



The ATON Project

Center for Research in Electronic Art Technology
University of California, Santa Barbara
Computer Vision and Robotics Research Laboratory
University of California, San Diego
CalTrans Test-Bed Center For Interoperability



ATON Report 2000.3: A Representation and Infrastructure for Flexible Sound Spatialisation

Stephen Travis Pope (stp@create.ucsb.edu), Frode Holm, and Alex Kouznetsov
CREATE, UCSB, June, 2000

Contents

- Preface1
- Introduction2
- The Basics of Spatial Hearing3
- Spatial Sound Rendering6
- The DRIVE Auralizer Version 18
 - The DRIVE System8
 - The Auralizer9
 - Auralizer Implementation10
 - Evaluation12
- Ambisonic Encoding, Processing, and Playback12
- Design of the Second-Generation Aural Renderer for DRIVE14
- References14

Preface

Within the DiMI ATON Project’s Research Thrust 5.2, we at UCSB are developing a flexible multi-user surround sound processor and 3D sound spatialiser. This document describes our motivations, presents the basics of spatial hearing and psychoacoustics, and introduces our applications for spatial sound and tools for spatial sound processing. We will outline the design of our first-generation real-time surround sound processor (the 1992 DRIVE auralizer), and then discuss a new encoding to be used within the second-generation system we are designing now.

Introduction

The ATON project partners plan to deliver an integrated traffic management system that uses remote sensing, robotics, sophisticated image processing, real-time databases, and advanced virtual-reality-based user interfaces. For this system's various users, we need to be able to generate realistic spatial sound cues over a wide range of sound output devices.

Within the DRIVE (Distributed Real-time Interactive Virtual Environment) system, virtual world objects and events can generate sound, and real-time video feeds can be coordinated with streaming sound input. Our goal is to develop a flexible and scalable multi-user spatial sound renderer for the new version of DRIVE.

Background

There are several domains of computer application that require accurately localized spatial sound for an enhanced user experience, for example virtual reality simulation, "eyes-free" displays and aural rendering (e.g., in airplane cockpits and automobiles), and music performance.

In the past, systems for multi-channel sound output were generally fixed in terms of the number of channels they support and the format they use for representing spatial sound. Simple examples of this are traditional stereo recording, binaural headphone recording, "quad" and more recently 5.1-channel theatrical surround sound.

In order to use VE systems more effectively, we need to integrate immersive output devices such as stereo-optic displays (head-mounted or goggle) and flexible spatialised sound output. Several solutions to the first problem are available off-the-shelf with the advent of low-cost head-mounted stereo-optic displays such as the new generation of immersive "goggles" and 3-D screen-based shutter glasses. To integrate these we need to customize the DRIVE visual renderers for the new output devices. We also intend to use an alternative output system based on the EON VE delivery software.

For increased realism of VE worlds, we require real-time multi-channel spatialised sound processing so that sounds can appear to emanate from their "virtual positions" and can be tracked by the user in harmony with the visual cues provided by the optical renderers. This kind of "aural renderer" has been difficult in the past because of the different latency introduced by the visual and aural output systems, the network overhead, high levels of latency jitter, and the complexity of high-quality models of sound spatialization. For maximum flexibility and scalability, we will develop a stand-alone spatial sound output system, and provide several kinds of interfaces to it: CORBA IDL, direct calls from the APIs of the VE delivery systems we choose (e.g., EON and/or MultiGen), and a low-level socket-based protocol for use by other programs.

The system will support the description of (a) sound "worlds" in terms of the architectural acoustical properties of a space (e.g., a concert hall or an automobile passenger compartment), and (b) the radiation characteristics of the sound sources in that space. The aural renderers for each of the (possibly many) VE users in that space would then perform the real-time digital sound mixing, filtering, and reverberation process (aural rendering) for that

user's particular position in the world. Because the description of the space and its sound sources is kept abstract, different aural renderers can support users with differing sound output hardware (e.g., one user listening over headphones and another with an eight-channel surround sound loudspeaker system) co-habiting the same VE world.

The research contribution of this work will be the new abstractions for sound and acoustical space description, and the real-time distributed aural rendering techniques we intend to develop. Potential commercial products include the description layer (e.g., for scalable computer games) and the several kinds of aural renderers we develop.

Previous Work

Previous and in-progress work at the partner labs has centered on the development of VE systems with diverse input/output media, and on complex world-building tasks (see the DRIVE project). We also have several past and current parallel projects related to spatial sound synthesis, processing, and distribution (performance), specifically, we are intend to update the first-generation spatial sound renderer (or aural perspective engine "APE" or "auralizer") that we build for DRIVE in 1992.

