THE AMBLR: A MOBILE SPATIAL AUDIO MUSIC BROWSER

Rebecca Stewart and Mark Sandler

Centre for Digital Music,
School of Electronic Engineering and Computer Science,
Queen Mary, University of London,
London, UK

ABSTRACT

Music collections are often visualized in a two-dimensional space to show relationships between songs. Some user interfaces interacting with these two-dimensional maps of songs use spatial auditory display to allow easier access to the audio content. A common auditory display is to have multiple songs playing simultaneously from differing spatial locations around the user. However, when using this style of interface the sonification of the collection needs to be limited to a local subset of the collection, usually three to six songs. This paper presents the amblr, a spatial audio music browser that allows a user to auralize the collection. It combines effective design from previous work with new approaches to create a novel interface. This allows for a more intuitive navigation of a virtual space populated by a large collection songs without relying on textual metadata.

Index Terms— music information retrieval, auditory display, binaural

1. INTRODUCTION

As mobile devices for listening to music become more pervasive and the music collections that those devices can access become larger, dedicated interfaces for music collection browsing are needed. This paper describes an interface designed specifically for a mobile device. Interfaces designed for desktop computers or immersive displays cannot be directly translated to mobile devices. The foremost change is the smaller screen size, but it would be remiss to assume that mobile devices cannot provide an immersive experience. The interface design cannot rely on a large visual display, but mobile devices are enriched with other sensors such as accelerometers, physical input from touch screen and/or buttons, haptic feedback through vibrations, and offer a more controlled audio playback environment through headphones.

When surveying the literature for music browsing interfaces, it is easily seen that maps are popular approach. They begin with the *Sonic Browser* [1] and continue through *nepTune* [2]. Within these interfaces, a music collection is arranged in a two or three-dimensional space and the user sonifies that arrangement by listening to a portion of the map. This usually means that multiple songs are played at the same time, but placed at different positions around the user.

Common issues emerge when using this approach such as overwhelming the listener with too much information. We try to address those issues while also using the features that worked well while adapting this approach to mobile devices. This paper proposes a new spatial audio interface for exploring a collection of music: the *amblr*. The *amblr* builds on established approaches from previous work and pulls together different approaches while incorporating additional novel design features.

2. INTERFACE DESIGN

Map interfaces usually rely on a visualization of the two or three-dimensional space populated by the music collection, but this is not conducive to small screens. Another mode of interaction is required. Auditory displays convey information via audio and are particularly advantageous for mobile devices with small screens that cannot present large or complex visual displays. The *amblr* has a visual display, but it is minimal so that is not necessary to look at the mobile device in order to use the interface – the auditory display is the primary mode of interaction. Though the interface is fully functional without any visuals, they do aid interaction by illustrating the different parameters of the interface. This has been shown to be necessary as the interface in [3] did not use a visual display and users had difficulties understanding the spatial audio environment.

The *amblr* draws strongly upon previous interfaces that use spatial arrangements of audio files. The Sonic Browser [1] and the cyclical interface described in [4] are the two primary influences. First, a music collection is arranged over two-dimensions – how this is accomplished is independent of the interface. The user then begins exploring the collection by listening from the center of the map. They can move towards or away from the songs that they hear and manipulate the parameters that will be discussed below to better understand the virtual space. When they find a song of interest, they can se-

978-1-61284-350-6/11/$26.00

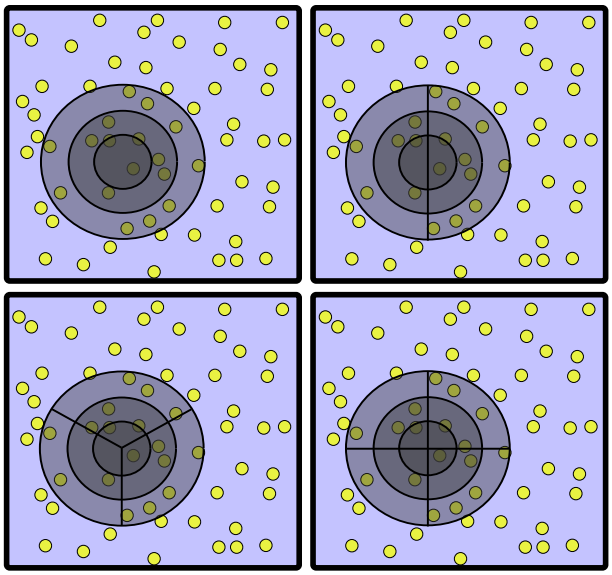

Fig. 1. Each square is an illustration of the same map with different interface parameters. The listening position is identical in each map and at the center of the concentric circles. The zoom level increases as the diameter of the circles in each map decrease so that less of the total map and only the nearest songs are heard. The number of concurrent sources allowed increases in each map.

lect that song to be used in another process such as playlist generation or purchasing from an online music store.

