Paint with Music

XXX
XXX

XXX
XXX

***Abstract*—The abstract goes here.**

I. Introduction

II. Prior Work

III. Sound Localization

points with the same difference of distance to two fixed points form a hyperbola in the 2D plane. Since each pair of microphones gives a hyperbola in the plane, localization becomes finding intersection of hyperbolas when more than two microphones are used. Localization relies on accuracte estimate of delay differences between microphones.

Generalized Cross Correlation(GCC) provides a framework to estimate delay differences t_0 between two signals $x_1(t)$ and $x_2(t)$:

$$t_0 = \arg\max_{\tau} \int_{-\infty}^{\infty} W(\omega) X_1(\omega) X_2^*(\omega) e^{j\omega\tau} d\omega \qquad (1)$$

, where $X_1(\omega)$ and $X_2(\omega)$ are Fourier Transform of $x_1(t)$ and $x_2(t)$. $W(\omega)$ provides a way to prefilter the signals passed to cross correlation estimator. We experiment with three ways of prefiltering the signal:

GCC	$W(\omega) = 1$. No prefiltering is done. This is normal cross correlation.
GCC_PHAT	$W(\omega) = \frac{1}{\lvert X_1(\omega) X_2^*(\omega) \rvert}$. Each frequency is divideded by its magnitude. Only phase information contributes to delay estimation.
GCC_PHAT_SQRT	$W(\omega) = \frac{1}{\lvert X_1(\omega) X_2^*(\omega) \rvert^{0.5}}$. This is somewhere between GCC and GCC_PHAT. part of magnitude information is included in delay estimation.

IV. System

The system uses two arrays, each with three microphones. Three microphones along with a micro-controller are mounted at three vertices of an 20cm equilateral triangle. Fig 1 shows a picture of the array. Micro-controller used in this project is *teensy 3.1*. The micro-controller has 64k RAM memory and the ADC is capable of sampling at 500kHz. In this project, the micro-controller collects microphone data on all three channels for 12 millisecond and then send the recorded data to a computer for localization.

To speed up computation for real time localization, instead of search for t_0 that maximizes equation 1. A grid-search in 2D grid is performed instead. With this approach, for each point in the grid, the arrival time difference to each microphone pair

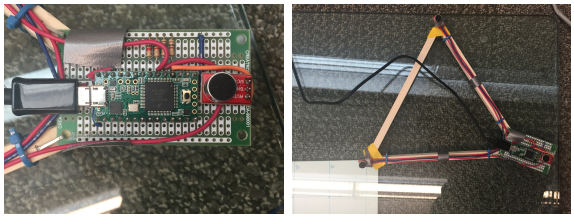
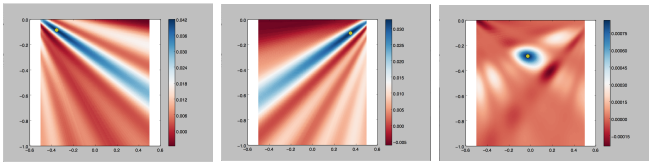


Fig. 1: array



(a) localization with only array 1 (b) localization with only array 2 (c) localization with both arrays

Fig. 2: Liklihood maps for localizating point at $(0.0, -0.3)$ m

can be precomputed. Then localization resolves to calculating GCC for each microphone pair and performing a lookup for each point in the grid. To further improve localization accuracy, instead of using a point estimate for difference of arrival time, a liklihoood map is built. Each entry in GCC output is used as the liklihood for that delay. With three microphones m_1, m_2, m_3, for each location (x, y) in the gird, the difference of arrival time to each microphones pair can be precomputed and stored in $D_{m_1,m_2}(x, y)$, $D_{m_1,m_3}(x, y)$, and $D_{m_2,m_3}(x, y)$. Let $R_{m_1,m_2}(\tau)$,$R_{m_1,m_2}(\tau)$, and $R_{m_1,m_2}(\tau)$ denote the GCC output for each microphone pair, then a liklihood map in the grid is calculated as:

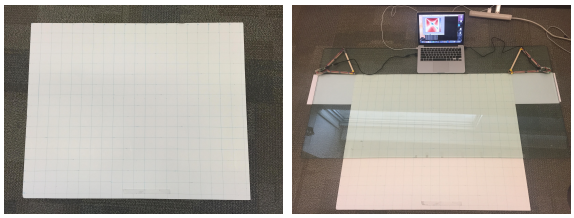
$$\begin{aligned} L(x, y) \quad = \quad & R_{m_1,m_2}(D_{m_1,m_2}(x, y)) + R_{m_1,m_3}(D_{m_1,m_3}(x, y)) \\ & + R_{m_2,m_3}(D_{m_2,m_3}(x, y)) \end{aligned}$$

