# ANTI

## Automatic Note Taker for the Impaired

## Final Report

Group 29:
Roland Anderson
Patrick Galloway
Casey Miville

**EEL 4915 Senior Design II**
Submitted to:
**Professor: Dr. Richie**

**Department: Electrical & Computer Engineering**
**University of Central Florida**
**Date: 5/1/15**

**Table of Contents**

# 1.0 – Executive Summary

## 1.1 – Abstract

The Automatic Note Taker for the Impaired (ANTI) is small robotic system that follows a professor or presenter using facial tracking. The system has speech to text software that makes notes for the user via a USB connected microphone on a portable Windows device. Using a high fidelity directed audio microphone to record audio and a high-resolution pan/tilt camera to record video with a post-processed text file for redundancy purposes. Therefore, the user will not only have the notes from the presentation or class but also audio, video, and a text file to follow, for a more detailed account of the event.

The ANTI device is required to function in a class sized environment with various external inputs. The device tracks the presenter and is able to pan over 180 degrees and tilt within a 90 degree range to cover the presenter's movement. The device is also approximately 8-12 inches high and has a footprint smaller than 6 inches in diameter. The size accounts for an average sized desk in a classroom that the device will have to be resting on alongside a laptop computer. The device is able to track movement from over 10 feet, with good lighting and is able to direct the microphone to the target for greater audio pickup. Software includes a GUI with easy to use options for disabled students and can be installed on any Windows laptop device with a USB interface. The device is powered separately through external power supply; though the device can be partially powered through USB connection with the user's laptop.

# 2.0 – Project Description

## 2.1 – Motivation

Our group saw a need for this low cost portable solution to help students with disabilities take accurate notes in class. The current method to help disabled students is the paid student note taker. This system is inherently flawed, because the disabled student is reliant on the note taking skills of another student, who might make mistakes or not take as detailed notes as possible to help the disabled student better understand the lecture. Some student note taker may ignore or miss something the professor or lecturer stated because they may think it not important or common knowledge. The simple basic fact is when someone is fully relying on notes from someone else's perception or understanding of a subject matter the account of that presentation will always be skewed. This system fixes that flaw by having a detailed comprehensive account of the lecture or event that is thorough and unbiased.

Every semester most of the UCF student body receive emails offering to pay students $150 per class to be note takers for the impaired. This system will save the school a lot of money

by offering a low cost, portable, all-in-one solution to multiple students at time, instead of each disabled student waiting on a note taker from his or her class to email the notes.

# 2.2 – Objectives

This device was modified in stages to add functionality to the finished product. Each phase represents a large leap in software or hardware design and integration. Phase 1 represents the base functionality that this design strives to accomplish while the latter phases are optional additions and bring forth new challenges to be overcome.

## 2.2.1 – Phase I

Phase I consists of accomplishing the goal of taking notes in a written form for the user. The main functionality of the project was split between two major devices. A laptop PC and the ANTI device's attached microphone are connected and allow the PC to acquire input from the said microphone. The ANTI device is an independent system that is capable of identifying a presenter, the target, and positioning the microphone to face the target. The ANTI device accomplishes the tracking via an on-board camera and pan/tilt servo base. The microphone mounts to the pan/tilt head, passes its output through a USB adapter, and connect directly to the laptop PC. The PC is responsible for running our "speech-to-text" software with the microphone as its input. Phase I requires the software to output a text file of what the target has spoken so that the lecture can be read, for review, at a later time.

## 2.2.2 – Phase II

Phase II consists of adding a static, second camera mounted to the laptop PC. The static camera records the entire scene in front of the user for greater context. Allowing the user to review visual information as well as the spoken information. Also, in Phase II, the software attaches the sound input to the video file. The software continues to create a text file, using Phase I's "speech-to-text" software. The output is represented in Figure 2.1.

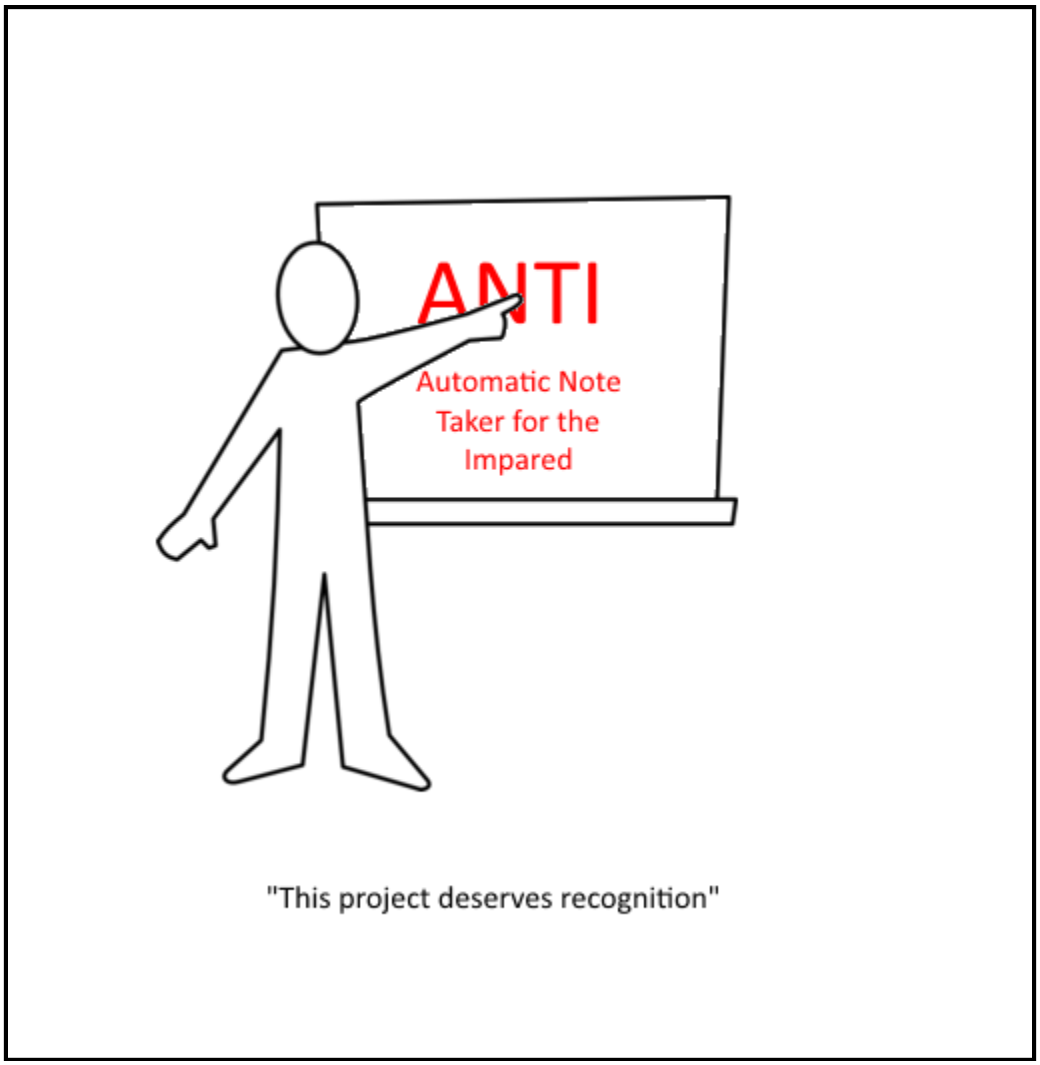


Figure 2.1 - Initial concept for Phase II output

## 2.2.3 – Phase III

Our current version of the project has yet to reach Phase III, but the group fins that this phase is a necessary step to make our product more marketable. Phase III consists of integrating the video from the ANTI recorder's embedded camera, and adding sub-titles to the video output. Since the camera is always focused on the presenter, that footage can be useful for the user to understand the presenter's message or concept by watching their gestures. To use this input, the video feed can be sent to the laptop PC for processing through an SSH connection. The software on the PC will overlay the input from the ANTI recorder in a corner of the final video output, in sync with the video. The output is represented in Figure 2.2

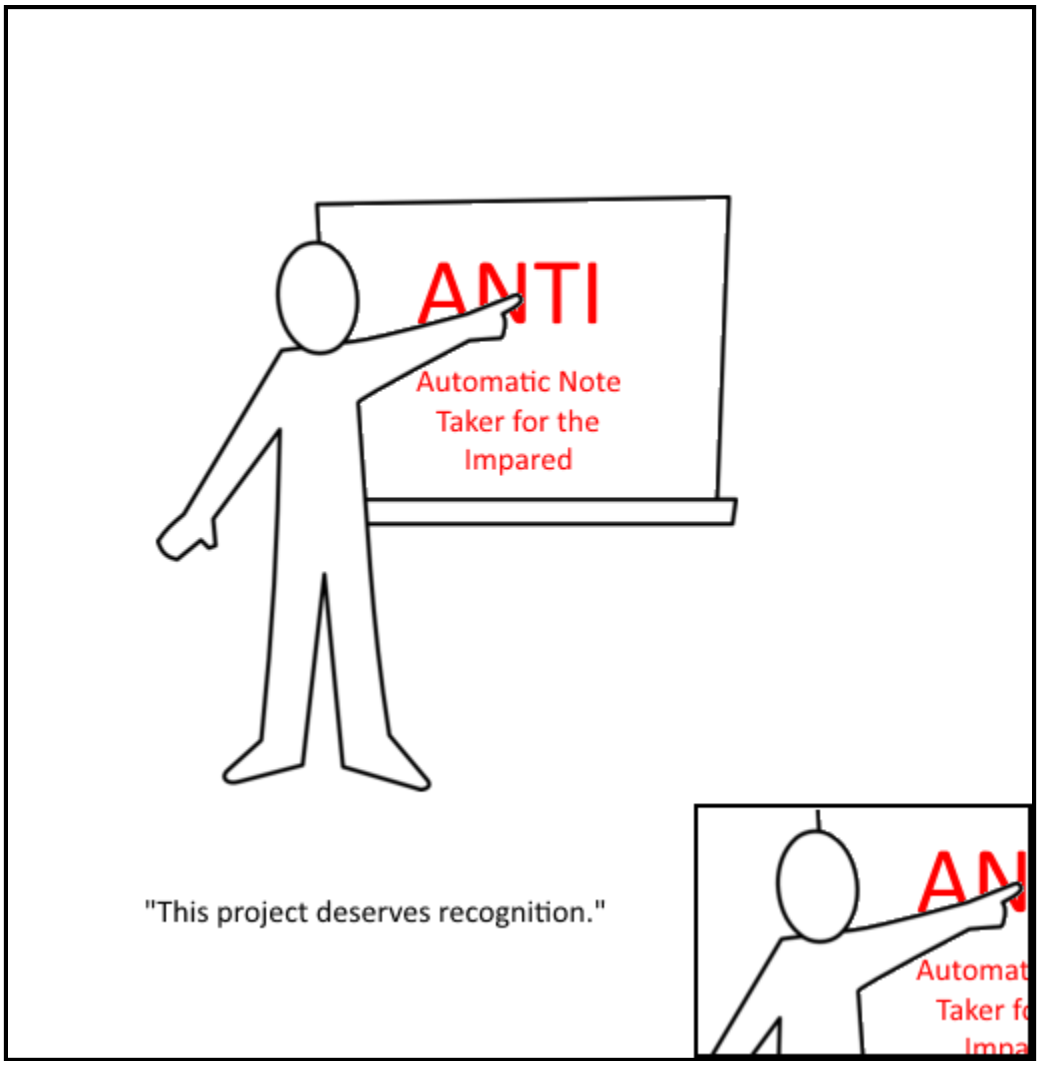


Figure 2.2 - Initial concept for Phase III output

# 2.3 – Specifications

The device is required to function in a class sized environment with various external inputs. The device tracks the presenter and is able to pan over 180 degrees and tilt within a 90 degree range to cover the presenter's movement. The device is approximately 8-12 inches high and has a footprint smaller than 6 inches in diameter. The size is to account for an average sized desk in a classroom that the device is resting on alongside a laptop computer. The device is able to track movement from over 10 feet and can direct the microphone to the target for greater audio pickup. Software includes a GUI with easy to use options for disabled students and can be installed on a portable Windows device with a USB interface. Device is powered through two 5 volt at 2 amp power cords, or one 5 volt 2 amp power cord and a USB connection with the uses laptop. The preliminary drawing for the entire integrated system is shown in Figure 2.3.

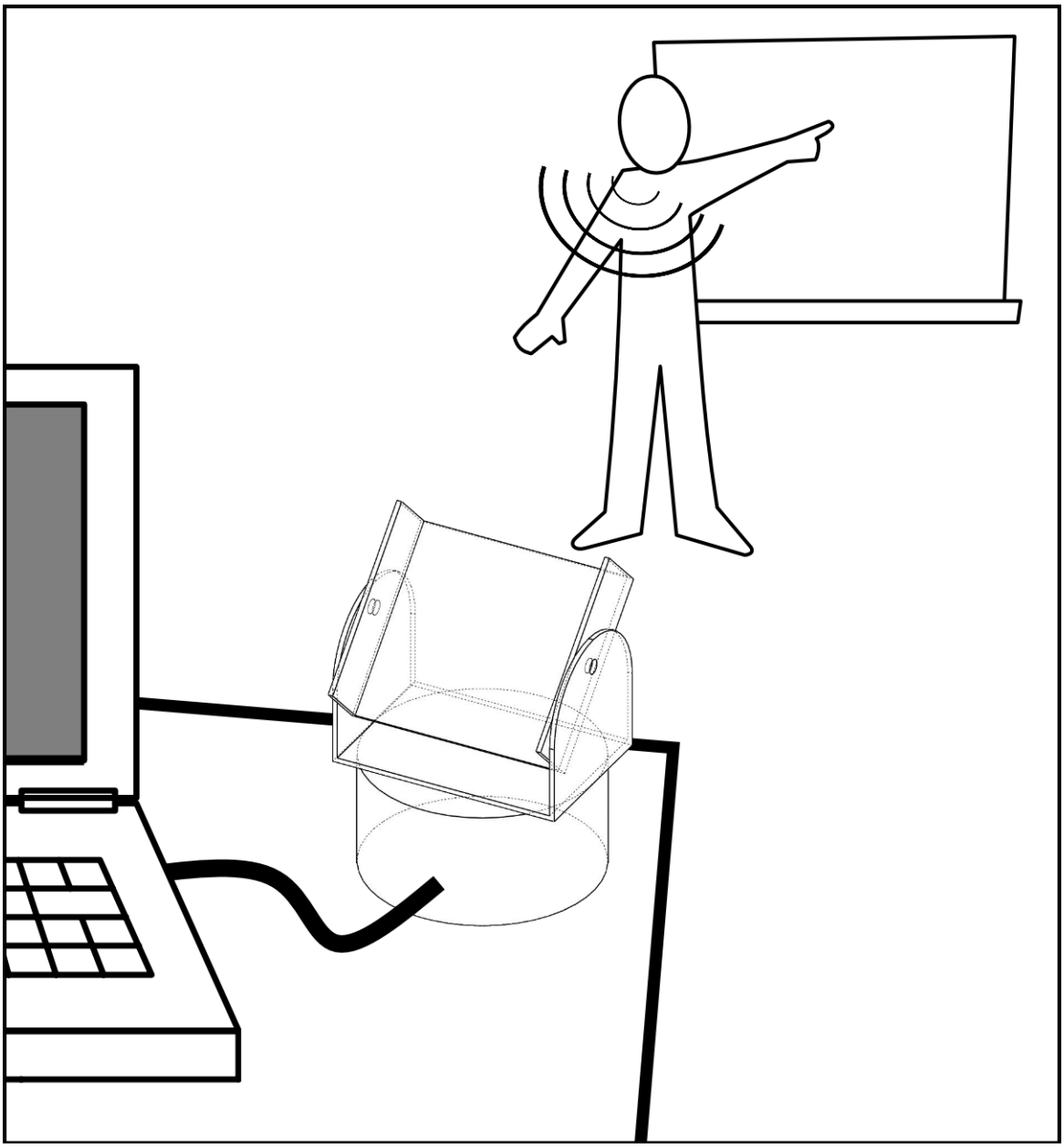


Figure 2.3 - Initial concept: Laptop PC and ANTI recorder in a classroom setting.

## 2.3.1 – Hardware

Figure 2.4 (below) shows the initial block diagram used in designing the hardware system for the ANTI device. This is will be seen later in section 4.0.

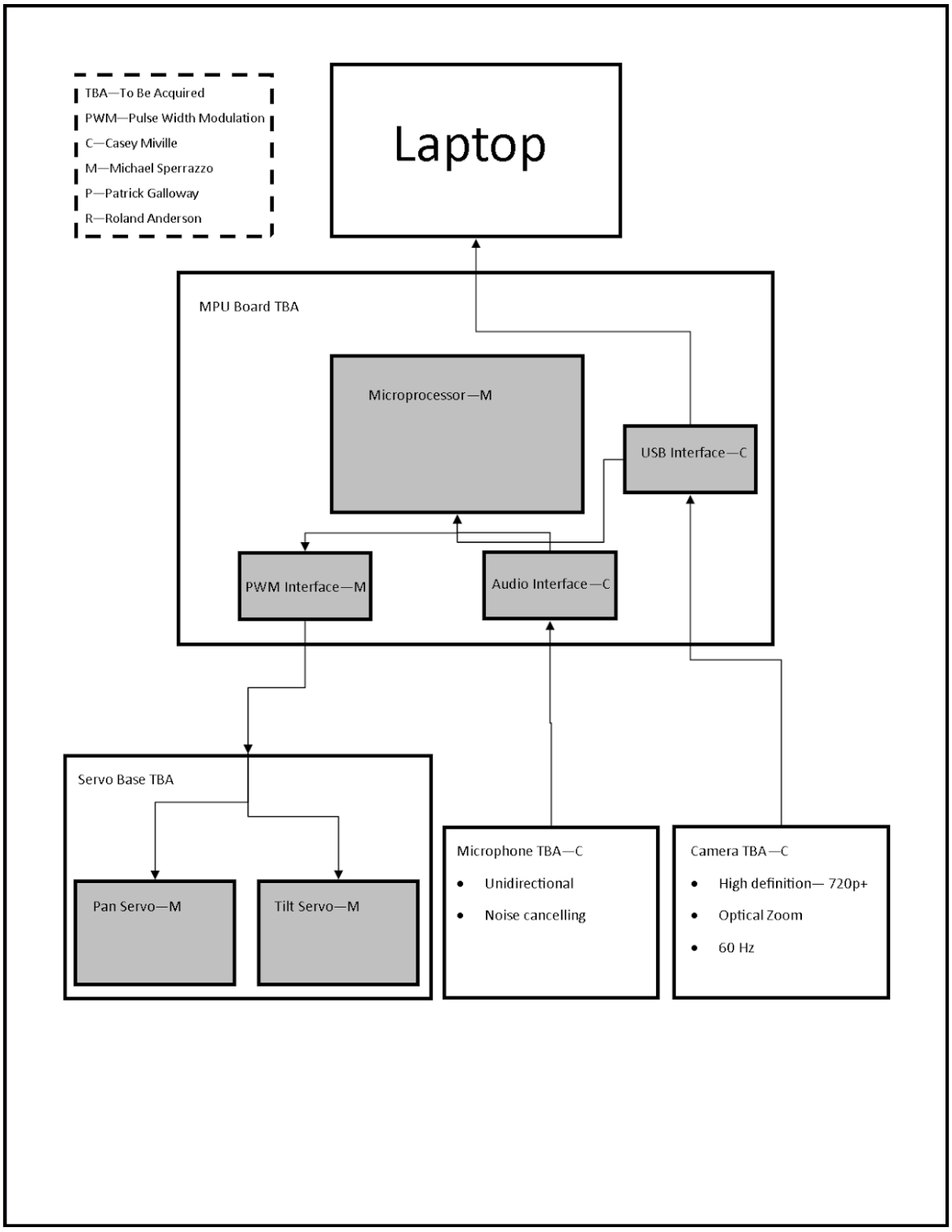



Figure 2.4 - Initial Block Diagram - Hardware

## 2.3.2 – Software

For this project there are two main sets of software specifications. One is for object tracking and the movement of the microphone. The other software specifications is the speech to text recognition, which takes place on the computer.

### 2.3.2.1 – Object Tracking

This section is an overview of the object tracing software and some of its features.

#### 2.3.2.1.1 – Introduction

##### *2.3.2.1.1.1 – Purpose*

The purpose of this software is to identify and track a singular significant object (in this project the object will be the lecturer) in front of the camera. The secondary purpose is to keep that object in the center of the captured images in order to properly record the object's sound/speech. This set of specifications outlines our group's options and potential approaches to solving this software problem.

##### *2.3.2.1.1.2 – Product Scope*

Where we want to go with this software is to make a portable software package that runs on potentially any embedded device. Our vision for this software is to quickly identify an object to be tracked and to keep that object relatively centered in a frame at all times. There are several goals we wish to achieve with this software, to make it portable, to make it reliable, and to work on an embedded operating system.

#### 2.3.2.1.2 – Overall Description

##### *2.3.2.1.2.1 – Product Functions*

The functions of the software used in this product are shown in Figure 2.5 below. Figure 2.5 shows the initial block diagram that represents the software structures.

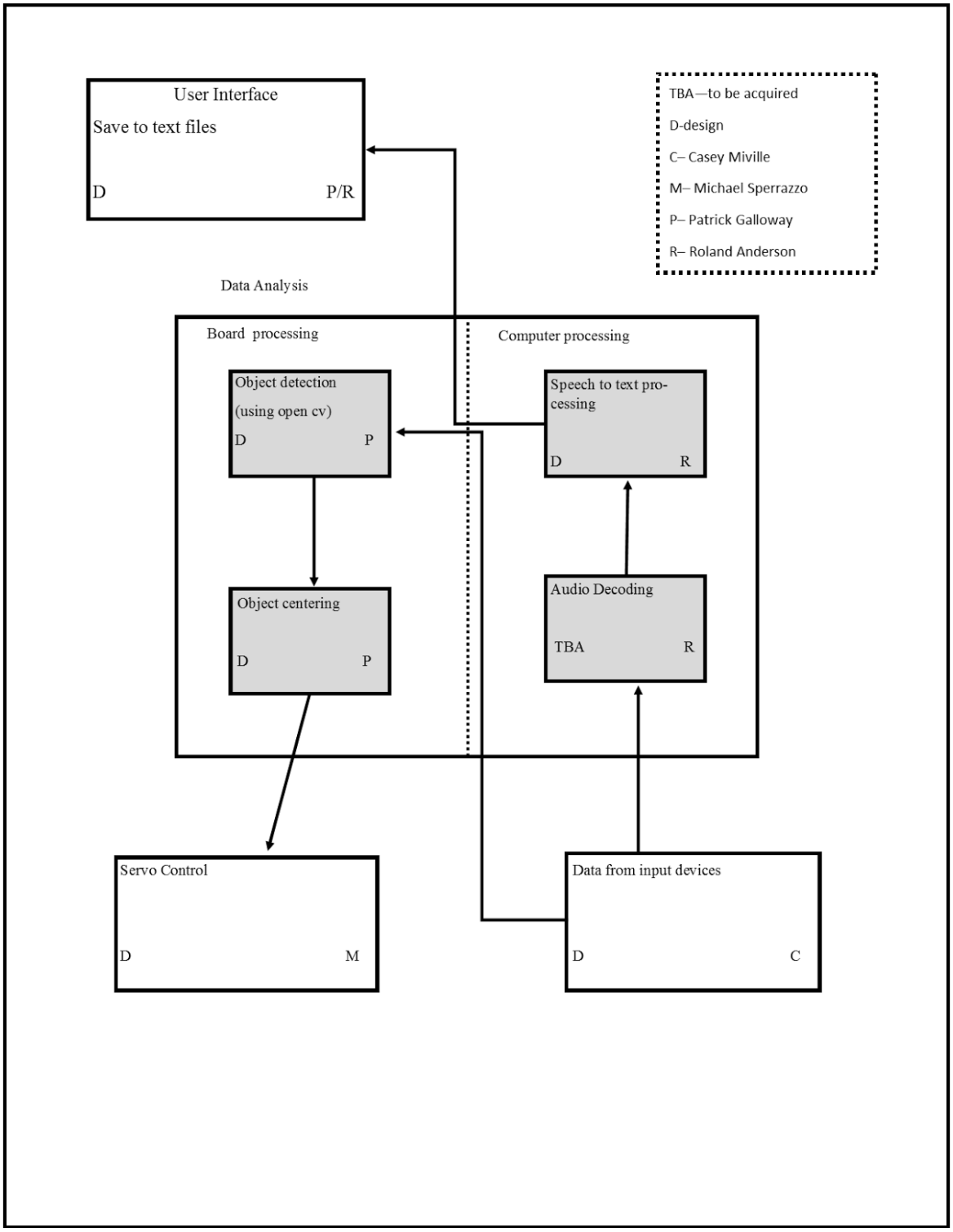


Figure 2.5 - Initial Block Diagram - Software

*2.3.2.1.2.2 – User Classes and Characteristics*

There are several way for us to achieve the same result for following objects. In the first case there will be two main classes that will cohesively come together to produce the final product.

- Object detection

  - Viola Jones algorithm if we utilize facial detection.
  - Color histogram detection assuming that the clothing of the lecturer is vastly different from the surrounding classroom
  - Blob detection similar to the color histograms but finds points and features that count as one object and then label it
  - motion detection; assuming that when the camera is stationary the only thing that will move is the lecturer we will follow this movement and turn off this function when the camera is moving to avoid jerkiness and error

- Object following

  - Object centering; The main idea we had was to line up the camera and microphone so that if we were to center the frame on the observe object the microphone would be perfectly to record the maximum amount of sound from the speaker. To center the object we will use the given parameters from the object detection class and then check different frames for movement. The output from this class will be a vector to plug into our servos to properly center the frame.

*2.3.2.1.2.3 – Operating Environment*

The operational environment will be on an embedded device. We have several options for the device in terms of operating systems.

- No operating system: This one would be tough because we'd have to manage memory and manually track the input and output ports on the device. we'd also have to create our own drivers and libraries for some of the software, which would be infeasible given the timeframe of our project and the current skill level of our members. The advantage of not having an operating system is that there would be less memory and processing power required for the overhead of the operating system. With just the software the embedded device can also be streamlined for its particular task with less fine tuning that an operating system would require.

- Linux operating system: This platform would make setup and use of the board a more surmountable task. With build in memory management and tools for streamlining data and functions to make software more efficient, this operating system would be best suited for our timescale. The disadvantage of having a Linux operating system kernel on our device is a loss of memory and processing power to the operating system . The other

downside is there are many options/ types of operating system environments to choose from and finding the correct one can and has been difficult.

*2.3.2.1.2.4 – Design and Implementation Constraints*

The main design constraint is going to be processing power. Video manipulation has traditionally been done on a full computer. That being said with the advent of newer processing technology we have been able to narrow down a few embedded processors that will be able to perform the task, if just barely.

The second constraint is going to be primary and secondary memory space. Computer vision software is generally a rather memory consuming task, with relatively large secondary memory space needed. This is going to be partially remedied by adding an SD card to give the board secondary memory. Since we are also considering an operating system we might have to test different amounts of memory to determine what is feasible.

Finally the last constraint is the software used for the computer vision. There are limited open source options that will conform to an embedded platform. Making computer visions applications difficult.

## 2.3.2.2 – Speech to Text

### 2.3.2.2.1 – Introduction

*2.3.2.2.1.1 – Purpose*

The purpose of this software is to accept incoming sound from a lecturer and translate that speech into readable text for a user. What our group wants to do is to create an environment where someone who might have difficulty taking notes for a class has a reliable source to transcribe the class for them.

*2.3.2.2.1.2 – Product Scope*

Where we want to go with this software is to make a portable software package that will run on any operating system. The vision we have for this software is to accurately and quickly transcribe sound from a speaker to text on a screen. We also want to be able to save this speech to a file so that the material can be reviewed later so that the user may study from this material.

### 2.3.2.2.2 – Overall Description

*2.3.2.2.2.1 – User Classes and Characteristics*

While there are many options in solving the speech to text we chose to break up the process into steps that seemed clear to us, the first being to decode the audio. In this step we will perform any necessary filtering and test for quality of sound. After the audio is decoded we

will send it to our speech to text algorithm. Currently the group is leaning toward a cloud based solution for speech to text, to help minimize the footprint on the user's computer. Also the simple fact remains that on board speech to text is just not possible to due to the processing requirements and vast amount of data needed. For example my ford touch has speech recognition just as many other car manufactures, but this system can only recognize a set of programmed words or phrases. This is why users have to say the command in a certain way for it to work. It is slow and at times frustrating which leave users to just use the buttons instead. This is a perfect example of why on board speech recognition is not practical or good at all.

*2.3.2.2.2.2 – Operating Environment*

There are several possible operating environments. The first being a desktop environment. On a desktop environment the program will be able to process the information effectively. The down side is the large amount of storage space required for speech to text to work. Speech to text requires storing a large database of voices wherever it is being processed. This leads up to cloud based speech to text. The advantage of cloud based speech to text is its low footprint on the user's computer. This will also mean that the user's computer will only really need an internet connection to run the program

*2.3.2.2.2.3 – Design and Implementation Constraints*

The main design constraint will be the user's ability to use the internet and the speed of that connection. The problem with our intended approach is that it requires high quality internet access to perform it's cloud computation and get back to the user in a reasonable time.

# 3.0 – Research

## 3.1 – Hardware Research

### 3.1.1 – Processor Selection

There will be two boards, a board for performing the computer vision processing, gathering data from sensors, and sending data to an external computer, and a board which will receive displacement vectors from the computer vision processor and run control software in an attempt to minimize this displacement error signal.

For the computer vision processor, there are several processing options to choose from. They come in three levels: the low end being the Cortex M4 microcontroller, the middle end being the Sitara Cortex A8 processor, and the high end being the Divinci Cortex A8/C674x+DSP processor.

For the servo controller board, only MCUs will be considered. The two under consideration are the TI Tiva 4C129x line and the TI Piccolo TMS32F2806x Line.

#### 3.1.1.1 – CV Processor Advantages

The advantages of the Cortex M4 microcontroller are its power consumption and the cost of development boards to test out functionality. The cortex M4 has very low power consumption and this will help with power management on the device. Because the servos require a relatively high amount of amps over a short time, a battery pack will be necessary to run them. The power of the processor is so low that we can power that part of the hardware from a USB input. This form of board power will be nice because it will allow us to modulate power and to not waste extra battery power on the board, which will help the batteries last longer.

Power is going to be the biggest benefit of the Cortex A8 processor. The computation power of the Cortex A8 is greater than that of the M4. Both are based on ARMv7 architecture, however the Cortex A8 uses the 'A' (Applications) profile and the M4 uses the 'M' (Microcontroller) profile. The M4 is limited to the Thumb/Thumb2 instruction sets, but the A8 can also use parts of the A32 set, which provides increased performance in critical applications. A very notable feature which the A-profile includes but the M-profile lacks is different processor modes with different levels of privileges, which is very important for operating systems such as Linux.

An obvious difference between the two is the drastically higher clock rates available in the A8 than in the M4. Additionally, while there is a slight caching component for instruction memory in the M4, there is no cache in the core for data memory. The A8, on the other hand, has a two level cache for data memory. This also improves the video and image processing quality of the processor. This is still an ultra low powered system, with most development boards the user only needs a USB plug for basic applications.

The Davinci processor has all of the advantages of the cortex A8 processor with a floating point DSP processor strapped on. This processor would be the pinnacle of the processing power required for this project. The DSP floating point processor would be perfect for analyzing and processing the incoming video quickly and efficiently. On top of being able to handle our video manipulation needs, this system will be able to handle the other computations necessary to make the system run smoothly.

#### 3.1.1.2 – CV Processor Disadvantages

The main disadvantage of the Cortex M4 is processing power. The processor might simply not be powerful enough to do the video processing required for the project. There is also a power balancing act that the M4 will bring to the table that will complicate things. If it is possible to run the board and the servos off of the limited power supplied by a USB batteries might not be necessary. This could be a detriment

because there might be points in time where the processor will need to be slowed down or turned off to give the servos enough power to do their jobs. This could interfere with the video processing algorithms that will need to be running in the background while the camera moves.

The Cortex A8 processor will be a problem because of the complexity of design. While performance is fairly optimal to be able to design a board around this interface will be a complicated task, even with a simplified design. Power management might also be a problem point for this chip, while the chipset is made to be run on ultra low power, the supporting components for the chip are rather costly to power. This will increase our need for battery power and will reduce battery life.

Power and Complexity are the two main problems with the Davinci A8/DSP Processor. Being capable of so many things lends to a complex design space for the chipset. That being said, the Davinci is not an ultra low power system, with even the most basic of its developement boards needing either battery or direct power, this will be costly to our battery packs and we will have to design around this to be operable of the required amount of time. This processor also might be too powerful for the application we intend to use it for.

### 3.1.1.3 – Motion Control System MCU Selection

Under consideration are two excellent microcontrollers: a Texas Instruments Tiva MCU and a Texas Instruments Piccolo MCU. A key difference is in the cores: the Tiva is built around an ARM Cortex M4 Central Processing Unit, while the Piccolo is built around a Texas Instruments C28x Digital Signal Processing Unit.

A block diagram for the Tiva is shown in Figure F3.1.1.3A, while a block diagram for the Piccolo is shown in Figure F3.1.1.3B.

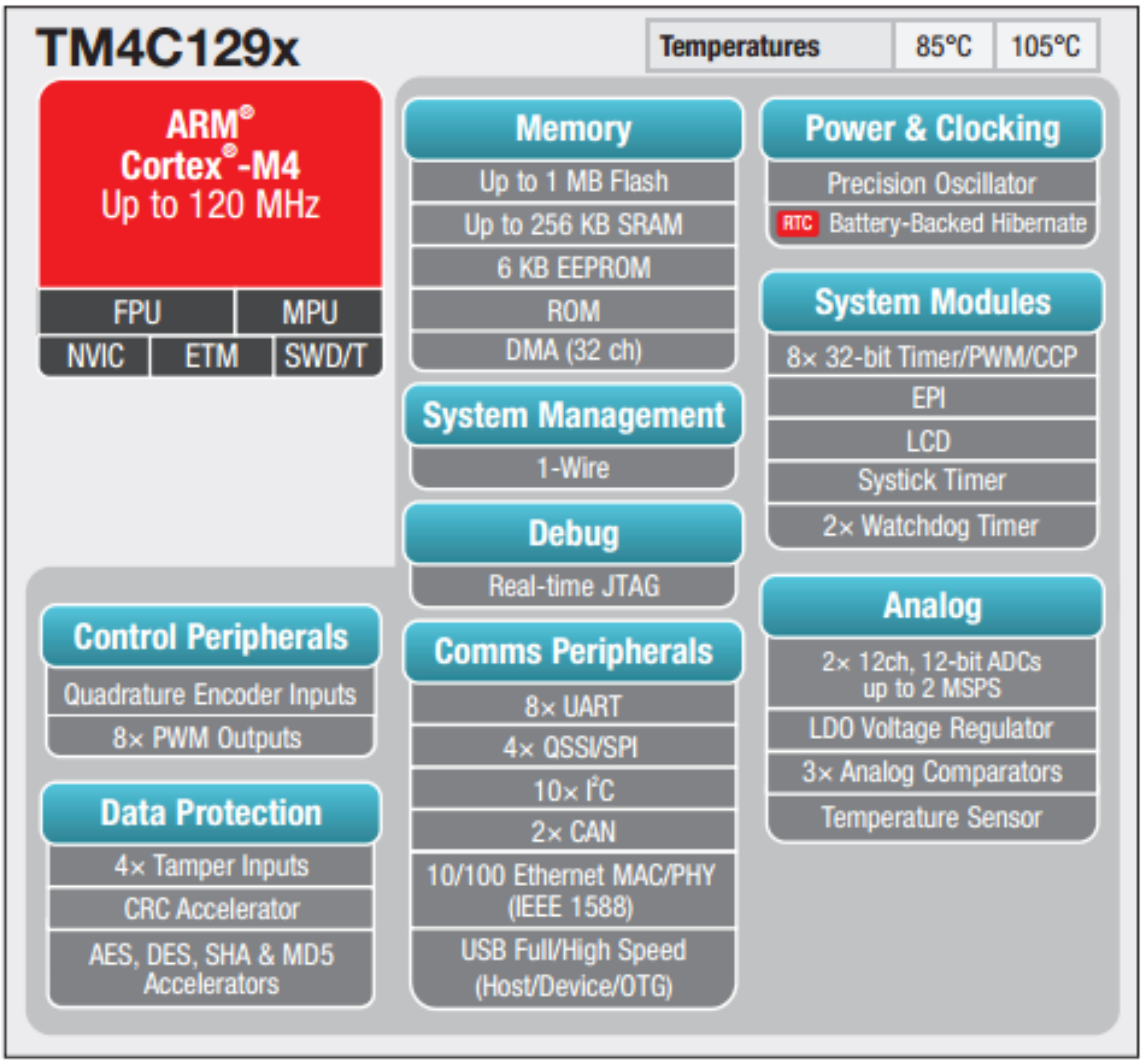



Figure F3.1.1.3A – Tiva Block Diagram
COURTESY OF TEXAS INSTRUMENTS

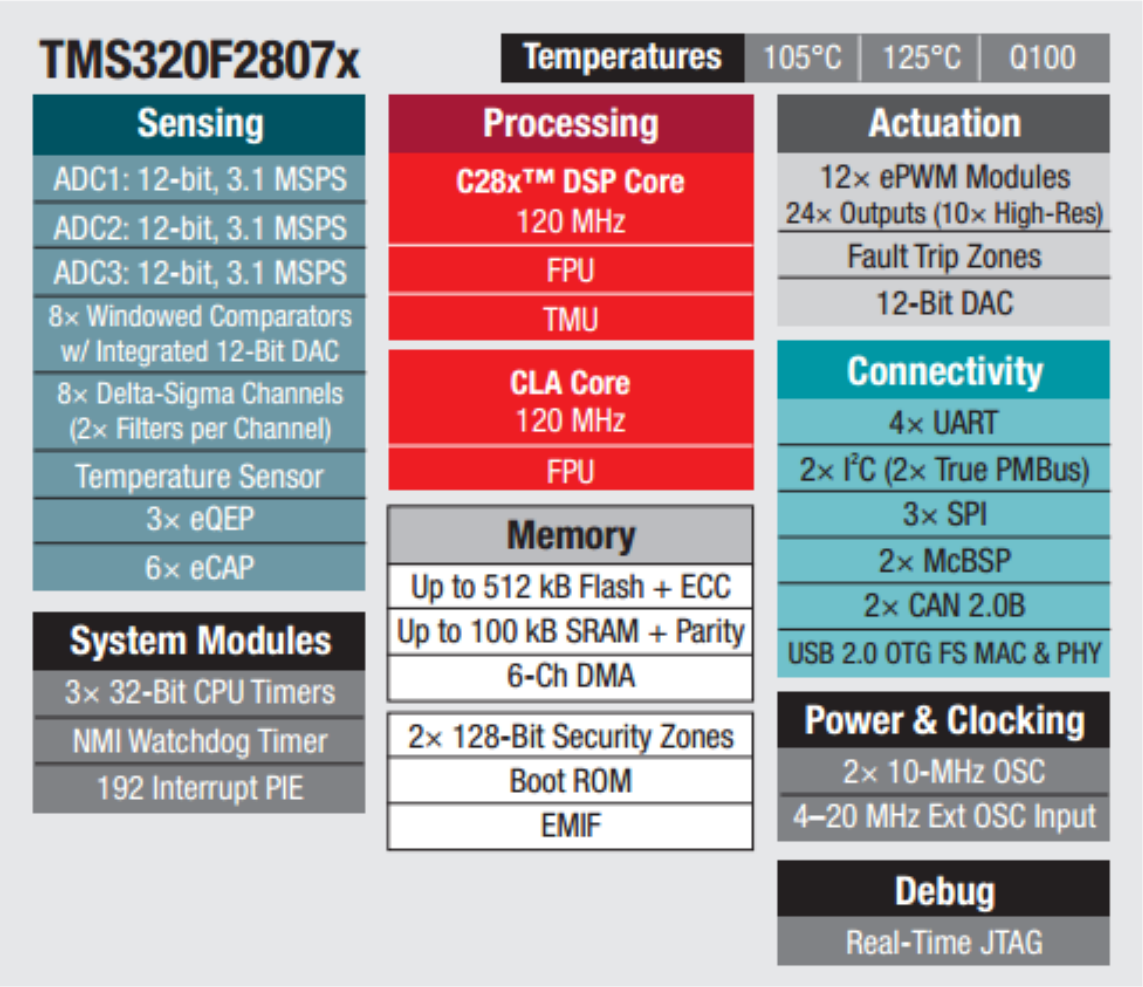



Figure F3.1.1.3B – Piccolo block diagram.

