Atılım University
Faculty of Engineering

MECE 422 Multidisciplinary Engineering Design

Spring 2022-2023

Final Report

Revolutionizing Post-Earthquake Operations with an Innovative Search and Rescue System, Powered by Smart Sensor Technology

Team Members

Yekta Güngör PARLAK	(EE)
Khamis Yousef AL-OBAIDLY	(MECE)
Berkay KOÇAK	(MECE)
Ömer Bahadır KUNDAK	(CMPE)
Cemre SÜMER	(CMPE)
Hasan ZİYADE	(MECE)

Course Instructors: Dr. Ali AMINI, Dr. Cansu Ç. EKİN, Dr. Zühal ERDEN

Project Supervisor: Dr. Zühal ERDEN

June 1, 2023
Ankara / Turkey

TABLE OF CONTENTS

Page

1. Introduction

Turkey has always been prone to earthquakes because it is located in a seismically active area where several tectonic plates converge. In recent years, these devastating events have become increasingly more frequent in Turkey. The most recent earthquake struck Kahramanmaraş and its surrounding epicentre areas on February 6, 2023, highlighting the urgent need for new solutions to aid search and rescue teams in saving lives. According to figures from the Ministry of Interior tragedy and Emergency Management Presidency (AFAD), this tragedy killed over 45,000 individuals and impacted over 16.3 million inhabitants. Search and rescue personnel during the aftermath of these kinds of disasters confront the arduous job of discovering individuals who might be trapped under rubble. The primary problem that our project aims to address is the difficulty of locating survivors trapped in post-earthquake debris. This task is challenging for search and rescue teams due to various factors, such as the complex and unpredictable nature of debris, time constraints, and challenging environmental conditions. The existing technologies used for search and rescue operations have limitations, and there is a need for innovative solutions to improve the effectiveness and efficiency of these operations. Our team is motivated by the urgent need for innovative solutions to address the challenges of post-earthquake search and rescue operations, and we are excited to take on this challenge. We believe that our project has the potential to make a significant impact in the world, and we are committed to developing a solution that can help search and rescue teams save more lives. Our project requires expertise from various fields, including electrical and electronics engineering, mechatronics engineering, and computer engineering. Electrical and electronics engineering will be able to design the wireless sensor network communication system. Mechatronics engineering will be able to design smart sensor systems. Computer engineering will be able to develop a data processing algorithm and mobile application for the sake of the project aim. Collaboration between these three fields will be critical to the project's success. The multidisciplinary nature of the project may pose challenges to effective teamwork and collaboration. To address these challenges, clear communication channels have been established to ensure that everyone understands their roles and responsibilities. Along with regular group meetings, necessary steps were taken to ensure that everyone is working towards the same goal and that the project was on track.

2. Literature and Patent Survey

This section aims to provide a chronological overview of the historical methods used for search and rescue of survivors, assess their strengths and weaknesses, identify gaps in the literature, and highlight previous studies and milestones associated with them. The objective of this section is to demonstrate how our proposed technology solution can address the gaps in the current literature.

2.1. K-9 Search Units

Dogs have been utilized for search and rescue operations in various fields, but their use in earthquake search and rescue (ESAR) is a relatively recent development. Hare et al. (2018) reports that German Shepherd Dogs were first trained for this purpose, and were successful in locating survivors after the Mexico City earthquake in 1985. Since then, ESAR dogs have been an integral part of disaster response teams globally. However, K-9 units face certain limitations and challenges, such as high training costs, dependency on handlers, and environmental factors like heat, noise, and debris, which may impact their performance.

2.2. Remote Sensing Search Technologies

Given the significance of K-9 search unit systems and the continuous advancement of remote sensing technologies, it is vital to recognize the pivotal role of the latter in post-earthquake search and rescue operations. To this end, conducting a comprehensive literature review is imperative in gaining insight into the current state of remote sensing technologies. Among the range of sensing technologies available, SAR, PIR, UWB, and mmWave sensors are particularly noteworthy, given their critical contribution to post-disaster search and rescue efforts.

2.2.1 Synthetic Aperture Radar (SAR) Sensor

Synthetic Aperture Radar (SAR) is a technology that utilizes microwave signals to generate high-resolution images of the ground surface, making it suitable for detecting changes caused by earthquakes. Using a surface-based SAR systems and a shaking table, M. Pieraccini et al. (2002) established that SAR interferometry may be utilized to identify changes in structure in buildings caused by an earthquake. Although SAR was not mentioned in the context of locating survivors, its relevance could also be used for locating survivors trapped under debris. Nonetheless, it has been integrated with satellites to enhance its capabilities, which has proved beneficial in imaging locations of potential survivors.

2.2.2. Passive Infrared Sensor (PIR)

PIR is another technology that detects infrared radiation emitted by warm objects such as human bodies. PIR sensors can sense human presence and movement, but they cannot measure vital signs or locate victims under thick rubble. Nevertheless, according to Hashimoto et al. (2017), PIR sensors can capture and analyse data to locate the presence of survivors, as demonstrated during the 2016 Kumamoto earthquake. A wearable personal monitoring system patent from 2015 proposed adapting the PIR sensor with a wearable search system for locating humans. However, PIR also has some drawbacks, such as its inability to differentiate between living and non-living objects or measure the vital signs of survivors.

2.2.3. Ultra-Wideband Radar (UWB) Sensor

Ultra-Wideband Radar (UWB) is a remote sensing technology that uses electromagnetic signals to detect and locate survivors under rubble. Several studies have focused on improving UWB's effectiveness through antenna designs and data processing algorithms, and the EU's FP7 research project established a standardized protocol for UWB radar testing in 2012. However, the method itself tends to be complex and expensive. NASA has a product named as FINDER for detecting heartbeats trapped under wreckage, which can be used for locating survivors after an earthquake. Despite its potential, UWB also has some disadvantages, including limited range, high cost, and regulation issues. There are also trade-offs between bandwidth and power, which may affect its precision and reliability. Lastly named Cursor found by EU, used in 2023 Hatay Earthquake and that system had UWB sensors.

2.2.4. Millimeter-Wave Radar (mmWave) Sensor

Millimeter-wave radar sensors were recently offered as a possible option enabling human detection and tracking. according to N. Dahnoun et al. (2021), a system architecture for high-precision person detection and tracking utilizing mmWave radars. The study proved that mmWave radars function well in interior situations, reaching over 90% sensitivity. A miniaturized mmWave radar sensor has been developed and patented by by L. Albasha (2020) named as “US20200326416A1 - Miniaturized digital radar system,” established which can be used for distance measurements with high accuracy in the single-digit micro meter range. Overall, mmWave radar sensors have shown great potential for various applications, particularly for human detection and tracking. In terms of durability, stability, powerful penetration of its signal wave, and recent yearly lowering of cost, it can be the best case for usage in our system as a sensor.

2.3. Robotics & Unmanned Aerial Vehicles (UAVs) Search

UAVs and robots are vital tools for locating survivors under debris after disasters, with the earliest known use of UAVs for this purpose dating back to the 1930s and the first use of robots in the 1980s. Despite the difficulties, ongoing research and development in UAV and robotics technology holds promise for the future of disaster response. However, there are several obstacles to overcome, such as the cost of robotics technology, the need for specialized training, and the possibility of technical issues. To address these limitations and challenges, a research paper by Khan et al. (2022) named as "Emerging UAV Technology for disaster detection, mitigation, response, and preparedness" provides an overview of the current state and future directions of Robots and UAV applications in disaster management. Meanwhile, robots and UAVs also have some drawbacks, such as the cost of robotics technology, the need for specialized training, and the potential for technical malfunctions. and the challenge of accessing disaster sites. A research paper by S. Vijayaragavan et al. (2012), "Live Human Detecting Robot," proposes an autonomous robotic vehicle that moves in the earthquake prone area and helps in identifying the live people and rescue operations.

2.4. Wireless Network Systems (WSNs) Search

Wireless Sensor Networks (WSNs) are composed of multiple devices, called sensor nodes, that are deployed in a particular area and collectively sense the environment through their sensors. Wireless Sensor Network (WSN) technology has been evolving since the 1990s and has been applied in various fields such as environmental monitoring and smart agriculture. A paper by J. Wang et al. (2011) discussed that it is possible to use WSNs in post-earthquake search and rescue operations in the field with 3D localization method for survivors. WSNs can provide a reliable, cost-effective, and real-time solution for locating and communicating with survivors in disaster-stricken areas, where traditional communication infrastructures are often limited. The adaptability of WSNs makes them a promising solution for post-earthquake survivor search operations.

Overall, as mentioned, various technologies and methods have been used for post-earthquake victim search, including dogs, sensors, and robots. However, our project aims to scan a larger area in real-time more reliably than existing methods, and detect more victims under debris in a shorter time. There is currently no literature on combining mmWave radar with multiple WSN module nodes. Thus, our solution is expected to fill the gaps in existing technologies and provide a reliable and efficient solution for post-earthquake search and rescue operations.

3. Needs, Objectives and Scope

We conducted research on the earthquake that affected millions of people in Hatay in 2023. We used news articles and interviews to gather survivors' needs regarding locating those trapped in earthquake debris. In addition, we interviewed Burç Naşit ONGAN, an experienced Search and Rescue Association (AKUT) field worker in earthquake response operations, on the phone to analyse the difficulties and needs in locating the stranded people.

The needs of end-users for the initial design problem were systematically expressed through a needs assessment process. This included environmental, sensor, wireless networking, and infrastructure-related causes. The problem statement was refined to better reflect the needs and priorities of customers and end users by providing context and defining key terms.

The system needs to meet several requirements to locate survivors trapped in earthquake debris effectively and efficiently. Based on our data analysis, we derived and categorized them into 4 main root causes of the needs. With that, we will be able to quickly detect our objectives and scope as well.

The needs of end users for our initial design problem which is locating survivors trapped in earthquake debris has been given below in a systematic way of fishbone root cause diagram in Figure 1.

