3deSoundBox – a Scalable, Platform-Independent 3D Sound System for Virtual and Augmented Reality Applications

Philipp Stampfl

Imagination Computer Services, VR Development, Austria

Abstract

Sound is one of the most important components in animations, presentations and especially immersive environments. Many aspects are recognized more intuitively when sound supports vision and it is easier to follow a story and to believe in a virtual world of illusion. The 3deSoundBox was designed to provide this important acoustic component for such kind of 3D applications in an easy and intuitive way. It is scalable, platform-independent, has an easy to use interface and offers a lot of new possibilities in the field of virtual and augmented reality applications. The 3deSoundBox system can drive any number of speakers and works with any application on any platform with the same range of service. Once programmed, an application can easily take advantage of these features, as it will work on any sound-system with any number of speakers without requiring any changes, due to the scalable architecture of the system.

Categories and Subject Descriptors (according to ACM CCS): H.5.1 [Multimedia Information Systems]: Audio input/output H.5.5 [Sound and Music Computing]: Signal analysis, synthesis, and processing I.3.7 [Three-Dimensional Graphics and Realism]: Virtual reality I.3.6 [Methodology and Techniques]: Device independence
Keywords: sound API, platform-independent, sound system-independent, scalable, sound engine, 3D sound, spatial sound, cluster use, multichannel sound, dolby digital, multi-speaker, multi-user, distributed

1. Introduction

Multichannel sound is currently very common in theaters but there is still a lack of 3D sound in most VR and AR applications, which demand a much higher level of immersion. As we do not deal with flat images in such applications but produce visual immersive environments, appropriate sound located in 3D space has to be used to support that imagination. Currently available 3D sound suffers from problems such as dependencies on specific hardware setups, speaker configurations or specific APIs as well as on specific platforms. An application has to be adapted every time the platform or even the sound setup changes which costs a lot of money and human resources. At the moment there is no market leading sound system for VR and AR which can be used on SGI[2] systems as well as on Linux clusters or other common configurations with the same range of service.

There are several good high-level sound APIs such as *FMOD*[4] and *Miles*[3] available at the moment but some of them cost a lot of money and others do not provide a high range of services on different platforms (eg. *OpenAL*[7]). Due to the fact that hardware support differs from platform to platform, providing the same range of service is difficult. Even if this problem were solved, spatial sound calculation needs system performance and will affect the visualization and framerate. Also the number of output channels would still be limited to the number of outputs of the soundcard used (e.g. the VIA Envy[8] chipset has eight channels). One system solved most of these problems. The *Lake Huron*[9] provides spatial sound on a high-level but is very expensive and only supports between 32 and 64 sound sources at the same time.

In the last few months some sound engines were directly implemented into current VR and AR frameworks such as *VRJuggler*[10] and *OpenSG*[11]. *VRJuggler* uses *SONIX*, an *OpenAL* based sound engine, while *OpenSG* integrated *TRIPS*[12] into their scenegraph framework. *TRIPS* currently only provides basic sound functions like positioning, playing and stopping sounds. Its advantage lies in the direct integration into a scenegraph and, due to that, its automatically calculated properties at any change of the scenegraph. But this is also a disadvantage. As a fix part of *OpenSG*, this sound system can now only be used with it and not with other systems. Another disadvantage is that it uses *FMOD* as sound engine, which is free for non-commercial use but can be expensive if someone wants to use an application developed with *OpenSG* and *TRIPS* for commercial purposes. The *VR-Juggler - SONIX* system is not really platform-independent due to the use of *OpenAL*, whose range of service highly depends on the soundcard drivers.

This panel describes a system which is able to produce accurate surrounding sound without affecting the framerate, especially for virtual and augmented reality applications using a well defined interface.

2. Previous Work

About two years ago, an idea was the first step into the 3D sound field. The *Augmented Sound Reality - ASR*[1] should offer the user the possibility to drag and drop sound sources with his own hand and thus get a better feeling of how acoustic changes are related to its position in space. As humans cannot see the sounds, the sources where visualized using augmented reality. Figure 1 shows a person placing 3D sound sources using a pen.

The main aim of this project was to show the possibilities of current consumer hardware in the field of 3D sound and to show the options of using sound to augment VR and AR applications in order to increase their realism. The *ASR* was presented at last year's SIGGRAPH and the user-feedback was enormous. People from leading companies using virtual reality tried the application and where impressed how 3D sound could augment their applications.

