TALLINN UNIVERSITY OF TECHNOLOGY
SCHOOL OF ENGINEERING
Department of Electrical Power Engineering and Mechatronics

RESEARCH AND DEVELOPMENT OF 3D ULTRASONIC SENSOR TO DETERMINE OBJECT POSITION IN THREE-DIMENSIONAL SPACE

3D ULTRAHELIANDURI UURIMINE JA ARENDAMINE OBJEKTI ASUKOHA MÄÄRAMISEKS KOLMEMÕÕTMELISES RUUMIS

MASTER THESIS

Student: Farzan Saleem

Student code: 195989 MAHM

Supervisor: Priit Ruberg

Tallinn 2021

AUTHOR’S DECLARATION

Hereby I declare, that I have written this thesis independently.
No academic degree has been applied for based on this material. All works, major viewpoints and data of the other authors used in this thesis have been referenced.

18.05.2021
Author: Farzan Saleem
/signature /

Thesis is in accordance with terms and requirements

“.......” .... 20….

Supervisor: Priit Ruberg
/signature/

Accepted for defence

“.......”....................20… .

Chairman of theses defence commission: ....
/name and signature/

Non-exclusive Licence for Publication and Reproduction of Graduation thesis[1]

I, Farzan Saleem (date of birth: 30/08/1992) hereby

1. grant Tallinn University of Technology (TalTech) a non-exclusive license for my thesis 3D ultrasonic sensor to determine object position in three-dimensional space,

supervised by

researcher, Priit Ruberg,

1.1 reproduced for the purposes of preservation and electronic publication, incl. to be entered in the digital collection of TalTech library until expiry of the term of copyright;

1.2 published via the web of TalTech, incl. to be entered in the digital collection of TalTech library until expiry of the term of copyright.

1.3 I am aware that the author also retains the rights specified in clause 1 of this license.

2. I confirm that granting the non-exclusive license does not infringe third persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

---

[1] *Non-exclusive Licence for Publication and Reproduction of Graduation Thesis is not valid during the validity period of restriction on access, except the university`s right to reproduce the thesis only for preservation purposes.*

______________ (*signature*)

______________ (*date*)

Department of Electrical Power Engineering and Mechatronics

THESIS TASK

Student: Farzan Saleem, 195989 MAHM

Study programme, MAHM02/18 - Mechatronics

main speciality: Mechatronics

Supervisor(s): Priit Ruberg, researcher

Co-supervisor: Even Sekhri, engineer

Thesis topic:

Research and development of 3D ultrasonic sensor to determine object position in three-dimensional space
3D ultrahelianduri uurimine ja arendamine objekti asukoha määramiseks kolmemõõtmelises

Thesis main objectives:

1. Development of 3D ultrasonic sensor
2. Improving the existing ultrasonic technology
3. Provide a solution to achieve an accurate and economical way of detecting objects in 3D space.

Thesis tasks and time schedule:

No	Task description	Deadline
1.	Study of the collected data. Design research or estimation for the prototype.	January 2021
2.	Thesis Writing, Hardware Design development and firmware programming	March 2021
3.	Thesis Writing and Python application programming	April 2021
4.	Development of the prototype	June 2021

Language: English **Deadline for submission of thesis:** 18.05.2021

Student: Farzan Saleem ....

Supervisor: Priit Ruberg ....

CONTENT

LIST OF FIGURES 6

PREFACE 8

LIST OF ABBREVIATIONS AND SYMBOLS 9

1 INTRODUCTION 10
- 1.1 Duplex Theory 12
- 1.2 Acoustic Location 12
- 1.3 Ultrasonic Sensor 13
  - 1.3.1 Working of Ultrasonic Sensor 13
  - 1.3.2 Industrial Applications of Ultrasonic Sensor 14
  - 1.3.3 Liquid Level Sensing 15
  - 1.3.4 Smart Waste Collection 15
  - 1.3.5 Ultrasonic Sensors in Parking Applications 16
- 1.4 Problem Statement 17

2 LITERATURE OVERVIEW 19
- 2.1 Key Information 22
- 2.2 Literature Analysis 23

3 METHODOLOGY 24

4 SIMULATION 27
- 4.1 Environment Setup 27
- 4.2 Results 32

5 DEVELOPMENT OF A SYSTEM 36
- 5.1 Hardware Design 38
- 5.2 Software Architecture 40

6 ASSEMBLY 45

7 TESTING 46
- 7.1 Case 1 46
- 7.2 Case 2 47
- 7.3 Case 3 49
- 7.4 Case 4 50

8 SUMMARY 52
- 8.1 Future work 52

9 KOKKUVÕTE 54
- 9.1 Edasiarendused 54

References 56

APPENDICES 58

LIST OF FIGURES

Figure 1. Visual example of azimuth and elevation 10
Figure 2. Cues in mammals to determine the location of the source of sound 11
Figure 3. Active acoustic location by bat 13
Figure 4. Working principle of ultrasonic 14
Figure 5. Ultrasonic liquid level sensor 15
Figure 6. Ultrasonic parking sensor 16
Figure 7. Ultrasonic beam width comparison at different frequencies 17
Figure 8. 3-MIC sitting and 4-MIC sitting arrangement setup 19
Figure 9. Interaural delay geometry 24
Figure 10. Transducer beam directivity 25
Figure 11. Simulation flowchart 28
Figure 12. Properties of ultrasonic waves propagation medium in simulation 29
Figure 13. Properties of the ultrasonic transmitter used in the simulation 30
Figure 14. Orientation of the ultrasonic receivers used in the simulation 31
Figure 15. Ultrasonic simulation in wave2000 software
(a) initial bursts
(b) propagation in air
(c) echos 32
Figure 16. Echo plot of the data received from the simulation 33
Figure 17. 2D plot of actual position and detected position of object 35
Figure 18. Transformer-Driven method 36
Figure 19. Direct-Driven method 37
Figure 20. Bandpass filter calibrated at the center frequency of 40kHz 38
Figure 21. Functional block diagram of MAX232 39
Figure 22. Sensor prototype block diagram 40
Figure 23. Software flowchart 41
Figure 24. Position algorithm cases 43
Figure 25. Python GUI 44
Figure 26. Sensor Image 45
Figure 27. Case 1 setup image 46
Figure 28. Case 1 sensor output 47
Figure 29. Case 2 setup image 48
Figure 30. Case 2 sensor output 48
Figure 31. Case 3 setup image 49
Figure 32. Case 3 sensor output 50

Figure 33. Case 4 setup image 50
Figure 34. Case 4 sensor output 51
Figure 35. MEMS based ultrasonic TOF sensor 53

PREFACE

Ultrasonic has an innumerable number of applications, and the author is very glad that he had the opportunity to work in such an interesting direction as the development of a 3D ultrasonic sensor. The author has a passion for embedded systems, robotics, and researching on the sonar navigation system of mammals. The thesis is about the development of the hardware and software of the 3D ultrasonic sensor. The sensor is developed to detect the position of the object in three-dimensional space.

The author would like to express his deepest gratitude to researcher Priit Ruberg for the immense support provided to me whenever I needed to conduct my thesis and for guiding me to write the thesis report.

A special thanks goes to my family for their consistent support. This thesis is dedicated to my wife who has always believed in me and has supported me in all difficult phases of my life.

Finally, I would like to thank Prof. Mart Tamre for approving the work on this project and Estonia for all the moral support given to me to fulfill my dreams in the best way possible.

Keywords: Ultrasonic; Time of Flight; Sound Localization; Master Thesis

LIST OF ABBREVIATIONS AND SYMBOLS

3D	Three Dimension
TOF	Time of Flight
RTOF	Rough Time of Flight
IP65	Ingress Protection Code 65
DSP	Digital Signal Processor
MIC	Microphone
SRP-PHAT	Steered-Response Power Phase Transform
PC	Personal Computer
PS	Phase Shift
SNR	Signal Noise Ratio
UTA	Ultrasonic Transmitting Array
SUT	Single Ultrasonic Transmitter
URA	Ultrasonic Receiver Array
HPBW	Half Power Beam Width
FEM	Finite Element Method
TR	Time Reversal
PWM	Pulse width modulation
GUI	Graphical User Interface
MEMS	Microelectromechanical systems
CMUT	Capacitive micromachined ultrasonic transducers
DSP	Digital signal processor
ASIC	Application-specific integrated circuit

1 INTRODUCTION

Sound localization is the technique that is used to identify the direction of the sound source. That technique is also used by several species of animals known as Echolocating animals. These animals emit ultrasonic waves to the environment and listen to the echoes that bounce back from the various objects near them.The mechanism of sound localization has been studied well in the last century. These studies [1] have primarily examined azimuth and elevation localization accuracy using a variety of reporting techniques based on the timing analysis, intensity level-differences, spectral information, correlation analysis, and pattern matching between all receivers.

There are some terms and some experimental techniques that have been used in spatial hearing research and to understand sound localization in a better manner. The most common and direct method of testing sound localization can be to present sounds at different locations remote from the listener and allow the listener to judge those locations [2].

Azimuth: Azimuth refers to the horizontal plan at which the specific sound source is situated and the angular spread of the sound sources of interest in the horizontal plane shown below in Figure 1.

Elevation: Elevation refers to the vertical plane at which the specific sound source is situated, also called as Zenith, and the angular spread of the sound sources of interest in the vertical plane shown below in Figure 1.

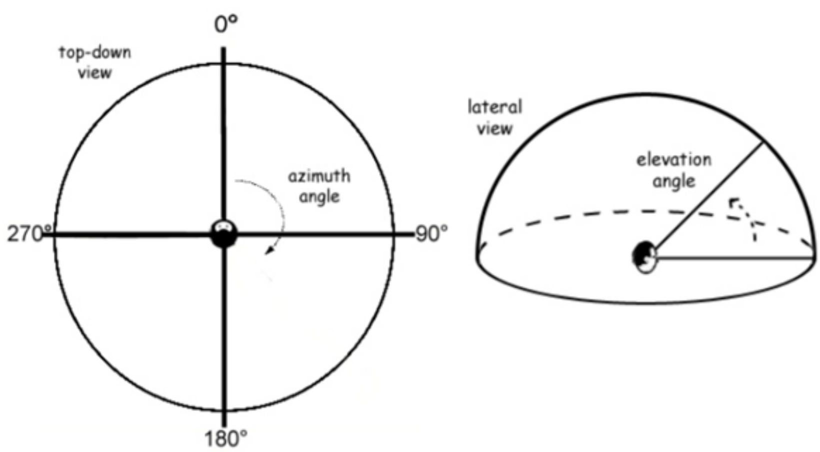

Figure 1: Visual example of azimuth and elevation [2]

In the bio life, the two ears on either side of the head act as receivers of sound waves. The distance between the two ears causes a time difference in the approaching of the sound waves. That timely difference helps to determine the location of the source. The azimuth of the source is determined by binaural factors involving both ears based on the time level difference but the elevation and the distance of the source is only determined monaurally using only one ear [3].

The binaural cues have the involvement of two ears and are defined as the ability to locate the horizontal position of the object using time arrival and intensity difference of the incoming sound between two ears. The auditory system in the mammals used monaural cues to analyze the elevation angle of the sound source. The elevation position of the sound source causes the head-related transfer function, due to interactions of the sound waves with the ear, the head, and the trunk of the head called the pinna. The pinna structure induces the change in the frequency profile of sound which allows the auditory system to reconstruct the elevation of the sound source.

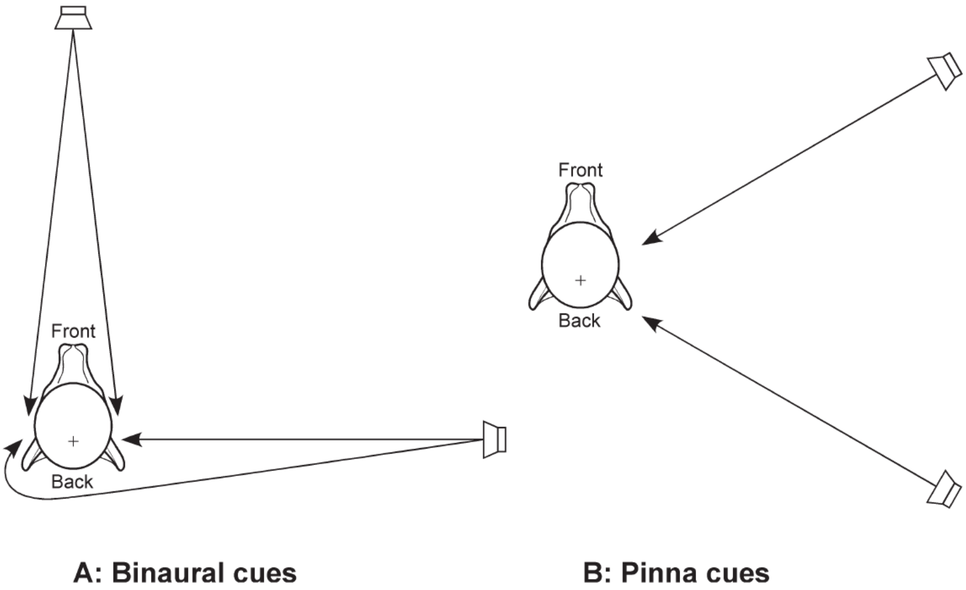

Figure 2: Cues in mammals to determine the location of the source of sound [3]

1.1 Duplex Theory

A sound source localization to the left or right of the relative to the midline plane is referred to as the "Horizontal Localization". The horizontal localization can be traced to work of Baron and lord Rayleigh, back in the 19th century where Rayleigh used the tuning fork as the sound source. During the experiment Rayleigh observed that the left-versus-right location of the tuning forks can be discriminated across a broad spectrum of frequencies. He realized that at relatively high frequencies, the head is larger than the wavelength of the sound and casts an acoustic shadow. This is the main reason that the ear close to the sound source has a greater sound level than the ear that is far, thereby providing a cue to the sound source location. The same effect was doubtful at relatively lower frequencies. At lower frequencies, the wavelength of the sound can be several times longer than the diameter of the human head so in this condition, the sound waves bend around the head and the difference in the level of the sound at two ears becomes negligible. This effect of sound on ears at different frequencies later termed as Duplex theory.

After a series of experiments, Rayleigh suggested that the difference in distance between the sound source and two ears introduces differential delays resulting in interaural phase difference [4].

1.2 Acoustic Location

Acoustic location is the sound localization via mechanical or electrical means. It is the use of sound to find the distance and position of its source or reflector. The acoustic location is further classified into two methods in which location can be done actively or passively.

Active acoustic location is the production of sound in order to create an echo, which is then analyzed to determine the position of the object.
Passive acoustic location is the detection of sound or vibration produced by the object being observed, which is then analyzed to determine the position of the object.
The most common examples of both types of acoustic locations can be found in mammal echolocation. Humans used the method of the passive acoustic location to determine the position of the source generator. But bats generate sound waves to create an echo to measure the location of their prey. Figure 3 shows active acoustic location used by bats.

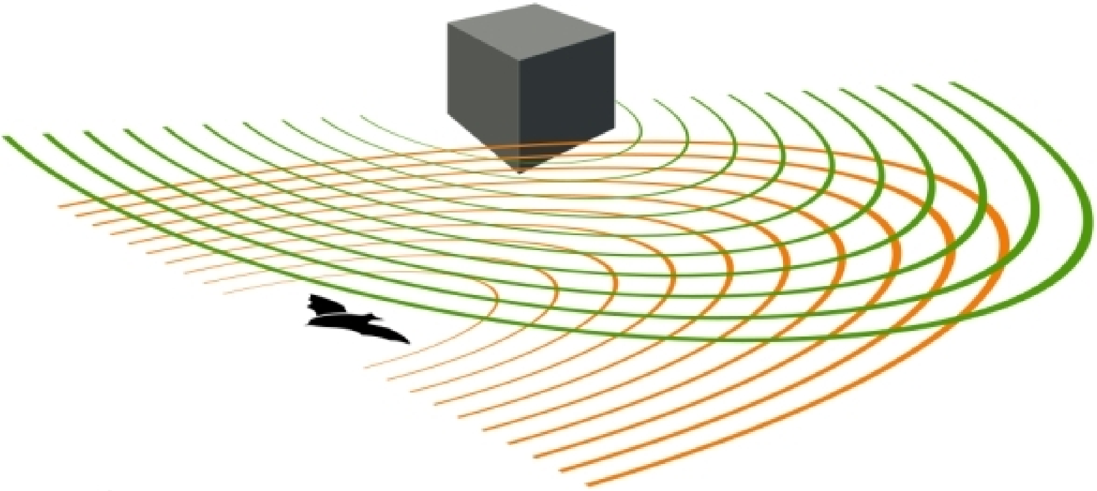

Figure 3: Active acoustic location by bat [5]

Sound Localization, its terms and techniques have been discussed in detail. Next subsection sheds light on the concept of ultrasonic sensor and the application of ultrasonic sensor.

1.3 Ultrasonic Sensor

Ultrasonic waves are sound waves whose frequencies generally exceed 20 kHz and they are inaudible to humans. Ultrasonic sensing is one of the most reliable ways to sense proximity and detect levels. The field of ultrasonic is still making strides towards perfection and it is widely used in several technologies e.g. car parking assistance, obstacle detection in robotics, etc. Ultrasonics are widely used as time of flight sensors to measure the distance or proximity of the object.

1.3.1 Working of Ultrasonic Sensor

The working principle of the ultrasonic sensor is to emit small, high-frequency sound pulses. These pulses travel at the speed of sound through the air. If they hit something, they're mirrored back to the sensor as echo signals, which the sensor uses to calculate the distance to the target based on the time between sending the signal and receiving the echo. Figure 4 shows the working principle of the ultrasonic time of flight measurement system.