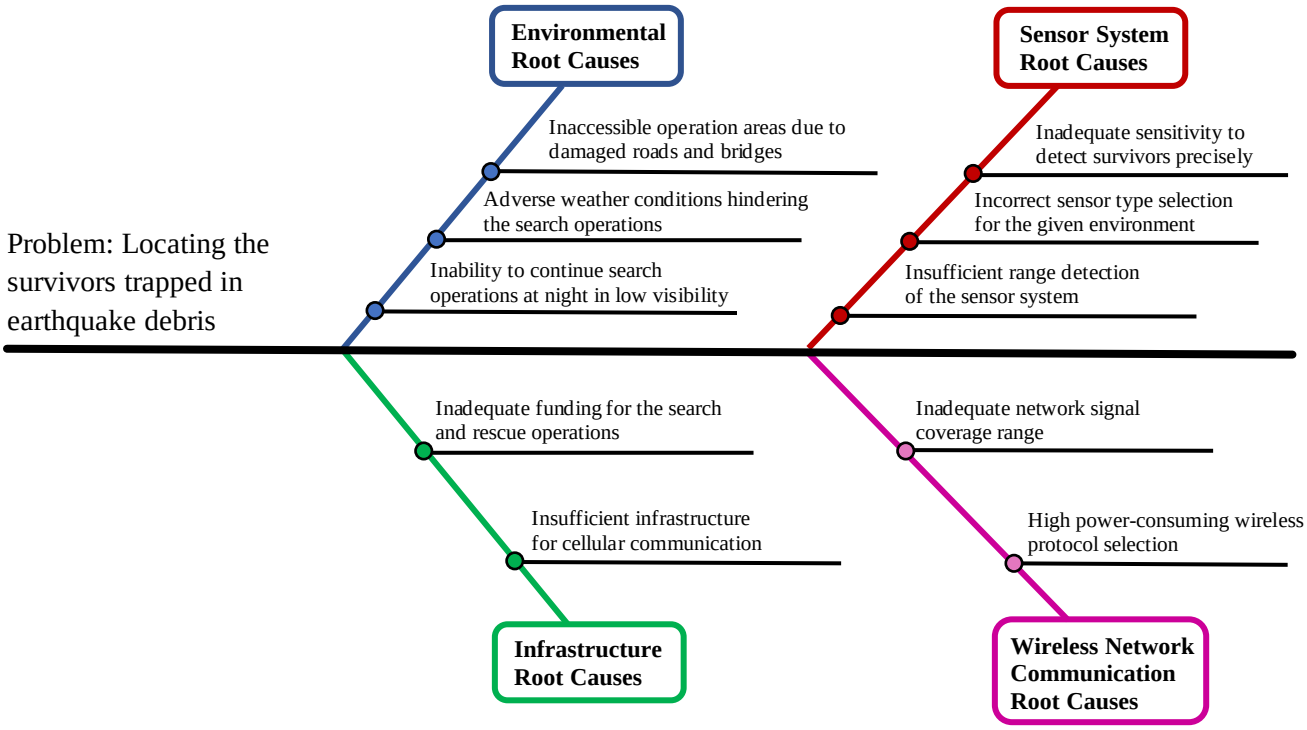


Figure 1. Fishbone Root Cause Analysis Diagram

After conducting a thorough needs assessment, it was identified that there is a problem related to locating survivors who are trapped in debris after an earthquake. The overall objective of this project is to develop a system that can detect the location of survivors in a timely and accurate manner in the event of an earthquake. To achieve this objective, the scope of the project includes to have a sufficiently long range to encompass a huge search region, be sensitive enough to detect small changes in physical phenomena in harsh environmental conditions without failing during critical moments and also be affordable. Additionally, the system needs to be lightweight and portable, have low power consumption and long battery life, and provide real-time location data of trapped survivors. The system should also be affordable to facilitate its widespread deployment in disaster-prone areas. Defining key terms such as "range", "sensitivity", "power usage", and "affordability " are necessary to better understand and address the problem.

Based on the customer needs assessment and project scope the specific objectives to achieve this goal are as follows:

1. Implement sensors with a sensitive, and long-range capacity inside the system.
2. Implement Wireless Sensor Network (WSN) communication protocol for efficient transmission of data from the sensors to the base station, with low power usage for the system.
3. Maximize the battery life of devices to enable long-term operation.
4. Provide an affordable and scalable solution for widespread deployment in disaster-prone areas.

The list of specific objectives provided has been specifically designed to achieve the overall objective of developing a system to detect the location of survivors in a timely and accurate manner in the event of an earthquake. Based on the needs assessment and project scope, the objectives cover the key aspects of system design and development. However, further details on how the objectives will be evaluated and validated, such as metrics to assess system performance and outcomes, will be added to improve the list in the following page.

Table 1 below summarizes the relationships between the project objectives and the customer needs.

Table 1: Relationship among project objectives and customer requirements

Customer Needs	**Project Objectives**
Accurate detection of survivors under the debris	Develop a sensitive, long-range capacity sensor system and.
Low power wireless transmission of location data from sensors to base station	Implement suitable Wireless Sensor Network System (WSN) protocol to ensure transmission data from sensor nodes to station of base in the way of wireless.
Long battery life detection devices	Design a system that can operate in remote environments with minimal power to prolong battery life, reduce the need for constant charging.
Affordable overall system	Implement cost-effective solution meets customer needs while maintaining a reasonable budget.

We defined objectives for our project, such as developing a system to locate survivors trapped in earthquake debris, and defined quantifiable metrics to measure their performance as follows:

1. The selection of sensor types is crucial to ensure the successful detection of survivors in earthquake debris. In order to penetrate thick debris materials such as walls and metals, it should be selected for sensors that operate at a high frequency, with a minimum of 20GHz.

2. The range of each sensor node should be at least 10 meters collectively to accurately detect human presence and cover a large search area. The system should be able to detect at least a $50m^2$ area, allowing for efficient search and rescue operations.

3. The selection of an appropriate communication protocol is essential for a wireless sensor network, as it should use maximum of 100 mW power to maximize battery life and enable long-term operation.

4. Affordability is also a key consideration for the project. A metric for an affordable system should be set at no more than 10,000TRY. This will help ensure the system remains competitive with current market costs of search and rescue technology systems, which typically cost around 100,000TRY.

4. Development of Conceptual Design Alternatives

4.1. 1st Design Alternative

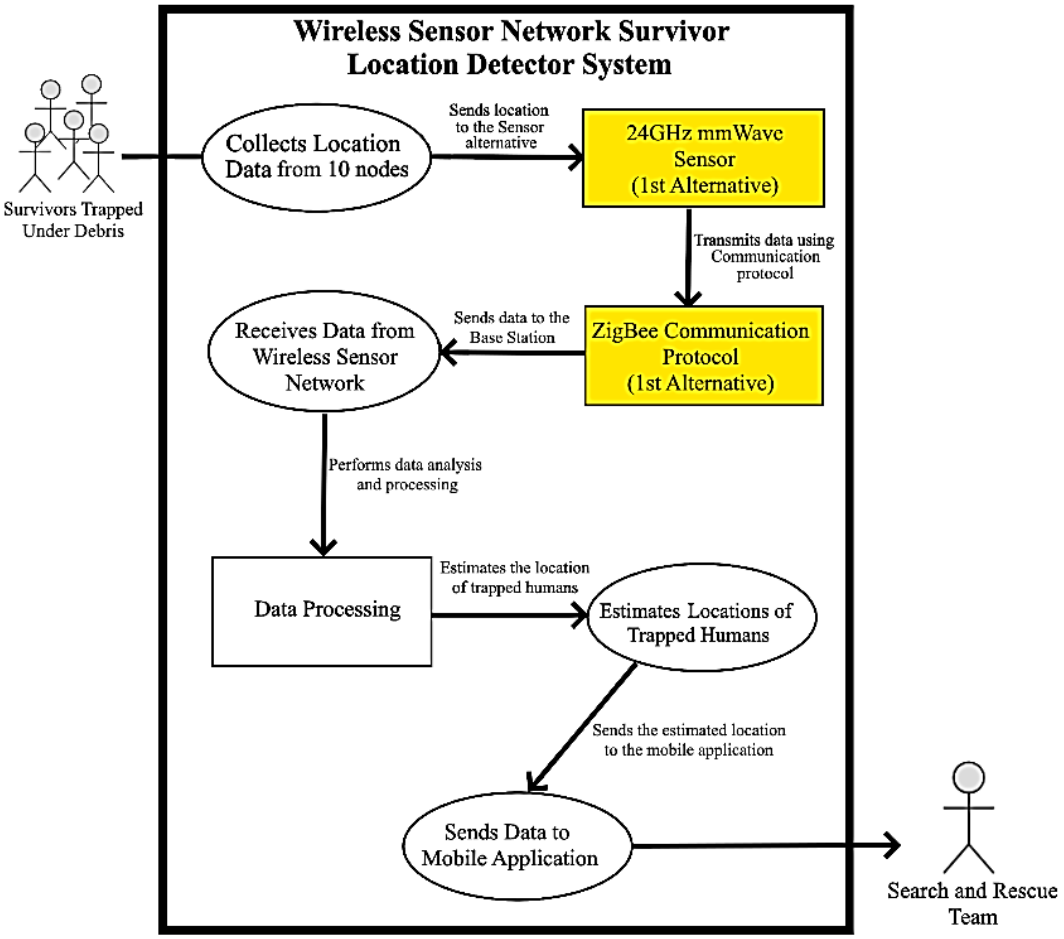


Figure 2. Use-Case Diagram for 1st Design Alternative

Our first design alternative involves using the 24GHz Millimeter-wave (mmWave) sensor to detect the presence of survivors and ZigBee protocol for WSN communication. The sensor is chosen due to its ability to penetrate thick debris materials, with long range capacity as 13 meters making it suitable for detecting survivors trapped in earthquake debris. ZigBee protocol is selected for its low power consumption and efficient data transmission. Advantages of this design alternative include the 24GHz mmWave sensor is reliable, sensitive, and has a long-range capacity, making it suitable for detecting survivors trapped under earthquake debris. ZigBee is a wireless communication technology with a low power, with a built-in mesh networking capability, making it scalable and widely deployed in disaster-prone areas. However, there are some potential disadvantages to consider. The use of a high-frequency sensor may lead to higher cost and complexity in implementation. So, the proper specific model which is not over cost should be find to integrated as mmWave sensors in our system Nonetheless, this design alternative represents a promising solution to the challenge of locating survivors trapped in earthquake debris. Our the code of Use-Case diagram was given as PlanUML in Appendix A.

4.2. 2nd Design Alternative

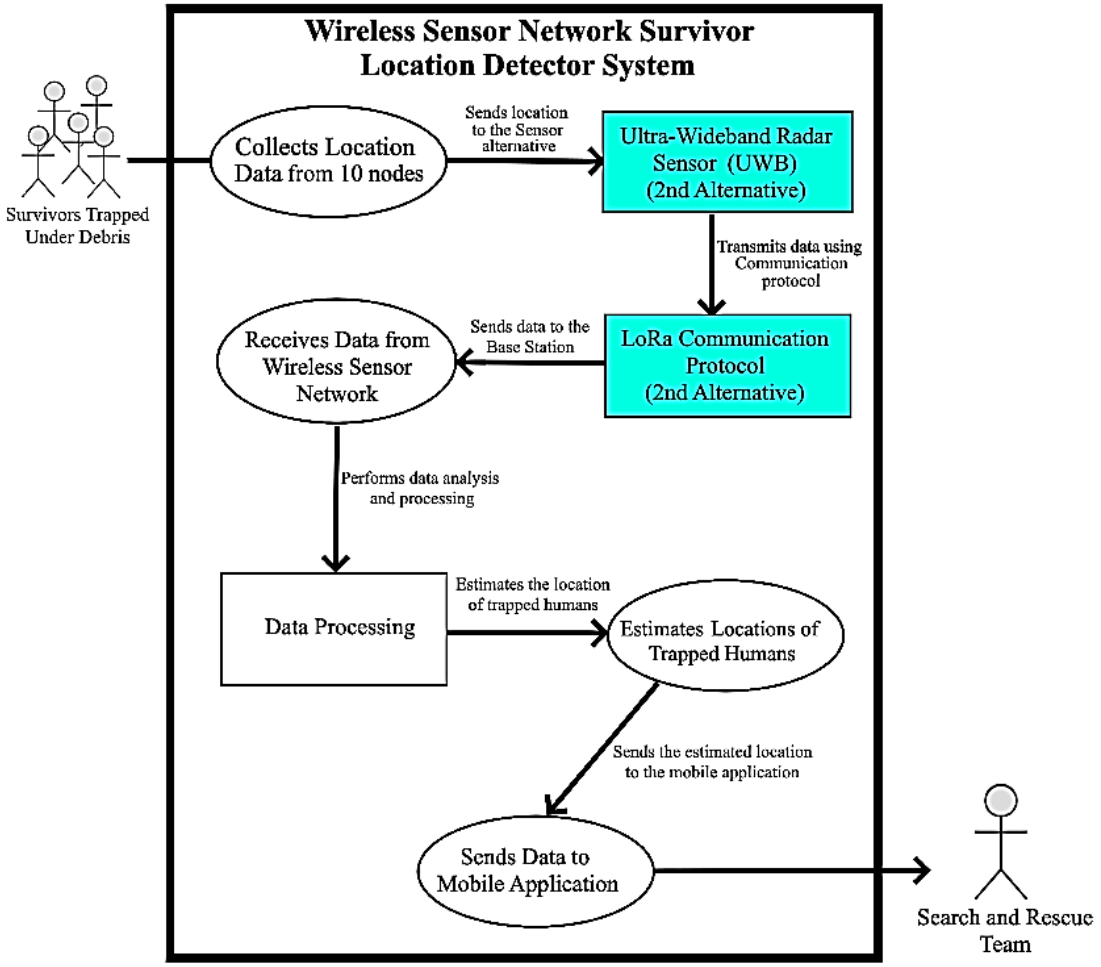


Figure 3. Use-Case Diagram for 2nd Design Alternative

Our second design alternative presents another approach to detecting survivors in emergency situations. This alternative incorporates the use of an ultra-wideband radar (UWB) sensor and the LoRa communication protocol. This sensor operates at a frequency between 3.1 and 10.6GHz and one of the significant advantages of this design alternative is the UWB radar sensor's ability to detect survivors even through walls, debris, and other obstacles, making it highly accurate in detecting survivors. Yet, its range is only 7 meters. This capability significantly decreases the chances of rescuing survivors in emergency situations. Additionally, the LoRa protocol is known for its low-power and long-range communication capabilities, making it ideal for use in disaster-prone areas. The high penetration capabilities of LoRa make it an excellent choice for communication in such areas. However, the implementation of UWB radar sensors may be more unstable, costly and complex than other sensor options. This may lead to an increase in the overall cost of the system. Furthermore, the LoRa protocol's slower data transmission rates compared to other protocols could potentially lead to delays in rescue operations, which may put survivors at risk and hinder the system’s real-time transmission capability.

