A Paradigm for Physical Interaction with Sound in 3-D Audio Space

Mike Wozniewski
Société Des Arts Technologiques
1195 Saint-Laurent boulevard
Montréal, Québec
mikewoz@sat.qc.ca

Zack Settel
Société Des Arts Technologiques
1195 Saint-Laurent boulevard
Montréal, Québec
zack@sat.qc.ca

Jeremy R. Cooperstock
McGill University
3480 University Street
Montréal, Québec
jer@cim.mcgill.ca

Abstract

Immersive virtual environments offer the possibility of natural interaction within a virtual scene that is familiar to users because it is based on everyday activity. The use of such environments for the representation and control of interactive musical systems remains largely unexplored. We propose a paradigm for working with sound and music in a physical context, and develop a framework that allows for the creation of spatialized audio scenes. The framework uses structures called soundNodes, soundConnections, and DSP graphs to organize audio scene content, and offers greater control compared to other representations. 3-D simulation with physical modelling is used to define how audio is processed, and offers a high degree of expressive interaction with sound, particularly when the rules of sound propagation are bent. Sound sources and sinks are modelled within the scene along with the user/listener/performer, creating a navigable 3-D sonic space for sound-engineering, musical creation, listening, and performance.

1 Introduction

In applications that require real-time interactive 3-D audio, such as games, surround sound post-production and virtual reality, virtual sound source simulation has been a primary focus for innovation and development. For an overview of such work, readers can see: (Begault 1994), (Naef et al. 2002), and (Jot 1999). Several 3-D audio libraries are available for developing these applications, including DirectX, OpenAL, X3D, and MPEG-4 AudioBIFS. These libraries provide proper real-time rendering of the sound sources that users can locate in their virtual scenes. However, the spatial audio models tend to be simplistic since much of the processing is focused on graphics rather than audio computation, and the degree of interactive control is limited. Moreover, due to their emphasis on sound sources, these libraries tend to offer limited support for sound sinks (e.g. listeners, virtual microphones) and typically provide support for only one listener. We consider this to be an important limitation, particularly given our interest in applications for music making, listening and performance. A comprehensive 3-D audio environment suitable for developing and hosting interactive music and sound applications is not yet available so we have set out to develop a suitable prototype.

The research framework described here is among only a few (e.g. (Pressing 1997), (Maki-Patola et al. 2005)) that have explored virtual environments from the perspective of musical performance. In previous work (Wozniewski, Settel, and Cooperstock 2006) we introduced our 3-D audio engine, and we now hope to propose a generic paradigm for interacting with audio in virtual environments, and particularly the method for real-time audio processing within 3-D scenes. The framework we have developed provides for the creation, organization, interconnection and control of sound sources and/or sinks, which we call *soundNodes*. User interaction with these nodes is based on 3-D spatial activity, and physical models of acoustic sound propagation are used to connect soundNodes together. An interconnected group of soundNodes is what we refer to as a *DSP graph*, and allows users to define areas in the scene where digital signal processing (DSP) occurs.

It is useful to draw parallels between our environment and those used for 3-D graphics modelling (e.g. OpenGL). From the latter, we have drawn inspiration and have borrowed concepts and techniques that we have adapted for use in: 1) the structuring of audio content, and 2) in the definition of spatial interaction with that content. Consequently, interaction with sound in our environment entails a high degree of spatiality and physicality. In addition, our description of DSP graphs is essentially geometric and based on physics. In several instances, such a description can be advantageous, if not uniquely adapted for use in applications for 3-D audiovisual display, interactive audio galleries, immersive listening, multitrack mixing, among others. Most important from an artistic

point of view, our approach offers an alternative and fundamentally novel way to explore sound, music, and music making, and ultimately, one's own imagination.

2 Background

The motivation for this research began with the need for one of the authors (Settel) to control a growing number of DSP units during live performance *without* having to interrupt the operation of his instrument: i.e., taking the hands off his saxophone to put them on a controller or computer peripheral device. To obtain this kind of transparent control, the first idea was to use a configuration with a separate microphone for each of the *N* DSP units. The microphones would be arranged in a circle around the performer, who would blow the horn directly into a particular microphone, depending on the desired processing to be applied. In theory, the configuration provides the performer with a reasonable degree of transparency, but remains quite limited in compositional range.

