FINAL 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:	149$

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 expect at this point, this platform is designed to be light, compact, accurate, robust and user friendly to endorse independent researches.

To achieve reliable operation for different environments, our product is always be aware of its surrounding by taking full rotation scans and observe its surrounding objects in two dimensions. While observing its surrounding, it is able to locate itself by using its own movements and observations in a continuous cycle of algorithms; therefore, product could easily understand where does it stands and location of the elements of its surrounding. Product is also 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 is created by identifying each landmark with fitting algorithms, and as final outcome, robot sends this map automatically to the client by over Wi-Fi connection.

As a team of engineers with different specializations, overcoming 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 is aimed. Now, Thunderbolt Corp. developed a prototype for an estimated cost lower than 200$, that can fit in a 25 cm cylinder and related the performance tests in a simulation environment are conducted with satisfying results. Final product is served with a software including interface for transferring the final map, a battery charger, 2 years of warranty, and necessary testing equipment. As Thunderbolt Corp. our final product has achieved ease the process of launching robots to space and provide the customer efficient solutions.

2. INTRODUCTION

This final report, is a detailed description of our company's final product on autonomous map extraction project. Overall results obtained during this 6-month-project process is summarized. First, a brief information is given about the company, then a detailed design description is made; explanations of the used algorithms, block diagrams explaining the main units and subunits, flowcharts of explaining the relationship of all subunits are given. Most importantly obtained results and analyses of performance tests are stated and operating principles of the robot is verified. Finally, a list of deliverables and budget analyses are given, following with a user manual specially prepared for our future customers.

3. COMPANY INFORMATION

Mission
Our mission is to provide products with high quality to our customers. We are trying to meet the expectations by creating the most efficient solutions.

Vision
We believe that we can shape the future with our new ideas and contribute the development of the technology around the globe by inspiring many others and making life worth living.

Thunderbolt is a newly founded dynamic, competitive and customer friendly start-up. Due to quite fast improvements in the autonomous systems and AI fields, robotics is taking more and more part in our everyday life. Unfortunately many companies which are already in this field can not keep them up to date. After our co-founders have realized such a need, an advanced robotic company Thunderbolt is founded. Beside making sophisticated robots, team is aiming to present a very basic, user friendly interface to the customers.

Our team consists of five electrical and electronics engineers who have different specializations. Different backgrounds, skills and experiences of our engineers complete each other and allows each member to contribute the project from different aspects. Organization structure of our company can be seen below.

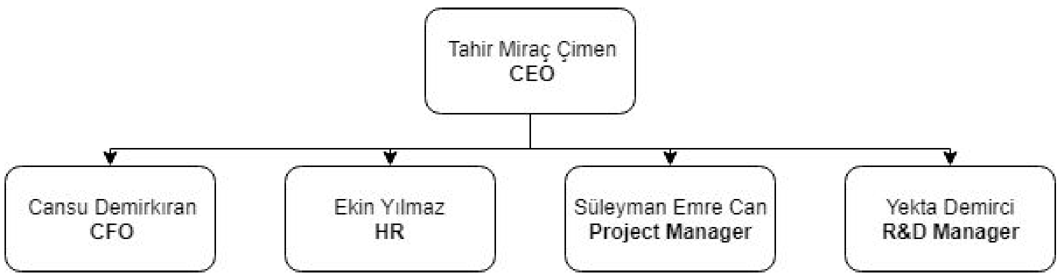

Figure 1: Organization structure of Thunderbolt Corp.

4. DESIGN DESCRIPTION

4.1. System Description

Main function of the system is to create a 2D map of the environment that it is currently in with also indicating the objects in that environment by giving proper rotations, shapes and centers of the objects. Overall system contains 6 main units which have their own unique subsystems that also include input-output relationships with each other. First, the working principles of the overall system will be shown as in flow chart form. Then, properties of each main and subsystem will be examined in detail in the following parts. The main and related subsystems, their input-output relationships and the working schematic of the overall system are given in the flowchart in Figure 2.Figure 2 can be seen in the additional page.

4.2. Sensing Unit

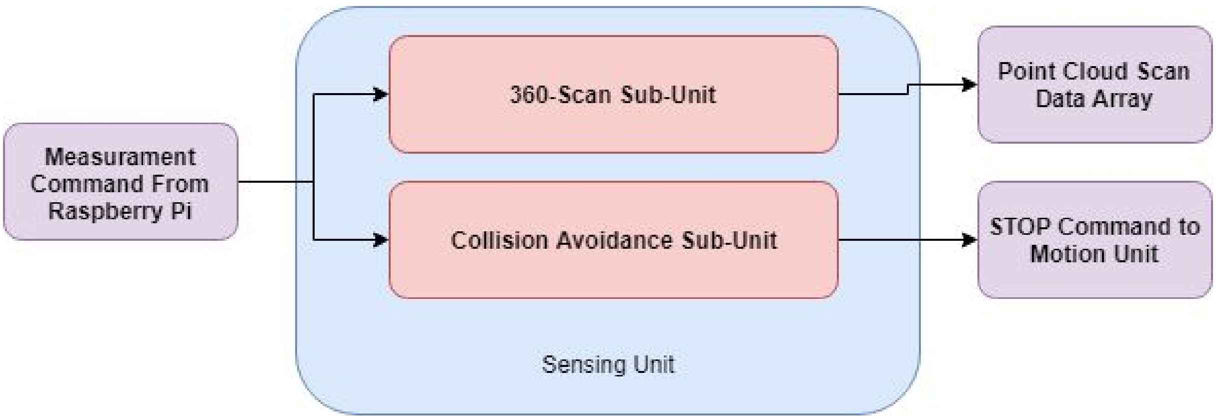

Figure 3: Block diagram for the sensing unit

As it can be seen from the overall flowchart, the key point for creating the map of the environment is being aware of the surroundings. This mission is handled by the first unit which is sensing unit. Sensing unit enables the robot to take many actions from the start till the end. The first and most important feature of this unit is that it gets the distance information between the robot and the walls and objects. That distance information is done by using 2 different types of sensors which serve different purposes. First one is the Time of Flight sensor which is used while obtaining the position data of the objects and walls and the second one is ultrasonic sensors which are used in preventing the robot from possible collisions during the motion in the field. These sensors are used in 2 different subunits which are 360 scan sub-unit and Collision avoidance sub-unit.

