Kalman Filtering for Audiovisual Speaker Tracking

Akshita Sharma

Department of Mathematics

College of Letters and Science

University of California, Davis

UNDERGRADUATE THESIS

Bachelor of Science in Applied Mathematics

July, 2021

Advisors: Timothy J. Lewis & Lee M. Miller

Abstract

In this work. we address the problem of audiovisual speaker tracking. We model the tracking problem as a system of linear equations and follow an approach which involves the incorporation of stream weights into the conventional Kalman filtering paradigm. Initially. fixed stream weights are used to define a strategy to methodologically incorporate information from both acoustic and visual sensory modalities. That approach is then augmented and optimized to allow for the incorporation of dynamic stream weights into the conventional multimodal Kalman filter. We present all aforementioned methods along with simulations and results of our programs. Our results show that the adopted techniques efficiently and effectively combine acoustic information with visual cues to perform speaker tracking and that the method of incorporating stream weights outperforms the tracking capabilities of the conventional Kalman Filter.

Contents

1 Introduction

To be able to make progress in the field of Auditory Neuroscience, we must steer our attention towards an unsolved problem in the discipline today. In this paper, we focus on an important aspect of automatic speech recognition (ASR). Healthy individuals are remarkably good at focusing their attention to one source amidst noisy and crowded environments by simply suppressing all other sounds and distractions. This phenomenon, known as the cocktail party effect, comes naturally to well abled individuals, but poses a major hindrance to those with impaired hearing capabilities. While being a widely studied phenomenon, still poses a major challenge to computers and therefore projects itself as a major field of research. To address this challenge, we consider an innovative method to amplifying attention to a specific source, an effect that is the byproduct of muting background noise.

The purpose of ASR is to convert audio data into data formats that, in their most sophisticated variations, resemble natural human conversation. It seems appropriate to rely on audio information to devise technology to successfully accomplish ASR. Given this restricted type of information, ASR technology will be limited in its capabilities. For instance, any movement of the source (speaker) or the sensor (listener) or any background noise will introduce a hindrance into the system. To overcome this issue, we introduce another dimension into our ASR problem solving strategy; namely, we add a visual component and thus motivate Audio-Visual Automatic Speech Recognition (AV-ASR).

We work to achieve audiovisual speaker tracking. This involves an appropriate manipulation of both audio and visual observations using the multimodal Kalman filter. We follow the approach presented in [14] and extend the strategies presented in this study to evaluate the prediction capability of the Kalman Filter to perform speaker tracking in a multi-talker environment.

This thesis is organised as follows. In chapter 2, we introduce the concept of filtering, present the unimodal Kalman Filter, and go through an example demonstrating the key concepts involved in the Kalman filtering paradigm. In the next chapter we proceed to define the problem under consideration, define the specifications of our proposed solution, and derive the the equations making up the multimodal Kalman Filter. Then, we introduce some beamforming and filtering algorithms that we incorporate into our program, present simulations of our program and discuss important results. Finally, in chapter 5, we provide concluding remarks and discuss possible directions for future research in this field of study.

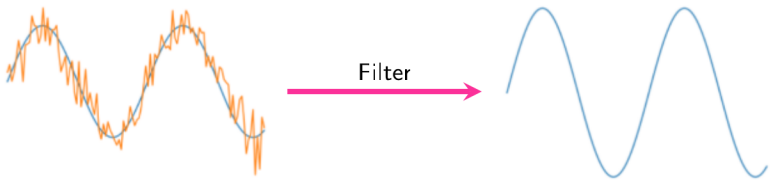
2 Background

2.1 Introduction to Filtering

Filtering is used in a variety of disciplines from pure mathematics to applied computer science. The intuition behind filtering can be understood by studying the mechanics of a water filter which removes impurities from water to make it safe to consume. Similarly, we can also consider the logistics of an air purifier which works by trapping dust and dirt particles to expel cleaner air. In both these examples, the filtration processes work to remove impurities

from raw components. This is the underlying principle of filtering: completely or partially suppressing unwanted components from an input stream of data.

Signal Processing is extensively used in almost all fields. It is employed in health care for its uses in X-rays, MRIs, and CT scans. In finance, signal processing is used for interpreting financial data, and decision-making in trading among other things. The entertainment industry uses it for multiresolution signal processing and then of course there is the ever-dynamic consumer electronic industry in which you get video games, smartphones, and other AI technologies [1]. Even though the applications of signal processing are quite varied, the underlying motivation among all its implementations stems from the same idea. Signal processing technology is used whenever there is a need to measure, amplify, suppress, or filter audio or other types of signals. When signals are picked up or transmitted in noisy environments or over long distances, they can become corrupted. This corruption or 'noise' that adds on to the signals makes it difficult to analyse and extract regions of the signals that we are actually interested in. So how do we extract a clean signal from one that is submerged in noise?



(a) Noisy Signal (b) Clean Signal

Figure 1: Filtering a noisy signal

2.1.1 Filtering Theory

The objective of filtering is to determine the values of the unobservable states of the system at time k, x_k, through studying the system's measurable output, z_k, which is influenced by the aforementioned unobservable values. There are generally three situations in which we would employ filtering. Firstly, we may use filtering simply because we're interested in estimating the state of the system. This is usually the case when the state of the system can be decomposed to give us valuable information of the physical quantities involved in the determination of z_k (e.g. position or velocity). Secondly, as is the case discussed in this thesis, we could be interested in the filtered outputs if we want to control a system described by a state-space model. In this case, we use state feedback controls which take the form $u_k = u(k, x_k)$, where x_k is determined by the systems estimate of the parameter at time k. Lastly, we may be interested in filtering if we want to replace the standard state-space model with a more appropriate external model, such as ARMAX [8].

A general filtering problem may be described as follows: given a set of observations $[Y_1, Y_2, \ldots, Y_n]$, we are interested in estimating the value of Y_0 to get the state of the system $\bar{Y}^T = [Y_0, Y_1, Y_2, \ldots, Y_n]$ such that we minimize the mean squared error $\mathbb{E}[Y_0 - g(Y)]^2$ where $g(Y)$ is the conditional expectation $\int_{-\infty}^{\infty} y_0 dF_{y_0\, y}(y_0\, y)$ and F is the joint distribution function of Y^T. For practicality, instead of the integral expression for the conditional expectation, we switch to a linear estimation problem $g(Y) = \alpha_1 Y_1 + \alpha_2 Y_2 + \ldots + \alpha_n Y_n$. This way the linear

estimation problem is reduced to finding the values $\alpha^T = [\alpha_1, \alpha_2, ..., \alpha_n]$ which minimize $\mathbb{E}[Y_0 - g(Y)]^2 = E\sum_{i,j=0}^{n} \alpha_i \alpha_j Y_i Y_j$ [12].

