EEE3097S 2023
Milestone 3: Second Progress Report

Group 2

15 October 2023

Aimee Simons (SMNAIM002)

Tilal Mukhtar (MKHTIL001)

Md Shaihan Islam (ISLMDS002)

1 TABLE OF CONTENTS

2 ABBREVIATIONS

GCC-PHAT: Generalized Cross-Correlation Phase Transform

GPIO: General Purpose Input Output

I2S: Inter-IC Sound

LSE: Least Squares Estimator

MEMS: Micro-Electro-Mechanical System

Mic: Microphone

NTP: Network Time Protocol

OS: Operating System

RPi: Raspberry Pi

SCP: Secure File Copy

SNR: Signal to Noise Ratio

SSH: Secure Shell

TDoA: Time Difference of Arrival

ToA: Time of Arrival

USB: Universal Serial Bus

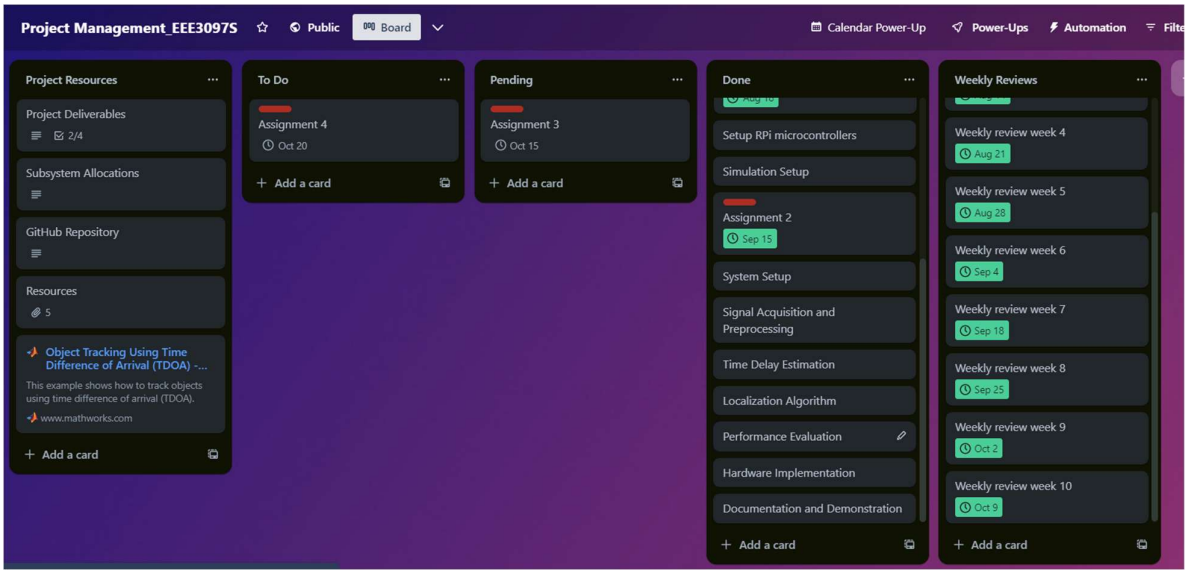
WBS: Work Breakdown Structure

3 ADMIN DOCUMENTS

3.1 CONTRIBUTIONS

Student Name	Student Number	Contributions
Md Shaihan Islam	ISLMDS002	System Implementation (Graphical User Interface) Evaluation (ATPs) Conclusion
Tilal Mukhtar	MKHTIL001	System Implementation Experimental Setup Results
Aimee Simons	SMNAIM002	System Implementation Experimental Setup Data Collection and Analysis Conclusion

3.2 PROJECT MANAGEMENT PAGE



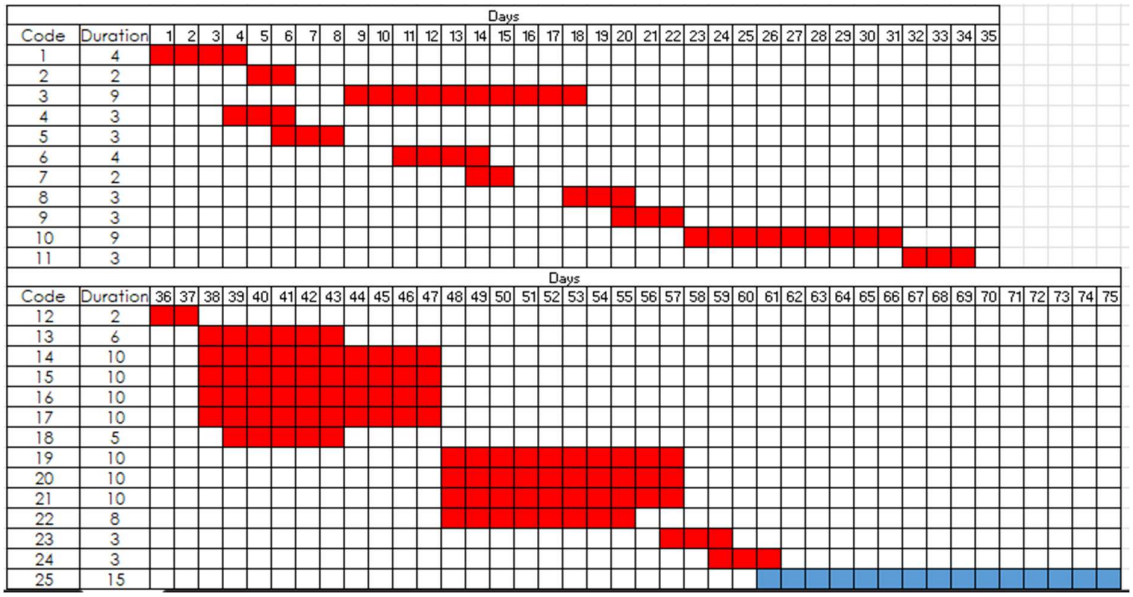
3.3 GITHUB REPOSITORY

Link to GitHub Repository:

https://github.com/tshaihan/EEE3097S_Group2

3.4 PROJECT TIMELINE

The cells shaded in red show the progress completed so far, whereas the cells in blue mean that the particular section is not complete and is still in progress.



From the Gantt chart above, it can be seen that the project development is behind schedule by eleven days. However, since extra time had been allocated for such situations, there will be just over a week to complete the final report write up.

The WBS for the remaining tasks is shown below:

Code	Activity	Duration	Dependency
25	Final Report Write-up	04/09/2023 - 18/09/2023	24

4 SYSTEM IMPLEMENTATION

4.1 OVERALL SYSTEM IMPLEMENTATION

4.1.1 Hardware

The overall system implementation consists of the following hardware components:

- 2 Raspberry Pi Zero W modules
- 4 Adafruit I2S MEMS microphone breakout boards
- An A1 (0.8x0.5 m) paper grid
- A laptop running the Microsoft Windows OS (Host Device)
- 2 Mobile Devices
- 2 Bluetooth Speakers
- 3 Micro-B USB cables
- A USB Hub
- 2 Breadboards
- Jumper cables
- Electrical Tape

The 2 RPi modules were each connected to a pair of the Adafruit I2S MEMS microphone breakout boards via jumper cables connected through the 2 breadboards. The connections between the RPi modules and the Mic breakout boards utilized the I2S interface. The Mics were each placed on a separate corner of the A1 paper grid. The coordinates of Mics 1 to 4 were (0, 0), (0, 0.5), (0.8, 0.5) and (0.8, 0) respectively. Electrical tape was used to secure the Mics and jumper cables in place. The RPi modules were connected to the host device wirelessly via a local network. Additionally, the RPi modules were connected to a USB hub via micro-B USB cables. The USB hub was connected to the host device via a micro-B USB cable. The USB hub was used to power both RPi modules from the host device. Lastly, the 2 Bluetooth speakers were used to produce the required audio signals on the grid. They were each wirelessly connected to a mobile device via Bluetooth. The mobile devices were used to select the audio signals.

With regards to challenges, one of the RPi modules provided were faulty and had to be replaced, which pushed the project behind schedule for a short period of time. Once replaced, the microphones showed signs of not recording, even though both RPi modules were configured correctly. This was suspected to be lack of secure wiring and connections. This was addressed by soldering each of the microphones onto Veroboard as well as soldering Female Header Sockets for easy and secure connections.

4.1.2 Software

The overall system consists of the 6 Python programs given below.

- **gui.py**: a graphical interface for the user to interact with the system.
- **signal_acquisition.py**: extracts and processes the audio signals from the recording files.
- **gcc_phat.py**: a library implementation of the GCC-PHAT algorithm.
- **time_delay_estimation.py**: estimates the time delays between a set of signals and a reference signal.
- **triangulation.py**: triangulates the 2D position of a signal based on the receiver positions and TDoA values.

Additionally, the system makes use of the following Python libraries:

- NumPy
- SciPy
- Matplotlib
- PySimpleGUI
- subprocess

Lastly, the system also makes use of the following collection of bash scripts:

- main.sh
- pi_one.sh
- pi_two.sh
- raspberrypi.sh

With regards to challenges, it was initially difficult to synchronise the start of the recordings on both microphones. To address this, thorough research on parallel scripts was conducted. This provided a good basis for understanding how scripts run and also how to execute them in the background. Further information on the execution of these scripts is discussed in the next section.

4.2 SUBSYSTEM IMPLEMENTATION

4.2.1 Signal Acquisition

The Signal Acquisition subsystem consists of two stages. The first stage is the capturing of the raw audio signals recorded from each microphone. The second stage is the preprocessing of the raw audio signals into processed audio signals that can be used by the other subsystems.

4.2.1.1 Signal Capture

The Signal Capture stage was implemented using a local network connecting the host device with the Raspberry Pi microcontrollers. A collection of bash scripts was created to initiate the recording of the audio signals and the transmission of these from the Raspberry Pi microcontrollers to the host device. The bash scripts would transmit a record instruction to both of the Raspberry Pi microcontrollers via a Secure Shell (SSH) connection. Once the recordings were completed, they were retrieved via the Secure Copy Protocol (SCP). Initially, a parallel-SSH was to be used to start the recording on both RPi microcontrollers simultaneously. It was found, however, that parallel-ssh was not compatible with the task and, therefore, an '&' character was used between the 2 Raspberry Pi scripts. The syntax and logic of these bash scripts will be discussed further in the Data Collection and Analysis section.

4.2.1.2 Signal Preprocessing

The microphones and algorithms used were also found to be incredibly sensitive to noise. What was not accounted for in the previous milestone, was the presence of a large amount of low frequency noise within the system. As such, a bandpass filter was implemented in place of a lowpass filter. The processing of each signal consists of the following stages:

1. The raw audio signal is split in two halves.
2. The first half is designated as the calibration signal.
3. The second half is designated as the source signal.
4. A Hann window is applied to each half.
5. Each half is filtered by a Butterworth bandpass filter with its corresponding frequency band.

6. Each half is normalised to have a maximum magnitude of 1.

With regards to challenges, it was found that the frequency bands of the source and calibration signals chosen as well as the order of the bandpass filter severely impacted the results obtained, as it could either let too much noise pass through, or it could distort the signal, leading to inaccurate results. This was addressed by testing the performance of the algorithms these quantities were varied, to determine their optimal values. This is discussed further in the Results section.

4.2.2 Synchronisation

The Synchronisation subsystem was implemented using a calibration signal as previously stated. Furthermore, the Network Time Protocol (NTP) was also implemented to determine if it would sufficiently synchronise the recordings. The frequency range of the calibration signal was moved to 6-10 kHz to accommodate the change in the source signal frequency to 1-5kHz. A dead band of 1kHz was kept between the signals to reduce the chance of frequency aliasing. A challenge with synchronisation was separating the calibration and source signals from the recordings. Thus, to solve the issue, time-slicing was implemented. The 16 second recording was divided into two halves. The first half was allocated to the calibration signal and the second half was allocated to the source signal. The synchronisation delays are estimated using the GCC-PHAT algorithm. The calibration signal acquired from Mic 1 was used as the reference signal for the cross correlation with the calibration signals from each Mic. This includes the correlation of the calibration signal of Mic 1 with itself to ensure that a delay of zero is obtained. The synchronisation delays calculated were then used to compensate for the desynchronisation between the signals from different Mics.

4.2.3 Time Delay Estimation

The Time Delay Estimation Subsystem was implemented using the GCC-PHAT algorithm. The time delay estimation was done using GCC-PHAT cross correlation of the source signal from Mic 1 with the source signals from each of the other Mics, including Mic 1. Mic 1 was included to detect any obvious errors in the time delay estimation. This process produces the estimated ToA delays of the audio signals at each Mic with respect to Mic 1. However, these ToA delays include the synchronisation delays, and this must be removed to produce accurate TDoA values. The TDoA values are calculated according to the following equation:

$$TDoA_{1i} = (ToA_1 - ToA_i) - (Sync\ Delay_1 - Sync\ Delay_i)$$

This equation is used to produce synchronised TDoA values for Mics 2, 3 and 4 with respect to Mic 1. The only significant change made to the physical implementation compared to the simulated implementation is that the synchronisation delays were compensated for by calculation rather than shifting the signals as this proved to be much simpler.

With regards to challenges, if the synchronization delays were extremely inaccurate, then the TDoAs will also be extremely inaccurate. Proper synchronization between the audio signals was, therefore, ensured to improve the accuracy of the TDoA.

4.2.4 Triangulation

Triangulation is done using a non-linear least squares estimation. The nonlinear LSE algorithm used is the trust-region-reflective algorithm that was used in the simulation implementation. This is implemented using the Python SciPy library curve_fit function which is equivalent to the MATLAB lsqcurvefit function used in the simulation implementation. As such, no real challenges emerged when implementing this subsystem. A function was defined that takes in the coordinates of the sound source and the positions of the Mics and outputs the differences in the distances from Mic 1

to the source and the other Mics to the source. The curve_fit function then estimates the source position using the estimated TDOA values multiplied by the speed of sound as the distances. Furthermore, this is done by iterating an initial position until if matches the expected distance values. The significant change from the simulation implementation was that a linear LSE was added to produce the initial position from the TDoA values and Mic positions. This is done instead of using the centre of the grid (0.4, 0.25) as the initial point as was done in the simulation interpretation. Additionally, a lower and upper bound are placed on the solution. The lower bound and upper bound coordinates were set to 0.001m greater than the grid size, so that estimates outside the grid could be detected and correctly rejected.

4.2.5 Graphical User Interface

Since the code was no longer in MATLAB but now in Python, a Python GUI had to be implemented to work with the other programs such as the triangulation, time delay estimation and signal acquisition programs. The “PySimpleGUI” library was used to create the GUI. Though previously the “Tkinter” library was considered initially, “PySimpleGUI” was found to be much easier to implement and contains much fewer lines of code for lower latency. The GUI consists of six buttons, a grid and 5 lines of text, consisting of the estimated synchronisation delays, estimated TDOA delays and other estimated values. The initial startup GUI setup can be seen below:

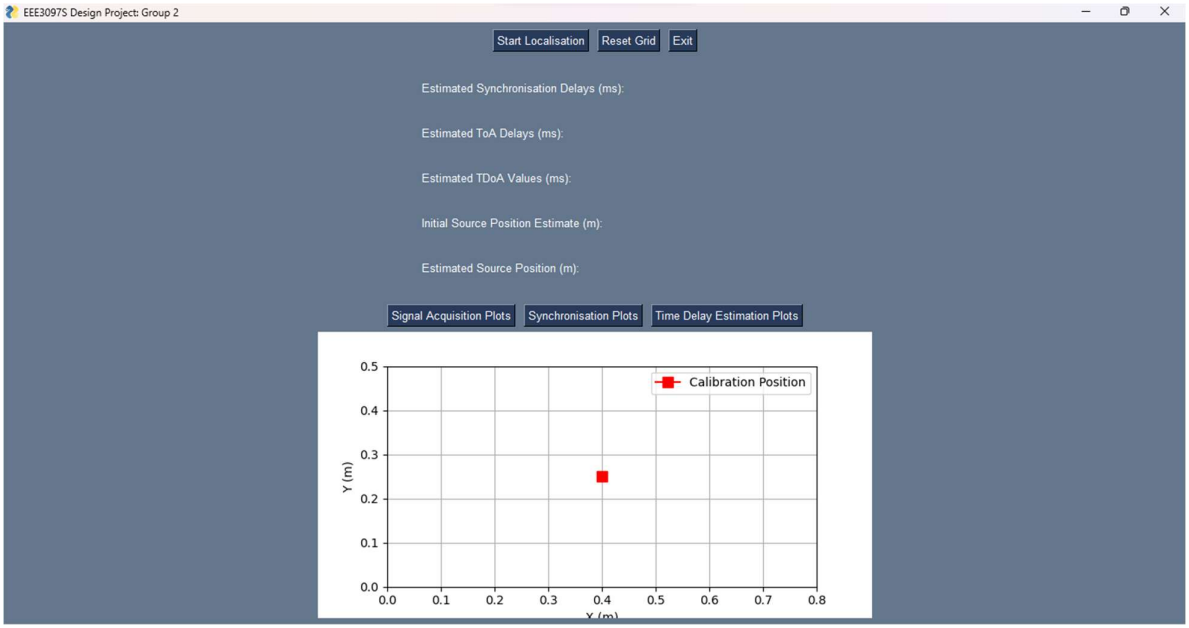


Figure 1: Project GUI

The ”Start Localisation” button calls an SSH command to tell the Raspberry Pis to start recording the sound. Thereafter, the data obtained from the Pis is analysed by the respective TDOA and triangulation programs and displayed on the GUI. The estimated coordinates are also plotted on the grid. The screenshot below shows the displayed information for the approximate coordinates (0.55, 0.45):

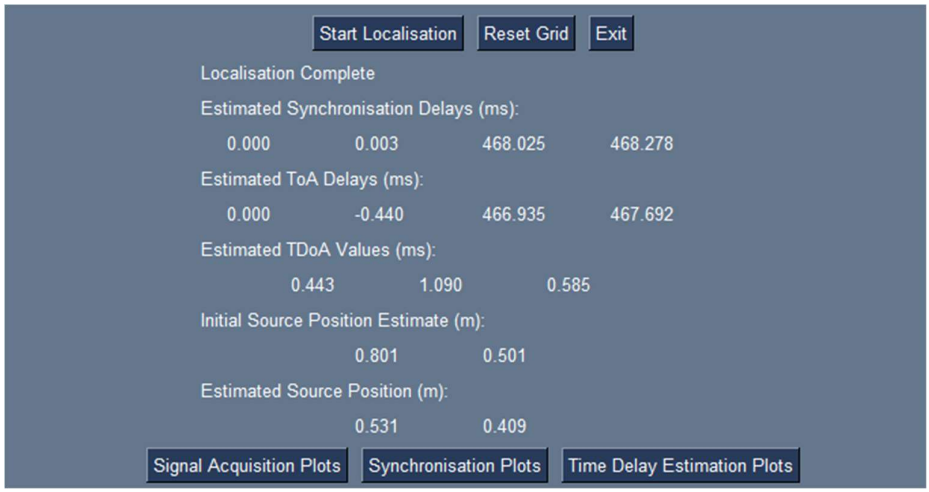


Figure 2: GUI TODA calculation outputs

The image below shows the grid, which contains the source signal positioned at (0.55, 0.45):

