CONCEPTUAL DESIGN REPORT

Autonomous Map Extraction Robot

To: Assoc. Prof. Dr. Lale Alatan
Middle East Technical University
Electrical and Electronics Department C-209
+90 312 210 2346

Company Owners:

Mr. S. Emre Can	+90 554 927 26 20
Ms. Ekin Yılmaz	+90 538 310 50 01
Mr. Tahir M. Çimen	+90 539 887 75 37
Ms. Cansu Demirkıran	+90 538 546 77 79
Mr. Yekta Demirci	+90 535 029 07 94

Project Starting Date:	03.12.2018
Project End Date:	02.06.2019
Overall Duration:	6 Months
Estimated Cost:	167$

Table of Contents

1. EXECUTIVE SUMMARY

Exponentially growing population needs and the lack of natural sources brought people to explore the untouched. Since it is not an option to send a human being to unexplored locations where no one has any opinion about the possible dangers and the required time, the best solution is using robots for such missions. Although automated mapping and localization is a trending topic where there are many options and solutions, we are providing more efficient and larger scale solutions to the search for explorations, which involves larger opportunities.

As Thunderbolt Corporation, we are focusing on space exploration missions. We are developing a mapping platform to achieve zero fatalities on space missions with significant reduction on costs. Robots carrying this platform will be sent to its mission location to explore the area autonomously, create a detailed map, and send the final version to the base. As one can guess at this point, this platform has to be(and will be) light, compact, fast, accurate, robust and user friendly to endorse independent researches.

To achieve reliable operation for different environments, our product always will be aware of its surrounding by taking full rotation scans and observe its surrounding objects in two dimensions. While observing its surrounding, it will locate itself by using its own movements and observations in a continuous cycle of algorithms; therefore, product will easily understand where does it stands and location of its surrounding. Product will also be able to cover every part of the mapping area in an efficient way by planning its path with an landmark based path planning algorithm. After scanning the whole mapping area, an accurate map containing location, orientation, type and the center position information for each object will be created by identifying each landmark with fitting algorithms, and as final outcome, robot will send this map automatically to the client by over wi-fi connection.

As a team of engineers with different specializations, we will overcome this challenges by distributing the tasks to competent engineers in a dynamic teamwork environment, where each challenge will be approached under microscope of engineers specialised in related areas.

During the following 6 months, Thunderbolt Corp. will develop a prototype and test all the requirements of this project on a simulation landscape on earth. The final product will be

served with a software including interface for transferring map from robot, a battery charger, two years warranty, and a demo environment. Final product will fit in a 25cm diameter cylinder with and have estimated cost lower than 200$, which will ease the process of launching robots to space and provide the customer efficient solutions.

2. INTRODUCTION

Autonomous robots that are aware of their surroundings is a basic building block for almost every self-controlled robot in robotics applications these days. This awareness provides the autonomicity to robots. As Thunderbolt Corp. being aware of the importance of this concept, we are determined to work on a project such as “Autonomous robot trying to extract the plan of its surroundings project” and as a developing robotics company we offer many solutions to this manner. Although the field is competitive, as Thunderbolt Corp. we offer the best quality and efficiency to our customers thanks to our highly experienced and ambitious team.

Simultaneous localization and mapping is also a concept which solves a very important problems in mobile robotics recently. For self-controlled i.e autonomous robots extracting the plan of its surroundings, being aware of the environment it is very crucial. This application can be considered separately i.e robots for mapping purposes only, or it can also be considered as a sub-unit of any autonomous robot too such as in the example of driverless car technology. It is a concept which can be integrated in so many different areas and can provide various solutions for commonly faced engineering problems. As Thunderbolt Corp. we are proud to contribute in such an important engineering subject.

This project basically provides an autonomous device which can extract the 2D map of its environment by exploring it autonomously and display it on the screen. The device will be autonomous which brings several advantages to the user. It allows the robot to navigate the field by itself without needing any human interaction. Our project suggests a solution for the discovery and mapping of the problematic places where human access is not possible or not desired.

This conceptual design report is a detailed explanation of the project that provides overall system operation with our company’s offered solutions. All sub-system interactions and interfaces are provided with their requirements and related solutions, are also given.

Necessary back up plans are also provided considering possible future obstacles to provide our users a better service and product. Test results obtained so far and module demonstration test results are also declared and evaluated. Alternative solutions considering practical applications are presented.

In addition, future time schedule with subsystem and overall system testing plans are provided with complete system integration plans towards the end product.

One can find any information needed about the problem to be solved, the solution offers, previous test results and future testing plans and the robot's overall conceptual design in this report. The path to be followed towards our company's end product is clearly stated in following sections.

3. PROBLEM STATEMENT

The project which its conceptual design will be explained in this report, is to make an autonomous map extraction robot and during the making process of the robot the problems that are needed to be solved will be clearly examined in this part.

Firstly, to be able to create a map from many different data defining its surroundings robot must be aware of the self location of itself to combine and match this data. Secondly, considering the fact that walls and objects are arbitrarily placed robot must move in the field without crashing into any of the walls and objects. Also, It must give the distance and location data with minimum error. Robot must go to the objects and the walls close enough to take accurate data and must take several data for each object for the accuracy of the map. It must not stop until scanning of all of the objects and walls is completed and also it must not scan the same object twice. To achieve these a proper path decision algorithm must be used. The starting angle of the sensor must not be change to prevent localization errors. After combining all the data, it must classify the object types and centers. Borders of the field and the objects are also needed to be identified and shown accurately by using an proper visualization algorithm. The final version of the output must be visible to the user.

4. SOLUTION PROCEDURES AND DEVELOPED METHODOLOGIES

Thunderbolt proposes a remarkable solution for the problems stated earlier in the problem statement part. The scanning process will be handled by Time of Flight distance sensor and precise moving in the field and approaching to the objects will be handled by the ultrasonic sensors. Before and after each scanning, robot will hold the location information that it currently has to compare the data with the old ones. Using omni-wheels enables the robot to keep its direction as in the beginning so that the front of the robot will show the same direction all the time to minimize the location errors that can stem from the motion in the field.

Before scanning the area robot will hold the data of the starting point at which it will be placed first. Then it will begin to scan the environment to explore the area. By using the data from scanning process it will check whether there are possible objects in the field or not. If it sees any objects, it will compare the distances between the objects and itself separately and find the nearest object. After finding the object, it will go towards that object until it reaches the minimum distance limit so that it can start to collect data. When it is finally at a point close enough to the object it will stop the motion, save the self location and start to collect data by using the ToF distance sensor with 360 degrees scanning.

The scanning process of an object will continue until the object is completely finished. In the case that there are no obstacles around the object, robot will collect data at the points 90 degrees rotated with respect to the previous location in the clockwise direction. If not, path decision algorithm will allow the robot to move around the objects properly so that if the robot encounters an obstacle while trying to complete the object, it will find the possible best locations to collect data and complete the object.

After one object is finished, it will be marked as “visited” and robot will find the nearest object that is not visited yet. If two objects are detected at the same distance from the robot, the algorithm will check the other objects and the robot will go towards the object which is closer to the other objects. When all the objects in the field are completed, robot will check whether walls are completed or not. If there are uncompleted walls in the map data robot will go to these missing walls and collect data to complete the map.

After all objects and walls in the field are completed, visualization algorithm will convert the point cloud data into the lines and circles to create the actual map image and also it will classify the objects and find their desired properties (diameter, center, lengths of edges). All visualization algorithms and the path decision, object classification, data collection, self localization algorithms will be run on Raspberry Pi. When creation of the map is completed, map image with the object classification and properties will sent to a computer via WiFi by using Raspberry Pi again.

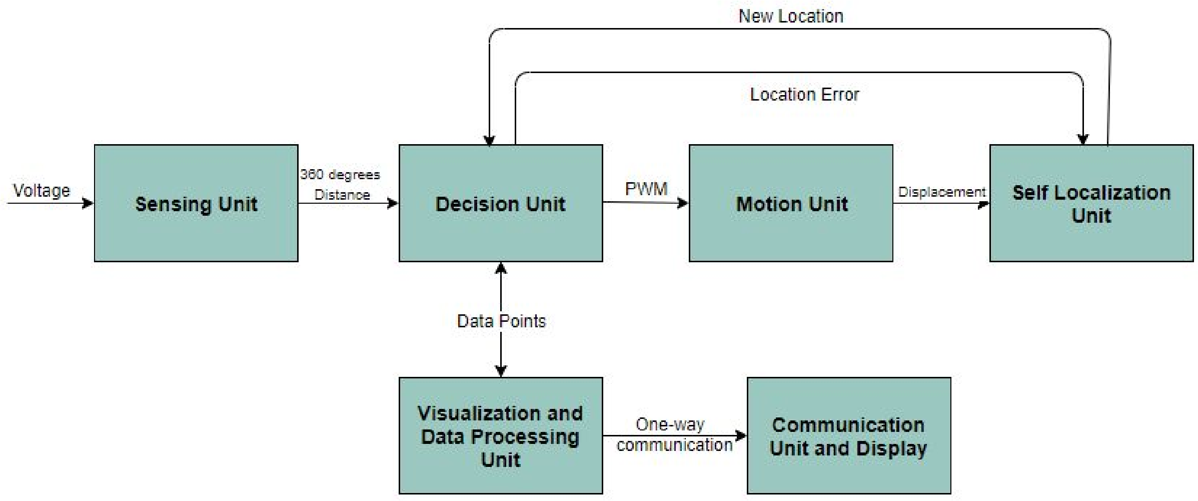

Diagram I: Input-Output relation between subsystems

System level flowchart of main decision logic is provided in Figure A1-1 in Appendices section under Appendix 1.

4.1. SENSING UNIT

For the robot to be aware of the surroundings, first of all, it should get distance information from the objects and the walls. To get this data, several types of distance sensors and several methods of sensing were considered. At the end, it was decided to use two types of sensors for different purposes which are mainly obtaining the positions of possible objects and secondly following a safe path without hitting the objects considering the size of the robot.

First aim of the robot is to detect objects and decide the direction to move. For this purpose, it was needed to get data from all directions (360 degrees view). The distance range and accuracy were the most important factors about the choice. After our research, it was decided to use VL53L0X sensor which uses time-of-flight concept to measure distance. In other words, by propagating infrared pulses and measuring the time between sending the pulse and receiving it after it is reflected from the object, the sensor can measure the distance up to 2m [1]. It also has an Arduino library and Raspberry Pi application programming interface which will ease the procedure of using it.

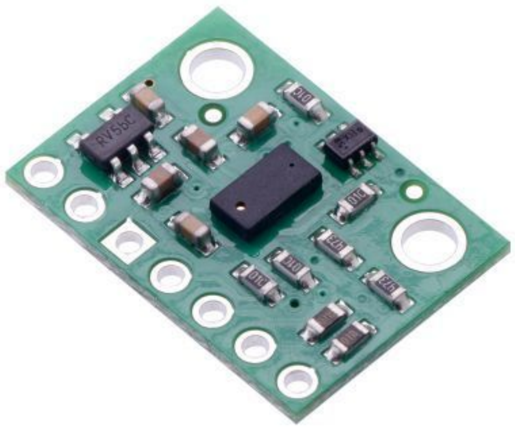

Figure 1: VL53L0X sensor

To get data from all directions, the sensor will be used with a stepper motor. The aim of the stepper motor is to maintain the accuracy while changing the direction of the sensor. The used motor has 200 steps. In every step of the motor, a distance information will be obtained which means one data for each 1.8 degrees.

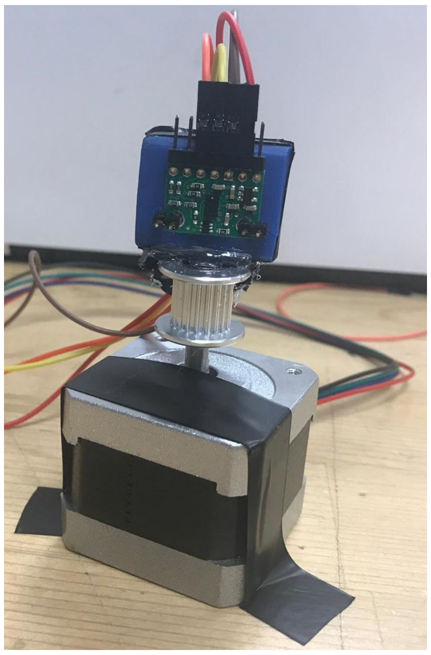

Figure 2: VL53L0X sensor with step motor

With VL53L0X, the data will be obtained only when the robot is stationary. An alternative approach is also proposed such as collecting data while moving if more accuracy is needed.

