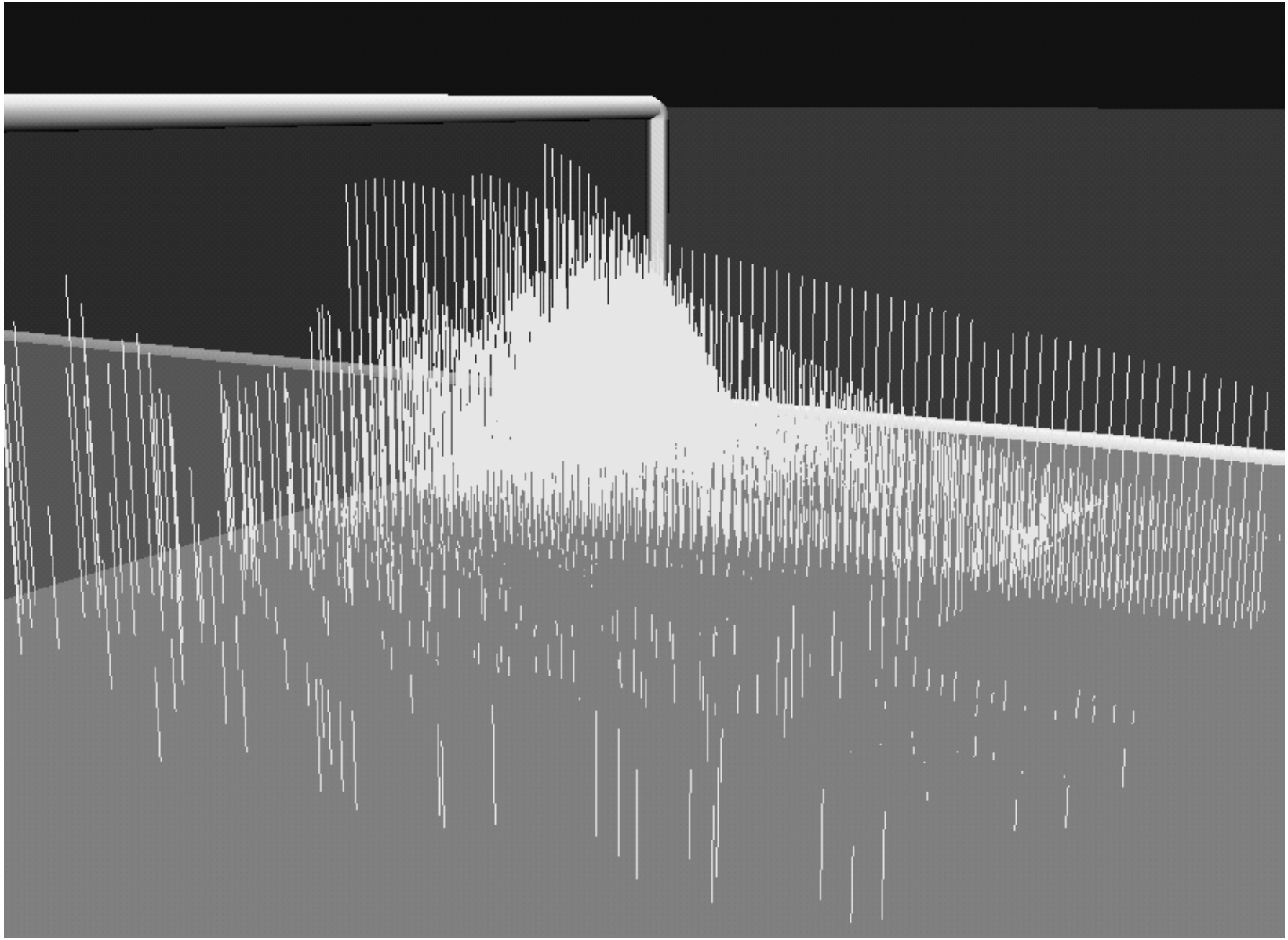


Figure 9. Picked spectral peaks. Each vertical line is a peak from a time-varying spectral representation of the sound.