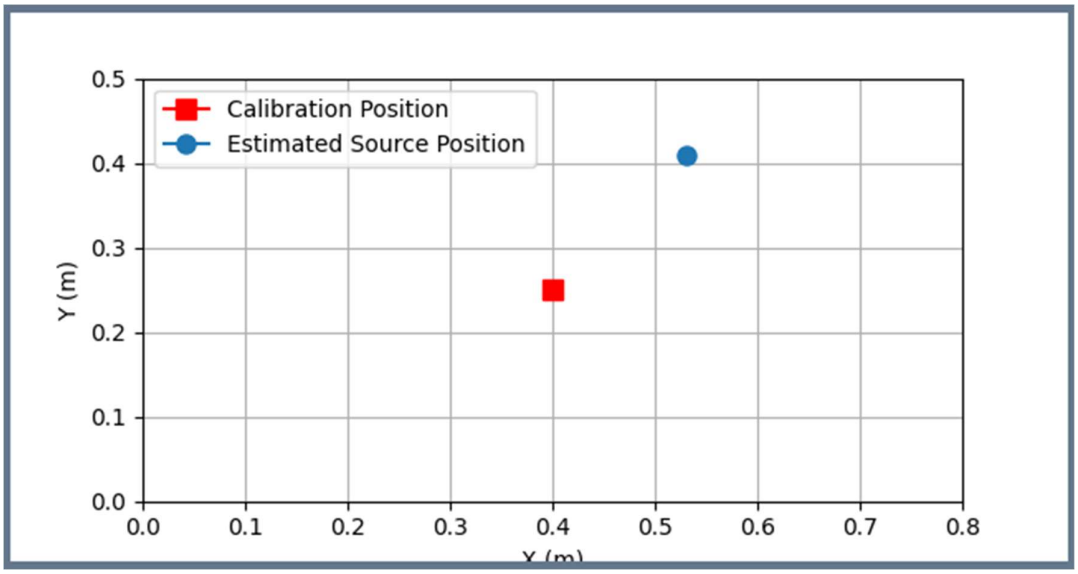


Figure 3: Displaying the source position with an input source position of (0.55,0.45)

The GUI also has buttons to display the signal acquisition plots, the synchronisation plots and the time delay estimation plots. For the example above the image below shows the signal acquisition plots:

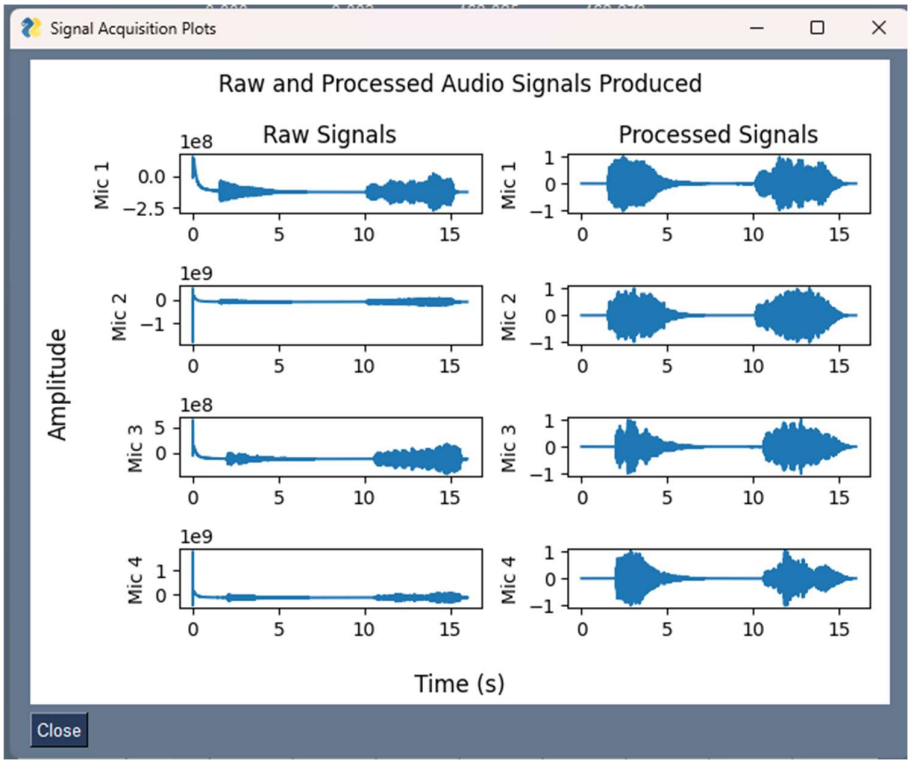


Figure 4: Signal Acquisition plots associated with the input source signal at position (0.55,0.45)

The image below shows the synchronisation plots for the same example:

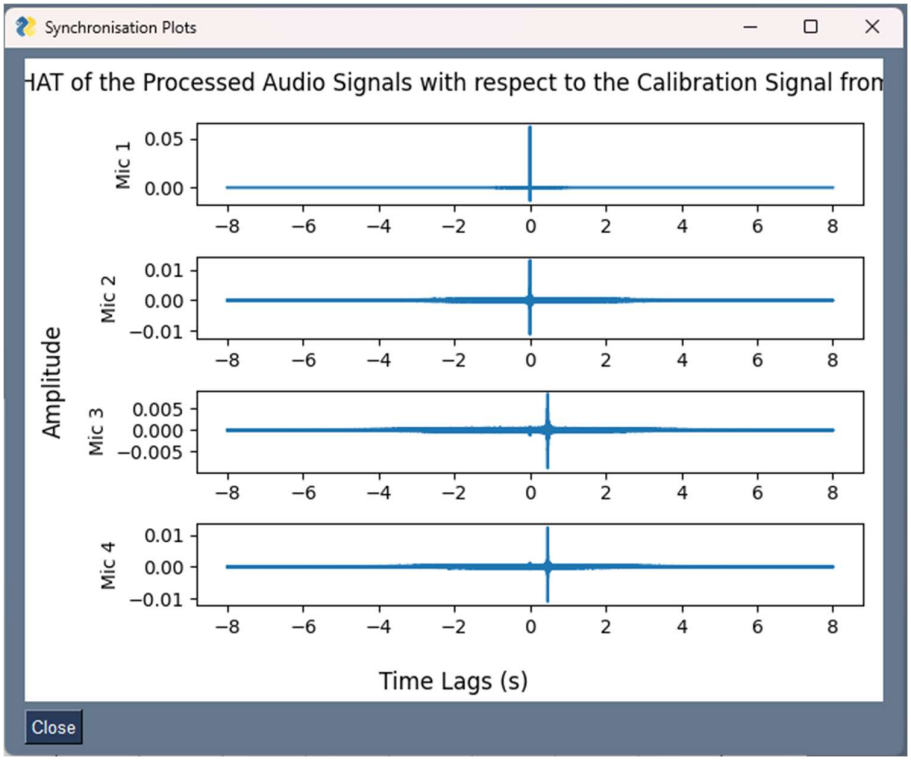


Figure 5: Synchronisation plots for an input source signal at position (0.55,0.45)

Finally, the image below shows the time delay estimation plots:

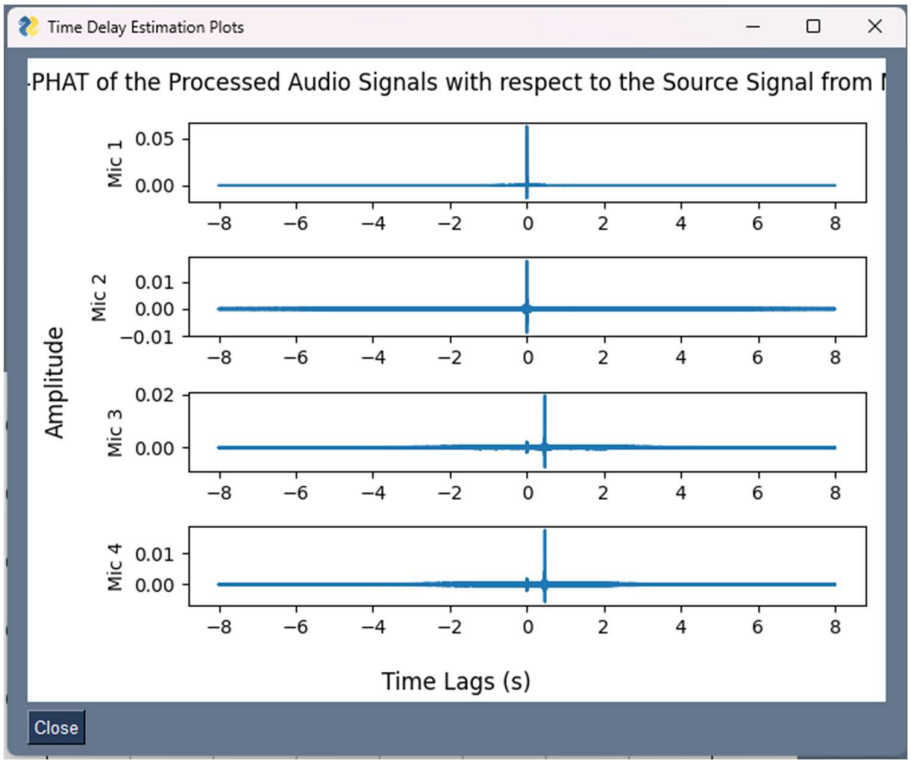


Figure 6: Time Delay Estimation plots for an input source signal at position (0.55,0.45)

The GUI contains a ‘’Reset” button, which clears the grid, allowing for the location of multiple different locations without having to restart the program.

There were multiple challenges faced in this section, with the most noteworthy one being the difficulty in positioning of GUI elements as compared to the MATLAB App Designer. Since the app is entirely coded, to position an item to one’s preference would require coordinate inputs for each element. Furthermore, the app experienced cross-compatibility issues, where the app would work very well on a certain laptop but not as well on another. There were also challenges faced during the creation of the ‘Reset’ button, as it would sometimes resize the graph or delete the gridlines.

5 EXPERIMENTAL SETUP

5.1 OVERALL SYSTEM

The following apparatus was used for this physical implementation:

- 1 x A1 grid
- 2 x Raspberry Pi Zero W modules
- 2 x Red Mushroom Bluetooth Speakers – Pictured in Figure
- Electrical Tape
- 4 x the Adafruit I2S MEMS microphone breakout boards
- 3 x micro-USBs
- 1 x mini-HDMI adapter
- 2 x Breadboards
- Jumper cables
- Veroboard
- Female Headers
- 1 x USB Hub
- 2 x IPads
- 1 x Mobile phone
- 1 x Asus Vivobook Laptop (Local Host)

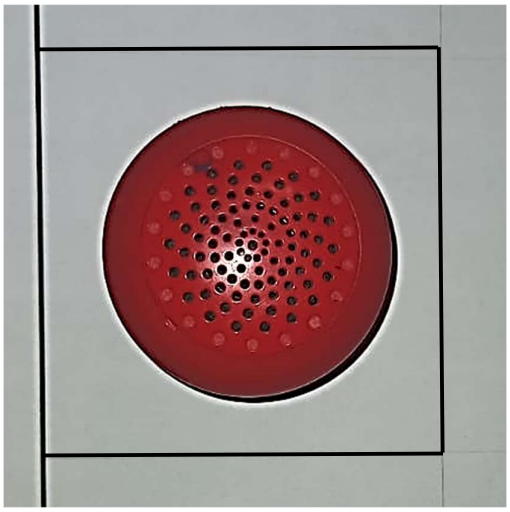


Figure 7: Image of the Mushroom Bluetooth speaker with a grid block as a reference.

In Figure 7, the size of the Bluetooth Speaker can be seen, with reference to a grid block. This provides insight to range of positions, which could be outputted by the triangulation algorithm.

The Microphones were soldered onto Veroboard, along with Female Headers, in order to ensure a reliable connection.

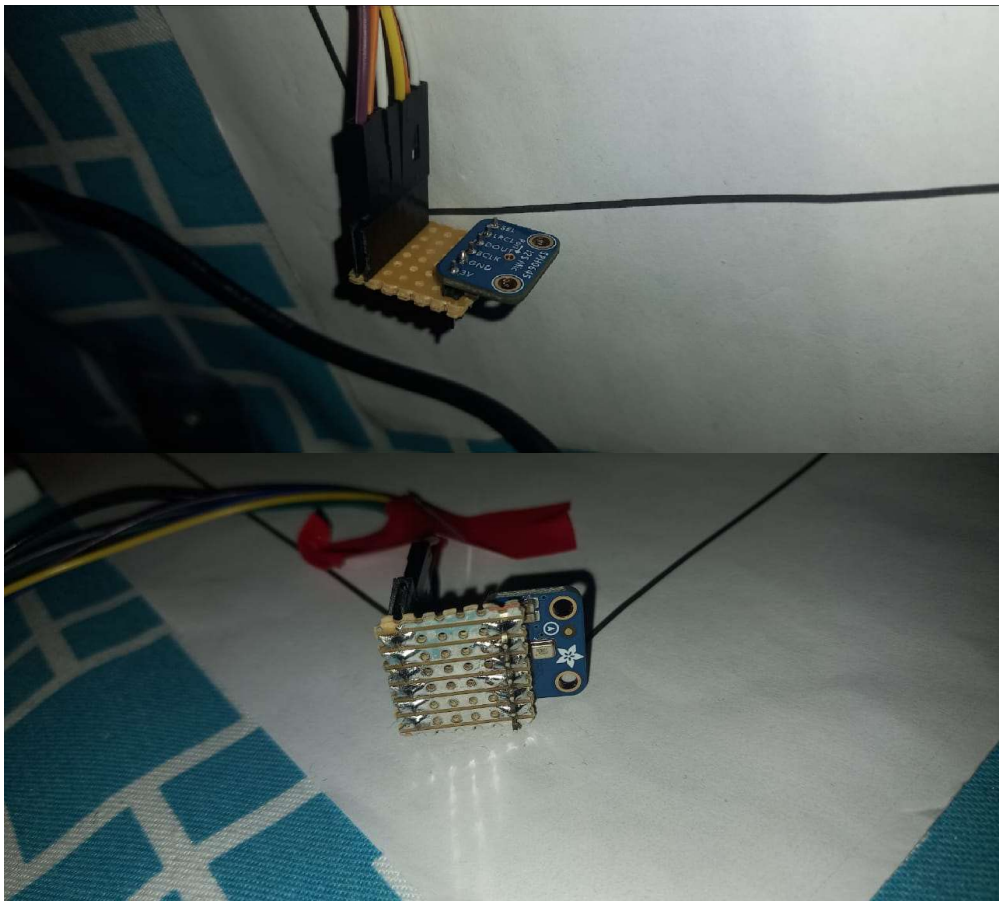


Figure 8: Connections associated the I2S Mics.

2 Microphones were connected to 1 Pi each, using a breadboard. Jumper cables were used to ensure these connections.

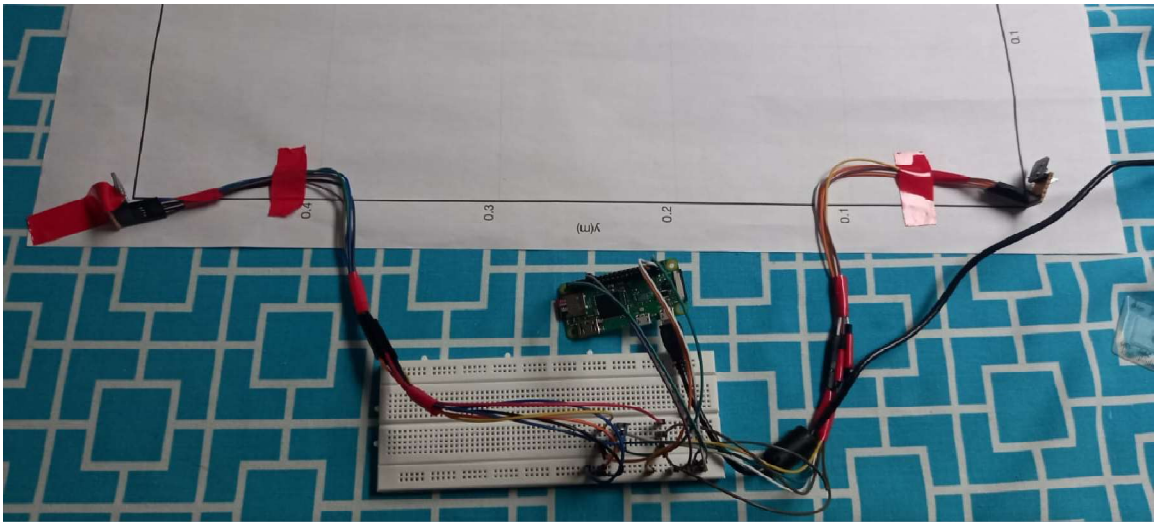


Figure 9: Connections between the RPI modules and the I2S Mics.

The Microphones were placed at each corner and made to point towards the centre of the grid. They were held steady using some electrical tape. Please note that Figure 9 is for demonstration purposes only, as the microphone on the right-hand side is not exactly correctly orientated.

The complete experimental setup is shown below:

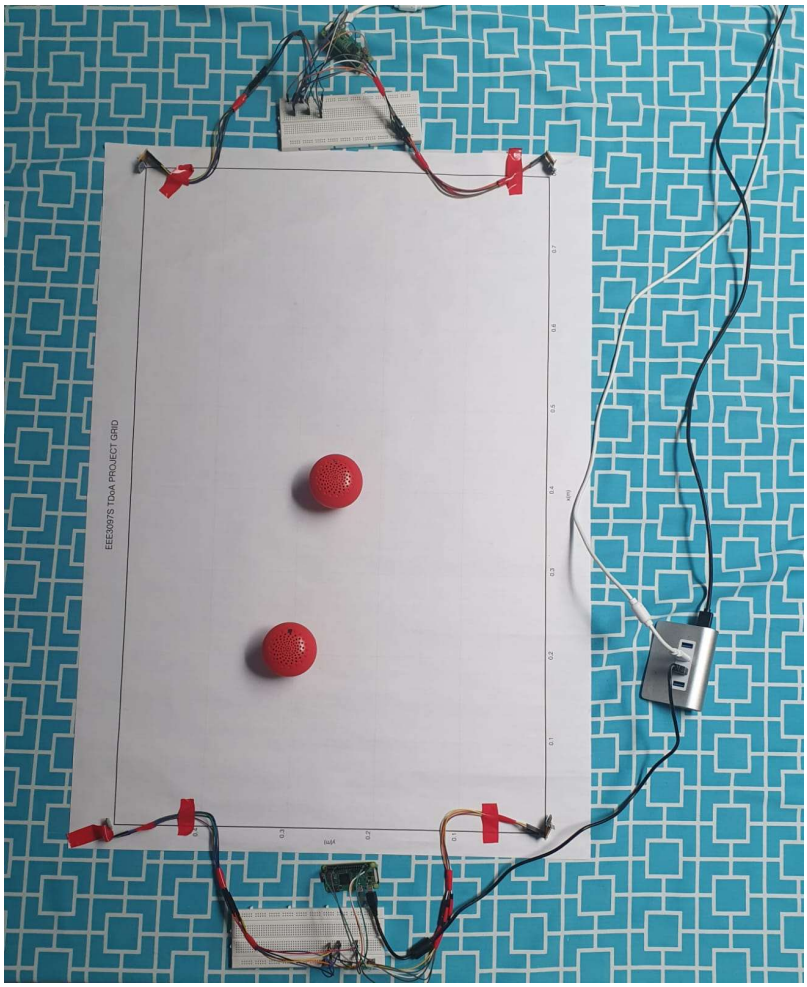


Figure 10: Overall System Experimental Setup.

