An Extensible Toolkit for Creating Virtual Sonic Environments

Hesham Fouad
Naval Research Laboratory
Code 5707
Washington DC 20375 USA
+1 202 404 3736
hesham@acm.org

James A. Ballas
Naval Research Laboratory
Code 5513
Washington DC 20375 USA
+1 202 404-7988
ballas@itd.nrl.navy.mil

Derek Brock
Naval Research Laboratory
Code 5513
Washington DC 20375
+1 202 767-0491
brock@itd.nrl.navy.mil

ABSTRACT
The Virtual Audio Server (VAS) is a toolkit designed for exploring problems in the creation of realistic Virtual Sonic Environments (VSE). The architecture of VAS provides an extensible framework in which new ideas can be quickly integrated into the system and experimented with.

Keywords
Virtual environment, sound server

INTRODUCTION
Producing an immersive auditory experience of any realistic environment is very difficult. Although audio technology has rapidly advanced, there are still some fundamental challenges that must be addressed. Foremost among these is the challenge of creating phenomena that are ephemeral. The fleeting nature of sound means that the auditory designer does not have a permanent entity to examine, analyze, copy and compare, as does the graphics designer. Any written description is wholly inadequate and acoustic measurements such as an impulse response, spectral and amplitude measures are typically incomplete and/or impossible to obtain. For example, in our applications we are interested in recreating military environments. But actual military battles are imprecisely documented, and the only known example of a recreated battle environment (75 Easting, reconstructed by the Institute for Defense Analysis) relied on participants' memories to recreate the sounds.

Therefore, the audio designer can only approximate the auditory environment but even this will require a flexible auditory scene modeling and rendering system. The system must be able to render sounds that will be attended as well as sounds that are part of the ambient. Sounds that might be attended will have to be precisely rendered in time, space and action. The system must also be able to render the ambient and therefore be capable of producing a large number of sounds. The system must model sounds that come from discrete events and must coordinate sounds that are interrelated. The system should be independent of the particular rendering employed since some VR environments are best rendered with speakers, others with headphones [5]. Finally it should have a control language written at the level of auditory scene elements so that the designer does not have to specify the minute details of sound control.

THE VAS FRAMEWORK
The Virtual Audio Server (VAS) was developed so that the problems associated with developing such auditory environments can be readily explored. The primary use of VAS is as a framework for studying problems associated with the creation of Virtual Sonic Environments (VSE). As such, the primary driving force behind the design of VAS is extensibility. This is necessary in order to enable the integration of new ideas into the toolkit without having to redesign the whole system. VAS is composed of four independent subsystems (fig. 1): High-level modeling constructs provide a basis for studying new techniques in modeling VSEs. Sound source processing is designed so that new sound representations can easily be integrated into the system. A rendering subsystem enables the seamless integration of novel spatialization techniques without affecting any of the existing design. Finally a scheduling mechanism enables the study of real-time scheduling techniques for the sound generation process.

Providing this level of extensibility requires well defined and functionally complete interfaces between each of the subsystems so that any new functionality can be accommodated. The use of object-oriented techniques greatly facilitates the definition of such interfaces through encapsulation, while inheritance makes the extension of existing functionality possible.

While other very useful tools have been developed by researchers for creating VSEs [1, 2, 4, 8, 9, 11], we believe that our goal of establishing an extensible toolkit for research is unique. Production oriented toolkits such as DirectSound3D™ and A3D™ do not provide the flexibility necessary to support research development. The integration of new spatialization techniques, modeling constructs, or resource management schemes into such toolkits is not supported.

 In the following sections we discuss the salient features of each of VAS's subsystems and describe how extensibility is achieved. We also describe some of our experiences in extending VAS to provide new functionality that was not available at the conception of the system.

