EAST WEST UNIVERSITY

Localization Of Mobile Submerged Sensor Using Cayley-Menger determinant and PSO Algorithm

Submitted By:

Mehebuba Naorin Lima:2013-2-63-040

Sumaiya Binte: 2013-2-63-024

Md. Faizul Ibne Amin: 2013-2-60-029

Supervised By:

Dr. Anisur Rahman

Assistant Professor

Department of Computer Science and Engineering

EastWest University

A Project Submitted in Partial Fulfillment of the requirements for the Degree of Bachelors of Science in Computer Science and Engineering to the Department of Computer Science and Engineering.

East West University
Dhaka, Bangladesh
April 2018

Declaration

We hereby declare that this project was done under CSE497 and has not been submitted elsewhere for the requirement of any degree or diploma or for any purpose except for publication.

_____

Mehebuba Naorin Lima

ID: 2013-2-63-040

Department of Computer Science and Engineering

East West University

_____

Sumaiya Binte

ID: 2013-2-63-024

Department of Computer Science and Engineering

East West University

_____

Md. Faizul Ibne Amin

ID: 2013-2-60-029

Department of Computer Science and Engineering

East West University

Abstract

Underwater wireless sensor Networks(UWSNs) are usually deployed over a large sea area and the nodes are usually floating, due to their special environment. This results in a lower beacon node distribution density, a long time for localization, and more energy consumption. Currently most of the localization algorithm in this field do not pay enough consideration on the submerged mobile nodes/sensors. This paper investigates the problem of localizing submerged mobile sensors in a random direction having water current and provides a new mechanism to determine the coordinates of those sensors using only one buoy. In underwater wireless sensor networks (UWSN), the precise coordinate of the sensors that actuate or collect data is vital, as data without the knowledge of its actual origin has limited value. In this study, the method of determining the underwater distances between beacon and sensor nodes has been presented using combined radio and acoustic signals, which has better immunity from multipath fading. Cayley-Menger determinant is used to determine the coordinates of the beacon nodes. The velocity of this beacon nodes is determined using Particle swarm optimization. Finally here the unknown mobile nodes position is determined and updated using the velocities of beacon node/sensors and unknown mobile nodes which have found using PSO algorithm. Simulation results validate the proposed mathematical models by computing coordinates of mobile nodes with negligible errors.

Letter of Acceptance

We hereby declare that thesis is from the student's own work and best effort of us, and all other sources of information used have been acknowledged. This thesis has been submitted with our approval.

Supervisor

_____

Dr. Anisur Rahman
Assistant professor
Department of Computer Science and Engineering
East West University, Dhaka

Chairperson

_____

Dr. Ahmed Wasif Reza
Associate Professor and Chairperson
Department of Computer Science and Engineering
East West University

Acknowledgment

We begin by praising Almighty Allah since all the praises due to him. It is His mercy that made our work and life easier.

Our most heartfelt gratitude goes to our beloved parents for their endless support, continuous inspiration, great contribution and perfect guidance from the beginning to end.

We owe our thankfulness to our supervisor Dr. Anisur Rahaman for his skilled, almost direction, encouragement and care to prepare us.

Our sincere gratefulness for the faculty of Computer Science and Engineering whose friendly attitude and enthusiastic support that has given us for four years.

We are very grateful for the motivation and stimulation from our good friends and seniors.

We also thank the researchers for their works that help us to learn and implement the Determination of average acoustic speed in Underwater Wireless Sensor Network.

Table of Contents

Abbreviation and Acronyms:

WSN	Wireless Sensor Network
UWSN	Underwater Wireless Sensor Network
RF	Radio Frequency
PSO	Particle Swarm Optimization
SLMP	Scalable Localization with Mobility Prediction
RSSI	Received Signal Strength Indicator
TDOA	Time Difference of Arrival
TOA	Time of Arrival

List of Figure

List of Table

Chapter 1

Introduction

The research of Underwater Wireless Sensors Networks has acquired a remarkable pace due to their plenty of application. While wireless communication has become more eminent in the terrestrial area, there are many difficulties in the underwater sensor field. Moreover, 2/3 area in the world is covered with water and most of them are unexplored. To explore and to make use of this vast unexplored area, we need innovative technologies that will provide us a number of application such as pollution monitoring, disaster prevention, natural resources invention, military surveillance and tactical surveillance for scientific purposes. To this purpose, the exploration of underwater, Underwater Wireless Sensors Networks (UWSN) is mentionable

technologies which are a fusion of wireless technology with extremely small micromechanical sensors having smart sensing, intelligent computing and communication capabilities.

1.1 Underwater Wireless Sensor Network(UWSN):

Nowadays marine rights and interests are getting increasingly consideration since it has seen quick improvement in numerous countries. Hence, looks into on UWSNs (Under Water Wireless Networks) has grown rapidly, giving fundamental specialized help, for example, sea condition checking, asset investigation, catastrophic event cautioning, military protection, medicinal services, mobile communication, surveillance, utilities. Since most of the world's surface is covered with water, more research is directed on the submerged area. Information accumulation and condition observing have turned out to be real parts.

The qualities of the submerged condition give analysts numerous difficulties exceptionally creating compelling sensor correspondence and limitation procedures. Since marine life is an immense asset it is in this manner essential to gather exact information utilizing submerged sensors and to create a proper instrument for activation assignments. Furthermore, due to water streams, there is the non-negligible node versatility which may make frequent changes in the system topology.

The underwater wireless sensor network is a very prominent and interesting research sector in recent years of WSN. Submerged Wireless Sensor Networks comprises various sensor nodes, stationary or portable, associated remotely by means of acoustic correspondence modules sent to monitor different occasions of interest cooperatively. The goal is accomplished by having an arrangement of independent devices in a system which can self-sort out and adjust to remote ocean conditions. Submerged correspondence is basically done utilizing low frequency and low information rate acoustic modems with an arrangement of nodes transmitting their information to a buoyant gateway that transfers the information to closest seaside checking and control station.

The localization of mobile nodes that will gather this information is essential and important for submerged sensor systems. With random versatility, the sensor nodes could understand the observing of the entire region. The sensor nodes situated at the ocean bed can't communicate straightforwardly with the nodes close to the surface level, they require multi-hop communication helped by the proper directing plan. Wireless sensor network comprises some sensing devices that can impart wirelessly and these devices can process and speak with its associates.

Depending on the type of deployment, the wireless sensor and actuator system may comprise surface checking stations, independent submerged vehicles, and different natural sensors. The restriction of mobile nodes in this procedure will help progressively following and for recognizing the pollution territory.

In this way the successful execution on the area of the various division of our system and administration particularly in the marine life utilizing WSN localization, it turns into the fascination of the analysts to make it more material in various route requiring little to no effort. In addition, the data accumulation from the submerged devices and transmission are acknowledged by the sensor nodes and the applicable information without a location will have neither rhyme nor reason, so the localization of mobile sensor nodes has enhanced and turned into an exploration hotspot and core interest.

1.2 Challenges for Underwater Wireless Sensor Network:

The characteristics of UWSN described above present great difficulties and challenges to the localization of underwater mobile nodes in a large-scale network. Underwater sensor networks are quite different from terrestrial sensor networks. Underwater networks because of the characteristic (large delay, long distance of communication) of the network, the communication is relied on physical means like acoustic sounds to transmit the signal. Traditional RF networks might not work efficiently in underwater networks. Low bandwidth and large latency result in a long end to end delays, and this brings in challenges in reliable data transfer and traffic congestion control. Besides underwater sensors are still expensive devices and so extra protection required for underwater environment and more complex transceivers needed. Battery power is limited and usually, batteries cannot be recharged as solar energy cannot be exploited. In addition, the mobility of nodes makes the topology change frequently.

The algorithm intended for static systems need to run the localization method intermittently to refresh the nodes' area, which may cause more energy utilization for correspondence in a submerged situation. Moreover, so under normal conditions, it is hard to accomplish higher localization exactness and localization scope rate in a submerged domain.

On the other hand, if 3D Euclidean separation estimation technique is utilized, it requires the need of a specific number of neighboring nodes to quantify between node separations and where the error is proliferated through the system because of its recursive nature. In one technique the researchers propose a localization plot in view of floats moored to the waterbed and mobile nodes that need to discuss specifically with these floats to get their location. This technique does not support dynamic condition in light of the fact that floats should be sent ahead of time in known areas.

Another challenge is sensors' powerlessness to communicate with each other is that the sensors are influenced by sea streams, which have uncertain portability, so sensors can't acquire satisfaction location data of some other sensors in an age. All these marvels results in the whole UWSNs having a high delay, and even a high likelihood of interruption condition.

1.3 Objective:

The main objective of the project is to find the coordinates of the submerged beacon and unknown sensors nodes having a surface buoy plane and beacon plane in parallel with each other when all the submerged unknown sensors are mobile. In recent times submerged wireless sensor network is an active research area due to the vast resource, security and environmental issue and our objective are to explore this unexplored submerged area. The goal of the work in this thesis is:

1. To localize beacon sensor; at first, we will find the beacon coordinates using Cayley-Menger determinant.
2. Then calculate beacon and unknown sensor nodes velocity.
3. And finally, localize the unknown mobile sensor nodes coordinates using PSO algorithm.

1.4 Methodology:

To explore the special environment of UWSN and the difficulties in mobile sensor localization, we propose a method based on using Cayley-Menger determinant and PSO algorithm and calculate the coordinates of the beacon and unknown sensor node. First, we measure the distances between the surface buoy and beacon sensors where beacon sensors are assumed in a plane which is parallel to the surface buoy. Then localize the beacon sensors coordinates using Cayley-Menger determinant. Then we calculate the beacon and unknown sensor nodes velocity using PSO algorithm. After calculating unknown nodes velocity we can update their locations.

Chapter 2

Background and Literature Review:

The characteristics of UWSN portrays present incredible troubles and difficulties to the restriction of submerged mobile nodes in a vast scale network. Acoustic correspondence has a greater propagation delay, lower bandwidth and higher error rate contrasted with radio correspondence. Utilizations of submerged detecting range from the oil industry to aquaculture, and include instrument checking, pollution control, atmosphere recording an expectation of natural disturbances, search and overview missions, and investigation of marine life. Wireless data transmission through the sea is one of the empowering advancements for the improvement of future sea perception systems and sensor systems.

2.1 Cayley-Menger determinant:

Cayley-Menger determinant is used to determine the coordinates of the sensor nodes where none of the nodes have a priori knowledge about location. Cayley-Menger determinants are used to characterize Euclidian spaces in terms of distances between points.

2.2 Particle Swarm Optimization:

Particle Swarm Optimization (PSO) is a computational strategy that enhances a problem by iteratively attempting to enhance a hopeful solution with respect to a given measure of quality. It solves an issue by having a population of applicant solutions, here named particles, and moving these particles around in the search space as indicated by simple numerical formula over the particle's position and velocity. Every particle's movement is affected by its nearby best-known position, but on the other hand, it also guided toward the best-known positions in the search space, which are updated as better positions are found by different particles. This is expected to move the swarm toward the best solutions [1].

PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called gbest. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called lbest [2].

2.3 Related Works:

Localization methods for wireless sensor networks can be divided into two types: range-based and range-free methods. The range-based methods such as the received signal strength indicator (RSSI), time difference of arrival (TDOA) and time of arrival (TOA) use hardware to measure the distance information. These kinds of the method have a higher accuracy, but they increase the network cost and energy consumption [3]. The range-free methods use the connectivity of the network to locate the unknown nodes. The typical range-free methods mainly include the DV-HOP, Convex Programming, and Centroid Localization algorithm. These methods have no additional hardware requirements, and they have lower energy consumption and shorter positioning time, but their accuracy is usually lower [4].

There are many types of research on terrestrial nodes localization. In [5] the authors proposed a range-free localization algorithm based on a sequential Monte Carlo localization method. It can exploit mobility to improve the localization accuracy. In [6], a Monte Carlo Localization algorithm with mobility prediction (MCL-MP) was proposed, and it can further improve the accuracy by using prediction and filtering for the unknown nodes based on dynamic sampling.

Studies on the localization of underwater mobile nodes always face some challenges, and most of them were designed for small-scale networks. For example, GPS Intelligent Buoys (GIB) based on surface buoys and one-hop communication under the water has been proposed. This approach has a high accuracy but the hardware is complex and the cost is high [7]. In [8], a so-called Silent Localization algorithm was proposed, which does not need time synchronization and is applicable to one-hop underwater networks. In addition, in [9], a Scalable Localization with Mobility Prediction (SLMP) method was proposed, and it is closer to localization in an actual environment. The SLMP algorithm has two stages to locate the unknown nodes. First, the velocity of beacon nodes will be estimated by the Durbin algorithm to perform the online linear prediction. The unknown nodes are then located by using the mobility prediction based on the spatial correlation of sensor nodes.