Deliverables

The results of ATON Research Thrust 5 are to be working telepresence systems that demonstrate various modes of navigation and presentation within virtual worlds based on several kinds of models. We will construct several virtual worlds for testing, and will construct navigation software ("vehicles") that use several different interaction devices. The DRIVE renderers will incorporate multi-channel spatial sound and limited-bandwidth video feeds.

The Basics of Spatial Hearing

There exists a significant literature in psychoacoustics that is devoted to understanding how humans determine the characteristics of spaces using aural cues and how we localize sound sources in a 3-D space. As in several other areas of perception psychology, there are several simple, well-understood mechanisms at play, and other, quite complex and still poorly understood, ones that are equally important. It is obvious that it is central to our ability to localize sound sources that we have two ears, that they are rather directional in their reception pattern, are separated by a small (and constant) distance, and face in different directions. Deeper reflection leads to the realization that our ears are asymmetrical in both the horizontal and vertical (cutting through the head) planes, that there are other cues (e.g., Doppler shift, or inter-ear delays), and higher level functions in the brain (see e.g., [Blauert 1983] or [Durlach et al. 1992]) that we use to localize sound sources.

A simple model of sound localization would use direction and distance to characterize the relative geometrical orientation of the sound source and the listener, and provide cues for these, such as relative and absolute amplitude, spectrum envelope and inter-ear delays. Simple reverberation as a cue for the characteristics of the space is also widely used, whereby reverberation time can be keyed to the volume of the space, and the direct-to-reverberated signal ratio is mapped onto the source-listener distance. The more complex relationships between the spectrum of the sound and its location and the characteristics of the space is still

a topic of significant research. It is interesting to note the extremely high time-domain resolution that the auditory system is capable of, especially for inter-aural delays. Some researchers claim on the order of microseconds! (Nordmark 1976)

Looking at the basic perceptual features of a sound, we can make a table relating each of them to one or more aspects of a spatial model as shown in Table 1 below.

- Amplitude (loudness) — distance to source
- Position (in 3D space) — direction to source
- Spectrum (many dimensions) — distance, direction, medium, space
- Reverberation (directional/dispersed sound ratio) — distance, space, environment
- Inter-aural delay time — direction to source (the Haas or precedence effect)

Table 1: Some Mappings between Sound Features and Spatial/Locational Cues

Geometry	Source	Filter (medium)	Listener	Spatial Cues
(seen from the top)				Distance $(d) = (b + a) / 2$
				Loudness $\propto d^2$
				L/R ratio $\propto (b - a) / e$
				Low-pass filter $\propto d^2$
				Direct/reverb ratio $\propto d^2$
				Spatial low-pass filter ratio $\propto \Theta$
				Spatial band-reject filter $\propto \Psi$
				Inter-aural time delay $\propto (a - b) / e$
				Initial/decay reverb ratio $\propto \Psi$
				(many more possible)
(seen from the back)				

There are several approaches to simulating the cues we use to localize sounds. Some of them are based on direct interpretations and implementations of the physics of sound propagation and the anatomy of our ears—so-called “sound ray-tracing” techniques that use the “pinna transform,” also called the “head-related transfer function” (HRTF)—and others are based on simple psychoacoustical insights. We will present an example of the latter kind of model—the type that we are trying to explore and improve upon in the DRIVE auralizer—below.

The Figure above shows the geometrical model and the basic mapping equations used in current psycho-acoustical models of localization. The distances—shown as a and b in the figure—between the source and the listener’s two ears, which are assumed to be separated by the fixed distance e , determine the relative amplitude of the signal that the listener hears. The left-right stereo volume balance is related to the ratio of the difference between $a - b$, and e (shown by the angle Q in the horizontal $[x/z]$ plane). The amplitude scale factor and direct-to-reverberated signal ratio are proportional to the square of the average distance $d = (a + b) / 2$. The absorption of high frequencies by the air between the source and the listener can be modeled as a low-pass filter whose slope (Q) and cutoff frequency vary proportional to d .

There are several ways to model the spectral changes in a signal according to its direction in the horizontal plane. Making sound sources behind the listener “less bright” than those in

front of, or to the side of, the listener is perhaps the most trivial model. This can be simulated simply by a low-pass filter whose Q and cutoff frequency vary (according to some non-linear look-up table) with Q .

The height of a sound source, measured as angle Y in the x/y plane, effects the inter-aural time delay, the sound's spectrum (via the HRTF and possibly the listener's shoulder), and the characteristics of the reverberation. The spectral effects are complex and still poorly-understood, but can be simulated in our "cheap" model by two filters; A fairly broad band-reject or "notch" filter in the upper speech range of frequencies (2-4 kHz) combined with a mild boost in the 7 kHz region. The change in the reverberation signal can be approximated by the addition of more early reflections to the signal, simulating echoes off of the "floor," "walls," and "ceiling" of the simulated environment (Kendall 1989) using a ray-tracing approach to reverberator construction.