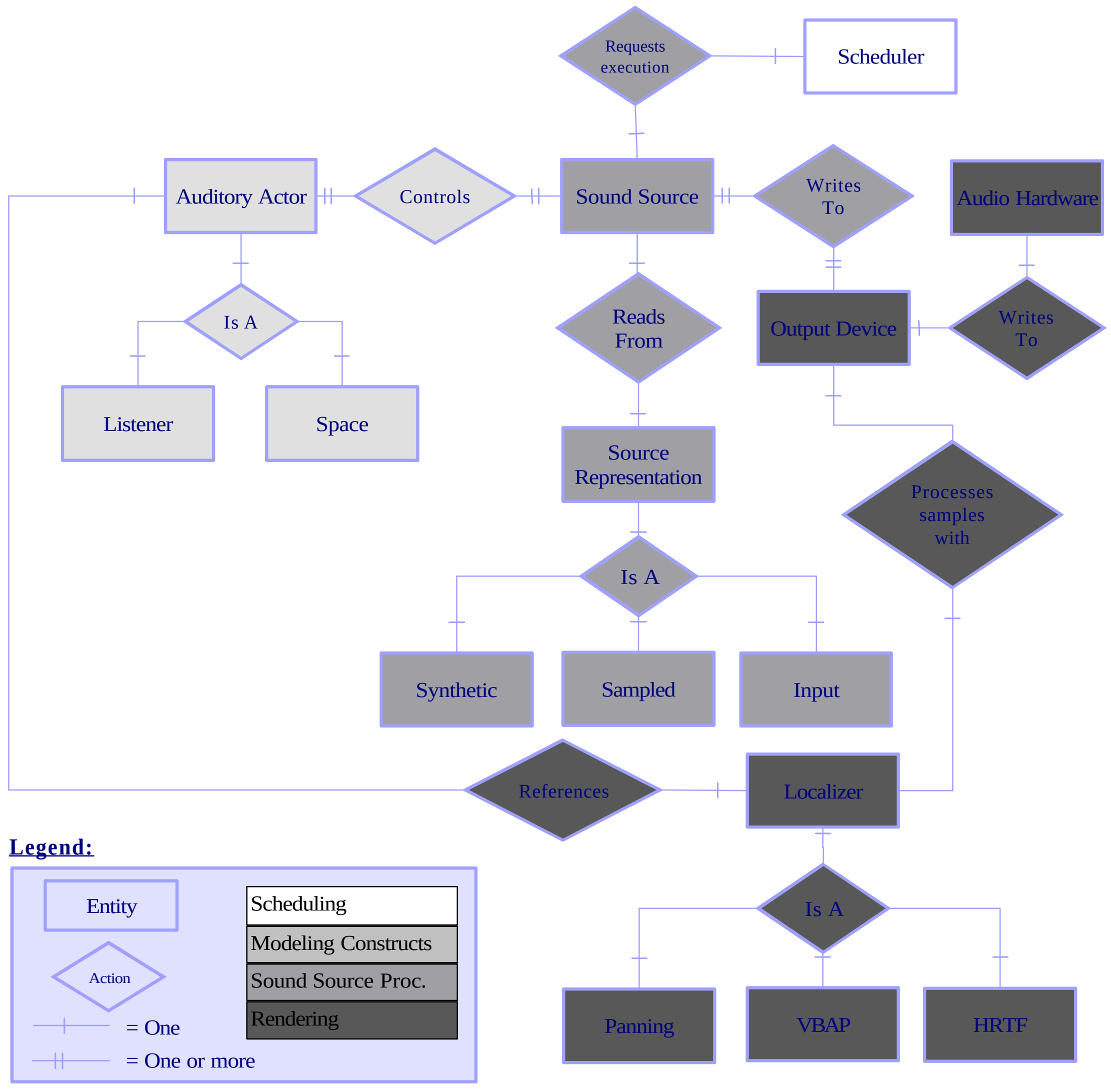

Figure 1 An entity relation diagram of VAS

Modeling Scene Elements

A Virtual Sonic Environment can be modeled concisely as a collection of sound sources, a location for the listener, and a collection of occluding objects. However, a representation based on these elements does not take into account the inherent logical groupings of sounds that occur in natural environments. Taking such groupings into account can facilitate the user's interaction with a VSE, especially when the sounds are correlated with graphical elements. We can look at the problem of sonically modeling a city street as an example. One of the main sound producing entities in such an environment are cars streaming past the listener. We can consider each instance of a car as four independent sound sources (sounds emanating from the engine, exhaust, wheels, and horn) each with a location and behavior. Such a model will unnecessarily burden the application because instead of dealing with one entity "car", the application is dealing with four independent entities that don't necessarily have any logical meaning to the application.

VAS allows such groupings to be modeled directly using the Auditory Actor construct. An Auditory Actor defines a local coordinate system where sound sources can be positioned and oriented relative to a fixed frame of reference. Once scene elements have been defined using the Auditory Actor, an application can manipulate a single entity without regard to how the individual sources are located.

