Conducting Psychoacoustic Experiments with the SoundScape Renderer

Matthias Geier, Sascha Spors

Address: Quality & Usability Lab, Deutsche Telekom Laboratories, TU-Berlin, Ernst-Reuter-Platz 7, 10587 Berlin
E-Mail: {`matthias.geier,sascha.spors,SoundScapeRenderer`}`@telekom.de`
Web: `http://qu.tu-berlin.de`

Abstract

The *SoundScape Renderer* (SSR) is a versatile software tool for real-time spatial audio reproduction. It is capable of generating driving signals for a variety of loudspeaker-based and headphone-based reproduction methods. Due to its extensive remote-control possibilities, it can be conveniently used as an audio backend in psychoacoustic listening and conversational experiments, for example in the area of spatial telephone conferencing.

The main features of the software are described and some aspects about its architecture are presented. Different possibilities are shown how the SSR has already been used for conducting subjective tests.

The *SoundScape Renderer* can be downloaded from `http://tu-berlin.de/?id=ssr` as *Free and Open Source Software*.

1 Motivation

In comparison to traditional speech and audio listening tests, subjective experiments in the context of spatial audio communication put higher demands on the test setup. In addition to temporal and timbral attributes, also spatial attributes have to be presented, controlled and analyzed.

For loudspeaker-based tests, the driving signal for each loudspeaker has to be calculated in real-time if the sound sources' positions or other parameters shall be changed interactively during the experiment. One problem of using loudspeakers for listening tests is the exact placement of the test subject withing the listening area and the controlled transition between listening positions. To achieve reproducible test conditions, often headphone-based reproduction is chosen. The spatial impression is most natural when head-tracking is used to compensate the test subject's head movements. This presentation method – which is commonly used for tests in the area of spatial telephone conferencing – is called *dynamic binaural (re-)synthesis*. Because of the necessity to compensate head movements, it is not possible to use pre-computed audio files for playback and the output signals have to be generated in real-time. Additionally, in some adaptive test methods, the signals change dynamically depending on the test subject's choices.

As a means of interaction between the test subject and the test setup often specialized *Graphical User Interfaces* (GUIs) are needed to realize certain test methodologies. For implementation and maintenance it is normally much easier to use separate programs for the presentation of the audio stimuli and for displaying the test GUI. Both programs have to share a common interface to exchange control data, for example via network sockets.

In the following sections a framework for spatial audio experiments is presented which is able to cope with all aforementioned requirements.

2 Technical Details

The *SoundScape Renderer* (SSR) [1] is a tool for *object-based* audio reproduction, i.e. input signals are represented as sound objects with a given position in space and several other parameters. The information for all sound objects is stored in a so-called *scene description*. Based on this scene description the output channel signals of the desired reproduction system are generated in real-time.

The SSR is written in C++ and compiled with the GNU compiler. It is running on Linux and with some effort it can also be installed on Mac OS X.

2.1 Reproduction Methods

The SSR is not limited to a single reproduction method or hardware setup. Some of the available rendering modules are:

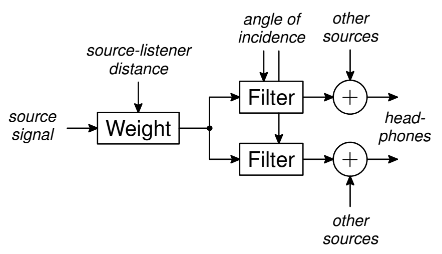
- loudspeaker-based:
  - *Wave Field Synthesis* (WFS) [2]
  - *Vector Base Amplitude Panning* (VBAP) [3]
  - *Ambisonic Amplitude Panning* (AAP) [4]
- headphone-based:
  - *HRTF-based Binaural Renderer*
  - *Binaural Room Synthesis* (BRS)
  - *Binaural Playback Renderer* (BPB)

When using the HRTF-based renderer, all sources share the same set of *Head Related Impulse Responses* (HRIRs) which are typically measured in a dry, non-reverberant environment. The BRS renderer uses *Binaural Room Impulse Responses* (BRIRs) which contain room information specific to the position where they were measured. Therefore, each source needs its own set of BRIRs and the source positions cannot be changed in the SSR.