This simple geometrical model has all the necessary information to map geometrical properties onto more sophisticated or higher-level cues, such as the provision of more complex filters for the front-back and height cues, and more location-dependent reverberation techniques.

Individual Differences

One of the factors that have frustrated researchers seeking to synthesize the perceptual cues involved in spatial hearing is the central role played by individual difference in ear size, shape, location, body volume, lung volume, etc.

This Figure shows physical wave reflections and the resulting filter frequency responses for sound coming from two different directions (above and in-front). It is obvious that subtle differences in ear physiology make for great differences in these spectra, and that each of us has learned over our entire lives to "listen through our ears."



There are many other details of localization, such as what cues we use to differentiate between sounds that have the same inter-aural time delay—those that lie on the so-called "cones of confusion" that surround each ear and are symmetrical with respect to the medial plane (the y/z plane passing through the listener's head between the ears)—and how we determine the difference between the characteristics of the space and of localized sound sources based on the nature of the signal's reverberation. These are, however, beyond the scope of our present model. The interested reader is referred to (Chowning 1971) or (Moore 1989) for more detailed discussions.

Spatial Sound Rendering

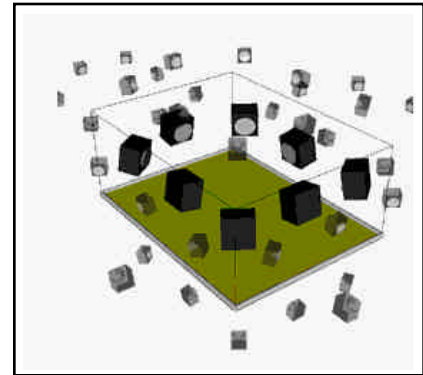
The Source-Filter-Listener Model

To better understand the psychological cues we use to localize spatial sound, we use a source-filter-listener model to disambiguate a sound source's position and orientation, its (possibly direction- and frequency-dependent) radiation pattern, the carrier medium's filtering characteristics, and the geometry and perceptual effects of the listener.

Spatial Modeling, Reverberation

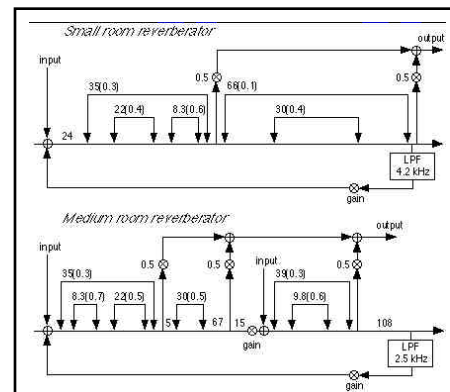
To model sound reflections and dispersion in a non-open environment such as a room or a concert hall, we can think of sound reflections from walls) as constituting "virtual sources," i.e., extra copies of the sound source that are repositioned, attenuated, and filtered according to the "virtual" distance, and the characteristics of the reflecting/dispersing surface.

The figure on the right shows a modelled room with 8 loudspeakers and (smaller) the virtual source loudspeakers for the first few reflections of each source off of each surface in the room. Clearly, the compute requirements for spatial rendering of complex rooms with highly reflective surfaces are quite large.



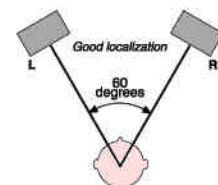
To write software that simulates the reflections of a reverberant sound environment, one often directly models the reflections and virtual sources using temporal delay lines, multiplying attenuation, and spectral filtering, as illustrated in the Figures to the right.

The top part of the figure shows the signal processing flow chart for a spatial reverberator for a small room, while the bottom flow chart describes a reverberator for a medium-sized room.

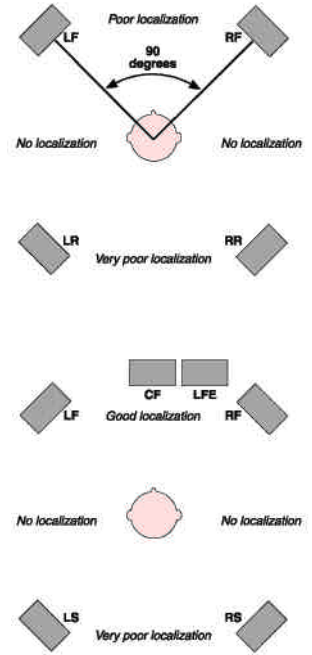


Standards for Multi-Channel Sound Playback

Throughout the long history of multi-channel sound recording, distribution, and synthesis, there have been many formats for standardizing recording format and speaker placement for 2, 4, 5, 5.1, 7.1, 8, and more channels of sound. Standard stereophonic reproduction assumes two speakers at an angle of approximately 30 degrees, and between 6 and 20 ft. of the listener in or near the media horizontal plane. This speaker placement is shown in the Figure to the right.



