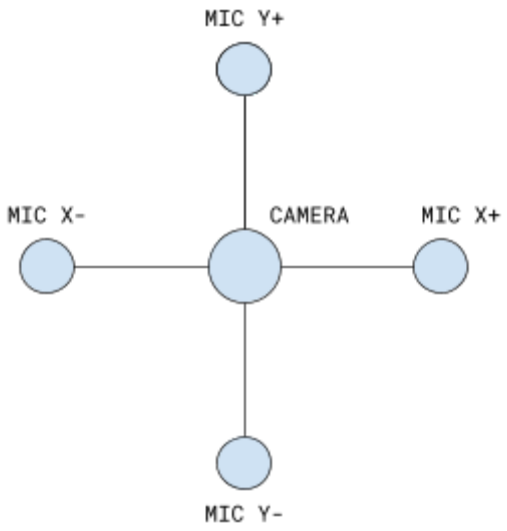
6.111 Project Proposal

Auditory Localization

Francis Wang and Keshav Gupta

Overview

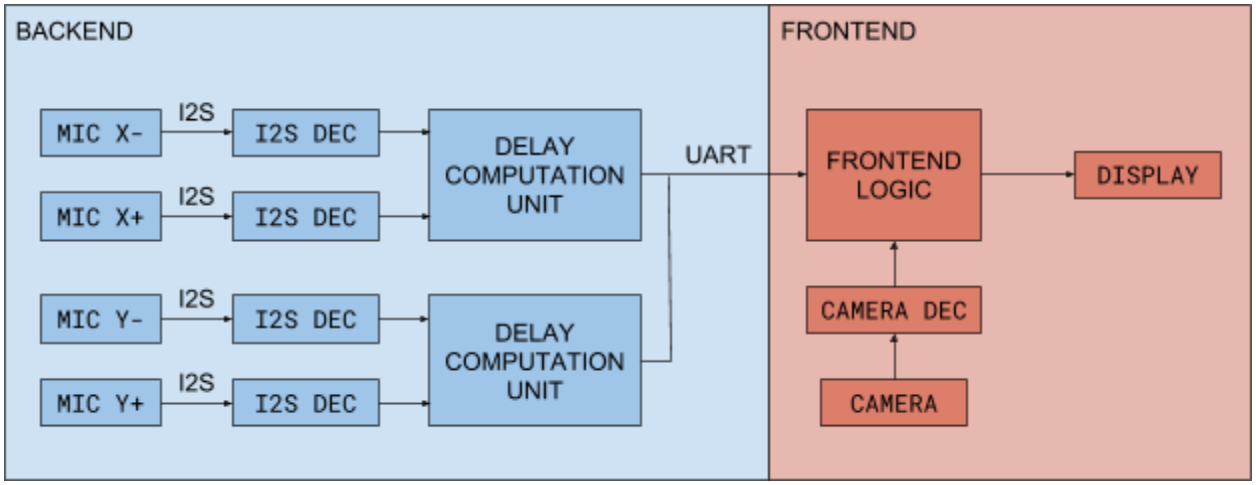
In this project we aim to demonstrate a system that is able to discern the direction of incoming sound through an array of microphones. The physical configuration of the system is as depicted in the diagram to the right. Four microphones will be arranged in a cross and the direction of the sound source will be determined by analyzing the relative delay of the waveforms measured by each pair of microphones in the X and Y axis. At the center of the cross will be a camera, and the position of the inferred sound source will be overlaid on the scene and displayed on the monitor.



For the microphones we plan on using the Adafruit I2S MEMS Microphone Breakout Board based on the SPH0645LM4H IC. The rationale for choosing a digital microphone over an analog one is that it would cut down on the noise introduced during data transmission. For the camera we plan to use a standard NTSC camera that is compatible with the ADV7185 video decoder on the labkit.