The Bluetooth Speaker in the centre is responsible for producing the calibration signal, while the other Bluetooth speaker is responsible for producing the source signal.

5.2 Signal Acquisition

Bluetooth Speakers were used instead of mobile devices because it was found that the mobile devices' speakers were not sufficient to produce a loud enough sound required to achieve the level accuracy that was needed. The Bluetooth speakers could also handle low frequencies, which could not be heard through the mobile devices' speakers.

The signal acquisition subsystem was tested in two steps. Firstly, the signals recorded from each mic were obtained. Secondly, the received signals were passed through a bandpass filter to reduce the high and low frequency noise in the signals.

The signal recordings comprised of a source signal and calibration signal. These are both 5 second chirp signals. Chirp signals were selected as they are optimal for computing time delays using cross correlation as mentioned in [1], although the algorithm will also work for regular sinusoidal signals.

As the mics are connected in pairs, a latency difference is evident between the signals from mics 1 and 2 and the signals from mics 3 and 4.

The high pass filter was implemented using the Python 'butter' function. The high pass function uses a 6^{th} order filter. The filter was designed to cause minimal distortion on the signals while still significantly reducing high frequency and low frequency noise.

To showcase the effectiveness of this subsystem, an experiment was run. The performance of the subsystem was tested by varying individual signal and filter parameters and recording the effects on the overall system accuracy.

5.3 SYNCHRONISATION

The frequency range of the calibration signal was moved to 6-10 kHz to accommodate the change in the source signal frequency to 1-5kHz. A dead band of 1kHz was kept between the signals to reduce the chance of frequency aliasing. As discussed in the Subsystem Implementation section, time-slicing was implemented. The audio recordings were each divided into two halves. The first half was considered the calibration signal, and the second half was considered the source signal. The synchronisation delays are estimated using the GCC-PHAT algorithm. The calibration signal acquired from mic 1 is used as the reference signal for the cross function. The calibration signal of mic 1 was cross correlated with itself to ensure that a delay of zero is obtained.

Additionally, to test further this subsystem, the position of the source was varied. A plot of the synchronised signals was displayed to ensure that the signals were correctly synchronised.

5.4 TIME DELAY ESTIMATION

The time delay estimation is done using GCC-PHAT cross correlation of the signal from mic 1 with the signals from the other mics. This gives an estimate of the TDOA values between mic 1 and the other mics. This assumes that the signals have already been correctly synchronised, and the calibration signal has been removed. No additional equipment or components were used for this subsystem, as it was entirely software based.

For the testing of this subsystem, the synchronised signals from the synchronisation experiment ran previously were the input signals to the GCC-PHAT cross correlation function. A plot of the GCC-PHAT cross correlations of the signals was taken to display the results.

Additionally, the TDOA values were also recorded for all the tests performed on the overall system and displayed on the GUI.

5.5 TRIANGULATION

Triangulation is done using a non-linear least squares estimation. The default trust-region-reflective algorithm of the MATLAB lsqcurvefit function is used. A function is defined that takes in the coordinates of the sound source and the positions of the mics and outputs the differences in the distances from mic 1 to the source and the other mics to the source. The lsqcurvefit function then estimates the source position using the estimated TDOA values multiplied by c as the distances. Furthermore, this is done by iterating an initial position until if matches the expected distance values. The centre of the grid (0.4, 0.25) is used as the initial position as it is closest to all positions on the grid. Additionally, a lower and upper bound are placed on the solution. The lower bound is at the origin and the upper bound is at the maximum coordinates of the grid.

The default parameters were used in the experimentation of this subsystem, along with the position of the sound source being set to (0.8,0.5), which is at the far-right corner of the grid. The iterations of nonlinear least squares estimation algorithm were plotted to illustrate the steps taken to reach the final estimated source position.

5.6 Graphical User Interface

In order to visualize the process more clearly, a Graphical User Interface (GUI) was created to showcase the accuracy and precision of the overall system, as well as the effectiveness of each of the various subsystems involved in the overall system. From Figure 6 buttons can be seen.

- 'Start Localisation Button' – Starts the recording process and, thereafter, calculates the estimated position.
- 'Reset Grid' – Clears the grid of all predictions.
- 'Exit' – Closes the GUI
- 'Signal Acquisition Plots' – displays the signals before being processed as well as after being processed.
- 'Synchronisation Plots' - displays the signals before and after being synchronised.
- 'Time Delay Estimation Plots' - displays the cross-correlation outputs obtained by the gcc-phat function.

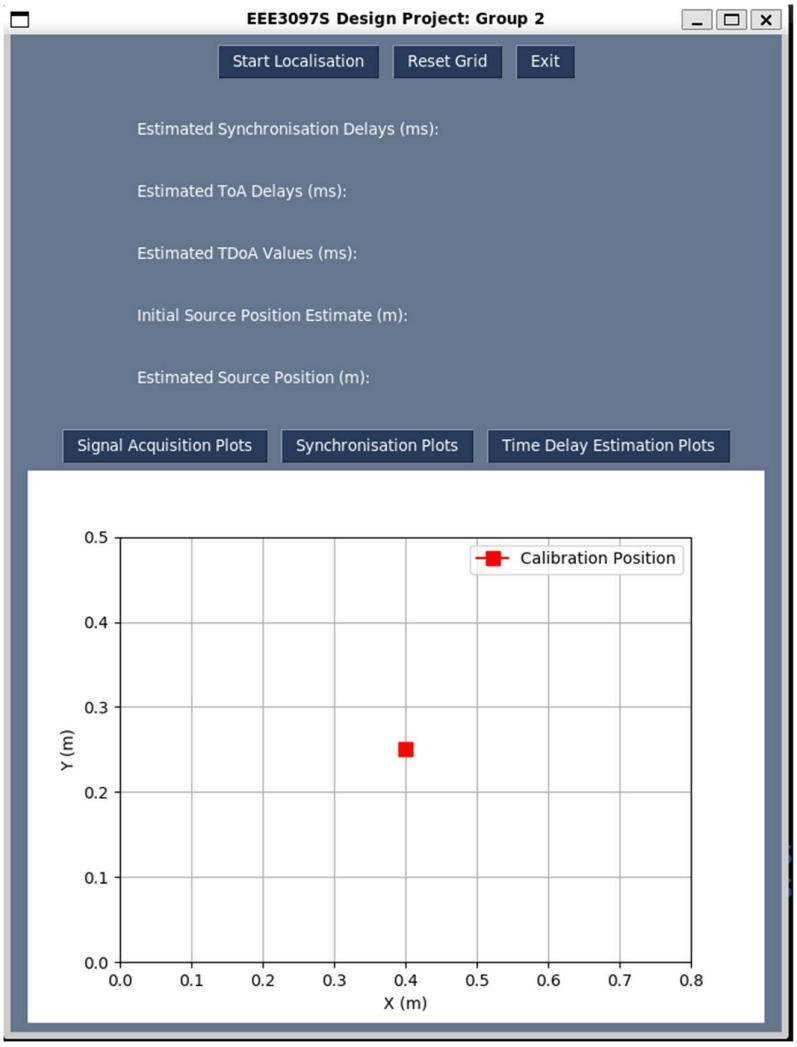


Figure 11: Experiment GUI

The GUI also comprises of a 0.8x0.5 grid, which emulates the A1 grid. The Calibration Signal is plotted at runtime, denoted as a red square. The user then presses the button labelled “Start Localisation”. This starts the recording of the microphones and, thereafter, calculates the estimated coordinates. In this case, the recorded sound was located at the point (0.4,0.4).

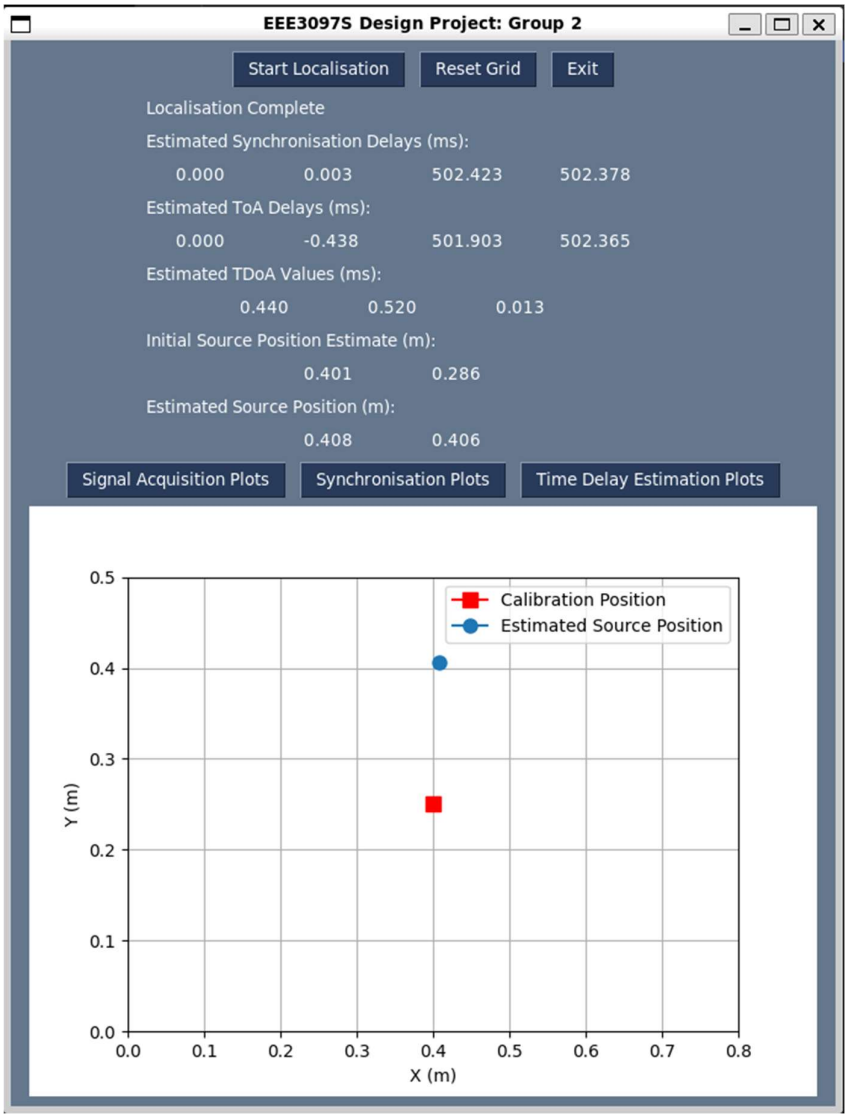


Figure 12: Showing the Actual and Predicted position on the grid.

The Sound Source location is thereafter displayed on the grid as a dot. The provided labels display the actual numerical values, the position error, and the TDOA estimation obtained when calling the respective Python functions.

The simulation also accounts for invalid coordinates provided, e.g, the coordinates are outside the bounds of the grid. The sound source was located at the point (0.4,0.6).

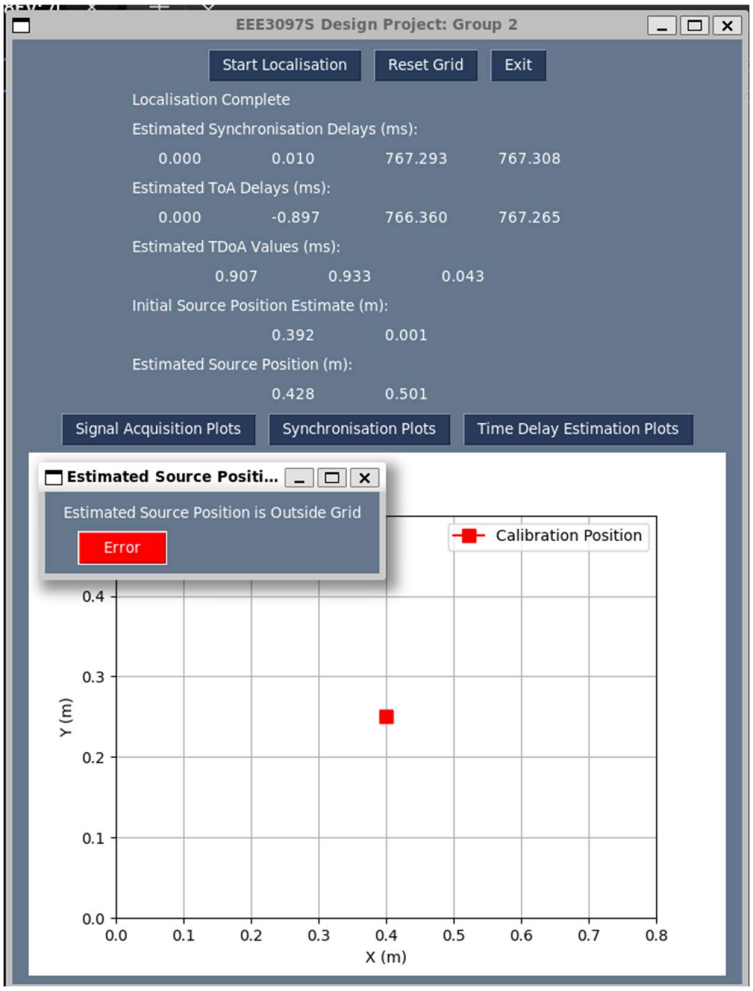


Figure 13: Error message when the coordinates are out of bounds.

The triangulation function will only generate a maximum x-coordinate of 0.801 and a maximum y-coordinate of 0.501.

6 DATA COLLECTION AND ANALYSIS

6.1 PROCESS FOR COLLECTING ACOUSTIC SIGNALS

As expressed in the System Implementation section, a series of bash scripts were used to record and store the signals for analysis. Both Raspberry Pis contained the bash script shown in Figure.

```
#!/usr/bin/bash

# Define the output file name
OUTPUT_FILE="recording.wav"

# Use arecord with the specified options
arecord -D plughw:0 -c2 -r 25000 -f S32_LE -t wav -V stereo -v -d 16 "$OUTPUT_FILE"

# Check if the recording was successful
if [ $? -eq 0 ]; then
    echo "Recording completed successfully."
else
    echo "Recording encountered an error."
fi
```

Figure 14: raspberrypi.sh script.

This bash script handles the recording of the audio signals. The 'arecord' command is set up to record in stereo mode for 16s, at a sampling frequency of 25000Hz. The output file is amended based on which RPI module the script is on.e.g. on Raspberry Pi 1, the output file was renamed to 'recording_1.wav'. The message "Recording completed successfully" is displayed when the recordings are finished.

The bash script shown in Figure 15, is named pi_one.sh and is used to ssh into Raspberry Pi 1. Since the Pi is password protected, 'sshpass', along with the password 'group2' as an argument is used to speed up the process, to prevent the program from requesting a password. The only differences between pi_one.sh and pi_two.sh is that pi_one.sh has a 'sleep' command for 3 seconds, and they each contain their respective IP addresses.

```
#!/usr/bin/bash
sleep 3
sshpass -p 'group2' ssh group2@192.168.242.97 'bash -s < raspberrypi.sh'
```

Figure 15: pi_one.sh script.

```
#!/usr/bin/bash
sshpass -p 'group2' ssh group2@192.168.242.43  'bash -s < raspberrypi.sh'
```

Figure 16: pi_two.sh script.

In order to implement both of these scripts at the same time, on the local host, the bash script shown in Figure 17 is executed. This script is named main.sh. The line "./pi_one.sh & ./pi_two.sh" starts the execution of both those bash scripts at relatively the same time. There is, however, a small delay between the execution of pi_one.sh and pi_two.sh. The next two lines are executed once the recordings are completed. These 2 lines copy the recording files onto the local host via SCP and sshpass.

```
#! /usr/bin/bash

./pi_one.sh & ./pi_two.sh

sshpass -p 'group2' scp group2@192.168.242.97:recording_1.wav .
sshpass -p 'group2' scp group2@192.168.242.43:recording_2.wav .
```

Figure 17: main.sh script.

The main.sh script is ultimately executed in the Python program as shown in Figure .

```
subprocess.run("bash main.sh", shell=True)
```

Figure 18: Script implementation in Python program.

The corresponding TDOA measurements are obtained through the synchronisation and TDOA algorithms expressed in the Experimental Setup section, denoted as 'Synchronisation' and 'Time Delay Estimation'.

6.2 Preliminary Analysis or Insights gained from the Collected Data.

It was found that there was a slight 'pop' at the beginning of the microphone recordings. This proved to cause inaccuracies in the estimated positions.

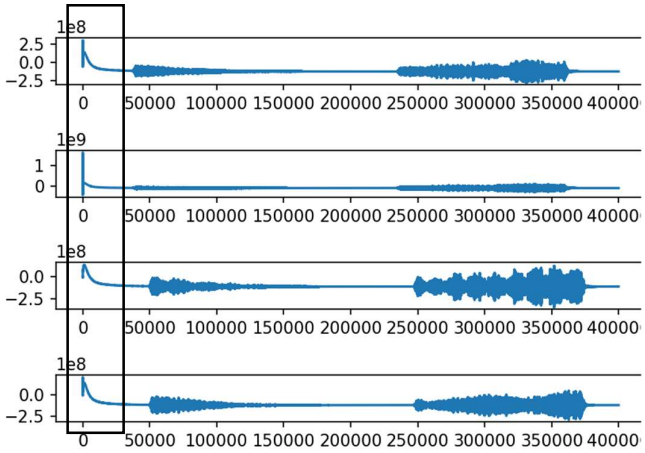


Figure 19: Original, raw audio recordings, illustrating the 'pop' at the beginning of the recording.

This is portrayed in Figure 19. In order to eliminate this 'pop', preprocessing procedures were conducted and were discussed in the System Implementation section, under the Signal Acquisition subheading.

It was also noted that the time between the calibration signal and the source signal should be at least 3s, in order to avoid clipping when the audio recordings are clipped in half. As such, in order to eliminate as much noise as possible, the volume of the audio signals would have to be increased to a maximum.