Since the sensor will not get the distance information continuously, another set of sensors are needed for keeping a margin between the robot and the surroundings. They will get measurements non-stop, enable the robot to stop at a safe distance and check if the robot's dimensions are small enough to follow and fit to the determined path.

These sensors will be more effective in short distances. For this purpose, four ultrasonic distance sensors are planned to be used. Since their distance range is starting from 2 cm, it will be suitable for the necessary application.

Figure 3: HC-SR04 sensor

In addition to these solution proposals which are tested at module demo phase, if more accuracy is needed, linear surrounding scanning can also be considered during motion, instead of scanning while robot's position is stationary/fixed. In this case processing of continuously changing location data is also necessary for matching distance measurement data with continuously reading measurements.

4.2. DECISION UNIT

Decision unit forms the most essential and important part of the robot. To create a map one should collect the data from surroundings and process them to create a digital visualization of the surroundings. The path will be decided according to the data coming from the ultrasonic sensors and the time of flight distance sensor. The data collected from the sensors will be processed by the path decision algorithm and all of the data will be combined and sent to the visualization unit to create the map.

4.2.1. Path Decision Algorithm: Object Focused Exploring

According to the corresponding tests we carried out, it is decided that focusing on objects is a more efficient way for exploring the environment. Our aim is to explore every part of the map as accurate as possible in an acceptable time period. After achieving this, the focus will be reducing the exploring time. Considering both of them, the algorithm is based on detecting objects, recording their positions and moving towards the closest object to take measurements around it. It was observed that taking measurement from a relatively close distance and taking measurements from different surfaces of the objects increase the accuracy significantly. Therefore, when the destination object is determined, the robot will get closer to it and get the 360 degrees distance information from the object's different sides. After finishing one object it will record the object as “visited”, it will check for other objects and continue with the closest one again. After finishing this path, if there is any recorded but unvisited object left, it will go back for the object. In the end, it will check if the walls construct a closed shape. If yes, it will finish mapping, otherwise, it will move towards the open area.

The exploring algorithm is expressed with Figure 4-7 as follows.

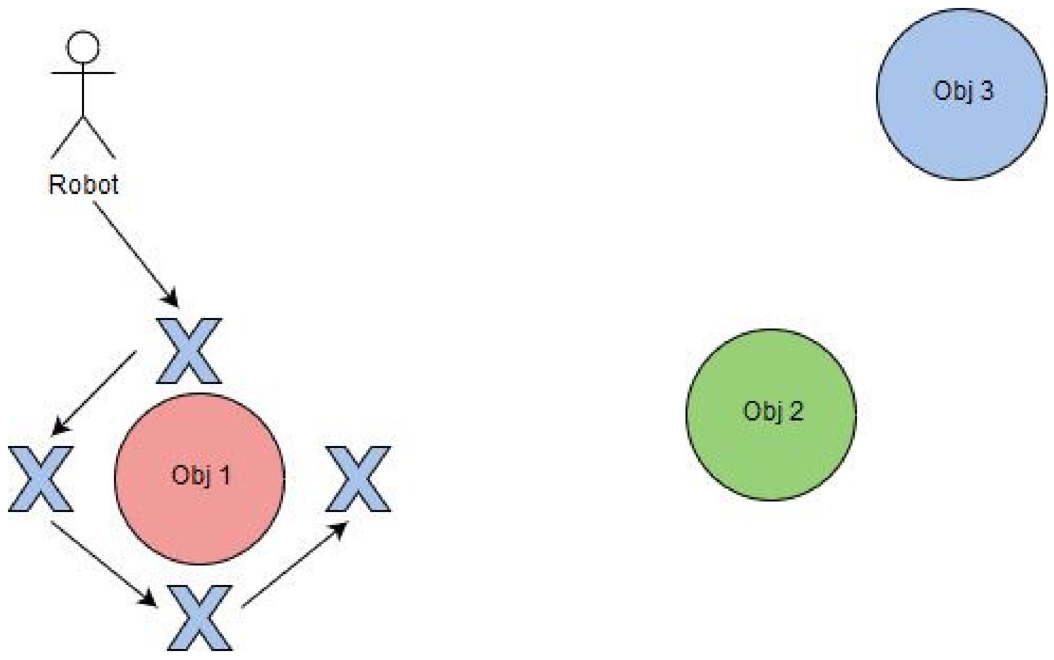

Figure 4: Identifying the first object

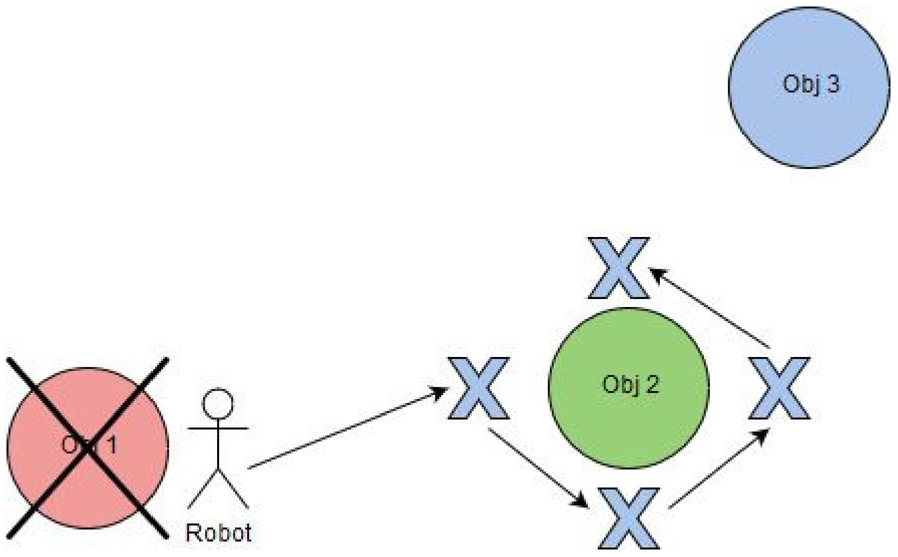

Figure 5: Identifying the second object

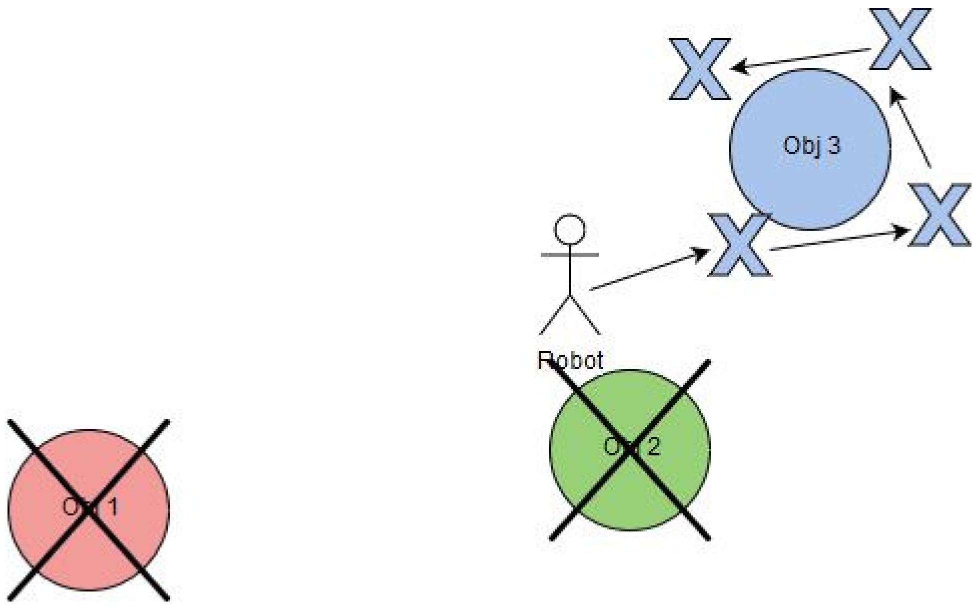

Figure 6: Identifying the third object

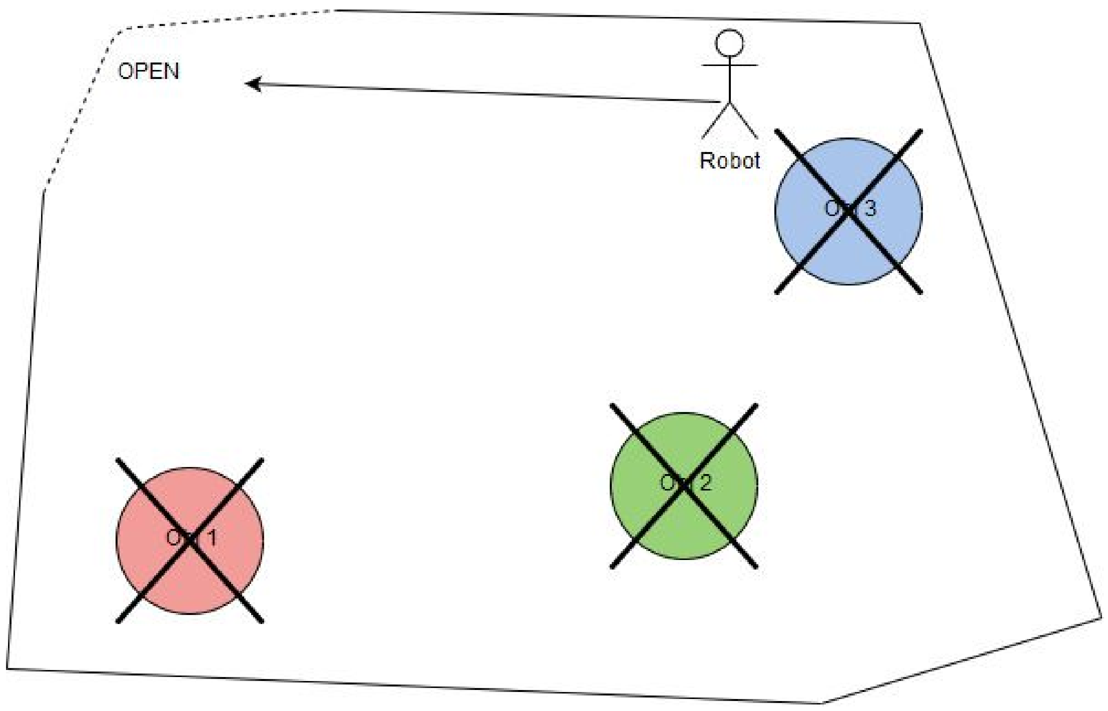

Figure 7: Measuring the missing outer map points

The algorithm also needs to cover the cases where objects are close to each other. In such a case, robot will try to follow the ordinary path, however, if there is an obstacle in its path, it will continue moving its direction and it will take another measurement, then try to go to the direction which was previously tried until it can. If it goes into a dead-end, it will follow the same path backwards and try to cover the object from the other direction. An example is shown in Figure 8.

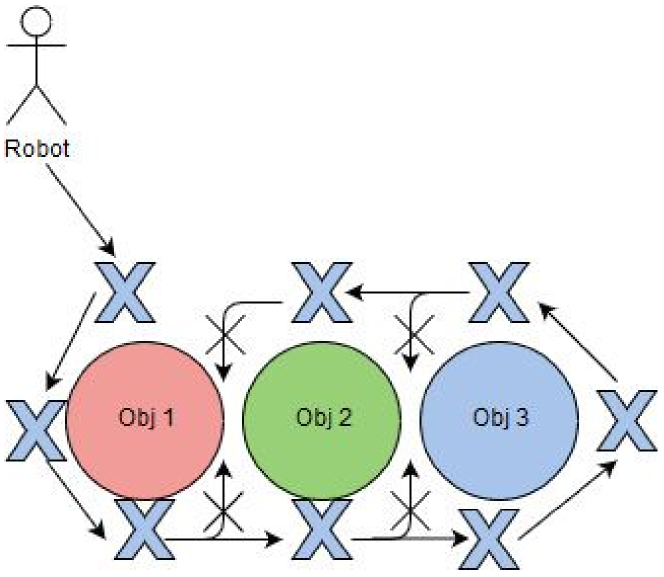

Figure 8: Identifying a group of objects

Algorithm will also decide the path that will be followed during the motion. Since robot will not take any data from the ToF during the motion it must decide the path without crashing with any of the walls or objects near. For this purpose ultrasonic sensors will be placed on the edges of the robot. These sensors will give the distance as the output and these distance data will play an important role on the path decision. Firstly, if robot goes on a straight path , algorithm will prevent robot from getting too close to the objects or walls on the sides and robot will follow the path that is far enough from any objects. Secondly, when robot reaches to a dead end, the sensors at the front will prevent it from any unwanted collisions. In such a case robot will find the suitable new path according to the decision algorithm to create the map and it can move in various directions to complete the map.