The control of sounds associated with an Auditory Actor is encapsulated in the actor. Sounds can be controlled manually by the application, through scripting or through an application-defined reactive control mechanism that can be associated with an Auditory Actor. This mechanism responds to application defined events by controlling the sounds associated with the Auditory Actor. This creates interesting possibilities in exploring novel sound control schemes.

As it turns out, the Auditory Actor provides a good foundation for deriving other elements comprising a VSE. For example the listener in VAS is a specialization of Auditory Actor. In addition to the functionality provided by the Auditory Actor, a listener maintains a head orientation and tracks the position of the listener's ears when the head is moved. This information is necessary for localization algorithms to render the environment. A logical grouping of sounds occurs with the virtual listener in a VSE due to self-produced sounds generated by the listener. This allows applications to easily attach near field sound cues such as movement and collision to the listener.

VAS models distinct spaces in a VSE as an extension of the Auditory Actor called an Auditory Space. VSEs can contain spaces that may or may not be enclosed, but none the less have unique characteristics in terms of ambience. Because the ambient sounds that occur in a space are naturally grouped, the Auditory Actor provides good foundation in order to model such spaces. The sounds associated with an actor modeling a space consist of the ambient sounds in the space. These sounds are generally not localized while the listener is inside the space since that they appear over the extent of the space. Barriers can be included in a space so that enclosures can be modeled.

Making Sounds

Below the high level modeling constructs used by applications in creating VSEs, VAS uses a set of lower level modeling construct to generate sounds. A sound source in VAS consists primarily of an execution thread for sound source processing. The interface to a sound source provides the ability to control the play behavior of a sound and control its playback parameters. Sound sources do not encapsulate a representation of the sound or a particular rendering method. A sound representation is modeled as an independent entity with its own interface. A sound source makes use of that interface to evaluate the representation at successive intervals in order to produce a sample stream. Once a sample block is produced, it is written to an output device object, which invokes a localizer to process the samples, and then outputs them to the audio hardware.

An output device provides an interface between the process of generating a sample stream, and the rendering of the sound. Because multiple listeners can coexist in the same VSE, multiple output devices may be associated with a sound source. Each output device renders a version of the VSE based on the position of its associated listener (fig. 2).

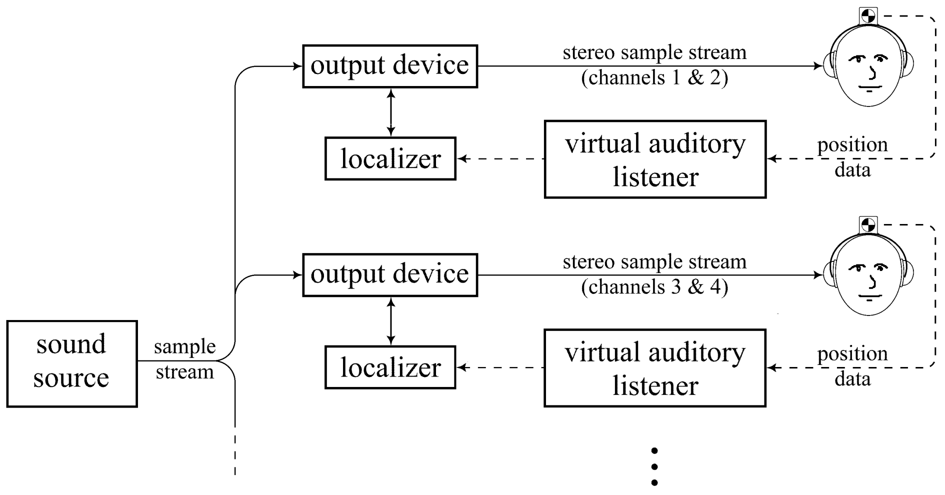

Figure 2 Sound rendering in VAS

The rendering mechanism invoked by the output device is an independent entity with a well-defined interface. The choice of localization algorithm is specified at the creation of the sound source. The following localization options are currently supported:

- Intensity Panning **-** uses 2 or 4 speakers to localize a sound in 2D.
- Vector Base Amplitude Panning – can use any number of speakers to localize a sound in 3D.
- HRTFs – Head Related Transfer Functions are used to localize sounds using loudspeakers or headphones.

The design of the sound source supports extensibility because the representation of a sound is independent of the sound source processing; new representations can be readily added to VAS. Also because the rendering of sounds is an independent mechanism, any number of rendering techniques can be incorporated into VAS. In fact, each sound source in a VSE can utilize a different localization technique. Finally this design enables the support of multiple listeners in the same VSE because each output device will render a different version of the VSE for each listener.

