COOLVR: IMPLEMENTING AUDIO IN A VIRTUAL ENVIRONMENTS TOOLKIT

Jarrell Pair and Rob Kooper
Virtual Environments Group
Graphics, Visualization and Usability Center
Georgia Institute of Technology
College of Computing
801 Atlantic Drive
Atlanta, Georgia 30332-0280
404-894-4488
jarrell@cc.gatech.edu, kooper@cc.gatech.edu

ABSTRACT

COOLVR (Complete Object Oriented Library for Virtual Reality) is a toolkit currently bei Visualization, and Usability Center (GVU) at Georgia Tech. The toolkit is written to all virtual environments (VE's) which will compile cross platform. Unlike most VE toolkits w visual senses, COOLVR aims to equally engage both the sense of sight and the sense of he goals of the COOLVR toolkit is to give the programmer an intuitive method to enrich the COOLVR uses a set of cross platform audio rendering modules to conduct real time sound p potential designers with the capability of easily integrating spatial audio in a virtual or presence can be achieved in COOLVR environments.

INTRODUCTION

Designers of virtual environments are seeking to create immersion for users. Primarily, this goal has been pursued by creating worlds with convincing graphics. This sense of presence is achieved by chiefly engaging the human sense of sight. Unlike sight, the sense of hearing is often neglected in the implementation of a virtual world. Recent work indicates that the integration of spatialized audio in a virtual environment enhances a user's sense of presence [6]. Regardless of considerable evidence on its immersive potential, audio is often banished as the poor stepchild of virtual reality. The plight of audio in interface design is explained in [2]:

> Audio alarms and signals have been with us since long before there were computers, but even though music and visual arts are sibling muses, a disparity exists between the exploitation of sound and graphics in interfaces. . . . For whatever reasons, the advancement of technology has been focused more on visual modes than aural.

This trend is in part due to technical source limitations of computer systems. Designers were forced to sacrifice audio quality for graphics performance. However, these restrictions no longer exist. In the pas (application specific integrated circuits) coupled with fast CPU's have made it possible to incorporate audio in graphically intensive virtual environments.

COOLVR (Complete Object Oriented Library for Virtual Reality) is a toolkit currently being developed at Georgia Tech's Graphics, Visualization and Usability Center (GVU). It is intended to succeed the Simple Virtual Environment (SVE) toolkit which the GVU Virtual Environments Group has used since 1992. SVE was designed p environment for the Silicon Graphics (SGI) platform. COOLVR is intended to provide a set of authoring tools that will allow a designer to quickly build a virtual environment that will run on both SGI and PC workstations. The goal is to create an array of functions that will empower users to create worlds that engage the sense of hearing equally the sense of sight.

IMPLEMENTATION DETAILS OF COOLVR

Until recently, complex virtual environments were developed almost exclusively for high performance SGI workstations. Only these machines could provide the performance required for VE applications. However, the Windows based PC has emerged as an increasingly powerful graphics platform. Increased performance and advantages of the PC

while maintaining the ability to run VE applications on SGI's, COOLVR was designed to compile cross-platform. Another key design decision was to create a dual set of modular rendering modules. One renderer handles the graphical components of the world. A separate sound renderer manages the environment's audio.

Details of the Audio Renderer

COOLVR (CVR) has been designed to support a set of modular objects (CVR objects) and renderers for both audio and graphics. The term "render" is used traditionally for graphics, but the concept can also be applied to audio [8]. COOLVR uses an audio renderer to accomplish real time sound processing in a virtual envi… audio renderer will function identically to the graphics renderer in that they both render objects. However, in this case the objects will encapsulate audio.

More formally, the audio renderer implements methods to render spatial (3D) attributes to instances of digital audio sample data. The sample data is read from files (currently only .WAV formats will be supported) and stored in audio VR objects. These objects can be positioned in the world as objects. For instance, an audio object can be associated with a graphics object to give the user the illusion that it is attached to the graphics object. If the graphics object moves in virtual space, the audio will move with it. This audio renderer also provides non-render of ambient soundscapes.

CVR_SetRenderMode() will select the audio rendering method. Currently the following mode
These modes are intended to give the designer a wide range of audio options to utilize c
of the environment, and the performance of the platform.

- CVR_NOTRICKS renders the audio sample data without spatial (3D) attributes. In other words, the sample data remains unmodified. If the sample data was preprocessed with spatial or other effects, these effects remain intact and static. This mode is intended for the playback of looped ambient au

- CVR_DISTANCE renders the audio sample data accounting for absolute distance. sound fro is attenuated or amplified depending on the distance.

- CVR_STEREO renders audio accounting for the left-right position from the listen This mode can be used when the platform lacks adequate resources to render spat audio.

- CVR_STEREODISTANCE renders audio accounting for the left-right position and absolute distance from the listener. Again, this mode is a viable option for environments running on platforms with performance limitations.