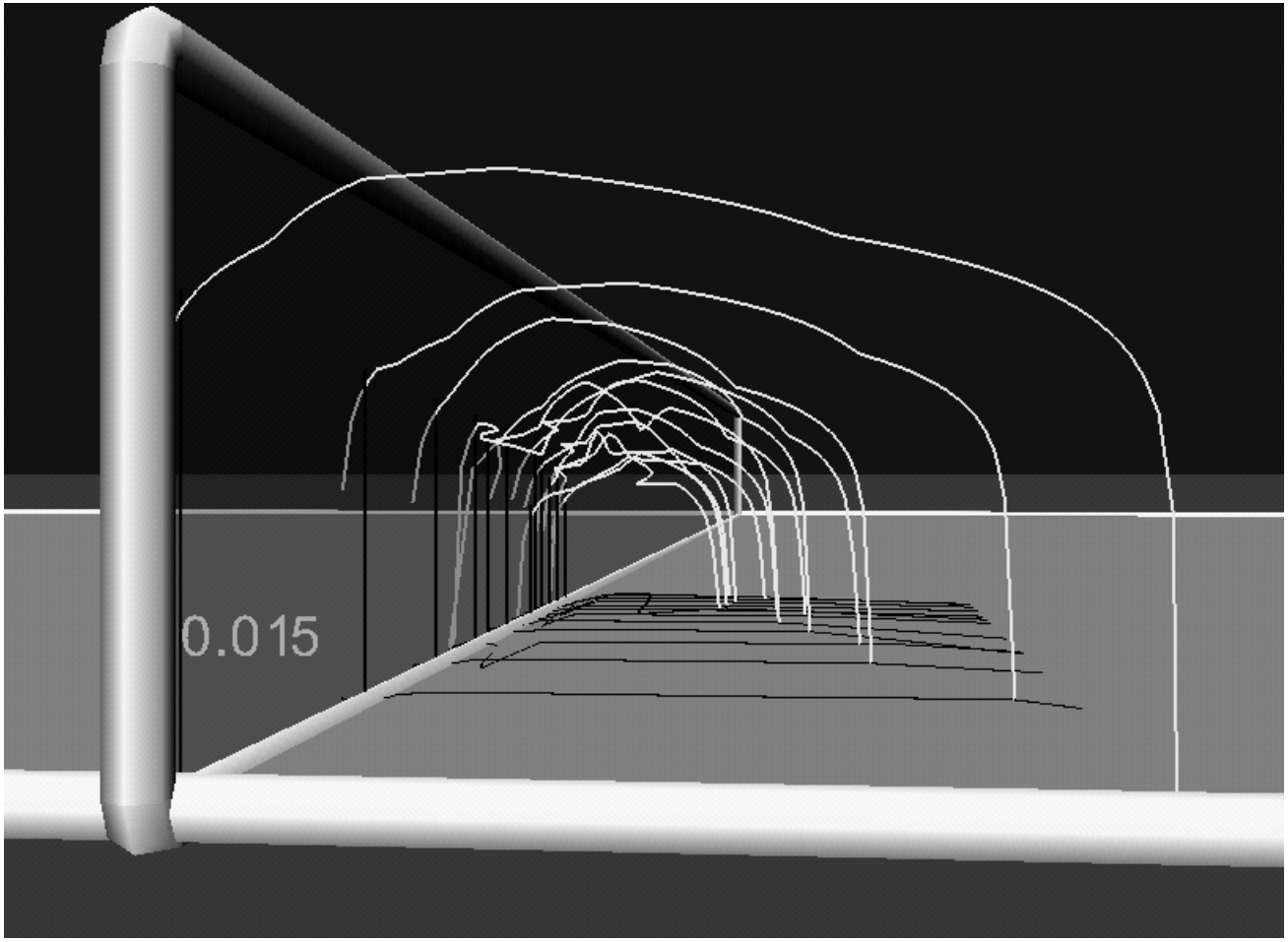


Figure 10. A sinudoidal track model. The shadows on the ground plane display the frequency of each track as a function of time. The projected shadows on the time-selection frame indicate the amplitude and frequency of each track at 15ms.