A three-dimensional deployment space is another important characteristic of UWSNs. In [15], an efficient localization scheme which can transform the three-dimensional localization problem into a two-dimensional counterpart via a projection technique was proposed; it can make all the nodes map to the same plane. Because the depth information can be obtained by a pressure sensor, this scheme not only makes the two-dimensional localization algorithm apply in the three-dimensional space, but also simplifies the amount of calculation for three-dimensional localization, and reduces the energy consumption of the process. In this paper, we also use this method to project the nodes on a certain plane and only consider the velocity of the nodes in one direction along with X-axis. The principle of this method is shown in **Figure 2.1**.

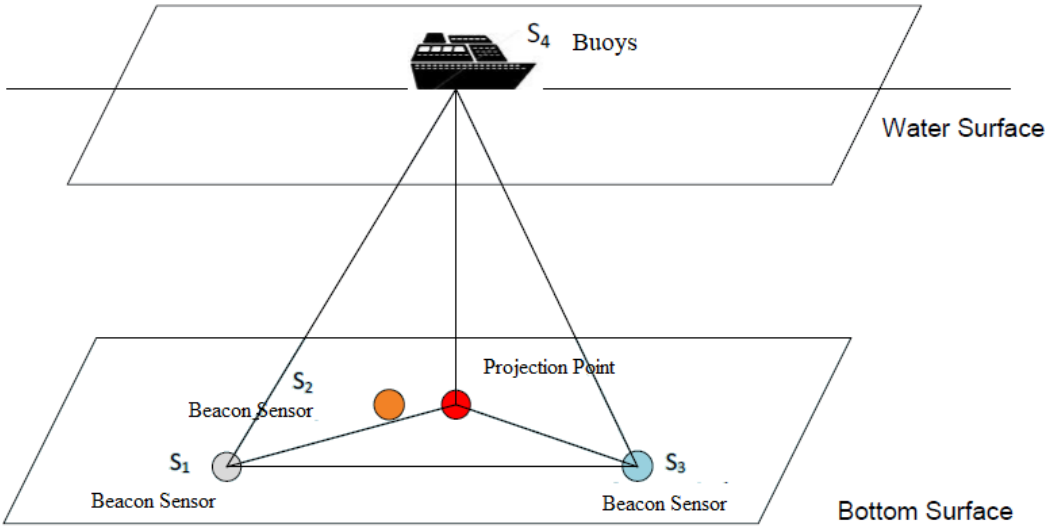

Figure 2.1A diagram of the projection method

As shown in Figure 2.1, S1, S2, and S3 are beacon sensor nodes in underwater networks. The projection point of the buoy(the ship) is shown on the bottom surface of the seabed. Then we only need to locate the nodes in the surface plane, and the final location can be obtained by simply adding the respective distance information.

2.4 Localization technique of UWSN:

Localization of sensor nodes is an important aspect of Wireless Sensor Networks (WSNs).This methods for wireless sensor networks can be divided into two types

- Range-based approach
- Range-free approach

2.4.1 Range-Based Approach

The range-based methods such as the received signal strength indicator (RSSI), time difference of arrival (TDOA) and time of arrival (TOA) use hardware to measure the distance information. These kinds of the method have a higher accuracy, but they increase the network cost and energy consumption.

Received Signal Strength Indicator (RSSI):

RSS is a common technique in localizing sensor nodes; this is due to the fact that almost all nodes have the ability to measure the strength of the received signal. RSS technique benefits from the fact that radio signals diminish with the square of the distance from the signal's source. In other words, the node can calculate its distance from the transmitter using the power of the received signal, knowledge of the transmitted power, and the path-loss model. The operation starts when an anchor node broadcasts a signal that is received by the transceiver circuitry and passed to the Received Signal Strength Indicator (RSSI) to determine the power of the received signal.

Since the path loss in underwater acoustic channels is usually time-varying and multipath effect can result in significant energy fading, the RSSI method is not the primary choice for underwater localization.

Time Difference of Arrival (TDOA)

TDOA can be measured based on the fact that the distances between the transmitter and different receivers are different. This means that the transmitted signal is delayed in time based on the distance to the receiver. Based on the two received signals, the distance to the transmitter can be determined. However, it is unsuitable for underwater localization because radio does not propagate well in water. Alternatively, the time difference of arrival between beacons from different reference nodes transmitted using acoustic signaling can be used in localization.

Time of Arrival (TOA)

TOA is defined as the earliest time at which the signal arrives at the receiver. It can be measured by adding the time at which the signal is transmitted with the time needed to reach the destination (time delay). The time delay can be computed by dividing the separation distance between the nodes by the propagation velocity. In TOA, the nodes have to be synchronized and the signal must include the time stamp information. To overcome these restrictions, Roundtrip Time of Arrival (RTOA) and Time Difference of Arrival (TDOA) are developed.

2.4.2 Range-Free Approach:

The range-free approach employs to find the distances from the non-anchor nodes to the anchor nodes. This approach uses the connectivity of the network to locate the unknown nodes. The typical range-free methods mainly include the DV-HOP, Convex Programming, and Centroid Localization algorithm. These methods have no additional hardware requirements, and they have lower energy consumption and shorter positioning time, but their accuracy is
usually lower.[11]

There are three basic localization techniques that are used as a base advanced techniques: [12]

1.Trilateration: This method determine the position of a node from the intersection of 3 circles of 3 anchor nodes that are formed based on distance measurements between its neighbors. The radius of the circle is equal to the distance measurement. However, in a real environment, the distance measurement is not perfect; hence, more than three nodes are required for localization.

2. Triangulation: This method is used when the direction of the node rather than the distance is estimated. It uses trigonometry laws of sines and cosines to calculate the position of the node based on the angle information from two anchor nodes and their positions.

3. Maximum Likelihood Multilateration: Trilateration technique cannot accurately estimate the position of a node if the distance measurements are noisy. A possible solution is to use the Maximum Likelihood (ML) estimation, which includes distance measurements from multiple neighbor nodes. This method intends to minimize the differences between the measured distances and estimated distances.

Three-Dimensional Underwater Target Tracking (3DUT) Scheme

A Three-Dimensional Underwater Target Tracking (3DUT) scheme is also proposed. As shown in Figure2.2, at least three anchor nodes float at the surface of the water. One of these nodes is the sink (node*A*) which collects the information from underwater sensor nodes and carries out the calculations. The black nodes collect and send information from the target to the sink. The gray node is the designated projector node. 3DUT is a two-phase algorithm. During the first phase, Passive Listening, sensor nodes listen to the underwater environment for potential targets. The second phase of the algorithm, Active Ranging, is to localize the target. 3DUT selects a projector node which sends pings periodically. The target is assumed to be a point target so that the echoes are radiated isotropically. Once the echo is received by the projector, it calculates its distance to the target and transmits to the sink node. Sink node uses trilateration to localize the target. The location and the calculated velocity of the target are then exploited to achieve tracking. Depending on the results of the calculations, sink node selects a new projector node.

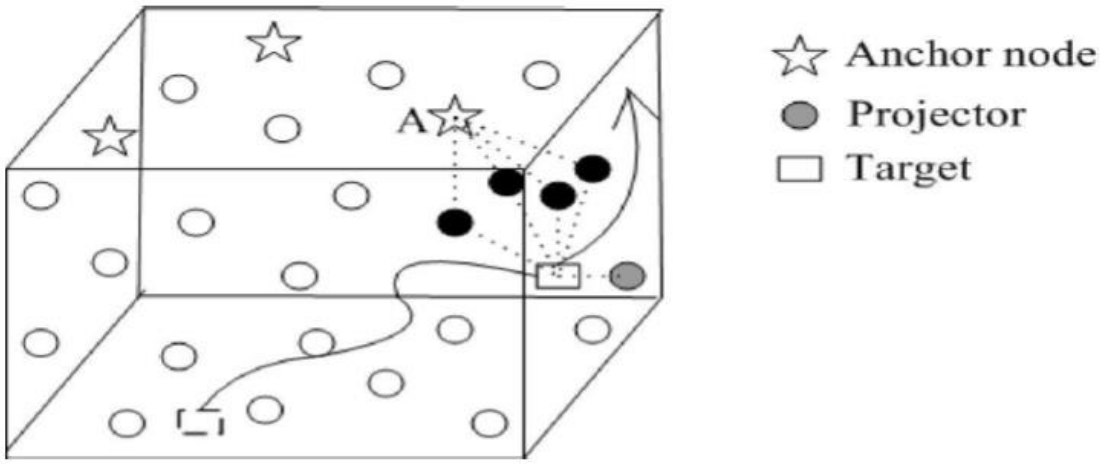

Figure 2.2 The Three-Dimensional Underwater Target Tracking.

To save energy, the nodes which are not located at the network edge have low duty cycles. The nodes which are at the boundary of the sensing region have higher duty cycles in order to detect the target entering into the sensing region immediately.

Therefore, to avoid rapid energy depletion of boundary nodes due to continuous surveillance, 3DUT employs an adaptive procedure to find, designate, and activate new boundary nodes. Furthermore, 3DUT does not depend on the number of nodes. The algorithm runs even if the number of sensor nodes changes. However, 3DUT can only track one target at a time. Moreover, the tracking accuracy is heavily influenced by the target's velocity.

Chapter 3

Proposed Method:

Research shows that the mobility of underwater object is influenced by water current, temperature, and some other factors, so we cannot use a unified model to describe the mobility of the nodes for all environments. In our proposed method, we designed a model to explore underwater things that contains great priority in research and environment. In our proposed model, at first, we localize the beacon sensors and calculate their coordinates by using Cayley-Menger determinant. And then we will calculate beacon sensors and unknown sensors velocity By PSO algorithm. Finally, we will locate and update the positions of submerged mobile unknown nodes. This will help us to explore the vast area of underwater.

3.1 Problem Statement:

In this paper, to determine the coordinates of the submerged sensors, our proposed method assumes at least three beacon sensors and a floating buoy. It is also assumed that the distance measurement between the buoy and beacon sensors are possible. In the marine environment, a boat or a buoy can be used as a surface buoy and beacon and sensors could be deployed in the water. While measuring the multiple distances between the buoy and beacon, those locations of the buoy are assumed to be in a plane, which is approximately parallel to the plane created by the three beacon (as shown in Figure. 3.1). But if these three beacons are mobile then we will need to find these sensors location using only one buoy.

For finding mobile beacons location from one buoy, at first measurements of the distances from six locations of the buoy are taken. As the general properties of a transducer, the buoy and beacon have the capability of generating radio and acoustic signal, whereas ordinary sensors might have the restricted capability of receiving the radio and acoustic signal for timing purpose as well as it is enabled with the acoustic transmission. A solvable configuration of one buoy with three submerged mobile beacons is denoted in Figure. 3.1. Our proposed mechanism exploits the advantage of both radio and acoustic signal propagation in seawater in 1.8-323m depth. Since most of the marine explorations take place in shallow water, our proposed model has wide-ranging practical applications.

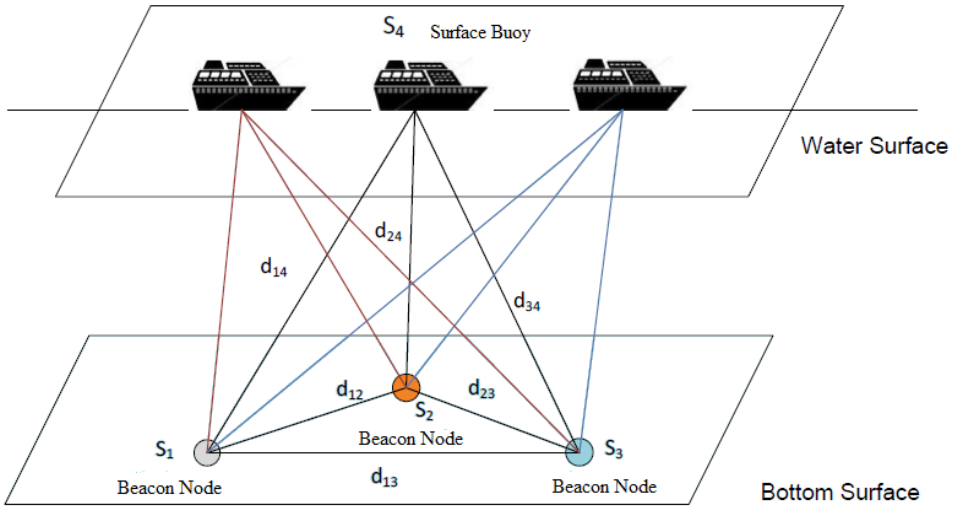

Figure. 3.1 A solvable configuration of one buoy with three submerged beacons

Here the position where the first beacon(S1) was deployed, we denoted that position as (0,0,0) on the x-axis. The position where the second beacon (S2) was deployed, we denoted that position as (0,y,0) and the position where the third beacon(S3) was deployed, we denoted that position as (x,y,0).

