University of Cape Town

EEE3097S

ECE DESIGN

Milestone 4

Author:	*Student Number:*
Murray Inglis	INGMUR002
Tinashe Timba	TMBTIN004
Dylan Trowsdale	TRWDYL001

August 15, 2024

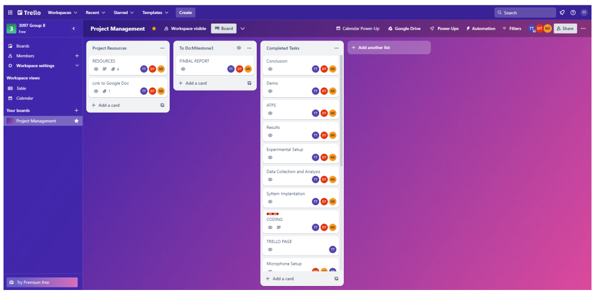
Contents

1 ADMIN DOCUMENTS

1.1 Contributions

Name	Student Number	Contributions
Dylan Trowsdale	TRWDYL001	Requirements Analysis, Conclusion
Murray Inglis	INGMUR002	Paper Design, Validation using final implementation
Tinashe Timba	TMBTIN004	Validation using Simulation, consolidation of ATPs, Admin documents

1.2 Trello



Project Resources
- RESOURCES
- Link to Google Doc
- Add a card

To Do:Milestone3
- FINBAL REPORT
- Add a card

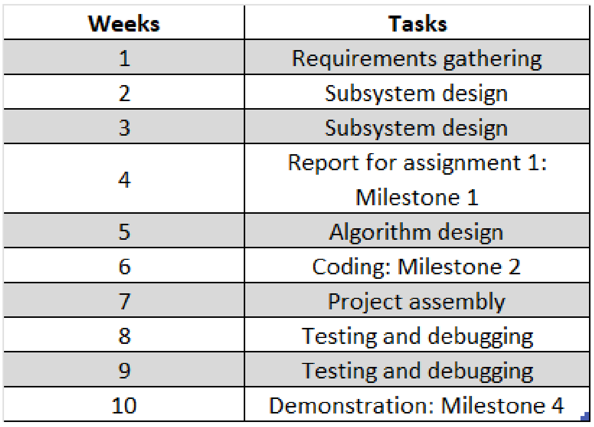
Completed Tasks
- Conclusion
- Demo
- ATPS
- Results
- Experimental Setup
- Data Collection and Analysis
- Syttem Implentation
- CODING
- TRELLO PAGE
- Microphone Setup
- Add a card

Figure 1: Project Management Tool

1.3 Link To GitHub

https://github.com/murrayinglis/3097-Group-8-2023

1.4 Timeline



Weeks	Tasks
1	Requirements gathering
2	Subsystem design
3	Subsystem design
4	Report for assignment 1: Milestone 1
5	Algorithm design
6	Coding: Milestone 2
7	Project assembly
8	Testing and debugging
9	Testing and debugging
10	Demonstration: Milestone 4

Figure 2: Project Management Tool

2 Requirements Analysis

2.1 Objectives:

- Triangulate a sound source by calculating its 2D coordinates on a grid. Involves using Time Difference of Arrival (TDoA) algorithms to determine the sound source's position based on the time delay between microphones.
- Display the triangulation result: Show the calculated position on a graphical user interface (GUI).
- Record data from each microphone simultaneously. The system should capture audio from all four microphones simultaneously.
- Implement noise reduction: Filter and process the captured audio to reduce noise interference.

2.2 Constraints:

- Must use TDoA to triangulate the sound source: TDoA methods require accurate time synchronization between the microphones.
- Must use a GUI to display the result: A user interface is necessary to visualize the calculated sound source position.
- Should be capable of recording and processing data in real-time (optional): If real-time processing is desired, it's important to consider the computational capabilities of the Raspberry Pi units.
- Must use 2 Raspberry Pi's and 4 microphones: These hardware components are

specified for the project.

- Configurable microphone setup: The system should allow adjusting the arrangement of microphones.
- Accurate location estimation within a specified tolerance: The calculated position should be within a certain range of the true sound source position.
- High signal-to-noise ratio to ensure accurate output: Minimizing noise interference is essential for accurate triangulation.

2.3 Performance Expectations:

- Should be capable of recording and processing data in real time (optional).
- Configurable microphone setup.

2.4 Parameters:

- The system should provide accurate location estimation within a specified tolerance.
- High signal-to-noise ratio to ensure accurate output.

2.5 Feasibility Analysis:

2.5.1 Strengths

- Microphone Integration: The integration of 4 microphone breakout boards with the Raspberry Pi units is feasible and common. The Pis have the necessary hardware interfaces and I2S pins to connect to the microphones.
- Signal Processing: TDoA calculations, time delay estimation, and triangulation algorithms are common techniques. There are many available libraries and code examples.

2.5.2 Weaknesses

- Timestamping: The Raspberry Pi's are running on separate clocks. An error of a millisecond will be an error of about 30cm which is a large portion of the grid size. Accurate timestamping may be hard to achieve with the software available.
- Coordination: Since the readings are being taken on 2 separate devices, it may prove difficult to transfer data reliably between them.

2.5.3 Opportunities

- Iterative Process: Iterative testing and refinement will be needed to ensure each subsystem works correctly and integrates seamlessly with others. This will be an important chance to verify our design and catch any errors preemptively.

2.5.4 Threats

- Time: Developing, testing, and integrating the various subsystems might take considerable time, especially being new to certain aspects of the project, like GUI development or synchronization mechanisms.
- Setbacks: These systems and calculations are unfamiliar. Unforeseen setbacks may occur.

2.6 Possible Bottlenecks:

- CPU performance (clock speed, sample rate): The clock speed of the Raspberry Pi and the sample rate for audio processing could impact the system's efficiency.
- Insufficient memory (RAM and external): Both RAM and external storage may become limiting factors, particularly for real-time data processing if this is done in real-time.
- Microphone quality (noise): The quality of the microphones affects the accuracy of captured audio data.
- Time management will be crucial to the project but will prove difficult with other assignments and projects.

3 Paper Design

3.1 Pi Synchronization:

3.1.1 Requirements:

- The Pis must be able to connect to the same network for data transmission and synchronization.
- Audio data captured by microphones should be synchronized between the two Pi devices.
- The synchronization mechanism should minimize data transmission delay.
- Wi-Fi connectivity for internet access.

3.1.2 Specifications:

- Use GPIO pins for synchronizing starting recording to minimize recording startup
- Set one Pi as a master that sets the pi high to start recording and one as a slave that listens for the pin to go high
- Use a synchronization 'ping' at the start of every recording, at the center of the grid to determine the delays between recording startups
- Timestamps for audio recording must be synchronized with an error of less than 0.5ms between the devices
- Transfer files via SSH over to the computer with the main program running
- Both Raspberry Pis will record audio.
- Connect the Raspberry Pi over Eduroam or home Wi-Fi networks.