The *amblr* does not consider how the map was computed; it treats the map generation as a black box. The interface is designed to interact with an arbitrary two-dimensional arrangement of songs.

2.1. Navigating the Map

The primary user interaction with a map of music is moving around the two-dimensional space towards and away from various songs. This is done in a similar manner to first person viewpoints in video games. The user occupies a point on the map and surrounding songs are spatialized according to their location and proximity to the user's position. This is illustrated in Fig. 1 where the same map is shown with four different sets of user interface parameters. For all four maps the listening position is identical and is at the center of the concentric circles.

2.2. Focus

To further exaggerate the audio environment, sound sources to the front of the user are louder than sources the same distance away but to the sides or rear of the user. This is to emphasize the information from the front and disambiguate from sources behind the user. Attenuating the sources not directly in front of the user has been implemented as a user-adjustable *focus of perception* in *soniXplorer* [5]. The idea of mimicking and enhancing focus is also explored in the *AudioStreamer* [6] and *Audio Hallway* [7] where the volume of sound sources increases by 10 dB when the user turns towards a source to indicate interest.

The amount of *focus* is adjustable by the user in *soniXplorer*, but is a fixed value in the *amblr* to reduce the number of parameters. The *focus* adjusts the gain of each sound source according to its angle relative to the user. The gain of a song is inversely proportional to its distance:

$$g_{distance}(d) = min(1, \frac{r}{d} - \frac{r}{d_{min}}) \quad (1)$$

where the gain $g_{distance}$ of a song is dependent on its distance d, the rate of decay per unit r, and the minimum distance d_{min} from the user where the sound source begins to decay. Songs directly in front of the user play at a volume only attenuated by the distance, but sources to the sides and rear have additional attenuation. A song located at φ radians relative to the direction the user is facing has a gain of:

$$g_{focus}(\varphi) = e^{-\varphi^2/4} \quad (2)$$

The gain for a song s located at position $\vec{p_s}$ with the user at position $\vec{p}$ is then the product of the attenuations from the distance and the angle of the song relative to the user:

$$g(s, \vec{p}) = g_{distance}(||\vec{p} - \vec{p_s}||) * g_{focus}(\angle(\vec{v}d, \vec{p_s} - \vec{p})) \quad (3)$$

where $\vec{v}d$ is the user's orientation.

2.3. Zoom

A primary concern expressed by users of music interfaces during user evaluations is that listening to more than one sound source can be confusing and that they can be quickly overwhelmed with too much information. The *Sonic Browser* coined the term *aura* as a means to restrict the number of sound sources, defining an aura as "a function that indicates the user's range of interest in a domain" [8]. An aura is commonly visually represented as a circular cursor on a map. It limits the number of audio files that play back concurrently and highlights a region of interest. This idea has been repeated as an "auditory torch" in *City Maps* [9] and in *SoundTorch* [10].

Auras or functionally similar tools are frequently requested if missing from an interface with multiple concurrent streams [1, 11]. Users testing *nepTune* [2] asked for "larger landscapes to allow focused listening to certain tracks in crowded regions." The *soniXplorer* [5] limits the number of audible sound sources by not allowing multiple songs to play from the same location. If multiple songs are located near each other, only the closest one will be heard.

The *amblr* refers to the functionality of an aura as zooming as it is similar to zooming in and out of an online geographical map interface. Zooming is illustrated by the concentric circles with different diameters in Fig. 1. When the interface is zoomed out, the user can listen to a larger area of the map. This may be beneficial when trying to gain an overview of the map, but may be too much information if searching for a particular song. The user zooms in to listen to only the songs closest to the listening position.

The maximum area the user can listen to corresponds with the distance gain attenuation described above. The interface only zooms out until the distance of a sound source directly in front of the listener has a gain of zero.

