EEE3097S 2023 MILESTONE REPORT 1

Acoustic Triangulation

Joshua King: KNGJOS007

Dante Webber: WBBDAN003

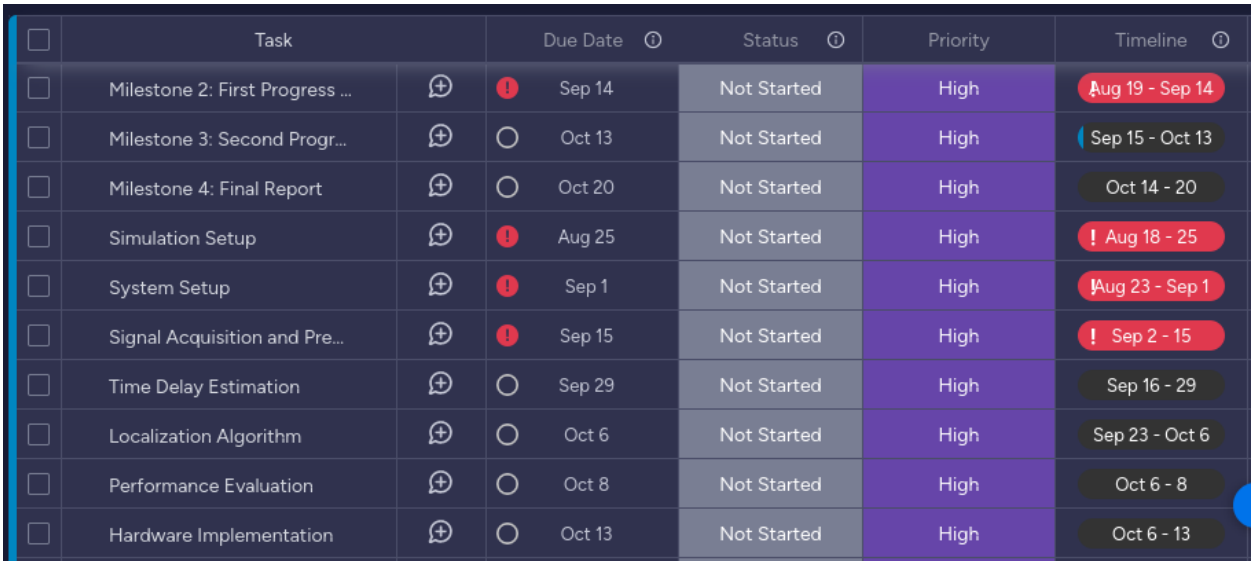
Michael Awe: AWXMIC001

Admin

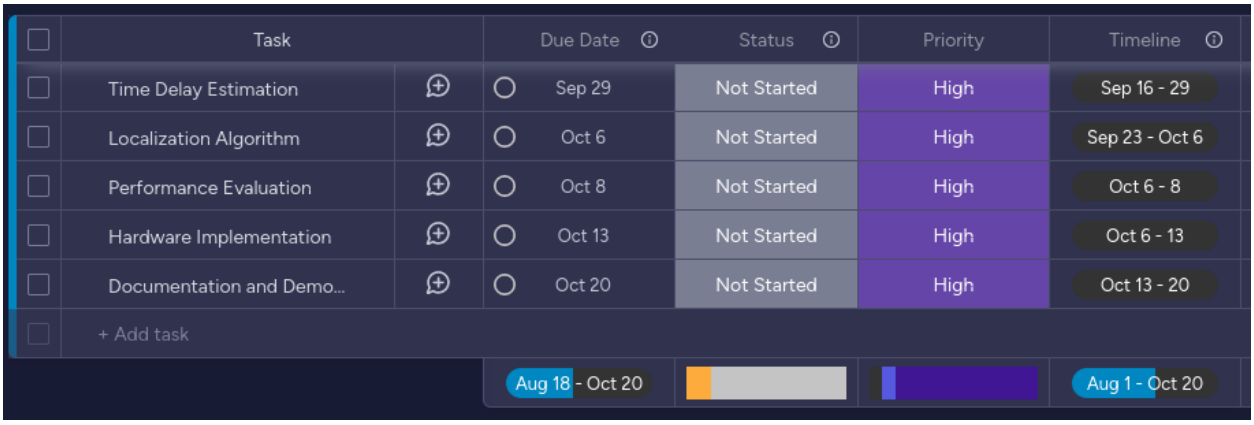
Individual contributions

Joshua King: KNGJOS007	Dante Webber: WBBDAN003	Michael Awe: AWXMIC001
Simulation set up	Triangulation and Result Analysis	Sub-System specification

Snap shot of time management tool (Monday.com)



Task	Due Date	Status	Priority	Timeline
Milestone 2: First Progress ...	Sep 14	Not Started	High	Aug 19 - Sep 14
Milestone 3: Second Progr...	Oct 13	Not Started	High	Sep 15 - Oct 13
Milestone 4: Final Report	Oct 20	Not Started	High	Oct 14 - 20
Simulation Setup	Aug 25	Not Started	High	Aug 18 - 25
System Setup	Sep 1	Not Started	High	Aug 23 - Sep 1
Signal Acquisition and Pre...	Sep 15	Not Started	High	Sep 2 - 15
Time Delay Estimation	Sep 29	Not Started	High	Sep 16 - 29
Localization Algorithm	Oct 6	Not Started	High	Sep 23 - Oct 6
Performance Evaluation	Oct 8	Not Started	High	Oct 6 - 8
Hardware Implementation	Oct 13	Not Started	High	Oct 6 - 13



Task	Due Date	Status	Priority	Timeline
Time Delay Estimation	Sep 29	Not Started	High	Sep 16 - 29
Localization Algorithm	Oct 6	Not Started	High	Sep 23 - Oct 6
Performance Evaluation	Oct 8	Not Started	High	Oct 6 - 8
Hardware Implementation	Oct 13	Not Started	High	Oct 6 - 13
Documentation and Demo...	Oct 20	Not Started	High	Oct 13 - 20
+ Add task				
	Aug 18 - Oct 20			Aug 1 - Oct 20

Link to Github

https://github.com/dantewebber/EEE3097S

Progress is unfortunately slightly behind, we were suppose to have started with the physical set up already but have not gotten to it yet. We are currently just finished with the simulation stage.

Table of Contents

1 Introduction

This milestone report presents the progress of our engineering design project, which aims to detect the position of a sound source on an A1 grid using two Raspberry Pi's and four distributed microphones.

The goal of this report is to detail our system design and implementation, focusing on the key subsystems of time delay estimation and triangulation. These subsystems are crucial to the successful operation of the system, as they enable the calculation of the time difference of arrival (TDOA) and the subsequent location of the sound source.

The report also discusses the overall system architecture, including the use of Raspberry Pi's and microphones, and the structure of the distributed sensor network. The simulation model used for the project is explained, along with the algorithms or techniques used for TDOA calculation.