4.2.2. Testing for Object Detection and Path Decision Algorithm Efficiency

As it is stated in project description, mapping area will be any arbitrary convex shape. With this a priori information, robot can conclude that any concave region occurred in the scan is an object. By using the sensor scan data taken from a test environment, testing of the object detection(without considering its shape) can be held on by checking the derivative of the distance data with respect to time. To make this testing, we used MATLAB for calculation. For the test scan given in Figure 9, where object is located at (0,100) coordinates test procedures and results can be seen from Figure 9, Figure 10 and Figure 11.

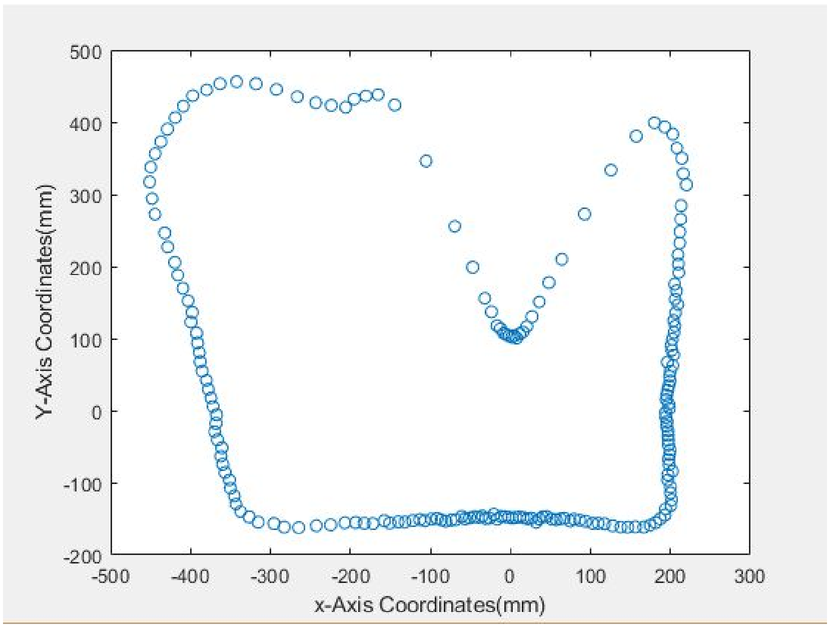

Figure 9: Point cloud representation of a measurement(cylinder object with rectangular outer border)

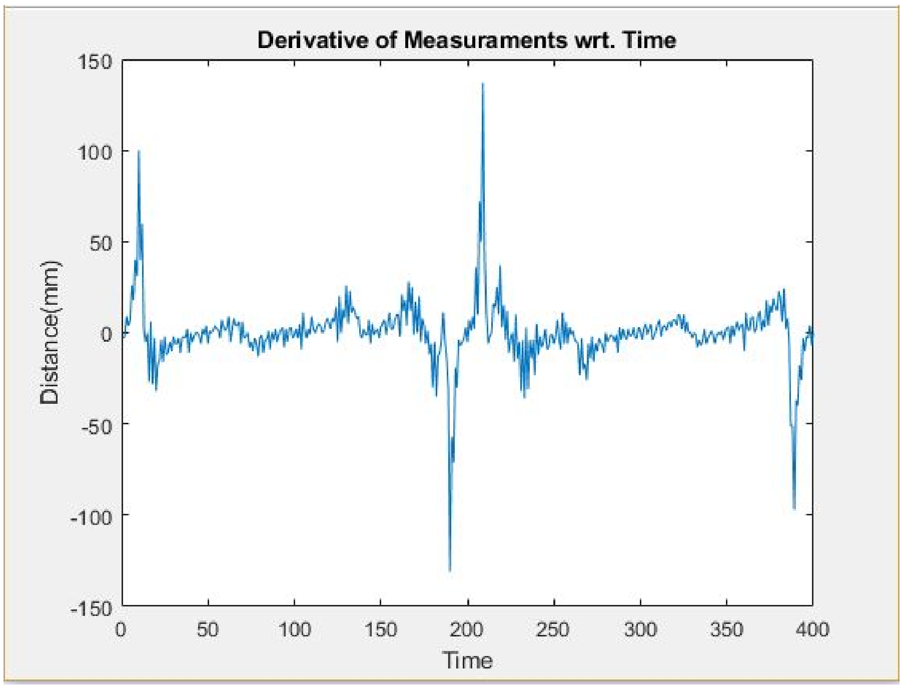

Figure 10: Derivative of the measured distances

As expected, from the derivative graph given in Figure 10 it is observed that there is a significant negative change in the distance followed by significant positive change, which yields an object due to concavity. By inspecting the local minimum and maximum in small windows, we can filter the data points taken on an object by using proper threshold value for amount of distance chance.

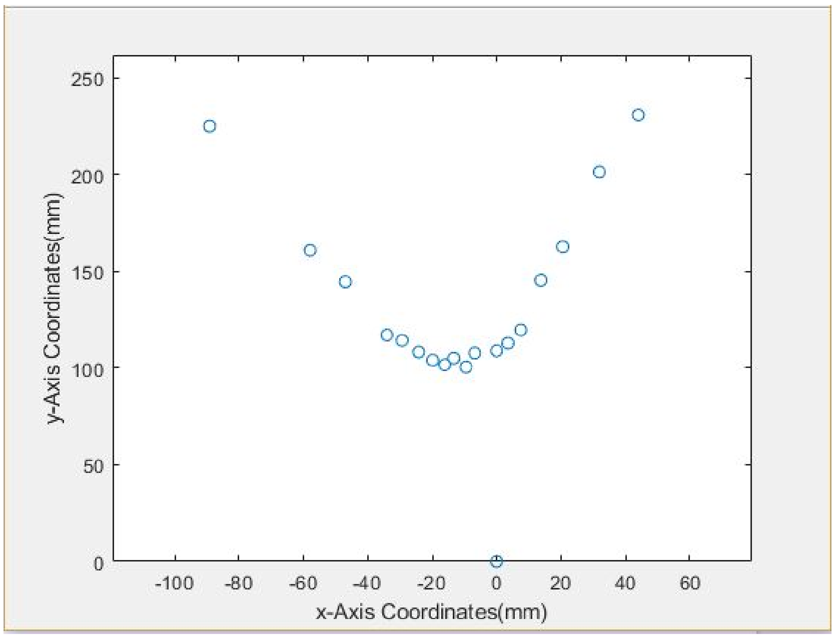

Figure 11: Derivative of the measured distances

As it is given in the Figure 11, object location can be obtained to decide robots path for closer scans. However, since it is a scaled and known environment, window length and threshold value is chosen specifically. During the further testings, these values will be chosen by an adaptive algorithm and implemented on the robot.

By assuming the correct object locations are known, efficiency of the path algorithm for accurate mapping is tested by taking data from the decided scan locations from the path algorithm on the testing environment.

After taking the measurement data from each decided points according to path decision algorithm, test results for whole map can be seen from Figure 12 and Figure 13.

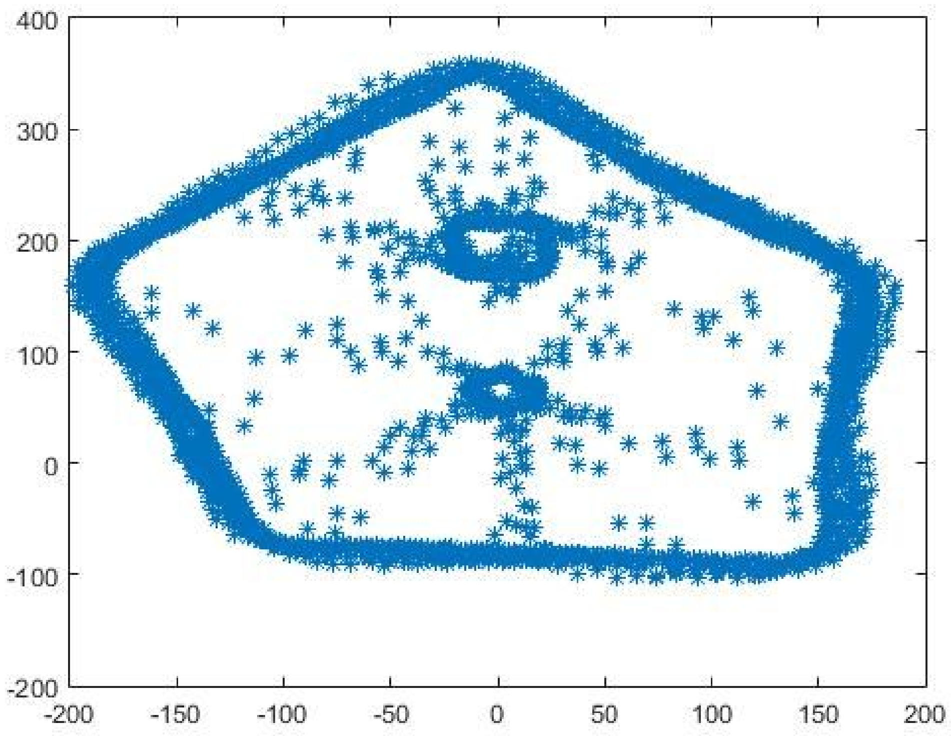

Figure 12.a: Point cloud data plot of the environment and the object via MATLAB

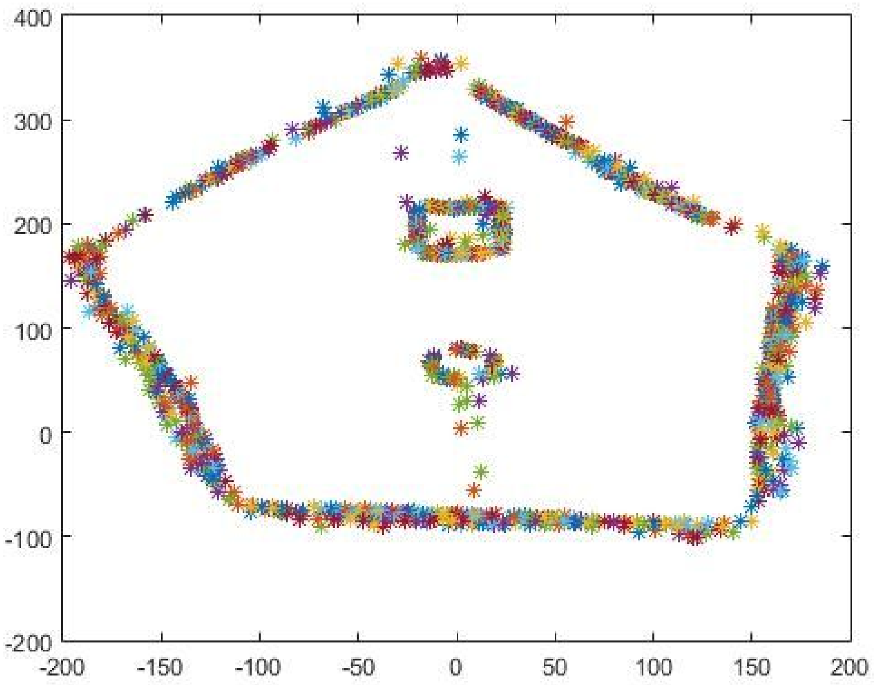

Figure 13.a: Filtered data plot via MATLAB

As it can be seen from Figure 13.a & Figure13.b, with this path decision algorithm, features of the objects can be observed accurately for square and cylinder objects.