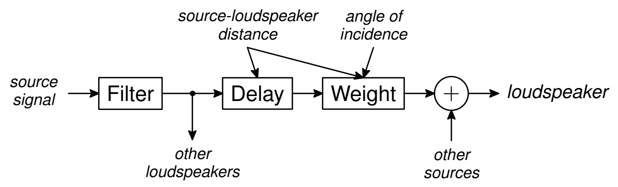
The BPB renderer is not calculating the headphone signals in real-time but is rather an audio file player with head-tracking capabilities. Binaural signals for each head rotation in steps of one degree have to be generated beforehand, the BPB renderer merely selects the appropriate channels depending on the head orientation and does a crossfade if the orientation changes. This is useful for investigating algorithms for which no real-time implementation exists, for example for rapidly moving sources. The BPB renderer is still in development and not yet included in the public release of the SSR.

2.2 Signal Processing

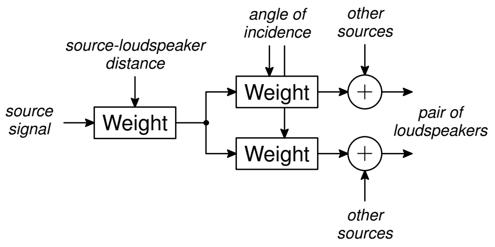
All renderers are realized as a combination of a few basic processing units: filtering (partitioned convolution), delay line and weighting.



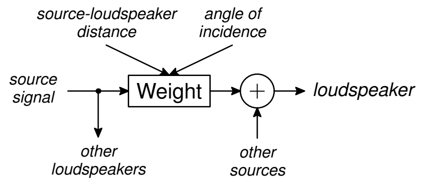
(a) Binaural Rendering



(b) Wave Field Synthesis



(c) Vector Base Amplitude Panning



(d) Ambisonic Amplitude Panning

Figure 1: Signal flow in different rendering modules

The BRS and HRTF renderers – figure 1(a) – are implemented by convolving the input signal with impulse responses corresponding to the left and right ear, dynamically chosen from a database of impulse responses depending on the source angle with respect to the listener position and the head orientation of the listener. Before that, the source signal is weighted with a value depending on its position to simulate distance attenuation.

The WFS renderer – figure 1(b) – is implemented by applying the same pre-filtering step to all input channels and then – individually for each combination of source and loudspeaker channel – delaying and weighting the resulting signals depending on the relative position and orientation of source and loudspeaker.

When using the VBAP renderer – figure 1(c) – only two loudspeakers are active for each sound source. A panning function is used to calculate the weights of the two loudspeakers. Another weighting factor is applied to both loudspeakers to account for the distance attenuation.

The AAP renderer – figure 1(d) – uses the principles of *Higher Order Ambisonics* (HOA) to calculate a panning function which potentially employs all loudspeakers to reproduce one source.

All renderers use the same renderer base class with a straightforward interface which makes it simple to implement new renderers. It also provides parallel processing capabilities, which can be utilized by all renderers.

2.3 Audio Input/Output

The *JACK Audio Connection Kit* (JACK) is utilized for input and output of audio data. This allows for a great flexibility in routing between different audio processing applications and the hardware ports. Any audio application with JACK support as well as the physical inputs of the sound card(s) can be used to provide the input signals. The SSR has been used with loudspeaker setups with up to 192 channels.

2.4 Control

The SSR can be controlled in different ways. On the one hand, it has a built-in *Graphical User Interface* (GUI) by means of which sound sources can be moved around and their volume and other parameters can be changed interactively. In experimental setups, however, the built-in GUI is mostly in the background or entirely deactivated.

On the other hand, there is a network interface to which a control application can connect via a TCP/IP socket. This socket interface has already been used by several GUIs written in *Python*, two GUIs written in *ActionScript*, two GUIs written in *Matlab*, a remote-control created with *Pure Data*, a GUI running on a mobile phone with *Android OS* and an *iPhone* application. Over its network interface, the SSR sends and receives text messages in a simple XML format. This way the network messages can be extended in a very flexible way and the network traffic can be analyzed easily in case of problems.