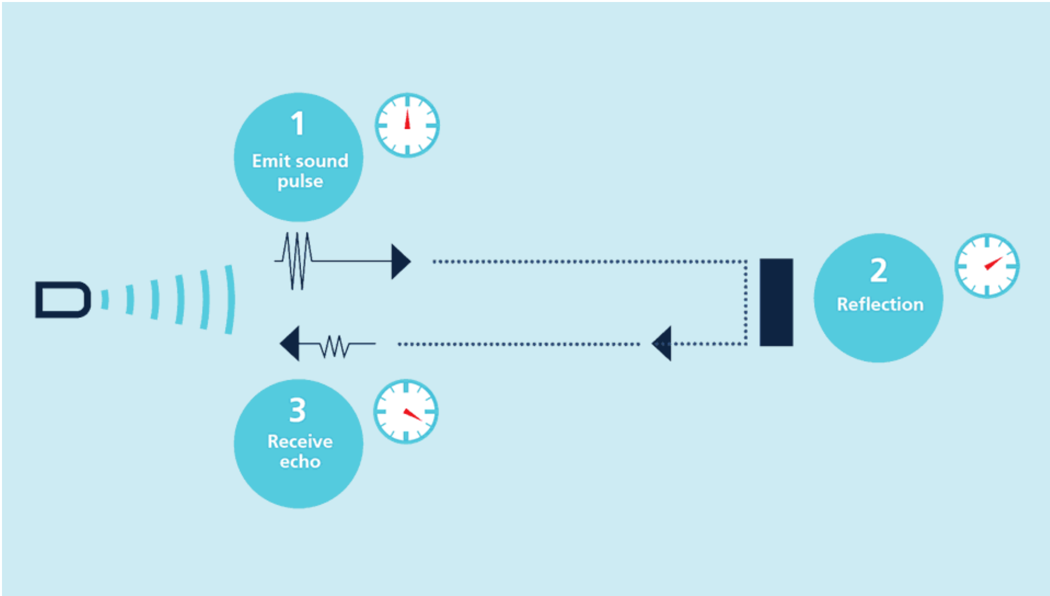

Figure 4: Working principle of ultrasonic [6]

1.3.2 Industrial Applications of Ultrasonic Sensor

Ultrasonic Sensor are widely used in industrial applications, they are characterized by their reliability and outstanding versatility: Ultrasonics Sensor are well equipped to solve even the most complex jobs related to object detection or level measurements even with millimetre precision, as their measuring method works reliably under almost all conditions.

Today many E-commerce companies are using robots to deliver their products to speed up the delivery process and cope up with consumer needs in the demanding market. Robots are used in the logistics of many companies e.g. Ocado or Kiva Robot from Amazon, ultrasonic sensors are used as obstacle detection in these robots.

The automation of the warehouse is growing day by day in many E-commerce, manufacturing, and retailer companies. In the Automotive manufacturing industry, automation can serve an efficient job to assemble, move and place the products about the facility, paint the parts as well as various other purposes [7].

There are several other applications of the ultrasonic sensors, such as anti-collision detection, object or people detection, contouring or profiling, presence detection, box sorting system, pallet detection, bottle counting on drink filling machines.

1.3.3 Liquid Level Sensing

Level measurements for corrosive liquids, fuel glues, resins, paints, rubbers, and plastics are among the applications for ultrasonic level sensors in the chemical industry. The liquid depth calculation is done by finding the distance between the transceiver and the surface of the liquid. A short ultrasonic pulse is transmitted by the sensor, the travel time of that pulse (the echo) to the liquid and back is to be measured. The fill level is measured by subtracting the distance measured by an ultrasonic sensor from the total depth of the tank. The assembly of the sensor is shown in Figure 5, it is very important to keep the position of the sensor at a fixed point above the stored water. This application is also beneficial in underwater submersion, and it greatly increases the lifespan of the sensor [8]. Figure 5 shows the liquid level sensor inside the container.

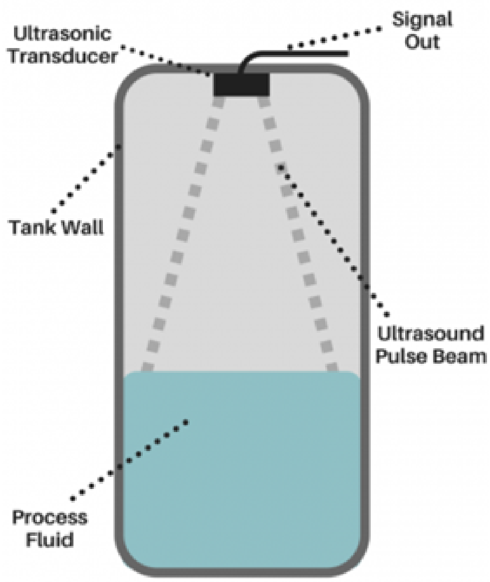

Figure 5: Ultrasonic liquid level sensor [8]

1.3.4 Smart Waste Collection

The ultrasonic sensor is one of the most common IoT platform products for waste level measurement applications. The waste level sensor uses ultrasonic technology to detect the level in waste bins and containers, a robust sensor tests the fill level, regardless of what has been deposited inside. As ultrasonic technology does not depend upon the type, shape, opacity, and environmental factors e.g. light intensity so it is the most suitable technology to use in the rugged environment like bins. The smart waste sensors send information about the level of waste or last collection to a data management system. Only Certain bins are marked for the waste collection and the vehicle only collects the full or overdue containers, do not visit all the containers. This application greatly reduces the collection trucks traveling time, human resource requirement, and fuel emission.

1.3.5 Ultrasonic Sensors in Parking Applications

Ultrasonic parking assist has long been used in Europe and other regions, where most parking is on roads or parallel parking on narrow private roads, well before the proliferation of Advanced driver-assistance systems functions. To confirm the presence of objects and measure the distance to obstacles, ultrasonic sensors emit ultrasonic waves, and a receiver detects waves reflected from obstacles.

Previously, systems gave warnings to drivers for vehicles to avoid colliding with hazards such as walls and other vehicles, but now, the adoption and use of automatic parking systems that regulate steering is growing. Ultrasonic parking assistance will continue to play an important role in automatic driving and parking. Figure 6 shows the ultrasonic parking assistance sensor.

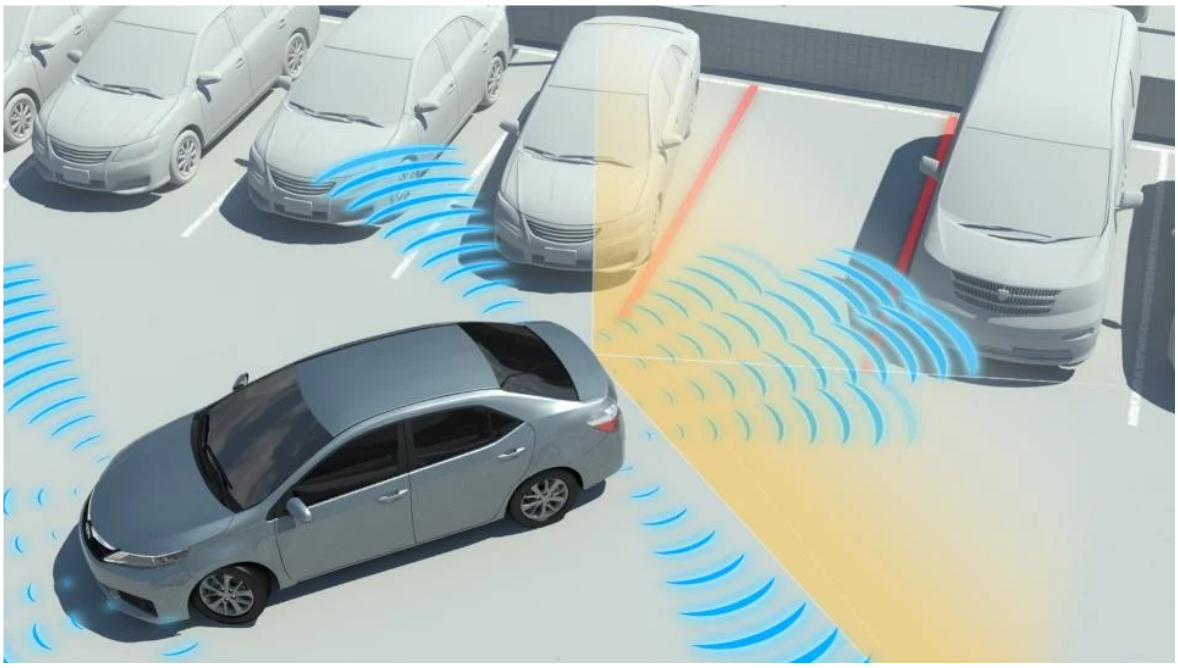

Figure 6: Ultrasonic parking sensor [9]

1.4 Problem Statement

Currently, the ultrasonic sensors are only available as Time of Flight (TOF) sensors as they are only capable of measuring the distance by calculating the TOF of the ultrasonic wave. The idea behind this thesis is to develop an algorithm and hardware for 3D ultrasonic sensors that can detect the position of objects in addition to distance. The sensor can be used for the navigation of autonomous cars and parking assistant applications.

The main constraint of ultrasonic technology is unwanted reflections from surfaces which make this technology limited for mapping techniques. The limitation can be suppressed by adding multiple receiver arrays and software filtering algorithms to extract meaningful data from the setup. The concept of the proposed technique is to replace a traditional single transmitter single receiver ultrasonic system with a single transmitter and multiple receiver array. The algorithm will extract the information from receivers and determine the distance, azimuth and elevation of the object.

The range and resolution of the ultrasonic waves depend upon their propagation frequency. The operating frequency of ultrasonic is inversely related to the width of the ultrasound beam. The higher the frequency, the narrower is the beam. The attenuation of the sound wave also depends upon the frequency of the ultrasonic wave. Ultrasonic sensors operating at high frequency have high attenuation thus they have very limited range and narrow beam as compared to ultrasonic operating at a lower frequency.The side-by comparison of the effect of frequency on the width of the beam is shown in Figure 7.

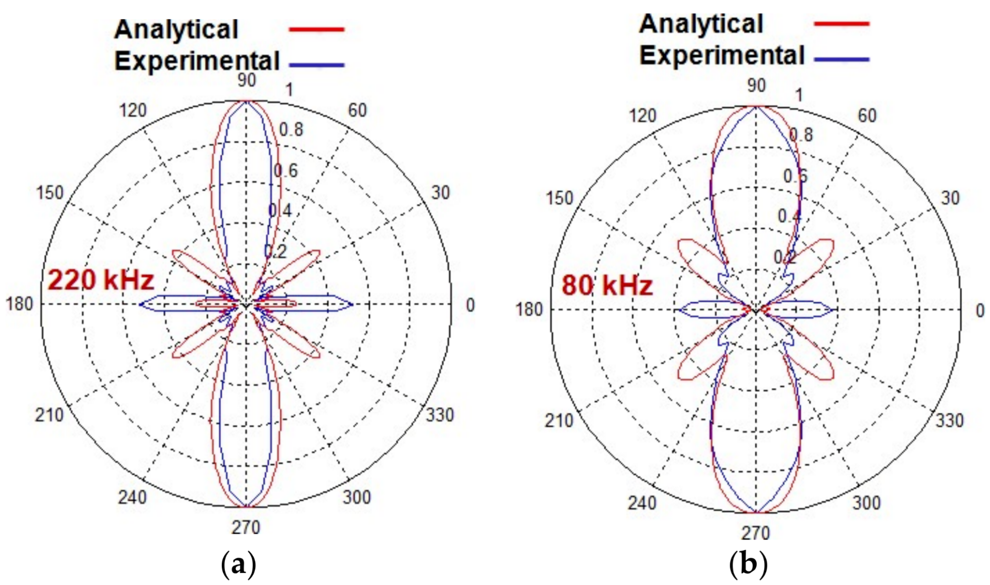

Figure 7: Ultrasonic beam width comparison at different frequencies [10]

For the development of a 3D ultrasonic sensor, it is required to have a wider beam and long-range to find the position of the object in the space so the sensor will operate at the frequency of 40kHz. Figure 7 shows that the beam directivity of the ultrasonic is narrow when operating at high frequency and also the attenuation is much higher as compared to the low-frequency ultrasonic waves. The higher frequency ultrasonic sensors are recommended for the narrow places e.g. fill level sensor inside tanks. While the lower frequency sensors generally ranging from 40kHz - 58KhZ are widely used for outdoor applications due to their wide beam directly.

2 LITERATURE OVERVIEW

In 2010, the technique of passive sound source localization was implemented using an array of MIC and applying the delay estimation algorithm. There is a comparison between two setups having 3 MIC system in the sitting position of the equilateral triangle and 4 MIC system in square sitting arrangement. The outcome of the experiment shows that the 4 MIC system has a higher positioning accuracy than the 3 MIC to some extent. The 3 MIC positioning system has been able to satisfy general requirements [11]. The 3 MIC and 4 MIC all have very good direction sensitivity. In this paper, quasi-L1-related technology accelerates the speed of delay estimation, improved anti-outliers interference.

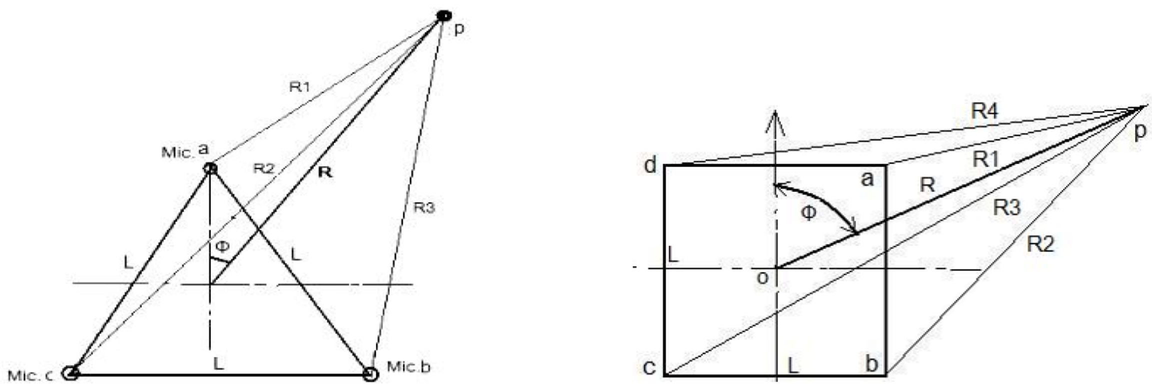

Figure 8: 3-MIC sitting and 4-MIC sitting arrangement setup [11]

In [12], a way to use a distributed array of microphones to reduce the approximation error in the position of sound source is presented. This paper introduced the concept of multisensor object localization using different sensor observability in order to account for different levels of access to each spatial position. The experimental results in the paper shows that using the two element arrays system yields more accurate results and the error reduces from 0.9m to 0.08m at a signal to noise ratio of 0db.

In [13] a robot navigation system was implemented using passive sound localization. A distributed array of 24 microphones was used to localize the position of the robot. With this system, the robot was programmed to conduct a guided laboratory tour. To travel between stations on the tour, the robot made a series of steps, and at each step sound localization was used to find the new location, taking advantage of the fact that the robot was speaking as part of the tour. The updated locations were used to calculate any needed course corrections. Experimental results showed that the error in the position of the robot at each station was approximately 7 cm close to the array. The robot detected obstacles with a set of touch sensors, and with the use of sound localization, the positions of the obstacles were saved in memory to be avoided for future tours.

This literature [14] provides a model-based sound localization system and its application to robot navigation. The setup consists of three-microphone arrangement in the triangular sitting arrangement, DSP, PC, and robot base. In the experiment, it is determined that the incoming sound has some portion of corrupted data due to the sound echoes. The only clean portions are echo-free onsets that have a sharp rising slope following a certain silent or low-level sound period. The research has shown that human ears have an inbuilt ability to filter out the echoes and inhibit the localization in echo portions which is called the precedence effect.

The paper [15] describes methods to improve noise filtering techniques in Sound Source Localization and Sound Source Tracking by increasing the number of microphones used while reducing computational load. In this paper, a method is introduced called SRP-PHAT. The SRP-PHAT is the scan of the 3D space over a wide grid then narrowing down the scan to a specific area. The theory is similar to the technique that is used by bats. Bats first used low-frequency waves to scan wider regions of the environment then narrowing down their search to a specific region by using high frequency focused waves. A special technique is used in which the data from the specific number of microphones is processed thus ignoring non-significant pairs of microphones. That technique allows to reduce the number of directions to scan thus decreasing the computational power. That technique is called microphone directivity. Paper also introduces the modified Kalman filter method for Sound Source Tracking that is capable of tracking in 3D the directions of sound sources. The performance of SRP-PHAT and modified Kalman filter shows the same results and performance while using 4 and 30 times less computing resources respectively.

The implementation of a spherical microphone array and a source localization technique based on the generalized cross correlation of the microphone signals have been considered [16]. The simplest microphone configuration is the regular one where the microphones have the same azimuth and elevation spacing. In this case, the noise source map obtained has many side lobes which may introduce artifacts and prevent the localization of sources with weaker amplitude.

In this work, the microphone positions of a spherical array were optimized in order to improve the source localization. A cost function independent of the source and based on the symmetry of the aperture angle map was defined. The Nonlinear Optimization by Mesh Adaptive Direct Search solver was used to maximize it. The optimized microphone array geometry allows for decreasing the side lobes amplitudes without increasing the main lobe surface. The noise source maps obtained numerically are always more accurate than the noise source map obtained with the regular geometry.

Another study has presented a digital signal processing method combining TOF and PS technology to achieve high accuracy in a rangefinder system. It mainly uses the self correlation of the received signal to identify the onset location and then estimate the RTOF. To drop the influence of noise, this paper proposes a de-noising method based on wavelet decomposition that can determine the optimal decomposition level and the appropriate denoising threshold automatically[17]. It is shown that the SNR of the noisy received signal can be increased approximately 6 dB through MATLAB simulation. Then a phase shift detection method within one ultrasonic period is proposed. Moreover, in order to obtain the total phase delay between the transmission and reception signals, an optimization algorithm to get the real integral multiple of the ultrasonic period through RTOF and PS is put forward in the paper. Finally, the distance can be calculated through the phase delay and the sound velocity which is compensated by the ambient temperature. Through the verification experiments in three aspects: wavelet de-noising, PS detection and temperature field compensation, the ranging error is less than 0.5 mm for the distance up to 5000 mm.