As mentioned above and as we see here, the values of the state of the systems at time $k+1$, x_{k+1}, aren't usually known beforehand. They are estimated using the values $x_1, x_2, ..., x_k$ and Y_{k+1} before they are fed into the state feedback control functions. For clarity, the variables z_k, x_k, and y_k in this thesis refer to the predicted position of the object being tracked and measured values of the visual and acoustic information at time k respectively. The challenge with this approach is that performing these estimations would require computations with larger and larger covariance matrices increasing the computational complexity of these problems. To overcome this issue, we instead use the recursion relation $Y_{0,n+1} = a_n Y_{0,n} + b_n Y_{n+1}$. That is, we determine the value of the state at the present time step using values of only the two previous time steps. This is one of the building blocks of the Kalman filtering paradigm [8].

2.2 Introduction to Kalman Filtering

Kalman filtering is an algorithm that uses a series of measurements over time to provide estimates of some unknown variables that are more accurate than the predictions made of those variables using a single measurement alone. The algorithm makes these estimates by approximating joint probability distributions over all involved variables at every time step. This technique is especially useful in applications in which there is a delay between data creation and information feedback.

2.2.1 Single-Sensor Tracking Using Kalman Filter

The Kalman filter is an excellent tool to measure predicted values. However, it can be challenging to understand the ins and outs of the algorithm without looking at a worked example. Let's begin by building up some intuition around the 1D Kalman filter.

The Kalman filter is an algorithm that uses a set of iterative processes and recursive relations that work with a continuous stream of input data to quickly and increasingly accurately estimate the true state (e.g., position, velocity, etc.) of the system being measured when the measurement values include random noise, errors, variations, and uncertainty. One way to obtain these estimations is gather large amounts of data, run a statistical distribution over them, find the average value, and claim that the obtained average must be close to the true value of the measured object. What the Kalman filter does instead is, rather than waiting for sufficient data to be gathered and averaged, it starts making predictions from the very first batch of information it receives, and it perpetually converges its estimates towards the true value by understanding the uncertainties in the data inputs [7].

Consider the graph in Figure 2:

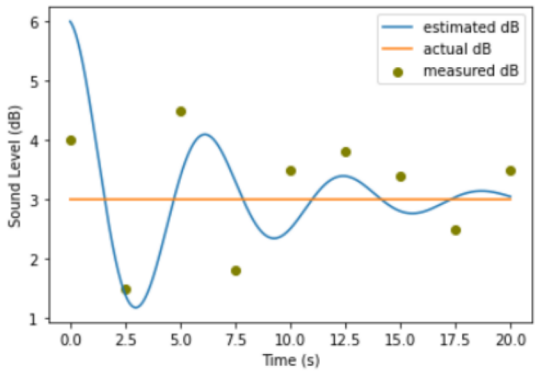


Figure 2: Kalman filter in action

Suppose we are trying to estimate the volume of a constant sound being emitted by a distant object. Due to disturbances in the environment, noise picked up the sensor, etc. the measurement values aren't as accurate. In the graph above, the orange line represents the true sound level being emitted by object, the green dots represent the measurement values picked up by a decibel meter, and the blue curve represents the Kalman filter's estimate of the true sound levels at every time frame. As we can see, the Kalman filter begins by making an initial estimate at the first time step. In addition to making that estimate, it also calculates an error term to quantify how incorrect its current estimate is. Then as more and more data points are received by the algorithm and more predictions and error terms are calculated by the Kalman filter, we can see from the blue curve how quickly the Kalman filter is able to use its structure to narrow in its estimates to make its predictions of the sound level of the emitted noise with decreasing deviations from the actual sound level.

2.2.2 Kalman Filter Autonomous Robot Example

Before we dive into an example, it is important to clarify some terminology:

- State propagation: State propagation defines how a propagated system follows its dynamics to move from one state to another before measurement values are factored in to make a final prediction on the state of a system
- Apriori estimates: Apriori estimates are the values computed during the state propagation process
- Posterior estimates: Posterior estimates are the predicted values we get once by combining apriori estimates and measurement values
- Error covariance: Error covariance matrices reflect the errors between the predicted states and the true values. They are used to help the Kalman Filter tune its predictions for future time steps

At each time step, the Kalman filter uses its state propagation system to calculate apriori estimates of the state of the system. Once that's done, the apriori estimate is combined with a measurement value to produce a posterior estimate. Upon every calculation of a posterior estimate, a new error covariance matrix is calculated to quantify the error between the

present prediction and true value of the system. This covariance matrix is used to improve the prediction performance of the Kalman filter algorithm over future time steps [17].

Let's see this algorithm in action using an example [10, 15].

Suppose we are designing an autonomous robot. All autonomous machinery would need to have the ability to perform continuous and interruptible tasks such as moving forward, making turns, dodging obstacles, etc. Therefore, a necessary feature of this autobot would be its ability to perform continuous self-localisation. To enable this device to perform self-localisation we can use the Kalman filter as it can assist in perpetually tracking and predicting the state of our system.

To be able to perform tracking using the Kalman filter we need at least two sources of measurement values. For our robot, suppose those two sources of information give the probability distributions of the possible values of position and velocity of the body at a particular time step in the graphs in figure 3.

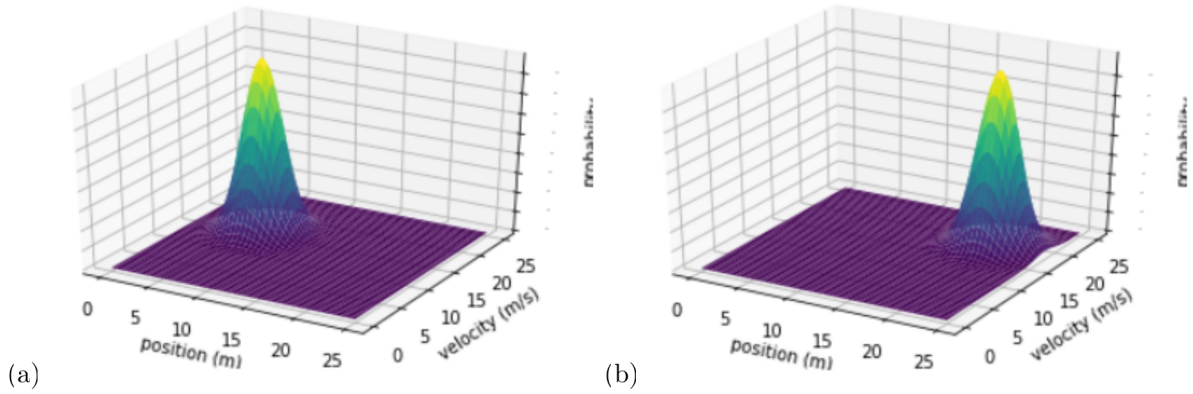


Figure 3: Probability distributions of position and velocity of a moving object

As we can see from both these graphs, both sources give slightly different estimates of what the robot's position and velocity would be at a particular time step. Instead of having to blindly pick which option we want to proceed with, we want a more comprehensive approach for making an estimate of the state of the system.