2.4. Auditory Landmarks

When interacting with a map of music, users need to retain information about the virtual space and not become disoriented. However, the musical content of a song develops over time, so that a 3 second from one section of a song may sound very different to a 3 second clip from another section. This means that audio files may need to be limited in time so that a particular sound or auditory landmark can be easily associated with a location in the virtual space. *nepTune* does this by limiting the audio playback to the middle 30 seconds of each song so that the music does not change too much if a user returns to a previously-visited virtual position. While automatic thumbnailing attempts to identify a short representative clip of a longer piece of music, it is not a solved problem. The *amblr* uses the simplified and popular approach of limiting the audio playback to the first 20 seconds of each song. Ideally, the *amblr* would be implemented with an automatic thumbnailer that can concisely summarize the audio content of an entire song in approximately 20 seconds.

Similarly, the *MusicCAVE* implementation of the *Audio Square* environment and the interface described in [3] have a return to "home" function so that users can return to a known location if they become disoriented within the virtual space. This is implemented in the *amblr* so that when requested, the user is returned to the center of the map, leaving the other parameters such as the orientation and zoom level unchanged.

2.5. Cyclical Playback

The cyclical interface described in [4] mixes multiple audio files into a single stream and weights each individual audio file to reflect its ranking as a result to a query (no spatialization is used). The effect is a short song clip fading out as another clip fades in so that multiple songs can be quickly reviewed. Visualization of the audio provides metadata and helps the user to differentiate between songs. The audio signal is analyzed in order to automatically present the sections of the song that may be of the most interest. The sections with the highest level of information in the frequency domain, high spectral entropy, are considered the most interesting. Only the first 20-30 seconds are considered.

Presenting multiple songs simultaneously can become overwhelming to some users, but the *amblr* uses a similar cyclical presentation. When a user decides to listen to only a single song at a time, they can still listen to multiple songs over a shorter period of time than would be required if they listened to each song sequentially. A clip of a song is played and then is immediately followed by a clip of another song. The songs played are all of the songs within the listening area defined by the zoom function. Each clip is played from the location defined by the map and the user's listening position.

All songs within the listening area are heard within 20 seconds of listening. In the top left map in Fig. 1, the largest circle or most zoomed out listening area contains 17 songs. A one second clip of each of the 17 songs would be played before repeating any of the songs. If the user zooms in to the smallest listening area which contains 2 songs, each song would be played for 10 seconds.

The length of time a song is played is:

$$t(n) = \begin{cases} 20/n & \text{if } 1 < n < 66 \\ 0.3 & \text{if } n \geq 66 \end{cases} \quad (4)$$

where t is the time in seconds that each song is played when there are n songs.

The user can choose to listen to up to four songs playing concurrently. Multiple simultaneous songs are best distinguished if they are in different locations. If more than one song is played concurrently, the listening area is dividing into regions to encourage concurrent playback of songs from differing locations. The length of time each song is played is then determined by the number of songs in that portion of the playback region. For example, in the top right map in Fig. 1, the listening area is divided into two portions so two songs at a time will play concurrently. In the largest, zoomed out listening area, 11 songs will play for a little under 2 seconds each to one side of the listener and 6 songs will play for a little over 3 seconds each to the other side.

2.6. Selection

The *amblr* is intended for music browsing, so if a song of interest is found, then a mechanism to select that song is needed. The selection of a song could be used for a variety of tasks such as becoming the seed to a search, being added to a playlist, or perhaps purchasing that song from an online store. All of those tasks require the browsing of a collection of songs and then the selection of a single song.

As there is no visual confirmation of the selected song within the *amblr*, an aural confirmation is presented. The user selects a song by approaching the song. When they indicate they would like to select it, they then listen only to the selected song to confirm their choice. The confirmation is needed then to make sure that the song the user believes they are selecting