A study published in 2016, explained the feasibility of a long range measurement system using ultrasonic sensors with the UTA in air is experimentally investigated. The UTA consisting of 144 transmitter elements exhibit 40 times higher ultrasonic sound pressure than the SUT. The system was operated with the pulse widths 2, 1, and 0.5 ms. The measurable range was probed using 2 ms pulse width with suitable pulse repetition period so that the transmitted pulse was completely received back by the receiver after reflection from an object; but before transmitting the next ultrasonic pulse [18]. The measurable range is significantly improved using the UTA with the URA. These results are useful to expand further applications of ultrasonic range sensors. Using the novel UTA system it has successfully demonstrated the range imaging of an object placed at different positions.

Most of the ultrasonic sensors operate at the frequency of 40kHz having a typical directivity of more than 20°. The objective of this research was to develop an ultrasonic transducer with a directivity of less than 5° while maintaining the same working distance as existing commercial products. The paper [19], describes the design, fabrication, and test of a new type of highly directional ultrasonic transducer. A highly directional ultrasonic beam with a HPBW of ±1.3° was generated using a parametric acoustic array with this transducer. The sound pressure level of the primary wave was more than 120 dB and that of the difference frequency wave was 83 dB. This is much better than other conventional ultrasonic transducers.

A model combining the continuum model and the compatibility condition was appropriate for designing the ultrasonic range sensor based on the difference frequency wave resulting from the nonlinear effect between two high-frequency components. This model can be applied to various types of Langevin transducers consisting of a horn and piezoelectric disk actuators. The analytical model provided an approximate solution for the optimal length and decreased the computation time of the finite element model by trial and error. It is confirmed that the resonant frequency of the model was similar to that of the prototype transducer. Notwithstanding its similarity, there is a difference of less than 4% between the theoretical result, FEM, and experimental one.

2.1 Key Information

From the detailed literature review, these are following point thats have been concluded:

- As presented in the above studies, the use of a single array of microphones is common in the localization of sound sources. The results of the experimental setup having less number of microphones but multiple number arrays of microphone does not only provide better performance but it also has the advantage of requiring less computational resources and less bandwidth [12][13].

- The 4 microphone sitting arrangement provides good direction sensitivity as compared to the 3 microphone setup. The quasi-Li related technology optimizes the process of speed delay estimation of the incoming sound waves [11].

- A transmitter array having *N* number of transmitters does not produce *N* times the sound pressure. The overall performance of the array depends upon the position, distance, and angle of the traducer [18].

- In the sound localization technique, the signal processing and the array geometry of the receiving microphones have to be chosen first. The simplest microphone configuration is the regular one where the microphones have the same azimuth and elevation spacing. It is recommended to use a spherical microphone array. It allows for decreasing the side lobe amplitudes without increasing the main lobe surface [16].

- In a nutshell the ultrasonic technology, the distance estimation is more accurate as compared to elevation/azimuth because the dynamic range of time difference caused by elevation/azimuth is very small. Environmental conditions such as fog, rain, and smoke tend to have a negative effect on the propagation of sound waves in the air. High power sound transmitter is required for long-distance object detection. The experimental setup having an array of sound transmitters gives better performance as compared to a single transmitter [12,13,14,15,16,17,18].

2.2 Literature Analysis

The test results of the sensor shows that the effect of temperature on the readings is significant as mentioned in the literature [17]. The speed of the sound affected by ambient temperature can add an error up to ±1cm. That error can be compensated by using a temperature probe in the future and by adjusting the speed accordingly.

The three sitting arrangements of the receivers were tested in the initial phase of the development of the prototype sensor but the results were inconsistent and the error in the angle calculation was ±10°. So the four sitting arrangement is adopted for the final prototype [11].

The field of view of the sensor depends upon the beamwidth of the transmitter. Transmitter having wider beamwidth covers a larger region which is recommended for 3D application but the operating frequency of ultrasonic is inversely proportional to the beamwidth of the transmitter and directly proportional to the resolution of receivers. So the selection of the operation frequency depends upon the balance of measurement resolution and field of view of the sensor.

3 METHODOLOGY

In this thesis, the active acoustic location principle is implemented on ultrasonic waves operating at the frequency of 40kHz to find the position of the object in 3D space. The operating principle of the sensor will be the same as discussed above, the 8 bursts of ultrasonic waves are transmitted and then by calculating the time arrival difference of the incoming waves deflected by nearest object is recorded by four receivers as the smallest detectable shift in the angular location of the sound deflection source.

The method uses a relatively low propagation speed of sound in the air. So there will be a measurable time difference between the two microphones based on the position of the object and the distance between receivers. The four receivers are placed in the rectangular sitting arrangement of 8 centimeters apart from each other. The measurable time difference will be approximately 50 microseconds between two receivers. To record such a short interval of time it is required to have a high speed timer operating at the base 100 nanoseconds.

The incident angle of the object from the receivers induces the time difference of the arrival of ultrasonic echo to the receivers. Figure 9 shows the geometry used for calculating sound direction based on interaural delay. Let's assume the object is placed at the left side of the two recievers. Due to the incident angle, the object is closer to Receiver 1 than Reciever 2. So the incoming ultrasonic wave represented by the red line shown in Figure 9 takes more time to reach Receiver 2.

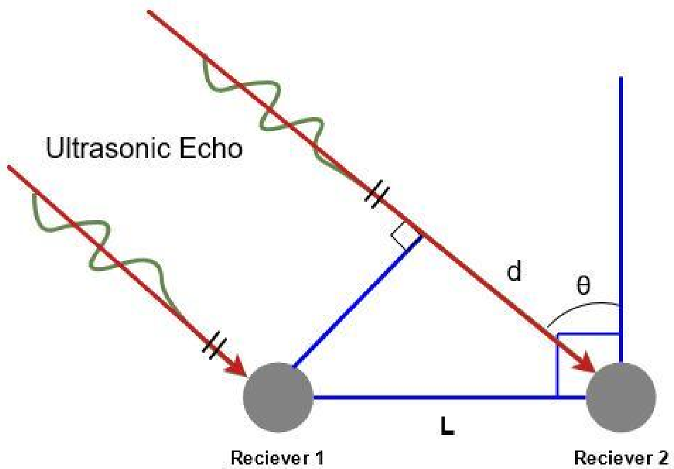

Figure 9: Interaural delay geometry

Calculating target bearing angle θ by using sine law on trigonometric ratios of Figure 9:

$$\theta = \operatorname{asin}\left(\frac{d}{L}\right) \tag{3.1}$$

$$\theta = \operatorname{asin}\left(\frac{D2 - D1}{L}\right) \tag{3.2}$$

$$\theta = \operatorname{asin}\left(\frac{(TOF2 - TOF1) * 0.0000000001 * Vsound}{L}\right) \tag{3.3}$$

where

d = time difference of the arrival of incoming wave between two receivers

L = distance between two receivers

D_1 = distance between object and receiver 1

D_2 = distance between object and receiver 2

TOF_1 = time of flight in Nanoseconds between object and receiver 1

TOF_2 = time of flight in Nanoseconds between object and receiver 2

V_{sound} = Speed of sound

Unlike passive sound source localization, the maximum angle bearing of the target is limited to the maximum beamwidth of the ultrasonic transducer. Ultrasonic transducer MSO-A1640H12TR is used in the setup. MSO-A1640H12TR [20] is a transceiver that can provide functionality as both transmitter and receiver. Figure 10 shows the beam directivity of MSO-A1640H12TR. The transducer has a beam directivity ranging from -40deg to +40deg. So the sensor cannot locate the target beyond these angles.

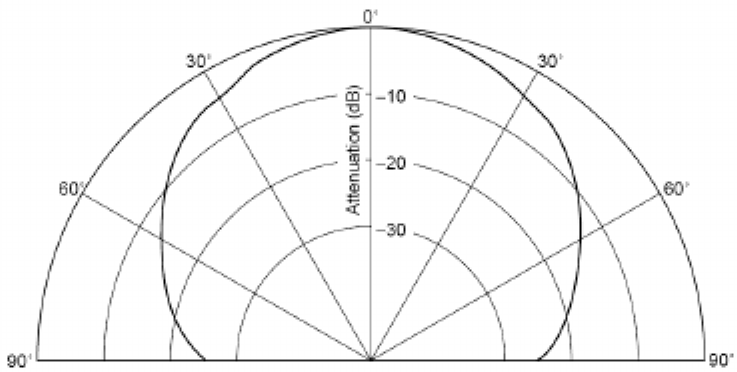

Figure 10: Transducer beam directivity [20]

Table 1 shows the maximum and minimum range of azimuth angle, elevation angle, and distance of the developed ultrasonic sensor solution. The range of the azimuth angle and elevation depends upon the beamwidth of the ultrasonic transmitter. The transmitter used in the solution has an overall beam directivity of 80°. The dead zone of the sensor is 5cm. The sensor is unable to calculate the distance and position of the object below the distance of 5cm. As the distance, less than 5cm is so small that the time difference of the incoming echo between all receivers is nearly zero. Therefore it is not possible to find the position of the object if it is less than 5cm from the sensor.

Table 1: Parameters of the developed ultrasonic sensor

	Minimum range	**Maximum range**
Azimuth angle	-40°	+40°
Elevation angle	-40°	+40°
Distance	5cm	200cm

4 SIMULATION

The wave2000 [21] simulation software is used to simulate the propagation and reflection of the ultrasonic wave. The software simulates the ultrasonic waveforms under a variety of spatial and temporal acoustic interrogations and generates solutions to practically any 2D ultrasonic elastic wave propagation problem. Using that software we will perform the analysis on the interaural intensity difference and the interaural time difference under the original conditions and we will validate our algorithm to calculate the bearing angle of the object.

4.1 Environment Setup

In this simulation, two ultrasonic receivers are placed at a distance of 4cm apart from each other. After that two objects are added at different positions, afterward a 40kHz ultrasonic burst is generated to record the time of arrival of the echo reflections from the two objects, recorded simultaneously by both receivers. Then all the echo data which is recorded by the receivers is extracted and plots are generated to find out the time difference of arrival of ultrasonic echo between two receivers. From that, the azimuth angle of the objects with respect to the transmitter can be calculated and find out the percentage error from actual values. Figure 11 shows the flow chart of the simulation.

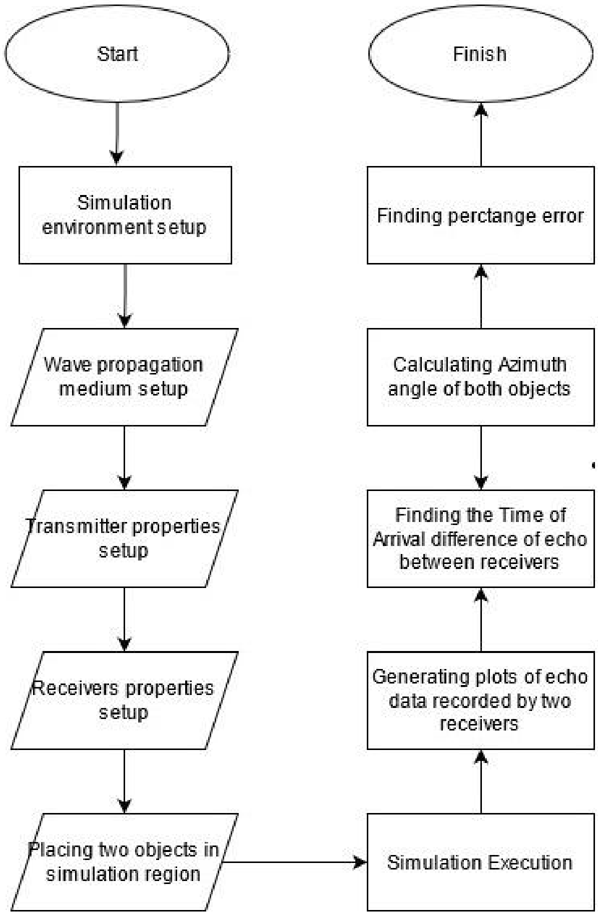

Figure 11: Simulation flowchart [21]

The first step in the software simulation is to set up the environment by defining the area of the simulation, which is called geometry dimension. As the Wave2000 [21] software comes with high-end features to provide reliable solutions concurrently there are few constraints associated with it as well. One of the constraints is that Wave2000 is designed to perform simulation in a small environment. As our target is to perform the timing analysis on the echo so the dimension of the simulation is not a factor to consider. So, the simulation is performed in a small dimension of 80mm x 80mm.

The second step is to add the property of the material, as we're using an airborne ultrasonic sensor so we have to specify the Air as an ultrasonic waves propagation medium. By using the Materials Properties window shown in Figure 8, we have added the type and parameters of a propagation material as Dry Air at 20 degrees Celsius in a simulation. Figure 12 shows all the parameters of ultrasonic waves propagation medium in simulation.

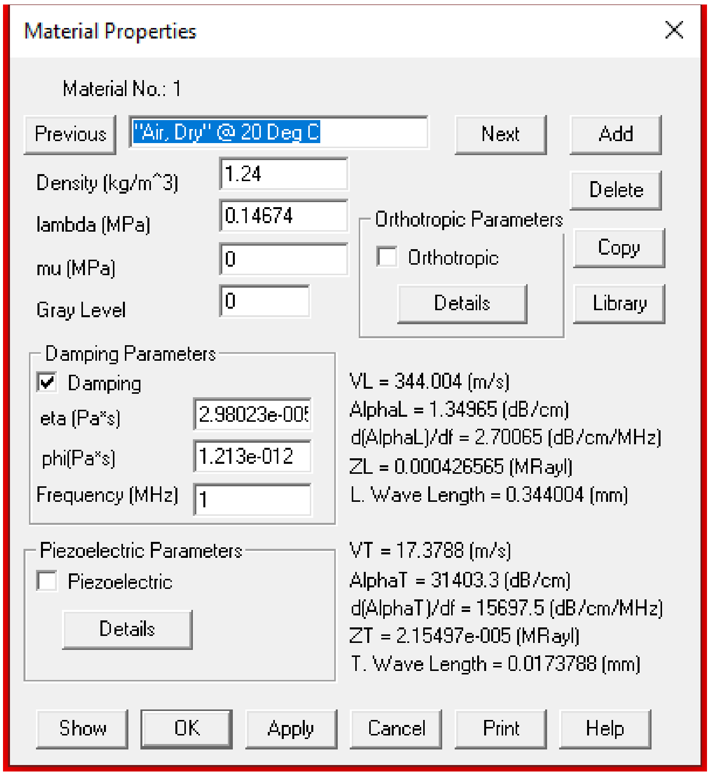

Figure 12: Properties of ultrasonic waves propagation medium in simulation [21]

The third step is to add the transmitter in the simulation, the 5mm long transducer is created at the center of the bottom edge. The transducer is set to generate a sine wave with a center frequency of 40kHz, having an amplitude of 1 for the duration of 5 microseconds (us). Figure 13 shows the parameters of the ultrasonic transmitter used in the simulation.

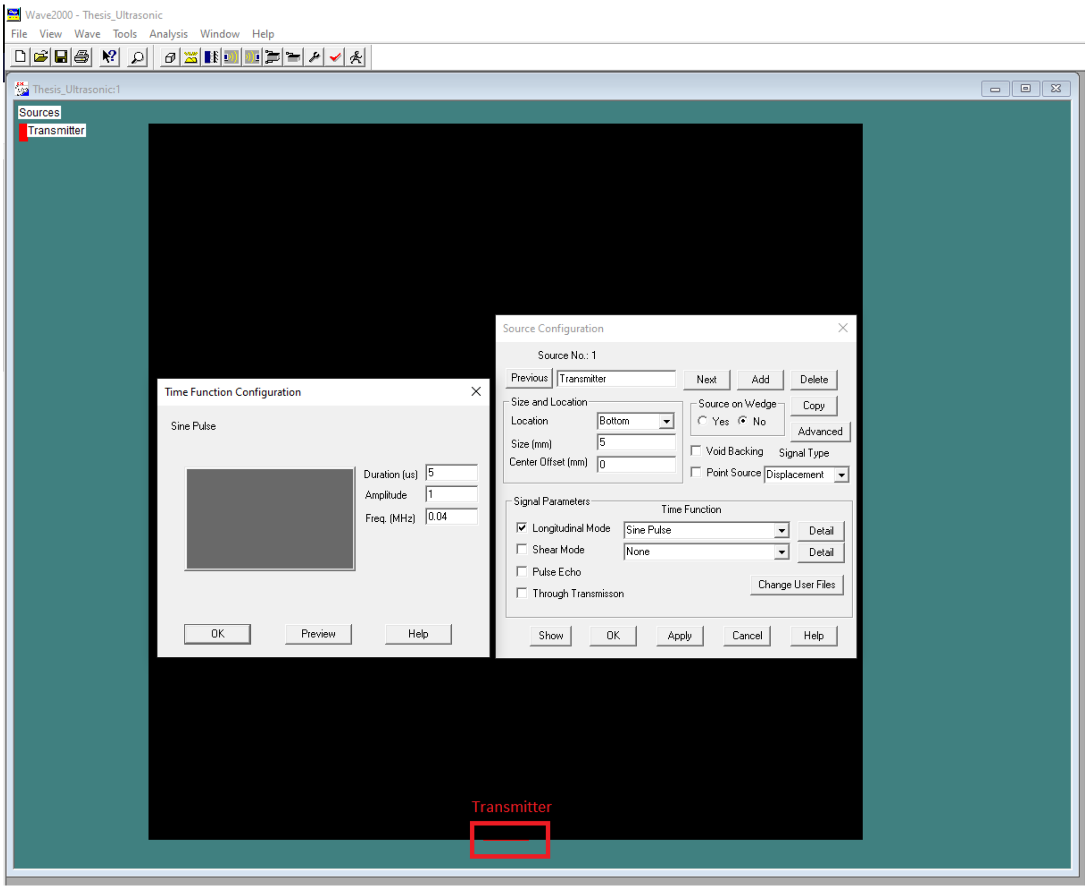

Figure 13: Properties of the ultrasonic transmitter used in the simulation [21]

After that, two receivers are added to the simulation to quantitatively measure the reflected/refracted ultrasound signals. The receivers are placed at a distance of 4cm apart from each other at the bottom edge. The receiver will not process any data for 130 microseconds, and this time can be classified as a blank zone. This feature will remove all the false echoes generated during the burst stage of the transmitter.