3 Previous Experiments and Setups

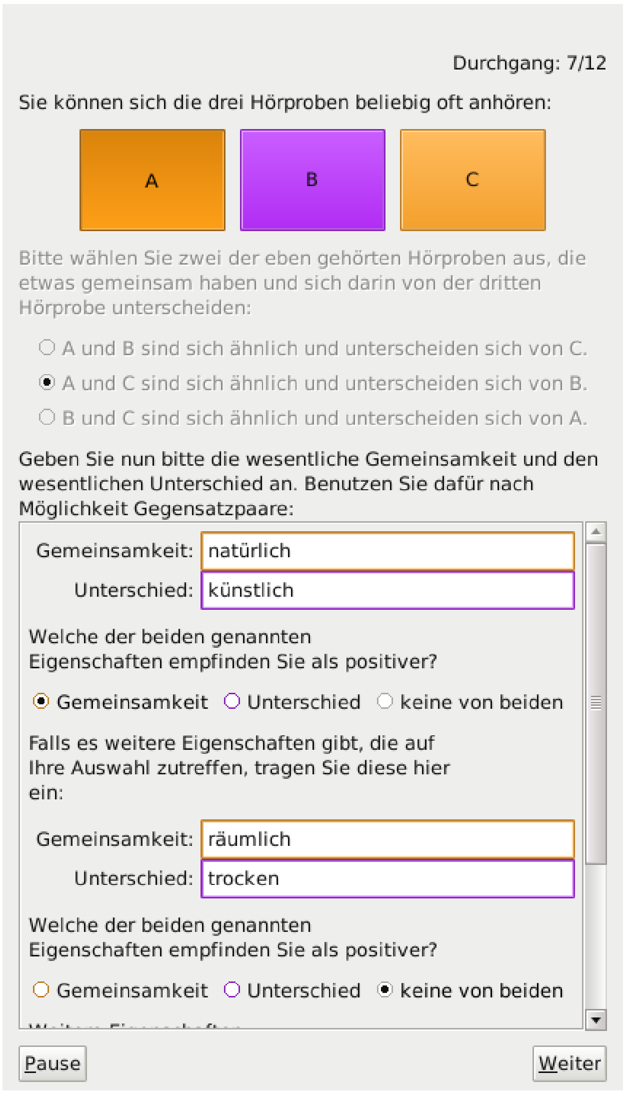
The modular architecture of the SSR allows its use in many different setups. It has already been used in about a dozen psychoacoustic experiments. Most of them were using headphone-based reproduction, but also experiments with loudspeaker arrays have been conducted.

Binaural synthesis has been realized using *Head Related Impulse Responses* (HRIRs) measured in non-reverberant rooms as well as *Binaural Room Impulse Responses* (BRIRs) measured in more or less reverberant rooms. The BRS renderer was also used with BRIRs calculated in *Matlab* to render virtual loudspeaker setups.

Audio playback was either done by the SSR or by an audio player created using *Pure Data* (a.k.a. Pd). All components were connected by means of the JACK audio server. In most experiments GUIs written in *Python* were used to control the SSR and Pd, but also GUIs written in *Matlab* were used.

In the following paragraphs a few experiments using the SSR are briefly presented and the test setup is explained.

A spatial telephone conference listening test has been conducted in [6] where recordings of three participants were played back using Pd and the BRS renderer of the SSR was used to generate test conditions as combination of different spatial presentation modes and different telephone bandwidths. For each condition a BRIR set was calculated using *Matlab*. A GUI written in *Python* was used to select the current condition and start the playback.



Durchgang: 7/12

Sie können sich die drei Hörproben beliebig oft anhören:

Bitte wählen Sie zwei der eben gehörten Hörproben aus, die etwas gemeinsam haben und sich darin von der dritten Hörprobe unterscheiden:

- A und B sind sich ähnlich und unterscheiden sich von C.
- [✓ A und C sind sich ähnlich und unterscheiden sich von B.]
- B und C sind sich ähnlich und unterscheiden sich von A.

Geben Sie nun bitte die wesentliche Gemeinsamkeit und den wesentlichen Unterschied an. Benutzen Sie dafür nach Möglichkeit Gegensatzpaare:

Gemeinsamkeit: natürlich
Unterschied: künstlich

Welche der beiden genannten Eigenschaften empfinden Sie als positiver?

[✓ Gemeinsamkeit] Unterschied keine von beiden

Falls es weitere Eigenschaften gibt, die auf Ihre Auswahl zutreffen, tragen Sie diese hier ein:

Gemeinsamkeit: räumlich
Unterschied: trocken

Welche der beiden genannten Eigenschaften empfinden Sie als positiver?

Gemeinsamkeit Unterschied [✓ keine von beiden]

Figure 2: Screenshot of the test GUI from [5], created with *Python*, *GTK+* and *glade*.