4.2.1. 360-Scan Sub-Unit

This sub-unit is used for obtaining point cloud data for each scan and contains 1 Tof sensor and a stepper motor. The sensor which is used in this sub-unit is the VL53L0X sensor. This sensor is the ToF sensor that is used for detecting the objects in the field and enables the robot to decide the direction that is being moved towards. Sensor simply propagates infrared pulses and measures the time between the sending and the receiving pulses and gives the distance information as the output. The reasons behind the choice of this sensor were its accurate distance measurement range(accurate up to 2m) and the fact that it can be easily integrated with the Raspberry Pi. As it was indicated in the overall flowchart, robot takes measurements from 360 degrees by using that Tof sensor. To get the 360 degrees data, ToF sensor is placed onto a stepper motor which has 200 steps. In this way, we obtained an increase in the accuracy of the measurements while getting the distance information since each data is taken at every 1.8 degrees. However, Tof sensor is only used when the robot is stationary and that causes the robot to be blind during the motion in the field.

Figure 4: ToF sensor integrated with step motor for main scans

4.2.2. Collision Avoidance Sub-Unit

To overcome this problem, collision avoidance sub-unit which is composed of 3 ultrasonic sensors is used. Ultrasonic sensor uses the sound signal instead of light while measuring the distance. 3 ultrasonic sensors are placed on the robot's right,left and front. To prevent the robot from collisions during the motion in the field, the ultrasonic sensors are operated before each move in the field until the creation of the map is completed. The reason behind the choice of these sensors was its range which starts from 2 cm. In this way, adjusting the proper distance between robot and any other objects or walls is provided and that enabled the robot to complete its path safely by checking the environment all the time.

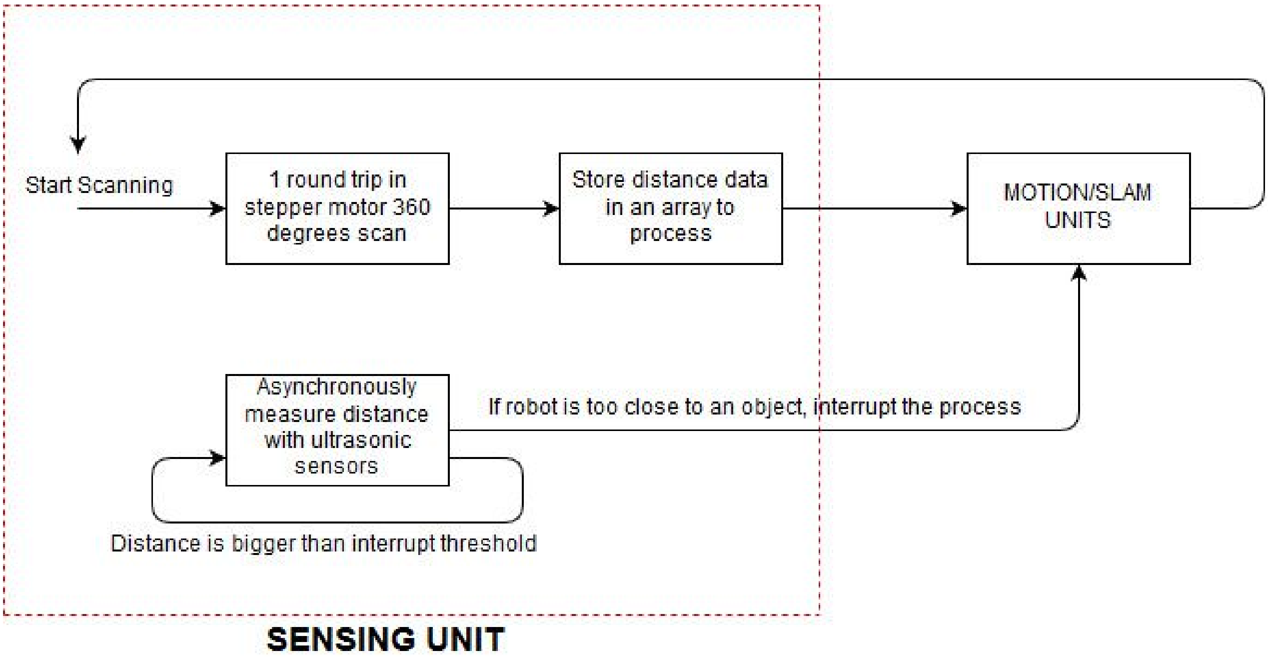

Figure 5: Flow chart for the sensing unit

Performance Results

With the different measurements taken from different distances, which are in range of our consideration in the map, following results stated in Table 1 are obtained:

Table 1: Actual distance reading results comparison with sensor readings

Actual Distance	Readings(mm w/+- 2mm reading deviation)
5cm	48
10cm	106
15cm	148
20cm	206
25cm	244
30cm	301
40cm	398
50cm	492
75cm	768
100cm	1012
150cm	1527

4.3. DECISION UNIT

One of the most essential parts of the process is to decide the path which will be followed during the motion. That part is handled by the second unit which is the decision unit. Decision unit enables the robot to follow a certain path to move around the objects and take the necessary measurements at the determined points. To do this, it receives the distance and point cloud information from the sensing unit and by using the path decision algorithm it creates the map to a destination point.

4.3.1 Path Planning Algorithm

To avoid collisions during the movement of the robot, path planning algorithm checks the point cloud data and returns a coordinate sequence for robot to follow. The algorithm follows the give flowchart while operating;