An infinite boundary condition is used in this simulation, which is labeled "Infinite BC" on the Boundary Condition tab. The program has four checkboxes that allow you to simulate an infinite boundary condition on any number of sides of the object. On any given side, an infinite boundary condition attempts to make the boundary appear as an infinite medium matched to the material just within the object's boundary. Figure 14 shows the properties and position of the ultrasonic receivers in the simulation.

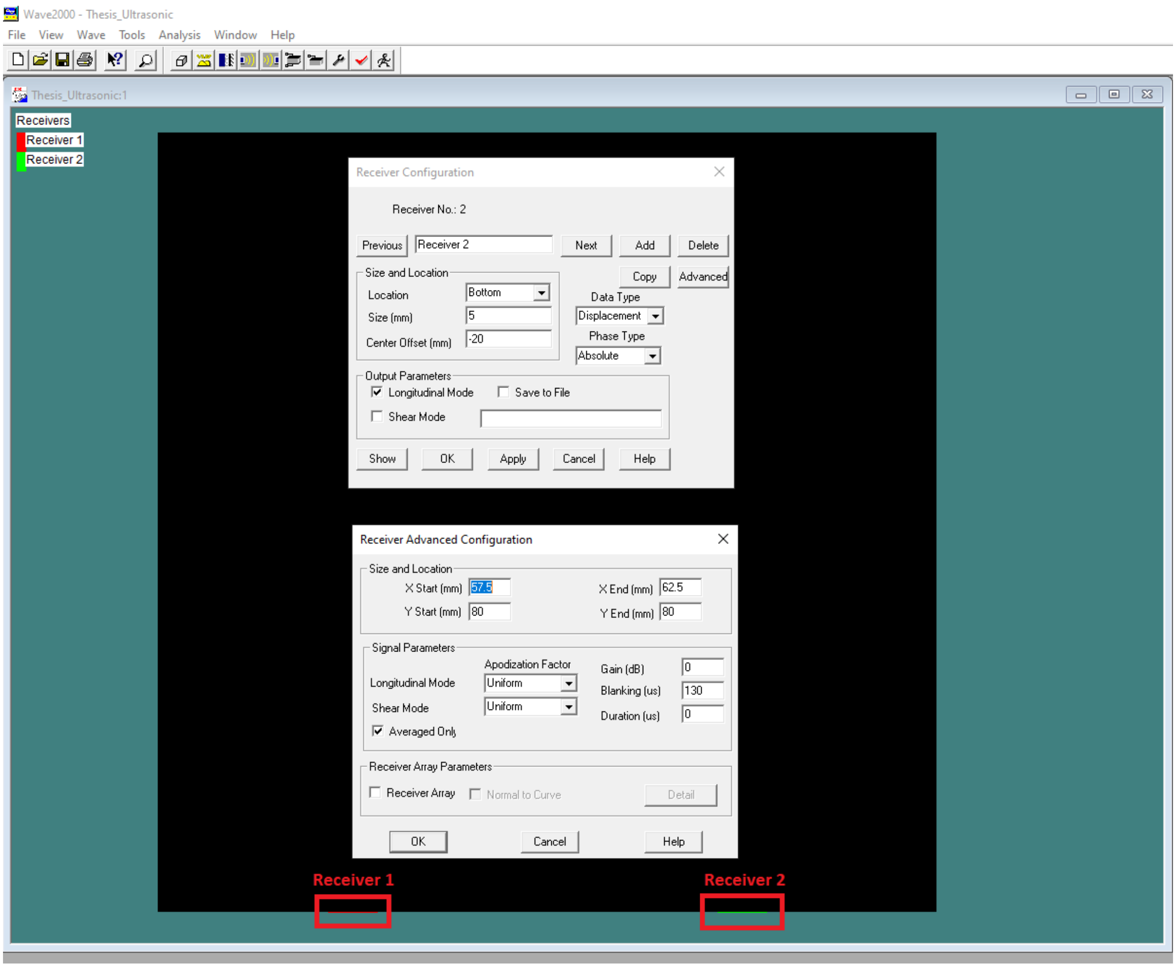

Figure 14: Orientation of the ultrasonic receivers used in the simulation [21]

At the end, we have placed two objects in the simulation, Object1 and Object2. Object1 and Object2 are placed at a distance of 55mm and 70mm respectively from the transmitter. To observe the interaural intensity difference and interaural time difference, we have placed Object1 at the left side of the transmitter, and Object2 is placed at the right side.

Now the software is ready to perform the simulation and after the simulation we will extract the data from both of the receivers to analyze the interaural time difference and interaural intensity difference by generating plots. The simulation in progress is shown in Figure 15 below.

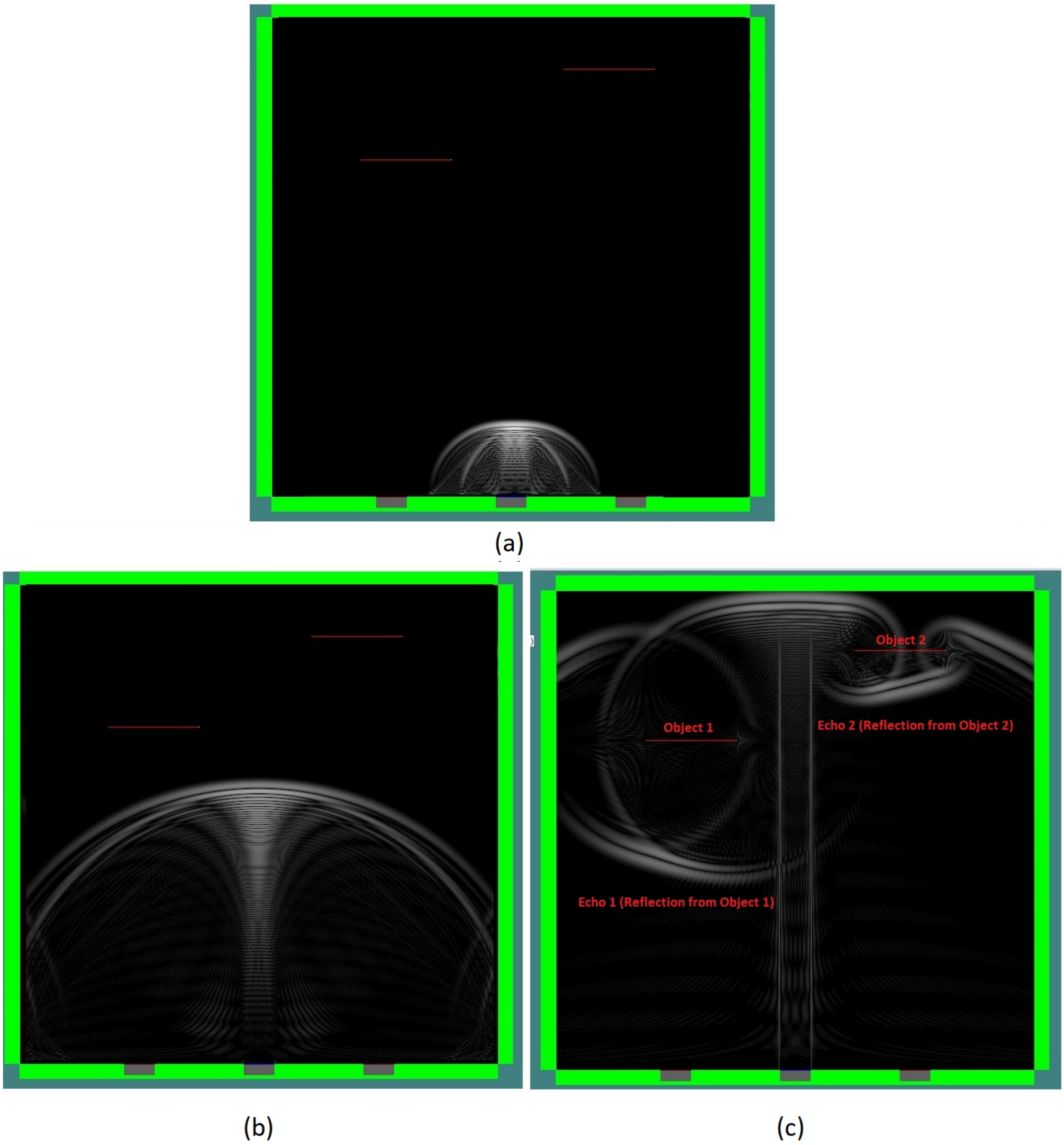

Figure 15: Ultrasonic simulation in wave2000 software (a) initial bursts, (b) propagation in air and (c) echos [21]

4.2 Results

To quantitatively measure the reflected signals, simulation echo plot in Figure 16 shows the time-domain signals received from two receivers. The plot shows the time delay and intensity difference generated by the reflected echoes from Object1 and Object2. The time difference, shape, and signal strength between two receivers are affected by the distance and azimuth angle of the objects in the 2D space.

As Object1 is closer to Receiver1 so the echo reflected from Object1 is reached in a shorter time to Reciever1 as compared to Receiver2, this time difference of the arrival

of echo between the two receivers is called interaural time difference. Due to the shorter distance from Reciever1, the intensity of the echo recorded by Reciever1 is higher as compared to Reciever2, which is called interaural intensity difference.
The same phenomena have occurred to Object2. As the Object2 is closer to Receiver2, the intensity of the echo recorded by Receiver2 is higher as compared to Receiver1 and the time arrival of echo is shorter at Receiver2.

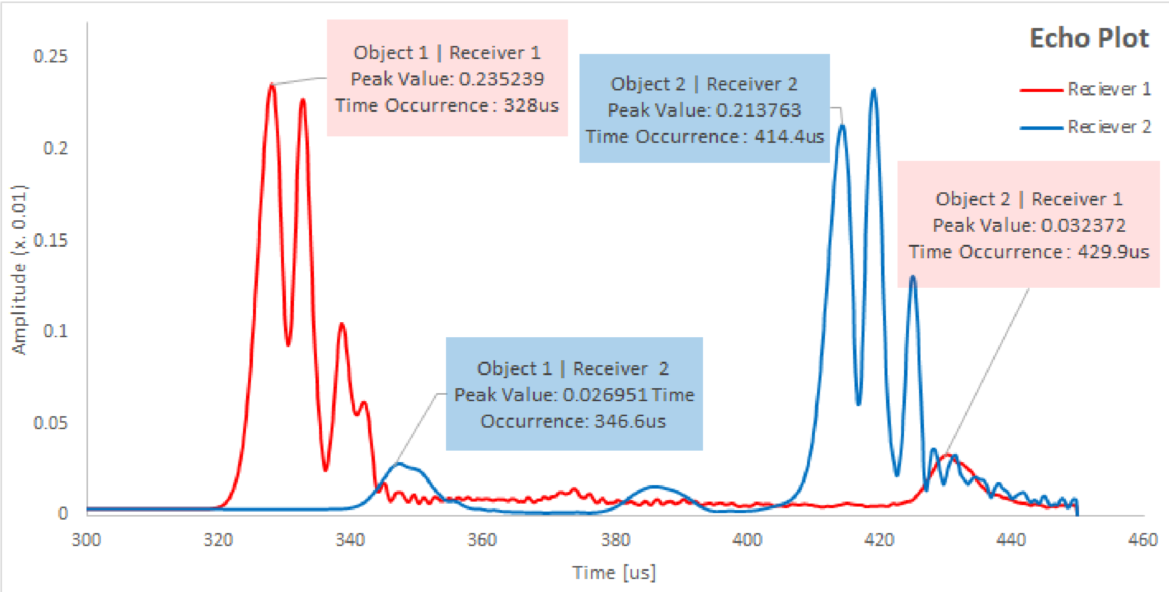

Figure 16: Echo plot of the data received from the simulation [21]

Using equation 3.3, azimuth angle can be calculated as:

$$\theta = \operatorname{asin}\left(\frac{(TOF2 - TOF1) * 0.0000000001 * Vsound}{L}\right)$$

(3.3)

First, the azimuth angle of the Object 1 is calculated. For that data is extracted from the echo plot. In the echo plot, the time of arrival of echo from Object 1 is 328.0us to Receiver 1 and 346.6us to Receiver 2. So the azimuth of Object 1 with respect to the transmitter can be calculated as:

TOF_1 = 328000ns

TOF_2 = 346600ns

L (length between two receivers in simulation) = 4cm

V_{sound} = 34315cm/s at 20 degrees Celsius

putting these values in equation 3.3

$$\theta = \text{asin}\left(\frac{(346600 - 328000) * 0.0000000001 * 34315}{4}\right)$$

$$\theta = 9.18$$

So Object 1 is at 9.18° left to the transmitter. Distance of the object from transmitter can be calculated as:

(distance = speed x time)

$$D = \frac{TOF * Vsound}{2} \tag{4.1}$$

where

D = distance of object from transmitter

TOF = Time of flight of object

V_{sound} = 343.15m/s = 0.034315cm/us at 20 degrees Celsius

Distance of the Object 1 can be calculated by using equation 4.1.

$$D_1 = \left(\frac{328 * 0.034315}{2}\right)$$

D_1 = 5.63cm

Similarly for the Object2

$$\theta = \text{asin}\left(\frac{(429900 - 414400) * 0.0000000001 * 34315}{4}\right)$$

$$\theta = 7.64$$

So Object 2 is at 7.64° right to the transmitter. Distance of the Object 2 can be calculated by using equation 4.1.

$$D_2 = \left(\frac{414.4 * 0.034315}{2}\right)$$

D_2 = 7.11cm

Percentage error between the actual values and measured values is calculated in Table 2. The reading of the distance has less error as compared to the azimuth angle because the dynamic range of time difference caused by elevation/azimuth is very small. The maximum error is 9.10% which is in an acceptable range and commonly induced in ultrasonic sensors due to the low resolution of ultrasonic receivers typically 5mm.

Tabel 2: Simulation results

	Measured value	**Actual value**	**% Error**
Θ_1	9.18	10.1	9.10
Θ_2	7.64	8.13	6.02
D_1	56.3	55	2.3
D_2	71.1	70	1.57

The detected position of two objects in the simulation is plotted against the actual position of the objects in the simulation shown in Figure 17. The results of implemented methodology based on time difference analysis have little error. That can be further reduced by adding more receivers or operating ultrasonic at higher frequency, as the high-frequency ultrasonic receiver has a high resolution typically 1mm.

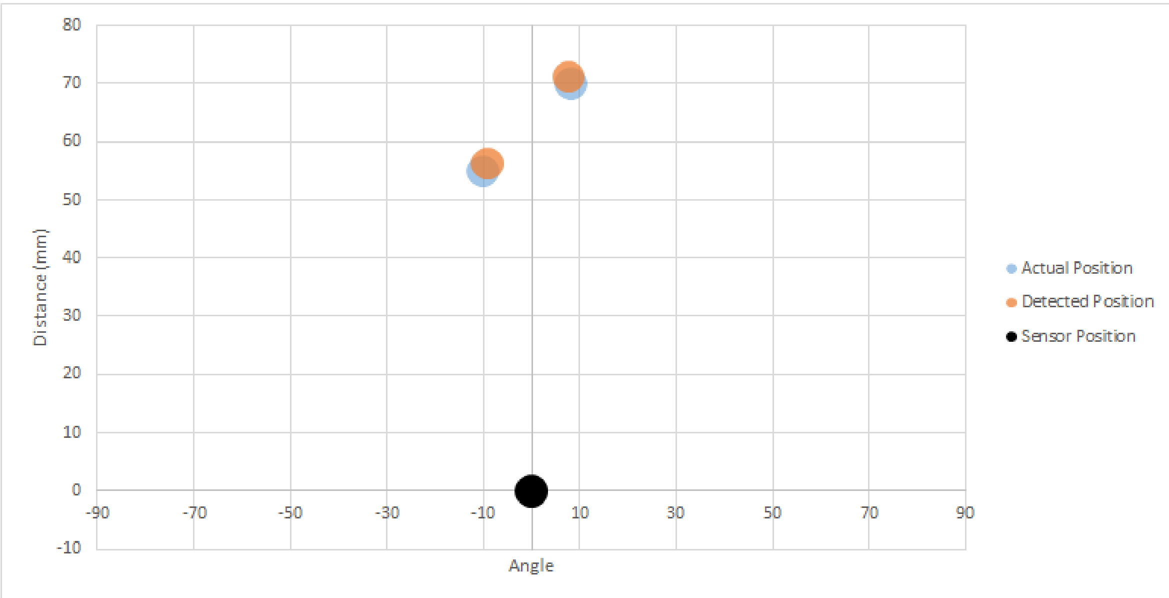

Figure 17: 2D plot of actual position and detected position of object [21]

5 DEVELOPMENT OF A SYSTEM

Ultrasonic sensors, also known as sonar sensors, work by sending a high-frequency sound wave at a target and then analyzing the reflected echo. These ultrasonic sensors can convert alternating current into ultrasound and the reverse;then ultrasound into alternating current. Some devices use different sensors for transmitting and receiving data, while others combine both functions into a single sensor known as mono-static configuration. Our target is to develop a system that is based on one transmitter and four recievers.

