So-Lo
Final Project Review Report

Andy Weng (CSE), Dan-Michael Tiamzon (EE), Suzet Nkwaya (EE), Ming Shuai Chen (CSE)

[1]*Abstract*— **So-Lo (Sound Locator) is a real-time automated system used for recording audio and video meetings or conferences in small to medium sized rooms. The video can be easily accessed by users who want to recap the meeting or for those who missed the meeting. So-Lo uses Time Difference of Arrival (TDOA) to locate the speaker, allowing the system to focus on the former while recording the video.**

I. INTRODUCTION

Meetings occur every day in which people gather and exchange essential information. Research found that the top five reasons why groups meet are: reconcile conflict, reach a group judgment or decision, solve a problem, ensure that everyone understands a specific topic, and facilitate staff communication [4]. Research has shown that most companies spend between seven and fifteen percent of their personnel budget on meetings [6]**.** Given the resources that are put into organizing meetings, it is imperative that all participants stay focused and remember the maximum amount of information discussed. In fact, according to a 3M study, "Unproductive meetings may cost organizations more than wasted dollars; time may be lost, morale may decline, and productivity may be reduced**"**[5].

The core takeaway from a meeting is information. The most common way information is recorded is through writing. Writing down information has its own problems such as "it may lose accuracy since the minute preparer may not remember or interpret correctly or is biased"[7]. Video and audio recording devices have been beneficial to keep information correct and be accessed easily. There are products on the market for this function however, most of the affordable options are either manually controlled, offer only audio, or a stationary camera. The high-end products are incredibly expensive. For instance, Polycom "CX5500 Unified Conference Station" costs around $5000 [10]. Our project, So-Lo, proposes a solution to this problem which delivers quality audio and video, easily accessible recordings, and is inexpensive.

So-Lo is a device that records information in video and audio format from a conference. So-Lo uses the Time Difference of Arrival (TDOA) technique to determine the location of the person speaking, and then focus the camera on the person. Three microphones and a Raspberry Pi are used to implement TDOA in So-Lo. Once the location of the speaker is determined, a motor equipped with a camera will point in the direction of the speaker. This will give a clear and focused recording of the person speaking so that playback will provide good quality and focused recording of the person speaking. The scope and implementation of So-Lo can go beyond the conference rooms. It can be used for security purposes. For example, the device can be installed in a store. During after hours, if someone were to break in, they would make a loud noise, causing the camera to face the intruder.

After extensive research and consulting with our advisor and other professors we have laid out our system requirements as shown in Table 1.

Specifications	Goal	Actual
Range	3ft	>3ft
Response Time	<1s	<1s
Cost	<$200	$100
Senitivity	Detect Sharp Sounds	Detect Voices
Frequency Range	300Hz-3000Hz	272Hz- 3187Hz
Video Resolution	1080p	720p
Video File	.mp4	.mkv

Table 1. System Requirements

II. DESIGN

Overview: So-Lo is designed to work in quiet, small to mid sized meeting rooms; where it will be assumed that only one person is talking.

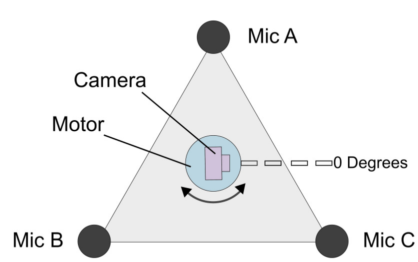

Figure 1.

[1]A. Weng majors in Computer Systems Engineering.
D. M. Tiamzon majors in Electrical Engineering
M. S. Chen majors in Computer Systems Engineering
S. Nkwaya majors in Electrical Engineering

So-Lo uses TDOA technique to obtain the sound source location. Compared to other sounds localization techniques, TDOA is relatively easy and cheap to implement. Some of the other methods of obtaining either sound source direction or sound source location include particle velocity or intensity vector and triangulation. For instance, triangulation requires to know the angle at which the sound is arriving at the microphone

To implement TDOA, So-Lo uses an array of three omnidirectional electret microphones to detect various sounds: in this case the project focuses on detecting human voice. The microphones will be placed one foot away from each other, forming an equilateral triangle as shown in Figure 1. When a person speaks the sound will travel and get to the 3 microphones at different times. This time difference will then be captured, and used by a custom software stored in a Raspberry Pi to determine the order in which the microphones detected the sound and use this data to calculate approximately the angle from which the sound came. The angle obtained is then used to point the camera in the speaker direction via a DC motor.