An improvement to the idea was to virtualize this configuration with computer simulation. By using one single microphone with an attached orientation sensor, we could now describe where the saxophone was pointing. The performer would physically be on stage, directing his sound to one of *N* surrounding "virtual" microphones, and his sound would be processed accordingly. Unlike the real configuration, the simulated version does not require the performer to move closer to a target microphone. Rather, it is enough to just point or "aim" the instrument towards it. In doing so, the performer's sound is proportionally bussed to the DSP module in question. Additionally, the simulation offers the performer the possibility to *spread* his sound more widely, directing his sound to more than one target. The physicality and directness of this sonic interaction was compelling enough to pursue further, leading us to a greatly expanded notion of spatial interaction with sound, and by extension, with music.

3 Fundamental Concepts

We present a generic spatial framework to support the definition and real-time control of audio configurations, while providing physical user interaction with sound in space. The following concepts, expanded upon below, underlie the framework, and reflect our research focus:

1. A paradigm for physical interaction with sound in space.
2. An immersive environment with physical and virtual aspects that share the same geometric space.
3. "Tweaked" Reality: the use of rule-bending in the acoustic model of sound propagation.
4. The control of sound using familiar spatial activity (moving, turning, pointing, grabbing, etc.).
5. A novel model for the organization and interconnection of audio sources and sinks.
6. A way of working with sound that leads to discovery of novel applications, revisiting conventional applications, and new methods for listening & performing.

3.1 Physical Interaction With Sound

Most important is the concept of *physical* interaction, which in fact conveys two separate ideas. First, we are dealing with a notion of direct physical activity by the human body to affect sounds. Rather than creating layers of abstraction through sliders, pedals, or other controllers, we would like to use natural human motion to control audio processing. This means that we must have an understanding of how the body interacts with the external world. The second idea is then the notion of modelling real-world physics, such as the kinematics associated with moving entities, and the physical properties of sound propagation.

3.2 Immersion

The concept of immersion is a fundamental aspect of the proposed interaction paradigm. Like research in the field of virtual reality, we model the user's position in virtual space. In fact, several aspects of the user's world such as speakers, microphones, walls, and furniture can all be included in the virtual model. The geometry between the physical and virtual worlds is merged so that all spatial relations are kept intact. An object one metre away in the real world will appear one metre away in the virtual world. This "merging" favours a more natural interaction for users, since real-world spatial understanding is all that is necessary to interact with the world and the sounds within it.

At the heart of this framework is an engine that models 3-D geometry and physics of both physical and virtual components. These components may be displayed either visually or sonically, but typically both. A user who is 'inside' of this world is thus offered with a literal rendering of the audio scene which provides direct feedback of the current state of the system.

It should be noted that accurate tracking of the user is needed to realize this system. The user's head position and orientation must be passed to the engine in order to achieve accurate rendering. Particularly important are physical audio devices such as headphones or microphones that are worn by the user. If other components in the real world are moving and have some importance in the audio scene, then they too should be represented and updated in the engine.

3.3 Tweaked Reality

Earlier we mentioned that the 3-D engine models the physical laws of sound propagation. This includes simulation of properties such as decay of sound intensity with distance and angle, diffraction, and Doppler shift. However, for the purpose of musical creation and performance, the user (or rather, musician) may choose to use the 3-D propagation of audio as a medium for musical expression, in which case, perceptually accurate models of sound propagation may not necessarily be desired. Rather, he/she may wish to exaggerate or diminish certain acoustic properties for artistic purposes. Also, rather than having a single propagation model for every part of the scene, the user may selectively specify the characteristics of propagation among particular soundNodes. For example, Doppler shift can be diminished or disabled to prevent detuning of pitched sound material in a tonal music context.