The basic idea of the unimodal Kalman filter is to combine or take the weighted sum of both probability distributions, as in figure 4. From the above graph we can observe that the combined probability distribution curve has a higher peak and a narrower width implying that the prediction returned from the combined probabilities is more reliable than using the probabilities returned from either of the sources alone.

The overall algorithm for the Kalman filter can be explained in two steps:

1. Predict: Obtain apriori estimates using state propagation
2. Update: Combine the apriori estimates with a measurement input to find posterior estimates which are the predictions of the state of the system

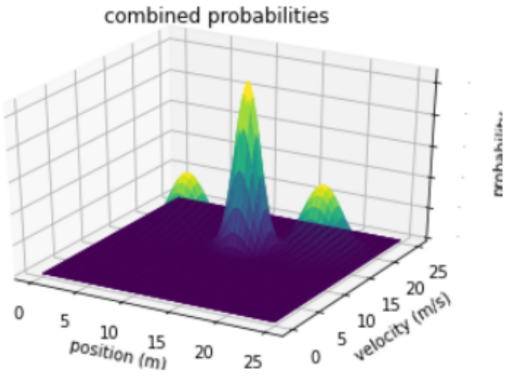


Figure 4: Combined probability distributions

Now that we have a basic idea of what the Kalman filter is doing we can define the motion models and derive the equations that define the Kalman filtering paradigm.

Given the state of the system $x_k = \begin{bmatrix} p_k \\ v_k \end{bmatrix}$, where p_k and v_k are the position and velocity at time k, and state to observation matrix $\mathbf{C}$, we can define the state transition model to be

$$p_k = p_{k-1} + v_{k-1}\Delta t$$

$$v_k = v_{k-1}$$

Rewriting this system in matrix form we get $x_k = \mathbf{A}x_{k-1}$ where and $\mathbf{A}$ is the state transition matrix

$$A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

To make our model more realistic, we must also add some process noise, $w_k \sim \mathcal{N}(0, Q)$, to our system.

Altogether, our constant velocity model looks like the following:

$$p_k = p_{k-1} + w_k$$

$$v_k = \mathbf{C}x_k + \omega_k$$

In the above expression, $\omega_k \sim \mathcal{N}(0, R)$.

Using this we can also create a similar model for our measurement inputs:

$$\acute{x}_k = \mathbf{A}\acute{x}_{k-1}$$

$$\acute{y}_k = \mathbf{C}\acute{x}_k$$

In the above equations $\acute{x}_k$ and $\acute{y}_k$ are the state of the system and the measurements made at time k respectively and $\mathbf{C}$ is a state to observation matrix. All other variables are defined as described above.

To quantify the error between the estimates found from the motion and measurement models where y is the known measurement at every time step, $y - \acute{y}$, and use it to progressively improve the estimates of the Kalman filter predictions, we introduce the Kalman Gain equation, which is given by

$$\mathbf{k}_k = \frac{P_k C^T}{C P_k C^T + R}$$

where P_k is the covariance matrix of the estimated state x_k and R represents the covariance matrix of the noise that stems from the measurement models. A complete derivation of the Kalman Gain equation can be found in [4].

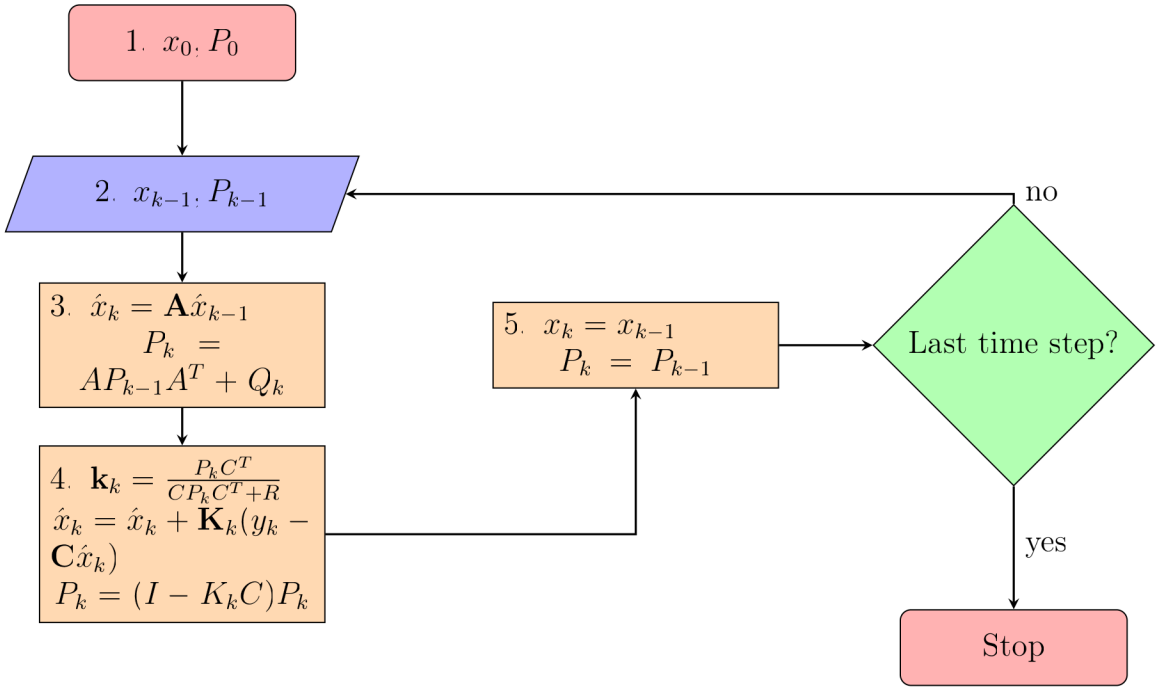
Finally, the prediction and update steps can be summarised more appropriately as the following:

1. Predict:
   $\acute{x}_k = \mathbf{A}\acute{x}_{k-1}$
   $P_k = AP_{k-1}A^T + Q_k$
2. Update:
   $\mathbf{K}_k = \frac{P_k C^T}{C P_k C^T + R_k}$
   $\acute{x}_k = \acute{x}_k + \mathbf{K}_k(y_k - \mathbf{C}\acute{x}_k)$
   $P_k = (I - K_k\mathbf{C})P_k$

All the variables above are as defined in the example presented in this section.

2.2.3 Single-Sensor Kalman Filter Flowchart

The single-sensor Kalman filter iterative algorithm can be encapsulated using the following flowchart.