Both have an incredible assortment of features, but there are several that stand out. On the Tiva, the quadrature encoder inputs would allow for very simple integration of an incremental optical encoder for the pan servo, the benefits of which will be given later. On the Piccolo, the High Resolution Pulse Width Modulation is what is particularly relevant for this application. Either an RC Servo or a Brushed DC Optically Encoded Servo can be controlled via PWM. The HRPWM channels increase the precision of such control greatly. On the other hand, a quadrature encoder input can only be used with an incremental encoder, and would remain disabled if RC servos were used. The Tiva has only standard precision PWM channels.

Another positive point for the Piccolo is that the main core is a DSP. This means that the Piccolo has built in floating point calculation support, while the M4 has to use its FPU compressor to do so. The Piccolo also has a coprocessor called the Control Law Accelerator which is capable of performing floating point calculations and executing code independently of the C28x core. The M4 has some advantages over the C28x core, but these are not very important to the present design because there will be a powerful Cortex A8 MPU at the heart of the project.

The Piccolo also includes peripherals called Enhanced Capture units, or eCAPs. Using an eCAP would allow precise timing of incoming data from the Cortex A8, which would allow for more precise control system calculations.

For the reasons given above, the Piccolo MCU was chosen over the TIVA for this application.

## 3.1.2 – Microphones

### 3.1.2.1 – Array Microphone

A system of several closely-positioned microphones is called microphone array.[1] A Microphone Array (or array microphone) is a microphone device that functions just like a regular microphone, but instead of having only one microphone to record sound input, it has multiple microphones (2 or more) to record sound. The concept is illustrated in Figure 3.1.

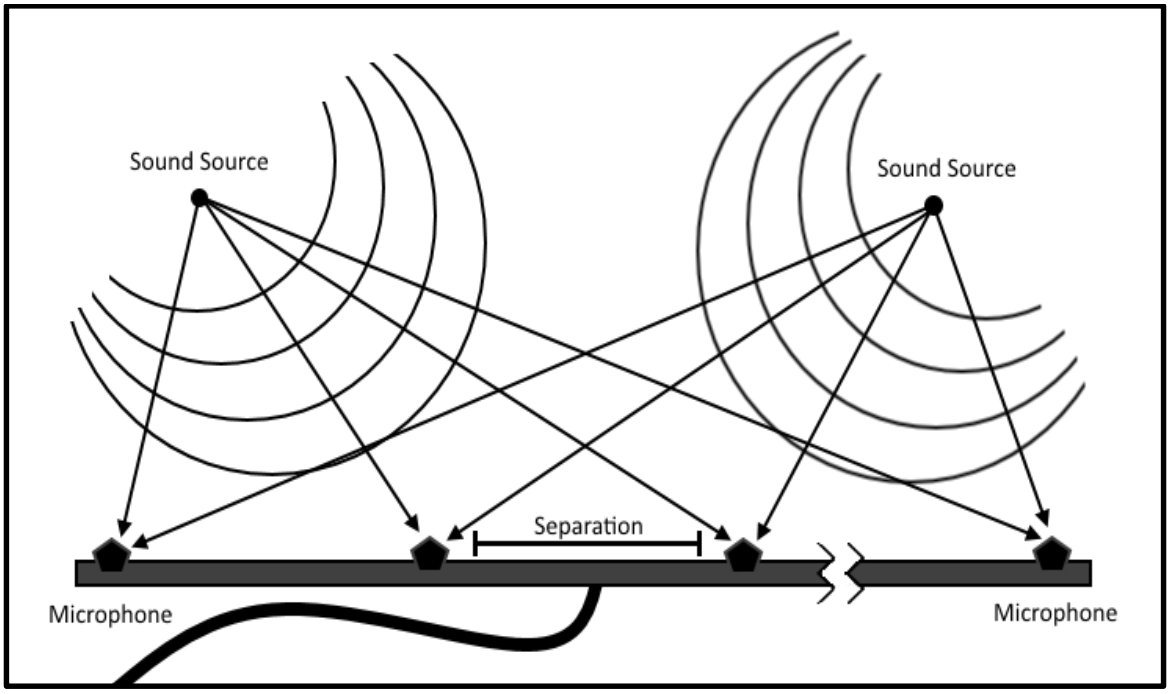


Figure 3.1 - Array Microphone

The microphones in the array device work together to record sound simultaneously. Microphone arrays can be designed to have as many microphones in them as needed or wanted to record sound output. A common microphone array, however, is a 2-microphone array device, with one microphone placed on the left side of the device and the other placed on the right side. With one microphone on each side, sounds can be recorded from both the left and right side of the room, making for a dynamic stereo recording which mimics

surround sound. When played back on a stereo headset, the separate left and right channel recording are distinctly different and noticeably heard.

The most important characteristic that must be present in microphone array devices is microphone matching. All of the microphones in an array must be similar and closely matched and in some aspects exactly the same in order for an array to pick up good recording. Otherwise, you can have a microphone array where one microphone has much higher gain than the other, or the microphones could be out of phase so that one microphone records before the other, or the problem where one microphone picks up sounds from all directions while the other picks up sounds from only a certain direction. These are all unwanted in audio applications and are some of the horrible consequences that can ensue when microphones in an array aren't carefully chosen. Therefore, the microphones must all be similarly matched and meet a number of specifications so there are no uneven sound recordings.

There are three aspects to consider for microphone matching in microphone arrays they are:

1. **Directionality-** The directionality of a microphone is the direction from which it can pick up sounds. Microphones are made to pick up sound from certain directions when spoken into. Some microphones are made to only pick up sounds from one direction, unidirectional microphones. Other microphones are made so that they can pick up sounds from all directions, omni-directional microphones. When building an array microphone, all the microphones must have the same directionality. Having one microphone pick up sounds only from a certain direction and the other pick up sounds from all directions would make for disastrous, imbalanced sound recording. Unless there is some unique situation where this would be the case, this is largely undesired. Therefore, microphone arrays are always made with microphones of the same directionality.

2. **Sensitivity-** Sensitivity is another aspect that must match for microphone arrays. Sensitivity is the gain that a microphone picks up when recording a signal. Sensitivity must be closely matched in microphone array devices, or else one microphone will be louder than the other, producing imbalanced sound recordings. This is why usually the maximum sensitivity difference allowed in array microphones is ±1.5dB so that there is no bigger than a 3dB difference in microphone sensitivity of the microphones.

3. **Phase-** Phase is the last important aspect that must match for microphone arrays. Phase is the degree line of reference for the time that a microphone begins recording, meaning, it determines the time that all microphones in an array start and stop recording. If microphones have drastically different phases, they will record signals at different times. This will lead to unsynchronized recording. Again, this is largely undesired. It is desired that microphones record signals at the same time so that there is no delay between signals. Just like sensitivity, there must be a maximum allowable tolerance for phase difference between microphones. This difference is usually ±1.5 degrees to ensure that signals record at the same time, leading to harmonized recording.

Microphone arrays are gaining increasing popularity in the audio industry because of the dynamic surround sound recording that they allow. There is literally no end or range to the number of microphones that they can use and how the sounds are synchronized to create dynamic recording. Some commercial audio devices use as many as 8 microphones to form a "listening beam" that locates and steers in on a talker to record speech. Others are strategically made so that they can create surround sound recording throughout a room, recording sounds as they occur in the direction from where the sounds originate. As new microphone arrays are designed and new applications made, we can only expect them to pick up more speed and carry more load in the microphone market.

#### 3.1.2.1.1 – Advantages

Microphone arrays have a distinct advantage as they place few constraints on the user, freeing them from the need to wear a microphone (as in lapel, headset, or mobile devices) or be near to and speaking towards the microphone (as in webcams and lecture microphones). A microphone array is a set of closely positioned microphones. Microphone arrays achieve better directionality than a single microphone by taking advantage of the fact that an incoming acoustic wave arrives at each of the microphones at a slightly different time. The main concepts that are used in the design of microphone arrays are beam forming, array directivity, and beam width. By combining the signals from all microphones, the microphone array can act like a directional microphone, forming what is called a "beam." This microphone array beam can be electronically managed to point to the originator of the sound, which is referred to in this paper as the "speaker."

In real time, the microphone array engine searches for the speaker position and acts as if it points a beam at the current speaker. The higher directivity of the microphone array reduces the amount of captured ambient noises and reverberated waves. Because the microphone array output contains less noise and has less reverberation than a single microphone, the stationary noise suppressor does a better job than it would with a signal from a single microphone.

During sound capturing, the microphone array software searches for the speaker's position and aims the capturing beam in that direction. If the person speaking moves, the beam will follow the sound source. The "mechanical" equivalent is to have two directional microphones: one constantly scans the workspace and measures the sound level and the other—the capturing microphone—points to the direction with highest sound level; that is, to the speaker.

The normal work band for speech capturing is from 200 Hz to 7,000 Hz, so wavelengths can vary by a factor of 35. This makes it difficult to provide a constant width of the microphone array beam across the entire work band. The problem is somewhat simpler in a typical office environment, where most of the noise energy is in the lower part of the frequency band—that is, below 750 Hz. Reverberation is also much stronger at these low frequencies and is practically absent above 4,000 Hz.

#### 3.1.2.1.2 – Disadvantages

Arrays have some disadvantages compared to individual microphones, including increased hardware and processing requirements due to the multiple channels and acoustic losses due to the distance from the speaker. The price of a good array microphone can be quite high. In addition, the size can make it impractical for some projects that require mobility and compact design.

### 3.1.2.2 – "Shotgun" Microphone

Shotgun microphones are long cylindrical devices that have the ability to pick up audio in a tight conical shape at a close range, while filtering out the rest. These types of microphones are used in "boom microphones" that are held just above the performer out of a camera's view box to isolate the audio of the performer in a video. The shotgun microphone is also used in handheld camera applications to record audio in the direction that the lens is facing. These microphones contain two major components that allow it to pick up sound in a single direction, the interference tube and the hypercardioid microphone[2] and are illustrated in Figure 3.2. These components give the shotgun microphone the ability to capture sound directionally. However, this system has limitations.

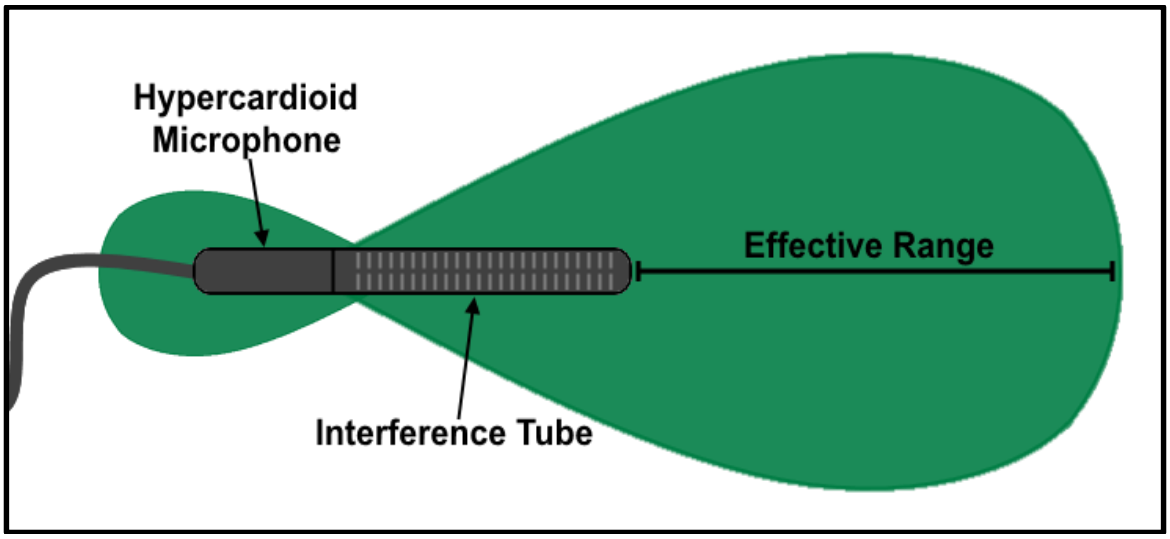


Figure 3.2 - Shotgun Microphone

A typical polar response for a shotgun microphone is shown to the right. Polar plots, seen in Figure 3.3, tell us the gain of the microphone for all the various directions that it points to in relation to the sound source. The 0° reference point is when the shotgun microphone is pointing directly in front of the sound source that it is recording. Here, at this 0° point, you can see that the microphone has its highest gain, which in this case is -11dB. As the microphone turns to its side, away from the sound source, the gain that it picks up begins to decrease. The 180° reference point is when the microphone is facing completely away from the sound source, pointing in the opposite direction. Here, at this position, is where the gain of the microphone is the lowest, which in this case is -28dB.

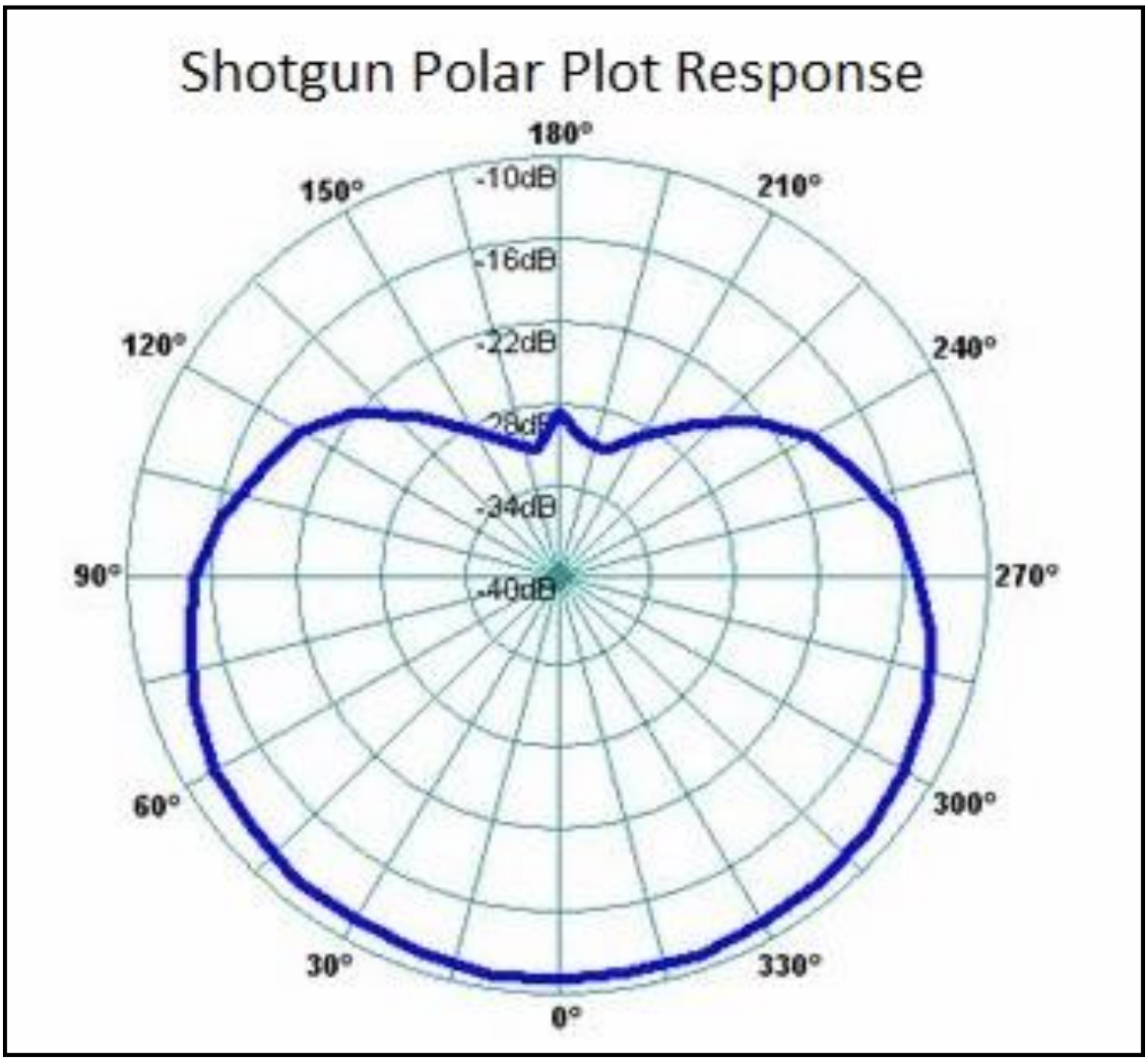


Figure 3.3 - Polar Response

These gain responses of a shotgun microphone illustrate that it picks up sound with high gain only when it points directly to the sound source and decreases when rotated to its side.

Shotgun microphones have the ability to cancel noise from surrounding inputs, but have a limited range of operation. In the case of the design of this project, if the presenter is a large distance away from the microphone, his voice will still sound faint and distant. Shotgun microphones are quite sensitive, so if the microphone experiences movement or force, the audio quality will not be as clear. These factors entail that the user of the device be close, but not too close, to the presenter when operating.

Restricting the user to operate at a close distance raises some other issues. Sitting close to the presenter may not always be possible. In addition, in having the device closer to the presenter, the presenter may leave the cone of operation of the microphone. If this happens, the microphone will have to be repositioned to face the presenter. Upon repositioning, the

microphone will experience lowered sound quality because of force being applied during movement. One may reduce the noise experience during redirection by supporting the shotgun microphone in a shock-absorbing mount.

#### 3.1.2.2.1 – Advantages

The advantage of shotgun microphones is they focus directly on the sound source in front and pick up the sound with high gain, while recording any other noises present in the environment very low, if at all, that may be to the sides or rear. This is advantageous because it can pick up the sound only that the user desires, directly in front, while disregarding any other unwanted sounds that may be present in the environment. Therefore, shotgun microphones are highly used in applications where only the sound directly in front of the microphone is to be recorded so that all other noises will be reduced and the sound source is in a fixed position, directly in front of the shotgun microphone, and does not move much.

Common uses of shotgun microphones are for talks or speeches in meetings, conferences, and lectures. In scenarios such as these, the speaker does not need to hold a microphone and speak into it or wire a microphone on his body in order to record his speech or lecture. He can just stand and talk at a distance as long as he talks to the front of the shotgun microphone. This way, the speaker can just focus on talking during his lecture without the cumbersomeness of having to have a microphone on him. Therefore, a shotgun microphone is advantageous in any situations where distance speaking will occur and the speaker will stay in vicinity where he talks in front of the microphone, such as a professor lecturing throughout a class who stands centered in the classroom throughout his talk.

#### 3.1.2.2.2 – Disadvantages

Shotgun microphones can be useful in some instances such a sound reinforcement as in talker on stage. The downsides of a shotgun microphone are that the type of application is its extreme dependence on the skill and alertness of the person point the microphone.

Many advantages are to be had with the implementation of this microphone type and fit into the scope of this design. The designed device must be able to have a reduced noise audio input for the Speech to Text software to utilize and the shotgun microphone promises this within a certain range. Cost to purchase a shotgun microphone is relatively inexpensive for entry level models, but increases as the range of operation and various other specifications increase. For this type of microphone to be implemented, the microphone would have to operate within specifications of the design.

### 3.1.2.3 – Wireless Microphone

The preceding sections were detailing devices designed to receiving audio from a relatively large distance away from the source while filtering out noises to isolate a high-quality input. This concept would be especially useful for the use in this project, as the audio is to

be processed remotely away from the source. Wireless microphones, however, would provide an advantage over the various other methods of audio acquisition. The presenter would have the microphone on their person and the project device would contain the receiver for the signal produced by the wireless microphone. Therefore, the presenter's audio would be acquired with minimal noise interference from in between the source and capture device.

There are various forms of wireless microphones that could be considered for use within the scope of this project: handheld, lavalier (lapel), headset, stationary. Each type presents advantages and disadvantages for the presenter. Handheld, wireless microphones would constrain the presenter's hand that is holding the device, but would not be invasive to the presenter's person. Lavalier, wireless microphones, being attached to the presenter's clothing, would not limit the presenter's ability to perform, but require the presenter to be "wired". Headset, wireless microphones, have similar characteristics to lavalier microphones in this application, but are worn on the presenter's head. Stationary, wireless microphones would allow the presenter full mobility and remove the invasive conditions, but would require the presenter to remain within a specific distance and project his/her voice in the direction of the microphone.

Having a wireless platform presents new issues to the user: power usage, broadcast/receiver range, signal interference. The wireless transmitter may have to be powered by a battery independent of the device attached to the laptop PC. If the broadcast range of the wireless microphone or receiver range of operation is smaller than the distance between the user and presenter audio will not be received at the device. Signal interference may be a factor in some situations if the signal channel cannot be modified. These are some of the factors that may inhibit the performance of the wireless microphone and, therefore, could make the device inoperable.

Within the scope of this project, the wireless microphone will not suffice. The device would require the presenter to have a wireless microphone to capture his/her voice. This removes the need for image tracking and repositioning unless the wireless signal receiver is required to be positioned to face the transmitter. Also, the ease of use of the project would be blemished as the presenter would have to be involved in the devices operation. The resulting device would be much smaller, however, as the previous methods of receiving a clear audio signal are larger than most wireless microphone receivers. Cost associated with the choosing this device is also a factor. Audio quality, operation range, form, size, and various other factors govern the cost of the wireless microphone.

## 3.1.3 – Camera

### 3.1.3.1 – Types of Embedded Cameras

There are a variety of types of digital embedded cameras available for use on the market today, but all share a common integral part, the image sensor. The primary goal of an

image sensor is to capture an optical image and translate it into electrical signals. The two main types of image sensor are charge-coupled devices (CCD) and active pixel sensors in complementary metal-oxide-semiconductor (CMOS). Either technology brings forth its own advantages and disadvantages that are not relevant to this project.

Digital image sensors on the market today are capable of being configured for a variety of output formats. These include, but are not limited to, interlaced video, progressive scan VGA, and JPEG. The specific output will be chosen by the consumer for their purposes.

There are primarily 2 types of video streaming formats: interlaced video and progressive scan. The primary difference between both methods is the method and frequency at which they update their scan lines. Scan lines are horizontal sets of pixels that run across the image that is being captured. As all scan lines are assembled, the frame becomes clear and recognizable.

Interlaced video is a format that updates only half of their scan lines every frame, greatly reducing the information required to keep the video stream running. Every other scan line is updated upon each captured frame, which is to say that the odd lines are updated on frame 1 and the even lines are updated on frame 2. This method is very beneficial for viewing applications, but may not be the correct method for image processing, since every frame is updated with only half of its scan lines and may throw continuity exceptions.

Progressive scan video is a much simpler format, but has the disadvantage of producing a stream that requires twice the information per frame and an interlaced video stream. In this format, every the scan line is updated progressively per frame. This means that if the user was sampling the stream for a stable image, one may be obtained that accurately represents the scene at that given time.

In the case of an image sensor that produces static JPEG format images, one may attempt to capture multiple consecutive frames with the intention of assembling a video stream. The JPEG format is widely used in image processing and can be easily integrated into image processing software.

In the case of this design, we have chosen a common CMOS design camera. The output of this device is up to a 2 megapixel, 1600x1200 pixel, JPEG format image updated at a maximum of 15 frames per second. The data from the captured image is transmitted over a serial RS232 connection with the PCB.

## 3.1.4 – Pan/Tilt Systems

A pan/tilt system is a mechanical device that allows two rotational degrees of freedom, which are named pan and tilt. Pan is rotation in the plane of the base, which – if the pan/tilt system is supporting a camera – will generally result in horizontal left and right motions relative to the camera's image. Tilt is rotation about a vector lying in the plane of the base and perpendicular to a vector in the direction the attached camera is supporting, and generally results in vertical up and down motions relative to the camera's image. Rotation

which causes the camera's direction vector to move towards the right (left) relative to the image is defined as positive (negative) panning, and rotation which causes the camera's direction vector to move up (down) relative to the image is defined as positive (negative) tilting in this application.

In Figure 3.4, a rotation of component A about the point of attachment of components A and B is called tilting, and a rotation of component B about the point of attachment of component B and the base is called panning. A camera module would be attached to component A.

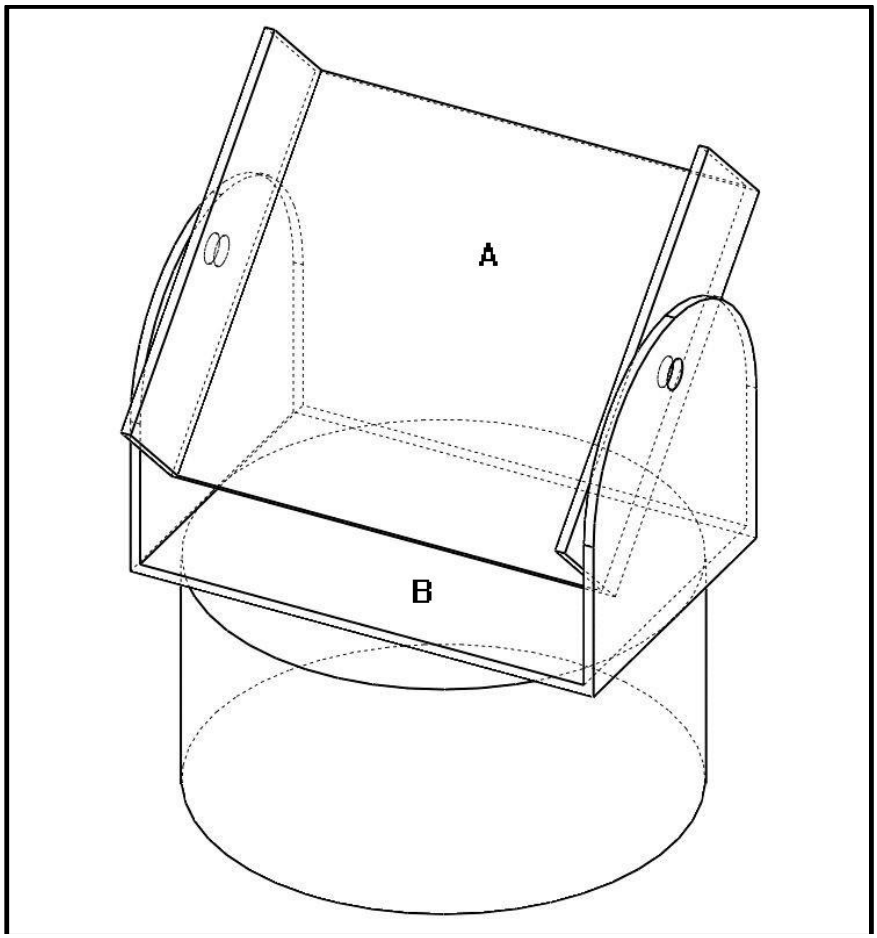


Figure 3.4

The result of mounting a camera on a pan/tilt system is that the camera can adjust its direction horizontally and vertically relative to its image. This allows the camera to monitor an angular region which exceeds the angular region defined by the viewing angle of the camera itself, although it is still limited to viewing an angular region the size of its built in region at one time. An additional limitation is that there is no roll angle degree of freedom. This means that the camera cannot reorient its image by rotating about its direction vector. While this may be an important requirement in applications where the base is not fixed, it is unimportant in the present application, where roll angle freedom would only serve to

complicate the image processing and motion control software, and would provide no real benefit.

Movement of a pan/tilt system is generally accomplished through rotational electric servo motors, and this is the method used in this application. Servo motors are discussed in section 3.1.5.

## 3.1.5 – Servo Motors

A servo motor is a motor – in this application, an electric motor – which provides some method of feedback to its controller about the motor shaft's angular position, speed, or both. The three most common ways that this information is obtained are optical encoders, magnetic encoders, and potentiometers.

Encoders are digital devices that have some way of detecting that the shaft has moved through a certain angle. A common feedback communication method is two wire quadrature encoding, which allows signaling of an interval detection and the direction that the angular interval was travelled in. This is accomplished by cycling the voltages on the pair of wires through four two bit states after each interval detection in an order such successive states are different by only one bit from the previous state, as in Gray code. For example, the reading "00, 01, 11, 10, 00, 01" indicates that six rotation intervals were detected with the shaft rotating in one direction, while reading "00, 10, 11, 01, 00, 10" indicates that six rotation intervals were detected with the shaft rotating in the opposite direction. When using an incremental encoder, the controller counts the number of intervals that the shaft has moved in both directions and calculates the position based off of this. Angular velocity can also be calculated by monitoring the time of the detected signals and numerically differentiating the position, although differentiation in this way has the effect of amplifying noise. Encoder servo motors can be purchased with varying numbers of intervals per shaft rotation, and can generally be chosen to be as precise as necessary for the application. However, precision comes at an increased cost – both through a higher price and through increased control system complexity – when compared with potentiometer servo motors.

Potentiometers are analog devices that change their resistance based off of the position of a contact which moves along the potentiometer. The two end contacts have a fixed resistance between them. The resistances between one end contact and the center contact and between the other end contact and the center contact are both variable. These two variable resistances sum to the constant resistance between the two end contacts. If a constant voltage is applied across the two end contacts, the potentiometer acts as a variable voltage divider when measured at the center contact, with voltage as the dependent variable and position (or angular position, in this case) as the independent variable.

In servo motors, the potentiometer is in the shape of a circular arc, and is fixed relative to the casing of the motor. The shaft has an extrusion which sets the position of the variable center contact of the potentiometer. The potentiometer is integrated in a network of

resistors and capacitors which serve to condition the voltage signal. A precision voltage reference is placed in parallel with this network, and the voltage at the center contact is measured and converted into a position signal for the motor shaft.

If the potentiometer was a complete circle instead of a circular arc so that the motor could assume positions at all 360 degrees, there would not be unique voltage values for each position, and the motor would not be successful. This limits the motor to a finite angular range less than 360 degrees which it cannot exceed. Additionally, there are generally physical constraints built into the motor which will not allow it to rotate past a certain angle. Encoder servo motors can generally spin continuously in both directions as many times as is necessary.

Potentiometer servo motors have the advantage of giving exact position, which allows the control system to tell the motor to go to its home position very simply. Conversely, a brushed DC motor with an incremental optical encoder could just be rotated into the optimum starting position and use that positon as home for the remainder of its run time. Home position will be defined as the center of the servo motor's range of motion. With incremental encoders, the motor generally needs another sensor to detect when the motor is in its home position. Absolute position encoders do exist, and are available on servo motors. The price point, however, is much higher than even price of incremental encoder servo motors.

Potentiometer servo motors – specifically, RC servo motors (described in section 3.1.6) will be used in this application for the tilt motor. Physical testing is required to determine if a low-end RC Servo can provide sufficiently smooth motion through small angles, or if a pricier option such as a high-end digital RC servo or a brushed DC motor with an incremental optical encoder would be justified. The cause for concern will also be described in section 3.1.6.

## 3.1.6 – Servo Control

The most prevalent form of potentiometer servo motors are RC servo motors, which are usually called RC servos (or just servos). The name of these motors comes from their use in remote-controlled vehicle applications. They are extremely cheap, relative to the price of most other types of servo motors. Their defining feature is that they have a built-in feedback control system.
The feedback control system for RC servos is usually an integrated circuit designed specifically for use in these motors. They have three external inputs: ground, supply voltage, and a control input. The IC has an integrated voltage reference and voltage sensor for use with the potentiometer, and has power-switching circuits to give power to the motor. This design allows RC servos to be very compact and easy to use. Originally, RC servo IC control systems were mostly analog. Now, however, there are RC Servos with digital control systems. Therefore, RC servos are now grouped into the categories of 'analog RC servos' and 'digital RC servos.' Digital servos generally perform better in all

categories than analog servos, with the exception of power consumption. Most notably, digital servos allow programmers to be used to change control system parameters.

The control input is a pulse width modulation (henceforth, PWM) input. Duty cycle is not considered when determining the position intended by the pulse. Instead, position is indicated by the length of the pulse (the 'on' time of the PWM signal). Generally, a 1500 microsecond pulse indicates the home position (also known as the neutral position, or zero degree position), but this may vary by motor model. A range of pulse-widths are associated one-to-one with a range of positions. That is, to move the motor shaft to a certain angle, the corresponding pulse-width is sent to the motor. Depending on the motor model, if no new pulses are given after the motor reaches the new position, the shaft will either be held at this position forcefully or allowed to spin freely. The latter is generally more common, but an RC servo used in an application must be tested to determine in which manner it operates, and the control system must take this information into account.

As mentioned in section 3.1.5, the motor is limited to a finite range of angles. Care must be taken not to attempt to move outside of this angular range, as the motor will attempt to do so and damage itself, unless it is given a new valid position before it overheats. For this reason, the motor control system for this application will saturate the motor's maximum angular positions at angles less than the maximum range given by the distributor as a factor of safety.
When a control pulse is received, the motor will attempt to move directly to that position. There is no speed control inherent in the control signal, although the frequency at which pulses are given will affect the effort of the motor to get to this position when the motor is heavily loaded, and the motor will move quicker when making a large position change than when making a smaller position change (this is what allows RC servos to be internally modified to sacrifice position control in exchange for speed control and continuous rotation). The effect of this on the present application will now be explained.

Suppose that the motion control system is given instructions to move to a new position every 500 milliseconds by the computer vision system, and that a pulse indicating the new position is immediately sent to the motor. If the position increment is small, as is likely in the case of tracking a professor giving a lecture, the motor will arrive at the new position before the next position instructions are received, and will sit idle at this new position until it receives the next pulse. This will result in jerky movement. In order to get smooth movement, successive pulses must be sent in this timeframe. The first pulse should have a width equal to the pulse width associated with the current position plus a small amount. Each pulse should be increased by this amount until the next instruction is received from the computer vision system. Testing will be required to find the relationship between the small incremental pulse width amount and the difference between the target position's associated pulse width and the original position's pulse width which results in the best performance, if this method is implemented.

The reason why it may not be implemented is the typical RC servo's dead band width, which is generally on the order of a few microseconds. While the dead band width of the servo is useful in many applications, in the present application it drastically limits the

effectiveness of the pulse width sweep which was just described on providing smooth travel along the prescribed trajectory. There are several methods that can be used to attempt to rectify this issue.

The first is to place a transistor (or some other form of variable impedance) in series with a standard RC servo motor. This would allow for some manner of speed control, although it would not be nearly as precise or smooth as with the high resolution pulse width modulation (HRPWM is described in section 4.1.2.2) sweep on a dead band free motor. Another downside is that this method will use more power and produce more heat than an uncompensated servo, although it would not be as detrimental to battery life as the other solutions likely would. Finally, the variable impedance of the transistor would unavoidably cause fluctuations not only on the motor, but on its control IC. Generally, great lengths are gone through to reduce such fluctuations on supply voltages, as it can cause errors in the ICs operation.

The other method being considered is to replace the motor. The first of two potential motor replacement types is a digital RC servo (as opposed to the cheaper and more common analog RC servos). Digital RC servos have programmable dead band widths (among other things), so its detrimental effects can be mitigated to an extent. However, RC servos which have the capability to decrease their dead band width to a level which would allow the HRPWM of the Piccolo to be beneficial are generally much more expensive. They are also generally designed for much larger loads than this application will encounter, and as such they consume more electrical power than a servo with a more appropriate mechanical power rating. Further contributing to a decreased battery life, digital RC servos generally consume much more power than comparable analog RC servos, due to the fact that they provide more frequent pulses of power to the motor. This increased control effort does, however, come with benefits to responsiveness, accuracy, precision, and general utility, as digital servo motors are programmable.

The other potential replacement motor type is a geared, brushed DC motor with an incremental optical encoder. Options of this type exist at the same price point as dead band free RC Servos, and would provide increased performance in almost every category if certain accommodations are made, such as increased motor control system computational effort and control effort. The benefits include greatly increased pan angular range, more accurate and noise-resistant position detection, much less power consumption when the motor is not moving than with RC servos, and position feedback (which both enables and requires more complex and higher performance control algorithms for the Piccolo). Disadvantages include higher power consumption compared to a standard size analog RC servo when changing position under loads, although it would likely not be too much larger than that of the aforementioned high-end digital RC servo.

# 3.2 – Software Research

## 3.2.1 – Video Software

### 3.2.1.1 – CMUcam

The CMUcam is a small device the provides simple vision capabilities to an embedded system. We were interested in this project because it closely followed the idea of finding targets and tracking them, along with other image processing features.. The benefit that the CMUcam has is that the object it is intended to track is generally a brightly colored object of a generic geometric shape. What we liked about the CMU project is that it performs object detection and following in an embedded environment. The CMUcam project is currently in its fifth iteration and what we can learn through their iteration process is that when the scope of what the user is tracking is kept simple, but the number of items tracked and the accuracy skyrockets when computational power increases. The original processor was a simple 16 pin microcontroller and now they are using a 204 MHz dual core processor. This has helped to show us how each power level of hardware can meet each specific task. Also this has helped to show that since our problem is relatively complicated from a software stand point we need hardware that will let the software do its job. Overall the CMUcam was where we derived several of our hardware baselines, as well as some of what we might need for our software.

### 3.2.1.2 – Operating Systems

The next thing that we researched was operating systems. What we didn't realize was the vast amount and scope of embedded operating systems. There uses range from simple user interfaces to complex backend server management. We looked at these operating systems for a few key features. First we wanted the operating system to be free, because we has a small budget we didn't want to waste extra funds buying proprietary software. Secondly we wanted to have the system general enough so that when we put our software onto it we can get the functionality we wanted. This is where we had the most trouble, because there are so many kinds of systems it is hard to figure out which ones will suit the project's needs the best. We finally wanted an embedded operating system that required the least overhead and utilized the hardware to a great extent. While embedded operating systems are all meant to be resource efficient, they are not all equally efficient in these areas. We quickly came to realize that a Linux embedded operating system would be optimal for this project. From there we narrowed down our search to operating systems that had desktop like qualities, but without taking up a desktop amount of resources. We also want to implement a Computer Vision design in our program which requires a lot of memory manipulation. The operating system is going to take a significant portion of the heavy lifting off of our hands so that we can focus on optimizing our program. After going through Ubuntu and other kernels we determined that Debian was not only an easily accessible embedded operating system, but it also had great options in terms of functionality. The operating

system can operate in many ways. First like a traditional operating system with a GUI interface. Secondly more like traditional embedded operating systems through a command line interface. We plan on using both functionalities to our advantage during the testing and implementation, though we want the final product to be self-sustaining and not need any input from an outside user to operate correctly.

### 3.2.1.3 – Programming

Embedded devices are generally quite compact computers, and as such do not traditionally offer much variety in terms of programming language variety. What we have found is the most basic to the most advanced embedded devices are generally programmed in C or C++. That being stated each device generally comes with its own utilities that can make its programming environment more robust. When the embedded device starts to get powerful enough to put an operating system on it, more programming languages become available. For example if you put a Linux device onto a board, not only will it run C programs, but it can also accept programs in object oriented languages like Java or a scripting language like Python. If a Microsoft operating system is put onto a board, it will accept programs in C#, .Net, as well as the traditional C and C++ languages. For the most part programming on an embedded device is very similar to programming on a desktop environment. The main difference is that access to the terminal of the embedded device is generally through a desktop, and that there is a limitation to the programs size.

### 3.2.1.4 – Computer Vision Software

Computer vision is a large part of our project and because of this finding libraries was a large task. OpenCV was the first and obvious choice for a solid Computer Vision library. Some of our teammates have had experience using the library. OpenCV is also a great environment because it has multiple language support. So no matter what the hardware and/or operating system that we choose this computer vision library adds functionality and suport. The only downside is that there is a fairly steep learning curve to this library. With it you have to know what type of masking you need and what kind of data you are processing. If we choose to use this software our worry is that we may not have enough time to learn it well enough to make it useful to our project.

SimpleCV is an extension of OpenCV in a python framework. The unique thing about SimpleCV is that it will handle all of the data pathing and manipulation for the user. This is helpful to our project because it will cut down on the learning time and we can then dedicate it to choosing and debugging different algorithms. The benefit of having a singular programming language is that it cuts down on the file size, which because we will have limited space is a huge consideration. The other benefit to this particular library is that is has its own instruction manual that was purpose built for it. While there are many guides for computer vision, most are just general and take examples from many different computer vision libraries. This will be helpful because it will give us a clear insight to what our program should look like.