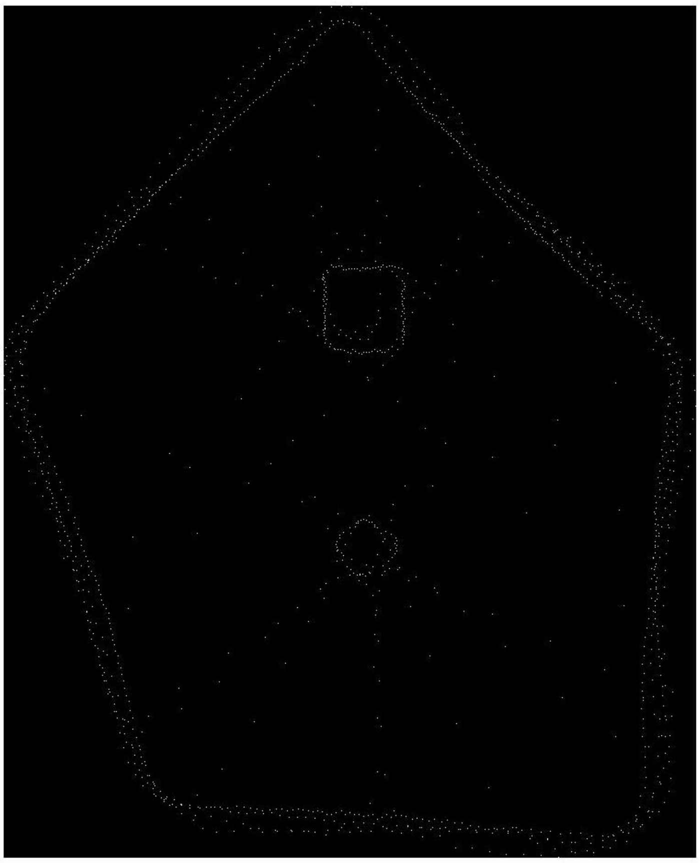

Figure 12.b: Pre-filtered data

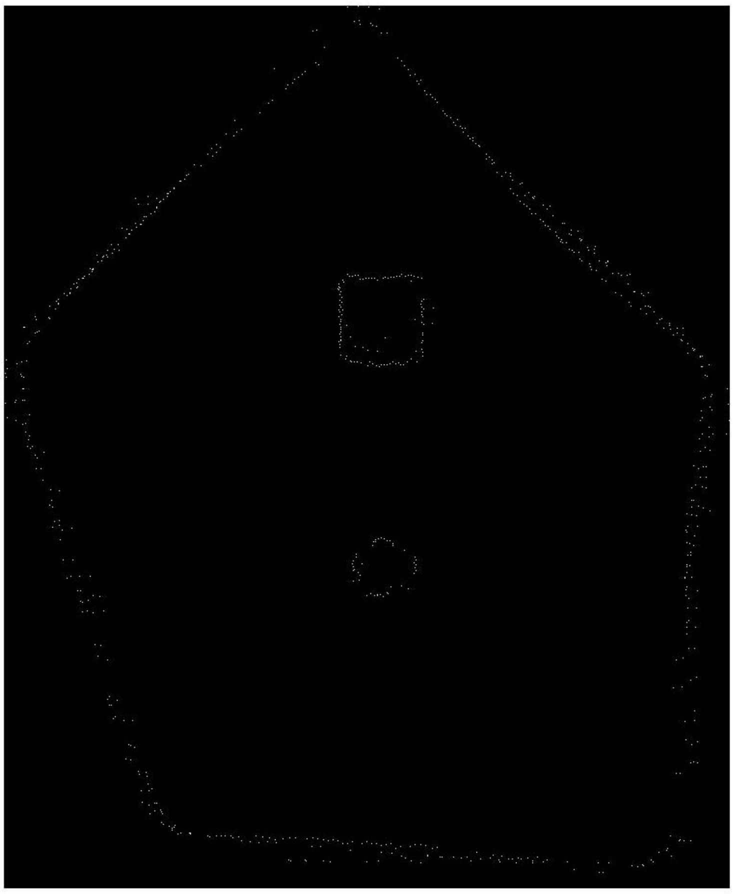
Figure 13.b: Filtered data

4.3. MOTION UNIT

Motion is one of the most important concepts in the robot's design. Increasing the accuracy of the measurements and decreasing the mapping time which are our main objectives are significantly related to the motion algorithm and method. The motion algorithm to determine the path was explained in detail in the 'Decision Unit' part.

As the method of the motion, it was decided to use omni-directional motion. Since differential drive causes errors which will accumulate, especially when robot is making turns, omni-directional drive will provide a more accurate computation. Omni-directional drive can be used to drive the robot to all linear directions, moreover in clockwise and counterclockwise directions. However, since it was decided to use motion along only two axes, x and y. This simplification of the robot’s movement will ease the self localization and decrease the amount of probable errors.

4.3.1. Motion Method: Omni-directional Wheels (Kiwi drive)

Omni wheels are capable of controlling each of their degree of freedom independently. Our robot has three degree of freedom and omni wheels enable us to control the robot in the exact direction that we want.

Our wheels will be placed as there will be 120 degrees between them and the axis of the rotation will be normal to the center of the robot as in the Figure 14. In other words, the wheels will be at -60, 60 and 180 degrees. The one should realize that the direction of the velocity will be perpendicular to the motor axis. Therefore, the velocity vectors will have the angles 90 degrees more/less depending on the direction of driving.

The movement of the robot can be expressed with the contributions of the all three vectors. Robot’s moving direction will change as the contributions to the displacement vectors by the motors is changing.

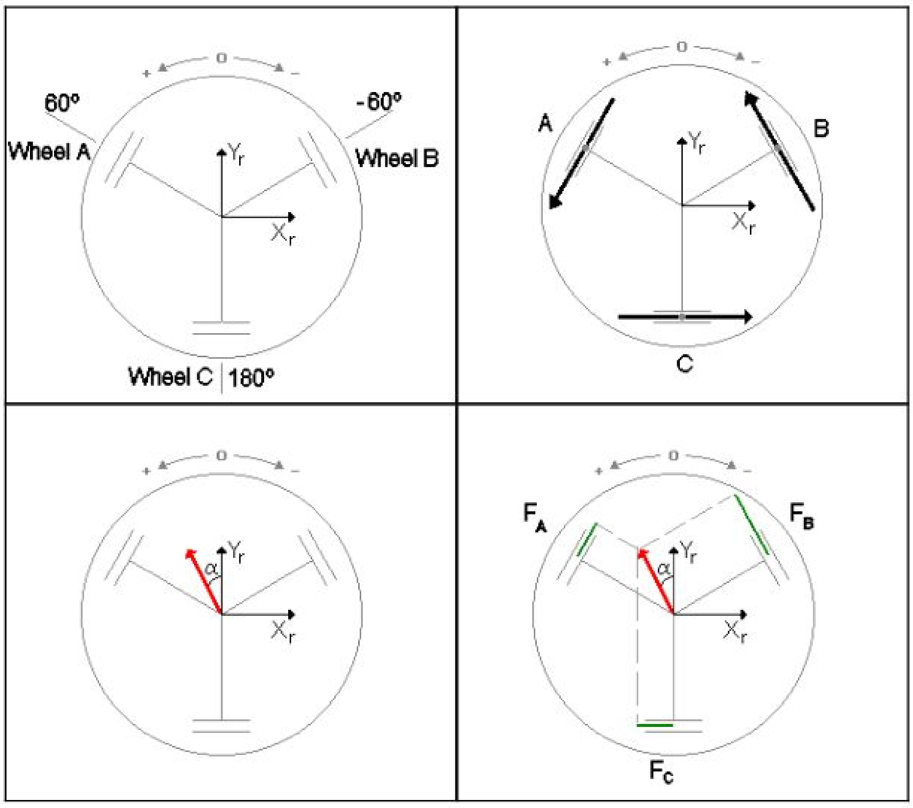

Figure 14: Direction representations of 3 omni wheel structure

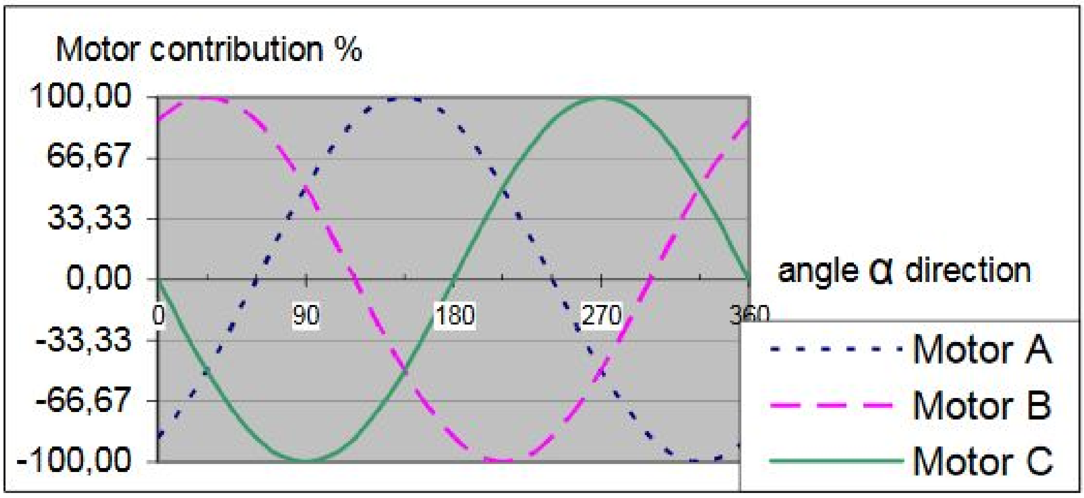

Figure 15: Omni wheels motor phases

Omni directional wheels are selected to reduce motion and position based errors. The selection of omni wheels allows us to rotate the motor without changing the position and direction of the sensor. This will prevent sensor’s position based measurement problems which may cause large errors.

A platform employing three omni wheels in a triangular configuration is generally called Kiwi Drive.

4.3.2.Test Procedure

To test the omnidrive system, body of the robot will be constructed with DC motors for locomotion and two axis encoder. By using directional commands from Raspberry Pi, we will test the motion unit.

4.3.3. Alternative Solution

As an alternative to the omni-wheel movement, differential drive of two symmetrical motor, with encoders on each motor, can also be used for motion and odometry data. However, to minimize the rotational error, an additional IMU sensor would also be added to have additional correction.

4.4. SELF LOCALIZATION AND MAPPING

Since the movement of the robot is limited by only two axes, it is needed to get the information of displacement in x and y directions. To achieve this, there are several methods to track the movement.

We have been inspired from one of the most basic and common encoder systems in our daily life: computer mouses. In general mouses use optical encoder system to detect where the cursor is. Turning a mouse into an XY encoder is an efficient and cheap way for our purpose.

As the output of this system, we will get the information of how much the robot moved from its previous position in X and Y directions which is referred as odometry data in the following parts.

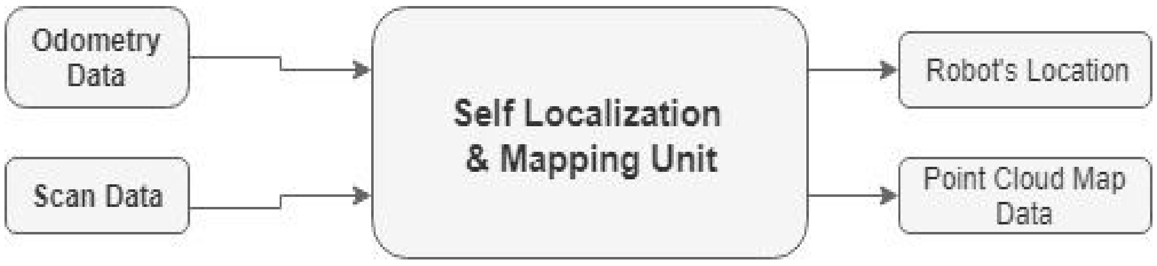

Diagram II: Input output relation of localization unit

To achieve accurate mapping, proper self localization is a must for an autonomous system. In this unit, the robot uses the landmark’s (or objects) location matching in between each scan and the odometry data (in x and y coordinates) taken from two axis encoder. By using this two inputs to the subsystem, robot will locate itself while creating a raw mapping data of the environment with the help of commonly known concept named ‘“Simultaneous Localization and Mapping (SLAM)”.