Block Diagram

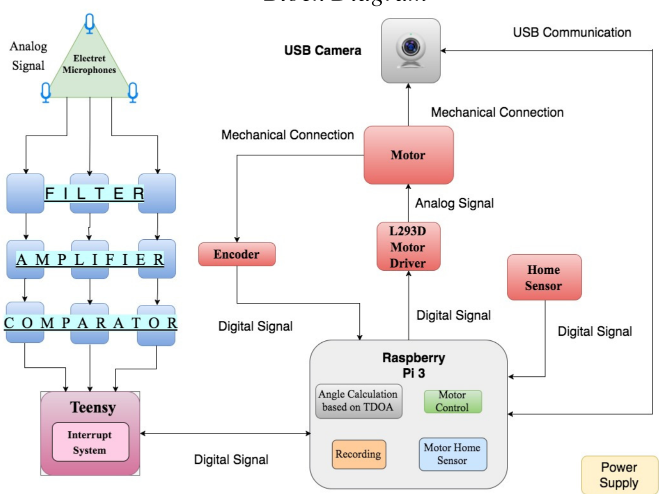

Figure 2. Block Diagram

A. Electret Microphone

The loudness of the sound depends on the distance between the speaker and the microphone. This means that in some cases one or two of the microphones may pick up the presence of a sound because they are relatively far from the speaker, and this will cause the time difference and the angle calculations to be wrong. In order to fix the problem of some microphones not picking up the sound, we used new microphones whose pre-amp is equipped with an automatic gain control (AGC). Using the AGC in the amplifier means that nearby 'loud' sounds will be quieted so they don't overwhelm & 'clip' the amplifier, and even quite, far-away sounds will be amplified, therefore, the chances of a microphone not picking up the sound will be reduced. This will improve the accuracy of the angle calculation.The schematic of the electret microphone and the preamp circuit can be seen on Figure 3. The chip at the heart of this amp is the MAX9814, and has a few configurable options. The default 'max gain' is 60dB, but can be set to 40dB or 50dB by connecting the Gain pin to VCC or ground

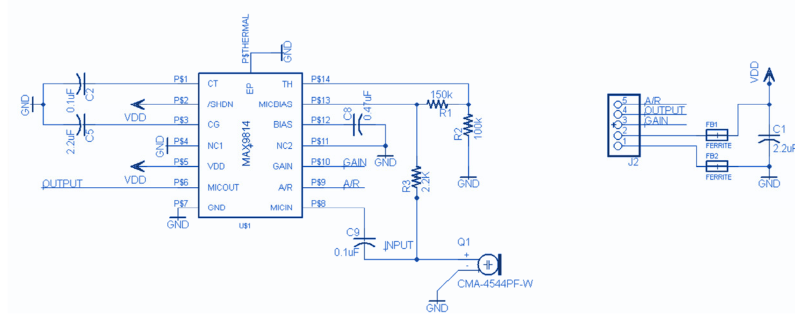

Figure 3. Adafruit AGC Electret-Microphone Amplifier

B. Filter

The sound detected by the electret microphones goes through different stages of analog signal processing before being used by the Raspberry Pi. The first step was the filtration of the signal. Figure 3 shows the filtering circuit used in So-Lo. The filter's goal was to reduce the amount of noise and focus on the human voice, the sound first goes through a lowpass filter that has a cutoff frequency of 3000 Hz. Then the signal gets passed through a highpass filter with cutoff frequency of 300Hz. The combination of the high-pass and low-pass filter makes a 300Hz-3000Hz bandpass filter, which is good enough to cover the human voice frequency range.

