SAFIR: LOW-COST SPATIAL SOUND FOR INSTRUMENTED ENVIRONMENTS

Michael Schmitz[1] and Andreas Butz[2]

[1]DFKI GmbH, Saarbrücken, Germany, schmitz@cs.uni-sb.de
[2]University of Munich, Munich, Germany, butz@ifi.lmu.de

ABSTRACT

We present a high-level Java API that allows developers of pervasive computing environments to integrate 2D or 3D spatial sound over loudspeakers into systems without any DSP knowledge. The system is platform-independent and allows most arbitrary speaker configurations, thereby providing a scalable tool for real-time spatialization with low-cost off-the-shelf hardware.

Keywords: spatial audio, toolkit, instrumented environments.

INTRODUCTION

Real-time 3D sound rendering has proven to be valuable across a wide range of domains and applications, including virtual reality, gaming and assistive interfaces for the visually impaired (e.g. Mereu and Kazman (1)). We believe that use of spatial audio can also be beneficial in pervasive computing environments by helping to create richer experiences and interactions for users. In this work, we present SAFIR (Spatial Audio Framework for Instrumented Rooms), an API that allows for the easy, low-cost, flexible integration of spatial sound into ubicomp environments.

SAFIR offers several benefits for use in pervasive computing environments over existing spatial sound libraries and APIs. Unlike many special-purpose spatial sound systems that use expensive hardware and proprietary interfaces (e.g. Bargar et al. (2)), SAFIR works with low-cost, off-the shelf hardware, and does not require developers to have DSP knowledge to install and configure spatial audio setups. Additionally, SAFIR is designed specifically to support the use of spatial audio in pervasive computing environments. Conventional spatial sound libraries that are built to support common surround standards and gaming applications under Windows operating systems assume that a user is stationary and oriented toward a single visual display (e.g. EAX[1])). Such libraries also typically assume the use of a conventional 5.1 speaker setup, and therefore do not support flexible speaker configurations. SAFIR's sound output allows users to move and interact freely within an environment. As a cross-platform system SAFIR is not limited to Windows operating systems and furthermore allows for flexible configuration and re-configuration of speakers in the physical space to suit the purposes of the environment.

SYSTEM DESIGN AND FEATURES

SAFIR was developed to meet the following design requirements:

- **Cost-efficiency**: The system should work with affordable, commonly available hardware.
- **User-Centred sound**: The sound rendering should adapt to a moving listener position, correctly preserving the spatial image of the acoustic scenery as much as possible.
- **Configuration flexibility:** The system should support an arbitrary number and flexible spatial configuration of speakers in order to be able to deploy the same system in different setups for different purposes.
- **Ease-of-development**: Ubicomp developers without digital sound processing experience should be able to use the system effectively.
- **Cross-platform compatibility**: The system has to be compatible with both Windows and Linux operating systems.

SAFIR works with almost any number of speakers. The minimum for any reasonable spatialization is 4 speakers, but the maximum is only set by the limits of PC sound hardware. More speakers increase the quality of spatialization. The volume spanned by the speakers defines the area in which sound can be freely spatialized. The developer defines the actual speaker positions in a configuration file when the system is set up. Generally, speakers should be distributed evenly in the listening area, but a higher spatial resolution in areas of particular interest can be achieved by positioning more speakers there. Developers use SAFIR by providing sound sources (audio files or streams) and their designated coordinates in the environment; other parameters such as volume or routes instead of simple coordinates are optional. If the user position can be determined by the environment, the coordinates can

[1] Creative Technology Ltd.: Environmental Audio Extensions: EAX 4.0 - Available at http://developer.creative.com/