3.4 Control Via Spatial Activity

It is worth noting that all interaction with the environment is achieved through familiar spatial activity. By simply moving around the space and orienting one's head, audio is rendered in different ways. In the simplest case, sounds are spatialized so that they appear to be located in a certain direction. However, more interesting scenarios can occur when we tweak the underlying physical models. For example, offering the user the possibility for extremely directional hearing, where small changes of head position or orientation will change the apperceived mix of different signals or effects. Similarly, a user's spatial activity may be coupled to a sound source. As the user moves, so does sound source, whose audio is directed accordingly into the audio scene, and processed by any sound sinks that are in its path. By varying the directivity (wide or narrow) of sound radiation, the number of sinks in the source's path can varied; thus for example, a variable number of "bus sends" can be managed with just one parameter!

Ultimately, the content of the scene is composed of spatially positioned sound-processing objects that coexist with the user. The progression of a piece of music could be achieved by travelling through virtual space. The user's ability to control how and where audio propagates within the scene suggests a powerful control paradigm, building on the human strength of spatial understanding.

3.5 Organization & Interconnection of Audio

Signal flow in our our DSP graph is based on physical models for acoustics. Unlike conventional analog or digital audio systems, connections between soundNodes (audio sinks and sources) in our DSP graph are not set to unity gain; rather the audio signals are processed (attenuated, delayed and filtered) according to relative spatial proximity and orientation. Complex DSP functionality, such as a multi-voice harmonization, is achieved by grouping (in space) interconnected soundNodes. Thus, the principal operators for mixing (e.g. bussing, panning, filtering) are spatial (e.g. translation, rotation, scaling etc.) or geometric (e.g. directivity of sound). When control systems that use the same parameters (e.g for position, orientation) are directly coupled to these operators, the user's interaction with the application can become remarkably direct, transparent and familiar.

3.6 Novel Solutions & Applications

In the same way that a particular audio signal representation may lend more readily to certain operations (convolution in the frequency domain), certain audio operations are greatly facilitated by our environment. For example, audio mixing is accomplished using a spatial mixing paradigm: virtual sources radiate sound, which is in turn, captured by virtual sinks (e.g. virtual microphones) based on their proximity to the sources. The quality of the resulting mix is given by: 1) the relative placement of these soundNodes in the audio scene; and 2) by the sound propagation model description. Output format specification of a multi-channel mix simply entails locating a given set of virtual microphones (stereo, quad, 5.1, etc.) among the sound sources in the audio scene; the captured audio can then be spooled to disk. Novel applications for *active listening* are also easily imagined and implemented in our environment, given its intrinsic ability to simulate the listener's spatialized audio experience of surrounding sound sources.

4 The *soundEngine*

There exist several libraries and APIs with which to develop interactive virtual environments, including DirectX, OpenAL, X3D, and MPEG-4. Unfortunately, these systems are not really geared towards interaction with music and sound, hence they often have an impoverished audio representation. For instance, most APIs have no method to specify directivity of sounds and instead consider all sources as omnidirectional. In the cases where directional sounds are supported, these are typically implemented with linear attenuation applied as a function of angle. There is usually no support for the complex radiation patterns of traditional musical instruments, or the cardioids that are commonly found in audio equipment. Furthermore, there is often only support for a single listener with a standard speaker configuration (e.g. stereo or 5.1 channel surround). Arbitrary speaker configurations are rarely supported, and the listener is usually assumed

to be standing in a centralized sweet spot that is equidistant from all speakers.

As for simulation of reverberation, there are very few real-time solutions. Creative EAX can be added to simulate rooms of various size. However, the user is forced to choose between presets with few modifiable parameters, which limits the accuracy of the representation of the room geometry. The DIVA (Digital Interactive Virtual Acoustics) project (Savioja et al. 1999) uses ray casting, similar to methods in 3-D graphics, to compute reflection paths of audio signals. Additional virtual sources are created at the reflection points. By modeling surfaces with absorption parameters, they achieve a much more realistic response of the environment.