The final computer vision library we found by mistake. MATLAB has a computer vision and image manipulation library. We found this while purchasing the product for a class. Being a proprietary software there is a barrier to entry with the cost of the software. Again this might be a problem with our group's relatively small budget. MATLAB has the benefit of being an easy to use environment, which can produce the exact result we are looking for. MATLAB also has the benefit of being able to export its programs to any form factor that we choose. This is great because if need be we can export to the specific binaries that are used by our board's chipset, bypassing the need for an operating system. We can also export it in a programming language that is optimized for the embedded operating system we choose. MATLAB is the most streamlined and complete software package that is available.

### 3.2.1.5 – Facial detection

The first research that was done into facial recognition was into algorithms that existed in our potential computer vision software libraries. The first library we looked into was OpenCV, because it is generally the basis in which the other CV libraries went off of.

The first facial recognition algorithm we came upon was the Eigenfaces method. This method takes a high dimensional image, finds key features and then compresses these features into a lower dimensional image, where there are fewer variances in the picture information. The algorithm then trains on given lower dimensional images and selects features that are common among the many different types of faces. After this training is done you can send a random image through the algorithm, it will compress the image, extract its feature and then run analytics on it. The key feature of Eigen faces is to make the algorithm more or less accurate you simply have to increase or decrease the number of passes through the trained algorithm.

The second facial recognition that we came across is one of the oldest and very commonly used in embedded devised called the Viola-Jones algorithm. This algorithm takes feature classifiers and then tests them to see how accurate they are. What was found was that there was no one classifier that would accurately detect a face. However what they did was to changes the algorithm to use many classifiers that together would build accuracy to facial detection. To make sure and choose the best classifiers the object detection framework is run through an adaptive boosting algorithm. What this does is it takes machine learning techniques and utilizes them to teach the algorithm which classifiers are the best for choosing faces. What the user then does is choose how many classifiers to use and the boosting method will find that number of classifiers, and what's even better is those will always be the best available classifiers. So as you increase classifiers, until you have too many and the algorithm slows down, the accuracy of the method increases.

### 3.2.1.6 – Facial tracking

Facial tracking turned out to be the hardest subject to research. Most of the scholarly articles that have been published on the subject are from france and are written in french. This made reading up on the topic cumbersome, and finding useful information difficult.

What ended up giving us a good lead was facial tracking for the defence industry. After scavenging through many different articles we found that there were a few algorithms that stood out and were used in many different situations. The first such algorithm is the KLT approach, which uses optical flow to determine where faces were and to determine where they were going to accurately track them across the scene. The second set of algorithms where what are called mean shift algorithms. WHat the mean shift approach does is to track density gradients. What the algorithm does it to take feature vectors of an image and then find where these features are densely clustered. This density is then fed through a function to detemine its importance and is then tracked if deemed that this is indeed worthy of tracking.

### 3.2.1.7 – Possible Problems

The biggest problem associated with facial tracking and facial recognition is privacy. These technologies make it more difficult to hide a person's identity than ever before. An not only that but more and more companies are using these technologies to construct assumptions about the individual who is being recognized or tracked. This leads to malicious individuals who might use facial recognition to stalk a person. If one of these individuals comes across old photos of their target they can determine where they work or live and follow that individuals movements. This also can spill over into the workplace where the potential employer takes a photo of the person being interviewed and then runs a facial recognition algorithm and searches the internet for any other picture of the person that might exist.

With these problems already at the forefront of people's minds we need to develop our product in a way that alleviates concerns for privacy. We can set up recommended models, so that people who are being tracked by our product can sign off that they understand that they are going to be tracked. There is also the uncomfortable feeling that people get when they feel like they are being watched by a machine. There is generally a mistrust as to what is going on in the software that has become more pervasive as technologies continue to come out that track individuals for security and surveillance. To try to overcome this mistrust we plan on doing several things. First of all since our target population is in an academic setting we can have information about the product available on request from those who might be using the product or those who have someone who uses our product in their classroom. We can also have indicators on the device that though it is tracking the presenter it is not saving that data, only the speech information from the class that could be obtained from any other student. Since our goal will be to have the robot used by the Office of Student Disabilities, we can have them put up and information page that describes exactly what our product does.

# 4.0 – Hardware

## 4.1 – Processor

### 4.1.1 – Processor Specifications

There will be two PCBs in this application, following a recommendation from the instructor of this course. The motivation for not having a single board running everything is the ability to work on things separately and simultaneously, and to reduce the possibility of one group member's code causing unexpected results in another's. The primary board will be a BeagleBone Black, the main processor of which is an AM3359 ARM Cortex A8 microprocessor. The secondary board will be a custom-built board for a TMS320F28069PNT Piccolo Microcontroller by Texas Instruments.

#### 4.1.1.1 – AM3359 (BeagleBone Black MPU)

The BeagleBone Black is a development board by Texas Instruments featuring an AM335x ARM Cortex A8 microprocessor with a 1 GHz clock rate and 32 bit registers, and has features to increase throughput during video and image processing computations. The BeagleBone Black runs various distributions of Linux, providing easy integration of many important libraries and communication protocols. This processor and board will perform most of the functions of the automatic note-taking device.

The main function this processor will be expected to perform is video processing using the OpenCV computer vision libraries. To this end, the MPU includes a NEON SIMD (single instruction multiple data) coprocessor. The NEON has 32 registers, each 64 bits wide, which can be used to perform simultaneous computations which do not rely on each other's results. These types of computations are typical in image and video processing, and will allow for much faster processing of the video data to provide more frequent tracking instructions to the motion control board. A common saying is to "make the common case fast," and that is the purpose of the NEON coprocessor.

The BeagleBone Black will also receive an audio stream from the directional microphone and send it through a USB connection to a computer running voice recognition software. Both the video data and audio data will benefit from the AM3359's enhanced DMA Controller, which provides 64 DMA channels and 8 Quick DMA (QDMA) channels to allow the processor to continue computations while transfers are ongoing.

A real time clock (RTC) is integrated in the processor, which provides options for easily expanding the utility of the note-taker. An example is time-stamping of information, which can be integrated into the final output of the voice recognition software for cataloging transcriptions by date and class.

### 4.1.1.2 – TMS320F28069PNT (TI Piccolo MCU)

The TMS320F28069 is a microcontroller in the Piccolo family by Texas Instruments. It has a C28x floating-point DSP core, which is in the C2000 series of TI's DSPs. The C28x core has a 90 MHz clock rate and 32 bit registers. This MCU will run the motion control system software, and will provide PWM outputs to the motors.

The AM3359, described in section 4.1.1.1, will be running a full Linux distribution. As such, it is not suited to real-time applications. In contrast, the Piccolo will be dedicated to motion control and will be suitable for such tasks. If necessary, the Piccolo MCU can run Texas Instrument's real time operating system (RTOS), TI-RTOS. The use of a RTOS takes slight overhead but allows for precisely timed operations

The MCU includes one Programmable Control Law Accelerator (CLA). This is a 32 bit math acceleration unit, which can perform calculations, including floating point calculations, while the processor is focused on other tasks. The CLA will be able to perform calculations for the upcoming PWM intervals while the processor carries out the PWM sweep accurately. A DMA with 6 channels will facilitate transfer of results from the CLA to memory and from memory to the processor/PWM peripherals as appropriate.

If a brushed DC motor with an encoder is used, a discrete-time PID or PI controller will most likely be implemented in the Piccolo. As the core is a floating point DSP with an independently executing floating point coprocessor and 6 DMA channels, running this algorithm should not significantly decrease the amount of time spent in power-saving modes, which drastically reduce power consumed by the microcontroller.

## 4.1.2 – Input/Output

### 4.1.2.1 – AM3359 (BeagleBone Black MPU)

The BeagleBone Black, as mentioned in 4.1.1.1, runs one of several available Linux distributions. This provides excellent support for communication schemes and allows easy use of open source libraries. The BeagleBone Black has a number of available I/O peripherals included, including USB, SPI, UART, I2C, and 2 PRUs which allow very customizable input and output methods.

The Linux distribution handles drivers and low-level protocols, making it easy to communicate with the computer using one of the two USB 2.0 peripherals. One of these will be used – with the computer as the USB host and the BeagleBone Black as the USB client – to send the computer the audio stream for voice recognition. A tentative expansion of the note-taking application would see this USB connection sending the video from the pan/tilt camera in addition to the audio stream for increased utility. The other USB peripheral, which is capable of operating as a USB host, will be used to receive the audio from the microphone, while a UART peripheral will be used to receive data from the camera using the RS232 standard.

The BeagleBone Black will, after calculating the lecturer's position relative to the center of the image, transmit a displacement vector to the Piccolo. This will be accomplished using one of the SPI buses. The BeagleBone Black will be the master and the Piccolo will be the slave, and communication will be one-way; data will be sent from the BeagleBone Black to the Piccolo, with the Piccolo sending no meaningful data back through the SPI bus. In the event that the Piccolo detects that the angular displacement has reached the saturation value, the general purpose input and output (GPIO) pins of the BeagleBone Black can be used as external interrupts to convey this information. Three BeagleBone Black GPIO pins will be used for this purpose. One pin will trigger the interrupt, and the other two will convey the nature of the problem: saturated pan (negative), saturated pan (positive), saturated tilt (negative), or saturated tilt (positive).

### 4.1.2.2 – TMS320F28069PNT (TI Piccolo MCU)

The Piccolo includes USB, SPI, UART, I2C, and other buses which are irrelevant to the present application. The Piccolo also includes Enhanced Capture (eCAP) and High-Resolution Capture (HRCAP) modules. Most importantly, the Piccolo includes 16 PWM buses, 8 of which are Enhanced Pulse Width Modulation (ePWM) buses, providing 8 High Resolution Pulse-Width Modulation (HRPWM) channels for precise sweeping of pulse widths.

As previously mentioned, in section 4.1.2.1, the SPI bus will be used to receive displacement vector instructions from the BeagleBone Black. This is straightforward. The eCAP module allows for precise time between events to be measured. Using the SPI chip select pin as the eCAP trigger pin, this allows precise for precise timing of the interval between the incoming displacement information. This information can be used in the pulse width sweep calculations to provide smoother control of the pan/tilt camera's position.

As mentioned, the key feature is the Enhanced Pulse Width Modulation bus, which features High Resolution Pulse Width Modulation. HRPWM utilizes micro edge positioning (MEP) technology to achieve very high resolution pulse-widths. Accuracy is on the order of 150 picoseconds, which is $1/10{,}000{,}000^{th}$ of the width of the home position pulse (1500 microseconds, of a typical RC Servo). MEP logic works by dividing up one system clock cycle of a standard pulse width modulator. A regular PWM clock signal is shown in figure F4.1.2.2A, while a HRPWM utilizing MEP is shown in figure F4.1.2.2B. MEP logic can be applied to provide precise control of the rising edge of the pulse, the falling edge of the pulse, or both the rising and falling edge of the pulse. This allows for not only increased resolution when setting the pulse width, but also less error in the pulse width. It is impossible to set the edge exactly at a certain time, but the edge is positioned in one of the two intervals between PWM subdivisions which border a chosen subdivision instead of one of two intervals between PWM divisions which border a chosen division.

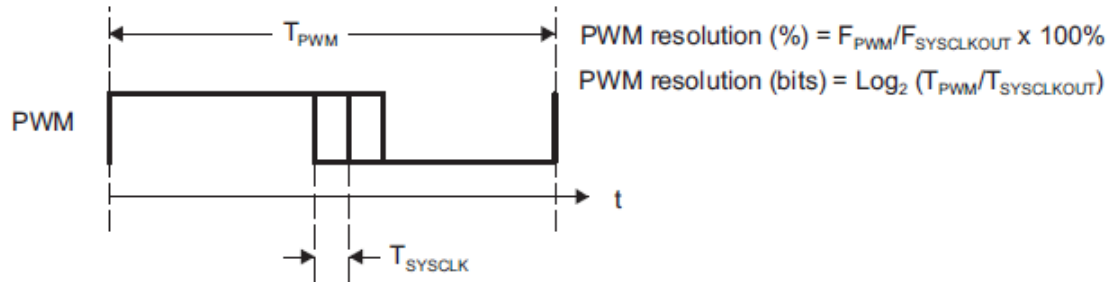



Figure F4.1.2.2A – Standard PWM Clock Cycle
COURTESY OF TEXAS INSTRUMENTS



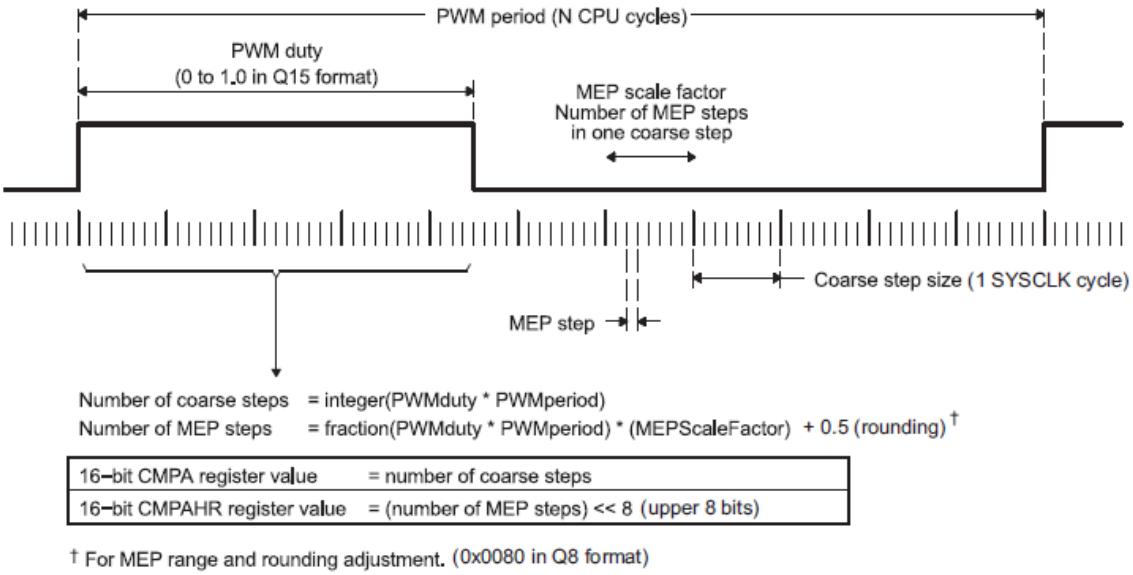



Figure F4.1.2.2B – Clock Cycle of HRPWM Using MEP
COURTESY OF TEXAS INSTRUMENTS

The result of using an ePWM peripheral with HRPWM instead of a standard PWM peripheral is that the individual pulse widths in the pulse width sweep can be precisely chosen. This reduces inconsistencies in the motion due to rounding to the nearest standard PWM division, and it allows the precise timing between the SPI transmissions from the eCAP to be taken advantage of. In the event that a brushed DC motor with an optical servo is used, the HRPWM can still be taken advantage of. The power delivered to a motor driven by an H-Bridge circuit can be controlled by providing a PWM control signal to one of the transistors' gates/bases. A HRPWM signal routed to this gate/base would provide more precise control over the power delivered to the motor, especially at operating points where the duty cycle of the PWM control signal is close to zero, but as the duty cycle gets higher the increased precision would be less effective at reducing the error (as a percentage) in the power delivered to the motor.

The High Resolution Capture module provides the ability to use capacitive touch sensors and many other applications, if the utility of the application is desired to be expanded through a touch interface on the exterior of the device.

# 4.2 – Microphone

## 4.2.1 – Specifications

Table 4.2.1 shows the manufacture's specifications for the selected microphone to be used in the final design.

Audio-Technica ATR6550:

| | |
|---|---|
| Element | Mono Condenser |
| Polar Pattern (Modes) | Normal: Cardioid<br>Tele: Supercardioid |
| Frequency Response | 70Hz - 18kHz |
| Open Circuit Sensitivity | Normal: -56 dB<br>Tele: -45 dB |
| Impedance | Normal: 1,000 ohms<br>Tele: 2,200 ohms |
| Cables | 1m cable with 3.5mm TRC plug |
| Weight | 4 oz |

Table 4.2.1 - Audio-Technica ATR6550 specifications

Device was chosen based upon the description from the manufacturer. “In its ‘Tele’ range setting, this cardioid condenser is engineered to pick up dialogue and sound effects at a distance, while bypassing ambient noise such as traffic, air-handling systems, room reverberation and mechanically coupled vibrations.”

## 4.2.2 – Output Interface

The output cable of the microphone is a male 3.5mm TRS connection, where TRS stands for “tip, ring and sleeve,” which describes the shape of the male audio connector. In Phase I, this cable will extend and connect directly into the microphone port on the laptop PC. If the PCB integrates a 3.5mm port, the microphone will attach to the PCB. The audio will pass through the PCB and out through the USB cable into the laptop PC.

### 4.2.3 – Integration

The microphone will be mounted to the head of the pan/tilt device. Since the microphone is very sensitive to force and movement, and the pan/tilt device will point towards the presenter, the mount needs to compensate for the movement. For easy removal and replacement, the mount also needs to be easily attached. Further testing will have to be done to determine what mount will work in this application.

All audio will be processed on the laptop PC and, therefore, the microphone must output to that platform. In the event that the PCB does not accommodate the integration of an audio input, the output cable from the microphone will run alongside and be tethered to the USB cable to attach to the laptop PC.

## 4.3 – Cameras

### 4.3.1 – Specifications

Tables 4.3.1A and 4.3.1B show the manufactures' specifications for the selected cameras to be used in the final design.

LS-Y201-2MP:

| | |
|---|---|
| Sensor | 2 Megapixel CMOS |
| Frame Rate | 15fps at 1600x1200 |
| Output | JPEG through serial RS232 |
| Baud Rate | 115200 |
| Voltage | 3.3V or 5V DC |
| Current | 80-100mA |
| Size | 32mm x 32mm |

Table 4.3.1A - LS-Y201-2MP specifications

Logitech B910 HD Webcam:

| | |
|---|---|
| Field of view | 78-degree wide-angle |
| Color Depth | 24-bit true color |
| Frame Rate | 30 frames per second @ 720p and VGA mode |
| Photo Capture Resolution | 5 million pixels |
| Microphone | Built-in dual microphones |
| Output Interface | Hi-speed USB 2.0 |
| HD video (720 x 1280) at 30 fps | • CPU: Quad core 2.0 GHz or higher<br>• RAM: 2 GB or more<br>• Upstream bandwidth: 1.5 mbps |
| VGA video (640 x 480) at 30 fps | • CPU: Dual core 1.9 GHz or higher<br>• RAM: 1 GB or higher<br>• Upstream bandwidth: 600 kbps |
| CIF (352 x 288) at 15 fps | • CPU: Single core 1.5 GHz or higher<br>• RAM: 512 MB or higher<br>• Upstream bandwidth: 250 kbps |

Table 4.3.1B - Webcam specifications

## 4.3.2 – Output Interface

The embedded camera (LS-Y201-2MP) mounted on the ANTI recorder device has an interface with 4 pins. 2 pins are dedicated to power input while the other are reserved for serial RS232 output. Serial specifications are listed in section [4.3.1].

The webcam, mounted to the laptop PC and used in Phases II and III, has a USB interface with the laptop PC and has full specifications listed in section [4.3.1].

## 4.3.3 – Integration

Embedded camera (LS-Y201-2MP) is mounted to the head of the pan/tilt device on the ANTI recorder. This camera is moved by the pan/tilt device to keep the presenter within view for tracking purposes. Phase III uses the output from the embedded camera to overlay the presenter's face in the recorded video. All 4 wires from the embedded camera will run to the PCB. Power will be obtained by the pins on the PCB and the 2 other wires will connect to pins for RS232 communication with the PCB. JPEG format images, of undetermined resolution, will be obtained at approximately 15 frames per second by the

PCB and held on a local memory device, detailed in section [4.4] for processing. In Phase III, the frames will be forwarded to the laptop PC via USB interface.

The laptop webcam will only be used in Phases II and III. This webcam will be attached to the laptop PC and will face towards the scene. The webcam was designed by the manufacturer for mounting to the top of laptop PCs and interfacing via USB. Drivers on the laptop PC will be required to interface with the device and are distributed by the manufacturer.

# 4.4 – Memory Components

## 4.4.1 – AM3359 (BeagleBone Black MPU)

The AM3359 has a three-level cache. There is 32 KB of instruction cache and 32 KB of L1 data cache with parity bit error detection, which detects single-bit errors; there is 256 KB of L2 data cache with error correction code; and there is 64 KB of internal L3 RAM shared between all masters in the processor.

The BeagleBone Black has 512 MB of DDR3 RAM external to the processor. The enhanced DMA controller mentioned in section 4.1.1.1 works for transferring data to and from both the internal memory and the external memory.

Non-volatile program memory is provided in the form of flash via a 4 GB 8-bit embedded multimedia card (eMMC).

## 4.4.2 TMS320F28069PNT (TI Piccolo MCU)

The Piccolo contains 256 KB of flash memory for program storage and 100 KB of RAM. In addition, it contains 2 KB of one-time programmable memory, which can be used to store identifying information about the particular unit (serial numbers, etc).

As mentioned in section 4.1.1.2, the Piccolo contains a 6 channel DMA controller for transfer of data between memory and the peripherals.

# 4.5 – Printed Circuit Board (PCB)

Designing a printed circuit board for a Cortex A8 processor running at 1 GHz, its peripherals, external memory, terminal connections, and taking into account EMI noise, thermal noise, signal reflections, and power considerations was considered. It was discovered that the BeagleBone Black – the board which will be acquired for this project – took years to design and work out the bugs. In addition to time constraints, financial constraints prohibit us from ordering the numerous repeated prototype PCBs that would likely be required due to the debugging procedure. Making such a board could be a challenging project for a Senior Design group, if that was the end goal. Since this is only

one component, designing such a board was deemed infeasible and out of the scope of this project. This board will be acquired, as described in section 6.1.

The Piccolo – which is an MCU, not an MPU like the Cortex A8 – on the other hand, was deemed feasible to design within the aforementioned constraints, as all of the particularly challenging component integrations have already been done inside the MCU. The remainder of section 4.5 will detail the design of a printed circuit board featuring the Piccolo.

## 4.5.1 – Related Designs used in Practice

Texas Instruments produces a number of evaluation and development boards for their microcontrollers and microprocessors. Among the cheapest and simplest are the controlSTICK evaluation boards. Specifically, part number TMDX28069USB was examined. A picture of this product can be seen in figure F4.5.1A for reference. It is a very minimal design.

Figure F4.5.1A - controlSTICK
COURTESY OF TEXAS INSTRUMENTS

The controlSTICK is an evaluation board about the size of a USB flash drive, with a USB port on one end. The USB port is used to power the device and to communicate with it through the on-board JTAG emulation IC. The schematic from this board, which can be found in greater detail in the controlSUITE application by Texas Instruments, is displayed in figure F4.5.1B. The narrow, vertically placed IC on the right-hand side, which is connected to the IC in the upper left, is the Piccolo MCU. The IC in the upper left is a TPS73233 Linear Voltage Regulator with an output voltage of 3.3V. By examining the hardware files of this controlSTICK, the hardware files for the controlCARD featuring the same processor, and especially from section 4.1 of the Piccolo TMS320F2806x ata sheet, which gives descriptions of all of the pins, the required supporting circuitry can be determined.

The on-board JTAG emulation allows for use and debugging of this evaluation board without the need to purchase a JTAG programmer.

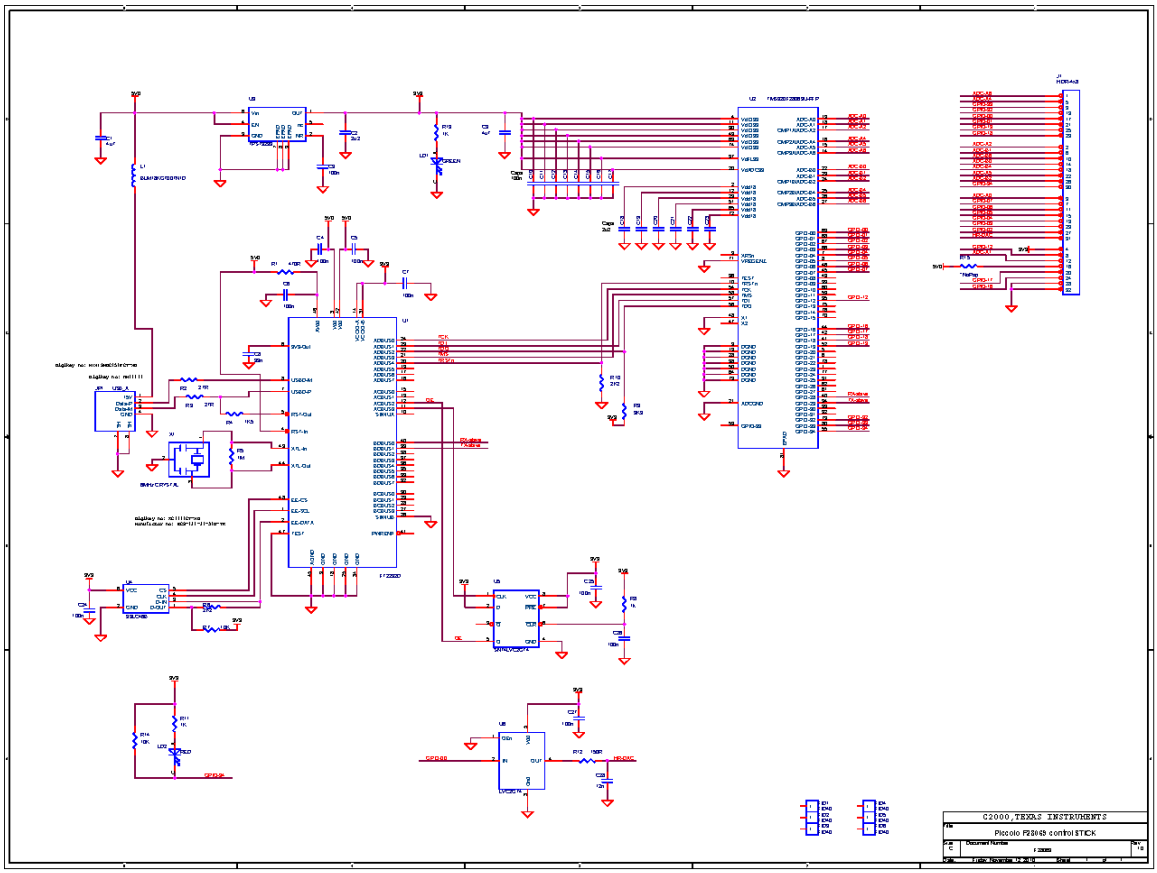


Figure F4.5.1B – controlSTICK Schematic
COURTESY OF TEXAS INSTRUMENTS

## 4.5.2 – Power Management

### 4.5.2.1 – Description

Power will be provided by a 12.8 V battery pack. The Piccolo MCU requires a 3.3 V DC input. The BeagleBone Black requires a 5 V DC input. The Servo Motors will also operate at 5 V.

Two switching regulators with 5 V outputs will be connected to the battery pack. Switching regulators are used for this stage because the BeagleBone Black can draw nearly 500 mA, and standard sized RC servos can draw about 1 A at stall (This is an estimation based on generalization, as stall current is almost never given for RC servo motors, as stalling them can cause rapid damage). This is an estimated maximum of 2.5 A, giving an estimated maximum power consumption of 12.5 W for these components. If a linear regulator was used, this would be a power loss of (12.8V - 5V)*(2.5A) = 19.5 W, which is more than one and a half times the maximum power consumed by the BeagleBone Black and the two motors. This gives an efficiency of (12.5 W) / (19.5 W + 12.5 W) = 39.1% at maximum current, which would only further decrease if the Piccolo board was included in the calculation. A switching regulator is more efficient when operated in the correct current

range, so much higher efficiencies are possible. In particular, the regulators chosen for this application are anticipated to run at about 90% efficiency, based on efficiency graphs from the datasheets. One of these switching regulator's output will provide power to the motors, and the other switching regulator's output will provide power to the BeagleBone Black and the linear regulator which provides power to the Piccolo.

The reason for having two switching regulators is twofold. The first reason is logistical: a regulator that fit the specifications required for being the sole regulator was not easy to locate. The second reason is that the motors will create a large deal of noise on an already noisy switching regulator. While capacitors can mitigate the noise somewhat, it is better to have separate regulators.

A linear regulator with a 3.3 V output will be connected to the output of the switching regulator which powers the BeagleBone Black, and the Piccolo will be connected to the output of this regulator. The Piccolo will, at maximum clock speed and with every peripheral running, draw around 300 mA. However, with only the peripherals necessary for this application running, the current will be much less. This is a maximum of (5V – 3.3V)*(300 mA) = 510 mW (although it will be significantly lower in the circuit, especially if it is determined that a clock rate of 90 MHz is unnecessary, which is likely to be the case) dissipated through the linear regulator, which is acceptable with regards to the overall power consumption of the device, and much less significant than the 19,500 mW of power loss by using a linear regulator elsewhere in the circuit. The connections between the battery, the regulators, and the powered components is shown in figure F4.5.2.

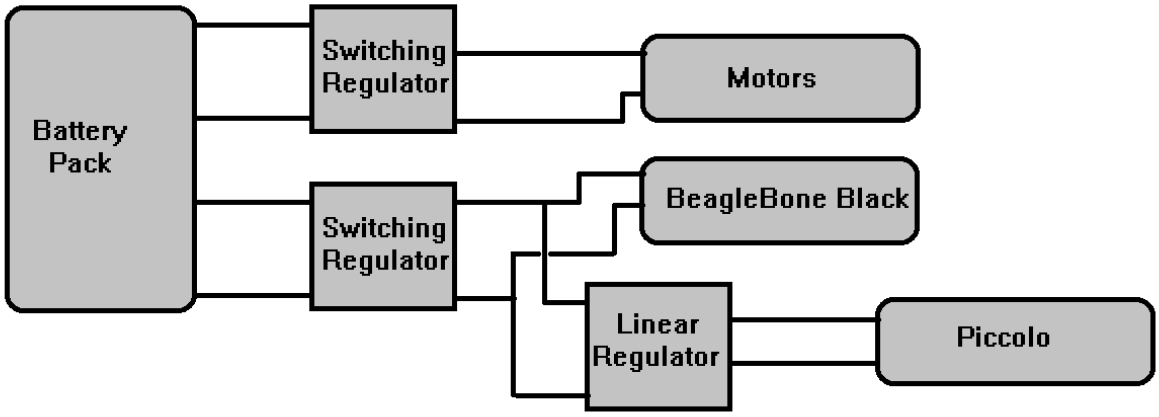



Figure F4.5.2 – Power Regulator Configuration

The battery pack will be Lithium Iron Phosphate ($LiFePO_4$) chemistry. Cells made using this chemistry have a working output voltage of 3.2 V, hence the 12.8 V supply voltage. $LiFePO_4$ batteries have a high energy density, as well as a fairly constant discharge curve, except for immediately after being charged and immediately before being depleted. The voltage decreases linearly with time very slowly until the battery's capacity has been nearly depleted, at which point the voltage rapidly decreases. Graphically, the rapid decrease in voltage of a nearly-depleted battery appears almost vertical. Energy density is important

because this device could be required to operate for several hours in a day. While a single battery may not be sufficient, high energy density allows battery packs which are sufficiently compact to the point where it is not infeasible to carry a spare or two. This battery chemistry also has a high maximum discharging rate and charging rate (although generally not when compared to Lithium Polymer batteries, which are also less safe), which could allow a user to make due with two batteries, charging one while the other is in use.

#### 4.5.2.2 – Regulator Specifications

Linear Voltage Regulator:

- Manufacturer: Texas Instruments
- Part Number: TPS73233
- Input Voltage (Min/Max): 1.7 V / 5.5 V
- Output Voltage: 3.3 V
- Quiescent Current: 0.4 mA
- Dropout Voltage: 40 mV
- Output Capacitor Type: None or Ceramic

Switching Voltage Regulator (provides power to motors)

- Manufacturer: Analog Devices, Inc.
- Part Number: ADP2303ARDZ-5.0-R7
- Input Voltage (Min/Max): 3 V / 20 V
- Output Voltage: 5 V
- Maximum Output Current: 3 A
- Efficiency: >90% for Output Current >500 mA

Switching Voltage Regulator (provides power to the BeagleBone Black and the Piccolo)

- Manufacturer: Analog Devices, Inc.
- Part Number: ADP2302ARDZ-5.0-R7
- Input Voltage (Min/Max): 3 V / 20 V
- Output Voltage: 5 V
- Maximum Output Current: 2 A
- Efficiency: >90% for Output Current >500 mA

## 4.5.3 – Component Layout and Connections

The main input of the printed circuit board will be the SPI bus connection to the BeagleBone Black board. The point of connection will be a female header, similar to the header rows on the BeagleBone Black. Male-male jumper wires will be used to connect the two female headers. Initially, the PCB for the Piccolo was intended to be constructed with male header rows on the bottom side so that it would plug into the BeagleBone Black. This is how ‘capes’ – the purchasable BeagleBone Black daughter boards – interface with the board. However, this has the downside that the other output pins on the headers which

are used by the BeagleBone Black, such as the UART pins for receiving the camera data, must be broken out in the Piccolo board. On a finished product, this would be acceptable. However, the purpose of separating the functions into two separate boards in this prototype is to allow both to be worked on separately, and having functions of one board built onto the other negates this benefit. Jumper wires allow for the boards to be rapidly combined and separated, and allow for them to be worked on separately.
In addition to the SPI input, other input peripherals of the Piccolo may be broken out in the final revision of this board to allow possible expansions if time permits.

The main output of the printed circuit board will be the two PWM channels which go to the two RC servos. These will be available on the board in the form of three consecutive male headers, which will allow standard RC servo connections to be used. The servo connector being referred to is shown in figure F4.5.3A, which is an image of a standard sized RC Servo motor. The supply voltage wire, ground wire, and PWM control input wire can be seen to terminate at a female header. Along with the PWM control signal header, there will be 5V and ground power supply headers from a switching regulator which will be mounted on one corner of the board. These headers will be located physically close on the board to the motor's switching regulator (the ADP2303), which will prevent excessive resistive losses in the traces.

There will be a ground plane on the reverse side of the PCB board, to which all ground connections required by the circuits on the printed circuit board will be made. This helps to keep trace lengths short, which minimizes the size of conductive loops in the circuit (and therefore decreases the effects of electromagnetic interference), as well as to provide a more consistent ground voltage throughout the board. This is done by terminating a trace at a hole drilled through the board. The inside of this hole is plated with copper, making contact with both the trace on the component side and the PCB.

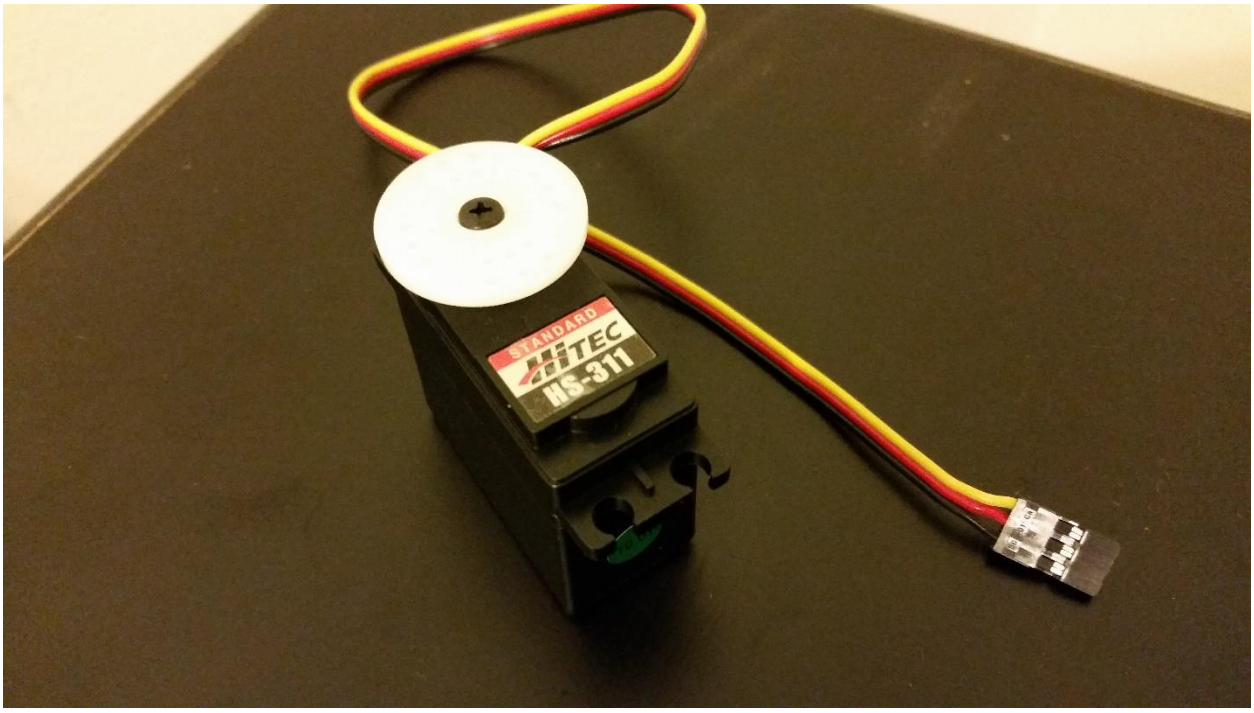


Figure F4.5.3A – A Standard Size RC Servo with Typical RC Servo Connector

The other switching regulator, the ADP2302, which will power the linear voltage regulator and the BeagleBone Black, will also be on this board, as will the linear voltage regulator. This second switching regulator will be located in the nearest corner to the other, so that the 12.8 V from the battery will not have to be routed too far along the board, while maintaining some distance to try to reduce the effect of noise from each regulator contributing to the noise in the other. The linear voltage regulator will be closer to the other side of the board, where the Piccolo IC will be mounted. It is beneficial to move these components away from EMI sources known as switching regulators. Physically close to the linear regulator, a 47 uF tantalum capacitor will be placed which will bridge the 5 V input of the linear regulator and ground. Tantalum is chosen because of low ESR and leakage. This capacitor serves to reduce noise and stabilize the ripple from the output of the switching regulator.

The linear voltage regulator's 3.3 V output will be connected to one lead of a 4.7 uF capacitor which is mounted close to the Piccolo. The other lead of this capacitor will be connected to ground. The lead to which the linear regulator's output is connected will have individual traces routed to each of the six $V_{DDIO}$ pins (digital I/O and peripheral power pins), the $V_{DD3VFL}$ pin (flash core power pin), and the $V_{DDA}$ pin (analog power pin). Each pin will have its own trace, because a power bus or daisy-chain configuration would cause uneven voltages due to differing currents along the trace as current branches out into each pin. Very near to each of these physical pins will be a 100 nF capacitor. The combination of the 4.7 uF capacitor and 100 nF capacitor provides a more stable and less noisy supply voltage.

The internal 1.8 V voltage reference will be used for this application, so the VREGENZ pin should be grounded. In this case, the Piccolo's datasheet mandates that the $V_{DD}$ pins (1.8 V CPU and logic pins) should have a 1.2 uF or larger capacitor between the $V_{DD}$ pin and ground. The TRST pin (JTAG test reset pin) must be low for normal operation, and requires an external pull-down resistor to allow debugging. Exact resistance depends on the debugger, but the datasheet suggests that 2.2 kOhm is usually sufficient. The XRS pin (external reset / watchdog reset pin) should be connected to $V_{DDIO}$ by a 2.2 kOhm to 10 kOhm resistor, and a 100 nF or smaller capacitor may be connected between the XRS pin and $V_{SS}$ (digital ground) for noise suppression. The $V_{SSA}/V_{REFLO}$ pin (analog ground) should be connected to ground, as should the $V_{SS}$ pins. The thermal pad on the Piccolo's case should be connected to ground.

An external CMEMS oscillator will be used, with a frequency of 20 MHz. The Piccolo's PLL clock multiplier is capable of multiplying by a variety of fractions, including 90 MHz, so this frequency allows for maximum clock rate, if that is deemed necessary. This oscillator requires very minimal supporting circuitry, and does not need that circuitry to be precise. Required for the chosen model is a 100 nF capacitor between ground and the $V_{DD}$ pin for noise suppression. Provided sleep mode was never desired, the active-high output enable (OE) pin could be tied to $V_{DD}$, and the capacitor would be all that is required.