A relatively high peak to peak voltage is needed to transmit ultrasonic pulses. The peak to peak voltage value of the transducer depends upon the type of transmitter. The intensity of generated sound pressure waves depend upon the driven voltage of transducers. It is necessary to operate an ultrasonic transmitter on its rated voltage to generate. There are two methods to generate high voltage to drive an ultrasonic transmitter, transformer driven solution and direct driven solution:

- For long-range measurements, ultrasonic transformers are widely used to excite closed-top transducers rated at +100Vpp. Since most high-frequency transducers are closed top and have a maximum drive voltage rating of hundreds of Vpp, they are often used in conjunction with transformers. In Figure 18 the typical transformer-driven method is shown. Using this method, the small input voltage VPWR is enough to operate high voltage ultrasonic transmitters due to the voltage step-up operation of the transformer by N factor depending upon its coil turn ratio. Figure 18 shows a schematic example of the Transformer-Driven method. The complementary PWM is required to be applied on the pins OUTA and OUTB to drive the transmitter.

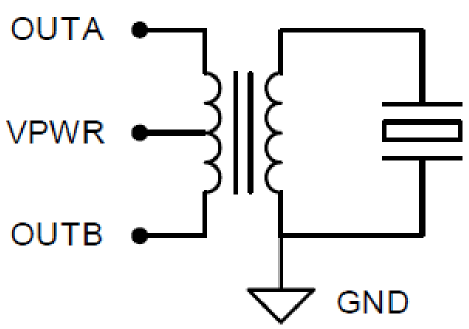

Figure 18: Transformer-Driven method [22]

Since the range of the proposed sensor is only 2m, a transformer might not be needed, allowing us to use a direct driven solution.

- In the direct driven solution, there is not any third component involved for the step-up conversion of voltage just like a transformer-driven solution. Instead of that, the transducer is driven by providing high voltage directly VPWR, generated by charge pumps or boost supply. The direct-driven solution has its drawback for mono-static ultrasonic solutions. As the transformer provides the counter inductance which helps to reduce the transducer's ring decay time, in the absence of that inductance there is not any mechanism for damping and the transducer is at the mercy of its resonance behavior therefore the dead zone is high in monostatic direct driven solutions. The dead zone is the time required for the transducer to become quiet after the transmission stage to allow the reception of a return echo. Figure 19 shows a schematic example of the Direct-Driven method.

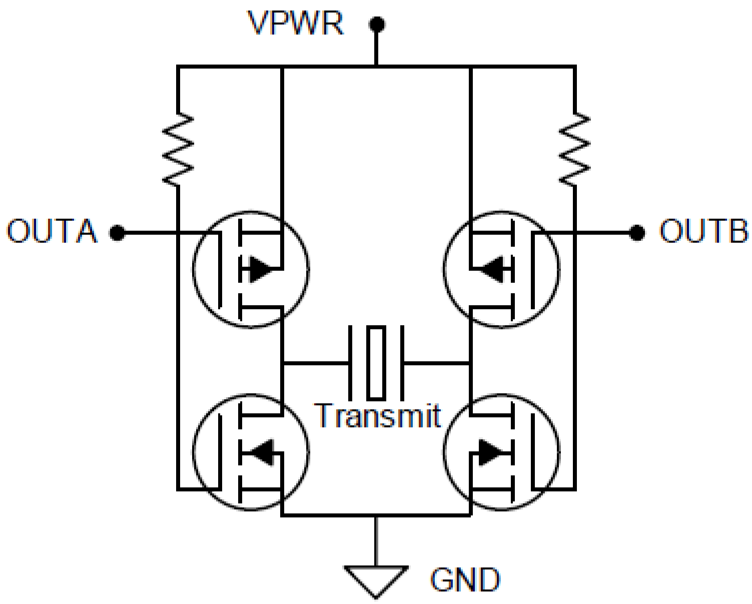

Figure 19: Direct-Driven method [22]

In the receiving stage, the echo reflected by the object produces a very small voltage at the receiver. The signal produced by the receiver needs to be filtered and amplified before sending it to the ADC. In the ultrasonic receiver, it is very important to implement a bandpass filter. Without the bandpass filter, the amplified output of the operational amplifier picks up a lot of noise (around 25Khz). The bandpass filter is calibrated to a center frequency matching the ultrasonic system's transmit carrier pulses (40kHz nominal). Figure 20 shows the bandpass filter calibrated at the center frequency of 40kHz ± 0.5kHz. This means that the response of the system is maximum at 40Khz with the tolerence of ± 0.5kHz and all other frequencies will be filtered out. The bandpass filter will remove all the noise from echo signals received by the ultrasonic receiver.

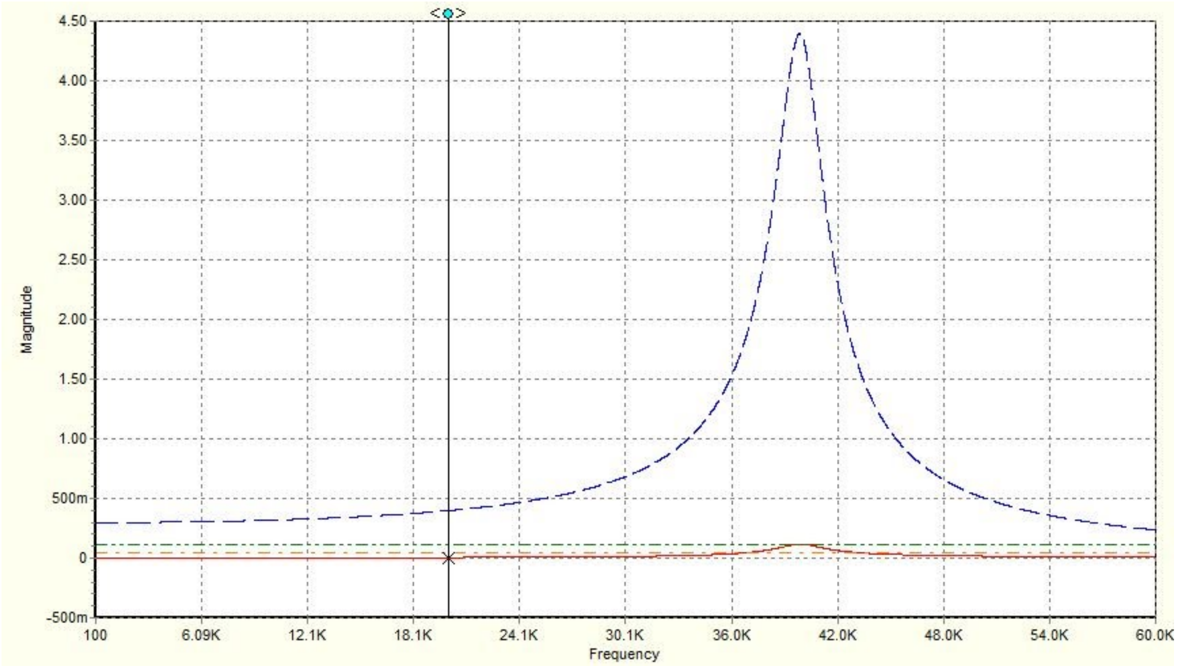

Figure 20: Bandpass filter calibrated at the center frequency of 40kHz [23]

5.1 Hardware Design

The circuit has five transducers one for the emission and four for the reception. The direct-driven topology is implemented for the transmission of ultrasonic waves. A relatively high voltage is needed to transmit ultrasonic pulses. To reduce the complexity of the circuit a MAX232 [24] is cleverly used to convert 5V to ±10V. A MAX232 is an RS-232 Driver/Receiver with an integrated charge pump and voltage inverter for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly where ±12V is unavailable. The functional block diagram of MAX232 is shown in Figure 21. Block diagram of MAX232 shows that it has an inbuilt charge pump, charge inverter, and buffers in a single package. It can be used to drive ultrasonic transmitters directly without any need for other circuitry.

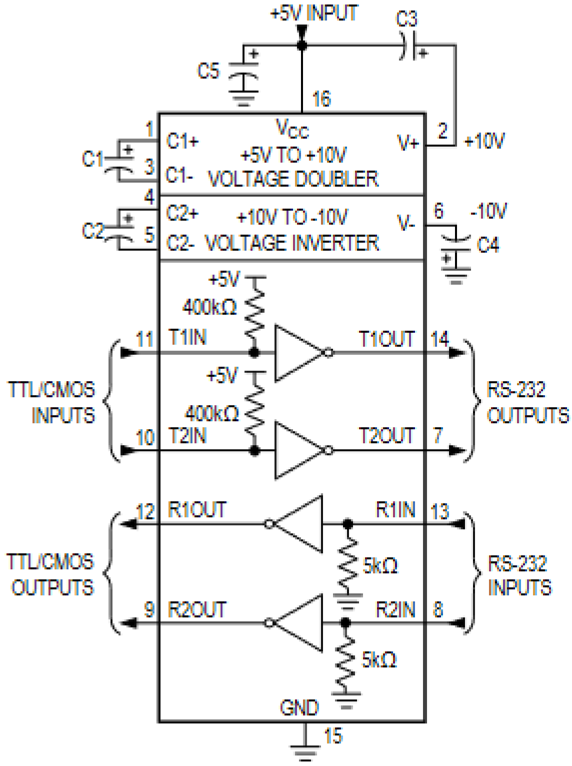

Figure 21: Functional block diagram of MAX232 [24]

The transducer is connected between two outputs of MAX232, powered by an internal charge pump and voltage inverter generating 20Vp-p. The input of the MAX232 is connected to the controller which generates the 8 pulses of complementary PWM at the frequency of 40KHz. The transistor is used as a switch to control the power of MAX232. Power is only applied to MAX232 through the transistor sometime before and during pulse emission because the internal switching charge pump is noisy. The MAX232 power is turned off when the circuit switches to receive mode.

On the receiver side, an LM324 is used which has four OPAMPs. The first OPAMP is just a times 6 amplifier. The second is a 1st order bandpass filter which is followed by another times 8 amplifier. The amplified output of the third OPAMP goes to the ADC of the controller for peak detection. After that, The last OPAMP is used together with the transistor as a hysteresis comparator. The output of the comparator goes to the digital pin of the controller to capture the time arrival of the echo. For each receiver, one LM324 is used. The block diagram of the prototype developed for thesis is shown in Figure 22.

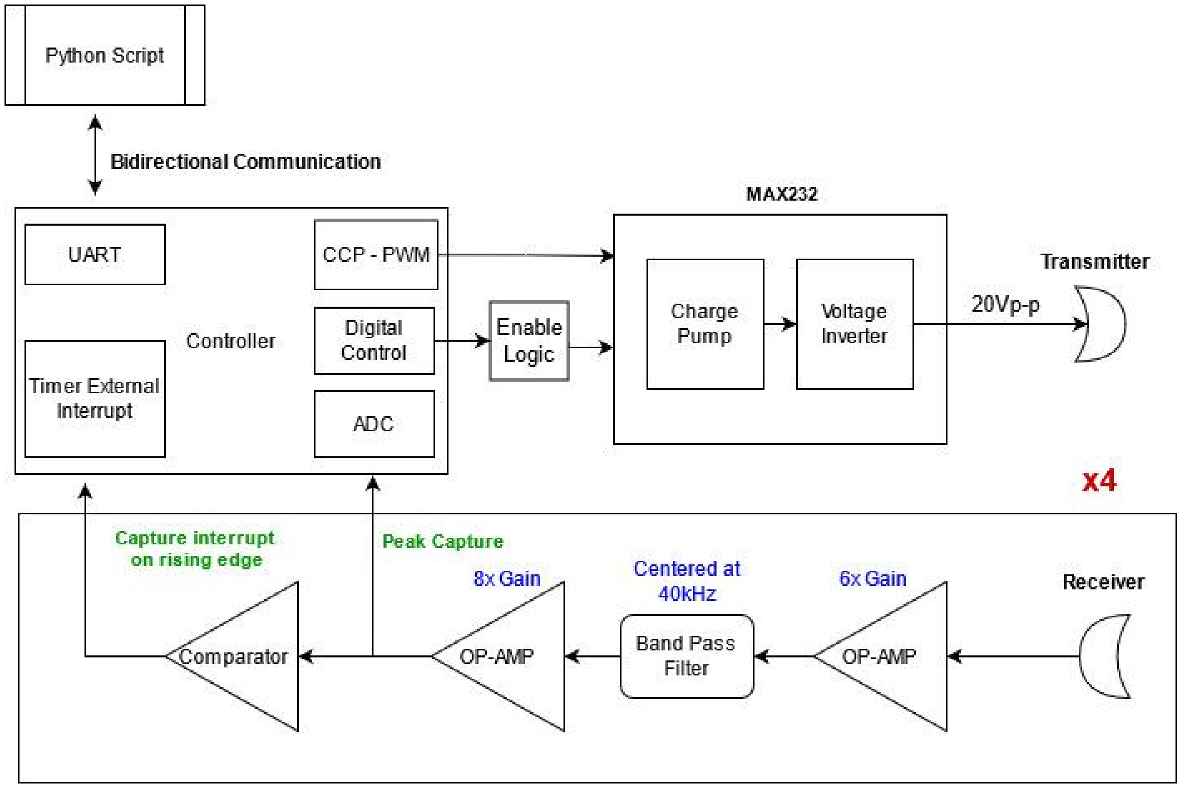

Figure 22: Sensor prototype block diagram

5.2 Software Architecture

The distance measured is a function of the time interval between the time the pulse is emitted and the time the echo is received. The sequence of taking measurements starts by turning on the supply of MAX232. After 248us, the charge-pump voltage ramps up to the +/-10V. The delay of 500us is implemented for the supply to get stable and then 8 pulses are granted to drive the transmitter. The pulse generation event is blocking and it is generated directly by toggling digital without the use of the timers. Once all the pulses are generated the supply of MAX232 is cut off by turning off the power transistor to reduce parasitic noises in the system. The output of the comparators of the four receivers are attached to the four digital pins of the controller and the analog output of OPAMP is attached to four analog pins. All four digital have external internal features to execute ISR when the pin goes from low to high on the reception of the echo pulse. After that, the timer is set to count for the time of 12ms with a base of 100ns. This means the timer has a resolution of 100ns. Meanwhile, the timer is running, whenever there is an interrupt on the digital pin the interrupt ISR will execute. In the ISR the current value of the timer is stored to a variable corresponding to the receiver based on a digital pin. At the same moment, the analog read operation is carried out to store the peak value of the receiver. When the timer is expired the digital and analog data from all receivers is to python script via UART interface. During the recording time of 12ms. The flowchart of the firmware architecture is shown in Figure 23.

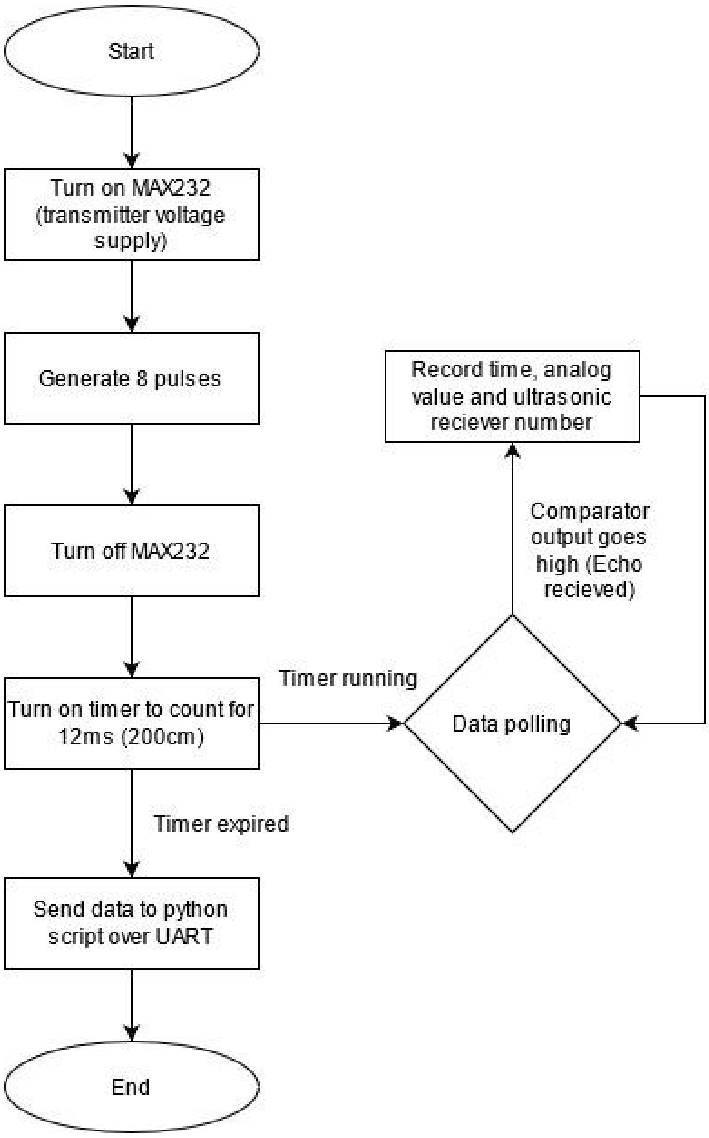

Figure 23: Software flowchart

The smart algorithm is developed in the python application to find out the azimuth and elevation angles from the raw values recorded by four receivers. There are a total of four receivers in the sensor but at a time, data from only three receivers are used to calculate the position of the object. Three receivers are selected based on the shortest time arrival of echo value.

When the python script receives data from the sensor, it finds out the lowest time of arrival value between the receivers. The receiver which has received the echo in the shortest time is marked as Receiver 1 in the script. That receiver will act as the pivot point of reicevers. The azimuth and elevation angle will be calculated by using the data from three receivers; Receiver 1, receiver next to pivot receiver in vertical (Receiver 2) and the receiver next to pivot receiver in horizontal direction (Receiver 3). The azimuth angle is calculated by using the time of flight value of echo from Receiver 1 and Receiver 2 and the elevation angle is calculated by using the time of flight value of echo from Receiver 1 and Receiver 3.