3.2 Environmental Limitations:

Normally, the submerged condition is more antagonistic than earthbound condition; in spite of those restrictions, it represents a few merits that could be misused in deciding directions. The water body is generally more homogeneous in light of the fact that the standard hindrances exhibit in water is littler in estimate than that of an earthly condition. The locale of enthusiasm on the ground is more probable involved with structures and trees which are the main considerations for multipath propagation[13].

Regarding signal propagation in water, the acoustic signal propagates much further compare to the radio signal; however, the speed of the acoustic signal is much slower than that of the radio signal.

TABLE 3.1 shows some limitations and typical measurements for radio and acoustic signals.

	Radio signal		Acoustic signal	
	Vacuum	Water	Vacuum	Water
Velocity	3×108 m/s	≈2.25×106 m/s	-	VA
Range	-	1.8-323 m	-	1 - 100 km

TABLE 3.1 Properties of Radio and Acoustic signal

The main environmental variable that we assume in our method to determine distances is the speed of acoustic signals in water. It depends on the temperature, salinity,and permeability. How the speed of acoustic will vary because of aforesaid factors is not considered in this study, but our mathematical model assumes it as a variable *V*A.

3.3 Distance Measurement:

In spite of the impediment of both radio and acoustic signal propagation in water in various viewpoint, we will abuse every one of their benefits in our proposed method to build the precision of the distance measurements. Differential speed amongst radio and acoustic signs will be utilized to figure the distance, while acoustic signal will be utilized for correspondence purposes. This technique will require a short communication in the middle of the Buoy and beacon nodes.

Despite the fact that the speed of radio flag is somewhat not as much as that of in the vacuum, considering about the problem domain, the speed variety won't have a critical effect on the proposed localization strategy. Besides, the speed of the acoustic signal, which shifts because of various environmental components, is the principal variable that we have to use for coordinate determination.

Assumptions:

• The buoy can generate radio and acoustic signals simultaneously.

• The environmental factors that affect the acoustic signal will be considered while measuring inter- beacon node distances.

• Beacon nodes are mobile.

• The base for all the beacons is same and the base is of tetrahedron shape.

• Each beacon node will have a unique ID.

The distance measurement calculation mentioned below:

1. Simultaneous generation of radio and acoustic signals by the buoy,$S_j, j = 4,5$ ... ...at t_0
2. For any submerged sensors $S_i, i = 1,2,3$
    - beacon receive the radio signal at $t_{R(rec)} = t_0 + \in$
    - beacon receives the acoustic signal at $t_{A(rec)}$; here $t_{A(rec)} \gg t_{R(rec)}$
3. Time is taken for the acoustic signal to travel from buoy to beacons :

$$T_{ij(Travel),i=1,2,3;j=4,5,6\ldots\ldots} = t_{A(rec)} - t_{A(tra)}$$
$$= t_{A(rec)} - t_{R(tra)}\ ; where\ t_{A(tra)} = t_{R(tra)}$$

So,

$$\therefore T_{ij(Travel)} \approx t_{A(rec)} - t_{R(rec)}; where\ t_{R(rec)} = t_0 + \in \approx t_{R(tra)}$$

4. Beacon nodes send back the time $T_{ij(Travel)}$ with individual beacon's ID to the buoy using acoustic signal.

5. Buoy compute the distance between the buoy and beacons: $d_{ij} = v_A * T_{ij(Travel)}$here,$v_A$ is average acoustic signal speed.

6. Sensor nodes send back the time $T_{ij(Travel)}$ with individual sensor's ID to the beacon using acoustic signal.

3.6 Coordinates of the Beacon:

The objective of localization algorithms is to obtain the exact position or coordinates of all the beacon nodes by measuring distances between buoy and beacon. Only measurements available here to compute is the distance and typically it is considered an optimization problem where objective functions to be minimized have residuals of the distance equations.

The variables of any localization problem are the 3D coordinates of the nodes. In principle,a number of distance equations are needed than a number of variables to solve this kind of problem. However, this approach known as the degree of freedom analysis may not guarantee the unique solution in a nonlinear system [13].

Trilateration or multilateration techniques that are nonlinear system usually used to determine the location or coordinates of the sensors in partial or full. According to Guevara et al. [14], the convergence of optimization algorithms and Bayesian methods depend heavily on initial conditions used and they circumvent the convergence problem by linearizing the trilateration equations.

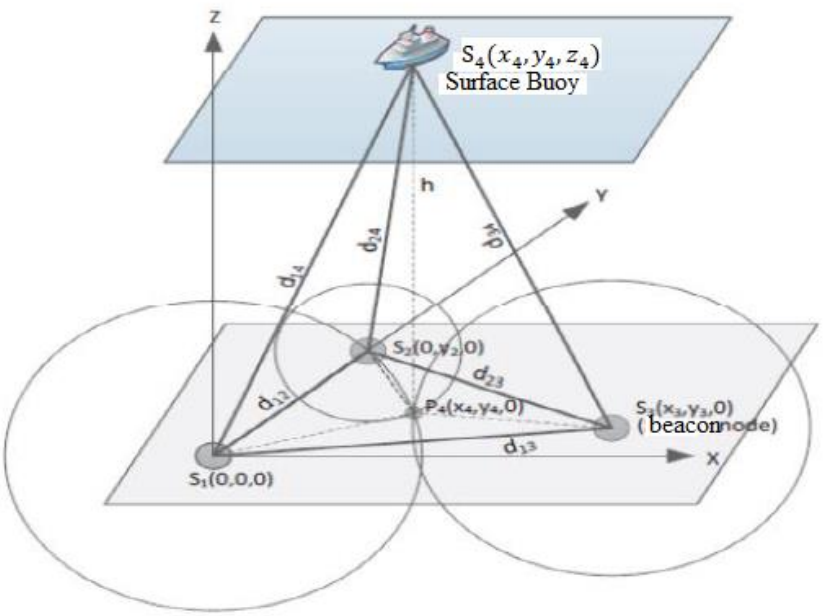

Figure 3.2: Coordinates determinations

Figure 3.2 shows the initial subset composed of the buoy*S j*,*j* = 4,5....9 and three beacon nodes *Si*,*i*=1,2,3. Without loss of generality, a coordinate system can be defined using one of the beacon nodes *Si*,*i*=1,2,3. as the origin (0,0,0) of the coordinate system. Now the trilateration equations can be written as a function of two groups of distance measurements. The distance between buoy and beacon d14, d24, d34.... which are measured data, and inter-beacon distances d12, d13, d23 and the volume of tetrahedron *Vt*(formed by the buoy and beacon), are unknown. By expanding and grouping known-unknown variables of (1), we obtain;

$$d_{34}^2(d_{12}^2 - d_{23}^2 - d_{13}^2) + d_{14}^2\left(\frac{d_{23}^4}{d_{12}^2} - d_{23}^2 - \frac{d_{13}^2 d_{23}^2}{d_{12}^2}\right) + d_{24}^2\left(\frac{d_{13}^4}{d_{12}^2} - \frac{d_{13}^2 d_{23}^2}{d_{12}^2} - d_{13}^2\right)$$
$$- (d_{14}^2 d_{24}^2 + d_{14}^2 d_{34}^2 - d_{24}^2 d_{34}^2 - d_{14}^4)\frac{d_{23}^2}{d_{12}^2}$$
$$- \left((d_{34}^2 d_{24}^2 - d_{14}^2 d_{34}^2 - d_{24}^2 d_{24}^2 - d_{24}^4)\frac{d_{13}^4}{d_{12}^2}\right) + \left(144\frac{V_t^2}{d_{12}^2} + d_{13}^2 d_{23}^2\right)$$
$$= (d_{24}^2 d_{34}^2 - d_{34}^4 + d_{14}^2 d_{34}^2 - d_{14}^2 d_{24}^2)$$

Here,

$(d_{12}^2 - d_{23}^2 - d_{13}^2)$	$(\frac{d_{23}^4}{d_{12}^2} - d_{23}^2 - \frac{d_{13}^2 d_{23}^2}{d_{12}^2})$	$(\frac{d_{13}^4}{d_{12}^2} - \frac{d_{13}^2 d_{23}^2}{d_{12}^2} - d_{13}^2)$	$\frac{d_{23}^4}{d_{12}^2}$	$\frac{d_{13}^4}{d_{12}^2}$	$(144\frac{V_t^2}{d_{12}^2} + d_{13}^2 d_{23}^2)$

are unknown terms.

The above expression can be written as follows,

$d_{14}^2 X_1 + d_{24}^2 X_2 + d_{34}^2 X_3$ $- (d_{14}^2 - d_{34}^2)(d_{24}^2 - d_{14}^2)(d_{24}^2 - d_{14}^2)(d_{34}^2 - d_{24}^2)X_5 + X_6$	$= (d_{24}^2 - d_{34}^2)(d_{34}^2 - d_{14}^2)$	(1)

Based on the local positioning system configuration of Figure. 3.4, we need to write equations that will include all known and unknown distances. For that matter, we express the volume of tetrahedron Vt using Cayley-Menger determinant as following:

$288V_t^2 = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & d_{12}^2 & d_{13}^2 & d_{14}^2 \\ 1 & d_{12}^2 & 0 & d_{23}^2 & d_{24}^2 \\ 1 & d_{13}^2 & d_{23}^2 & 0 & d_{34}^2 \\ 1 & d_{14}^2 & d_{24}^2 & d_{34}^2 & 0 \end{bmatrix}$	(2)

The Equation (2) in fact resembles the linear form of $a_1x_1 + a_2x_2 + \cdots + a_nx_n = b_1$. As we have six unknown in (1), we need at least six measurements, which could be done following the the same procedure described earlier steering the buoy$S\,j$,j=4,5....9 to six different locations and measuring the distances in the vicinity of S_4. Finally, we get m-linear equations of the form;

$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$	
$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$	
$\vdots$	
$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m$	(3)

If we omit reference to the variables, the system of equations in (3) can be represented by the array of all coefficients known as the augmented matrix of the system, where the first row of the array represents the first linear equation and so on. That could then be expressed in AX =b linear form. Then, the system of equations can be written as:

$$A = \begin{bmatrix} d_{14}^2 & d_{24}^2 & d_{34}^2 - (d_{14}^2 - d_{34}^2)(d_{24}^2 - d_{14}^2) - (d_{24}^2 - d_{14}^2)(d_{34}^2 - d_{24}^2) & 1 \\ d_{15}^2 & d_{25}^2 & d_{35}^2 - (d_{15}^2 - d_{35}^2)(d_{25}^2 - d_{15}^2) - (d_{25}^2 - d_{15}^2)(d_{35}^2 - d_{25}^2) & 1 \\ \vdots & \vdots & \vdots & \vdots \\ d_{19}^2 & d_{29}^2 & d_{39}^2 - (d_{19}^2 - d_{39}^2)(d_{29}^2 - d_{19}^2) - (d_{29}^2 - d_{19}^2)(d_{39}^2 - d_{29}^2) & 1 \end{bmatrix}$$

$X = \begin{bmatrix} (\frac{d_{23}^4}{d_{12}^2} - d_{23}^2 - \frac{d_{13}^2 d_{23}^2}{d_{12}^2}) \\ (\frac{d_{13}^4}{d_{12}^2} - \frac{d_{13}^2 d_{23}^2}{d_{12}^2} - d_{13}^2) \\ (d_{12}^2 - d_{23}^2 - d_{13}^2) \\ \frac{d_{23}^4}{d_{12}^2} \\ \frac{d_{13}^4}{d_{12}^2} \\ (144\frac{V_t^2}{d_{12}^2} + d_{13}^2 d_{23}^2) \end{bmatrix}$	$b = \begin{bmatrix} (d_{24}^2 - d_{34}^2)(d_{34}^2 - d_{14}^2) \\ (d_{25}^2 - d_{35}^2)(d_{35}^2 - d_{15}^2) \\ \vdots \\ (d_{29}^2 - d_{39}^2)(d_{39}^2 - d_{19}^2) \end{bmatrix}$

From the above representation, after finding,X_1,X_2, X_3 ,X_4, X_5 and X_6 we calculate d_{12},d_{13} andd_{23} as follows:

$d_{12}^2 = \frac{X_3}{(1 - X_4 - X_5)}$	$d_{13}^2 = \frac{X_3 X_5}{(1 - X_4 - X_5)}$	$d_{23}^2 = \frac{X_3 X_4}{(1 - X_4 - X_5)}$

From the above values the unknown variables can be computed as follows:

$y_2 = d_{12}$	$y_3 = \frac{d_{12}^2 + d_{13}^2 - d_{23}^2}{2d_{12}}$	$x_3 = \sqrt{\left(d_{13}^2 - (\frac{d_{12}^2 + d_{13}^2 - d_{23}^2}{2d_{12}})^2\right)}$

, ,

where d_{12}, d_{13},andd_{23}, are known computed distances Table 3.2 summarizes the coordinates of the sensors for this system.

