MICROPHONE ARRAY BASED SURVEILLANCE SYSTEM

By

BLAIR ARMAND BALDRIDGE

Bachelor of Science in Computer Engineering

Oklahoma State University

Stillwater, Oklahoma

2013

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
July, 2015

MICROPHONE ARRAY BASED SURVEILLANCE SYSTEM

Thesis Approved:

Dr. Qi Cheng

Thesis Adviser

Dr. George Scheets

Dr. Keith Teague

Name: BLAIR ARMAND BALDRIDGE

Date of Degree: JULY, 2015

Title of Study: MICROPHONE ARRAY BASED SURVEILLANCE SYSTEM

Major Field: ELECTRICAL ENGINEERING

Abstract: This work attempts to explore an alternative surveillance method through the usage of a microphone array. Most of the current audio based surveillance work focuses on the detection of a single sound source. For any future real world applications it is very plausible that multiple sound sources will occur simultaneously at some point in time. In this work a method of detecting and localizing multiple sound sources is presented. Three state-of-the-art techniques are given that put together allow the system to estimate the approximate location of one or multiple sound sources, separate the sounds, and then properly identify the sounds.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Surveillance systems at one point in time were considered to be a high tech luxury item, but have now become a commonly featured component within a lot of households. There are many reasons that people decide to buy security systems. Crime and home invasions are generally a fear shared by most consumers. The use of a surveillance system can make the user feel safe, making it much easier to sleep at night knowing that no one has entered their home or personal space. Surveillance systems also tend to discourage potential criminals from attempting a robbery. If a robbery or some other crime does occur information the surveillance system provides can be used as evidence against the perpetrator. Other advantages today's security systems provide allow the user to stay informed about their property while they are away. In today's world security systems can communicate and be controlled through the usage of a smart phone letting the user know all their doors are indeed locked or that no one is on inside the premises.

Although there are plenty of different security monitoring techniques out there the use of microphone arrays for the purposes of security monitoring can help revolutionize the security industry by either giving more options to the user/customer, or by increasing the performance of a current security monitoring method.   The use of audio as a surveillance method could either to aid a current system in recognizing the surrounding environment or be a stand-alone system. This work uses audio for surveillance, by mimicking some of the abilities we have as humans to determine the locations of different sound sources and identify those sources.

Improving surveillance applications is becoming increasingly important in both the private and public sectors, but it is important to understand that there is no perfect system. Current surveillance applications can feel intrusive to the owner, while other surveillance applications just cannot collect enough information about the surrounding environment to truly fit the needs of the user. For example motion detector based systems do not feedback any information about the environment, except the fact that there is movement in the vicinity of the detector. [1] There are motion detector systems that can be used in the homes of people who have pets if their pets are small, but most of these systems limit the user to one pet. By having more than one pet or a very large pet they can be make themselves susceptible to a higher false alarm rate. This leaves motion detector based systems at a disadvantage, at least on their own, for people who have house animals, and expect there to be some movement in the home while they are away. Camera based systems can also have their disadvantages. [1] For example the placement of the cameras can possibly leave blind spots that could leave the user more vulnerable.

Surveillance systems often bring up the issue of privacy. The use of cameras as a security monitoring method might be socially acceptable in some locations, but in places such as a bedroom or bathroom cameras might not be considered socially acceptable. In [1] the issue of privacy and what is considered a socially acceptable security monitoring method inside a bedroom or bathroom is discussed. In this work they opt to use microphones instead of cameras or motion detectors in order to monitor the activities of daily living for the elderly. Motion sensors could only acquire a limited range of information pertained to movement within the room, and cameras are considered socially unacceptable in the given environment. This shows that in order to appeal to the general community multiple security monitoring methods are necessary, and that microphone array based security monitoring does have its place in today's security market.

Microphone arrays could be used in order to sound an alarm if a burglar is present, or ignore sounds that naturally occur in the surrounding environment. This feature is useful since most users would want to set up their system in an environment where a very diverse range of sounds could naturally occur. For instance if the system were to be set up inside of a home the user may have a pet, and it would be undesirable for the system to sound an alarm because of sounds the pet made. There also might be a fan or an A/C unit that is loud enough to trigger the microphone array. In most cases these types of sounds should not trigger the alarm, and need to be categorized as a naturally occurring sound.

A security monitoring method that utilizes microphone arrays is proposed in [2] that can detect, and identify a single sound source. The method proposed in [2] divides an audio segment into frames, and estimates the pitch within each frame in order to determine if that frame is speech or non-speech. In the case that a frame is considered non-speech, further classification uses a time delay neural network as a classifier.

This thesis will cover the usage of three different state-of-the-art techniques that are used to make up this security monitoring system. The first technique is a localization algorithm that allows the system to determine a direction of arrival of a specific sound source or of multiple sound sources. The next technique utilizes the Frost Beamforming algorithm which allows the system to extract the sound from a specific sound source in the presence of other interfering sound sources. The final technique covered will be an audio recognition algorithm used to classify the sound sources in the presence of the microphone array. Each chapter will have sections covering these three topics. Chapter 2 will consist of a review of the principles of microphone array based security monitoring. Chapter 3 will contain the methodology used to complete the system, and Chapters 4 and 5 will consist of the results and conclusion respectively.

1.1 Sound Source Localization

In this work a uniform circular microphone array is used in order to perform sound source localization. In [3] the multiple sound source localization problem is solved by performing a single source localization technique on selected received time frequency (TF) points. The TF points are selected based off a power ratio based selection process that determines the active/inactive points within the received signal at each microphone. In a reverberant environment this allows for the selection of TF points that are more likely related only to the direct path between the microphone array and the sound source.

1.2 Beamforming

Beamforming is a spatial filtering technique that in this case is used to control the directionality of the received signal at the microphone array. This is a useful tool since it allows for the reception of a single sound source in a given look direction, while reducing the signal power from interfering sound sources. In this work the frost beamformer is used due to its ability to reduce the power from interfering sound sources by continuously learning the statistics of the noise arriving from directions other than the assigned look direction. Frost's Beamformer is made up of an array containing M sensors, and J filter taps per sensor. Each filter tap has a weight that is updated using the Constrained Least Mean Square Algorithm, in order to minimize the power of the output signal while satisfying a constraint.

1.3 Audio Recognition

In order to perform the audio recognition it is important to look into the extraction of different audio features in order to keep the feature vector size as small as possible, while still maintaining a highly precise algorithm. In this work we explore the extraction of Mel-Frequency Cepstral Coefficients, Zero Crossing Rates, Spectral Flatness, and Short Time Energy as features. A Gaussian Mixture Model (GMM) is used as a classifier for its unique ability to form sharp approximations to arbitrary distributions.

CHAPTER II

PRINCIPLES OF AUDIO BASED SECURITY MONITORING

2.1 Audio Surveillance

This work attempts to cover the use of audio as a viable surveillance method. Surveillance systems of all types attempt to give the user peace of mind, allowing them extra protection against burglars, and unwanted intrusions. The use of audio as a surveillance method is not a new topic. In [4] microphones are placed throughout a room and a subspace based signal enhancement method is used by applying the Karhunen-Loeve transform (KLT). Audio recognition is later performed by extracting Independent Component Analysis (ICA) transformed Mel-Frequency Cepstral Coefficients (MFCCs), while using the Hidden Markov Model (HMM) as a classifier. The complication this method presents is that practically multiple sound sources can be present at the same time. There can also be sounds that are considered acceptable from specific locations that might not be acceptable coming from another location. A more detailed audio surveillance method needs to be explored that can determine the locations of multiple sound sources, and classify them.

Another audio surveillance method proposed in [2] attempts to solve the localization problem along with the classification problem. A sound source in [2] is localized, and a technique called beamforming is used in order to enhance the audio signal from the sound source. The sound source is then classified using a Time Delay Neural Network (TDNN), along with Mel-Frequency Cepstral Coefficients (MFCCs) Power Spectrum (PS), and the Magnitude Distribution (MD) as extracted features. The work outlined in [2] can handle a single sound source, but another method still needs to be proposed for handling multiple sound sources.

2.2 Audio Recognition

The primary focus of an audio based surveillance system is the ability to properly identify different sounds. This is generally done through the use of some audio recognition algorithm. Some of these algorithms are more tailored for speech applications, while others are more tailored to handle non-speech sounds. This work primarily focuses on recognizing and identifying non-speech sounds. In order to recognize an acoustic signal certain features from that signal first need to be extracted. A number of features considered in the literature can be categorized as Temporal Features, Perceptual Features, Energy Features, or Spectral Features. [5] A list of the different types of features are shown below. The extracted features are generally used to make up a Gaussian Mixture Model (GMM). The GMM is then used as a classifier for its unique abilities to form sharp approximations to arbitrarily shaped densities.

- **Temporal Features:** Zero-Crossing Rate (ZCR)
- **Perceptual Features:** Mel-Frequency Cepstral Coefficients (MFCCs)
- **Energy Features:** Short Time Energy (STE)
- **Spectral Features:** Spectral Flatness Measure (SFM)

Temporal Features such as ZCRs have been utilized for both speech and audio recognition. The work performed in [6] attempts to use ZCRs and STEs as a feature for determining voice and unvoiced portions of a signal. If the ZCR is high and the STE is low for a given audio frame, it is determined that the frame contains an unvoiced portion of the signal, while if the ZCR is low and the STE is high, it is considered a voiced portion of the signal. Using ZCRs and STE to determine a voiced/unvoiced portion of an audio signal based off a threshold is useful if speech is the known incoming audio signal. This could decrease the amount of processing for speech recognition. However, as a way of detecting speech versus other sounds this method is not sufficient.

In [7] an interesting approach is used in order to explore the use of an audio based surveillance system for both scream and gunshot detection. The work performed in [7] describes a system that has the ability to automatically detect audio events in a public place such as screams or gunshots. The system utilized the GMM as a classifier, and the classifier was trained using ZCRs, spectral moments, spectral flatness, MFCCs, spectral slope, spectral decrease, spectral roll-off, and correlation based features. The precision and the false rejection rate are used in order to optimize the size of the final feature vector. The problem with this method is that the results show the optimum feature set being quite large which can increase the computational cost, making it unsuitable for as a real time system.

2.3 Audio Localization

Although audio recognition for most is probably considered the most important aspect of an audio based surveillance system, having the ability to determine the locations of a single or multiple sound sources can be equally important. In order to do this a microphone array is required. In [7] a localization algorithm is presented that utilizes a T-shaped array composed of 4 microphones, where the center microphone is considered the reference sensor. The Generalized Cross Correlation (GCC) method is used for estimating the time delays between the reference microphone and other microphones. The minimization of the spherical error function [8] is used in order to estimate the source location. This makes this localization algorithm different from most since it does not require the *far field* hypothesis.

The objective of this work is to eventually be able to place the microphone array based security system inside of a reflective/reverberant environment. In many outdoor situations reflections are not really an issue but inside of an indoor environment there can be multiple paths the signal can take, that can cause false DOA estimates. The two localization algorithms outlined in [9], and [3] have been tested inside of a reflective/reverberant environment. Both [9] and [3] have introduced modern methods used for calculating Direction of Arrivals (DOA) from one or multiple sound sources to a circular microphone array. These techniques for sound source localization both assume the far field model, and both search for TF points where there is one dominant source. Those TF points along with the single sound source localization algorithm outlined in [10] are used to estimate a DOA. The main difference between these two methods come from how they select the proper TF points to be used for DOA estimation.

In [9], single source TF zones are calculated through a correlation coefficient method, and the TF points with the largest power are selected for DOA estimation. The problem with this method is that it is hard to assume the sparseness of the received signals in the TF domain inside of a reverberant environment. TF points from the direct path can be highly correlated with TF points coming from a reflected path leading to the selection of bad TF points. In [3] it is assumed that the direct path from a sound source to the microphone array is the shortest path. This leads to the detection of the active/inactive portions of the incoming signals for DOA estimation. In [3], the active/inactive portions of the incoming signals are determined from the power ratio between adjacent TF points in time. The inactive portions of the incoming signal should have a lower power while the active portions should have a larger power. The idea is that boundary points are more likely to be sparse TF points.

2.4 Beamforming Based Audio Separation

Most environments that the microphone array could be placed in would be considered noisy environments where the desired signal is distorted by one or many competing interfering signals. To handle this issue there is a need to properly separate source signals from interfering signals before performing audio recognition. Beamforming is a type of spatial filtering, which filters out interfering signals, and focuses on signals coming from a specific look direction. In [11] it is shown that most beamformers can be categorized as either a deterministic beamformer, or a statistically optimum beamformer. The deterministic beamformer does not depend on the incoming microphone signal, and has a set desired response. The statistically optimum beamformer is designed based on the properties of the incoming signal or interfering signals. This category of beamformer usually optimizes some function that makes the beamformer optimum for a specific purpose.

The simplest type of beamformer is the Delay and Sum beamformer. This beamformer fits into the deterministic class of beamformer. Figure 1 shows a Delay and Sum Beamformer with M microphones/sensors, each having a stage. Each signal is delayed based off of the time difference between each microphone and reference at a specified look angle. The goal is that the portion of the signal that comes from the specified look angle will add up constructively while the noise and interference will not.

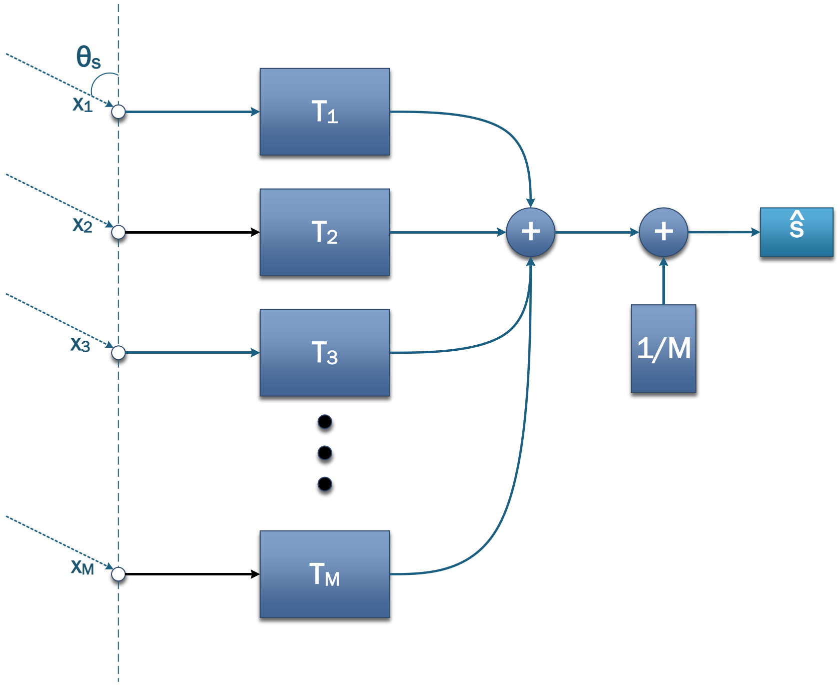


Figure 1: Delay and Sum Beamformer

There are some issues with this method. The Delay and Sum Beamformer does not perform well in a reverberant environment, and does not perform well in the presence of interfering sound sources. This method works best if there is a single sound source and the goal is to increase the signal to interference ratio of the received signal from a specific look angle, and just decrease the noise power. Other methods must be explored in order to handle the presence of interfering sound sources.

The Frost Beamformer outlined in [12] falls into the category of a statistically optimum beamformer for its abilities to adapt based on the statistics of an incoming signal. This beamformer utilizes the Constrained Least Mean Squares algorithm in order to minimize the total output power while fix the power in a specific look-direction. The beamformer has M sensors and J filter taps for each sensor. (See Figure 32 in Ch. III)

The General Side Lobe Canceller (GSC) outlined in [13] can be described as an alternative approach to Frost's adaptive beamforming algorithm outlined in [12]. The GSC is formed by a fixed beamformer, a blocking matrix, and an adaptive noise canceller. This method works well with narrowband signals, such as those encountered in communications, or radar based applications. The reason for this lies with the design of the fixed beamformer used with the GSC. The design of this fixed beamformer in typical applications utilizes filter weights that are chosen based on the array beamwidth, and average sidelobe level [13] [14]. This can be problematic when dealing with an acoustic signal since the beamwidth and sidelobe levels are frequency dependent.

2.5 Circular vs. Linear Microphone Array Geometry

Beamforming is a technique that has been around for some time. For instance the frost beamformer used in this work was originally proposed back in 1972 by Otis Lamont Frost. Acoustic beamforming does present some unique challenges that might not be as prevalent when dealing with narrowband signals such as in the communications sense. The array geometry plays a vital role that relates to the system's ability to properly extract a source signal from a given look direction. In [15] both the linear and circular microphone array topologies are compared, by their respective beam patterns and by their overall performance after extracting audio from some sound source. The beamformer used in [15] is a simple delay and sum beamformer used for a single sound source, but the use of a more complex adaptive beamformer could have increased their performance. It is shown that the beam patterns of the linear array tend to produce some ambiguity lobes which can decrease the system’s ability to properly extract sounds within a specific frequency range. The circular microphone topology often presents one main lobe with several side lobes. Both topologies though tend to have worse performance when extracting sources that consist of primarily low frequencies.

CHAPTER III

SURVEILLANCE SYSTEM DESIGN

3.1 System Overview:

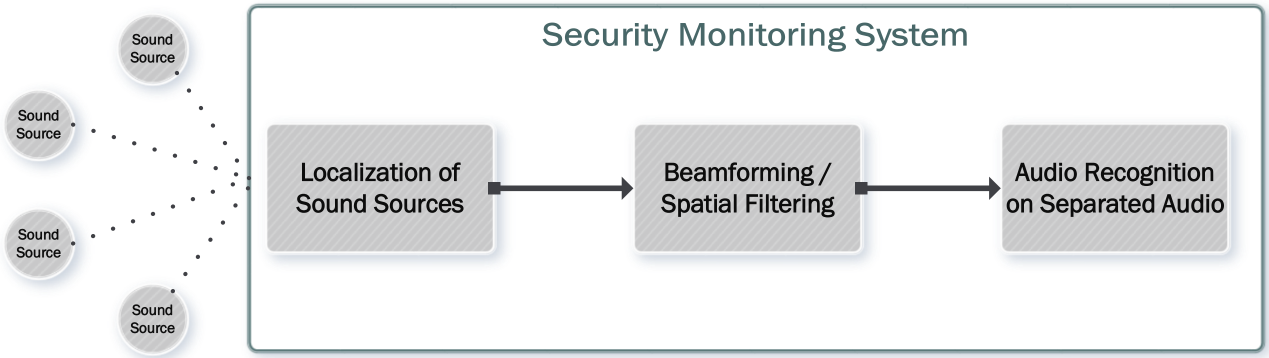

Security Monitoring System
Sound Source
Sound Source
Sound Source
Sound Source
Localization of Sound Sources
Beamforming / Spatial Filtering
Audio Recognition on Separated Audio

Figure 2: Security Monitoring Top Level Block Diagram

The Security Monitoring System that is being proposed consists of three major blocks. The Localization block determines DOAs to all the sound sources, and that DOA is used in order to open up a Beamformer for each look direction. The output of the beamformer at a specific look direction theoretically should contain only the portion of the signal coming from the observed sound source. The beamformer output(s) are then sent through the audio recognition algorithm. Figure 2 above shows the Security Monitoring Top Level Diagram. This chapter will cover the details within each block of the proposed system.

3.2 Design of the Circular Array

In Chapter II the Linear vs. Circular microphone array geometries were reviewed. The frequency response of the microphone array can be different based on the array geometry and it is important to understand how the frequency response changes based on the design of the microphone array, and the frequency of the received signal. Since this research primarily focuses on acoustic signals, it is best to have an array design that can be representative over a broad frequency spectrum. The response of a receiving aperture, such as a microphone array is inherently directional due to the amount of signal seen by the array at various DOAs. The response of the microphone array based on the DOA and the frequency of the source signal is known as the directivity pattern [16]. Looking at the directivity pattern for different array geometries can aid in the decision of the type of array geometry to use. The Directivity pattern for both the Linear Array and the Circular Array are shown in Figure 4 and Figure 5. The Linear Array offers a more simplistic geometry than the Circular Array, but the linear array starts getting ambiguity lobes earlier than the Circular array as the received signal frequency is increased. Both of the compared array geometries are shown in Figure 3.