This orientation cannot be maintained for 4-channel quadrophonic playback without changing the aspect ratio of the “quad square” to an extremely long/narrow rectangle. This would have a strong negative impact side placement. The speaker placement for “square” or 90-degree quadrophony is shown in the Figure. Note the compromise in the front speaker placement, and the related deterioration in the quality of the front stereo image.



More recent video-oriented home theater surround sound systems add a front-center channel, and a low-frequency “sub woofer/effects” channel, giving “5.1” channels. The Figure to the right shows the standard speaker placement for a 5.1-channel home theater system. To fill in the side images on some systems, two additional direct-side speakers are placed, giving 7-1 channels (and better circular localization)

Pluriphonic (Many-Channel) Systems

Electroacoustic music composers have used many-channel (pluriphonic) sound playback systems since the 1930s, and especially since the advent of multi-channel tape recorders in the 1950s and of digital multi-track recorders in the 1990s. These two Figures show different configurations of the UCSB Creatophone spatial sound performance system, which can entail up to 20 channels of processing and playback at present.

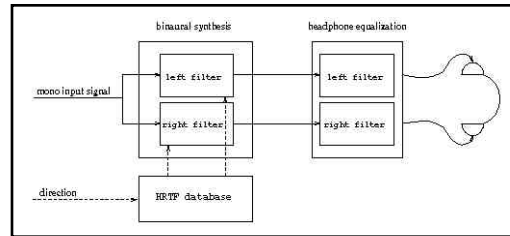


Processing Sound for Multi-Channel Playback

There are many alternatives for creating and performing pluriphonic sound. Standard source and interchange formats support multi-channel transfer in track multiples of 8) through such systems as the Alesis ADAT and Tascam DA-88 formats (both of which store 8 channels of digital sound on video cassettes).



For playback through headphones, one can use the geometrical information about the sound sources, and a database of individual hearing response data to accurately place sounds in the listener's acoustical space.

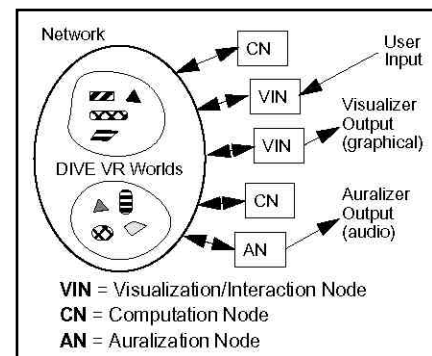


The DRIVE Auralizer Version 1

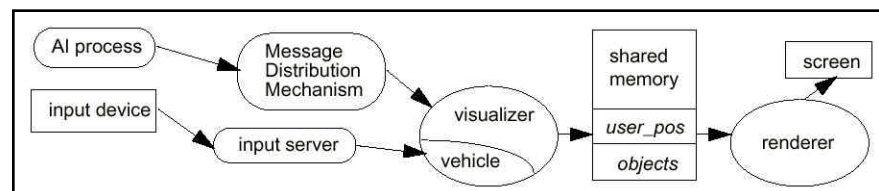
Most VR systems, are mainly concerned with a 3-D rendering of a simulated virtual world, and with user navigation among, and interaction with, objects in this world (Hesler and Roth 1991). In these systems, some “database” of simulation-style or purely graphical objects exists, along with some data about the “user’s” position in the world (the position and perspective from which the world will be rendered). This data is used by the rendering component, which generates a visual display. The user interaction, database management, and rendering systems can be relatively independent of each other (as they are in the DRIVE), with the interface asynchronously driven by changes in the user’s position and the state of the objects in the world each time the renderer generates a new frame of video information for display. The renderer can thus be said to “poll” the world and user state for each frame it displays.

The DRIVE System

The basic unit of activity in the DRIVE is the AI, or application—a UNIX process running on some node in the network that updates the shared world and object databases or broadcasts change messages via the system’s event distribution mechanism (currently based on the ISIS distributed programming tool kit [Birman and Cooper 1990]). An AI might, for example, represent a clock that updates itself every second, distributing messages throughout the network as to its new visual appearance. Visualization AIs that are in the same world as the clock would receive this message, and update their renderings of it in case it is in the user’s field of vision.