Beacons	Coordinates
S_1	(0.0.0)
S_2	$(0, d_{12}, 0)$
S_3	$\sqrt{(d_{13}^2 - (\frac{d_{12}^2+d_{13}^2-d_{23}^2}{2d_{12}})^2)}, \frac{d_{12}^2+d_{13}^2-d_{23}^2}{2d_{12}}, 0$

Table 3.2 Coordinates of the beacons with known measurements

3.7 Coordinates of the Beacons with respect to the Buoy:

Up to now, we have been able to determine the coordinates of the beacon nodes with respect to *S*1. In order to find the coordinate with respect to the buoy, we follow the following steps.

We assume that with the use of appropriate beacons, the depth *h* can be measured [15].After measuring the vertical distance *h* in between the buoy$S_4(X_4, Y_4, Z_4)$ and the *XY* plane, we can assume the projected coordinate of the beacon node $S_4(X_4, Y_4, Z_4)$on the plane *XY*is$P_4(X_4, Y_4, 0)$To find *x*4 and *y4*, we can apply trilateration in the following manner assuming the distances between S_1, S_2, S_3,and$P_4$are D_{14},D_{24},andD_{34}respecti vely.

$D_{14}^2 = x_4^2 + y_4^2$	(4)
$D_{24}^2 = x_4^2 + (y_4 - y_2)^2$	(5)
$D_{34}^2 = (x_4 - x_3)^2 + (y_4 - y_3)^2$	(6)

From equation (4), (5) and (6) we obtain the projected beacon's coordinates $P_4(X_4, Y_4, 0)$

$X_4 = \sqrt{\frac{1}{2D}(2d_{12}D_{14}^2 - D_{14}^2 + D_{24}^2 + d_{12}^2)}$
$Y_4 = \frac{1}{2d_{12}}(D_{14}^2 - D_{24}^2 + d_{12}^2)$

As ford_{14}, d_{24} and d_{34} are the hypotenuse of the$\Delta S_1P_4S_4$, $\Delta S_2P_4S_4$and$\Delta S_3P_4S_4$respectively, so it is possible to obtain D_{14},D_{24},and D_{34}usin g Pythagorean Theorem. So the coordinate of the buoy $S_4(X_4, Y_4, Z_4)$would be $S_4(X_4, Y_4, h)$where all the elements are known.

So,

$$S_4(X_4, Y_4, 0) = S_4\left(\begin{array}{c}\left(\sqrt{\frac{1}{2D}(2d_{12}D_{14}^2 - D_{14}^2 + D_{24}^2 + d_{12}^2)}\right), \\ \left(\frac{1}{2d_{12}}(D_{14}^2 - D_{24}^2 + d_{12}^2)\right), h\end{array}\right)$$

If the origin of the Cartesian system is transferred on to the coordinate of the buoy, then it is possible to find the coordinates of other beacons with respect to the buoy S4. A linear transformation would give the results as in Table 3.3.

Beacons	Coordinates	Beacons	Coordinates
S_4	(0, 0, 0)	S_2	$(x_2\text{-}x_4\ ,\ y_2 - y_4\ ,\ \text{-}z_4)$
S_1	$(\text{-}x_4\ ,\text{-}\ y_4\ ,\ \text{-}z_4)$	S_3	$(x_3\text{-}x_4\ ,\ y_3 - y_4\ ,\ \text{-}z_4)$

Table 3.3 Coordinates of the beacons with respect to a buoy for a parallel situation

3.8 Coordinates of the Beacons with respect to the Buoy when they are mobile:

Upon finding the coordinates of the beacons initially i.e when they were static, in here using those measurements we find out their coordinates with their velocity in their mobile form. In order to do that, when the beacons and the buoy move from their previous position to their next, letting them do that, we get the beacon to its previous position for the measurement purpose.

So, after that, we use the triangulation technique and measure the distance between the buoy's current position to beacon's previous position. Finding the distance we again use the Cayley-Menger determinant to find out its position then we add the distance that it covered with its speed to find out its actual positions coordinates. That's how we localize the mobile beacons.
Now finding out the beacons positions based on the axes.

When the Beacon S1 is on X-axis:

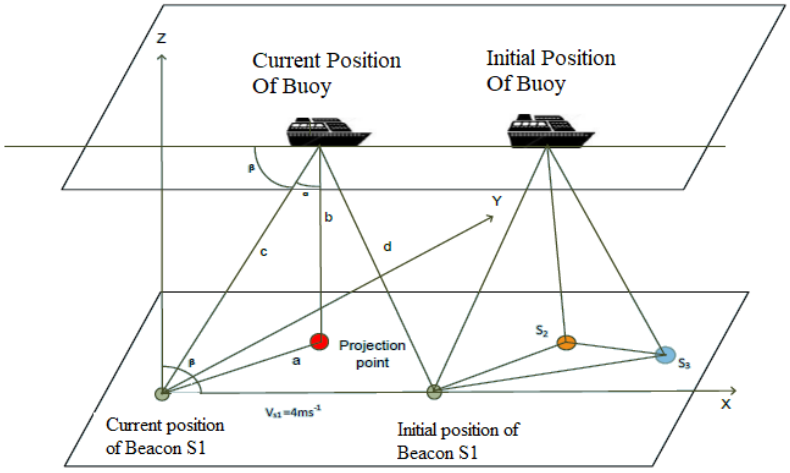

Figure 3.3: beacon S1 on X-axis

From the above figure, we first find out the angle α,

$$a^2 = b^2 + c^2 - 2.b.c.\cos a$$

Then we find out the angle β,

$$\boldsymbol{\beta = 90° - \alpha}$$

then finally we find out the distance between the buoy's current position to beacon's previous position by using the equation which is given below:

Let, $e = V_{s1} ms^{-1}$
$d^2 = c^2 + e^2 - 2.c.e.\cos\beta$
So, $d = \sqrt{c^2 + e^2 - 2.c.e.\cos\beta}$

When the beacon S2 is on Y-axis:

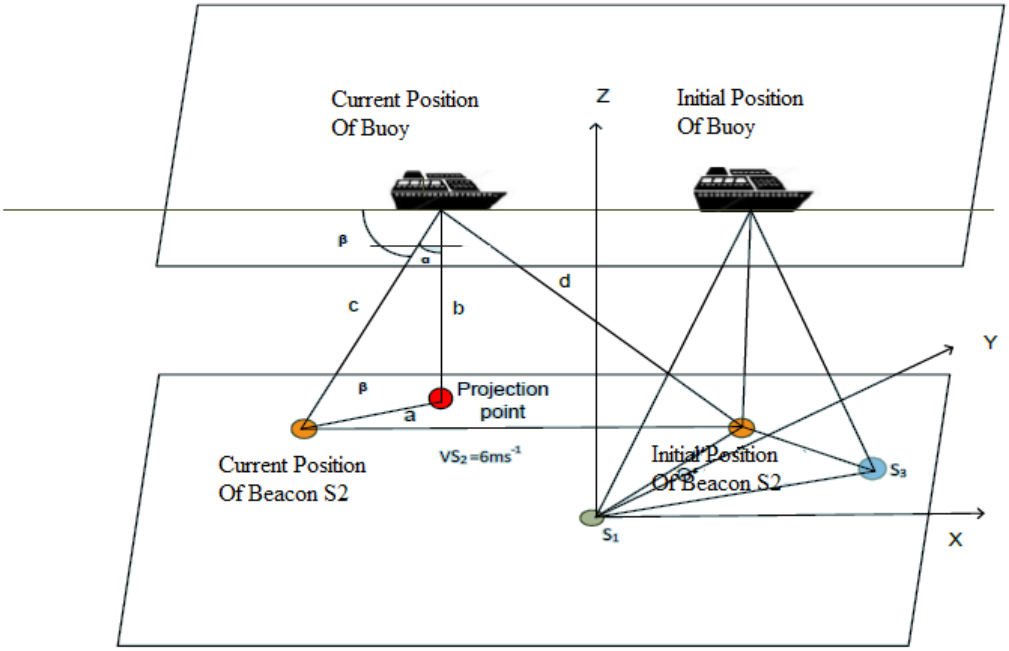

Figure 3.4: beacon S2 on Y-axis

From the above figure, we first find out the angle α,

$a^2 = b^2 + c^2 - 2.b.c.\cos a$

then we find out the angle β,

$\boldsymbol{\beta = 90° - \alpha}$

then finally we find out the distance between the buoy's current position to beacon's previous position by using the equation which is given below:

Let, $e = V_{s2} ms^{-1}$
$d^2 = c^2 + e^2 - 2.c.e.\cos\beta$
So, $d = \sqrt{c^2 + e^2 - 2.c.e.\cos\beta}$

When the beacon S3 is out of axis:

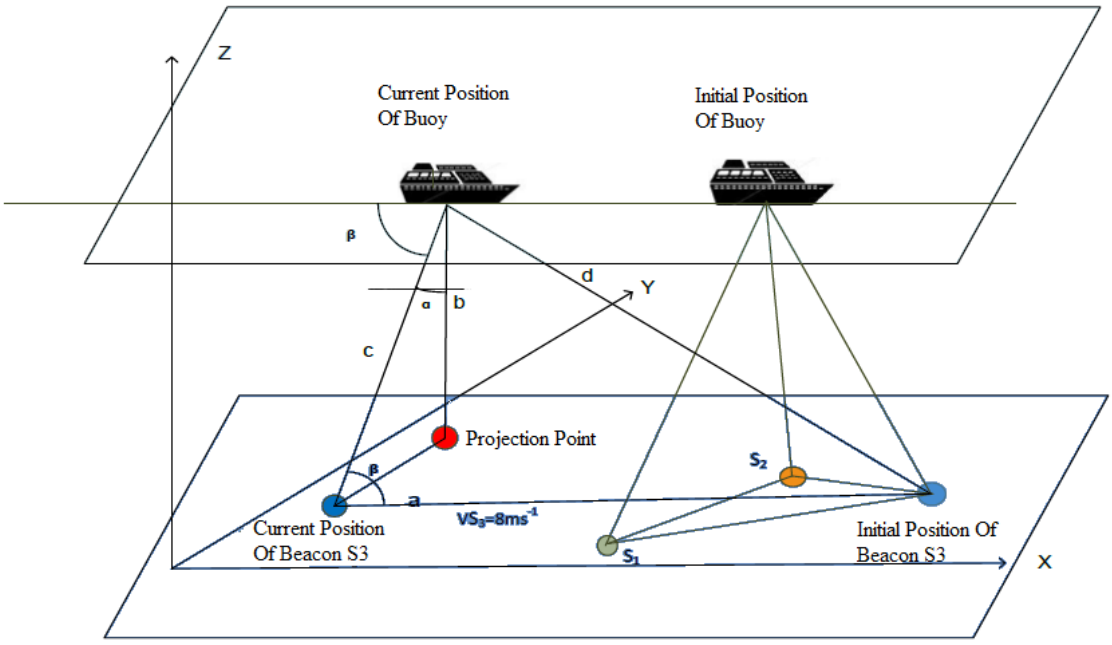

Figure 3.5: beacon S3 out of axis

From the above figure, we first find out the angle α,

$a^2 = b^2 + c^2 - 2.b.c.\cos a$

then we find out the angle β,

$\boldsymbol{\beta = 90° - \alpha}$

then finally we find out the distance between the buoy's current position to beacon's previous position by using the equation which is given below:

Let, $e = V_{s3} ms^{-1}$
$d^2 = c^2 + e^2 - 2.c.e.\cos\beta$
So, $d = \sqrt{c^2 + e^2 - 2.c.e.\cos\beta}$

3.9 Velocity Calculation:

- Calculate the velocity of beacon sensor according to localization result.
- The sensor node sort their reference nodes in descending order according to the confidence value
- If the number of reference node is more than M, then select the M reference nodes with larger confidence coefficients to calculate the velocity, if not, use all the reference nodes to calculate the velocity.
- Update the locations of the unknown sensor nodes.

3.9.1 The Calculation of Beacon Nodes Velocity:

The range-based localization algorithm needs more energy and computation. In real practical applications, the number of beacon nodes in real networks is relatively less. These nodes have rather strong computational ability and more energy, so they can run the localization algorithm periodically to get their velocities of the previous time points [16].