- CVR_SPATIAL renders audio accounting for X, Y, Z position of audio in respect t mode includes the implementation of distance rolloff employed in C and CVR_STEREODISTANCE.

Audio Distance Cutoff

In a virtual environment, graphic objects are rendered only if they can be seen by the user. This task is accomplished through the process of visible surface determination algorithms such as z-buffering [1]. For illustration purposes, this graphics rendering technique is analogous to the audio cutoff scheme employed in COOLVR. Each audio object is assigned a maximum cutoff distance and a minimum cutoff distance. If the audio object is spatial equal to the maximum cutoff distance, the sound file is not played. Likewise, the audio object is played only if it is positioned at a point less than or equal to the minimum cutoff distance. Because the fil are not played back, computational resources are conserved. This scheme differs from the distance attenuation algorithm in CVR_DISTANCE and CVR_STEREODISTANCE. These rendering modes guarantee audio file playback with the volume level varying as a function of the distance. Audio distance cutoff makes it possible for specified audio obj to be audible only relatively close to the user. For example, a whispering voice would be assigned a minimum and maximum cutoff distance of only a few feet. However, the sound of a waterfall would be assigned a much larger cutoff distance, thus allowing it to be played over long distances. If employed properly, this cutoff method can enhance the user's impression o

The initial outline of COOLVR audio distance cutoff functions are described below. These
the playback bounds of an audio object.

- CVR_AudioSetMaxDistance() sets the distance from the user beyond which an audio object's sample data will not be played back.

- CVR_AudioSetMinDistance() sets the shortest distance from the user at which an

CROSS PLATFORM CAPABILITY

COOLVR can be used to develop environments that can be run on either SGI's, PC's, and future platforms. The audio and graphics renderers will be written specifically for each platform. On the PC, the DirectX API, or OpenGL will be used to implement the graphics rendering modules. On the PC, spatial audio is rendered by making calls to Microsoft's DirectSound3D application programming interface (API). The use of this API facilitates the use of spatial audio acceleration hardware such as Diamond Multimedia Monster Sound. The SGI graphics rendering module is built upon the OpenGL graphics library. The audio module for SGI machines will be based on head related transfer functions (HRTF) convolution techniques discussed in [4] utilizing the KEMAR dummy HRTF data set [5].

Modular Approach

To be able to implement the different renderers on all platforms and still maintain a consistent interface, we decided to implement COOLVR as a set of modules. These modules will provide a flexible, upgradable degree of functionality. The modules can be replaced with updated versions that take advantage of new audio hardware or libraries that may become available. Similarly, several different variants of a rendering module can exist for the needs of a us of an application.

FUTURE DIRECTIONS

The version of COOLVR currently being developed was implemented with the audio functionality intended by the head mounted display (HMD) based virtual environments utilized at Georgia Tech's GVU. The audio rendering techniques implemented are intended for headphone (binaural) playback. Once the initial release of COOLVR is complete we plan to include functions to support speaker playback and simple voice recognition. These features will be needed to spaitial immersive displays (SID) such as the CAVE (Cave Automatic Virtual Environment) [3] and table projection systems similar to the responsive workbench [7] .

CONCLUSIONS

Virtual environments have been primarily oriented visually. With the introduction of new audio API's coupled with recent perceptual research, VE developers have begun to realize the indispensability of auditory cues in virtual environments. The COOLVR toolkit will enable users to intuitively implement audio in both PC and SGI based virtual environments. Resultingly, COOLVR environments can be authored to achieve a high sense of presence for

REFERENCES

1. Catmull, E., A Subdivision Algorithm for Computer Display of Curved Surfaces, Ph.D. UTEC-CSc-74-133, Computer Science Department, University of Utah, Salt Lake City, UT, December 19

2. Cohen, M., and Wenzel., E. M. The Design of Multidimensional Sound Interfaces. In W. Barfield and T. Furness III, editors, Virtual Environments and Advanced Interface Design, pages 291-346. Oxford Un New York, 1995.

3. Cruz-Neira C., Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE, Computer Graphics Proceedings, Annual Conference Series, 1993, 135-142.

4. Gardner, W.G., Transaural 3-D audio. MIT Media Lab Perceptual Computing Technical Re

5. Gardner, W.G., and Martin, K.D. HRTF Measurements of a KEMAR Dummy Head Microphone. MIT Media Lab Perceptual Computing Technical Report #280, May 1994.

6. Hendrix, C., and Barfield, W., The Sense of Presence Within Virtual Environments. Presence: Teleoperators and Virtual Environments 5, 3 (Summer 1996), 290-301.

7. Krüger, W., Bohn, C-A., Fröhlich, B., Schüth, H., Strauss, W., and Wesche, G. The Responsive Workbench: A Virtual Work Environment. Computer 28, 7 (July 1995), 42-47.

8. Takala, T., and Hahn, J. Sound Rendering. Computer Graphics (Proceedings of SIGGRAPH '92) 26, 2 (July 1992), 211-220.