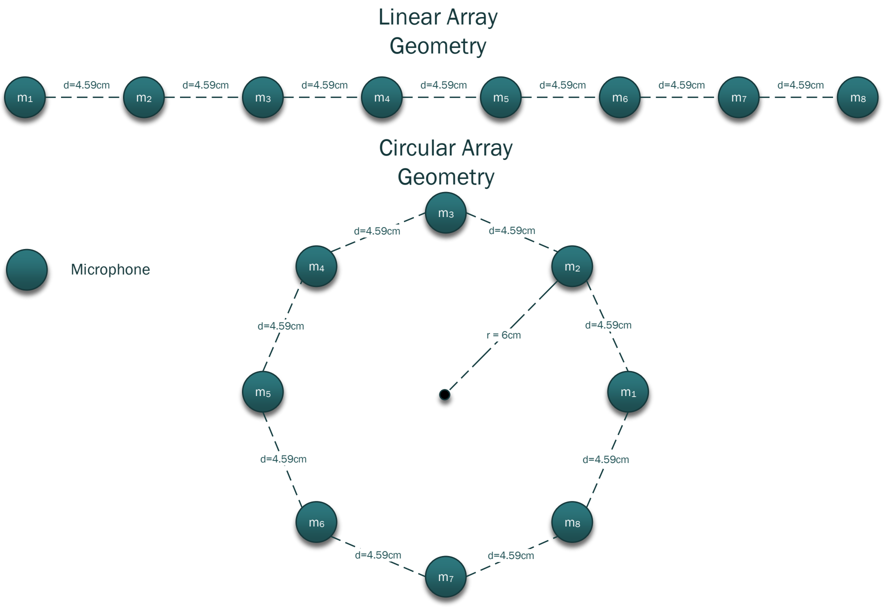


Figure 3: Linear vs. Circular Array Geometries

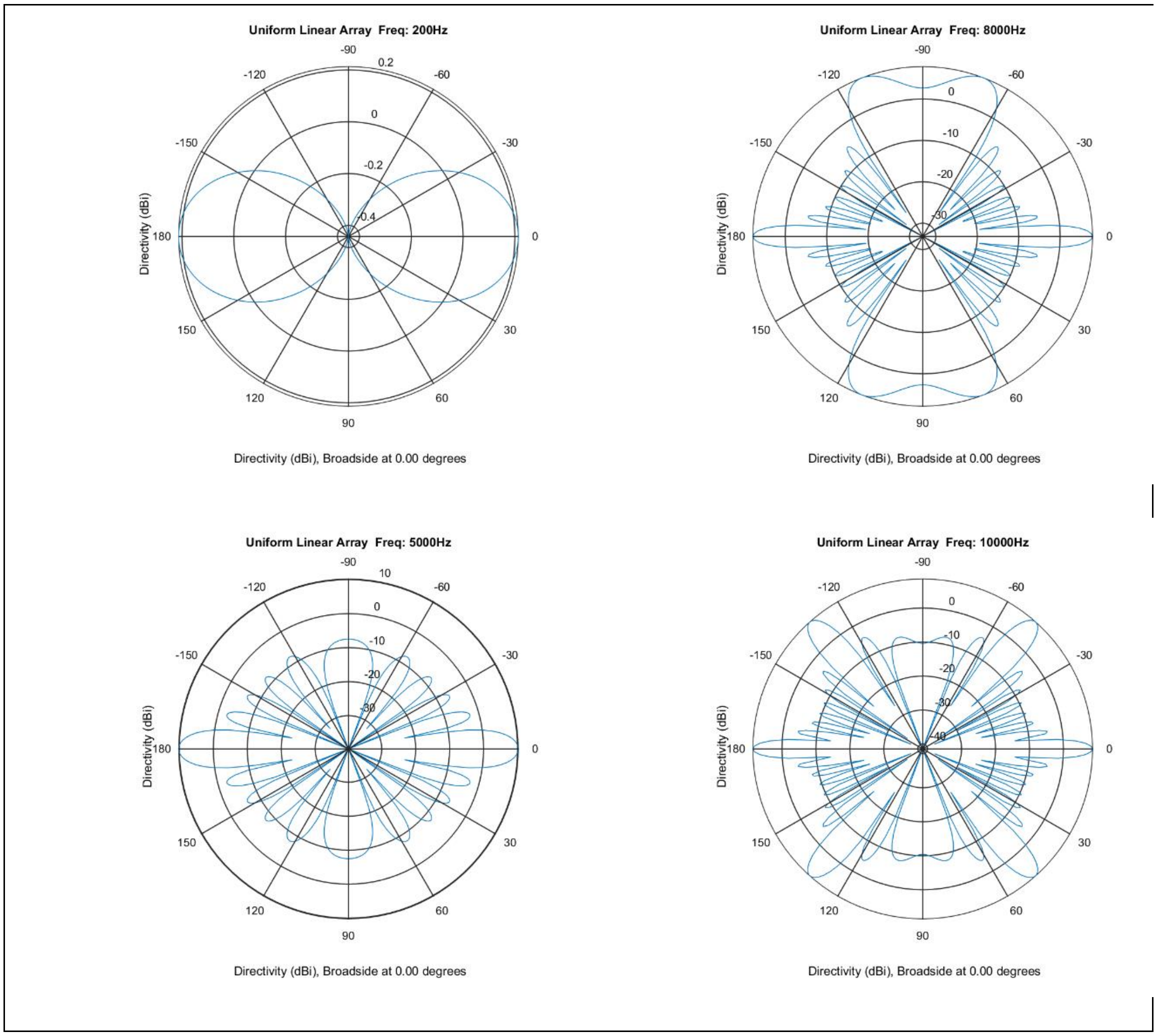


Figure 4: Uniform Linear Array Directivity Pattern @200Hz, 5000Hz, 8000Hz, and 10000Hz

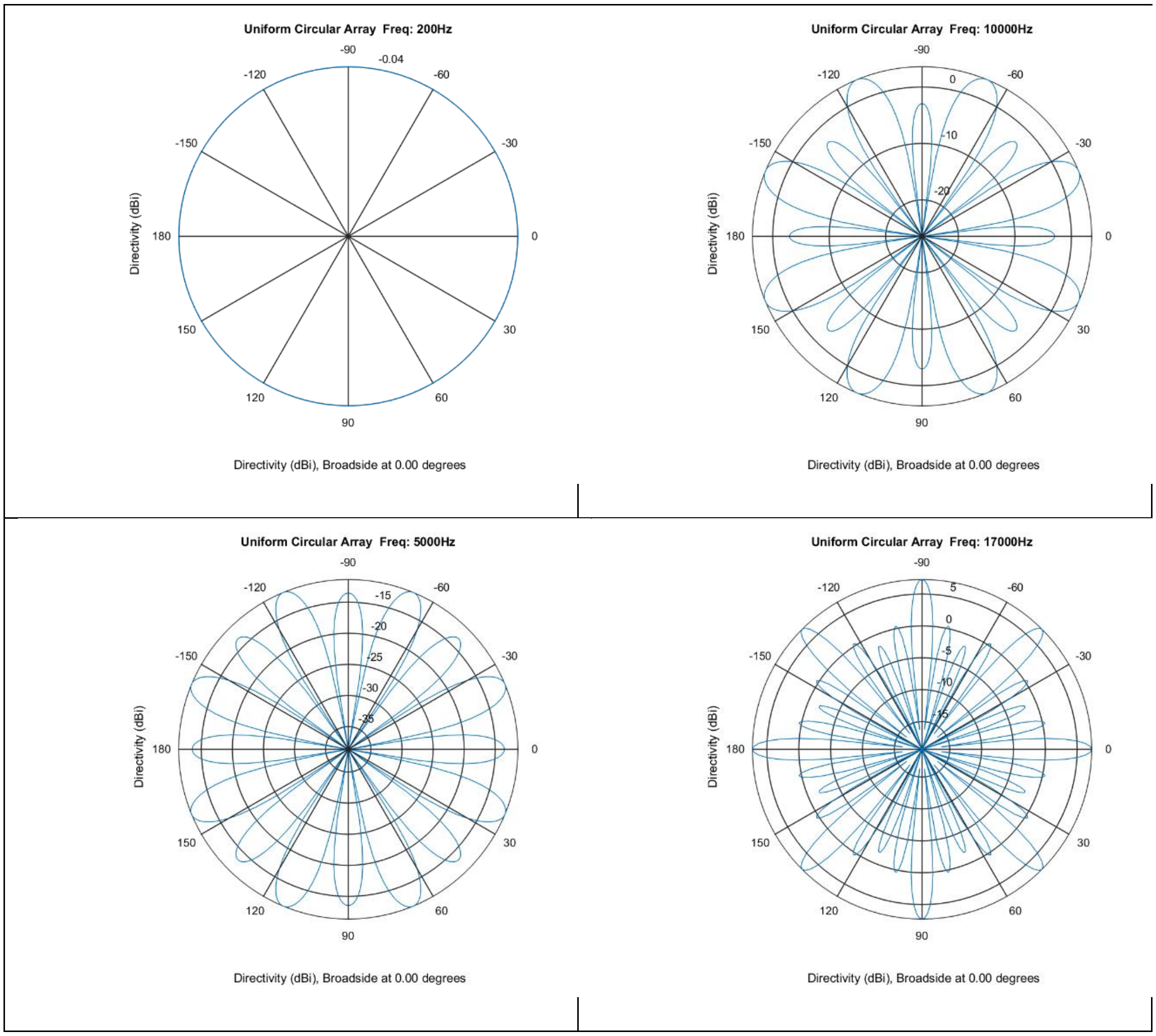


Figure 5: Uniform Circular Array Directivity Pattern @200Hz, 5000Hz, 10000Hz, and 17000Hz

3.3 Audio Recognition:

The block diagram shown in Figure 6 below describes the different stages of the audio recognition algorithm presented in this work. The audio from the microphone is buffered in 1s increments and the entire contents of the buffer are then sent into the audio recognition algorithm. The first stage is the framing stage. The audio here is broken up into different frames usually between 10ms-30ms for this kind of application. In this case each frame is 20ms. The next stage is the feature extraction state. In this stage features are extracted from the framed audio signals. This work explores the use of Mel-Frequency Cepstral Coefficients, Zero-Crossing Rates, Short Time Energy, and Spectral Flatness as possible features. All of these features were not used in the final implementation of the audio recognition algorithm, but they all were explored through the use of a feature selection algorithm, that determined the final feature vector.

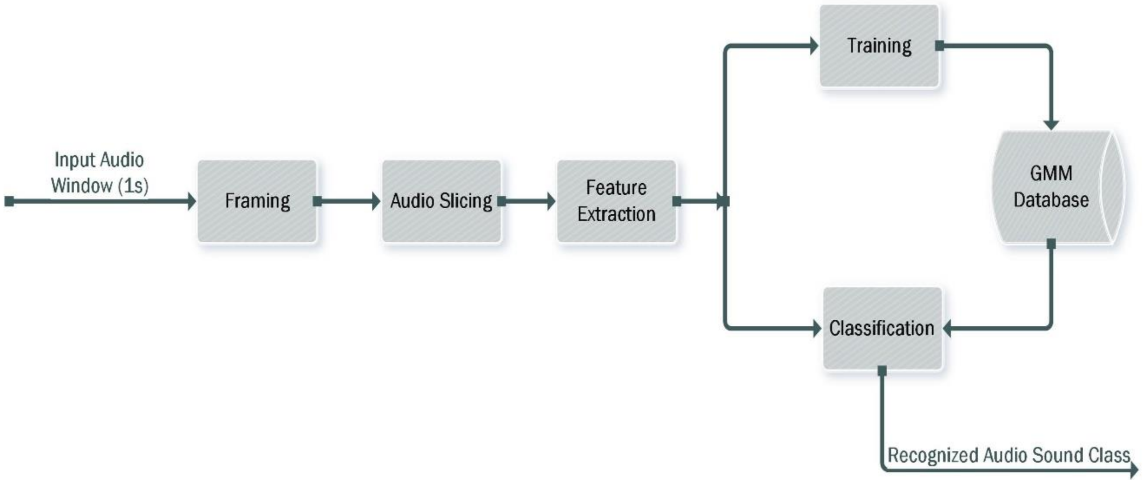


Figure 6: Audio Recognition Block Diagram

Before audio class recognition can occur, training must first be done for each class of expected sound. In this case examples of a sound class might be footsteps, or the sound of breaking glass. The training takes the extracted features, and uses that data to learn a Gaussian Mixture Model for each class. This information is then saved into a Database. During the Classification stage, (usually done during real-time use) the extracted features are compared with the different models from the database, and this comparison then leads to a decision/detection of an audio class.

3.3.1 Framing:

Framing is an important process, and is the first block for this audio recognition system. An incoming audio signal is cut into small frames usually ranging from 10-30ms. [17] The importance of this comes from the fact that we want to extract some features from an incoming audio signal while the signal is not stationary. If a frame size is too large, the signal just changes too much throughout the frame to obtain reliable features. If the frame size is too small, the signal just does not contain enough samples to obtain reliable information. After the signal has been broken up into frames different features can be extracted from each of the frames. Figure 7 shows a depiction of the framing process.

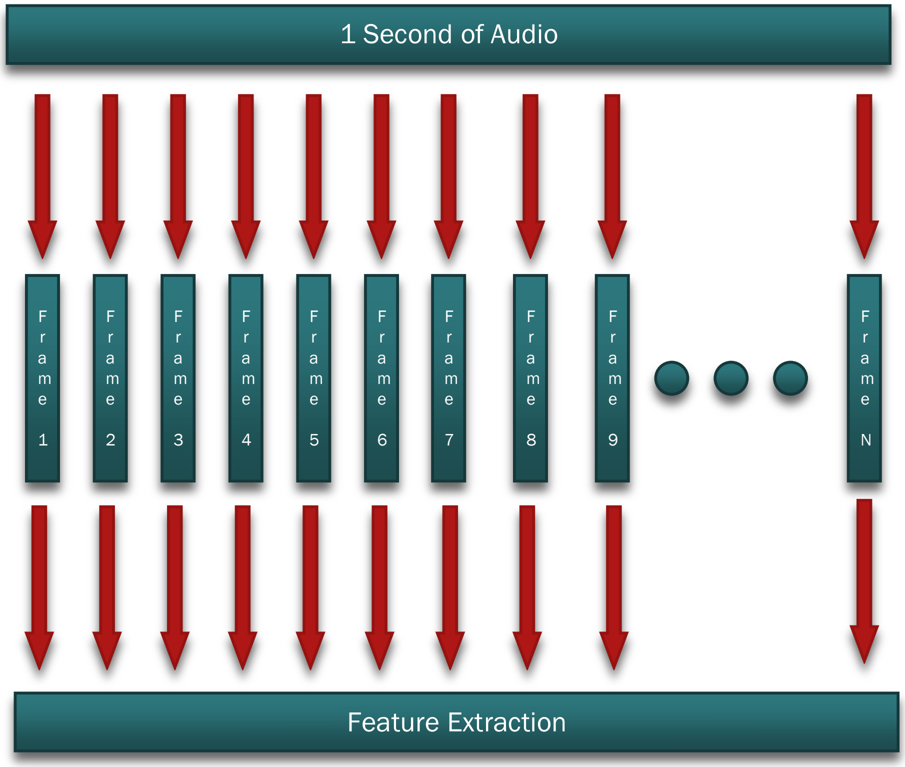


Figure 7: Depiction of Framing

3.3.2 Audio Slicing:

As we know in a room there is going to be some ambient noise that can affect the signal we are trying to recognize. Audio slicing attempts to take out the portions of the signal that are going to be most affected by the ambient noise from the room. Performing audio recognition from an input sound requires that different features be extracted from that input signal, but when the signal power drops into a range that can be highly affected by ambient noise the information that is extracted becomes less useful. Audio Slicing takes these portions of the signal out, while keeping the integrity or useful information from the signal intact.

The reason we can say that all the useful information from the signal is kept intact, is due to the way features are calculated. The audio signals are split into small frames, and features are calculated for each individual frame. Audio Slicing takes out all of the low power frames by creating a power threshold based on audio from the room, or the ambient noise in the room. Essentially this is a way of finding the noise floor for the room, and when the signal power drops near this level frames start getting deleted from the original input signal. Since frames are being deleted from the original signal, this process also reduced processing during the training and recognition sections.

$$P_x = \frac{1}{N}\sum_{n=1}^{N} |x(n)|^2 \qquad (1)$$

$$P_x(dBW) = 10 * log_{10}(P_x) \qquad (2)$$

$$Threshold(dBW) = median\big(P_x(dBW)\big) + abs\left(median\big(P_x(dBW)\big)\right) * \ \alpha, \qquad 0 < \alpha < 1 \qquad (3)$$

In order to obtain a good threshold the average power is taken from each individual frame using (1). These values are then converted to decibel watts using (2). The median power level in decibel watts is calculated from the ambient noise audio signal and used as seen in (3), where α is a threshold adjustment factor. The threshold adjustment factor is used if it is desired to increase the threshold by some percentage above the median power level.

Once the median power threshold has been obtained from an ambient noise audio sample, when obtaining either training audio, or audio to be recognized there is now a basis to determine which frames are to be deleted. The audio slicing is used on both the training, and recognition sections. An example of the audio slicing being used on a glass breaking training signal can be seen in the figures below.

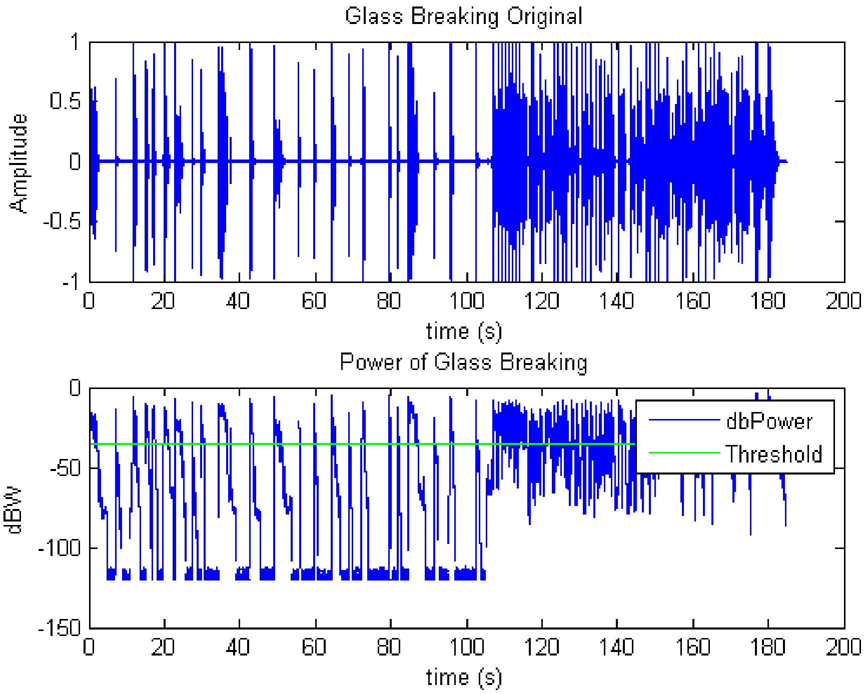


Figure 8: Original Glass Breaking

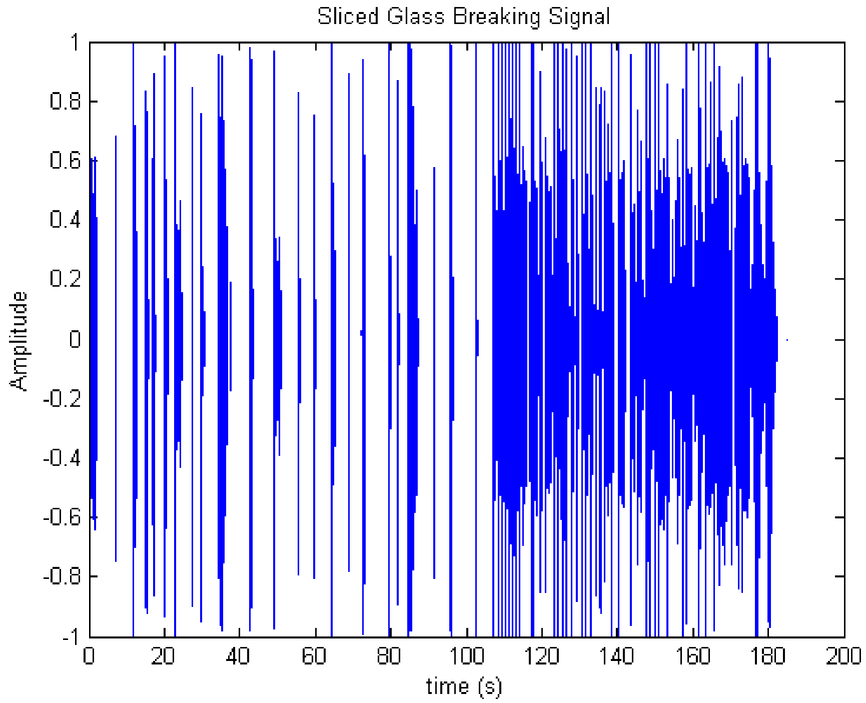


Figure 9: Sliced Glass Breaking

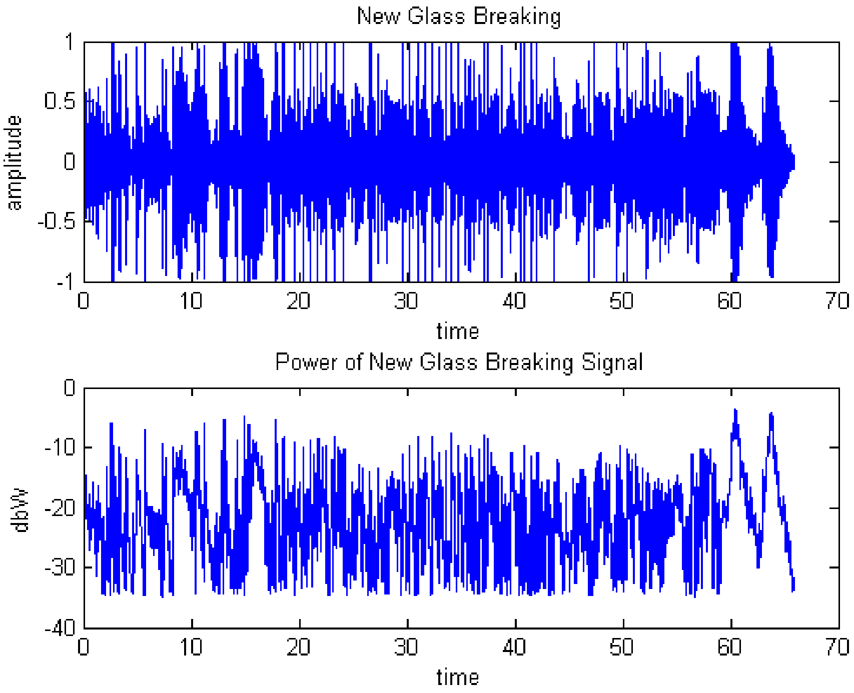


Figure 10: Glass Breaking after Slicing and Re-Combine

Figure 8 is an example of a glass breaking training signal before the audio slicing has been applied. It shows graphs of the signal and the average signal power in each frame. The green line shown is where the median power threshold was determined from (3). Figure 9 shows the sliced glass breaking signal with all of the deleted frames. From looking back at Figure 8 it can be seen that all the frames below the green threshold line were deleted leaving only the higher power portions, or the more useful portions of the signal intact. Figure 10 shows the new glass breaking training audio signal after the audio slicing and frames re-combined. This figure shows the average power taken in each frame to show that the average power never drops below the power threshold obtained from (3). It can also be seen that the glass breaking training signal length went from being around 180 seconds down to a little more than 60 seconds in this example.

3.3.3 Audio Features and Feature Selection

3.3.3.1 Mel-Frequency Cepstral Coefficients (MFCCs)

Mel-Frequency Cepstral Coefficients (MFCCs) are commonly used in audio recognition based applications, and since they were introduced by Davis and Mermelstein back in the 1980's they have been considered as one of the best features used for audio recognition. Some aspects of MFCCs are modeled in the same manner as human hearing, so in order to fully understand MFCCs it is important to have a basic understanding of the human ear. [18]

There is an organ in the human ear called the cochlea that transforms sound waves into electrical information the brain can recognize. Inside the cochlea there are around 20,000 to 30,000 reed-like fibers. These fibers each vibrate at different frequencies, so when a sound wave reaches the human ear different fibers vibrate based off of the frequency of the sound. [19] The vibration of these fibers are not linear with our interpretation of frequency in Hz. This is where the Mel-Scale becomes important, since the Mel-Scale was developed primarily to linearize our interpretation of frequency. Humans are better at determining small changes in pitch at lower frequencies than at higher frequencies. The Mel Scale was developed by testing human hearing, and their perception of pitches to be equal in distance from one another. Eq. (4) converts frequency in Hz to the mel-scale. Since the mel-scale was developed by testing the human auditory system, there is not one single equation used for converting frequency to the mel-scale, but eq. (4) is the most commonly used equation. Figure 11 shows a plot of the frequency in Hz versus the Mel-Scale based off of eq. (4). [20] Eq. (5) converts the frequency in Mels back to Hz.