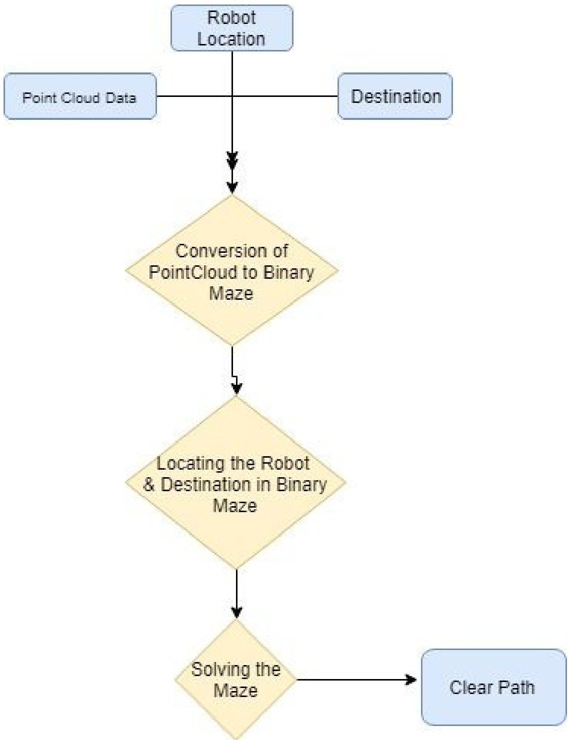

Figure 6: Path planning algorithm explained in a flowchart

After converting the map to a binary maze, where the zeros are yielding clear way for movement, algorithm assigns the destination and current location on the map. If the destination is not an empty point, algorithm assigns a new destination point which is nearest clear point to the previous destination. After the assignment, the algorithm solves the maze by focusing on the shortest path and low computational expanse. Basically it forces its movement to the outside of the obstacle and continues its movement towards to object During this process, our algorithm does not hold any unnecessary data for possible path routes. It simulates the movement on the maze and shortens the resulted path as short as possible. As the output of this algorithm, a list of coordinates are returns for robot to follow.

Performance Results

Considering the following case for illustration,

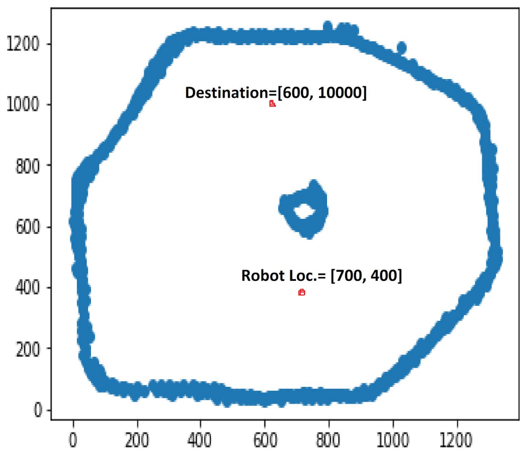

Figure 7: Example map for path planning algorithm

The algorithm creates the following binary maze with grids of 5mm squares, where certain amount of margin is added around to points to avoid collision of robot's body.

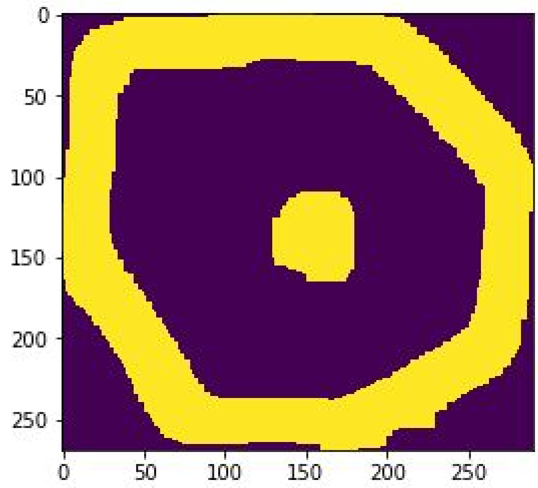

Figure 8: Created binary maze for path planning

After the solution of the maze, following path points are returned

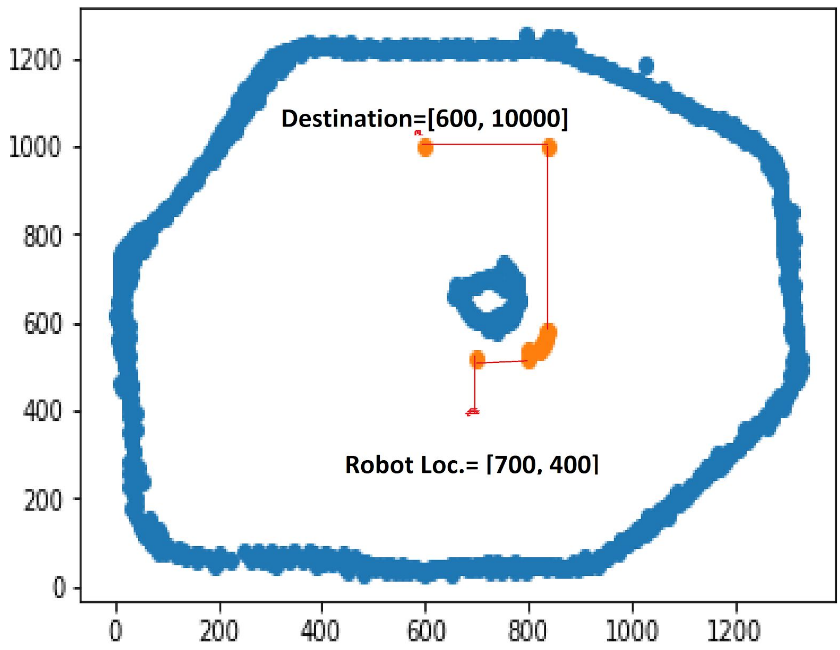

Figure 9: Further path points results for path planning

As it can be seen from the returned path coordinates, a route without collision is obtained to send motion unit. For this configuration, symmetric route would be shortest; however, due to our computational expense reduction goal, an acceptable route is chosen as final path.

Although the algorithm can overcome path planning, there are some collisions are observed during the object scanning algorithm, which takes close measurement around the object and cause failures due to accumulated errors in localization.