One noteworthy standard for describing interactive audio scenes is AudioBIFS from MPEG-4 (ISO 14496-1) (Scheirer et al. 1999). BIFS, which stands for Binary Format for Scene Description, is in an extension of VRML/X3D, with but with a focus on describing audiovisual scenes in a compact and object-oriented fashion that ultimately leads to streamable interactive content. The audio component is quite interesting and serves as inspiration for the work presented here. AudioBIFS borrows the scene-graph hierarchy from VRML to organize nodes, but introduces specialized nodes for audio processing. Particularly, defined nodes such as: 'Sound', 'AudioSource', 'AudioFX', and 'ListeningPoint', can be attached to graphical objects anywhere within the scene. This allows developers to create scenes for interactive mixing and musical control within a 3-D context.

Our *soundEngine* implementation is similar to AudioBIFS, using a scene graph to organize content. We use an open source library called OpenSceneGraph (OSG) to manage the arrangement of objects in space, and to efficiently perform any geometric transformations. Our scenes are composed of sound processing entities called *soundNodes*, which contain parameters relating to spatial position and DSP. A *soundConnection* is made between pairs of soundNodes that defines the sound propagation model between them.

PureData (Pd) is currently used as the front-end interface and also as a method to manage the signal processing and audio hardware. Every node in the scene has its own Pd 'patch' where DSP routines can be programmed. The main difference between our system and AudioBIFS however, is that we provide for dynamic control and reconfiguration of the entire scene in real-time. BIFS are always binary in format, hence they must be authored separately and then compiled for distribution. Interactivity is accomplished by allowing certain fields to be *exposed*, and updated by the content server, but this control is not implicit. Rather, the developer must predefine all possible interactions during the authoring phase. In our system, there is no differentiation between an authoring mode or interactive mode; they are one and the same.

4.1 The *SoundNode*

The soundNode is the fundamental building block of a virtual audio scene, and contains a number of important parameters. Table 1 lists these parameters and provides brief descriptions.

ID	Unique identification label
POSITION	Position in 3-D space (x, y, z)
SENS	3-D vector representing direction of sensitivity (sink orientation)
RADN	3-D vector representing direction of radiation (source orientation)
SENSROLLOFF	Rolloff function/table representing attenuations for various angles of sensitivity
RADNROLLOFF	Rolloff function/table representing attenuations for various angles of radiation
CHILDREN[]	List of child nodes whose transforms are relative to this parent
SCALE	3-D scale
GFX	Pointer to the 3-D graphical model
DSP	Pointer to a DSP routine (Pd patch with mono input, mono output)

Table 1: Properties of a soundNode.

A soundNode can be either a *source* (which emits sound), a *sink* (which absorbs sound), or both of these at the same time. The case where a soundNode represents both a source and sink is particularly important in the context of musical interaction since this is how signal modification is realized. We can allow a node in space to absorb sound, apply some sort of signal processing, and then emit the result back into the scene. The node labelled B in Figure 1 shows this dual-function node, while node A is purely a source, and nodes C and D are purely sinks. The DSP can be any arbitrary routine defined by the user, as long as it has one input and one output. Polyphonic DSP is accomplished by grouping nodes together (this is discussed later in Section 4.3).

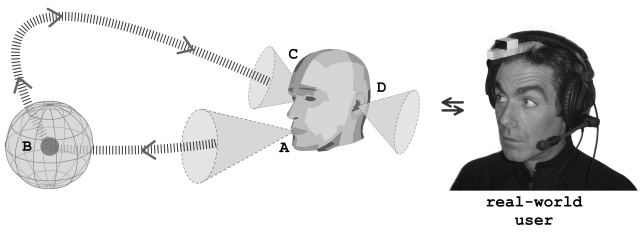

Figure 1: Example of how to model a simple scene with soundNodes.

Source nodes in our representation are similar to those found in many other systems. They typically represent input

from microphones, sound files, or sound synthesis. The concept of sink nodes is however a little more particular. Most traditional systems only render the audio scene for one listener at one position in space. In our representation, the concept of a listener is more abstract since it is just a particular type of sink node (or group of sinks) at a given location in space, where the audio absorbed is written to hardware buffers. Sink nodes are however used variously for other things. For example, the audio scene could be simultaneously rendered at specific locations to produce surround recordings, or multiple listeners with user-specific rendering, could occupy the environment at the same time.

