SoundSling: A Framework for Using Creative Motion Data to Pan Audio Across a Mobile Device Speaker Array

Anthony T. Marasco
School of Music and CCT
Experimental Music & Digital Media
Louisiana State University
Baton Rouge, LA
amarasco@lsu.edu

Jesse Allison
School of Music and CCT
Experimental Music & Digital Media
Louisiana State University
Baton Rouge, LA
jtallison@lsu.edu

ABSTRACT

SoundSling is a framework used to translate motion-based data into audio diffusion trajectories across a crowd of networked mobile devices. The intent is to allow a performer to distribute audio across audience mobile devices in creative ways with motion data that mimics various patterns of movement found in the natural world. As a sound is "slung" around the room, the software intuitively adjusts each audience member's gain as it moves past their location. SoundSling adapts dynamically to the total number of devices as users connect to or disconnect from the network. This helps to ensure that the performer's chosen diffusion patterns and motion trajectories can be scaled properly to the array of currently-participating devices. Existing as a collection of MaxMSP abstractions and easily-editable web page templates, a focus has been kept on making the tool as adaptable to a performer's current musical set-up as possible.

1. INTRODUCTION

1.1 Inspiration

In recent years, performances that feature distributed sound across networked mobile devices have become more common place both inside and outside of the Web Audio community. While the origins of this performance practice stretch back into the mid-twentieth century[7], recent evolutions in the Web Audio API and the process of creating meshed data distribution networks have allowed contemporary composers to embrace this technique to fruitful effect. Audiovisual works such as *Fields*(2015), by **Tim Shaw** and **Sébastien Piquemal**[6] and *Diamonds in Dystopia*(2016) by **Jesse Allison**, **Frederick Ostrenko** and **Vincent Cellucci** [1] feature distributed performance systems in order to localize audio events across audience mobile devices connected to a centralized server.

As more artists seek to experiment with distributed performance systems, the development of new tools that allow for a streamlined and user-friendly approach into the practice is necessary. Recent tools for distributed audio such as the Soundworks and Nü Soundworks frameworks provide artists with a litany of options for tracking device location and rearranging the speaker array configuration[5][4], but customization of the framework's structure is fairly complex and can be daunting to those without previous JavaScript experience. Therefore, SoundSling contains a series of easy-to-learn web page templates that communicate with the framework's MaxMSP abstractions in the hopes that beginners will be able to start simple and advance their coding skills as their project needs grow.

Integrating SoundSling into audiovisual performance setups allows for the creation of a framework that responds to physics-based motion data and translates that data into creative audio panning trajectories (See Figure 1). Responding to motion-based data such as easing functions (which are used to create realistic motion trajectories of animated objects) or the oscillations of virtual sound-mass objects(such as tracking the oscillation of a virtual string) allows for a more direct correlation between a sound's realized location in the concert hall and any visual elements/performative gestures generated during the performance.

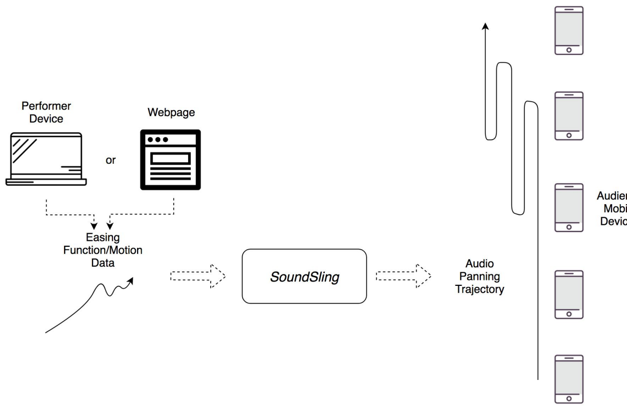

Figure 1: Diagram showing a performance setup utilizing the SoundSling tool set.