That said, there are two pins on the Piccolo that allow an external clock signal to be input. The default of these is also used for JTAG debugging (it is the JTAG test clock pin), and the alternative pin is also used for SPI communication. If the default pin is used, the datasheet says that there should be hooks integrated which can disable the external clock. If the external clock cannot be disabled, then there can be issues during debugging. As such, a pull-up resistor is connected between $V_{DD}$ and the OE pin, along with a way to pull the pin to ground. A small switch will likely be used to accomplish this.

Figure F4.5.3 shows a screenshot of the schematic capture stage of the Piccolo printed circuit boards design.

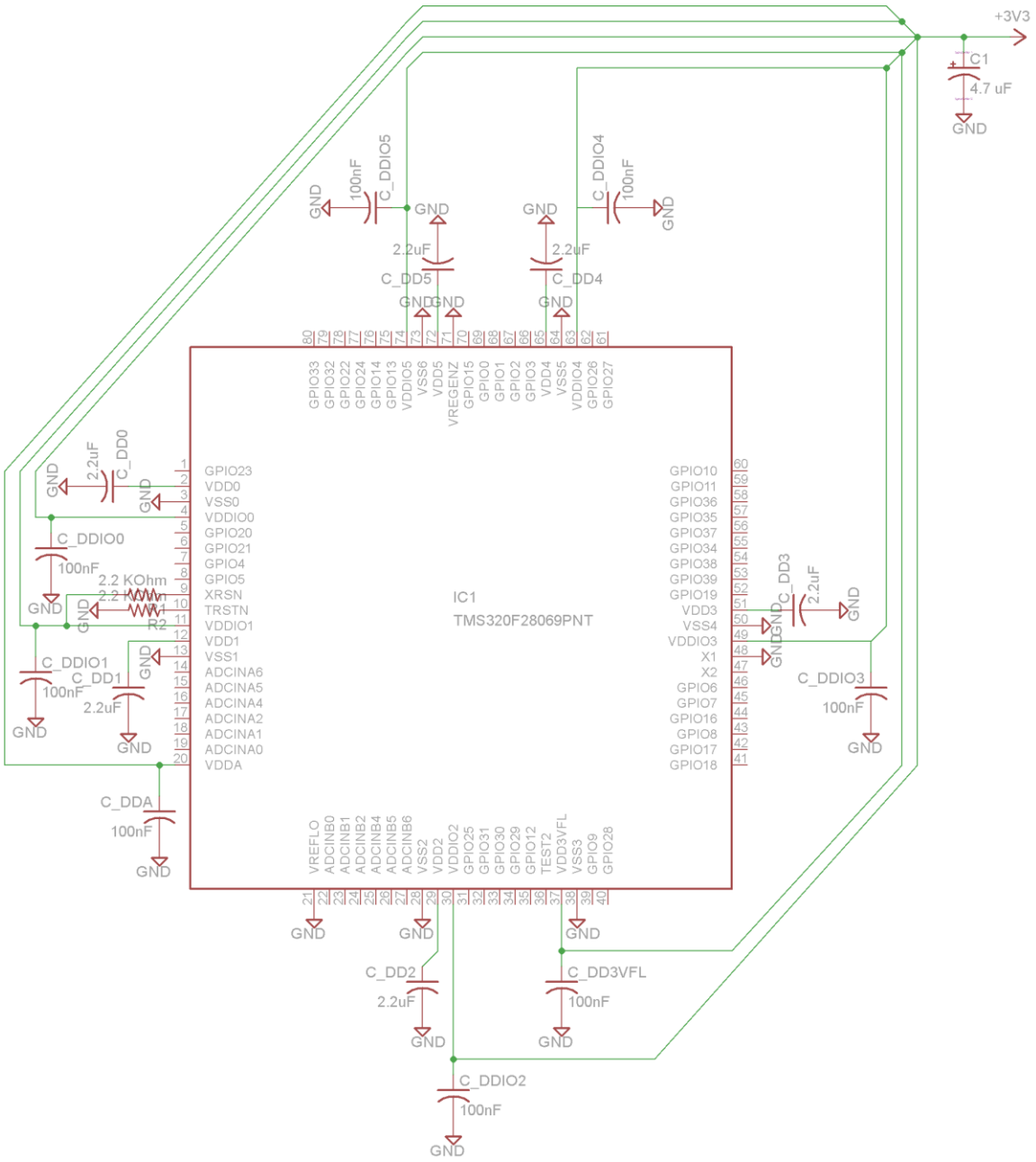



Figure F4.5.3 – Bare-Minimum Passive Support Circuitry

The external oscillator is not shown in this image, as it is not a passive component. However, as it will be used, the X1 pin is grounded in the image. This is required when the pins are not connected to an on-board crystal. XCLKIN is not labeled in this schematic, but the all of the pins are shown, including the pins which can operate as an XCLKIN input pin. The default XCLCK pin is pin 54, which is labeled as GPIO 38. The alternative pin is pin 52, which is labeled as GPIO 19.

If the CMEMS oscillator was included in the schematic, there would another wire leaving capacitor C1 (located in the top-right corner of the image) which would connect to the

supply voltage pin of the oscillator (which would also have a 100 nF capacitor connected to the pin and to ground). The 3.3 V source in the top-right corner which is connected to the capacitor C1 represents the output of the 3.3 V linear voltage regulator.

The model numbers for all components discussed in this section can be found in section 6.1 – Component Acquisition.

## 4.5.4 – Determining Component Values

Not mentioned in the previous section are the voltage regulator circuits and passive component values. For the LDO linear voltage regulator, as the circuitry on either side of the linear regulator circuit is nearly identical as those in the controlSTICK, the same component values can be used. For the switching regulators, the datasheet must be consulted. As the typical voltage of the battery will be very close to the 12 V input which is extensively discussed in the datasheet, the information in Table 11 of the ADP2302/ADP2303 datasheet may be used to quickly determine component values, which will soon be listed here. It is noteworthy that the components each have certain type recommendations which are highly encouraged to be followed:

Capacitors: Capacitors with either X5R or X7R are recommended because they have low effective series resistance (ESR), and because they have low temperature coefficients.

Inductor: Shielded ferrite core inductors are recommended due the high frequency of regulator switching.

Diode: Schottky diodes should be used, since they have a fast switching speed and a low forward voltage drop (approximately 0.4 – 0.45 V), which is a factor appearing in almost all of the component value equations.

ADP2302 with $V_{IN}$ = 12 V, $V_{OUT}$ = 5 V, and $I_{LOAD(max)}$ = 2 A

- L 6.8 uH
- $C_{OUT}$: 2x 22 uF capacitors in parallel
- $R_{TOP}$: 52.3 kOhm, 1% tolerance
- $R_{BOT}$: 10 kOhm, 1% tolerance

ADP2303 with $V_{IN}$ = 12 V, $V_{OUT}$ = 5 V, and $I_{LOAD(max)}$ = 3 A

- L 4.7 uH
- $C_{OUT}$: 47 uF
- $R_{TOP}$: 52.3 kOhm, 1% tolerance
- $R_{BOT}$: 10 kOhm, 1% tolerance

To select the actual parts, the other tables in the datasheet were compared with the component values listed above.

The diodes were both chosen to be SSB43L Schottky diodes, as listed in Table 7 of the datasheet. The SSA33L would have been sufficient for the ADP2302, but the prices were very similar, so two of the same diode were selected in an attempt to save time when laying out the printed circuit board by having to make one less component package.

Using Table 8 of the datasheet, and with no luck finding significantly better alternatives, the VLF10040T-6R8N4R5 was chosen for the ADP2302, and the VLF10040T-4R7N5R4 was chosen for the ADP2303.

Using Table 9 of the datasheet, two GRM31CR60J226KE19L capacitors were selected for the ADP2302, and a GRM32ER60J476ME20L capacitor was selected for the ADP2303. However, it was then determined that the GRM32ER60J476ME20L capacitor has become obsolete since the time of the datasheet's publication date and was in stock with none of the major distributors. A similar capacitor, the GRM31CR60J476ME19L, was located on Digi-Key and selected as the output capacitor of the ADP2303. Both selected capacitors are X5R capacitors, following the previous recommendation from the datasheet

$R_{TOP}$ was selected to be a 52.3 kOhm ERJ-8ENF5232V thick film resistor, and $R_{BOT}$ was chosen to be a 10 kOhm MCA12060C1002FP500 thin film resistor. Both have 1% tolerance, and a much higher power rating than is necessary to attempt to keep components at a reasonable size for hand-soldering (they are 1206 packages, which is roughly 3mm by 1mm).

# 4.6 – Implementation

The implementation of our hardware did not go as planned; this was due to a change in personnel. Due to this change, we were unable to acquire our designed hardware components. This forced us to purchase equivalent components to meet our hardware needs. Rather than creating a PCB with the PWM controls for the servos, we handled this control with the BeagleBone Black. Power control and regulation was also a portion of the design that left us with the personnel change. To compensate with this we made our system wall powered and acquired a Step Up/Down Voltage Regulator, to make sure our servos were held at a constant voltage. As you can see in Figure 4.6 A this is a picture of the Voltage regulator. Table 4.6 A has the specifications of Figure 4.6A.

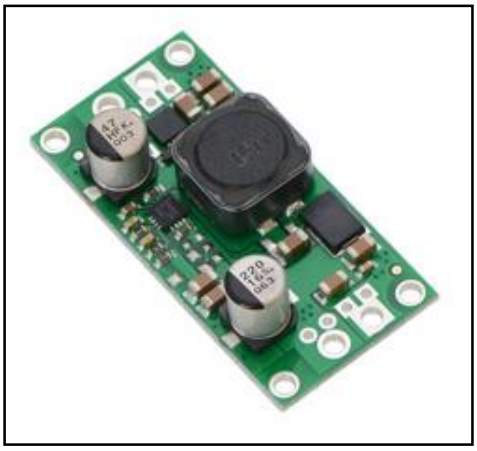


*Figure Figure 4.6 A Step Up/Down Voltage Regulator*

5V STEP UP/DOWN VOLTAGE REGULATOR SPECIFICATIONS

| | |
|---|---|
| Operating Voltage | 4.8V-6.0V |
| Operating Temperature | -20 to +60 °C |
| Operating Speed(4.8V) | 0.19sec/60° at no load |
| Stall Torque(4.8V) | 42 oz.in |
| Current Drain(4.8V) | 7.4mA/idle 160mA no load operating |
| Connector Wire Length | 11.81" |
| Weight | 1.52 oz |

Table 4.6 A

The Base, Pan and tilt servos and tilt head, were all purchased by the group and put together. This step proved to be a fast one, this allowed for the integration of the Software to happen almost immediately after acquiring the Hardware.

# 5.0 – Software

## 5.1 – Facial Tracking

### 5.1.1 – Reasons for facial tracking

The biggest reason to integrate facial tracking for our project is because we want to use a directional microphone to record and decode speech from a single individual. Facial tracking will ensure that we direct the microphone to the source of the desired sound, with the least amount of effort. With facial tracking we are assured that the mic will be pointed in the most accurate position to insure the highest quality of sound to microphone. If we use blanket object tracking the mic might not be guaranteed to be pointed to the mouth of the desired individual, (ie it could follow a moving hand or leg) and then vital sound quality would be lost and make the speech to text editor's job much harder. Another reason for facial tracking is that there are many classifiers on the face that are unique from other objects. Eyes and mouths move quite differently from other objects in a classroom environment, and as such are good things to track to follow a person around, rather than a chair.

### 5.1.2 – Methods of facial tracking

There are several methods of tracking, Facial recognition algorithms are used in these algorithms to identify and to enforce new faces. There is not necessarily a direct translation from facial recognition and facial tracking, Rather, the elements are blended together so that the good points of recognition are used to track. To see the different options for facial

recognition see section 5.2.2. Facial tracking is still a relatively new and developing field, there are many approaches still in development.

A widely used way to track faces is the Kanade-Lucas-Tomasi (KLT) approach. This approach is based off of an optical flow version of tracking. This type of tracing uses three main assumptions, The first assumption is that the brightness from one frame to another frame isn't going to significantly change, especially if the camera and algorithm has a significant enough frame rate. The second assumption is temporal consistency, which means that an object will not change the general shape from frame to frame. The third assumption that objects will stay relatively in the same place over each frame, and that if an object is moving that it will keep a similar velocity over frames. This algorithm is relatively fast and doesn't take a significant amount of memory to implement.

The next family of Algorithms uses a more iterative method of facial tracking. This method determines estimates of density gradients, to find probable targets. This algorithm rapidly finds the maximum of each density, then determines the maxima of those densities. The data is then discretized to be used in a feature space, where it is then iterated through to find the most probable position of the current target. This algorithm turn out to be a fast method to tracking objects, and through a few simple modifications can accurately do real time facial tracking. This algorithm falls short when faced with multiple modes of probability density functions.

Where the last algorithm fell short dealing with multiple modes of probability density functions, the sequential Monte-Carlo method excels. This algorithm is for solving state estimation problems, where many densities are unknown and occlusion is high. This is great for recent tracking approaches which use a tracking by detection strategies. This method is great for highly populated areas where there is a bunch of overlap and many different faces and objects. The type of algorithm helps to speed up other facial tracking algorithms, by taking what are normally exponential run time to a linear run time.

In all of these algorithms there are a few key things that they will look for to enhance their object detection abilities. For facial tracking, there are a few things that are important. The first is local object that describe the environment, such as a white board and desk for a classroom. Then the algorithms will look at gradients, histogram these gradients to find different anomalies that might be significant. Finally color may sometime play a role in face detection if there are distinctive features from the background.

## 5.1.3 – Possible Facial Tracking Classes

Figure 5.1.3A shows an activity diagram of one possible Facial tracking class. The main feature in this class is the looping nature where the image is continually being checked for movement.

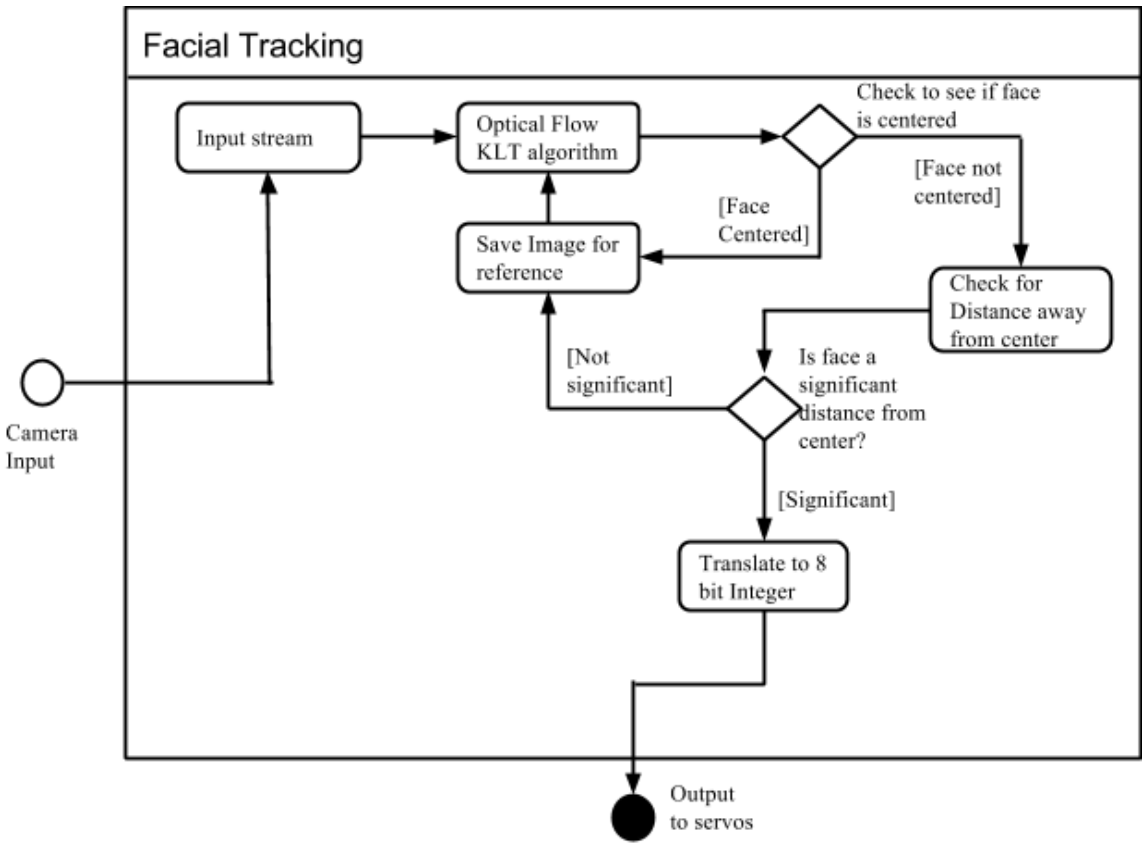



Figure 5.1.3A

In Figure 5.1.3B we have a very similar activity diagram compared to Figure 5.1.3A. The main difference is that in Figure 5.1.3B there is a loop to halt the algorithm while the servos are moving. This change is important because we feel that if the is camera moving it will mess up the optical flow algorithm by giving false vectors. Stopping the algorithm from running while the movement takes place eliminates this factor.

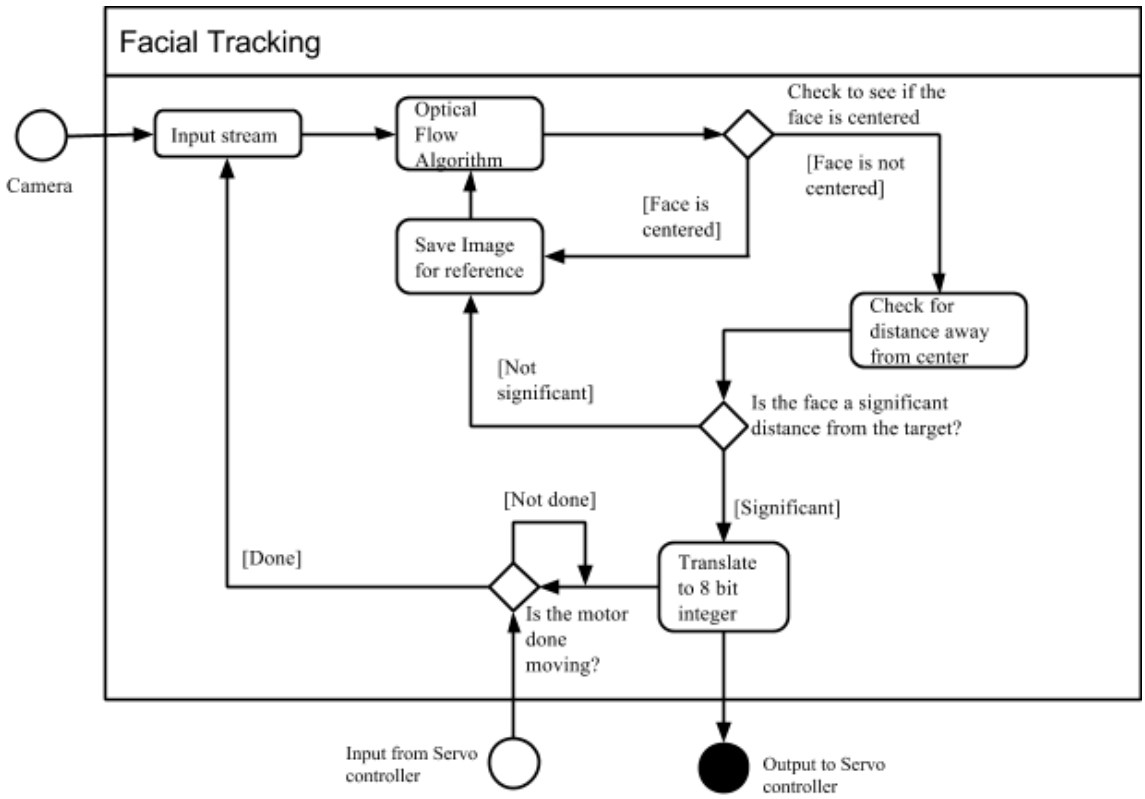



Figure 5.1.3B

In Figure 5.1.3C, we have an activity diagram that takes into account the first figure of 5.2, the facial location method, describing how we may incorporate facial recognition to make our program more efficient, by breaking up the work it in our facial tracking program. Since the facial location will only send information if the face is not in the center of the frame, the activity diagram can become more streamlined, which will hopefully result in a more streamlined program.

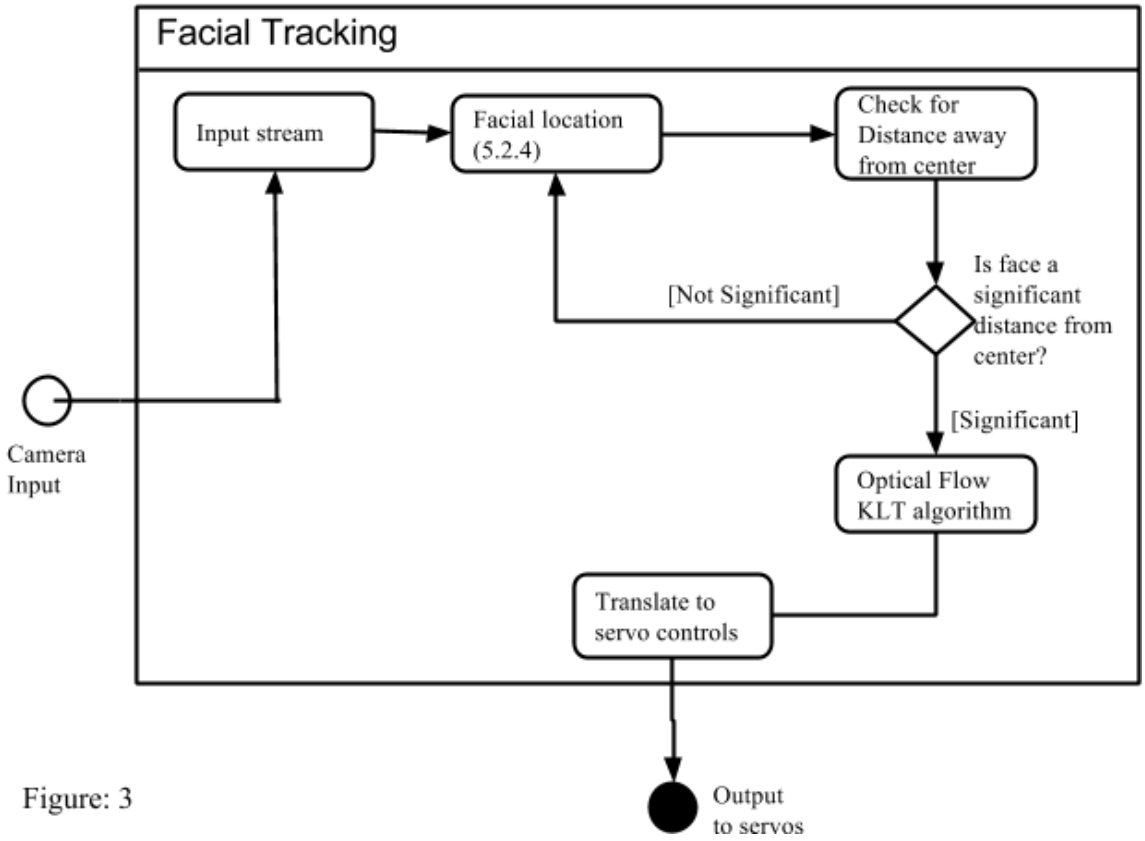



Figure: 3

Figure 5.1.3C

In Figure 5.1.3D, we have an activity diagram that takes into account the second figure of 5.2.4, the facial centering method, describing how we may incorporate facial recognition to make our program more efficient, by breaking up the work it in our facial tracking program. The facial centering method, we put into Figure 5.1.3D, will take the movement information from the facial tracking algorithm and translate that into servo commands, to make the process more efficient.

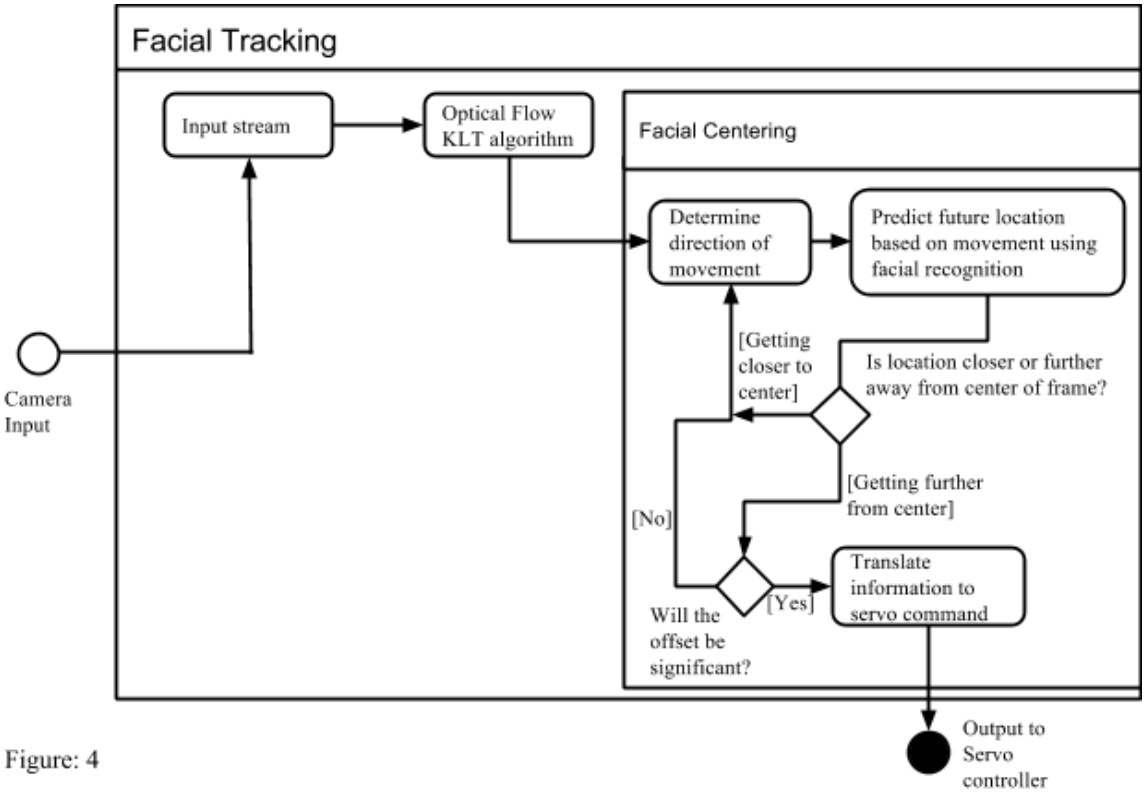



Figure 5.1.3D

## 5.1.4 – Other Applications

Current technology uses facial tracing in several applications. As a commercial product facial tracking is used for augmented realities. The purpose of most smartphone applications that use facial recognition is to place pictures or masks over the user's face. The facial tracing is used to keep the new image in front of the user's face at all times. These apps have been growing in popularity and have begun to offer more advanced features as developers become more comfortable using more advanced facial tracking algorithms. Some of the higher end products use facial tracking to try and predict the user's mood and make suggestions during video chat sessions with startling accuracy. Sadly these advanced features are still in development, but are not very far off from the marketplace.

The next field that has pushed facial tracking to new heights is animated movies. With audiences expecting more realistic and human movements from even the strangest of animated creatures, facial tracing and then modeling to an animated character has begun to rise in popularity. Most people are somewhat familiar with motion capture for large scale movements. Generally characterized by people in green unitards and wearing random white balls all over. Facial capture brings this technology to a more intricate scale with less use of obvious markers to model movement. So to improvise many filmmakers and animators have been utilizing and pushing for the optimization of tracing algorithms that can map full facial movements with all of its nuances. This has helped animators to sync mouth movements with the sound coming from the actor.

# 5.2 – On-Board Facial Detection

## 5.2.1 – Different libraries for vision software

While there are tons of open source software kits online, only some of these kits stood out as being the most comprehensive and easy to use with a large user community to provide helpful feedback. The criteria we used to select these libraries was that they had to be robust in scope and relatively easy to implement.

The first and main open source computer vision software is OpenCV. This is the leader in open source vision applications. From hobbyist to businesses utilize this software package to enhance and build their vision applications. OpenCV has interfaces in most programing languages and can be exported to MATLAB. It is supported by every major operating system. OpenCV can also be used by embedded linux devices, and can now be used by GPUs to exploit their ability to quickly calculate floating point equations. The real advantage of OpenCV is that it emphasises real-time vision applications. This is good for our project because we will need the program to run in real time and this means that our program is guaranteed to get algorithms that are optimized for performance.

Another software utility that was viable for our applications was SimpleCV. SimpleCV is a Python derivative of OpenCV that optimizes OpenCV libraries for Python in a more simple shell. Since SimpleCV is in python it will be good to use with an embedded linux device, because linux has a native Python kernel. The down side is that it might be difficult to put a Python compiler onto the board if an operating system is not used. The benefit of the software is the large user base with lots of tutorials, examples and several books written on how to use the product from simple to advanced applications.

The final utility is the only proprietary software that we considered, which is MATLAB. MATLAB is a numerical computing environment, that is part statistical software and part code generator. With interfaces in most programming languages MATLAB program can go into almost any computing environment. There is even add ons to be able to send out code that can be read by specific chipsets, including embedded chips. MATLAB is a modular technology that has its functionality determined by which module is installed at any given moment in time. For our project there is a computer vision module and an image processing module. With these there would be a simple framework to program and the ability to export the program in the exact format to match our particular embedded device.

## 5.2.2 – Methods of detection

There are several methods of facial recognition. The simplest, but least reliable method is color detection. This method utilizes the different shaded regions of the face, i.e. the bridge of the eyebrow, eyelashes, and shading that comes from hair and light. This process is a problem because it is very limited in scope. Differences in lighting conditions, and different

skin tones can drastically alter the accuracy of this method and is not a reliable unit of measure.

Motions detection is another way to do facial detection. This method uses characteristics such as the fact that most blinks have the eyes closing synchronistically. It also uses things like mouth movement to determine if the object is a face or not. Just like the color detection algorithm this algorithm is limited in scope and is an unreliable method of facial detection. Researchers did try to remedy this method by also including color detection. While this combined method is better than either individual method, it is still not a robust way to detect faces.

The next method is a reasonably reliable way to do facial recognition, neural networks. In this method the face is broken up into many different neurons, usually one pixel per neuron. The pixels are then analyzed and approximated. They are then run through a network where they are compared to other pixels to find features and unique properties. The network then approximates what these features mean and then gives an output. To get this method to work training sets are used to let the network know it's doing things correctly. This method is very reliable, because of the number of features that can be extracted and analyzed. Neural networks suffers from the fact that if the problem is significant; many training cases for the network will be needed before facial detection can be used. This has the tendency to make the networks a time consuming problem. The benefit of the algorithm is that with even a few training sets it is very accurate. This accuracy may offset any problem that the method has with memory or time, because once an accurate network is established it can be utilized many times.

Finally weak classification is the final and best method to use for facial detection. Weak classification algorithm takes a large training database where some of the images are faces and the others are not. It then takes randomly generated classifiers and then checks them off of the given images. After passing through the training a few times there will be some clear classifiers that detect most of the faces, these are chosen to be weak classifiers, the algorithm is best when the user keeps running the algorithm until most faces are detectable. As long as the database is large enough and the user selects a sufficient amount of classifiers, this method of facial detection is the most reliable and robust one available. Most if not all commercial and private products with facial detection features use this type of recognition.

## 5.2.3 – Algorithm Size Considerations

The size of our final program has to be fine tuned to the embedded environment that we choose for it. To make sure that the program can operate on our selected hardware we have to be very careful with the algorithms that place into the program. We will have to find algorithms that are memory efficient or are small enough that their inefficiency doesn't matter. Secondary memory space is also a critical consideration. If the program is too large and does not fit into our limited secondary memory we will not be able to complete our concept. The main concern from a programming aspect is that the back end algorithms might be too bloated to be handled on an embedded environment. Though we plan on

having enough processor strength to run the algorithms, the concern is running out of primary memory while critical processes are running. For this purpose we must carefully consider the type of facial recognition we will use.

Color recognition and movement are not only sub optimal forms of recognizing faces, but they are also costly algorithms. Since they were early forms of detection and they are not used commonly the algorithms were never streamlined to be memory efficient. this means that we cannot choose these algorithms without first having supporting software that will make efficient use of them. While the more common algorithms, such as the neural network and the weak-classifier algorithms, have been optimized for use in many applications. This approach can be problematic from a memory aspect, because if the network is too long there may be a memory leak, and we want to have as robust facial recognition algorithm as possible. The best algorithm that is currently used in most embedded applications is the weak classifier algorithm. Classifiers are chosen through a larger algorithm that is very large and takes a long time to run. The benefit is that this program can be precompiled and then the classifiers saved to be used by the other part of the facial recognition algorithm. We can be set to have a relatively small number of classifiers (e.g. 100-200) which means that even if the rest of the algorithm isn't the most efficient it can be run using a limited amount of memory and still be effective and fast enough to be run in real time.

## 5.2.4 – Potential Uses In the Program

One use for facial recognition is to help the tracking algorithm to find and confirm that the object being tracked is a face. This will not be a likely implementation that we will use because of the space constraints of our hardware. What is likely to happen is to use the less hardware taxing facial recognition algorithms to determine the exact location of the subjects face, chosen by the tracking algorithm. We will then use these positions to determine the speed and direction that the servos must travel to get the subjects face in the center of the frame. One such method is Figure 5.2.4A which show the activity diagram of our possible method that helps with finding the location of the persons face in the frame and then sends the tracking algorithm the location of the face so that corrections can be made.

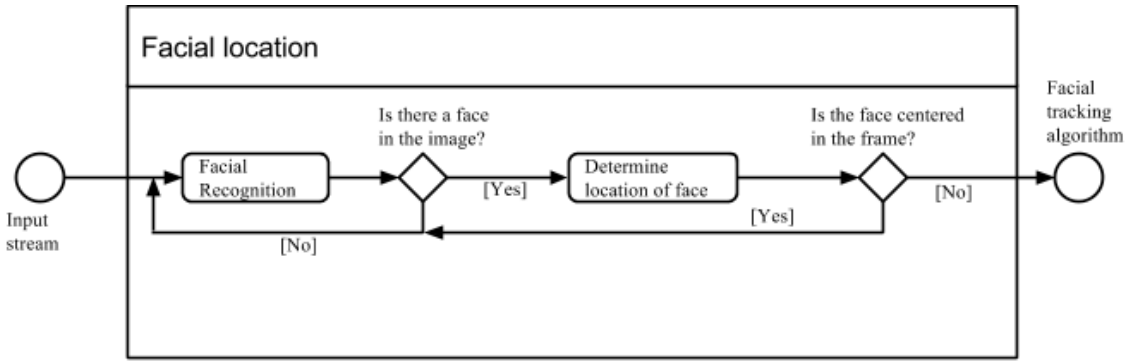



Figure: 1
Figure 5.2.4A

In figure 5.2.4B we do the opposite process, we have the facial tracking algorithm determine where the face is moving and then the facial recognition algorithm is used to determine the signal strength to send to the motors.

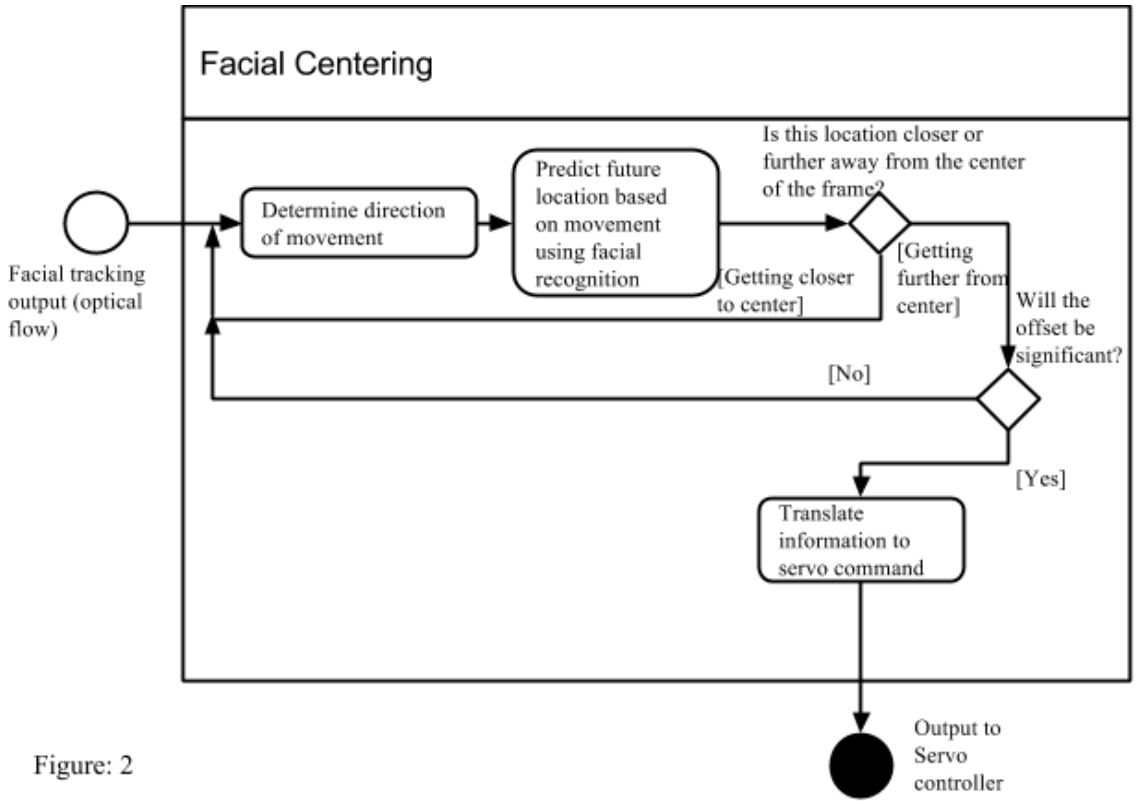



Figure 5.2.4B

## 5.2.5 – Operating systems

The embedded system that we currently plan to implement is a relatively complicated design. This lead us to consider whether an embedded operating system would be advantageous to our project, or if the control we'd be giving the operating system would be better handled by us. An embedded operating system operates essentially the same as a normal desktop operating system. The main differences is that they are designed to be compact, resource efficient, and reliable. Most of the time embedded operating systems strip out most of the functions of a traditional operating system, and try to maximize performance on the usually singular task that they perform.

### 5.2.5.1 – Advantages

The advantage of having an operating system is that all of the background work will be handled automatically and won't have to be dealt with by the operator/programmer. The operating system gives the programmer a reliable platform to make software on. For example, if in a program there are several inputs that have to be handled, with an operating

system this problem will be handled automatically. Embedded operating systems are efficient at resource usage and can do this much better than any individual could. This will enhance our project because most computer vision application programs are resource intensive, and if the operating system can handle them natively we can better use utilize our time on creating effective software. The other advantage to an embedded operating system is that there are many kinds. This is because each operating system is designed for a specific task. We can use this to our advantage, by focusing on a single task we can optimize performance of our program.

### 5.2.5.2 – Disadvantages

The variety of operating systems for embedded devices can be a negative thing. Most of these operating systems are targeted towards niche environments. While this is great for that specific application, but for applications that don't fit neatly it may be elusive if not impossible to find the correct operating system for the intended application. For example in our program we not only intend to use it for vision applications but to also control electric motors as well; an operating system in these circumstances may cost us vital resources that could be better utilized. Another disadvantage to having an operating system is the space it takes up on the board; of the operating system in use is a general functioning one, like the one we intend to use, it can take up a significant portion of the secondary memory on the embedded device (if the device has secondary memory) if not all of that memory. If the embedded device is simple enough it may be easier to make a program without the operating system and handle the memory by yourself. In rare instances embedded operating systems cannot utilize particular features in ways that are suboptimal to the particular problem at hand.

### 5.2.5.3 – Types to consider

While there are hundreds of different types of embedded operating systems for specific applications, we have only found a couple embedded operating systems that are general enough for our projects purposes. These operating systems are quite advances, with GUI interfaces, but they still are stripped versions of their desktop rivals.

The first such operating environment is embedded Linux. Linux had many different forms and functions because it uses a modular approach to its design. Embedded Linux is a fully functional operating system, just like a desktop operating system, but with most of its kernel stripped out. This is a benefit because this will make an embedded device essentially function just like a traditional computer. Also a major benefit is that a user can upload and use particular modules as needed to best fit their project. The other good part about embedded Linux is it’s open source nature; it is widely available and can be used on any processor, free.