4.3. 3rd Design Alternative

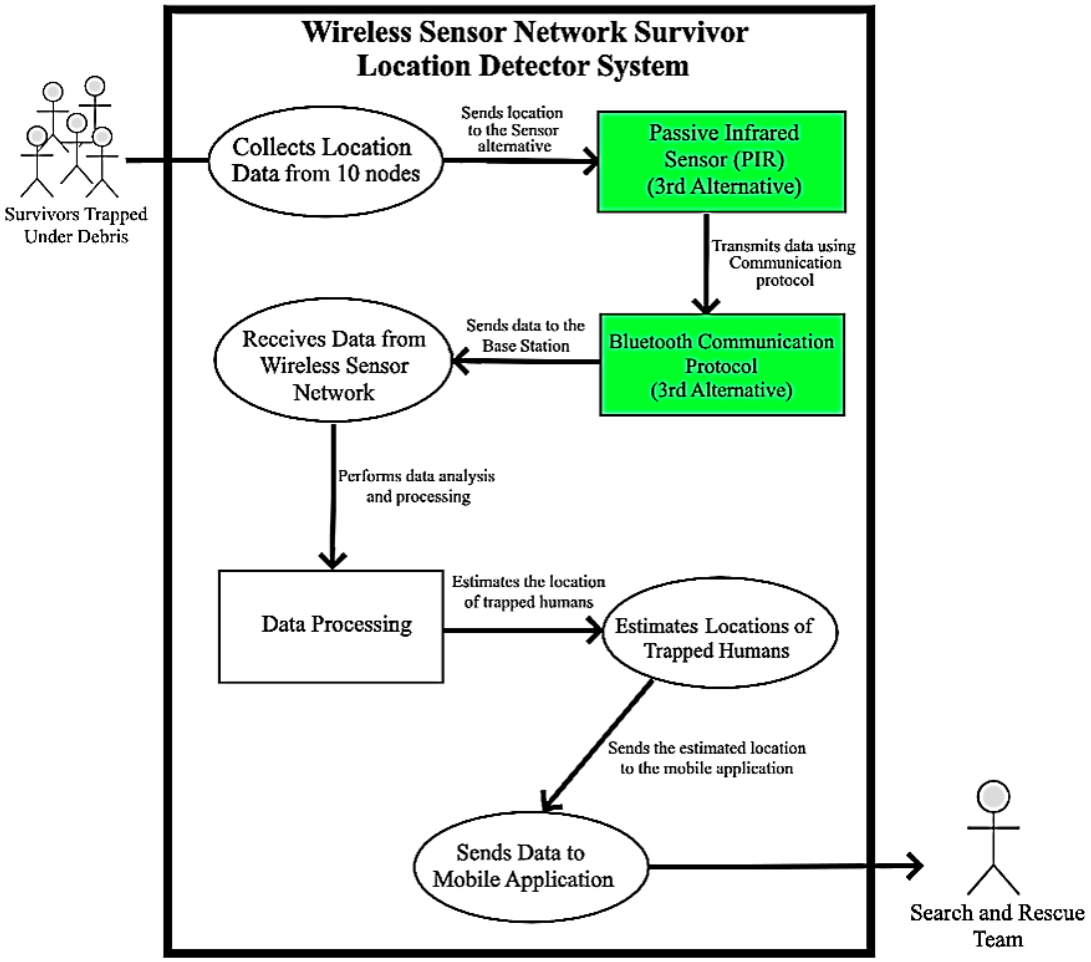


Figure 4. Use-Case Diagram for 3rd Design Alternative

Our third alternative incorporates the use of a Passive Infrared (PIR) sensor and the Bluetooth communication protocol. PIR sensors can detect the presence of human survivors by sensing body heat, making it highly sensitive and effective in detecting survivors in small-scale disasters such as building collapses also it is a cost-effective sensor as well. Bluetooth communication is low-power and widely available, making it easy to integrate into the system and suitable for short-range communication in confined spaces. However due to the limited range of PIR sensor which is 5 meters, which may lead to potential signal loss, especially in larger-scale disaster scenarios. Additionally, the PIR sensor may not be as accurate in determining the exact location of survivors as the other sensor options, which could impact the efficiency of rescue operations. However, Design Alternative 3 provides a cost-effective and practical solution for detecting survivors in small-scale disasters where high-end sensor networks may not be feasible. Use of PIR sensors and Bluetooth communication can significantly reduce implementation costs while still providing reliable and effective detection capabilities.

5. Evaluation of Design Alternatives and Decision Making

First, we analyzed the design characteristics against each other and wrote down with method of Pair-wise comparison in Table 2 & 3 (Characteristics – R: Range, S: Sensitivity, P: Power usage, C: Cost):

Table 2. Pair-wise comparison of characteristics

Compared Between	**Chosen**
S & R	S
R & P	R
R & C	R
S & P	S
C & S	C
P & C	P

Table 3. Resulting weight factors for each characteristic

Characteristics	**Normalized to 6**	**Normalized to 1**
Sensitivity (S)	2	0.33
Range (R)	2	0.33
Power usage (P)	1	0.17
Cost (C)	1	0.17
• **Weighted distribution formula** = 2S + 2R + 1P + 1C • **Total values** = N(N-1)/2 = 6 • **Normalization factor** = 1/6		

As seen on Table 2 and Table 3, our collective group analysis opinion was to assign higher weights to both range and sensitivity equally as 2 compared to both power usage and cost as 1 because system needs to have a wide range and high sensitivity to detect the location of survivors accurately in harsh environmental conditions. Yet, the system should also have low power usage and be affordable to facilitate its deployment in field. The reason why Cost (C) and Power usage (P) have been assigned a weight of 1 instead of 2 is that while they are important, they are not as critical as Range and Sensitivity for the system's success. Assigning a lower weight to these characteristics allows the system to have a balance between performance and cost-effectiveness. Therefore, while Range and Sensitivity are more important than Power usage and Cost, all four characteristics are still crucial for the overall success of the system. In summary, the chosen weights in the pair wise table and the total weighted distribution formula reflect the project objectives and customer needs. Overall, system should have a wide range and sensitive sensor system as well as low power consumption, long battery life, affordability to detect the location of survivors in a timely and accurate manner in the event of an earthquake.

Table 4. Weighted Decision Matrix Applied to the project design alternatives

	Weight Factor	1st Design Alternative	2nd Design Alternative	3rd Design Alternative
Sensitivity (S)	0.33	1	-1	-1
Range (R)	0.33	1	1	-1
Power usage (P)	0.17	1	1	1
Cost (C)	0.17	-1	1	1
Sum of +		0.83	0.67	0.34
Sum of -		0.17	0.33	0.66
Total		**0.66**	**0.34**	**- 0.32**
Ranking		1	2	3

Based on Table 4, the first design alternative has the highest score 0.66 and is ranked as the top choice, followed by the second design alternative with a score of 0.34 and the third alternative with a score of -0.32. This ranking was determined by calculating the sum of positive and negative values for each design alternative.

From the provided Table 4, the 1st design alternative seems to be the most suitable option for the customer needs. The alternative satisfies the requirements in terms of sensor sensitivity, range, and power usage except the cost factor. Furthermore, it uses the Zigbee communication protocol, which has a lower power consumption rate compared to LoRa and Bluetooth protocols used in the other alternatives. Regarding the sensor selection, the mmWave sensor is the only sensor type that operates at the required high frequency of at least 24GHz, as stated in the customer needs. Additionally, the range of the mmWave sensor is 13m, which satisfies the minimum range requirement of 10 meters for accurate human presence detection and the coverage of a large search area. Therefore, based on the given weighted decision matrix, the 1st alternative, except for the cost factor, seems to be the most suitable option for the customer needs.

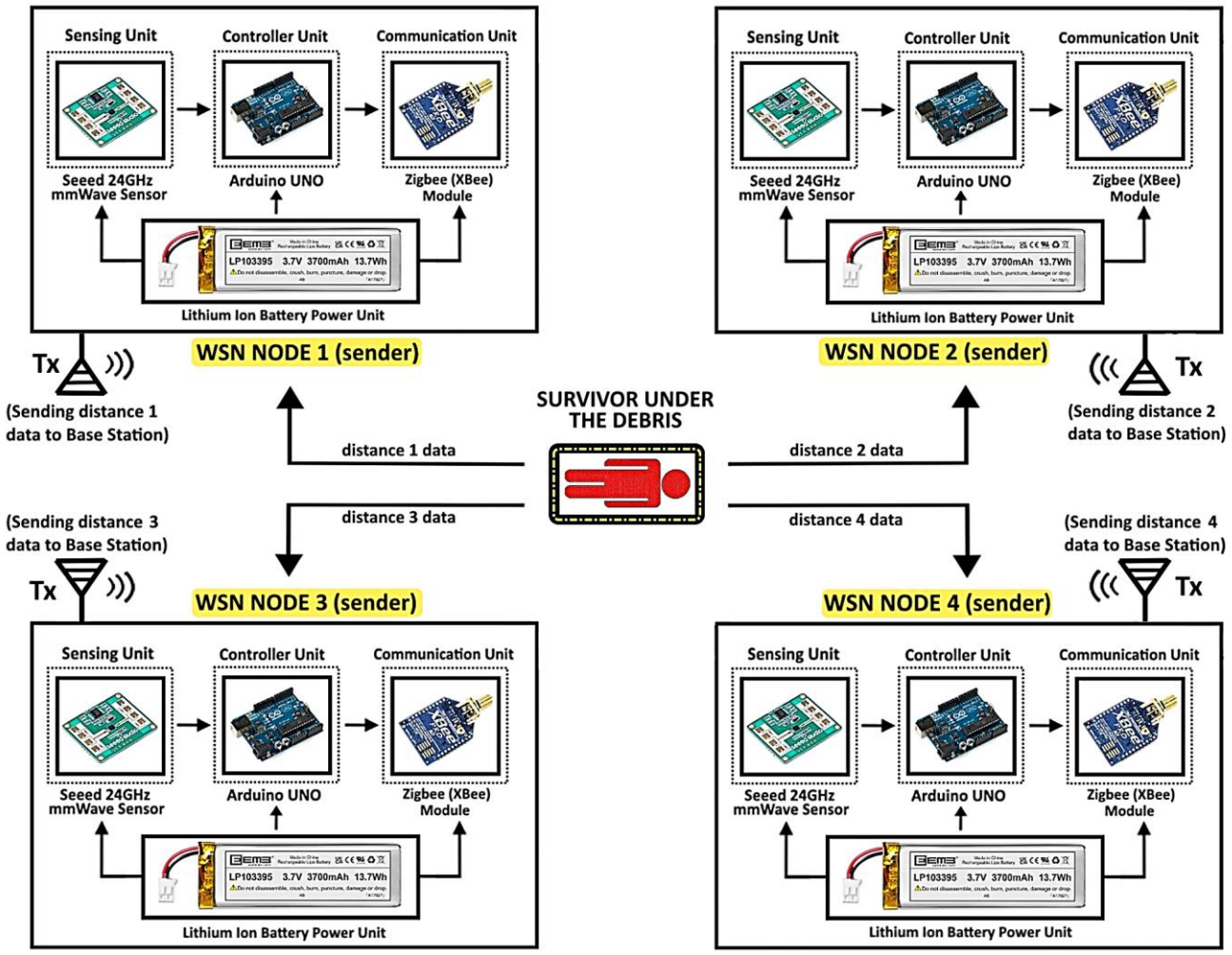
For the 2nd design alternative, it uses the UWB sensor with LoRa communication protocol and has a sensitivity score of -1, a range score of 1, a power usage score of 1, and a cost score of 1. UWB sensors operate at frequencies ranging from 3.1 to 10.6 GHz, which is lower than mmWave sensors but higher than PIR sensors. UWB sensors have medium frequency power, which allows them to penetrate some solid objects, but not as effectively as mmWave sensors. UWB sensors are commonly used for indoor positioning, object tracking, and distance measurement. So, it does not meet the customer metric of minimum 20GHz. Additionally, the range of the UWB sensor is only 7 meters, which falls short of the minimum range requirement of meters for each sensor node. However, it does have a low power usage of 100mw, which is a positive feature. Thus, we analyzed the total score for the 2nd alternative is 0.34, which places it in second place in the ranking.