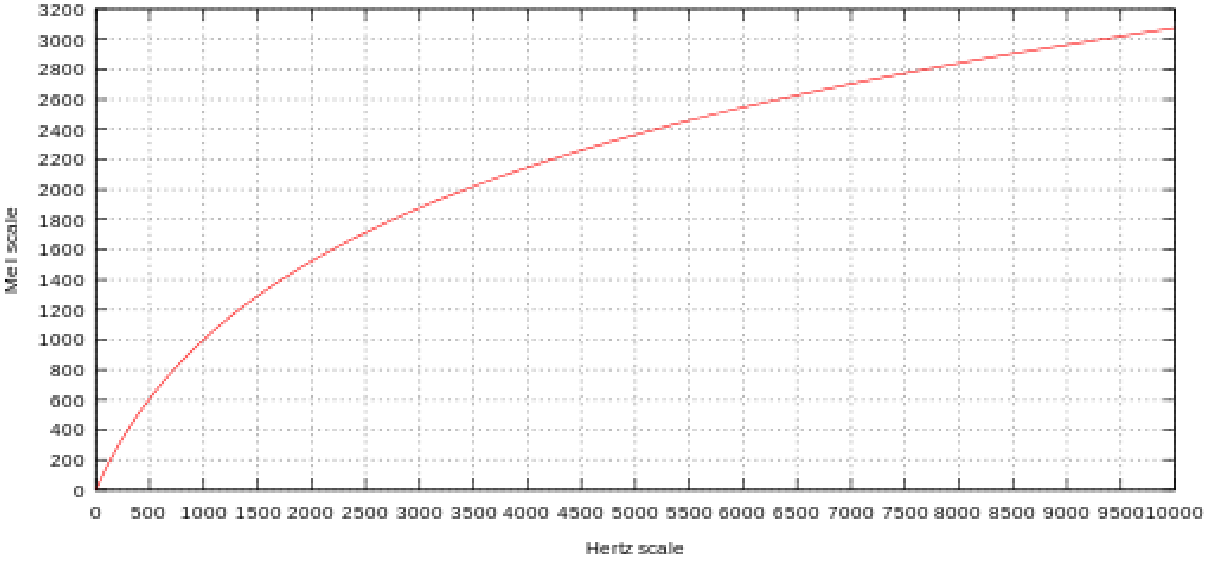


Figure 11: Mel Scale vs. Hertz Scale

$$mel = 2595 * log_{10}(1 + \frac{f}{700}) \qquad (4)$$

$$f(Hz) = 700 * (10^{\frac{mel}{2595}} - 1) \qquad (5)$$

Mel-Frequency Cepstral Coefficients (MFCCs) make up a Mel-Frequency Cepstrum (MFC). The MFC is the short term power spectrum based on a linear cosine transform of a log magnitude spectrum on a mel-scale of frequency. Figure 12 shows a flow chart describing how the MFCCs are calculated.

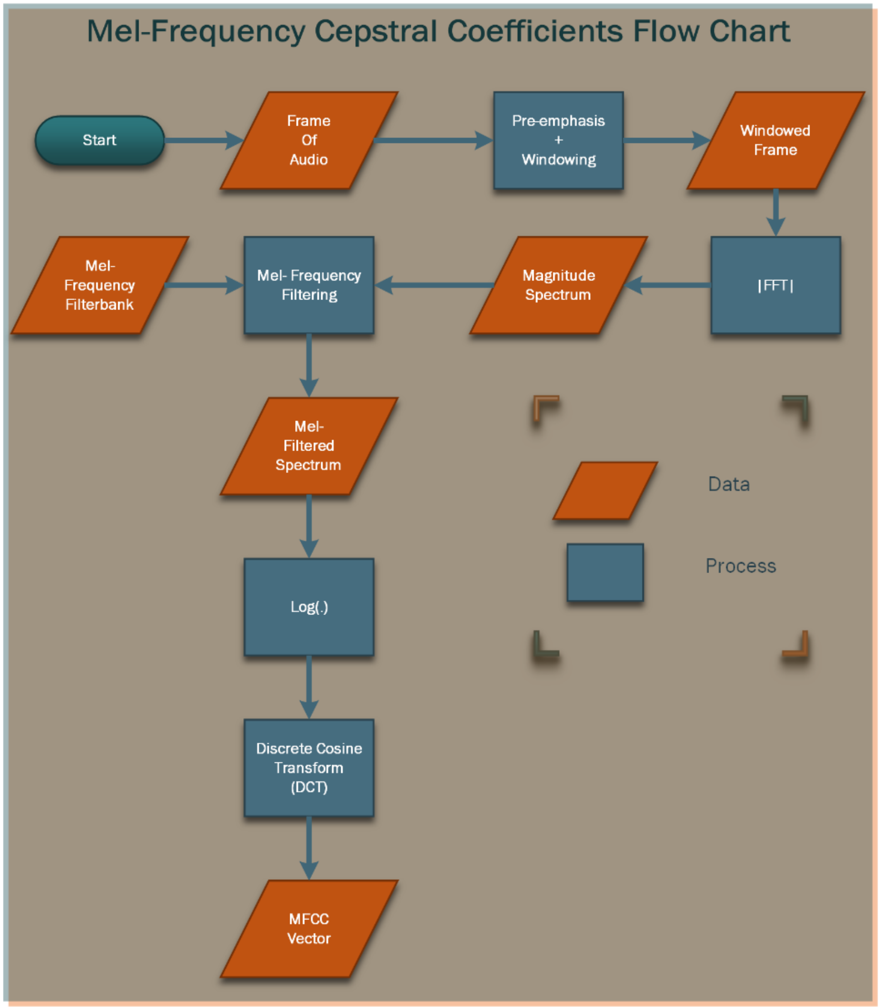


Figure 12: Mel-Frequency Cepstral Coefficients Flow Chart

Mel-Frequency Cepstral Coefficients are calculated for every audio frame. The signal is sent through a low pass filter in order to alleviate any noise outside of the human hearing range. The magnitude spectrum is calculated for each frame by taking the absolute value of the Fast Fourier Transform. The magnitude spectrum contains more information than can be used for the purpose of audio recognition, so instead of using the magnitude spectrum directly we want to see how much energy exists in various frequency regions. The way this is done is partially based on human hearing. We want to take the magnitude spectrum and multiply it by the Mel-Frequency Filter Bank. This filter bank is a group of triangularly shaped filters that are evenly spaced on the Mel-Scale. An example of the Mel-Frequency Filter Bank can be seen in Figure 13. As the frequency increases the filters start to get wider because they are evenly spaced on the mel-scale. If the scale is changed back to frequency in Hz that graph looks like that seen in Figure 13. Typical filter-bank parameters can be seen here in Table 1.

Typical Filter-bank Parameters	
Overlap	50%
Number of Filters in Mel bank	24-30
Number of Cepstral Coefficients to retain ≈	(# filters)/2

Table 1: Typical Filter-bank Parameters

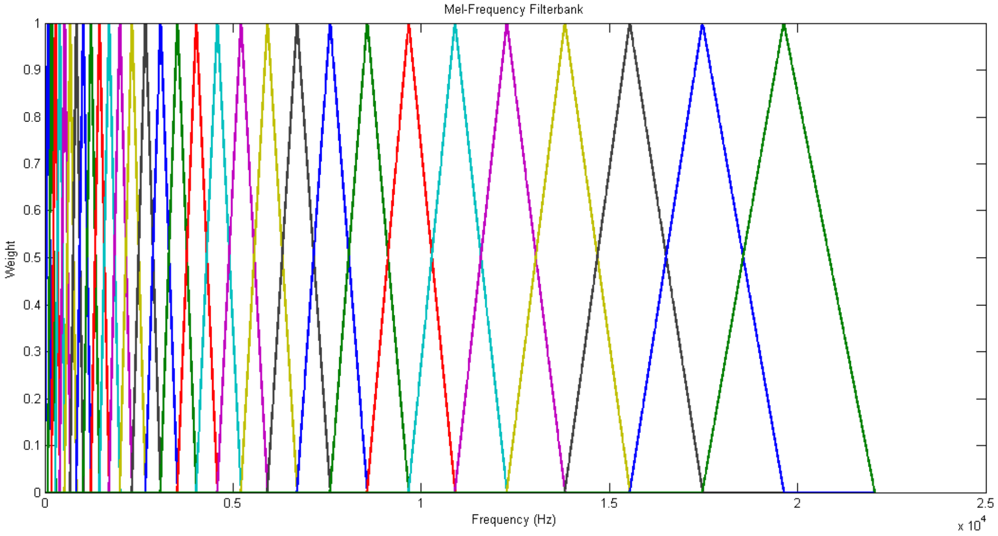


Figure 13: Mel-Frequency Filter-bank

Once all the energies have been calculated from all the various frequency regions, the log of these energies needs to be calculated. The reasoning behind taking the log of these energies is that we as humans do not perceive loudness on a linear scale. Taking the log of these energies makes our features more similar to what humans actually hear. Since the filter-banks overlap the filter-bank energies are strongly correlated. The discrete cosine transform de-correlates these energies. The equation for the discrete cosine transform can be seen in eq. (6). [21] The m_j represents the energy in the j^{th} filter-bank, N describes the number of filters, and c_i represents the coefficient.

After taking the DCT the first coefficient should be discarded because this coefficient is proportional to the distance from the microphone. Leaving the first coefficient will most likely yield unwanted results if sound sources are at different distances from the microphone. An example of MFCCs can be seen in Figure 14.

$$c_i = \sqrt{\frac{2}{N}} * \sum_{j=1}^{N} m_j * cos(\frac{\pi}{N}\left(j - \frac{1}{2}\right) * i), \qquad i = 1 \ldots\ldots N \qquad (6)$$

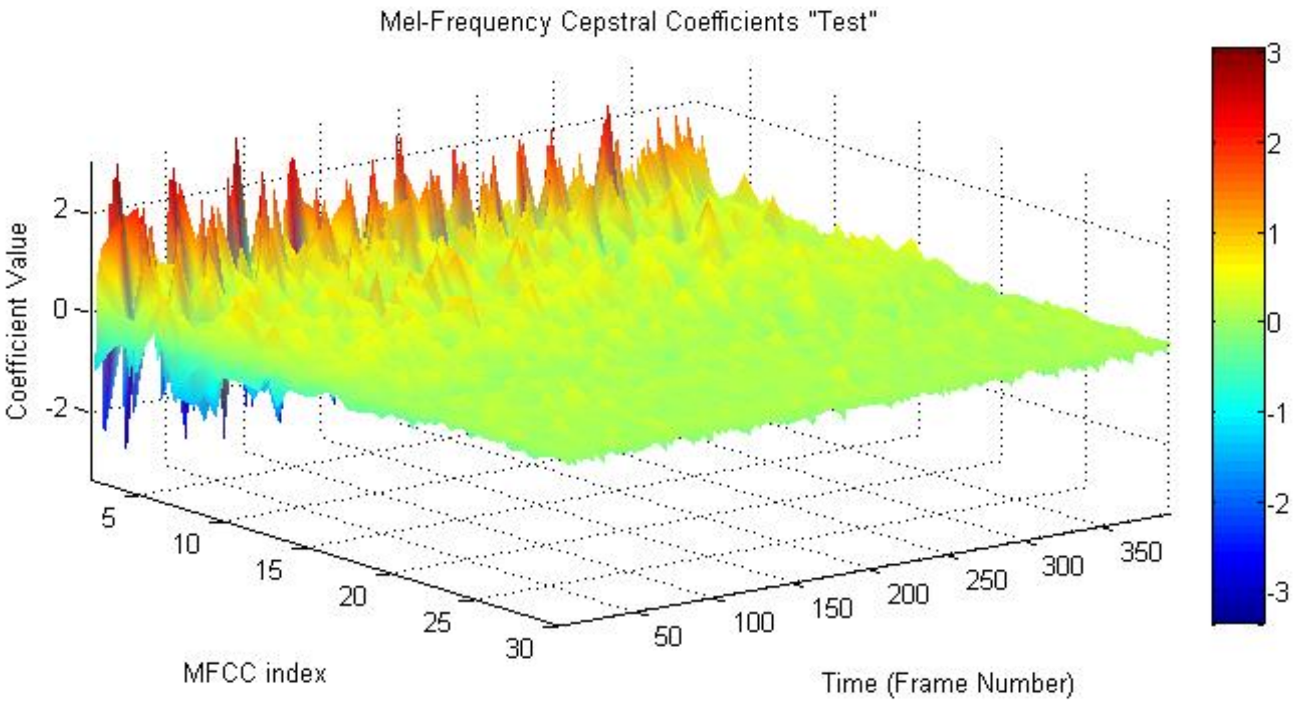


Figure 14: MFCCs

3.3.3.2 Zero Crossing Rates (ZCR) and Short Time Energy (STE)

Zero crossing Rates are often extracted for the purpose of Audio Recognition, and can be a key feature used for classifying sounds. They are a measure of the number of times that the algebraic sign of the signal changes, or the number of times the amplitude of the signal crosses zero.   An example of a zero crossing can be seen in Figure 15. The short time average zero crossing rate is defined in eq. (7). [6] The average of Z_n is taken within every frame, and those averages can be used for audio classification. An example of the Zero crossing rate for every frame can be seen for the sound of footsteps in Figure 16, and for the sound of a fan in Figure 17.

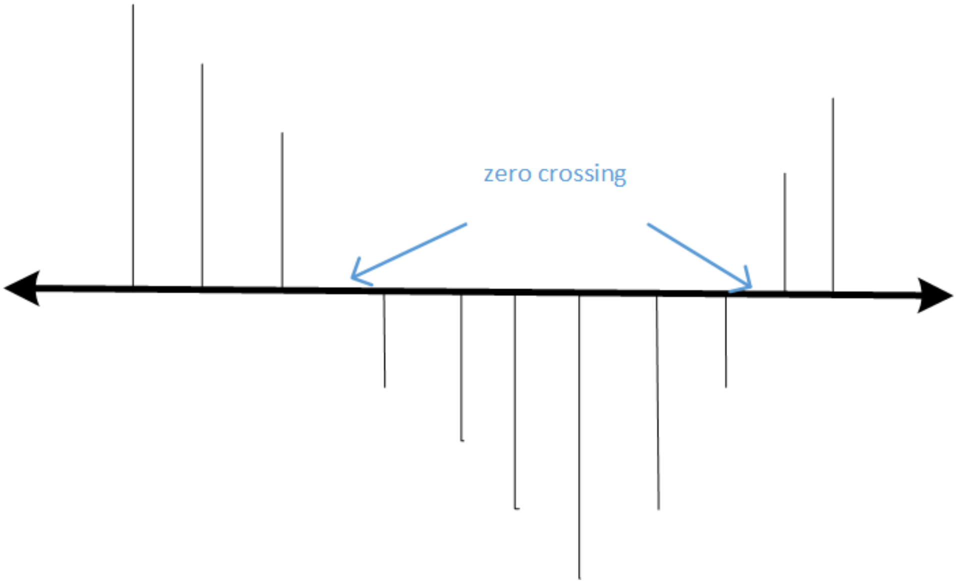


Figure 15: Zero Crossing

$$Z_n = \sum_{m=-\infty}^{\infty} |sgn[x(m)] - sgn[x(m-1)]|w(n-m) \tag{7}$$

$$sgn[x(n)] = \begin{cases} 1, & x(n) \geq 0 \\ -1, & x(n) < 0 \end{cases} \tag{8}$$

$$w(n) = \begin{cases} \dfrac{1}{2N}, & 0 \leq n \leq N-1 \\ 0, & otherwise \end{cases} \tag{9}$$

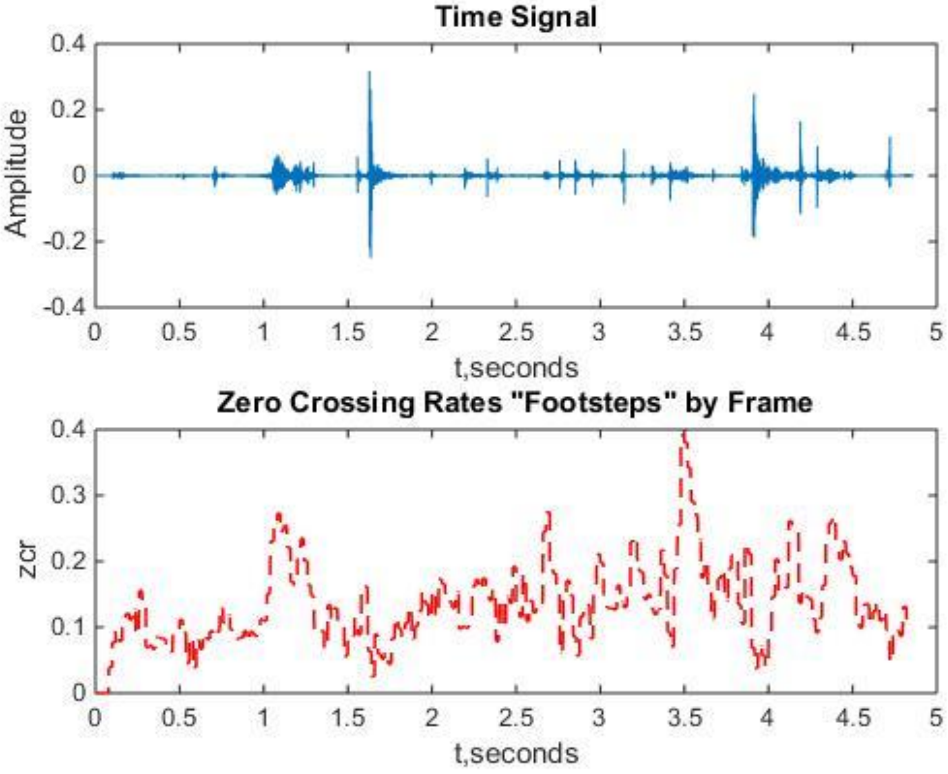


Figure 16: Zero Crossing Rates for Footsteps

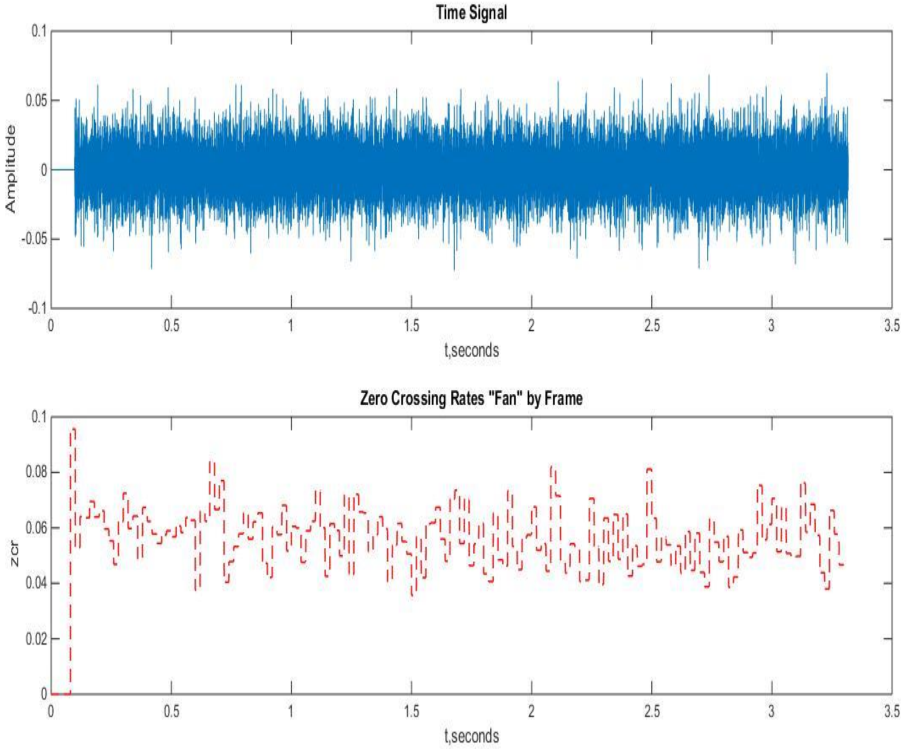


Figure 17: Zero Crossing Rates for a Fan

For speech processing Zero Crossing Rate along with Short Time Energy are commonly used to determine which frames of an audio signal contain speech. For our purpose we want to explore the use of Zero Crossing Rates and Short Time Energy as extracted features for audio recognition. Just as with the other extracted features the Short Time Energy needs to be calculated for each individual frame. The equation used to obtain the short-time energy can be seen in eq.(10) where eq. (9) represents the window used. [6] Figure 18 shows the short time energy for the sound keys.

$$E_n = \sum_{m=-\infty}^{\infty} [x(m)w(n-m)]^2 \tag{10}$$

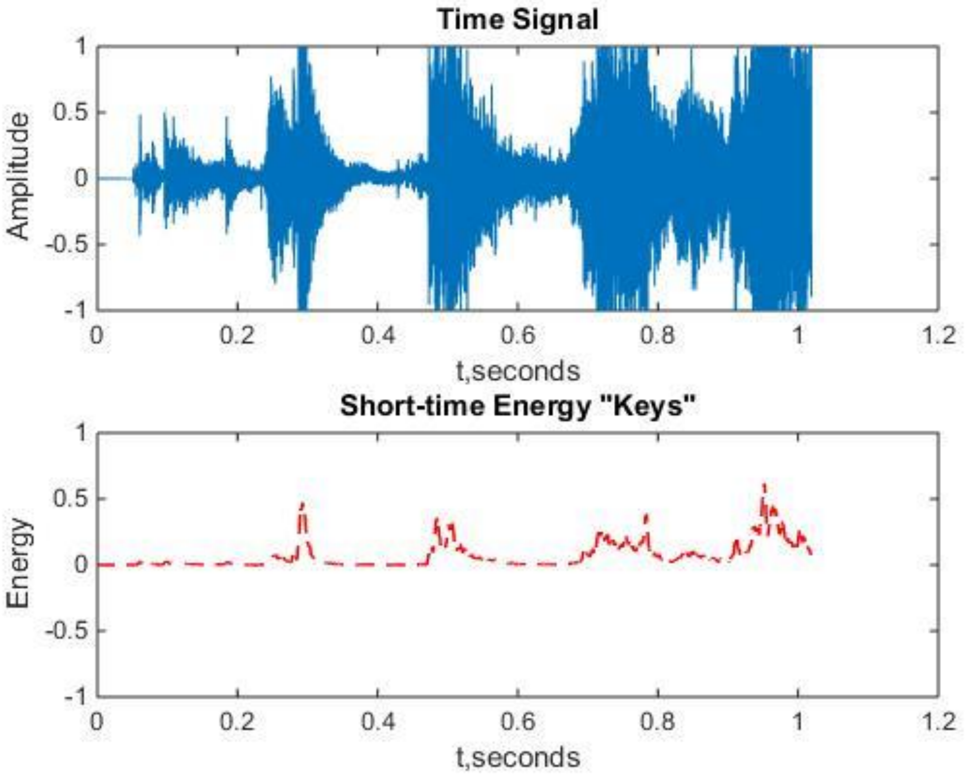


Figure 18: Short Time Energy of Keys

3.3.3.3 Spectral Flatness Measure (SFM)

Spectral Flatness is a useful feature in digital signal processing that is used to characterize the tonality of an audio signal. This means that the Spectral Flatness Measure (SMF) can be considered as a useful feature for audio recognition, because it can determine how tonal a signal is versus how noise-like. SFMs are determined from the power spectral density of an audio signal. Signals that have a similar amount of power in all spectral bands will produce a spectral flatness of approximately 1. As we know white noise should have a flat power spectral density so white noise should produce a spectral flatness of about 1, where a pure tone should produce a spectral flatness of approximately 0. Spectral Flatness is defined as the ratio of the geometric mean to the arithmetic mean, which can be seen in eq. (11). [22] Figure 19 shows the spectral flatness measure by frame for the sound of jingling keys. Figure 20 shows the spectral flatness measure for the same sound of jingling keys, with some added noise. By looking at Figure 19 and Figure 20 it can be seen that by adding noise, the spectral flatness measure was increased, indicating that the signal was more noise-like.

