sonic surfer

a binaural audio game by Sam Smith

overview

The initial vision for this game was based on control systems such as the Wii Fit and mobile phone games like Temple Run. Yet, the model for this game does not rely on beating levels or defeating "bosses". Instead, Sonic Surfer relies on auditory feedback and physical control: the player leans on a custom built platform in the direction of the noises that seem happy/safe. The hardware was built using an arduino uno, 4 Force Sensing Resistors (FSRs) and a laser cut enclosure to house the components. The software was programmed primarily in max/MSP in addition to java to report the sensors data to max/MSP (fig. 1-2). This game relies on intense focus and extreme coordination.

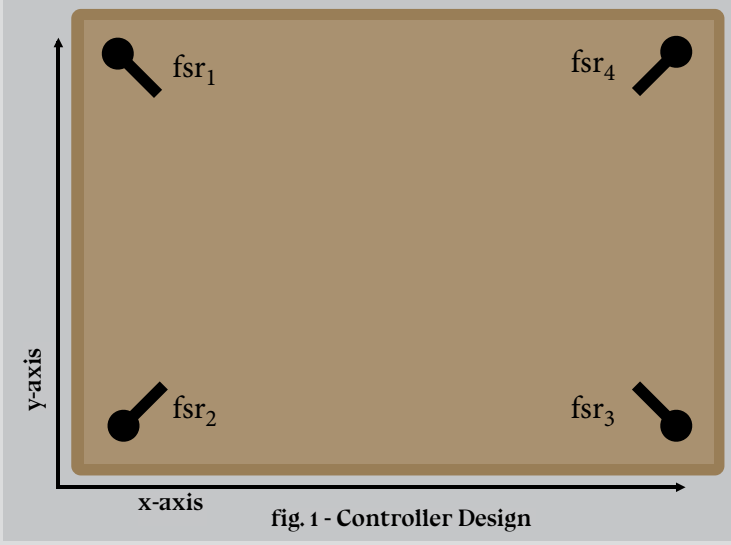

fig. 1 - Controller Design

introduction

Binaural audio is a format of audio encoding and decoding that allows for sound source spatialization within a pair of headphones. Through the usage of Head Related Transfer Functions (HRTFs), a mathematical equation is used to determine how a given sound will be perceived by one's ear. This technology allows for spatialization of audio in a home setting.

Binaural audio has long been used as an alternative to more hardware intensive and intricate systems like ambisonics. Recently, binaural audio has made its way into video games such as Mario-cart, Modern Warfare and the like, under the pseudonym "3d audio". This audio format gave players with headphones the advantage of hearing sounds behind them so that they can proactively enhance their performance. The challenge of creating audio-only games is designing compelling interaction and feedback, which Sonic Surfer seeks to explore and improve upon.

hardware

This model of interaction was based around designing a controller with intuitive control. Sonic Surfer's controller stands on a panel with 4 FSRs to detect the level of pressure at each corner and help me determine where the user was leaning on the board. I used an ardiuno uno in order to read the data.

interaction and feedback

The interaction between the user and the game is based around the human ability to localize sound. The player is prompted to lean towards the "good sounds" and to avoid the "bad sounds". This interaction model allows for interpretation by the player of what is good and what is bad. The control board is hyper sensitive even to the smallest shifts in weight, making it extremely effective for accurate localization of sound. I chose to engage with binaural audio because it allowed for me to use the forward/backward lean of the player as a mode of control as opposed to just left/right.