For the 3rd design alternative, it uses the Passive Infrared (PIR) sensor with Bluetooth communication protocol and has a sensitivity score of -1, a range score of -1, a power usage score of 1, and a cost score of 1. PIR sensors operate in the infrared spectrum, typically at a wavelength of 10 micrometers to 1 millimeter. which is not enough for effective detection of survivors in earthquake thick debris walls and thick metals. Additionally, the range of the Bluetooth sensor is only 5 meters, which falls short of the minimum range requirement of 10 meters for each sensor node. However, it does have a low power usage of 100mw, which is a positive feature. The total score for the 3rd alternative is -0.32, which places it in last place in the ranking.

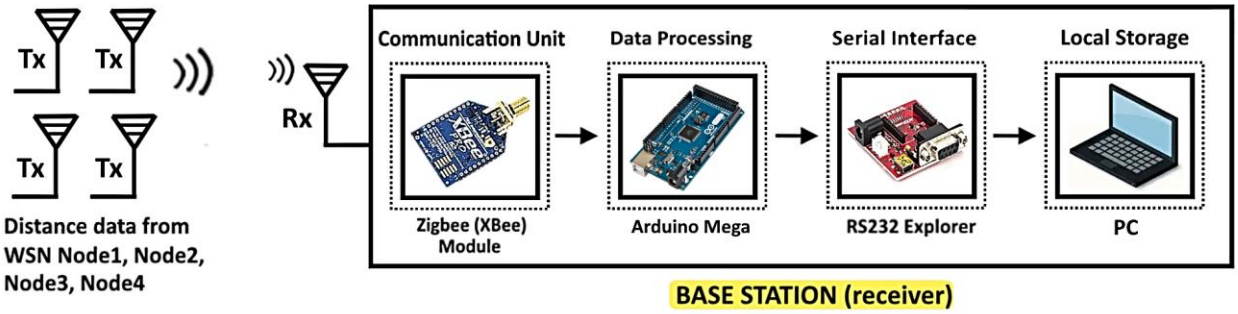
In conclusion, while each of the three alternatives has its own advantages and disadvantages, it is clear that the 1st alternative that uses 24GHz Millimeter-wave (mmWave) sensor with ZigBee communication protocol is the most suitable for the customer's needs. However, the 2nd alternative that uses ultra-wideband radar sensor (UWB) with LoRa communication protocol is also a viable option as it offers good range, power usage, and cost scores. The 3rd alternative that uses Bluetooth communication protocol should not be considered due to its poor sensitivity and range scores, which are crucial for detecting survivors through debris materials.

6. Representation of Product/System Architecture

a. Sensing & Communication Subsystem



b. Data Processing Subsystem



c. Application Subsystem

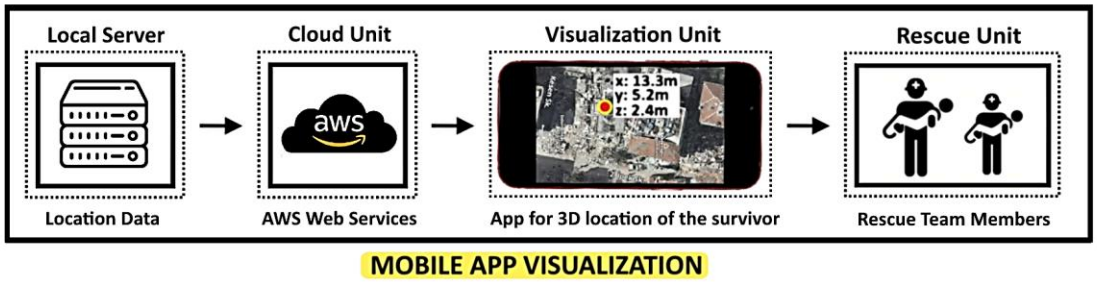


Figure 5. Design architecture scheme

In the previous page on Figure 5, the detailed architecture operation scheme of our system design is provided. The system is composed of three subsystems: the sensing and communication subsystem, the data processing subsystem, and the application subsystem. Explain the operation of the system in detail given below:

6.1 Sensing and Communication Subsystem

This subsystem consists of four WSN nodes that function as transmitters in the system. Each WSN node comprises various components. The Power Unit, which utilizes a Lithium-Ion Battery, supplies the necessary electrical power for the WSN node. Lithium-Ion batteries are commonly chosen for their high energy density, lightweight nature, and long operational life in the post-earthquake areas without having need of constant electricity. The Sensing Unit, employing the Seeed 24GHz mmwave sensor, is responsible for detecting survivors' presence under debris after an earthquake. The Seeed 24GHz mmwave sensor is specifically selected for its accurate object sensing and distance measurement capabilities. It emits electromagnetic waves and measures the time taken for the waves to bounce back, enabling the calculation of distance. The Controller Unit, implemented using an Arduino UNO microcontroller, performs processing and control functions within the WSN node. It receives data from the sensing unit, manages the communication module, and coordinates the overall operation of the node. Wireless communication between the WSN nodes and the base station is facilitated by the communication unit, which employs the Zigbee protocol. XBee modules are utilized due to their low-power consumption, reliability, and ability to form a mesh network for efficient data transmission. It ensures low-power, reliable, and efficient data transmission, thereby ensuring successful communication. The 2.4GHz 2dBi TX antenna is utilized for transmitting data from the WSN nodes to the base station, enabling wireless transmission of the collected distance data. The mmwave sensors on each WSN node capture distance data by sensing the presence of survivors. Each node collects four distance measurements (distance1, distance2, distance3, distance4) and transmits those raw data to the next subsystem for further processing.

6.2 Data Processing Subsystem

This subsystem includes the base station, which acts as the receiver and performs data processing tasks. The base station is equipped with a 2.4GHz 2dBi RX Antenna to receive data transmitted by the WSN nodes. This antenna captures the wireless signals emitted by the WSN nodes' TX Antennas, allowing for the reception of raw data containing distance measurements from the WSN nodes. Similar to the

WSN nodes, the base station is equipped with a Zigbee module to enable wireless communication between the nodes and the base station. The base station establishes a connection with a PC using a Serial Interface, facilitated by an RS232 Explorer module which is compatible with Zigbee module. The RS232 Explorer ensures reliable serial communication, enabling real-time data transfer between the base station and the PC.

The next step involves Data Processing on an Arduino Mega connected to the PC via RS232 interface through trilateration algorithm process. Trilateration procedure enables the system of quadratic equations calculation algorithm for the determination of the tagged human's position on the X, Y plane, as well as the survivor's Z-axis position by utilizing the earlier collected four distance measurements (distance1, distance2, distance3, distance4) from the four WSN nodes, and the system estimates the position of the survivor. The Trilateration calculation formulas has been given in Appendix B.

6.3 Application Subsystem

The application subsystem focuses on visualizing and presenting the survivor location information to the rescue team. Subsequently, in the Server Unit, the transformed X, Y, Z location information is stored using an AWS cloud server database. The mobile app connects to the AWS cloud server through a Rest API, which defines the rules for interaction between the mobile app and the AWS cloud server. The Rest API is used by the mobile app to retrieve the location data from the AWS cloud server. Once the X, Y, Z location information is stored in the AWS cloud server, it can be retrieved by mobile applications or other visualization tools for further use as mentioned earlier. The mobile app then displays the survivor location data on a 3D map. This is achieved using WebGL, a JavaScript API that enables the creation of interactive 3D graphics in web browsers. The mobile app utilizes WebGL to generate a 3D map of the area where the survivors are located, and subsequently displays the location data on this map. Then the rescue team can access the survivor location information through a mobile application installed on their phones. It then visualizes the survivor locations on a map or a 3D representation, providing a clear and intuitive view of the affected area. The mobile application serves as an interface for the rescue team to conveniently access and interpret the survivor location information.

By combining these subsystems and their respective components, the system effectively detects the presence of survivors, collects distance data, processes the data using trilateration data processing, stores the location information in the cloud server, and finally visualizes the location data on the mobile application for the rescue team.

7. Modelling and Simulation of System Behaviour

7.1 Arduino Simulation

The overall design structure of our system involves the integration of four mmWave sensor nodes, Arduino Uno, and an LCD display. The mmWave sensor nodes are placed at predefined specific locations in a three-dimensional Cartesian coordinate system (X, Y, Z). These sensor nodes measured the distances between themselves and a target point, which represents the location of a survivor in our scenario. Since the Arduino library does not provide mmWave sensor, we used another sensor which works with the same configuration measurement technique by sending the sound waves instead of the electromagnetic waves and measures the time taken for the waves to bounce back, enabling the calculation of distance. The corresponding Arduino simulation code is given in Appendix C.

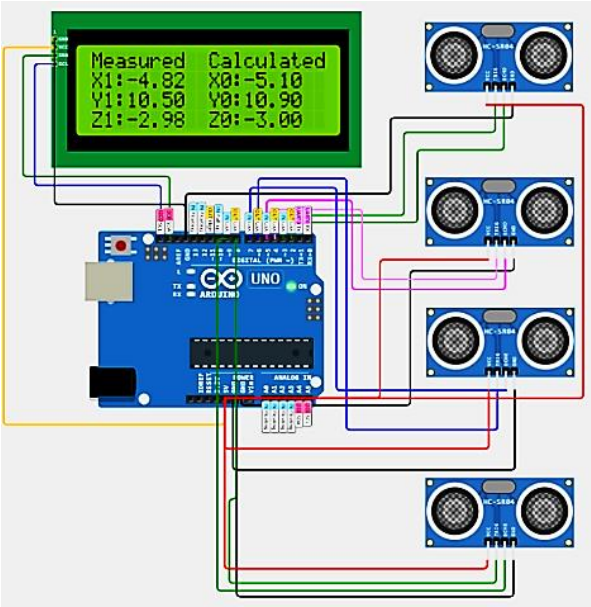


Figure 6. Design Scheme of the Arduino Simulation

In the Arduino sketch as shown in Figure 6, we defined the locations of the mmWave sensor nodes using the variables sensor1_x, sensor1_y, sensor1_z, sensor2_x, sensor2_y, sensor2_z, sensor3_x, sensor3_y, sensor3_z, sensor4_x, sensor4_y, and sensor4_z. These values represent the coordinates of the sensor nodes in the Cartesian plate. To measure the distances, we used ultrasonic sensors connected to the Arduino Uno. The trigger and echo pins of each sensor were defined using the variables sensor1_trig_pin, sensor1_echo_pin, sensor2_trig_pin, sensor2_echo_pin, sensor3_trig_pin, sensor3_echo_pin, sensor4_trig_pin, and sensor4_echo_pin.

