The Multi-Touch SoundScape Renderer

Katharina Bredies, Nick Alexander Mann, Jens Ahrens,
Matthias Geier, Sascha Spors and Michael Nischt
Deutsche Telekom Laboratories / Technische Universität Berlin
Ernst-Reuter-Platz 7, 18. Floor
10587 Berlin, Germany
katharina.bredies@telekom.de

ABSTRACT
In this paper, we introduce a direct manipulation tabletop multi-touch user interface for spatial audio scenes. Although spatial audio rendering existed for several decades now, mass market applications have not been developed and the user interfaces still address a small group of expert users. We implemented an easy-to-use direct manipulation interface for multiple users, taking full advantage of the object-based audio rendering mode. Two versions of the user interface have been developed to explore variations in information architecture and will be evaluated in user tests.

Categories and Subject Descriptors
H.5.2 [**User Interfaces**]: Graphical User Interfaces; H.5.5 [**Sound and Music Computing**]: Systems

1. INTRODUCTION
More and more spatial sound reproduction systems are used in practice. They differ in the particular reproduction technique used and the speaker layout. Traditionally, audio is produced and transmitted on a per reproduction channel basis and hence for a particular speaker layout. In order to handle the variety of sound reproduction systems object-based audio is a promising approach. Here the audio objects together with side information are transmitted and the local terminal renders the sound for the local speaker layout using a suitable spatialization algorithms.
This object based approach to spatial audio opens up the freedom of local user interaction with the spatial audio scene. It also offers new possibilities for audio editing tools and interaction techniques, as sounds can be handled as discrete objects. However, currently spatial audio tools are mostly used by expert users and consequently interface design did not play a major role so far. These tools do also not exploit all potentials of the object-based audio approach. However, as soon as the technology will reach a mass market in telecommunications or entertainment, there will be a need for more intuitive interfaces.
The development of the Soundscape Renderer user interface was an opportunity to explore direct manipulation and collaborative interaction for object-based spatial sound scenes. We built a Frustrated Total Internal Reflection (FTIR) based multi-touch table similar to the one presented in [8]. We then adapted an existing mouse-based previous iteration of the interface and implemented two versions: One that focused on gestures and multi-touch input, heavily relying on the idea of direct manipulation. The other version emphasizes the collaborative aspect and provided individual menus for each user, assembling the available information and functions in one place.
This paper is organized as follows: We describe the hardware setup, the development of the graphical user interface for the spatial sound renderer software and the two versions of the interface that we designed for user testing.

2. RELATED WORK
A lot of interactive tabletop devices for musical expression have been developed as music controllers in the past years, both as multi-touch and tangible devices. Popular examples are the reactable[10] and the audiopad[14], that use physical pucks as interactive objects. Examples for multi-touch interface are the LEMUR sensor pad [9] and the synthesis interface for the original FTIR table presented in [4]. They provide an overview of audio-related visualizations, gestures and interaction techniques that range between physical and virtual space. On the other hand, spatial mappings have been exploited in domains where they are inherent, like landscape or interior architectural planning scenarios[2, 16]. This approach is also very obvious for controlling spatial sound sources. However, only few research has been dedicated to multi-touch tabletop devices for spatial sound scenes.
The ISS Cube [15] uses a surround sound system to create spatial audio scenes. The sources are represented as acrylic pucks that can be put into the scene space and freely moved around. In the Audiocube [1] system, the sounds are represented as cubes. Each side of the cube plays a different sample once it is put down on the table. Both systems are developed for exhibitions and offer only reduced functionality like spatial positioning, volume and sample changes. Also, the scenes are spatially restricted and do not support animation of sound sources.
The IOSONO system [7], on the other hand, uses a Wacom pen tablet and virtual representations to work on spatial sound scenes. Developed at Fraunhofer Institute for Digital Media Technology (IDMT), it focuses on expert users and

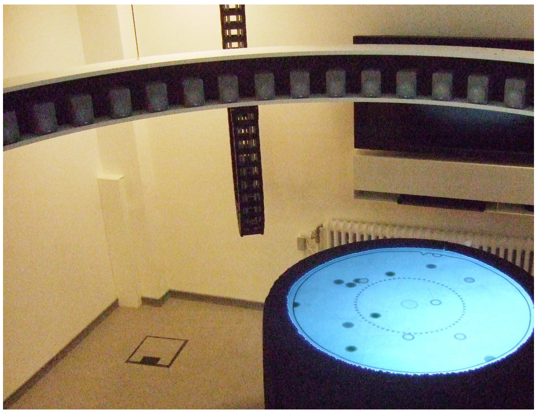

Figure 1: The wave field synthesis lab room with the circular speaker array and the multi-touch table in the center.

refers to established audio software interfaces. Furthermore, there seems to be no multi-touch functionality implemented on the tablet, although the mapping is similar to a tabletop device.