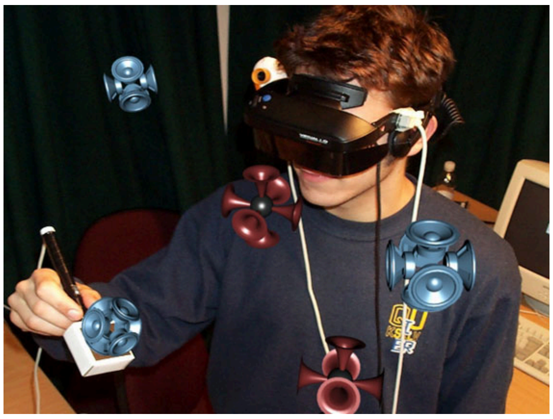

Figure 1: *Positioning 3D sound sources with the user's hand.*

The demonstration showed us that there is a big need for a 3D sound creating system but there is still a lot of work to do in order to create an easy and intuitive-to-use system. In addition to excellent acoustic quality, the usability of the system is also important. Sound can also help to give an idea of the final look and feel of a prototype. This is why a fast integration of sound into VR and AR applications is essential in order to provide the ability of rapid-prototyping.

3. Exposition

The so-called *3deSoundBox* provides an easy and intuitive way to integrate real 3D sound in VR and AR applications, in order to create full immersive sound environments. To have a clear interface between the sound component and the visual component, an external device (the *3deSoundBox*) was designed which can be accessed via a C++ API (currently available for Irix, Unix, Linux and Windows). Due to the explicit split between visualization and sound creation the performance of the graphics engine is less affected.

Moreover, more of these *SoundBoxes* can be combined (more information provides section 5), each driving up to a 7.1 Dolby Digital[5] speaker system (also possible are 2, 4, 5.1 and 6.1 speaker systems and even an accurate playback on headphones), in this way forming a configurable sound rendering environment. As the API is defined on an abstract level, the user is independent from the number of speakers (and *SoundBoxes*) used in an actual installation. An application can be developed with a small office sound system (e.g. a 5.1 speaker system), which can give the programmer a pretty good first idea of the final sound and afterwards the same application can be directly used with a big theater sound system (e.g. two or more 7.1 speaker systems), without changing a singe line of code. This works because the application is always accessing the so-called *MasterBox* which coordinates up to four *SlaveBoxes* each driving speakers on different positions in 3D space in order to create a realistic impression of the sound's position.

This feature makes it possible to use a completely programmed application with any number of speakers. This feature can be very important if someone considers using an application at many different locations, where the audio equipment may not be identical. In comparison with headphone solutions using *Head Related Transfer Functions (HRTF)* or common speaker setups that combine *HRTF* with *Cross Talk Cancelation*, common problems, such as front to back confusion, could be improved with this setup due to the free mobility of the user's head, in combination with real speakers located around him. Due to the free configuration of the speakers in the room, real speakers located at the bottom-back, middle-back or top-back of the listener can generate a realistic elevation from the sound sources. For virtual reality systems like the CAVE[13] speaker-rings are used as in these applications the point of interest can be anywhere around the user and not just in front as in theaters. Figure 2 shows a typical setup using two *SoundBoxes*. Section 5 describes more complex combinations of the *3deSoundBox* in VR and AR applications.

Another useful tool is the shared use option of the *Sound-*

Boxes. Defined as a *Shared node*, more than just one application can access the *SoundBox* system and generate a collaborative soundscape. This can be used if multiple computers are combined in a cluster to render a common virtual environment, where every node needs to create sound events. Another use of this feature is the creation of distributed sound which is described in more detail in section 6.