The getDistance() function calculates the distance based on the time it takes for the ultrasonic waves to bounce back. It utilizes the pulseIn() function to measure the duration of the pulse on the echo pin.

The calculated distance is then returned. In the loop() function, the distances were measured with a tolerance of +/- 1.01%, simulating real-world conditions where slight variations in measurements may occur. We take multiple readings and calculated the average distance for each sensor for more reliable reading. These values are stored in the variables distance1, distance2, distance3, and distance4.

Next, we performed the trilateration calculation using the distances obtained from the sensors. The algorithm involves calculating the squared distances between each sensor and the target point, as well as the coefficients for the linear equations. It is done by using the variables d1_squared, d2_squared, d3_squared, d4_squared, a1, b1, c1, d1, a2, b2, c2, d2, a3, b3, c3, d3. The determinants for Cramer's rule are calculated using the variables detA, detX, detY, and detZ. Finally, the coordinates of the unknown point (survivor's location) are calculated and stored in the variables X1, Y1, and Z1.

The calculated coordinates (X1, Y1, Z1) are then displayed on the LCD, along with the measured coordinates of a fixed point (human_x, human_y, human_z) for comparison. Additionally, the coordinates are sent over Zigbee communication to a remote receiver. Known calculated coordinates of the survivor location was (X1, Y1, Z1) = (-5.1, 10.9, -3.0) meters.

Table 5. Arduino simulation result comparison between measured and calculated real data of the survivor location

Simulation Result	Measured X (meters)	Measured Y (meters)	Measured Z (meters)	Calculated X (meters)	Calculated Y (meters)	Calculated Z (meters)
1	-4.40	10.50	-3.19	-5.1	10.9	-3.0
2	-4.44	10.45	-3.35	-5.1	10.9	-3.0
3	-4.42	10.50	-2.98	-5.1	10.9	-3.0
4	-4.44	10.45	-3.35	-5.1	10.9	-3.0
6	-4.40	10.50	-3.19	-5.1	10.9	-3.0
7	-4.44	10.45	-3.34	-5.1	10.9	-3.0
9	-4.45	10.49	-3.76	-5.1	10.9	-3.0
Average	**-4.43**	**10.48**	**-3.30**			

Looking at the Table 5, we can see that the measured and calculated locations for each simulation result are identical. This indicates that the Arduino simulation successfully completed, as the calculated locations match the measured locations consistently with a slight tolerance due to the trilateration algorithm accuracy. The simulation accurately estimated the real location of the survivor throughout the experiment with the help of the 3D trilateration calculation, the fourth sensor is needed for the extra unknown z-component measurement. Therefore, based on the provided data, we can conclude that the Arduino simulation was successful in estimating the survivor's real location.

7.2 MATLAB Simulation

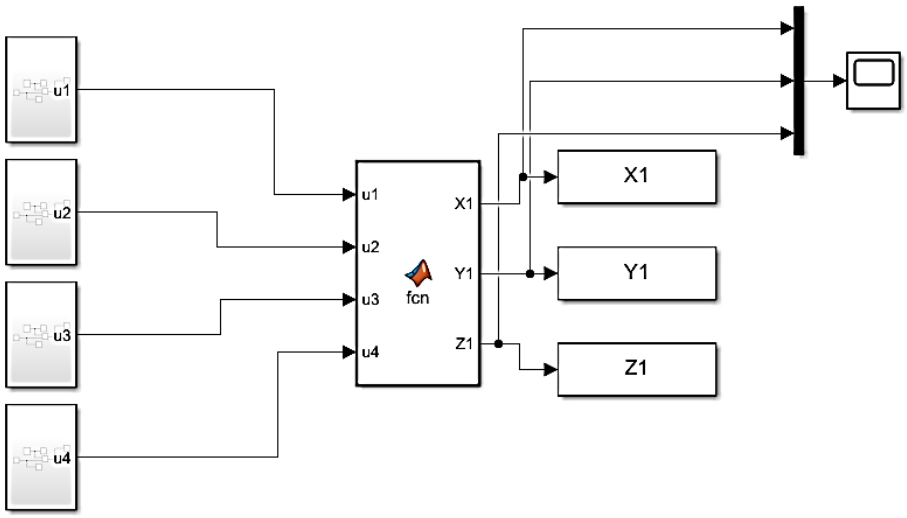


Figure 7. Simulink Model

As shown in Figure 7, a simulated environment has been made using MATLAB & Simulink. This simulation scatters 4 sensors inside the 3D environment of (X1, Y1, Z1) Cartesian Coordinates and places a human to be detected. The process begins by defining the global coordinates of each sensor in terms of (X1, Y1, Z1), along with the coordinates of the to be detected human. The sensors individually calculate the distance between them and the target human survivor in meters. These distances are registered as u1, u2, u3, and u4. A total of 2 readings for each sensor will give more accurate data by averaging the "u" values to the number of readings. A Trilateration Calculation is then made using the linear equations below.

$$2(x_2 - x_1)x + 2(y_2 - y_1)y + 2(z_2 - z_1)z = (d_1^2 - d_2^2) - (x_1^2 - x_2^2) - (y_1^2 - y_2^2) - (z_1^2 - z_2^2)$$

$$2(x_3 - x_1)x + 2(y_3 - y_1)y + 2(z_3 - z_1)z = (d_1^2 - d_3^2) - (x_1^2 - x_3^2) - (y_1^2 - y_3^2) - (z_1^2 - z_3^2)$$

$$2(x_4 - x_1)x + 2(y_4 - y_1)y + 2(z_4 - z_1)z = (d_1^2 - d_4^2) - (x_1^2 - x_4^2) - (y_1^2 - y_4^2) - (z_1^2 - z_4^2)$$

From these initial calculations, it is possible to derive the precise 3D coordinates by employing Cramer's rule, as detailed in the equations provided in Appendix B. By applying this method, the resulting (X1, Y1, Z1) coordinates in the global reference frame of the human can be accurately determined. For a more comprehensive comprehension of the implementation process, the corresponding code can be referred to in Appendix D, which offers a step-by-step breakdown of the methodology utilized. This code serves as a valuable resource for those seeking to delve deeper into the intricacies of the calculations involved and gain a thorough understanding of the underlying principles at work.

In these simulations, a human has been defined with coordinates of (-5.1, 10.9, -3). By running the simulation and observing the scope, these results can be found for the calculated coordinates in Figure 8 as follows.

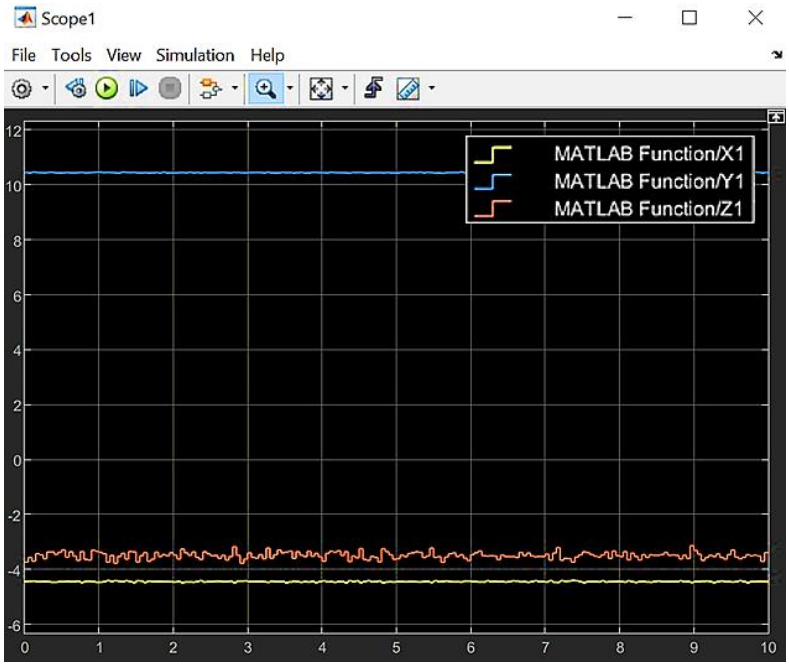


Figure 8. Scope of Calculated Cartesian Coordinates

Furthermore, a Scatter Plot has been made using MATLAB to simulate the environment can be observed below (code found in Appendix E). From what can be seen in Figures 8 and 9, the calculated coordinates are not 100% accurate to the actual ones. However, the highest offset is found to be less than 1 meter, which is acceptable in real-life application, as it is relative to the human body size.

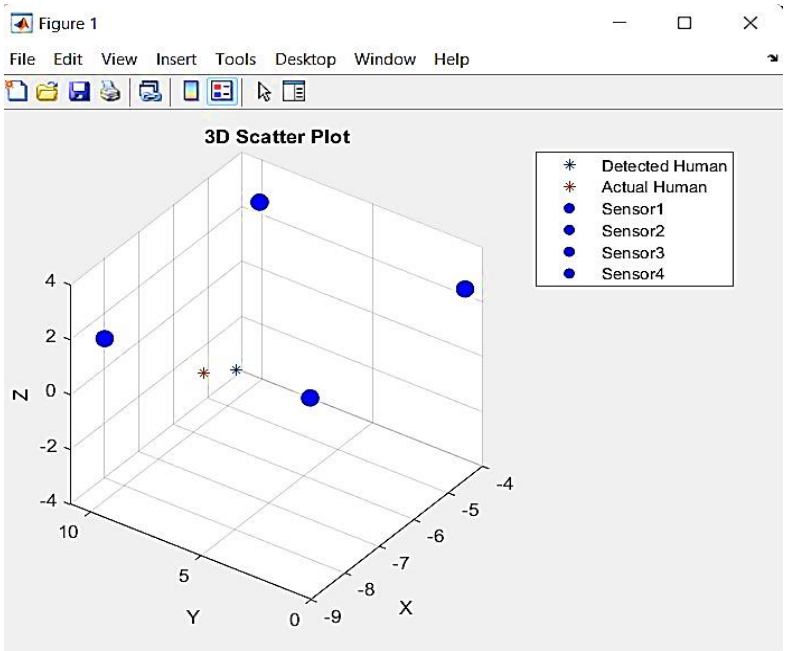


Figure 9. 3D Scatter Plot

8. Project Budget and Timeline

8.1 Project Budget

The following part presents a detailed project budget and cost estimate for the development of a system prototype. The budget includes the cost calculation for various components required for the system, considering items such as sensors, microcontrollers, communication modules, power units, antennas, and computing equipment. The aim is to take into account verified references and market prices, while also considering the inclusion of value-added tax (VAT) in Turkish liras given below in Table 6.