To implement the SLAM algorithm with odometry data, we will use extended kalman filter (EKF) approach. In our EKF SLAM approach we will use the following algorithm given in block diagram[3].

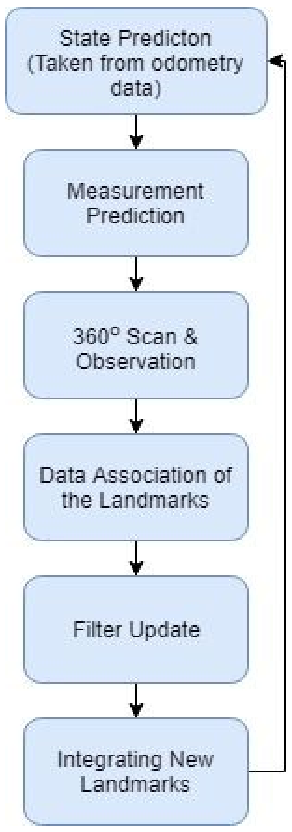
State Predicton (Taken from odometry data)
Measurement Prediction
360° Scan & Observation
Data Association of the Landmarks
Filter Update
Integrating New Landmarks

Diagram III: Algorithm of localization

- **State Prediction:** After locating the detected objects, robot decides where to locomote and move closer to the nearest object, which will result a predicted location of the robot's new position.
- **Measurement Prediction:** After movement, self localization and mapping unit will have a predicted object location relative to the robot's new position.
- **Scan & Observation:** At its new location, robot makes another scan and locate object visually.
- **Data Association:** Unit associates the taken data from state prediction and measurement prediction.
- **Filter Update:** Due to data association results, kalman filter coefficients will be updated
- **Integrating New Landmark:** Unit integrates the new observed objects and returns the state prediction step.

4.4.1. Self Localization Testing

To test the self localization from scanning data, a simulation with decided angular resolution and zero error scan data is constructed on MATLAB, as given in Figure 16. Very basic measurement results are given in the mentioned figure below, by placing the sensor around the cylindrical object and changing its location 4 times to scan all of its surface. As it can be seen, red dots are the points scanned and perceived by the sensor which is the equivalent of the multiple-mentioned scan data which is used for distance measurement.

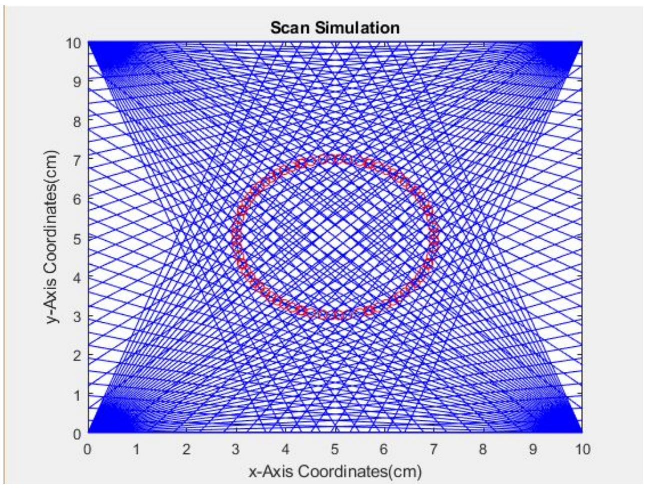

Figure 16: Simulation of an object sensing

By running the scanning simulation from different places and obtaining the object data points, we will test our self localisation algorithm by using scan matching options such as feature based matching or grid base matching. After obtaining successful 2D rotational and translational information, we will add error to the placement location (odometry data) and noise to the scan data.

There will be also parallel testings of the 2 axis encoder on the demo surfaces to observe the amount of error caused by odometry data to feed simulation error and improving localization and mapping algorithm to optimal level.

After observing the test results and limitations of the self localization and mapping algorithm, we will tune the path decision algorithm to meet our requirements.

4.4.2. Alternative Solutions

In case of not achieving satisfactory results or not meeting the requirements, for taking the odometry data, IMU sensor can be used to achieve more accurate state prediction. Also for achieving more accurate scan data, additional sensors(which will mainly used for collision avoidance) can be implemented as additional observation sensors.

In addition, another method that can be used to correct the errors caused by miscalculated locations, can be implemented ourselves by mimicking Kalman filter. The first measurement can be considered as the non-errored(reference) measurement. Then further measurements can be combined with the reference measurement by applying a penalty to them. The number of rotation of the wheels can be used to estimate the penalty and by applying this penalty to the measurements, the error can be tried to minimized.

4.5. VISUALISATION AND PROCESSING THE MEASURED DATA

4.5.1 Processing Measured Data

Obtained data from the sensors will be stored in the form of point cloud(X and Y direction values for each point) since it can be perceived easily by humans and further processing can smoothly be implemented on it. To visualize the overall data and create a map of objects, it’s needed to combine the data from sensing unit with the robot’s current location and motion information.

The raw measurements from the sensing unit are obtained in terms of centimeter. Furthermore the angle at each measurement is known precisely since a step motor is used. Multiplying the obtained centimeter values with cosine and sine, the location of the measured points can be obtained in X and Y direction. As it is explained in the decision algorithm, several measurements are taken from different points respect to the object. By adding the offset values to the cosine and sine operations,measurements from different points can be combined to construct point cloud dataset for the measured area. An important thing here is to measure the offsets correctly, since the error at each shift may result with unmatched figures.

After getting measurements from several locations and combining them, it is seen that shape of the object is obtained clearly however there some noisy points occur in the dataset. This happens because the sensing unit emits ‘O’ shaped light which can not detect sharp edges, and it returns average distance of the measured points that causes an error. To avoid these noisy points, a basic but useful method is designed that is inspired from mathematical morphology techniques. A kernel is convolved with the dataset. If the kernel has a matching above a threshold value with the dataset, the points in the dataset are kept, if the matching is below the threshold then the points are erased. The method is like ‘erosion’. By using this method by the end of each object measurement, a clear point cloud map will be obtained.

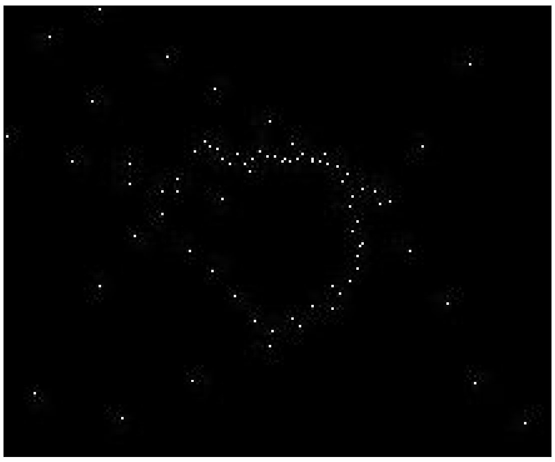

Figure 17: Point cloud representation of the detected points before processing

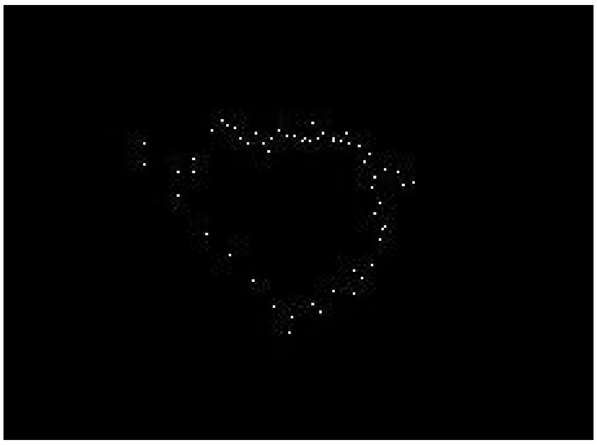

Figure 18: Point cloud representation of the detected points after processing

4.5.2 Finding Centre Location and Rotation of the Objects and Map Borders

In order to find centre of the objects, a method called K-Means clustering can be used. When the outer borders are neglected and only the points of objects are used, by giving the data and the number of objects to the function, k-means algorithm can find the centre points. The important point here is to find number of objects correctly, otherwise the found centres by the K-means algorithm will not be correct. At the worst case, elbow method might be used to cross check the found number of object even though it is computationally expensive for a microprocessor. The algorithm can be run several times to check if the centroids fall into the same object. The small errors on the perimeter of the objects would not affect the found centre much. Since measurements are taken from symmetrical points,due to their average, these errors are tolerated.

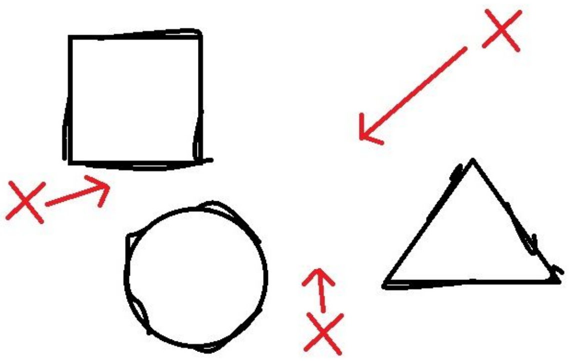

Figure 19: Three object with some perimeter error, three random centroids are initialized

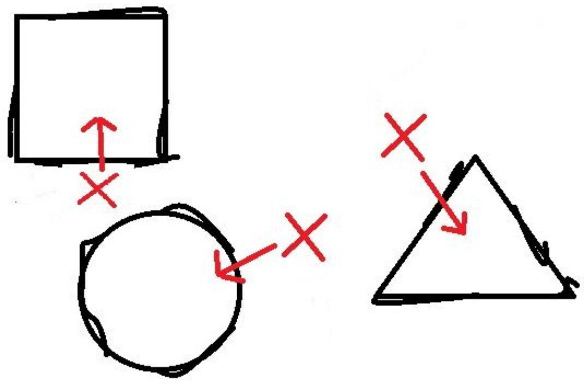

Figure 20: Centroids move towards the cumulated points

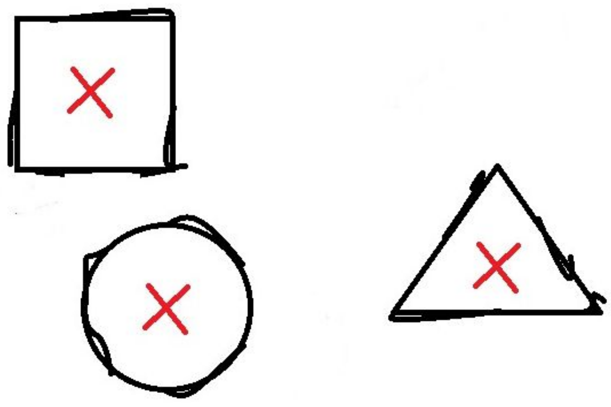

Figure 21: Centroids convert at the centre of the objects.

In addition, for this project there is an advantage of knowing the objects perimeters beforehand. If shape and rotation of the objects can be estimated through measurements, by using pre information, the results can be cross checked and the centres can be found easily. When the centre and the shape of an object is known, by applying a brute force convolution between a preknown perimeter kernel and the measured object, more precise rotation can be found even though the measured object has noise on it. The kernel can be spinned according to the centre point of an object. When the maximum value is obtained, the rotation can be found easily. It might be a power consuming method but successful. This method might be used after the tests conducted using wheels. For instance in Figure 22.a, black triangles represents the measured object and red ones are the preknown perimeters. By taking convolution and spinning the kernel the maximum result will be obtained at c. After d there is no need to keep spinning because best match is found at c.