Liklihood map from each array can be combined into final liklihood map:

$$L(x, y) = L_1(x, y) L_2(x, y) \qquad (2)$$

, where $L_1(x, y)$ and $L_2(x, y)$ represents the liklihood map from array 1 and array 2

To see the effect of using multiple arrays, fig 2 shows the individual liklihood map from each array and also the combined liklihood according to equation 2. Individual array gives accurate angle estimate, but has high uncertainty in distance estimate. By using two arrays, the angle estimate can be effectively combined to estimate distance.



(a) 1 meter by 1 meter grid (b) array placement

Fig. 3: Setup for localization accuracy testing

From a timing point of view, the micro controller spends 12 millisecond on sampling microphone data before sending the data to the computer for processing. Sending data through USB port takes another 18 millisecond, and processing on computer takes around 50 millisecond. Therefore, the totoal time lag between sound source and localization is around 80 millisecond.

V. Experiment

To test localization accuracy, an one meter by one meter grid was set up and the arrays are placed at the top left and top right corners of the grid. Fig 3 shows a picture of the setup.

A total of 32 positions are chosen uniformly in this region where microphone data is recorded. To test how accuracy varies with window size, the algorithm is fed with recorded microphone data with different segment length. Fig 4 shows how accuracy changes with window length for three GCC algorithms. The error lowers as window size increases and plateaus after window size exceeds around 10 milli-second. The lowest error achieved is 2.53 centimeters. It is achieved when window size is set to 12 millisecond and GCC_PHAT is used to estimate difference of delay

Although accuracy improves with window length, the calculation time also increases with window length. The part of calculation that depends on window length is using cross correlation to estimate difference of arrival time. cross correlation can be calculated with FFT and the runtime is of order $O(N \log N)$. We measured how the computation time varies with window length and Figure 5 shows the result. The runtime increases approximately linearly in the window size region of interest.

We also calculated the localization error for each tested point in the region. Figure 6 shows a heatmap of the error distribution inside the grid. The error is below 3 cm for most areas inside the region. There is one error spike in the mid-left region and we contribute this to audio source placecment error becuase the error is fairly low and consistent around that spike region.

To test how well the arrays track movement, we mounted a rotating disk 40 centimeter in diameter onto the grid at $(x = 0, y = -0.3)$. Fig 7 shows a picture of the sestup. A sound source is placed on the edge of rotating disk and the arrays localizes the sound source as it rotates in a circle. In this experiment, we tested how accuracy changes with:

- different window sizes

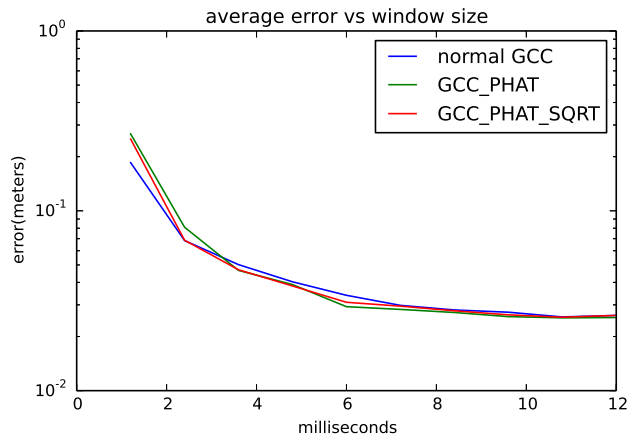


Fig. 4: accuracy versus window size

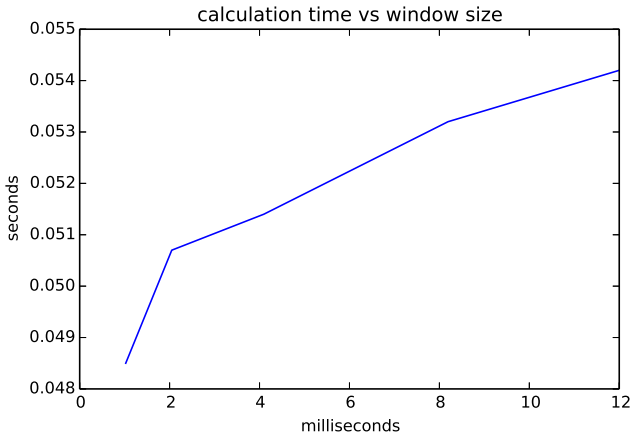


Fig. 5: speed versus window size

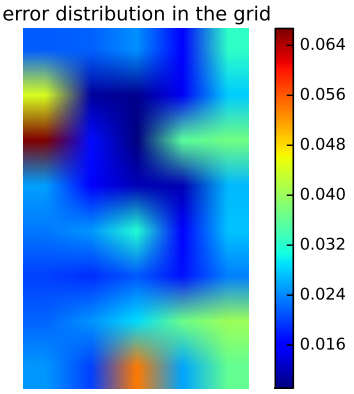


Fig. 6: error distribution in the grid

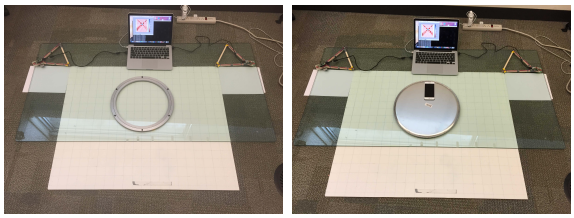


Fig. 7: Setup for circle movement localization

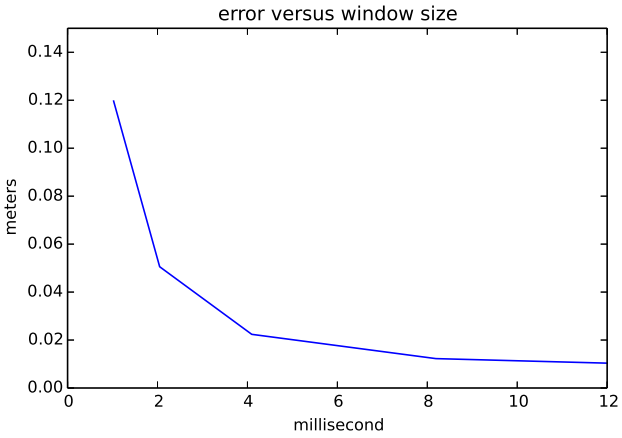


Fig. 9: Localization error versus window size

- different audio sources
- different movement tracking filters
- different movement speeds

Fig 9 gives an intuitive representation of how accuracy changes with window size. When window size is small(1.02 millisecond), the audio does not contain enough information to reliably estimate arrival time differences. The localization is noisy. As window size increases, the localization converges to the shape of ground truth circle. Fig 9 shows how the error changes with window size. The general trend is similar to that in point localization case. The error decreases as window size increases and plateaus after window size exceeds around 10 milliseconds.

To test how different sound sources impact localization quality, we conducted three experiments on the same movement track with three different sound sources:

White Noise	A recording of white noise.
Music A	A music that has normal audio amplitude throughout experiment period was chosen. "Honest Eyes" by Black Tide was chosen
Music B	A music with intermittent low amplitude sections was chosen. "Canon" was used.

To test how sound source movement speed affects localization quality, each of three experiments were conducted at two different speeds:

Normal	An angular speed of 0.5 rad/s was maintained, which translates to linear speed of 10 cm/s.
Fast	An angular speed of 1.0 rad/s was maintained, which translates to linear speed of 20 cm/s.

For each experiment conducted, two different movement filters are tests:

Averaging filter	localization for past 0.5 seconds are averaged and outputed as current estimate.
Kalman filter	A 2nd order Kalman filtering is used.

Fig 10 shows results for experiments with at normal speed, and Fig 11 shows results at fast speed. By comparing localization error for each audio source between normal movement speed and fast movement speed, we find the localization error does depend on how fast the sound sorce is moving. For example fig 10b and fig 11b shows that localization error is 1.289 cm at normal movement speed and 1.291 cm at fast movement speed.

From fig 10, localization error is 0.9 cm for white noise, 1.29 cm for Music A, and 2.9 cm for Music B. Localization accuracy is the best for white noise source, and worst for Music B. This is consistent with our expectation because low amplitude regions in Music B would make arrays lose track of where the source is. This can also be seen from the "blank" regions in fig 10c

Fig 10 also shows that raw detection has the most amount of jiggling. Kalman filter reduces the amount of jiggling from raw detection. Averaging filter has the least amount of jiggling. However, averaging filter averages detection outputs from past 0.5 seconds, which makes the filtered output lags the real movement.