4.3.2 Object Scanning Algorithm

Since the exploration algorithm is object focused, the robot first takes a scan of the environment and determines the location of the nearest object. Then, by using the point cloud data that was obtained, it looks whether the path contains any obstacles or not. If there are any obstacles in the path, it creates a new path by using path planning algorithm . After getting closer to the nearest object, object scanning algorithm is started to be used. The object scanning algorithm works in this way:

For an object we have 6 locations that the robot goes and stands for a while. This locations are kept in a counter variable. After robot identified the object and got close to it, robot starts scanning first. Then, it starts to turn around the object clockwise direction. After first scanning, robot checks whether it can go left or not by using its left ultrasonic sensor. If it cannot go the counter becomes 6 immediately and robot starts to turn in counterclockwise direction. If it can move, it turns 90 degrees and moves to the first location,**counter becomes 1** and checks whether its right-side is empty or not by using the ultrasonic sensor on its right side. If it is empty, it follows the basic scanning path which is turn -90 degrees, check if it can go further and decide(if it can go further counter becomes 2). If it is not empty, it immediately turn -90 degrees and scans, and then again turn 90 degrees and checks if it can move forward. If it can move forward, it goes to the next location and at this location counter is not changed. Since counter is not changed, it **repeats the same procedure** for the case of counter 1. By constantly checking the front and the right ultrasonic sensors, robot follows a path around the object until the counter number becomes 6. If robot encounters an obstacle and cannot go further, the algorithm turns the counter to 6 immediately. Then robot starts to follow a counterclockwise path to close the loop. For better understanding, the working principle of the object scanning algorithm for some cases can be seen in figures 10,11 and 12.

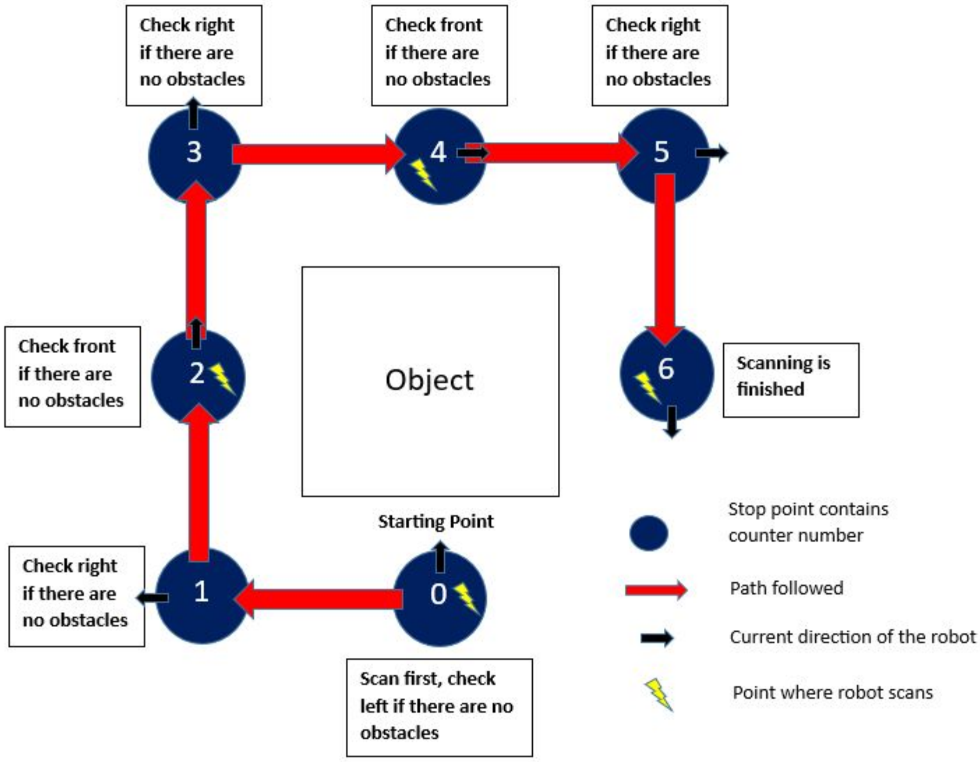

Figure 10: Principle clockwise movement of the robot around the object without any obstacles

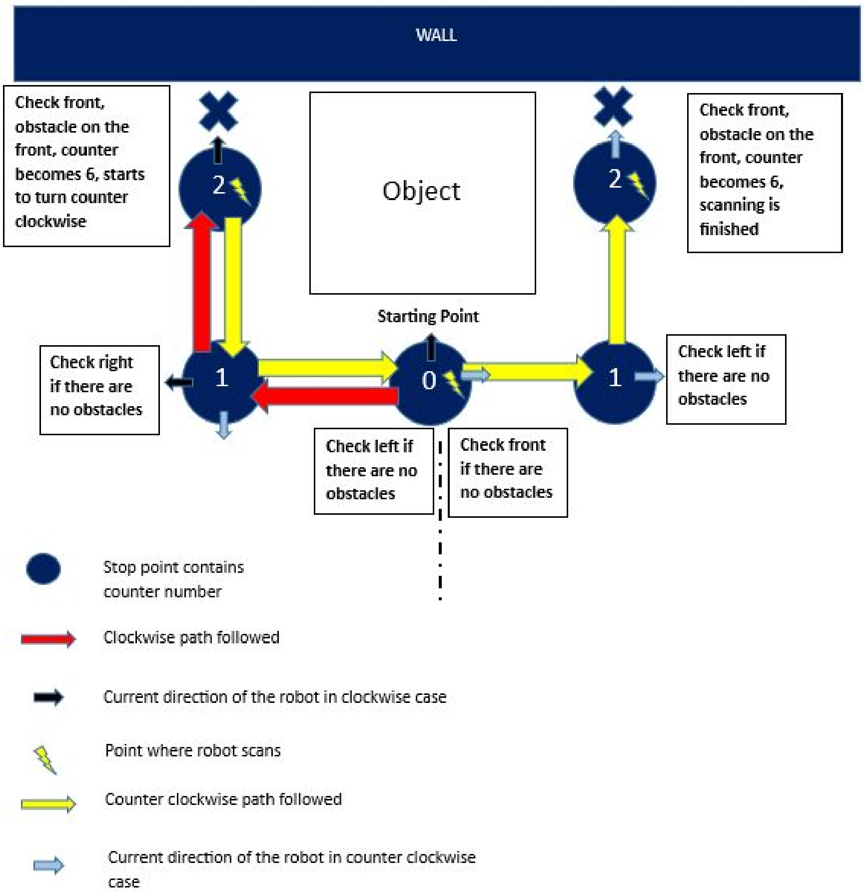

Figure 11: Movement of the robot around an object in a more challenging case with an obstacle