1. Initial approximation of system, including covariance matrix
2. State of system with covariance matrix at time $k-1$
3. Use state transition model to predict new state of the system with covariance matrix at time k
4. Incorporate measurements in optimal way
5. Update prediction and time values

3 Methods

3.1 System Description

So far we have described the Kalman filter. In certain situations, however, prediction of the state of a system can be done using the input from more than one sensory modality (measurement). The goal of enabling those systems to perform sensor fusion is to reduce uncertainty in the prediction model compared to when any one of the sensors is used individually. Here, we are approaching the problem of audiovisual speaker tracking. Audiovisual speaker tracking is the calculated integration of both acoustic and visual cues to track the location of a mobile speaker. Automatic Speech Recognition using multimodal sensor fusion is a widely studied area. Location information from such systems can be used to track the movement of people and vehicles and even be used to augment the capabilities of those with unhealthy hearing in noisy and reverberant conditions.

The audiovisual source localisation problem can be broken down into the several sub problems, each of which is explained below:

1. Define and Understand the Dynamics of the Acoustic and Visual Models:
   To be able to perform perpetual object tracking using acoustic and visual information, it is important to understand the dynamics of how sound and visual cues propagate from source to sensor.
2. Weigh the Audio and Visual Observations Based on their Time Dependent Reliability:
   To be able to optimize the performance of our prediction model, there needs to be a well defined algorithm that we use to guide the multi-sensor information fusion.
3. Use the Multimodal Kalman Filter to Perform Target Tracking:

Finally, it is necessary to find methods that encapsulate all the steps mentioned above.

3.1.1 Defining the Multi-Sensory System

To define the model, we follow along with the work presented in [14].

The linear dynamical system that models the audiovisual speaker tracking task can be described as follows:

$$x_k = A_k x_{k-1} + v_k \tag{1}$$

$$y_{A,k} = C_{A,k} x_k + w_{A,k} \tag{2}$$

$$y_{V,k} = C_{V,k} x_k + w_{V,k} \tag{3}$$

Here, (1) represents the motion model and (2),(3) represent the independent acoustic and visual measurement models. (1) is the model representing the motion of the speaker and it is assumed to be Brownian random motion. (2) and (3) are the linear transformations of the state plus zero-mean Gaussian noise. $x_k \in \mathbb{R}^N$ is the state of the system at time k and $y_{A,k} \in \mathbb{R}^{M_A}$, $y_{V,k} \in \mathbb{R}^{M_V}$ are the conditionally independent acoustic and visual observations. $A_k \in \mathbb{R}^{NxN}$ is the state transition matrix and $C_{A,k} \in \mathbb{R}^{M_A xN}$, $C_{V,k} \in \mathbb{R}^{M_V xN}$ are the state to observation matrices as explained in the example above. Finally, $v_k \sim \mathcal{N}(0, Q_k)$, $Q_k \in \mathbb{R}^{NxN}$, $\omega_{A,k} \sim \mathcal{N}(0, R_{A,k})$, $R_{A,k} \in \mathbb{R}^{M_A xM_A}$, and $\omega_{V,k} \sim \mathcal{N}(0, R_{V,k})$, $R_{V,k} \in \mathbb{R}^{M_V xM_V}$ are the zero-mean Gaussian noise matrices.

We can translate these equations into a script that allows us to observe the evolution of the state of the system over time for a particular simulation. An example of this is shown in figure 5.

Let the state of the system be given by $x_k = \begin{bmatrix} p_{x_k} \\ v_{x_k} \\ p_{y_k} \\ v_{y_k} \end{bmatrix}$ and let $y_{A,k}$ and $y_{V,k}$ be the variables representing the acoustic and visual observations at time k.

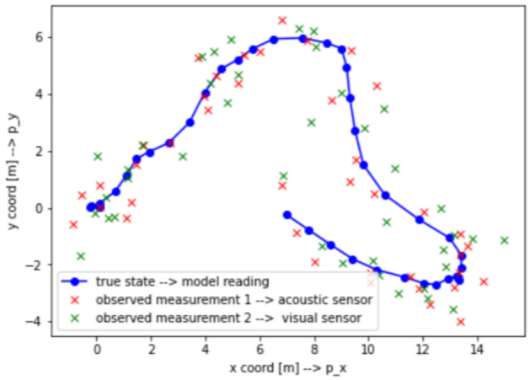


Figure 5: Evolution of the state of a system over time

Assuming Markovian System Dynamics, the state transition probability can be represented as

$$p(x_k \; x_0, \ldots, x_{k-1}) = p(x_k \; x_{k-1})$$

To factor in reliability of each of the contributions from the acoustic and visual sensors at each time step, we can write the above probability more explicitly as

$$p(y_{A,k}, y_{V,k} \; x_0, \ldots, x_k) \propto p(y_{A,k} \; x_k)^{\lambda_k} p(y_{V,k} \; x_k)^{1-\lambda_k}$$

where $\lambda_k \in [0, 1]$ is a dynamic stream weight. λ_k enables our system to put more emphasis on the dominant sensory contribution at each time step.

As we discussed earlier, we will be using the the multimodal Kalman filter to perform speaker tracking. Therefore, we know that the solution to this state estimation problem will follow a two step approach:

1. Letting $Y_{A,k}$ and $Y_{V,k}$ represent sequences of audio and visual observations, the ultimate probability distribution or the expression that will find the posterior estimates of the location of the speaker is given by

$$p(x_k \; Y_{A_{k-1}}, Y_{V_{k-1}}) = \int p(x_k \; x_{k-1}) p(x_{k-1} \; Y_{A_{k-1}}, Y_{V_{k-1}}) dx_{k-1}$$

2. The corresponding update step is given by

$$p(x_k \; Y_{A,k}, Y_{V,k}) \propto p(y_{A,k} \; x_k)^{\lambda_k} p(y_{V,k} \; x_k)^{1-\lambda_k} p(x_k \; Y_{A,k-1}, Y_{V,k-1}) dx_{k-1}$$

Since all the probabilities are Gaussian distributed, we can rewrite the linear dynamical system described in eqs. (1)-(3) as follows:

$$\begin{aligned} p(x_k \; x_{k-1}) &\sim \mathcal{N}(A_k x_{k-1}, Q_k) \\ p(y_{A,k} \; x_k) &\sim \mathcal{N}(C_{A,k} x_{k-1}, R_{A,k}) \\ p(y_{V,k} \; x_k) &\sim \mathcal{N}(C_{V,k} x_{k-1}, R_{V,k}) \\ p(x_k \; Y_{A,k-1}, Y_{V,k-1}) &\sim \mathcal{N}(\hat{x}_{k-1}, \hat{\Sigma}_{k-1}) \end{aligned}$$

where $\hat{x}_{k-1}, \hat{\Sigma}_{k-1}$ are the posterior estimates of the state and covariance matrices at the previous time step.

3.1.1.1 Dynamic Stream Weights

Thus far, we have introduced the idea of incorporating an additional sensory modality to boost the performance of a conventional audio based speaker tracking algorithm. To overcome the difficulties present when performing automatic speech recognition in noisy and reverberant conditions, as acoustic and visual cues are conditionally independent of each other, we study how the addition of a visual modality can enhance the performance of a speaker tracking algorithm.