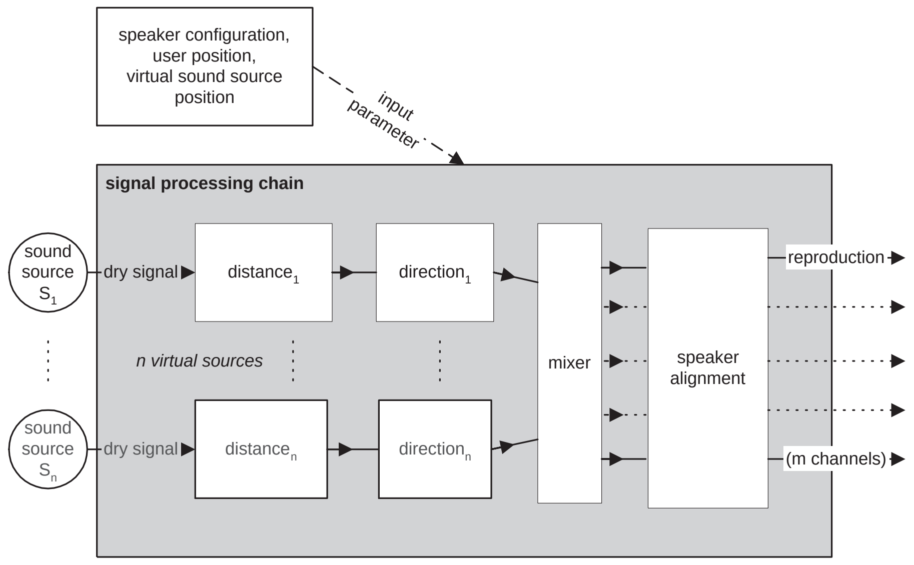

Fig. 1. The Structure of SAFIR

be passed in to the spatial sound engine, which will adjust the synthesis of currently playing sounds to it, which especially improves the spatial perception significantly when the user moves around a virtual sound source.

IMPLEMENTATION

We implemented SAFIR in Java to provide cross-platform compatibility using the JSyn library[2] for real-time sound rendering. The VBAP (Vector Based Amplitude Panning) algorithm by Pulkki (3) is used for the general spatialization, because it provides computationally cheap functions to simulate the direction of a virtual sound source. Since VBAP only simulates the direction of a sound, we added additional mechanisms to create a feeling of distance and moving sound sources as well.

JSyn classes are based on the traditional unit generator model, therefore we also structured SAFIR in a similar manner, which can be seen in figure 1. It describes the signal flow from the sound source signal on the left to the final speaker signals on the right, with each square in between representing one step of the signal processing chain. Other input parameters relevant for the processing steps are shown in the box on top, namely the positions of loudspeakers, user and the virtual sound source. SAFIR first generates distance, then directional cues and finally adjusts the speaker channels. This procedure will be described more detailed in the remainder of this chapter.

Distance

Human perception of distance does not work as accurate as of direction of sound sources. The main distance cue is the decreasing intensity of a sound source over growing distance due to air absorption. The $1/r^2$ law, also used by SAFIR, expresses this behaviour sufficiently. This means for example that the intensity will be multiplied by $1/2^2 = 1/4$ if the sound source moves twice as far away. Another (indoor) distance cue is the time and energy relation between direct and reflected signal. Sound signals do not only travel directly from a source to a receiver but also reflect from walls and indirectly reach the listener marginally later and with less energy due to air absorption. This will not be perceived as a separate sound event but interpreted by the brain to estimate the distance of the source. To keep the computational costs low, SAFIR only considers the first reflection and assumes that it only occurs from the same direction as the original source, which would be true in a spherical room. Using such a sphere with a given radius as a simplified room model, it is possible to compute the additional travelling distance of the reflection for all positions. SAFIR now sets the radius to the distance of the furthest loudspeaker (which will probably be in a corner of the room) and mixes a delayed and attenuated copy of the source signal to itself.

The Doppler effect is created by another delay that is controlled by a linear function of distance to the virtual sound and the listener and updated whenever the sound source position changes: This will create a frequency shift, which will temporarily increase the pitch of the sound source when it moves towards the user and decrease it when it moves away.

[2] JSyn - Java Audio Synthesis - Available at http://www.softsynth.com/jsyn/

Direction

SAFIR uses the VBAP algorithm to simulate the directions of a virtual sound source. It is a multi-channel spatialization technique that allows positioning sound sources on a surface that is spanned by a set of loudspeakers. VBAP chooses speakers that will emit the signal and computes the intensity of these speakers (local intensity panning). In case that all loudspeakers lie in the same horizontal plane, e.g. on a ring around the listener, the algorithm runs in a simplified 2D mode. The number of used loudspeakers and their positions is generally arbitrary, but a higher density of speakers will improve the accuracy of the spatialization.