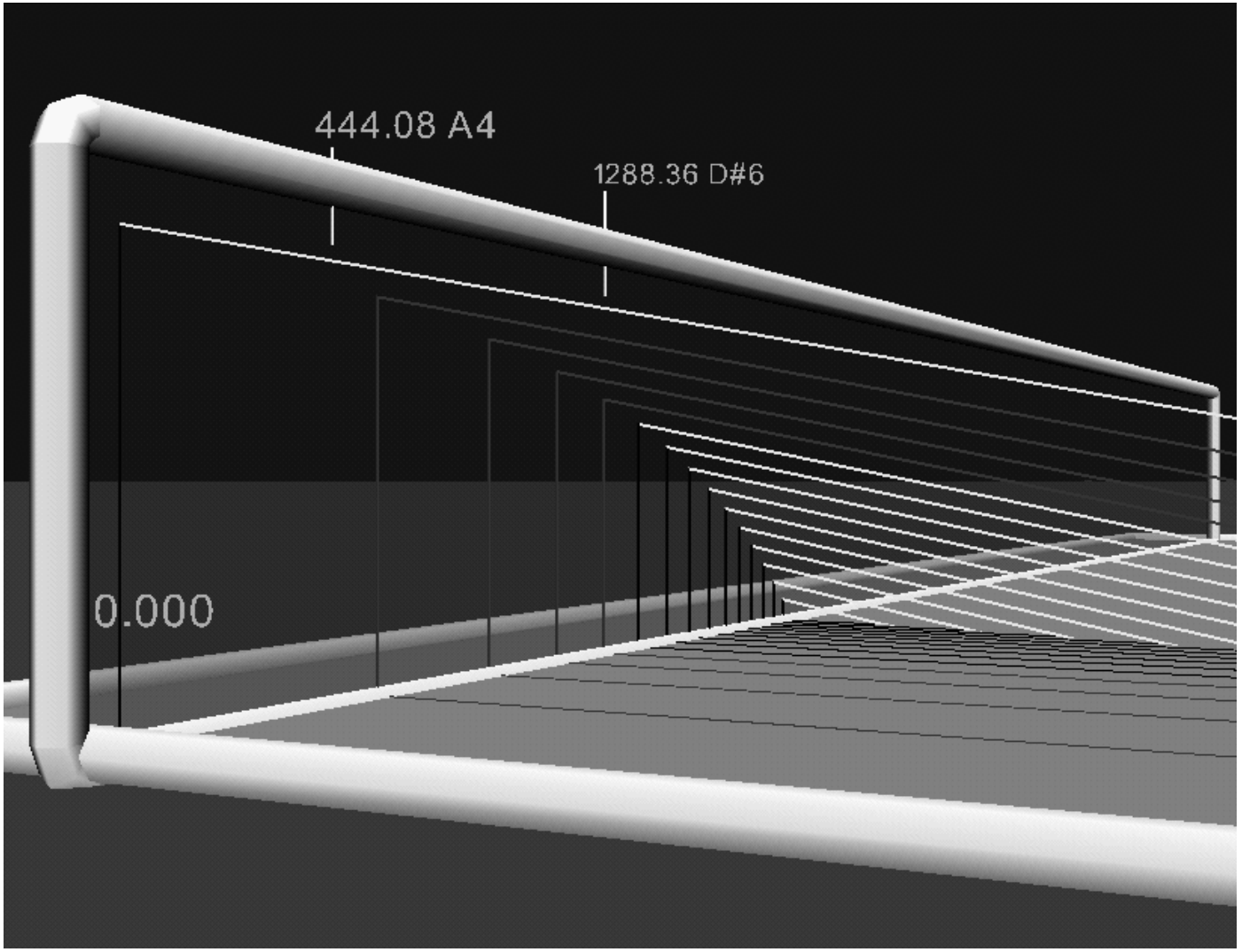


Figure 11. A resonance model viewed from the time-amplitude plane. The resonances decay as time increases towards the right.