Nodes that mainly interact with the shared database are called “computation nodes” (CNs) and nodes that interact

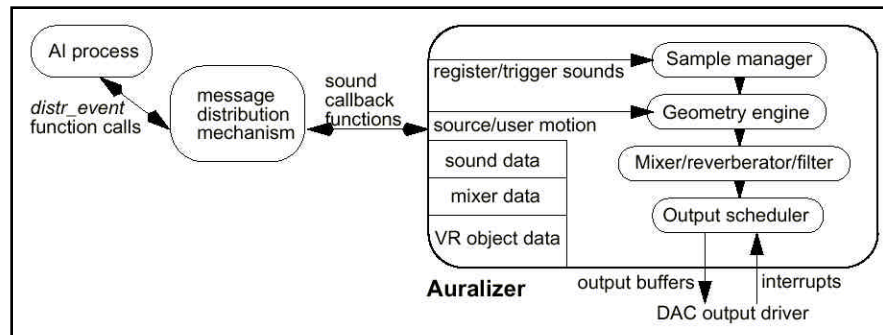


with the database and the user interaction and some kind of visualization are called “visualization and interaction nodes” (VINs). The DRIVE architecture allows multiple “virtual worlds” (which may or may not be connected by “gateways”) to be active on the same network, with one or more users in each of them at any one time. It is this distributed, multi-world, multi-user and non-server architecture that is perhaps the main interesting feature of

DRIVE. Figure 2 shows a schematic overview of the architecture. Visualization and interaction nodes manage the graphical output and gestural input for the user. The visualization nodes can potentially support stereo-optic output systems such as head-mounted stereo displays. Computation nodes can execute AI processes such as ticking clocks (i.e., those that generate animated, possibly with some kind of “semi-intelligent” behavior, virtual objects in the synthetic world), or “supporting” processes such as a collision detector or gravity simulator, which update the state of other objects in the world database without introducing any of their own. The degenerate case is that all of these processes are running on the same workstation, though we use between three and six in common practice.

The Auralizer

The Auralizer node shown in the lower-right part of the DRIVE system architecture Figure above is the component that we have added to the system in this project. Like a visualizer, it will “render”



the state of the world object data base, but using audio (2 channels at present) instead of video as its output medium. This architecture is illustrated in a different schematic representation in the Figure above, which shows the flow of control, and the residency of world data between an AI and the processes in a VIN. The AI also has its own copy of the shared memory (not shown here); more importantly it sends out messages that are distributed among the processes in its world (e.g., other AIs or visualizers). These messages will generally change the state of the shared object data in each participating member. The input server—e.g., one connected to a 3-D input device such as a magnetic tracker—updates the visualizer’s notion of the location and orientation of the virtual “user,” which defines the perspective from which the graphical model will be rendered. The visualizer is broken up into the components called the visualizer proper, the vehicle (which controls the mapping between the input device and changes in the user’s location and orientation), and the renderer (which generates image frames at some specified target rate, typically 10-20 Hz).

To incorporate audio output into such a VR system, it is necessary to extend the simulated object model of the virtual world so that “events” can trigger sounds, and so that continuous sounds can be produced by AIs or other components. This means that the auralizer must have a different architecture than a visualizer/renderer, and must be connected to AIs in a different manner. Returning to the clock example we mentioned above, if the clock advances one second, it changes its graphical rendition as it is stored in the shared database (it sends out a message via DRIVE’s distribution mechanism), and assumes that any renderers that are “watching” will update their renditions of it (perspectives on it), as appropriate. If this clock wants to make a “tick” sound (or to say the word “tick,” or to play an e-minor guitar chord), it is likely that it must broadcast that information in a different format than it uses for updat-

ing its graphical perspective. This is because of the architecture described above, whereby renderers “poll” the state of object memory when rendering each frame of graphics, whereas sound events can (and will) occur asynchronously and must be scheduled with less latency than graphical ones. The fact that audio is not “frame-based” is an important difference that will naturally effect how the two media are handled differently.

There are a great many issues in the software architecture of an aural renderer for a system such as DRIVE; they are described in more detail in (Pope and Fahlén 1993) and primarily involve how one models event distribution vs. polling for handling sound triggering and user position updates, and the software factoring of the digital signal processing needed for implementing the localization model so as to be portable, easily extensible, and distributable among several workstations. We have also avoided the issue of AIs that generate their sounds in real time by requiring them to pre-register sound samples, giving a sound file name and a sound index and perspective (which can be thought of as a sample name and channel and key numbers in MIDI synthesizer terminology). We do this because we do not want to have to address the issues of network bandwidth with multiple real-time sound sample streams, or of abstract timbre description languages, in the present system, which is intended for experimenting with localization models and 3D audio environments.

Application programs “register” sounds with the Auralizer by sending out messages that include their own unique identifiers (“client IDs”), a sound file name, and a sound index and perspective. The AI can later “play” this sound by sending out a message that includes the above information (without the sound file name), along with their current positions, and the relative amplitude of the sound (and a few other parameters that are beyond the scope of this introduction—see below).