$$Flatness = \frac{\sqrt[N]{\prod_{n=0}^{N-1} x(n)}}{\frac{\sum_{n=0}^{N-1} x(n)}{N}} = \frac{exp\left(\frac{1}{N}\sum_{n=0}^{N-1} ln[x(n)]\right)}{\frac{1}{N}\sum_{n=0}^{N-1} x(n)} \qquad (\ 11\ )$$

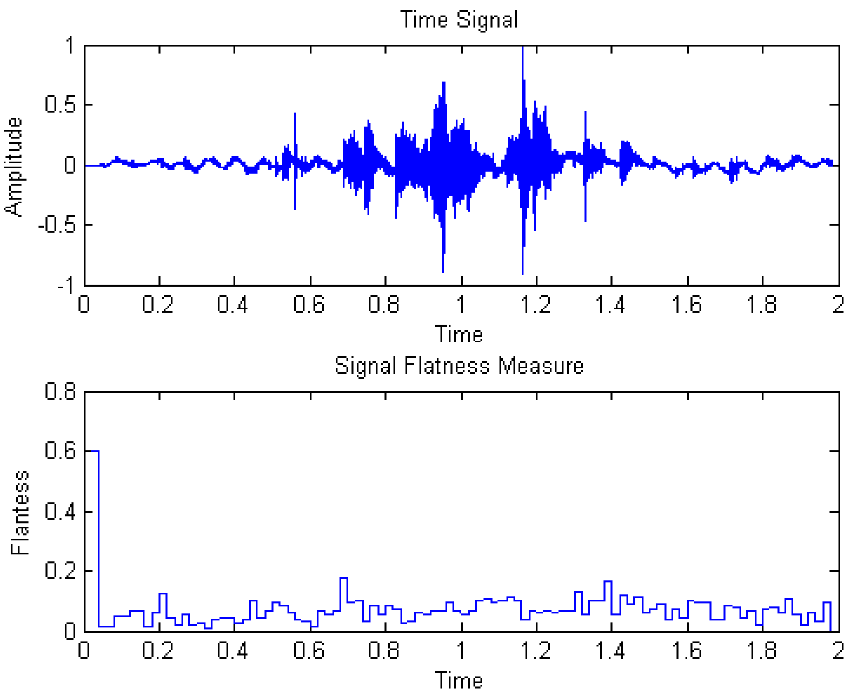


Figure 19: Spectral Flatness Measure of Keys

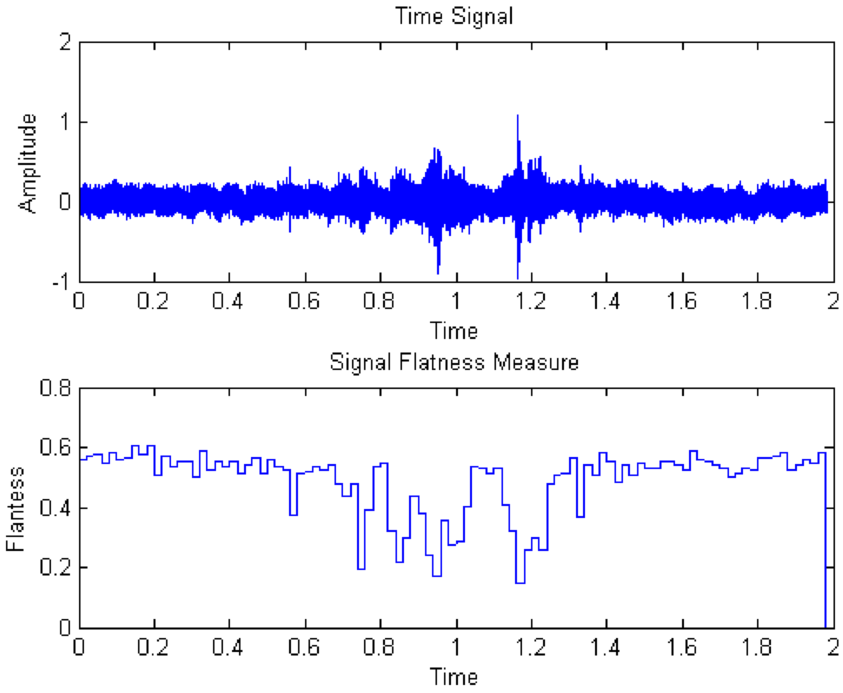


Figure 20: Spectral Flatness Measure of Keys with Added Noise

3.3.3.4 Feature Selection:

Feature selection algorithms are widely used in order to improve the speed and accuracy of different systems. Increasing the feature vector size more than necessary can sometimes be computationally expensive. As the size of a feature vector increases, it becomes more likely that specific features will contain redundant information that can even hurt performance. Feature selection algorithms determine a subset of features from within a full set of features that can optimize performance. The feature selection algorithm proposed in this work attempts to decrease the number of features used for classification, while statistically maintaining the maximum performance. This is done by combining the Sequential Forward Selection (SFS) method which is the simplest greedy method, with hypothesis testing for the difference between proportions.

3.3.3.4.1 Sequential Forward Selection (SFS):

Sequential Forward Selection is often used for its simplicity and speed. The SFS algorithm is considered a bottom up search procedure, because it starts with an empty feature vector, or feature set, and continually adds features based on some objective function. In this work the objective function is based on the precision of the audio recognition algorithm. After each iteration of the SFS algorithm new features are selected from the remaining available features, or features that have not been chosen in the past. The new chosen feature to add should maximize the objective function within each iteration. [23]

For this algorithm we will assume that the set $\boldsymbol{T} = \{t_1, t_2, \ldots, t_N\}$ contains all the possible features, where N is the total number of possible features. Let the set $\boldsymbol{G_i}$ denote the selected features, and $J(\cdot)$ represents the objective function.

Steps for Sequential Forward Selection:

1. Start with empty set $\boldsymbol{G_0} = \{\phi\}, \quad i = 0$
2. Update $i = i + 1$
3. $k^* = \max[J(\{\boldsymbol{G_{i-1}}, T_k\})],\ where\ k = 1, \ldots, N; \quad T_k \notin \boldsymbol{G}; \quad \boldsymbol{G_i} = \{\boldsymbol{G}_{i-1}, T_{k^*}\}$
4. Repeat steps 2 & 3 until $i = N$

The SFS algorithm performs best when the total number of features is small. As the feature size gets very large the area covered or the number of combinations analyzed becomes narrower. This is due to the fact that if the SFS algorithm is close to the full set, most features have already been chosen. When using the original SFS algorithm there is another step that decides the final feature set to be the $\max[J(\boldsymbol{G})]$, which would make the final feature set equal the set that had the largest precision from $\boldsymbol{G}$. In this work the goal is to reduce the number of features to be as small as possible, so instead of choosing the final feature set to be the feature set with the highest precision, we perform a hypothesis test for the difference between proportions to determine statistical significance between proportions.

3.3.3.4.2 Hypothesis testing for difference between proportions:

Hypothesis testing allows us to make some conclusion about a hypothesis. In the previous section the objective function is based on precision values for each set of features. In this case we want to test whether or not there is a significant difference between two proportions, or whether or not there is a significant difference between two precision values for two sets of feature vectors. A significant difference determines whether or not we can attribute the difference seen to some random variations, or to the difference in the feature vector. For hypothesis testing a null hypothesis, and an alternative hypothesis must clearly be defined. The null hypothesis usually represents the status quo, and is not rejected unless the results strongly imply that the null hypothesis is false. The alternative hypothesis contradicts the null hypothesis, and is accepted in the case that the null hypothesis is rejected. Eq. (12) shows the null and alternative hypotheses that are used in this work. It is important to understand that to prove the two proportions are statistically different, a two-tailed test must be performed at a chosen significance level. The significance level expresses how sure someone might be in the data. For instance a 1% significance level makes it much more unlikely that the alternative hypothesis will be accepted than a 5% significance level. The steps for performing a two tailed hypothesis test are shown below. [24]

$$\begin{gathered} H_0: p_1 - p_2 = 0 \\ H_a: p_1 - p_2 \neq 0 \end{gathered} \qquad (12)$$

(Null and Alternative Hypothesis)

Steps for Two Tailed Hypothesis Testing:

1. Define the null hypothesis *H_0* and alternative hypothesis *H_a*.
2. Assume that *H_0* is true.

3. Compute the test statistic.

4. Compute the z-score for a chosen significance level.

5. Determine if $statistic > zscore \; or \; statistic < \; -zscore$

6. If step 5 is true then the Null Hypothesis gets rejected and we accept the alternative hypothesis.

$$\hat{p}_1 = \frac{X_1}{n_1}, \quad \hat{p}_2 = \frac{X_2}{n_2}, \quad \hat{p} = \frac{n_1\hat{p}_1 + n_2\,\hat{p}_2}{n_1 + n_2} \tag{13}$$

(Sample Proportions)

$$statistic = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}(1-\hat{p})\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}} \tag{14}$$

(Test Statistic)

Eq. (13) and (14) are used to determine the value for the test statistic. The sample proportion $\hat{p}_1$ are determined by the number of successes X_1divided by the total number of samples n_1. The same is also done for the second sample proportion.

3.3.3.4.3 Combining SFS and Hypothesis Testing:

As the feature size increases it becomes more likely that the system will have redundant information that can even hurt performance. The Feature Selection algorithm proposed utilizes the SFS method, and hypothesis testing to determine significance. The Sequential Forward Search algorithm needs to be applied, and the set $\boldsymbol{G_i}$, needs to be determined for the chosen or selected features. The precision and the selected features for each $\boldsymbol{G_i}$ should be known. The feature subset $\boldsymbol{G_i}$ that has the maximum precision from the entire SFS process, determines the starting point for the hypothesis testing. This means that if the feature vector that has the maximum precision has 2 features, and feature vector that has more than two features can immediately be discarded.

Feature Selection Steps after SFS:

1. Find Max Precision and the corresponding feature subset $\boldsymbol{G_i}$.
2. Look for $\boldsymbol{G_j}$ s.t. $j < i$
3. If the feature subset with the maximum precision has just one feature choose, this feature subset.
4. When computing significant differences between precisions always compare set with one less feature at each iteration. (Unless there is only one feature in feature vector with max precision).

3.3.4 Gaussian Mixture Model (GMM)

GMMs have a unique ability to form sharp approximations to arbitrarily shaped densities, which allows an audio recognition systems to have the ability to train to different sound classes, based off of feature extracted training data. These Mixture Models are gaining increased attention in the audio recognition area for their use as a classifier. Generally for the purpose of audio recognition a GMM is made for each individual audio class, and this collection of GMMs makes up a database. This is a useful feature, because if new sound classes need to be added in the future, this can be done easily without having to re-train the entire database. A new model for the new sound class would just need to be added to the database. When new data needs to be classified, it can be compared to all the different GMMs in the database, and the model that has the highest likelihood of representing the data is chosen as the correct audio class.

The Gaussian Mixture Model is a probabilistic model that can be used to represent a subpopulation, from an overall population. It can generally be described as a summation of K, different Gaussian density components in order to represent some data $\boldsymbol{X}$, where $\boldsymbol{X} = \{x_1, x_2, \ldots, x_N\}$, with a set of N observations. The Gaussian densities can each be described by their different means μ, and variances σ^2. A one-dimensional example of a GMM with K=8 mixtures, fit to some data can be seen in Figure 21.

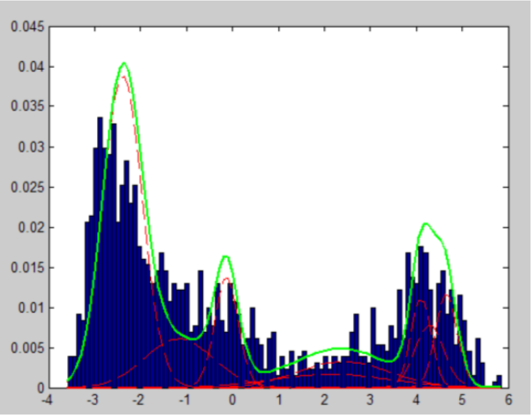


Figure 21: 1-Dim GMM Example *[25]*

3.3.4.1 1-Dimensional GMM Representation:

Since a Gaussian mixture model is made up of a summation of Gaussians, each Gaussian can be described by (15) where *j* represents each individual component.

$$g(x|\mu_j,\sigma_j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(x_i-\mu_j)^2}{2\sigma_j^2}}, \quad j = 1 \dots K \tag{15}$$

The model parameters μ_j and σ_j can be represented as θ_j, and θ_j needs to be determined for each Gaussian component. This is generally done by performing an Expectation Maximization Algorithm. The aim is to optimize the parameters of each Gaussian component to best fit the training data. The EM algorithm is an iterative process that constantly revaluates an E-Step or expectation step, and an M-Step, or maximization step, until the likelihood function is maximized. The EM Algorithm can be summarized into these simple steps [26]:

1. Initialize Parameters to starting Values.
2. Evaluate the likelihood values obtained from the initialized parameters.
3. Evaluate the Posterior Probabilities.
4. Use the Posterior Probabilities to estimate the new parameters and calculate the likelihood.

5. Re-evaluate the likelihood and repeat steps 3 and 4 until the likelihood converges.

The initial parameters can be chosen randomly. After the initial parameters have been selected the likelihood should be calculated, and used to get the posteriors *p_i*. The evaluation of the posterior probabilities is considered the E-Step. The posteriors are then used to estimate the new parameters (means and variances) from (20) and (21). The evaluation of the posteriors, the estimation of the new parameters, and the calculation of the new likelihoods are considered the M-Step. These two steps are repeated iteratively until the likelihood value meets some convergence threshold. Once the likelihood has converged and been maximized, the parameters for the new model are then finalized.

$$P(x_i|\theta_j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(x_i-\mu_j)^2}{2\sigma_j^2}}, \quad j = 1 \ldots . K \tag{16}$$

(Likelihood)

$$p_{i,j} = P(\theta_j|x_i) = \frac{P(x_i|\theta_j)P(\theta_j)}{P(x_i)} \tag{17}$$

(Posterior)

$$\alpha_j = \frac{\sum_{i=1}^{n} p_{i,j}}{n} \tag{18}$$

(Prior for component *j*)

$$P(x_i) = \sum_{j=1}^{K} P(x_i|\theta_j)P(\theta_j) \tag{19}$$

(Total Probability for $P(x_i)$)

$$\mu_j = \frac{\sum_{i=1}^{n} p_{i,j} x_i}{\sum_{i=1}^{n} p_{i,j}} \tag{20}$$

(Mean of component *j*)

$$\sigma_j^2 = \frac{\sum_{i=1}^{n} p_{i,j} * \left(x_i - \mu_j\right)^2}{\sum_{i=1}^{n} p_{i,j}} \qquad (21)$$

(Variance of Component *j*)

3.3.4.2 Multi-Dimensional GMM Representation:

Once the 1-dimensional GMM is understood, it is fairly simple to produce a multi-dimensional GMM using the same basic principles from the 1-dimensional case. First, all the equations from the 1-dimensional case need to be reconstructed in order to handle multiple dimensions. The process for maximizing the likelihood, or adjusting the parameters to best fit the data is done the same way as it was in the 1-dimensional case, but with the new equations seen in eq. (22) through eq. (25).

$$P\left(\theta_j \middle| \vec{x_\iota}\right) = \frac{P\left(\vec{x_\iota} \middle| \theta_j\right) P\left(\theta_j\right)}{\sum_{j=1}^{K} P\left(\vec{x_\iota} \middle| \theta_j^T\right) P\left(\theta_j^T\right)} \qquad (22)$$

(Posteriors)

$$\mu_{j,r} = \sum_{i=1}^{N} \frac{P\left(\theta_j \middle| \vec{x_\iota}\right)}{nP\left(\theta_j\right)} x_{i,r} \qquad (23)$$

(Mean of component *j*,*r*)

$$(\Sigma_\theta)_{r,k} = \sum_{i=1}^{N} \frac{P\left(\theta_j \middle| \vec{x_\iota}\right)}{nP\left(\theta_j\right)} (x_{i,r} - \mu_{j,r})(x_{i,k} - \mu_{j,k}) \qquad (24)$$

(Covariance Matrix)

$$P(\overrightarrow{x_i}|\theta) = \frac{1}{\sqrt{2\pi|\Sigma_\theta|}} exp\left(-\frac{1}{2}\sum_{r=1}^{d}\sum_{k=1}^{d}(x_{i,r} - \mu_{j,r})\,(\Sigma_\theta{}^{-1})(x_{i,k} - \mu_{j,k})\right) \qquad (25)$$

(Likelihood)

Because it can be more computationally difficult to determine the necessary parameters in multiple dimensions using the method above, it can be helpful to introduce the log-likelihood. Using the log-likelihood function can make the maximum likelihood estimation calculations analytically easier. The Log-likelihood function can be seen in (26). If we say that $L(\theta)$ are the parameters that are used to estimate the expectation, and $L(\theta^t)$ are the new parameters after each iteration $L(\theta^t)$ should be greater than $L(\theta^{(t-1)})$. Unlike some other clustering algorithms the GMM does not have a set point to stop iterating. A good stopping criteria that is most commonly used is a check to see if the Log-Likelihood has converged. Once it has converged the parameters are considered optimal.

$$L(O|X) = log\prod_{i=1}^{N} p(x_i|O) = \sum_{i=1}^{N} log(\sum_{j=1}^{K} \alpha_j g_j(x_i|\,\theta_j)) \qquad (26)$$

(Log-Likelihood)

3.3.4.3 Graphical Example of GMM:

During the training phase of the audio recognition system a different Gaussian Mixture Model is made for each type of audio class. This section examines an example of how this data can be clustered, and shows how the EM algorithm fit the parameters of the GMM to the training data presented. Figure 22 shows a 2-dimensional view of how MFCC1 and MFCC2 are clustered for door sounds, keys, and the natural sound of the room. From the figure it can be seen that for this example the data is separated into distinct clusters. This is a feature that the GMMs are good at exploiting. During the recognition portion of the audio recognition algorithm when new data comes into the system the goal is to recognize how the new data is clustered and make a comparison to the previous models created. This is why it is so important during training to accurately create a model that fits to the training data.

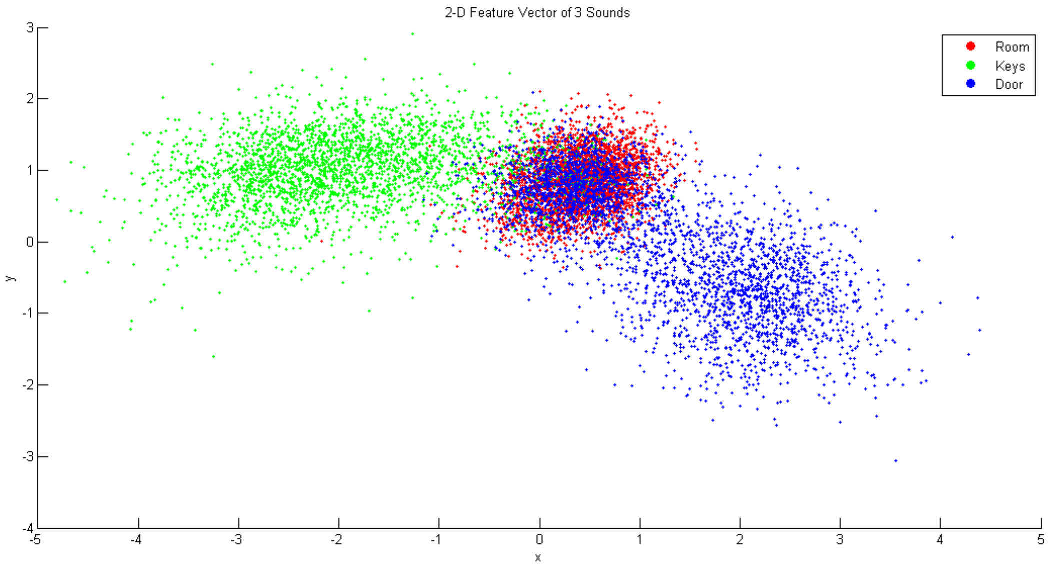


Figure 22: 2-D Feature Vector using MFCC1 and MFCC2

Figure 23 shows how well the EM algorithm fit a mixture of 2 Gaussians to the training data for door audio. From looking at the figure it can be seen that the model accurately approximates the distribution of the training data.