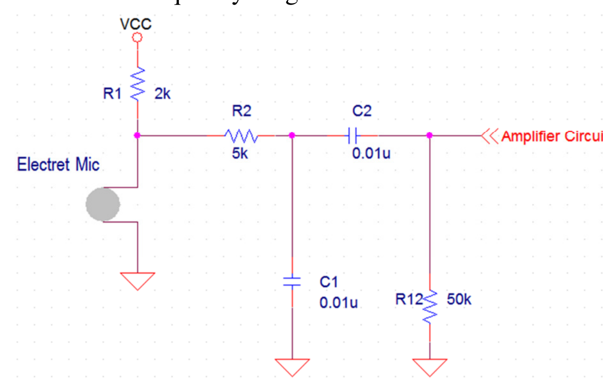

Figure 4. Filter circuit

B. Amplifier

Because the signal coming from the microphone was very small (a few millivolts), it had to be amplified in order to make the processing easier to be used as the input for the rest of the system. The amplification was done by the TL074CN chip, which contains four independent op-amps. So-Lo has two amplification stages (see Figure 5 below). The first stage has a 15V/V, the second stage has a gain of 2V/V. In addition to increasing the overall gain of the circuit, the second stage give s the ability to tune the amplification of the system. When the electret microphone are not connected to an amplifier, the output was in order of millivolts. The team tested the amplification of the signal coming from the microphone by speaking or playing music then recording the value of the voltage at the output of the amplifier. The amplification stage allowed the output to go up to 15 V. Because no two components have the same exact properties, even if the same circuit is replicated for all three microphones, they may end up having different gains. Having the ability to tune the gain makes it possible to match them. After the Amplification stages, the signal goes through another high-pass filter to remove the DC offset.

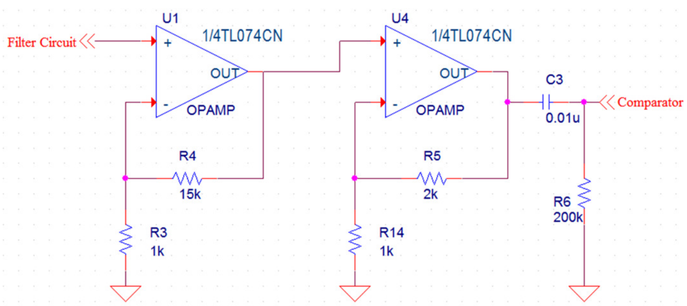

Figure 5. Amplifier Circuit

C. Comparator

Originally, after filtering and amplifying the signal, our team planned to use an analog to digital converter (ADC). However after testing, it was concluded that the ADC was not ideal for this project. The problem with ADCs is that to be able to convert a sound signal, they need a high sampling rate, and their implementation can be challenging. That's why the ADC was replaced with an LM311N comparator (see Figure 6 for comparator circuit). The comparator was set up to output 0V when there is no sound and 15V when a sound is detected (when someone speaks). In order to achieve this, the comparator took in the amplified sound signal as an input and compared it to a reference voltage that corresponded to the voltage created by the ambient noise. During lab testing, when there was just ambient noise in the vicinity of microphone, the output of the comparator was 0 V; however, when a person spoke or clapped, the output of the comparator went from 0V to about 15 V, this change was displayed on an oscilloscope. Since all meeting rooms would have different settings, and therefore different ambient noise levels, the reference voltage can be set to a desired value using a potentiometer. The comparator's 15V output went through a voltage divider before going to the GPIO of the Raspberry Pi because the latter's GPIO pins has a voltage rating of 3.3V. This signal was then used by the Raspberry Pi to trigger an interrupt system.