3.1.3 UML Diagram:

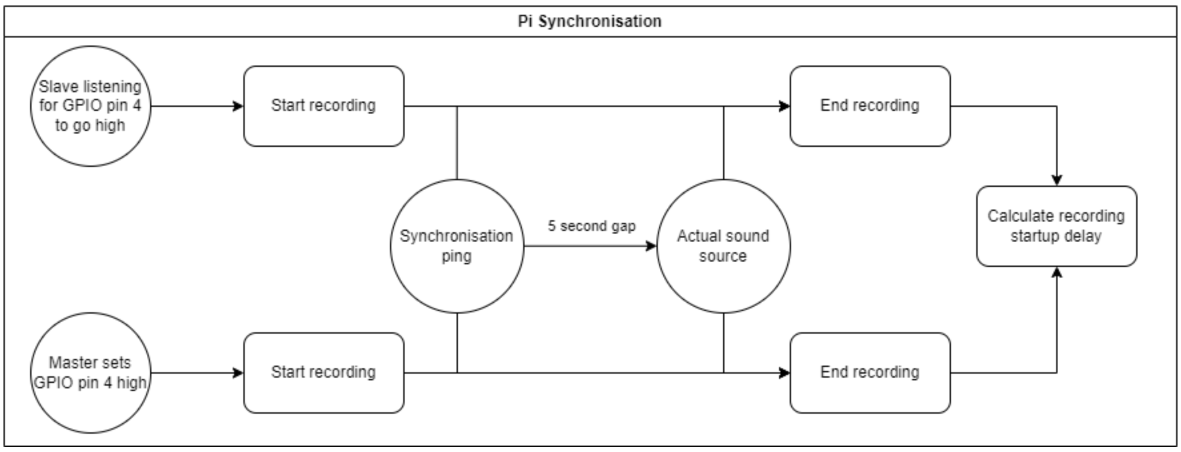


Figure 3: Pi Synchronization

3.2 Signal Acquisition:

3.2.1 Requirements:

- Use 4 microphone breakout boards and 2 Raspberry PIs to capture audio
- Signal acquisition needs to be autonomous without user involvement.
- Sample the audio above the required Nyquist rate.
- Bandlimit the input sound with filtering.
- The microphones should be physically small and efficient enough to be run on a Raspberry Pi.
- The microphones should draw a small supply of current.

3.2.2 Specifications:

- 4 Adafruit I2S MEMS microphone breakout boards will be used.
- 2 Raspberry Pi zero w v1.1 will be used.
- The microphone breakout boards must be able to record 100 Hz to 10 kHz.
- The signal will be sampled at 64kHz.
- I2S serial communication protocol will be used to communicate between the Raspberry Pi and the microphones.
- The microphones must have an SNR of above 50dB
- The supply current for the microphones should be less than 3 mA
- The microphones should operate at a supply voltage of 3.3V
- There will be 2 microphones per Raspberry Pi

3.2.3 UML Diagram:

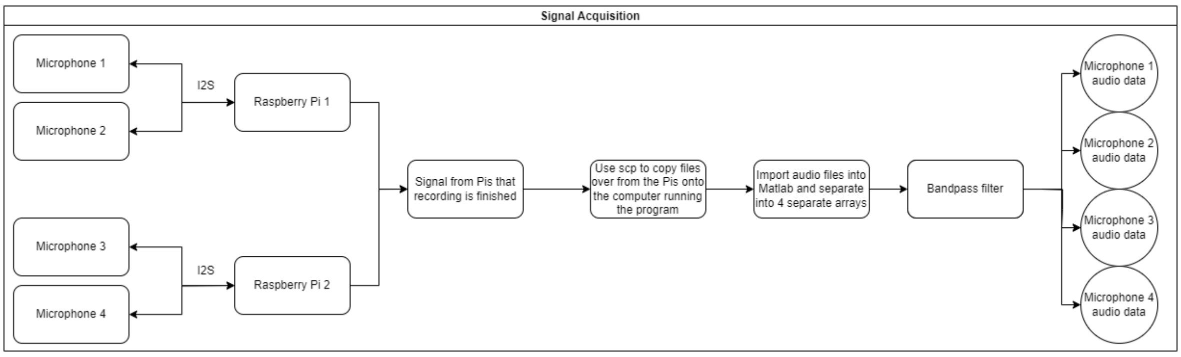


Figure 4: Signal Acquisition UML Diagram

3.3 Time Delay Estimation:

3.3.1 Requirements:

- Must be able to show the time delay on arrival for each microphone
- Must use cross-correlation to calculate the time delay.
- The sound source must have a simple frequency structure
- The sound source must be as close to an ideal spherical radiation pattern as possible

3.3.2 Specifications:

- The accuracy range must be ±5%.
- The sound source is a 400Hz to 2900Hz frequency sweep.
- One microphone will be placed at (0,0) and will be used as a reference. The other microphones will have their inputs cross-correlated against the reference microphone.
- The recording startup delay calculated with the startup synchronization ping is subtracted from the time delays calculated between microphones on different Pis.