Ensuring Real-time response
As mentioned earlier, a sound source provides the thread of execution in which sound source processing occurs. In fact the sound source makes use of a scheduler in order to manage the execution of this thread. VAS provides the option of utilizing a built-in real-time scheduler for managing the execution of the sound source processing. When overload conditions occur, this scheduler utilizes a graceful degradation scheme in order to maintain the real-time generation of sound. Details of this work can be found in [6, 7].

RECENT EXTENSIONS

Three capabilities were recently added to VAS demonstrating the flexibility of this approach. For the first, VAS was used to generate spatialized audio in an augmented reality system for mobile military combatants (fig. 3). Users of this augmented reality system were to be able to receive augmenting sound including communication from team members, threat alerts, or navigation beacons. By spatializing the team communications, the user would get position information about the other members of the team. The threat alerts would provide direct information about the location of the threat, and the navigation beacon would provide a source to walk to. The system broadcast three sound streams to a pair of headphones mounted in a

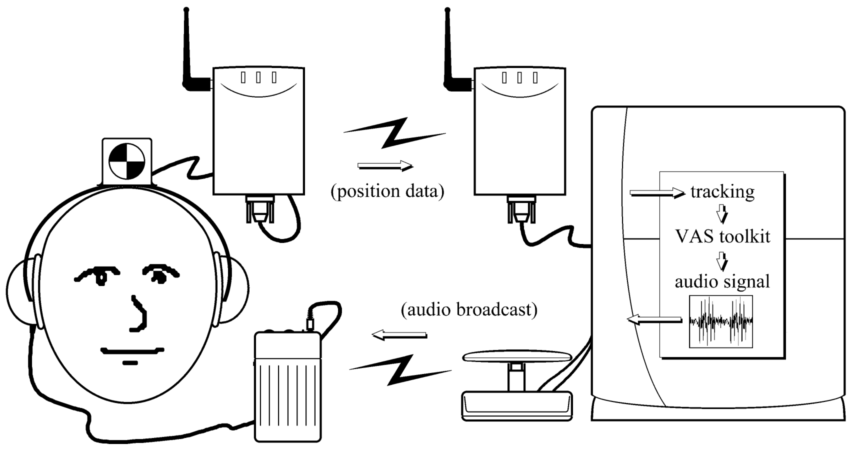

Figure 3 Spatialized audio in an augmented reality system

standard military helmet. A microgryo head tracker was attached to the helmet and head orientation was transmitted to the VAS server. Using this information, VAS spatialized the three streams in real time to fix them at the intended locations. The three sounds included a synthesized bell, a live speech stream, and a looping sampled sound. VAS was readily extended to support this new functionality by creating a new specialized version of a sound representation to support live input. This new sound representation read sound samples from an input port into a circular buffer. Sound sources could then read samples from this new representation and the processing beyond that point was not modified.

Another modification to VAS was to support Vector Base Auditory Panning (VBAP) [10], a three-dimensional spatialization technique using speaker panning. VBAP localizes a sound by choosing an appropriate triplet from the speaker configuration and panning the sound within the triangle formed by the speaker triplet. The clear advantage of VBAP is that is conveys both azimuth as well as elevation information, and it can support an arbitrarily large number of speakers. VBAP was incorporated into VAS be creating a new specialized localizer. The base localizer provides rendering algorithm with necessary information such as listener location and orientation, speaker locations, and sound source location. The availability of this information made possible the implementation of a VBAP localizer. Again, VAS was not modified to support this functionality beyond the creation of a new localizer.

Finally, in a recent project, VAS was used to add sound to a virtual walk-through of a military training site in a CAVE [3] environment. The VE was comprised of a small town center occupied by empty buildings and surrounded by open fields. Sounds were attached to helicopters and airplanes that would periodically fly over the town. With doppler effect, the sounds produced were very convincing. The problem was that during quiet periods, the town was completely silent which detracted from the experience. In order to counteract this, field crickets were to be added to the fields surrounding the town in order to recreate an environment more similar to what one might expect to encounter at such a site. Creating a realistic sound field with crickets' chirps was a problem however because there is generally a multitude of crickets active in a field and the number of sound sources required to accurately model this effect would be unmanageable. Instead, a new modeling construct was added to VAS that facilitated the creation of volumetric sounds. A new feature was added to Auditory Spaces that we refer to as a Randomized Sound Field. Sound sources are attached to an Auditory Space along with an offset o, period p and jitter value j. Every p seconds the Auditory Space object moves the sound source to a random location within the space that it delineates and activates the sound once. The Jitter value is a random number such that $-j \geq n \geq +j$. This value is added to the start time of the sound upon every invocation in order to produce a more natural effect. Finally the offset is used to offset the initial start time of the process so that two sounds with the same period could be kept out of phase. This new construct created a very convincing effect of field crickets with only two sound sources.