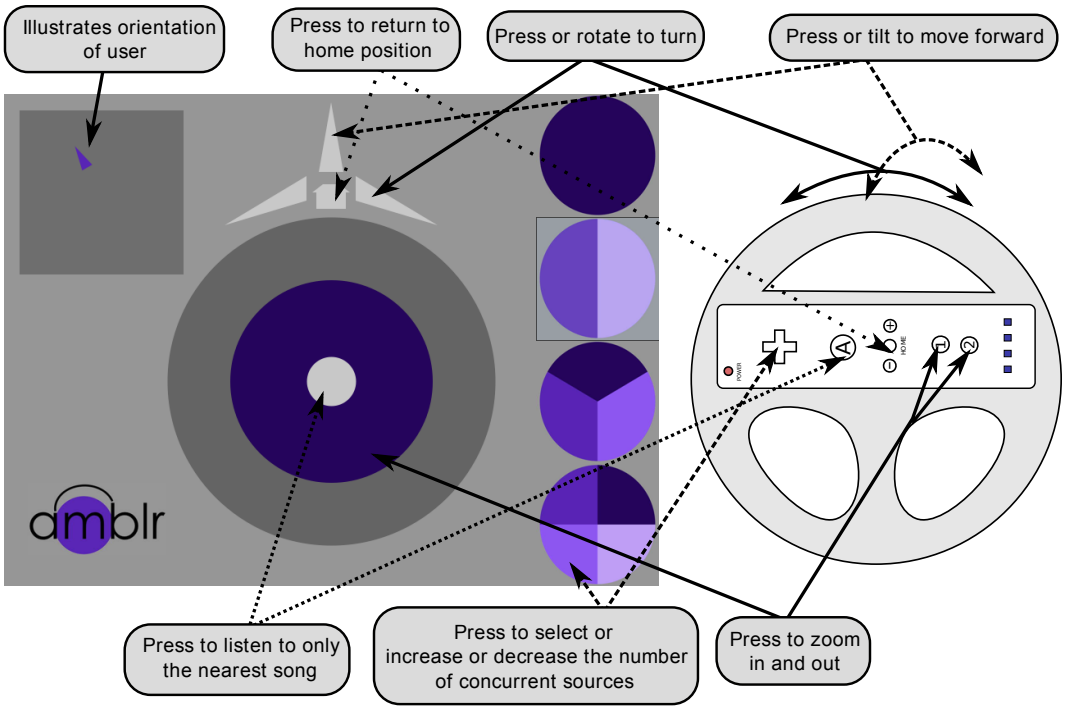

Fig. 2. Screenshot of the GUI and diagram explaining how the mouse and Wii remote are used to control the *amblr*.

is in fact the nearest song. This is akin to a dialog box in a graphical user interface confirming an action.

3. USER INTERFACE

The interface has two modes of input: a physical controller (a Wii remote) and a graphical user interface (GUI). The Wii remote is used a prototype of a mobile device as it contains the most common sensors for feedback and input found on mobile devices. The visual display is optional if using the physical controller instead of the mouse (or touchscreen if on a mobile device). The audio output is identical when using either controller and all functionality is completely duplicated.

3.1. Graphical User Interface

The *amblr*'s GUI can be seen in Fig. 2. It displays the location and orientation of the user within the two-dimensional map to give the user structure and prevent them from becoming lost. The interface explicitly does not show the location of any of the songs nor does it display any metadata about the collection. This is so the auditory display is still the primary mode of interaction and is only complemented, but cannot be replaced, by the visual display.

The central section of the visual display is a series of concentric circles. The outermost grey circle and innermost white circle do not change their sizes, but the middle purple circle will change its diameter. The new diameter adjusts to where the user clicks the mouse with the white and grey circles being the minimum and maximum sizes. The size of the purple circle controls the zoom level with the larger the circle the larger the listening area of the map.

Clicking the white circle in the center of the concentric circles selects the nearest song. Only that song is then heard; if any other songs are playing, they are muted. Clicking on the white circle again brings back the full virtual space and all songs are heard according to the parameters of the application.

The collection of triangles at the top of the concentric circles allow the user to move around the virtual space. The triangle pointing up moves the user forward and the other two arrows rotate the user to the left or right. By clicking on the icon of a house, the user is returned to the home position at the center of the map.

On the right-hand side of the visual display is four circles each divided into one to four partitions. They each represent the number of concurrent sound sources and illustrate how the listening area is partitioned. The user clicks on the appropriate circle to change the number of sources.

3.2. Haptic Interface

The *amblr* uses a Wii remote held in a steering wheel configuration and holder as for Wii racing games. The steering wheel gives context for how the physical interface is used and is less ambiguous as to how it is held. The accelerometers are used for rotating the sound field and for moving the user forward,