1.2 Advantages for Users

Designed to be user-friendly, SoundSling[1] was developed

Web Audio Conference WAC-2018, September 19–21, 2018, Berlin, Germany.

[1] Git: https://github.com/tmarasco125/soundSling, video: https://vimeo.com/user5885752

for performers who may have minimal experience with distributing sound to audience devices in their artistic work. The SoundSling framework has major usability advantages that make it an attractive choice for electroacoustic and audiovisual performers:

1. SoundSling is scalable, allowing for multiple, unique slings to be generated and run simultaneously during a performance.

2. Due to its messaging scheme, SoundSling is capable of limiting a sound's trajectory to specified locations/-groups of audience member mobile devices in a single concert hall.

3. All audio playback and intuitive amplitude adjustments are handled client-side on the audience's mobile devices, avoiding unnecessary CPU strain on the performer's device.

4. Motion-based data can be generated and streamed into SoundSling abstractions on-stage by the performer, or from a remote web page (useful in sound art installation settings).

5. SoundSling's MaxMSP abstractions are built using the JWeb object, which allows for communication with a web page or node server with minimal risk of being blocked by a firewall.

6. The communication elements of SoundSling are handled by Rhizome[2], a Node.js server application that offers a flexible messaging scheme. This simplifies the ability of the performer to trigger playback events, update the sound location, and receive updated layouts of the audience speaker array during a live performance. Rhizome servers can be deployed locally or from a cloud service, allowing for the potential for globally-networked performances.

2. SYSTEM DESCRIPTION

2.1 Implementation

SoundSling can be launched via the terminal, or by creating an instance of any SoundSling abstraction in MaxMSP and triggering its internal shell object in order to automate the corresponding Rhizome configuration script. Once running, Rhizome hosts and establishes connection between two web pages: one, a Communicator web page that routes publishes and routes messages to any subscribing clients, and another that serves as the Audience UI/Performance page, containing a pre-loaded audio file. When designing these pages, the performer can declare specific subscription tags to route sound location data through, effectively creating a single, unified speaker array across the entire audience or designing specific speaker nodes by splitting the audience into multiple subsections based on their location. During performance, motion-based data streams and trigger messages are routed into a SoundSling abstraction and used to determine the audio panning trajectory for a triggered sound file. Users can choose between multiple MaxMSP SoundSling abstractions (soundsling.single, soundsling.multi, or soundsling.2D) based on the desired number and direction of individual slings.

The interface between the Rhizome server and the performer's MaxMSP patch consists of a simple set of commands that are translated to OSC internally:

```
/speakerBleed 0.5
/soundLocation 0.79
/playCue 1
```

For the use of multiple slings, an instance of each message channel is made:

```
/speakerBleed/1 0.3
/soundLocation/2 0.45
```

Audience members pick their location in the concert hall by logging onto the Audience UI/Performance page and choosing X/Y grid of available seats. With each new addition to or subtraction from the speaker array, a graphic UI object in the performer's Max patch provides them with a visual layout of all connected devices and their location in the hall, allowing them to make decisions about sling trajectories on the fly.

2.2 Intuitive Amplitude Changes

Each device's location is used as the mean of a calculated Gaussian curve. The tail ends of each curve overlap with the curve of the adjacent audience mobile device. The slingIt() function translates these curves into individualized amplitude response curves, which react to the sound location data being sent from the performer's device:

```
function calcGaussian(x, mean, spread, scale) {
    return (Math.exp((-1 * (x - mean) * (x - mean)) /
    (2 * spread * spread))
    / (spread * Math.sqrt(2 * Math.PI))) * scale;
}


function slingIt() {

    var ampCurve = calcGaussian(soundLoc, speakerMean,
    0.25 * speakerBleed, 0.627 * speakerBleed);

    if (ampCurve) {
        return ampCurve;
    } else {
        return 0.;
    }
}
```

As the performer "slings" a sound around the room, each audience member's device is triggered to playback the audio file simultaneously and increase or decrease its amplitude as the sound crosses over their location. This approach to generating intuitive amplitude changes allows for a seamless transition of sound localization as the audio file travels across one device to the next (see Figure 2). Performers can adjust the width of the amplitude response curves in real-time, allowing for various amounts of speaker bleed-over between devices. This feature is useful to create more natural sonic transitions based on the density of connected devices in single row or subsection of the concert hall.