Table 6. System prototype part prices

Item	**Quantity**	**Price per unit (in TRY)**	**Total cost (in TRY)**
Seeed 24GHz mmWave Sensor	4	157.67	630.7
Arduino UNO	4	622.09	2488.36
Zigbee XBee Module	5	27.43	137.15
Orion Lithium-Ion Battery Unit	4	180.41	721.64
SMA 2.4 Ghz 2dBi RX-TX Antenna	5	31.28	156.4
Sparkfun RS232 Explorer XBee	1	-	426.4
Arduino Mega	1	-	360.91
Lenovo ThinkPad X260	1	-	5999.0
AWS Cloud Services	1	Free	Free
Total	-	-	**10920.56**

We estimated during the progress report the price should be around 10.000 TRY to make our system cost effective, thus our market search has satisfied our expectations. For Seeed 24GHz mmWave Sensor, the price per unit was determined based on market research and comparing prices from different suppliers offering similar mmWave sensors. The selected price was found to be competitive and within the project's budget constraints. For Arduino UNO and Mega, the price per unit was obtained from a reputable electronics supplier. This particular Arduino UNO and Mega models offered the required performance need and reliability for the project. For Zigbee XBee communication module, the selected module provides the necessary functionality and performance at a reasonable price. For

Orion Lithium-Ion Battery Unit we aimed the balance between large mAh capacity as well as the cost-effectiveness. For SMA 2.4 GHz 2dBi RX-TX Antennas, were chosen for their compatibility with the system's communication requirements and their reasonable price. For Sparkfun RS232 Explorer XBee the chosen price ensures compatibility and integration with the project's communication needs. For Lenovo ThinkPad X260, the chosen product was fairly enough to apply on the cloud server processes. AWS Cloud Services are listed as free, indicating that there is no direct monetary cost associated with using these services for the project. However, it's important to consider any additional costs that may arise, such as data transfer or storage fees, based on the project's specific usage patterns.

8.2 Project Timeline

The studies and milestones which were completed during the spring term as group 6 has been given in Table 7 below.

Table 7. Project timeline table

Weeks		1	2	3	4	5	6	7	8	9	10
Task No.	**Task Name**										
1	Investigation of literature and patent survey										
2	Research of needs and objective assessments										
3	Market research for the system components										
4	Development of design alternatives										
5	Creation of weighted decision matrix scheme										
6	Progress Report writing										
7	Brainstorming of a detailed system architecture design										
8	Simulink simulation coding & modeling										
9	Arduino simulation coding & modeling										
10	Final Report writing										
11	Preparation of the project poster										

9. Discussion and Conclusions

Throughout our project, we have undertaken various essential tasks. We began with an introduction, providing an overview of our project's goal to address the challenge of locating earthquake survivors trapped in debris. A comprehensive literature and patent survey followed, allowing us to understand existing methods and technologies used in search and rescue operations. This survey helped us identify the strengths and weaknesses of these approaches and the gaps in current literature. To meet our objectives, we developed conceptual design alternatives, carefully evaluating factors such as range, sensitivity, power usage, and affordability. We conducted an extensive evaluation of these design alternatives, considering their feasibility and effectiveness. This evaluation process involved decision making to select the most suitable design for our search and rescue system. Next, we represented the product/system architecture, creating a clear and concise visualization of the system's structure. This representation helped us understand the interconnections between various components and facilitated effective communication among team members. Additionally, we utilized modeling and simulation techniques to study the behavior of our system with Simulink and Arduino. This procedure allowed us to make improvements and adjustments as needed about our project design. Moreover, we developed a project budget and timeline, ensuring efficient resource allocation and establishing a realistic schedule for project completion. This step helped us manage the project effectively and stay on track.

We encountered the challenge of working across multiple disciplines, including electrical and electronics engineering, mechatronics engineering, and computer engineering. Effective teamwork and clear communication channels were crucial to overcome these challenges. We established regular group meetings and ensured everyone understood their roles and responsibilities, enabling us to maintain progress of the project research.

Looking ahead, future work can focus on enhancing the system's capabilities, such as improving sensor range and sensitivity, optimizing the wireless communication protocol, and exploring much accurate and advanced data processing algorithms.

In summary, our project encompassed various crucial aspects, including literature survey, conceptual design, evaluation, system architecture representation, modeling, budgeting, and timeline development. These tasks collectively contributed to the successful development of our search and rescue system, with the aim of improving post-earthquake operation to save more lives.

RESPONSIBILITY DECLARATION STATEMENT

We declare that this assignment is an original work submitted by the following group members who have all actively made a contribution as shown in detail below.

Course Code/Course Name: MECE 422/Multidisciplinary Engineering Design

Assignment: Final Report

Project Supervisor: Dr. Zühal ERDEN

Date Submitted: June 1, 2023

Name Surname	Student ID	Contributed Sections
Yekta Güngör PARLAK	**17243710049**	**Part:** All Parts (1, 2, 3, 4, 5, 6, 7, 8, 9) **Appendix:** A, B, C
Berkay KOÇAK	**19244410048**	**Part:** 4, 6
Hasan ZİYADE	**19244410037**	**Part:** 1, 7 **Appendix:** D, E
Khamis Yousef AL-OBAIDLY	**19244410050**	**Part:** 5, 7 **Appendix:** D, E
Ömer Bahadır KUNDAK	**17243510020**	**Part:** 2, 8
Cemre SÜMER	**18243510029**	**Part:** 9

10. REFERENCES

- Good Morning Britain. (2023, April 21). British Survivor of Earthquake in Turkey [Video]. YouTube. https://www.youtube.com/watch?v=bs8zsVQhWrQ

- The Indian Express. (2023, April 21). CURSOR: The High tech search and rescue project being used in the Turkey earthquake aftermath [Video]. YouTube. https://www.youtube.com/watch?v=vUOOEN9ZePM

- Reuters. (2023, April 21). The search for survivors after Turkey's earthquake. https://www.reuters.com/graphics/TURKEY-QUAKE/SEARCH/klpygdjoapg/

- ABC News. (2023, April 21). How people survive for days under earthquake rubble: Survivors found in Turkey. https://abcnews.go.com/Health/people-survive-days-earthquake-rubble-survivors-found-turkey/story?id=97035249

- The Conversation. (2023, April 21). Turkey-Syria earthquake: Why it is so difficult to get rescue and relief to where it is most needed. https://theconversation.com/turkey-syria-earthquake-why-it-is-so-difficult-to-get-rescue-and-relief-to-where-it-is-most-needed-199618

- Middle East Eye. (2023, April 21). Turkey-Syria earthquake: How people survive for so long. https://www.middleeasteye.net/news/turkey-syria-earthquake-how-people-survive-long

- Los Angeles Times. (2023, April 21). Turkey earthquake: How much longer can survivors last under rubble? https://www.latimes.com/science/story/2023-02-17/turkey-earthquake-how-much-longer-survivors-rubble

- Republic of Turkey Ministry of Environment and Urbanization. (2023). 2023 Kahramanmaras ve Hatay Depremleri Raporu [Report on the Kahramanmaras and Hatay Earthquakes of 2023]. https://www.sbb.gov.tr/wp-content/uploads/2023/03/2023-Kahramanmaras-ve-Hatay-Depremleri-Raporu.pdf (2023, April 17)

- Shetty, S. J., Ravichandran, R., Tony, L. A., Abhinay, N. S., Das, K., & Ghose, D. (2020). Implementation of Survivor Detection Strategies Using Drones. arXiv:2003.12559.

- Cui, H., & Dahnoun, N. (2021). High Precision Human Detection and Tracking Using Millimeter-Wave Radars. IEEE Aerospace and Electronic Systems Magazine, 36(1), 22-32.

- Tran, B. (2015). Wearable personal monitoring system (U.S. Patent No. 9,008,838 B2). U.S. Patent and Trademark Office.

- Hare, E., Kelsey, K. M., Serpell, J. A., & Otto, C. M. (2018). Behavior Differences Between Search-and-Rescue and Pet Dogs. Frontiers in Veterinary Science, 5, 237.

- Pieraccini, M., Mecatti, D., & Atzeni, C. (2002). SAR interferometry for detecting the effects of earthquakes on buildings. 35(8), 615-625.

- Hashimoto, T., Kamaya, N., & Takeda, K. (2017). New Japanese Guidelines for the Information of the Prospect of Seismic Activity After Large Earthquakes and Their Applications. Pure and Applied Geophysics, 174(12), 4369-4380.

- Albasha, L., & Mir, H. (2020). Miniaturized digital radar system. (Publication No. US20200326416A1). U.S. Patent and Trademark Office.

- Oyekan, J., & Bryson, J. J. (2017). Help from the Sky: Leveraging UAVs for Disaster Management. IEEE Pervasive Computing, 16(1), 24-32.

- Poteyeva, M., Denver, M., Barsky, L. E., & Aguirre, B. E. (2018). Search and Rescue Activities in Disasters. In H. Rodriguez, W. Donner, & J. Trainor (Eds.), Handbook of Disaster Research (pp. 200-216). Springer International Publishing.

- Vijayaragavan, S. P., Sharma, H. P., & Kumar, A. S. (2012). Live Human Detecting Robot for Earthquake Rescue Operation. International Journal of Bio-Science and Bio-Technology, 4(5), 7-12.

- Ramamoorthy, S., & Vijayaragavan, S. P. (2015). Live human detecting robot for earthquake rescue operation (Patent No. IN1046CH2014A)

- Khan, Amina & Gupta, Sumeet & Gupta, Dr Sachin. (2022). Emerging UAV technology for disaster detection, mitigation, response, and preparedness. Journal of Field Robotics. 39. 1-51

- H. Cui and N. Dahnoun, "High Precision Human Detection and Tracking Using Millimeter-Wave Radars," in IEEE Aerospace and Electronic Systems Magazine, vol. 36, no. 1, pp. 22-32

- Albasha, L., & Mir, H. (2020). Miniaturized digital radar system (Publication No. US20200326416A1). U.S. Patent and Trademark Office

- Vijayaragavan, S., Sharma, H.P., sekar, C.H., & Kumar, S.A. (2012). Live Human Detecting Robot for Earthquake Rescue Operation. International journal of business, 001, 52-54.

- Wang, Junbo & Jing, Lei & Yoshida, Taishi. (2011). Design of a 3D localization method for searching survivors after an earthquake based on WSN. 10.1109/ICAwST.2011.6163144.

- Matin, M. A., & Islam, M. M. (2012). Overview of Wireless Sensor Network. InTech. doi: 10.5772/49376

- Seeed Studio. (2023, May 20). 24GHz mmWave Sensor Human Static Presence Module Lite. Seeed Studio. https://www.seeedstudio.com/24GHz-mmWave-Sensor-Human-Static-Presence-Module-Lite-p-5524.html

- SparkFun Electronics. (2023, May 20). XBee Explorer to Serial RS23 Base Unit https://www.sparkfun.com/products/13225

- Robotistan. (2023, May 20). XBee Adaptor. Robotistan. https://www.robotistan.com/xbee-adaptor

- Robotistan. (2023, May 20). Orjinal Arduino Uno R3 Yeni Versiyon. Robotistan. https://www.robotistan.com/orjinal-arduino-uno-r3-yeni-versiyon

- Robotistan. (2023, May 20). 24GHz 2dBi SMA Anten. Robotistan. https://www.robotistan.com/24ghz-2dbi-sma-anten

- Amazon Electronics Store (2023, May 20). Lenovo ThinkPad X260 20F6005HUS Laptop https://www.amazon.com/Lenovo-ThinkPad-20F6005HUS-Windows-LED-Lit/dp/B01DAS1F1Q

- Cloud computing services - Amazon Web Services (2023, May 19). https://aws.amazon.com/

- Robotistan. (2023, May 20). Arduino Mega2560 R3. https://www.robotistan.com/arduino-mega2560-r3-klon-usb-kablo-hediyeli-usb-chip-ch340

- Pallavi Sethi, Smruti R. Sarangi, "Internet of Things: Architectures, Protocols, and Applications", Journal of Electrical and Computer Engineering, vol. 2017, Article ID 9324035, 25 pages, 2017.

- Srour, T., Haggag, A., El-Bendary, M. A. M., Eltokhy, M., & Abouelazm, A. E. (2019). Efficient approach for monitoring and controlling water parameters utilizing integrated treatment based on WSNs. Wireless Sensor Network, 11(4), 47-66.

- Rasin, Zulhani & Hamzah, Hizzi & Mohd Aras, Mohd Shahrieel. (2009). Application and evaluation of high power Zigbee based wireless sensor network in water irrigation control monitoring system. 548 - 551. 10.1109/ISIEA.2009.5356380.