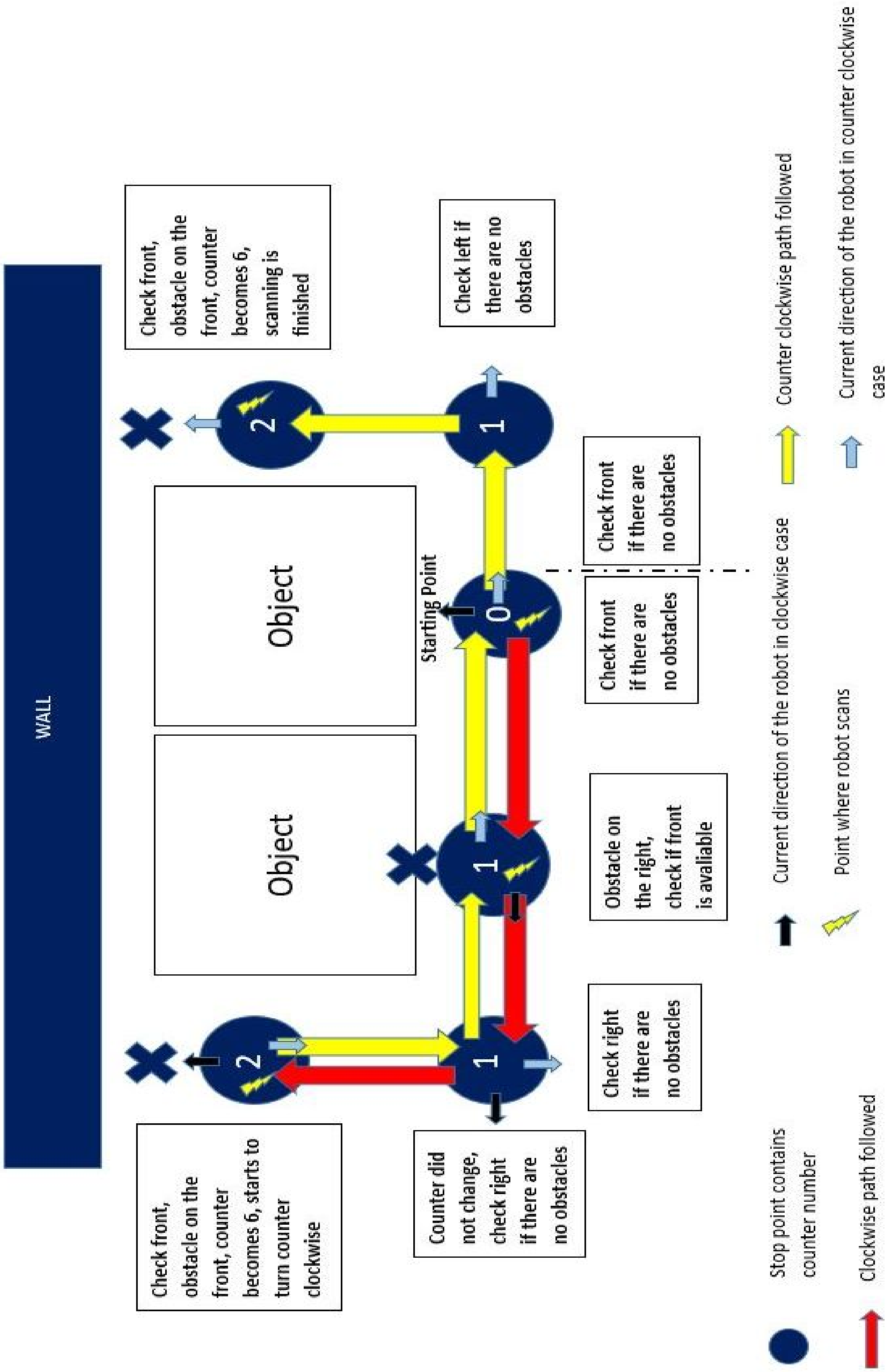

Figure 12: Further scanning positions of the robot in the case of multiple objects with an obstacle around

After finishing the scan of the object, the object is marked as visited and the robot moves through the nearest object that has not been visited yet by using the last point cloud data. Path decision and scanning algorithms are used until all the objects in the field are visited. In the final part, it checks whether all the walls are completed or not. If there are still empty edges of the map, it completes the map by taking additional measurements in the field. After the map is completed, scanning is finished and the data is sent to the visualization unit to create a visual representation of the environment that the robot is currently in.

4.4. MOTION UNIT

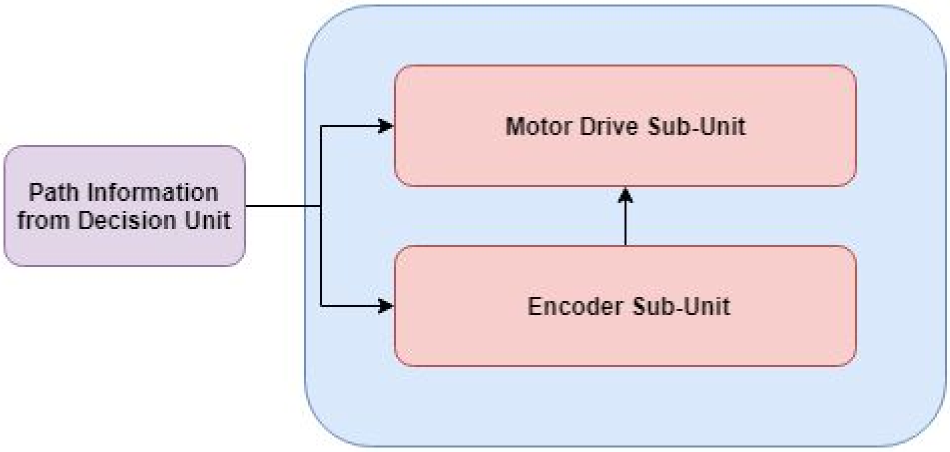

Figure 13: Block diagram of the motion unit

The movement of the robot in the field is held by the motion unit which is the third main unit of the system. For the motion, differential drive is used by considering the advantages of it which are being easy to control and slipping less during the movement with respect to the other motion methods. 2 DC motors, two encoders and two differential wheels forms the motion unit of the system. Each of the encoders and wheels is connected to each motor.