Auralizer Implementation

The Auralizer runs as a separate process, probably on a network node with high-quality stereo audio output hardware, such as a suitably-equipped Sun or SGI workstation. It maintains a table of the sounds that have been registered by AIs in the current world and responds to sound output messages by playing the chosen sounds, possibly mixing them with other active sounds and spatialising the result to provide for an aural model of the virtual space. To trigger a sound, the Auralizer needs to know which sound is being requested, the position of the sound source (possibly in motion), the position and orientation of the listener, and the characteristics of the virtual space. It will use this information to select and process the stored sample according to the source/filter/listener model described above. There can be several types or levels of auralizers depending on the amount of computation power that can be dedicated to the auralization (i.e., if it is running on the same workstation as the visual renderer, or if special digital signal processing (DSP) hardware is provided), and on which output format is being used (e.g., monophonic CODEC or multichannel CD-quality output). The first auralizer that was built mixed 16-bit 8 kHz monophonic sound samples into a stereo aural image using left-right sound placement and amplitude as spatial cues. The current generation runs at 16 kHz and incorporates stereo reverberation, a simple filter-based front-back directional model and inter-ear time delay. More sophisticated filters (modeling the HRTF),

and dynamic reverberation algorithms to provide up-down directional cues and more stable cues in general are planned.

The method of integration of the Auralizer into the DRIVE system was debated for some time, and several approaches were prototyped. The final design was factored into several components: (1) the interface to the network distribution mechanism; (2) sample structure storage and management; (3) the geometry functions for determining the relative locations of sound sources and mapping the geometry onto the parameters of the mixer, filter and reverberator; (4) sound mixer, filter and reverberator; and (5) the real-time output handlers for the DACs. The rough architecture of the APE and its interconnection with the rest of DRIVE is illustrated in Figure 4.

The auralizer's sample storage uses an model of a multi-dimensional array mentioned above. The items stored in this array are special structures that include the actual samples of the sound, its duration, sample rate, sample format, and other data. When a client registers a new sound, the auralizer allocates space for the sound's sample array, reads the samples from the disk, and places the sound structure in the array such that it can be located later when the client wants to play it. (The addressing and structure management to do this in real time is non-trivial.) The second component of the Auralizer is its geometry functions. Each time a client plays a sound, it is possible that both the client and the listener may have moved, so it is necessary to get the coordinates and orientation of the user (defined as the user's "eye" vector in DRIVE), and of the client AI in order to determine the relative position and amplitude of the source. In our simple initial system, we used the relative positions to set the stereo position and scale the loudness of the sound samples. The more sophisticated current models also use inter-aural time delay, non-linear front-back filters, and a configurable reverberator. A client can also register an event for continuous updating of its position if, for example, it is moving quickly or is making a sound with a long duration. This should probably be the default case, but the current mechanism for calculating relative locations continuously was thought too computationally expensive for the present system.

The sound mixer functions can manage multiple active clients at different locations, and perform the summation of up to 32 monophonic sources into a stereo output stream. The reverberation model adds a small amount of reverberation to the mixed sound sample array. The ratio of the direct and reverberated signals is dependent on the distance between the source and the listener. The interface to the network distribution mechanism uses the DRIVE's platform-independent tools to communicate via the ISIS system with AIs and visualizers. It registers handlers for the collection of callback functions that AIs can call with sound output, source movement, or other auralizer-related messages. Finally, the real-time drivers and interrupt handlers for the sound output via the DACs are implemented so that different DAC hardware can be substituted with relative ease (hopefully).

The real-time I/O module must handle the interface to the low-level sound output functions or de-vice driver, and the setup and clean-up for each mix block. It must also know what to do if the system cannot keep up with real-time, i.e., if the next buffer is not ready when the output driver generates an interrupt.

Evaluation

The first-generation DRIVE aural renderer performed acceptable (by 1992/3 standards) and supported multiple users (up to 5 or 6 per world) and relatively complex worlds (up to 20 sources, typically), with reasonable sound fidelity (16 kHz sample rate, 2- or 4- channel output). We did identify scalability problems in each of these dimensions, and hope to design a new version of the same basic architecture that (1) uses a more advanced spatial processing model, and (2) is implemented on top of a very high-performance distributed processing infrastructure from the ATON HPDM Project.