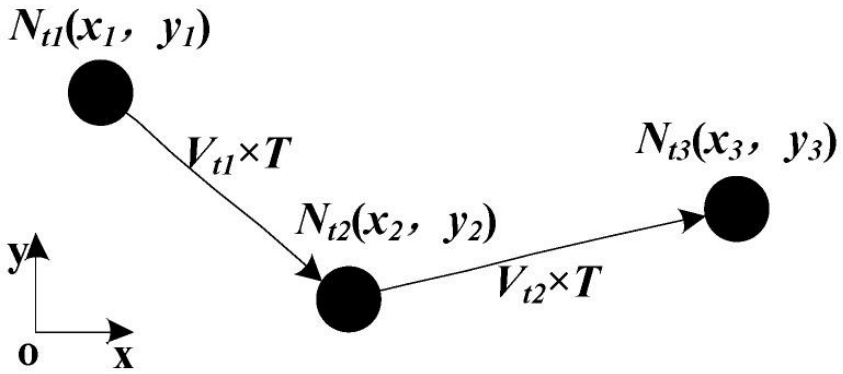

Figure 3.6: Calculation of beacon nodes velocity

As shown in Figure 3.7 the beacon node N moves from N_{t1} to N_{t2} and then to N_{t3} , and the position offsets of the two movements are V_{t1} × T and V_{t1} × T respectively. We have obtained the value of coordinatesN at t_1 and t_2 and then the velocity at $t_1$for the beacon node can be calculated by Equation

$v_x = \frac{x_2 - x_1}{T}$	$v_y = \frac{y_2 - y_1}{T}$

where *T* is the localization period, and the value of *T* may affect the localization result.

Namely, if the value of *T* is too large, the estimation of the velocity of the beacon node may be imprecise. The instantaneity of velocity for the underwater object is strong, so if we use the velocity at a certain time to represent the velocity of a period in the past, it may cause large errors. On the other hand, if the *T* value is too small, the localization algorithm running frequency will be too high, which may cause more energy consumption and computational overhead.

3.9.2 The Calculation of Unknown Sensor Nodes Velocity:

The beacon nodes broadcast a packet in the network after getting their velocities, and the packet contains the identity, velocity and time identification information. Time identification represents the moment of positioning for this node. Since all the nodes cannot complete their positioning at the same time, the referenced nodes selected should be in the same positioning round as the unknown nodes, that is to say, they should have the same time identification. The unknown node *W* receives the packets from different beacon nodes, and it will sign the beacon node which has the same time identification as a referenced node. Finally, the list of reference nodes will be set up, and it includes the identity, velocity, time identification information, and the received signal strength. The unknown node *W* can then calculate its mobile velocity with Equation 1

$\begin{cases} v_x(w) = \sum_{i=1}^{M} \sigma_{iw} v_x(i) \\ v_y(w) = \sum_{i=1}^{M} \sigma_{iw} v_y(i) \end{cases}$	(1)

Where M is the number of referenced nodes, ς_{iw} is the weight of referenced nodes, and it can be calculated by Equation

$\sigma_{iw} = \frac{r_{iw}}{\sum_{i=1}^{M} r_{iw}}$	(2)

where, r_{iw} is the signal strength received from the referenced node i by unknown node W. This method can decrease the errors of the referenced nodes which are far away from node W, and make the velocity of W more close to the real value.

Considering the beacon nodes are relatively sparse in the underwater networks, we adopt a cooperative mechanism to calculate the velocity of the unknown nodes. The unknown nodes which already get their velocities can be taken as the new reference nodes. The principle of this method is shown in Figure 3.8

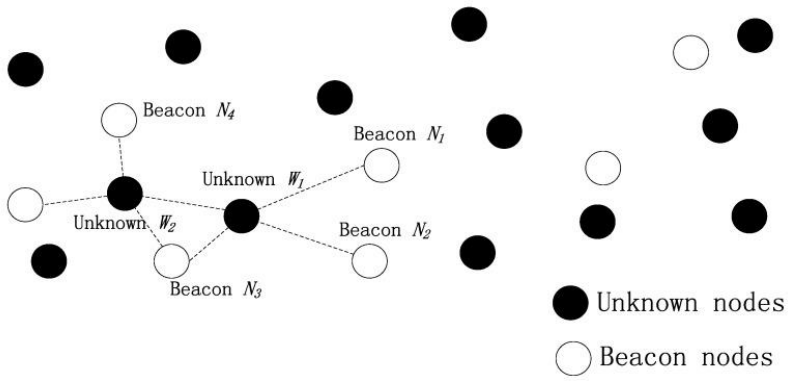

Figure 3.7: Calculation of unknown sensor nodes velocity

As shown in Figure 3.8, the unknown node W_1 calculates its velocity using beacon nodes N_1, N_2 and N_3, and then W_1 broadcasts a packet containing the identity, velocity and time identification information, so the unknown node W_2 can calculate its velocity by using beacon nodes N_3, N_4, N_5 and the unknown node W_1 as the reference nodes.

The velocities of unknown nodes have relatively more errors, especially when the new referenced nodes are far away from the unknown node, it will cause larger error accumulation. In order to further decrease the error, the unknown node will distribute a confidence coefficient to each referenced node.

We sort the referenced nodes according to their confidence coefficient from the biggest to the smallest and select the first *M* nodes of among them to participate in the velocity calculation. According to Equations (1) and (2), we can know that the number of referenced nodes can affect the velocity calculation. If the value of *M*is too small, the velocity of the unknown nodes will be calculated by only a few nodes, which cannot take full use of the spatial correlation of underwater objects' mobility. On the other hand, if the value of *M* is too large, the nodes which are far away from the unknown node will be involved in the velocity calculation, and it will cause large errors, so the value of *M* is determined by the number of all the nodes and the proportion of beacon nodes in the network. The confidence coefficient can be calculated by Equation 3

$h_{iw} = \frac{r_{iw}}{r_{wmax}} - 0.05k - 0.1r$	(3)

where $r_{iw}$is the signal strength received from reference node$i$by unknown node W, r_{wmax} is the maximum of r_{iw},k is the number of unknown nodes which are involved in the calculation of the velocity for this reference node, and r is the times that the node is taken as the reference node. As an example in Figure 3.8, the values of k and r for beacon nodes are all 0. When we calculate the velocity of W_2, the values of k and r of reference node $W_1 the$ are 0 and 1, respectively. When the node W_2 works as a reference node, the value of k is 1, and the value of r is 2.

Updating Of Unknown Sensor Nodes Location:

After calculating the unknown nodes' velocity, we can update their locations. In fact, we just need to store the velocity of each node and update the location when it is necessary.

The location can be updated with Equation 4

$\begin{cases} x' = x + \sum_{i=1}^{n} v_{xi} \,.\, T \\ y' = y + \sum_{i=1}^{n} v_{yi} \,.\, T \end{cases}$	(4)

where, T is the positioning period, and$\sum ni = 1vi.\, t$the displacement between two positioning moments.

Chapter 4

Results Analysis:

A simulation of the proposed method to determine the coordinates of submerged beacon sensors and mobile unknown sensors as described in chapter 3 was performed to verify the method. As the distance measurement between a surface buoy and beacon sensors is possible. So at first we assume the three sensors are placed at (0, 0, 0), (0, 70, 0) and (85, 90, 0) and a floating buoy moved towards one direction, assumed to be X-axis which is in a plane and that plane is parallel to the imaginary XY plane where the sensors are in 3D- space. The coordinates of the beacon sensors are randomly chosen. Z-coordinates of sensors is always kept zero to satisfy that all sensors are situated in the same plane and for computational simplicity one of the coordinates are placed at the origin. Then we have calculated the velocities of three located beacon sensors from the initial and after positions of the beacon sensors using PSO algorithm. From the velocities of beacon sensors, we have determined unknown mobile sensors which have mentioned. Eventually, we have updated the positions of these submerged mobile sensors which is our main objective of this proposed method.

This proposed method has been simulated using Matlab (v2010). To simulate the proposed method 18 datasets were taken. While calculating the true Euclidian distance from six different beacon nodes to sensors S1, S2 and S3.Then to calculate the velocities of beacon nodes we have taken the initial and after positions of these beacon nodes. After this to calculate unknown mobile nodes velocity we have calculated the weight of the reference nodes and the velocities of the beacon nodes which has discussed already. Finally, here we have taken four unknown nodes and updated their positions. The simulation result of our proposed method is given below:

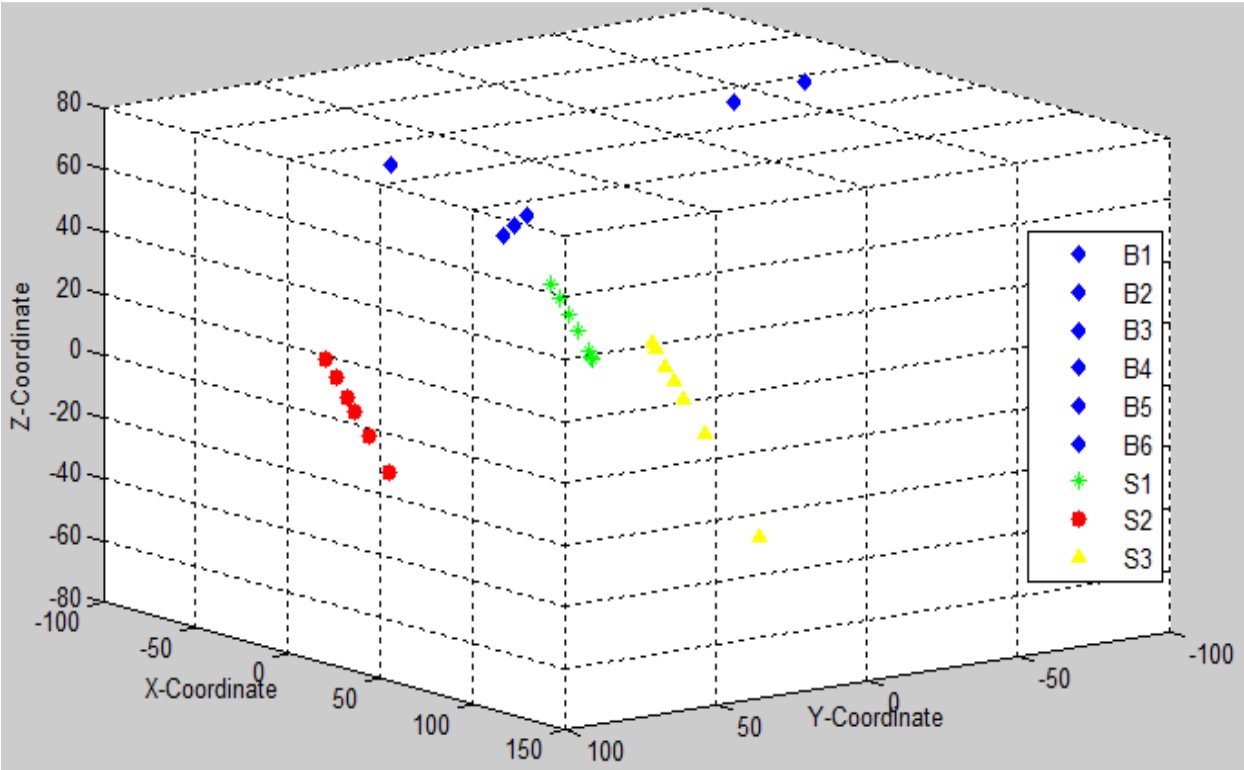

Figure 4.1: Calculated beacon sensors positions with the proposed method

To calculate the coordinates of sensors we need the inner distances between sensors S1, S2 and S3. After solving the linear equation (3) which is formed by Cayley-Menger determinant equation (2) we find the inner distances between sensors S1, S2, and S3. After that when the buoy and the beacon sensors start moving, we calculate the distance between beacon's current position to the sensors previous position using triangulation and trilateration technique. Calculating the distance we use it to find the latest coordinate of the sensor.