VI. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

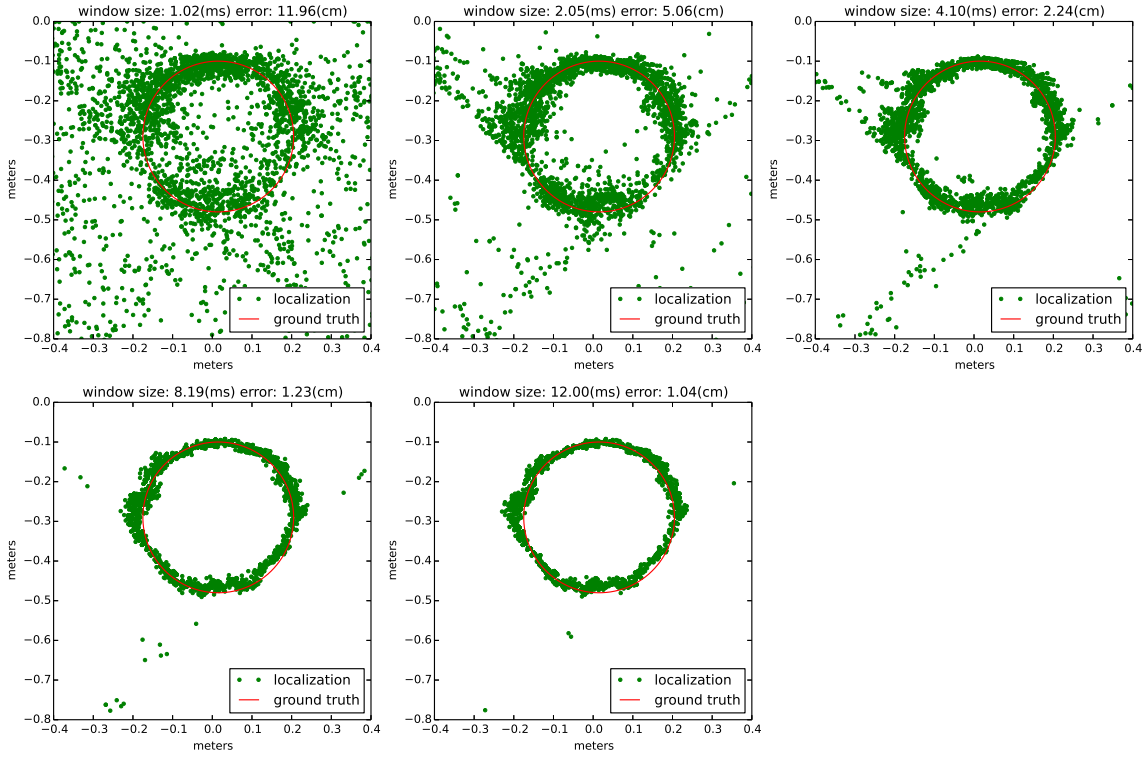
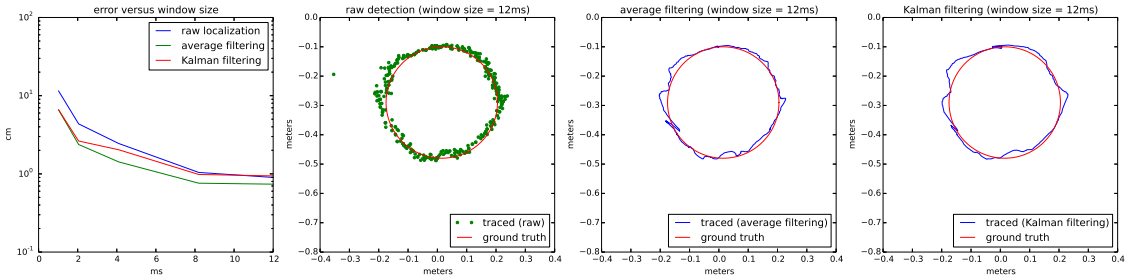
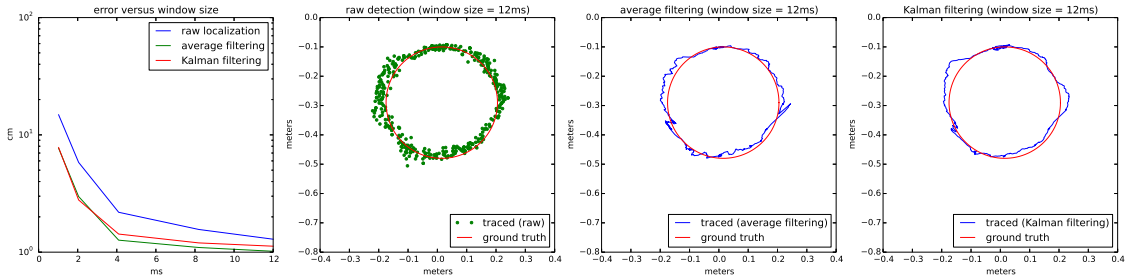