3.3.3 UML Diagram:

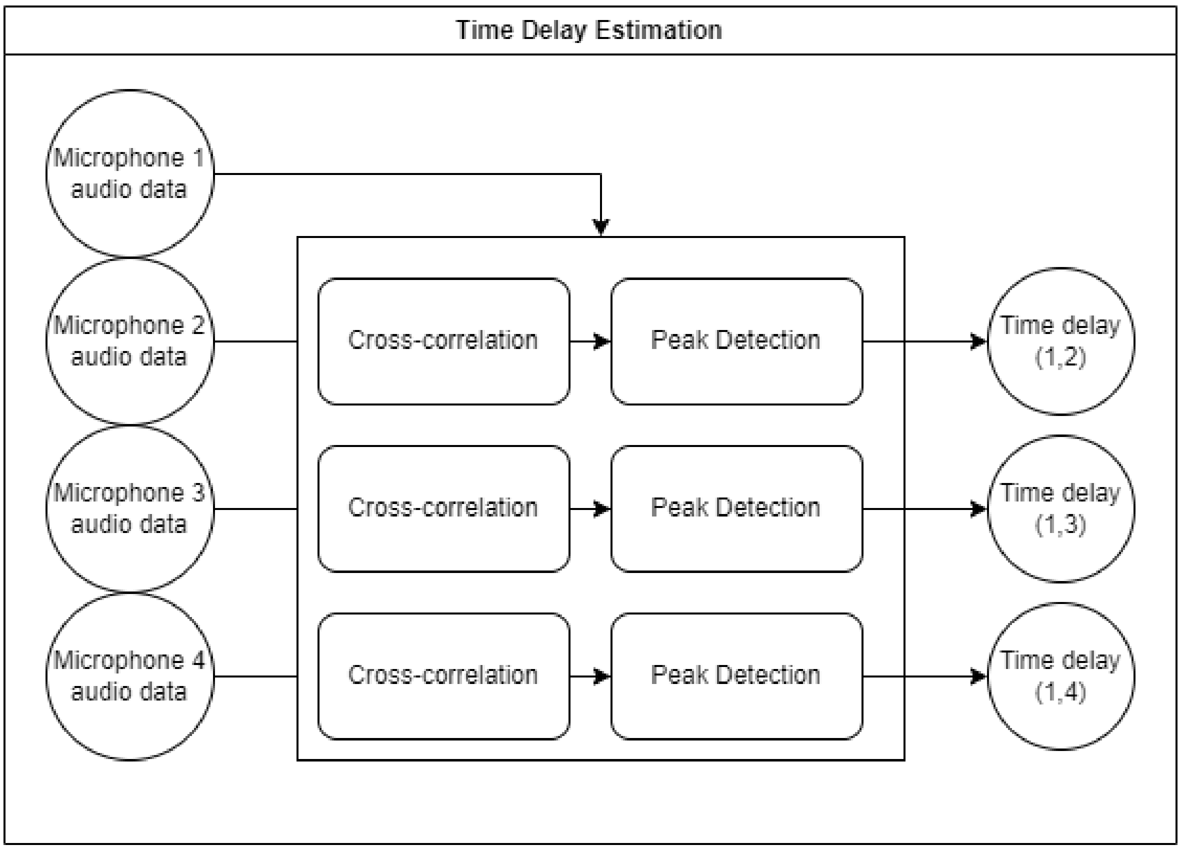


Figure 5: Time Delay Estimation UML Diagram

3.4 Triangulation:

3.4.1 Requirements:

- Must be able to calculate the position of the sound source within a given accuracy range.
- Must use 4 sensors, thus creating a system of 4 nonlinear equations.
- The position of the microphones must be known by the system and configurable.
- The position of the microphones must be configurable and programmable.

3.4.2 Specifications:

- The accuracy range must be ±5%.

- The time delay from each microphone to the reference is used to create distance equations.
- The equations are solved using an iterative algorithm on Matlab using the *fsolve* function. This function solves the equations using the Levenberg-Marquardt algorithm.

3.4.3 UML Diagram:

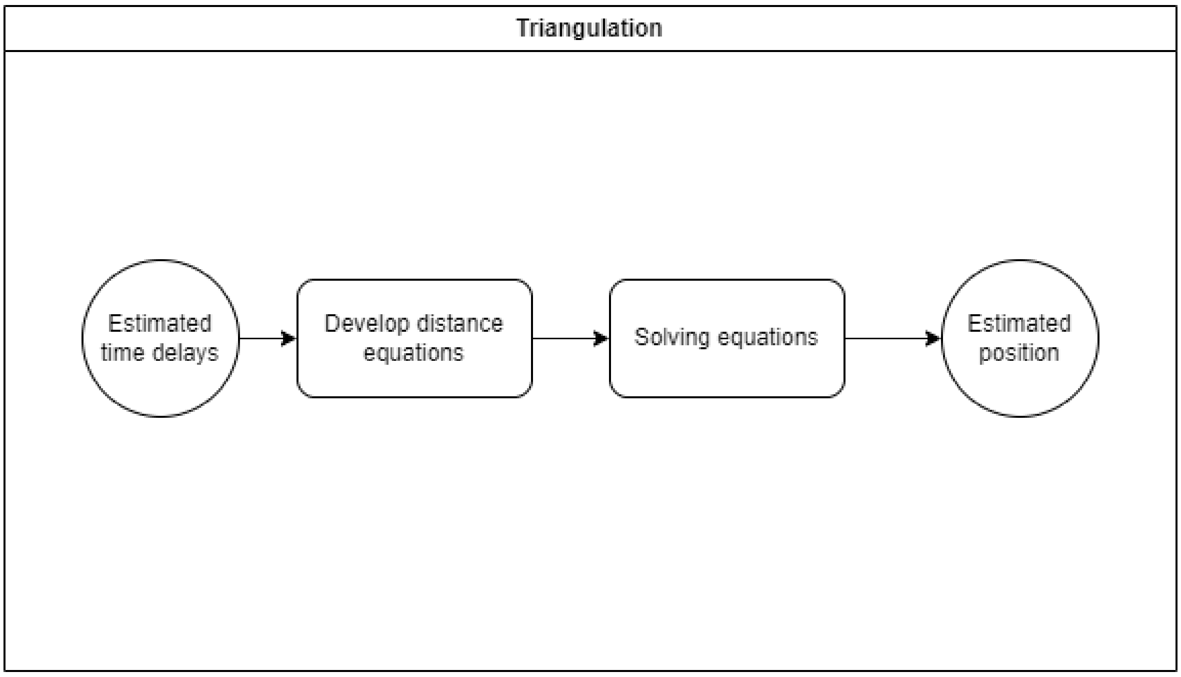


Figure 6: Triangulation UML Diagram

3.5 User Interface:

3.5.1 Requirements:

- Must use a Graphical user interface to display the coordinates of the predicted results.
- Be able to start and stop measuring.
- Log the data related to each localization session for analysis of results.
- Notify a user when each localization session is finished or if errors have occurred.

3.5.2 Specifications:

- A graphical user interface is to be used for a user to start and stop the sound localization session and display the output results of the predicted source location on the grid to the output space of the GUI.
- Prompt the user when to play the synchronization ping and when to play the ping at the desired source location.
- Output notification messages to the user to notify them of the status of the process at each stage (recording stage, file transfer stage, calculation stage).
- Each section must be able to be run independently from the terminal for proper debugging.