There are five possible positions of the object in front of the sensor, these conditions are calculated as five cases by the python algorithm. The objects positions in the space calculated by the python in addition to elevation and azimuth angle is given below:

- Case 1 TL: Object in the top left position of sensor
- Case 2 TR: Object in the top right position of sensor
- Case 3 BL: Object in the bottom left position of sensor
- Case 4 BR: Object in the bottom right position of sensor
- Case 5 FF: Object in the front position of sensor

Visual representation of cases is shown in Figure 24.

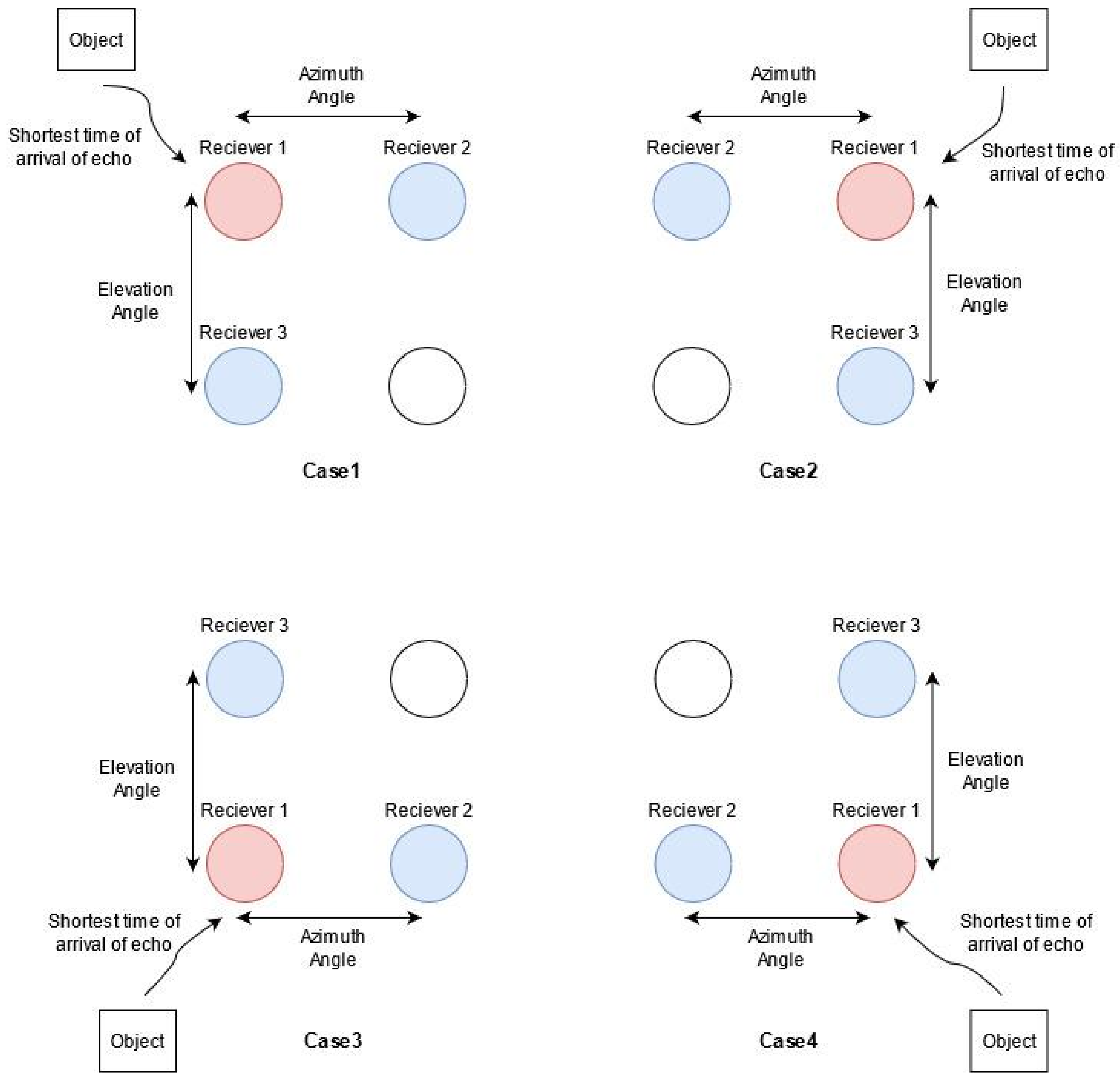

Figure 24: Position algorithm cases

The graphical user interface named Ultrasonic Localization Utility is developed in python to visualize the output of the sensor graphically. The Ultrasonic Localization Utility plots the position of the detected object in 3D space based on the calculated azimuth angle, elevation angle, and distance of the object from the sensor. The utility also has a section which shows that from which three receivers the data is used to calculate the position of the object. That feature pinpoints the receiver which is closest to the detected object. Figure 25 shows the working of Ultrasonic Localization Utility. The detail of features is:

1. Connection box. Used to establish connection with sensor. Shows the status of connection.
2. Button to send the command to the sensor to start executing the process and send back the readings.

3. The plot of the sensor in 3D dimensional space.
4. Sensor position in 3D space.
5. Detected object position in 3D space with respect to sensor.
6. Shows the receiver which is closest to the sensor and shows which three receivers are used to calculate angle.
7. Numerical display of calculated angles and distance.

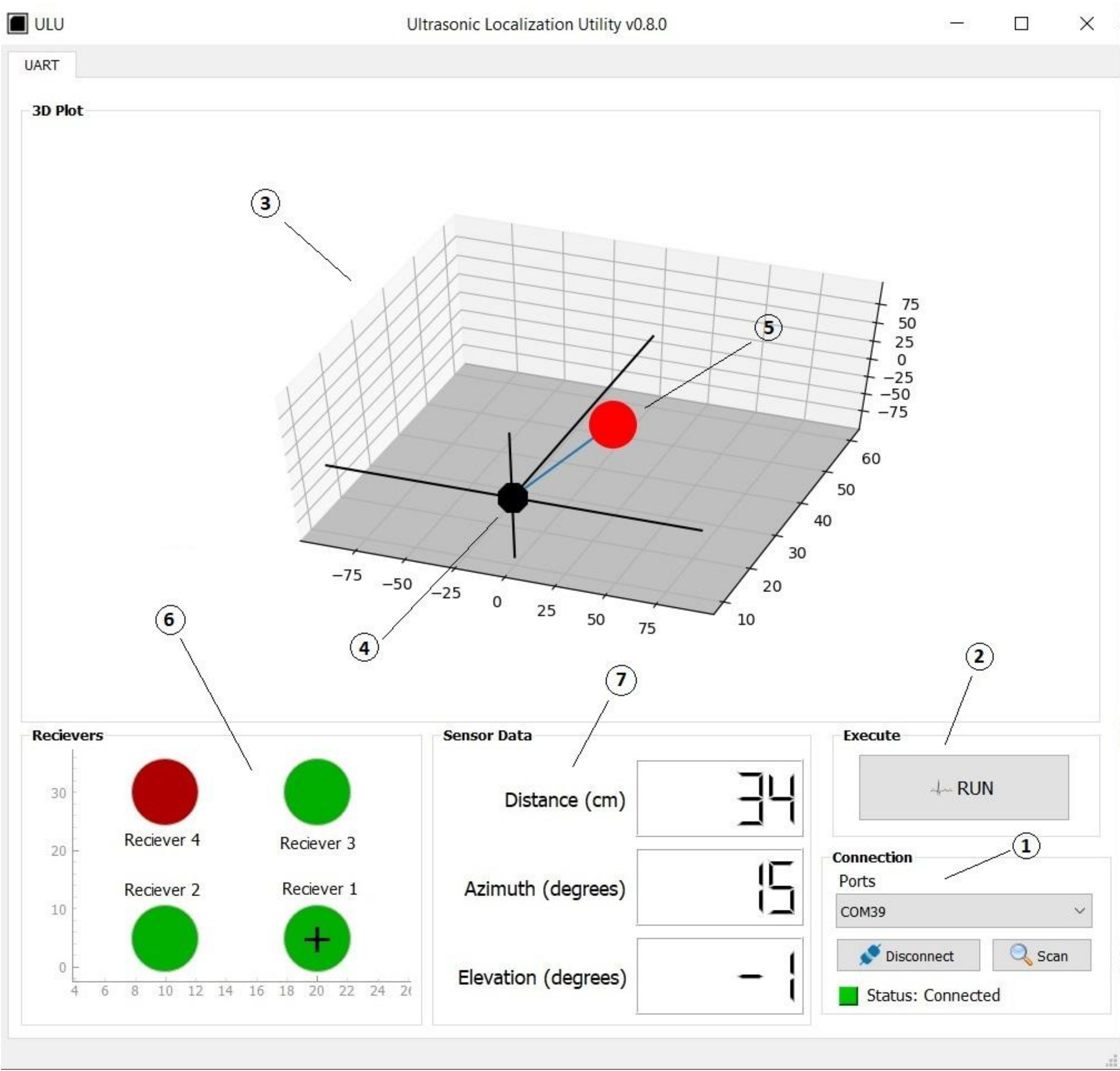

Figure 25: Python GUI

6 ASSEMBLY

The final step is to assemble all the hardware in one place and create a casing that can hold all the transducers firmly together at a specific distance. The receivers of the sensor should not move; they should be placed exactly at the same angle facing the top plane. The slight difference in the angle of the transducers would result in a change in the readings and it may induce a high percentage in the readings. The casing is designed in CAD software Fusion 360 [19]. Fusion 360 is the first cloud-based 3D CAD, CAM, and CAE tool of its kind, connecting the entire product development process into one cloud-based platform. The transducers are placed at a distance of 8cm apart from each other. The overall dimension of casing is 11.5cm x 11.5cm x 1.8cm. After designing the casing the casing is printed with the material PLA by using the 3D printer. All the hardware is placed inside the casing and transducers are mounted precisely to make sure their position will not affect the accuracy of the results. The assembled sensor is shown in Figure 26.

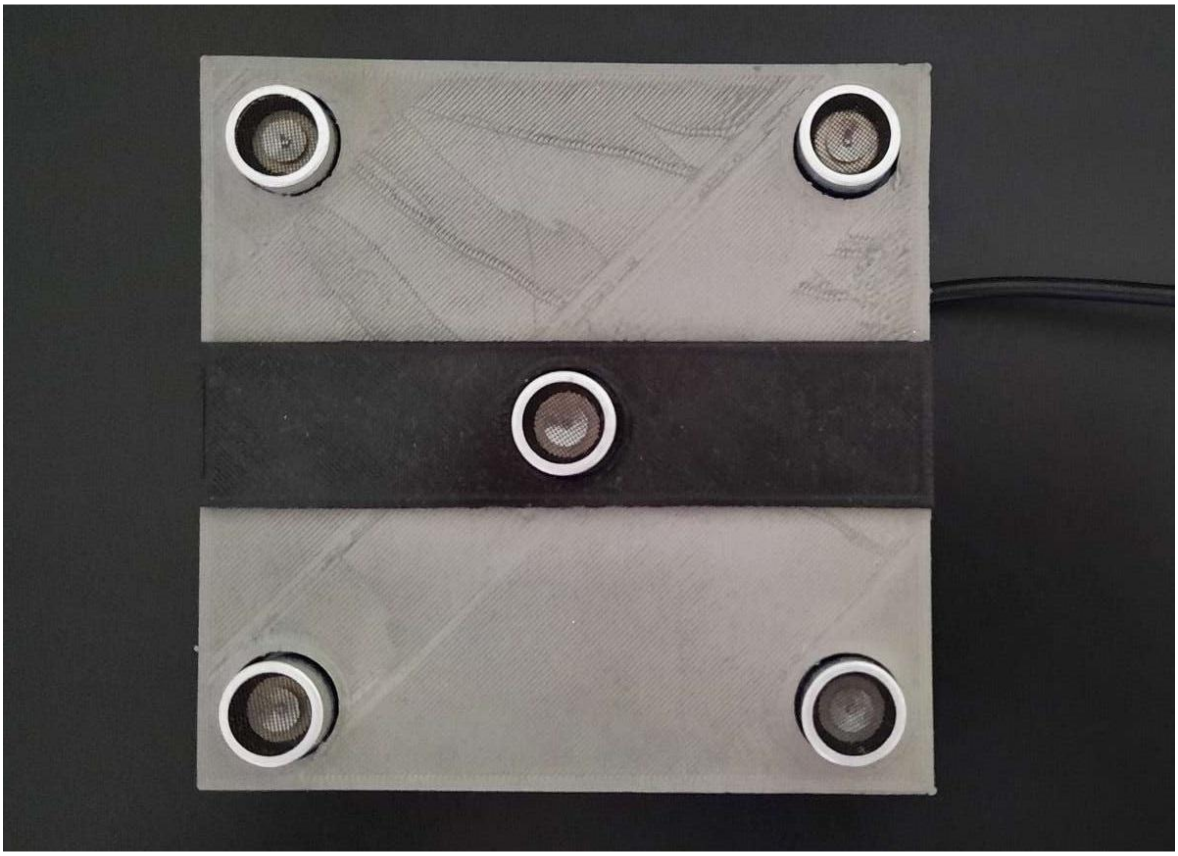

Figure 26: Sensor Image

7 TESTING

The four cases shown in Figure 24 are implemented in a real world environment to test the output of the sensor. For that, the box of dimensions 32cm x 23cm x 12xcm is placed at four different positions according to four cases. The position of the box is measured by the sensor and 3D plot in the python GUI. Four test cases are mentioned below:

- Case 1: Object is placed to the top left side of sensor
- Case 2: Object is placed to the top right side of sensor
- Case 3: Object is placed to the bottom left side of sensor
- Case 4: Object is placed to the bottom right side of sensor

7.1 Case 1

In case 1, the box is placed at a distance of 52cm from the sensor. The vertical distance of the box is 8cm to the left side of the sensor. The height of the box from the ground is 10cm. Figure 27 shows the setup of case 1 from the top view and side view. As the algorithm is designed to detect the position of the closest object, so the standing base of the target box has not interfered with readings.

Figure 27: Case 1 setup image

Figure 28 shows that box is detected in the top left plane of the sensor. The measured values of the position of the box by the sensor are:

- Distance: 53cm
- Azimuth angle: -16°
- Elevation angle: 8°

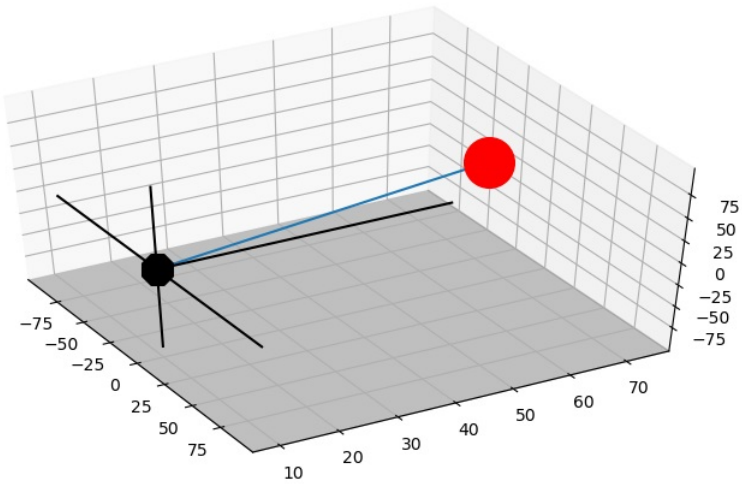

Figure 28: Case 1 sensor output

7.2 Case 2

In case 2, the box is placed at a distance of 52cm from the sensor. The vertical distance of the box is 8cm to the right side of the sensor. The height of the box from the ground is 10cm. Figure 29 shows the setup of case 2 from the top view and side view. All the distances in this case are the same as in case 1, the only difference is that the box is placed at the right side of the sensor.

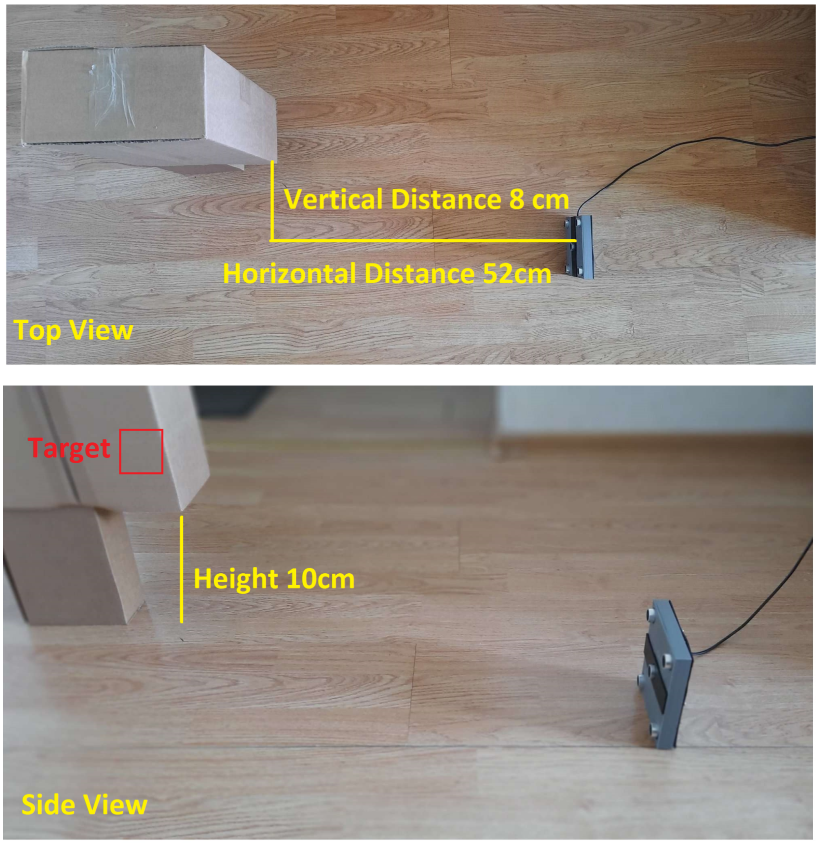

Figure 29: Case 2 setup image

Figure 30 shows that box is detected in the top right plane of the sensor. The measured values of the position of the box by the sensor are:

- Distance: 52m
- Azimuth angle: 14°
- Elevation angle: 4°

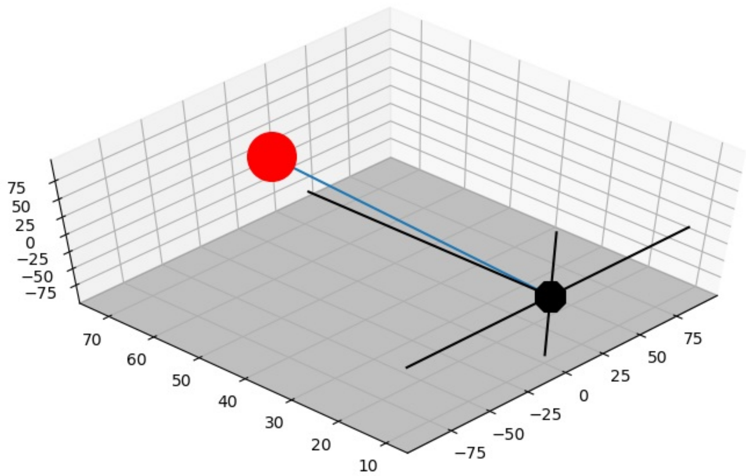

Figure 30: Case 2 sensor output

7.3 Case 3

In case 3, the box is placed at a distance of 85cm from the sensor. The vertical distance of the box is 12cm to the left side of the sensor. The box is 44cm below from the sensor. Figure 31 shows the setup of case 3 from the top view and side view.