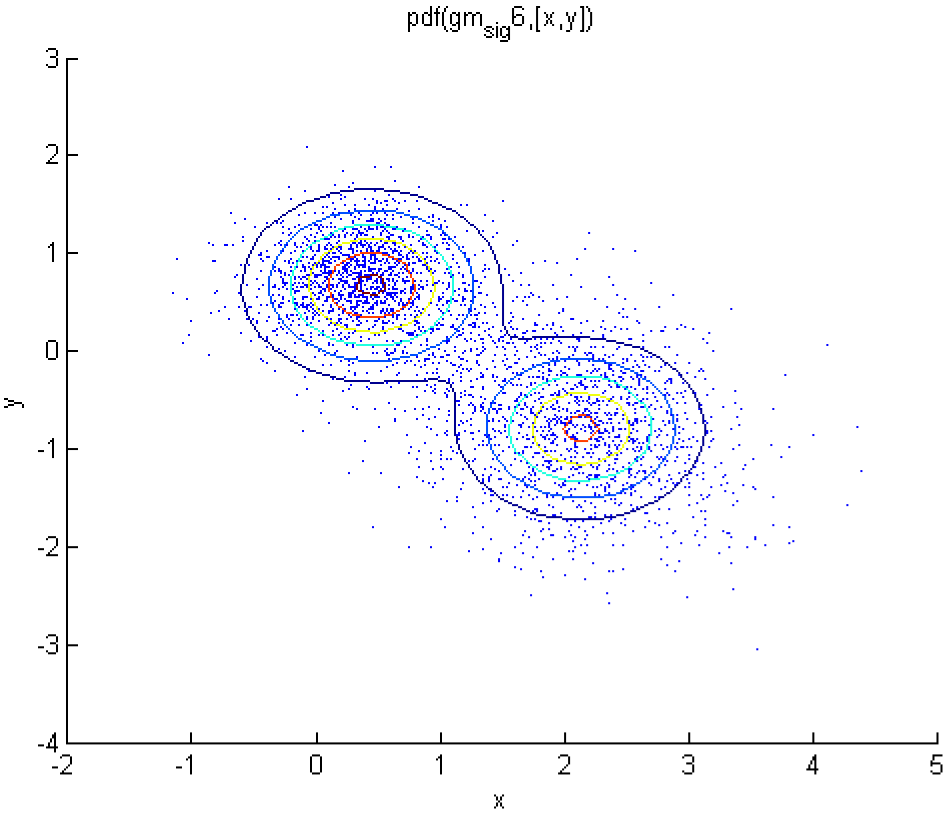


Figure 23: Gaussian Mixture Model of Door Audio

3.3.4.4 Audio Recognition using a GMM as a Classifier:

Figure 23 showed how well the EM algorithm fit 2 Gaussian mixtures to training data for door audio. A Gaussian Mixture Model like this is created for every audio class, and when new audio data is present the new audio data is compared to each of the different models from every sound class. The Log-Likelihood $L(O|X)$ is calculated for every audio class. The GMM that best fits the data is then chosen as the recognized sound class. Figure 24 shows a graphical example of how this comparison is performed, and an audio class is selected.

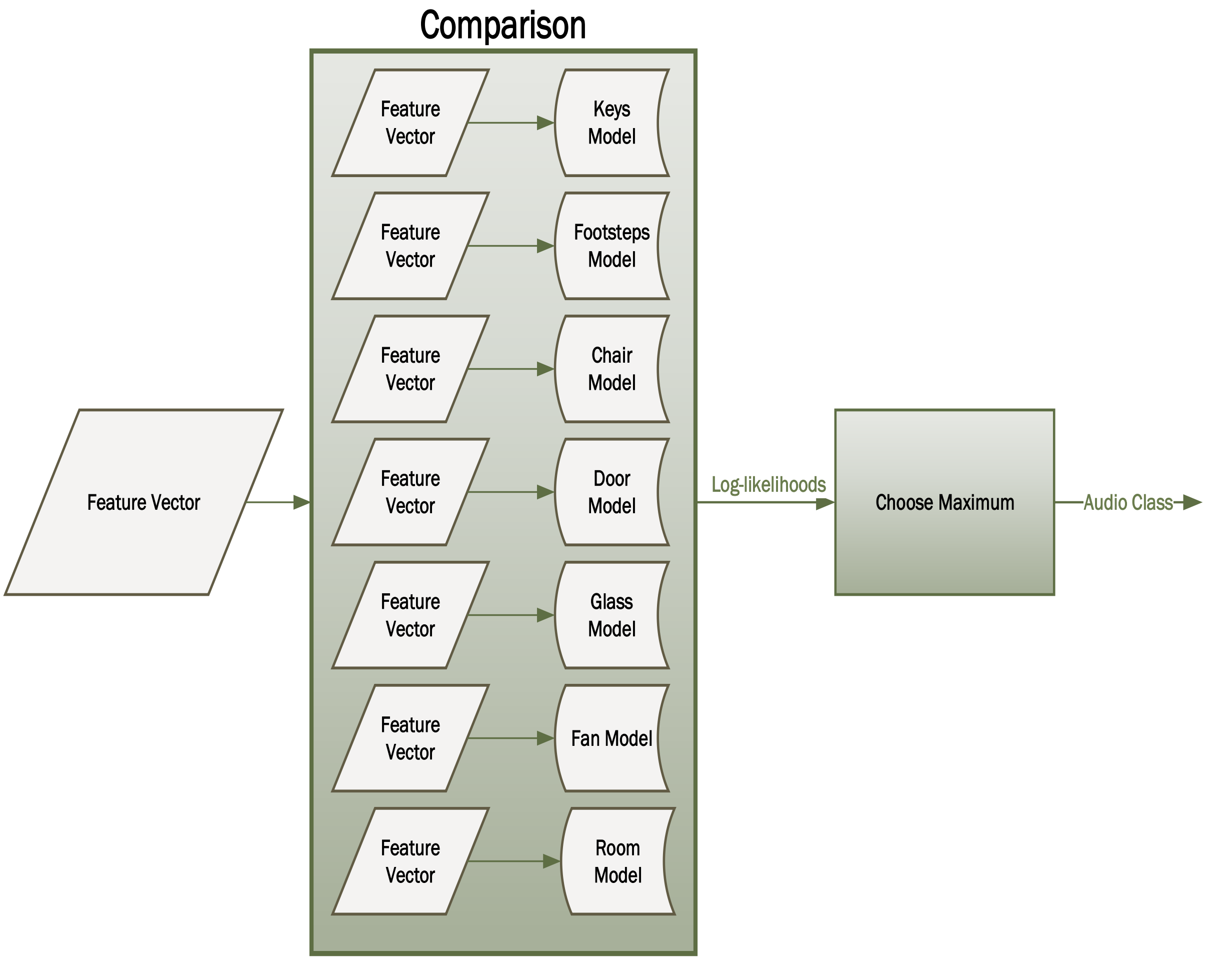


Figure 24: Audio Class Selection

3.4 Localization:

3.4.1 Problem Description:

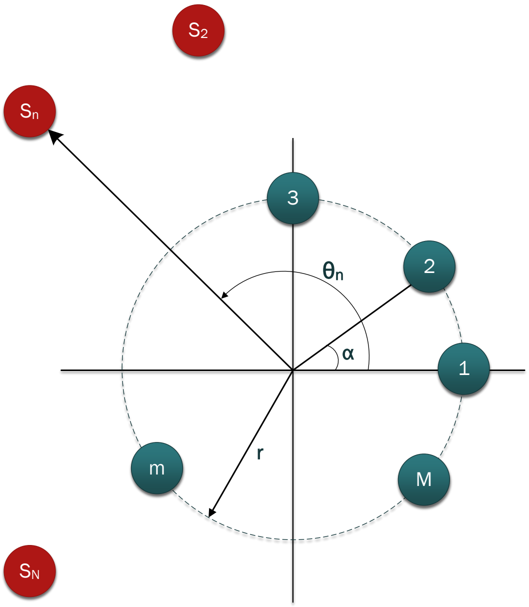


Figure 25: Microphone Array Configuration

$$x_m(t) = \sum_{n=1}^{N} \sum_{l=1}^{L_{m,n}} h_{mn}(\tau_l) s_n(t - \tau_l) + n_m(t), \qquad m = 1, \dots, M, \tag{27}$$

$$X_m(k, f) = \sum_{n=1}^{N} H_{m,n}(f) S_n(k, f) + \ N_m(k, f), \quad m = 1, \dots, M \tag{28}$$

A uniform circular microphone array is used for the purpose of environmental audio signal collection. The microphone array configuration can be seen in

Figure 25. The circular microphone array is used compared to a linear microphone array due to its ability to overcome the ambiguities seen from linear microphone arrays. This circular array

$$x_m(t) = \quad n = 1 \square N \square\, l = 1 \square L_{m,n} \square h_{mn}(\tau_l) s_n(t - \tau_l) + n_m(t,\ m = 1, \dots, M, \qquad) \square \qquad (\ 27\ )$$

contains M microphones. The distance *l* is the distance between any two adjacent microphones, and the radius *r* is the distance from the center of the microphone array to any of the microphones. The distance $l = 2r\sin(\frac{\alpha}{2})$ can easily be found where, $\alpha = \frac{2\pi}{M}$. The mixture $x_m(t)$ seen on microphone *m* is represented by the equation below.

n_m is additive white Gaussian noise seen at microphone *m*. s_n is the observed sound source where *n = 1, ..., N*. h_{mn} is the Room Impulse Response (RIR) between microphone *m* and source *n*. τ_l is the time delay of the significant paths, and L_{mn} is the length of the RIR.

3.4.2 DOA Estimation:

The problem description shows that there can be any number of sound sources, with M mixed signals received by the microphones. The goal is to be able to use each signal $x_m(t)$ in order to determine a Direction of Arrival (DOA) from a specific sound source to the microphone array.

$$q_m(k, f) = \frac{|X_m(k, f)|^2}{|X_m(k-1, f)|^2}, \qquad for\ m = 1, \dots, M \qquad (\ 29\ )$$

$$q(k, f) = \frac{\sum_{m=1}^{M} q_m(k, f)}{M} \qquad (\ 30\ )$$

In [3], the TF points are chosen by looking for the boundary of active/inactive portions of the signal through the usage of the power ratio described in (29). The average power ratio over all microphones is shown in (30). A high $\mathrm{q(k, f)}$ generally represents an inactive to active boundary point. This will likely allow for the selection of TF points that relate to the direct path from a sound source to the microphone array. The top *p*% of TF points that relate to the highest average power ratio are chosen in order to reduce the computation time, while still maintaining accuracy. The median of the average power ratios is taken as a threshold ε for simplicity.

Using this method the objective is to select TF points that can be used along with a single source localization algorithm. Single source localization is performed by taking the circular integrated cross spectrum (CICS) outlined in [10], and defined as,

$$CICS^{(k,f)}(\theta) = \sum_{m=1}^{M-1} \frac{X_m(k,f)X_{m+1}^{*}(k,f)}{|X_m(k,f)X_{m+1}^{*}(k,f)|} G_{m\rightarrow 1}^{(f)}(\theta) \tag{31}$$

where

$$G_{m\rightarrow 1}^{(f)}(\theta) = e^{-j2\pi f\left(\tau_{1,2}(\theta)-\tau_{m,m+1}(\theta)\right)}. \tag{32}$$

Eq. (31) is a function of θ which is dependent on both the locations of the microphones and the relative time delay between each microphone. The time delay between two adjacent microphones *m* and *m+1* for a source at a specific DOA θ, can be represented by eq. (33), where *c* is the velocity of sound in air.

$$\tau_{m,m+1}(\theta) = \ t_{m,n}(\theta) - t_{m+1,n}(\theta) = \frac{l\,sin\left(\pi - \theta + \left(m - \frac{1}{2}\right)\alpha\right)}{c} \tag{33}$$

The angle that maximizes the magnitude of eq. (31) is considered the DOA estimate $\hat{\theta}(k,f)$. This relation is shown in eq. (34). Figure 26 shows the Circular Integrated Cross Spectrum vs. the Direction of Arrival, with a true DOA at 160 degrees.

$$\hat{\theta}(k,f) = arg \max_{0<\theta<2\pi} \left|CICS^{(k,f)}(\theta)\right| \tag{34}$$

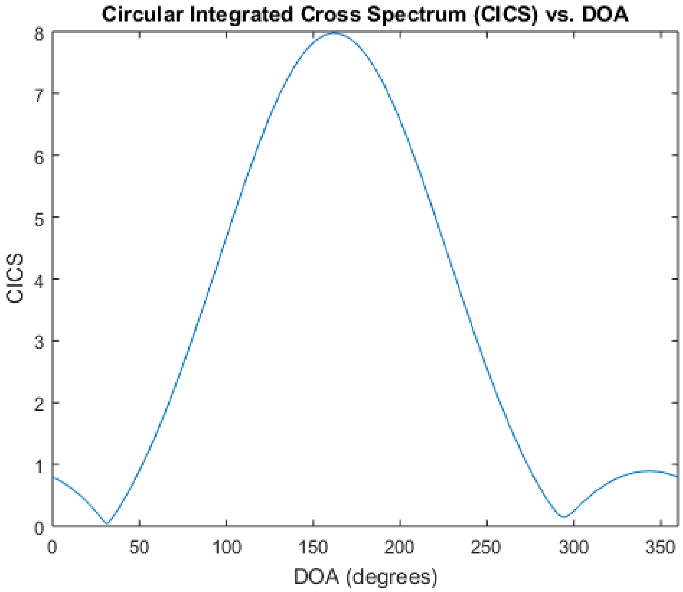


Figure 26: Circular Integrated Cross Spectrum (CICS) vs. DOA

Multiple estimates are taken and used to build a histogram to find the final estimate within a 1 second timer interval. In this work the number of sound sources is assumed known *a priori*, allowing for the final estimate(s) to equal the *N* highest peaks in the histogram where *N* is the number of sound sources. An example of this can be seen in Figure 27 where the three highest peaks are clearly defined and represent the three DOA estimates. The true DOAs used to make Figure 27 were 10, 120, and 280 degrees.

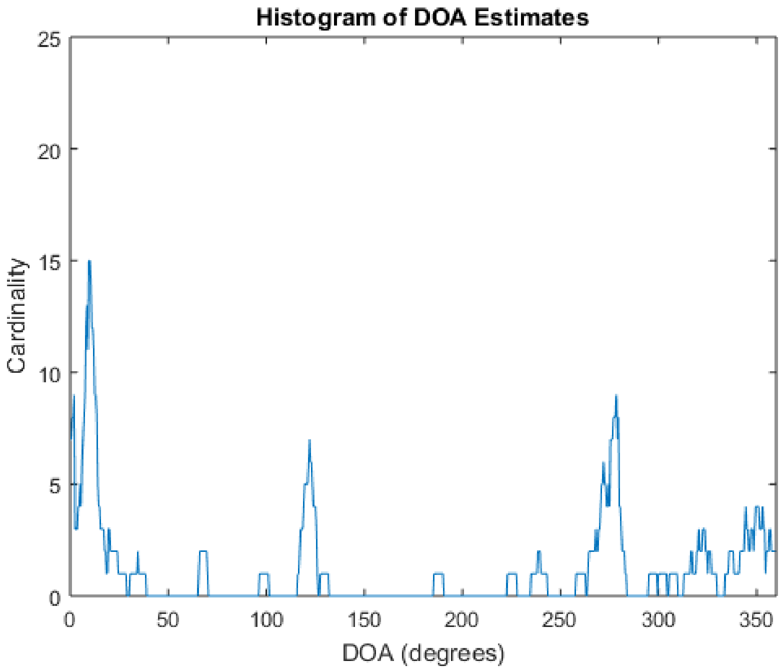


Figure 27: Histogram of DOA Estimates with true DOAs at 10, 120, & 280 degrees

The precision of the localization algorithm can be increased by adding extra sensor arrays that each independently produce their own estimates. Adding multiple circular arrays at different radiuses can help show peaks in the histogram that are not prevalent previously due to the frequency response of the microphone array at a specific radius. Figure 28 shows the histogram of DOA estimates from the array with a radius of 2cm. The sounds source at 185 degrees is very prominent, but the sounds source at 20 degrees is obscure. Figure 29 and Figure 30 both show a prominent curve at 20 degrees and at 185. By fusing all the estimates from each array into one histogram both of the sources can be seen very well as is shown in Figure 31. The maximum peaks are used as the final DOA estimates.

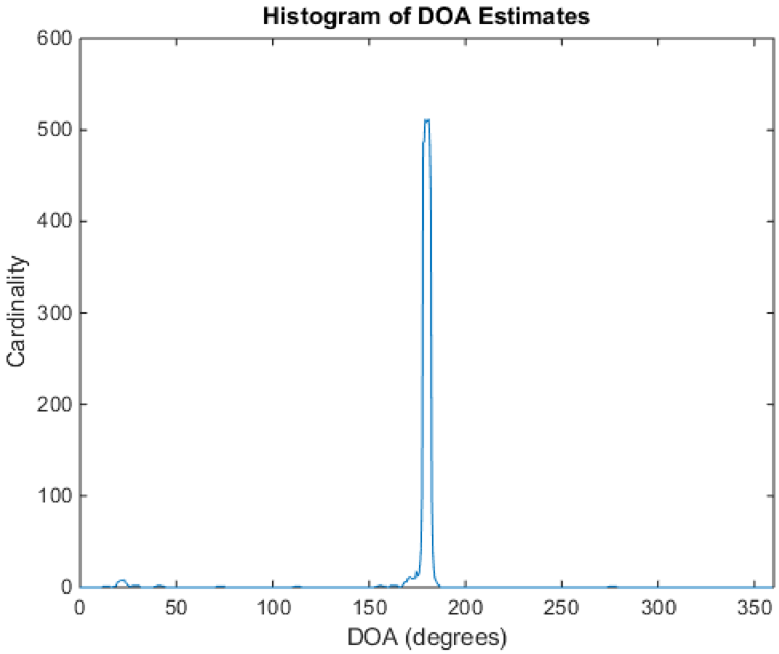


Figure 28: DOA estimates from r=2cm Array

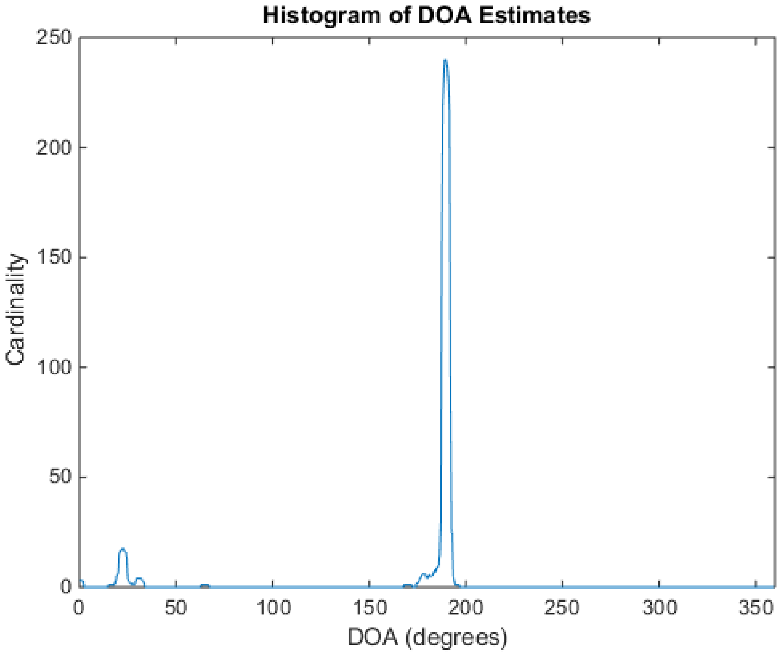


Figure 29: DOA estimates from r=4cm Array

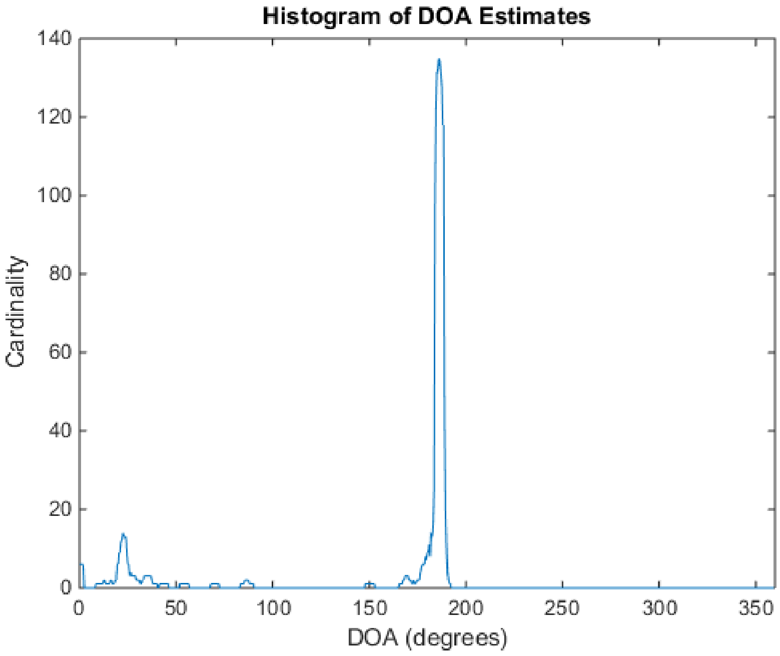


Figure 30: DOA estimates from r=6cm Array

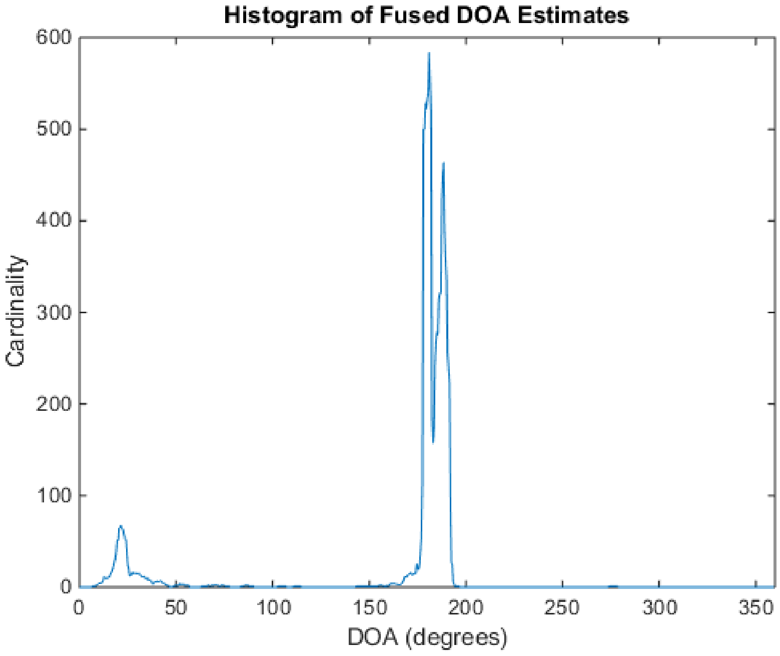


Figure 31: DOA estimates from all Arrays

3.5 Frost Beamformer:

The Frost Beamformer in this work uses the Constrained LMS (Least Mean Squares) algorithm in order to optimize or adjust an array of microphones to focus on a signal from a given look direction. The beamformer consists of *M* microphones, and *J* filter taps. The look direction frequency response is determined from *J* constraints, so the minimization of the total non-look-direction power, can be completed so long as the signal voltages on the taps is uncorrelated with the noise at these taps. The desired scenario in fact would be that the noise voltages are uncorrelated amongst themselves, while uncorrelated with the signal voltages. Minimizing the total output power from the array breaks down to minimizing either the noise power or the power from interfering signals. Increasing the number of filter taps increases the number of filter weights. Non-look-direction noise waveforms most likely will not produce the same voltages on all the taps in a column while signals coming from the look direction should produce similar voltages on all the taps in a column. There are two ways this beamformer is able to eliminate noise from a non-look-direction. The adaptive array can reduce the weight on any set of taps that contains a large amount of uncorrelated noise or interferer power. Increasing the amount of taps will increase the adaptability of the beamformer, by providing an increased reduction in non-look direction signal power. Second this design will also add the voltages from the taps coherently at the output, while noise signals should add up destructively at the output. This is done in a similar fashion as what was discussed with the delay and sum beamformer. The filter weights from the frost beamformer are used to minimize the total output power from the array. The algorithm is able to maintain a given frequency response in a given look direction while minimizing the output noise power due to a relation between the frequency response and the filter weights.