3.5.3 UML Diagram:

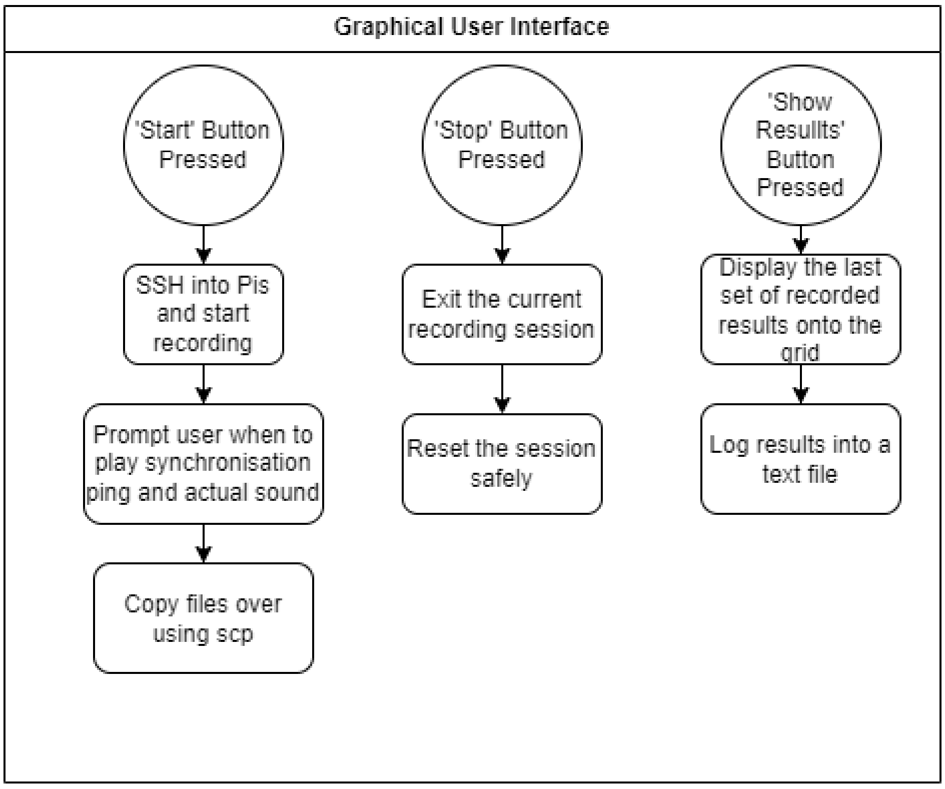

Graphical User Interface
'Start' Button Pressed
SSH into Pis and start recording
Prompt user when to play synchronisation ping and actual sound
Copy files over using scp
'Stop' Button Pressed
Exit the current recording session
Reset the session safely
'Show Resullts' Button Pressed
Display the last set of recorded results onto the grid
Log results into a text file

Figure 7: Graphical User Interface UML Diagram

3.6 Inter Sub-System Interaction

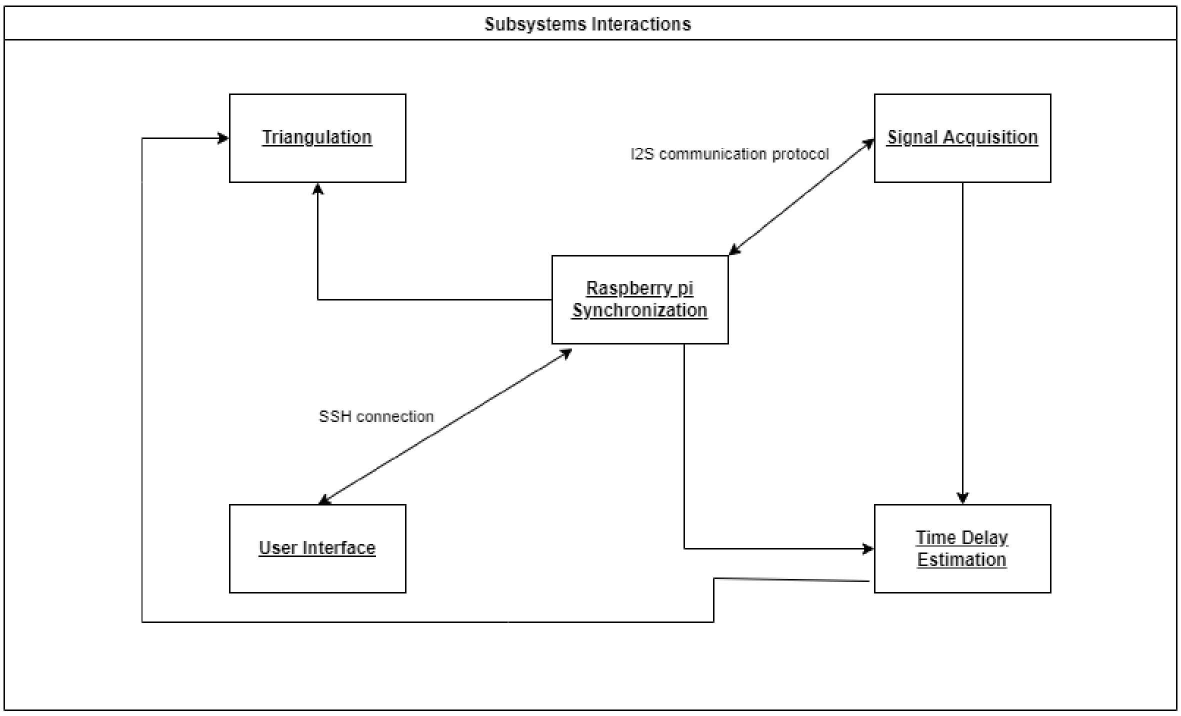


Figure 8: Sub-Sytsem Interactions

4 Validation using simulations

4.1 Need for Simulation-Based Validation

Simulation decreases any risks during the design and development of a project. By testing various situations, we can identify any potential design flaws before design implementation which helps reduce issues and defects in the final system. In the case of Acoustic triangulation, simulation is essential to assess the performance of the algorithm in a controlled and repeatable system. It allows for extensive testing and adjustment and allows us to optimize the localization without real-world constraints or risks such as component breakdown or the risk of incorrect setup without a foreseeable expectation. The main objective for simulation includes evaluating the accuracy, range, and reliability of the system under varying conditions verifying the compliance with requirements and specifications, and identifying potential issues we otherwise could have missed in the real-world system.

4.2 Steps for Simulation-Based Environment

1. Model Development
2. Definition of Varying Situations
3. Defining Acceptable Performance
4. Analysis and Optimization

4.3 Discussion of Steps

1. Model Development

In the acoustic triangulation system, we employed four microphones positioned at different points on a grid. The system's objective is to detect sound from a defined source. In a real-world scenario, due to physical separation and the finite speed of sound, each microphone detects the sound with varying arrival times. In our simulation, we defined the positions of the microphones as discrete points on a grid. The separation between each microphone and the defined source point was modeled using the Pythagorean theorem for calculating the distance between two points. The ideal speed of sound was utilized to determine the relative times of arrival:

$$Time = \frac{Distance}{Speed} \tag{1}$$

To replicate the real-world scenario, we added time delays to each signal definition to simulate signals arriving at each microphone at different times.

Our primary objective in this step is to determine the Time Difference of Arrival (TDOA), which is a crucial parameter for estimating the source point. We employed a cross-correlation function, using microphone 1 as the reference microphone, to identify the peak index, thus calculating the TDOA. A generalized cross-correlation function is defined as taking in the reference signal (the signal acquired by the reference microphone) and the signal received by another microphone. It then uses the xcorr() function to cross-correlate the two signals. The result is divided by the sampling frequency to acquire an estimated time delay. This gives the time delay on arrival from the reference mics to the other three mics.