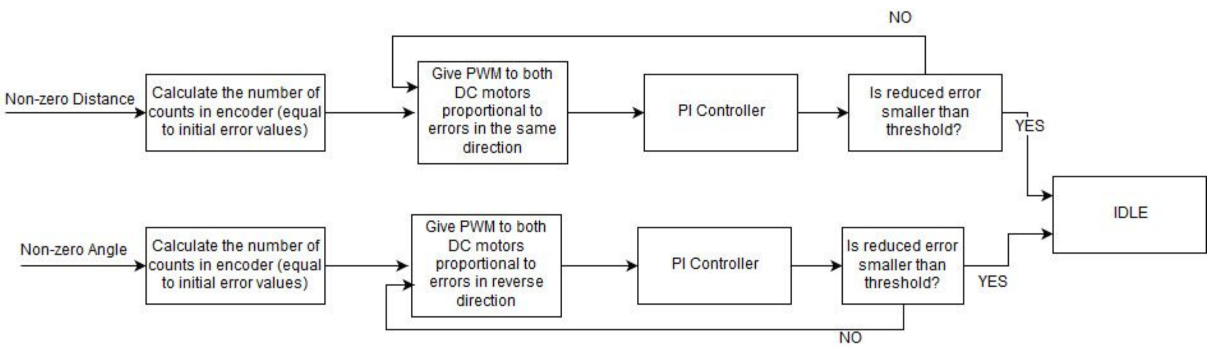

Figure 14: Flowchart explaining the working principle of motion unit

4.4.1. Motor Drive Sub-Unit

The movement of the robot is adjusted by giving desired rotations and distance information to reach a destination point. To reach the desired destination accurately, PI controller is used. In this way, zero steady state error and minimum oscillations is achieved. The controller takes the required rotation and distance information for the movement in the field as its inputs and the error value is provided by the encoder subunit which will be explained in the following part.
The closed loop system initializes errors for each motor with the desired distance value. At each step, it gives PWM to motors proportional to error values and error decreases as encoder count gets closer to the desired value. By adjusting the coefficients of proportional and integral controllers, we were able to make smooth movements with less error and to reach the destination without overshoot which is an unwanted result. In this way, more accurate results are obtained.

Performance Results

As it is expected from the PI motor controller, robot is able to compensate steady-state error and the disturbances which are not causing any slippage on the wheels.

4.4.2. Encoder Sub-Unit

Encoder sub-unit collects the rotation data from the magnetic encoders attached to the DC motors. These encoders are sensing the magnetic change in the magnetic discs, where both of them has 11 count per rotation resolution, attached to the rotor of the motors. This unit counts the magnetic change obtained by the hall effect sensors of its and returns the amount of rotation that the motors did to provide error value to the PI controller in the motor drive sub-unit.For an ideal case, it is expected 0.02 cm error from 75:1 motor gear ratio and encoder resolution of 11 per revolution, which we are using in our robot.

Performance Results

The physical structure of the encoders are returning correct amount of counts as stated in data sheet (823 count per rotor turn).

4.5. SELF LOCALIZATION AND MAPPING UNIT

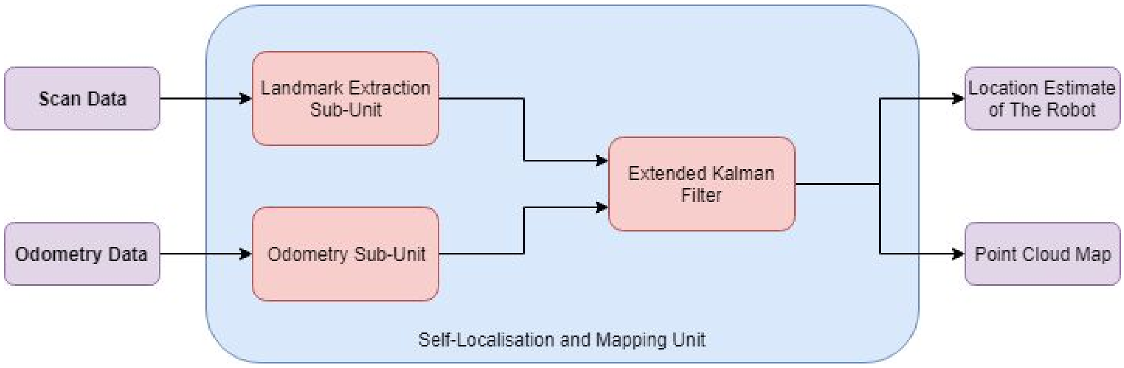

Figure 15: Block diagram of SLAM unit

Fourth main unit is the self localization and mapping unit. This unit collects the odometry data provided from the encoders, and the scan data taken from ToF sensor to estimate the position of the robot globally in the map, estimate the position of the objects, reduce and optimise the possible error occurred during the estimation process, and provide a point cloud map data for visualization unit. To achieve this goal, three main sub-units is used.

4.5.1. Landmark Extraction Sub-Unit

To identify the objects from walls in the surroundings of the robot, this sub-unit processes the scan data. Since it is already known, the shape of the map will have convex geometry, any spike at the data which yields a concavity in the map will be considered as an object.

Firstly the set of two consecutive scan data elements' difference is calculated as plotted in figure 16. Then the peaks of the new data set is taken into consideration. For an object, it is expected that the measured distance values first decrease and then increase to the walls distance, as observed in the figure 17.