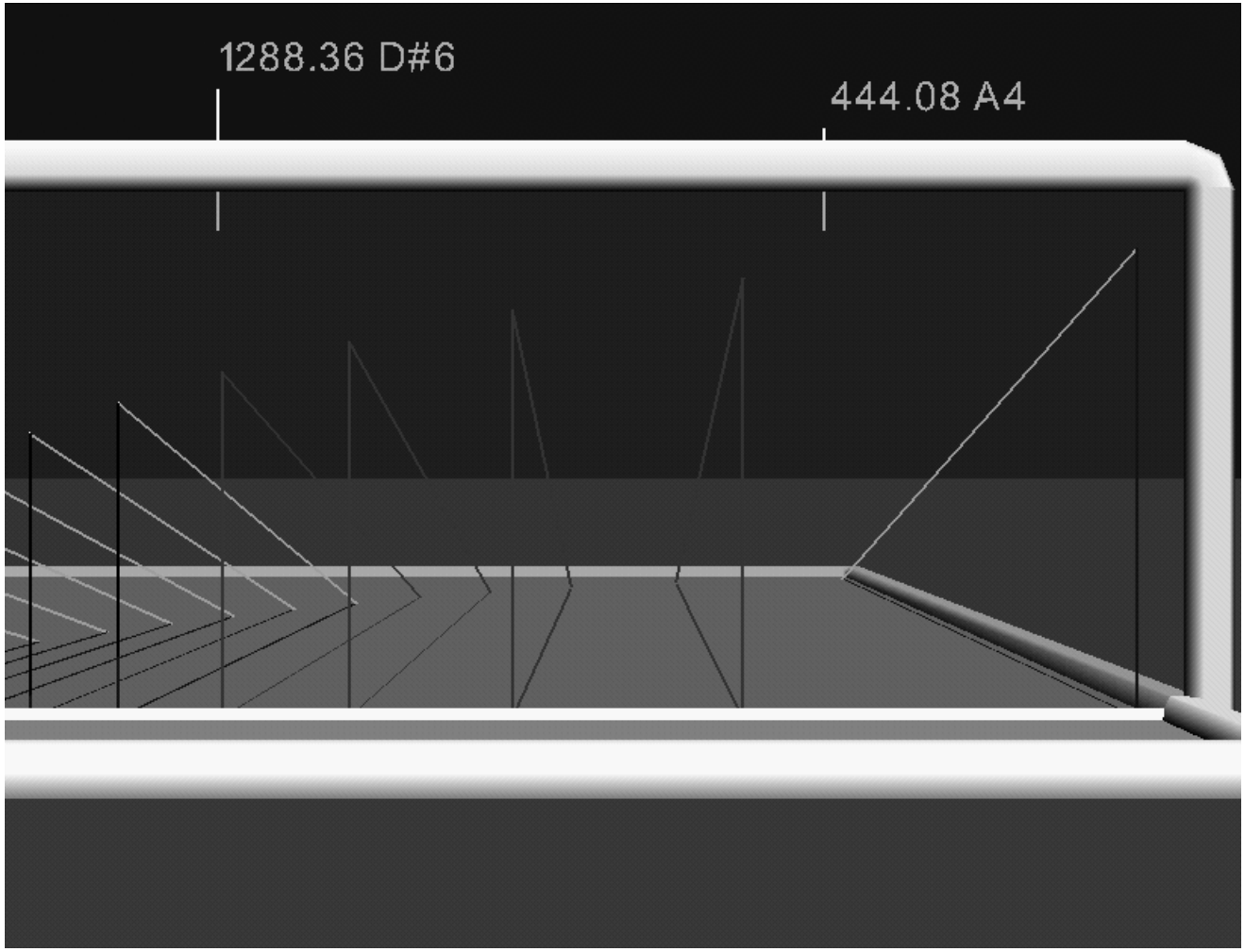


Figure 12. The resonance model from figure 11 as viewed from the frequency-amplitude plane. Partials decay with increasing time towards the distance. The partials between the frequencies A4 and D#6 are selected and can be scaled or cut.