The values after simulation:

Beacon sensor 1

	Experimental coordinates of sensors			**Actual coordinates of sensors**		
S1	Ex	Ey	Ez	Ax	Ay	Az
	15.589	7.589	-10.410	13.034	8.154	-9.324
	22.882	14.882	-3.117	23.912	14.001	-4.523
	30.002	22.002	4.001	31.000	23.654	3.999
	-5.661	-13.661	-31.661	-6.023	-14.001	-29.672
	7.801	-0.199	-18.199	8.995	-1.312	-20.101
	-2.598	-10.598	-28.598	-3.655	-9.991	-30.001

Beacon sensor 2

S2	-26.535	37.464	-50.535	-24.886	35.990	-51.535
	-14.835	49.164	-38.835	-13.654	50.010	-37.110
	1.883	65.883	-22.116	1.999	66.997	-20.776
	10.442	74.442	-13.557	11.045	77.882	-13.661
	10.441	74.441	-13.558	11.453	75.661	-13.995
	-44.181	19.819	-68.180	-45.549	20.734	-70.000

Beacon sensor3

S3	61.256	72.256	-15.743	60.351	73.572	-14.618
	85.047	96.047	8.047	84.454	97.482	9.065
	69.459	80.459	-7.541	70.158	79.294	-8.992
	88.241	99.241	11.241	90.162	100.021	11.201
	-4.506	6.493	-81.506	-3.183	7.026	-80.472

After this, we have calculated the coordinates and updated the four unknown mobile nodes using PSO algorithm. Calculated values of the updated the positions of unknown mobile nodes:

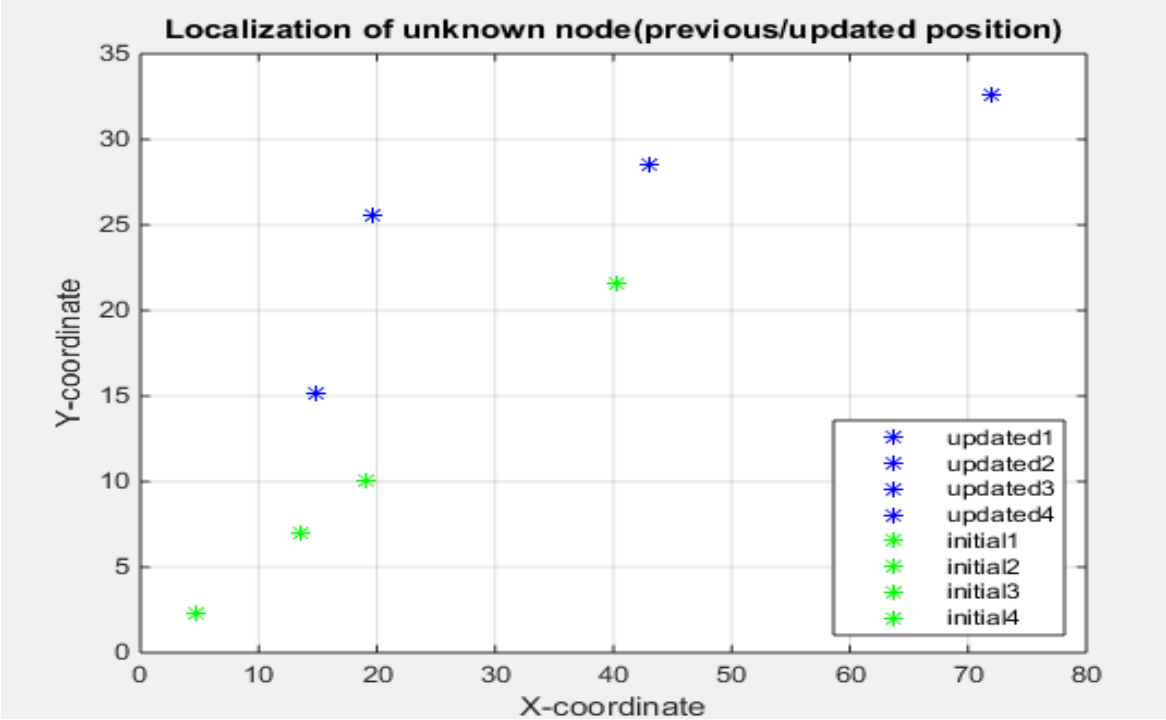

Figure: Calculation of updated mobile unknown nodes

Values after simulation:

Experimental coordinates of mobile sensors		Actual coordinates of mobile sensors		Confidence coefficient to reduce errors
X	Y	X	Y	
4.68595	2.3202	5.5	10.5	0.2481
13.58	7.0207	-7.5	11.5	0.1256
40.28	21.53	8.5	-10.5	0.0056
18.9918	10.0135	5	8.5	-0.1293

Table 4.1: Simulation result after calculation

Chapter 5

Conclusion and future work:

Focusing on the problems of underwater WSNs, like sparse deployment and mobility of the nodes, in this paper we presented a mathematical model to localize submerged mobile sensors using single buoy. Finally, simulation results validate that proposed mathematical model generates a negligible error in the coordinate determination of the sensors when distances between buoy and beacon sensors are true Euclidean distance. The simulation results show that this method can provide highly accurate localization of beacon nodes and unknown mobile nodes, and the positioning accuracy and positioning coverage rate can be kept at a better level.

While UWSN is a promising new field and may help in exploring the unfathomed world that lies underwater, there are many challenges and opportunities as well. Facing the challenges of underwater WSNs, a mathematical model to determine the coordinates of submerged sensors which are all mobile using single buoy is presented. We use multilateration, trilateration and triangulation technique to determine the location of the sensors with respect to the beacon node where the distance between them is measured considering the acoustic and radio signal. Cayley–Menger determinant and Particle Swarm Optimization algorithm is used to determining the nodes coordinates, it reduces the impact of distance measurement error on the location estimation. Simulation results validate that proposed mathematical model though it generates a negligible error in coordinate determination error on the location estimation. Simulation results validate that proposed mathematical model though it generates a negligible error in the coordinate determination of the sensors when distances between buoy and beacon sensors are true Euclidean distance. It also shows that coordinates are within the acceptable error range. Besides this using PSO algorithm finding unknown mobile nodes coordinates it may have some acceptable errors while referencing beacon nodes as the reference node.

In future work, we plan to consider more accuracy in referencing beacon nodes as reference node so that the localization of unknown nodes in random direction may have more accurate values than we have calculated. The amount of challenges in designing of UWSNs makes it an interesting area for researchers to work on. With the advancement in sensor and wireless technologies, UWSNs have attracted a lot of researchers and have contributed significantly to this field. However, the window is still wide open for upcoming research and opportunities.

References:

[1]. Anon, (2018). [online] Available at: https://en.wikipedia.org/wiki/Particle_swarm_optimization. [Accessed 5 Apr. 2018].

[2]. Swarmintelligence.org. (2018). Particle Swarm Optimization: Tutorial. [online] Available at: http://www.swarmintelligence.org/tutorials.php. [Accessed 5 Apr. 2018].

[3]. AlHajri, M.I.; Goian, A.; Darweesh, M.; AlMemari, R.; Shubair, R.M.; Weruaga, L.; Kulaib, A.R. Hybrid RSS-DOA technique for enhanced WSN localization in a correlated environment. In Proceedings of the 2015 IEEE International Conference on Information and Communication Technology Research (ICTRC), Abu Dhabi, United Arab Emirates, 17–19 May 2015; pp. 238–241.

[4] El Assaf, A.; Zaidi, S.A.R.; Affes, S.; Kandil, N. Range-free localization algorithm for heterogeneous wireless sensor networks. In Proceedings of the 2014 IEEE Wireless Communications and Networking Conference (WCNC), Istanbul, Turkey, 69 April 2014; pp. 2805–2810.

[5]. Hu, L.; Evans, D. Localization for mobile sensor network. In Proceedings of the ACM 10th annual international conference on Mobile computing and networking, Philadelphia, PA, USA, 26 September–1 October 2004; pp. 45–57.

[6]. Soltaninasab, B.; Sabaei, M.; Amiri, J. Improving Monte Carlo localization algorithm using time series forecasting method and dynamic sampling in mobile WSNs. In Proceedings of the 2010 Second International Conference on Communication Systems, Networks and Applications, Hong Kong, China, 29 June–1 July 2010; pp. 389–396.

[7]. Alcocer, A.; Oliveira, P.; Pascoal, A. Study and implementation of an EKF GIB-based underwater positioning system. Control Eng. Pract. **2007**, 15, 689–701.

[8]. Cheng, X.; Shu, H.; Liang, Q.; Du, D.H.C. Silent positioning in underwater acoustic sensor networks. IEEE Trans. Veh. Technol. **2008**, 57, 1756–1766.

[9]. Zhou, Z.; Peng, Z.; Cui, J.H.; Shi, Z.; Bagtzoglou, A.C. Scalable localization with mobility prediction for underwater sensor networks. IEEE Trans. Mob. Comput. **2011**, 10, 335–348.

[10]. Cheng, W.; Teymorian, A.Y.; Ma, L.; Cheng, X.; Lu, X.; Lu, Z. Underwater localization in sparse 3D acoustic sensor networks. In Proceedings of the 27th IEEE Conference on Computer Communications INFOCOM, Phoenix, AZ, USA, 13–18 April 2008; pp. 236–240.

[11] J. Guevara, A. R. Jimenez, A. S. Morse, J. Fang, J. C. Prieto, and F. Seco, "Auto-localization in Local Positioning Systems: A closedform range-only solution," in Industrial Electronics (ISIE), 2010 IEEE International Symposium on, 2010, pp. 2834-2840.

[12] A. R. Kulaib and R. M. Shubair, M. A. Al-Qutayri and Jason W. P. Ng, “An Overview of Localization Techniques for Wireless Sensor Networks”.

[13] A. Rahman, V. Muthukkumarasamy and E. Sithirasenan, ″Coordinates Determination of Submerged Sensors Using Cayley-Menger Determinant,″ 2013 IEEE International Conference on Distributed Computing in Sensor Systems, Cambridge, MA, 2013, pp. 466-471.

[14]. Hu, L.; Evans, D. Localization for mobile sensor network. In Proceedings of the ACM 10th annual international conference on Mobile computing and networking, Philadelphia, PA, USA, 26 September–1 October 2004; pp. 45–57.

[15]. Alcocer, A.; Oliveira, P.; Pascoal, A. Study and implementation of an EKF GIB-based underwater positioning system. Control Eng. Pract. **2007**, 15, 689–701.

[16]. [7]Y. Zhang, J. Liang, S. Jiang and W. Chen, "A Localization Method for Underwater Wireless Sensor Networks Based on Mobility Prediction and Particle Swarm Optimization Algorithms", 2018.

Appendix
Code:

% The depth of senors S1 ,S2 & S3 are same H1=H2=H3=H=60 ;

```
H=70;
```

% Beacon Nodes

```
B1 = [ 100 90 H];

B2 = [90 80 H];

B3 = [ 80 70 H];

B4 = [ -10 60 H];

B5 = [ -20 -60 H];

B6 = [ -30 -90 H];

% Sensor Coordinates

S1 = [0 0 0 ] ;

S2 = [0 70 0];

S3 = [85 90 0];
```

% from Beacon's 1st position

```
D14= pdist2(B1,S1,'euclidean');

d24 = pdist2(B1,S2,'euclidean');

d34 = pdist2(B1,S3,'euclidean');

Dist_Between_Senosrs_and_B1 = [d14 d24 d34];
```

% from Beacon's 2nd position

```
d15 = pdist2 (B2,S1,'euclidean');

d25 = pdist2 (B2,S2,'euclidean');

d35 = pdist2 (B2,S3,'euclidean');
```

```
Dist_Between_Senosrs_and_B2 = [d15 d25 d35];

% from Beacon's 3rd position

d16 = pdist2 (B3,S1,'euclidean');

d26 = pdist2 (B3,S2,'euclidean');

d36 = pdist2 (B3,S3,'euclidean');

Dist_Between_Senosrs_and_B3 = [d16 d26 d36];

% from Beacon's 4th position
d17 = pdist2 (B4,S1,'euclidean');

d27 = pdist2 (B4,S2,'euclidean');

d37 = pdist2 (B4,S3,'euclidean');

Dist_Between_Senosrs_and_B4 = [d17 d27 d37];

% from Beacon's 5th position

d18 = pdist2 (B5,S1,'euclidean');

d28 = pdist2 (B5,S2,'euclidean');

d38 = pdist2 (B5,S3,'euclidean');

Dist_Between_Senosrs_and_B5 = [d18 d28 d38];

% from Beacon's 6th position

d19 = pdist2 (B6,S1,'euclidean');

d29 = pdist2 (B6,S2,'euclidean');

d39 = pdist2 (B6,S3,'euclidean');

Dist_Between_Senosrs_and_B6 = [d19 d29 d39];

%Adding Gaussian Error

% 1st Beacon Node
```

```
d14 = d14+erf(d14);

d24 = d24+erf(d24);

d34 = d34+erf(d34);

% 2nd Beacon Node

d15 = d15+erf(d15);

d25 = d25+erf(d25);

d35 = d35+erf(d35);

% 3rd Beacon Node

d16 = d16+erf(d16);

d26 = d26+erf(d26);

d36 = d36+erf(d36);

% 4th Beacon Node

d17 = d17+erf(d17);

d27 = d27+erf(d27);

d37 = d37+erf(d37);

% 5th Beacon Node

d18 = d18+erf(d18);

d28 = d28+erf(d28);

d38 = d38+erf(d38);

% 6th Beacon Node

d19 = d19+erf(d19);

d29 = d29+erf(d29);

d39 = d39+erf(d39);
```