The work detailed in this report represents the second milestone in our project. The simulations and experiments conducted so far have provided valuable insights and have guided the development of the system. This report aims to document these findings and outline the next steps in the project.

2 Simulation Setup

2.1 Description of Simulation Environment and Tools Used

Given the goal of the assignment, we chose MATLAB as our simulation environment due to its extensive support for signal processing, mathematical modeling, and its potential to easily generate visual representations of our data. We also chose MATLAB for it’s ease of use as the entire team was familiar with the software.

The Raspberry Pi's and Adafruit MEMS microphones were not directly used in the simulation. Instead, we developed and tested all algorithms and techniques in MATLAB before implementing our design on the provided hardware.

2.2 Rationale Behind Chosen Simulation Approach

The multilateration technique, which utilizes the concept of time difference of arrival (TDoA) of a signal at multiple points (microphones in our case), has been chosen because of its accuracy and ease of implementation. This method is often used in global navigation satellite systems, radio navigation systems, and mobile cellular communication.

Our simulation strategy involved creating a hypothetical sound signal, adding random noise, and producing time delays (to simulate the distance of each microphone from the sound source). This artificial data provide a controlled way to understand how our proposed algorithm performs under different conditions before we apply it to an actual system. MATLAB provides sufficient functionality for simulating our algorithm and was thus the platform of choice.

2.3 Simplifications and Assumptions Made During the Simulation

Several assumptions and simplifications were made to facilitate the simulation. It was assumed that the sound source does not move and can be treated as a point source. It was also assumed that the noise added to the sound signal is random and uncorrelated.

Furthermore, we also applied a modification to the nature of the sound signal itself. Rather than creating a single-frequency sine wave, we generated a sound signal that includes a chirp frequency superimposed on the original sine wave. This allowed us to test our locating techniques across a range of frequencies simultaneously, thereby enhancing the robustness of our algorithm.

While these assumptions might not hold in a real-world scenario, they allow us to build a baseline simulation model that we can progressively refine and optimize. After validation of results, more complexities can be gradually introduced to closely simulate a real-world scenario.

3 System Design and Implementation

3.1 Subsystem Simulations

3.1.1 Time Delay Estimation

The first subsystem to be simulated is the time delay estimation. This process involves determining the time difference between the sound signal reaching different microphones. In our MATLAB simulation, we create a hypothetical sound signal and add noise to it. Then, we delay this signal by random amounts to simulate the sound reaching different microphones at different times. The objective of this simulation is to develop an algorithm that can accurately estimate the time delay, which is a crucial factor in locating the sound source.

3.1.2 Triangulation

The second subsystem to be simulated is triangulation. Once the time delay between the sound signals reaching the microphones is known, we can use triangulation to calculate the position of the sound source. This process involves creating a set of equations based on the known distances (i.e., the speed of sound times the time delay) and solving them to find the coordinates of the sound source. The simulation in MATLAB involves implementing and testing this algorithm.

3.2 Overall System Architecture

The overall system architecture includes two Raspberry Pi Zero W's, each connected to two Adafruit MEMS microphones, making a total of four microphones. The Raspberry Pi Zero W's act as the central processing units, responsible for acquiring the sound signals from the microphones, processing these signals, and estimating the time delays.

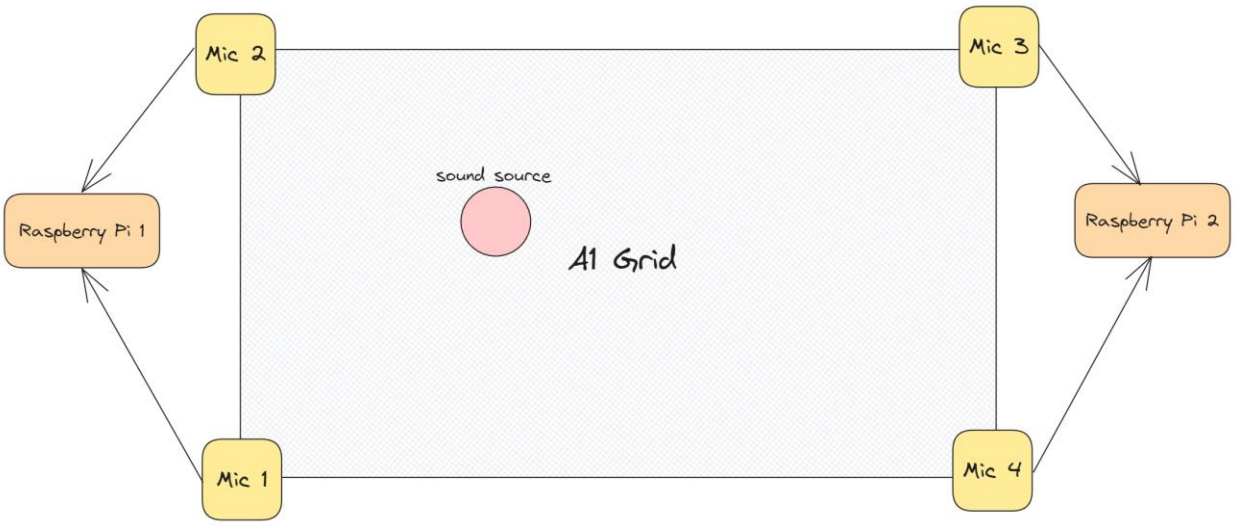


Figure 1: An illustration of the overall system architecture

The four microphones, being the primary sensing elements, are positioned onto the corners of the grid to capture the sound signals. They convert the acoustic signals into electrical signals which are then fed to the Raspberry Pi Zero W's for further processing. The Raspberry Pi Zero W's are interfaced with the microphones through the I2S (Inter-IC Sound) interface, which is a serial bus interface standard used for connecting digital audio devices.

The system architecture is designed in such a way that the two Raspberry Pi Zero W's work in tandem, each processing the signals from two microphones. The time delay calculations from each Raspberry Pi Zero W are then combined to triangulate the position of the sound source on the A1 grid.

3.3 Distributed Sensor Network Structure

The distributed sensor network is structured with a pair of microphones connected to each Raspberry Pi Zero W. The microphones are placed on each corner of the A1 grid, ensuring a wide coverage area for sound source detection. This structure is designed to leverage the benefits of a distributed sensor network, including increased system reliability, enhanced coverage, and improved localization accuracy.

In the MATLAB simulation model, we replicate this distributed sensor network structure. We assume that the microphones are at known, fixed positions on the grid. We then simulate the sound signals reaching the microphones at different times, based on the varying distances from

the sound source. This simulation setup allows us to emulate real-world conditions and test the effectiveness of our time delay estimation and triangulation algorithms.