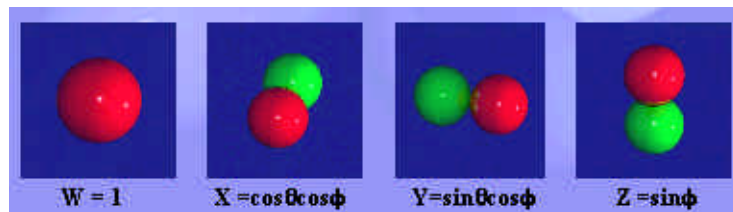
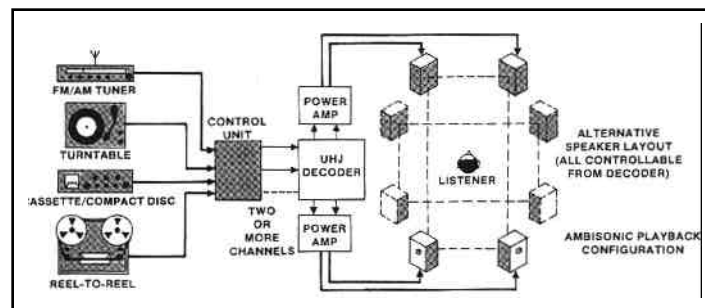
Ambisonic Encoding, Processing, and Playback

In order to provide a more efficient, flexible, and scalable surround sound renderer for DRIVE, we are investigating the use of the Ambisonic coding of spatial sound signals. In this representation, spatialized sound is not represented according to concrete channel signals (as in stereo, 5.1-channel home theater surround, etc.), but rather into abstract directional information based on simple geometry

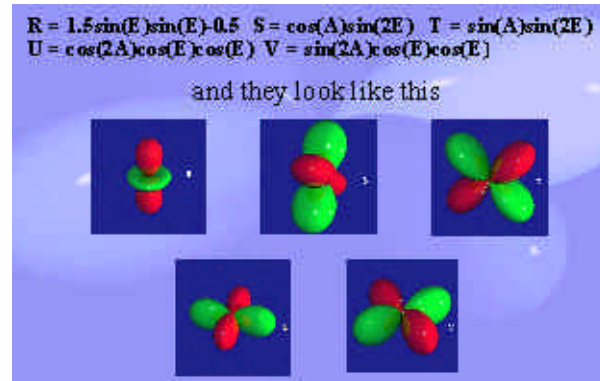
Ambisonic sound encoding stores the sound radiation pattern decomposed as a polynomial equation of four or eight terms involving the spherical harmonic series.

This means that a system that uses the first-order (4-term) Ambisonic representation stores spatialized sound as 4-channel data, but can render (perform) this for any number of channels (including 8 or more).

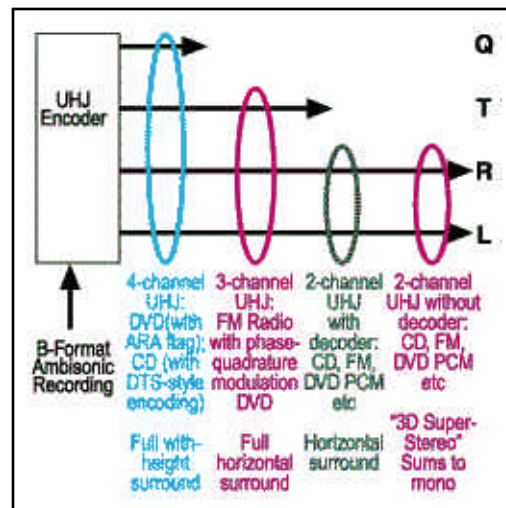
The first-order polynomial involves the non-directional (mono) signal W and figure-8 shaped spherical functions that point in the 3 spatial dimensions of x , y , and z . This is shown in the Figure to the right, and described by the equations under each picture.



The second-order spherical harmonics are shown with their equations and function plots in this Figure. These are used in advanced 8-track Ambisonic systems that can generate even more convincing spatial cues (including a stronger height cue).



Given a stored (or real-time-generated) Ambisonic signal in 4, 6, or 8 tracks, one can produce a realistic spatial projection over any standard speaker geometry, including headphones. Encoders for various Ambisonic formats process (typically 4-channel) information into other surround sound formats.



An (albeit slanted) comparison of Ambisonic surround and 5.1-channel home-theater surround sound is given in the Table to the right.

Design of the Second-Generation Aural Renderer for DRIVE

There are two main areas where the 1992-era DRIVE auralizer needs to be updated: (1) the basic representation used for spatial sound, and (2) the renderer itself has to be broken up into independent (and potentially distributed) components for user/source tracking, geometry mapping, sound synthesis or streaming input, generation of spatial cues, reverberation, filtering, and mixing. Each of these two areas will be the focus of concrete efforts during the Summer of 2000 and beyond.

Ambisonics	5.1
Only requires four channels.	Needs six channels.
Includes height information.	No height information.
Only one mix required for DVD.	At least two mixes required for DVD.
Lower data density: lossless compression can be used.	Higher data density: lossy compression needed.
Imaging equally accurate in all directions.	Imaging only accurate across front stage.
Imaging possible between the speakers.	Almost non-existent inter-speaker imaging.
Sounds can be placed within the array.	Sounds appear around the edge of the array.
Speakers can be placed almost anywhere.	Speakers must be in special positions.
'Sweet spot' large enough to enjoy image outside the array.	'Sweet spot' only covers a small area: image changes as you move.