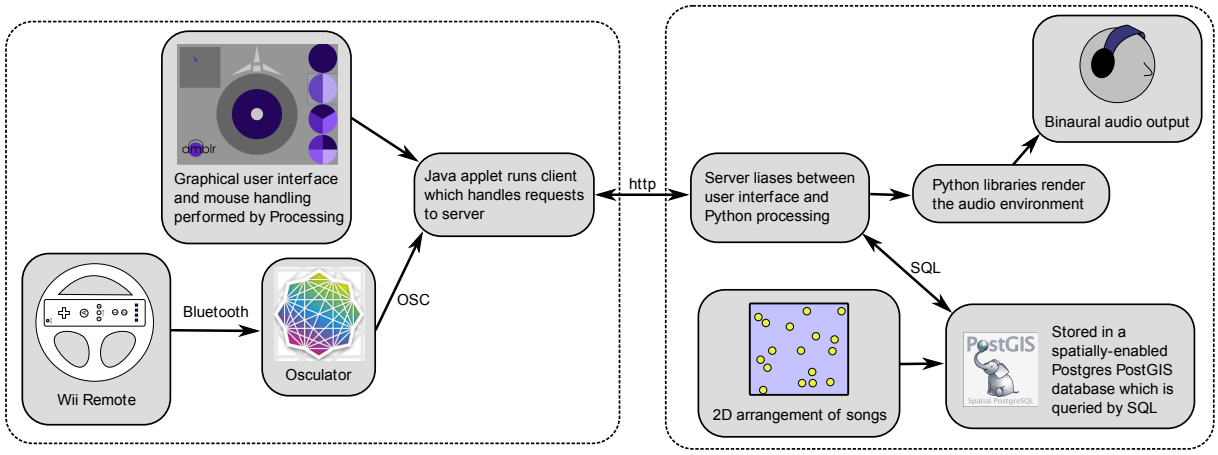

Fig. 3. System architecture.

identical to the forward, left, and right arrows in the GUI. The rest of the parameters are controlled by buttons. Turning the steering wheel to the left or right as if driving rotates the user in the corresponding direction. Tilting the steering wheel forward moves the user forward.

The Wii remote has a cross button on the left of the controller, a large circular button in the center, and two smaller circular buttons on the right. The user increases the number of concurrent sound sources by pressing either up or right on the cross button and can decrease by pressing down or left. The zoom level is increased by pressing the left smaller circular button (labeled "2") and decreased by pressing the right smaller circular button (labeled "1"). In the center of the Wii remote is a button labeled "HOME." Pressing the button will return the user to the center of the map.

The Wii remote vibrates when the user approaches and "hits" a song. If the user presses the large circular button (labeled "A"), they hear only the nearest song. If they hit the button again they return to the full virtual environment.

4. SYSTEM IMPLEMENTATION

The *amblr* system is based on a client-server architecture. While the entire system is run locally on a single computer, the design is intentionally kept flexible so that it could instead stream audio from a central database of music to a client running the system. This prototype plays the music back from the machine hosting the server, but ideally the spatialized music would be streamed to the client device.

The backend of the system is run by a CherryPy[1] server. The audio is rendered with custom Python libraries and uses Python bindings to PortAudio[2] to output binaural audio to the soundcard. The map of songs is managed by a PostGIS[3] database. PostGIS is a spatially-enabled PostgreSQL[4] database. The database is queried with SQL to determine what songs are nearest to the user's position on the map.

The Wii remote connects to the computer through Bluetooth. Osculator[5] is used to map the accelerometers, buttons, and haptic feedback to OSC (Open Sound Control)[6].

The GUI is written in Java using Processing[7] to render the visual display and handle mouse input. The Java client also sends and receives OSC to the Wii remote via Osculator. The client updates the Python server with HTTP requests using a REST-like structure. It deviates from REST because a client-stateless-server architecture is not maintained [12]. Instead, the client is considered stateless while the server maintains the parameter values and context until updated with new parameters by the client. The server is updated with the user's location, orientation, zoom, and number of concurrent sound sources. Information such as the user's current location and orientation and whether the user has "hit" a song is reported to the client via JSON[8].

4.1. Audio Rendering

The *amblr* uses virtual Ambisonics to create a binaural audio signal. The decoded loudspeaker signals of an Ambisonics decoder are convolved with head-related transfer functions (HRTFs) instead of fed to loudspeakers. The primary computational advantage of this approach is that a static set of HRTFs are used and there is no need for interpolation. See [13] for a derivation of a B-format to binaural decoder.

[1] http://www.cherrypy.org
[2] http://www.portaudio.com/
[3] http://postgis.refractions.net/
[4] http://www.postgresql.org/
[5] http://osculator.net
[6] http://opensoundcontrol.org/
[7] http://processing.org
[8] http://www.json.org/

5. DISCUSSION

This paper presented a new music browser called the *amblr* which is influenced by previous music browsers that interact with two-dimensional maps of songs. The *amblr* builds on: auras first described in [1]; perception of focus as implemented in [5]; auditory landmarks as discussed in [11]; and cyclic presentation of audio content as explored in [4]. Additionally, the *amblr* utilizes audio confirmation of a selection, haptic interaction, and an auditory display that does not rely on a visual display.