Each node is also represented with a parametric directivity (or roll-off) pattern, giving the user highly accurate control over objects with which a node can interact. This becomes an essential tool when using the framework for spatial signal processing. The nodes can thus become virtual effects units that process audio at some location in space, and a user can mix among them using spatial interactions.

4.2 The *soundConnection*

In conventional systems, every sound source is usually rendered in the same fashion. Other than changing position and orientation, the rest of the audio propagation model (i.e. the directivity, the type of filtering) is statically encoded into the system. However, our representation constructs *soundConnections* between source-sink pairs in the scene. This connection is in fact a data-structure that stores various propagation parameters that describe how the sound should travel through the space between the nodes. Furthermore, it allows the creator of the scene to organize nodes into connected groups that accomplish various DSP tasks, similar to the way that patch cords connect audio equipment. The difference however is that the mixing is not controlled with sliders and knobs, but is instead determined by spatial position and orientation.

The *connection features* described in Table 2 describe some propagation behaviours of a soundConnection. Once a connection is established, a user can specify the intensity of each feature, allowing for various musical effects. For example, a user preparing a scene rich with harmonies might like to prevent frequency shifting and preserve the tonal content of the music. This can easily be accomplished by removing the effects of distance decay and Doppler from a connection, which effectively "teleports" a sound signal between two distant nodes.

Another example becomes evident when we consider a node that processes audio and re-emits a result with greater sound intensity (e.g. a boost or gain). The presence of such a node means that there is no guarantee of diminishing sound energy as audio makes its way through the scene. To prevent the occurrence of intensifying feedback, the user would organize his connection chain to ensure that no loops are present. Conversely, feedback may be desirable from an artistic perspective, so the user may wish to create feedback potential among certain nodes.

DISTANCE ATTENUATION	The attenuation of the sound signal with distance
ANGULAR ATTENUATION	The amount of attenuation relative to the angle of propagation
DOPPLER EFFECT	A variable delay as a function of distance
PROXIMITY EFFECT	A low-shelf filter for small distances (close proximity)
ABSORPTION FILTER	A low-pass filter simulating air absorption as a function of distance
INCIDENCE FILTER	A low-pass filter to simulate frequency-dependent attenuation with angle
REVERB	A filter with an impulse response representative of the current scene size

Table 2: Features of a sound connection.

4.3 Audio Scene Description

The soundConnection object ultimately allows a user to organize soundNodes into graphs that describe how signals are processed. These *DSP graphs* graphs are directed, starting from a source node and terminating at one or more sink nodes. They are also potentially cyclic graphs, meaning that recursive filtering can easily be developed. We note that a single soundNode, as described in Table 1, only has the ability to provide monophonic processing. Polyphony is achieved by interconnecting several soundNodes, and copying signals between them. When these features are considered together, the ability to create audio processing of arbitrary complexity (subject to available computational resources) becomes apparent.

In addition to DSP graphs, there must also be a method of interconnecting soundNodes so that their geometric properties can be shared. For example, in Figure 1, we see that nodes A, C, and D all share a common geometric reference. If the user's head moves or rotates, then all three of these nodes will move and rotate together. We borrow another graph structure from the field of 3-D graphics that describes this relationship, known as a *scene graph*. This is a tree-structured graph where spatial transformations applied to a node are automatically propagated to its children.

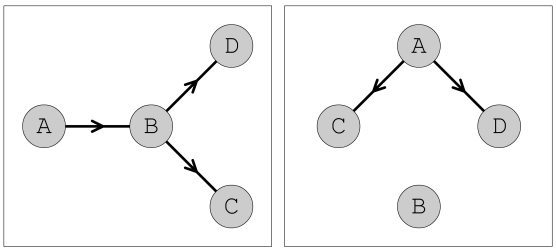

Figure 2: The DSP graph (left) and scene graph (right) associated with Figure 1.

Figure 2 shows both the DSP graph and scene graph associated with the simple example presented in Figure 1. If the user rotates his head to speak in a different direction, the location of the sink nodes representing his ears ears will rotate accordingly. However, node B is not affected since it has no geometric relation to the user's head.

Readers should understand that scene graphs have no bearing on the DSP chain. They are simply structures that organize the geometry of the scene. On the other hand, if there is a cluster of objects that represents a polyphonic sound generator, it may be important that these move through space in a grouped and continuous fashion.