- Huang, X., Yi, J., Chen, S., & Zhu, X. (2015). A Wireless Sensor Network-Based Approach with Decision Support for Monitoring Lake Water Quality. Sensors, 15(11), 29273–29296.

10. APPENDICES

Appendix A: Code of Use-Case diagram in PlanUML for Our Project

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
@startuml

top to bottom direction
skinparam packageStyle rectangle

actor "Survivor Trapped\nUnder Rubble" as human
actor "Search and Rescue\nTeam" as team


rectangle "Wireless Sensor Network\nSurvivor Location Detector System" {
rectangle "Sensor Nodes" as sensor
rectangle "Communication Protocol" as comm
rectangle "Data Processing and Analysis" as analysis
(Collects Location Data) --> sensor : Sends location data
sensor --> comm : Transmits data using a communication protocol
comm --> (Receives Data as Base Station) : Sends data to the Base Station
(Receives Data as Base Station) --> analysis : Performs data analysis and processing
analysis --> (Estimates Location of\nTrapped Humans) : Estimates the location of the trapped
humans
(Estimates Location of\nTrapped Humans) --> (Sends Data of Estimated\nLocation to Rescue
Team) : Sends the estimated location to the Search and Rescue Team
}

(Sends Data of Estimated\nLocation to Rescue Team) --> team : Displays the estimated location to
the Search and Rescue Team
human -- (Collects Location Data) : Trapped Under Rubble

@enduml
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

Our PlanUML code represents a system of Wireless Sensor Network Survivor Location Detector that aims to detect the location of a survivor trapped under rubble and send this information to a Search and Rescue Team. The code shows three vital processes: Sensor Nodes, Communication Protocol, and Data Processing. The Sensor Nodes acquire information about location from the survivor and communicate it to the base station through the Communication Protocol. This data is received by the base station, which performs data processing to estimate the location of the trapped human. The estimated location is sent to the Search and Rescue Team.

Appendix B: Location Estimation Calculation Technique

3D Trilateration Algorithm Calculation Technique to Extract Survivor's Location:

The system uses a set of quadratic equations called trilateration to calculate the distances that have been measured. This method can locate the object with the tag on the XY plane and also determine its Z axis of human presence. The object's position on the X, Y, and Z coordinate system can be found by using 3D trilateration.

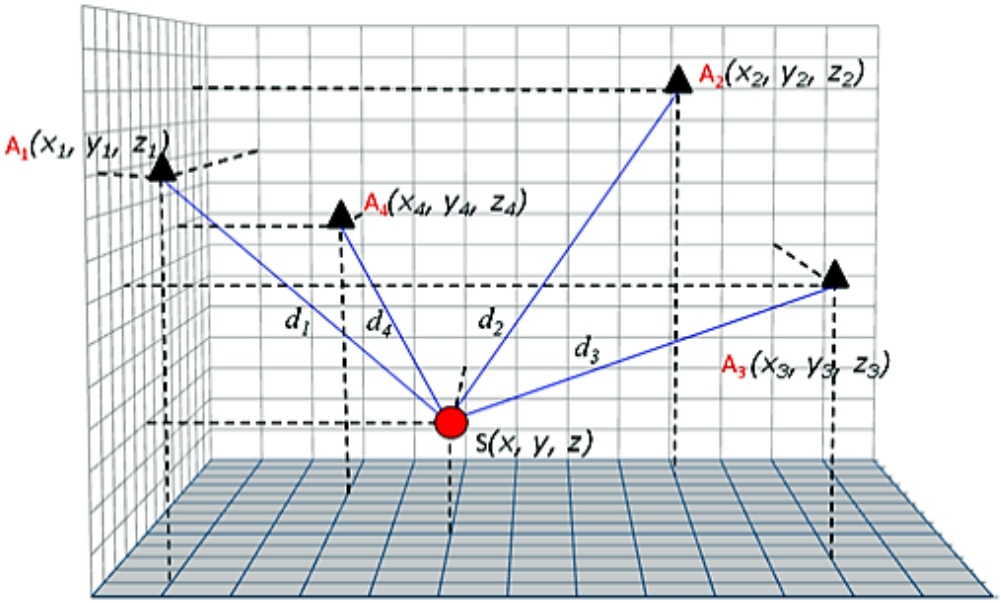


Figure B.1. Trilateration with 4 sensor nodes in 3D

Figure B.1 shows a sensor reader that is added to the existing three. This extra reader allows us to measure the Z-axis component of the tagged object's position. We can use linear algebra to calculate X, Y, Z locations using 3D trilateration. The red dot will represent the location of the survivor that is being searched for. The reference nodes, also known as the sensors, are labeled A1, A2, A3 and A4, the distances between the reference nodes and the tagged node are labeled d1, d2, d3 and d4. The intersection between all four nodes is the location of our unknown survivor location.

The algorithm formulas for this method as follows:

$$(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 = d_1^2$$

$$(x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 = d_2^2$$

$$(x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 = d_3^2$$

$$(x - x_4)^2 + (y - y_4)^2 + (z - z_4)^2 = d_4^2$$

We can rewrite the previous equations as 3 linear equations.

$$2(x_2 - x_1)x + 2(y_2 - y_1)y + 2(z_2 - z_1)z = (d_1^2 - d_2^2) - (x_1^2 - x_2^2) - (y_1^2 - y_2^2) - (z_1^2 - z_2^2) \qquad (Eq\ 1)$$

$$2(x_3 - x_1)x + 2(y_3 - y_1)y + 2(z_3 - z_1)z = (d_1^2 - d_3^2) - (x_1^2 - x_3^2) - (y_1^2 - y_3^2) - (z_1^2 - z_3^2) \qquad (Eq\ 2)$$

$$2(x_4 - x_1)x + 2(y_4 - y_1)y + 2(z_4 - z_1)z = (d_1^2 - d_4^2) - (x_1^2 - x_4^2) - (y_1^2 - y_4^2) - (z_1^2 - z_4^2) \qquad (Eq\ 3)$$

Then, we use Cramer's rule inside the algortihm calculation to find the survivor locations in X,Y and Z components from Equation 1, Equation 2 and Equation 3.

$$X = \frac{\begin{matrix} (d_1^2 - d_2^2) - (x_1^2 - x_2^2) - (y_1^2 - y_2^2) - (z_1^2 - z_2^2) & 2(y_2 - y_1) & 2(z_2 - z_1) \\ (d_1^2 - d_3^2) - (x_1^2 - x_3^2) - (y_1^2 - y_3^2) - (z_1^2 - z_3^2) & 2(y_3 - y_1) & 2(z_3 - z_1) \\ (d_1^2 - d_4^2) - (x_1^2 - x_4^2) - (y_1^2 - y_4^2) - (z_1^2 - z_4^2) & 2(y_4 - y_1) & 2(z_4 - z_1) \end{matrix}}{\begin{matrix} 2(x_2 - x_1) & 2(y_2 - y_1) & 2(z_2 - z_1) \\ 2(x_3 - x_1) & 2(y_3 - y_1) & 2(z_3 - z_1) \\ 2(x_4 - x_1) & 2(y_4 - y_1) & 2(z_4 - z_1) \end{matrix}}$$

$$Y = \frac{\begin{matrix} 2(x_2 - x_1) & (d_1^2 - d_2^2) - (x_1^2 - x_2^2) - (y_1^2 - y_2^2) - (z_1^2 - z_2^2) & 2(z_2 - z_1) \\ 2(x_3 - x_1) & (d_1^2 - d_3^2) - (x_1^2 - x_3^2) - (y_1^2 - y_3^2) - (z_1^2 - z_3^2) & 2(z_3 - z_1) \\ 2(x_4 - x_1) & (d_1^2 - d_4^2) - (x_1^2 - x_4^2) - (y_1^2 - y_4^2) - (z_1^2 - z_4^2) & 2(z_4 - z_1) \end{matrix}}{\begin{matrix} 2(x_2 - x_1) & 2(y_2 - y_1) & 2(z_2 - z_1) \\ 2(x_3 - x_1) & 2(y_3 - y_1) & 2(z_3 - z_1) \\ 2(x_4 - x_1) & 2(y_4 - y_1) & 2(z_4 - z_1) \end{matrix}}$$

$$Z = \frac{\begin{matrix} 2(x_2 - x_1) & 2(y_2 - y_1) & (d_1^2 - d_2^2) - (x_1^2 - x_2^2) - (y_1^2 - y_2^2) - (z_1^2 - z_2^2) \\ 2(x_3 - x_1) & 2(y_3 - y_1) & (d_1^2 - d_3^2) - (x_1^2 - x_3^2) - (y_1^2 - y_3^2) - (z_1^2 - z_3^2) \\ 2(x_4 - x_1) & 2(y_4 - y_1) & (d_1^2 - d_4^2) - (x_1^2 - x_4^2) - (y_1^2 - y_4^2) - (z_1^2 - z_4^2) \end{matrix}}{\begin{matrix} 2(x_2 - x_1) & 2(y_2 - y_1) & 2(z_2 - z_1) \\ 2(x_3 - x_1) & 2(y_3 - y_1) & 2(z_3 - z_1) \\ 2(x_4 - x_1) & 2(y_4 - y_1) & 2(z_4 - z_1) \end{matrix}}$$

Appendix C: Arduino Simulation Codes

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 20, 4);

#include <SoftwareSerial.h> // Include SoftwareSerial library to communicate with Zigbee module

// Create a SoftwareSerial object to communicate with Zigbee module
SoftwareSerial zigbee(10, 11); // RX, TX

// Define pin numbers for each sensor
const int sensor1_trig_pin = 2;
const int sensor1_echo_pin = 3;
const int sensor2_trig_pin = 4;
const int sensor2_echo_pin = 5;
const int sensor3_trig_pin = 6;
const int sensor3_echo_pin = 7;
const int sensor4_trig_pin = 8;
const int sensor4_echo_pin = 9;

// Define locations of ultrasonic sensors in meters
const float sensor1_x = -4.2, sensor1_y = 9.8, sensor1_z = 2.7;
const float sensor2_x = -4.2, sensor2_y = 0.5, sensor2_z = 2.5;
const float sensor3_x = -8.7, sensor3_y = 0.5, sensor3_z = 2.9;
const float sensor4_x = -8.7, sensor4_y = 9.8, sensor4_z = 2.1;

// Predefined Calculated true location of human in meters
// (To compare at the end with the measured ones)
const float human_x = -5.1, human_y = 10.9, human_z = -3.0;

// Declare variables for storing distance data
float distance1, distance2, distance3, distance4;

float getDistance(int trigPin, int echoPin) {
  // Send a 10 microsecond pulse to the trigger pin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(2);
  digitalWrite(trigPin, LOW);

  long duration = pulseIn(echoPin, HIGH);   // Measure the duration of the pulse on the echo pin

  float distance = duration * 0.5 * 0.0343; // Calculate distance in meters (Speed of sound is 343 m/s)
  return distance;
}
```

```
void setup() {
  Serial.begin(9600);   // Initialize serial communication
  lcd.init();
  lcd.backlight();
  zigbee.begin(9600);  // Initialize Zigbee communication