Figure 31: Case 3 setup image

Figure 32 shows that box is detected in the bottom left plane of the sensor. The measured values of the position of the box by the sensor are:

- Distance: 85cm
- Azimuth angle: -18°
- Elevation angle: -27°

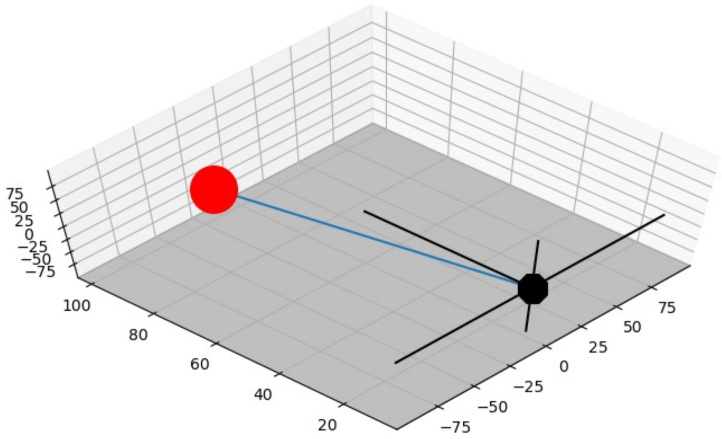

Figure 32: Case 3 sensor output

7.4 Case 4

In case 4, the box is placed at a distance of 85cm from the sensor. The vertical distance of the box is 12cm to the right side of the sensor. The height of the sensor from the ground is the same as in case 3. Figure 33 shows the setup of case 4 from the top view and side view. All the distances in this case are the same as in case 3, the only difference is that the box is placed at the right side of the sensor.

Figure 33: Case 4 setup image

Figure 34 shows that box is detected in the bottom right plane of the sensor. The measured values of the position of the box by the sensor are:

- Distance: 86cm
- Azimuth angle: 20°
- Elevation angle: -22°

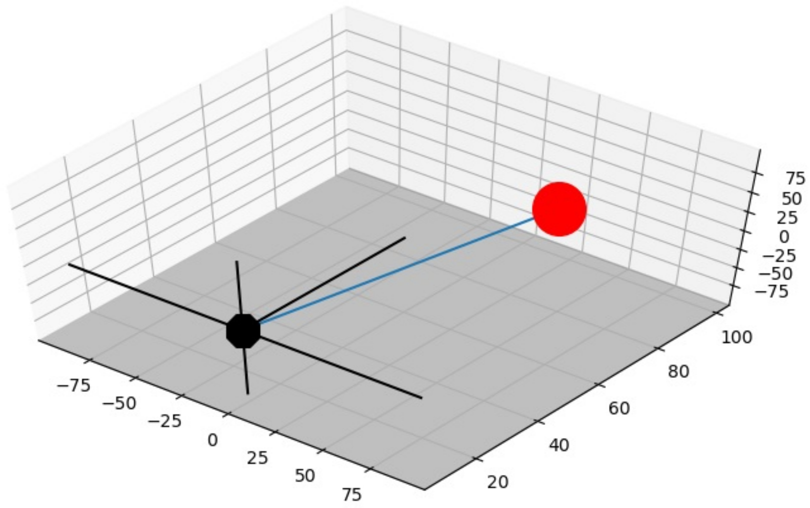

Figure 34: Case 4 sensor output

Table 3 shows the results of testing. The results of the four cases show that the sensor is capable of detecting the object in four planes. The accuracy of the measurement of distance is high as compared to position angles because the dynamic range of time difference caused by elevation/azimuth is very small.

Tabel 3: Testing results

	Actual plane	Detected plane	Plane detection accuracy	Azimuth angle accuracy	Elevation angle accuracy	Distance accuracy
Case 1	top-left	top-left	100%	±5°	±5°	±1cm
Case 2	top-right	top-right				
Case 3	bottom-left	bottom-left				
Case 4	bottom-right	bottom-righ t				

8 SUMMARY

Although light-based proximity sensors and ultrasonic sensors can be used for the same purposes, sound-based sensors are far more common as it requires less complexity and time to design and build them. In some circumstances, they may detect objects more effectively than light-based sensors. For example, while light-based sensors struggle to process transparent plastic correctly, ultrasonic sensors have no such issues. In reality, the color of the material they're sensing has no effect on them. On the other hand, readings would be unreliable if an object is made of a material that absorbs sound or is formed in such a way that it reflects sound waves away from the receiver. There are also other factors which affect the reading of ultrasonic sensors.

The air temperature has the greatest effect on the ultrasonic sensor measuring precision. The sensor uses the speed of sound to determine the distance to the object after measuring the transit time of the reflected ultrasonic pulse. The speed of sound, on the other hand, increases by 0.17 percent per degree Kelvin as the air temperature changes.To compensate for this effect, almost all ultrasonic sensors have a temperature probe. This probe measures the ambient temperature and calculates the speed of sound accordingly this is known as temperature compensation.The humidity is also an external factor that affects the speed of sound thus affecting the readings of the ultrasonic sensor. At room temperature and lower temperatures, humidity has no or very little impact on the speed of sound. However, at higher air temperatures, the humidity has a significant effect on the speed of sound. However, unlike light-based sensors, these factors can be compensated easily by taking the measurement using sensor probes and adjusting the ultrasonic readings accordingly.

8.1 Future work

Ultrasonic transducers made of microelectromechanical systems (MEMS), also known as capacitive micromachined ultrasonic transducers (CMUTs), have emerged as viable alternatives to conventional bulk piezoelectric transducers. MEMS is a technology that can be described as miniaturized mechanical and electro-mechanical elements (i.e., devices and structures) produced using microfabrication techniques in its most general form. MEMS critical physical measurements range from far below one micron on the lower end of the dimensional spectrum to several millimeters on the high end. MEMS devices range in complexity from basic structures with no moving elements to highly complex electromechanical systems with numerous moving elements regulated by

integrated microelectronics. MEMS are also available to receive and transmit high-frequency sound waves generally characterized as ultrasonic waves
Figure 35 shows the miniature size MEMS in a 3.5 mm x 3.5 mm package, MEMS-based ultrasonic technology combines a Time-of-Flight (ToF) range sensor with a power-efficient digital signal processor (DSP) ASIC.

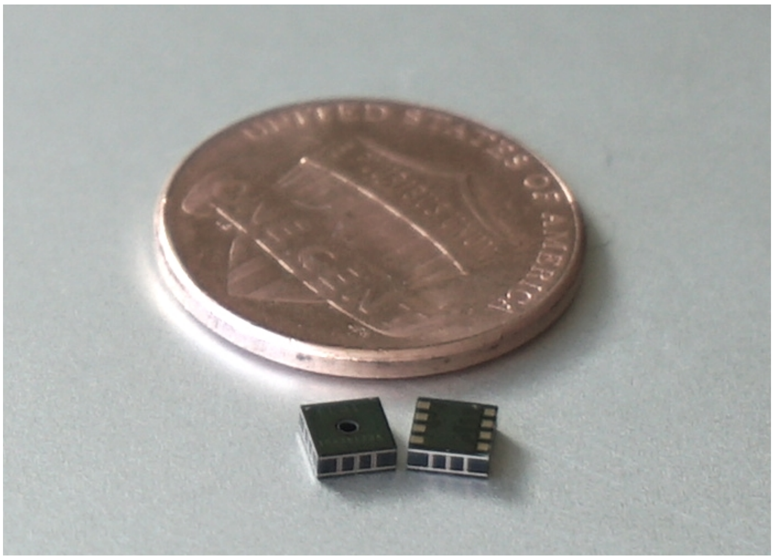

Figure 35: MEMS based ultrasonic TOF sensor [25]

MEMS are replacing traditional ultrasonic transducers due to their miniature size and less complexity to operate them. In the future following changes can be implemented to improve the performance of the sensor.

- 3D ultrasonic sensors can be developed using MEMS which can reduce the size by 1/10 to traditional piezoelectric transducers.
- The current algorithm gives the position of the sensor in the form of angels. It can be modified by adding the functionality of calculating the absolute position of the object in terms of distance coordinates.
- Slightly increasing the operating frequency of the sensor would improve the resolution of the receivers. Which will increase the accuracy of the detection of the object.
- Multiple object detection features can be implemented.

9 KOKKUVÕTE

Kuigi valguspõhiseid lähedusandureid ja ultraheliandureid saab kasutada samadel eesmärkidel, on helipõhised andurid palju levinumad, kuna nende arendamine ja valmistamine nõuab vähem keerukust ja aega. Mõnel juhul võivad nad objekte tuvastada valgusanduritest isegi tõhusamalt. Näiteks kui valguspõhised andurid ei suuda läbipaistvast plastikust infot ammutada, siis ultraheliandurítel selliseid probleeme pole. Samuti ei mõjuta neid nende tajutava materjali värv. Teiselt poolt ei oleks näidud usaldusväärsed, kui objekt on valmistatud materjalist, mis neelab heli või on loodud nii, et see peegeldab helilaineid vastuvõtjast eemal. Ultraheliandurite lugemit mõjutavad küll aga muud tegurid.

Õhutemperatuuril on ultraheliandurі täpsusele mõõtmisel kõige suurem mõju. Pärast peegeldunud ultraheliimpulsi läbimisaja mõõtmist kasutab andur helikiirust objekti kauguse määramiseks. Heli kiirus seevastu suureneb õhutemperatuuri muutudes 0,17 protsenti Kelvini kraadi kohta. Selle efekti kompenseerimiseks on peaaegu kõigil ultraheliandurítel lisaks temperatuuriandur. See andur mõõdab ümbritsevat temperatuuri ja arvutab vastavalt sellele kohandatud helikiiruse. Seda nimetatakse temperatuuri kompenseerimiseks. Niiskus on ka väline tegur, mis mõjutab helikiirust, mõjutades seega ultraheli anduri näite. Toatemperatuuril ja madalamal temperatuuril ei mõjuta niiskus helikiirust või mõjutab seda väga vähe. Kõrgema õhutemperatuuri korral mõjutab niiskus aga helikiirust märkimisväärselt. Kuid erinevalt valguspõhistest anduritest saab neid tegureid hõlpsasti kompenseerida, võttes arvesse keskkonna tegureid ja kohandades vastavalt ultrahelinäidud.

9.1 Edasiarendused

Elektromehaaniline mikrosüsteem (MEMS) valmistatud ultraheliandurid, tuntud ka kui mahtuvuslikud mikrotöötlusega loodud ultrahelimuundurid (CMUT), on kujunenud tugevaks alternatiivideks tavapärastele piezoelektrilistele muunduritele. MEMS on tehnoloogia, mida võib kirjeldada kui miniatuurseid mehaanilisi ja elektromehaanilisi elemente (st seadmeid ja struktuure), mis on toodetud mikrotöötlustehnika abil selle kõige üldisemal kujul. MEMS füüsilised mõõtmed ulatuvad mõõtmete spektri alumises otsas kaugelt alla ühe mikroni kuni mitme millimeetrini. MEMS-seadmed on keerukad alates liikuvate elementideta põhistruktuuridest kuni keerukate elektromehaaniliste süsteemideni, kus integreeritud mikroelektroonika reguleerib arvukalt liikuvaid elemente. MEMS on saadaval ka selliste kõrgsageduslike helilainete vastuvõtmiseks ja edastamiseks, mida tavaliselt iseloomustatakse ultrahelilainetena

Joonisel 35 on kujutatud miniatuurse suurusega MEMS 3,5 mm x 3,5 mm pakendis. MEMS asendab traditsioonilisi ultrahelimuundureid nende miniatuurse suuruse ja nende kasutamise keerukuse tõttu. Anduri jõudluse parandamiseks saab tulevikus sisse viia järgmisi muudatusi.

- 3D ultraheliandureid saab välja töötada MEMS-i abil, mis võib vähendada traditsiooniliste piesoelektriliste muundurite suurust 90% võrra.
- Praegune algoritm annab anduri asukoha ingel-graafikute kujul. Seda saab muuta, lisades objekti absoluutse asukoha arvutamise funktsionaalsuse kauguskoordinaatidena.
- Anduri töösageduse suurendamine parandaks vastuvõtjate eraldusvõimet. See omakorda suurendab objekti tuvastamise täpsust.
- Rakendada saab mitut objekti tuvastamise funktsiooni.

References

[1] B. Grothe, M. Pecka, and D. McAlpine, 'Mechanisms of Sound Localization in Mammals', Physiol. Rev., vol. 90, no. 3, pp. 983–1012, Jul. 2010, doi: 10.1152/physrev.00026.2009.

[2] A. A. Scharine and T. R. Letowski, 'Factors Affecting Auditory Localization and Situational Awareness in the Urban Battlefield', p. 61.

[3] H. E. Heffner and R. S. Heffner, 'The Evolution of Mammalian Sound Localization', p. 8.

[4] J. C. Middlebrooks, 'Sound localization', in *Handbook of Clinical Neurology*, vol. 129, Elsevier, 2015, pp. 99–116. doi: 10.1016/B978-0-444-62630-1.00006-8.

[5] 'Decenei | Ultrasound and infrasound - applications and protection measures'. https://decenei.com/2015/12/07/ultrasunetele-si-infrasunetele-aplicatii-si-masuri-de-protectie (accessed May 15, 2021).

[6] 'ultrasonic principle | ultrasonic sensors | microsonic'. https://www.microsonic.de/en/support/ultrasonic-technology/principle.htm (accessed May 16, 2021)..

[7] 'Factory Automation: Increase Productivity, Increase Quality.', *MaxBotix Inc.*, Oct. 23, 2017. https://www.maxbotix.com/articles/factory-automation.htm (accessed May 12, 2021).

[8] 'Using an Ultrasonic Level Sensor', *MaxBotix Inc.*, Nov. 08, 2019. https://www.maxbotix.com/articles/water-depth-sensing.htm (accessed May 12, 2021).

[9] 'Parking Sensors Differences Explained', Nationwide Vehicle Contracts. https://www.nationwidevehiclecontracts.co.uk/blog/parking-sensors-differences-explained (accessed May 15, 2021).

[10] K. A. Tiwari, R. Raisutis, L. Mazeika, and V. Samaitis, '2D Analytical Model for the Directivity Prediction of Ultrasonic Contact Type Transducers in the Generation of Guided Waves', Sensors, vol. 18, no. 4, Art. no. 4, Apr. 2018, doi: 10.3390/s18040987.

[11] Q. L. D. M. Jing Fan, „Localization Estimation of Sound Source by Microphones Array," Elsevier Ltd., 2010.

[12] P. Aarabi, „The Fusion of DistributedMicrophone Arrays," *Applied Signal Processing,* 2001.

[13] T. I. P. A. Qing Hua Wang, „Acoustic robot navigation using distributed microphone arrays," Elsevier B.V., 2003.

[14] T. S. l. T. F. W. Jie Huang, „A model-based sound localization system and," Elsevier Science B.V, 1999.

[15] F. M. François Grondin, „Lightweight and optimized sound source localization and tracking,“ Elsevier B.V, 2019.

[16] O. D. F. S. A. B. Thomas Padois, „Optimization of a spherical microphone array geometry,“ Elsevier Ltd, 2019.

[17] J. Z. L. Y. M. S. B. X. Z. L. Jun Wu, „A highly accurate ultrasonic ranging method based on onset,“ Elsevier Ltd, 2013.

[18] H. F. Sahdev Kumar, „Long-range measurement system using ultrasonic range sensor with,“ Elsevier B.V, 2016.

[19] Y. J. H. L. W. M. Jongkyu Park, „Design of an ultrasonic sensor for measuring distance and detecting obstacles,“ Elsevier B.V, 2009.

[20] 'MSO-A1640H12TR-aidnoBpoKliRliSrjpnrrlmi.pdf'. Accessed: May 15, 2021. [Online]. Available:https://ikrorwxhjiilll5q.ldycdn.com/MSO-A1640H12TR-aidnoBpoKliRliSrjpnrrlmi.pdf

[21] 'CyberLogic Wave2000 ultrasound simulaiton software'. https://www.cyberlogic.org/wave2000.html (accessed May 15, 2021).

[22] A. Whitehead, 'PGA460 full-bridge driver solutions for ultrasonic transducers', p. 15, 2017.

[23] 'Ultrasonic Ranger - Dushyant Mehta'. https://sites.google.com/site/mehtadushy/ultrasonic-ranger (accessed May 15, 2021).