Once way to effectively incorporate the contributions of more than one sensory modality is through the use of dynamic stream weights. In the description of the system above we introduced the concept of dynamic stream weights. In this section, we closely follow the ideas presented in [5] to describe a framework the determine value of the stream weight, λ_k, at every iteration of the speaker localisation program.

To find an expression that calculates the dynamic stream weights, we need to solve the following equation:

$$\lambda^* = \arg\max_{\lambda \in \mathbb{R}} p(Y_A, Y_V, \lambda \; X).$$

That is, we need to find the value of λ which maximizes the value of $p(Y_A, Y_V, \lambda \; X)$.

Using Bayes' Theorem we get the following

$$\lambda^* = \arg\max_{\lambda \in \mathbb{R}} p(Y_A, Y_v, \lambda \; X) = \arg\max_{\lambda \in \mathbb{R}} \frac{p(x \; Y_A, Y_v, \lambda) p(Y_A) p(Y_V) p(\lambda)}{p(X)}.$$

Simplifying we get

$$\frac{p(X \cap (Y_A \cap Y_V \cap \lambda))}{p(Y_A) \cap p(Y_V) \cap p(\lambda)} * \frac{p(Y_A)p(Y_V)p(\lambda)}{p(X)} = \frac{p(X) \cap p(Y_A) \cap p(Y_v) \cap p(\lambda)}{p(X)}.$$

Therefore, what we essentially have is

$$p(Y_A) \cap p(Y_V) \cap p(\lambda).$$

Since Y_A, Y_V, X are sequences of observations and λ is a sequences of dynamic stream weights we get

$$\lambda^* = \arg\max_{\lambda \in \mathbb{R}} \prod_{k=1}^{k} p(\lambda_k) \prod_{k=1}^{k} p(Y_{A,k\ x_k})^{\ \lambda_k}\ p(Y_{V,k\ x_k})^{1-\lambda_k}$$

$$= \arg\max_{\lambda \in \mathbb{R}} \sum_{k=1}^{k} \log(p(\lambda_k)) + \sum_{k=1}^{k} \lambda_k\ \log(p(Y_{A,k}\ x_k)) + (1-\lambda_k)\ \log(p(Y_{V,k}\ x_k)).$$

To be able to find the value of λ, we first need a prior distribution function that represents our belief of the behaviour of λ before te information from the next time step is taken into account.

Since all our probabilities are taking the form of a Gaussian distribution, we can assume a Gaussian prior here:

$$p(\lambda_k) = \frac{1}{\sqrt{(}2\pi\sigma^2)} * e^{\frac{-1}{2}\frac{(\lambda_k - \mu_k)^2}{\sigma_k^2}}$$

$$\lambda^* = \arg\max_{\lambda \in \mathbb{R}} \sum_{k=1}^{k} \log(\frac{1}{\sqrt{(}2\pi\sigma^2)} * e^{\frac{-1}{2}\frac{(\lambda_k - \mu_k)^2}{\sigma_k^2}}) + \sum_{k=1}^{k} \lambda_k \log(p(Y_{A,k}\ x_k)) + (1-\lambda_k) \log(p(Y_{V,k}\ x_k)).$$

Taking the derivative of this function with respect to λ and solving for λ_k, we get the following expression to find the dynamic stream weights

$$0 = \frac{\mu_k}{\sigma_k^2} - \frac{\lambda_k}{\sigma_k^2} + \frac{\log(p(Y_{A,k}\ x_k))}{\log(p(Y_{V,k}\ x_k))}$$

$$\lambda_k = \mu_k + \sigma_k^2 [\frac{\log(p(Y_{A,k}\ x_k))}{\log(p(Y_{V,k}\ x_k))}].$$

Lastly, we finalise our stream weight values by truncating them such that each stream weight lies between 0 and 1.

It is worth mentioning that [5] includes a detailed explanation of the methods used to find the optimal values of μ and λ. However, for the purposes of this study, these hyperparameters were fixed such that the prediction algorithm converges to optimal estimates of the speakers location at a reasonable rate.

3.1.2 Multimodal Kalman Filter Equations

The last piece of the puzzle, before we can proceed to running simulations of the multimodal Kalman filter, is deriving the prediction and update equations of this variant of the Kalman filtering paradigm. To do this we follow and adapt the derivation of the unimodal Kalman filter equations in [16].

Since the number of sensory inputs we incorporate have to affect the way the motion model returns the apriori predictions of the state of the system, it is easy to see why they stay the predict equations stay the same as those of the unimodal Kalman filter algorithm:

Predict:
$\hat{x}_{k\ k-1} = \mathbf{A}_k \hat{x}_{k-1}$
$\hat{P}_{k\ k-1} = A_k \hat{P}_{k-1} A_k^T + Q_k$

To find the update equations we must recall the multivariate Gaussian distribution equation

$$p(x_0) = det(2\pi\Sigma_0)^{-\frac{1}{2}}\ exp\{-\frac{1}{2}(x_0 - \mu_0)^T \Sigma_0^{-1}(x_0 - \mu_0)\},$$

and the integral expression derived in [11] that gives us the apriori estimate of the state of the system

$$\bar{bel}(x_t) = \eta \int \underbrace{exp\{-\frac{1}{2}(x_k - \hat{x}_{k\ k-1})\Sigma_k^{-1}(x_k - \hat{x}_{k\ k-1})\}}_{L_t}.$$

We can adjust L_t to represent the inclusion of an additional sensory modality by adapting it according to the probability distribution functions derived in section 2.1.1

$$\begin{aligned} L_t = \lambda_k (y_{A,k} - C_{A,k}x_k)^T R_{A,k}^{-1}(y_{A,k} - C_{A,k}x_k) + \\ (1-\lambda_k)(y_{V,k} - C_{A,k}x_k)^T R_{V,k}^{-1}(y_{V,k} - C_{V,k}x_k) + \\ (x_k - \hat{x}_{k\ k-1})^T \Sigma_{k\ k-1}^{-1}(x_k - \hat{x}_{k\ k-1}). \end{aligned}$$

The first derivative of L_t gives us the state equation

$$\Sigma_{k\ k-1}^{-1}(x_k - \hat{x}_{k\ k-1}) - \lambda_k C_{A,k}^T R_{A,k}^{-1}(y_{A,k} - C_{A.k}\hat{x_k}) - (1-\lambda_k) C_{V,k}^T R_{A,k}^{-1}(y_{V,k} - C_{V,k}\hat{x_k})$$

The inverse of the second derivative gives us the covariance matrix:

$$\hat{\Sigma_k} = (I - \lambda_k K_{A,k} C_{A,k} - (1-\lambda_k) K_{V,k} C_{V,k}) \hat{\Sigma}_{k\ k-1}$$

Similarly, we can extended the unimodal Kalman filter's equation for the Kalman Gain, $k_t = \Sigma_t C_t^T Q_t^{-1}$, to find the Kalman Gain equation for the multimodal version of the algorithm

$$K_{A,k} = \hat{\Sigma_k} C_{A,k}^T R_{A,k}^{-1}$$

$$K_{V,k} = \hat{\Sigma_k} C_{V,k}^T R_{V,k}^{-1}$$

Now we are ready to run some simulations!