First step of the VBAP algorithm is to determine the active loudspeakers. Depending on the relative position of the virtual sound source to the listener, up to three speakers with the shortest angular distance to the sound source become active while all others stay mute. In 2D mode, only up to two speakers become
active and elevation will not be considered. In special cases when e.g. the position of the virtual source matches with a speaker, only that speaker will be the active one. Now the relation of the gain factors of the speakers will be determined on vector basis as shown in the figure 2:

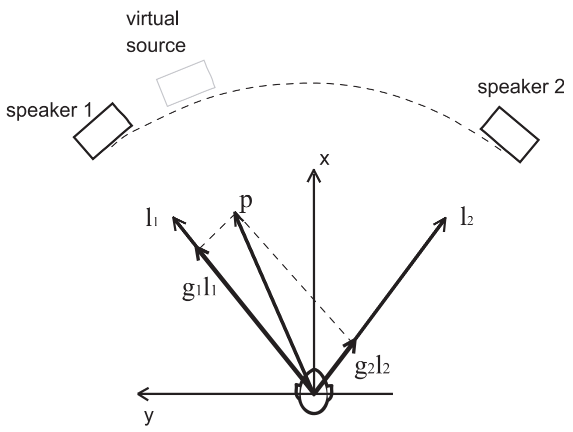

Fig. 2. VBAP formulation in 2D mode

The unit vectors l_1 and l_2 point towards the active speakers. Unit vector p, pointing towards the sound source, can be formulated as linear combination of speaker vectors l_1 and l_2:

$$p = g_1l_1 + g_2l_2$$

g_1 and g_2 are the resulting intensity factors for the active speakers and can be computed by simple equations. This algorithm works accordingly for 3D mode.

Output of SAFIR's direction module are between one and three copies of the signal with certain intensities, each of which will be routed to one speaker.

Speaker Channel Adjustments

After the sound sources have passed the distance module, they were routed by the direction block to their active speaker channels as described above. Now after mixing all sounds that are playing in parallel, SAFIR adjusts the speaker channels incorporating the different distances of the speakers to the user. Same principles as in the distance simulation of sound sources are applied to attenuate and delay speakers that are closer than others. It is especially important, since VBAP assumes that distances of all speakers to the listener are equal. This gives us more flexibility in positioning the speakers in the room and creates a larger 'sweet spot'.

System Usage

The system requires the coordinates of the loudspeakers as input parameters - currently supplied by a plain text file. Basically two Java classes are needed to manage the audio playback: The *AudioContext* provides methods to specify the user's position at runtime and other parameters that effect the overall audio reproduction e.g. the general volume. Instantiating this class will start the audio server and allow generating instances of *AudioObject*s. Each *AudioObject* represents a virtual sound sources and has (among others) methods to interactively position them in space. This separation between context and objects easily allows to supply several rooms with one server if sufficient soundcards are available. Sound sources can either be files, streams over the network or live recordings from microphone/line-in channels.

APPLICATIONS

We deployed SAFIR in the Saarland University Pervasive Instrumented Environment (S.U.P.I.E. as described in Butz and Krüger (4) at a hardware cost of approximately $1000 (for 8 speakers, soundcard, amplifier and cables). It is used to support interactions in a variety of scenarios: Spatial sound is coupled to the projection of a steerable projector for combined audio-visual presentations, such as an embodied conversational agent which appears as a virtual inhabitant of the environment and migrates between different displays and devices, such as stationary projectors and wall-mounted displays. It appears visually as a cartoon character on arbitrary surfaces in the environment. Acoustically, it talks to the user with speech output

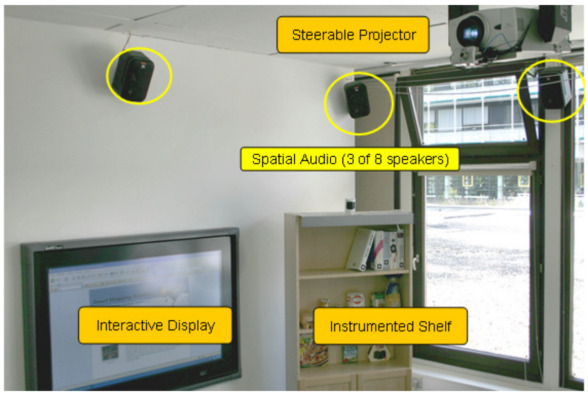

Fig. 3. The SAFIR system integrated into our instrumented environment.

rendered from the direction in which the character is currently visible (see figure 4).