This section will first focus on some of the basic notation, and then will focus on the LMS weight vector and show how the optimum weight vector is determined.

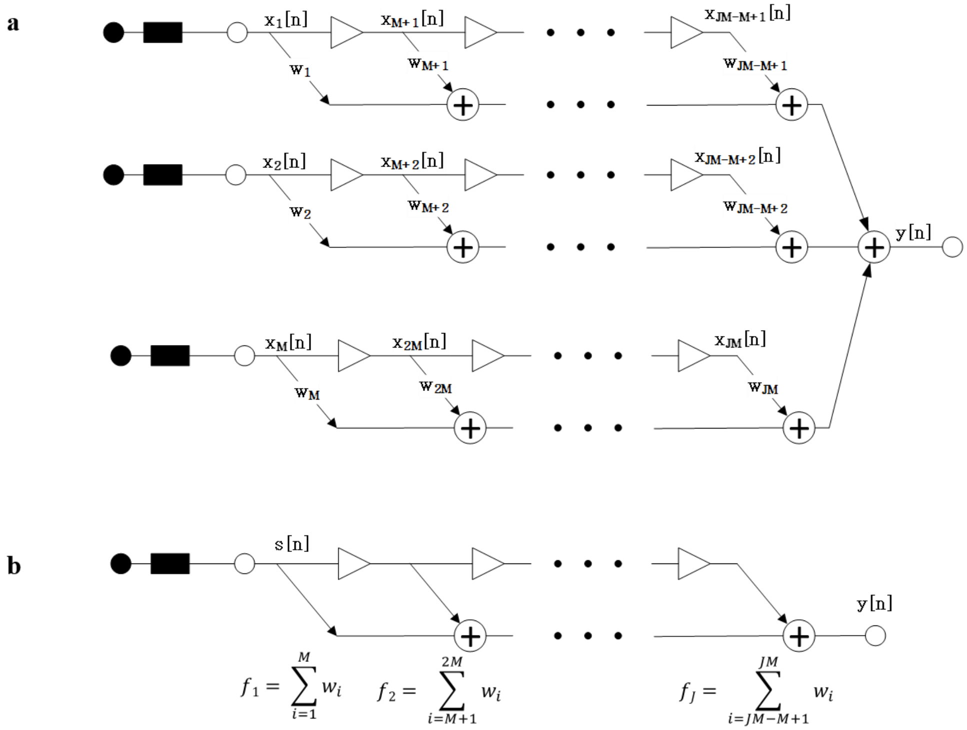


Figure 32: (a) Frost Beamformer Framework (b) The equivalent Frost Beamformer Framework for signals coming from the look direction

To start the discussion on the frost beamformer let us first define some needed terms. The noisy input signal from the *M* microphones are formed by both the clean source signal *s(t)* and the some added noise *n(t)*. The sampled signal at each tap is represented by X[n] and the vector ***W*** consists of all of the filter weights. The vector F contains the constrained impulse response, and the matrix C is used for constraint formulation. The vector of tap voltages can be written as,

$$X^T[n] \triangleq [\, x_1[n], x_2[n], \dots, x_{MJ}[n] \,] \tag{35}$$

The vector weights at each tap can be written as,

$$W^T \triangleq [\, w_1, w_2, \dots, w_{MJ} \,] \tag{36}$$

There is a constraint that all the weights on a given *jth* column of taps will sum to a chosen f_i. These constraints allow the beamformer to be tuned to match a specific frequency response. This is especially useful for communications purposes where the received signal should be within some frequency range. The beamformer could match the necessary frequency response needed to alleviate noise at the output. For the purposes of an audio signal, the received signal will not be a narrow band signal as is described above. The signal intended to be classified is a broadband audio signal so the frequency response of the filter should be all pass and linear phase (distortionless). This will be important for the audio recognition algorithm. More about this will be explained in detail in the next section. The requirement for the constraints can be expressed by,

$$c_j^T W = f_j \qquad j = 1, 2, \dots, J \tag{37}$$

The full constraints matrix C can be defined as,

$$C \triangleq [\, c_1 \;\cdot\;\cdot\; c_j \;\cdot\;\cdot\; c_J \,] \tag{38}$$

where C is a (MJ – J) dimensional matrix.

$$c_j^T = [\underbrace{0 \cdots 0}_{\mathbf{M}} \cdots \underbrace{0 \cdots 0}_{\mathbf{M}} \; \underbrace{1 \cdots 1}_{j^{th}\text{ group of M elements}} \; \underbrace{0 \cdots 0}_{\mathbf{M}}] \tag{39}$$

The vector F is J dimensional and is composed of equivalent tapped delay line filter weights from Figure 32. F can be described by,

$$\mathrm{F} = [f_1 \;\; f_2 \;\; f_3 \;\cdots\; f_J] \tag{40}$$

and the constraints can be shown by,

$$C^T W = \mathscr{F} \tag{41}$$

3.5.1 Optimizing the Weight Vector

In [12], it is explained that since the look direction frequency response is fixed by the relation shown in eq. (41), the minimization of the power from interfering sources is the same as minimizing the total output power which can be represented by,

$$E[y^2[n]] = E[W^T X[n] X^T[n] W] = W^T R_{xx} W \tag{42}$$

where $R_{xx}$is the autocorrelation matrix. The optimization problem and constraints are represented by,

$$\min_W W^T R_{xx} W \tag{43}$$

s.t.

$$C^T W = \mathscr{F} \tag{44}$$

The Lagrange Multipliers method is used in order to obtain the Wiener solution for W_{opt}. W_{opt} is then represented by,

$$W_{opt} = R_{xx}^{-1} C [C^T R_{xx}^{-1} C]^{-1} \mathscr{F} \tag{45}$$

The adaptive LMS algorithm, [12] is given by

$$\begin{gathered} W[0] = F \\ W[n+1] = P\big[W[n] - \ \mu y[n] X[n]\big] + F \end{gathered} \tag{46}$$

where the projection matrix P and the vector F can be represented as

$$F \triangleq C(C^T C)^{-1} \mathscr{F} \tag{47}$$

$$P \triangleq I - C(C^T C)^{-1} C^T \tag{48}$$

and the output $y[n]$ can be represented as,

$$y[n] = X[n]^T W[n]. \tag{49}$$

The choice of μ from (46) effects the ability of the system to converge, and the amount of time it takes for the weights to converge. Since μ is considered the step size, optimally it should be chosen to be as large as possible while still meeting all convergence criteria. Doing this should decrease convergence time. Griffiths [13], shows that (50) can be calculated and used to determine an upper bound for the step size.

$$0 < \mu < \frac{2}{3tr(R_{xx})} \tag{50}$$

CHAPTER IV

SIMULATION RESULTS

4.1 Audio Recognition

In this section different experiments were performed to determine how well the audio recognition algorithm worked with real audio samples. This led to the creation of an audio library that consisted of 1169 audio samples that were recorded from 7 different sound classes. The 7 different sound classes represented the sounds of footsteps, chair sliding, a fan, keys, door knocking, glass breaking, and the room when quiet. The audio recognition algorithm was tested in both the no noise case, and with added white noise. In the case where noise was added, a SNR degredation test was performed where the SNR was taken from 50dB down to 0dB.

4.1.1 Training:

In the training phase as we know a GMM needs to be created for all of the different sounds classes. In this experiment all 7 different sound classes were trained to without noise and again with noise. This led to the creation of 14 different GMMs or sound classes that mapped to only 7 different sounds. So for every sound class, there is a clean model and a noisy model. The noisy training sounds used for the noisy models had the SNR randomly varied uniformly between 50dB and 0dB. The training database stayed the same for all the experiments performed.

4.1.2 Performance:

The performance of the audio recognition algorithm is measured by three conditions, the precision, false rejection rate, and the false detection rate. The precision is a measure of the number of events that were correctly detected and classified, to the total number of events that were detected. The false rejection rate measures the number of events that were not detected when there was an event to detect, and the false detection rate measures the number of times an event was detected when there was no actual event. The performance equations can be seen in (51) through (53). [7]

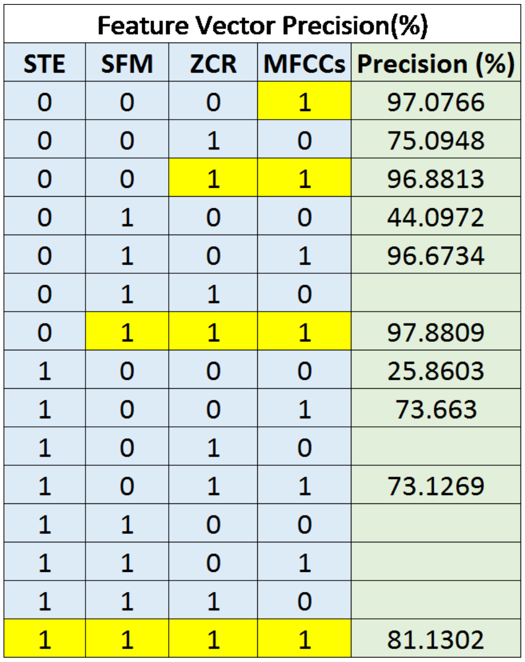
$$precision = \frac{Number\ of\ Events\ Correctly\ Detected}{Number\ of\ Events\ Detected} \tag{51}$$

$$False\ Rejection\ Rate = \frac{Number\ of\ No\ Event Results}{Number\ of\ Events\ to\ Detect} \tag{52}$$

$$False\ Detection\ Rate = \frac{Number\ of\ Detected\ Events\ that\ were\ the\ Room}{Total\ Number\ of\ Room\ Samples} \tag{53}$$

4.1.3 Feature Extraction/Selection:

The features that were investigated for feature extraction were Mel-Frequency Cepstral Coefficients, Zero Crossing Rates, Spectral Flatness, and Short Time Energy. There were 15 different combinations of these features from which the feature selection process could possibly choose from. The feature selection algorithm was used in order to determine the feature vector that required the least amount of features that statistically maximized the precision of the audio recognition algorithm. The feature selection algorithm determined that MFCCs were the most relevant of all the extracted features from the precisions. Figure 33 shows all the different combinations of features that could be placed in the final feature vector, and all the precisions found during the feature selection process. All of the boxes in yellow are the features that were chosen after each iteration of the SFS algorithm. The hypothesis testing portion of the feature selection algorithm used a significance level of 5% in order to determine if the difference between proportions were significant. MFCCs were determined to be the most significant feature. The precision values seen in Figure 33 were calculated with no added noise to the testing samples. The features were selected based off of the audio samples without added noise, and that set of features was later tested with noise during the SNR degredation test.



Feature Vector Precision(%)				
STE	SFM	ZCR	MFCCs	Precision (%)
0	0	0	1	97.0766
0	0	1	0	75.0948
0	0	1	1	96.8813
0	1	0	0	44.0972
0	1	0	1	96.6734
0	1	1	0	
0	1	1	1	97.8809
1	0	0	0	25.8603
1	0	0	1	73.663
1	0	1	0	
1	0	1	1	73.1269
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	81.1302

Figure 33: Precision of Feature Vectors

4.1.4 SNR Degredation Performance:

The SNR degredation performance test added noise to all of the testing audio samples to meet specific SNR values ranging from 0-50dB. This experiment was meant to test the performance of the algorithm in the presence of a noisy environment. The results of the SNR degredation test can be seen in Figure 33 through Figure 36. These results represent the precision, false rejection rate, and false detection rate of the audio recognition algorithm at different SNR’s ranging from 0-50dB. In order to get a more broken down view of the performance of the algorithm a confusion matrix was created at every SNR seen in the figures below. The confusion matrix shows the expected output versus the detected output. All of the performance metrics proposed in this paper can be calculated from the confusion matrix. The confusion matrix for 50dB testing samples, and 0dB testing samples can be seen in Table 2 and Table 3 respectively.

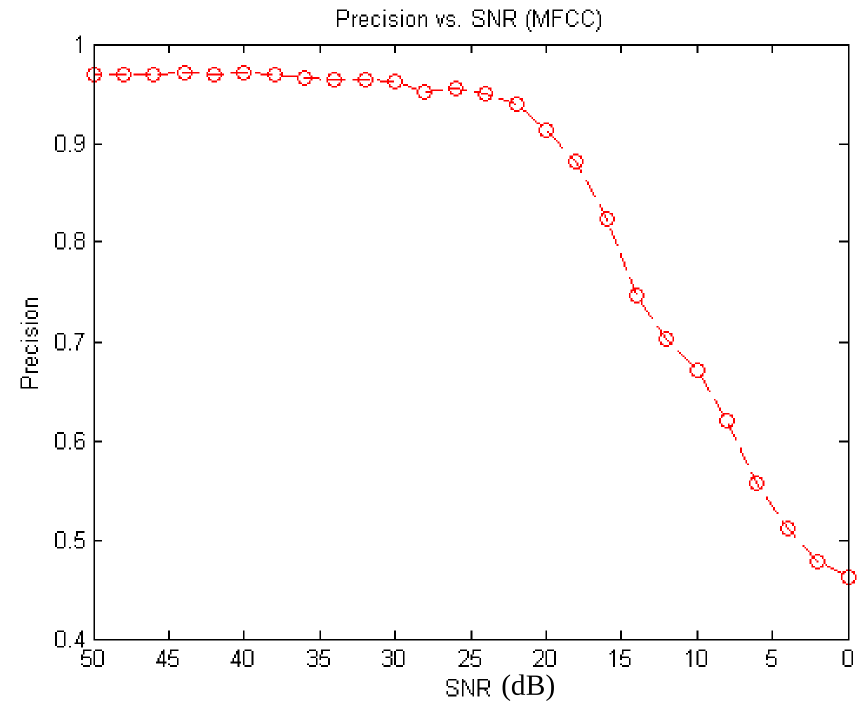


Figure 34: Precision vs. SNR

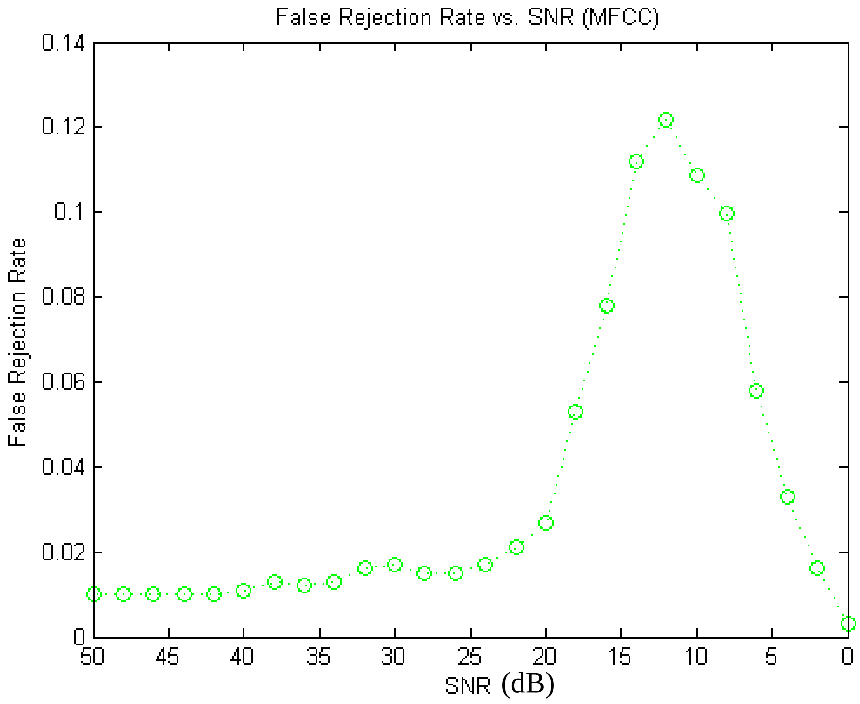


Figure 35: Rejection Rate vs. SNR

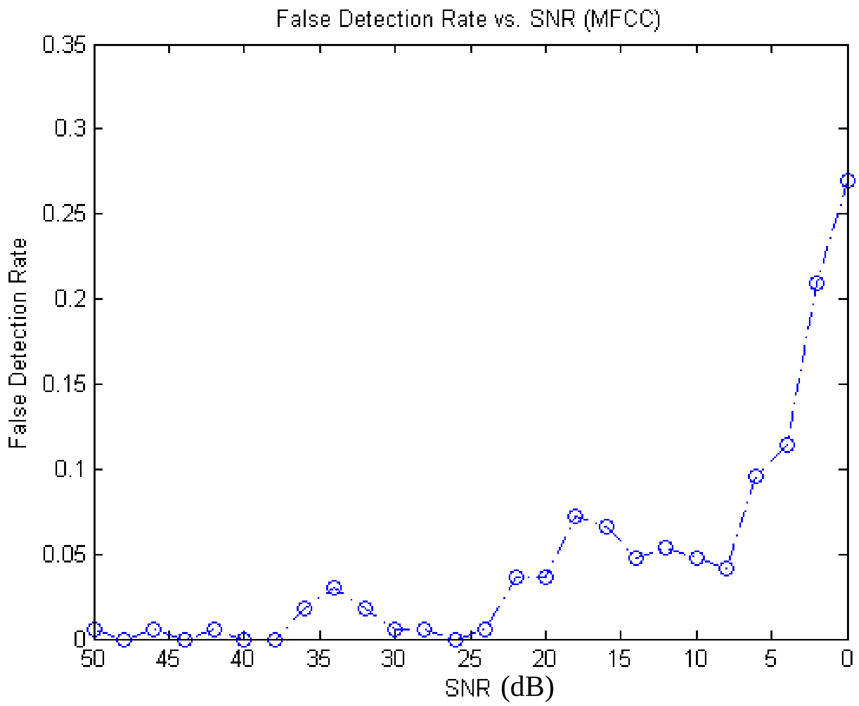
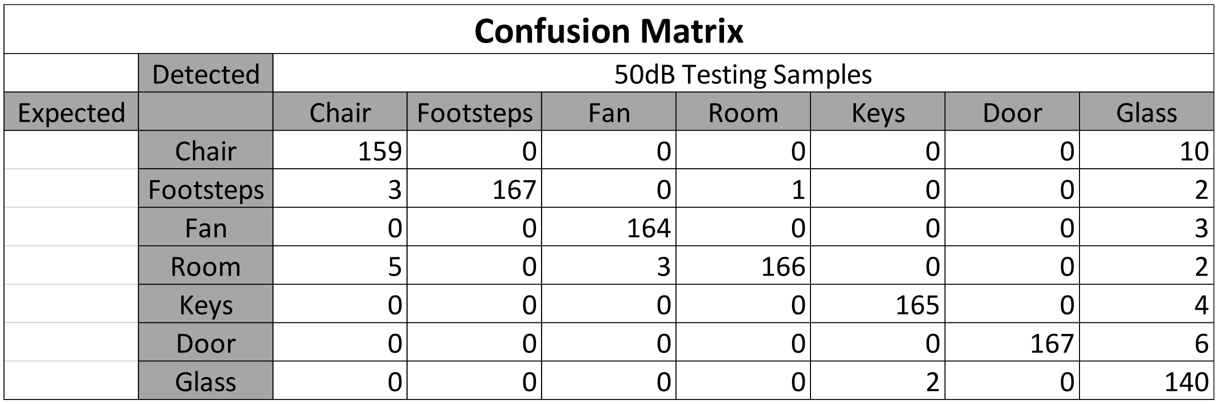
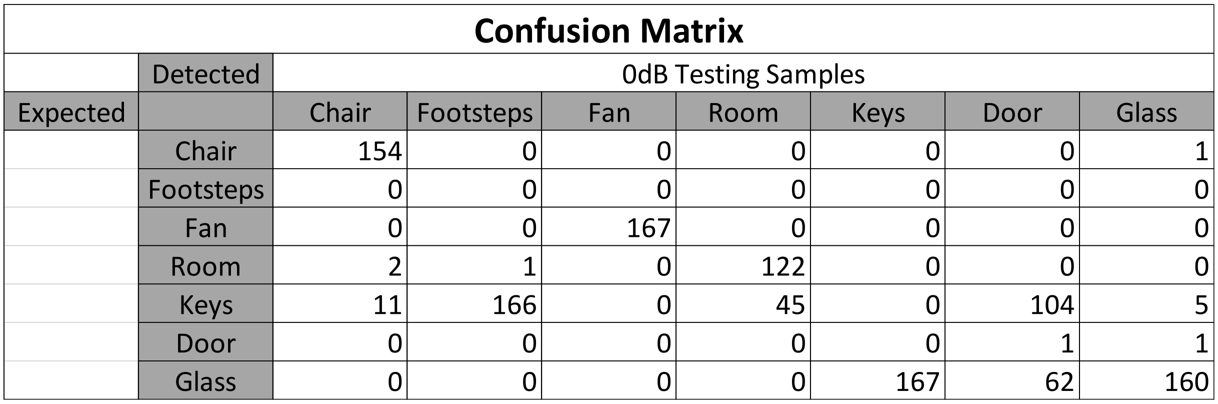


Figure 36: False Detection Rate vs. SNR



Confusion Matrix								
	Detected	50dB Testing Samples						
Expected		Chair	Footsteps	Fan	Room	Keys	Door	Glass
	Chair	159	0	0	0	0	0	10
	Footsteps	3	167	0	1	0	0	2
	Fan	0	0	164	0	0	0	3
	Room	5	0	3	166	0	0	2
	Keys	0	0	0	0	165	0	4
	Door	0	0	0	0	0	167	6
	Glass	0	0	0	0	2	0	140

Table 2: Confusion Matrix with 50dB Testing Samples



Confusion Matrix								
	Detected	0dB Testing Samples						
Expected		Chair	Footsteps	Fan	Room	Keys	Door	Glass
	Chair	154	0	0	0	0	0	1
	Footsteps	0	0	0	0	0	0	0
	Fan	0	0	167	0	0	0	0
	Room	2	1	0	122	0	0	0
	Keys	11	166	0	45	0	104	5
	Door	0	0	0	0	0	1	1
	Glass	0	0	0	0	167	62	160

Table 3: Confusion Matrix with 0dB Testing Samples

4.2 Simulation

The following tests performed in the rest of this Chapter were based on the simulated mixed signals seen on each microphone. Simulating a signal coming from a specific Direction of Arrival (DOA) can be done since for a sound source the approximate time difference between microphones can be calculated, and the phase of the signal can be compensated to match this effect. Doing this will simulate a sound coming from a specific direction, but will be more characteristic of an outdoor environment.