A specific variant of embedded Linux that we found useful is called Debian. Debian operates under a GNU general public license, which means it’s free to own and use in anyway the user sees fit. While any person can edit this software, there is a group called the Debian project that creates and releases newer official versions of the software. One of

the benefits of Debian is that it is one of the earliest Linux distributions, which means that it has seen many iterations and has developed naturally based on user feedback and requirements. Another benefit to the Debian software is that it has a very small kernel. It has a small kernel for desktop use, which means that when conforming to an embedded platform not much functionality is lost.

Another embedded operating system under consideration is Windows CE. Like linux, Windows CE can use a GUI interface and work just like a traditional operating system. Also like its Linux cousin Windows CE is optimized for systems with very little memory and power. An advantage is the many development tools available to program on the operating system. There is also a very efficient power management and is quite portable to many embedded devices. The downside of this software is that it is proprietary and the user has to pay to use it, which is a bit of a burden with our projects tight budget.

## 5.2.6 – Other Applications

When one thinks of facial recognition, most of the time the thought is on surveillance. Cameras looking onto people to see who they are and where they are. This has been a large field of funding for security services for a long time. With accurate facial recognition algorithms police can find and track subjects by using security camera images. SWAT and tactical military teams that use drones to scout can now have the added benefit of the robot pointing out if it finds any faces in a room. This can be beneficial to make sure a room is clear, or maybe it has innocent people in it that would have otherwise been harmed. Facial recognition has helped to identify repeat offenders who have had their imaged placed into a database. Other systems help casino's to catch card counters and people who might be banned from their facilities.

Away from the security and surveillance sector, there is also a home for facial detection in commercial applications. The most common way we see facial detection is when using a camera. Whenever the camera draws a box over someone's face to show where they are and to enhance the image, facial detection is being used. This technology has been around for a while and is now so pervasive the people assume that this enhancement is happening even when no boxes are drawn on a screen. Another commercial use is in tagging faces in social media. Facial recognition is run on images placed on social media so that the user can identify who is in the picture more easily. Some sites practice this automatically, trying to find better ways to get customers to share where they have been and who they were with when the picture was taken. Another used for facial recognition is for unlocking technology. Companies have been trying to personalize log-in's to make them faster and more secure. One way they found to do this is to do facial recognition on the person trying to access their site or trying to open unlock their smartphone, to make the process fast and secure. Some technologies completely replace passwords while others merely substitute for it, so that customers that have a variety of users for their device can all get in. In the commercial space there is also a market for facial recognition in video games. Players can use their webcams to track their mood while playing a horror game so that the games knows when best to scare them. This allows for features on games like changing the scariness depending on your mood. So if you are already screaming the game might continue to

pursue you, or back off depending on the settings. Another feature that uses facial recognition is to identify shows, movies and objects you have seen before, to help jog your memory. You can download an app that with a picture of the television screen can tell you what show you're watching. This can also be used by advertisers to see what shows users of this app are watching and use that information to better sell their products.

# 5.3 – Speech

## 5.3.1 – Motivation for software implementation

Although there may be other avenues of approach the simple fact remains that speech to text is hardware intensive and cannot be implemented on board or hardware only. The best and so far most successful approach to implementing speech to text has been with software. This way a developer can cater to the user's needs by implementing an intuitive graphic user interface that has a low learning curve. As in our case, the user interface is very important since the user being targeted by this project will need a very basic ease of use that requires very little to no instructions what so ever. Developer can ensure the a very fast user experience without any lag and if there arise any issues they can be fix quickly an update or patch. The main goal is make sure the software looks inviting, fun to use and of course functional. The software will be installed on the laptop via USB or automatically installed when the hardware is plugged into the users' laptop.

Before diving straight into coding one was must first assess the impact of different parts of software implementation. I did not become a better coder until I learned to first design my program on a white board and break it into small parts. Once the program is broken into objectives and how best to achieve those goals, pseudo code may be implemented to ensure a well thought out execution. When writing the code a developer can refer back to the white board or the design and make sure the code implementation is as such. Straying from a good design can wreak havoc on software and make it seem as if the hardware is not up to par when in fact the code implementation is the problem most of the time. As it shows in figure 1, the simple fact is design plays a critical role in software and more importantly, its scalability. If software is poorly designed, there will be problems further down the line when it is scaled up for consumer uses. This is apparent in several instances of an operating systems release, hence why companies use beta testers to slowly scale up and see if there are any software glitches, or poor design.

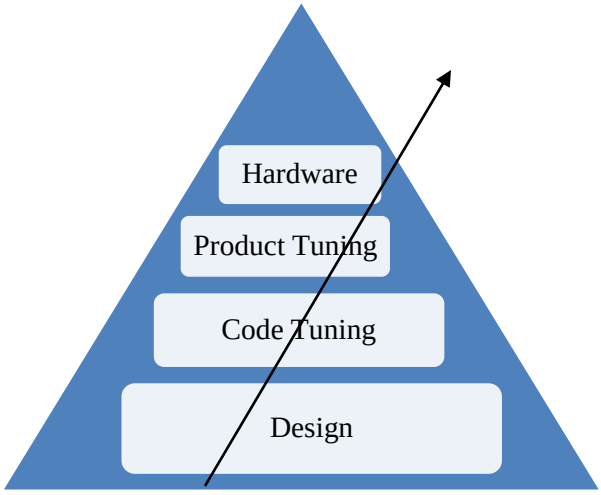


Figure 5.2.5 - Impacts on software scalability

As the scalability pyramid indicates, fast hardware, software, and tuning are only a small part of the scalability equation. At the base of the pyramid is design, which has the greatest influence on scalability. As you move up the pyramid through decreasingly important factors, the ability to affect scalability decreases. What the pyramid illustrates is that smart design can add more scalability to an application than hardware. When designing for scalability the primary goal is to ensure efficient resource management. Designing for scalability is not limited to any particular tier or component of an application. Developers must consider scalability at all levels, from the user interface to where the data is store. According to a Microsoft research paper on scalability, the five commandments of designing for scalability below can be useful when making design choices.

- Do Not Wait
- Do Not Fight For Resources
- Design For Commutability
- Design For Interchangeability
- Partition Resources and Activities

Do Not Wait: A process should never wait longer than necessary. Each time slice that a process is using a resource is a time slice that another process is not able to use that resource. You can place processes into two separate categories, synchronous and asynchronous. There are times when applications must perform actions synchronously. Some actions may need to wait for an action to return a result before continuing, or they may need to verify that an action was successful to ensure atomicity. That is, all of the actions associated with the operation must completely fail or succeed before performing another operation. However, when applications are limited in this manner, resources become a source of contention that negatively affects scalability.

One way to achieve scalability is by performing operations in an asynchronous manner. When operating asynchronously, long-running operations are queued for completion later by a separate process. For example, some e-commerce sites perform credit card validation

during the checkout process. This can become a bottleneck on a high-volume e-commerce site if there is difficulty with the validation service. For e-commerce sites that must physically ship a product to fulfill an order, this process is a good candidate for asynchronous operation. Because possession of the product does not shift from the seller to the buyer during the online transaction, the retailer can complete the credit card validation offline. The application can then send e-mail to the customer confirming the order after validation of the credit card transaction.

Do Not Fight for Resources: Contention for resources is the root cause of all scalability problems. It should come as no surprise that insufficient memory, processor cycles, bandwidth, or database connections to meet demand would result in an application that cannot scale. Regardless of design, all distributed applications possess a finite amount of resources. Besides expediting processes by not waiting on long-running operations, you can take other steps to avoid resource contention. A developer should order resource usage from plentiful to scarce. For example, when performing transactions that involve resources that are scarce and thereby subject to contention, use those resources as late as possible. By doing so, transactions that are aborted early will not have prevented or delayed a successful process from using these resources.
Acquire resources as late as possible and then release them as soon as possible. The shorter the amount of time that a process is using a resource, the sooner the resource will be available to another process. For example, return database connections to the pool as soon as possible. If possible, do not even use a contentious resource. Sometimes a process uses a transaction that does not require a resource when performing a function. For example, methods that require a transaction should be placed in a component separate from ones that do. As a result, you can avoid creating a transaction when it is not needed.

Design for Commutability: Designing for commutability is typically one of the most overlooked ways to reduce resource contention. Two or more operations are said to be commutative if they can be applied in any order and still obtain the same result. Typically, operations that you can perform in the absence of transaction are likely candidates. For example, Amazon is an e-commerce site that continuously updates the inventory of its products could experience contention for record locks as products come and go. To prevent this, each inventory increment and decrement could become a record in a separate inventory transaction table. Periodically, the database sums the rows of this table for each product and then updates the product records with the net change in inventory.

Design for Interchangeability: Whenever you can generalize a resource, you make it interchangeable. In contrast, each time you add detailed state to a resource, you make it less interchangeable. Although personalization can give a user a unique experience, it comes at the expense of creating a custom presentation that you cannot reuse for another user.

Partition Resources and Activities: Finally, developers should partition resources and activities. By minimizing relationships between resources and activities, you minimize the risk of creating bottlenecks resulting from one participant of the relationship taking longer than the other. Two resources that depend on one another will live and die together.

Partitioning of activities can help ease the load that you place on high cost resources. By separating methods that do not require transactions from those that do, developers do not needlessly impose the overhead required for a transaction on methods that do not require one. However, partitioning is not always a good choice. Partitioning can make your system more complex. Dividing resources that have dependencies can add costly overhead to an operation.

One of the biggest benefits of a software implementation is scalability. Scalability is the ability to increase the total system output when the load is increased, therefore allowing several users to use software without issues. A process should never wait longer than necessary. Each time slice that a process is using a resource is a time slice that another process is not able to use that resource. You can place processes into two separate categories, synchronous and asynchronous. There are times when applications must perform actions synchronously. Some actions may need to wait for an action to return a result before continuing, or they may need to verify that an action was successful. All of the actions associated with the operation must completely fail or succeed before performing another operation. However, when applications are limited in this manner, resources become a source of contention that negatively affects scalability. One way to achieve scalability is by performing operations in an asynchronous manner. When operating asynchronously, long-running operations are queued for completion later by a separate process. As seen in figure 1, different parts can have a huge impact on the scalability of the software. While most people think cutting edge expensive hardware can make up for poor software or poor design it is far from the truth. The best-designed software can minimize the severely impact the scalability and implementation of software. As the old saying goes, "the answer is simplicity itself."

## 5.3.2 – Methodologies of Speech

### 5.3.2.1 – Best properties of speech encoder

The main goal of speech coding is either to maximize the perceived quality at a particular bit-rate, or to minimize the bit-rate for a particular perceptual quality. The appropriate bit-rate at which speech should be transmitted or stored depends on the cost of transmission or storage, the cost of coding (compressing) the digital speech signal, and the speech quality requirements. In almost all speech coders, the reconstructed signal differs from the original one. The bit-rate is reduced by representing the speech signal with reduced precision and by removing inherent redundancy from the signal, resulting therefore in a lossy coding scheme as shown in Figure 5.3.2.1.

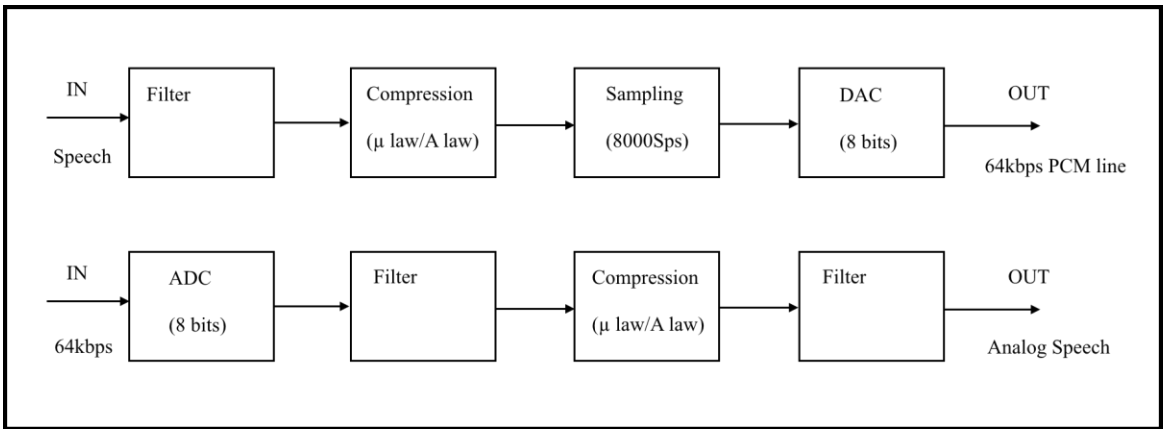


Figure 5.3.2.1 - Speech coding

Desirable properties of a speech coder include:

- Low Bit-Rate: The lower the bit-rate of the encoded bit-stream, the less bandwidth is required for transmission, leading to a more efficient system. It also helps minimize problems with the system as far is data transfer. I/O speeds are limited by the capacity of the development board; therefore it is essential that a low bit rate is maintained. This requirement is in constant conflict with other good properties of the system, such as speech quality. In practice, a trade-off is found to satisfy the necessity of a given application. The goal is to find a bit-rate that can ensure quality and task the system as much as a higher bit rate would.

- High Speech Quality: The decoded speech should have a quality acceptable for the target application. There are many dimensions in quality perception, including intelligibility, naturalness, pleasantness, and speaker recognition.

  - Quality perception: Deals with the all processes that are connected with assessing the auditory and quality of voice and speech

  - Intelligibility: A measure of how comprehensible speech is, or the degree to which speech can be understood. Intelligibility is affected by spoken clarity, explicitness, lucidity, comprehensibility, perspicuity, and precision.

  - Naturalness: This deals with the speech sounding authentic and human rather than computer generated.

  - Pleasantness: Some accents are more pleasant to the ear than others depending on the user, for most a female voice is easier to understand.

  - Speaker Recognition: Ability to of the user to understand the speaker and recognize the voice clearly.

- Robustness across Different Speakers / Languages: The underlying technique of the speech coder should be general enough to model different speakers (adult male, adult female, and children) and different languages adequately.

- Low Memory Size and Low Computational Complexity: In order for the speech coder to be practicable, costs associated with its implementation must be low; these include the amount of memory needed to support its operation, as well as computational demand.

- Low Coding Delay: In the process of speech, encoding and decoding, delay is inevitably introduced, which is the time shift between the input speech of the encoder with respect to the output speech of the decoder. An excessive delay creates problems with real-time two-way conversations, where the parties tend to talk over each other.

## 5.3.2.2 – Encoding Delay

Consider the delay as shown in Figure 5.3.2.2A. The delay obtained in this way is known as coding delay ,or one-way coding delay which is given by the elapsed time from the instant a speech sample arrives at the encoder input to the instant when the same speech sample appears at the decoder output. The definition does not consider exterior factors, such as communication distance or equipment, which are not controllable by the algorithm designer. Based on the definition, the coding delay can be decomposed into four major components (see Figure 5.3.2.2B):

1. Encoder Buffering Delay. Many speech encoders require the collection of a certain number of samples before processing. For instance, typical linear prediction (LP)-based coders need to gather one frame of samples ranging from 160 to 240 samples, or 20 to 30 ms, before proceeding with the actual encoding process.

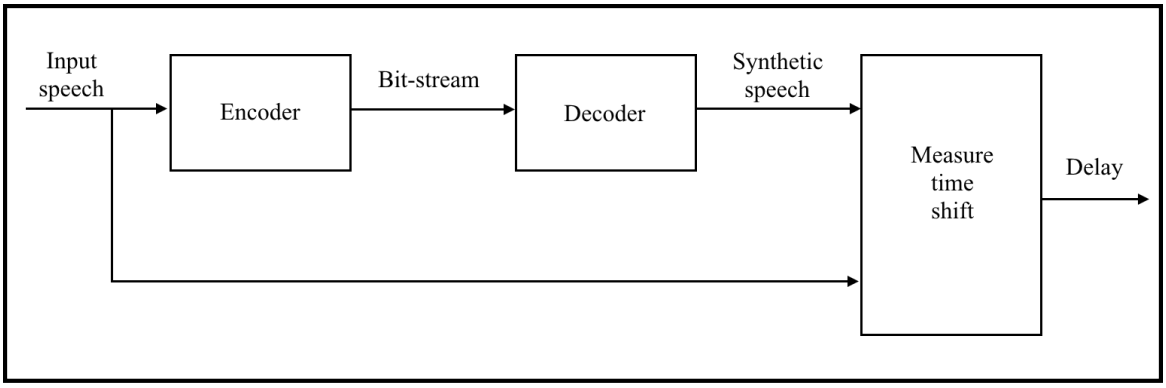


Figure 5.3.2.2A - Speech delay

2. Encoder Processing Delay. The encoder consumes a certain amount of time to process the buffered data and construct the bit-stream. This delay can be shortened by increasing the computational power of the underlying platform and by utilizing efficient algorithms. The processing delay must be shorter than the buffering delay; otherwise the encoder will not be able to handle data from the next frame.

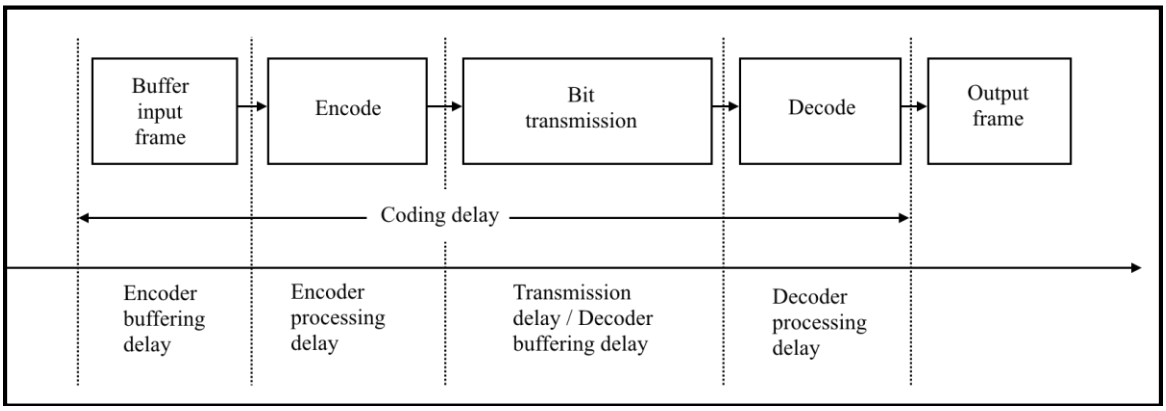


Figure 5.3.2.2B - Components of coding delay

3. Transmission Delay. Once the encoder finishes processing one frame of input samples, the resultant bits representing the compressed bit-stream are transmitted to the decoder. Many transmission modes are possible and the choice depends on the particular system requirements. There are two transmission modes: constant and burst. Figure 5.3.2.2D depicts the situations for these modes.

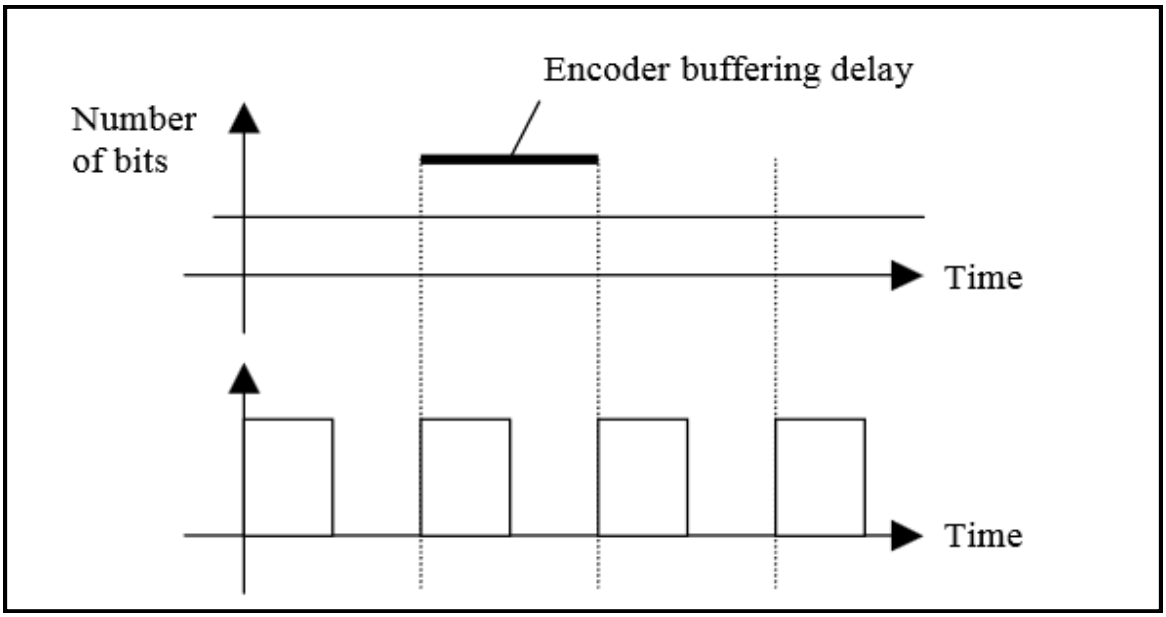


Figure 5.3.2.2D - Bit-stream, constant mode (top) and burst mode (bottom).

In constant mode the bits are transmitted synchronously at a fixed rate, which is given by the number of bits corresponding to one frame divided by the length of the frame. Under this mode, transmission delay is equal to encoder buffering delay: bits associated with the frame are fully transmitted at the instant when bits of the next frame are available. This mode of operation is dominant for most classical digital communication systems, such as wired telephone networks.

In burst mode all bits associated with a particular frame are completely sent within an interval that is shorter than the encoder buffering delay. In the extreme case, all bits are released right after they become available, leading to a negligibly small transmission delay. This mode is inherent to packetized network and the internet, where data are grouped and sent as packets. Transmission delay is also known as decoder buffering delay , since it is the amount of time that the decoder must wait in order to collect all bits related to a particular frame so as to start the decoding process.

4. Decoder Processing Delay. This is the time required to decode in order to produce one frame of synthetic speech. As for the case of the encoder processing delay, its upper limit is given by the encoder buffering delay, since a whole frame of synthetic speech data must be completed within this time frame in order to be ready for the next frame.

As stated earlier, one of the good attributes of a speech coder is measured by its coding delay, given by the sum of the four described components. As an algorithm designer, the task is to reduce the four delay components to a minimum. In general, the encoder buffering delay has the greatest impact: it determines the upper limit for the rest of the delay components. A long encoding buffer enables a more thorough evaluation of the signal properties, leading to higher coding efficiency and hence lower bit-rate. This is the reason why most low bit-rate coders often have high delay. Thus, coding delay in most cases is a trade-off with respect to the achievable bit-rate.

In the ideal case where infinite computational power is available, the processing delays (encoder and decoder) can be made negligible with respect to the encoder buffering delay. Under this assumption, the coding delay is equal to two times the encoder buffering delay if the system is transmitting in constant mode. For burst mode, the shortest possible coding delay is equal to the encoder buffering delay, where it is assumed that all output bits from the encoder are sent instantaneously to the decoder. These values are not realistic in the sense that it is achievable only if the processing delay is zero or the computational power is infinite: the underlying platform can find the results instantly once the required amount of data is collected. These ideal values are frequently used for benchmarking purposes, since they represent the lower bound of the coding delay. In the simplest form of delay comparison among coders, only the encoder buffering delay is cited. In practice, a reasonable estimate of the coding delay is to take 2.5 to 3 and 1.5 to 2.5 times the frame interval (encoder buffering delay) for constant mode transmission and burst mode transmission, respectively.

## 5.3.2.3 – Speech Coder Classification

Speech coders can be classified by various approaches such as: bit rate, waveform, parametric, hybrid, single and multi-mode coders.

- Bit Rate Classification: All speech coders are designed to reduce the reference bit-rate of 128 kbps toward lower values. Depending on the bit-rate of the encoded bit-stream, it is common to classify the speech coders according to Figure 5.3.2.3A. Different coding techniques lead to different bit-rates. A given method works fine at a certain bit-rate range,

but the quality of the decoded speech will drop drastically if it is decreased below a certain threshold. The minimum bit-rate that speech coders will achieve is limited by the information content of the speech signal. Judging from the recoverable message rate from a linguistic perspective for typical speech signals, it is reasonable to say that the minimum lies somewhere around 100 bps.

| Category | Bit Range |
|---|---|
| High | Greater than 15 kbps |
| Medium | 5→15 kbps |
| Low | 2→5 kbps |
| Very low | Less than 2 kbps |

Figure 5.3.2.3A - Classification according to bit rate

- Waveform Classification: These coders are better suited for high bit-rate coding, since performance drops sharply with decreasing bit-rate. This is an attempt to preserve the original shape of the signal waveform, and hence the resultant coders can generally be applied to any signal source. These coders work best at a bit-rate of 32 kbps and higher in practice at least. Signal-to-noise ratio (SNR) can be utilized to measure the quality of waveform coders. Some examples of this class include various kinds of pulse code modulation (PCM) and adaptive differential PCM (ADPCM).

- Parametric Classification: In the framework of parametric coders, the speech signal is assumed to be generated from a model, which is controlled by some parameters. During encoding, parameters of the model are estimated from the input speech signal, with the parameters transmitted as the encoded bit-stream. This type of coder makes no attempt to preserve the original shape of the waveform, and hence SNR is a useless quality measure. The perceived quality of the decoded speech is directly related to the accuracy and sophistication of the underlying model. Due to this limitation, the coder is specific to a signal, having poor performance for non speech signals. This class of coders works well for low bit-rate. Increasing the bit-rate normally does not translate into better quality, since it is restricted by the chosen model. Typical bit-rate is in the range of 2 to 5 kbps.

- Hybrid Classification: A hybrid coder combines the strength of a waveform coder with that of a parametric coder. Just like a parametric coder, it relies on a speech production model; during encoding, parameters of the model are located. Additional parameters of the model are optimized in such a way that the decoded speech is as close as possible to the original waveform, with the closeness often measured by a perceptually weighted error signal. As in waveform coders, an attempt is made to match the original signal with the decoded signal in the time domain. This class dominates the medium bit-rate coders, with the code-excited linear prediction (CELP) algorithm and its variants the most outstanding representatives. From a technical perspective, the difference between a hybrid coder and a parametric coder is that the hybrid coder attempts to represent the excitation signal to the speech production model, which is transmitted as part of the encoded bit-stream. The parametric coder, however, achieves low bit-rate by discarding all detail information of the excitation signal; only coarse parameters are extracted. A hybrid coder tends to behave

like a waveform coder for high bit-rate, and like a parametric coder at low bit-rate, with fair to good quality for medium bit-rate.

- Single Mode & Multi Mode Classification: Single-mode coders are those that apply a specific, fixed encoding mechanism at all times, leading to a constant bit-rate for the encoded bit-stream. Examples of such coders are pulse code modulation (PCM) and regular-pulse-excited long term prediction (RPE-LTP). Multimode coders were invented to take advantage of the dynamic nature of the speech signal, and to adapt to the time-varying network conditions. In this configuration, one of several distinct coding modes is selected, with the selection done by source control, when it is based on the local statistics of the input speech, or network control, when the switching obeys some external commands in response to network needs or channel conditions. Figure 5.3.3.2B shows the block diagram of a multimode coder with source control.

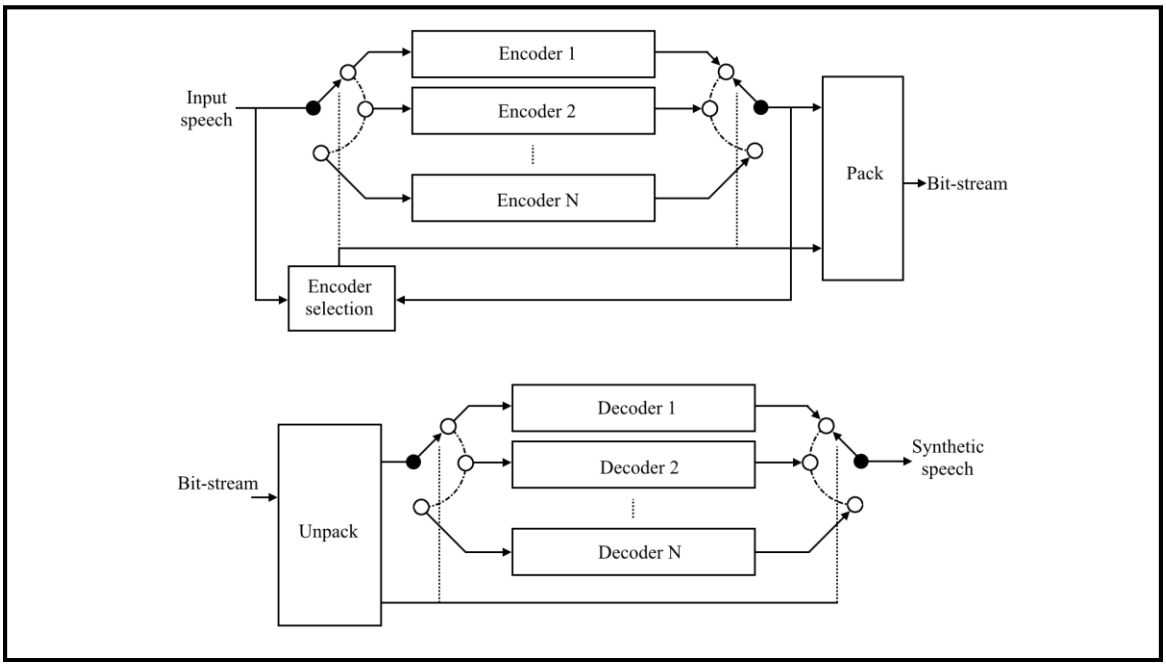


Figure 5.3.3.2B - Multi Mode encoder

In this system several coding modes are selected according to the properties of the signal at a given interval of time. In an open-loop system, the modes are selected by solely analyzing the input signal. While in a closed-loop approach, encoded outcomes of each mode are taken into account in the final decision. The mode selection information is transmitted as part of the bit-stream, which is used by the decoder to select the proper mode. Most multimode coders have variable bit-rate, where each mode has a particular, fixed value. Keeping the bit-rate varied allows more flexibility, leading to improved efficiency and a significant reduction in average bit-rate.

## 5.3.2.4 – Speech signal classification and modeling

A speech signal can be classified as voiced or unvoiced. Voiced sounds are generated when the vocal cords vibrate in such a way that the flow of air from the lungs is periodically interrupted, creating a sequence of pulses to excite the vocal tract. The vocal cords stationary, the turbulence created by the flow of air passing through a constriction of the vocal tract generates unvoiced sounds. In time domain, voiced sound is characterized by strong period present in the signal, with the fundamental frequency referred to as the pitch frequency, or simply pitch. For men, pitch ranges from 50 to 250 Hz, while for women the range usually falls somewhere in the interval of 120 to 500 Hz. Unvoiced sounds, on the other hand, do not display any type of uniformity and are essentially random in nature. Figure 5.3.2.4A shows an example speech waveform uttered by a male subject, where both voiced and unvoiced signals are present.

Figure 5.3.2.4A shows the randomness nature of speech signals, where statistics of the signal change constantly with time. For the voiced frame, there is clear reoccurrence at regular intervals in time domain, where the signal repeats itself in a semi uniform pattern; and also in frequency domain, where a harmonic structure is observed. The spectrum indicates more low-frequency contents, due mainly to the relatively low value of the pitch frequency. For the unvoiced frame, however, the signal is essentially random. From the spectrum we can see that there is a significant amount of high-frequency components, corresponding to rapidly changing signals. It is necessary to indicate that the voiced / unvoiced classification might not be absolutely clear for all frames, since during transitions (voiced to unvoiced or vice versa) there will be randomness and semi uniformity that is difficult to judge as strictly voiced or strictly unvoiced.

For most speech coders, the signal is processed on a frame-by-frame basis, where a frame consists of a finite number of samples. The length of the frame is selected in such a way that the statistics of the signal remain almost constant within the interval. This length is typically between 20 and 30 ms, or 160 and 240 samples for 8-kHz sampling.

In general terms, a model is a simplified representation of the real world. It is designed to help us better understand the world in which we live and, ultimately to duplicate many of the behaviors and characteristics of real-life phenomenon.

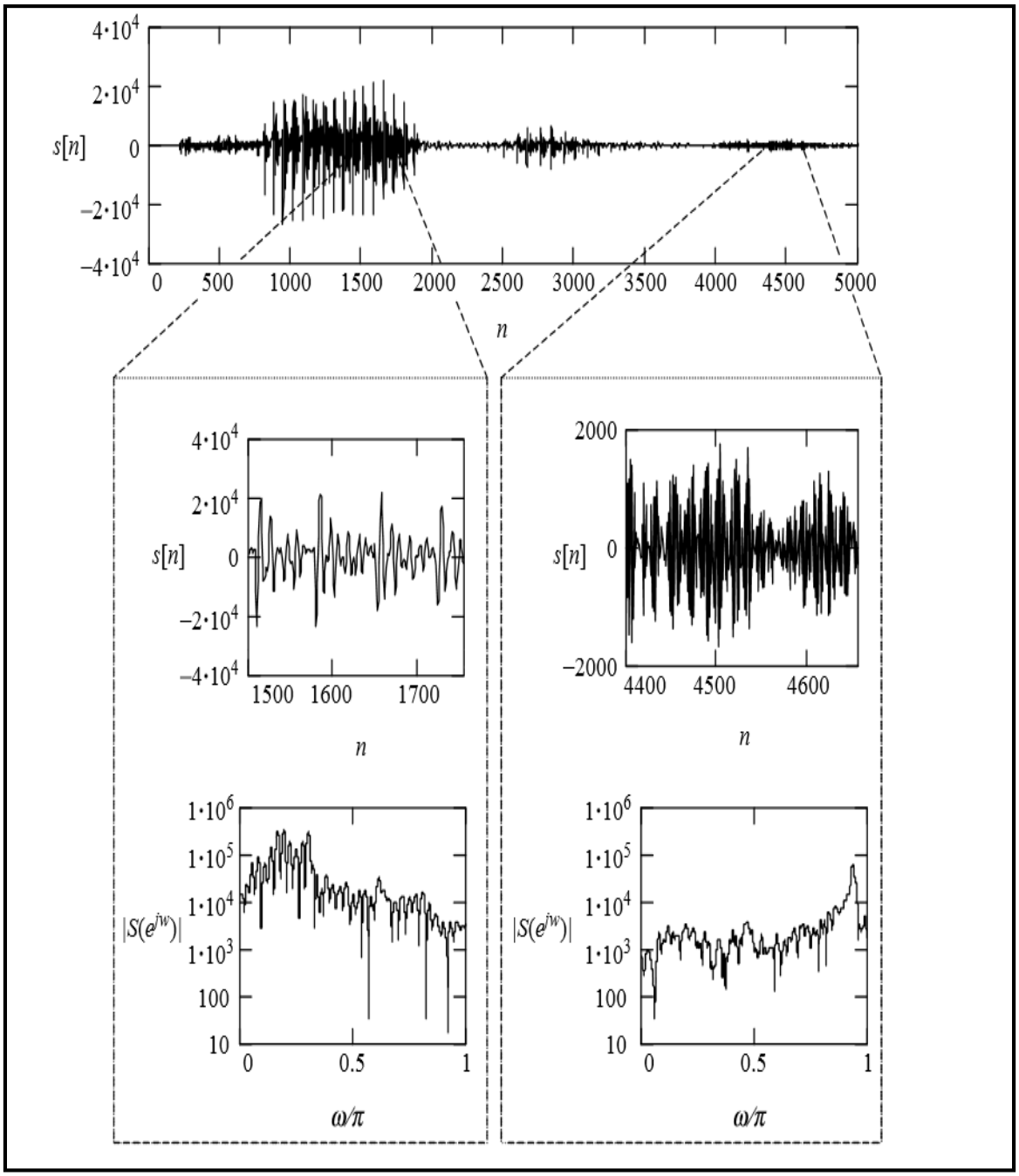



Figure 5.3.2.4A - Speech waveform showing a male subject. Voice and unvoiced frame show with Fourier transform plotted.

However, it is incorrect to assume that the model and the real world that it represents are identical in every way. In order for the model to be successful, it must be able to replicate partially or completely the behaviors of the particular object or fact that it intends to capture or simulate.

The human speech production system can be modeled using a rather simple structure: the lungs-generating the air or energy to excite the vocal tract- is represented by a white noise

source. The acoustic path inside the body with all its components is associated with a time-varying filter. The concept is illustrated in Figure 5.3.2.4B.

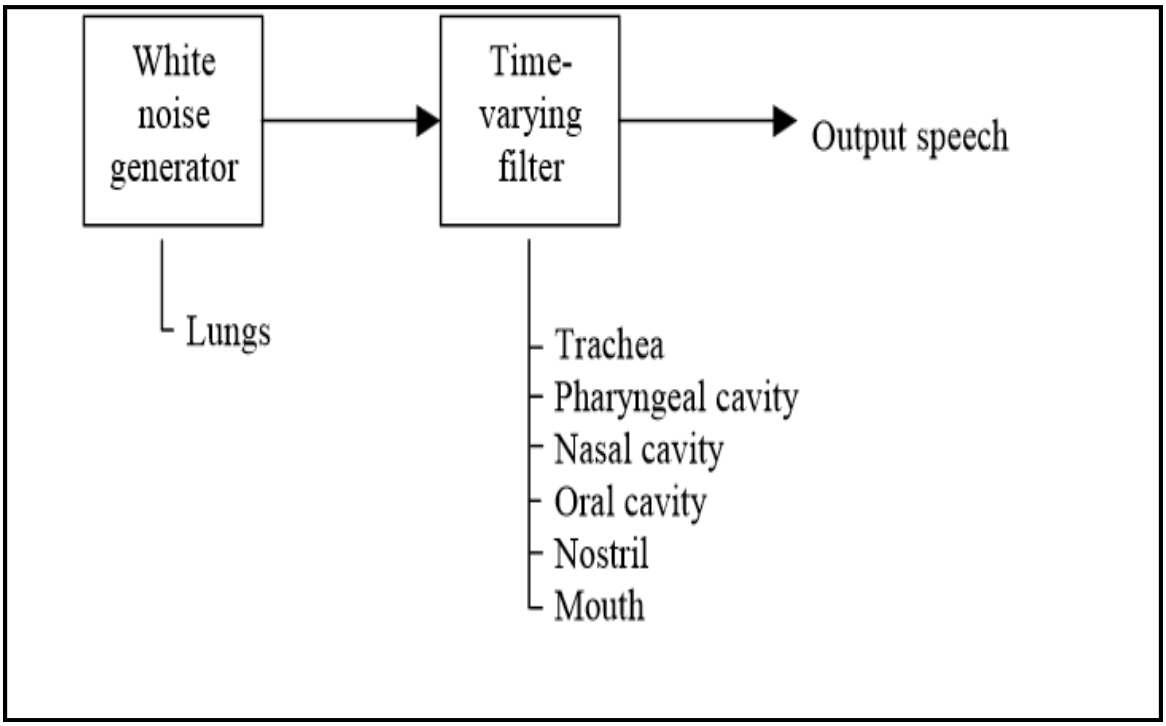



Figure 5.3.2.4B - Human speech production system

This model is the structure of many speech coding algorithms. By using a system identification technique called linear prediction, it is possible to estimate the parameters of the time-varying filter from the observed signal. The assumption of the model is that the energy distribution of the speech signal in frequency domain is totally due to the time-varying filter, with the lungs producing an excitation signal having a flat-spectrum white noise. This model is rather efficient and many analytical tools have already been developed around the concept. The idea is the well known autoregressive model.

## 5.3.2.5 – Structure of a Speech Coder

Speech coding is the application of data compression of digital audio signals containing speech. It uses speech-specific estimation of parameters using processing of audio signal techniques to model the speech signal, combined with generic data compression algorithms to represent the resulting modeled parameters in a compact bit stream. The two most important applications of speech coding are mobile communications and Voice over IP (VOIP).

The techniques used in speech coding are similar to that in audio data compression and audio coding where knowledge in psychoacoustics is used to transmit only data that is relevant to the human auditory system. For example, in narrowband speech coding, only

information in the frequency band 400 Hz to 3500 Hz is transmitted but the reconstructed signal is still adequate for intelligibility.

Speech coding differs from other forms of audio coding in that speech is a much simpler signal than most other audio signals, and that there is a lot more statistical information available about the properties of speech. As a result, some auditory information which is relevant in audio coding can be unnecessary in the speech coding context. In speech coding, the most important criteria is preservation of intelligibility and "pleasantness" of speech, with a constrained amount of transmitted data.