Another idea that we are currently investigating is talking objects, as in Wasinger and Wahlster (5) and Schmitz et al (6): Users communicate directly with smart objects, inspired by Reeves and Nass's observation that people attribute human qualities to objects (Reeves and Nass (7)). Users engage in dialogues with physical objects which - in most cases - cannot produce audio on their own. These objects are instrumented with sensors to detect their location. Our system supports the talking object metaphor by providing audio channels that can be spatially related to objects in our instrumented environment. We are also exploring how acoustically spatialized instructions can improve performance in navigation tasks.

CONCLUSIONS & FUTURE WORK

We presented a high-level Java API for developers of instrumented environments to integrate spatial sound over loudspeakers without any DSP knowledge and demonstrated how this platform-independent tool is being utilized in our lab.

As additions to the current low-cost base setup, we also plan to integrate specialized devices into the system. One such example is the use of private audio channels for users in the same room through wireless headsets or a steerable AudioBeam[3], a directional speaker that can send a very narrow sound beam only audible by individuals in its emanation path. Control of such devices will then be incorporated into the system to provide extensions for multiple users and privacy settings. In the near term, we plan to invite other researchers and developers to use SAFIR to gather more insights and feedback about the value of the interface and of spatial sound in ubicomp environments in general.

[3] Sennheiser Electronic GmbH & Co.KG, Press Archive: www.sennheiser.com/sennheiser/icm.nsf/root/press_archiv_2_2001_aesconvention110_1

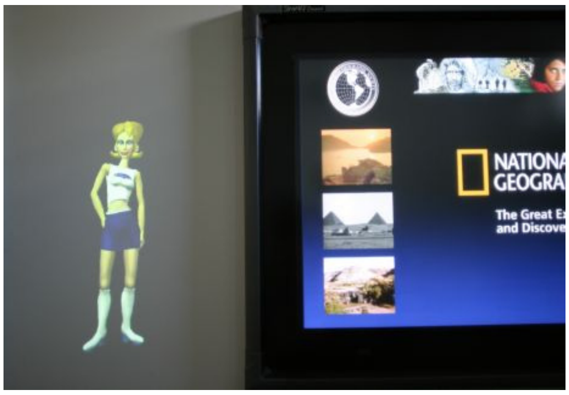

Fig. 4. The virtual room inhabitant Kruppa et al (8)

REFERENCES

1. S. W. Mereu and R. Kazman, "Audio Enhanced 3D Interfaces for Visually Impaired Users", 1996, Proc. of CHI'96, 72-78

2. R. Bargar, I. Choi, S. Das and C. Goudeseune, "Model-based interactive sound for an immersive virtual environment", 1994, Proc. of ICMC, 471-474

3. V. Pulkki, "Virtual sound source positioning using vector base amplitude panning", 1997, J. of the AES, 45, 456-466.

4. A. Butz and A. Krüger, "A mixed reality room following the generalized peephole metaphor", 2006, IEEE Computer Graphics & Applications, 24-31

5. R. Wasinger and W. Wahlster, 2006, "The Anthropomorphized Product Shelf: Symmetric Multimodal Interaction with Instrumented Environments", Chapter in: True Visions: The Emergence of Ambient Intelligence, Springer, Heidelberg, Germany

6. M. Schmitz, J. Baus and S. Schmidt, "Anthropomorphized Objects: A Novel Interaction Metaphor for Instrumented Spaces", 2006, Pervasive 2006 Workshop Proc., 487-493

7. B. Reeves and C. Nass, 1996, "The media equation: How people treat computers, television, and new media like real people and places", CSLI Publications and Cambridge university press.

8. M. Kruppa, M. Spassova and M. Schmitz, "The Virtual Room Inhabitant: Intuitive Interaction With Intelligent Environments", 2005, Proc. of the 18th Australian Joint Conference on Artificial Intelligence (AI05), 225-234