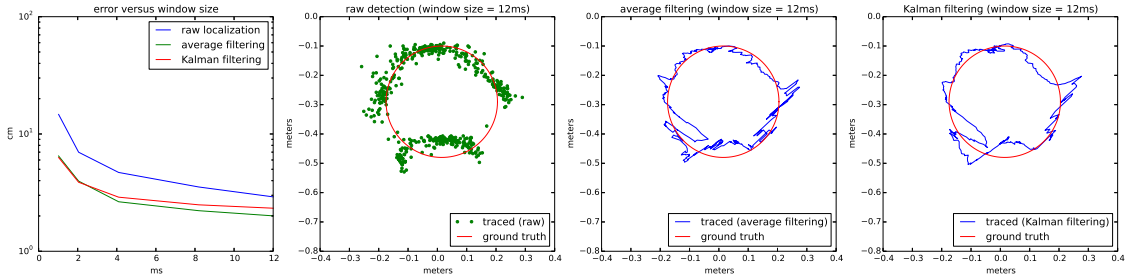
Fig. 8: Localization quality versus window size



(a) white noise

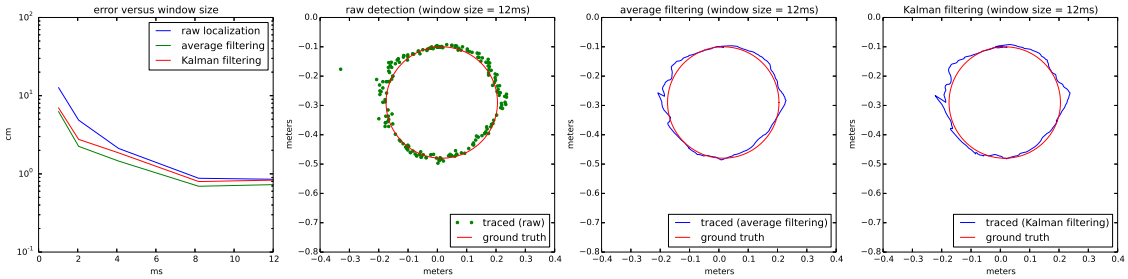


(b) music A

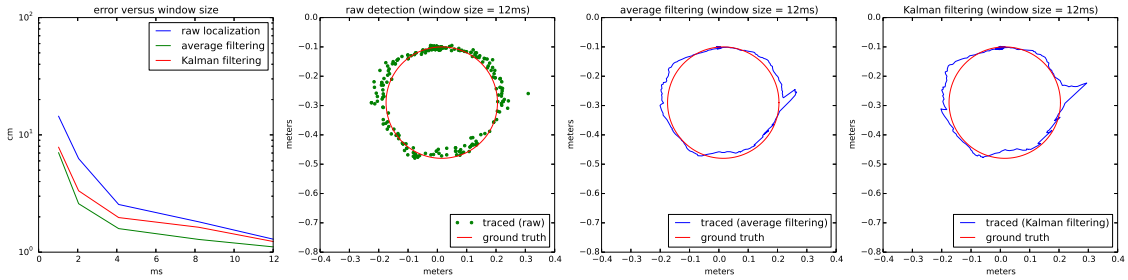


(c) music B

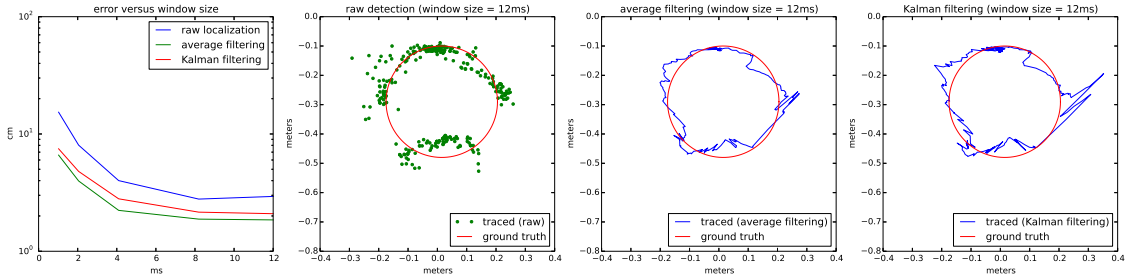
Fig. 10: Localization of circle movement with different sound sources. Sound source is moving at 10 cm per second



(a) white noise



(b) music A



(c) music B

Fig. 11: Localization of circle movement with different sound sources. Sound source is moving at 20 cm per second