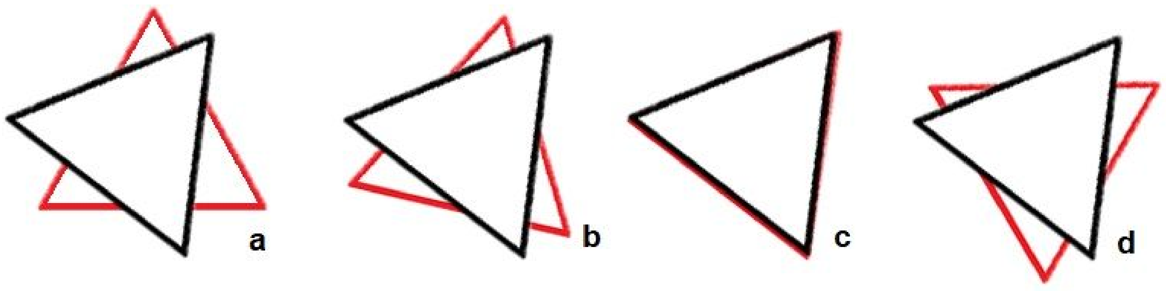

Figure 22: Black triangles represents the real rotation where red one shows kernel

For the outer boundary of the map, since several measurements are taken, there is a higher density of points compared to objects. By taking their average and using Ramer-Douglas-
Peucker algorithm fine tuning can be done. They are all available in Python libraries.

Any desired of output frame like number of objects, their rotation and centre etc. can be shown by using the point cloud based data when the map exploring is done.

4.6. COMMUNICATION PROCEDURE

After collecting data and visualizing it as a map image, the image will be sent to a computer for the user to see the final version of the map. Communication between the robot and the PC will also be held through Raspberry Pi module. Pi module has many options for sending data and enable us to apply these various types of communication channels for our robot.

For the main solution for communication, SFTP(SSH File Transfer Protocol) will be used for transferring the map image(created on Raspberry Pi) to client's PC. With this protocol, SSH server will be created on the Raspberry Pi and the data will be transferred to PC over Wi-Fi. Thanks to robustness of SSH, we expect maximum 30 seconds transfer time to consider data transfer as successful.

4.6.1. Test Procedure for Communication

By assuming the map image is created successfully by Raspberry Pi, we will create different size and format of image files on Raspberry Pi and try to access files on a PC by using the 'meturoam' network. Also by repeating the same procedure on different locations to observe communication speed, we will be able to make proper arrangements to achieve subsystem requirement in each situation.

4.6.2. Alternative Solutions

In case of facing problems on SSH protocol, another way to send the data could be using an free cloud service such as dropbox. After creating the image, this unit would upload the file to cloud service and allow client to reach map manually.

Similar to previous solution, sending the final file over an email could be also useful due to reliability of common email services such as Gmail.

4.7. ERROR SOURCES

This project can be considered as a prototype of a autonomous map extraction robot that will be used in discovery of undiscovered places, places with undesirable extreme environmental conditions where human access is not possible. Hence, robustness is a crucial feature.

There are these following error sources that may cause unwanted results. In this section, these factors are summed up and previously mentioned solution ideas are matched with them accordingly.

Self localization error can arise because of unexpected motion error such as extra sliding in wheels due to external conditions or error from encoder system giving the movement data with an offset. These errors could also end up accumulating with each measurement and causing a larger error at the end. This type of error can affect both the information related to robot’s location and object’s location. To prevent such a case, a feedback loop can be created from the computational unit which checks and corrects the self localization data after each measurement step.

Sensor accuracy error may occur because of the long distance between the sensor and the object. We have face this problem when the area to be extracted is not limited by the walls. But when the previously defined extraction area is constructed exactly with the necessary objects and walls no problem occured. In addition to the tests that have been already made, new tests can also be applied to exactly state the distance limit of our prototype.

Sensor’s position based errors are aimed to be eliminated by using omni-wheels in motion unit. By this way sensor will always be facing to the same direction and it will come back to its initial position since the scanning will be done for 360 degrees.

5. FUTURE PLANS

5.1 WORKLOAD SHARING

This project is composed of different systems and these systems are part of different specific fields. Therefore, to create such a robot different kinds of knowledge about different fields are necessary. Since each member of our team is experienced at least one specific area, the tasks that need to be fulfilled were assigned to each group member according to their knowledge, experience and interests. In this way, it is aimed to achieve maximum working efficiency and speed. Robot and the algorithms were designed by everyone by coming to a consensus. Every team member contributed to the project equally. Also, each team member conducted tests on the systems with respect to their knowledge about the field that is related with the test subject.

Süleyman Emre Can: Construction and implementation of the self localization and mapping unit
Cansu Demirkıran: Creating the motion algorithm and implementing omni-directional movement
Tahir Çimen: Processing the filtered point cloud data and communication
Ekin Yılmaz: Processing of the obtained data to identify object features, creating testing algorithms and conveying testing and simulation process
Yekta Demirci: Filtering of the point cloud data and converting them into geometric shapes

5.2. WORK PLAN

Work plan is provided as the gantt chart form in the Appendices part.

5.3. FORESEEABLE DIFFICULTIES AND CONTINGENCY PLAN

To provide better service and create a robust product, as Thunderbolt Corp. we came up with backup solutions in the case of any contingencies as stated earlier in solutions section under each section titled as “Alternative Solutions”.

5.4. TEST PLANS

The created simulation program for module demo test will be developed, improved and used in future subsystem tests too. Accuracy of the overall map will be decided and improved by comparing it to the simulation of the created testing environment. Any prepared area with the walls and objects will be created in this testing module manually according to the location of the objects placed on a measurable floor and then the extracted map will be compared to this simulation. As stated earlier, these readings will be used understanding robot's location according to its surroundings i.e. self-localization.

Also the object type, shape and size identification algorithms will be tested on a similar simulation program which will be created firstly with errorless simulation data, then adjustable artificial measurement and position errors will be added. Finally, after all the improvements made on it, the algorithm will be tested with the real data. Related object scan data is given below in Figure 24-26 with zero and non zero errors. These will be the inputs of out object identification algorithms during testing process. Since the errors are adjustable, algorithms can be improved further.

The possibility of using artificial data and adding errors in an adjustable range is very useful while developing and improving all of the self-localization, mapping and object identification algorithms since by this way it is possible to learn the accuracy limits of the developed algorithms more efficiently.

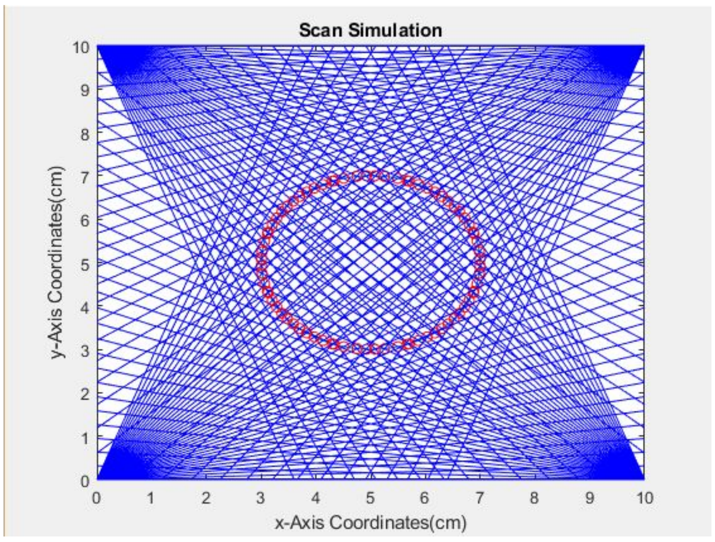

Figure 23: Combined measurement results for 4 corner scanning

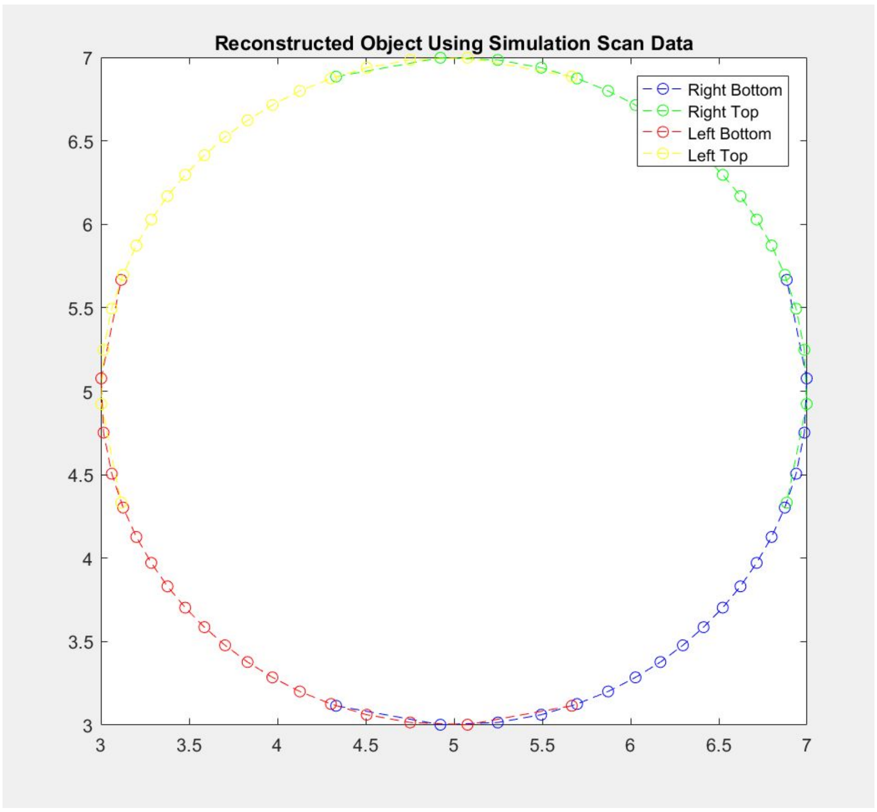

Figure 24: Reconstructed object map using simulation scan data with zero error

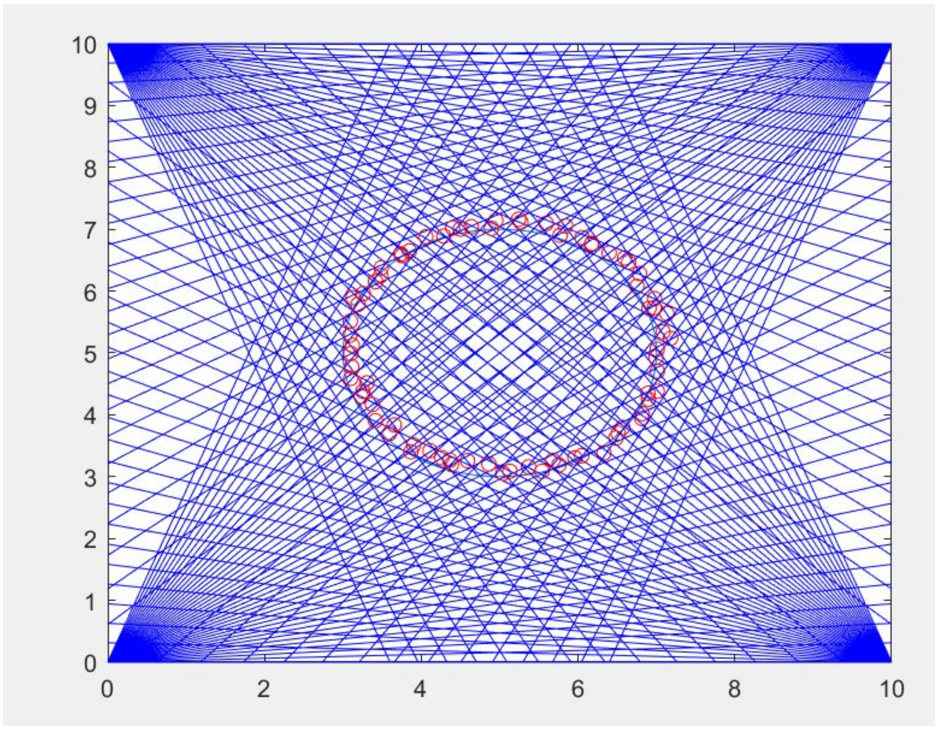

Figure 25: Combined measurement results for 4 corner scanning with random measurement errors added