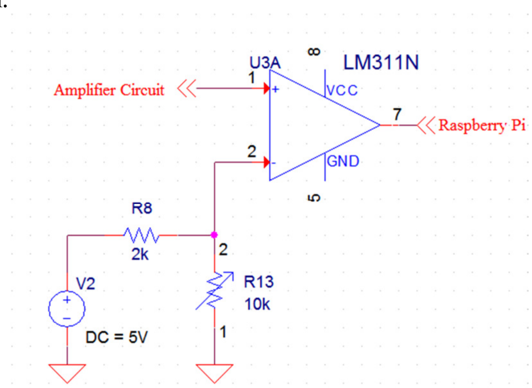

Figure 6. Comparator Circuit

D. Interrupt System

The system was required to pick up sounds from three microphones. Since the sound could originate from anywhere, the system must know which microphone received the signal first, second, and third. Previously the interrupt system was implemented on the Raspberry Pi but due to some issues, we implemented the system on a Teensy instead. The Interrupt System was designed similarly as if it was on the Raspberry Pi in which the microphones are connected to 3 I/O pins. The pins were set up at logic low and receive a signal from the microphone array circuit. When a sound was picked up by the microphones, they produced a high voltage, which which triggered the interrupts.

The instant a sound was picked up by the microphones, three signals would be sent to the I/O channels. The channel that received a signal first will have its time of arrival marked. The same procedure will apply to the remaining two channels that are waiting for an interrupt. Once all times and order of microphones were recorded, the system the time difference between the 1st and 2nd microphone and the 1st and 3rd microphone were calculated. The system checked if these time differences are within reasonable values in terms values. We implemented error trapping code to make sure we throw out any unreasonable time differences that could be caused by random noise like echoes. If unreasonable time differences were detected, then nothing was sent to the Pi and the microphones started listening again until reasonable time differences were obtained.

To transmit the data to the TDOA calculation on the Raspberry Pi, we used Serial communication. The Teensy was connected directly to the Raspberry Pi via USB cable.

In order to assure accurate time measurements, the Teensy and the Raspberry Pi were tested with 3 generated test signals, shown in Figure 7, with known time differences in between them. These signals were fed into the Raspberry Pi then the Teensy and the results shown in Figure 8 illustrate incorrect times given by the Raspberry Pi. The incorrect time measurements have no consistency either which shows that the interrupt system was unreliable on the Raspberry Pi. Figure 9 illustrate the time measurements on the Teensy which show the correct time difference between the first rising edge and the second rising edge of the signal.

Figure 7. Generated test signals with a 50ms delay in between.

Although the Raspberry Pi 3 Model B has a CPU speed of 1.2GHz and the Teensy 3.5 only has 120MHz, the Raspberry Pi added latency to the time stamping of the pins in which the time the signal arrived at the pin is captured with some delay. The precision of these time differences is essential to be accurate to the 10th of a millisecond. The Raspberry Pi produced latency due to its Operating System running in the background. The Teensy was a perfect substitute as it had no

Operating System and delivered the accurate time measurements.

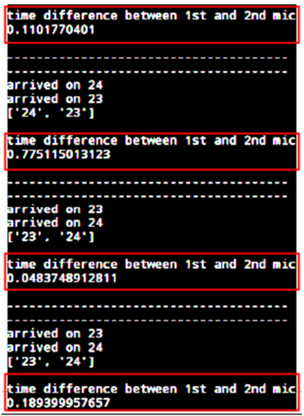

Figure 8. Raspberry Pi printed time measurements based on generated test signals.

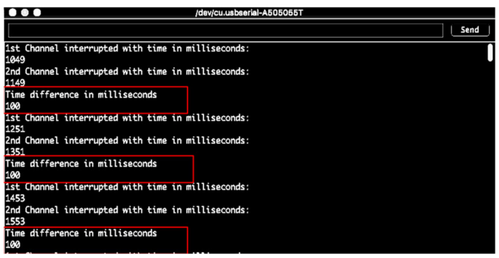

Figure 9. Teensy printed time measurements based on generated test signals

.

E. Angle Calculations

The angle calculation portion of this project is very essential to the overall system. It is purely software based and it takes in an output from the interrupt subsystem using nonlinear calculation methods and finally the pythagorean theorem. It outputs an angle which directs the motor.