3.4 Time Difference of Arrival Calculation

The Time Difference of Arrival (TDOA) calculation is a fundamental aspect of our system, providing the data necessary for the triangulation process. TDOA refers to the difference in the arrival times of the sound signal at various microphones. This difference is a result of the varying distances of the microphones from the sound source.

In our MATLAB simulation, we simulate the TDOA by introducing random delays to the hypothetical sound signal. These delays emulate the time differences that would occur in a real-world scenario when a sound signal reaches different microphones at different times.

To estimate the TDOA, we employ the method of cross-correlation. Cross-correlation is a measure of similarity between two signals as a function of the time-lag applied to one of them. By cross correlating the signals received at two microphones, we can determine the time delay between them.

In the MATLAB simulation, we implement an algorithm that uses cross-correlation to estimate the TDOA from the delayed signals. This simulation allows us to validate and refine this algorithm, ensuring it can accurately estimate the TDOA. This accuracy is critical, as the TDOA estimates are used in the triangulation process to calculate the position of the sound source with high precision.

4 Simulation Results & Analysis

4.1 Signal Simulation

As mentioned above, for the simulation scenario, seeing as there weren’t any actual microphones and hence no actual recorded signals, the recorded signals had to be simulated based on the generated sound source. The figure below shows the outcome of the simulated signals with no noise added.

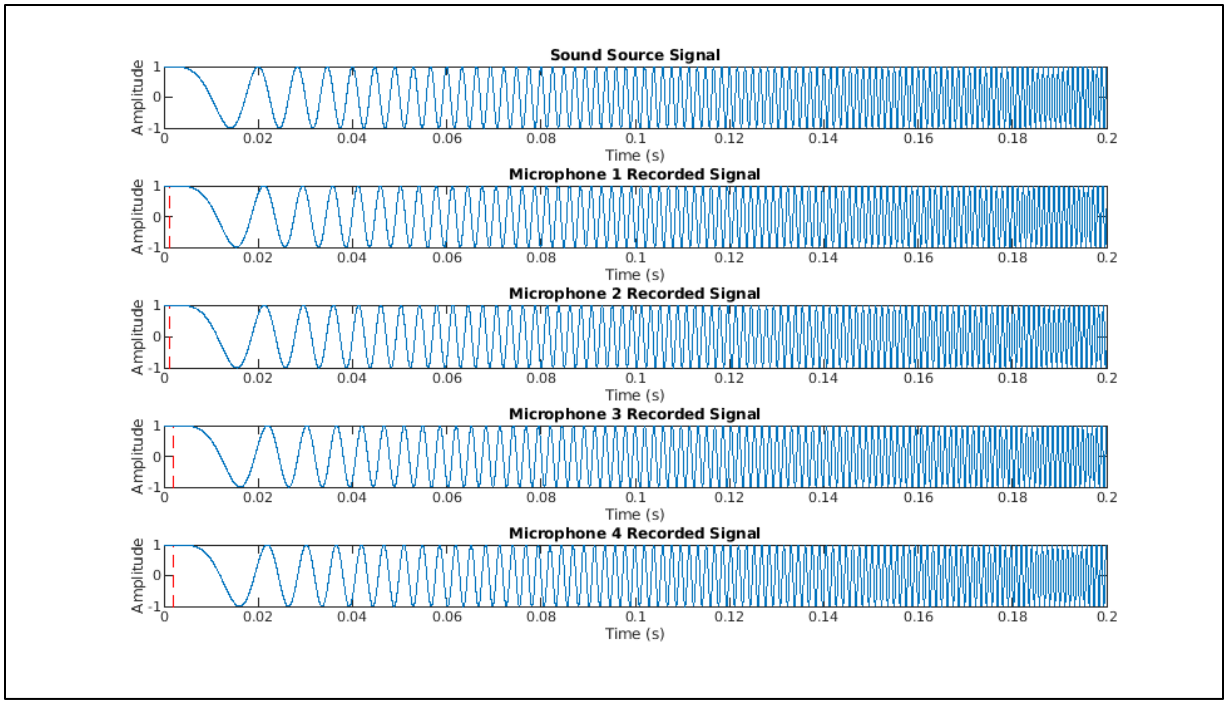


Figure 2: Figure of plots showing the simulated sound source signal and 'recorded' signals with no added noise. The dashed red line shows the time delay simulated in the signal.

Random noise was then added to each simulated, 'recorded' signal. See the figure below.

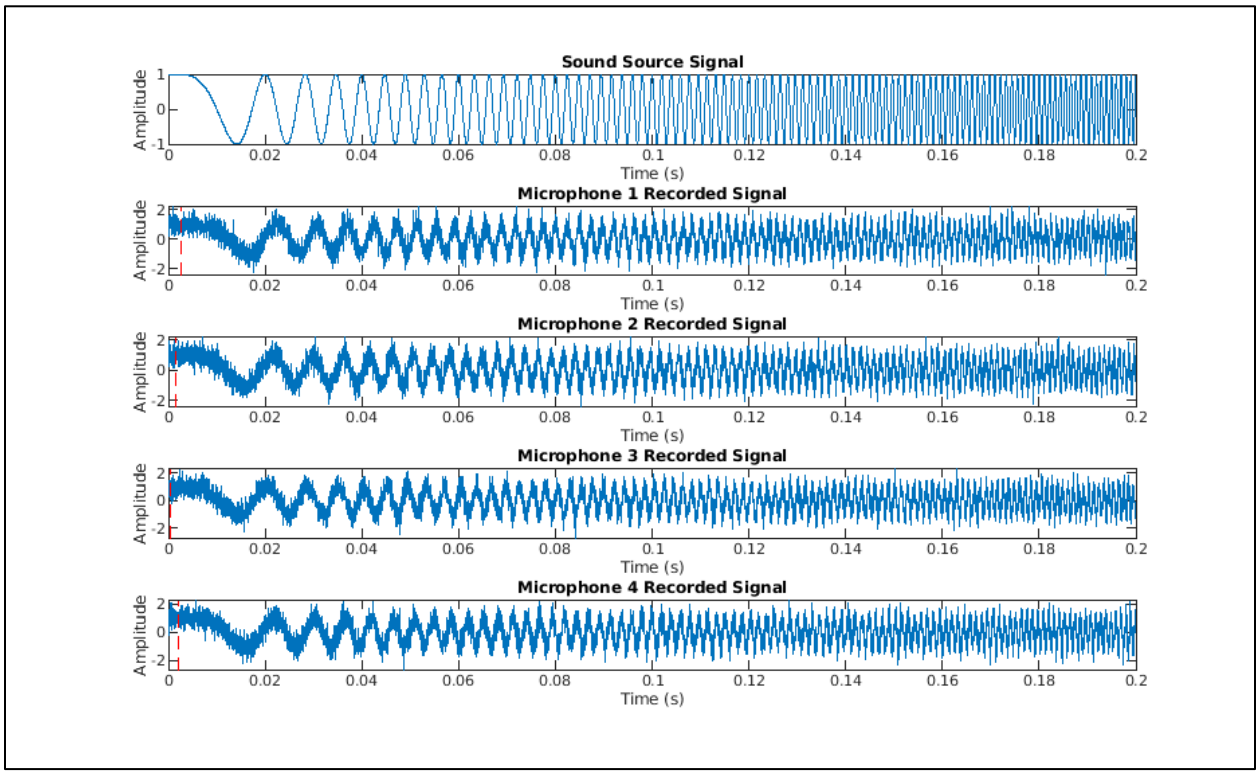


Figure 3: Figure of plots showing the simulated sound source signal and 'recorded' signals with added random noise.