% From Cayley - Menger Determinant

a11=d14^2; a12=d24^2; a13=d34^2; a14=-(d14^2-d34^2)*(d24^2-d14^2); a15=-(d24^2-d14^2)*(d34^2-d24^2); a16=1; b1=(d24^2-d34^2)*(d34^2-d14^2);

a21=d15^2; a22=d25^2; a23=d35^2;a24=(-(d15^2-d35^2))*(d25^2-d15^2); a25=(-(d25^2-d15^2))*(d35^2-d25^2);a26=1; b2=(d25^2-d35^2)*(d35^2-d15^2);

a31=d16^2; a32=d26^2; a33=d36^2;a34=-(d16^2-d36^2)*(d26^2-d16^2);a35=-(d26^2-d16^2)*(d36^2-d26^2); a36=1;b3=(d26^2-d36^2)*(d36^2-d16^2);

a41=d17^2; a42=d27^2; a43=d37^2;a44=-(d17^2-d37^2)*(d27^2-d17^2);a45=-(d27^2-d17^2)*(d37^2-d27^2); a46=1;b4=(d27^2-d37^2)*(d37^2-d17^2);

a51=d18^2; a52=d28^2; a53=d38^2; a54=-(d18^2-d38^2)*(d28^2-d18^2);a55=-(d28^2-d18^2)*(d38^2-d28^2); a56=1;b5=(d28^2-d38^2)*(d38^2-d18^2);

a61=d19^2; a62=d29^2; a63=d39^2;a64=-(d19^2-d39^2)*(d29^2-d19^2); a65=-(d29^2-d19^2)*(d39^2-d29^2);a66=1; b6=(d29^2-d39^2)*(d39^2-d19^2);

% Augmented Matrix

A = [a11 a12 a13 a14 a15 a16

a21 a22 a23 a24 a25 a26

a31 a32 a33 a34 a35 a36

a41 a42 a43 a44 a45 a46

a51 a52 a53 a54 a55 a56

a61 a62 a63 a64 a65 a66];

Matrix = A;

cond = cond(A);

% Result

B = [b1

b2

```
b3

b4
b5
b6];
% value of x
x = A\B;
% x = pinv(A)*B;

% Unknown Inner Distances Between Sensors

% Distance Between S1 & S2

d12 = sqrt(x(3)/(1-x(4)-x(5)));

% Distance Between S1 & S3

d13 = sqrt((x(3)*x(5))/(1-x(4)-x(5)));

% Distance Between S2 & S3

d23 = sqrt((x(3)*x(4))/(1-x(4)-x(5)));

Dist_Between_Senosrs_and_B1 = [d14 d24 d34];

% Final Coordinates Respect of Submerged Sensors S1 , S2 , S3

y2 = d12;

y3 = (d12^2+d13^2-d23^2)/(2*d12);

x3 = sqrt(d13^2-((d12^2+d13^2-d23^2)/(2*d12))^2);

% Matrix Representation of Sensors

S = [ 0 0 0

0 y2 0

x3 y3 0];
% Coordinates of the sensors with respect to the Beacon

% Distance Between Projected Beacon's coordinates P4 and Sensors Using Pythagorean Theorem
D14 = sqrt(d14^2-H^2);
```

```
D24 = sqrt(d24^2-H^2);
D34 = sqrt(d34^2-H^2);
Dist_Between_Sensors_and_P4 = [D14 D24 D34];
% Coordinates of Beacon

y4 = (D14^2 - D24^2 + d12^2)/(2*d12);

x4 = sqrt(D14^2 - y4^2);

% Projected Coordinates of Beacon -> P4

P4 = [x4 y4 0];

% Coordinates of Beacon B1

z4=H;

S4 = [x4 y4 z4];
% After Cartesians Transformation, Sensors Coordinates with respect to Beacon

fS4 = [0 0 0]; % Considering Beacon position on Origin

fS1 = [-x4 -y4 -z4]; % Coordinates of S1 respect of Beacon

fS2 = [-x4 (y2-y4) -z4]; % Coordinates of S2 respect of Beacon

fS3 = [(x3-x4) (y3-y4) -z4]; % Coordinates of S3 respect of Beacon
Final_Sensors_Coordinates = [ fS4
fS1
fS2
fS3];

% Distance between 1st Beacon Node and Calculated Sensors Coordinates

Cal_d14 = pdist2(fS4,fS1,'euclidean');

Cal_d24 = pdist2(fS4,fS2,'euclidean');

Cal_d34 = pdist2(fS4,fS3,'euclidean');

Cal_Dist_Between_Sensors_and_B1 = [Cal_d14 Cal_d24 Cal_d34];

%Part: 2 :

% distance from beacon's current position to 1st sensor's previous position on X-Axis
```

```
a1=50 ;a2=60 ;a3=70; a4=80; a5=90; a6=100;

b1=70;b2=70;b3=70; b4=70;b5=70;b6=70;

c1=80 ;c2=90 ;c3=100;c4=71;c5=88;c6=83;

A1 = ((a1^2-b1^2-c1^2)/(-2*b1*c1));

A2 = ((a2^2-b2^2-c2^2)/(-2*b2*c2));

A3 = ((a3^2-b3^2-c3^2)/(-2*b3*c3));

A4 = ((a4^2-b4^2-c4^2)/(-2*b4*c4));

A5 = ((a5^2-b5^2-c5^2)/(-2*b5*c5));

A6 = ((a6^2-b6^2-c6^2)/(-2*b6*c6));

AB1=acosd(A1);

ang1=90-AB1;

AB1=acosd(A1);

ang1=90-AB1;

AB2=acosd(A2);

ang2=90-AB2;

AB3=acosd(A3);

ang3=90-AB3;

AB4=acosd(A4);

ang4=90-AB4;

AB5=acosd(A5);

ang5=90-AB5;

AB6=acosd(A6);

ang6=90-AB6;
```

```
q1=80 ;q2=90 ;q3=100;q4=71;q5=88;q6=83;

r1=4;r2=8;r3=12;r4=16;r5=20;r6=24;
distp1=sqrt(q1^2+r1^2-2*q1*r1*cosd(ang1));

distp2=sqrt(q2^2+r2^2-2*q2*r2*cosd(ang2));

distp3=sqrt(q3^2+r3^2-2*q3*r3*cosd(ang3));

distp5=sqrt(q5^2+r5^2-2*q5*r5*cosd(ang5));

distp4=sqrt(q4^2+r4^2-2*q4*r4*cosd(ang4));

distp6=sqrt(q6^2+r6^2-2*q6*r6*cosd(ang6));

% distance from beacon's current position to 2nd sensor's previous position on Y-axis

xxx1=40 ;xxx2=45 ;xxx3=55; xxx4=58; xxx5=67;xxx6=80;
yyy1=70;yyy2=70;yyy3=70; yyy4=70;yyy5=70;yyy6=70;
zzz1=40 ;zzz2=56 ;zzz3=77;zzz4=88;zzz5=93;zzz6=54;

X1 = acosd((xxx1^2-yyy1^2-zzz1^2)/(-2*yyy1*zzz1));

X2 = acosd((xxx2^2-yyy2^2-zzz2^2)/(-2*yyy2*zzz2));

X3 = acosd((xxx3^2-yyy3^2-zzz3^2)/(-2*yyy3*zzz3));

X5 = acosd((xxx5^2-yyy5^2-zzz5^2)/(-2*yyy5*zzz5));

X4 = acosd((xxx4^2-yyy4^2-zzz4^2)/(-2*yyy4*zzz4));

X6 = acosd((xxx6^2-yyy6^2-zzz6^2)/(-2*yyy6*zzz6));

Yang1=90-X1;

Yang2=90-X2;

Yang3=90-X3;

Yang4=90-X4;

Yang5=90-X5;

Yang6=90-X6;

zz1=40 ;zz2=56 ;zz3=77;zz4=88;zz5=93;zz6=54;
```

```
yy1=6;yy2=12 ;yy3=18; yy4=24;yy5=30;yy6=36;

distxx1=sqrt(yy1^2+zz1^2-2*yy1*zz1*cosd(Yang1));

distxx2=sqrt(yy2^2+zz2^2-2*yy2*zz2*cosd(Yang2));

distxx3=sqrt(yy3^2+zz3^2-2*yy3*zz3*cosd(Yang3));

distxx4=sqrt(yy4^2+zz4^2-2*yy4*zz4*cosd(Yang4));

distxx5=sqrt(yy5^2+zz5^2-2*yy5*zz5*cosd(Yang5));

distxx6=sqrt(yy6^2+zz6^2-2*yy6*zz6*cosd(Yang6));
```
% distance from beacon's current position to 2nd sensor's previous position on out-of-axis
```
f1=34;f2=45;f3=55;f4=65;f5=76;f6=91;

g1=70;g2=70;g3=70;g4=70;g5=70;g6=70;

h1=90;h2=80;h3=77;h4=98;h5=80;h6=50;

F1= acosd((f1^2-g1^2-h1^2)/(-2*g1*h1));

F2= acosd((f2^2-g2^2-h2^2)/(-2*g2*h2));

F3= acosd((f3^2-g3^2-h3^2)/(-2*g3*h3));

F4= acosd((f4^2-g4^2-h4^2)/(-2*g4*h4));

F5= acosd((f5^2-g5^2-h5^2)/(-2*g5*h5));

F6= acosd((f6^2-g6^2-h6^2)/(-2*g6*h6));
Nang1=90-F1;

Nang2=90-F2;

Nang3=90-F3;

Nang4=90-F4;

Nang5=90-F5;

Nang6=90-F6;

hh1=90;hh2=80;hh3=77;hh4=98;hh5=80;hh6=50;
```

```
gg1=8;gg2=16;gg3=24;gg4=32;gg5=40;gg6=48;

distff1=sqrt(hh1^2+gg1^2-2*hh1*gg1*cosd(Nang1));
distff2=sqrt(hh2^2+gg2^2-2*hh2*gg2*cosd(Nang2));

distff3=sqrt(hh3^2+gg3^2-2*hh3*gg3*cosd(Nang3));
distff3=sqrt(hh3^2+gg3^2-2*hh3*gg3*cos(Nang3));

distff4=sqrt(hh4^2+gg4^2-2*hh4*gg4*cosd(Nang4));

distff5=sqrt(hh5^2+gg5^2-2*hh5*gg5*cos(Nang5));

distff6=sqrt(hh6^2+gg6^2-2*hh6*gg6*cosd(Nang6));
```

% After Adding Distance coordinates of sensors with respect to beacon

%Coordinates of S1 with respect to Beacon

```
FS1_1 = [-x4+distp1-4 -y4+ distp1 -z4+ distp1];

FS1_2 = [-x4+distp2-4 -y4+ distp2 -z4+ distp2];

FS1_3 = [-x4+distp3-4 -y4+ distp3 -z4+ distp3];

FS1_4 = [-x4+distp4-4 -y4+ distp4 -z4+ distp4];

FS1_5 = [-x4+distp5-4 -y4+ distp5 -z4+ distp5];

FS1_6 = [-x4+distp6-4 -y4+ distp6 -z4+ distp6];
```

%Coordinates of S2 with respect to Beacon

```
FS2_1 = [-x4+distxx1-6 (y2-y4)+distxx1 -z4+distxx1];

FS2_2 = [-x4+distxx2-6 (y2-y4)+distxx2 -z4+distxx2];

FS2_3 = [-x4+distxx3-6 (y2-y4)+distxx3 -z4+distxx3];

FS2_4 = [-x4+distxx4-6 (y2-y4)+distxx4 -z4+distxx4];

FS2_5 = [-x4+distxx5-6 (y2-y4)+distxx5 -z4+distxx5];

FS2_6 = [-x4+distxx6-6 (y2-y4)+distxx6 -z4+distxx6];
```

%Coordinates of S3 with respect to Beacon

```
FS3_1 = [(x3-x4)+ distff1-8 (y3-y4)+ distff1 -z4+ distff1];
```

```
FS3_2 = [(x3-x4)+ distff2-8 (y3-y4)+ distff2 -z4+ distff2];

FS3_3 = [(x3-x4)+ distff3-8 (y3-y4)+ distff3 -z4+ distff3];

FS3_4 = [(x3-x4)+ distff4-8 (y3-y4)+ distff4 -z4+ distff4];

FS3_5 = [(x3-x4)+ distff5-8 (y3-y4)+ distff5 -z4+ distff5];

FS3_6 = [(x3-x4)+ distff6-8 (y3-y4)+ distff6 -z4+ distff6];

x1 = 0;

y1 = 0;

z1 = 0;

x2 = 0;

z2 = 0;

z3 = 0;
```