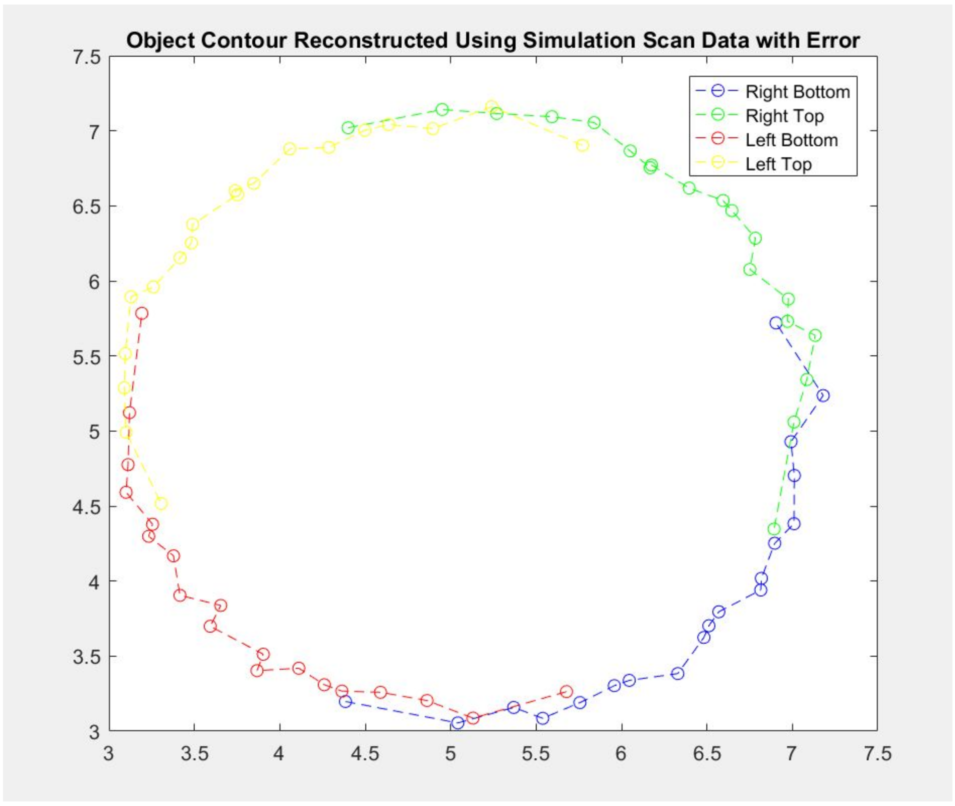

Figure 26: Reconstructed object contour using simulation scan data with adjustable sized random error

5.5. MEASURE OF SUCCESS

As a constraint for our prototype, devices must separately explore the playfield and by taking necessary actions, they try to locate and identify the objects in the field, in terms of their position, size and orientation and display the resulting field plan on a computer screen. The team that provides a better plan of the field in minimum time wins. Considering this constraint following measures are needed to be satisfied. The limits are determined using our previous scans and estimated operating time of the components our team is planning to use.

- Complete exploring the map in less than 15 minutes
- Creating the final geometric version of the map by using all the data obtained during the exploring less than 4 minutes
- Measuring the distance with an error less than %5 for every environmental conditions
- Working time with full battery is 25 minutes
- Creating a robust robot design

- Detecting a group of objects and identifying them separately
- Identifying all the detected objects correctly
- Locating all the detected objects correctly
- Finding rotation of the objects correctly (especially for triangulars, square objects)
- Determining size of all the detected objects correctly
- Creating the final map with labels for each object and sending it to screen
- Sending the final map to the screen in less than 1 minute
- Overall map extraction time less than 20 minutes

5.6. COST ANALYSIS

Each component's contribution to overall system cost is shown below in Table I.

Table I: Cost Analysis

Equipment	**Quantity**	**Cost**
Raspberry Pi 3B+	**1**	**40$**
VL53L0X time-of-fight sensor	**1**	**9.95$**
HC-SR04 Sensor	**6**	**9$ (1.50$ each)**
Stepper Motor	**1**	**10.5$**
DC Motor	**3**	**53.85$(17.95$ each)**
Omni-directional wheels	**3**	**12$(4$ each)**
Li-Po Battery(2s 1350mAh)	**1**	**16$**
Body	**1**	**15$**
		Total
		166.3$

5.7. DIMENSION AND WEIGHT ANALYSIS

For dimension analysis, technical drawing was made. The dimensions are given in Figure 27-28. Technical drawing snapshots from other point of views can be found in Appendix. Weight contribution of each component can be seen in Table II.

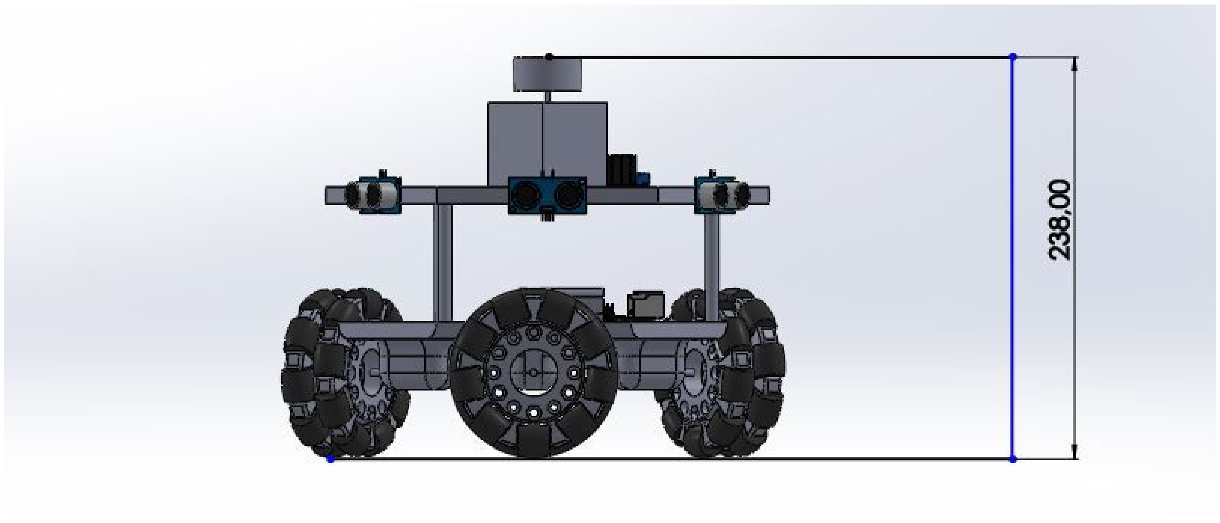

Figure 27: Technical drawing from front view and height

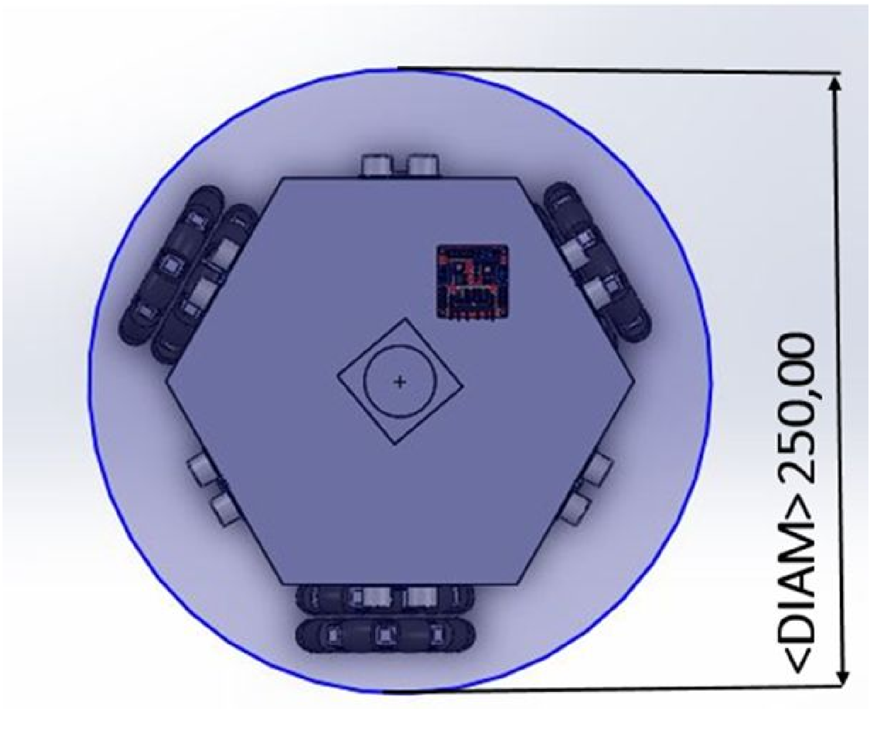

Figure 28: Technical drawing from top view and diameter

Table II: Weight contribution to overall construction from each component

Equipment	Quantity	Weight
Raspberry Pi 3B+	**1**	**42 gr**
VL53L0X time-of-fight sensor	**1**	**0.5 gr**
HC-SR04 Sensor	**6**	**51 gr (8.50 gr each)**
Stepper Motor	**1**	**150 gr**
DC Motor	**3**	**28.5 gr(9.5gr each)**
Omni-directional wheels	**3**	**180 gr(60gr each)**
Li-Po Battery(2s 1350mAh)	**1**	**85 gr**
Body	**1**	**120 gr**
		Total
		506.55 gr

5.8. POWER CONSUMPTION ANALYSIS

Table III: Power contribution from each element

Equipment	Quantity	Current(Worst Case)
Raspberry Pi 3B+	**1**	**2A**
VL53L0X time-of-fight sensor	**1**	**10mA**
HC-SR04 Sensor	**6**	**90mA (15mA each)**
Stepper Motor	**1**	**900mA**
DC Motor	**3**	**1.5A (500mA each)**
		Total
		4.5A

Approximate power calculations are stated in the Table III. More accurate results will be provided after testing process is completed.

According to the power analysis results, with the battery of our choice, robot can operate 20-25 minutes in the worst case without recharging.

5.9. DELIVERABLES

- A robot prototype
- Interface for robot and client communication
- Battery charger
- 2 years guarantee after the delivery of the prototype
- Demo presentation before the delivery
- Simulation set composed of 6 light yellow walls with 40 cm height each and 4 objects (2 cylinders of 10 cm diameter, 2 cylinders of 5 cm diameter, 2 square prisms with 7 cm edge length, 2 prisms with an equilateral triangular base of 8 cm edge length)

6. CONCLUSION

To sum up, overall solution approaches, input and output relations between subsystems, conducted and planned tests, some alternative methods, power analysis, planned cost and expected objectives and requirements from the project are given in this report without explaining the sub systems in detail.

The robot is composed of six sub-blocks which could be stated as:

The first one is the sensing unit, there is a step motor and a flight time based sensor to measure the surrounding objects. In addition there 6 more ultrasonic sensors to make sure robot does not hit any obstacle through the movement. Most of the tests are with the step motor and the flight time based sensor so far.

The second block is the decision unit. The micro computer decides where to go based on processed data, basically this is the part that follow the algorithms to complete the task. This part is very crucial since it is heart and brain of the project and different units send their data to.

The third part is the motion unit. Omni wheels are used instead of derivative or any other motion type since the rotation error is wanted to be minimized. Otherwise the rotation error would stack and would cause huge problems after some measurements.In addition, if the robot can accurately track how much it moves relative to the initialization point, data processing can be done much easily. Therefore encoders are also used at this part to track the motion in X and Y direction

The fourth part is the self localization unit. This is actually done within raspberry pi based on the information sent from the motion unit. However this is considered as another unit since it is an important concept. Kalman filters are going to be used via Python libraries.

The fifth parth is the visualization unit. The gathered information by the sensing units are processed based on some image processing techniques like mathematical morphology or K-means clustering. Shape, rotation, number of the objects are identified by this unit. At the end, the overall found data can be sent to a monitor with any desired interface.

The sixth unit is the communication unit. When all measurements and processings are done, the results are sent to a computer via wi-fi just to monitor them. SFTP(SSH File Transfer Protocol) is going to be used.

Since several tests are already implemented and promising results are obtained, it is aimed to obtain satisfying results at the end. Also, some alternative methods are developed in case of unexpected events.

7. DISCLAIMER

Design specified in the report complies with the standards of the selected project.