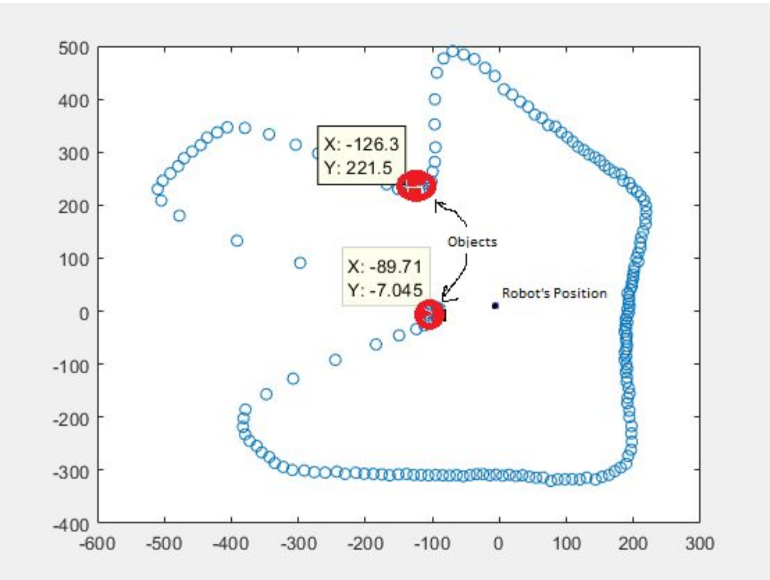

Figure 16: Test Scan for Landmark Extraction

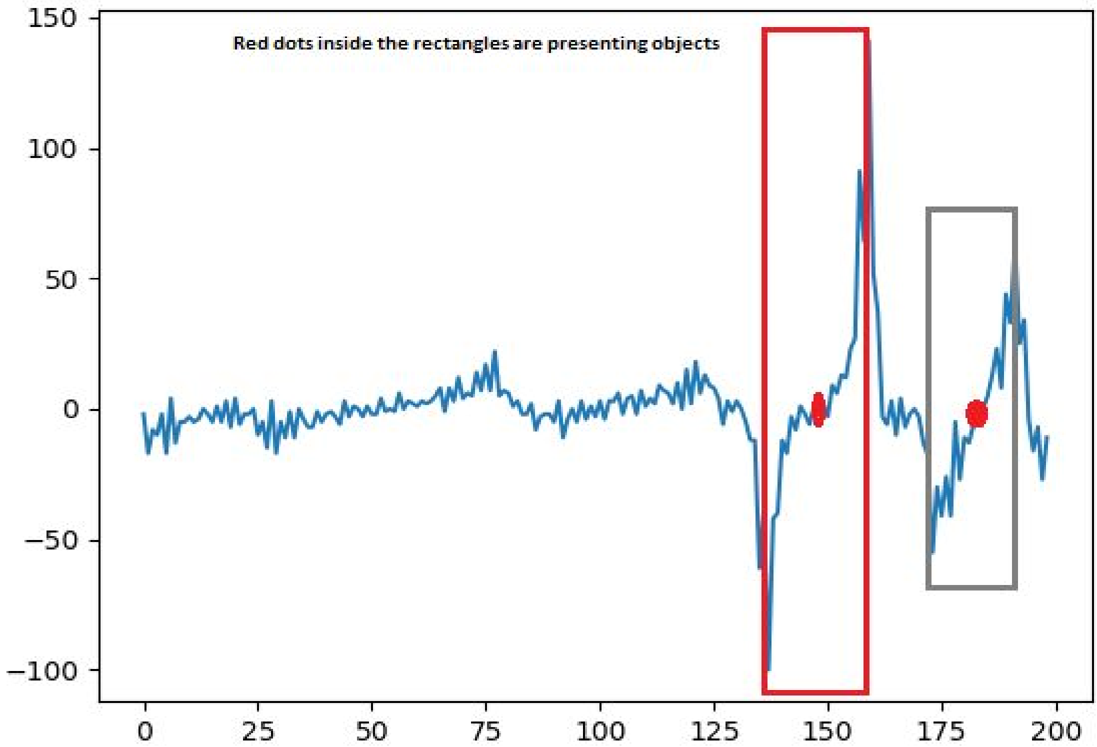

Figure 17: Difference plot of the distance measurements

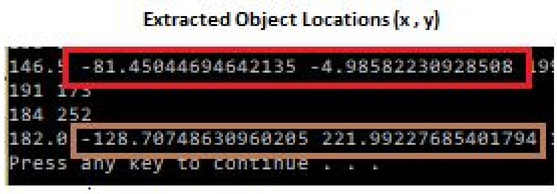

Figure 18: Location results of the closest points of the objects returned from landmark extraction algorithm.