Considering the system requirements, the optimal approach for source point determination involves using the intersection of three hyperbolas. These hyperbolas are constructed based on the TDOA and the known microphone positions. We defined three equations for each microphone's hyperbola and employed non-linear regression techniques and averaging to pinpoint the center of the triangle formed by the intersection of these hyperbolas. The equations are as below:

$$t_{d12} \times 343 = \sqrt{(x-x_1)^2 + (y-y_1)^2} - \sqrt{(x-x_2)^2 + (y-y_1)^2} \tag{2}$$

$$t_{d13} \times 343 = \sqrt{(x-x_1)^2 + (y-y_1)^2} - \sqrt{(x-x_3)^2 + (y-y_3)^2} \tag{3}$$

$$t_{d14} \times 343 = \sqrt{(x-x_1)^2 + (y-y_1)^2} - \sqrt{(x-x_4)^2 + (y-y_4)^2} \tag{4}$$

These equations can be transformed into a hyperbolic form and will have three intersection points, forming a triangle. The estimated source position is taken as the center of this triangle.
The equations are solved using Matlab's numerical solver, *fsolve()* to give the estimated position of the source.

To account for real-world environmental conditions and potential noise, we introduced noise modeling. Environmental conditions, including noise, were simulated using MATLAB's random number generator, replicating scenarios encountered in practical applications. This model development phase establishes the foundation for our acoustic triangulation system simulation, enabling us to replicate real-world conditions and assess the system's performance under various scenarios.

2. Definition of Various Situations

In our simulation, we aimed to maintain consistent microphone positions to minimize variance in the collected results. To achieve this, we kept the microphone positions constant throughout the simulation runs. To mimic the variability we anticipate in our future implementation, we generalized the source point. We allowed it to vary based on user input, which provided different x, and y coordinates. This approach enables us to simulate diverse scenarios where the sound source can be located at different positions.

In our simulation, we recognized that signal frequency may vary in real-world scenarios. Therefore, we did not constrain the signal frequency to a constant value. Instead, we made it subject to change. This flexibility allows us to assess the system's performance across different frequency ranges. Another variable we expected to change in real-world conditions was noise. To account for this variability, we introduced random noise as a dynamic factor in each run of the simulation. The use of MATLAB's random number generator enabled us to simulate a realistic environment, where noise levels might differ with each instance of sound detection. This step in our simulation design ensures that our system can adapt to various situations that may arise in practical applications. By considering changes in source points, signal frequency, and noise levels, we aim to create a robust system that can handle real-world challenges effectively.

3. Defining Acceptable Performance

Acceptable performance is achieved when the majority of the estimated positions deviate by less than 2cm from the specified source position. Additionally, at least 80% of the positions should fall within a ±5cm range. In this context, acceptable performance in the simulation stage means that the microphones can reliably capture sounds within the specified frequency range (100 - 10,000 Hz) which can easily be defined in the MATLAB code. It's also important that noise has minimal impact on the system's accuracy hence the fileter needs to attenuate the additional noise. In the case of synchronization stability and accuracy, acceptable performance would involve maintaining errors within a predefined range, such as within a few milliseconds; for the simulation, this can not be tested as the hardware is not being implemented.

4. Analysis and Optimization

In the initial testing phase, it became evident that the simulation was not meeting the expected performance standards. This was primarily due to code errors within the simulation environment. To address these issues, we used MATLAB's code sectioning capabilities, allowing us to isolate and analyze individual sections of the code. Through a systematic code inspection, we identified and located the bugs that were affecting the simulation's performance. These issues were addressed through code revisions and debugging efforts. Once the bugs were successfully identified and rectified, the simulation began to operate smoothly without errors. Following the bug fixes and improvements to the code, it was observed that the system was running efficiently and effectively. At this stage, further optimization was not deemed necessary, as the simulation was achieving the intended outcomes and adhering to performance standards. This phase involved a meticulous analysis of the code, the identification and resolution of code errors, and the assurance that the simulation was operating optimally, meeting the required performance criteria.

4.4 Experimental Setup and Results

For the simulation-based validation, we employed the following experimental setup:

A MATLAB-based simulation environment was used to model the acoustic triangulation system. Simulated microphones with known characteristics were placed in set positions. A range of scenarios was defined, including different sound source distances and signal noise levels. Data generated from the simulations were analyzed and compared against system specifications and requirements. A Matlab script, *localize.m*, was written and can be found on the group GitHub repository. This script can be called in Matlab using *localize(x,y)*. The signals with time delay received by each microphone will be plotted, the hyperbolas used for triangulation will be plotted and the estimated source position will be outputted.

Results

Signals were defined and acquired successfully. An example is as below:

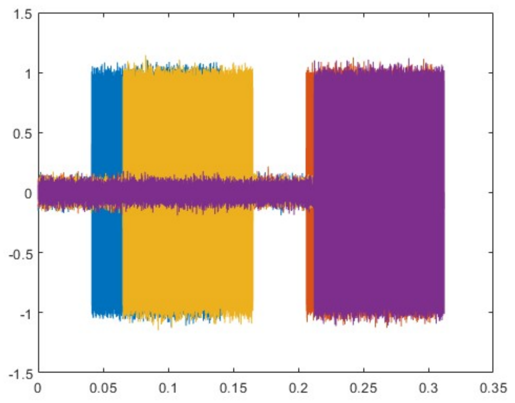


Figure 9: Signals received

The simulation was run using different source points. The results are tabulated below:

Index	Source Point	Estimated Point	Deviation in x	Deviation in y
1	0.2, 0.4	0.19967, 0.398	0.000033	0.002
2	0.4, 0.4	0.4, 0.40216	0	0.00216
3	0.7, 0.3	0.69934, 0.29893	0.000066	0.00107
4	0.2, 0.1	0.20066, 0.10124	0.00066	0.00124
5	0.4, 0.1	0.4, 0.097841	0	0.002159
6	0.7, 0.0	0.70033, 0.0015767	0.00033	0.0015767
7	0.6, 0	0.60125, 7.1781e-05	0.00125	7.1781e-05
8	0.6, 0.3	0.59988, 0.29848	0.00012	0.00152
9	0.1, 0.5	0.099196, 0.49712	0.000804	0.0.00288

Table 1: Position Accuracy Results For Simulated System

The deviation noted from the simulation is extremely small and it validates that the algorithm works in simulation.

5 Validation using final implementation

It is necessary to do a hardware-based validation so that the final results can be declared accurate and correct. The step to do a hardware-based validation would be to validate the functioning of each separate subsystem and the interactions between them. These steps are followed below for the rest of this section, by going through each subsystem and checking that the results are expected, then checking that the final result showing the interaction between all subsystems is correct.

5.1 Validation of Pi Synchronisation

The scripts used to SSH into the Pis, start the master and slave scripts, and start the recording allow the user to see the progress in this process. It was observed that recording started around the same time, since the synchronization ping is used to calculate the difference in recording startups, this delay is acceptable and expected.