4 Determining the Contributions from Each Sensory Modality

Interest in speaker localisation and tracking applications has soared over the past few years. From classroom aids, to assistant AIs, the uses of an array of acoustic sensors to perform this task have become quite versatile in this field of research. This has driven the need to develop increasingly sophisticated algorithms to deal with problems that arise in speech recognition in noisy and reverberant conditions.

In this project, we continue in the footsteps of those developing algorithms to achieve efficient speaker tracking by presenting a framework which does so by picking up cues from a combination of acoustic and visual sensors. In essence, we are introducing a structured solution to the problem of audiovisual speaker tracking using a data fusion algorithm that is commanded by a multisensor Kalman filtering approach. So far we have defined the multisensory system, derived a function to calculate dynamic stream weights, and presented the multimodal Kalman filter which can return its own optimal prediction of the speaker location based on fused audio and visual signals. The questions that remain are the following: What is sensor fusion and does it help in the design of our system?

Sensor fusion is the art of combining inputs from multiple information acquisition devices such as microphones, cameras, radars, lidars, etc. to recreate an accuracte depiction of the environment around the receiver. Fused sensor models tend to perform better than their single sensored counter parts because fused models are able to exploit the strenghts of different sensors. Each sensory modality has its own set of advantages and drawbacks. For example, microphones do well in picking up incoming sounds from all directions but have no ability to pick up any type of visual cues. On the other hand, cameras are strong at reporting the usefullness of a visual signal at any instant but aren't capable of giving any indication of the quality of a sound signal. In this project, we use software algorithms to fuse the acoustic and visual sensory modalities to improve the predictive capabilities of the multimodal Kalman filter for audiovisual speaker tracking.

4.1 Sensor Fusion with Fixed Stream Weights

One method to fuse both sensors is to simply weigh their contributions equally at each time step. That implies setting the stream weight, λ, equal to 0.5 for the entirety of the simulation. We tested the efficacy of this method by comparing it against simulations in which we performed sensor fusion by alternatively setting stream weights of either 0 or 1 at each time step of the simulations.

Table 1: RMSE Values

	$\lambda_1 > \lambda_2$	$\lambda_1 = \lambda_2$	$\lambda_1 < \lambda_2$
$\lambda = 0.5$	0.50749	0.57850	0.51609
$\lambda \in \{0, 1\}$	0.85986	0.98227	0.86495

In the graphs above, we see a model simulation depicting the differences in prediction accuracy between using a fixed stream weight of 0.5 and using an alternatively defined method

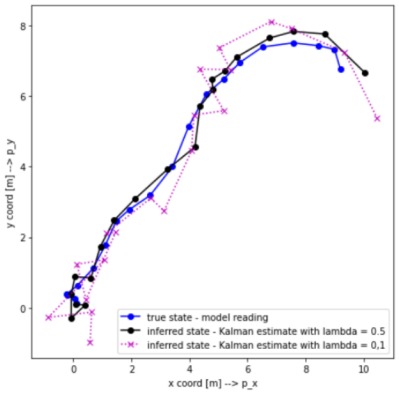


Figure 6: $\lambda_1 > \lambda_2$

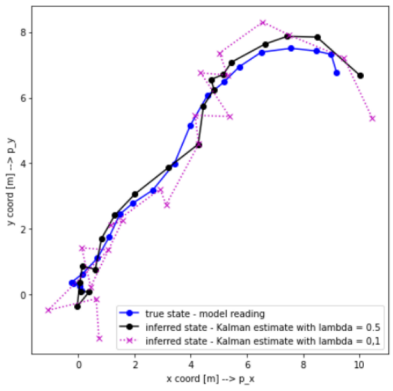


Figure 7: $\lambda_1 = \lambda_2$

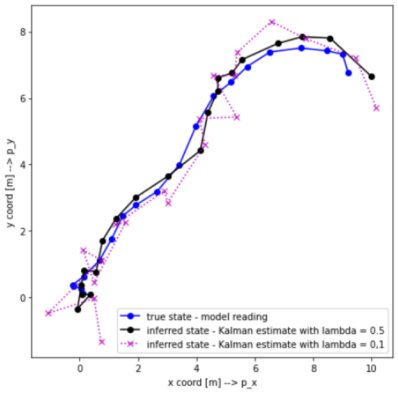


Figure 8: $\lambda_1 < \lambda_2$

of accommodating contributions from both sensory modalities. Note here that λ_1 and λ_2 represent the amount of zero-mean Gaussian noise added to the data received by each sensor respectively.

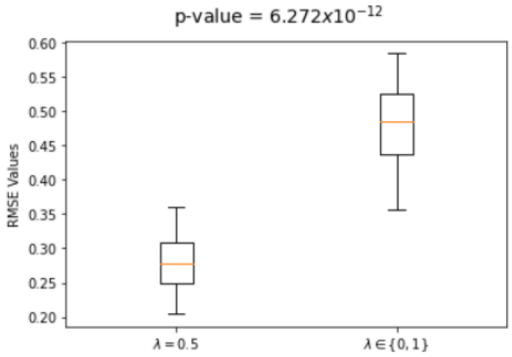


Figure 9: Statistically significant improvement seen after incorporating stream weights

From the boxplot above, it is easy to see that the performance of the multimodal Kalman filter is significantly better while using a stream weight of 0.5. The performance improvements were statistically verified using a t-test with $p < 0.05$.

4.2 Sensor Fusion with Dynamics Stream Weights

Now that we have shown that fixed stream weights can work in favor of the prediction capability of the Kalman Filter, we can strive to develop an even more sophisticated approach to sensor fusion. Conceptually, the true power of stream weights can be harnessed if there is a way to dynamically change them in response to fluctuating visual and acoustic conditions. We devised a framework to calculate dynamic stream weights in section 3.1.1.1, but implementing this theory requires understanding some components key to signal processing.

In this section, we will go over the steps necessary to implement dynamic stream weighting. For this study, as mentioned in section 3.1.1, we used two measurement models—one for representing each sensory modality. Going forward we explain how the Gaussian measure-

ment models are replaced by the Time Delay of Arrival algorithm and an object detection algorithm for the auditory and visual sensory modalities respectively and elaborate on how they work together to inform the calculations of the dynamic stream weight, λ (and $1 - \lambda$).