It was also observed that an excess in noise caused the start of the source signal to become indistinguishable from that noise, which yielded inaccurate results.

7 RESULTS

7.1 OVERALL SYSTEM

The performance of the overall system was characterized by the position error the estimated source positions with respect to the actual source positions. The position error was defined as the distance - in metres - between the actual position and the estimated position.

7.1.1 Default Parameters

The result of the overall system performance at default parameters is given in the figure below:

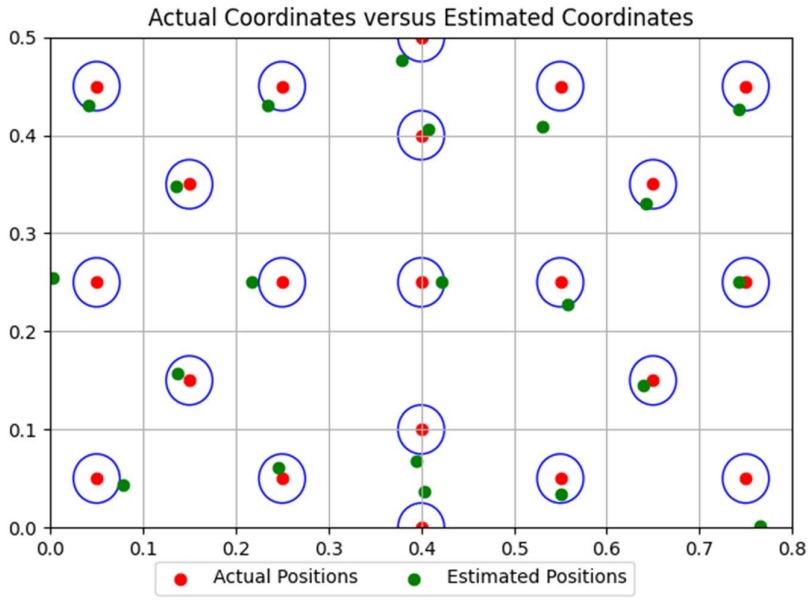


Figure 20: Graph of actual coordinates versus estimated coordinates

It can be observed that Figure 20 illustrates the accuracy of the system. Additionally, the blue circles depict the footprint of the speakers used. The size of the speakers represents an uncertainty in the actual position of the source signal. The use of smaller sized speakers would decrease this uncertainty, however there were no other speakers available for use. Figure 20 shows that 13 of the 21 positions estimated lie within the extent of the speakers. Moreover, most of the estimates that are not within this limit are visually close to it. This suggests that the system functions as intended despite the errors present. Furthermore, the position estimates do not appear to have a consistent error that could be accounted for by a constant offset. Lastly, the position estimates were rounded to a precision of 3 decimal places as this provides millimetre level precision which is the most that can be reliable measured physically. The precision of the estimated positions with respect to the repeated recordings at the same positions was not measured due to time constraints limiting the number of recordings that could be taken.

The position errors of estimated positions of the audio recordings taken are given in the figure below:

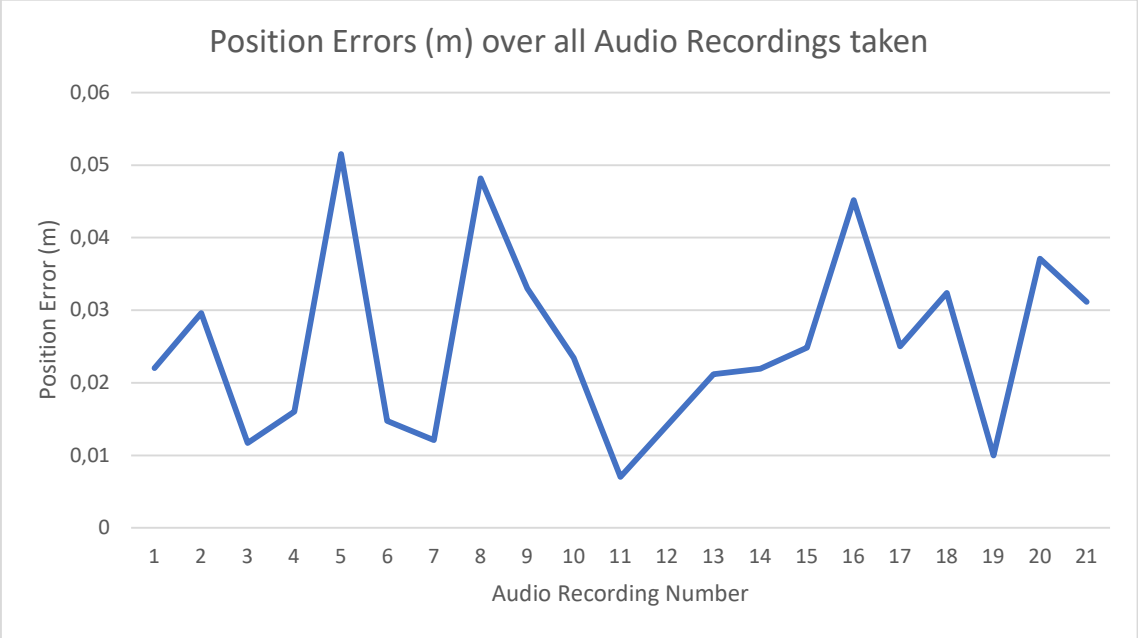


Figure 21: Graph of the position errors for the audio recordings taken.

Figure 21 shows that the position errors can fluctuate by around 5cm. There appear to be 3 outliers that are much greater than the rest of the values. These correspond to audio recordings 5, 8 and 16 which occur at positions (0.75, 0.05), (0.05, 0.25), (0.55, 0.45) respectively. These points are all close to the edge of the grid and which suggests the system may have degraded performance at the edges.

The statistics of the position error of the overall system are given in the table below:

Mean Position Error (m)	0.025
Median Position Error (m)	0.023
Minimum Position Error (m)	0.007
Maximum Position Error (m)	0.052
Standard Deviation of Position Error (m)	0.013

Table 1: Statistics of the Position Error of the Overall System

The statistics can be more intuitively illustrated by the Box and Whisker plot below:

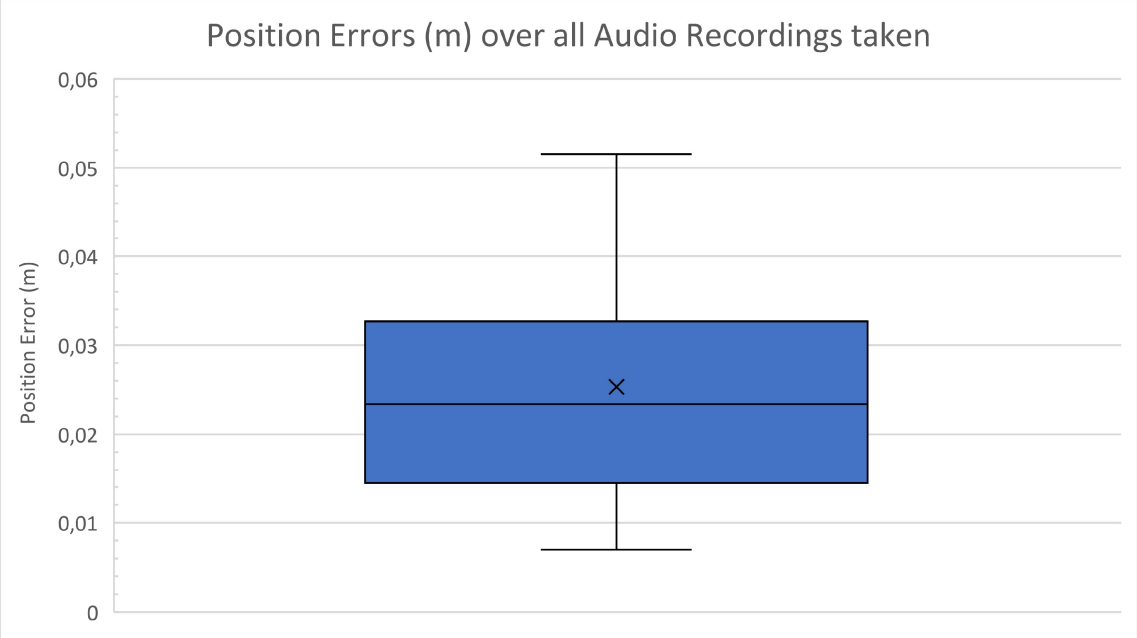


Figure 22: Box and Whisker Plot of the Position Errors of the Overall System

Figure 22 illustrates the distribution of the position errors over the recordings taken. Furthermore, it shows that 75% of the recordings produced a position error less than approximately 0.032m. This represents a significant majority of the recordings taken. And it is only 0.007m or 7mm greater than the extent of the speakers used. This further supports the observations made about figure.

The physical system can be compared to the simulated system to determine how closely its performance matches the performance predicted by the simulation. The performance of the simulated system is shown in the figure below:

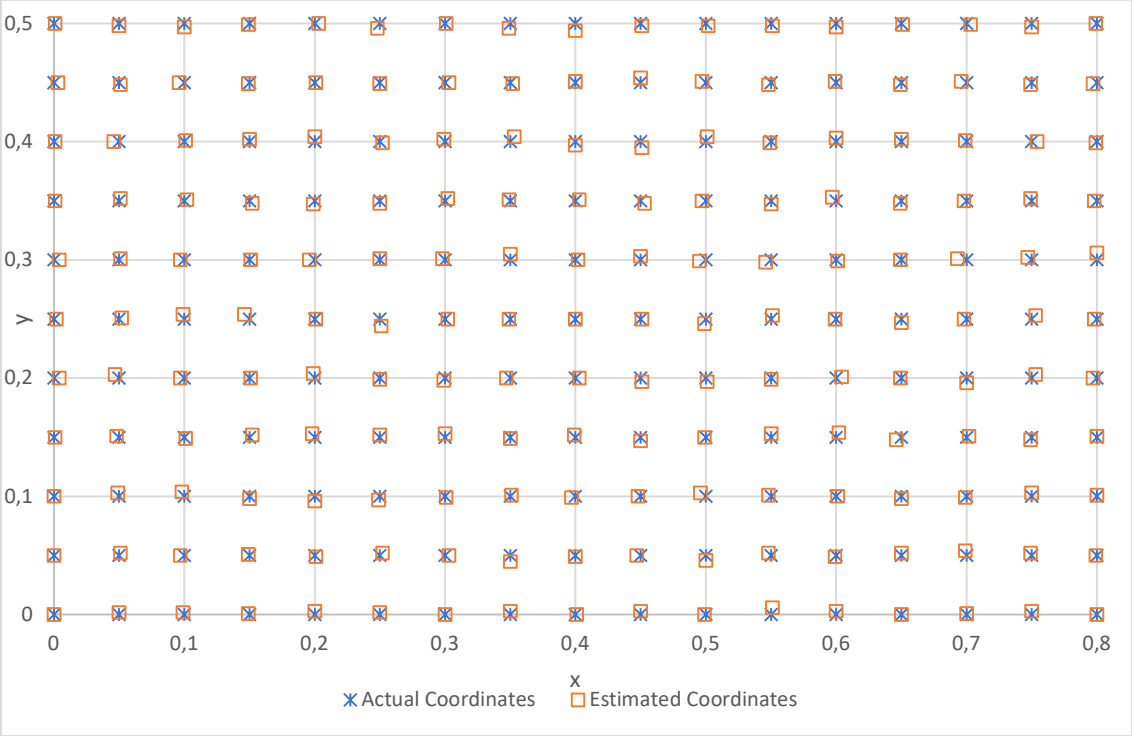


Figure 23: Simulation Results.

This shows that the results of the physical implementation are significantly worse than simulated. This likely due to the simplifications made in the simulation that do not take some of the physical non-ideal conditions into account. For example, the simulation was performed using signals with constant amplitude and only White noise was considered. Thus, this shows the significance of the non-ideal parameters. Improvements to the system implementation can be made by focusing on these areas.

As another point of comparison, the statics of the position error of the simulated system are given in the table below:

Mean Position Error (m)	0.003
Median Position Error (m)	0.002
Minimum Position Error (m)	0
Maximum Position Error (m)	0.007
Standard Deviation of Position Error (m)	0.001

Table 2: Statistics of the Simulation Position Error of the Overall System

These statistics suggest that the physical system performs approximately 10 times worse than expected from the simulations. This implies that there is significant room for improvement by optimizing the system to account for the non-ideal effects. This would likely involve more sophisticated signal processing techniques such as Kalman filtering which could not be applied due to time constraints.

7.1.2 Varying White Noise Levels

The results of the performance of the overall system with varying levels of background White Noise is shown below:

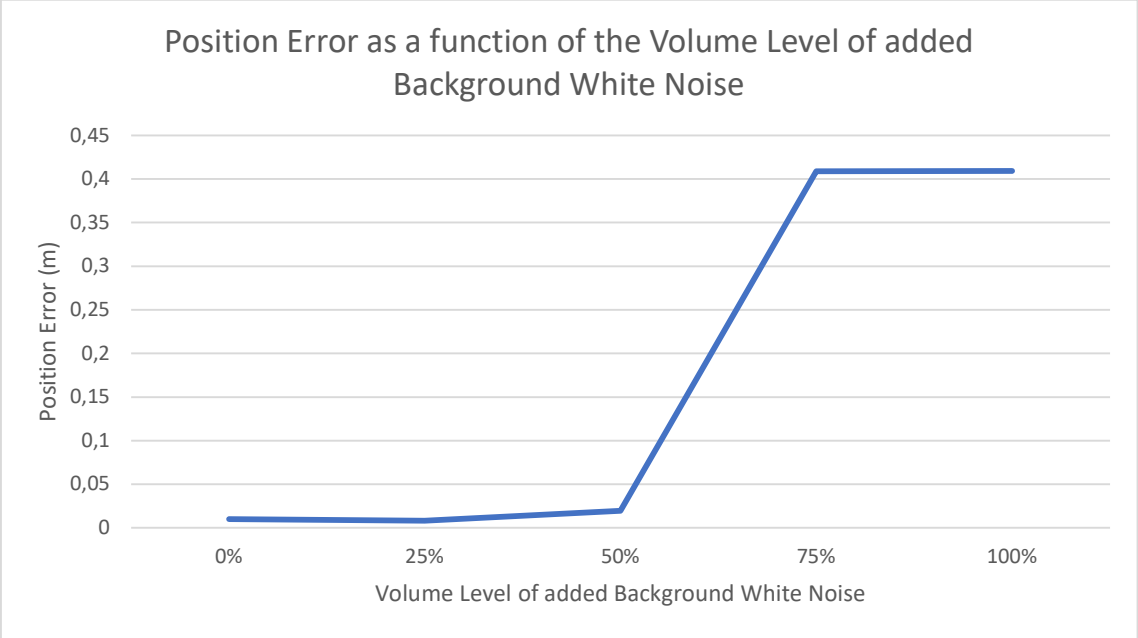


Figure 24: Graph depicting the position error as a function of the volume level of background white noise.

This shows that the system performs optimally for a certain level of noise and is totally inaccurate above a certain level of noise. The threshold lies somewhere between 50% and 75% of the max noise volume produced. Due to technical limitations the volume of noise cannot be meaningfully quantified due to technical limitations in the production of the noise. Thus, only a qualitative analysis is performed on this data. The implications are that the system is strongly sensitive to background noise above this unknown threshold. The threshold could be determined experimentally by measuring the noise signal prior to performing the source position estimation. However, this was not done due to time limitations. Thus, it is suggested to limit the application of the current system implementation to optimal environments with low levels of noise.

7.1.3 Varying Source and Calibration Signal Frequency Bands

The performance of the system for varying source and calibration signal frequency bands is illustrated by the table below:

Source Signal Frequency Range (kHz)	Calibration Signal Frequency Range (kHz)	Position Error (m)
1-2	3-4	0.413
1-3	4-6	0.032
1-4	5-8	0.018
1-5	6-10	0.01
1-2	8-10	0.413

Table 3: Position Error with reference to the Source Signal and Calibration Signal range varied.

The table shows that the position error is inversely proportional to the frequency bandwidths of the source and calibration signals. This suggests that a larger bandwidth maximises the performance of the system. Additionally, the comparison between the first and last values shows that the gap

between the source and calibration signal bands has no noticeable effect on the position error. This is likely because time slicing has been utilised to separate the source and calibration signals. However, a larger gap between their bands could allow for frequency slicing to be implemented which has the potential to be more versatile. This is because the signals would not have to be sent during fixed time intervals which proved to be a challenge to reliably achieve during the recording process.

As a point of comparison, the performance of the simulated system as a function of the maximum frequency of the source signal is shown below:

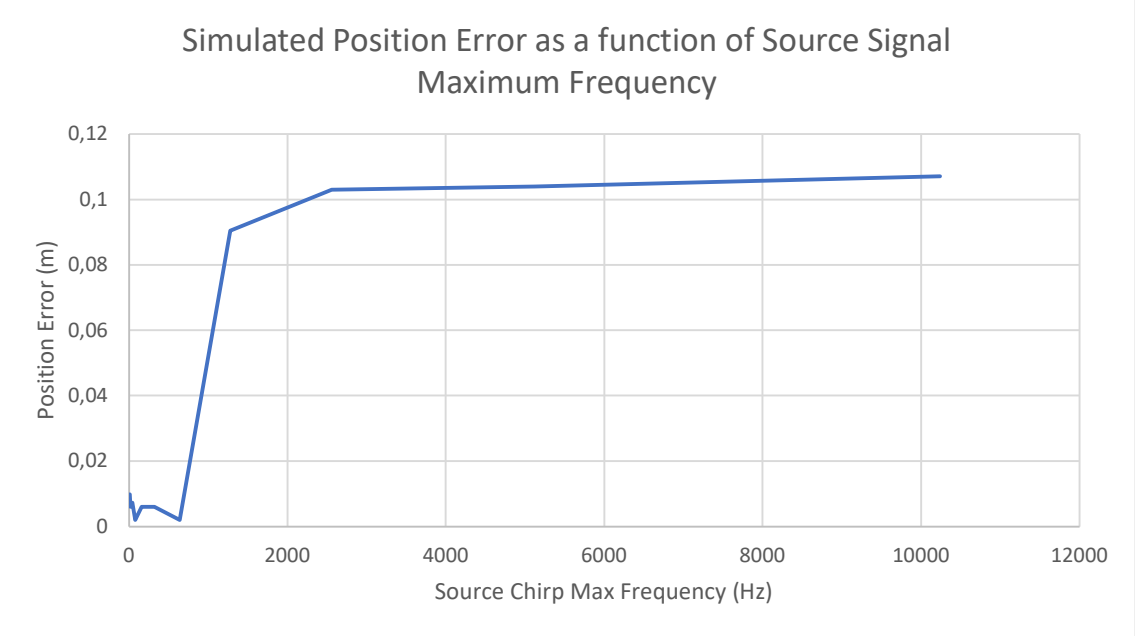


Figure 25: Graph depicting the Simulated Position Error as a function of the source signal maximum frequency.