Calculated time delays from the calibration signal for one of the recordings:

```
Time delay between mics 1 and 2: 4.6875e-05
Time delay between mics 1 and 3: 0.023703
Time delay between mics 1 and 4: 0.023703
Time delay between mics 3 and 4: 0
```

It can be seen that the time delays between microphones on the same Pi (between 1 and 2 and between 3 and 4) are negligible or actually 0. However, the time delays between microphones from different Pis have a significant value. This value can be used to accurately offset the signals.

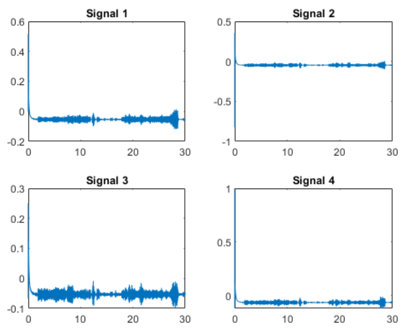
It is also possible to run each step separately and observe the slave and master start recording soon after the master sets GPIO pin 4 high.

5.2 Validation of Signal Acquisition

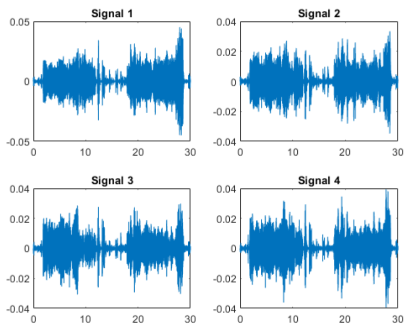
To decrease the amount of noise in the signals, a bandpass filter covering a frequency range from 300Hz to 3000Hz.

A noticeable pop sound is present at the start of each recording. This was significantly affecting the results from the cross-correlation. To remove this pop sound, the first 3ms of the recording are trimmed off.

To verify the filtering and trimming worked, the waveforms of the signals before and after processing are plotted and analyzed. It can be seen that the pop at the start has been removed and the noise reduced.



(a) Signal waveforms before processing



(b) Signals waveforms after processing

Figure 10: Waveforms verifying the signal processing

5.3 Validation of Triangulation

Below are readings taken from the final implementation. The actual source position, the estimated position outputted from the system, and the deviation are tabulated. The hyperbolic distance equations are plotted to show the intersection.

Index	Source Point	Estimated Point	Deviation in x	Deviation in y
1	0.2, 0.4	0.22074, 0.40506	0.02704	0.00506
2	0.4, 0.4	0.45578, 0.43876	0.05578	0.03876
3	0.7, 0.3	0.73337, 0.29822	0.03337	0.01288
4	0.2, 0.1	-15120.6981, -36623.6907	15120.04981	36623.5907
5	0.2, 0.1	-15120.6981, -36623.6907	15120.04981	36623.5907
6	0.2, 0.1	0.22682, 0.10536	0.02682	0.00536
7	0.4, 0.1	0.40474, 0.10699	0.00474	0.00699
8	0.7, 0.0	0.75671, -0.037232	0.05671	0.037732
9	0.6, 0	0.57209, 0.016396	0.02791	0.016366
10	0.6, 0.3	0.68596, 0.30504	0.08596	0.00504
11	0.1, 0.5	0.13252, 0.48276	0.03252	0.01724