4.2.1 Time Delay of Arrival Localisation

To eliminate the Gaussian measurement model impersonating the presence of an acoustic sensor, we incorporated the Time Delay of Arrival (TDoA) Localisation algorithm. TDoA is a technique for locating an acoustic source with respect to a microphone array. It takes as input a set of sound signals and uses the time difference with respect to when each of the signals reach the distanced microphones to return the coordinates of the sound source relative to the sensors. The complete mathematical derivation of this algorithm is in [11].

A model simulation of the TDoA algorithm is shown below.

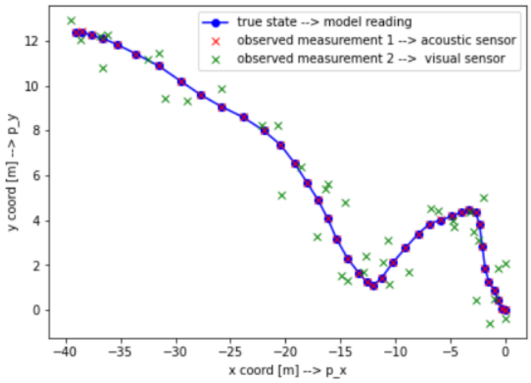


Figure 10: TDoA localisation

In the above graph we can see the blue and green data points which come from the Gaussian motion and measurement models respectively. Additionally, we see the red points which we find using the TDoA localisation algorithm. By noticing that the red crosses almost perfectly coincide with the blue points, we can infer that the TDoA algorithm does a great job at localising the sound source.

4.2.2 Object Detection

We proceed in our goal of achieving audiovisual speaker localisation by devising a strategy to replace the second measurement model which mimics the behaviour of a visual sensor. We begin by animating the trajectory of the mobile speaker based on the positions we get from the Gaussian motion model. After that, we implement an object detection algorithm (using Python's OpenCV library) on the animation to continuously track the location of the moving speaker.

Below is a model frame of the object detection program running on the animated speaker trajectory.

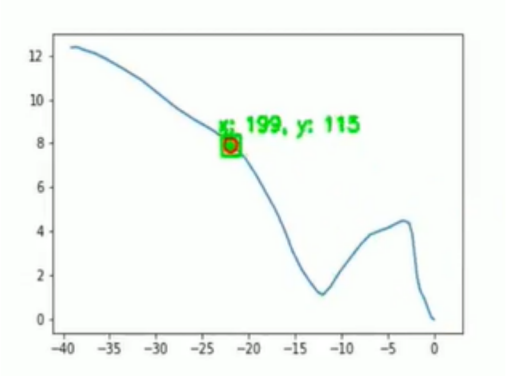


Figure 11: Object detection on a particular frame using Python's OpenCV

Finally, using the TDoA localisation and object tracking algorithms, we have the following model simulation of the program for 40 time steps. In the below graph we see the blue points which we get from the motion model, red crosses which we get from the TDoA localisation algorithm and the green crosses that we get from the object detection algorithm.

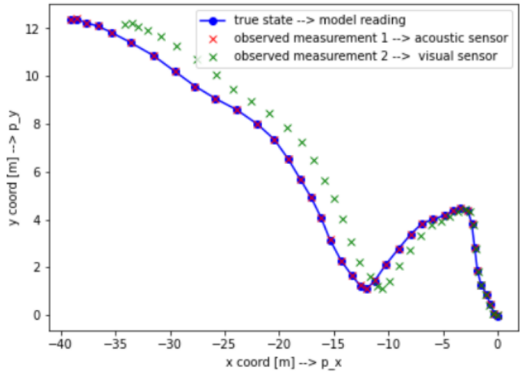


Figure 12: Audio and visual localisation

Together, the data points from the motion model work with the information we received from both the acoustic and visual sensors to get the Kalman Filter estimate of the mobile speaker position.

A graph of the motion model estimates along with the Kalman Filter estimates for the above simulation looks like the following.

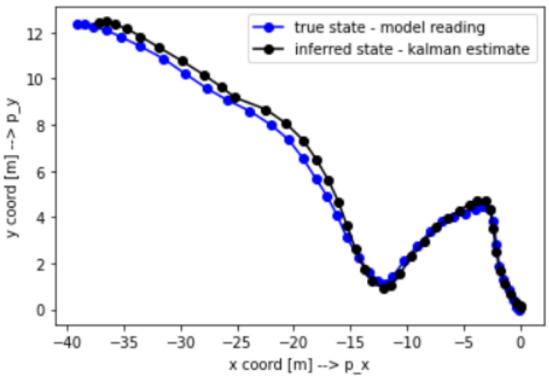


Figure 13: Model reading and Kalman filter estimate

4.2.3 Results

We test the efficacy of our algorithm by running some simulations. To simulate a dynamic acoustic background, we first adjust the λ_1 values to be linearly increasing and then to be sinusoidally varying over time. Recall that λ_1 is the noise term for the acoustic signal. In

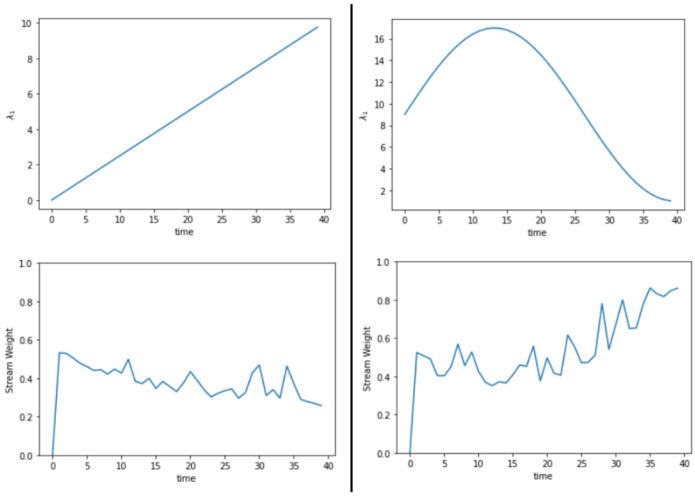


Figure 14: Adjusting λ_1 values

both cases we see that the dynamic stream weights adjust appropriately to reflect changes in the level of noise being added to the acoustic data. As λ_1 goes up the stream weight values decrease and vice versa.

Furthermore, we verify the statistical significance of the improvements in the predictions of the multimodal Kalman filter using RMSE values and t-tests with $p < 0.05$.