Since the minimum frequency of the source signal was kept constant the Figure 25 shows the effect of increasing the source signal bandwidth. The simulation results appear to contradict the behaviour of the physical system as the position error increases with the increase of the source signal bandwidth. This is likely due to the differences in the implementation of the simulated system and the physical system. Namely, the frequency of the calibration signal and the separation of the source and calibration signals. The simulation relied on the source and calibration frequencies not overlapping and this condition was broken as the source signal's bandwidth was increased.

7.1.4 Varying Bandpass Filter Order

The performance of the system for variations in the order of the bandpass filter is shown below:

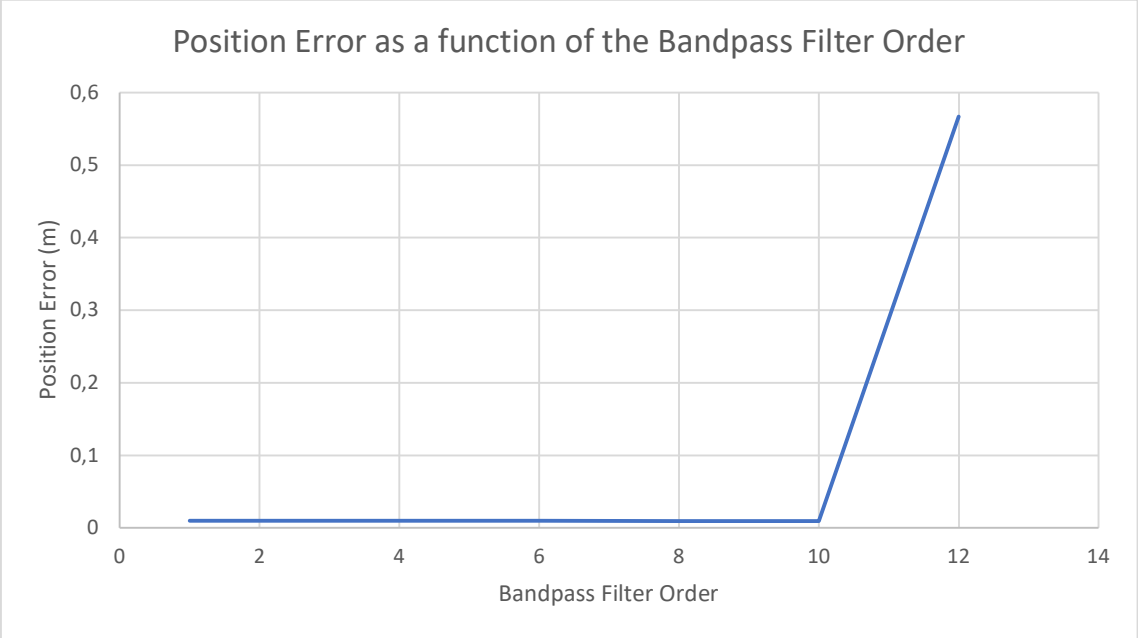


Figure 26: Position Error as a function of the Bandpass Filter Order.

The figure shows that the position error remains constant for orders below 10. Above an order of 10 the bandpass filter appears to introduce too much distortion to the signal. This suggests that filtering with a lower order filter is more resilient to distortions introduced by the bandpass filter. This must be balanced with the effect of the noise outside the frequency bands of the source and calibrations signals. A filter order of 6 was found to function acceptably according to the recordings taken at default parameters.

7.2 Signal Acquisition

The Signal Acquisition subsystem is responsible for capturing and preprocessing the audio signals. The performance of the subsystems preprocessing was used as a measurement for its performance. The SNR of the raw captured signals and processed signals were estimated to assess this performance. It should be noted that the SNR estimated were done by assuming that all the power within the frequency bands of the source and calibration signals are signal power and the rest is noise power. This assumption greatly simplifies the SNR estimation but also overstates the performance of the bandpass filter used to process signals.

7.2.1 Default Parameters

The performance of the Signal Acquisition subsystem is illustrated by the figure below:

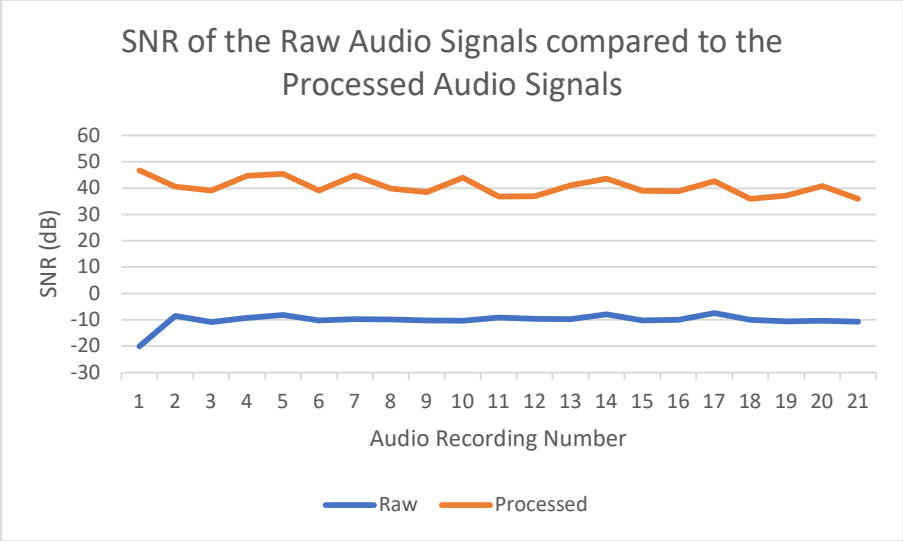


Figure 27: Graph of the SNR of the raw and processed audio signals recorded.

Figure 27 suggests that the Signal Acquisition subsystem performs well as it increases the SNR of the raw signals by approximately 50dB. This is represents a linear increase of 100,000 which is relatively large. However, as stated previously, the SNR measurement used overstated the performance of the bandpass filter. A more accurate measurement would thus be a comparison of the position error to the SNR of the processed signals.

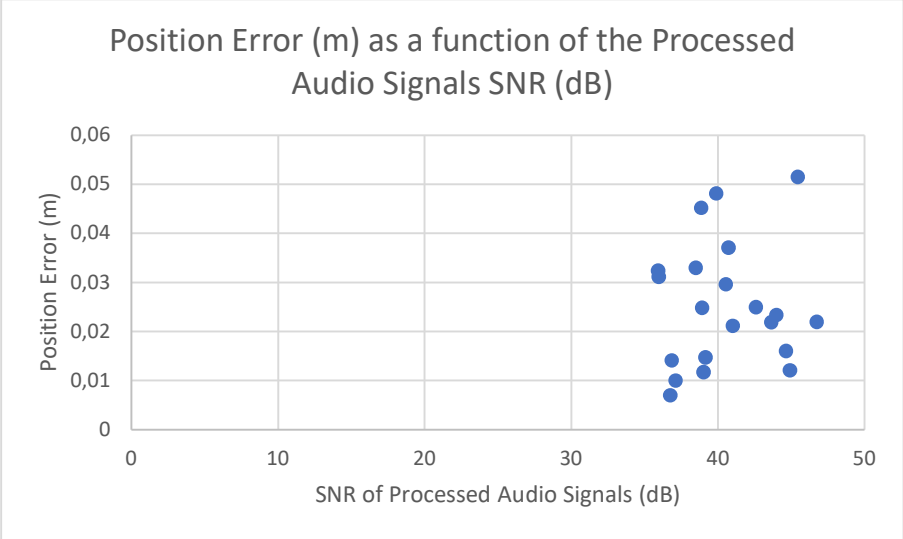


Figure 28: Graph depicting the position error as a function of the processed audio signals' SNR(dB).

Figure 28 shows that there is no notable correlation between the SNR of the processed signals and the position error. Although, this may be due to the limited range of SNR values observed. It is possible that that the system performance would degrade at SNR values below those observed. This will be examined using added background White noise which decreases the SNR below the levels observed ordinarily.

For comparison the performance of the simulated system with varying SNR of the raw audio signals is shown below:

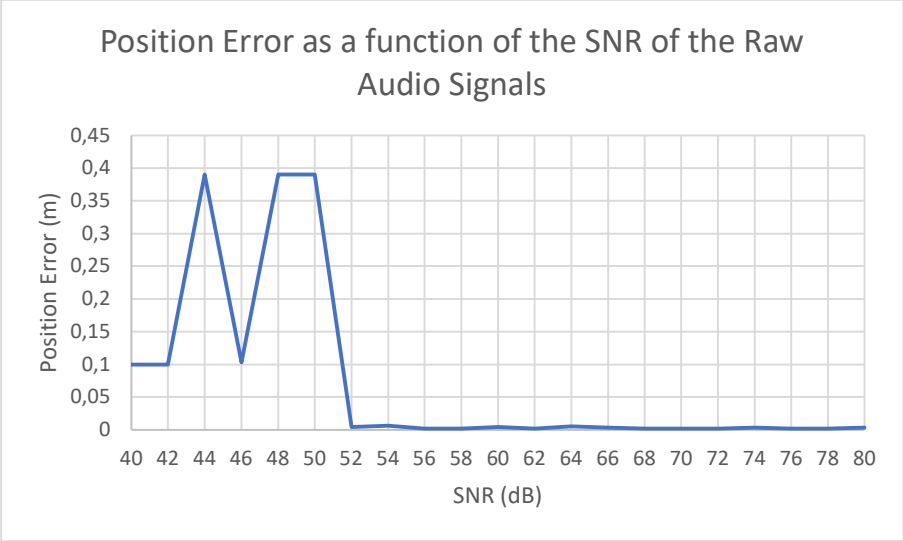


Figure 29: Graph depicting the position error as a function of the raw signals' SNR(dB).

Figure 29 shows that the system performance deteriorates significantly for SNR values below 52dB on the raw audio signals. This does not match the behaviour of the observed system which functions acceptably at raw audio signal SNR values as low as -10dB and processed audio signal SNR values as low as 35dB. This is likely due to differences in the SNR measurement processes. Additionally, the majority of the raw signal noise for the physical system was outside the frequency bands of the source and calibration signals. This means that it could be easily filtered out to achieve a much larger increase in SNR for the processed signals. The SNR of the simulated system was evenly distributed and thus had a larger effect even after filtering.

7.2.2 Varying White Noise Levels

The performance of the Signal Acquisition subsystem for varying levels of background white noise is shown in the figure below:

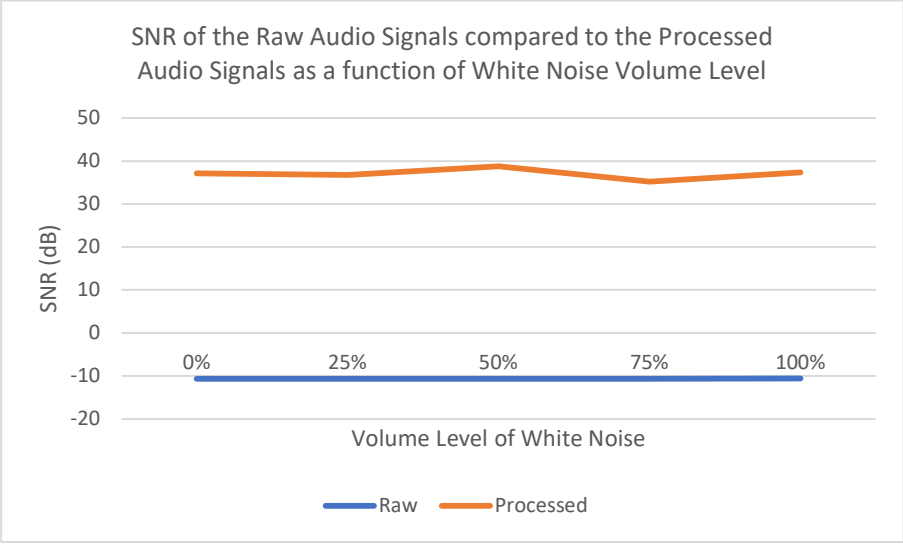


Figure 3020: Graph of position error as a function of the lowpass filter cutoff frequency.

The figure shows that the added background white noise has no noticeable effect on the SNR values of the raw and processed signals. Considering the increase in the position error at the for volume levels above 50% this indicates that the current SNR measurements are inadequate to properly characterise the performance of this subsystem. Thus, a more sophisticated SNR estimation technique is required to assess the performance of the Signal Acquisition subsystem more accurately. This could not be implemented due to technical and time constraints.

7.2.3 Varying the Source and Calibration Frequency Bands

The performance of the Signal Acquisition subsystem for varying frequency bands of the source and calibration signals is shown in the table below:

Source Signal Frequency Range (kHz)	Calibration Signal Frequency Range (kHz)	SNR of Raw Audio Signals (dB)	SNR of Processed Audio Signals (dB)
1-2	3-4	-7.88	34.01
1-3	4-6	-6.78	40.68
1-4	5-8	-9.75	40.69
1-5	6-10	-10,69	37,1
1-2	8-10	-14.83	37.35

Table 4: Signal Frequency Variation and associated SNRs.

The table shows that the bandwidths of the source and calibration signals have no significant or consistent effects on the SNR of the processed audio signals. This seemingly contradicts the observations made about the effects of the bandwidths on the performance of the overall system. This further suggests that the SNR measurements are not sufficiently accurate.

7.2.4 Varying the Bandpass Filter Order

The performance of the Signal Acquisition subsystem for varying values of the bandpass filter order is shown in the figure below:

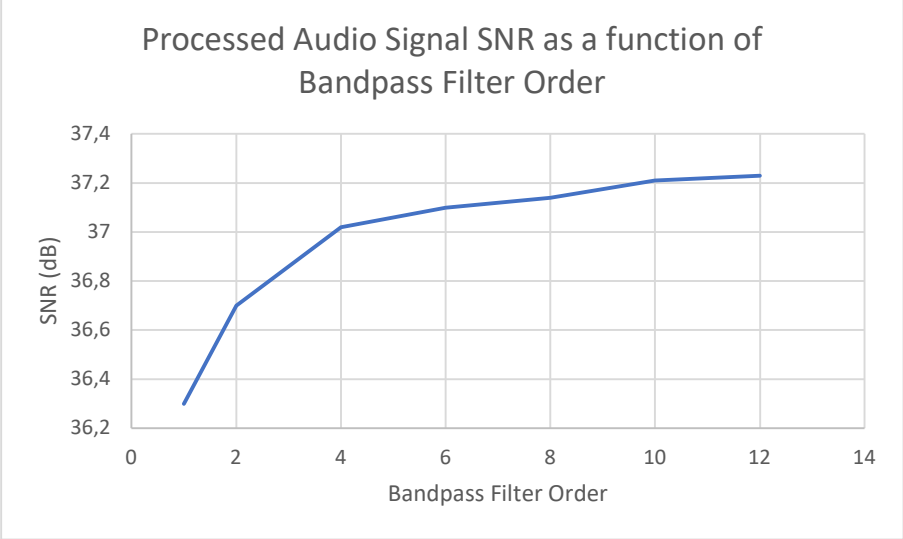


Figure 31: Graph Depicting SNR as a function of Bandpass Filter Order.

The figure suggests that increasing the filter order improves the SNR of the processed audio signals. This matches what is intuitively expected. However, the effect of this increase in SNR greatly diminishes as the filter order is further increased. Furthermore, it was observed earlier that increasing the filter order too much may significantly distort the signals. Thus, it can be seen from

Figure 31 that a filter order from 4 to 8 may be optimal. Thus, the chosen filter order of 6 falls within this optimal range.

7.3 SYNCHRONISATION

The performance of the synchronisation subsystem could not be quantitively measured as there was no way to measure the true synchronisation delay between the audio signals. Thus, the estimated synchronisation delays are compared to what the expected behaviour.

7.3.1 Default Parameters

The estimated synchronisation delays for the audio recordings take are shown in the figure below:

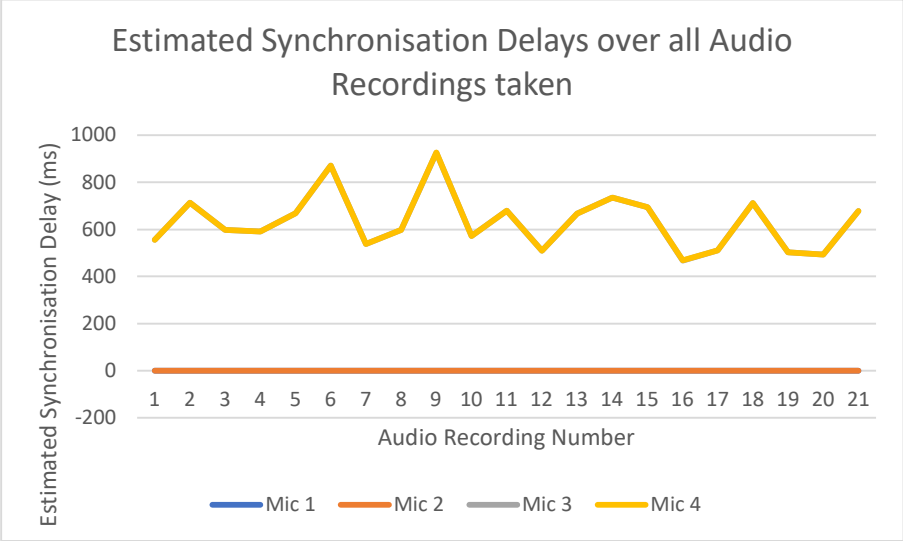


Figure 32: Estimated Delays for all audio recordings.

The figure shows that the estimated synchronisation delays are as expected. Since the calibration signal from Mic 1 is used as the reference the delays are expected to be around zero for Mics 1 and 2. Furthermore, the delays are expected to be some non-zero value that is approximately equal between the signals from Mics 3 and 4. Thus, the subsystem behaves as expected. The figure also illustrates that the synchronisation delays are much greater than 10ms despite utilising NTP. This suggests that NTP synchronisation performed significantly worse than expected. Consequently, the synchronisation could not be achieved solely through the use of NTP. Therefore, implementation of the calibration signal was necessary to achieve acceptable synchronisation between the RPi modules.

The statistics of the estimated synchronisation delays are shown in the table below:

Mean Synchronisation Delay (ms)	632.687
Median Synchronisation Delay (ms)	598.746
Minimum Synchronisation Delay (ms)	468.151
Maximum Synchronisation Delay (ms)	926.025
Standard Deviation of Synchronisation Delay (ms)	120.916

Table 5: Statistics of the Synchronisation Delays.