An experiment about binaural monitoring of WFS systems has been done in [7]. The same audio scene was presented using three different methods: A real WFS system, a "virtual" WFS system simulated using the BRS renderer with BRIR measurements of the real loudspeakers and a binaural rendering using dry HRTFs. All three conditions were realized using the SSR and the switching was done manually by the experimenter.

In [5], the perception of focused sources in WFS has been evaluated. A "virtual" WFS system has been realized using the BRS renderer. For each stimulus a BRIR set has been generated in *Matlab*. Figure 2 shows the GUI which was presented to the test subjects.

A very similar setup was used in an experiment about the role of the precedence effect for the perception of artifacts in WFS [8]. Again, *Matlab* was used to generate BRIR sets to create a "virtual" WFS system with the BRS renderer and head-tracked headphones.

The perceptual relevance of using fractional delays and of errors regarding loudspeaker placement in WFS was investigated in [9]. Except different BRIR sets, a different test method and a different GUI, the test setup was very similar to the previously mentioned experiment.

In addition to the aforementioned research papers, the SSR was also used in several diploma theses. In this context a perceptual comparison of WFS and HOA was done – again using a "virtual" loudspeaker setup – using the BRS renderer. Both the GUI and the audio file playback was done in *Matlab*.

Another diploma thesis was proposing different panning techniques for rapidly moving sources in WFS. A listening test was done using the *Binaural Playback* (BPB) renderer. Audio playback was done by the SSR and a GUI written in *Python* was used to mute and un-mute stimuli according to the test subject's selections.

4 Conclusions and Future Work

The *SoundScape Renderer* (SSR) is a versatile tool for spatial audio reproduction which is freely available as *Open Source Software*. It can be used in psychoacoustic experiments in the context of spatial audio reproduction and transmission. It can be connected to a variety of control applications using its network socket interface.

Future plans include the implementation of *OpenDAFF* [10] support for directional impulse responses (`http://www.opendaff.org`) and the definition of a common scene format for control and interchange of spatial audio scenes between different reproduction systems.

Visit `http://tu-berlin.de/?id=ssr` to get your own copy of the SSR and feel free to contact `SoundScapeRenderer@telekom.de` if you have questions or suggestions. Any comments are greatly appreciated!

References

[1] Matthias Geier, Jens Ahrens, and Sascha Spors. The SoundScape Renderer: A unified spatial audio reproduction framework for arbitrary rendering methods. In *124th Convention of the Audio Engineering Society*, May 2008.

[2] Sascha Spors, Rudolf Rabenstein, and Jens Ahrens. The theory of Wave Field Synthesis revisited. In *124th Convention of the Audio Engineering Society*, May 2008.

[3] Ville Pulkki. Virtual sound source positioning using Vector Base Amplitude Panning. *Journal of the AES*, 45(6):456–466, June 1997.

[4] Martin Neukom. Ambisonic panning. In *123th Convention of the Audio Engineering Society*, October 2007.

[5] Matthias Geier, Hagen Wierstorf, Jens Ahrens, Ina Wechsung, Alexander Raake, and Sascha Spors. Perceptual evaluation of focused sources in Wave Field Synthesis. In *128th Convention of the Audio Engineering Society*, May 2010.

[6] Alexander Raake, Claudia Schlegel, Matthias Geier, and Jens Ahrens. Listening and conversational quality of audio conferencing. In *40th International Conference of the Audio Engineering Society*, October 2010.

[7] Matthias Geier, Jens Ahrens, and Sascha Spors. Binaural monitoring of massive multichannel sound reproduction systems using model-based rendering. In *NAG/DAGA International Conference on Acoustics*, March 2009.

[8] Hagen Wierstorf, Sascha Spors, and Alexander Raake. Die Rolle des Präzedenzeffektes bei der Wahrnehmung von räumlichen Aliasingartefakten bei der Wellenfeldsynthese. In *German Annual Conference on Acoustics (DAGA)*, March 2010.

[9] Jens Ahrens, Matthias Geier, and Sascha Spors. Perceptual assessment of delay accuracy and loudspeaker misplacement in Wave Field Synthesis. In *128th Convention of the Audio Engineering Society*, May 2010.

[10] Frank Wefers. OpenDAFF: a free, open-source software package for directional audio data. In *German Annual Conference on Acoustics (DAGA)*, March 2010.