IMPLEMENTATION

VAS was implemented on SGI™ machines running the IRIX™ operating system. The system is comprised of two C++ shared object libraries that can be linked with applications. One version of the libraries is used for local execution where VAS executes within the application's process space on a single machine. The other version of the library was designed to give applications remote access to a VAS server running on a remote machine. The remote version of the library maintains the same interface as the local version so that applications can be made to run the local or the remote version without any modification. The remote versions of the VAS classes maintain state information and communicate with the server using RPC when their state changes.

Utilizing the most computationally expensive spatialization technique, namely HRTFs presented over speakers, a single sound source consumes approximately ten percent of an R10000 195 MHz processor. This performance metric was measured with a sampled sound and delay processing (Doppler) active. Because the VAS scheduler can utilize multiple processors when available, VAS can theoretically support a very large number of concurrent sounds. Unfortunately, current limitations in SGI's Audio Library limit that number to 16.

FUTURE WORK

The flexibility afforded by the design of VAS creates a number of interesting research possibilities that would otherwise be difficult to realize. One idea is that of a heterogeneous localization technique. Currently there are two dominant techniques for spatializing sounds: *Sound Field Simulation* and *Perceptual Synthesis*. Each of those techniques has its merits and drawbacks [5]. We would like to explore a new approach that would automatically choose the technique best suited for spatializing a sound based on the location and properties of the sound source and listener.

CONCLUSION

VAS provides researches with a well-defined framework in which new ideas can be integrated with minimal effort. Our experiences in adding new functionality to VAS without modifications to the underlying architecture demonstrate the utility of this architecture.

REFERENCES

1. Bargar, R., Choi, I., Sumit, D., Goudeseune, C. Model-based Interactive Sound for an Immersive Virtual Environment, *Proceedings of the International Computer Music Conference '94,* 471-474, 1994.

2. Burgess, D.A., Verlinden, J.C. An Architecture for Spatial Audio Servers, *GVU Technical Report*, GIT-GVU-93-34.

3. Cruz-Neira, C., Sandin, D.J., DeFanti, T.A. Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE, *Proceedings of SIGGRAPH '93 Computer Graphics Conference, ACM SIGGRAPH*, 135-142, August 1993.

4. Das, S., DeFanti, T., Sandin, D. An Organization for High-Level Interactive Control of Sound, *Proceedings of the Second International Conference on Auditory Displays, ICAD '94*, 203-216, November 1994.

5. Evans, M.J., Tew, A.I., Angus, J.A.S. Spatial Audio Teleconferencing – Which Way is Better?, *Proceedings of the Fourth International Conference on Auditory Displays, ICAD '97*, 29-37, November 1997.

6. Fouad, H., Narahari, B., Hahn, J.K. A Real-Time Parallel Scheduler for the Imprecise Computations Model, *PDCP, Special Issue on Parallel and Distributed Real-Time Systems 2*, 1 (1999), 15-23.

7. Fouad, H., Hahn, J., Ballas, J. Perceptually Based Scheduling Algorithms for Real-time Synthesis of Complex Sonic Environments. *Proceedings of the Fourth International Conference on Auditory Displays, ICAD '97*, 77-81, November 1997.

8. Hamman, M., Goudeseune, C. Mapping Data and Audio Using an Event-Driven Audio Server for Personal Computers, *Proceedings of the Fourth International Conference on Auditory Displays, ICAD '97*, 83-87, November 1997.

9. Huopaniemi, J., Savioja, L., Takala, T. DIVA Virtual Audio Reality System, *Proceedings of the Third International Conference on Auditory Displays, ICAD '96*, 111-116, November 1996.

10. Pulkki, V. Virtual Source Positioning Using Vector Base Amplitude Panning, *Journal of the Audio Engineering Society 45*, 6 (1997), 456-466.

11. Storms, R.L. NPSNET-3D Sound Server: An Effective User of the Auditory Channel, *Master's Thesis, Naval Postgraduate School*, Monterey California, September 1995.