It should be emphasized that the intelligibility of speech includes, besides the actual content, also the ability to recognize the speakers' identity, emotions, pattern or melody of speech changes, which are all important for perfect intelligibility. Since it is possible that degraded speech is completely intelligible however, subjectively it can be annoying to the listener. The more abstract concept of pleasantness of degraded speech is a different property than intelligibility. In addition, most speech applications require low coding delay, as long coding delays interfere with speech interaction.

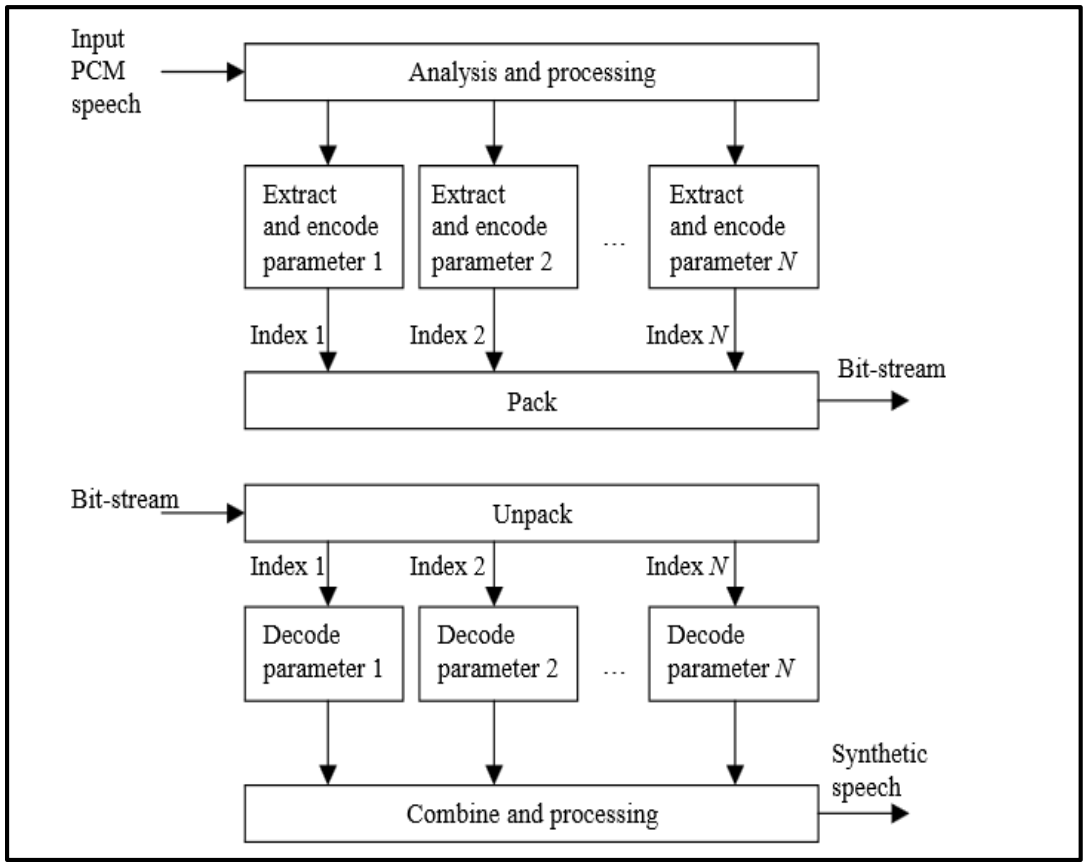


Figure 5.3.2.5 - Human speech production system

Figure 5.3.2.5 shows the generic block diagrams of a speech encoder and decoder. For the encoder, the input speech is processed and analyzed so as to extract a number of parameters representing the frame under consideration. These parameters are encoded with the binary

indices sent as the compressed bit-stream. The indices are packed together to form the bit-stream; that is, they are placed according to certain predetermined order and transmitted to the decoder. The speech decoder unpacks the bit-stream, where the recovered binary indices are directed to the corresponding parameter decoder so as to obtain the right parameters. These decoded parameters are combined and processed to generate the synthetic speech. It is the responsibility of the developer to decide the functionality and features of the various processing, analysis. Their choices will determine the performance and characteristic of the speech coder.

## 5.3.2.6 – Speech Algorithms

In speech recognition there are two important parts of the statistical based algorithm, the acoustic and language based modeling. Speech recognition requires two files to recognize speech; an Acoustic model is created by making recoding of speech and their textual representations. Afterwards, software is used to create a statistical representation of sounds that would then make up each word. They also require a language model which is really a grammar file that contains sets of predefined combination of words. This is what a speech recognition engine uses to recognize speech. According to a Microsoft research paper:

- Acoustic Modeling: Modeling of speech typically refers to the process of establishing statistical representations for the feature vector sequences computed from the speech waveform. Hidden Markov Model (HMM) is one most common type of acoustic models. Other acoustic models include segmental models, super-segmental models (including hidden dynamic models), neural networks, maximum entropy models, and (hidden) conditional random fields, etc.

Acoustic modeling also encompasses "pronunciation modeling", which describes how a sequence or multi-sequences of fundamental speech units (such as phones or phonetic feature) are used to represent larger speech units such as words or phrases which are the object of speech recognition. Acoustic modeling may also include the use of feedback information from the recognizer to reshape the feature vectors of speech in achieving noise robustness in speech recognition.

Speech recognition engines usually require two basic components in order to recognize speech. One component is an acoustic model, created by taking audio recordings of speech and their transcriptions and then compiling them into statistical representations of the sounds for words. The other component is called a language model, which gives the probabilities of sequences of words. Language models are often used for dictation applications. A special type of language models is regular grammars, which are used typically in desktop command and control or telephony IVR-type applications.

In Figure 5.3.2.6A it shows how the Hidden Markov Model is used for speech recognition.

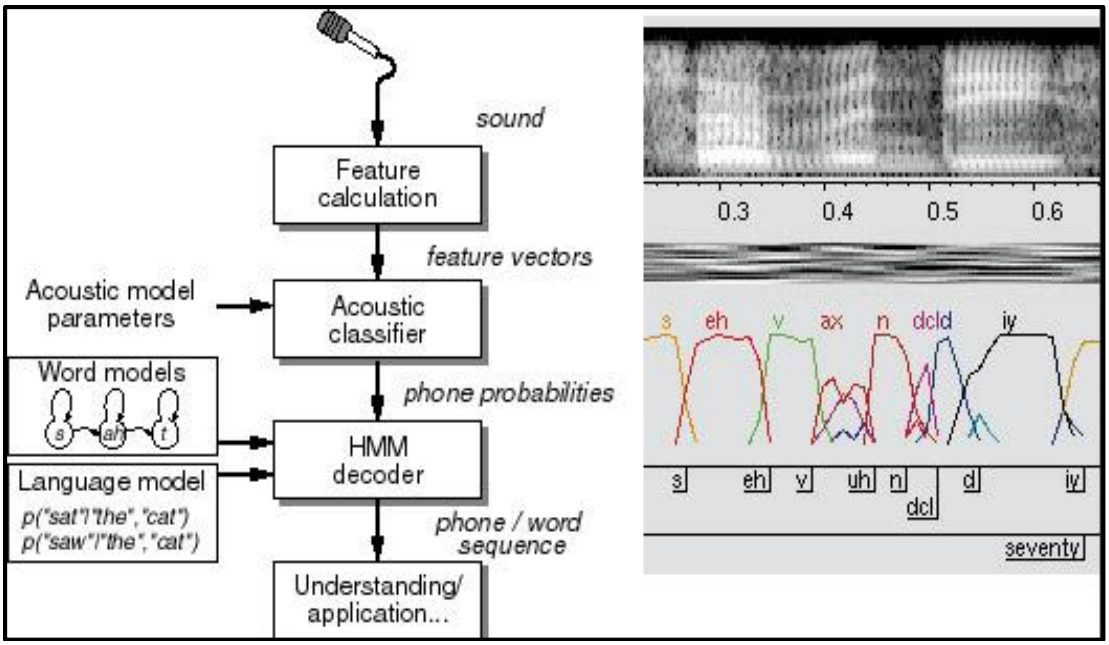


Figure 5.3.2.6A - Implementation of Speech recognition using HMM

- The language model assigns the probability to a word sequence using probability distribution. This model provides context to differentiate between words and phrases that sound similar. There are many instances when phrases that may not sound familiar to the human ear do so in fact to a computer, such as "recognize speech" and "wreck a nice beach." These common mistakes are what a great language model can help eliminate all together. One possible solution to make sure the probability of word depends on the preceded word or n word is the N-gram model. In this model it makes a probability calculation based on this chain rule where p is probability and w is word:

P(W1,W2,W3.....WN)= P(W1) P(W2|W1) P(W3|W1,W2).....P(Wn|Wn-1)

Figure 5.3.2.6B and Figure 5.3.2.6C are a great example of the language model used in day to day life. The feature many have come to love, swipe and hate at times, autocorrect, both use predictive features that are indicative of the language model.

Figure 5.3.2.6B - Swipe

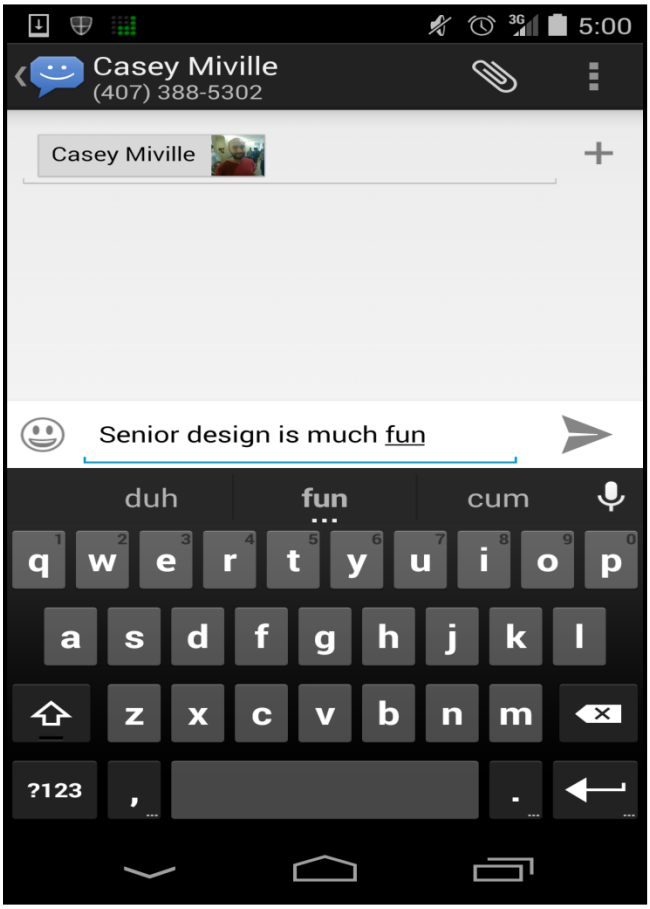


Figure 5.3.2.6C - Autocorrect

In speech recognition, the ability to compress sound data is valuable to keep the system requirements down such as computation and memory. The two logarithmic algorithms used for this are Mu law and A law. Simply they are compression for 16 bit sound that is

lossy, which means they are degraded from the original source. Both algorithms provide a 2:1 ratio, which means the compressed state will be half that of the original file. A Texas Instruments research paper defines mu law as:

- Mu law: a form of logarithmic data compression for audio data. Due to the fact that we hear logarithmically, sound recorded at higher levels does not require the same resolution as low-level sound. This allows us to disregard the least significant bits in high-level data. This turns out to resemble a logarithmic transformation. The resulting compression forces a 16-bit number to be represented as an 8-bit number

Simply stated since the human ear can only hear the middle of the sound wave, data can be removed from both lower and upper extremes of the sound frequency. Both algorithms take advantage of this by compressing the 16 bit audio in a way that is still intelligible to the human ear. The most significant difference between mu law and A law is that Mu law keep the top five bit of precision and uses a log function to determine the last three bits. A law algorithm keeps the first four bits and uses a log function to determine the last four bits. The following table shows a pulse code modulation numerical value representation.

| Numerical Value | Mu-Law<br>12345678 | A-Law<br>12345678 |
|---|---|---|
| +127 | 10000000 | 11111111 |
| +96 | 10011111 | 11100000 |
| +64 | 10111111 | 11000000 |
| +32 | 11011111 | 10100000 |
| 0 | 11111111 | 10000000 |
| 0 | 01111111 | 00000000 |
| -32 | 01011111 | 00100000 |
| -64 | 00111111 | 01000000 |
| -96 | 00011111 | 01100000 |
| -126 | 00000001 | 01111110 |
| -127 | 00000000 | 01111111 |

# 5.4 – On Board Speech to Text

## 5.4.1 – Advantages

The premise of on board speech to text sounds great at first. It can simplify the development process by making the entire software run on a development board. Developers have to consider the computational cost given the amount of input data. The goal is always to quickly process data and get the desired output. Utilizing a DSP board as shown in Figure 5.4.1, developer can have an addition and multiplication done in one step. DSP boards, as pictures in Figure 5.4.1, are very good for the development of fast algorithms and allows for simulations of real filters only using digital circuitry. DSP are cost effective and applicable for applications such as lossless compression. They can be easily upgraded due to their versatility and easily modified.

Figure 5.4.1 - Texas Instruments DSP board

## 5.4.2 – Disadvantages

The disadvantage with using a DSP is that it requires high power consumption. This is a huge disadvantage especially if a developer is building something that needs mobility and a small footprint. Power consumption is a very critical area when designing a system that will be tasked with speech recognition. Another critical downside of on board development is that a DSP takes samples of a signal in intervals therefore information can be lost, which in constant speech is really important. Lastly there are limitations when processing signals at the two extremes of the sound frequency.

## 5.4.3 – Options for implementation and feasibility

Several weeks of research has led me to the disappointing conclusion that on board speech to text is not possible due to the processor requirements and data. As shown in Figure 5.4.3, the most that can be done on board is speech recognition, where a certain set of words or commands can be uttered and then recognized by the system, i.e. Arduino. Or text to speech modules like the EMic 2, textspeak.com's TTS-EM HD series.

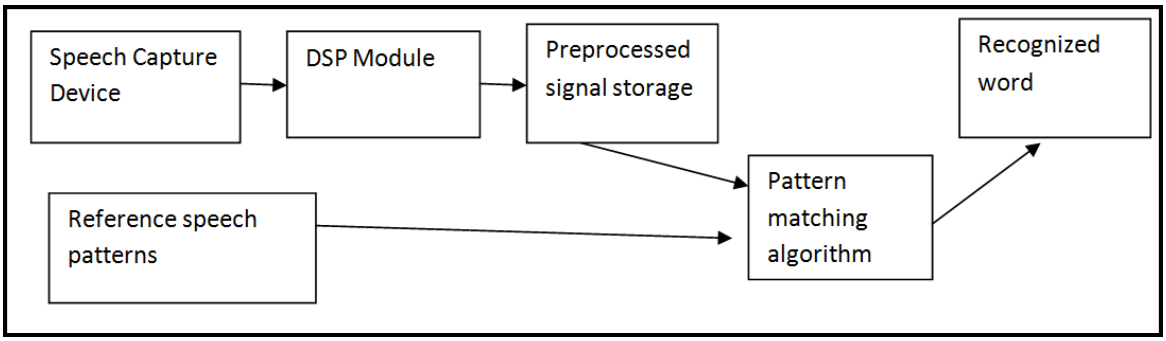


Figure 5.4.3 - Speech recognition on board

# 5.5 – Cloud Based Speech to Text

Cloud based speech to text utilizes Hidden Markov models(HMM), Dynamic Time warping(DTW) based speech recognition or Neural networks on a cloud server to process inquiries from different devices. As shown in Figure 5.5, it uses various combinations of techniques to improve results.

- HMM: This is a popular algorithm due to that fact that is can be automatically trained and feasible computation wise. When decoding the speech, a Viterbi algorithm is used to find the most probable path through a probabilistic model that is scored by time and state order.

- Viterbi: A search algorithm to find the best path. It does this in a very efficient manner. Utilizes forward dynamic programming in an efficient, recursive manner that performs exhaustive optimal search. It exploits the first-order Markov property which only needs to keep the most probable path of each state. At each state it keeps the most probable path and discards the rest

- DTW: This algorithm was using prior to the more popular HMM algorithm. It measures sequence similarities that vary in speed or time. It can be utilized for video or audio or graphics. Any data set that can be looked at in a linear representation can be analyzed using this algorithm. It has been used for automatic speech recognition.

- Neural Network: This acoustic model makes no assumptions as HMM would. It depends largely on the amount of inputs.

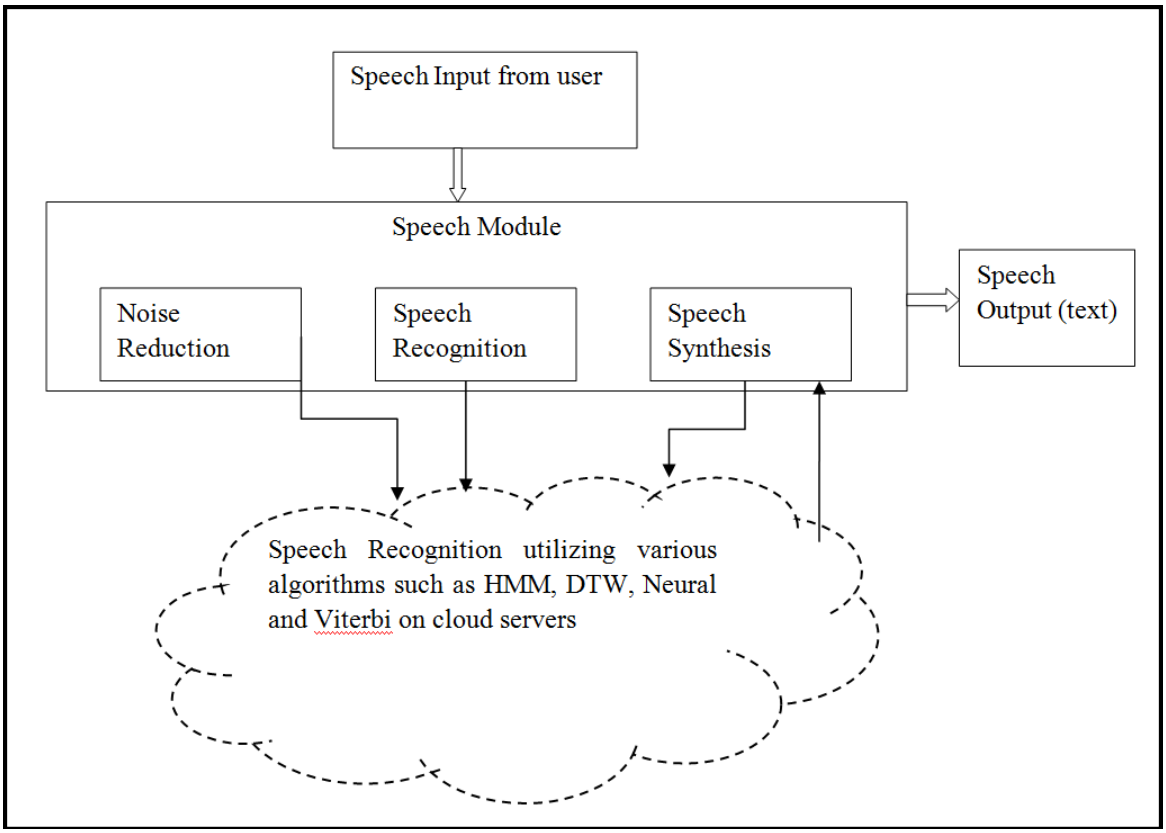


Figure 5.5 - Cloud Based Speech to Text

## 5.5.1 – Advantages

The advantages of cloud based speech to text far outweighs the on board speech to text which is not feasible. Developers do not have to worry as much about computational resources since all the processing is done on the cloud server which has more computation power than a desktop or mobile device. Increased speed and agility matter when a user is awaiting a response, the goal is almost instant speech to text which takes great resources that cannot be done on board. In a cloud based environment automatic updates are done without the application developer having to implement it on the mobile or user device. Scalability is not as much an issue since cloud based environment can scale up to meet any demand. Cloud based services are pay as you go model so there is no waste of money. As shown in Figure 5.5.1, there are several databases of acoustic and language models at the disposal of cloud servers that cannot be matched on desktop or mobile devices alone.

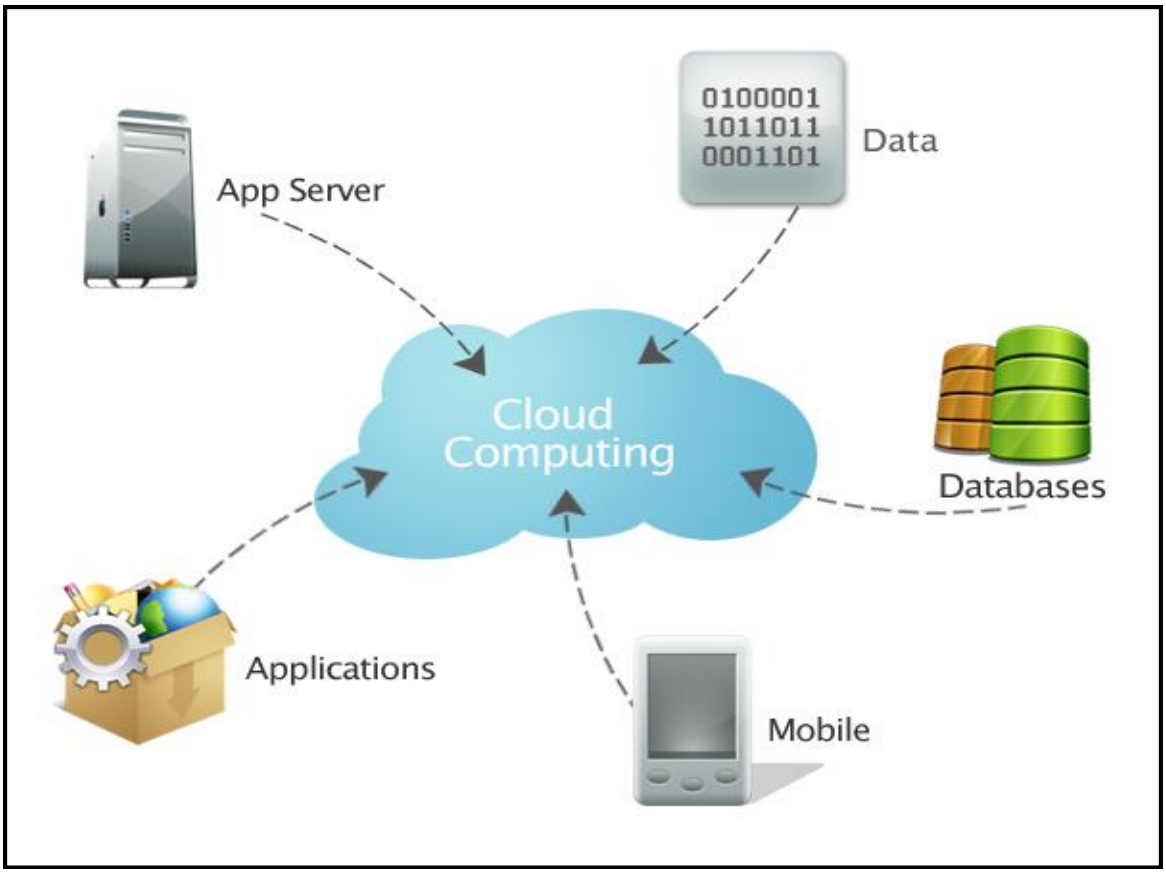


Figure 5.5.1 - Cloud computing environment

## 5.5.2 – Disadvantages

The primary disadvantage of cloud based speech to text is that is relies solely on the cloud. This network dependency is what makes its greatest flaw. When the network is down, so are all cloud services. If the connection is low or the network is being used during peak hours, all services that are linked to it run slowly as well. Another issue is compatibility, running different operating systems and ensuring that the handshake between the users' device and the cloud go right every time. Lastly security, the bigger the target the more risk there is involved. Since most cloud services are interconnected, once on system is breach the rest fall in a domino effect. Many of these disadvantages are due to the fact that this technology is still fairly new and will be improved upon as most services adopt cloud computing.

## 5.5.3 – Options for implementation and feasibility

As shown in figure 5.5.3A and 5.5.3B the options for cloud based speech to text are good. Most of these services are not free and require a subscription model. Google speech API has a quota that only allows for a limited query and only fifty requests per day. Further research has led me to the realization that although the Google speech API would have been great, it is no longer the best candidate for a production environment since it is: 1.

No longer being supported 2. Limits the requests of a developer's software. AT&T's speech API is also a subscription model but it offers a free sandbox access meaning a developer and try it out for free prior to production. The other services are also subscription based and primarily only offer web based speech to text or a transcription model that takes a while to return the transcribe text.

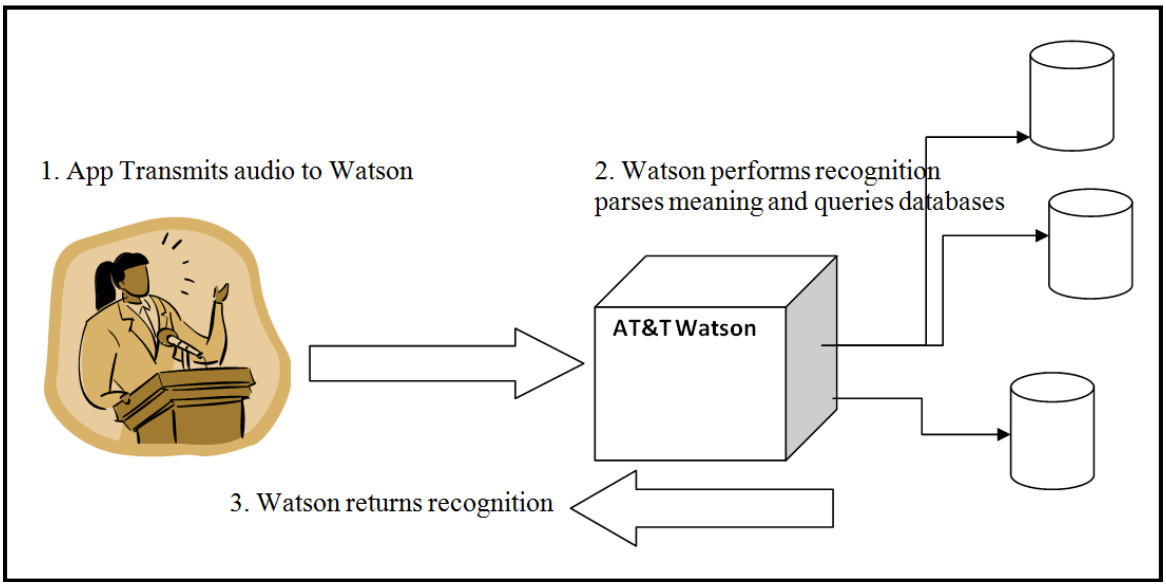



Figure 5.5.3A - AT&T Watson

| Application | Description | Website |
|---|---|---|
| Text Shark | Cloud based speech to text transcription | Textshark.com |
| Synthema Speech Scribe | Automatic Speech Recognition for audio and video transcription. | Synthema.it |
| VoxSigma Rest Api | Integrates speech to text into any application | Vocapia.com |
| AT&T Speech Api | Based on AT&T Watson speech recognition. | Reasearch.att.com |
| Google speech API Webspeech api | Using chrome or speech api, this cloud based speech to text can performs speech recognition very well | Developer.google.com |

Figure 5.5.3B - Options for Cloud based speech to text

## 5.5.4 – Graphic User interface

The graphic user interface I envisioned for this project is a very simple one that is intuitive and easy to use. Due to the fact that this project is targeted to the impaired I wanted to make the learning curve very low. The goal is that a user can intuitively find out how to use the program without any instructions or guidance. When the user presses the start

button, the program will automatically start a full screen capture of the audio, video and text on the fly. The text transcription will be saved to a notepad file for note taking so the student can have a backup. Also in case any problem or errors occur with the speech to text, there will be audio and video saved to the same folder as the text file. Figure 5.5.4 shows the graphic user interface I would like to implement.

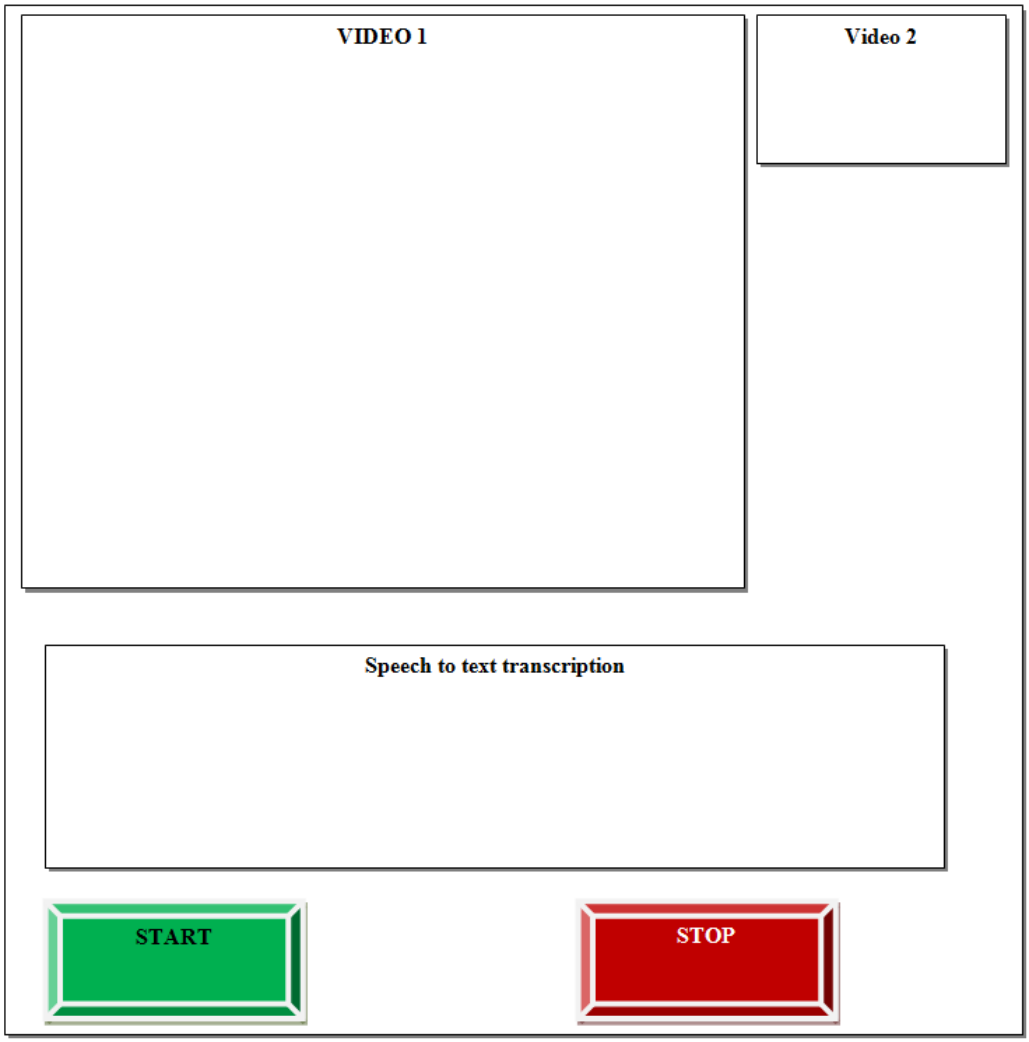


Figure 5.5.4 - Graphical User Interface

## 5.5.5 – Application packaging

Application packaging is the process of creating a small program that automatically installs software across numerous computers. It will have a set of default properties for the application it will install. Once the software is completed there are a couple of application packages to wrap an exe file to a windows installer(MSI). Application repackaging tool would capture the installation and run it and make basic settings. Here are the options so far for application packaging

1. MSI Generator: used for MSI packaging and creation

2. Admin Studio: includes install shield and various tools and a great alternative to MSI generator
3. App Deploy Repackager: Freeware windows installer that generates silent MSI setups to be deployed.
4. Advanced installer: packages and creates MSI setup, works with Visual Studio and Eclipse.

# 5.6 – Implementation

Software implementation happened twice, since we have two platforms that require their own unique software.

## 5.6.1 – Embedded Device Software Implementation

On the embedded device we decided to implement a Facial detection algorithm that uses Haar Cascades to distinguish if there is a face in the picture taken by our on device camera. The software then determines where the face is in the scene, and if that location is a significant distance from the center. To make sure to get accurate movement no matter the distance the face is from the frame, the software employs angles to move specifically to the target. Then embedded software also include the PWM controls, which take in the location that the device should point to and then move the servos to that corresponding location; first in the x- direction, then in the y- direction. Figure 5.6.1 A/B is a view of our final facial detection class diagram and its accompanying Servo control algorithm.

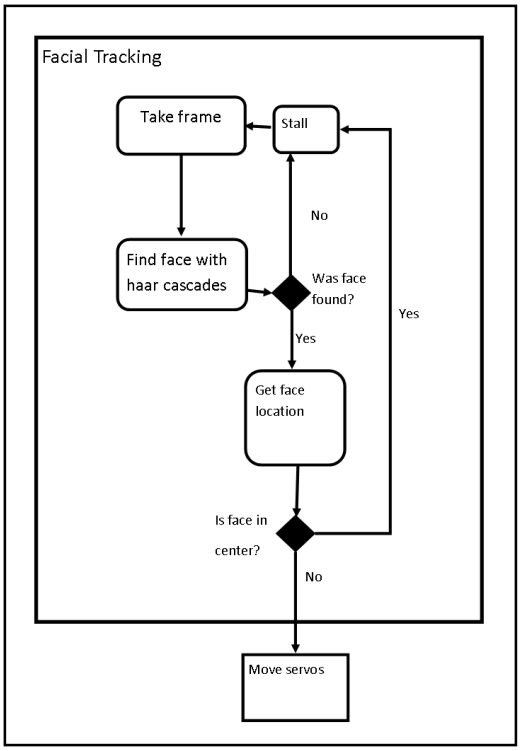



*Figure 5.6.1 A*

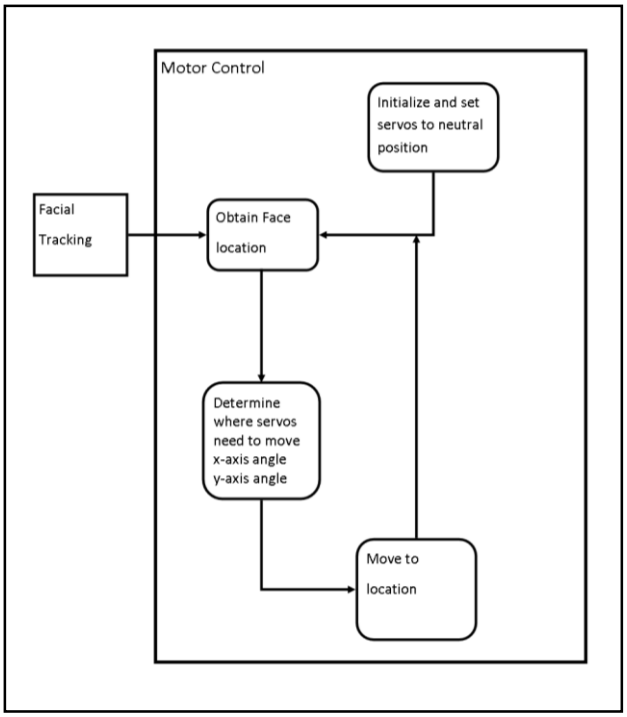



*Figure 5.6.1 B*

## 5.6.2 – Laptop Software Implementation

On the laptop environment we implemented the speech to text algorithm and the software that encoded our video and audio. The implementation looks very similar to our prototypes. We used Microsoft's speech recognition software, built into Visual Studios to program the speech to text aspect of the software. We then use Direct X Capture to give the user camera and audio from our given inputs. The software is given the ability to save the text information and the video, with any devices audio perfectly overlaid. Figure 5.6.2 A is a class diagram outlining the final implementation of the software and Figure 5.6.2 B is a picture of the program that the users sees.

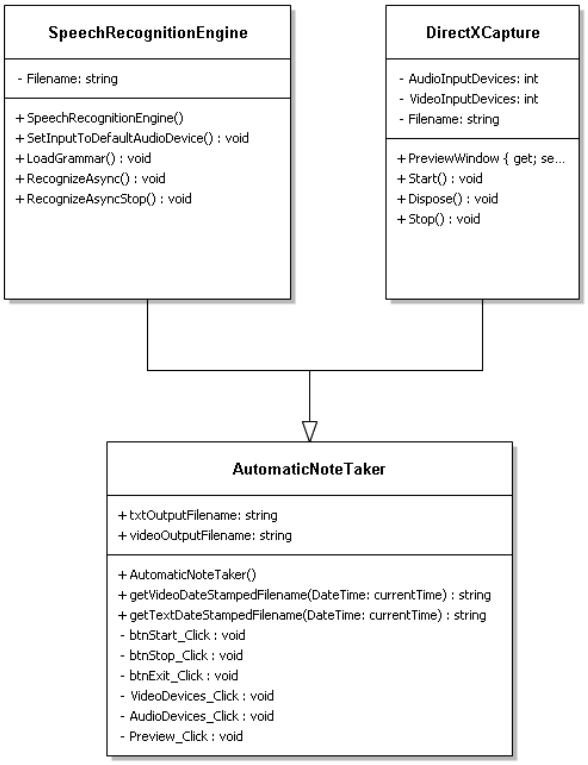



*Figure 5.6.1 A*


*Figure 5.6.1 B*

# 6.0 – Prototype Construction

## 6.1 – Component Acquisition

### 6.1.1 – Circuit Board Components

The following is a list of parts required for construction of the printed circuit board. This list gives the amount of each part required to build one circuit board. Multiples of each component will be purchased to prevent defective parts or construction errors from causing unnecessarily long delays in the production of a prototype.

Unless otherwise mentioned, all circuit board components will be sourced from Digi-Key, so both Digi-Key and manufacturer part numbers are provided.

Tantalum Capacitor (47 uF)

- Quantity: 1
- Price (Price for All): $0.99
- Digikey Part Number: 511-1500-1-ND
- Manufacturer: Rohm Semiconductor
- Manufacturer Part Number: TCTAL1A476M8BRS
- Package/Case Size: 1206 (3216 Metric)

Tantalum Capacitor (4.7 uF)

- Quantity: 1
- Price (Price for All): $1.02
- Digikey Part Number: 399-8453-1-ND
- Manufacturer: Kemet
- Manufacturer Part Number: T49B475K016AT
- Package/Case Size: 1411 (3528 Metric)

Ceramic Capacitor (47 uF)

- Quantity: 1
- Price (Price for All): $0.53 ($1.06)
- Digikey Part Number: 490-3907-1-ND
- Manufacturer: Murata Electronics North America
- Manufacturer Part Number: GRM31CR60J476ME19L

Ceramic Capacitors (22 uF)

- Quantity: 2
- Price (Price for All): $0.50 ($1.00)
- Digikey Part Number: 490-1824-1-ND
- Manufacturer: Murata Electronics North America
- Manufacturer Part Number: GRM31CR60J226KE19L

Ceramic Capacitors (2.2 uF)

- Quantity: 8
- Price (Price for All): $0.56 ($4.48; price break to $0.53ea at quantities above 10)
- Digikey Part Number: 399-9335-1-ND
- Manufacturer: Kemet
- Manufacturer Part Number: C1206C225K8PACTU
- Package/Case Size: 1206 (3516 Metric)

Ceramic Capacitors (100 nF)

- Quantity: 11
- Price (Price for All): $0.11 ($0.86 including price break for quantities of 10 - 49)
- Digikey Part Number: 399-9305-1-ND
- Manufacturer: Kemet
- Manufacturer Part Number: C1206C104K4RACTU
- Package/Case Size: 1206 (3216 Metric)

Thick Film Resistors (52.3 kOhm)

- Quantity: 2
- Price (Price for All): $0.10 ($0.20)
- Digikey Part Number: P52.3KFCT-ND
- Manufacturer: Panasonic Electronic Components
- Manufacturer Part Number: ERJ-8ENF5232V

Thin Film Resistors (10 kOhm)

- Quantity: 2
- Price (Price for All): $0.23 ($0.46)
- Digikey Part Number: MCA1206-10.0K-CFCT-ND
- Manufacturer: Vishay Beyschlang
- Manufacturer Part Number: MCA12060C1002FP500
- Package/Case Size: 1206 (3216 Metric)

Thin Film Resistors (2.2 kOhm)

- Quantity: 2
- Price (Price for All): $0.23 ($0.46)
- Digikey Part Number: P2.2KADCT-ND
- Manufacturer: Panasonic Electronic Components
- Manufacturer Part Number: ERJ-PO6J222V
- Package/Case Size: 0805 (2012 Metric)

Linear Voltage Regulator ($V_{OUT}$ = 3.3 V)

- Quantity: 1
- Price (Price for All): $4
- Digikey Part Number: *Will be obtained through TI*
- Manufacturer: Texas Instruments

- Manufacturer Part Number: TPS73233

Switching Voltage Regulator ($V_{OUT}$ = 5 V)
- Quantity: 1
- Price (Price for All): $3.39
- Digikey Part Number: ADP2303ARDZ-5.0-R7CT-N
- Manufacturer: Analog Devices, Inc.
- Manufacturer Part Number: ADP2303ARDZ-5.0-R7

Switching Voltage Regulator ($V_{OUT}$ = 5 V)
- Quantity: 1
- Price (Price for All): $3.06
- Digikey Part Number: ADP2302ARDZ-5.0-R7CT-ND
- Manufacturer: Analog Devices, Inc.
- Manufacturer Part Number: ADP2303ARDZ-5.0-R7

CMEMS Oscillator (20 MHz, 20 ppm)
- Quantity: 1
- Price (Price for All): $1.09
- Digikey Part Number: 336-2905-ND
- Manufacturer: Silicon Laboratories Inc
- Manufacturer Part Number: 501JCA20M0000BAG

Shielded Wire-Wound Inductor (6.8 uH)
- Quantity: 1
- Price (Price for All): $1.60
- Digikey Part Number: 445-3561-1-ND
- Manufacturer: TDK Corporation
- Manufacturer Part Number: VLF10040T-6R8N4R5

Shielded Wire-Wound Inductor (4.7 uH)
- Quantity: 1
- Price (Price for All): $1.60
- Digikey Part Number: 445-3560-1-ND
- Manufacturer: TDK Corporation
- Manufacturer Part Number: VLF10040T-4R7N5R4

Through participation in TI's student contest, this project has a $200 credit to the TI store, and TI also samples ICs. This credit and sample program will be utilized to obtain the following components:

BeagleBone Black development board
- This board is unlikely to be damaged during prototyping, so only one will be necessary.