The subsystem works by utilizing the Time Difference of Arrival (TDOA) technique which is also used in the interrupt subsystem. Because there are three microphones placed a certain distance away from each other in a triangle formation, there will be a time delay for when each of the mic picks up the source of sound. Each microphone will yield an equation of a circle which all have different center points and radiuses. Looking at Figure 6. Concept of TDOA, there will be three nonlinear equations that all have three unknowns, (x0, y0, r). Where x0 and y0 is the position on the coordinate plane and r is the distance to the closest microphone, but we don't use r.

Sequentially, after the (x0,y0) position of the source of sound is known, the angle can be calculated. Figure 7. Example Calculations of TDOA demonstrates how the angle is calculated. First, an imaginary coordinate plane is drawn with the origin centered at the center of the equilateral triangle above. Using offsets, the angle can be calculated using the Pythagorean theorem.

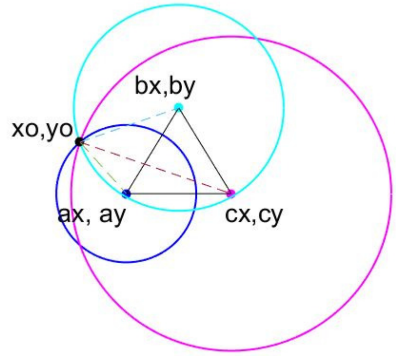

Figure 10. Concept of TDOA

There are a few things that should be noted here. The position of the center of the triangle is at (1012, 1009.46). The units are in inches and the imaginary coordinate plane centered at the triangle is only used for angle calculations. The main graph itself is at the upper right hand first quadrant because the graph starts at (1000,1000). The reason for that is so negative numbers don't come into the equation. Figure 8. TDOA Equations are the three main equations used to solve this problem.

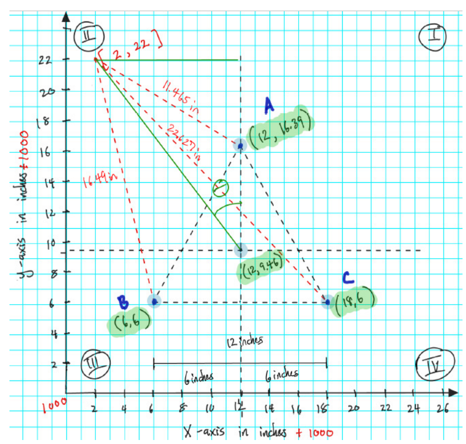

Figure 11. TDOA: Example Calculations

$$(a_x - x_o)^2 + (a_y - y_o)^2 = r^2$$
$$(b_x - x_o)^2 + (b_y - y_o)^2 = (r + 346 b_s)^2$$
$$(c_x - x_o)^2 + (c_y - y_o)^2 = (r + 346 c_s)^2$$

346m/s => 13622 inches/sec

Figure 12. TDOA Equations

Looking at Figure 8, the green colored variables are what is known, (ax,by) is the position of microphone A, and so on. The variable "bs" is the difference in time it takes the source of sound to get to microphone B and A and the variable "cs" is the difference in time it takes the source of sound to get to microphone A and C. Note that "bs" and "cs" is not always going to be related to microphone B and microphone C. These variables can change depending on the order of microphone which received the source of sound first, (bs comes before cs always). The two variables are the outputs from the interrupt subsystem and is the only input for this subsystem required to make an angle calculation output for the motor. The red colored variables are the three unknowns that need to be calculated. (x0,y0) is the position of the source of sound.

The technology used to implement the calculations is the Python language which is flexible because it can run on the Raspberry Pi 3 OS and PC/MAC OS. Moreover, to solve a system of linear equations, several Python packages had to be installed such as numpy and scipy. These modules can accurately solve systems of nonlinear equations. It can be ran on both Python 2 and 3.