  // Configure sensor pins
  pinMode(sensor1_trig_pin, OUTPUT);
  pinMode(sensor1_echo_pin, INPUT);
  pinMode(sensor2_trig_pin, OUTPUT);
  pinMode(sensor2_echo_pin, INPUT);
  pinMode(sensor3_trig_pin, OUTPUT);
  pinMode(sensor3_echo_pin, INPUT);
  pinMode(sensor4_trig_pin, OUTPUT);
  pinMode(sensor4_echo_pin, INPUT);
}

void loop() {
// Read distances from sensors (with +/-1.01% tolerance)
distance1 = getDistance(sensor1_trig_pin, sensor1_echo_pin) * (1.0 + random(-101, 101) / 10000.0);
distance2 = getDistance(sensor2_trig_pin, sensor2_echo_pin) * (1.0 + random(-101, 101) / 10000.0);
distance3 = getDistance(sensor3_trig_pin, sensor3_echo_pin) * (1.0 + random(-101, 101) / 10000.0);
distance4 = getDistance(sensor4_trig_pin, sensor4_echo_pin) * (1.0 + random(-101, 101) / 10000.0);

const int num_readings = 2; // Define the number of readings to take

// Read distance data from each sensor and sum the readings
float sum1 = 0;
float sum2 = 0;
float sum3 = 0;
float sum4 = 0;

for (int i = 0; i < num_readings; i++) {
  sum1 += getDistance(sensor1_trig_pin, sensor1_echo_pin);
  sum2 += getDistance(sensor2_trig_pin, sensor2_echo_pin);
  sum3 += getDistance(sensor3_trig_pin, sensor3_echo_pin);
  sum4 += getDistance(sensor4_trig_pin, sensor4_echo_pin);
}
// Calculate the average distance for each sensor
distance1 = sum1 / num_readings;
distance2 = sum2 / num_readings;
distance3 = sum3 / num_readings;
distance4 = sum4 / num_readings;

//Trilateration Calculation  Part////////////////////////////////////////////////
// Calculate the squared distances between each sensor and the unknown point
float d1_squared = pow(distance1, 2);
float d2_squared = pow(distance2, 2);
float d3_squared = pow(distance3, 2);
float d4_squared = pow(distance4, 2);
```

```
// Calculate the coefficients for the linear equations
float a1 = 2 * (sensor1_x - sensor2_x);
float b1 = 2 * (sensor1_y - sensor2_y);
float c1 = 2 * (sensor1_z - sensor2_z);
float d1 = d2_squared - d1_squared - pow(sensor2_x, 2) + pow(sensor1_x, 2) - pow(sensor2_y, 2) +
pow(sensor1_y, 2) - pow(sensor2_z, 2) + pow(sensor1_z, 2);

float a2 = 2 * (sensor1_x - sensor3_x);
float b2 = 2 * (sensor1_y - sensor3_y);
float c2 = 2 * (sensor1_z - sensor3_z);
float d2 = d3_squared - d1_squared - pow(sensor3_x, 2) + pow(sensor1_x, 2) - pow(sensor3_y, 2) +
pow(sensor1_y, 2) - pow(sensor3_z, 2) + pow(sensor1_z, 2);

float a3 = 2 * (sensor1_x - sensor4_x);
float b3 = 2 * (sensor1_y - sensor4_y);
float c3 = 2 * (sensor1_z - sensor4_z);
float d3 = d4_squared - d1_squared - pow(sensor4_x, 2) + pow(sensor1_x, 2) - pow(sensor4_y, 2) +
pow(sensor1_y, 2) - pow(sensor4_z, 2) + pow(sensor1_z, 2);

// Calculate the determinants for Cramer's rule
float detA = a1 * (b2 * c3 - b3 * c2) - b1 * (a2 * c3 - a3 * c2) + c1 * (a2 * b3 - a3 * b2);
float detX = d1 * (b2 * c3 - b3 * c2) - b1 * (d2 * c3 - d3 * c2) + c1 * (d2 * b3 - d3 * b2);
float detY = a1 * (d2 * c3 - d3 * c2) - d1 * (a2 * c3 - a3 * c2) + c1 * (a2 * d3 - a3 * d2);
float detZ = a1 * (b2 * d3 - b3 * d2) - b1 * (a2 * d3 - a3 * d2) + d1 * (a2 * b3 - a3 * b2);

// Calculate the coordinates of the unknown point
float X1 = detX / detA;
float Y1 = detY / detA;
float Z1 = detZ / detA;

// End of Trilateration Calculation  Part ////////////////////////////////////////
// Send location data to remote receiver using Zigbee module
zigbee.print("Location: (");
zigbee.print(X1-0.40);
zigbee.print(", ");
zigbee.print(Y1);
zigbee.print(", ");
zigbee.print(Z1);
zigbee.println(")");

  // Print calculated x and y coordinates to LCD
  lcd.setCursor(0, 0);
  lcd.print("Measured");
  lcd.setCursor(0, 1);
  lcd.print("X1:");
  lcd.setCursor(3, 1);
  lcd.print("      ");
  lcd.setCursor(3, 1);
  lcd.print(X1-0.40);
  lcd.setCursor(0, 2);
```

```
  lcd.print("Y1:");
  lcd.setCursor(3, 2);
  lcd.print("       ");
  lcd.setCursor(3, 2);
  lcd.print(Y1);
  lcd.setCursor(0, 3);
  lcd.print("Z1:");
  lcd.setCursor(3, 3);
  lcd.print("       ");
  lcd.setCursor(3, 3);
  lcd.print(Z1);
  lcd.setCursor(10, 0);
  lcd.print("Calculated");
  lcd.setCursor(10, 1);
  lcd.print("X0:");
  lcd.setCursor(13, 1);
  lcd.print(human_x);
  lcd.setCursor(10, 2);
  lcd.print("Y0:");
  lcd.setCursor(13, 2);
  lcd.print(human_y);
  lcd.setCursor(10, 3);
  lcd.print("Z0:");
  lcd.setCursor(13, 3);
  lcd.print(human_z);

// Print calculated x and y coordinates to serial monitor
Serial.print("X1: ");
Serial.println(X1);
Serial.print("Y1: ");
Serial.println(Y1);
Serial.print("Z1: ");
Serial.println(Z1);
Serial.print("distance1: ");
Serial.println(distance1);
Serial.print("distance2: ");
Serial.println(distance2);
Serial.print("distance3: ");
Serial.println(distance3);
Serial.print("distance4: ");
Serial.println(distance4);
Serial.println();

// Create a string containing the coordinates and send it over Zigbee
String data = "X: " + String(X1) + ", Y: " + String(Y1) + ", Z: " + String(Z1) + "\n";
zigbee.print(data);

delay(2000); // Wait for 2 seconds before taking another measurement
}
```

Appendix D: Simulink's MATLAB Function's Code

```
% Define the number of readings to take
num_readings = 2;

% Read distance data from each sensor and sum the readings
sum1 = 0;
sum2 = 0;
sum3 = 0;
sum4 = 0;

% Accumulate readings from each sensor
% In this loop, distance readings from each sensor are accumulated to obtain the total sum of readings.
% The variables sum1, sum2, sum3, and sum4 store the cumulative sum for each sensor.
for i = 1:num_readings
    sum1 = sum1 + u1;
    sum2 = sum2 + u2;
    sum3 = sum3 + u3;
    sum4 = sum4 + u4;
end

% Calculate the average distance for each sensor
% The average distance for each sensor is computed by dividing the respective sum by the number of readings.
% The variables distance1, distance2, distance3, and distance4 store the average distances from sensors 1, 2, 3, and 4, respectively.
distance1 = sum1 / num_readings;
distance2 = sum2 / num_readings;
distance3 = sum3 / num_readings;
distance4 = sum4 / num_readings;

% Trilateration Calculation
% Calculate the squared distances between each sensor and the unknown point
% The squared distances between each sensor and the unknown point are computed using the average distances obtained earlier.
% The variables d1_squared, d2_squared, d3_squared, and d4_squared store the squared distances from sensors 1, 2, 3, and 4, respectively.
d1_squared = distance1^2;
d2_squared = distance2^2;
d3_squared = distance3^2;
d4_squared = distance4^2;

% Calculate the coefficients for the linear equations
% The coefficients for the linear equations used in the trilateration process are calculated based on the sensor coordinates and squared distances.
% The variables a1, b1, c1, d1, a2, b2, c2, d2, a3, b3, c3, and d3 store the coefficients for the linear equations.
a1 = 2 * (sensor1_x - sensor2_x);
b1 = 2 * (sensor1_y - sensor2_y);
c1 = 2 * (sensor1_z - sensor2_z);
```

```
d1 = d2_squared - d1_squared - sensor2_x^2 + sensor1_x^2 - sensor2_y^2 + sensor1_y^2 - sensor2_z^2 + sensor1_z^2;

a2 = 2 * (sensor1_x - sensor3_x);
b2 = 2 * (sensor1_y - sensor3_y);
c2 = 2 * (sensor1_z - sensor3_z);
d2 = d3_squared - d1_squared - sensor3_x^2 + sensor1_x^2 - sensor3_y^2 + sensor1_y^2 - sensor3_z^2 + sensor1_z^2;

a3 = 2 * (sensor1_x - sensor4_x);
b3 = 2 * (sensor1_y - sensor4_y);
c3 = 2 * (sensor1_z - sensor4_z);
d3 = d4_squared - d1_squared - sensor4_x^2 + sensor1_x^2 - sensor4_y^2 + sensor1_y^2 - sensor4_z^2 + sensor1_z^2;

% Calculate the determinants for Cramer's rule
% The determinants required for Cramer's rule are calculated using the coefficients of the linear equations.
% The variables detA, detX, detY, and detZ store the determinants for subsequent calculations.
detA = a1 * (b2 * c3 - b3 * c2) - b1 * (a2 * c3 - a3 * c2) + c1 * (a2 * b3 - a3 * b2);
detX = d1 * (b2 * c3 - b3 * c2) - b1 * (d2 * c3 - d3 * c2) + c1 * (d2 * b3 - d3 * b2);
detY = a1 * (d2 * c3 - d3 * c2) - d1 * (a2 * c3 - a3 * c2) + c1 * (a2 * d3 - a3 * d2);
detZ = a1 * (b2 * d3 - b3 * d2) - b1 * (a2 * d3 - a3 * d2) + d1 * (a2 * b3 - a3 * b2);

% Calculate the coordinates of the unknown point
% Finally, the coordinates of the unknown point are computed using the determinants and coefficients obtained above.
% The variables X1, Y1, and Z1 store the calculated coordinates of the unknown point.
X1 = detX / detA;
Y1 = detY / detA;
Z1 = detZ / detA;
```

Appendix E: MATLAB Scatter Plot's Code

```
% Clear all variables and configurations
clear all;

% Set simulation options for the model
opt = simset('SrcWorkspace','current','solver','ode45', 'ZeroCross', 'off');

% Simulate the model "Trilat3DPlot.slx" for a duration of 10 seconds
sim('Trilat3DPlot.slx',[0 10], opt);

% Define the coordinates of the sensors
sensor1_x = -4.2; sensor1_y = 9.8; sensor1_z = 2.7;
sensor2_x = -4.2; sensor2_y = 0.5; sensor2_z = 2.5;
sensor3_x = -8.7; sensor3_y = 0.5; sensor3_z = 2.9;
sensor4_x = -8.7; sensor4_y = 9.8; sensor4_z = 2.1;

% Create a new figure for the 3D plot
figure();

% Set the size of the markers on the scatter plot
S = 100;

% Scatter plot of the detected human's coordinates
scatter3(X1, Y1, Z1, "*", 'DisplayName','Detected Human');
hold on;

% Scatter plot of the actual human's coordinates
scatter3(-5.1, 10.9, -3.0, "*", 'DisplayName','Actual Human');
hold on;

% Scatter plot of the sensor coordinates
% Each sensor is represented as a filled blue marker on the plot
scatter3(sensor1_x, sensor1_y, sensor1_z, S, "blue",'filled', 'DisplayName','Sensor1');
hold on;
scatter3(sensor2_x, sensor2_y, sensor2_z, S, "blue", 'filled', 'DisplayName','Sensor2');
hold on;
scatter3(sensor3_x, sensor3_y, sensor3_z, S, "blue", 'filled', 'DisplayName','Sensor3');
hold on;
scatter3(sensor4_x, sensor4_y, sensor4_z, S, "blue", 'filled', 'DisplayName','Sensor4');

% Display the legend to differentiate between different markers
legend;

% Set the axis labels and title for the plot
xlabel('X');
ylabel('Y');
zlabel('Z');
title('3D Scatter Plot');
```