Several Piccolo TMS320F28069FPT MCUs

- Quantity will depend on leftover credit at order time
- Possibly may be damaged during prototyping, as will be explained in section 6.2, and require replacements.

Several TPS73233 LDO Linear Voltage Regulator

- Quantity will depend on leftover credit at order time
- Possibly may be damaged during prototyping, as will be explained in section 6.2, and require replacements.

## 6.1.2 – Motion Components

The mechanical base of the pan/tilt system will be purchased from Servo City (www.servocity.com). The camera and microphone weigh less than seven ounces, so the following model was chosen:

1x SPT-100 Pan/Tilt System

- Rated for a load of up to 10 ounces.
- Height
  - 2 inches with the tilt bracket in the vertical position, measured from the flat bottom of the pan bracket (the point at which the pan RC servo attaches) to the flat top of the tilt bracket (the point at which the load attaches).
  - Approximately 3.75 inches, including the height of a standard size RC servo (which vary slightly in height from model to model).
  - Adding a load will increase the height.
- Weighs 1.5 ounces with no motors or load.
- Angular Range
  - Tilt Angle Range - Greater than 90 degrees. Limited by the angular range of the motor, except in cases where the bottom of the servo extends into the workspace of the tilt bracket, in which case the angular range of the servo will extend further in the negative (or positive) tilt angular direction than in the positive (or negative) tilt angular direction.
  - Pan Angle Range – Limited by the angular range of the motor. A DC motor with a digital encoder in place of potentiometric feedback would be capable of an angular range of 360 degrees or more, and would primarily be limited in angular range by the length of the tilt servo motor's cables. Continuous rotation could be realized for such a motor, but would come at the cost of significant hardware additions. This is unnecessary for the present application.

The tilt motor will be a standard size RC servo motor, likely a HiTec HS311. The design group has several of these size of motors on hand from previous projects, so these motors will not need to be acquired. Tilt angular range is not nearly as important for this application as is pan angular range, so the cheaper 90 degree angular range servos are sufficient.

The pan motor is not yet definitively picked due to the dilemma described at the end of section 3.1.6. Physical experimentation will be necessary before the pan motor component can be considered to be finalized. For now, the pan motor will be assumed to be the same servo as the tilt motor is.

### 6.1.3 – Sensor Components

Directional Microphone

- Price: $55.24
- Price Source: Amazon.com
- Manufacturer: Audio-Technica
- Manufacturer Part Number: ATR6550

Video Camera

- Price: $55.99
- Price Source: Logitech (via Amazon.com)
- Manufacturer: Logitech
- Manufacturer Part Number: B910 HD

## 6.2 – PCB Fabrication and Assembly

There are many possible ways to fabricate printed circuit boards, even at home. The basic of the DIY methods is to start with a silicon board with a layer of copper on one or both sides, and to remove all of the copper except for where the traces and pads will be. Etching is a popular method among hobbyists without access to expensive equipment. In the etching process, the traces are covered with an etchant-resistant material (usually, in the case of a hobby level etching, by a laser printer) before exposure to the etchant. The uncovered areas are removed via a chemical reaction, which is called etching. Recently, there has been an increase in the amount of hobbyists who are using home-made CNC routers to remove unwanted copper from their printed circuit boards instead of using a chemical process. Both of these processes are known as subtractive processes, as they remove the excess copper from the board.

Additive processes require less copper to produce a board and waste less of what they do use. A common method is to cover a bare silicon board (not plated in copper) with a film. The film is removed in areas where the traces are to be located, and then the exposed substrate is treated to allow it to be electroplated. This is then followed up by electroplating the exposed areas with copper, and then removing the film. Semi-additive process are very common, as they make it easy to fabricate the board with vias, which are conductor-plated holes connecting layers of a PCB. Vias are how the connections to the ground plane will be made in the printed circuit board designed for this application.

Only the two methods, both subtractive, mentioned at the start of this section are generally available to hobbyists looking to make the board themselves, with etching being far more common. Unfortunately, hobbyist boards are not generally fabricated with enough precision to use chips such as the Piccolo, which has a pitch of 0.5 mm (pitch is the distance between the center of one pin on an IC and the center of the adjacent pin on the IC). There are occasionally boards made with DIY CNC routers which have such precision, but these are very rare. As this is the case, the printed circuit board for this application will be ordered from a professional PCB fabricator.

PCB fabricators generally require a minimum number of boards to be ordered, although many companies allow orders with one to three boards if the design fits within certain limitations. Fabricator companies generally call these something along the lines of prototype boards, compared to fully featured or production boards. Additionally, there are generally a set of features which, when included in a design, make the board custom instead of standard, which can affect cost and lead time. Of particular interest to this project are student specials. Advanced Circuits (www.4pcb.com) offers 60 square inch 2-layer circuit boards for $33 each with a minimum order of three boards. However, with an academic email address, the minimum order quantity is waived and green solder mask with white silkscreen legend is included.

A computer aided design (CAD) program is used to make a schematic of the circuit and then design a printed circuit board layout for it. Depending on the complexity of the components involved in the circuit and of the CAD program itself, the schematic capture portion of the CAD program may allow spice simulations to be performed on the circuit (as with MultiSim/UltiBoard), or it may only check to see that no contradictions to electrical rules or pin declarations are present. Eagle CAD and DipTrace are two PCB CAD programs which have been used in the design process of the automatic note-taker so far. The schematic in section 4.5.3 was designed using Eagle's schematic capture software. The TMS320F2806x was not present in the libraries of either software, so a boundary scan description language (BSDL) file was obtained for the Piccolo F28069. An Eagle package (which contains a numbered list of pads for layout of an object onto a PCB) was created for the Piccolo, and then – after debugging errors of omission in the BSDL file – the BSDL file and package were used to implement the Piccolo in Eagle.

Some fabricators may offer a printed circuit board design program for use with their company, or they may accept the native files of a particular 3rd party printed circuit board design program, such as with DipTrace. However, the industry standard format which is accepted everywhere (and may be the only format accepted by a fabricator) is the Gerber format. The Gerber format is a 2D vector image format with only two 'colors', such as in an image composed only of black and white.

Gerber images are produced for each layer of a printed circuit board design when the model is exported from printed circuit board CAD software, since the two colors could not represent multiple layers of the board at one time. There will be a Gerber image for each layer of copper, for the solder mask, and for silkscreen printing. The file extension of Gerber files is .grb, but often in older versions of Gerber, a combination of representative

file names and non-gerber file extensions were used to clarify the function of a given file. Figure F6.2A is a screenshot of the list of files inside of the Gerber folder inside the hardware files folder of the Piccolo F2806x controlCARD's section of the controlSUITE program by TI, as well as a screenshot of the contents of one of these files when opened in Notepad. This is what is called a "Gerber Photoplot File", which is an image of the board design which was generated using a Gerber file, as can be seen in the first line of the Notepad view. Figure F6.2B shows the results of viewing this file using the free, in-browser PHO file viewer provided by gerber-viewer.com. An additional bit of information which can be obtained from a glance at Figure F6.2 is that the drill information is included as its own set of files. There are a number of file types which can be used to represent drill data. Gerber can represent drill data, but this is uncommonly done. In this set of files, the .drl file is an NC Drill file according to both the filename of the .drl file and the contents of the readme.txt file.

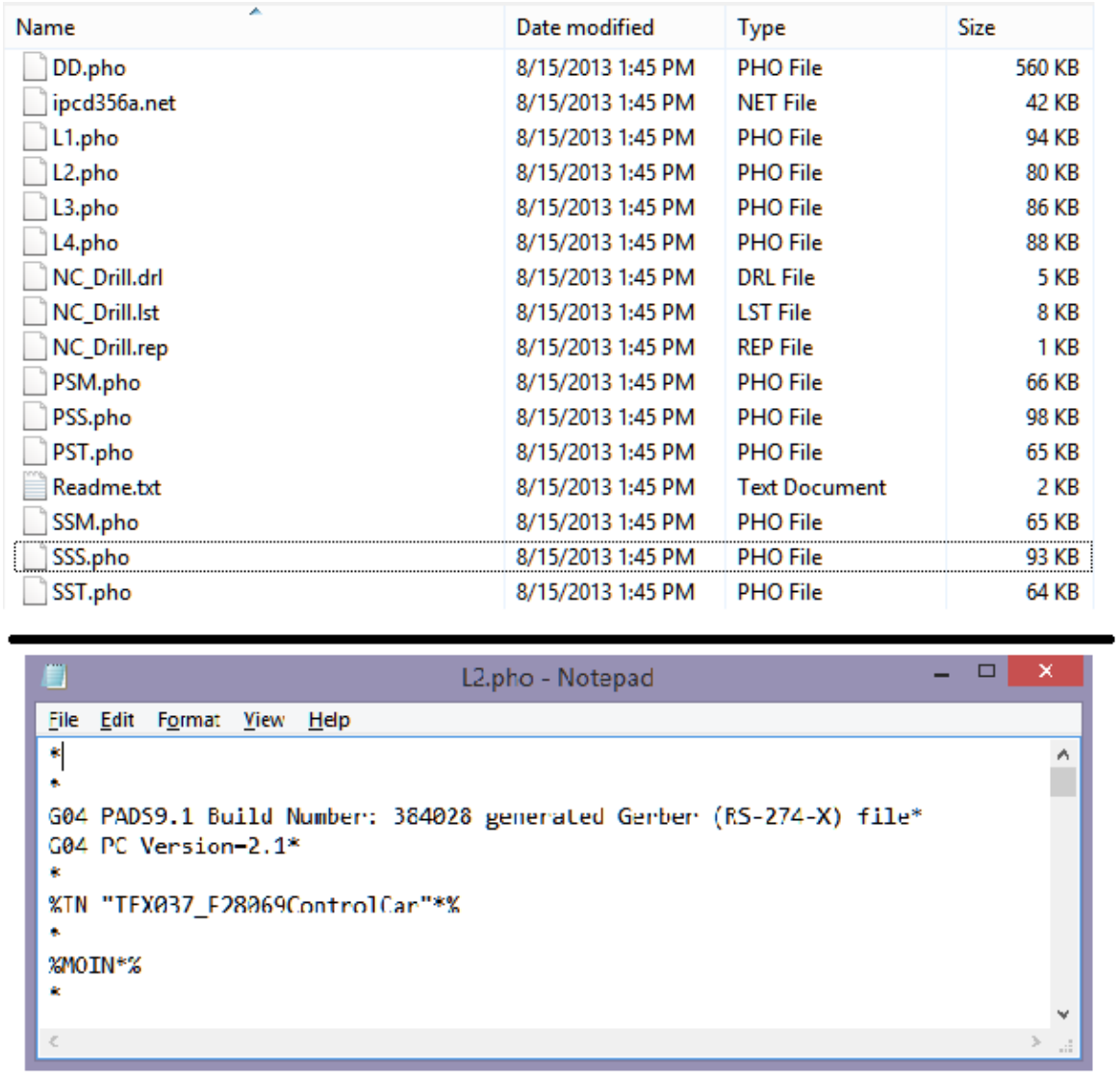



Figure F6.2A – Content of the "Gerber" Folder

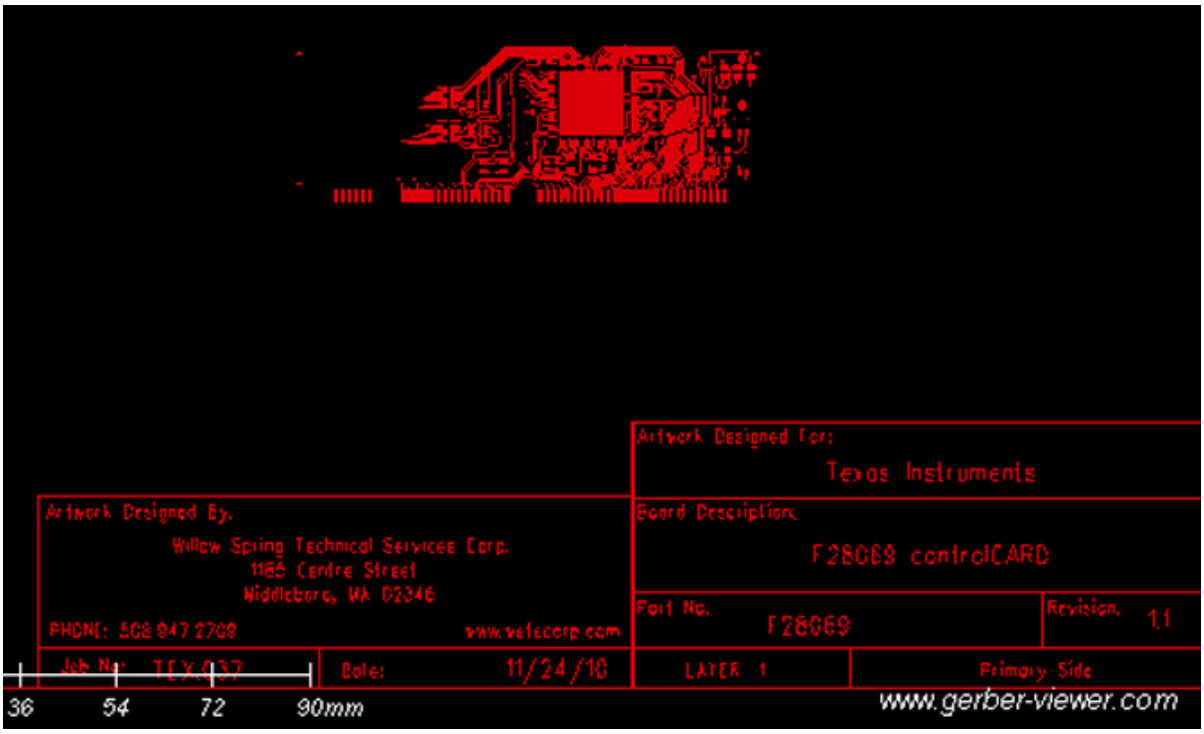


Figure F6.2B – Gerber Photoplot

# 6.3 – Subsystem Integration

The automatic note-taking system detailed in this design report consists of two major subsystems, which are explained in the following sections. Necessary and sufficient conditions for the subsystems to be considered a single subsystem with all of their sub-subsystems properly integrated are also given. These conditions focus only on the ability of the different sub-subsystems to interact with each other correctly, and not on the ability of the sub-subsystems to perform their associated duties correctly.

As an example, if the BeagleBone Black is sending the wrong data to the Piccolo board, and the Piccolo board is receiving this incorrect data completely and without transmission errors and using it to instruct the motors to move to the wrong positions, the two boards would still be considered properly integrated. Individual component's performance of associated tasks will be considered in chapter 7.

## 6.3.1 – Data Processing and Delivery Subsystem

The data processing and delivery subsystem consists of the two printed circuit boards an external desktop or laptop computer running the voice recognition software. Each of these can be considered its own sub-subsystem; namely, they are the computer vision and data distribution sub-subsystem (the BeagleBone Black), the power management and motion control sub-subsystem (the Piccolo PCB board), and the voice recognition and user

interface sub-subsystem (the external computer). To integrate these three sub-systems into the data processing and delivery subsystem, the following connections need to be made:

- Computer Vision and Data Distribution ⇔ Voice Recognition and User Interface
  - Connection 1: USB 2.0: For one-way data transfer to the external desktop or notebook computer
  - For the sub-subsystems to be considered integrated, the external computer should be able to, at minimum:
    - Detect that it is being sent data over its USB
    - Load the data from its USB input buffer into its RAM
    - Demonstrate that the data was received accurately by outputting the transmission through the its most relevant I/O device for the type of data that was sent so that the integration can be verified

- Computer Vision and Data Distribution ⇔ Power Management and Motion Control
  - Connection 1: SPI Bus: For one-way data transfer to the Piccolo board from the BeagleBone Black
  - Connection 2: A four-wire connection between three of the BeagleBone Black's GPIO pins (set as inputs) and three of the Piccolo board's GPIO pins (set as output)
  - Connection 3: A 2-wire cable connected between the output and ground of the 5 V switching regulator on the Piccolo board and the 5 V and ground input electrodes of the 5V DC input port on the BeagleBone Black.
  - For the sub-subsystems to be considered integrated
    - The Piccolo Board should be able to, at minimum:
    - Detect that it is being sent data over its SPI bus
    - Load the data from its SPI peripheral's input buffer into its RAM
    - Demonstrate that the data was received correctly by means of manipulating its GPIO pins in a way which allows the integration to be verified.

    - The BeagleBone Black should be able to, at minimum:
    - Enable external interrupts on one of the pins.
    - Execute an interrupt service routine (ISR) which reads the other two connected pins' values
    - Demonstrate that the correct interrupt condition was detected by manipulating its GPIO pins in a way which allows the integration to be verified.
    - Operate normally with the Piccolo board providing the sole power input.

## 6.3.2 – Data Acquisition Subsystem

The data acquisition subsystem contains two logical sub-subsystems: the data sensor sub-subsystem and the motion actuation sub-subsystem. The data sensor sub-subsystem consists of the microphone and camera modules, which are both oriented facing the same direction. The motion actuation sub-subsystem consists of the two motors and the pan/tilt system which is verified to move to and hold the expected positions sent by pulse-width

modulation without slowly and deviating from the position which it should be holding, and without generating excessive noise or heat.

For the sub-subsystems to be considered integrated:

- The camera and microphone should, at minimum:
  - Be physically attached to the pan/tilt system and still be oriented such that they are facing the same direction
  - Not reorient themselves relative to their original orientations in the reference frame of the tilt bracket or reposition themselves relative to their original positions in the reference frame of the tilt bracket as the motors reorient the pan/tilt system.

- The motors and pan/tilt system should, at minimum:
  - Be able to, with the camera and microphone attached, repeat the same angular position instructions which were used to demonstrate that the motors and pan/tilt system were properly working as a sub-subsystem without unacceptable performance degradation.

# 6.4 – System Integration

This section describes the necessary and sufficient conditions for the data processing and delivery subsystem and the data acquisition subsystem to be considered integrated into the complete note-taking system. As before, the performance of the individual subsystems is not relevant to this section.

In order for the data processing and delivery subsystem and the data acquisition subsystem to be considered integrated:

- The motors should, at minimum:
  - Operate with the Piccolo Board as their sole power source
  - Respond to the PWM signal received from the Piccolo board as expected

- The camera should, at minimum:
  - Operate with its USB connection to the BeagleBone Black as its sole source of power
  - Stream data to the BeagleBone Black through the USB connection.

- The microphone should, at minimum:
  - Stream audio data to the BeagleBone Black

- The BeagleBone Black should, at minimum:
  - Detect that it is being sent audio and video data from the microphone and camera
  - Move the data from the input buffers to the RAM in a timely manner so that information is not lost

- Demonstrate that the data being received is valid audio and video data by playing it back

- The Piccolo board should, at minimum:
  - Send PWM inputs to the motors as expected

# 6.5 – Component Mounting

Surface mount devices are typically attached to a printed circuit board by reflow soldering. This process begins by putting solder paste – a mix of solder suspended in flux – on the pins where all components will be placed. The board is typically then put onto a conveyor track which moves through a reflow soldering oven. The temperature in this oven varies in different locations, so the temperature which board is exposed to varies over time. The board is slowly warmed up initially, then heated quickly to melt the solder, and then cooled according to some prescribed temperature profile. The graph of temperature vs time shown in Figure 6.2, which is from Texas Instrument's "AN-2029 Handling and Process Recommendations" Application Note, is known as a solder profile, and it can be found in the literature for many devices. An additional method of industrial PCB soldering is wave soldering. Before wave soldering a board, all components are placed in their respective places, and any SMD components are glued to the board to keep them in place. The board is then moved along a conveyor throw a flowing wave of molten solder, which is wicked into the gaps between the metal pins and the metal pads, and the board is soldered.



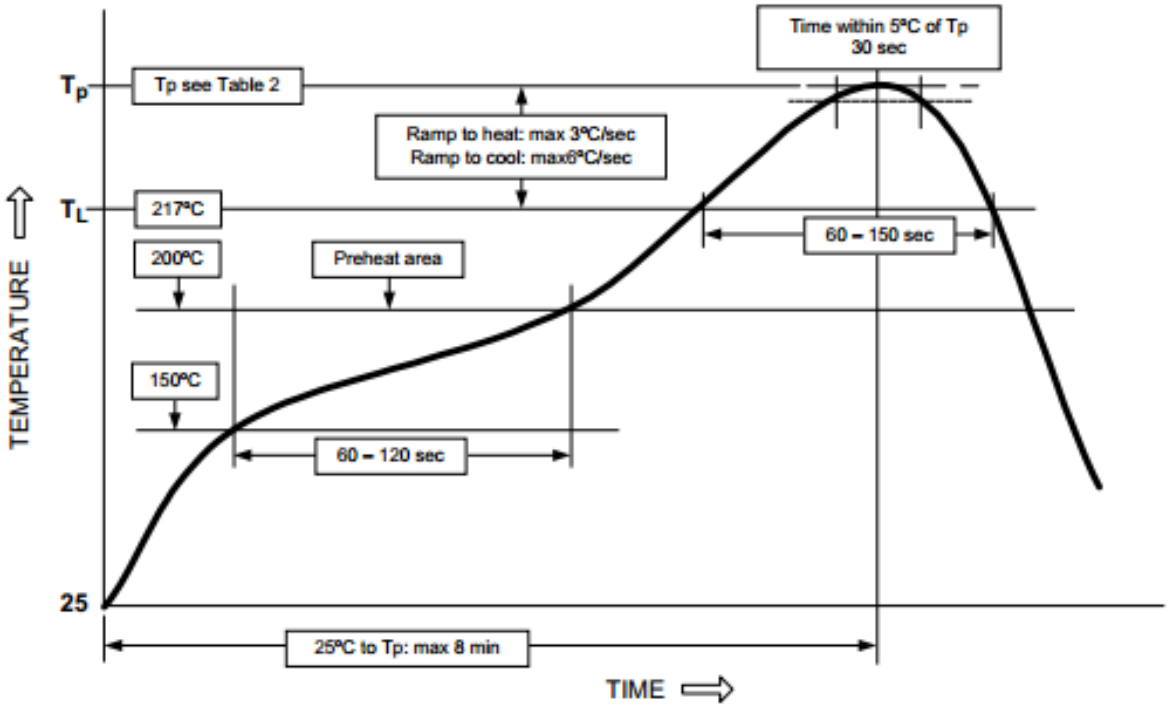



Figure 6.2 – Example Solder Profile

COURTESY OF TEXAS INSTRUMENTS

PCB manufacturers generally offer assembly services where they will solder your components to the board for you if you send them the components or, in some cases, pay an additional fee for the manufacturer to acquire the parts for you. PCB assembly services are generally offered on a cost-per-pin fee schedule somewhere in the neighborhood of 5 - 15 cents per pin. However, while this may make sense for large orders, there are generally tooling and startup charges which are charged at a flat rate per order regardless of the size, and are usually several hundred dollars. This is especially true if the board requires soldering of surface-mount devices (SMDs). All components on the board designed for this application are SMDs. As this project does not have a sponsor outside of the group itself, the board will be hand-soldered by group members.

The ICs will be soldered to the board by applying solder paste to the pads and heating them as similarly to their solder profile as possible by means of a hot air soldering station. Even very small pitch ICs can be soldered by hand using a soldering iron, but pads underneath the IC are problematic. The particular model of the Piccolo TMS320F28069 chosen is the PNT variant, which indicates that it is a plastic package with 80 pins and has a thermally enhanced package. The thermally enhanced package has a large metal contact on the bottom side, which should be soldered to a metal pad on the board that is connected to the ground plane. This allows the IC to use the board as a heat-sink. The components will be soldered one at a time, as opposed to the reflow or wave soldering methods. The largest components will be soldered first, as they will require the most heat to accidentally re-melt their solder joints while soldering other components, but the soldered components will be attempted to be shielded from the hot air, and the board will be allowed to cool in-between soldering different components. The smaller components, such as the capacitors and resistors, will be soldered using a pencil style soldering iron which is part of the same hot air soldering station.

# 7.0 – Prototype Testing

## 7.1 – Hardware Test Environment

### 7.1.1 – Microphone Test Environment

The Audio-Technica ATR6550 shotgun microphone testing environment is quite simple. The microphone will be positioned in a large room representative of an average college classroom. This environment must satisfy the following criteria:

- Sufficient space for multiple students to sit in desks
- Various objects (desks, chairs, equipment) must be placed in the room
- The front of the room must be clear enough for the presenter to have the ability to stand, walk, and gesture
- Little to no wall coverings

- Distance between front row of desks and presenter must not exceed 20'
- During testing rounds, no extraneous object is to be moved, including but not limited to doors, chairs, coffee mugs, etc.
- ATR6550 must be connected to PC before testing
- PC sound levels are to be normalized
- PC microphone input gain is to be set at 0dB

To prevent noise for interfering with the tests the following criteria must be met:

- Testing team must remain stationary
- Testing team must remain silent
- All air handlers must be off (air conditioning / heating)
- Doors and windows must remain closed
- Cellular devices within a large range of ATR6550 must be turned off

The microphone is to be positioned in such a way that the user would place it on a table. In the case of testing the ATR6550 on a stationary stand, the stand must be adjusted to the height of the user's desk.

All criteria is to be satisfied during testing only. Air handlers running, doors opening, power failures, etc. can be removed from the data.

## 7.1.2 – Camera Test Environment

The LS-Y201-2MP CMOS camera testing environment is easily managed in many spaces. The camera will be positioned in a room representative of an average college classroom. This environment must satisfy the following criteria:

- Sufficient space for multiple students to sit in desks
- Various objects (desks, chairs, equipment) must be placed in the room
- The front of the room must be clear enough for the presenter to have the ability to stand, walk, and gesture
- Little to no wall coverings
- Variable ambient lighting configurations
- Distance between front row of desks and presenter must not exceed 20'
- During testing rounds, no extraneous object is to be moved, including but not limited to doors, chairs, coffee mugs, etc.
- LS-Y201-2MP must be connected to PCB before testing

To prevent noise for interfering with the tests the following criteria must be met:

- Testing team must remain stationary, with exception of the presenter
- Testing team must remain silent
- All air handlers must be off (air conditioning / heating)

- Doors and windows must remain closed
- Ambient lighting must remain constant
- Cellular devices within a large range of LS-Y201-2MP must be turned off

The Camera is to be positioned in such a way that the user would place it on a table. In the case of testing the LS-Y201-2MP on a stationary stand, the stand must be adjusted to the height of the user's desk.

Between each test of the LS-Y201-2MP, data acquired must be transferred and viewed on a PC. While transferring data, tester must not move the stationary camera mount.

All criteria is to be satisfied during testing only. Air handlers running, doors opening, power failures, etc. can be removed from the data.

## 7.1.3 – Pan/Tilt and Servo Motor Test Environment (A)

The pan/tilt system test environment is somewhat different than the other environments. The microphone and camera modules will not be present in this test environment. A dummy camera and microphone of roughly the same shape, size, and weight will be mounted in their places, and an Arduino which has been proving to have working PWM outputs will be used in place of the application boards. The pan/tilt system, now significantly less delicate, will be placed on a large flat rectangular piece of hard wood which is thick enough not to flex when supported only at its ends. This environment must satisfy the following criteria:

- Semi-soft surface, such as thin economy carpeting, which is unlikely to significantly damage the system if it tips over onto it and yet is not springy enough to provide an unstable platform.
- Sturdy object of adjustable height, or a group of objects spanning a range of heights, which is/are capable of supporting the weight of the hardwood and the pan/tilt system if the hardwood is leaned on the object.

To prevent noise for interfering with the tests the following criteria must be met:

- The wood must be prevented from all non-negligible motions (translating, vibrating, or flexing) either by choosing wood with physical characteristics that meet this requirement or by external damping and weights.
- Testing team must remain stationary except to prevent the pan/tilt system from tipping or slipping off of the wood
- All air handlers must be off (air conditioning / heating)
- Doors and windows must remain closed

## 7.1.4 – Pan/Tilt and Servo Motor Test Environment (B)

Environment (B) for performing tests on the pan/tilt system and its servo motors is much more similar to the environments described in 7.1.1 and 7.1.2 than Environment (A) described for testing these same components. The camera will be mounted on the end-effector of the pan/tilt system while in this testing environment. This environment must satisfy the following criteria:

- Sufficient space for multiple students to sit in desks
- A marked position somewhere in the room where the center of the pan/tilt system's base will be placed.
- The front of the room must be clear enough to have a clear view of the wall.
- Little to no wall coverings.
- The walls should have 9 markings on them which – when viewed from the marked pan/tilt position – divide an angular range equal to the max range of the pan servo motor into 8 equal angular intervals.
- Bright ambient lighting which allows clear viewing of the marks on the wall.
- Distance between marked pan/tilt system position and the furthest mark on the wall should not exceed 20 feet.
- During testing rounds, the base of the pan/tilt system should not be allowed to translate or rotate at all. It should be physically forced to stay in this position, if necessary, but not by hand.
- LS-Y201-2MP must be connected to PCB before testing

To prevent noise for interfering with the tests the following criteria must be met:

- Testing team must remain out of view of the camera
- All air handlers must be off (air conditioning / heating)
- Doors and windows must remain closed
- Ambient lighting must remain bright and constant
- Cellular devices within a large range of LS-Y201-2MP must be turned off

While in this Environment, homing the motors should put the center marking in the center of the camera's viewing angle (at least horizontally).

Between each test of the LS-Y201-2MP, data acquired must be transferred and viewed on a PC. While transferring data, tester must not move the stationary camera mount or any of the wall markings.

All criteria is to be satisfied during testing only, with the exception of those regarding movement of the pan/tilt system or the wall markings, which must remain in the same position throughout the entire experiment. Air handlers running, doors opening, power failures, etc. can be removed from the data.

# 7.2 - Hardware Specific Testing

## 7.2.1 - Microphone Specific Testing

The Audio-Technica ATR6550 shotgun microphone will be tested in a controlled environment to eliminate various noise inputs and to isolate each desired testing datum. The testing environment is described above in section 7.1 and only must be maintained for the duration of the recording. For the purposes of these tests, we assume that there is no noise inputs to the system. 4 tests will be performed on the ATR6550 to assure that it will perform under the specifications set by this project. All data collected will be referenced to ensure the ATR6550 performs as expected. The "acceptability" range is set at the discretion of the tester and should be designed to denote a quality recording.

### 7.2.1.1 - Range Test

1. Mark range in 5' intervals in quiet interior location.
2. Create spreadsheet. must include data cells for each 5' interval.
3. Mount to static stand at 0' mark.
4. Ensure no object that may cause interference has direct contact with ART6550 or stand.
5. Position PC behind, out of its sensing radius.
6. Hook ART6550 output to microphone input on adjacent PC.
7. Open spreadsheet.
8. Turn on ATR6550.
9. Set ATR6550 to "tele" mode.
10. Run "Audacity" (open source audio editor) on PC.
11. Configure Audacity to accept ATR6550 as an input for recording.
12. Position portable speaker at 5' mark.
13. Aim ART6550 at portable speaker.
14. Configure speaker to play predetermined sound at a very low volume.
15. Click on the record button.
16. Play sound on portable speaker.
17. Stop recording.
18. Note peak input volume and record on spreadsheet.
19. Playback recording and record "acceptability" range on spreadsheet.
20. Save file.
21. Repeat Steps 12-20 but position portable speaker on the next 5' mark and repeat for the extent of marked range.
22. Repeat Steps 12-21 but increase volume on each round of tests.

### 7.2.1.2 - Noise Acceptance Test

1. Mark range in 5' intervals in quiet interior location.
2. Create spreadsheet, must include data cells for each 5' interval.
3. Mount to static stand at 0' mark.
4. Ensure no object that may cause interference has direct contact with ART6550 or stand.

5. Position PC behind, out of its sensing radius.
6. Hook ART6550 output to microphone input on adjacent PC.
7. Open spreadsheet.
8. Turn on ATR6550.
9. Set ATR6550 to "tele" mode.
10. Distribute extra noise speakers
11. Run "Audacity" (open source audio editor) on PC.
12. Configure Audacity to accept ATR6550 as an input for recording.
13. Position noise speakers in varying locations in the room, but do not position forward of the ART6550 sensor.
14. Configure noise speakers to play constant noise at constant volume.
15. Position portable speaker at 5' mark.
16. Aim ART6550 at portable speaker.
17. Configure speaker to play predetermined sound at a very low volume.
18. Click on the record button.
19. Play sound on portable speaker.
20. Stop recording.
21. Note peak input volume and record on spreadsheet.
22. Playback recording and record "acceptability" range on spreadsheet.
23. Save file.
24. Repeat Steps 15-22 but position portable speaker on the next 5' mark and repeat for the extent of marked range.
25. Repeat Steps 15-23 but increase volume on each round of tests.
26. Repeat Steps 15-24 and redistribute noise speakers.

### 7.2.1.3 - Cutoff Test

1. Mount to static stand.
2. Create spreadsheet, must include data cells for each volume level.
3. Ensure no object that may cause interference has direct contact with ART6550 or stand.
4. Position PC behind, out of its sensing radius.
5. Hook ART6550 output to microphone input on adjacent PC.
6. Open spreadsheet.
7. Turn on ATR6550.
8. Set ATR6550 to "tele" mode.
9. Run "Audacity" (open source audio editor) on PC.
10. Configure Audacity to accept ATR6550 as an input for recording.
11. Position portable speaker 1' from ATR6550.
12. Aim ART6550 at portable speaker.
13. Configure speaker to play predetermined sound at a very low volume.
14. Click on the record button.
15. Play sound on portable speaker.
16. Stop recording.
17. Note peak input volume and record on spreadsheet.
18. Playback recording and record "acceptability" range on spreadsheet.

19. Save file.
20. Repeat Steps 14-19 but increase volume on each round of tests.
21. Observe cutoff volume.

#### 7.2.1.4 - Movement Acceptance Test

1. Mount to Servo Base stand.
2. Create spreadsheet, must include data cells for each movement speed.
3. Ensure no object that may cause interference has direct contact with ART6550 or stand.
4. Position PC behind, out of its sensing radius.
5. Hook ART6550 output to microphone input on adjacent PC.
6. Open spreadsheet.
7. Turn on ATR6550.
8. Set ATR6550 to "tele" mode.
9. Run "Audacity" (open source audio editor) on PC.
10. Configure Audacity to accept ATR6550 as an input for recording.
11. Position portable speaker 1' from ATR6550.
12. Aim ART6550 at portable speaker.
13. Configure speaker to play predetermined sound at a very low volume.
14. Run small, slow movement function on Servo Base, ensuring the speaker does not leave the ART6550's sensor range.
15. Click on the record button.
16. Play sound on portable speaker.
17. Stop recording.
18. Stop Servo Base movement.
19. Note peak input volume and record on spreadsheet.
20. Playback recording and record "acceptability" range on spreadsheet.
21. Save file.
22. Repeat Steps 14-21 but increase Servo Base speed on each round of tests.

### 7.2.2 - Camera Specific Testing

The LS-Y201-2MP CMOS camera will be tested in a controlled environment to eliminate various noise inputs and to isolate each desired testing datum. The testing environment is described above in section 7.1 and only must be maintained for the duration of the recording. For the purposes of these tests, we assume that there is no noise inputs to the system. The "acceptability" range is set at the discretion of the tester and should be designed to denote a quality image.

#### 7.2.2.1 - Quality Test

1. Create spreadsheet. must include data cells for each resolution.
2. Mount camera to static stand.

3. Ensure no object that may cause interference has direct contact with LS-Y201-2MP or stand.
4. Position PCB behind, out of its sensing radius.
5. Position presenter 5' in front of LS-Y201-2MP.
6. Connect LS-Y201-2MP pins to PCB.
7. Open spreadsheet.
8. Turn on PCB.
9. Configure LS-Y201-2MP to capture 1600x1200 images.
10. Aim LS-Y201-2MP at presenter.
11. Run capture program on PCB and output to memory.
12. Presenter gestures in view of the LS-Y201-2MP.
13. Stop recording.
14. Playback recording and record "acceptability" range on spreadsheet.
15. Repeat Steps 9-14 but decrease capture resolution on LS-Y201-2MP.

#### 7.2.2.2 - Lighting Test

1. Create spreadsheet. must include data cells for each lighting: lightest to darkest.
2. Mount camera to static stand.
3. Ensure no object that may cause interference has direct contact with LS-Y201-2MP or stand.
4. Position PCB behind, out of its sensing radius.
5. Position presenter 5' in front of LS-Y201-2MP.
6. Connect LS-Y201-2MP pins to PCB.
7. Open spreadsheet.
8. Turn on PCB.
9. Set ambient lighting in recording space to very high.
10. Configure LS-Y201-2MP to capture 1600x1200 images.
11. Aim LS-Y201-2MP at presenter.
12. Run capture program on PCB and output to memory.
13. Presenter gestures in view of the LS-Y201-2MP.
14. Stop recording.
15. Playback recording and record "acceptability" range on spreadsheet.
16. Repeat Steps 10-15 but decrease ambient lighting.

### 7.2.3 – Pan/Tilt and Servo Motor Specific Testing in Environment (A)

The test procedures in this section require the environment described in section 7.1.3, the Pan/Tilt and Servo Motor Environment (A). The purpose of these tests are to determine what range of slopes the system operate on without knocking itself over. The servos will be given random position instructions with each position received will have a higher probability of being further away from the last given position than being near to it, hopefully resulting in quick, erratic motions from the servos. If vertical balance is lost by

such motions at small angles, it is clear that something must be done to rectify this as most classroom desks are sloped instead of parallel with the ground like a table top is.

#### 7.2.3.1 – Vertical Balance Test

1. Create a spreadsheet. Must include data cells for each angle tested.
2. Lay the wood flat on the semi-soft surface. Push down and ensure that the semi-soft surface gives very little under pressure.
3. Place Pan/Tilt System onto the piece of wood.
4. Ensure no object may come into direct contact with the dummy sensors on the end-effector plate, or with any other part of the device.
5. Record the current angle relative to the ground in the spreadsheet (zero degrees on the first iteration, and greater than zero degrees on all successive iterations).
6. Begin the random motion generation program.
7. Record in the associated data cell the intensity of the motion on a subjective scale of one to ten (one being negligible, and 10 being imminently about to fall over.
8. Stop the random motion generation program.
9. If the current angle was greater than or equal to the pre-determined stopping point or if at least one observer recorded an intensity level of 9 or 10 or if two or more experimenters recorded an intensity level of 8, end the experiment.
10. If neither of the cases in step 9 were true, remove the device from the wood and increase the angle relative to the ground by increasing the elevation of one side of the wood.
11. Constrain the other side of the wood with a stopper if necessary to prevent it from sliding.
12. Return to step 3.