The plots shown above clearly show that each of the signals is a chirp signal, which is just a sinusoidal wave with increasing frequency.

4.2 Time Difference of Arrival Testing

4.2.1 TDOA accuracy for varying amplitudes of added noise

Microphone Pair\Noise Amplitude	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7
Reference Mic and Mic 2	96.45	96.45	96.45	96.45	96.45	93.23	96.45	99.66
Reference Mic and Mic 3	98.75	98.75	98.75	98.75	98.75	98.75	98.75	98.75
Reference Mic and Mic 4	95.99	95.99	95.99	95.99	99.84	91.82	91.82	95.99
Average	**97.06**	**97.06**	**97.06**	**97.06**	**98.35**	**94.6**	**95.67**	**98.13**

Table 1: Table showing the accuracy of TDOA calculations (in %) for signals with varying noise signal amplitude between different microphone pairs. The above data was recorded with the sound source at position (0.522, 0.618), and the duration of the signal was set to 2 seconds for each iteration.

Each of the tests shown in the table above were run multiple times (more than what is shown in the table). However, for all of the noise amplitudes except 0.6 and 0.7, the values for accuracy didn't change, so including the results in the table would be wasteful. Seeing as the values for 0.6 and 0.7 did change, an average of their average accuracies was calculated:

- **Noise Amplitude = 0.6:**
  - **Average** = 96.77%
- **Noise Amplitude = 0.7:**
  - **Average** = 98.84%

The table above shows the results obtained from the time difference of arrival (TDOA) algorithm in our simulation. The results above are for varying levels of noise.

These same tests were run for signals at single frequencies, but the data is not shown because the results were identical to that shown above. It should also be noted that the noise added was random and set to change with every iteration to simulate real world noise.

4.2.2 TDOA accuracy for signals with varying duration

Another test to determine the accuracy of the TDOA algorithm is to test the algorithm by keeping the same signal but varying the duration of the signal. The table below shows the results from the test. The algorithm was run 4 times for each duration and the average accuracies are cumulated in the final row of the table. (Highlighted in green)

Microphone Pair\Duration	0.05	0.1	0.5	1	2	4
ref & 2	99.66	99.66	96.45	96.45	96.45	96.45
ref & 3	99.9	99.9	99.9	98.75	97.4	98.75
ref & 4	95.99	95.99	95.99	99.84	95.99	95.99
Average	**98.52**	**98.52**	**97.45**	**98.35**	**96.61**	**97.06**
ref & 2	96.45	99.66	96.45	96.45	96.45	96.45
ref & 3	97.4	98.75	98.75	98.75	98.75	97.4
ref & 4	91.82	95.99	95.99	95.99	95.99	95.99
Average	**95.22**	**98.13**	**97.06**	**97.06**	**97.06**	**96.61**
ref & 2	93.23	99.66	96.45	99.6	96.45	96.45
ref & 3	97.4	99.9	97.4	98.75	98.75	98.75
ref & 4	91.82	91.82	95.99	95.99	95.99	95.99
Average	**94.15**	**97.13**	**96.61**	**98.11**	**97.06**	**97.06**
ref & 2	96.45	96.45	93.23	96.45	96.45	96.45
ref & 3	97.4	98.75	98.75	98.75	98.75	98.75
ref & 4	91.82	99.84	91.82	91.82	95.99	95.99
Average	**95.22**	**98.35**	**94.6**	**95.67**	**97.06**	**97.06**
TOTAL AVERAGE	95.78	98.03	96.43	97.3	96.95	96.95

Table 2: Table showing results of TDOA test with varying signal duration. All results are accuracies in percentage (%). The label ref & n means that the result shown is the accuracy of the TDOA between the reference microphone and microphone n.

As seen in the table above, the highest accuracy was recorded when the signal had a duration of 0.1 seconds (s). The lowest accuracy was when the signal had the shortest duration of $0.05s$. The combined average over all the durations is 96.91%.

The last test of the TDOA algorithm compares accuracies over varying source positions, while keeping the noise amplitude and the signal duration the same. The results are recorded in the table below. The coordinates for the sound source are given in the top row of the table, the grid size is 0.594 m (x-axis) by 0.841 m (y-axis).

Microphone Pair/Position	(0.3113, 0.4946)	(0.5791, 0.0653)	(0.1296, 0.5187)	(0.4888, 0.3358)	(0.3144, 0.8234)
ref & 2	91.8	98.02	95.56	93.87	98.84
ref & 3	96.37	95.26	73.1	90.91	98.31
ref & 4	52.92	97.04	94.08	96.76	0
Average	80.36	96.77	87.58	93.85	65.72

As seen in the final column, the accuracy for the combination of the reference microphone and microphone 4 was calculated at 0% when the sound source was at the coordinates (0.3144, 0.8234). This might be a unique case that causes an error with the algorithm, or within the accuracy calculation. To determine if this was an issue the triangulation algorithm was run for

the same signal at the same coordinates. The resulting accuracy was above 95% and the position was accurate within the defined ATP limits. For this reason and because of the limited time which was given to complete the simulation, this error was tabled for evaluation at a later stage, as it does not cause detrimental error in the overall system.

4.3 Triangulation/Multilateration

4.3.1 Triangulation accuracy for varying amplitude of added noise signal

The first test that was performed to test the triangulation algorithm used different values for the amplitude of the simulated noise added to the source signal. This test is used to determine the effect of increasing noise on the triangulation algorithm accuracy. See the figures and tables below for results.