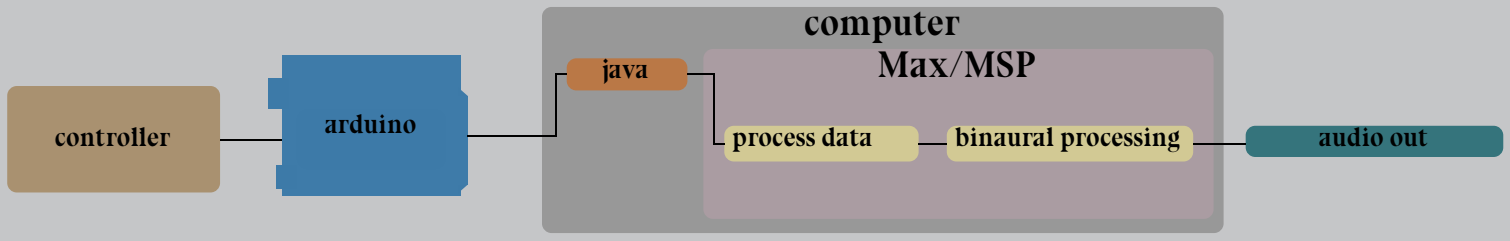

fig. 2 - signal flow

software

note generation

The melodies for the synth voices are generate by randomly selecting a sequence length (number of notes). For each of these notes, a randomly selected scale degree is assigned to the sequence. Note length is also assigned to each note in the sequence. The note sequence is then cycle'd through and fed to two voices, both based around subtractive synthesis.

Additionally, as the game progresses, changes are made to the parameters on the note generation algorithm. Some parameters include note length as well as the scale that the generator uses. These changes correspond to the success of the player. As the player makes more and more mistakes the note lengths become much shorter and the scale is switched to a minor or chormatic scale. This makes it even harder to differentiate the voices.

coordinate generation

The two primary sound sources (synth voices) are localized via a randomized coordinate generator. We can describe the sound locations in this game on a cartesian coordinate system from -1 to 1 (front to back, left to right). The coordinate generator utilizes the max/MSP [drunk] object to hop from value to value randomly while sweeping across the map. These sound sources move around each other and intersect at time. From here the coordinate generators send their data to the binaural audio encoder and subsequently encode the individual sound sources to the coorresponding location. Throughout the game, the coordinates from the controller are compared to those of the good and bad sound sources. Based on how close these values are the game either becomes more or less difficult. The players score is ultimately a reflection of how long they were able to play the game before failing.

sound design

Both the good and bad synth voices were generated through digital synthesis within max/MSP. The voices have pitch bend and detuning controls. The detuning control changes the pitch of the triangle wave, while the square wave would stay in tune. The bad voice utilizes these controls a lot more than the good voice since the sliding tone of the pitch bend was very ominous.

As was mentioned before, as the game progreses, the sounds change in somewhat unpredictable ways. This is to ensure that the game is different each time it is played and to make the player question their performance in the game. This adds one more layer of difficulty to the model.

The atmospheric noise heard throughout the game is generated through a series of noise objects which have filters that modulate at different rates. This noise is the only audio feedback given according to where the player is currently leaning. This feature helps the user feel more immersed within the audio because their motions have a direct impact on what they are hearing and where in their headphones they sense it.

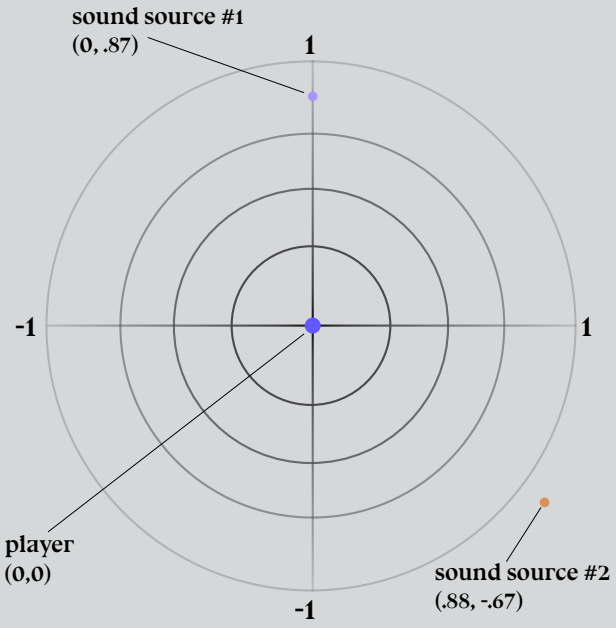

fig. 3 - example sound field

more information

game demo - https://bit.ly/sonic_surfer
source code - https://bit.ly/GH_source