We found that tangible tabletop interfaces did not offer the required flexibility, but provided a very comprehensive mapping. Real-time audio applications are a good source for existing and well-working gesture and multi-touch input possibilities. From other applications with spatial mappings, we could also learn about the particularities and problems with spatial layouts and refer to an established gesture repertoire.

3. HARDWARE SETUP

We used the Frustrated Total Internal Reflection technique introduced to multi-touch sensing in [8] to built a custom-made, round-shape tabletop multi-touch device. The FTIR technology seemed appropriate because of its relatively low costs and nonetheless sufficient performance for multi-touch purposes. We decided upon a round shape to avoid that the table would have a particular orientation, to fit it into the circular wave field synthesis speaker system, and to support an undetermined number of users (see Fig.1). For the spatial mapping, the horizontal orientation of the display seemed very appropriate, and it would also reduce fatigue during use.
In contrast to many other audio tabletop applications, we decided against using tangible parts like markers. Without tangible components, we preserved more liberty and reduced the danger of accidentally changing the setup. We also had no restrictions regarding the amount and placing or intersection of the sources. Besides, we could include animation in our interface.
The deployed software dealing with concurrent touch inputs was developed in house as an open source library build upon the $Java^{TM}$ Platform. Besides a small native part for the camera driver grabbing the input image sequence, the software is organized into two independent modules. The first one is responsible for determining and describing the touch inputs. This is done by labeling connected components in a gray-scale image, whereby each of them corresponds to single touched area. The other module focuses more on interaction. It contains a set of composable manipulators to be linked with user interface. For example it allows to move, scale and rotate a single widget with an unlimited amount of touches contributing. However, in this project it was not truly used, because of the $Flex^{TM}$ based user interface, which receives touch states accordingly to the TUIO Protocol [11]. Although we knew that it is not perfectly suitable for a multitouch table as its focus lies on rigid physical object markers, we welcome the simplicity and it proved good enough for our purposes.

4. SOUNDSCAPE RENDERER

The Soundscape Renderer (SSR) is a software framework for real-time rendering of spatial audio using a variety of algorithms [6]. At its current development stage it is able to render virtual acoustic scenes using either Wave Field Synthesis (WFS), binaural rendering or Vector Based Amplitude Panning (VBAP). It provides, amongst other key features, a graphical user interface and a network interface to interact in real-time with the auditory scene. Multiple clients can connect to a central SSR and modify the scene and system parameters. This way any type of interface or tracking system can be connected easily and control the SSR. The multi-touch GUI was implemented as a Flash Shockwave application that includes all the necessary functionalities to connect to a SSR and interact in real-time with the auditory scene.

5. SSR GUI DEVELOPMENT

For the binaural rendering mode of the SSR, we had designed a graphic user interface before for demonstration purposes. So we already had a set of requirements for the user interface and informal experience concerning its usability.
As an initial step, we conducted a short informal video inquiry, asking people to perform some crucial functions that we wanted to implement while recording their movements. It turned out that they stick closely to the interaction style with mouse and keyboard they were already familiar with. But we also recognized some patterns on how a couple of basic actions were performed (like rotating and scaling). Those were, on the other hand, geared to the interaction with physical objects, or simply known from other gesture-based devices. However, the test subject reported dissatisfaction about their own creative output and ascribed it to not being prepared and not having the time to reconsider their expressions. We therefore took the video inquiry as an inspiration rather than a strict requirement.
We then decided upon correspondent multi-touch input for the SSR interface. One particular problem was to find ways to display information in the scene and on the sources. As GUIs for spatial audio are not widespread, we had few conventions to stick to. So we borrowed some interactions from existing audio editors as well as from tangible audio interfaces like home stereos. Besides, we referred to the various examples of musical tabletop devices[1, 10, 9, 14] and multi-touch literature [5, 13, 12, 16, 3, **?**, **?**].
We also did paper prototyping with the first designs to find inconsistencies and logical gaps in an early state. Here, we checked the size of the graphic representations especially important for touch input and gestures. According to the paper prototyping results, we implemented the graphic user interface for the table.