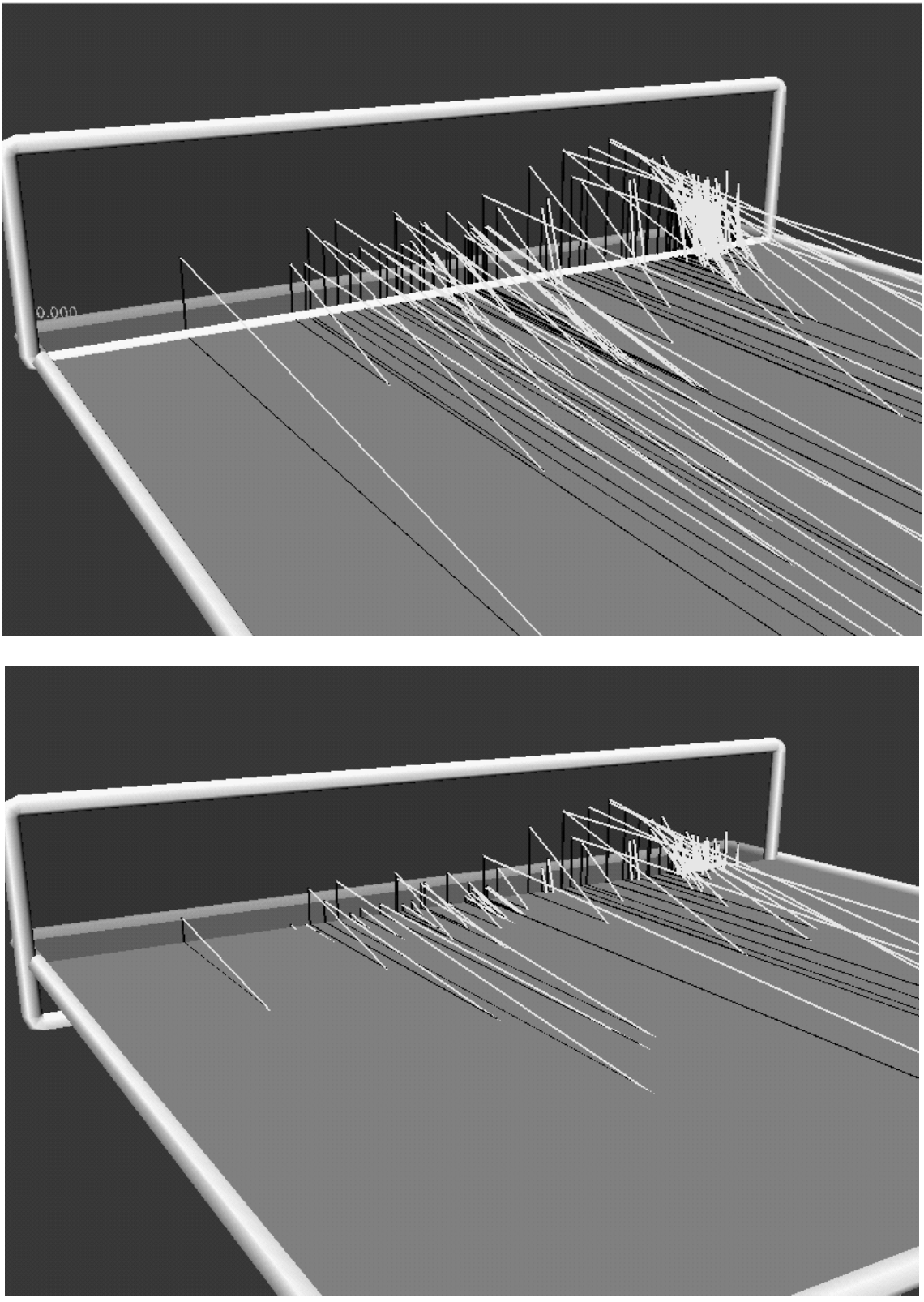


Figure 13. When the amplitude-threshold frame is raised, only the high-amplitude partials are visible.