A test implementation was conducted during MDR presentation for this subsystem. The code in Python asks for user input of the source of sound (x0,y0). The code then outputs the calculated distances from the source of sound to each of the three microphones as well as the time it takes to reach the three microphones. The code then outputs the (x0, y0) which mentioned above is the source of the sound. This position should be precisely the same as what the user inputted, which makes the check successful. The program then outputs the angle that the motor should turn. This angle assumes that the imaginary x-axis centered on the triangle is the 0º mark. If you look at Figure 7, the user input is (2,22). The program would output about 128º (90º from first quadrant and 38º from second quadrant).

The results of these tests show that under ideal circumstances and assuming that the speed of sound at room temperature is 346m/s, the calculations based on the ideal bs and cs will give the correct user position (source of sound) and it will output the correct angle. [11]

After we tested that the calculations work under ideal time differences, we connected the angle output to the L293D motor driver chip to direct the motor and USB webcam. This subsystem keeps track of current position of the motor using the H21A1 Photo-interrupter as well as having a current position stored within the Pi. Because of issues with the slip ring motor, the subsystem is smart enough to not have the motor rotate more than 360 degrees. This is done by never having the motor turning pass the dotted x-axis between Microphone A and Microphone B in Figure 11.

F. Motor

The purpose of the motor is to direct the camera towards the direction of the speaker. This part of the system used a DC brush motor, a motor driver, a photo-interrupter, and an encoder. The motor that was used for So-Lo is the DC Maxon Motor 310005 [3]. The motor driver that was used was the L293D motor driver chip. The photo-interrupter that was used was the H21A1 photo-interrupter.The encoder that was used was the E5 Optical Encoder. By itself, the motor is only capable of spinning in one direction. This was a slight problem in the functionality of our device because there can be a shorter path to a certain angle if the motor spins in the other direction. For example, if the motor is facing 0º and someone speaks at 90º, the motor would turn clockwise 270º to face that angle. To address this problem, the L293D motor driver chip was used.

The L293D chip was used to control the direction the motor turns by changing the direction of the current through the motor. Its secondary purpose was to supply power to the motor since the Raspberry Pi is incapable of supplying enough power to the motor directly. [2]

The H21A1 photo-interrupter was used to determine a home position for the device. This was necessary because if the device was off and someone were to tamper with the direction of the motor, then the motor will have a new initial position at startup. This would cause the motor to point to incorrect angles. A home position would help remedy this problem because it would keep the initial starting point of the system the same each time it starts up. A picture of the H21A1 can be seen in Figure 10. An optical transmitter on the side labeled "E" transmits a light to the optical receiver on the side labeled "D". When an object passes through the gap, it blocks the transmitted light, causing the optical receiver to change from a high to a low signal. By putting the photo-interrupter at our desired initial starting point, we could accurately "reset" the motor to this position by attaching an object onto the motor that will block the transmitted light. [12]

Figure 10. H21A1 Photo-interrupter

In order for the motor to accurately determine the direction it is facing after the initial starting point, an encoder was used. The encoder of choice was the E5 Optical Encoder. The encoder contains a disk that is wrapped around a shaft. As the motor spins, the shaft spins which causes the disk on the encoder to spin. There are opaque and transparent regions on the disk. A light shines through the disk and into a sensor on the other side. If the sensor senses the light shining on it, pin A on the encoder outputs a 1. As the disk spins, the region that the light shines through alternates between opaque and transparent. The result is a series of pulses from channel A and B. A diagram of the basics of an encoder is shown on Figure 9. Both channels together are needed to determine the direction the motor is spinning but since we are already able to control the direction the motor spins, only one channel is required. By counting the number of pulses, the motor can determine the direction it is facing relative to its initial position. When the motor makes one revolution, the encoder outputs ~18000 pulses. To turn the motor to 90º, just turn the motor counter-clockwise until 4500 pulses are counted since 90º is a quarter of 360º. In addition, since a slip ring was not used, the camera's wire could get tangled. This was accounted for by making sure the motor never turns a full 360°. This was implemented in the software in the Raspberry Pi. [1]