[24] 'MAX220–MAX249 - +5V-Powered, Multichannel RS-232 Drivers/Receivers', p. 39.

[25] 'MEMS-based "sonar on a silicon chip" ultrasonic time-of-flight sensors', Electronics-Lab.com, Jun. 26, 2019. https://www.electronics-lab.com/mems-based-sonar-silicon-chip-ultrasonic-time-flight-sensors/ (accessed May 16, 2021).

APPENDICES

Appendix 1 MainWindow Python Script

```
from ui.mainwindow import Ui_MainWindow
from ui_uart import TabUART
import sys, os
from PyQt5 import QtWidgets
from PyQt5.QtGui import QIcon
from PyQt5.QtGui import *
import version

class ApplicationWindow(QtWidgets.QMainWindow):
  def __init__(self):
      super(ApplicationWindow, self).__init__()
      self.ui = Ui_MainWindow()
      self.ui.setupUi(self)

      self.setWindowTitle("ULU".ljust(80) + "Ultrasonic Localization Utility v" +
      version.__version__)
      self.setWindowIcon(QIcon(self.resource_path('icons/logo/icon.ico')))

      self.tu = TabUART(self.ui)

  def resource_path(self, relative_path):
      """ Get absolute path to resource, works for dev and for PyInstaller """
      try:
          # PyInstaller creates a temp folder and stores path in _MEIPASS
          base_path = sys._MEIPASS
      except Exception:
          base_path = os.path.abspath(".")

      return os.path.join(base_path, relative_path)

if __name__ == "__main__":
  app = QtWidgets.QApplication(sys.argv)
  application = ApplicationWindow()
  application.show()
  sys.exit(app.exec_())
```

Appendix 2 3D Plot Widget Python Script

```
from PyQt5.QtWidgets import *
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg as
FigureCanvas
from matplotlib.figure import Figure

class MplWidget(QWidget):

  def __init__(self, parent=None):
     QWidget.__init__(self, parent)

     self.canvas = FigureCanvas(Figure())

     vertical_layout = QVBoxLayout()
     vertical_layout.addWidget(self.canvas)

     #self.canvas.figure.
     self.canvas.axes = self.canvas.figure.add_subplot(111, projection="3d")
     self.setLayout(vertical_layout)

     #self.canvas.axes.set_zlabel("Distance (cm)")
     min_axis = -90
     max_axis = 90
     self.canvas.axes.set_xlim3d(min_axis, max_axis)
     self.canvas.axes.set_ylim3d(0, 60)
     self.canvas.axes.set_zlim3d(min_axis, max_axis)
     self.canvas.axes.grid(True)
     self.canvas.axes.w_zaxis.set_pane_color((0.5, 0.5, 0.5, 0.5))

  def addPlot(self, x, y, z, clr):
     min_axis = -90
     max_axis = 90
     axis_mid = 0

     #target plot
     self.canvas.axes.scatter( x, y, z, c = 'r', marker='o', s=30**2)

     #target line plot
     self.canvas.axes.plot([axis_mid,x[0]], [10,y[0]], [axis_mid,z[0]])
     # sensor plot
     self.canvas.axes.scatter(axis_mid, 10, axis_mid, c='k', marker='8', s=20 ** 2)
     # x-line
     self.canvas.axes.plot([min_axis, max_axis], [10, 10], [axis_mid, axis_mid],
c='k')
     # y-line
     self.canvas.axes.plot([axis_mid, axis_mid], [10, 60], [axis_mid, axis_mid],c='k')
     # z-line
     self.canvas.axes.plot([axis_mid, axis_mid], [10, 10], [min_axis, max_axis],
c='k')
     self.canvas.draw()

  def origin(self):
     min_axis = -90
     max_axis = 90
     axis_mid = 0
```

```
        # sensor plot
        self.canvas.axes.scatter(axis_mid, 10, axis_mid, c='k', marker='8', s=20 ** 2)
        # x-line
        self.canvas.axes.plot([min_axis, max_axis], [10, 10], [axis_mid, axis_mid],
c='k')
        # y-line
        self.canvas.axes.plot([axis_mid, axis_mid], [10, 60], [axis_mid, axis_mid],
c='k')
        # z-line
        self.canvas.axes.plot([axis_mid, axis_mid], [10, 10], [min_axis, max_axis],
c='k')
        self.canvas.draw()


    def clear(self):
        self.canvas.axes.clear()

    def resize(self, range):
        self.canvas.axes.set_xlim3d(0, 100)
        self.canvas.axes.set_ylim3d(0, 100)
        self.canvas.axes.set_zlim3d(0, range)
        self.canvas.draw()
```

Appendix 3 UART Driver Python Script

```python
import serial
import serial.tools.list_ports as ports
from PyQt5.QtWidgets import QMessageBox
from PyQt5.QtGui import QIcon

class uart:
  def __init__(self):
    self.ser = serial.Serial()

  def getPorts(self):
    buffer = []
    comPorts = list(ports.comports())
    for dev in comPorts:
      buffer.append(dev.device)
    return buffer

  def connect(self, portname):
    if self.ser.is_open:
      self.ser.close()
      return 1
    else:
      try:
        self.ser.setPort(portname)
        self.ser.baudrate = 115200
        self.ser.timeout = 1
        self.ser.open()
        return 0
      except:
        msgBox = QMessageBox()
        msgBox.setWindowTitle("Error")
        msgBox.setWindowIcon(QIcon('icons/logo/broken_connection.jpg'))
        msgBox.setIcon(QMessageBox.Critical)
        msgBox.setText("Unable To Open Port")
        msgBox.exec()
        return 1


  def transmit(self, data):
    try:
      self.ser.write(data)
      return 0
    except:
      msgBox = QMessageBox()
      msgBox.setWindowTitle("Error")
      msgBox.setWindowIcon(QIcon('icons/logo/broken_connection.jpg'))
      msgBox.setIcon(QMessageBox.Critical)
      msgBox.setText("Communication Error")
      msgBox.exec()
      return -1

  def receive(self, bytes):
    try:
      return self.ser.read(bytes)
    except:
      return -1
```

Appendix 4 Main Algorithm Python Script

```
import os, sys
from PyQt5.QtGui import QIcon
from drivers.dr_uart import uart
import numpy as np
import math

class TabUART():
  def __init__(self, ui):
     self.ui = ui

     # assign functions to button events
     self.ui.pushButton_run.clicked.connect(self.runClicked)
     self.ui.ut_button_connect.clicked.connect(self.buttonConnectClicked)
     self.ui.ut_button_refresh.clicked.connect(self.buttonScanPortsClicked)

     # Main graph instance
     self.graph = 0

     # global varaibles to hold UART data
     self.transmit_frame = 0
     self.receive_frame = 0

     # Initialize UI
     self.initUi()

  def resource_path(self, relative_path):
     """ Get absolute path to resource, works for dev and for PyInstaller """
     try:
        # PyInstaller creates a temp folder and stores path in _MEIPASS
        base_path = sys._MEIPASS
     except Exception:
        base_path = os.path.abspath(".")

     return os.path.join(base_path, relative_path)

  def initUi(self):
     # Initialise graphs
     self.init2DGraph()
     self.init3DGraph()

     # Assign icons to buttons

self.ui.ut_button_connect.setIcon(QIcon(self.resource_path('icons/connect.png')))
     self.ui.ut_button_refresh.setIcon(QIcon(self.resource_path('icons/scan.png')))
     self.ui.pushButton_run.setIcon(QIcon(self.resource_path('icons/echo.png')))

     # Initialize UI widgets to default state
     self.ui.ut_status_led.setStyleSheet("background-color:rgb(200,0,0)");

     # Creat UART instance
     self.usart = uart()
     self.serial_ports = []

     # Scan UART ports
     self.buttonScanPortsClicked()
```

```
    def init2DGraph(self):
        self.ui.ut_widget_graph.setBackground('w')
        self.graph = self.ui.ut_widget_graph.plotItem
        self.graph.setXRange(5, 25)
        self.graph.setYRange(0, 35)
        #BL
        self.graph.plot([10],[5], symbol='o', symbolSize=60, symbolBrush=((175, 0, 0)))
        #TL
        self.graph.plot([10], [30], symbol='o', symbolSize=60, symbolBrush=((175, 0, 0)))
        #BR
        self.graph.plot([20],[5], symbol='o', symbolSize=60, symbolBrush=((175, 0, 0)))
        #TR
        self.graph.plot([20], [30], symbol='o', symbolSize=60, symbolBrush=((175, 0, 0)))

    def clear2DGraph(self):
        self.graph.setXRange(5, 25)
        self.graph.setYRange(0, 35)
        #BL
        self.graph.plot([10],[5], symbol='o', symbolSize=60, symbolBrush=((175, 0, 0)))
        #TL
        self.graph.plot([10], [30], symbol='o', symbolSize=60, symbolBrush=((175, 0, 0)))
        #BR
        self.graph.plot([20],[5], symbol='o', symbolSize=60, symbolBrush=((175, 0, 0)))
        #TR
        self.graph.plot([20], [30], symbol='o', symbolSize=60, symbolBrush=((175, 0, 0)))
        self.ui.label_R1.setText("")
        self.ui.label_R2.setText("")
        self.ui.label_R3.setText("")
        self.ui.label_R4.setText("")

    def init3DGraph(self):
        self.ui.MplWidget.origin()


    def buttonScanPortsClicked(self):
        ports = self.usart.getPorts()
        self.ui.ut_comboBox_port.clear()
        for range in ports:
            self.ui.ut_comboBox_port.addItem(range)

    def buttonConnectClicked(self):
        port = self.ui.ut_comboBox_port.currentText()
        if self.usart.connect(port):
            self.ui.ut_button_connect.setText("Connect")
            self.ui.ut_label_connection_status.setText("Status: Disconnected")
            self.ui.ut_status_led.setStyleSheet("background-color:rgb(200,0,0)");
        else:
            self.ui.ut_button_connect.setText("Disconnect")
```

```
        self.ui.ut_label_connection_status.setText("Status: Connected")
        self.ui.ut_status_led.setStyleSheet("background-color:rgb(0,200,0)");
        self.buttonScanPortsClicked()

    def runClicked(self):

        avg_azimuth_buffer = []
        avg_elevation_buffer = []
        count_buffer = 0
        average_count = 15

        # Distane between two transducers in cm
        TRANSDUCER_DISTANCE = 8.0
        PI = 3.14

        transducer_pivot = 0xFF
        transducer_elevation = 0xFF
        transducer_azimuth = 0xFF

        angle_elevation = -1.0
        angle_azimuth = -1.0

        while count_buffer != average_count:
            self.transmit_frame = [0xAA]
            self.usart.transmit(self.transmit_frame)
            rx_buffer = self.usart.receive(10)

            echo_time_BL = ((rx_buffer[2] << 8) & 0xFF00) | rx_buffer[1]
            echo_time_BR = ((rx_buffer[4] << 8) & 0xFF00) | rx_buffer[3]
            echo_time_TL = ((rx_buffer[6] << 8) & 0xFF00) | rx_buffer[5]
            echo_time_TR = ((rx_buffer[8] << 8) & 0xFF00) | rx_buffer[7]

            distBL = echo_time_BL / 29.0 / 2.0
            distBR = echo_time_BR / 29.0 / 2.0
            distTL = echo_time_TL / 29.0 / 2.0
            distTR = echo_time_TR / 29.0 / 2.0

            data_buffer = [echo_time_BL, echo_time_BR, echo_time_TL, echo_time_TR]

            echo_shortest_index = data_buffer.index(min(data_buffer))

            # First echo received by Bottom Left transducer
            if (echo_shortest_index == 0):
                transducer_pivot = 0
                # Calculate azimuth angle
                timeDifference = echo_time_BR - echo_time_BL
                try:
                    angle = math.asin((timeDifference * 0.000001 * 34300.00) /
(TRANSDUCER_DISTANCE))
                    angle = (angle / (2 * PI)) * 360
                    angle = angle /2
                    angle_azimuth = int(0 - angle)
                    transducer_azimuth = 1
                except:
                    angle_azimuth = -40

                # Calculate elevation angle
```

```
    timeDifference = echo_time_TL - echo_time_BL
    try:
        angle = math.asin((timeDifference * 0.000001 * 34300.00) /
        (TRANSDUCER_DISTANCE))
        angle = (angle / (2 * PI)) * 360
        angle = angle / 2
        angle_elevation = int(0 - angle)
        transducer_elevation = 2
    except:
        angle_elevation = -40

# First echo received by Bottom Right transducer
elif (echo_shortest_index == 1):
    transducer_pivot = 1
    # Calculate azimuth angle
    timeDifference = echo_time_BL - echo_time_BR
    try:
        angle = math.asin((timeDifference * 0.000001 * 34300.00) /
        (TRANSDUCER_DISTANCE))
        angle = (angle / (2 * PI)) * 360
        angle = angle / 2
        angle_azimuth = int(angle)
        transducer_azimuth = 0
    except:
        angle_azimuth = 40

    # Calculate elevation angle
    timeDifference = echo_time_TR - echo_time_BR
    try:
        angle = math.asin((timeDifference * 0.000001 * 34300.00) /
        (TRANSDUCER_DISTANCE))
        angle = (angle / (2 * PI)) * 360
        angle = angle / 2
        angle_elevation = int(0 - angle)
        transducer_elevation = 3
    except:
        angle_elevation = -40

# First echo received by Top Left transducer
elif (echo_shortest_index == 2):
    transducer_pivot = 2
    # Calculate azimuth angle
    timeDifference = echo_time_TR - echo_time_TL
    try:
        angle = math.asin((timeDifference * 0.000001 * 34300.00) /
        (TRANSDUCER_DISTANCE))
        angle = (angle / (2 * PI)) * 360
        angle = angle / 2
        angle_azimuth = int(0 - angle)
        transducer_azimuth = 3
    except:
        angle_azimuth = -40

    # Calculate elevation angle
    timeDifference = echo_time_BL - echo_time_TL
    try:
        angle = math.asin((timeDifference * 0.000001 * 34300.00) /
```

```
            (TRANSDUCER_DISTANCE))
            angle = (angle / (2 * PI)) * 360
            angle = angle / 2
            angle_elevation = int(angle)
            transducer_elevation = 0
        except:
            angle_elevation = 40

    # First echo received by Top Right transducer
    elif (echo_shortest_index == 3):
        transducer_pivot = 3
        # Calculate azimuth angle
        timeDifference = echo_time_TL - echo_time_TR
        try:
            angle = math.asin((timeDifference * 0.000001 * 34300.00) /
            (TRANSDUCER_DISTANCE))
            angle = (angle / (2 * PI)) * 360
            angle = angle / 2
            angle_azimuth = int(angle)
            transducer_azimuth = 2
        except:
            angle_azimuth = 40

        # Calculate elevation angle
        timeDifference = echo_time_BR - echo_time_TR
        try:
            angle = math.asin((timeDifference * 0.000001 * 34300.00) /
            (TRANSDUCER_DISTANCE))
            angle = (angle / (2 * PI)) * 360
            angle = angle / 2
            angle_elevation = int(angle)
            transducer_elevation = 1
        except:
            angle_elevation = 40

    distance_buffer = [distBL, distBR, distTL, distTR]
    distance = min(distance_buffer)

    avg_azimuth_buffer.append(angle_azimuth)
    avg_elevation_buffer.append(angle_elevation)

    count_buffer += 1

    if (count_buffer == average_count):
        azimuthal = (int)(self.Average(avg_azimuth_buffer))
        elevation = (int)(self.Average(avg_elevation_buffer))

        self.ui.lcdNumber_distance.display((int)(distance))
        self.ui.lcdNumber_azimuth.display(azimuthal)
        self.ui.lcdNumber_elevation.display(elevation)
        self.clear2DGraph()
        self.update2DGraph(transducer_pivot, transducer_azimuth,
        transducer_elevation)
        self.ui.MplWidget.clear()
        self.ui.MplWidget.addPlot([azimuthal], [(int)(distance)], [elevation], 'r')

        #print(avg_azimuth_buffer, "Common:", common(avg_azimuth_buffer), "
```

```
            Average:", Average(avg_azimuth_buffer))
            # print("Distance:", "{:.2f}".format(distBL)," " , "{:.2f}".format(distBL))

            avg_azimuth_buffer.clear()
            avg_elevation_buffer.clear()

        transducer_pivot = 0xFF
        transducer_elevation = 0xFF
        transducer_azimuth = 0xFF

    # function to get average of a list
    def Average(self, lst):
        return (int)(sum(lst) / len(lst))

    # function to find out most common element in list
    def common(self, lst):
        word_counter = {}
        for word in lst:
            if word in word_counter:
                word_counter[word] += 1
            else:
                word_counter[word] = 1

        popular_words = sorted(word_counter, key=word_counter.get, reverse=True)

        top = popular_words[:1]
        return top[0]

    def update2DGraph(self, pivot, azimuth, elevation):
        if (pivot == 0 or elevation == 0 or azimuth == 0 ):
            # BL
            self.graph.plot([10], [5], symbol='o', symbolSize=60, symbolBrush=((0, 175,
            0)))
        if (pivot == 1 or elevation == 1 or azimuth == 1):
            # BR
            self.graph.plot([20], [5], symbol='o', symbolSize=60, symbolBrush=((0, 175,
            0)))
        if (pivot == 2 or elevation == 2 or azimuth == 2):
            # TL
            self.graph.plot([10], [30], symbol='o', symbolSize=60, symbolBrush=((0,
175,
            0)))
        if (pivot == 3 or elevation == 3 or azimuth == 3):
            # TR
            self.graph.plot([20], [30], symbol='o', symbolSize=60, symbolBrush=((0,
175,
            0)))
        if (pivot == 0):
            self.ui.label_R1.setText("+")
        elif (pivot == 1):
            self.ui.label_R2.setText("+")
        elif (pivot == 2):
            self.ui.label_R3.setText("+")
        elif (pivot == 3):
            self.ui.label_R4.setText("+")
```