4.4 Computation of audio propagation

In addition to explicit soundConnections made between soundNodes, the propagation of audio in our scene is based on the physics of sound propagation. Since sound travels as a spherical wavefront, it means that the initial sound source energy spreads over an increasing area as it propagates. Thus sound intensity should be attenuated with increasing distance. Additional effects such as reflection, refraction, and diffraction will also occur since sound is a wave. The reflections occur when the audio encounters an object that is much larger than the wavelength of the sound, while diffraction occurs when the wavelength and object size are roughly equal. We will examine a computational method of simulating these properties using the soundNode parameters described in Table 1.

Attenuation based on distance & angle: Figure 3 shows two connected nodes: source A and sink B, with direction vectors specified by $\mathbf{A}_{\text{radn}}$ and $\mathbf{B}_{\text{sens}}$, respectively. This notation is necessary to differentiate the *radiation* of sound from A while node B is instead *sensitive* to sound. For another connection, A may be acting as a sink, in which case vector $\mathbf{A}_{\text{sens}}$ would be used to describe its directional sensitivity.

The most basic model of sound propagation between nodes is to attenuate the signal according to distance and angle. For

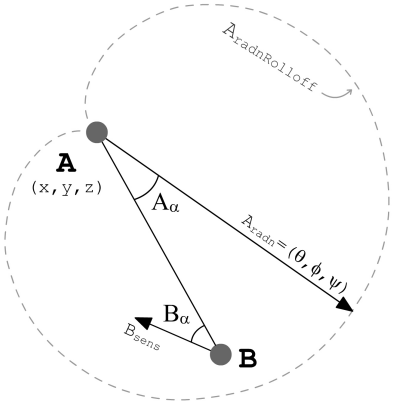

Figure 3: A sound node in more detail.

distance, sound decays exponentially, so we compute a distance gain value as:

$$G_{\text{dist}} = \frac{1}{(1 + |\mathbf{B} - \mathbf{A}|)^{\beta}} \tag{1}$$

This gain value will always be in the range of [0,1], and represents the amount by which the amplitude of the signal should be scaled to simulate decay of sound due to distance. The additional control parameter, β, helps to control the steepness of the exponential decay. When $\beta = 2$, this model is identical to the natural inverse square law (Rossing 1990).

The *angle of incidence*, α, describes the difference between a node's radiation/sensitivity and the vector connecting the source and sink. This can be computed for the source as follows:

$$A_{\alpha} = \cos^{-1}\left(\frac{\mathbf{A}_{\text{radn}} \bullet (\mathbf{B} - \mathbf{A})}{|\mathbf{A}_{\text{radn}}|\,|\mathbf{B} - \mathbf{A}|}\right) \tag{2}$$

The sink incidence, B_{α}, would be computed exactly the same way except using the sink's sensitivity vector $\mathbf{B}_{\text{sens}}$ instead of it's radiation vector.

Each node has a rolloff function that returns an attenuation value based on this incidence value. The user can choose from a variety of such functions, or even specify a lookup table if a desired effect cannot be described algebraically. An example of a common rolloff is the cardioid function shown in Figure 4. If this was the rolloff for sink A, we would expect attenuation values according to the following equation:

$$G_{\text{radn}} = \left[\frac{(1 + \cos(A_{\alpha}))}{2}\right]^{\gamma} \tag{3}$$

We can see that when the radiation direction of the source is directly aligned towards the sink (i.e. the angle of incidence

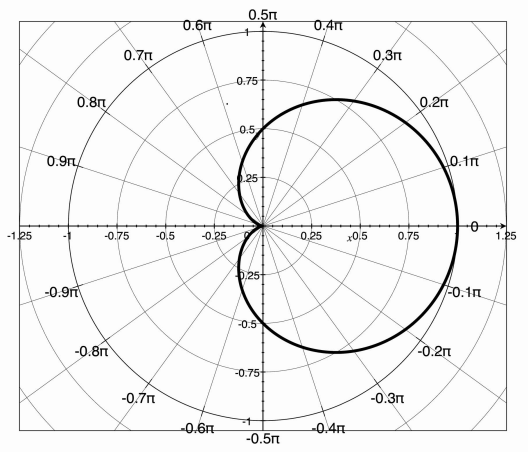

Figure 4: The cardioid function of Equation 3 when $\gamma = 1$.