4.3 Localization

The localization algorithm was tested from the testing library with 1000 different sounds representative of all of the sound classes except for the room sounds. These sounds were then mixed on each microphone and given a simulated DOA. Testing of the localization algorithm allowed for the setting of the number of array used in order to perform DOA estimation. The RMSE (Root Mean Square Error), and the Median Error (ME), were the statistics used in order to establish the performance of the localization algorithm. The ME was used along with the RMSE in order to give a better understanding about the distribution of the error in the estimates. K is the total number of samples and N is the number of sound sources. Before looking at the results in this section it is important to know that the resolution of the localization algorithm is set to 0.5 degrees. This is due to the fact that the Circular Integrated Cross Spectrum needs to be calculated from 0 to 360 degrees to come up with estimates, and a resolution of 0.5 degrees was chosen.

$$RMSE = \sqrt{\sum_{k=1}^{K}\sum_{n=1}^{N}\frac{\left(\theta_{n,k}-\hat{\theta}_{n,k}\right)^2}{N\cdot K}} \qquad (54)$$

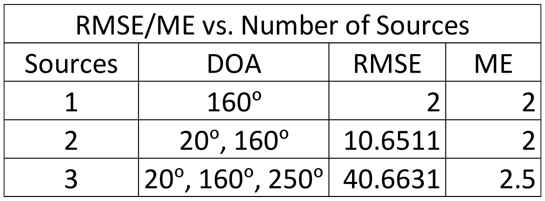
$$ME = median(\theta_{n,k}-\hat{\theta}_{n,k}) \qquad (55)$$

It was determined that increasing the number of arrays decreased the RMSE. The median error stayed pretty constant but the number of outlier estimates were decreased by forming a single set of estimates from multiple arrays. The downside to increasing the number of arrays is that there is extra processing cost, so the goal is not to increase the number of arrays to be too large. After increasing the number of arrays past 3 the RMSE didn’t decrease by a significant amount so the number of arrays were set to 3. Table 4 shows the results of the RMSE and the median error as the number of arrays were increased. The median error stayed constant at 2 degrees while the RMSE with 3 arrays dropped to 10.65 degrees. The reason the median error happens to be a nice round number is due to the resolution of the localization algorithm.

2 sound sources (20°,160°)			
number of arrays	radius(cm)	RMSE	median error
1	2	18.5538	2
2	2,4	12.1281	2
3	2,4,6	10.6511	2

Table 4: Number of Arrays vs. Error with 2 Sound Sources

The number of sources does affect the accuracy of the localization algorithm. After the number of arrays for the localization had been set to three, a test was performed in order to determine how increasing the number of sources affected the RMSE and the ME. It can be seen in Table 5 that the RMSE goes from 2 degrees with one sound source up to 40.66 degrees with 3 sound sources, but the median error only jumps to 2.5 degrees. When there are multiple sound sources the fan sounds seem to sometimes cause large errors in the localization. These errors do not occur that often, but the errors are large enough to pull the RMSE to a larger value. The RMSE along with the ME can be used to help form an idea of how the errors are distributed. If the fan sounds are taken completely out of the test, the RMSE is much lower. The distribution of the Localization Error can be seen in Figure 37.



RMSE/ME vs. Number of Sources			
Sources	DOA	RMSE	ME
1	160°	2	2
2	20°, 160°	10.6511	2
3	20°, 160°, 250°	40.6631	2.5

Table 5: Root Mean Square Error / Median Error as Sources Increase

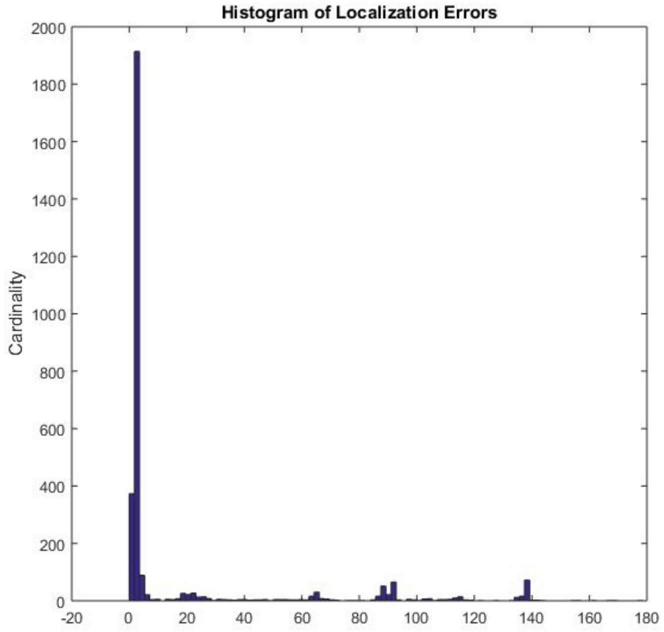


Figure 37: Histogram of Localization Error with three sound sources

4.4 Beamforming:

Since the objective of the frost beamformer was to allow the system to be able to handle multiple sound sources, the testing library consisting of 1000 different sounds was used along with the audio recognition algorithm to determine the performance of the beamformer. Each sound was randomly selected at a specific DOA, and sent through the audio recognition algorithm. For example if there were 2 sound sources the test would go through 1000 iterations where any sound from the testing library could be selected for a specific DOA, and the same for the second DOA. No sound that was previously selected would be selected again and the test would run until all sounds have been exhausted in each look direction.

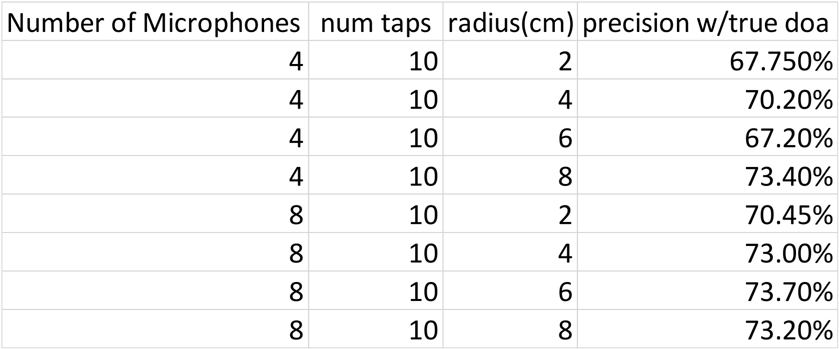
Testing determined that MFCCs did not hold up that well as a stand-alone feature in the presence of interfering sound sources. In order to determine the feature set that would hold up best in the presence of interfering signals the feature selection algorithm outlined in Chapter 3 was used with 2 sound sources, 8 microphones, 10 taps, and a radius of 12cm. The feature selection chart can be seen in Table 6. The most significant feature subset was the {SFM, ZCR, MFCCs}. Short Time

Energy was left from testing in this section due to its poor performance from the audio recognition section.

Audio Recognition			
10 Taps 10º&160º			
SFM	**ZCR**	**MFCCs**	**Precision (%)**
0	0	1	70.9
0	1	0	61.85
0	1	1	75.45
1	0	0	47.25
1	0	1	76.85
1	1	0	
1	1	1	79.8

Table 6: Precision of Feature Vector with 2 sound sources, r=12cm, and 10 filter taps

The frost beamformer structure has some criteria that needs to be defined such as the number of microphones, the number of taps, and the radius of the microphone array. From looking at the directivity patterns, the best response seems to come from using 8 microphones with a radius of 6cm. In order to test this the number of microphones was set to be either 4 or 8. The radius of the array was incrementally increased and the precision of the algorithm was recorded. The number of taps were set to 10 and later increased after the number of microphones and the radius of the array was set. This was done to reduce computational time as the experiment was performed, and the number of taps could later be increased until the performance no longer increased significantly. The number of microphones and the radius that yielded the best precision was 8mics, with a radius of 6cm. From above it can be seen that a radius of 12cm was used in order to determine the final feature vector to be used. The reason this radius did not get tested more is because the beamwidth at this radius was too small and doesn't allow for enough variability in the estimated DOA. Increasing the radius decreases the beamwidth, while increasing the number of microphones decreases the sidelobes.



Number of Microphones	num taps	radius(cm)	precision w/true doa
4	10	2	67.750%
4	10	4	70.20%
4	10	6	67.20%
4	10	8	73.40%
8	10	2	70.45%
8	10	4	73.00%
8	10	6	73.70%
8	10	8	73.20%

Table 7: Microphones vs. Radius Precision

Since the incoming signal is not a narrowband signal it can be a little difficult to come up with an exact optimum radius or number of microphones. This is due to the fact that the beam patterns are frequency dependent. One way a design can be chosen is by having some knowledge of the frequency characteristics of the expected signal. Knowing this information can aid in the design by looking at beam patterns at various frequencies that are representative of the incoming expected signal. After looking at the beam patterns for the different microphone/radius choices it was determined that having 8 microphones with a radius of 6cm would be effective for the beamformer. This coincides with the results seen from Table 7.

The number of taps used for the system were incrementally increased until the precision of the system stabilized or did not increase by a significant amount. This is how the number of filter taps were determined. At this point the radius and the number of microphones had already been set, the goal was to increase the number filter taps and push the performance/precision as high as possible.

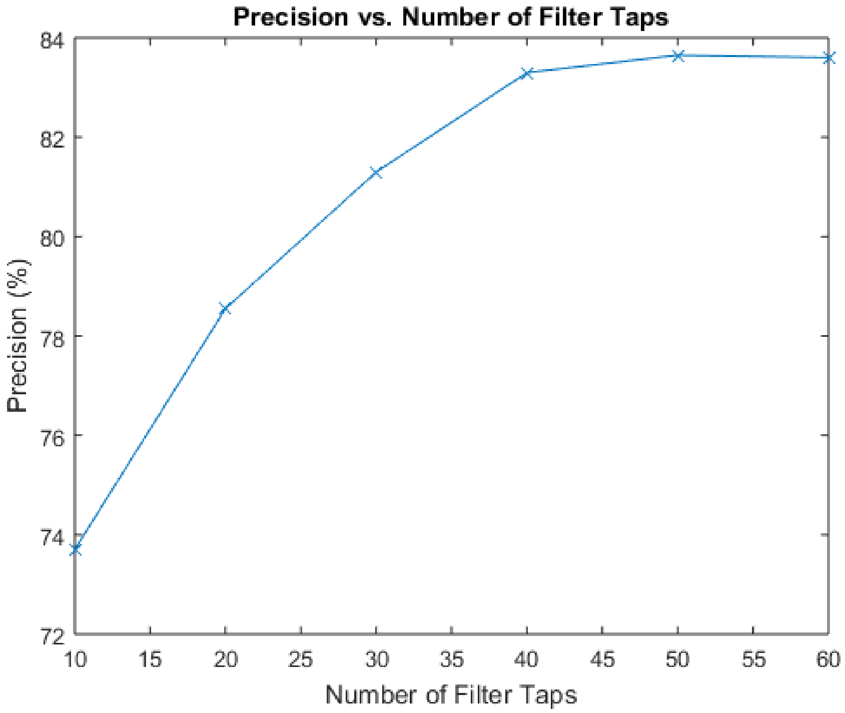


Figure 38: Precision vs. Number of Filter Taps

4.4.1 Separation Results:

The purpose of the beamformer as used in this work is to be able to separate out a single sound source in the presence of interfering sound sources, while maintaining the characteristics of the original source. Figure 39 shows the two individual signals before they were mixed together, and the mixed signal. Since there are two different sound sources two different beamformers can be opened up to focus on both of the look directions. The mixed signal is sent through the beamformer, and the beamformer outputs are shown below for both the Glass Breaking and Chair signals. By inspection there is a strong resemblance between the beamformer output and the original signals before they were mixed. Figure 40 shows the filter weights as they were updated over time. Table 8 shows the DOA for each signal and the SIR(dB) at the output of each beamformer focused in the specified look direction.

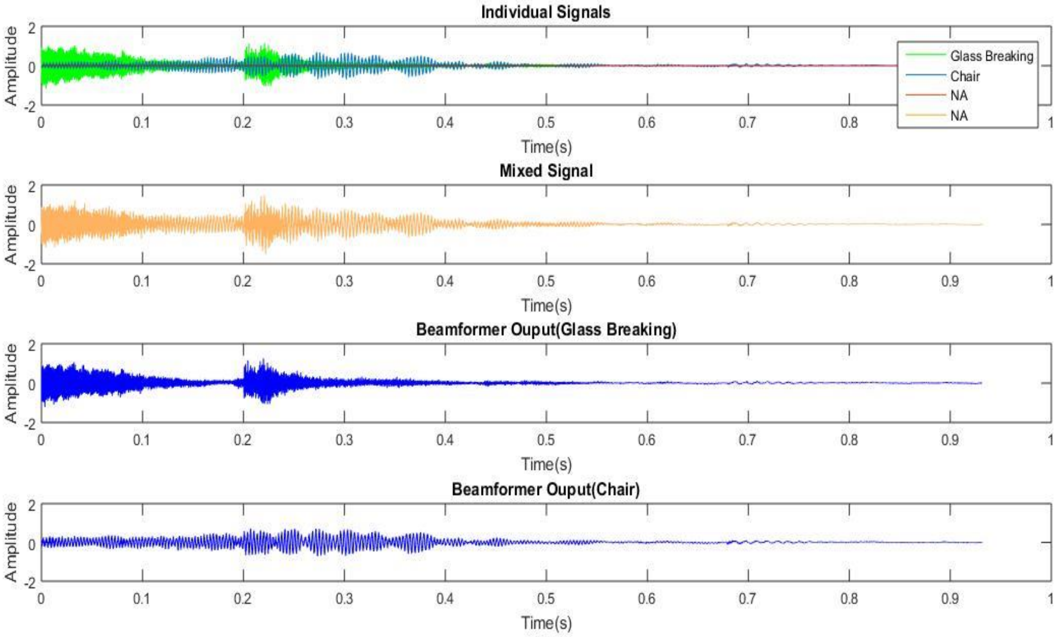


Figure 39: Beamformer input/output for Glass Breaking and Chair Sounds at 160 & 20 degrees respectively

Mixed Signal (Glass/Chair)		
Signal	**DOA**	**SIR(dB)**
Glass Breaking	160	25.2494
Chair	20	77.7312

Table 8: Mixed Signal Glass/Chair

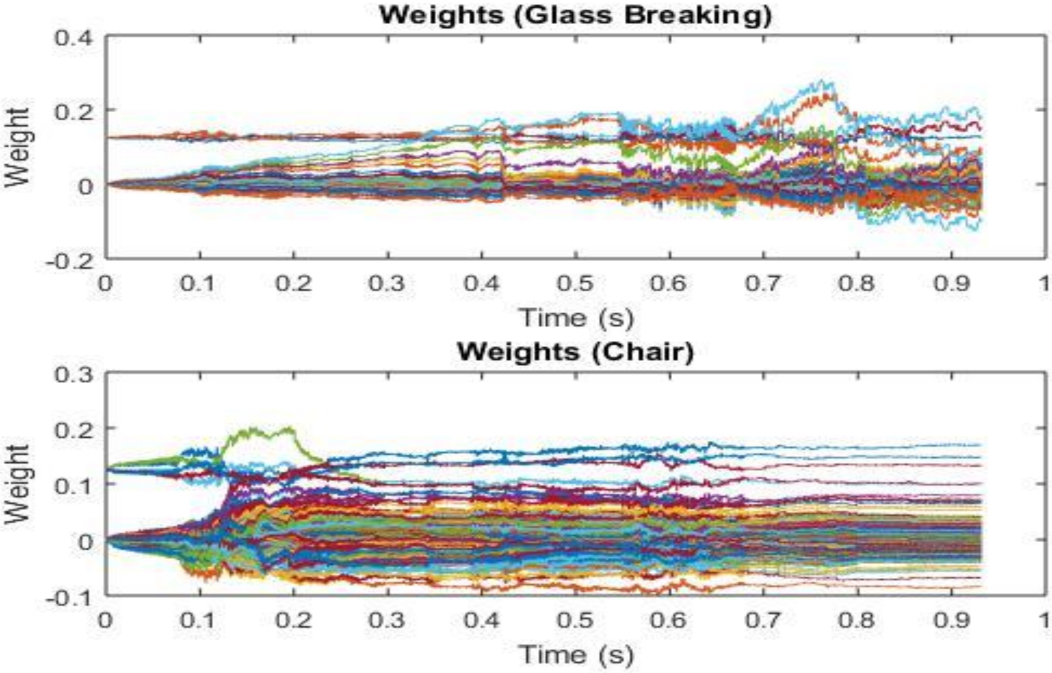


Figure 40: Filter Weights vs. Time (Glass/Chair)

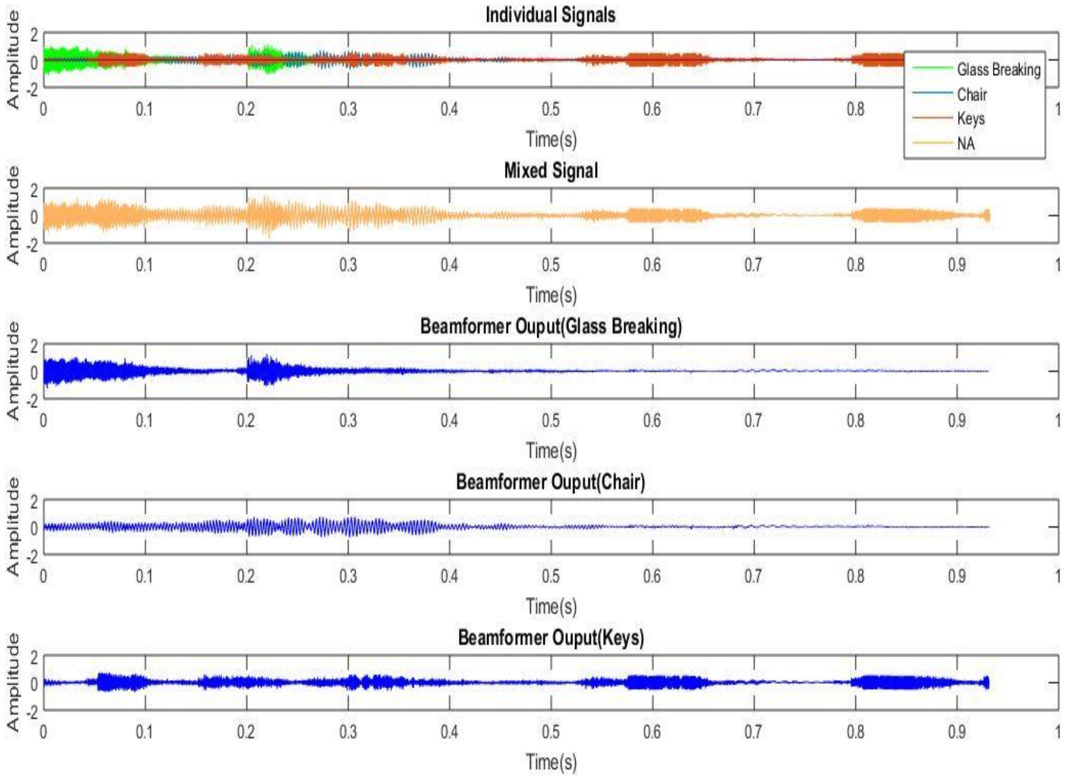


Figure 41: Beamformer input/output for Glass Breaking, Chair Sounds, and Keys at 160, 20, and 240 degrees respectively

Mixed Signal (Glass/Chair/Keys)		
Signal	**DOA**	**SIR(dB)**
Glass	160	24.7429
Chair	20	37.5086
Keys	240	15.0457

Table 9: Mixed Signal (Glass/Chair/Keys)

The beamformer can potentially be used to perform sound source localization by performing a 360 degree power scan. The 360 degree power scan could keep track of the received signal power after each iteration. The same as how the highest peaks were used in the localization algorithm from this work the highest peaks in the 360 degree scan can be selected as the DOA to specific sound sources. There would be an extreme cost to performing sound source localization this way. The computational cost compared to the localization algorithm used in this work would be much higher. An example of a 360 degree scan can be seen in Figure 42 below.

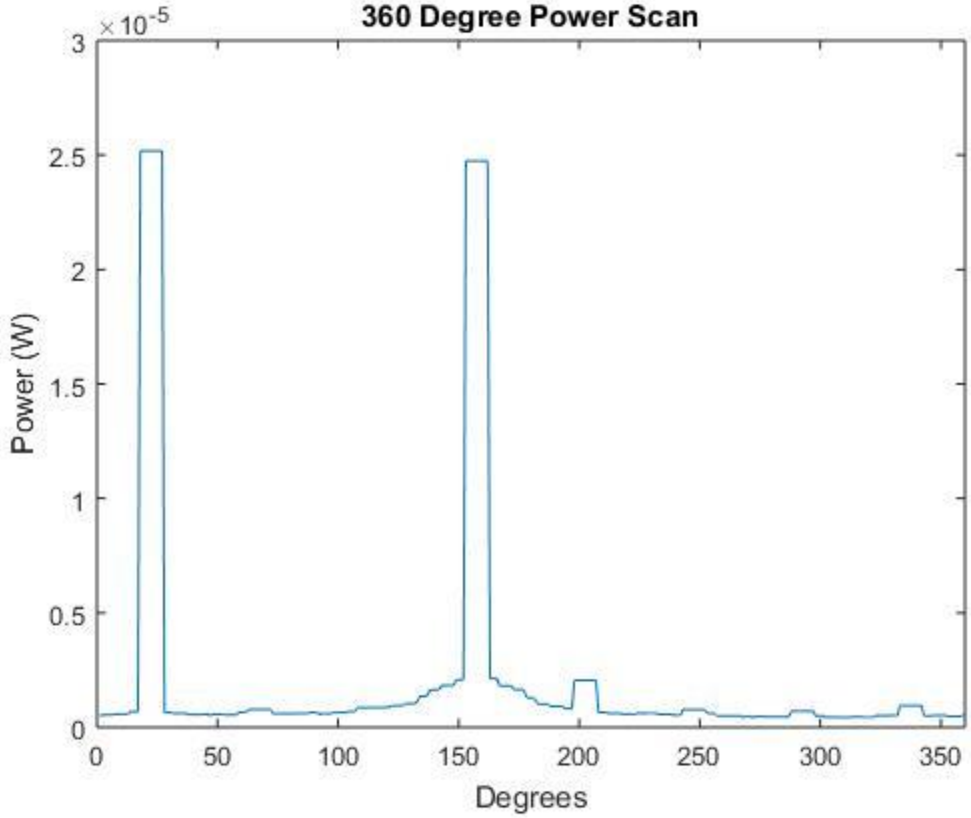


Figure 42: 360 Degree Power Scan Glass/Keys at 20 and 160 degrees

4.5 Beamforming and Audio Recognition Results

In this section both the number of sound sources, and the true Direction of Arrival for each sound source is assumed to be known *a priori*. This gives an understanding of the results with no localization error. The Audio Recognition feature set uses the SFM, ZCR, and MFCCs. A thousand sounds were randomly selected at each DOA and the precision of the test is recorded in Table 10.