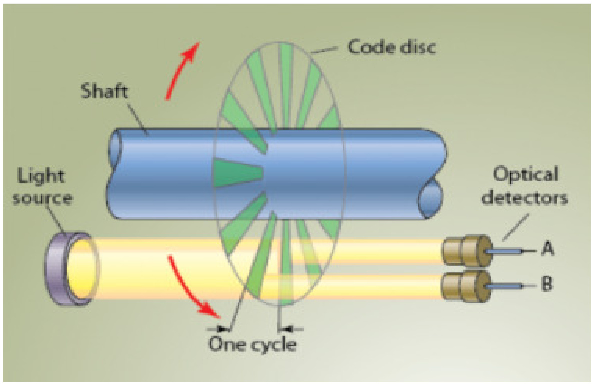

Figure 9. Fundamentals of an optical encoder

The motor driver and encoder was programmed using Python code on the Raspberry Pi. The techniques to programming on the Raspberry Pi to control the motor driver and encoder was similar to what we have learned in ECE 353 and ECE 354, Computer Systems Engineering I & II. In those courses, we gained experience with embedded systems and how to program an FPGA. The Raspberry Pi was significantly different from an FPGA, however. Therefore, we had to learn the layout of the functionality of the Raspberry Pi to build this block.

To test the motor functionality, we programmed the motor to accept a series of angles that will come from the angle calculation block. The motor's initial position is 0º. We input the angles 90º, 300º, 210º, and 40º. This tested if the motor could turn to a certain angle, turn clockwise and counter-wise, and verify the camera's wire never gets tangled.. The motor successfully passed the test.

G. Camera

The camera that was used for So-Lo was the TeckNet C016, however, any USB camera would be compatible. When the device starts up and resets to the home position, the camera will begin recording. It is capable of recording video and audio with its in-line microphone. While the camera was capable of recording videos at 720p, video processing at that resolution with the Raspberry Pi proved to be far too CPU intensive while working in conjunction with the angle calculation and motor code. Therefore we opted to reduce the video recording to 480p. Audio had to be recorded separately. However, we synced up the audio and video and combined it into one video. The videos were saved as .mkv files since that video file works best for the Raspberry Pi. When the video finished processing, it was uploaded to the user's Dropbox account whose credentials had to be hard coded. To connect to Dropbox, the Raspberry Pi had to be connected to a permanent Wi-Fi network. We connected our Raspberry Pi to eduroam which required the user's credentials to be hard coded in. All the code (interrupt system, angle calculations, motor, video, audio, Dropbox script) runs automatically as soon as the Raspberry Pi boots up which requires two power outlets.

H. Power Supply

The entire system used a triple output power supply and a power supply for the Raspberry Pi which are both connected to a wall socket. As shown in Figure 10, the triple output power supply provides -15VDC, +5VDC, and 15VDC. The comparators and amplifiers were powered with -15VDC and +15VDC and components such as microphones, motor driver, encoder, home sensor, and motor are powered with 5VDC.

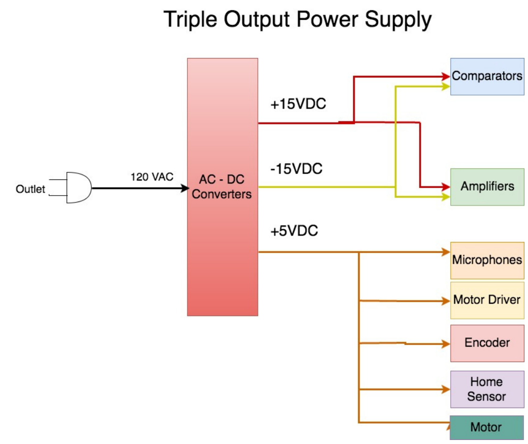

Figure 10.

III. PROJECT MANAGEMENT

The project had four major subsystems, distributing one subsystem to each team member. S. Nkwaya designed the microphone array circuit, D. Tiamzon designed the interrupt system, M. Chen designed the angle calculation software, and A. Weng programmed the motor.

The team succeeded in completing our MDR deliverables. Table 2 provides the deliverables proposed for MDR and the results. The main purpose of our MDR Deliverables was to demonstrate the concept of TDOA.