Additionally, the estimated synchronisation delay statistics are shown visually in the figure below:

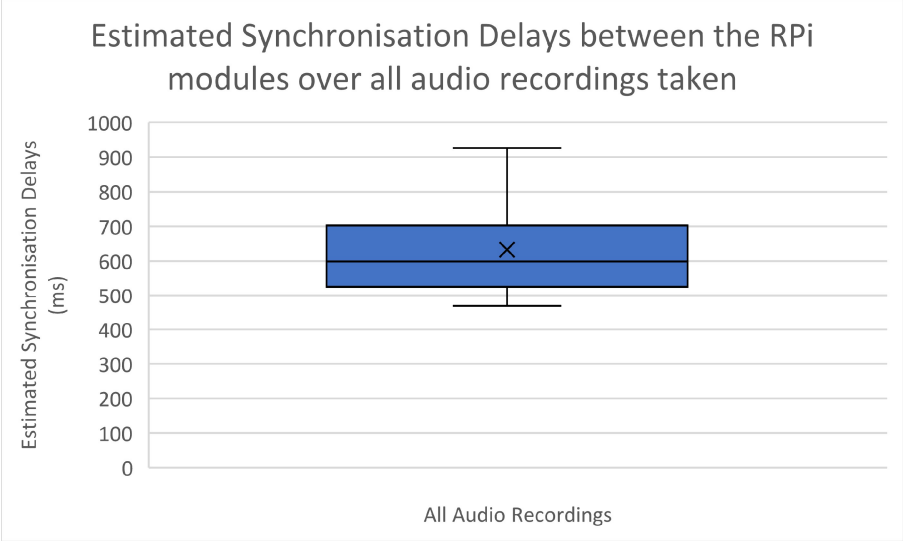


Figure 33: Box and Whisker Plot of the Estimated Synchronisation Delays for all audio recordings.

Both the table and the figure confirm that the estimated synchronisation delays are much greater than the expected 10ms expected from NTP synchronisation.

7.3.2 Varying White Noise Levels

The performance of the subsystem for varying levels of background white noise is shown in the figure below:

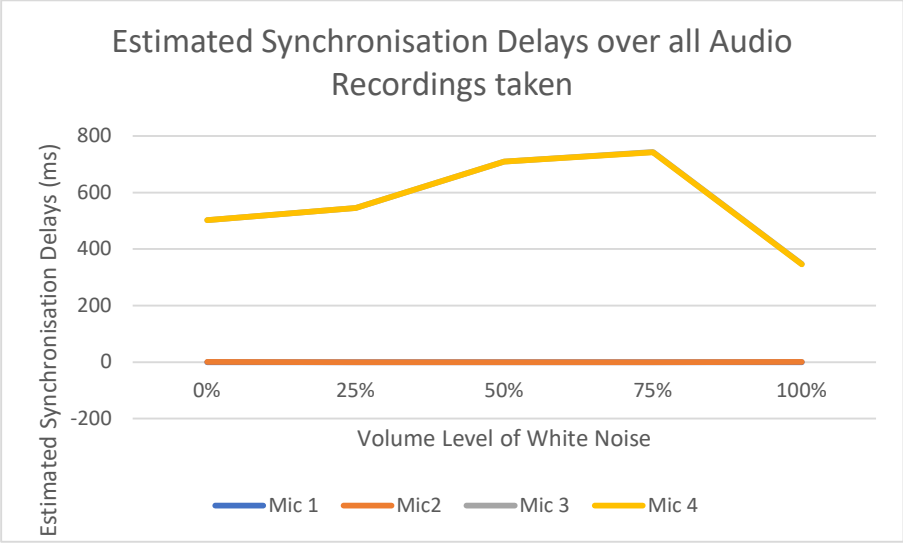


Figure 34: Estimated Synchronisation Delays under white noise conditions.

The figure shows that the subsystem performs equally well at all the levels of white noise tested. This is shown by the fact that there are no noticeable deviations from its expected behaviour.

7.3.3 Varying Source and Calibration Signal Frequency Bands

The performance of the Synchronisation subsystem for varying frequency bands of the source and calibration signals is shown in the table below:

Source Signal Frequency Range (kHz)	Calibration Signal Frequency Range (kHz)	Estimated Synchronisation Delays of Mic 1 (ms)	Estimated Synchronisation Delays of Mic 2 (ms)	Estimated Synchronisation Delays of Mic 3 (ms)	Estimated Synchronisation Delays of Mic 4 (ms)
1-2	3-4	0	0,038	801,072	801,088
1-3	4-6	0	0,115	419,61	419,535
1-4	5-8	0	0,023	472,768	472,95
1-5	6-10	0	0,002	502,422	502,378
1-2	8-10	0	0	407,315	415,07

Table 6: Estimated Delays with reference to Signal Frequency Variation.

The table shows that there are no noticeable deviations from the expected behaviour at all the tested frequency bands.

7.3.4 Varying Bandpass Filter Order

The performance of the Synchronisation subsystem for varying orders of the bandpass filter is shown in the table below:

Bandpass Filter Order	Estimated Synchronisation Delays of Mic 1 (ms)	Estimated Synchronisation Delays of Mic 2 (ms)	Estimated Synchronisation Delays of Mic 3 (ms)	Estimated Synchronisation Delays of Mic 4 (ms)
1	0	0,002	502,422	502,378
2	0	0,002	502,422	502,378
4	0	0,002	502,422	502,378
6	0	0,002	502,422	502,378
8	0	0,005	502,422	502,378
10	0	0,008	502,422	502,378
12	0	0,062	0,002	502,378

Table 7: Estimated Delays with reference to Bandpass Filter Order.

The table shows that the subsystem behaves as expected at all orders of the bandpass filter except the 12^{th} order. This suggests that the increased position error observed at this filter order was at least partially due to incorrect synchronisation. This is because estimated synchronisation delay of Mic 3 is close to zero, but it is expected to be close to the value of the estimated synchronisation delay of Mic 4.

7.4 Time Delay Estimation

The performance of the Time Delay Estimation subsystem was measured using the error in the estimated TDoA values. This was defined as the difference between the estimated TDoA values and the ideal TDoA values. The ideal values were calculated according to the actual position of the source for each of the audio recordings taken.

7.4.1 Default Parameters

The performance of the Time Delay Estimation Subsystem is shown in the figure below:

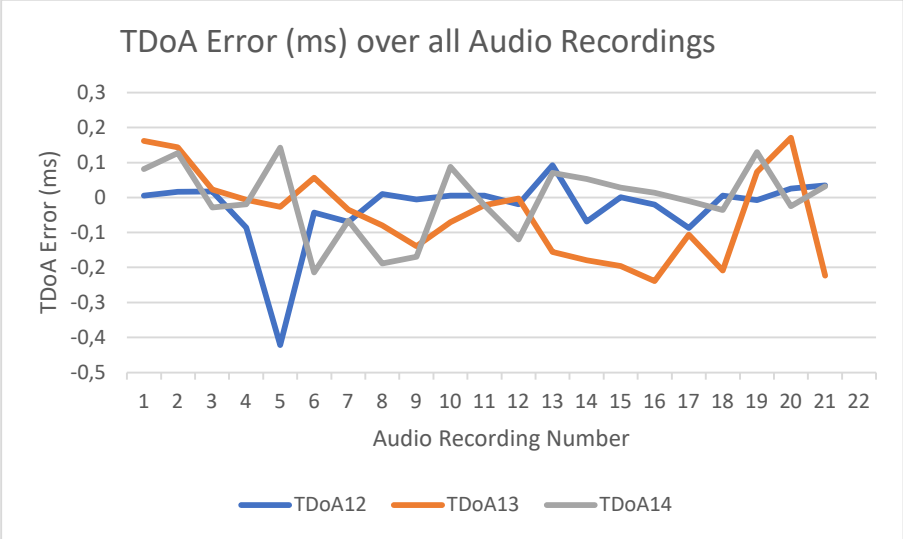


Figure 3521: TDoA errors for all audio recordings.

The figure shows that the TDoA errors do not vary consistently or in similar ways. The TDoA errors appear to lie mostly between -0.2ms and 0.2ms with a bias towards the negative values. The negative values indicate that the estimated TDoA value was less than the ideal TDoA value. This suggest that the system is more likely to underestimate the TDoA values than it is to overestimate them. This asymmetry could potentially be reduced by applying a small constant offset to the TDoA values so that their mean and median values are closer to zero.

The statistics of the TDoA Errors are shown in the table below:

Mean TDoA Error (ms)	-0,029
Median TDoA Error (ms)	-0,010
Minimum TDoA Error (ms)	-0,421
Maximum TDoA Error (ms)	0,171
Standard Deviation of TDoA Error (ms)	0,110

Table 8: Statistics of the TDoA Errors.

Further, the statistics of the TDoA Errors are visually illustrated in the figure below:

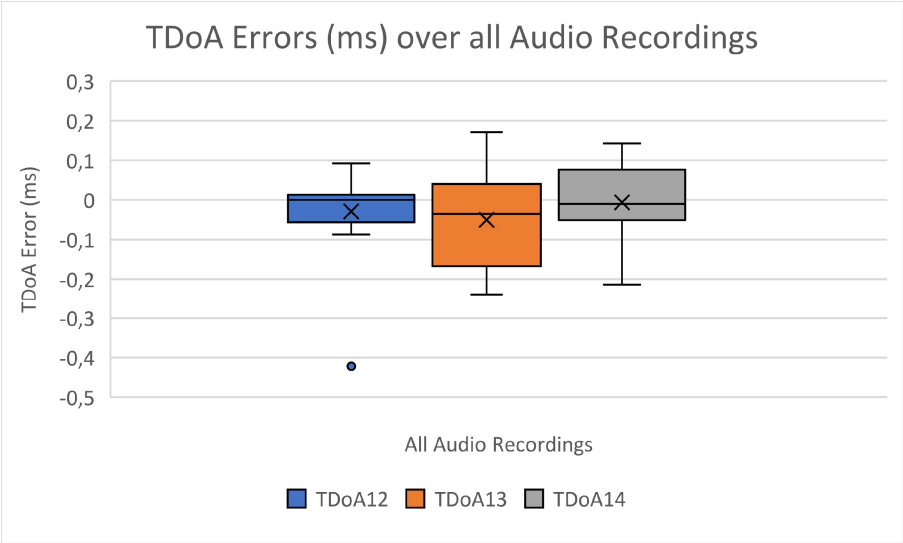


Figure 36: TDoA Errors for all audio recordings.

The table and figure confirm that the TDoA errors are biased towards the negative values. Additionally, the figure shows the differences in the distribution of the TDoA errors between Mic 1 and Mics 2, 3 and 4. The TDoA errors appear to correlate with the distances between Mic 1 and the other Mics. This suggests that Mics that are closer together produce more accurate TDoA values. The suggests that an alternative Mic arrangement could potentially be more optimal than the current implementation with the Mics at the corners of the grid. However, this was not tested due to time constraints. Additionally, TDoA errors one standard deviation away from the mean value would be from -0.139ms to 0.089ms. This accounts for approximately 68% of the values and would theoretically produce a maximum position error of 0.048m for 68% of the values. This is not what was observed in the performance of the overall system which indicated that approximately 75% of the position errors were below 0.032m. This is likely explained by the fact that the TDoA errors of each Mic with respect to Mic 1 do not appear to be strongly correlated. Thus, their combined effects result in a lower position error. Thus, the performance of the Time Delay Estimation subsystem can be considered acceptable.

7.4.2 Varying White Noise Levels

The performance of the Time Delay Estimation subsystem for varying levels of background white noise is shown in the figure below:

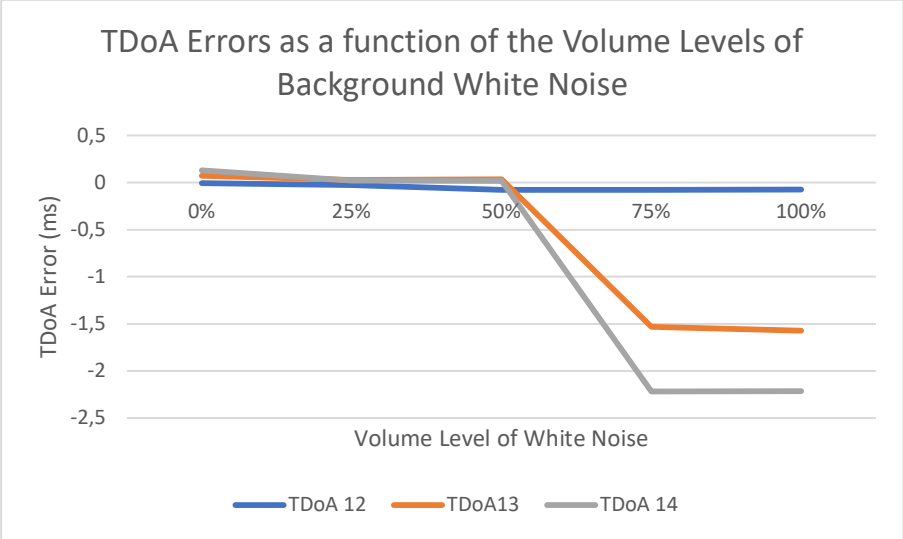


Figure 3722: TDoA Errors as a function of Background White Noise.

The figure shows that the system performs optimally for noise levels up to 50% of the maximum white noise volume level. Above this level the TDoA errors increase significantly. This is likely due to errors in both the synchronisation delays and ToA delays which are combined to produce the estimated TDoA values. Thus, the increase in the TDoA errors is likely the cause of the observed increase in position errors at noise levels above the threshold. This suggests that a more robust time delay estimation technique may be required for the system to be more resilient to varying noise levels. This was not tested due to technical and time constraints. Consequently, this supports the suggestion of the limiting the use of the system in ideal environments where the noise levels are below the threshold.

7.4.3 Varying Source and Calibration Signal Frequency Bands

The performance of the Time Delay Estimation Subsystem for varying frequency bands of the source and calibration signals is shown in the table below:

Source Signal Frequency Range (kHz)	Calibration Signal Frequency Range (kHz)	TDoA 12 Errors (ms)	TDoA 13 Errors (ms)	TDoA 14 Errors (ms)
1-2	3-4	-0,28016	800,6248	801,088
1-3	4-6	0,090843	0,029843	-0,043
1-4	5-8	0,010843	-0,13516	0,032
1-5	6-10	-0,00716	0,072843	0,13
1-2	8-10	-0,31716	406,9098	415,07

Table 9: TDoA Errors with referemnce to Signal Frequency Variation.

The table shows that the TDoA errors are extremely large at bandwidths of 1kHz for the source and calibration signals. Additionally, it shows that the performance of the Time Delay Estimation subsystem remains acceptable for bandwidths from 2kHz to 4kHz. This supports the observations of the effect of the bandwidth on the position errors.

7.4.4 Varying Bandpass Filter Order

The performance of the Time Delay Estimation subsystem for varying orders of the bandpass filter are shown in the table below:

Bandpass Filter Order	TDoA 12 Errors (ms)	TDoA 13 Errors (ms)	TDoA 14 Errors (ms)
1	-0,01216	0,072843	0,013
2	-0,01216	0,072843	0,013
4	-0,01216	0,072843	0,013
6	-0,00716	0,072843	0,13
8	-0,01916	0,072843	0,013
10	-0,01516	0,072843	0,013
12	0,050843	-502,347	0,013

Table 10: TDoA Errors with reference to Bandpass Filter Order.

The table shows that the TDoA values remain optimal for bandpass orders up to 10 but significantly deteriorate at a bandpass order of 12. This is likely caused by the errors in synchronisation observed at this filter order. Thus, this results in the observed position error at this filter order.

7.5 TRIANGULATION

The performance of the triangulation subsystem is measured through the estimated source positions and the associated position errors. However, the same measurement was used to characterise the overall system. Thus, the results already presented in the overall system section will not be repeated but they apply to the Triangulation subsystem as well as the overall system. Instead, alternative measurements of the performance of the subsystem will be presented.

7.5.1 Performance of the Linear LSE

The linear LSE was added to the physical implementation of the system while it was absent from the simulated system. Since it was added to reduce number of iterations performed by the nonlinear LSE the performance of the linear LSE is determined by the initial position error. This is defined as the distance between the initial position given to the nonlinear LSE and the estimated position produced by the nonlinear LSE. Previously the centre point of the grid was used as the initial point as it the closest to all other points on the grid. Thus, the performance of the linear LSE can be evaluated by whether the initial errors it produces outperforms those produce by the centre point. The comparison of the initial position errors of the linear LSE and centre point are shown in the figure below:

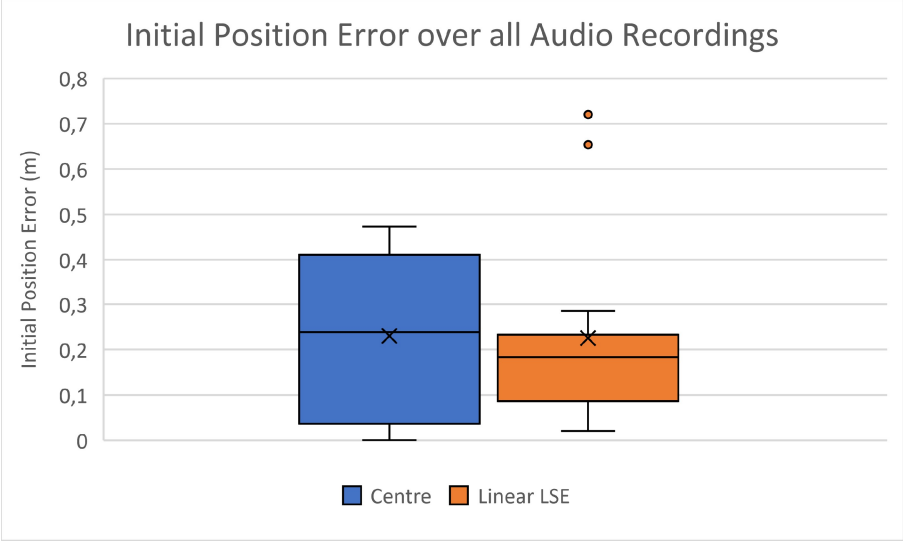


Figure 38: Box and Whisker Plot for Initial position for all audio recordings.

The figure shows that the linear LSE does outperform the centre point by a relatively significant margin. The range, mean and median of the initial position errors of the linear LSE are all less than those of the centre point. Additionally, 75% of the linear LSE initial position errors fall below approximately 0.22m while only approximately 50% of those of the centre point fall in the same region. Thus, the results show that the linear LSE sufficiently outperforms the centre point to justify its use with the added processing overhead. This is because the overhead of solving the linear LSE is relatively minor compared to that of additional iterations of the nonlinear LSE.