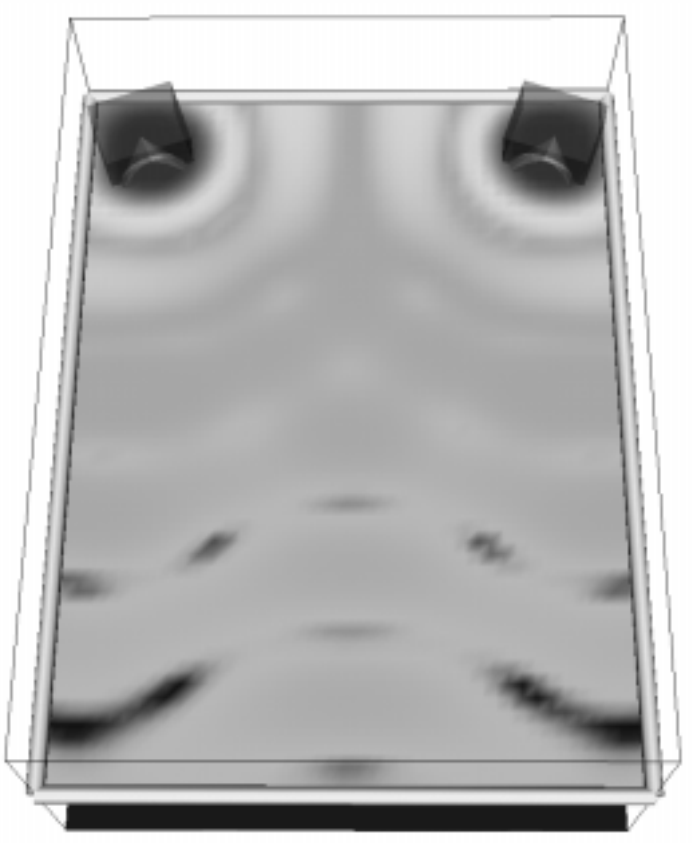


Figure 14. A listening room model in `sview`. The speakers represent sources, and the energy at different locations in the room is plotted on the moveable plane.

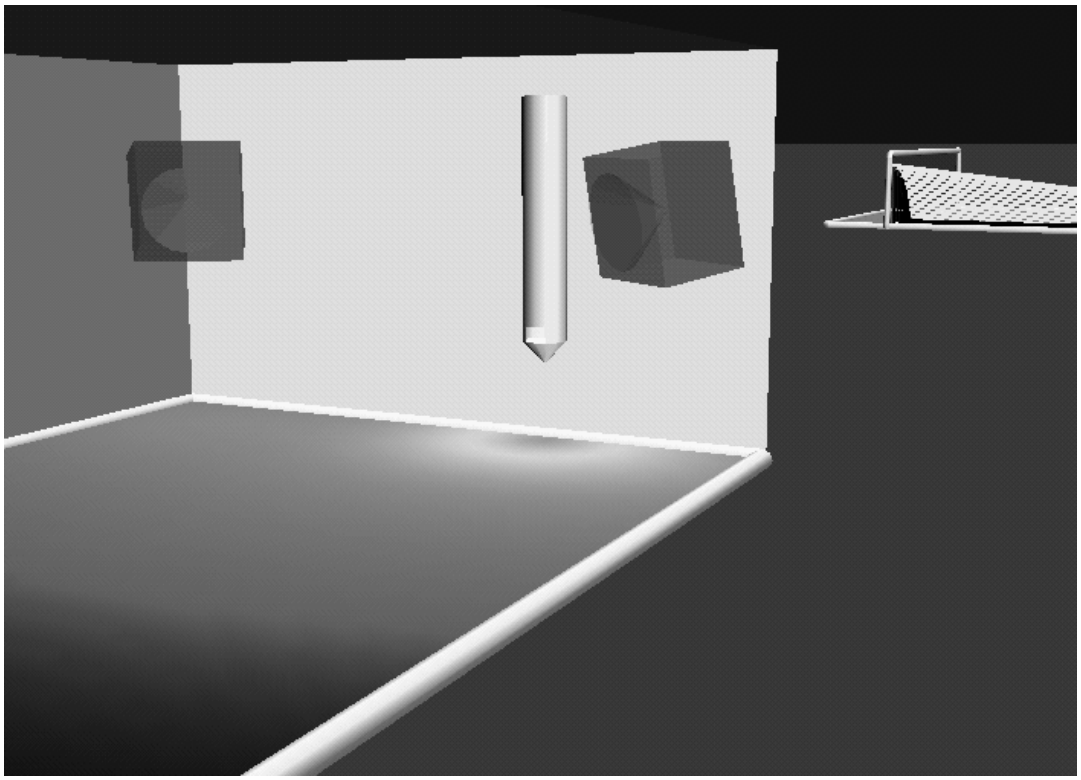


Figure 15. A listening room model in OSE. The speakers and organ pipe represent iconic sound sources that can be linked to other OSE views, such as the nearby resonance model (in the upper right).