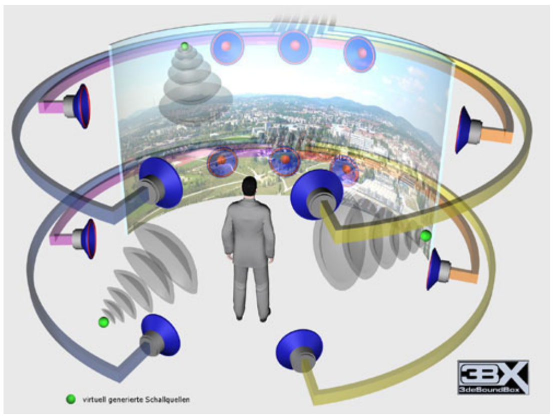

Figure 2: *A curved screen with two 3deSoundBoxes driving 14 speakers.*

To further improve the sound space, several psychoacoustic algorithms from *Sensaura*[6, 17] were implemented for example to represent big sound objects[16] or near field sound effects[15]. One of the most important parts of realistic room sound creation is reverb. Each room has its typical reverb. A hand clap in a church sounds different than a clap in a bathroom. Creating a highly complex soundscape without the option of adding realistic reverb would not create high degree of realism. *Creative's EAX Library*[20] is one of the best current realtime reverb engines. It supports the *IASIG - Interactive Audio Special Interests Group* Level 1[18] and Level 2[19] specifications and is currently available in its third version. The *3deSoundBox* has special reverb features implemented such as the blending between different reverb presets which can be used in a walk from one room into another (e.g. a walk from a hangar into an office).

In the following sections some features of the *3deSoundBox* are described in more detail.

4. The 3BX Sound-API

The *3BX API* is currently available for *Unix*, *Irix*, *Linux* and *Windows*. It can be used as a Library that offers a number of functions to create 3D sound and to manipulate it. There are only four lines of code needed to create a sound source, place it in 3D space and play the sound. With the API 3D sound as well as stereo sound sources can be created. Up to 64 3D sound sources and their characteristics can be calculated by the hardware. Up to 128 3D sound and up to 128 stereo sound sources can be additionally calculated in software and in realtime. Stereo sources have properties such as playback frequency, pan and volume. 3D sounds are far more complex. Each 3D source can have a position, a velocity which is used for the doppler effect calculation, as well as an orientation (for directed sound sources), a far and a near distance (used in the distance model which defines the volume change according to the distance) and further settings such as *ZoomFX*, *MacroFX* and common effects like *Flanger, Compressor, Echo* and many more.

```
SoundBox sndBox=SoundBox("Test Application");
sndBox.createNewBuffer("demo.wav","buffername",
   true);
sndBox.getBuffer("buffername").setPosition(
   1.0f,2.0f,3.0f,1);
sndBox.getBuffer("buffername").play(true);
```

The *3BX API* even offers the possibility to capture from an external input and use this captured file as sound source in the *3deSoundBox* system. In addition to the acoustic control functions, the API also offers *SoundBox* control functions, such as the connection status and automatic reconnect. Connections can be configured via config file as well as hardcoded as an automatic fall back if the config file could not be read. To enhance the performance of the sound system, the API has a proxy server implemented which automatically caches all settings and only submits changes to the *3deSoundBox*. An application which uses this API is totally independent from the number of *SoundBoxes* and speakers working together as it always accesses one *MasterBox* which coordinates and synchronizes the whole sound system. More details can be found in section 5.

As the API and the *SoundBox* are physically separated, an update of the *SoundBox* can easily be done without an influence on the driving application.

5. Scaleable

Each *3deSoundBox* can either be *Master* or *SlaveBox*. A *MasterBox* can have up to four other boxes configured which will then be controlled (accessed and synchronized) by it. As one *SoundBox* only has eight channels, this feature can be used to increase the number of total output channels. Due to this feature, one *SoundBox* can easily be afforded for development, while a combination of more can be used in the VR center where the application should finally run. Each *SoundBox* can drive between two and eight speakers. It is totally free to configure any combination such as one *SoundBox* driving eight speakers and one driving only two, which are mounted on the back of the user to prevent front-to-back confusion. With this feature almost any combination of *SoundBoxes* and speakers is possible. This combination can even be done a third time: a *MasterBox* drives four *MasterBoxes* which each drive another four *SlaveBoxes*.

The maximum number of *SoundBoxes* is currently 21 which means 168 output channels. This limit is due to the emerging latency between the command arriving at the first *MasterBox* and the last *SlaveBox*. Figure 3 shows a setup with nine *3deSoundBoxes* which can drive up to 72 speakers.