### 7.2.4 – Pan/Tilt and Servo Motor Specific Testing in Environment (B)

The test procedures in this section require the environment described in section 7.1.4, the Pan/Tilt and Servo Motor Environment (B). The purpose of these tests are to study the factors that affect the smoothness and speed of a motion from one angular position to another. Only the pan angle is considered, as pan movements will be larger and more frequent than tilt movements in practice.

#### 7.2.4.1 – Pulse Width Sweeping Instructions vs. Direct Position Instructions Test

1. Create a spreadsheet. Must include timing data cells for each angle sampled.
2. Start recording video.
3. Instruct the pan servo to move to the most positive pan angle.
4. Wait for the servo to reach its destination.
5. Instruct the pan servo to move to the most negative pan angle.
6. Wait for the servo to reach its final position.
7. Stop recording video.

8. Review the video, writing down each time when it finishes moving through an additional one eight of the total travel distance, including the total travel time. Call the time taken to travel the first one-eighth of the total travel distance **t**.
9. Instruct the servo to move to the most positive pan angle.
10. Wait for servo to reach its destination.
11. Start recording video.
12. Send an instruction to move 1/8 of the total travel distance in the negative pan direction.
13. After each additional time **t**, instruct the servo to move an additional 1/8 of the way further than the last instruction, such that at time 7**t** you are instructing it to move to the most negative pan angle.
14. Wait for the servo to finish moving.
15. Instruct the servo to move to the most positive pan angle.
16. Wait for the servo to finish moving.
17. Repeat steps 12 through 16 an additional 2 times, except instead of the distance interval being 1/8 of the total travel interval, use 1/16 the first repetition and 1/32 the second repetition. Use the same time **t** as the time interval between position instructions being sent to the servo in each repletion.
18. Stop recording.
19. Review the video and use the data to graph angular position vs. time for each positive to negative travel that was performed on the same graph, and compare the results.

#### 7.2.4.2 – Input Voltage Tests (Extension of experiment 7.2.4.1)

1. Decrease the servo's voltage supply to 75% of the original voltage used the first iteration performing of experiment 7.2.4.1.
2. Repeat experiment 7.2.4.1.
3. Decrease the servo's voltage supply to 50% of the original voltage used the first iteration of experiment 7.2.4.1.
4. Produce 4 graphs of angular position verses time. Each graph should contain a position vs. time graph of one run from the original experiment 7.2.1 as well as the graphs of the corresponding run from the second and third repetition of the experiment.

# 7.3 – On-Board Software Test Environment

## 7.3.1 – Hardware Environment for Software Testing

The development board used for software testing of the facial tracking is the BeagleBone Black. This board has a Sitara A8 1GHz processor, 512MBs of DDR3 RAM, and 4GBs of Flash memory. The board uses a floating point graphics accelerator and can render HDMI output images. The board uses between 210-460 mA @5V power while operating without any additional attachments. These specifications will help with the project because it will heavily rely on video processing. The board will meet our needs exactly and will provide enough power to process and produce the desired results. We chose this board because of its Sitara A8 processor and the large amount of RAM that the system had available. It also

has a floating point graphics accelerator that will help our program to analyze the incoming video. These features were better than that of other boards of comparable design and price range. The board also has the bonus of having a Linux operating system pre-built into it using onboard ROM. This is great because it frees up the SD card slot to put in external memory that we will use to program the board. We will be using a USB webcam for testing purposes. The issue with this is that the board will have to be powered through a wall connection, because if powered through the USB there will only be 40-190 mA to power the camera. The camera requires at minimum 300 mA @5V power.

The camera being used to determine facial placement is a Logitech B910. This webcam gives 30 frames per second video at 640x480 pixels. This is good because most video software requires a 640x480 pixel format. It will also help with rendering the correct output with a great rate of accuracy, because of the relatively small pixel size. The camera also optimal because it has universal compatibility with any type of operating system and does not need drivers to operate. This is good because with a limited amount of space the drivers would have taken up vital secondary memory space, and there would be no guarantee that drivers exist for our specific embedded device. Drivers, especially for webcams are generally made for desktop operating systems and would have posed a problem with the embedded operating system that we are using.

## 7.3.1 – Software Environment

The BeagleBone Black comes pre-loaded with Debian GNU/Linux distribution. This operating system can work either through the command line of another operating, or as a free standing GUI. Since Debian is a Linux distribution there are many programming languages that work directly in the command line. Having embedded Linux installed also helps with memory and resource management. This version of Linux, being specifically for embedded device has some enhanced memory management for low powered machines, this is good because it will keep our program running smoothly with the limited memory available. We will exploit this by using an open source Computer vision library that is in one of these languages.

Simple CV is going to be the software library used. Simple CV is an open source program for computer vision applications. This software helps to cut down on development by using the common OpenCV libraries and some extras, without having the user manually set up or manage the overhead. SimpleCv also has a python shell, which is natively compiled in the Linux distribution that is already on the board. This environment will help with rapid development, as there are many algorithms prebuilt in. The prebuilt libraries is a large help because there are many different facial tracking algorithms that can be utilizes that are now just function call away. The Simple CV package is also very helpful because there is a large number of training programs, books and an internet tutorial on how to use the software package.

# 7.4 – On-Board Software Specific Testing

## 7.4.1 – Algorithms

The first algorithm to be tested is the KLT approach, because it uses optical flow. The main idea for our project is to get the flow of a face and then to center the image on it. Klt will help because it is doing this flow natively and the output can then be easily translated to something usable by the servos. in this approach the face will have vectors that come off of it and then we will try to measure and average these vectors. After the vectors are averaged we will then try to predict movement. with the movement predicted we will again sample a frame and check it for accuracy. If these frames come out to be exuding similar behavior we will then engage the servos to move to the predicted position. To avoid rigid movement there will be a wind up and a wind down of movement. This may cause slight offset of the object from the center, but we will do this to try and benefit the other component and to also try and not mess up the accuracy of this image. To create this wind up and down the plan is to try and normalize the movement to prevent drastic change in movement.

The next algorithm that we will be testing is going to be a simple combination of Facial detection to create bounding boxes over the face of the desired person. The algorithm should then check each image to see the pixel change of the centroid of this bounding box. If the change is too large the algorithm should assume that the facial recognition messed up and to discard the results. If the centroid of the bounding box is in the same location or a similar location it will assume the object is the same and then check the movement to see if it should move the camera. If the camera is moved the algorithm should shut off until the movement has stopped ( no more than several milliseconds). Once this process is complete the algorithm should essentially reset. This process is to continuously loop until the object is centered. Once the object is centered the algorithm will check and make sure it's centered, if the object moves it will then go back to it's previous movement loop.

The third algorithm that is being considered for this project is the CamShift algorithm, which is a type of iterative gradient method of facial tracking (see 5.1.2). What this algorithm will do is to help with movement of the object that does not have their face perfectly towards the camera. This will help when the object is doing something such as writing on the board or turns to the side to point to a presentation. This algorithm will be used in conjunction with the KLT approach so that we can track the differences in movement on a more accurate target. What we hope to prevent in this testing set is loss of the target.

## 7.4.2 – Desired outputs

Simply stated the output should be a vector or integer to move the servos. That of course is the final output and there will be several incremental steps to get to this point. The first set of outputs will be to display a face with a bounding box over it. Then then next step is to get the program to follow a person's face. This step will be done by using one of the

methods above, most likely the KLT method. Once facial tracking is established we will then have the program output the location of the persons face. There are several way that we are thinking of displaying location. First to display the exact pixel in a i by j fashion. Secondly by displaying which quadrant the face is in (separated by splitting the picture in half horizontally and vertically). The final way we'd like to display location is based of the center of the image. After we get the person's location to display we will then have that location continuously update to tell us where their face is in real time. Once we get the location of the face we will then determine how far away from the center of the frame the face is, and display the Euclidean distance from that spot to the center. After this is determined we will put the direction that needs to be traveled. That will be the main portion of the initial output.

The second phase of the output will be to start determining force vectors to the center to help and move the face to the center of the image at all times, while not overshooting the center and keeping the servos from moving too fast or forcefully. this will be done by determining a vector strength base on how far the face is from the center of the image. We will then attempt to draw the vector coming from the relative center of the face. Once we determine what a good vector is at different static places we will then need an equation to scale that vector at all points. To do this we intend to normalize the vector based on the center being zero and each direction pointing to the center and scaling the vector drastically when hitting the edges of the image, but not while it is relatively close to the canter. Once we find good scaling factors we will heavily user test this.

After the vectors respond on screen as we desire we will then try to translate those vectors to readable measures that the servos can understand. Once this translation occurs we will test the system with the camera in a static position and having the servos move nothing. We will try to get the speed of the servos between being too slow to be effective and too fast. Once we feel that the servos are within this range of speeds we will attach the camera. We will again have a person move back and forth across the camera's view, this time will the camera trying to keep them centered. It is my belief that this point will cause the most headache for the program, because up until this point the camera has been static. I think that this will lead us to fine tuning the vector equation to make it the most responsive to the realistic scenarios that we will present it with.

## 7.4.3 – Projected issues

The projected issues with the testing will be several. The first problem that the program will face will be the robustness of our facial tracking algorithm. We are going to try and use the most speed efficient algorithm to track the face in real time, but this might give us a decrease in performance. To accommodate with this we plan on implementing multiple different facial tracking algorithms, this will hopefully make the program robust enough to handle difficult situations where the subject's face isn't in good view of the camera.

Creating the appropriate vectors will the next big problem we expect to encounter. This will be particularly difficult because of the unique nature of the problem. Because this is not a widely practices phenomenon these isn't much material to guide in producing our

desired result. we plan to combat this with extensive and repetitive testing to try and create an optimal result. The current idea is to draw a line from the centroid of the face to the center of the image, measure it and then to try and paste a vector onto this line that will match the intensity. This process will have the challenge of finding the center of the face, which right now we plan to do by running a separate facial detection algorithm, getting a bounding box and then finding the approximate center of the face from the dimensions of the bounding box.

The next problem will be the addition of moving parts into our program. Having a stationary object rack and send out signals is relatively easy compared to trying to send the same signals while in motion. We have several ideas to try and help this problem. The first idea is to design the program so that while the camera is moving there is no update on the target information. What we will be trading off here is the speed and reliability of the setup to keep the subjects face in the center of the frame, but this will help to reduce jerkiness and will trim down on the programs overhead by separating actions with processing. The next proposition that we came up with is to severely limit new motion from old motion. This approach will help the camera move at a steady rate even if the subject drastically changes location. This approach also takes into account the angularity of the stand, for a small movement left or right of the camera this could translate into large movement on behalf of the subject. The last idea we came upon is from robotics, call a vector field. What happens here is that the center would act as a goal and have force vectors pointing towards it. The face would act as the subject to be sent to the goal. what we would then do is to have the camera adjust position to try and keep the subject in the goal.

The final problem that is expected with our approach is that we are basing our assumptions on the fact that there will only be one subject moving in the front of the camera. What will have to be implemented is a sort of priority assignment placing emphasis on old subjects and discounting new subjects until they become relevant. For example we wouldn't want the camera to track a person going to sit down, but we would however want to track a new presenter. We can make a couple assumptions when dealing with this problem. the first assumption is that the presenter will be moving, but not by much. However we can assume a bystander will obstruct the scene and try to relatively quickly un-obstruct it. With this we can discount certain types of movement if they are consistent from the beginning to the end of the objects life in the scene. The next assumption is that any new speaker will move more than a previous speaker. To enact this kind of response we will weigh motion with the longevity of a subject in a scene. We will first have longevity be a predominant factor (to eliminate the camera from following  someone just walking across the scene) and then after a certain point we will focus on the subject with the most movement. This distinction is made with the knowledge that small things like mouth movement will be tracked and measured. Hopefully this will allow a smooth transition from one subject to another.

## 7.4.4 – Testing cases

The testing will be done using several different cases at each point during the testing process. The idea behind the different phases is if there is a problem then these different tests should pinpoint where the problem is arising. The testing environment will be kept

fairly empty, with the exception of a couch, which is too big to eliminate from the scene. During the testing the camera will be positioned in the same location and not moved (with the exception of when we finally connect the servos). In the beginning of testing we will only have one subject in front of the camera. At the later stages of development we will introduce multiple subject to test the robustness of our program.

The first test case being used is a relatively still subject sitting/ standing to determine location. These test cases will be used to determine if the program can determine where the subject is. These tests may be done in several ways. the first way we can render these scenes is by taking several pictures and just having the program reference the pictures. This method is not preferred because that means programming in a separate loader to process the image, rather than just a stream of images, which is what we'd like our final project to do. During this testing phase we want to use a variety of participants to ensure that the facial tracking method being used is useful and robust. This will ensure that when being used in practical settings there will be minimal failure.

The second test case that will be implemented will be a subject moving across the scene slowly and facing the camera. This will be used to test our vector algorithm's reliability and robustness. This testing case will also be the longest because it is the first main building block of our project. Just like our other test case this case will require many different subjects to ensure that the facial following algorithms are accurate and can perform in real time. We want the subject to move in many ways, back and forth and cross the center many times to make sure the program outputs the correct location of the subject and the correct vector strength at all locations. This is important because on startup the system will not necessarily be trained on the subject, so we want the system to be able to handle a subject walking into the frame or be at the extremes of the frame and have the program still pick them up and center then in its view.

The next training set will be a where the subject 's face might be obstructed or is not facing toward the camera. What we want to happen in this scenario is for the camera to try and track the user to a certain extent, because we also don't want it to freak out. To test this case we will have different subject walk across the scene, relatively slowly and sometimes stopping. We will also have them change their face's angle to the camera. This will test the robustness of the facial tracking algorithms and will help us determine the limitations of someone who is presenting while facing not directly at the camera. We also want to see what happens if for example there is a board and the subject is completely turned from the camera. We would like to make sure that the program can correct for this and that if the subject moves during this time, the program can catch them and follow them again.

The next case is going to be the second case but at a faster rate. We want to know is there is a speed that is unacceptable for our program. This is important because the end environment is going to be controlled and we can try to factor out tracing on object that move too quickly. This is also important because presenters have many styles of presenting and might move faster or slower around the room depending on many factors. Overall this case will simply to test limitations in our system rather than using the data to enhance our project. What may be significant is the possibility of the facial tracking algorithms breaking

down after a certain speed of the subject across the scene. This will be important because we can try to use other factor to track the subject if this occurs. Assuming of course that this speed of movement isn't a ridiculous rate for a classroom environment.

## 7.5 – PC Software Test Environment

The application will be run mainly through a simple intuitive graphic user interface. The application will be programmed using Microsoft Visual Studio 2013 in C# programming language or if not feasible then Java, which will be programmed using eclipse. During programming, a Windows 7, core i7 desktop will be used that has an array microphone and Logitech 1080p video camera plugged in via USB as shown in Figure 7.5 This will ensure that microphone and video inputs will work properly via USB with the software.

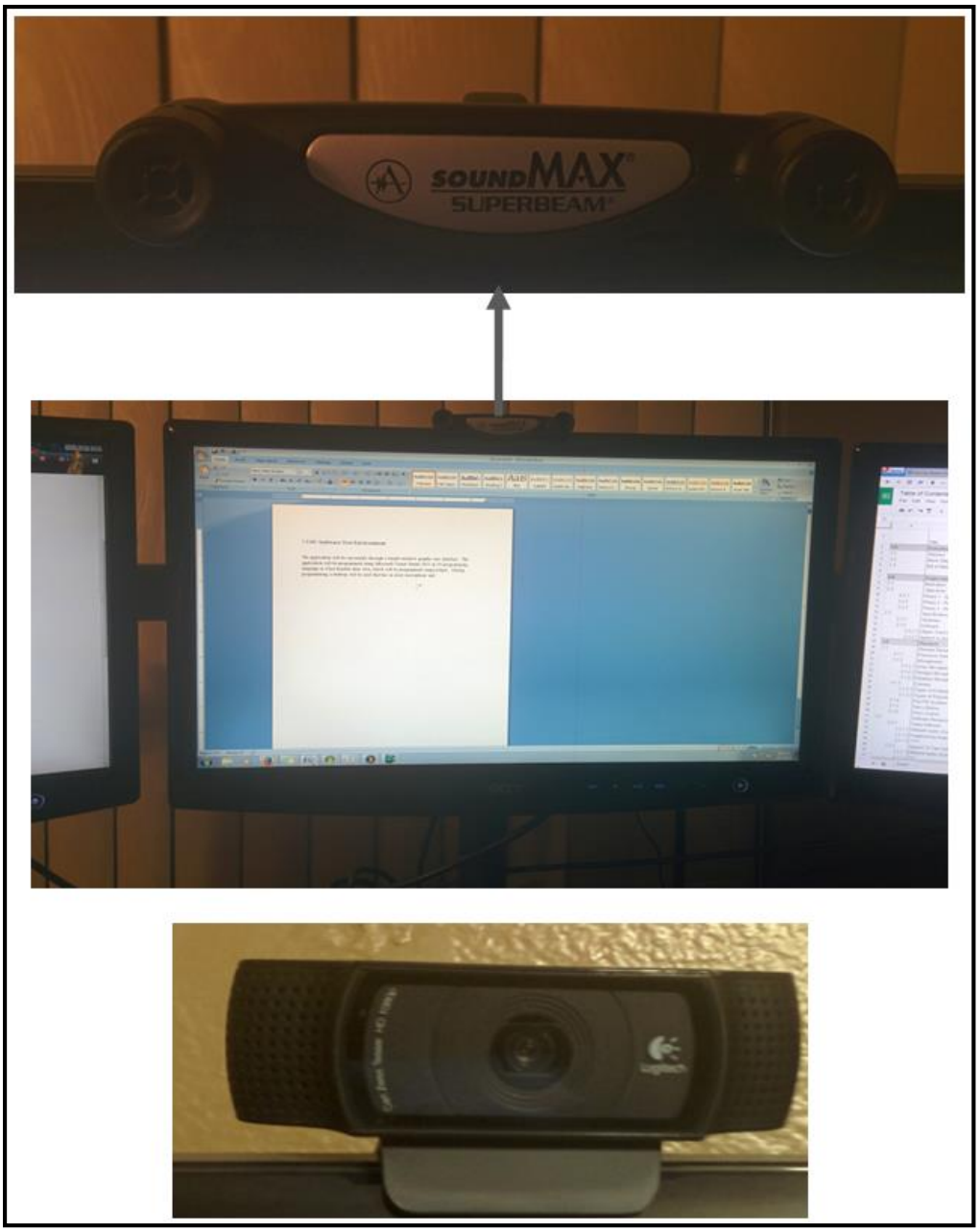


Figure 7.5: PC test environment

# 7.6 – PC Software Specific Testing

Due to the fact that the front end of this app will be a very simple graphic user interface. On the back end there will have to be extensive checks to make sure it is running correctly without any glitches. The first check on the front end is to ensure that the graphic user interface works as its suppose to, meaning all the buttons work and video feeds into the designated frame i.e. video 1 and video 2 as shows in Figure 7.6. Once development is completed the software will be deployed on a Dell M3800 laptop running Windows 8.1, which will fully utilize the built in touch screen capability. This will help simulate the

user's experience and help tremendously in finding out any bugs or glitches on the front end. There has to be system error catches for problems such as pusing the start or stop button numerous times. Ensure that once the start button is pressed it cannot be activated again until the stop button is pressed. The stop button should stop the recording, program and exit.

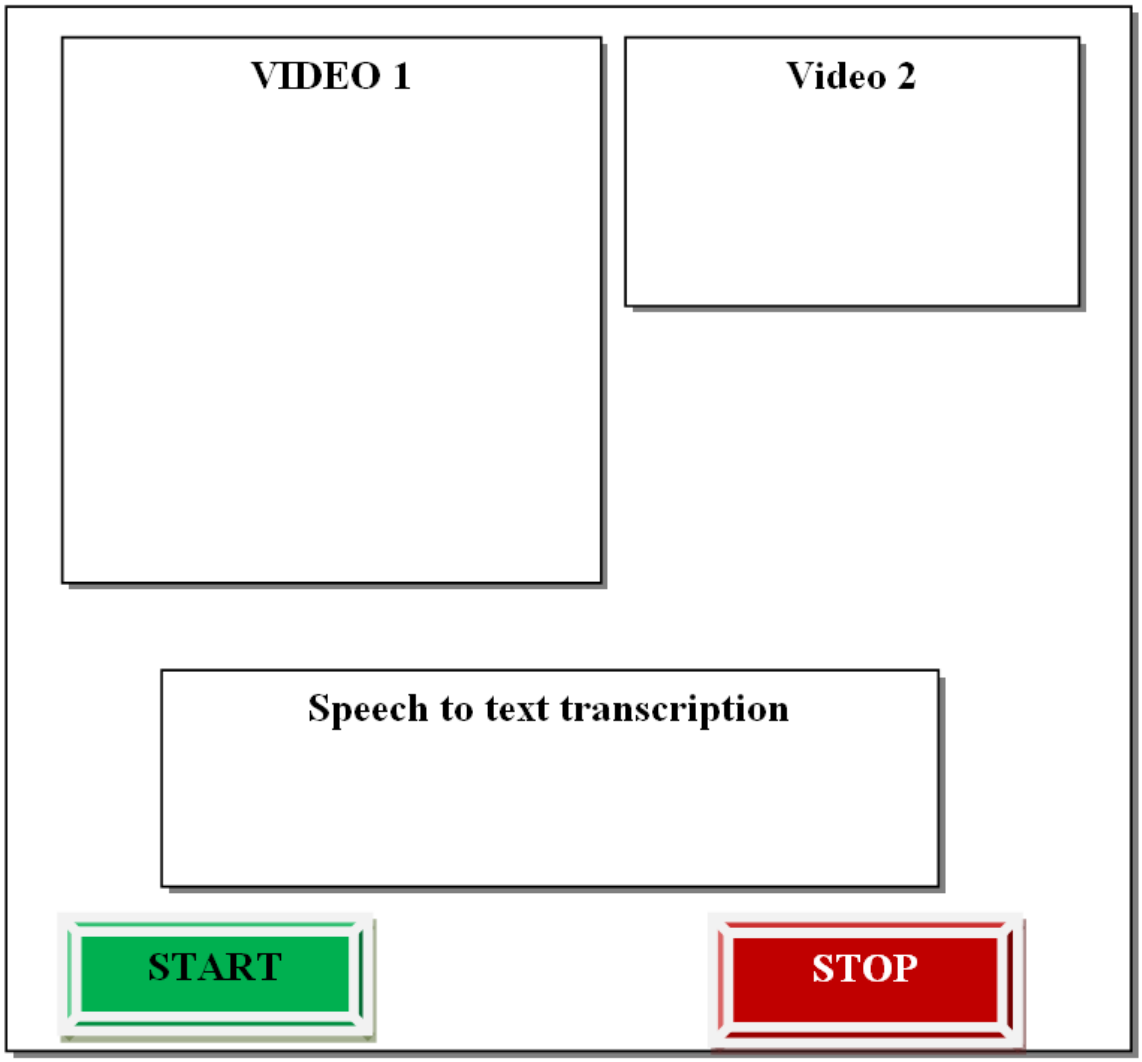



Figure 7.6 - Graphic User Interface testing

On the back end when a user presses the start button, there will be a full screen recording of video 1, 2 and the speech to text box which will be deposited to the video output folder as shows on figure 7.6.1. The speech to text will be saved to as a notepad document in the text output folder. The audio stream will be saved in the audio output folder. These folders will be checked to ensure that the respective files are being deposited and working when opened. Lastly depending on the cloud speech to text service that is chosen, I have to ensure the accuracy of the speech to text transcription and that the delay is not noticeable. Redundancies should be in place in case the speech to text doesn't work properly or

problems with accuracy during recording such as utilizing the video and adding subtitles to it after recording.

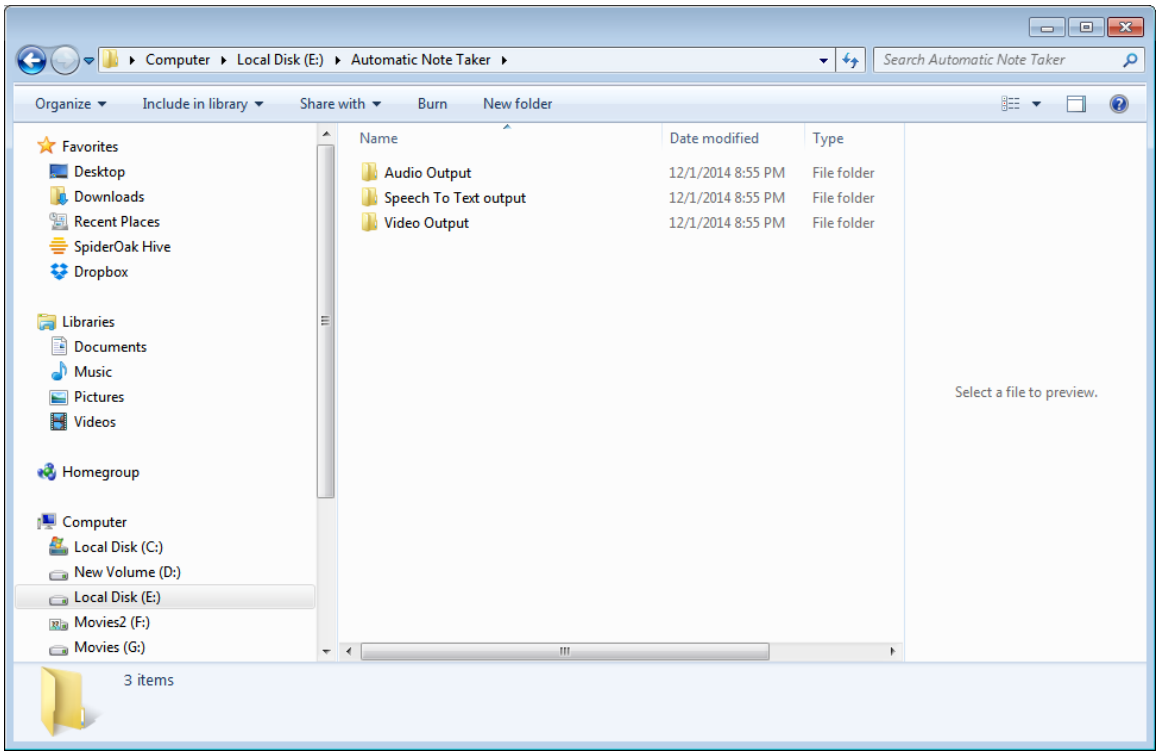


Figure 7.6.1- Folder Structure

# 8.0 – Administrative

## 8.1 – Milestones

### 8.1.1 – Senior Design I

The following tables represent the various milestones to occur within the life of this project. Each milestone includes a due date, a date of completion, and a brief description of the content to be completed. Table 8.1 (below) shows the milestones that are assigned to be completed by Senior Design I and are complete with the submission of this document.

| Milestone | Type | Description | Date Due | Date Completed |
|---|---|---|---|---|
| Organizational Software | Website | Set up Podio.com workspaces for future use | N/A or Unknown | 8/31/2014 |
| Initial Concept | Document | Initial design concept to be submitted to Dr. Richie | 9/9/2014 | 9/9/2014 |
| Final Document Outline | Document | Outline to be submitted to Dr. Richie. To serve as Table of Contents for Final Document | 10/9/2014 | 10/9/2014 |
| Update Table of Contents | Document | Update Table of Contents based upon Dr. Richie's input. | N/A or Unknown | 11/7/2014 |
| Final Document Draft | Document | Draft of the Final Document due on 12/4/2014. Outlined by the Table of Contents submitted 10/9/2014. Final Document is to be at least 120 pages. | 11/13/2014 | 11/13/2014 |
| Final Document | Document | No less than 120 pages. To be based on Final Document Draft submitted 11/13/2014 | 12/4/2014 | 12/4/2014 |
| Component Selection | Part Selection | Select parts to specifications described in Final Document | 12/4/2014 | 12/1/2014 |

Table 8.1 - Milestones for Senior Design I

## 8.1.2 – Senior Design II: Early Spring

Table 8.2 (below) represents the initial process to be completed early in Senior Design II. This table details the milestones associated with individual component acquisition and testing. Most milestones in Table 8.2 are expected to be reached early in the spring semester, while the PCB order and testing may fall in the later months.

| Milestone | Type | Description | Date Due | Date Completed |
|---|---|---|---|---|
| Order Microphone | Part Acquisition | Update Budget Worksheet | Early Spring 2015 | 1/25/2015 |
| Test Microphone | Hardware Testing | Testing environment and procedure documented in Final Document | Early Spring 2015 | 3/15/2015 |
| Order Embedded Camera | Part Acquisition | Update Budget Worksheet | Early Spring 2015 | 12/01/2014 |
| Test Embedded Camera | Hardware Testing | Testing environment and procedure documented in Final Document | Early Spring 2015 | 12/07/2014 |
| Order Pan/Tilt Base | Part Acquisition | Update Budget Worksheet | Early Spring 2015 | n/a |
| Test Pan/Tilt Base | Hardware Testing | Testing environment and procedure documented in Final Document | Early Spring 2015 | 3/15/2015 |
| Order Servo Motors | Part Acquisition | Update Budget Worksheet | Early Spring 2015 | n/a |
| Test Servo Motors | Hardware Testing | Testing environment and procedure documented in Final Document | Early Spring 2015 | 3/15/2015 |
| Order Processor | Part Acquisition | Update Budget Worksheet | Early Spring 2015 | 12/07/2014 |
| Download Open Source computer vision software | Software Acquisition | Select software to specifications described in Final Document | Early Spring 2015 | 1/12/2015 |
| Download Open Source dictation software | Software Acquisition | Select software to specifications described in Final Document | Early Spring 2015 | 1/20/2015 |
| Design PCB | Design | Design PCB with I/O and memory | Early Spring 2015 | 3/10/2015 |
| Order PCB | Part Acquisition | Update Budget Worksheet | Mid Spring 2015 | n/a |
| Test PCB | Hardware Testing | Testing environment and procedure documented in Final Document | Mid Spring 2015 | n/a |

Table 8.2 - Milestones for early Senior Design II: Component Acquisition and Testing

## 8.1.3 – Senior Design II: Late Spring

Table 8.3 (below) shows the milestones to be met by the latter part of the spring Senior Design II semester. The dates of completion for these step are dependent upon component acquisition and testing shown above in Table 8.2. Completing the milestones in the following table ensures that the project meets specifications entailed in this report. Further steps may be taken to improve functionality of the device, but are not required to meet the product's intended specifications and are listed in Table 8.4.

| Milestone | Type | Description | Date Due | Date Completed |
|---|---|---|---|---|
| Integrate Vision Software | Integration | Upload program to PCB | Late Spring 2015 | 3/25/2015 |
| Test embedded vision software | Software Testing | Testing environment and procedure documented in Final Document | Late Spring 2015 | 3/31/2015-4/03/2015 |
| Integrate Embedded Camera | Integration | Mount Camera to Pan/Tilt head and connect to PCB I/O and power | Late Spring 2015 | n/a |
| Integrate Servo Motors | Integration | Connect Servo Motors to PCB I/O and power | Late Spring 2015 | 3/25/2015 |
| Test and set movement thresholds for vison software | Software Design | Determine when the camera needs to be turned to keep presenter in view. | Late Spring 2015 | 3/31/2015-4/03/2015 |
| Test dictation software | Software Testing | Testing environment and procedure documented in Final Document | Late Spring 2015 | 3/31/2015-4/03/2015 |
| Integrate Microphone | Integration | Attach microphone to Pan/Tilt head. Previously unattached for use in testing dictation software. | Late Spring 2015 | 3/31/2015-4/03/2015 |
| Final Testing | Hardware Testing & Software Testing | Testing environment and procedure documented in Final Document | Project Completion date assigned by Dr. Richie | 3/31/2015-4/03/2015 |

Table 8.3 - Milestones for Senior Design II: Integration and Software Testing

## 8.1.4 – Senior Design II: Phase II & III

The optional milestones will that exceed the project's specifications are listed below in Table 8.4. These milestones are grouped into two "phases" that have overlapping hardware and software needs. Therefore, in the "Phase" column of Table 8.4 the phase associated with the milestone is listed. As these steps are optional, completion date is not relevant to the table. All milestones listed are modular and are intended not to interfere with meeting the specifications created for this project.

| Milestone | Type | Description | Phase | Date Completed |
|---|---|---|---|---|
| Order Webcam | Part Acquisition | Update Budget Worksheet | Optional - Phase II & III | 1/20/2015 |
| Test Webcam | Hardware Testing | Testing environment and procedure documented in Final Document | Optional - Phase II & III | 3/31/2015-4/03/2015 |
| Download Open Source captioning software | Software Acquisition | Select software to specifications described in Final Document | Optional - Phase II & III | n/a |
| Test captioning software | Software Testing | Testing environment and procedure documented in Final Document | Optional - Phase II & III | n/a |
| Stream video from PCB to Laptop PC | Software Design | Export frames captured by embedded camera to Laptop PC | Optional - Phase III | n/a |
| Overlay video stream from PCB | Software Design | Overlay frames from PCB onto video stream. | Optional - Phase III | n/a |

Table 8.4 - Optional milestones for Phases II & III

# 8.2 – Budget

| Category | Type/Subcomponent | Product Name | Estimated Quantity | Estimated Price | Estimated Total | Purchase Quantity | Purchase Price | Purchase Total |
|---|---|---|---|---|---|---|---|---|
| Circuit board: | | | | | | | | |
| | PCB | | 1 | $30.00 | $30.00 | - | - | - |
| | Processor | | 1 | $15.00 | $15.00 | - | - | - |
| | Memory Component | | 1 | $10.00 | $10.00 | - | - | - |
| | Cables | | 2 | $20.00 | $40.00 | - | - | - |
| | Wire | | 4 | $2.50 | $10.00 | 1 | $8.00 | $8.00 |
| | Other | | 1 | $50.00 | $50.00 | - | - | - |
| | BeagleBone Black | | - | - | - | 1 | $55.00 | $55.00 |
| | Expansion HeadeR | | - | - | - | 1 | $9.95 | $9.95 |
| | Voltage Regulator | | - | - | - | 1 | $14.99 | $14.99 |
| | | | | | | | | |
| Pan/Tilt: | | | | | | | | |
| | Pan Base | | 1 | $25.00 | $25.00 | 1 | $39.99 | $39.99 |
| | Tilt Head | | - | - | - | 1 | $24.99 | $24.99 |
| | Servo Motor | | 2 | $9.00 | $18.00 | 2 | $7.99 | $15.98 |
| | | | | | | 1 | $14.99 | $14.99 |
| Input devices: | | | | | | | | |
| | Microphone | | 1 | $55.00 | $55.00 | 1 | $53.00 | $53.00 |
| | Embedded camera | | 1 | $55.00 | $55.00 | - | - | - |
| | Webcam | | 1 | $30.00 | $30.00 | 2 | $79.99 | $159.98 |
| | | | | | | | | |
| Shipping: | | | | | | | | |
| | Shipping Total | | 1 | $50.00 | $50.00 | 1 | $50.00 | $50.00 |
| | | | | | | | | |
| | | | | | $388.00 | | | $467.73 |

Table 8.5 - Project Budget

## 8.3 – Financial Contributions

### 8.3.1 – Personal Contributions

$104.27 – Casey Miville – 10/28/14 - Set aside for project.
$79.99 – Roland Anderson – Acquire Webcam
$134.00 – Patrick Galloway – 12/07/2014 - Acquire one webcam and Beagle Bone Black
$149.47 – Casey Miville – 3/15/2015 – Acquire base, servos, and voltage Regulator

# 9.0 – Impact Of Project

Our group has designed our project to have a positive economic impact for the university system. Since our devices will ultimately save Universities money that they would spend on students. The project will also have an impact on the disabled community. Our objective with this project is to help the disabled community to perform on a competitive level compared to that of the general population. Our project will do this by giving this community the best possible information available. Our group hopes that this will have a positive social impact and help those who might not have received degrees or thrived in a college setting to succeed equally to all counterparts. We also expect to be able to use this project to help any student who wishes to study notes and materials, get an accurate representation of the material of the class/ lecture that they attended.

# 10.0 – Standards

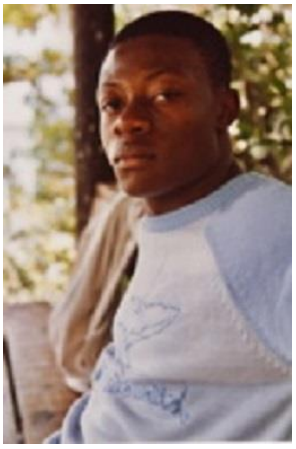
Systems and software engineering standard; ISO 12207
Barrel jack; IEC 60130-10
Beagle Bone Black: IEC 60130-10, EN 61000- 1:2006
Logitech Webcams: EN 55022:2010 +AC:2011 , EN 55024:2010
Voltage regulator ISO 1043 PA46-GF30

# 11.0 – Operation

- To operate our device, the user needs to have the ANTI Device and a copy of the ANTI Software exe on their laptop.
- The user then needs to power the ANTI device by plugging in its power adaptor to the wall, and the baler jack into the female receptor.
- The user must then plug in the microphone (if available) to the laptop.

- The user will then start the ANTI Software
- Once the software has started the user will then select their device
    - Go to the Device tab
        - Select the appropriate webcam
        - Select the appropriate Microphone
- After the device is selected the user may go again to the device tab to preview the scene
- The user can then press start to start recording the scene
- Then user can then press Stop to stop recording or press Exit to stop recording
    - The recording will be saved into a specific file location
    - The recording will use a time and date stamp to uniquely identify it
- To operate the ANTI Device just plug the USB cable from the Laptop to the ANTI Device
    - The Device should move automatically

# 10.0 – Personnel



Roland Anderson, currently a senior at University of Central Florida will receive his Bachelors' of Science in Computer Engineering in May of 2015. He plans on continuing on to the Master's program in Computer Engineering with a focus on Intelligent System and Machine Learning. He is currently a Software Developer Level 1 for Wyle Laboratories.

Patrick Galloway is a computer engineering student planning on graduating May 2015. Patrick plans on working for a year, currently in IT at UCF, and looking for programming work. Then applying for graduate school for either machine learning, or computer networks and security.

Casey Miville, currently a senior at University of Central Florida will receive his Bachelors' of Science in Electrical Engineering in August of 2015. He will enter the workforce with hope of obtaining an engineering position with an aerospace contractor. He also plans on continuing his education to obtain his Ph. D, become a professor, and run a research lab.

# Appendix A – Cited Sources


1. Microsoft .com - “Microphone Array” - http://research.microsoft.com/en-us/projects/Microphone_Array

2. “Shotgun Microphones” - http://www.bhphotovideo.com/explora/audio/buying-guides/shotgun-microphones

3. "Acoustic Modeling." *Research.microsoft.com*. N.p., n.d. Web. 17 Nov. 2014. <http://research.microsoft.com/en-us/projects/acoustic-modeling/>.

4. Brokish, Chuck. *U-Law Compression on the TMS320C54x*. N.p.: n.p., n.d. *Ti*. Web. 20 Nov. 2014. <www-s.ti.com/sc/psheets/spra267/spra267.pdf>.

5. Chu, Wai C. *Speech Coding Algorithms: Foundation and Evolution of Standardized Coders*. Hoboken, NJ: J. Wiley, 2003. Print.

6. Koehn, Phillip. *Statistical Machine Translation*. Cambridge: Cambridge UP, n.d. Print.

7. Malcolm Slaney, Elizabeth Shriberg, and Jui-Ting Huang, Pitch-Gesture Modeling Using Subband Autocorrelation Change Detection, in *Proceedings of Interspeech 2013*, ISCA, 29 August 2013

8. George Dahl, Dong Yu, Li Deng, and Alex Acero, Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition, in *IEEE Transactions on Audio, Speech, and Language Processing (receiving 2013 IEEE SPS Best Paper Award)* , vol. 20, no. 1, pp. 30-42, January 2012

9. Frank Seide, Gang Li, and Dong Yu, Conversational Speech Transcription Using Context-Dependent Deep Neural Networks, in *Interspeech 2011*, International Speech Communication Association, August 2011