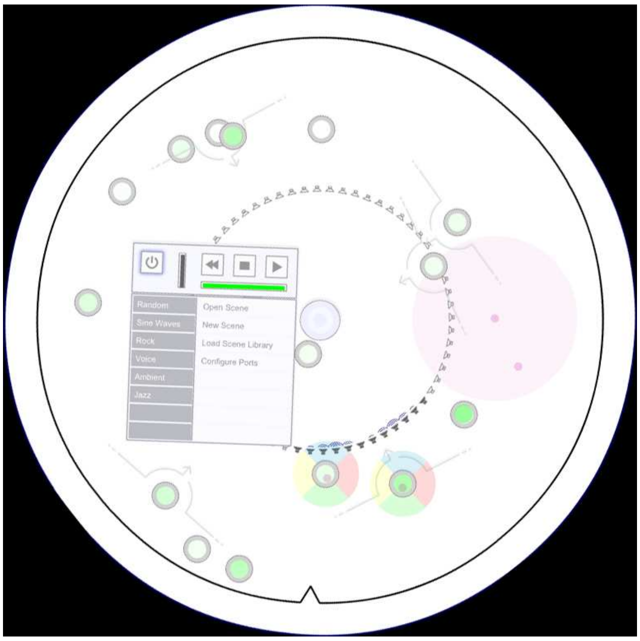

Figure 2: Version A of the SSR multi-touch GUI. The center application menu, the scene translation halo and the source context shortcuts are visible.

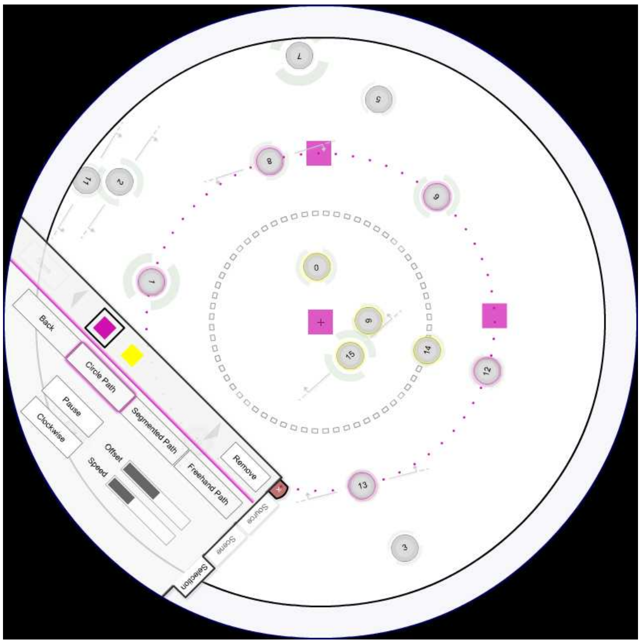

Figure 3: Version B with the edge menu and some sources animated to follow a circle path.

6. RESULTS

The preliminary result of our development are two versions of the multi-touch SSR user interface. They apply two different extremes of interface logic that we came upon during the information architecture design and sketching phase. The first one (that we will refer to as version A), inherits the organization of the predecessor mouse-based interface for the SSR and enhances it with gesture input. The second one (called version B) focuses on making the interface appropriate for multiple users. Both version provide sound sources as single objects, a surrounding space that we named the "scene space", and a menu with application-related functions.
In the following, both versions will be described in detail.

6.1 Version A

Version A organizes the features directly in place. All functionality and information belonging to a particular source is attached to it. Each sound source has its own four-part shortcut menu that is arranged around the core. It consists of three toggles (to mute and solo and to display the information panel) and a gesture slider for the volume. The user can simply tap the toggle shortcuts to perform the allocated action. To change the volume, he has to put one finger down in the shortcut area and do a scaling gesture (see Fig.2). The volume level is then indicated as a green arc around the source, increasing and decreasing with the volume. The intensity of the sound each source emits is represented as a green circle in the source center, that gets brighter and darker. An arrow at the rim of the table indicates the reading direction and justification for the shortcut menu. Touching the rim readjusts all source menus.
If the user touches the scene space, a pink halo appears around his touch. He can scale and rotate the whole scene by pressing another finger down in this area and moving it relative to the first one. Moving the first finger will move the whole scene. The translations from several users are added up against each other.
Pressing the center of the table shows a menu with features for the whole application. The user can jump to another preset audio scene, adjust the volume, pause and play the audio rendering.
In this version, the most important functions are accessible through gestures. We tried to make the interface as "direct" as possible, so the features would be easy to find. Anyway, one clear drawback of gesture input is that it is not made visible in the interface; there is no "affordance" what gestures could be performed. So we stuck to a very small set of gestures that we assumed would be the most common, relying on our short video inquiry.