Beamforming and Audio Recognition Results		
Number of Sound Sources	DOA	Precision (%)
1	160º	97.6
1	300º	97.6
2	20º, 185º	84.4
2	50º, 190º	83.85
3	20º, 160º, 240º	70.533
3	50º, 190º, 300º	70.7333

Table 10: Beamforming and Audio Recognition Results

Confusion Matrix							
	Detected	DOA: 160°					
Expected		Chair	Footsteps	Fan	Keys	Door	Glass
	Chair	161	0	0	0	0	7
	Footsteps	6	167	0	0	0	0
	Fan	0	0	167	0	0	1
	Keys	0	0	0	166	0	5
	Door	0	0	0	0	167	5
	Glass	0	0	0	0	0	148

Table 11: Confusion Matrix Source at 160 Degrees

Confusion Matrix							
	Detected	DOA: 20°, 185°					
Expected		Chair	Footsteps	Fan	Keys	Door	Glass
	Chair	304	0	2	0	0	33
	Footsteps	29	255	8	0	53	2
	Fan	0	0	323	0	0	1
	Keys	0	0	0	323	0	30
	Door	0	79	0	11	278	62
	Glass	0	0	0	0	2	205

Table 12: Confusion Matrix Source at 20 and 185 Degrees

Confusion Matrix							
	Detected	DOA: 50°, 190°, 300°					
Expected		Chair	Footsteps	Fan	Keys	Door	Glass
	Chair	433	0	1	0	0	73
	Footsteps	66	379	20	1	227	11
	Fan	0	0	479	0	0	0
	Keys	0	0	0	404	0	92
	Door	0	120	0	95	271	169
	Glass	1	0	0	0	2	156

Table 13: Confusion Matrix Sources at 50, 190, and 300 Degrees

4.6 Full System Results

In this section the results of the full system, or all of the blocks working together are shown. The testing in this section as in previous sections assumes the number of sound sources is known *a priori*. In Table 15 the full system test results are shown. The system is tested with one, two, and three sources. The DOAs shown in the table are the true DOAs. During the testing the DOA to each sound source is estimated, and a beamformer is opened up in each of the resulting look directions. The outputs from each of the opened beamformers are sent through the audio recognition algorithm for classification. In future work a source counting method should be implemented to make the system into a more practical application. In practice it is not likely that the number of sound sources are going to be known to the system beforehand. The Final System Specifications can be seen in Table 14.

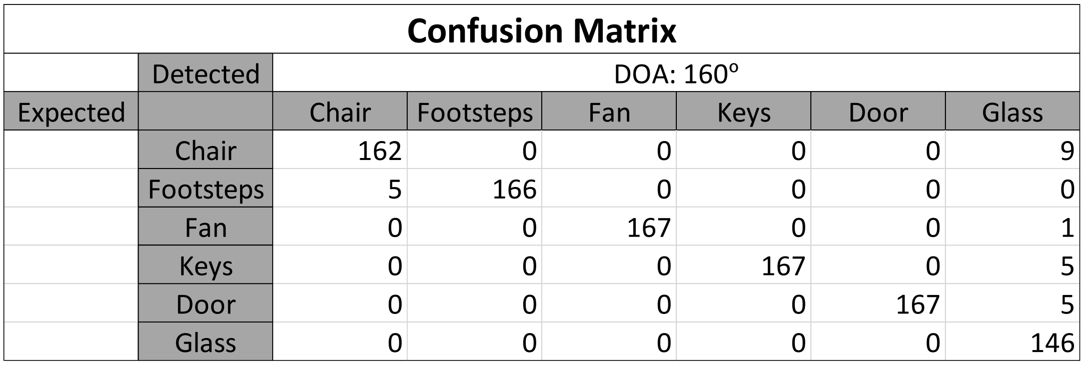
Final System Specifications	
Feature Set	SFM, ZCR, MFCCs
Number of Localization Arrays	3
Radius of Localization Arrays	2cm, 4cm, 6cm
Radius of Beamformer Array	2cm
Number of Taps	50
Number of Microphones	8

Table 14: Final System Specifications

The Final System Specifications can be seen in Table 14. In order to get more accurate audio classification even in the presence of small localization errors the beamformer radius needed to be reduced. Decreasing the beamformer radius increases beamwidth allowing for more variability in the estimated DOA from the true DOA. The Confusion Matrices from the Full System Test can be seen in Table 16 through Table 18.

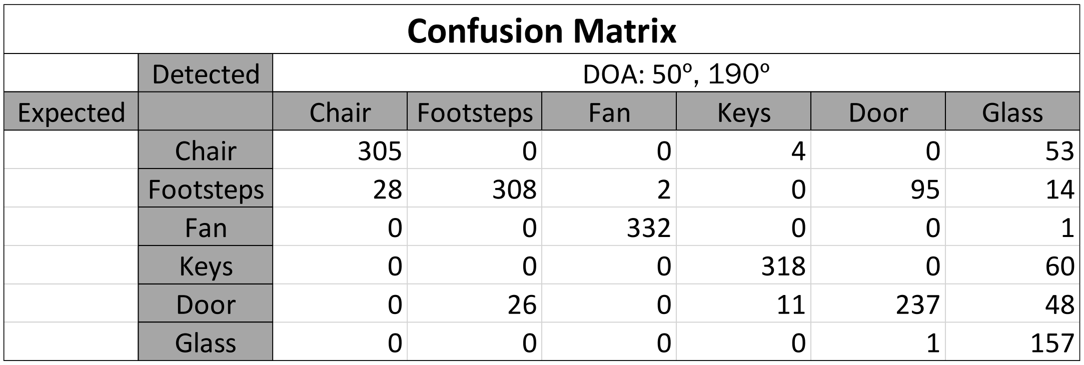
Full System Test		
Number of Sound Sources	DOA	Precision (%)
1	160°	97.4
1	300°	97.5
2	20°, 185°	78.10
2	50°, 190°	82.85
3	20°, 160°, 240°	60.70
3	50°, 190°, 300°	62.10

Table 15: Full System Test Results



Confusion Matrix							
	Detected	DOA: 160°					
Expected		Chair	Footsteps	Fan	Keys	Door	Glass
	Chair	162	0	0	0	0	9
	Footsteps	5	166	0	0	0	0
	Fan	0	0	167	0	0	1
	Keys	0	0	0	167	0	5
	Door	0	0	0	0	167	5
	Glass	0	0	0	0	0	146

Table 16: Full System Confusion Matrix DOA at 160 Degrees



Confusion Matrix							
	Detected	DOA: 50°, 190°					
Expected		Chair	Footsteps	Fan	Keys	Door	Glass
	Chair	305	0	0	4	0	53
	Footsteps	28	308	2	0	95	14
	Fan	0	0	332	0	0	1
	Keys	0	0	0	318	0	60
	Door	0	26	0	11	237	48
	Glass	0	0	0	0	1	157

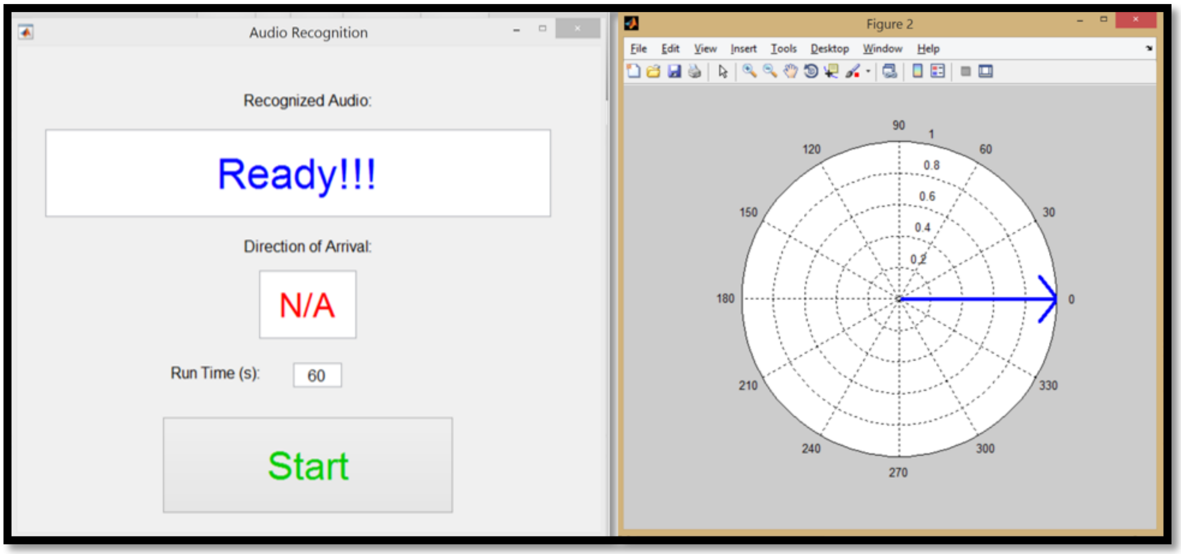
Table 17: Full System Confusion Matrix DOA at 50 and 190 Degrees

Confusion Matrix							
	Detected	DOA: 20º, 160º, 240º					
Expected		Chair	Footsteps	Fan	Keys	Door	Glass
	Chair	386	37	1	18	14	143
	Footsteps	103	369	89	9	274	69
	Fan	8	49	407	12	32	9
	Keys	0	11	0	391	3	91
	Door	2	33	2	63	174	95
	Glass	0	2	0	7	3	94

Table 18: Full System Confusion Matrix DOA at 20, 160 and 240 Degrees

4.7 GUI Development

A Graphical User Interface is useful since it adds appeal, and makes it much easier for people to grasp the main points from the work that has been done. A GUI was created for handling a single sound source that showed the Recognized Audio Class, and showed the Direction of Arrival from the sound source to the array. The single sound source GUI can be seen in Figure 43.



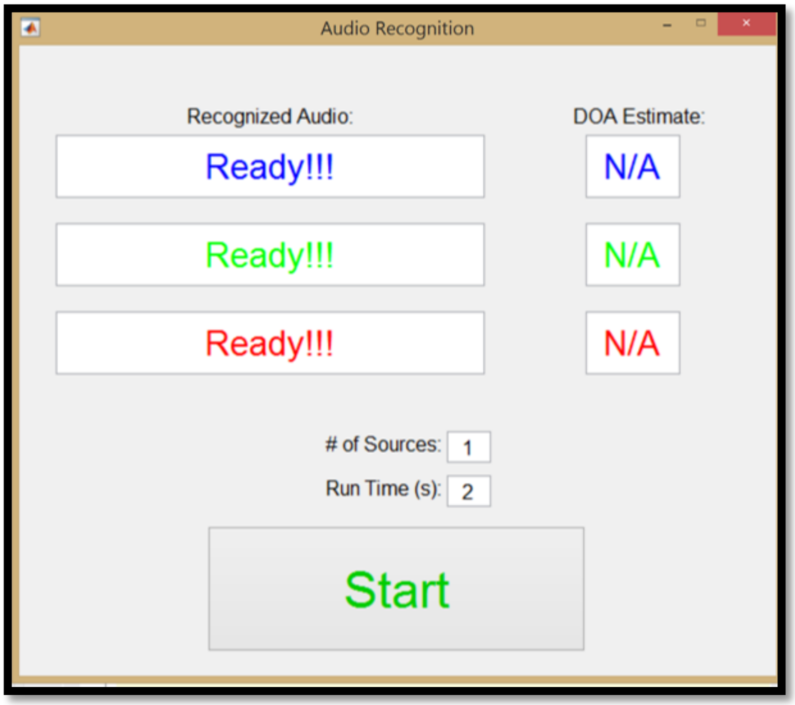
Recognized Audio: Ready!!!

Direction of Arrival: N/A

Run Time (s): 60

Figure 43: Single Source Graphical User Interface

Since the GUI that was created to handle a single sound source did not have the ability to handle multiple sound sources, an update was made that allowed for the detection and display of up to three different sound sources along with their corresponding DOAs. The GUI for handling multiple sound sources can be seen in Figure 44. Utilizing the frost beamformer for the purpose of spatial filtering uses up a lot of processing power, and currently takes much longer than one second to separate the sounds coming from a specific look direction. Future work needs to be done to speed up the frost beamformer. MATLAB offers a function that allows .mat files to be converted into c code. Doing this for the entire system should greatly increase the speed of the localization, separation, and detection. Once this is done a GUI already exists that can handle up to three sound sources.



Recognized Audio: DOA Estimate:

Ready!!! N/A

Ready!!! N/A

Ready!!! N/A

# of Sources: 1

Run Time (s): 2

Figure 44: Multiple Sound Source Graphical User Interface

For those who are interested there is a link below to some videos through different stages of testing. Some of the videos are performed in real time, while others are based on simulations. There should be a brief description of each video explaining the test being performed.

https://www.linkedin.com/pub/blair-baldridge/b1/a2a/2b2

CHAPTER V

CONCLUSION

In this work it was shown that microphone array based security monitoring can be achieved even in the presence of multiple sound sources. In the case that there is only one sound source audio recognition can be performed with just a single microphone, but in reality it is very possible that multiple sounds can appear at the same time. It is also very possible that a sound might be considered normal from a specific location, but not from another location. This leads to the need for a system that can separate simultaneously appearing sound sources, determine their location, and classify them. In order to do this a microphone array is required. Most of the work that is currently being done in the acoustic based security monitoring field is focused around classification, and recognizing the surrounding sound. Very little work has been done that outlines the ability of these types of systems to handle the classification of simultaneously appearing sound sources.

In this work there are three components to the microphone array based security monitoring method that is being proposed. Audio recognition is the primary component of any acoustic based security monitoring system. There has to be some ability to classify different sounds sources. Since it is also important to separate the sound sources before the audio is classified, a localization algorithm along with the frost beamformer were respectively set as precursors to the audio recognition algorithm.

Each of the individual blocks of the proposed method work well both individually and together as a whole system. The localization algorithm with a single source is able to maintain a RMSE (Root Mean Square Error) of around 2 degrees with a single sound source, and up to around 40 degrees with a 3 sound sources. The ME (Median Error) for a single sound source is around 2 degrees and for three sound sources only jumps to 2.5 degrees. The beamformer is able to separate individual sound sources from interfering signals while maintaining a high SIR (Signal to Interference Ratio). The audio recognition algorithm for a single source used only the MFCCs as the extracted features. This led to a precision of around 97.07%. Later test results showed that MFCCs did not hold up as well as expected in the presence of interfering sound sources. This led to the feature selection process being done again, but this time in the presence of interfering sound sources. From this it was determined that SFMs, ZCRs, and MFCCs, were the best feature set to use. The overall system with all of the blocks put together results in a precision of 97.5% for a single sound source, 82.85% for two sound sources, and 60.70% for three sound sources. The original radius that was going to be used for the full system was going to be 6cm. In the end after testing was done having a radius of 6cm did not allow for enough variability from the true DOA. In order to solve this problem the radius of the array was changed to 2cm to increase beamwidth.

The precision from the above results show that it is very possible to create a microphone array security monitoring system. Future work should focus on adding the idea of sensor fusion, to increase the precision of the system even more. Imagine a room with multiple sensors scattered throughout in a grid pattern. If every sensor knows the location of the other sensors an exact distance can be calculated to a specific sound source along with a Direction of Arrival. Another benefit would be that based off of the localization specific sensors can be chosen to perform the beamforming, and audio recognition that are closer to the sound source. The set of chosen sensors could then decide on the audio class based on multiple sensors versus the single system.

Future work should also focus on optimizing the beamformer array geometry based on the Power Spectral Density (PSD) of the incoming signal. The beamformers performance at high and low frequencies partially depends on the radius of the circular array. If there are multiple circular arrays at different radiuses it may be possible to actively change which circular array is used for beamforming based off of the incoming signals PSD.

This work should just be considered a starting point. Microphone Arrays have applications in many different fields such as Robotics, or Smart Home Design. In the robotics field there are security monitoring methods that can be improved by the use of the suggested system. A robot can be used to possibly investigate and be steered towards specific sound sources. As people become more interested in smart homes and interconnecting all of their electronic devices, a need of a system that can properly recognize the surrounding environment becomes more prevalent. This is to show that microphone arrays have future not only in security monitoring, but in other highly advantageous applications as well.

REFERENCES

[1] J. Chen, A. H. Kam, J. Zhang, N. Liu and L. Shue, "Bathroom Activity Monitoring Based on Sound," Proceedings of the Third International Conference on Pervasive Computing, Munich, Germany, 2005.

[2] A. R. Abu-El-Quran and R. A. Goubran, "Security-Monitoring using Microphone Arrays and Audio Classification," in *Instrumentation and Measurement Technology Conference*, Ottawa, Canada, 2005.

[3] L. Sun and Q. Cheng, "Indoor Sound Source Detection and Localization using a Circular Microphone Array," *IEEE/ ACM Transactions on Audio, Speech, and Language Processing.*, submitted for publication. ***

[4] Jia-Ching Wang; Hsiao-Ping Lee; Jhing-Fa Wang; Cai-Bei Lin, "Robust Environmental Sound Recognition for Home Automation," *Automation Science and Engineering, IEEE Transactions on* , vol.5, no.1, pp.25,31, Jan. 2008.

[5] R. Levorato, *GMM Classification of Environmental Sounds for Surveillance Applications,* 2010.

[6] R. G. Bachu, K. S., A. B. and B. B.D., "Separation of Voiced and Unvoiced using Zero crossing rate Energy of the Speech Signal," ASEE, 2008.

[7] V. G., M. Politecnico di Milano, G. L., T. M., A. E. and S. A., "Scream and gunshot detection and localization for audio-surveillance systems," in *Advanced Video and Signal Based Surveillance*, London, 2007.

[8] Y. Huang and J. Benesty, Audio Signal Processing for Next-Generation Multimedia Communication Systems, New York, Boston, Dordrecht, London, Moscow: Kluwer Academic Publishers, 2004, pp. 91-148.

[9] Pavlidi, D.; Griffin, A.; Puigt, M.; Mouchtaris, A., "Real-Time Multiple Sound Source Localization and Counting Using a Circular Microphone Array," *Audio, Speech, and Language Processing, IEEE Transactions on* , vol.21, no.10, pp.2193,2206, Oct. 2013.

[10] Karbasi, A.; Sugiyama, A., "A new DOA estimation method using a circular microphone array," *Signal Processing Conference, 2007 15th European* , pp.778,782, 3-7 Sept. 2007.

[11] H. Adel, M. Souad, A. Alaqeeli and A. Hamid, "Beamforming Techniques for Multichannel audio Signal Separation," Immersive Audio Signal Processing, 2012.

[12] O. L. Frost, "An Algorithm for Linearly Constrained Adaptive Array Processing," *Proceedings of the IEEE,* vol. 60, no. 8, pp. 926-935, 1972.

[13] Griffiths, Lloyd J.; Jim, C.W., "An alternative approach to linearly constrained adaptive beamforming," *Antennas and Propagation, IEEE Transactions on* , vol.30, no.1, pp.27,34, Jan 1982.

[14] Dolph, C.L., "A Current Distribution for Broadside Arrays Which Optimizes the Relationship between Beam Width and Side-Lobe Level," *Proceedings of the IRE* , vol.34, no.6, pp.335,348, June 1946.

[15] A. AlShehhi, M. L. Hammadih, M. S. Zitouni, S. AlKindi, N. Ali and L. Weruaga, *Linear and Circular Microphone Array for Remote Surveillance: Simulated Performance Analysis,* Sharjah, Unites Arab Emirates.

[16] I. A. McCowan, J. Pelecanos and S. Sridharan, "Robust Speech Recognition using Microphone Arrays," in *Proc. A Speaker Odessey- The Speaker Recognition Workshop*, 2001.

[17] G. L. Sarada, T. Nagarajan and H. A. Murthy, "Multiple frame size and multiple frame rate feature extraction for speech recognition," in *International Conference on Signal Processing and Communications*, 2004.

[18] Davis, S.; Mermelstein, P., "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *Acoustics, Speech and Signal Processing, IEEE Transactions on* , vol.28, no.4, pp.357,366, Aug 1980.

[19] Lyon, R.F.; Mead, C., "An analog electronic cochlea," *Acoustics, Speech and Signal Processing, IEEE Transactions on* , vol.36, no.7, pp.1119,1134, Jul 1988.

[20] A. Hossan, S. Momon and M. A. Gregory, "A Novel Approach for MFCC Feature Extraction," in *International Conference on Signal Processing and Communication Systems (ICSPCS)*, Gold Coast, QLD, 2010.

[21] N. Ahmed, T. Natarajan and K. R. Rao, "Discrete Cosine Transform," *IEEE Transactions on Computers,* Vols. C-23, no. 1, 1974.

[22] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project," Paris, France, 2004.

[23] D. Ning, "Developing an Isolated Word Recognition System in MATLAB," *MATLAB Digest*, pp. 1-6, 2009.

[24] M. A. Rodriguez, "Implementation of Gaussian Mixture Models in .Net technology for automatic speech recognition," 2011.

[25] M.-C. A., Q.-D. J., C.-J. M.G. and A. D., "Feature Selection Using Sequential Forward Selection and Classification applying Artificial Metaplasticity Neural Network," in *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, Glendale, AZ, 2010.

[26] J. F. Rudolf, J. W. William and L. M. Donna, Statistical Methods, 3rd ed., Burlington, MA; San Diego, CA; London, UK: Elsevier, Inc., 2010.

[27] L. J. Griffiths, "A Simple Adaptive Algorithm for Real-Time Processing in Antenna Arrays," in *PROCEEDINGS OF THE IEEE*, 1969.

VITA

Blair Armand Baldridge

Candidate for the Degree of

Master of Science

Thesis: MICROPHONE ARRAY BASED SURVEILLANCE SYSTEM

Major Field: Electrical Engineering

Biographical:

Education:

Completed the requirements for the Master of Science in your major at Oklahoma State University, Stillwater, Oklahoma in December, July, 2015.

Completed the requirements for the Bachelor of Science in Computer Engineering at Oklahoma State University, Stillwater, Oklahoma in May, 2013.

Experience:
Engineering Intern, Boeing, Oklahoma City, OK, 05/2013 – 08/2013
Hardware Engineering Intern, National Instruments, Austin, TX, 06/2012 – 08/2012

Professional Memberships:

Institute of Electrical and Electronics Engineers (IEEE)
Eta Kappu Nu Honor Society (HKN)