Each subsystem was distributed to suit the expertise and interests of each member. S. Nkwaya is an electrical engineering major whose forte is signal processing and therefore most comfortable with working on the microphone subsystem. D. Tiamzon is also an electrical engineering major who is interested in interfacing with hardware and software; so the interrupt system best suited his interests. M. Chen is a computer systems engineering major who likes programming so the angle calculation was best suited for him. A. Weng is also a computer system engineering major who likes in programming in the Raspberry Pi so the motor subsystem was best suited for him.

Despite our expertise in our individual parts, there were unavoidable challenges where the team needed to help each other out. Anytime any of us was stuck on a problem, we would ask the group for advice on a solution or workaround or even tackle the problem together.

Our team met on a weekly basis with our advisor to present updates and concerns. From September to April, we would all be present in the lab most days of the week. We often found ourselves working individually on each of our subsystems but at least twice a week we would all be present at the lab to collaborate.

IV. Conclusion

The system is fully integrated, and functional. Due to the sensitivity of the microphones, So-Lo works best in a quiet environment in which one person speaks at a time. So-Lo accurately points to the the source of a sound with errors of less than 1 degree. So-Lo has a time delay between each calculation meaning that when a person speaks and after the motor turns to them, it will wait a certain amount of time until it starts to listen again. For this project we set the time delay to be five seconds. This time delay can be changed to adjust to a specific situation. The motor was implemented at a specific RPM to accommodate pointing to the angle precisely.
The camera is able to record a video while the system sound locating. 30 seconds of video is recorded while speakers talk and the system points to them. The video was set up to be automatically uploaded to Dropbox. Our system also successfully recorded an hour long video footage which is the appropriate time frame for meetings. So-Lo was able to run for 4 hours continuously as during demo day, So-Lo was on for the entire duration without issues.

Acknowledgment

We would like to thank Professor Polizzi and Professor Ciesielski for their feedback that helped us improve our project. We would also like to thank Professor Soules who took the time to meet with us each week, and helped us stay on track and set our goals high and Professor Hollot for his support. We would like to thank Fran for his valuable services especially during busy times.

References

[1] US Digital, “E5 Optical Kit Encoder” E5 datasheet [Accessed 20-Dec- 2016].
[2] Texas Instruments, “L293x Quadruple Half-H Drivers” L293D datasheet September 1986 – Revised January 2016
[3] Maxon Motors, “30mm, Graphite Brushes, 60 Watt” Maxon DC Motor 310005 datasheet May 2008
[4] Monge, P.R., McSween, C., & Wyer, J. A profile of meetings in corporate America: results of the 3M meeting effectiveness study, Annenberg school of communications, University of Southern California, Los Angeles, CA, 1989.
[5] 3M Meeting Management Team, and Drew, J. Mastering meetings : discovering the hidden potential of effective business meetings. New York: McGraw-Hill, 1994.
[6] Doyle, M., & Straus, D. How to make meeting work: the new interaction method. New York, NY: Jove Books, 1982
[7] Alex Waibel, Michael Bett, and Michael Finke, “Meeting browser: Tracking and summarising meetings,” in Proceedings of the DARPA Broadcast News Workshop, 1998.
[8] Texas Instruments, “TL07xx Low-Noise JFET-Input Operational Amplifiers ” TL074CN datasheet September 1978– Revised June 2015
[9] Texas Instruments “LM111-N/LM211-N/LM311-N Voltage Comparator ” LM311-N datasheet mayr 199– Revised March 2013
[10] Polycom “Polycom CX5500 Unified Conference Station” Polycom CX5500 datasheet [Accessed 5- Feb- 2017]
[11] Process91, “Sound Triangulation,” in StackExchange, 2012. [Online]. Available: http://math.stackexchange.com/questions/187819/sound-triangulation. [Accesed 8-Nov-2016]
[12] *H21A1 / H21A2 / H21A3 Phototransistor Optical Interrupter Switch*. 1st ed. Fairchild Semiconductor, 2001. Web. 10 May 2017.