Table 2: Position Accuracy Results For Actual System

Visualized Results From Recordings

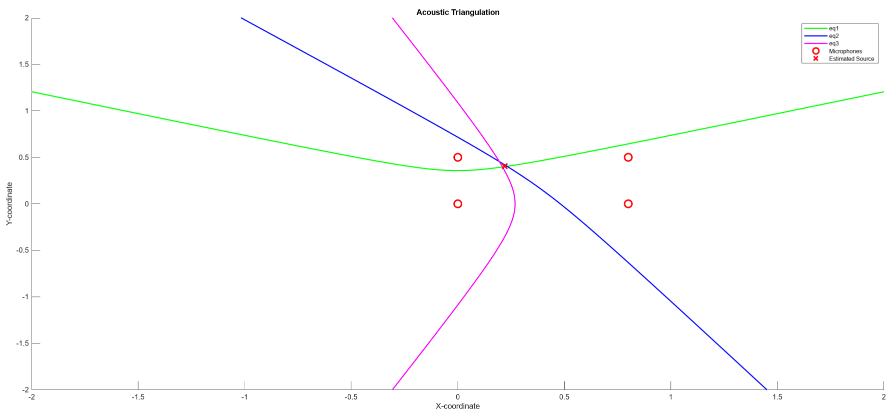


Figure 11: Index 1

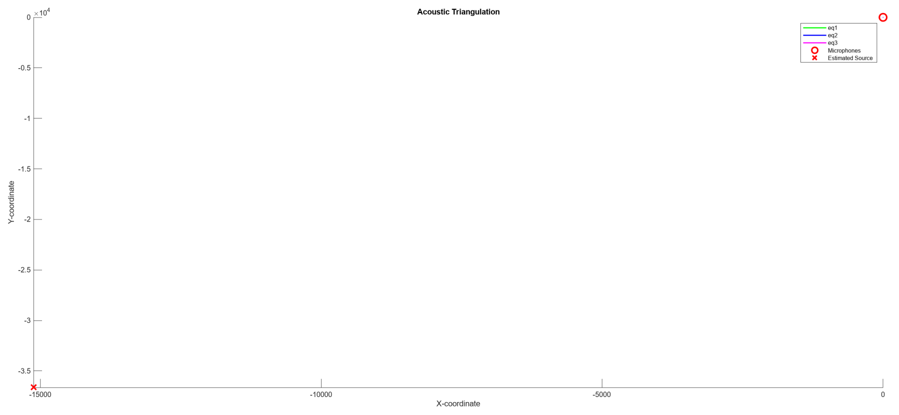


Figure 12: Index 4

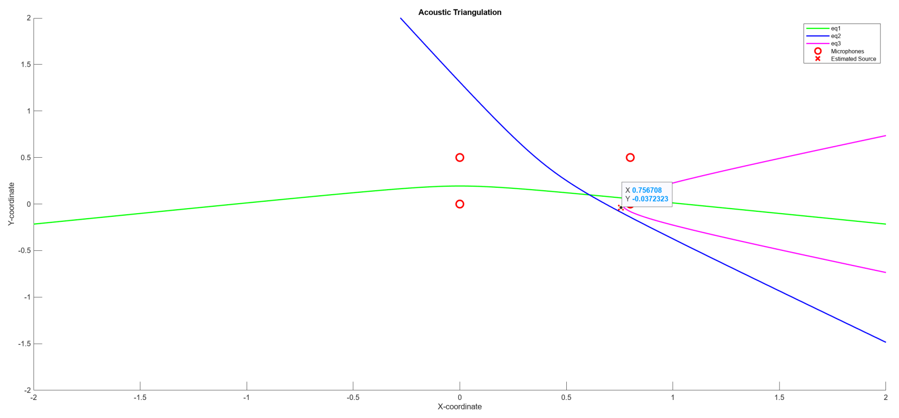


Figure 13: Index 7

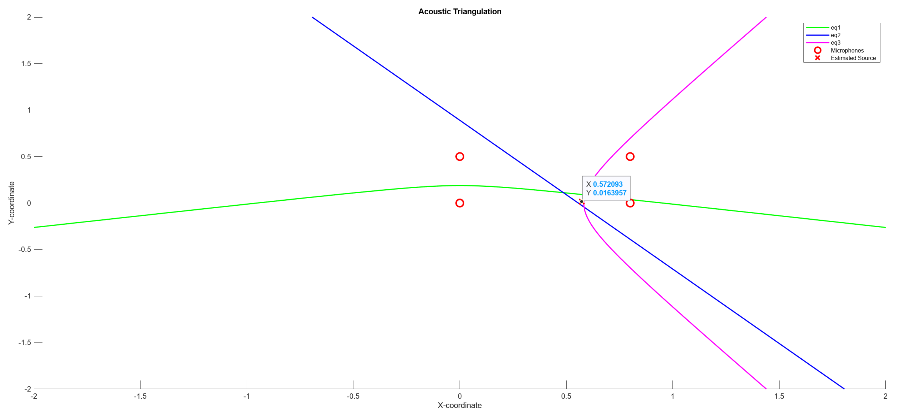


Figure 14: Index 8

It can be seen that every result other than index 4 and 5 accurately estimated the source location well within the required 5% deviation. The anomaly of results in indexes 4 and 5 is most likely explained by a synchronization error, reflections of sound off of nearby objects, or too high a level of noise. We found that with the phone speaker used to play the sounds, the level of background noise due to people talking nearby would throw off the results. A louder speaker provided robust results that had no deviations like this. If the phone speaker was also tilted slightly, the time delay between recording startup was also incorrect and caused errors in the estimation.

The number of readings and consistency in the results from the readings taken, validate that the system implementation is correct and works for a variety of points in the grid. Edge cases that could cause problems, such as near the microphones and on the borders of the grid are considered and shown to have accurate estimations.

6 Consolidation of ATPs and Future Plan

6.1 Figures of Merits

1. Position Accuracy via TDoA calculation: The deviation between the calculated position using the TDoA algorithm and the actual position of the sound source.
2. Sound Detection and Signal-to-Noise Ratio: The frequency range for which the microphones are able to pick up the sound clearly. The ratio of the signal generated to the surrounding noise.
3. Synchronization Stability and Accuracy: Testing the pi synchronization is consistent.
4. Usability Testing of GUI: The ease of use of the user interface.
5. Unit test for code: Debugging and testing algorithm.

6.2 Position Accuracy Test

Setup:

- Place the microphones at a standard position on the grid.

Procedure:

1. Emit sound at specified grid points.
2. Calculate position using the TDOA algorithm.
3. Measure position deviation and processing time.
4. Repeat for multiple points.

Acceptable Performance:

- Mean position deviation within ±2cm.
- 80% of positions within ±5cm.

6.3 Sound Detection and SNR Test

Setup:

- Place the microphones randomly and ensure the sound source is available

Procedure:

1. Play a sound of known frequency along with real-world noise
2. Detect and process the sound
3. Calculate the SNR
4. Vary the frequency and gain of the sound source.

Acceptable Performance:

- The microphones are able to pick up sound from 0-20 kHz.
- The SNR is high.
- Minimal effect of noise on position accuracy.

6.4 Synchronization Stability and Accuracy

Experiment Design:

1. Continuously monitor the clock synchronization between the Raspberry Pi units over an extended period.
2. Measure synchronization accuracy over time.

Acceptable Performance:

- Synchronization error should remain within the specified range (e.g., within a few milliseconds) consistently.

6.5 Usability Testing of GUI

Experiment Design:

1. Engage users who are not familiar with the system.
2. Have them interact with the GUI to perform tasks like starting/stopping measurement and changing settings.

Acceptable Performance:

- Users should be able to easily navigate and perform tasks using the GUI without confusion.

6.6 Unit tests for the Raspberry Pi code

Experiment Design:

1. Test code section by section.
2. Debug the code using the Raspbian debugger.

Acceptable Performance:

1. The code runs without errors.

6.7 ATP ANALYSIS

ATP	Stated Acceptable Performance	Discussion
Position Accuracy Test	• Mean position deviation within ±2cm. • 80 % of positions within ±5cm.	The Acceptance Test Procedure (ATP) was deemed satisfied as 80% of the estimated positions fell within the desired uncertainty range. For the more precise positions, deviations were within the 2cm margin specified.
Sound Detection and SNR	The microphones are able to pick up sound from 100 - 10 000 Hz. • The SNR is high. • Minimal effect of noise on position accuracy.	The evaluation of our Acoustic Time of Arrival (TDOA) system revealed that the sound source utilized fell within the specified frequency range of 400Hz to 2900Hz. However, it was observed that a decrease in signal volume, accompanied by a reduction in the Signal-to-Noise Ratio (SNR), resulted in a large deviation from the intended positional accuracy. For example, reading 4 and 5 in Table 1 could have been due to high background noise, possible sound reflections, and possible synchronization errors. Additionally, a significant influence was noticed regarding noise levels on the accuracy of the TDOA measurements. This was a result of the microphones likely being optimized for recording human voices (frequency ranges 100Hz-8kHz) [3]. This means that if humans talk whilst the system is recording the test signal, the voice frequencies within the range of the test signal would not be filtered out, thus causing errors in the time delays. Furthermore, this effect was attributed to the source of our sound, originating from mobile devices, which provided relatively lower sound intensity when compared to a more robust sound source such as a speaker.

Usability Testing of GUI	• Users should be able to easily navigate and perform tasks using the GUI without confusion.	User feedback indicates that the Acceptance Test Procedure (ATP) was successfully met. Users found the instructions to be clear and straightforward, making the application easy to use.
Unit tests for the Raspberry Pi code	• The code runs without errors	The code was debugged thoroughly and executed without encountering any errors.
Synchronization Stability and Accuracy and synchronization	• Error should remain within the specified range (e.g., within a few milliseconds) consistently	The presence of Time Difference of Arrival (TDoA) caused by synchronization errors was effectively mitigated by incorporating a calibration sound into the recording process. As a result, this Acceptance Test Procedure (ATP) mostly met but some errors but some anomalies were present.

Table 3: ATP ANALYSIS

6.8 Possible Improvement and Future Plans:

NB: This assumes that we have our own budget and access to additional hardware components.