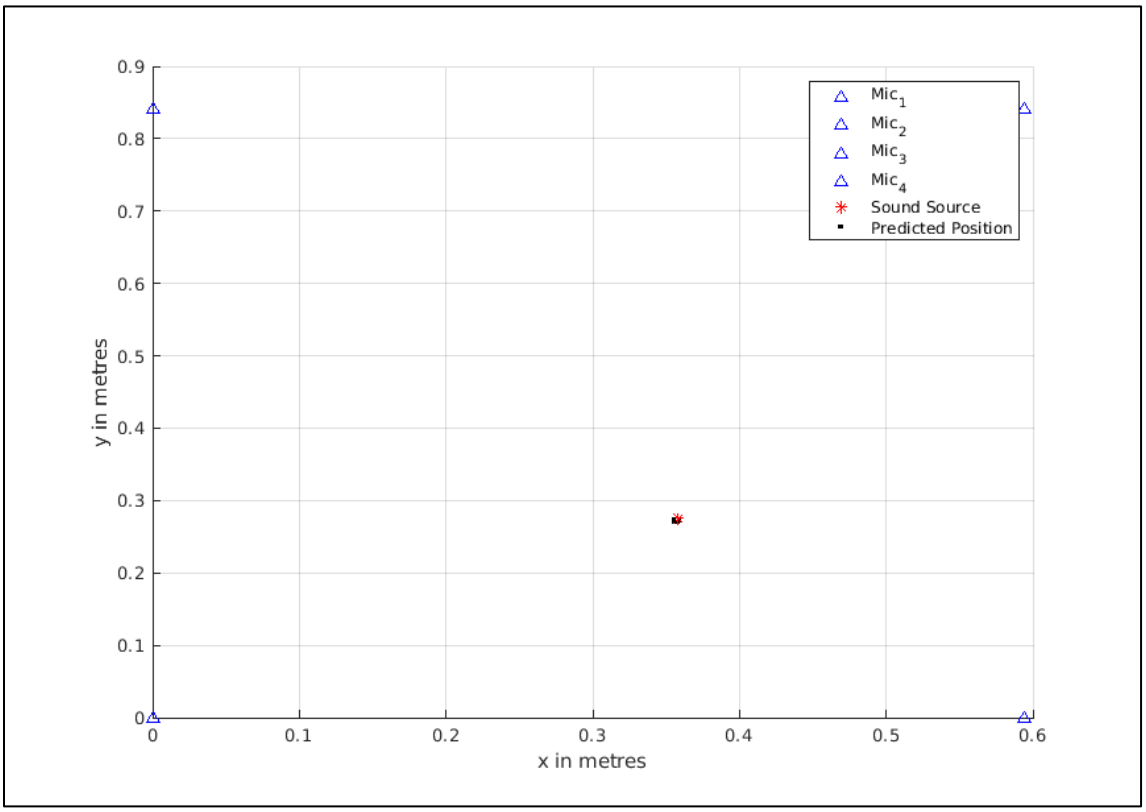


Figure 4: Simulated grid with actual sound source position as (*) and triangulated positions as (.) for varying noise amplitudes.

The results obtained from the varying noise amplitude test are shown in the table below.

Noise Signal Amplitude	0	0.1	0.2	0.3	0.4
Accuracy	99.532%	99.532%	99.498%	99.532%	99.336%
Distance from actual position (mm)	5	5	5	5	7

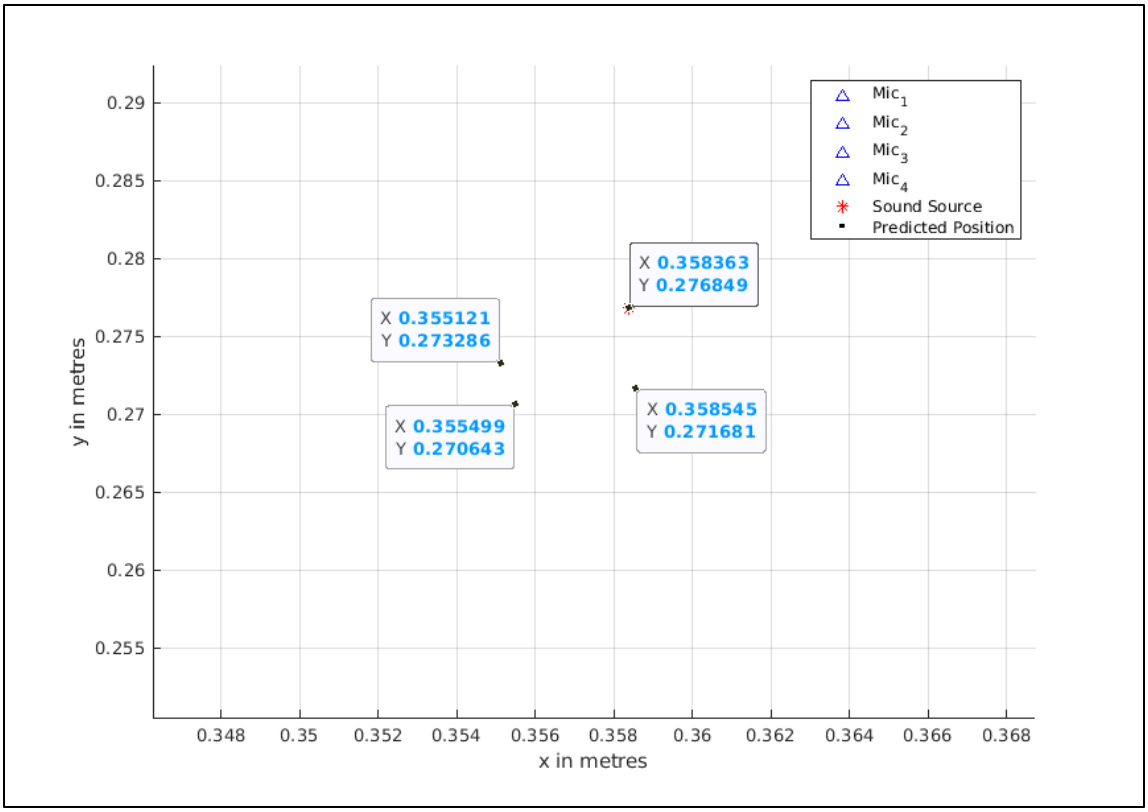


Figure 5: Zoomed in Figure of Figure 2 above, showing the labelled positions of the points.

4.3.2 Triangulation accuracy for varying sound source position

The second test of the triangulation algorithm is to observe how changing the position of the sound source within the grid affects the accuracy of the triangulation algorithm. The sound source was moved within the grid to 10 random locations. The accuracy of the triangulation algorithm across these different sound source locations is shown in the table below.