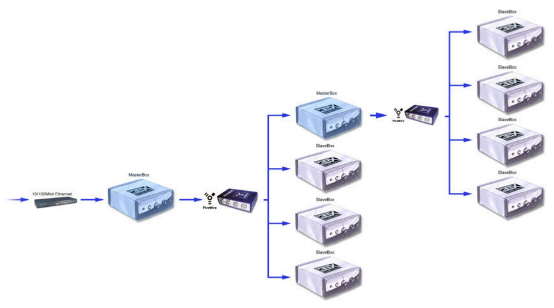

Figure 3: *Master and SlaveBoxes can be stacked in order to increase the number of output channels.*

6. Distributed 3D sound

Another feature of the *3deSoundBox* is its shareability. If configured as *SharedNode*, more than just one application can access the *SoundBox* system in order to share the resources and collaborately create a soundscape. If a *SoundBox* should not be shared, it can be configured as *SingleNode*. This should be done with any but the first *MasterBox* of a *SoundBox* system to ensure the integrity of the whole system (i.e. each *SounBox* can only be driven from its *MasterBox*). If the first *MasterBox* is configured as *Shared Node*, the whole system will share its resources according to the idea of one *MasterBox* coordinating the whole system, independent ofits configuration complexity.

This feature can easily be used to create a distributed soundscape. Each *SoundBox* needs to be configured as *SharedNode* and has to have the other *SoundBoxes* configured as *SlaveBoxes*. Due to this configuration, each generated sound will automatically be created at the other *SoundBoxes* and the other way around. This also means that any application can share its soundscape by simply changing the configuration of the driven *MasterBox* and without any adjustment of the application! Figure 4 shows a typical setup for a shared (distributed) sound environment.

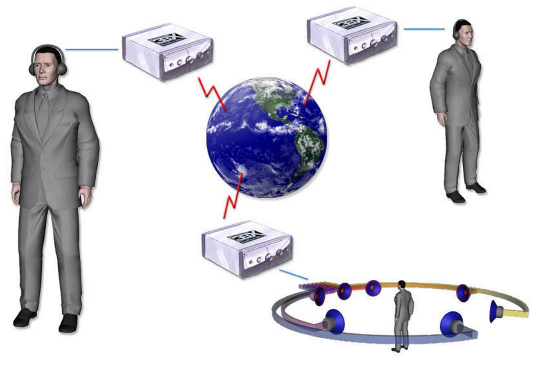

Figure 4: *Sound can automatically be generated at different locations. Each environment shares its sound with others.*

7. Extensible Toolkit

Trials have shown that there is a need for special settings which appear in various VR and AR setups. One problem is that the origin of the application is often displaced from the origin (or sweetspot[21]) of the sound system (e.g. if the screen is the visualization origin which is mounted three meters in front of the sound systems sweetspot). A sound played at the position (0.0,0.0,0.0) would then appear almost direct in the listener's head instead of being placed three meters in front of him. This is a mistake that many common sound implementations make. Also, the placement of different *SoundBoxes* has to be configurable. Another typical problem is that different visualization systems use different coordinate systems. The *3deSoundBox* offers the possibility to directly recalculate the given coordinates from the driving application into the left-handed coordinate system used in the *SoundBox*. There is also the possibility to scale the coordinates as the *SoundBox* uses meters and a driving application might use another linear dimension. These features ensure easy and fast integration into any application as no additional work has to be done to forward the application values to the *SoundBox*.

The main settings are described in more detail in the following paragraphs. Figure 5 shows the coherence between them.

- SoundBox origin: defines the position of the sound system driven by this *SoundBox* in relation to a listener at the room's sweetspot.
- Application origin: defines a translation from the *SoundBox* origin to the application origin.
- Scaler: defines the scale from the application's measure units to meters which are used in the *SoundBox*.

8. Conclusions and future work

Our experiments with 3D sound in combination with immersive virtual enviroments have shown that the impression of space was much more intensive than without sound or simple stereo sound. Users managed to identify objects in the environment much more easily than without sound[14]. Also interaction with objects could be improved when an acoustic feedback was used to signal a set action (e.g. pushing a button or a wheel squeaking if turned).