6.2 Version B

In version B, interaction with the sound sources in the scene space was reduced to positioning, selecting, deselecting and grouping. Tapping on several sources one after another would store them in a preliminary group selection that was always highlighted in blue. Using the rim menu, the groups could then be stored permanently. Tapping into the void would deselect all sources. The user can also drag lassos around a bunch of sources. We kept two gesture shortcuts: Drawing a lasso around one pressed-down finger would select all sources; drawing a lasso around two pressed fingers selects all stored groups. The emitting sound of a source is visualized by pulsing transparent arcs around it that get bigger the louder the source sound is. The reading direction for the type always orients to the edges, so the label of a source changes direction when the source is moved over the table.
The whole functionality - source information and type, scene features, application features and group features - was relo-

cated in a rim panel. Several panels could open up in front of each user (see Fig.3). The panel contains three tabs: The first one is showing the source menu, the second one the preset scene selection and scene features, and the third one grouping and animation features.
The source menu displays all information and functions allocated to the source. The user can pick the source in numerical or alphabetical order from a top menu bar, then change the volume and source type on the panel. The selected source is highlighted with a pulsating blue halo in the scene space. However, it is not possible to allocate a source selected in the scene space to a particular panel.
The scene menu tab shows a number of preset scenes as well as the scene master volume, scale and source file.
In the third tab, the user can store his selection in permanent groups and apply some animation behavior, like following a circle path or moving between a number of points. Again, particular groups can only be selected in the menu.
In this version, the connection between the individual panel and a single source is much weaker than in version A, but it offers more detailed interaction, like assembling, storing, changing and animating groups.

7. CONCLUSIONS

In this paper, we described the development process of a multi-touch user interface for a spatial sound reproduction software. We produced two versions of the GUI, working with different visualizations and with different focus. We believe that the two extremes represent a general problematic of interfaces with spatial layouts and that the results can be applied to different domains of simulation and planning. The ongoing work will therefore aim at further refining the interface, especially assimilating the functionality of both versions while purifying their different approaches on the one hand. On the other, we are interested in exploring the usability of both approaches with regard to the information architecture.
Although one can assume the strengths and weaknesses of each version, they have not been evaluated in user testings so far. We will therefore conduct formal user tests to evaluate the actual advantages and drawbacks of each version. Besides, we use the table as a testbed not only to reveal usability requirements for audio software tools, but to observe emergent user behavior for new application areas.

8. REFERENCES

[1] Audite. Audiocube. `http://www.audite.at/en/projects_audiocube.html`.

[2] E. Ben-Joseph, H. Ishii, J. Underkoffler, B. Piper, and L. Yeung. Urban simulation and the luminous planning table bridging the gap between the digital and the tangible.

[3] H. Benko, A. D. Wilson, and P. Baudisch. Precise selection techniques for multi-touch screens. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1263–1272, New York, NY, USA, 2006. ACM.

[4] P. L. Davidson and J. Y. Han. Synthesis and control on large scale multi-touch sensing displays. In *NIME '06: Proceedings of the 2006 conference on New interfaces for musical expression*, pages 216–219, Paris, France, France, 2006. IRCAM — Centre Pompidou.

[5] C. Forlines, D. Wigdor, C. Shen, and R. Balakrishnan. Direct-touch vs. mouse input for tabletop displays. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 647–656, New York, NY, USA, 2007. ACM.

[6] M. Geier, J. Ahrens, A. Möhl, S. Spors, J. Loh, and K. Bredies. The soundscape renderer: A versatile framework for spatial audio reproduction. In *Proceedings of the DEGA WFS Symposium*, Ilmenau, Germany, Sept. 2007. Deutsche Gesellschaft für Akustik (DEGA).

[7] I. GmbH. Iosono. `http://www.iosono-sound.com/`.

[8] J. Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 115–118, New York, NY, USA, 2005. ACM.

[9] Jazzmutant. Lemur. `www.jazzmutant.com`, 2004.

[10] S. Jordà, G. Geiger, M. Alonso, and M. Kaltenbrunner. The reactable: exploring the synergy between live music performance and tabletop tangible interfaces. In *TEI '07: Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 139–146, New York, NY, USA, 2007. ACM.

[11] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E. Costanza. Tuio - a protocol for table based tangible user interfaces. In *Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation (GW 2005)*, Vannes, France, 2005.

[12] M. Masoodian, S. McKoy, and B. Rogers. Hands-on sharing: collaborative document manipulation on a tabletop display using bare hands. In *CHINZ '07: Proceedings of the 7th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction*, pages 25–31, New York, NY, USA, 2007. ACM.

[13] T. Moscovich and J. F. Hughes. Multi-finger cursor techniques. In *GI '06: Proceedings of Graphics Interface 2006*, pages 1–7, Toronto, Ont., Canada, Canada, 2006. Canadian Information Processing Society.

[14] J. Patten, B. Recht, and H. Ishii. Interaction techniques for musical performance with tabletop tangible interfaces. In *ACE '06: Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*, page 27, New York, NY, USA, 2006. ACM.

[15] Something. ISS cube. `http://www.somethingonline.org/index.php?main=done&sub=iss_cube`.

[16] M. Wu and R. Balakrishnan. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In *UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 193–202, New York, NY, USA, 2003. ACM.