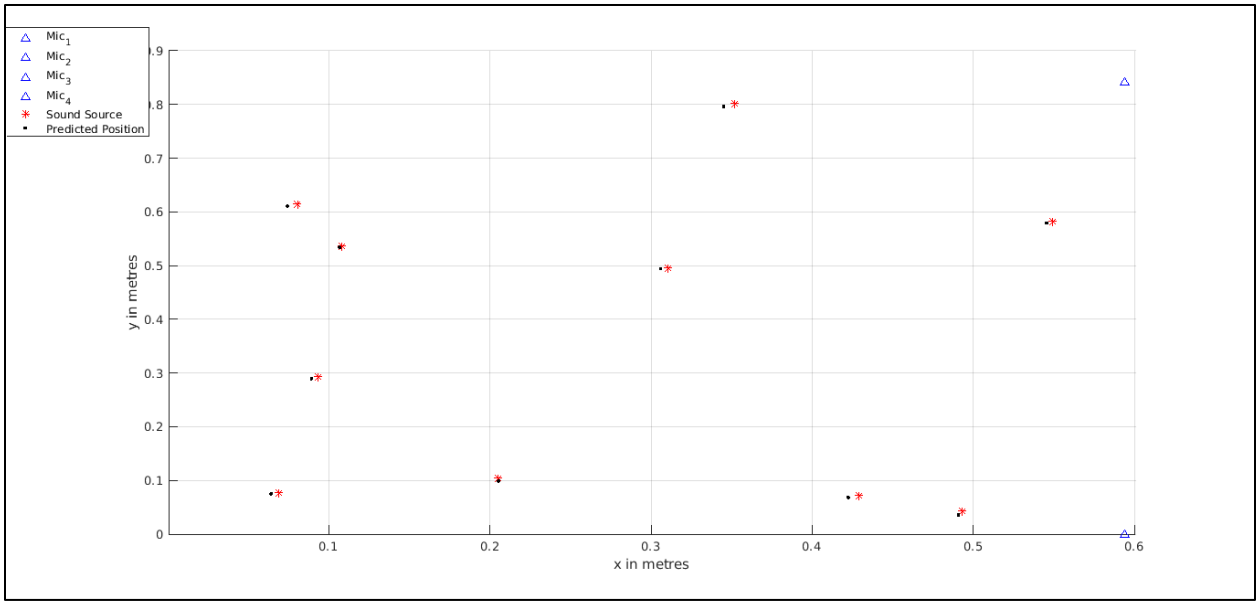


Figure 6: Figure showing 10 different sound source locations (*) and respective triangulated locations (.)

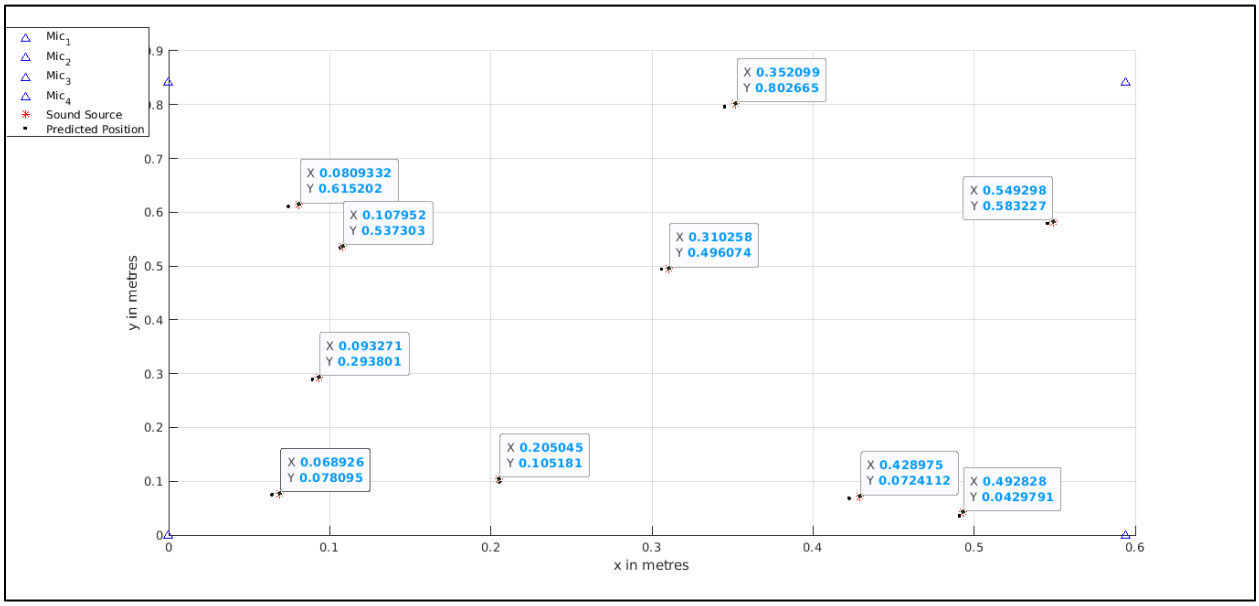


Figure 7: Identical to figure 4 but showing coordinates of sound source.

Source position	(0.31, 0.496)	(0.081, 0.615)	(0.108, 0.537)	(0.352, 0.803)	(0.093, 0.294)	(0.069, 0.078)	(0.205, 0.105)	(0.429, 0.072)	(0.549, 0.583)	(0.493, 0.043)
Accuracy	99.55 %	99.22%	99.63%	99.08%	99.45%	99.47%	99.42%	99.24%	99.47%	99.29%
Position error (mm)	5	8	4	9	6	5	6	8	5	7

The table above shows the accuracy of the calculated position from Multilateration vs. the actual position.

4.3.3 Triangulation Precision Test

To test the precision of the triangulation algorithm, everything was kept the same in the signal except for the noise, which is random simulated noise added to the sound source signal. The noise changes for each 'recorded' signal. The position of the sound source was held constant at the coordinates $(0.522, 0.618)$. The algorithm was then run 10 times to track the spread of the data caused by varying noise. This is to simulate a precision test in real life, where everything possible will be control variables, the only uncontrollable variable being the noise, which would vary randomly between triangulation executions. The figure below shows the sound source with the triangulated positions scattered close by.