Süleyman Emre CAN Tahir Miraç Çimen Cansu Demirkıran

Yekta Demirci Ekin Yılmaz

8. APPENDICES

APPENDIX 1

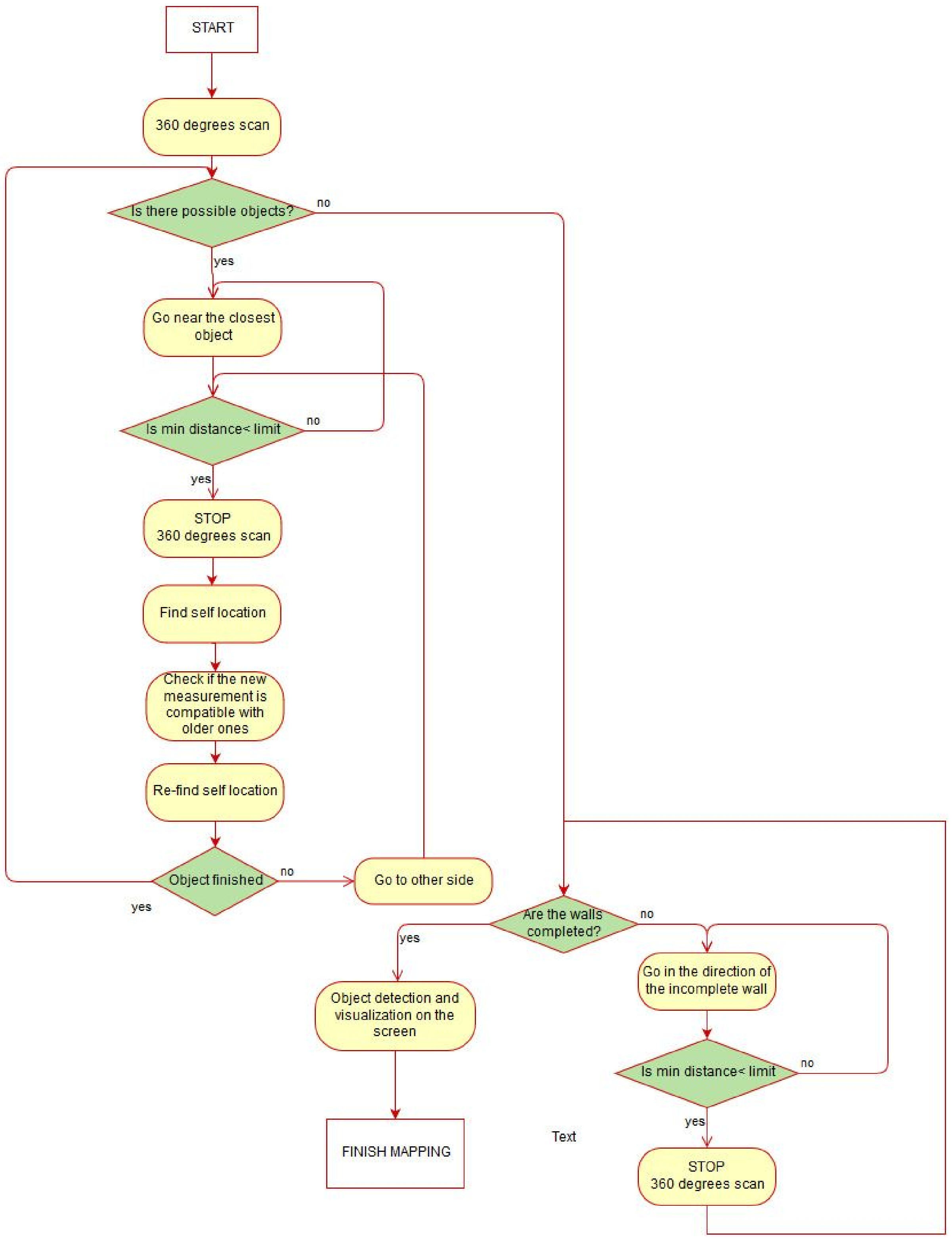

Figure A1-1: System level flowchart of main decision logic

APPENDIX 2

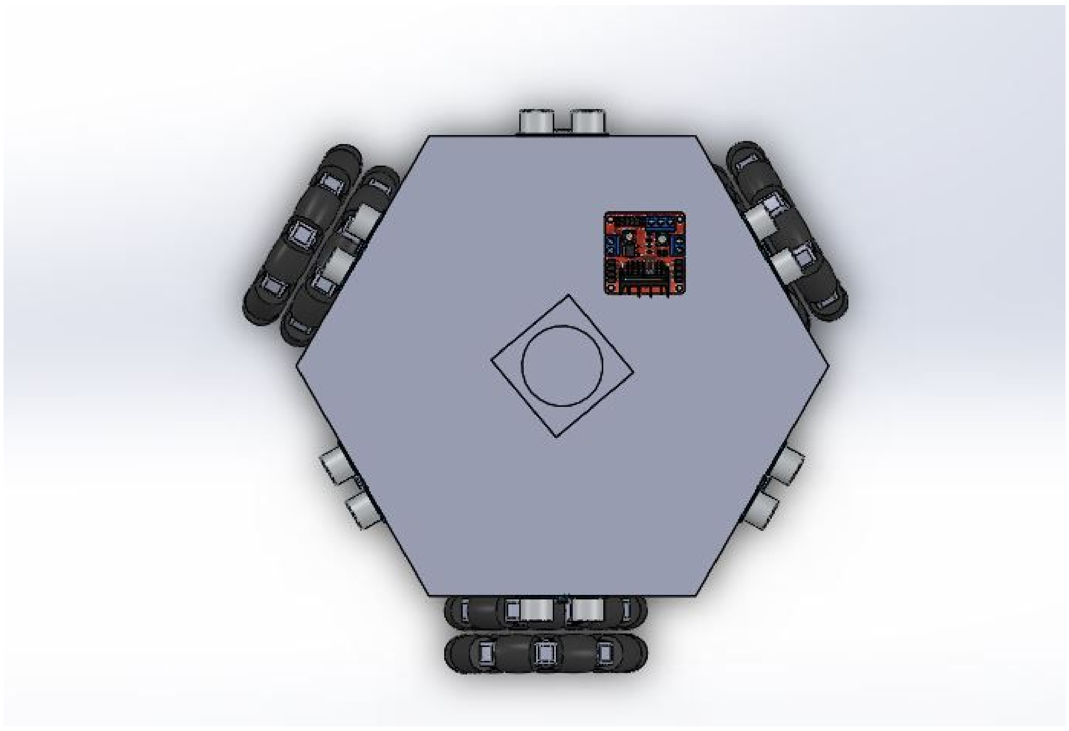

Figure A2-1: Bottom view of the robot

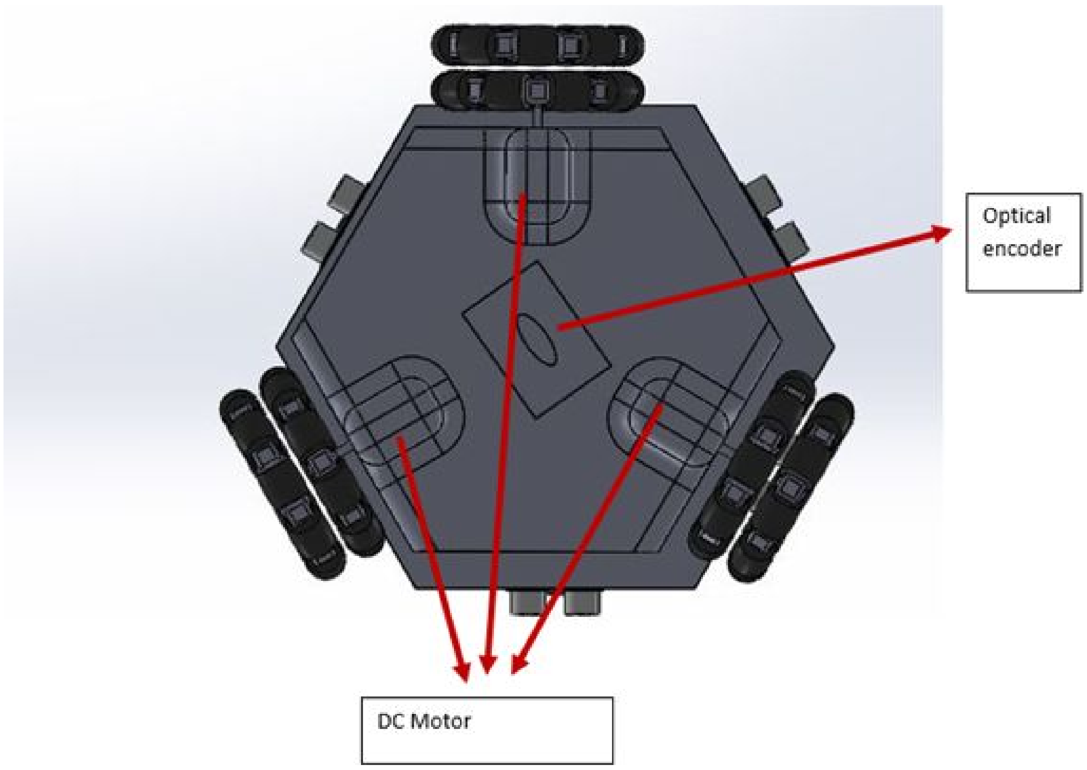

Figure A2-2: Bottom view of the robot with detailed descriptions

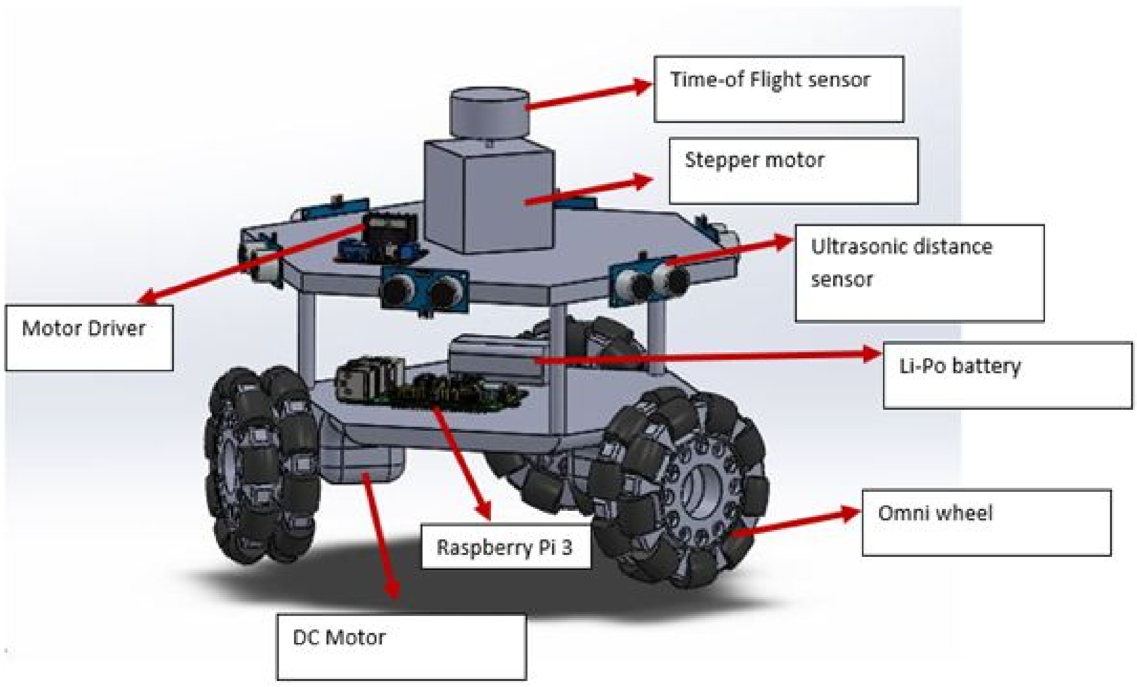

Figure A2-1: Detailed description and side view of the robot

9. REFERENCES

[1] VL53L0X Datasheet:
https://www.st.com/content/ccc/resource/technical/document/datasheet/group3/b2/1e/33/77/c6/92/47/6b/DM00279086/files/DM00279086.pdf/jcr:content/translations/en.DM00279086.pdf

[2] Ribeiro, A. F., Moutinho, I., Silva, P., Fraga, C., & Pereira, N. (2004). Three omni-directional wheels control on a mobile robot.

[3] Stachniss, C. Robot Mapping EKF SLAM Retrieved December 25,2018 from http://ais.informatik.uni-freiburg.de/teaching/ss12/robotics/slides/12-slam