7.5.2 Performance of Non-Linear LSE

An alternative measurement of the performance of the Triangulation subsystem besides position error is the coordinate errors. This is defined as the errors in the estimated x and y coordinates compared to their ideal values. The performance of the Triangulation subsystem with respect to the coordinate errors are shown in the figure below:

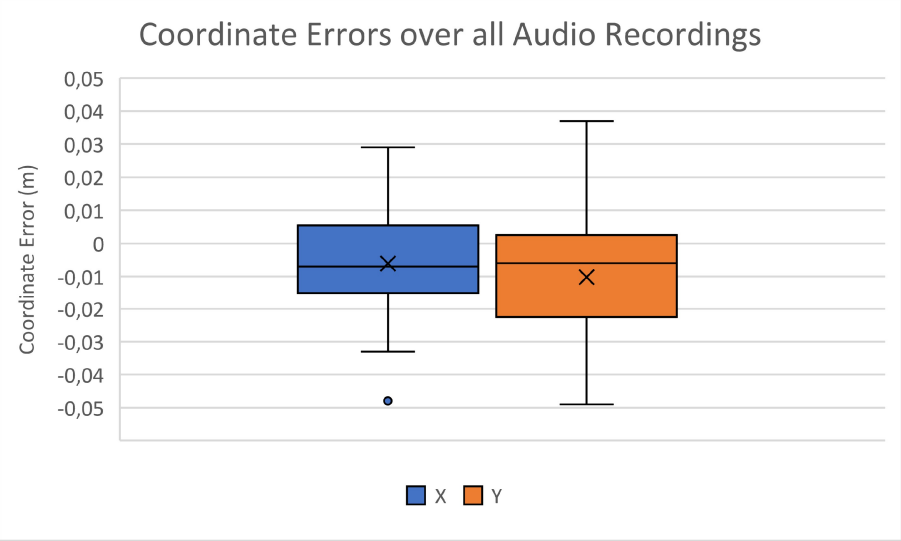


Figure 39: Box and Whisker Plot for Coordinate Errors for all audio recordings.

The figure shows that the errors in the x and y coordinates are not identical. This is expected as the x coordinates have a greater range that the y coordinates. However, the y coordinate errors appear to have a larger range than the x coordinate errors which is not explained by this. This could possibly be due to the fact that the positions and angles of the Mics result in more accurate x coordinate values. The complete cause of this cannot be determined without further investigation. This was not done due to time constraints.

The performance of the Triangulation subsystem can also be illustrated qualitatively analysing whether the estimated position is appropriate for the give TDoA values. This is because the position errors measured relative to the ideal position are influenced by the errors in the TDoA values as well as the performance of the Triangulation subsystem. Thus, analysing the estimated position without regarding the ideal position gives a more neutral measurement of performance. Five points were chosen on the grid to illustrate this performance. The points were (0.4, 0.25), (0.15, 0.15), (0.35, 0.15), (0.15, 0.65) and (0.35, 0.65). Plots illustrating the hyperbolas produced by the TDoA values acquired at these points and the estimated source position were generated. These plots are shown below:

Figure 40: Graph of the hyperbolas produced by the TDoA values acquired at a source position of (0.4, 0.25) and the associated estimated source position

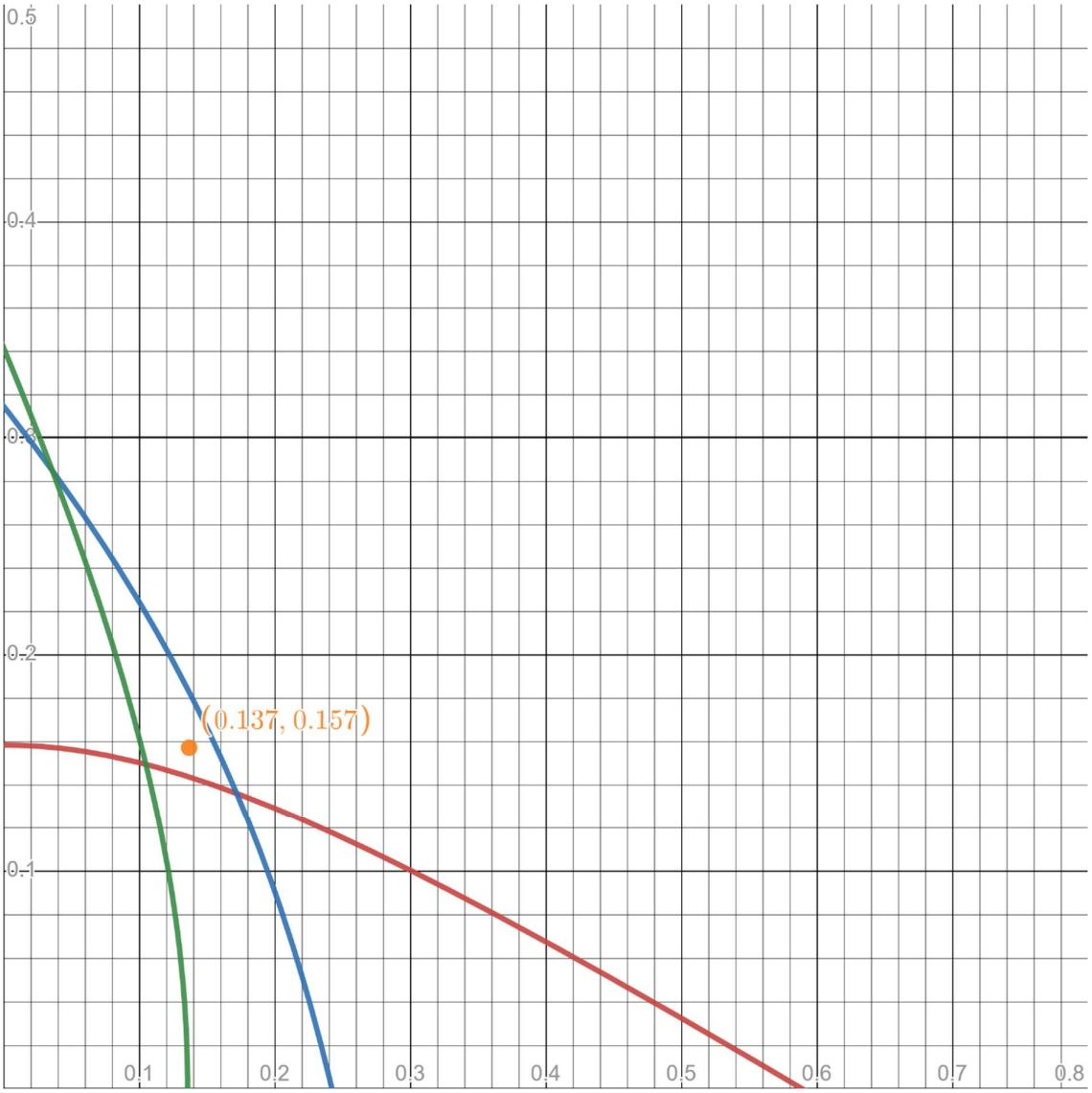


Figure 41: Graph of the hyperbolas produced by the TDoA values acquired at a source position of (0.15, 0.15) and the associated estimated source position

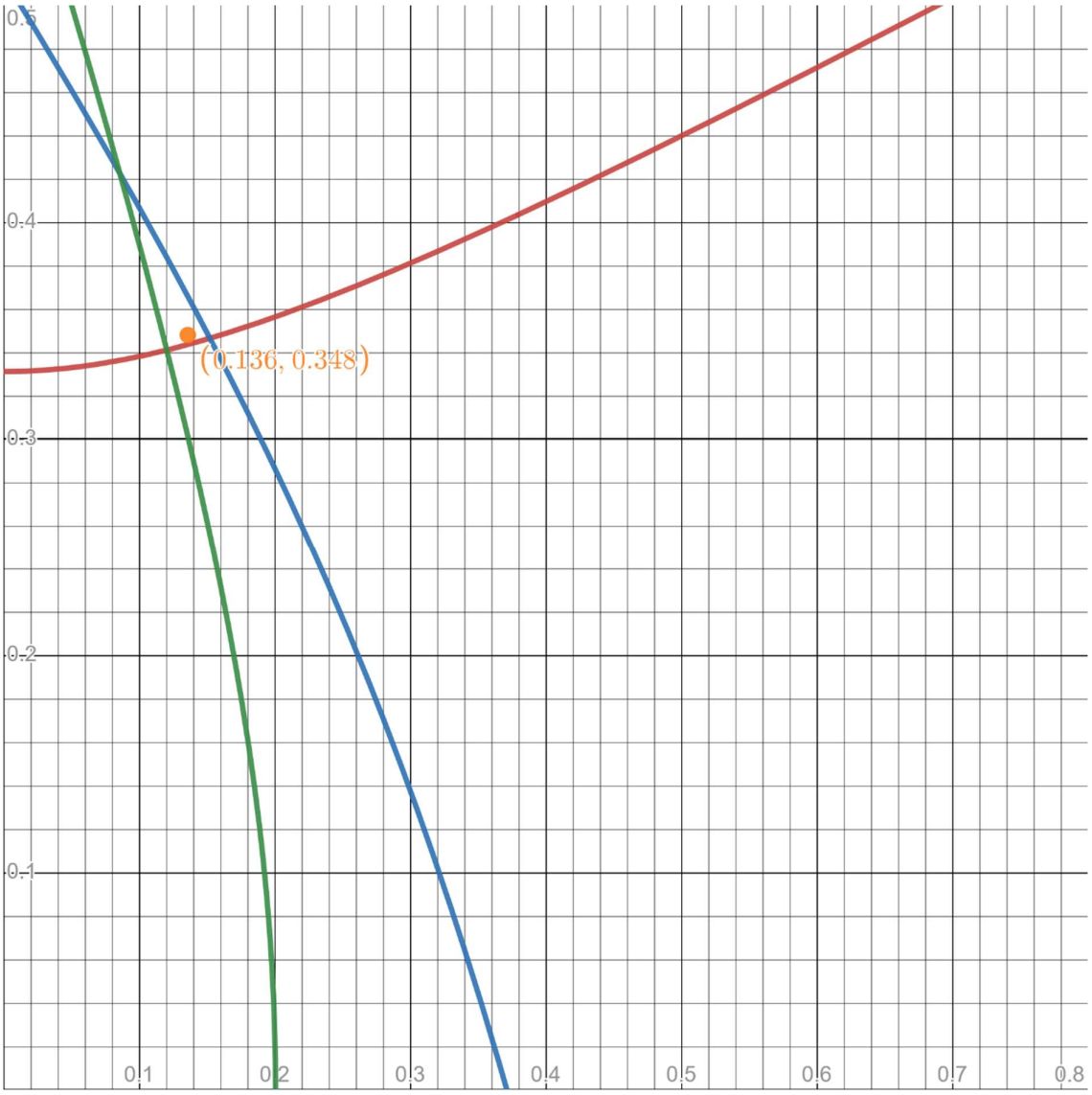


Figure 42: Graph of the hyperbolas produced by the TDoA values acquired at a source position of (0.15, 0.35) and the associated estimated source position

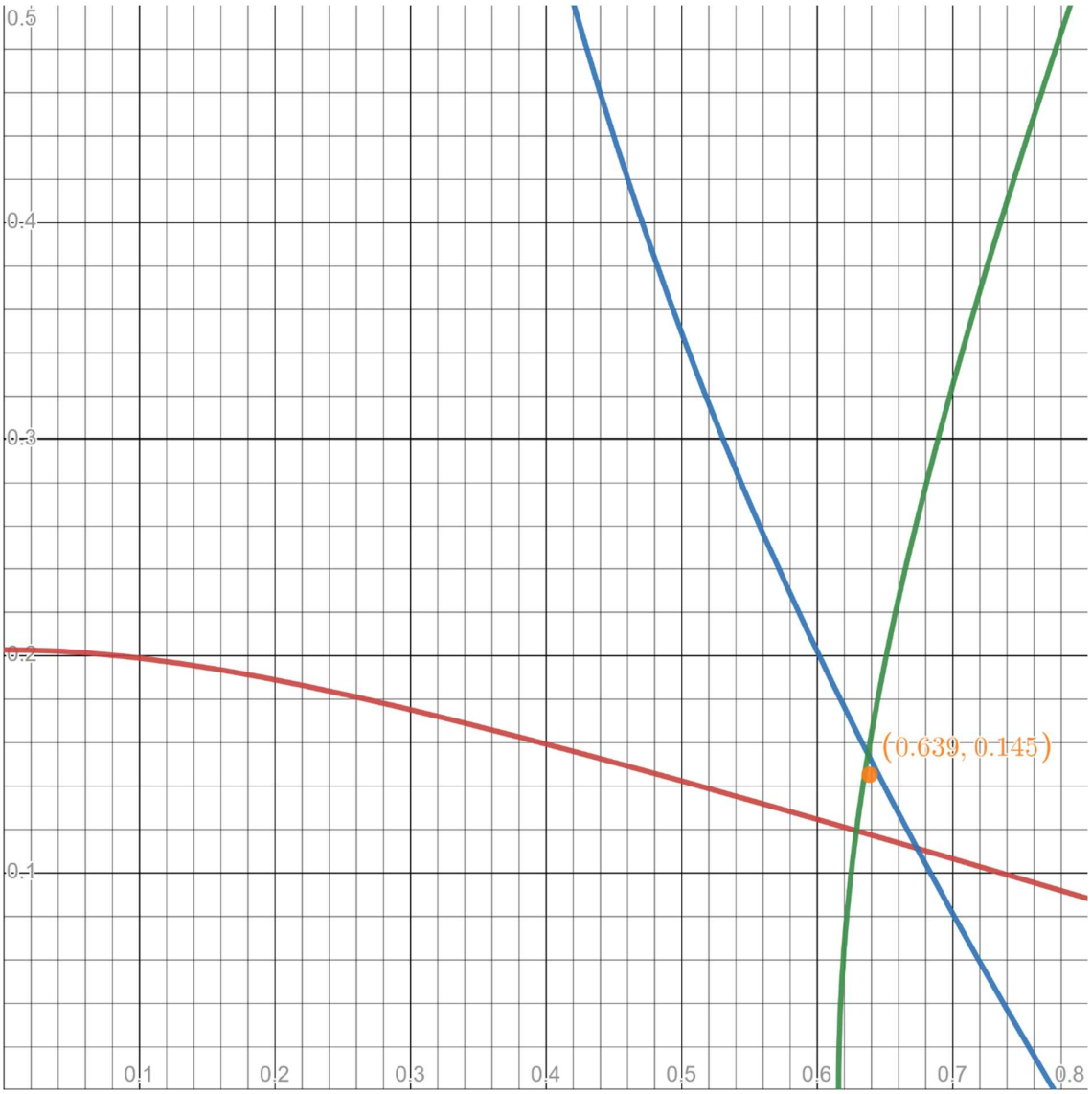


Figure 43: Graph of the hyperbolas produced by the TDoA values acquired at a source position of (0.65, 0.15) and the associated estimated source position

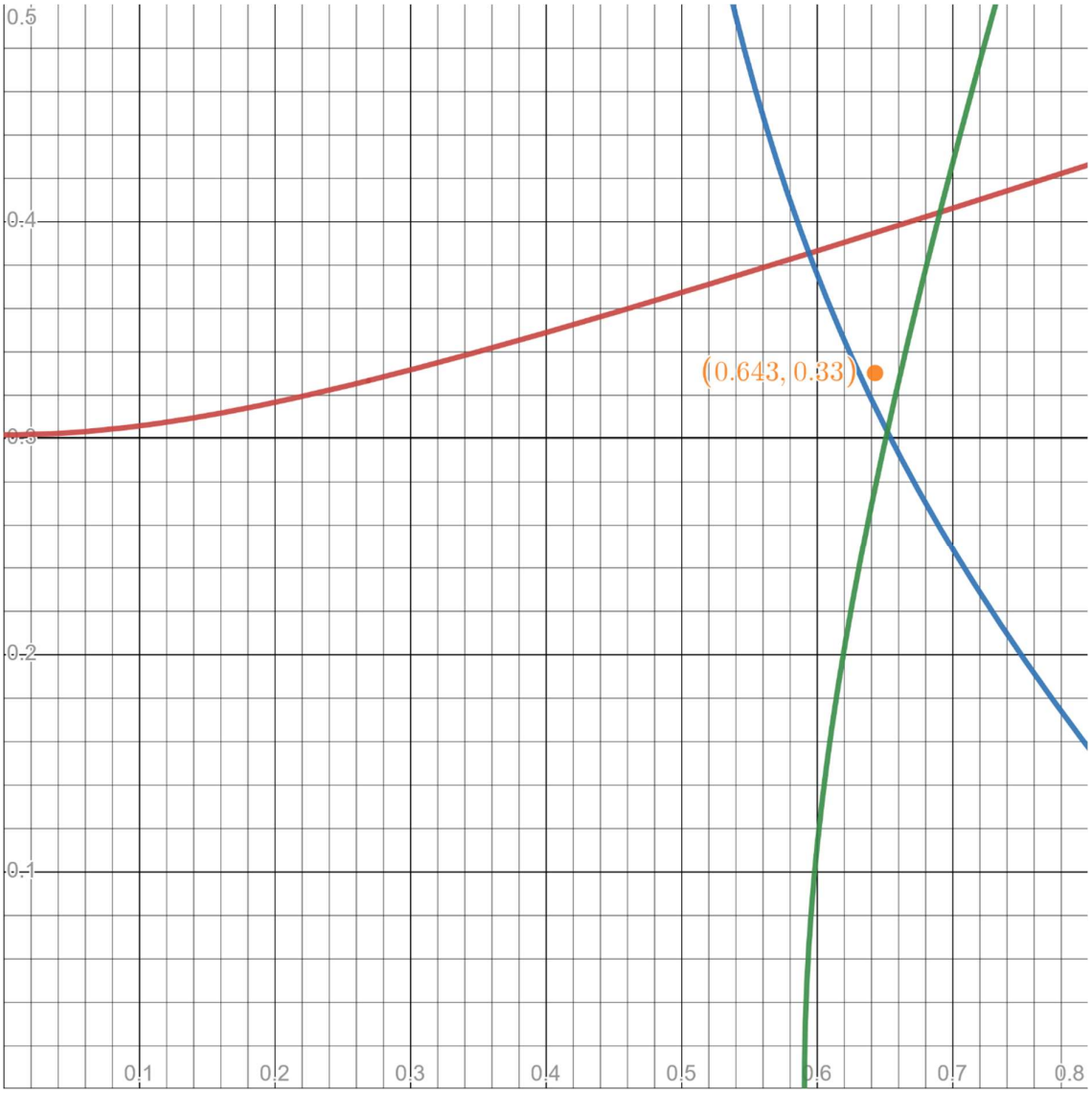


Figure 44: Graph of the hyperbolas produced by the TDoA values acquired at a source position of (0.65, 0.35) and the associated estimated source position

The plots show that for all of the positions chosen the estimated source position matches what is expected by the hyperbolas generated by the estimated TDoA values at the respective positions. Additionally, the plots show that the hyperbolas usually do not have a single intersection point and this explains the poor performance of the linear LSE initial position estimate compared to the nonlinear LSE position estimate. Thus, it can be concluded that the Triangulation subsystem performs acceptably.

8 EVALUATION (ATPS)

For the physical implementation, the following ATPs were tested:

ATP 1.1: Pi Communication

Acceptance test for Pi communication: To ensure bidirectional communication, a test message will be sent from each of the RPi's to each other.