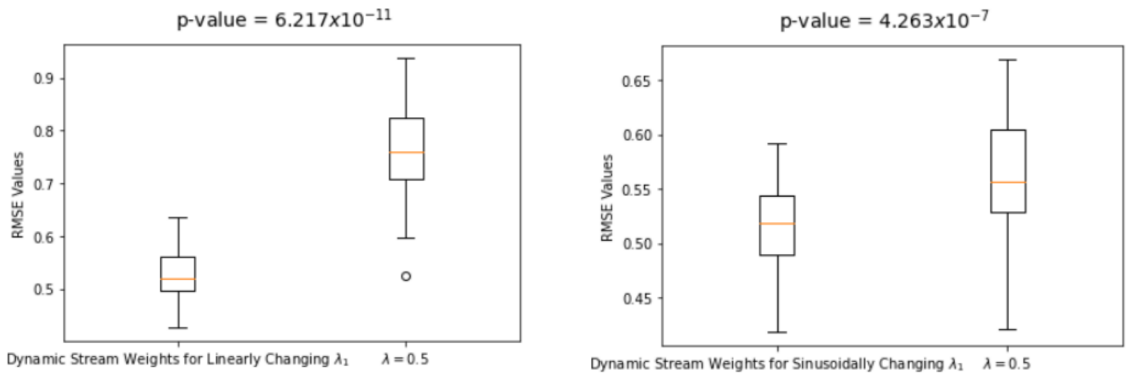


Figure 15: Statistically significant improvements seen upon incorporating dynamic stream weights into prediction algorithm

Finally, although not shown here, we also try the performance of the algorithm using simulated sound signals. In this case we allow λ_1 to fluctuate depending on the signal-to-noise ratio calculated by the algorithm at every time step. Here as well, the use of dynamic stream weights confirmed statistically significant improvements compared to using fixed stream weights or no stream weights at all.

4.2.4 Summary

The complete speaker tracking task is summarised in the pseudocode below:

Algorithm 1 Multimodal Kalman filter for AV-ASR

Require: Audiovisual sequences of observations $\{y_{A,1}, ..., y_{A,K}\}$, $\{y_{V,1}, ..., y_{V,K}\}$, dynamic stream weights$\{\lambda_1, ..., \lambda_K\}$
  for k = 1:K **do**
    Compute $\hat{x}_{k\ k-1}$, $\hat{P}_{k\ k-1}$
    Obtain $y_{A,k}$, $y_{V,k}$
    Apply TDoA localisation algorithm to receive acoustic sensor measurements
    Apply object detection algorithm to receive visual sensor measurements
    Calculate λ_k (DSW)
    Compute Kalman Gains $K_{A,k}$, $K_{V,k}$
    Update $\hat{x}_k$, $\hat{P}_k$
  end for
  return Estimated speaker path $\hat{x}_1, ..., \hat{x}_K$

5 Concluding Remarks and Future Directions

In this study, we have evaluated the performance of our prediction algorithm using fixed and dynamic stream weights for mobile speaker tracking. Through experimentation, we found that dynamic stream weights tend to perform better then fixed stream weights, as expected, and also verified that both methods perform superior to the conventional Kalman filter for the given task.

It is worth noting that there are some limitations in our current system. One constraint

of our program is that it is currently working with simulated data. A next step would be to adapt the program to work in real time. For that we would explore incorporating parallel computer architectures and study hardware-software interfaces.

The work presented in this thesis is a small step taken in the direction of achieving near perfect automatic speech recognition. The methods presented here have the ability to be exploited for many research directions. One particular interest would lie in augmenting a speaker tracking system with other speech enhancement software, such as DeepSpeech [9] or Cochlearity [6], to manufacture an even more sophisticated sensory perception device.

6 Acknowledgements

I would like to begin by acknowledging that the University of California, Davis is built on the land of the Patwin people. As a researcher at UC Davis and a beneficiary of the university settlement over the Patwin land, I would like to show my gratitude for the Patwin elders by honoring their legacy and connection to this land.

Additionally, I would like to express my sincere gratitude towards professors Timothy Lewis and Lee Miller. Professor Lewis encouraged me to pursue research and Professor Miller introduced me to the world of mathematical neuroscience three years ago by taking a chance on lowly, but eager to learn, undergraduate. Together they advised me on my first independent research project and for that I will always be grateful. Their patience, sincerity, and eagerness to see me succeed made this journey the most challenging, confidence-boosting, and fulfilling experience of my undergraduate career.

Furthermore, I would like to thank my colleague and mentor, Britt Yazel. Britt not only acted as a mentor to me in lab but also took up the positions of friend and role model. I wouldn't have had the knowledge, experience, or confidence to take on this project if I didn't have him as a guiding light throughout my undergraduate research experience. Along with him, I am also thankful for my friends Isabella Vo and Samantho Cho who are always there for me during the good times and even the times when my *experiments just won't work*.

Finally, I would like to thank my parents and sister for their continuous love and support and most importantly my cockatiel, Munu, who made her one way trip to heaven in December, 2018 after being my best friend for three years.

References

[1] 3 reasons why signal processing is the career of the future. https //signalprocessingsociety org/publications-resources/blog/3-reasons-why-signal-processing-career-future#. Accessed: 2021-05-17.

[2] Delay sum beamforming. http //www labbookpages co uk/audio/beamforming/delaySum html. Accessed: 2021-05-17.

[3] How delay-and-sum-beamforming works in the time domain. http //www labbookpages co uk/audio/beamforming/delaySum html. Accessed: 2021-05-17.

[4] The kalman gain. https //www kalmanfilter net/kalmanGain html. Accessed: 2021-05-17.

[5] A. H. Abdelaziz, S. Zeiler, and D. Kolossa. Learning dynamic stream weights for coupled-hmm-based audio-visual speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(5):863–876, 2015.

[6] M. H. Anderson, B. W. Yazel, M. P. F. Stickle, F. D. Espinosa iñguez, N. S. Gutierrez, M. Slaney, S. S. Joshi, and L. M. Miller. Towards mobile gaze-directed beamforming: a novel neuro-technology for hearing loss. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 5806–5809, 2018.

[7] Michel Van Biezen. Lectures in the kalman filter. http //www ilectureonline com/lectures/subject/SPECIAL%20TOPICS/26/190.

[8] M.H.A. Davis and R.B. Vinter. *Stochastic Modelling and Control*. Chapman and Hall, London, 1985.

[9] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition, 2014.

[10] Jonathan Hui. Self-driving car object tracking: Intuition and the math behind kalman filter, April 2018. [Online; posted 8-April-2018].

[11] Steven Li. tdoa_localization. https //github com/StevenJL/tdoa_localization.

[12] T.S. Lindsey. On the kalman filter and its variations. Master's thesis, University of Kansas, 2014.

[13] Prajoy Podder, Md. Mehedi Hasan, Md. Rafiqul Islam, and Mursalin Sayeed. Design and implementation of butterworth, chebyshev-i and elliptic filter for speech signal analysis. *International Journal of Computer Applications*, 98(7):12–18, Jul 2014.

[14] C. Schymura, T. Isenberg, and D. Kolossa. Extending linear dynamical systems with dynamic stream weights for audiovisual speaker localization. In *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pages 515–519, 2018.

[15] James Teow. Understanding kalman filters with python, May 2018. [Online; posted 3-May-2018].

[16] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT Press, Cambridge, Mass., 2005.

[17] Vivek Yadav. Kalman filter: Intuition and discrete case derivation, March 2017. [Online; posted 4-March-2017].