3. CONCLUSIONS & FUTURE DIRECTIONS

Initial tests with SoundSling have proven fruitful. Additional work on to the SoundSling framework will focus on

[2] https://github.com/sebpiq/rhizome

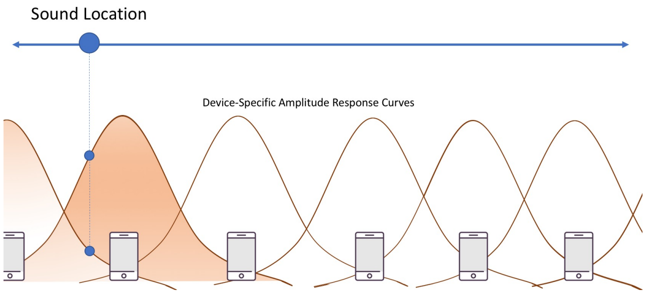

Figure 2: Device-specific amplitude response curves, overlapping with adjacent audience member devices. As the sound location changes, each device adjusts its amplitude accordingly.

adapting the current set of MaxMSP abstractions into a set of Max4Live devices, as well as versions for the Pure Data environment in order to accommodate a wider variety of performers. In the future, SoundSling will be implemented into NexusHub.[3]With recent developments in Web Audio implementations on embedded microcomputers[2], future experiments will focus on the ability to integrate SoundSling into embedded electroacoustic instruments and controllers outside of a visual coding environment.

Moving away from the limitations of triggering preloaded audio files from the Audience UI/Performance pages, a new set of web templates and abstractions that focus on the transmission of OSC-blobs are currently in development. Live streaming of audio from the stage will be a subject of future tests and research into the complications of integrating this feature[3] into the current SoundSling framework have already begun.

4. REFERENCES

[1] J. Allison, F. Ostrenko, and V. Cellucci. Active server roles for extended distributed performance complexity in diamonds in dystopia. In *Proceedings of 3rd Web Audio Conference, London, August 2017*. Queen Mary University of London, 2017.

[2] J.-T. Milde. Synchronized mobile devices using web audio technology on a raspberry pi. In *Proceedings of 3rd Web Audio Conference, London, August 2017*. Queen Mary University of London, 2017.

[3] M. Paradis, C. Pike, R. Day, and F. Melchior. A novel approach to streaming and client side rendering of multichannel audio with synchronised metadata. In *Proceedings of 2nd Web Audio Conference, Atlanta, 2016*, 2016.

[4] D. Poirier-Quinot, B. Matuszewski, N. Schnell, and O. Warusfel. Nü soundworks: Using spectators smartphones as a distributed network of speakers and sensors during live performances. In *Proceedings of 3rd Web Audio Conference, London, August 2017*. Queen Mary University of London, 2017.

[5] S. Robaszkiewicz and N. Schnell. Soundworks - a playground for artists and developers to create collaborative mobile web performances. In *Proceedings of 1st Web Audio Conference, IRCAM, January 2015*. IRCAM, 2015.

[6] T. Shaw, S. Piquemal, and J. Bowers. Fields: An exploration into the use of mobile devices as a medium for sound diffusion. In E. Berdahl and J. Allison, editors, *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 281–284, Baton Rouge, Louisiana, USA, May 2015. Louisiana State University.

[7] B. Taylor. A history of the audience as a speaker array. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 481–486, Copenhagen, Denmark, 2017. Aalborg University Copenhagen.

[3] https://github.com/nexus-js/nexusHub