While a formal evaluation was not conducted, approximately 40 people have informally offered feedback. When the interface is initially described to new users, some people have expressed concerns over concurrent playback, but they then find that they are not overwhelmed when using the interface. Several users have made particularly good suggestions like incorporating more traditional auditory display elements. Though auditory icons are a major portion of auditory display design for tasks other than exploring music, none of the interfaces cited here nor the *amblr* make use of them. Frustration occurs when the interface occurs no longer appears to be responding. The user may assume that the application is no longer functioning, when instead they are misunderstanding how to use it or are asking it to do something incorrectly. Auditory icons may be a way to resolve this issue.

We believe that the comments collected from the informal evaluation would be echoed in a more formal study. The overriding tone of the feedback is that a map-based approach to browsing a collection may be a fun exploratory game, but it is not a tool where someone could easily become an expert user. This ultimately may be a fault of using a map to describe an entire collection. Collection level browsing makes less sense as collections grow in size, certainly when considering the processing constraints of analyzing an entire collection. Additionally, maps are not intuitive – a space that makes sense to one user, may hold no meaning to another. As automatic one dimensional arrangements of music (e.g. playlists) are not solved, perhaps then automatically arranging two or three dimensions is beyond the abilities of current research.

Map-based approaches for mobile applications which cannot support immersive visual displays expect more from spatial audio than it can perhaps deliver. Without the visual cues to reinforce the virtual auditory environment, a user can easily become disoriented and confused. When considering the drawbacks when using maps in a mobile music browsing context, we feel that other interaction paradigms should be considered.

In conclusion, we find that carefully restricting the audio playback to allow the user to have greater control over the interface is needed for a map interface, but future interface designs should look beyond the map for scalable, intuitive methods for music collection navigation.

6. REFERENCES

[1] Mikael Fernström and Eoin Brazil, "Sonic browsing: an auditory tool for multimedia asset management," in *ICAD '01*, Espoo, Finland, August 2001.

[2] Peter Knees, Markus Schedi, Tim Pohle, and Gerhard Widmer, "An innovative three-dimensional user interfacefor exploring music collections enriched with meta-information from the web," in *ACM Multimedia*, 2006.

[3] Rebecca Stewart, Mark Levy, and Mark Sandler, "3D interactive environment for music collection navigation," in *DAFx '08: 11th Internatioal Conf. on Digital Audio Effects*, Helsinki, Finland, September 2008.

[4] S. Ali and P. Aarabi, "A cyclic interface for the presentation of multiple music files," *IEEE Transactions on Multimedia*, vol. 10, no. 5, pp. 780–793, August 2008.

[5] Dominik Lübbers and Matthias Jarke, "Adaptive multimodal exploration of music collections," in *ISMIR'09: 10th International Society for Music Information Retrieval Conference*, Kyoto, Japan, 2009, pp. 195–200.

[6] Chris Schmandt and Atty Mullins, "AudioStreamer: exploiting simultaneity for listening," in *CHI'95*, May 1995.

[7] Chris Schmandt, "Audio Hallway: a virtual acoustic environment for browsing," in *ACM UIST*, San Francisco, CA, 1998.

[8] Mikael Fernström and Caolan McNamara, "After direct manipulation - direct sonification," in *ICAD '98*, 1998.

[9] Wilko Heuten, Niels Henze, and Susanne Boll, "Interactive exploration of city maps with auditory torches," in *CHI'07*, San Jose, CA, USA, April 2007.

[10] Sebastian Heise, Michael Hlatky, and Jö Loviscach, "Aurally and visually enhanced audio search with SoundTorch," in *Human Factors in Computing Systems - CHI*, Boston, MA, USA, April 2009.

[11] Piotr D. Adamczyk, "Seeing sounds: exploring musical social networks," in *MULTIMEDIA '04: Proceedings of the 12th Annual ACM International Conference on Multimedia*, New York, NY, 2004, pp. 512–515.

[12] Roy Thomas Fielding, *REST: Architectural Styles and the Design of Network-based Software Architectures*, Doctoral dissertation, University of California, Irvine, 2000.

[13] Bruce Wiggins, *An investigation into the real-time manipulation and control of three-dimensional sound fields*, Ph.D. thesis, University of Derby, Derby, UK, 2004.