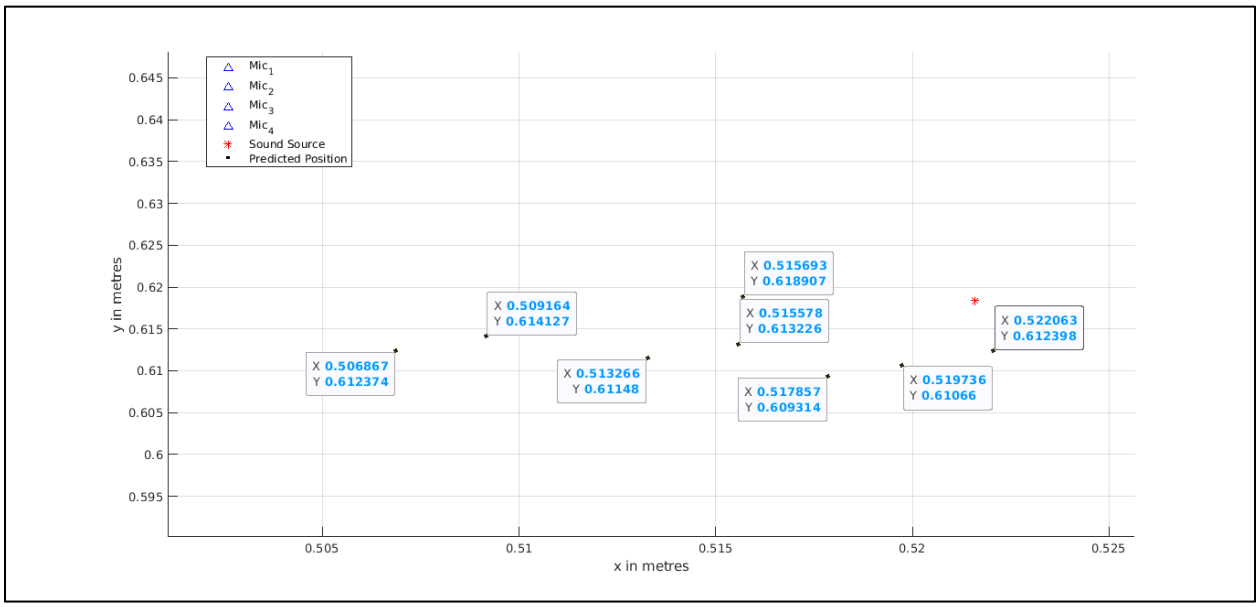


Figure 8: Figure showing the sound source position (*) and triangulated positions (.) for the triangulation precision test. The figure is zoomed in by a factor of 23 from the actual grid size, to show the scatter of the triangulated positions.

The table below shows the results from the test illustrated in figure 8 above.

Execution #	1	2	3	4	5	6	7	8	9	10
Accuracy (%)	98.947	99.228	98.454	99.426	99.228	99.043	99.412	99.228	98.723	99.224
Distance from actual position (mm)	11	8	16	6	8	10	6	8	13	8
Triangulated Pos. *(x, y)* coordinates	(0.513, 0.611)	(0.516, 0.613)	(0.507, 0.612)	(0.516, 0.619)	(0.516, 0.613)	(0.518, 0.609)	(0.522, 0.612)	(0.516, 0.613)	(0.509, 0.614)	(0.52, 0.611)

To calculate the precision from the data shown above, the average deviation from the mean position of all the triangulated positions was calculated:

$$Average\ position = (\bar{x}, \bar{y}) = \left(\frac{x_1 + x_2 + \cdots + x_n}{n}, \frac{y_1 + y_2 + \cdots + y_n}{n}\right)$$
$$= \left(\frac{0.513 + 0.516 + 0.507 + 0.516 + 0.516 + 0.518 + 0.522 + 0.516 + 0.509 + 0.52}{10}, \frac{0.611 + 0.613 + 0.612 + 0.619 + 0.613 + 0.609 + 0.612 + 0.613 + 0.614 + 0.611}{10}\right)$$
$$= (0.5153, 0.6127)$$

Hence the deviation from the mean can be calculated: $dev\ from\ mean = (|\bar{x} - x|, |\bar{y} - y|)$

These values are shown in the table below.

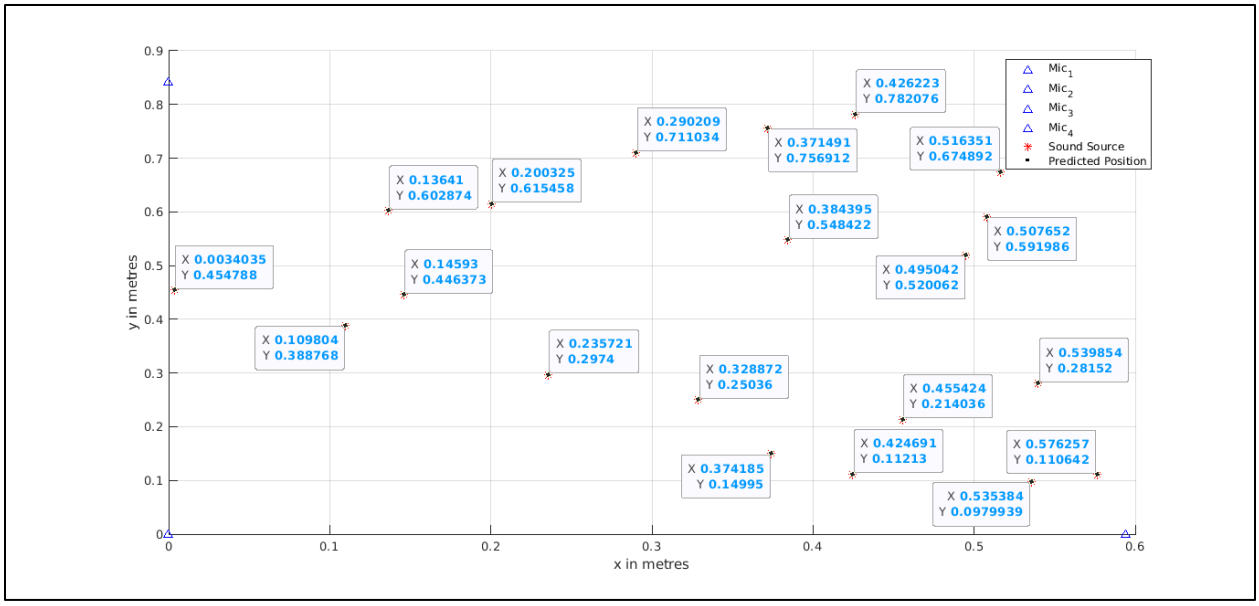
Triangulated Pos. (m)	(0.513, 0.611)	(0.516, 0.613)	(0.507, 0.612)	(0.516, 0.619)	(0.516, 0.613)2	(0.518, 0.609)	(0.522, 0.612)	(0.516, 0.613)3	(0.509, 0.614)	(0.52, 0.611)
Deviation from mean (mm)	(2.3, 1.7)	(0.7, 0.3)	(8.3, 0.7)	(0.7, 6.3)	(0.7, 0.3)	(2.7, 3.7)	(6.7, 0.7)	(0.7, 0.3)	(6.3, 1.3)	(4.7, 1.7)

And finally, the mean deviation from the mean position is calculated as the mean of the deviations in the above table. This yields a mean deviation of 3.38 mm in the x direction and 1.7 mm in the y direction. Finding the mathematical norm of theses 2 points gives the **overall precision** as a deviation from the mean position:

$$norm\ (x_{dev}, y_{dev}) = \sqrt{x_{dev}^2 + y_{dev}^2} = \sqrt{3.38^2 + 1.7^2} = 3.78\ mm$$

4.3.4 Triangulation 100% Accuracy test

This tests the accuracy of the isolated triangulation function. The sound source is moved to different locations within the grid and the actual TDOA (NOT CALCULATED) is passed into the triangulation algorithm. This should yield 100 % accuracy in the triangulated positions. See figure below for the execution of this test:



As seen in the above diagram, the accuracy was 100 % for all of the points evaluated.