We will break this project up into two sections, which we will refer to as the “backend” and the “frontend.” The backend will be responsible for decoding the information that streams in from the microphones and performing the digital signal processing required to extract the time delay information. The frontend takes this delay information, as well as imagery data streaming in from the camera, and synthesizes it into a user interface, perhaps in the form of a game. In order to separate the computationally complex DSP from the more fluid user interface layer, we will be developing our system on two FPGAs. The backend will be implemented on the Nexys 4 while the frontend will be implemented on the Virtex 2 labkit. The two systems will communicate over the UART protocol based on an agreed upon data exchange format.

The division of work for this project will be as follows: Francis will develop the backend and Keshav will develop the frontend. A high level block diagram of the proposed system is presented below.

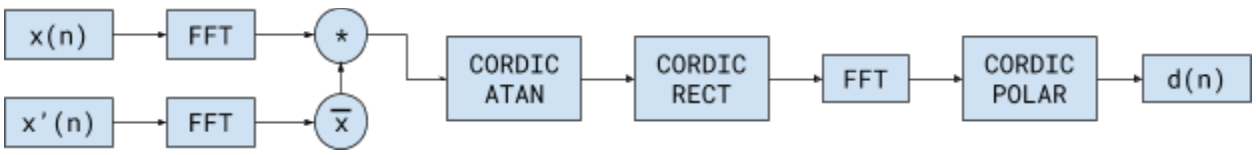


Backend

The microphones will be sampled at 64 kHz and the data will be broken up into frames of 256 samples and fed into the delay computation unit. We plan to use the generalized cross-correlation phase transform (GCC-PHAT) method to extract the delay information, although if that proves to be too demanding we will fallback to a time domain cross correlation computation. Given two input samples $x(n)$ and $x'(n)$, the time delay profile is given by $d(n) = |g(n)|$ where

$$G(f) = \frac{X(f)X'(f)^*}{|X(f)X'(f)^*|}.$$

$X(f)$, $X'(f)$, and $G(f)$ denote the discrete fourier transform of $x(n)$, $x'(n)$, and $g(n)$ respectively. The block diagram of the GCC-PHAT data processing pathway is presented below.



We will use the Fast Fourier Transform and CORDIC IP cores provided by Vivado for the mathematical manipulation. The main difficulty we expect is in configuring

the cores and interfacing with them over the AXI4-Stream protocol. A completely parallelized implementation would require six distinct FFT modules which could be beyond the hardware capabilities of the Nexus 4. Although it may be feasible to multiplex the FFT computations through time-sharing, we hope to avoid this situation through an appropriate downsampling of the data stream. The delay profile $d(n)$ will be sent over UART to the frontend system for post-processing and visualization.

As a baseline, we aim to have a resolution of 4 bits over the height and width of the camera display with a refresh rate of 1 Hz. Theoretically, at a sample rate of 64 kHz we can discern differences in path length down to an accuracy of 5 mm. An appropriate stretch goal would therefore be a resolution of 8 bits with a refresh rate of 60 Hz.

Frontend

The front end will receive the delay data from the backend over a UART link, as a sequence of 512 bytes (256 bytes for each axis), each byte representing the weightage for delay between -**t** and +**t**. The value of **t** needs to be measured. This information will then be stored in a 512 byte RAM block through it's write port. Since VGA pixel update will not be synchronized with incoming data (the incoming data will arrive once every few milliseconds with significant jitter, whereas display will update at 60 Hz which is not necessarily an integer multiple of incoming data frequency), the VGA module will use the read port of the RAM and update contents of the display at 60 Hz.