is zero), then G_{radn} will be 1.0. As the incidence increases to 90°, the gain drops to 0.5 until it finally reaches 0.0 at an angle 180°. There is also an additional control parameter, γ, which describes the sharpness of the cardioid. Figure 4 shows a regularly shaped cardioid with $\gamma = 1.0$. When $\gamma = 0$, it flattens out resulting in omni-directional radiation, while $\gamma > 1.0$ results in a hyper-cardioid.

The final attenuation due to distance and angle is simply the product of these various gain values:

$$G_{\mathrm{final}} = G_{\mathrm{dist}} G_{\mathrm{radn}} G_{\mathrm{sens}} \tag{4}$$

Spatial filtering: It should be noted that the attenuation values computed above are applied to the entire sound signal. However, this is not an accurate model of sound decay in nature, where different frequencies are attenuated to greater extents. For example, high frequencies are quite directional and usually much quieter behind the source while low frequencies tend to wrap around and are heard equally in all directions. Likewise, higher frequencies are also lost due to air absorption as the distance between source and sink increases. These behaviours must therefore be modelled using filters that are parametrized by geometric properties.

For example, the loss of high frequencies behind an object can be simulated with the use of a low-pass filter (that we call an *incidence filter*), whose cutoff frequency decreases with the angle of incidence. The shape of the object will also play an important role, since audio wavelengths that are roughly the same size as an object will diffract and bend around the object. We use a soundNode's GFX (an associated graphical 3-D model) and scale parameters to determine its shape. Many methods from 3-D graphics can be used to extract meaningful information, such as the convex hull (bounding 3-D contour), the average radius, or the volume that the object occupies.

High frequencies are also lost due to air absorption as the distance between the source and sink increases. This absorption or scattering is simulated with an *absorption filter*, which again is a low-pass filter where the cutoff frequency decreases inversely with distance.

A common effect known to sound recording engineers, the *proximity effect*, models the boost of low frequency energy when the distance between source and sink is very small. This is accomplished with the use of a low-shelf filter.

Another spatial effect that we consider is a simulation of reverberation by filtering with an impulse response function. While this impulse function should ideally be computed directly from the scene geometry, this has not yet been implemented in an automatic fashion. For now, the user must adjust a few parameters (1^{st} & 2^{nd} order reflection gains, reverberation gain, room size, reverberation time, etc.) to achieve a desired effect. In future work, we plan to examine the ray casting techniques similar to the DIVA project (Savioja et al. 1999) mentioned earlier.

Doppler Shift: One final computation relates to the travel time of sound between source and sink. If either of the nodes are moving in space, this travel time changes, which results in an apparent shift in the sound frequency. This effect, known as *Doppler shift*, is reproduced using a variable delay embedded in each connection. The delay is is computed based on the speed of sound in air (approximately 343 m/s) and the distance between the source and sink of the connection.

Choosing between connection features: The above computations can be used to a lesser or greater extent in each soundConnection. Each of the connection features in Table 2 has an associated intensity (specified as a percentage from 0-100%). This is an important feature since sometimes it may be useful to construct a scene which is not perceptually accurate, by enabling or disabling some of the connection features. For example, when 'Angular Attenuation' is turned on, it may be impossible to hear something directly behind a sink. This would never occur in nature, but could be interesting since providing listeners with hyper-sensitive hearing might allow for improved auditory navigation, or new musical listening experiences.

Another example which was mentioned before involves Doppler shift. The frequency changes that result from Doppler might ruin musical passages with rich harmonic content. Thus moving objects that contain tonal sound content should probably have Doppler disabled, while objects used for spatialization, simulation, or sound effects could include Doppler.