Position Accuracy, Sound Detection and SNR

A higher-order band-pass filter with a steeper roll-off could be implemented to ensure noise is completely filtered out.
A change in the frequency range of the test signal could also be made so that it is not within the same frequencies as human voices which would reduce the noise levels and eliminate unwanted signals at the same frequency as the test sound.
In addition, a speaker with a close-to-ideal spherical radiation pattern could be used to produce the sound signal instead of relying on the phone speaker.
Our system did not have sufficient error detection and error correction facilities. It would have been possible to detect when one of the calculated time delays is greater than the maximum possible time delay and instead use the intersection of the other 2 hyperbolas. In the case where all time delays are above this limit, the system could state that there is an error in the processing and the experiment should be repeated.

Usability Testing of GUI

No further improvements are needed for this subsection

Unit tests for the Raspberry Pi code

No further code improvements could be made. However, the SSH script was shown to fail to make a successful connection depending on network stability, especially on the network at the UCT campus. Hence a possible improvement could be to use a different means of communicating with the Pi that does not require network connectivity such as SPI or UART. This was not within the scope of what was possible for this assignment since it required too much extra hardware.

Synchronization Stability and Accuracy

Access to Ethernet cables could have made a more efficient means of synchronization compared to relying on GPIO pins and a calibration signal. Additionally, several other synchronization techniques such as PTP, and GPS Time stamping could have been useful. Although the use of a calibration signal removed most inaccuracies due to synchronization errors this proved to make data collection a lot longer than needed.

Additional/New Specifications

ATP NOT MET	OLD SPECIFICATION	NEW SPECIFICATIONS
Synchronization Stability and Accuracy and synchronization	• Use GPIO pins for synchronizing starting recording to minimize recording startup. • Set one Pi as a master that sets the Pi high to start recording, and one as a slave that listens for the pin to go high. • Use a synchronization 'ping' at the start of every recording, at the center of the grid to determine the delays between recording startups.	• Use an Ethernet cable between Pis to ensure Pis start recording at the same time.
Sound Detection and SNR	• The microphone breakout boards must be able to record 100 Hz to 10 kHz.	• The microphone breakout boards must be able to record 100 Hz to 10 kHz, and the recording must be filtered by a sixth-order bandpass filter to remove noise.

Table 4: Comparison of Specifications

Future Plans

In the course of developing our acoustic triangulation system and conducting simulations, we identified areas for potential improvements and enhancements that could be implemented in future iterations of the system:

One avenue for improvement is to design the system to operate within the same frequency range as human voices. By doing so, the system could better reduce noise levels and eliminate unwanted signals operating at the same frequency as the test sound. This modification would enhance the system's capability to differentiate target signals from ambient noise.

In future implementations, the use of a speaker with a close-to-ideal spherical radiation pattern should be considered to produce the sound signal. Relying on a speaker with such a radiation pattern would enhance the consistency and reliability of sound propagation, resulting in more accurate measurements.

To improve the robustness of the system, the inclusion of error detection and correction mechanisms is essential. Future versions of the system should incorporate error detection for calculated time delays that exceed a predefined threshold. When such an event occurs, the system could switch to alternative methods for source point determination, such as using the intersection of the other two hyperbolas. Additionally, if all time delays surpass the threshold, the system could indicate an error in processing and prompt for the experiment to be repeated.

While the current code functions effectively, network stability issues were observed when using SSH for communication with the Raspberry Pis, especially in certain network conditions. A potential improvement could involve exploring alternative means of communication that do not rely on network connectivity, such as SPI or UART. However, such changes may require additional hardware components beyond the scope of the current assignment.

To enhance synchronization stability, Ethernet cables could be employed to provide a more efficient means of synchronization compared to GPIO pins and a calibration signal. Alternatively, techniques like GPS time-stamping could be explored for further accuracy. The use of a calibration signal, while effective in reducing synchronization errors, may also result in longer data collection times. Future iterations of the system could aim for more efficient synchronization methods.

In summary, these potential future implementations represent areas where the acoustic triangulation system can be further enhanced to meet evolving requirements and address identified limitations.

7 Conclusion

In conclusion, during the course of this project, we have set out to design and implement an Acoustic Localization system according to the given requirements, specifications, and constraints. The team moved through the stages of design, beginning with a clear set of and analysis of requirements, specifications, and constraints which was followed by the Paper design, simulations, experimental implementation, and full system implementation. This culminated in a robust system capable of accurately estimating the position of sound sources and meeting most of the ATP's set out for this project.

The significance of our Acoustic Localization project lies in its various real-world applications. Our Acoustic localization system offers use to enhance small-scale object localization applications where tracking and locating sound sources are critical such as home automation and robotics.

The extensive simulation validation of the system allowed us to understand the system's capabilities and limitations under controlled and ideal conditions. This allowed us to refine triangulation algorithms, sound detection, GUI usability, and code functionality. These insights gained from the simulations were invaluable for refining the system for practical experimentation and implementation.

Experimental implementation of our system introduced real-world complexities, noise, and synchronization challenges between the Raspberry Pis. With some creative problem-solving skills, the team was able to address the challenges faced due to the constraints of the system and the outcome resulted in a system that was able to accurately and effectively estimate source positions in most scenarios. As evident from the results, the system fulfilled the necessary ATPs. However, with additional environmental noise (human speech as explained above), the system's accuracy and reliability deteriorated.

In the journey from project conception to implementation, the team identified areas for potential improvement and optimization. This included enhancing the synchronization, time-stamping, and communication capability of the Raspberry Pis with the addition of extra hardware, extending and optimizing the system's frequency range to frequencies outside that of human speech or common environmental noise frequencies, selecting or using more appropriate speaker for sound generation for the sound source and implementing error detection and correction.

The Acoustic Localization system has evolved from an idea to a working prototype. The success of the project showcases the value of multidisciplinary collaboration in signal processing, hardware engineering, mathematical algorithm application, and software development.

References

[1] Help on BibTeX entry types. http://nwalsh.com/tex/texhelp/bibtx-7.html. Accessed: 2015-03-12.

[2] How to locate an unknown point using the tdoa method of trilateration, mathematics stack exchange. https://math.stackexchange.com/questions/3367330/how-to-locate-an-unknown-point-using-the-tdoa-method-of-trilateration. Accessed: 2023-09-10.

[3] Human voice frequency range. https://seaindia.in/blogs/human-voice-frequency-range/#:~:text=A%20Female%20voice%20frequency%20range,Harmonics%20is%20900Hz%20to%208KHz. Accessed: 2023-10-15.

[4] i2s datasheet. https://cdn-shop.adafruit.com/product-files/3421/i2S+Datasheet.PDF. Accessed: 2023-08-17.

[5] i2s microphone breakout pdf. https://cdn-learn.adafruit.com/downloads/pdf/adafruit-i2s-mems-microphone-breakout.pdf. Accessed: 2023-08-17.