Moreover, 3D sound offers the possibility to direct the

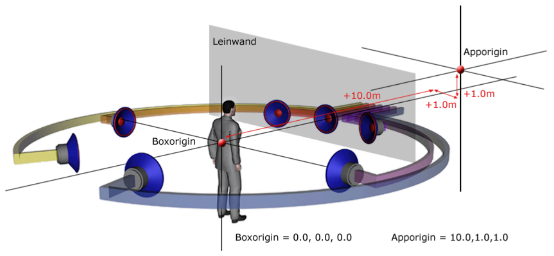

Figure 5: *With the application- and the SoundBox-origin, the sound system can be configured for any application demands.*

viewpoint of the user to certain spots in 3D space using intensive sound sources at that spot. This extended the possibilities of digital story telling with an important additional *directing* feature.

The trials have also shown that adding accurate 3D sound was easy and fast using the *3deSoundBox*. Realistic acoustic sound environments could be achieved using its standard features. The power of the *SoundBox* could imposingly be demonstrated when the sound system setup was changed and more *SoundBoxes* were added, with the effect of an increasing immersive soundscape and no performance change at the visualization.

In the future, the stacking option of the *SoundBox* will be enhanced in order to combine more than 21 *SoundBoxes* synchronously. There will also be a Plugin for *Maya*[22] to easily place sound sources and set their options in 3D space as a preset as most 3D models for VR and AR applications are first modeled in a 3D software tool. The future will be full of new developments which have to find their practical use in the *3deSoundBox*. Future versions of the *3BX API* will be backwards compatible but offer these new features in order to keep generating accurate sound environments for the virtual and augmented reality application of tomorrow.

References

1. Dobler D., Stampfl P., Haller M. "ASR - Augmented Sound Reality", *Sketches and Applications at SIGGRAPH 2002*, p. 148 (July 2002). 2
2. Silicon Graphics Inc. http://www.sgi.com/ 2003. 1
3. RAD Game Tools Inc., "Miles Sound System", http://www.radgametools.com/miles.htm 2003. 1
4. Firelight Technologies Pty., "FMOD Sound System", http://www.fmod.org/ 2003. 1
5. Dolby Laboratories Inc., "Dolby-Digital", http://www.dolby.com/ 2003. 2
6. Sensaura Ltd., "Sensaura acoustic algorithms", http://www.sensaura.com/ 2003. 3
7. Loki Entertainment Software, "Open Audio Library", http://www.openal.org/home/ 2003. 1
8. VIA Technologies, "VIA Envy Audiochipset", http://www.via.com.tw/en/Digital%20Library/PR030428envy24pt7-1.jsp 2003. 1
9. Lake Technology Limited, "Huron20 - Digital Audio Convolution Workstation", http://www.lake.com.au/driver.asp?page=main/products+and+customers/audio+vr+(huron)/huron20 2003. 1
10. Cruz-Neira C., "VRJuggler", http://www.vrjuggler.org/ 2003. 1
11. Reiners D., VoSS G., Behr J., "OpenSG", http://www.opensg.org/ 2003. 1
12. Neumann T., Fünfzig C., Fellner D., "TRIPS - A Scaleable Spatial Sound Library for OpenSG", http://www.eg.org/EG/DL/PE/OPENSG03/07neumann.pdf 2003. 1
13. Cruz-Neira C., Sandin D., DeFanti T., "Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE", *Proceedings of SIGGRAPH 1993* http://www.evl.uic.edu/EVL/RESEARCH/PAPERS/CRUZ/sig93.paper.html 1993. 2
14. Sibbald A., Sensaura Ltd."An introduction to sound and hearing", 1999. 4
15. Sibbald A., Sensaura Ltd."MacroFX 2.0", 2001. 3
16. Sibbald A., Sensaura Ltd."ZoomFX for 3D-Sound", 2000. 3
17. Dawley R., Sensaura Ltd."Sensaura SDK 1.0", 2000. 3
18. 3D Working Group of the Interactive Audio Special Interrests Group, "3D Audio Rendering and Evaluation Guidelines, Level 1.0", 1998. 3
19. 3D Working Group of the Interactive Audio Special Interrests Group, "3D Audio Rendering and Evaluation Guidelines, Level 2.0", 1999. 3
20. Jot J-M., Boom M., Creative Technology Limited, "Environmental Audio Extention, EAX 2.0", 2001. 3
21. Tomlinson H., Focal Press, "5.1 Surround Sound, Up and Running", 2000. 4
22. AliasWavefront, "Maya 5", http://www.aliaswavefront.com/en/products/maya/index.shtml 2003. 5