%%%for figure

```
figure,

scatter3(100,90,70,' d','filled', 'b');

hold on, scatter3(90,80,70,'d','filled', 'b');

hold on, scatter3(80,70,70,'d','filled', 'b');

hold on, scatter3(-10,60,70,'d','filled', 'b');

hold on, scatter3(-20,-60,70,'d','filled', 'b');

hold on, scatter3(-30,-90,70,'d','filled', 'b');

hold on, scatter3(x1,y1,z1,'*', 'g');

hold on; scatter3(x2,y2,z2,'o','filled','r');
hold on; scatter3(x3,y3,z3,'^','filled','y');

legend('B1','B2','B3','B4','B5','B6','S1','S2','S3');
```

%figure for S1 sensor with respect to Beacon in its mobile form

```
hold on, scatter3(-x4+distp1-4,-y4+ distp1,-z4+ distp1,'*','g');
hold on, scatter3(-x4+distp2-4,-y4+ distp2,-z4+ distp2,'*','g');
hold on, scatter3(-x4+distp3-4,-y4+ distp3,-z4+ distp3,'*','g');
hold on, scatter3(-x4+distp4-4,-y4+ distp4,-z4+ distp4,'*','g');
hold on, scatter3(-x4+distp5-4,-y4+ distp5,-z4+ distp5,'*','g');
hold on, scatter3(-x4+distp6-4,-y4+ distp6,-z4+ distp6,'*','g');
```

%figure for S2 sensor with respect to Beacon in its mobile form

```
hold on, scatter3(-x4+distxx1-6,(y2-y4)+distxx1,-z4+distxx1,'o','filled','r');
hold on, scatter3(-x4+distxx2-6,(y2-y4)+distxx2,-z4+distxx2,'o','filled','r');
hold on, scatter3(-x4+distxx3-6,(y2-y4)+distxx3,-z4+distxx3,'o','filled','r');
hold on, scatter3(-x4+distxx4-6,(y2-y4)+distxx4,-z4+distxx4,'o','filled','r');
hold on, scatter3(-x4+distxx5-6,(y2-y4)+distxx5,-z4+distxx5,'o','filled','r');
hold on, scatter3(-x4+distxx6-6,(y2-y4)+distxx6,-z4+distxx6,'o','filled','r');
```

%figure for S3 sensor with respect to Beacon in its mobile form

```
hold on, scatter3((x3-x4)+ distff1-8,(y3-y4)+ distff1,-z4+ distff1,'^','filled','y');
hold on, scatter3((x3-x4)+ distff2-8,(y3-y4)+ distff2,-z4+ distff2,'^','filled','y');
hold on, scatter3((x3-x4)+ distff3-8,(y3-y4)+ distff3,-z4+ distff3,'^','filled','y');
hold on, scatter3((x3-x4)+ distff4-8,(y3-y4)+ distff4,-z4+ distff4,'^','filled','y');
hold on, scatter3((x3-x4)+ distff5-8,(y3-y4)+ distff5,-z4+ distff5,'^','filled','y');
hold on, scatter3((x3-x4)+ distff6-8,(y3-y4)+ distff6,-z4+ distff6,'^','filled','y');
```

%figure for S1 sensor with respect to Beacon in its mobile form with error

```
figure,
hold on, scatter3(19.603, -6.929, 11.647, 'o','r')
```

```
hold on, scatter3(-4.307, 6.292, -31.086, 'o','r')

hold on, scatter3(-3.2, -6.984, 0.050, 'o','r')

hold on, scatter3(-6.010, -2.428, 6.703, 'o','r')

hold on, scatter3(-13.274, -84.832, -9.462, 'o','r')

hold on, scatter3(-28.919, 6.075, -4.677, 'o','r')

hold on, scatter3(-x4+distp1-4,-y4+ distp1,-z4+ distp1,'*','g');

hold on, scatter3(-x4+distp2-4,-y4+ distp2,-z4+ distp2,'*','g');

hold on, scatter3(-x4+distp3-4,-y4+ distp3,-z4+ distp3,'*','g');

hold on, scatter3(-x4+distp4-4,-y4+ distp4,-z4+ distp4,'*','g');

hold on, scatter3(-x4+distp5-4,-y4+ distp5,-z4+ distp5,'*','g');

hold on, scatter3(-x4+distp6-4,-y4+ distp6,-z4+ distp6,'*','g');
legend('Error of S1');
```

%figure for S2 sensor with respect to Beacon in its mobile form with error

```
figure,

hold on, scatter3(6.626, 4.096, -1.940,'o','filled','r');
hold on, scatter3(8.649, -1.692, 4.648, 'o','filled','r');

hold on, scatter3(-5.803, -1.663, 5.883, 'o','filled','r');

hold on, scatter3(-5.459, -1.612, -3.123, 'o','filled','r');

hold on, scatter3(-8.836, -1.614,-3.123, 'o','filled','r');

hold on, scatter3(-3.003, -4.413, -2.6, 'o','filled','r');

hold on, scatter3(-x4+distxx1-6,(y2-y4)+distxx1,-z4+distxx1,'^','filled','b')

hold on, scatter3(-x4+distxx2-6,(y2-y4)+distxx2,-z4+distxx2,'^','filled','b')

hold on, scatter3(-x4+distxx3-6,(y2-y4)+distxx3,-z4+distxx3,'^','filled','b')

hold on, scatter3(-x4+distxx4-6,(y2-y4)+distxx4,-z4+distxx4,'^','filled','b')
```

```
hold on, scatter3(-x4+distxx5-6,(y2-y4)+distxx5,-z4+distxx5,'^','filled','b')

hold on, scatter3(-x4+distxx6-6,(y2-y4)+distxx6,-z4+distxx6,'^','filled','b')

legend('Error of S2');
```

%figure for S3 sensor with respect to Beacon in its mobile form with error

```
figure,

hold on, scatter3(-0.544, -0.941, -46.649, 'o','filled','r')

hold on, scatter3(1.499, -1.789, 7.696, 'o','filled','r')

hold on, scatter3(0.702, -1.472, -11.230, 'o','filled','r')

hold on, scatter3(-0.996, 1.469, -16.137, 'o','filled','r')

hold on, scatter3(-2.131, -0.779, 0.357, 'o','filled','r')

hold on, scatter3(41.565, -7.586, 1.285, 'o','filled','r')

hold on, scatter3((x3-x4)+ distff1-8,(y3-y4)+ distff1,-z4+ distff1,'^','filled','y');

hold on, scatter3((x3-x4)+ distff2-8,(y3-y4)+ distff2,-z4+ distff2,'^','filled','y');

hold on, scatter3((x3-x4)+ distff3-8,(y3-y4)+ distff3,-z4+ distff3,'^','filled','y');

hold on, scatter3((x3-x4)+ distff4-8,(y3-y4)+ distff4,-z4+ distff4,'^','filled','y');

hold on, scatter3((x3-x4)+ distff5-8,(y3-y4)+ distff5,-z4+ distff5,'^','filled','y');

hold on, scatter3((x3-x4)+ distff6-8,(y3-y4)+ distff6,-z4+ distff6,'^','filled','y');

legend('Error of S3');
```

%Part :3:

```
t=1;
```

%% Sensor 1

```
A=[13.034 23.912 31.000 -6.023 8.885 -3.655];

B=[15.589 22.882 30.002 -5.661 7.801 -2.598];

C=[8.154 14.001 23.654 -14.001 -1.312 -9.991];
```

D=[7.589 14.882 22.002 -13.661 -0.199 -10.598];

Vx_sensor_1=((B-A)/t);

Vy_sensor_1=((D-C)/t);

%% Sensor 2

A=[77.012 60.351 84.454 70.158 90.162 -3.183];

B=[76.593 61.256 85.047 69.459 88.241 -4.506];

C=[88.425 73.572 97.482 79.294 100.021 7.026];

D=[87.593 72.256 96.047 80.459 99.241 6.493];

Vy_sensor_2=((D-C)/t);

Vx_sensor_2=((B-A)/t);

%% Sensor 3

A=[77.012 60.351 84.454 70.158 90.162 -3.183];

B=[76.61 61.256 85.047 89.459 88.241 -4.506];

C=[77.012 73.572 97.482 78.294 100.021 -7.026];

D=[76.593 72.245 96.047 80.459 99.241 6.493];

Vx_sensor_3=((B-A)/t);

Vy_sensor_3=((D-C)/t);

%% unknown Node 1

ans_iw_1 = 30/(33 + 30 + 34);

Vx_unknown_node_1 = sum(ans_iw_1 * Vx_sensor_1 + ans_iw_1 * Vx_sensor_2 + ans_iw_1 * Vx_sensor_3);

```
Vy_unknown_node_1 = sum(ans_iw_1 * Vy_sensor_1 + ans_iw_1 * Vy_sensor_2 + ans_iw_1 * Vy_sensor_3);

%% unknown Node 2

ans_iw_2 = 33/(33 + 34 + 36);

Vx_unknown_node_2 = sum(ans_iw_2 * Vx_unknown_node_1 + ans_iw_2 * Vx_sensor_2 + ans_iw_2 * Vx_sensor_3 );

Vy_unknown_node_2 = sum(ans_iw_2 * Vy_unknown_node_1 + ans_iw_2 * Vy_sensor_2 + ans_iw_2 * Vy_sensor_3 );

%% unknown Node 3

ans_iw_3 = 34/(37 + 34 + 36);

Vx_unknown_node_3 = sum(ans_iw_3 * Vx_unknown_node_2 + ans_iw_3 * Vx_unknown_node_1  + ans_iw_3 * Vx_sensor_3);

Vy_unknown_node_3 = sum(ans_iw_3 * Vy_unknown_node_2 + ans_iw_3 * Vy_unknown_node_1  + ans_iw_3 * Vy_sensor_3);

%% unknown Node 4

ans_iw_4 = 36/(37 + 36 + 38);

Vx_unknown_node_4 = sum(ans_iw_4 * Vx_unknown_node_3 + ans_iw_4 * Vx_unknown_node_2 + ans_iw_4 * Vx_unknown_node_1 );

Vy_unknown_node_4 = sum(ans_iw_4 * Vy_unknown_node_3 + ans_iw_4 * Vy_unknown_node_2 + ans_iw_4 * Vy_unknown_node_1  );


%% Plot of the data

%plot(Vx_unknown_node_1,Vy_unknown_node_1,'*');

%figure

%plot(Vx_unknown_node_2,Vy_unknown_node_2,'*');
```

```
%figure
%plot(Vx_unknown_node_3,Vy_unknown_node_3,'*');
%figure
%plot(Vx_unknown_node_4,Vy_unknown_node_4,'*');
```

%% Show value

```
Vx_unknown_node_1
Vy_unknown_node_1
Vx_unknown_node_2
Vy_unknown_node_2
Vx_unknown_node_3
Vy_unknown_node_3
Vx_unknown_node_4
Vy_unknown_node_4
```

%% unknown node location

```
position_periode_t = 2;


node_value_x_1 = 5.5;
node_value_x_2 = -7.5;
node_value_x_3 = -8.5;
node_value_x_4 = 5;
node_value_y_1 = 10.5;
node_value_y_2 = 11.5;
node_value_y_3 = -10.5;
```

```
node_value_y_4 = 8.5;
updated_position_x_1 = node_value_x_1 + Vx_unknown_node_1 * position_periode_t ;
updated_position_x_2 = node_value_x_2 + Vx_unknown_node_2 * position_periode_t ;
updated_position_x_3 = node_value_x_3 + Vx_unknown_node_3 * position_periode_t ;
updated_position_x_4 = node_value_x_4 + Vx_unknown_node_4 * position_periode_t ;

updated_position_y_1 = node_value_y_1 + Vy_unknown_node_1 * position_periode_t ;
updated_position_y_2 = node_value_y_2 + Vy_unknown_node_2 * position_periode_t ;
updated_position_y_3 = node_value_y_3 + Vy_unknown_node_3 * position_periode_t ;
updated_position_y_4 = node_value_y_4 + Vy_unknown_node_4 * position_periode_t ;



plot(updated_position_x_1,upadeted_position_y_1,'*b');
hold on
plot(updated_position_x_2,upadeted_position_y_2,'*b');
plot(updated_position_x_3,upadeted_position_y_3,'*b');
plot(updated_position_x_4,upadeted_position_y_4,'*b');

%% Plot of the data on unknown Var
hold on
plot(Vx_unknown_node_1,Vy_unknown_node_1,'*g');
plot(Vx_unknown_node_2,Vy_unknown_node_2,'*g');
plot(Vx_unknown_node_3,Vy_unknown_node_3,'*g');
```

```
plot(Vx_unknown_node_4,Vy_unknown_node_4,'*g');
title('Localization of unknown node(previous/updated position)')
grid on
```