References

The final reports from the CREATE's two previous (1997) projects in the ATON domain are available on the Web from the following references.

IDOT (Impact of Distributed Object Technology Using CORBA):
<http://www.create.ucsb.edu/idot/report.html>

DRIVE (Distributed Real-Time Interactive Virtual Environments):
<http://www.create.ucsb.edu/drive/phase2/report.html>

Distributed VE

Frécon, Emmanuel and Märten Stenius. "DIVE: A scaleable network architecture for distributed virtual environments." Available at <http://www.sics.se/~emmanuel/publications/dsej/>. See also <http://www.sics.se/publications/dive.html>

Hagsand, Olof. "Interactive MultiUser VEs in the DIVE System." IEEE Multimedia Magazine, 3(1), 1996.

"Standard for information technology, protocols for distributed interactive simulation." DIS-ANSI/IEEE Standard 1278-1993, American National Standards Institute, 1993.

Pope, Stephen T. et al. "CREATE DRIVE Project: Distributed Real-Time Interactive Virtual Environments: Phase 2 Final Report." See <http://www.create.ucsb.edu/drive/phase2/report.html>.

Spatial Sound

Adler, D. "Virtual Audio - Three-Dimensional Audio in Virtual Environments." SICS Internal Report, <ftp://ftp.sics.se/pub/SICS-reports/Reports/SICS-T--96-03--SE.ps.Z>, 1996, ISRN: SICS-T--96/03-SE.

Begault, D. R. 1991. "Challenges to the Successful Implementation of 3-D Sound." *Journal of Audio Engineering Society* 39(2): 864-870.

Birman, K., and R. Cooper. 1990. "The ISIS Project: Real Experience with a Fault-Tolerant Programming System." *Proceedings of the Workshop on Fault-Tolerant Distributed Systems*. New York: ACM Press.

Blauert, J. 1983. *Spatial Hearing*. Cambridge, Massachusetts: MIT Press.

Carlsson, C. and Hagsand, O. 1992. "The Architecture of the MultiG Distributed Interactive Virtual Environment." *Proceedings of the Fifth MultiG Workshop*. Stockholm: KTH.

Chowning, J. 1971. "The Simulation of Moving Sound Sources." *Journal of Audio Engineering Society* 19: 2-6.

Durlach, et al. 1992. "On the Externalization of Auditory Images." *Presence: Telepresence and Virtual Environments* 1(2): 251-257.

Fahlén, L. E. 1991. "The MultiG Telepresence System." *Proceedings of the Third MultiG Workshop*. Stockholm: KTH.

Helsel, S., and J. Roth, eds. 1991. *Virtual Reality Theory, Practice, and Promise*. Menlo Park, California: Addison-Wesley.

Kendall, G. S. et al. 1989. "Spatial Reverberation: Discussion and Demonstration." in M. V. Mathews and J. R. Pierce, eds. *Current Directions in Computer Music Research*. Cambridge, Massachusetts: MIT Press: 89-103.

Kendall, Gary. "A 3-D Sound Primer: Directional Hearing and Stereo Reproduction." *Computer Music Journal* 19:4, Winter, 1995.

Kendall, Gary. "The Decorrelation Of Audio Signals and its Impact on Spatial Imagery." *Computer Music Journal* 19:4, Winter, 1995.

MacDonald, Alistair. "Performance Practice in the Presentation of Electroacoustic Music." *Computer Music Journal* 19:4, Winter, 1995.

Malham, David G. and Andrew Myatt. "3-D Sound Spatialization using Ambisonic Techniques." *Computer Music Journal* 19:4, Winter, 1995.

Moore, F. R. 1989. "Spatialization of Sounds over Loudspeakers." in M. V. Mathews and J. R. Pierce, eds. *Current Directions in Computer Music Research*, Cambridge, Massachusetts: MIT Press: 65-87

Nordmark, J. O. 1976. "Binaural Time Discrimination." *Journal of Acoustical Society of America* Vol 60: 870-880.

Pope, Stephen T. "The Use of 3-D Audio in a Synthetic Environment." in Proceedings of the 1993 International Computer Music Conference.

Pope, Stephen T. "Sound and Music Processing in SuperCollider." See <http://www.create.ucsb.edu/htmls/sc.book.html>

Roads, Curtis, Stephen T. Pope, Giovanni De Poli, and Aldo Piccialli, eds. "Musical Signal Processing." Swets and Zeitlinger, 1997.

Rocchesso, Davide. "The Ball within the Box: A Sound-processing Metaphor." *Computer Music Journal* 19:4, Winter, 1995.

Wenzel, E. M. 1992. "Localization in Virtual Acoustic Displays." *Presence: Telepresence and Virtual Environments* 1(1): 80-107.