Instead of using a test message to check whether the Raspberry Pi modules were able to establish bidirectional communication with each other, the IP addresses of the two Pi's were pinged from the different Pi module. The two modules were named Raspberry Pi 1 and Raspberry Pi 2, and the IP address of Pi 1 was pinged from Pi 2, while the IP address of Pi 2 was pinged from Pi 1. The pictures below display the results of the two corresponding pings:

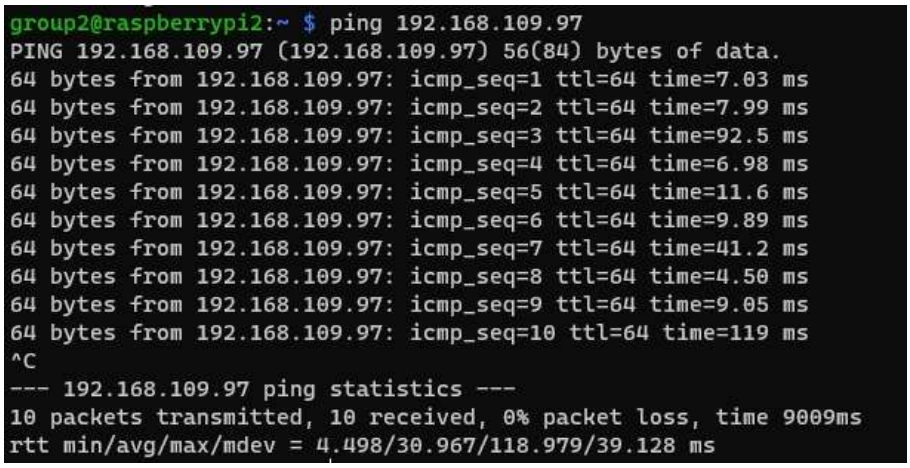


Figure 45: Raspberry Pi 1 being pinged from Raspberry Pi 2

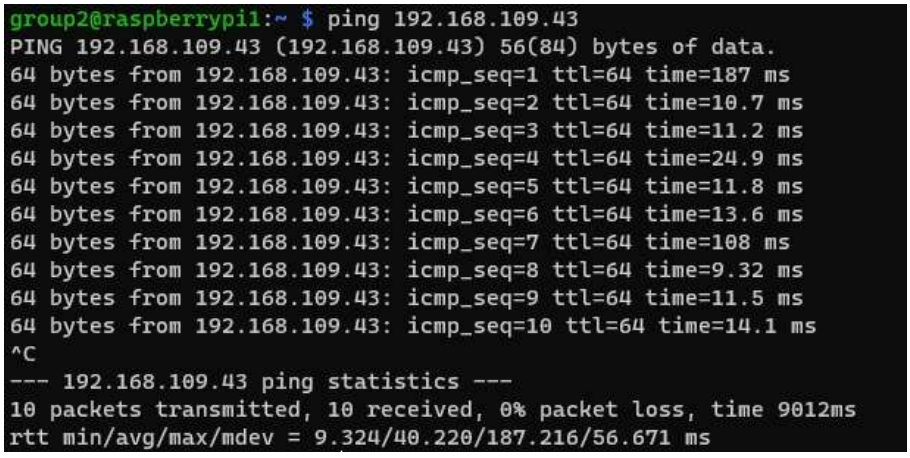


Figure 46: Raspberry Pi 2 being pinged from Raspberry Pi 1

Based on the results of the acceptance test above, it can be seen that in both cases, 10 packets are transmitted, and 10 packets are received, indicating that there is absolutely no packet loss and the Pi's are able to communicate effectively with each other.

Acceptance Test: Passed

ATP 1.2: Pi Timing

Acceptance test for Pi timing:For this acceptance test, the synchronisation time delays of each microphone will be recorded in relation to microphone 1.

The table below displays the plotting of various coordinates and their respective synchronisation delays:

x	y	Mic 1	Mic 2	Mic 3	Mic 4	Between Pi's
0.4	0.25	0	0.002	555.79	556	555.76
0.05	0.05	0	-0.085	713.195	713.218	713.249
0.25	0.05	0	0	598.278	598.3	598.289
0.55	0.05	0	-0.072	590.302	590.328	590.351
0.75	0.05	0	0	668.84	669.098	668.969
0.15	0.15	0	-0.07	871.363	871.387	871.41
0.65	0.15	0	-0.072	539	538.642	538.666
0.05	0.25	0	0.012	598.635	598.868	598.7455
0.25	0.25	0	0.008	925.91	926.148	926.025
0.55	0.25	0	0.008	571.012	571.178	571.091
0.75	0.25	0	0.012	680.278	680.295	680.2805
0.15	0.35	0	0.008	510.39	510.56	510.471
0.65	0.35	0	0.075	667.15	667.392	667.2335
0.05	0.45	0	0.02	734.468	734.72	734.584
0.25	0.45	0	0.002	694.422	694.675	694.5475
0.55	0.45	0	0.002	468.025	468.278	468.1505

Table 11: Coordinates and their respective synchronisation delays.

The important information to note in the table above is the synchronisation delay between the Raspberry Pi's. From the table above, the average synchronisation delay between the two Pi's is 632.6867381 milliseconds. Although this synchronisation delay is higher than the expected 10 milliseconds expected from NTP, the results are consistent with what is expected. Furthermore, validation of the sync measurements may be flawed as the true desynchronisation cannot be measured.

Acceptance Test: Passed

ATP 2.1: Signal Capture:

Acceptance test for signal capture: For signal capture, an acceptance test will be performed to determine whether the microphone was able to capture the sound being emitted from the source. Furthermore, a test can be conducted to ensure the microcontroller sends a response message once it has been able to capture the sound signal.

A successful capturing of signals yields the following response message:

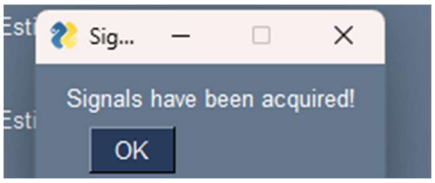


Figure 47: Response Message when Signals are acquired.

Furthermore, to check the validity and correctness of the signals being captured, the signal acquisition plots for the approximate point (0.55, 0.45) were compared below:

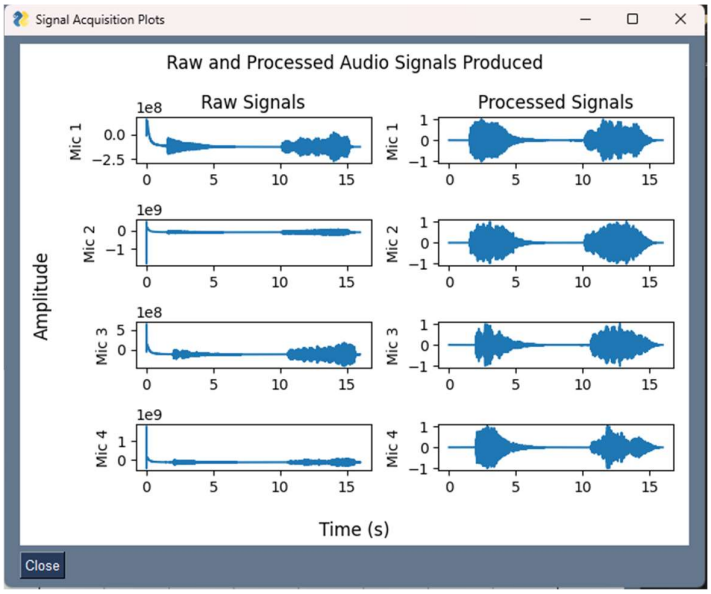


Figure 4823: Signal Acquisition Plots for position (0.55,0.45).

With the exception of the initial spike, the raw signal seems to accurately pick up both the calibration signal (the first few spikes) and the sound source signal (the second section of spikes). The signals look different for each microphone, which is to be expected since the sound source is not the same distance away from the microphones.

In conclusion to this test, it can be seen that the microphones are able to successfully detect sound within the grid and record the signal for a specified period of time.

Acceptance Test: Passed

ATP 2.2: Signal Preprocessing

Acceptance test for Signal Preprocessing: Tests will also be conducted to measure the signal-to-noise ratio (SNR). Information regarding the noise in the system will be taken from the filter that will be used to attenuate the noise from the signal, which can then be used to calculate the SNR.

The following table shows the input SNR vs the output SNR for a number of recordings:

Input SNR	Output SNR	SNR Difference
-20.09	46.73	66.82
-8.61	40.53	49.14
-10.84	39.01	49.85
-9.24	44.64	53.88
-8.14	45.43	53.57
-10.24	39.14	49.38
-9.79	44.89	54.68
-9.9	39.88	49.78
-10.15	38.47	48.62
-10.33	43.98	54.31
-9.14	36.73	45.87
-9.57	36.83	46.4
-9.71	40.99	50.7
-7.86	43.63	51.49
-10.18	38.91	49.09
-10.02	38.85	48.87

Table 12: Input SNR vs Output SNR.

From the table above, the average SNR of the input signal is -10.1495 dB. In stark comparison, the average output SNR is 45.59429 dB. This is a satisfactory level of performance from the filter to attenuate noise in order to increase SNR of the original signal.

Acceptance Test: Passed

ATP 3: Time Delay Estimation

Acceptance test for Time Delay Estimation: Correctness of the time delay will be tested for. Information regarding the time delay will be computed using the host device, and the distance that was calculated using the Time-Difference-of-Arrival equations will be outputted.

The table below show the results of the TDoA acceptance test. It consists of the TDoA error (in milliseconds) for each microphone:

TDoA 1	TDoA 2	TDoA 3	TDoA 4
0.005	0.162	0.082	0.083
0.015873408	0.143825382	0.12728904	0.09566261
0.017523415	0.023520538	-0.02818637	0.004285861
-0.086293092	-0.006290215	-0.01881363	-0.037132313
-0.421463658	-0.026415633	0.14271096	-0.101722777
-0.043289735	0.056845827	-0.213611753	-0.06668522
-0.06954242	-0.034677982	-0.066388247	-0.05686955
0.01	-0.080430206	-0.188430206	-0.086286804
-0.005	-0.139390939	-0.169390939	-0.104593959
0.005	-0.070609061	0.087390939	0.007260626
0.005	-0.021569794	-0.021569794	-0.012713196
-0.018710265	-0.002322018	-0.119864438	-0.046965574
0.09254242	-0.155845827	0.070864438	0.002520344
-0.068873408	-0.178584367	0.052951975	-0.064835267
0.000476585	-0.195709785	0.027997123	-0.055745359
-0.019706908	-0.238520538	0.014002877	-0.081408189

Table 13: TDoA Error for each Mic.

Averaging the TDoA for each microphone is shown below:

TDoA 1: -0.029047619 ms (29 µs)

TDoA 2: -0.050333333 ms (50 µs)

TDoA 3: -0.006142857 ms (6.14 µs)

TDoA 4: -0.028507937 ms (28.5 µs)

Although the TDoA delay values are a bit higher than the recommended maximum of 10 µs, the values are still low enough to ensure an accurate triangulation and localisation.

Acceptance Test: Passed

ATP 4: Triangulation

Acceptance test for Triangulation: The sound source will be placed at various positions within the grid, for example: close to a microphone, at the center of the grid, as well as outside the grid. The system will then be tested to see if provides the appropriate coordinate information.

The table below shows all the different positions that the sound source was placed, along with their estimated coordinates and position errors:

x	y	Estimated x	Estimated y	Position Error
0.4	0.25	0.422	0.25	0.022
0.05	0.05	0.079	0.044	0.029614186
0.25	0.05	0.246	0.061	0.0117047
0.55	0.05	0.55	0.034	0.016
0.75	0.05	0.766	0.001	0.051546096
0.15	0.15	0.137	0.157	0.014764823
0.65	0.15	0.639	0.145	0.012083046
0.05	0.25	0.002	0.254	0.048166378
0.25	0.25	0.217	0.251	0.033015148
0.55	0.25	0.558	0.228	0.0234094
0.75	0.25	0.743	0.25	0.007
0.15	0.35	0.136	0.348	0.014142136
0.65	0.35	0.643	0.33	0.02118962
0.05	0.45	0.041	0.43	0.021931712
0.25	0.45	0.234	0.431	0.024839485
0.55	0.45	0.531	0.409	0.045188494
0.75	0.45	0.743	0.426	0.025
0.4	0.1	0.395	0.068	0.032388269
0.4	0.4	0.408	0.406	0.01
0.4	0	0.403	0.037	0.037121422
0.4	0.5	0.379	0.477	0.031144823

Table 14: Estimated Coordinates vs Actual Coordinates.

Since the triangulation algorithm is bounded, whenever a sound source is outside the bounds of the grid, it defaults to the closest corner. To fix this issue, the bounds were adjusted to ensure that the corners would only determine a reading that is outside the grid bounds.

Based on the values of the table above, the average position error is 0.012655059 units.

This value is very close to the maximum threshold of 0.01 units; hence this passes the acceptance test.

Acceptance Test: Passed

ATP 5.1: User Interface Functionality

Acceptance test for user interface functionality: Each input and output element will be tested for functionality and correctness. The inputs will be executed numerous times, and the output of these inputs will also be recorded, to determine whether the input achieves the desired output/function.

Each component in the graphical user interface (GUI) works as intended. While most aspects of the user interface are responsive, the 'Start Localisation' button is considerably slow, taking between 10-15 seconds before the answer is displayed on the GUI. The most likely reasons for this occurrence are the initialisation of the microphones, the recording of the signal, filtering and processing the signal, applying cross-correlation and finally calculating the TDoA and triangulating the position. These processes contribute a significant amount to the delay, but besides that the GUI works as intended.

Acceptance Test: Passed

ATP 5.2: User Interface Design

Acceptance test for user interface design: The user interface will be tested in its ergonomic aspect and ease of use. A volunteer, with no prior knowledge of the project, will be asked to use the program and comment/rate on how easy or difficult the interface is to use and understand.

A student at UCT was invited to review the GUI. The student praised the design of the GUI, citing it to be simplistic and easy to use. The GUI was also described to be interactive, with pop-up windows and changing text. The inclusion of the Signal Acquisition Plots, Synchronisation Plots and Time Delay Estimation Plots was also applauded as it gave the user a means to debug in case something goes wrong. The student also recommended a 'Help' button that could provide the user with necessary insight on how to use the app.

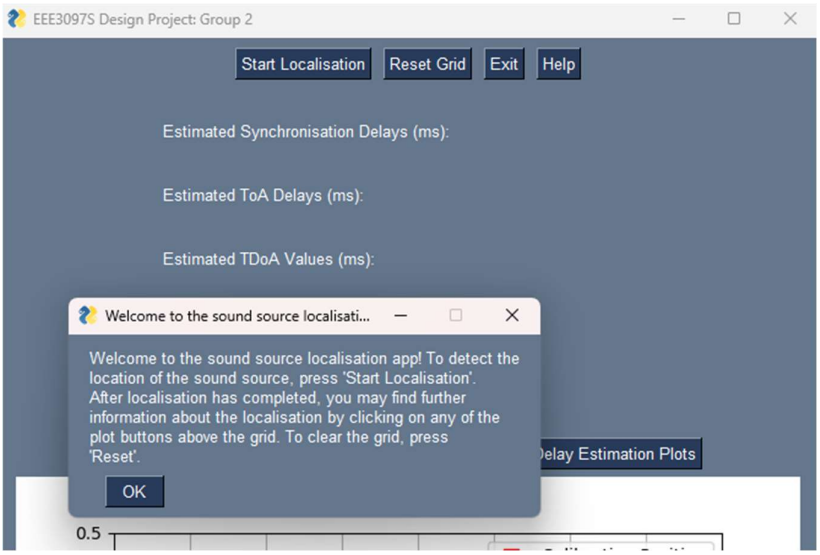


Figure 49: Improved GUI.

Acceptance Test: Passed

9 CONCLUSION

To summarize all the key findings:

- The overall system achieved a mean position error of 0.025 m.
- The physical implementation is sensitive to high levels of noise.
- The optimal Source Signal Frequency range, generating a chirp signal, was found to be 1 kHz to 5 kHz.
- The optimal Calibration Signal Frequency range, generating a chirp signal, was found to be 6 kHz to 10 kHz.
- The Order of the Bandpass filter should be between 4 and 6 poles.
- Whilst the errors in the TDoA and triangulation results may be a bit over the theoretical maximum limit, they were still able to produce an accurate localisation of the sound source.
- It is imperative that the speaker (sound source) and microphones synchronise the playback, otherwise it will lead to too much clipping of the sound source and may severely affect the triangulation and TDoA.

In conclusion, it can be said that the physical implementation of the sound detection by triangulation was a success. During the development of the project, numerous lessons were learnt, and much insight was gained, as follow.

The lessons learned and insights gained:

- Communication between group members is vital to prevent issues such as clashes in git commits and possible overwriting of files, potentially nullifying changes made by other group members.
- Building on the point above, it is also very important to take note of commits made by group members so that one knows what changes have been made, and whether these changes are reflected after pulling the git repository.
- Testing should always be conducted in advance to allow enough time for debugging.

10 BIBLIOGRAPHY

[1] Da Costa, D.G. (2022) Time Difference of Arrival Acoustic Triangulation Using a Distributed Sensor Network. Dissertation

[2] "Raspberry pi documentation," Raspberry Pi. [Online]. Available: https://www.raspberrypi.com/documentation/

[3] "Adafruit I2S MEMS microphone breakout," Adafruit Learning System, [Online]. Available: https://learn.adafruit.com/adafruit-i2s-mems-microphone-breakout/overview

[4] B. Jin, X. Xu, and T. Zhang, "Robust time-difference-of-arrival (tdoa) localization using weighted least squares with cone tangent plane constraint," Sensors (Basel, Switzerland), vol. 18, 03 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/3/778

[5] D. Dalskov and S. K. Olesen, 'Locating acoustic sources with multilateration', Master's, 2014. [Online]. Available: https://vbn.aau.dk/ws/files/198526294/Measurements/Source%20signals

[6] M. Pollefeys and D. Nister, 'Direct computation of sound and microphone locations from time-difference-of-arrival data', in 2008 IEEE International Conference on Acoustics, Speech and Signal Processing, 2008, pp. 2445–2448. [Online]. Available: https://cvg-pub.inf.ethz.ch/WebBIB/papers/2008/001_PollefeysICASSP08.pdf

[7] T. Ho et al., "Acoustic Source Localization," Georgia Institute of Technology. [Online]. Available: https://eceseniordesign2022spring.ece.gatech.edu/sd22p11/finalwrittenreport.pdf

[8] N. R. Kumarasiri, Development of novel algorithms for localization in wireless sensor networks. The University of Toledo, 2014.

[9] "I2S Output Digital Microphone Datasheet", SPH0645LM4H-B, Rev. B, Knowles, 2015. [Online]. Available: https://cdn-shop.adafruit.com/product-files/3421/i2S+Datasheet.PDF