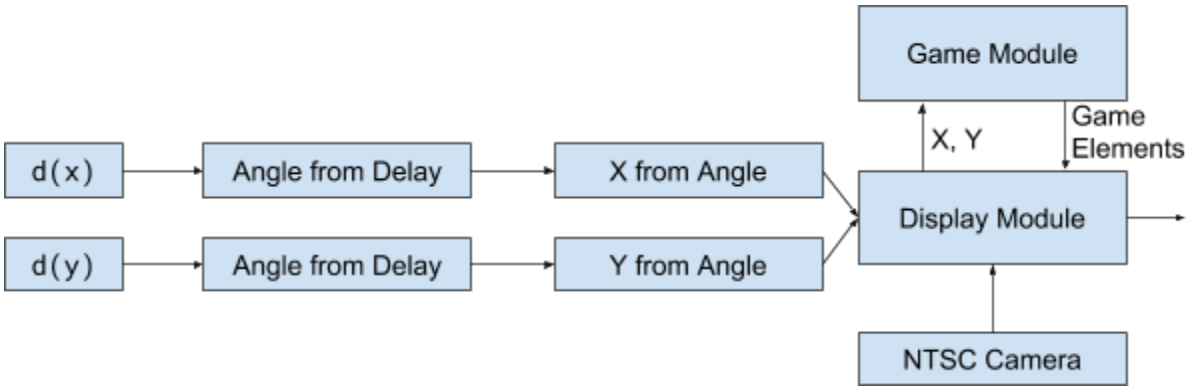
The most probably delays on the two axes are used to calculate the angle of the location at which sound was produced with respect to the axes. These angles (when compared to the field of view of the camera measure beforehand) give the X and Y location on screen. The angle can be calculated from delay as

$$\theta = \frac{\delta c}{2d}$$

where δ is the delay, c is the speed of sound, $2d$ is the separation between the microphones

(Proof in appendix 1)

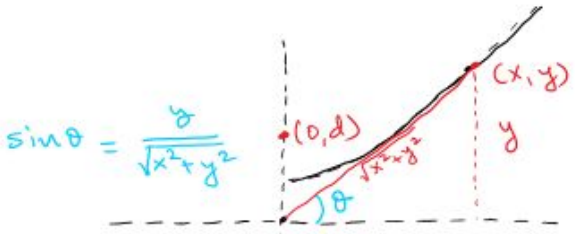
The standard goal would be to display a hotspot on the screen over the predicted region at which sound was produced. One of the stretch goals is to implement a bug squashing game where users need to position their hands over the on-screen location of bugs, and then snap their fingers to squash the bugs.



External Components

1. **Adafruit I2S MEMS Microphone Breakout - SPH0645LM4H (Qty: 4)**
   Adafruit link: https://www.adafruit.com/product/3421
   Status: already purchased
2. **NTSC Camera (Qty: 1)**
   Provided by the lab
3. **Ribbon Cable, Headers, etc**
   Provided by the lab

Appendix 1



Consider a delay δ between the two sound signals. Let the speed of sound be c.

The hyperbola is given by

$$\sqrt{x^2+(y+d)^2} - \sqrt{x^2+(y-d)^2} = (\delta c)$$

Simplifying, we get

$$2(x^2+y^2+d^2) - 2\sqrt{(x^2+y^2+d^2)^2 - 4y^2d^2} = (\delta c)^2$$

Assume the sound came from a far away location in space

$x, y \gg d$

$$\rightarrow \quad x^2+y^2 - \sqrt{(x^2+y^2)^2 - 4y^2d^2} = \frac{(\delta c)^2}{2}$$

$$\rightarrow \quad 1 - \sqrt{1 - \frac{4d^2y^2}{(x^2+y^2)^2}} = \frac{\delta^2 c^2}{2(x^2+y^2)}$$

This term is very small

The assumption $\sqrt{1+x} = 1 + \frac{x}{2}$

holds when x is very small.

So, $$1 - \left(1 - \frac{2d^2y^2}{(x^2+y^2)^2}\right) = \frac{\delta^2 c^2}{2(x^2+y^2)}$$

$$\rightarrow \quad \frac{2d^2}{x^2+y^2} \cdot \frac{y^2}{x^2+y^2} = \frac{\delta^2 c^2}{2(x^2+y^2)}$$

$$\rightarrow \quad \sin^2\theta = \frac{\delta^2 c^2}{4d^2}$$

$$\rightarrow \quad \sin\theta = \frac{\delta c}{2d}$$