5 Evaluation (ATPs)

For the ATPs in this progress report, only the procedures that apply to the simulation phase are evaluated. The rest of the ATPs will be evaluated with the physical set-up and testing of the system.

5.1 ATPs for TDOA

5.1.1 TDOA Noise variation

This ATP can only be performed during the simulation phase.

Add random noise to the signal, varying the amplitude of the noise from 0 to 40% of the amplitude of the sound source's signal. Record the calculated TDOA and compare it to the actual TDOA. The accuracy should not drop below 95% for this ATP to be met.

5.1.2 TDOA duration variation

Change the duration of the source signal in the following way:

$Duration\ (in\ seconds) = n, 2n, 4n, 8n, 16n, 32n$. Where n is a short realizable period of time, this will vary between the simulated system and the physical system, as the duration of the sound source signal can be controlled much more precisely and accurately in the simulation. In the simulation the duration should start at around $n < 0.1\ seconds$. In the physical set up, a bigger value for n should be chosen, around $n = 1\ second$.

Record the calculated TDOA each time and compare it to the actual TDOA. The accuracy should not drop below 95% to pass this ATP.

5.1.3 TDOA source position variation

Vary the position of the sound source within the grid, keep the duration of the signal as constant as possible and reduce noise as much as possible. The purpose of this ATP is to ensure that the TDOA algorithm maintains accuracy when the sound source is anywhere withing the grid.

Run this test a minimum of 5 times, position the sound source near each corner of the grid and near the center of the grid. Record the calculated TDOA each time and compare it to the actual TDOA. The accuracy should not drop below 95% to pass this ATP.

5.2 ATPs for Triangulation/Multilateration

5.2.1 Triangulation Noise Variation

This ATP can only be performed during the simulation phase.

Add random noise to the signal, varying the amplitude of the noise from 0 to 40% of the amplitude of the sound source's signal. Compare the Triangulated position to the actual position of the sound source. The accuracy should not drop below 98% or $\pm 15\text{mm}$ for this ATP to be passed.

5.2.2 Triangulation sound source position variation

Vary the position of the sound source within the grid, keep the duration of the signal as constant as possible and reduce noise as much as possible. The purpose of this ATP is to ensure that the Triangulation algorithm maintains accuracy when the sound source is anywhere withing the grid.

Run this test a minimum of 5 times, position the sound source near each corner of the grid and near the center of the grid. Record the triangulated position each time and compare it to the actual sound source position. The accuracy should not drop below 98% or $\pm 15mm$ to pass this ATP.

5.2.3 Triangulation precision

Fix the sound source position, the signal duration and everything else. The only variable that should be allowed to change is the noise in the system which cannot be controlled in the physical system and in the simulated system should be simulated to match a physical scenario as much as possible.

Execute the triangulation a minimum of 10 times and record the triangulated positions. Calculate the precision of the record triangulated positions. The precision should be calculated using deviation from a mean position and to pass this ATP, the precision should not be greater than 5 mm.

5.3 Table of Evaluated ATPs

ATP	TDOA Noise variation	TDOA duration variation	TDOA source position variation	Triangulation Noise variation	Triangulation sound source position variation	Triangulation precision
Required	>95%	>95%	>95%	>98% OR < 15mm	>98% OR < 15mm	<5mm
Simulation result	97.1	96.91	84.86	99.49% OR 5.4 mm	99.38% OR 6.3 mm	3.78 mm
ATP Met?	YES	YES	NO	YES	YES	YES

The only ATP that was not met in the simulation was the TDOA source position variation. To validate whether this ATP could be provisionally passed, the propagation of the error through to the triangulation algorithm was evaluated. Using the same sound source position and exactly the same signal, the triangulation algorithm returned triangulated positions that passed the ATP for triangulation. Hence the TDOA source position variation ATP will be considered provisionally passed.

6 Conclusion

Overall, the simulation was a success, almost all of the ATPs were passed successfully. The ATP that wasn't met didn't cause lasting errors in the whole system. Other conclusions are drawn in the individual sections.

7 References

Boschen, D. (2018, October 4). *GPD CA Signal Aquisition*. Retrieved from StackExchange: https://dsp.stackexchange.com/questions/52370/gps-ca-signal-acquisition

Fernandez, A., & Dang, D. (2013). Chapter 8 - Analog: The Infinite Shades of Gray. In A. Fernandez, & D. Dang, *Getting Started with the MSP430 Launchpad* (pp. 81-125). Retrieved from https://www.sciencedirect.com/science/article/abs/pii/B978012411588000008X

Fromaget, P. (2019). *How to Sync Time with a Server on Raspberry Pi*. Retrieved from RaspberryTips: https://raspberrytips.com/time-sync-raspberry-pi/

INTERSPEECH2021. (2022, January 10). *Time Delay Estimation for Speaker Localization Using CNN-Based Parametrized GCC-PHAT Features - (Oral presentation)*. Retrieved August 16, 2023, from YouTube: youtube.com/watch?v=q4o9eGjE658

Khan, K. (2023, February 1). *Time Synchonization on the Raspberry Pi*. Retrieved from Hackster.io: https://www.hackster.io/kamaluddinkhan/time-synchronization-on-the-raspberry-pi-d11ece

MATLAB. (2014, December 4). *Determining Signal Similarities*. Retrieved August 16, 2023, from YouTube: https://www.youtube.com/watch?v=oCcUm0_rUJw

Mohan, J., Pandu, V., & Kanagasabai, A. (2019). Real-Time ECG-Based Biometric Authentication System. In S. Geetha, & A. V. Phamila, *Countering Cyber Attacks and Preserving the Integrity and Availability of Critical Systems* (pp. 275-289). doi:10.4018/978-1-5225-8241-0

NI. (2023, February 21). *LabVIEW Sound and Vibration Toolkit*. Retrieved from ni: https://www.ni.com/docs/en-US/bundle/labview-sound-and-vibration-toolkit/page/svtconcepts/svsyncsampling.html

Rajashekar, U., & Simoncelli, E. P. (2009). Chapter 11 - Multiscale Denoising of Photographic Images. In U. Rajashekar, & E. P. Simoncelli, *The Essential Guide to Image Processing* (2nd ed., pp. 241-261). Retrieved from https://www.sciencedirect.com/science/article/abs/pii/B9780123744579000111

TechOverflow. (n.d.). *How to enable or disable NTP time synchronization on the Raspberry Pi*. Retrieved from TechOverflow: https://techoverflow.net/2023/03/14/how-to-enable-or-disable-ntp-time-synchronization-on-the-raspberry-pi/

Tips, R. (2020, March 5). *How to sync time with a server on Raspberry Pi?* Retrieved August 17, 2023, from YouTube: https://www.youtube.com/watch?v=Nyr5DI0fuAY