5 Applications

The ability to simulate not only sources and sinks, but also to chain DSP nodes has opened the door to a host of novel applications for directing and manipulating sound in 3-D. Usually, performing musicians control their music (i.e., samplers, synthesizers, computer processes, etc.) with knobs, pedals, and sliders. We argue that this method requires too much concentration and effort on behalf of the performer. It is very difficult to manage the levels of more than a few effects at once; the performer after all has only two hands and two feet. Also, the state of a slider or knob is not known until the performer actually looks at the current position. By using the entire human body, a performer will always know where he/she is pointing, and multiple effects can be controlled at once using spatial relationships.

To accomplish effects routing using our framework, we define a certain number of sink nodes, each containing a particular type of DSP such as reverb, delay, flanging, ring modulation, etc.. The nodes may be located in the space around the performer, who can direct sound to the effect(s) by just pointing his/her instrument towards them. If the sound is being generated in the computer via sampling or synthesis, the performer can equip their hands with 6 DOF sensors and simply point to the various DSP units.

Additionally, by performing an grasping gesture, the performer can grab and relocate nodes during performance, thus extending musical range. Such ability can prove quite useful, since the performer can group certain nodes together in space so that they are always activated together. Nodes can also be moved closer or further away which effectively controls the gain of the applied effect (due to decay with distance).

Applications relating to *listening* are also well-suited for deployment using our framework. The soundEngine allows us to "focus" hearing in a particular direction or diminish the effect of distance attenuation so that distant sounds are heard more clearly. This ability to finely direct one's listening can lead to novel experiences for audiences; creation can enter into the picture given a particular listener's unique experience of a given scene. The listener can in fact become a composer or re-mixer by placing looped sample nodes about them in a scene and selectively choosing what is heard.

One last note to make is that a virtual scene can be shared between many different people by simply using more sensors and creating appropriate soundNodes to represent the additional users. This opens the door to distributed/network performances and tele-presence applications where performers and audience members share a virtual space even when are geographically separated. By allowing multiple participants to share common DSP units and interact with each other in this spatial context, we foresee many new forms of collective interactive sonic activities.

6 Conclusion

In several audio and audiovisual applications, the representation and interface we propose for working with sound can be clearly advantageous. In music and arts applications, the possibility to discover and invent novels ways to experience, explore and interact with sound and music is invaluable. Though we are at a relatively early stage in our research, we are confident that our approach to physical interaction with sound in space is on the right track, and that it will offer users (including artists) rich and novel possibilities for expressive interaction with sound, music and audiovisual content.

7 Acknowledgements

The authors wish to acknowledge the generous support of NSERC and Canada Council for the Arts, which have funded the research and artistic development described in this paper through their New Media Initiative. In addition, the authors are most grateful for the resources provided to the project by La Société Des Arts Technologiques and to the Banff Centre for the Arts for offering a productive and stimulating environment in which several of the applications described here were prototyped.

References

Begault, D. R. (1994). *3-D sound for virtual reality and multimedia*. Academic Press Professional Inc.

Jot, J. M. (1999). Real-time spatial processing of sounds for music, multimedia and interactive human-computer interfaces. *Multimedia Systems 7*(1), 55–69.

Maki-Patola, T., J. Laitinen, A. Kanerva, and T. Takala (2005). Experiments with virtual reality instruments. In *Proceedings of NIME*, pp. 11–16.

Naef, M., O. Staadt, and M. Gross (2002). Spatialized audio rendering for immersive virtual environments. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pp. 65–72. ACM Press.

Pressing, J. (1997). Some perspectives on performed sound and music in virtual environments. *Presence 6*(4), 482–503.

Rossing, T. D. (1990). *The Science of Sound (2nd edition)*. Addison-Wesley.

Savioja, L., J. Huopaniemi, T. Lokki, and R. Vaananen (1999). Creating interactive virtual acoustic environments. *J. Audio Eng. Soc. 47*(9), 675–705.

Scheirer, E., R. Väänänen, and J. Huopaniemi (1999). Audiobifs: Describing audio scenes with the mpeg-4 multimedia standard. In *IEEE Trans. Multimedia*, Volume 1, pp. 237–250.

Wozniewski, M., Z. Settel, and J. Cooperstock (2006). A framework for immersive spatial audio performance. In *Proceedings of NIME*.