Performance Results

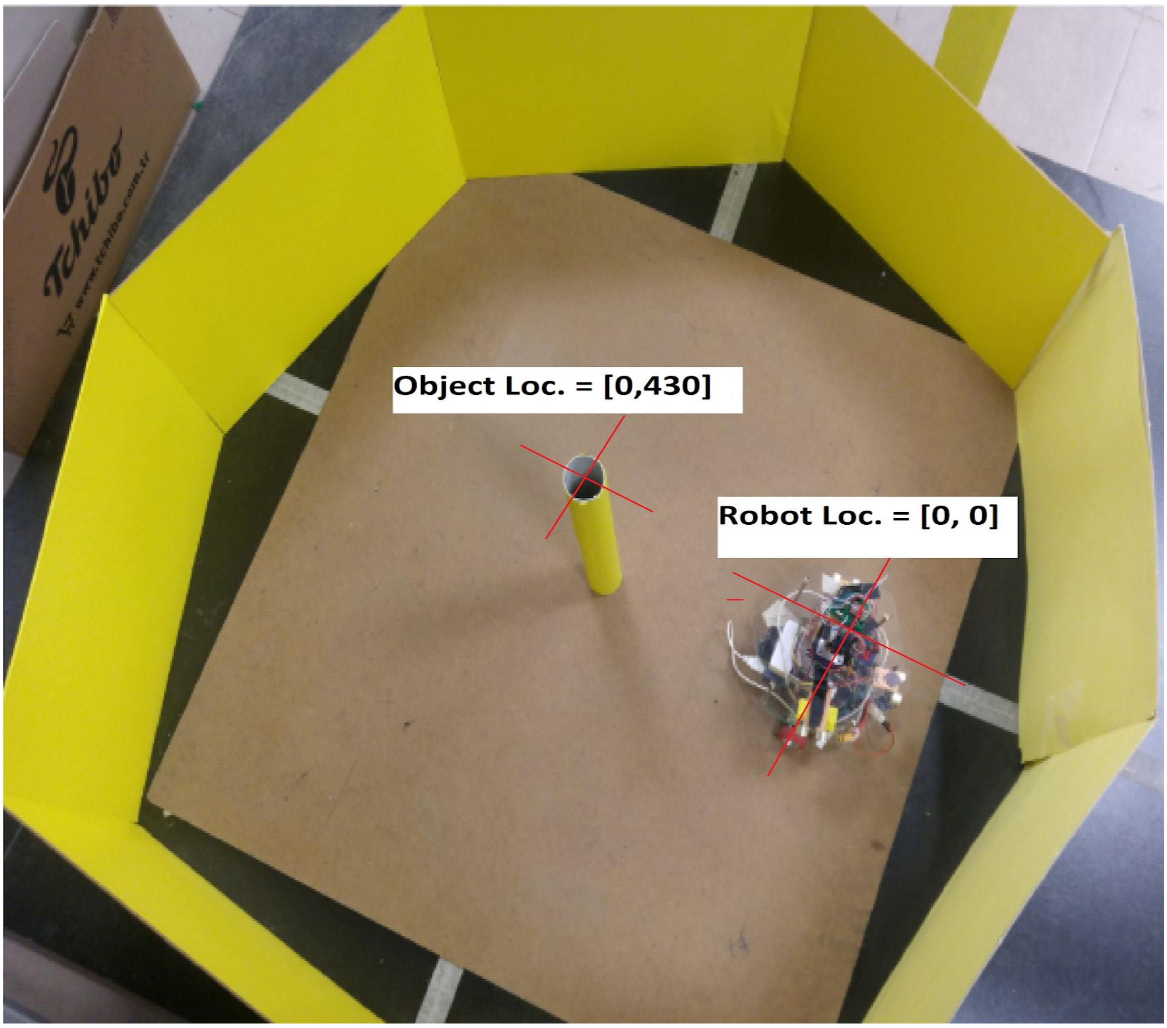

Figure 19: Bird's eye view of the environment with the measured locations of the robot and the cylindrical object

By testing the landmark extraction algorithm's performance on the given test area given in figure 19 we obtained the following center location for each object type;

- **Triangular Object:**

```
objMtx=,oldMtx [[-0.228948736636211, 431.5390591399221, 0]]
```

As it can be seen from the result, algorithm didn't face any problem in locating the triangular object with error of 1.5mm in y-axis. However, testing results of this object changes within a range of 0-20mm due to the orientation of the object.

- **5-cm diameter Cylinder Object:**

```
objMtx=,oldMtx [[41.89408056437446, -438.0970846524363, 0]]
```

With the 5cm cylinder, algorithm locates the object with around 2.5 cm error in x-axis and 8cm error in y-axis. The error is considerably large due to the small amount of

- **Square Object:**

```
objMtx=,oldMtx [[-12.41322285453619, -404.87918543719036, 0]]
```

As it can be seen from the result, square object's center location is found with a error of 1.2 cm in x-axis, and 2,6 cm error in y-axis. Although the error in the x-axis is negligible in this case with the consideration of misplacement of the robot in testing, the error in the y-axis is is deviating according to the rotation of the object, which determines the closest point of the object to the robot.

- **10cm Cylinder Object:**

```
objMtx=,oldMtx [[-0.1780071788988038, 442.66652913017344, 0]]
```

With this object, algorithm detects the object location with less than 1mm error in x-axis, and 1.2cm error in y-axis, which meets our objectives.

Although the estimated center points of the objects are logged during the mapping process, final results of the center points are calculated by using K-means clustering algorithm.

4.5.2. Odometry Sub-Unit

Odometry is the use of data from motion sensors to estimate change in position over time.This sub-unit uses the revolution count data taken from the encoders of the robot and maps it to certain change in the x & y location and rotation of the robot. Magnetic encoders attached to the back of DC motors' rotors have 11 count resolution per revolution from its two hall effect sensors. The motors also have 75:1 gearboxes; therefore, the rotation and the distance that the robot locomote can be calculated with the following equations, where the rotation radius of the robot is 9cm, also please note that with this components and 4 cm radius wheels, the measurement resolution of the encoders are 0.02 cm ;

Distance (cm): (2 x 11 x 2π x 2) x (Desired input distance)
Rotation(degree): (22 x 2π x 9)x (Desired input angle)

4.5.3. Scan Matching and Kalman Filter

Since the localization of the robot is crucial for constructing an accurate map, an probabilistic estimator to consider possible error deviation occurred due to wheel slippage, accumulated encoder count error or distance measurement errors. For compensating this error, extended kalman filter (EKF) will be used. With this EKF, relative location of the landmarks, which are the convex-hull points of the map stated as an convex shape, and the odometry data taken from odometry sub-unit will be associated to update state matrix of the robot according to the covariance matrix, which is initialised by using the actual error characteristics of the final product. As the output of this unit, a more precise location estimate of the robot is returned, and the required offset data to combine each scanned point cloud returns.

The overall self localization and mapping unit will follow the given flowchart in Figure 20.

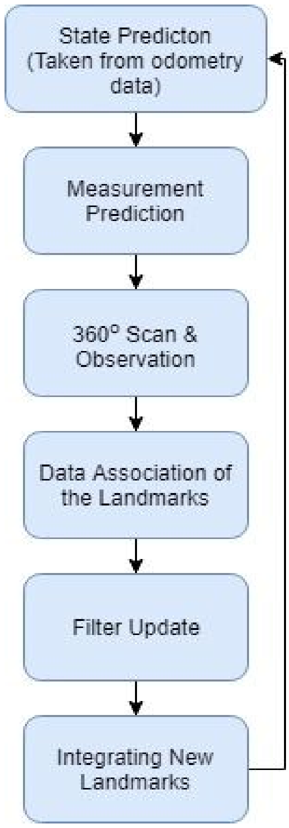

Figure 20: Flowchart explaining the SLAM unit's operation

- **State Prediction (from odometry sub-unit):** 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 (on EKF):** After movement, self localization and mapping unit will have a predicted robot location due to the scan matching of the last scan and map point cloud.
- **Scan & Observation (from landmark extraction sub-unit):** At its new location, robot makes another scan and locate object visually.
- **Data Association (on KF):** Unit associates the taken data from state prediction and measurement prediction.
- **Filter Update(on KF):** Due to data association results, kalman filter coefficients will be updated
- **Integrating New Landmark (from landmark extraction sub-unit):** Unit integrates the new observed objects and returns the state prediction step.

4.5.4. Error Corrector

Error corrector was a sub-solution we have developed which performed up to 0.5cm offset correction on x and y direction and 3 degree angle offset on rotation. It is a solution that gives really high accuracy, however it takes quite time for raspberry pi to run this algorithm. Because is is a brute force map checking algorithm. In other words, the map is shifted by by the means of rotation and distance. Then this map is checked with the previous measurement, when the maximum matching is found these offset values are returned. The reasons why this implementation works well are the borders. Most of the measured points are borders so checking their relative position gives good matching results.

However since this algorithm is required after each measurement and it takes quite time. Still instead of kalman filter this approach is used.

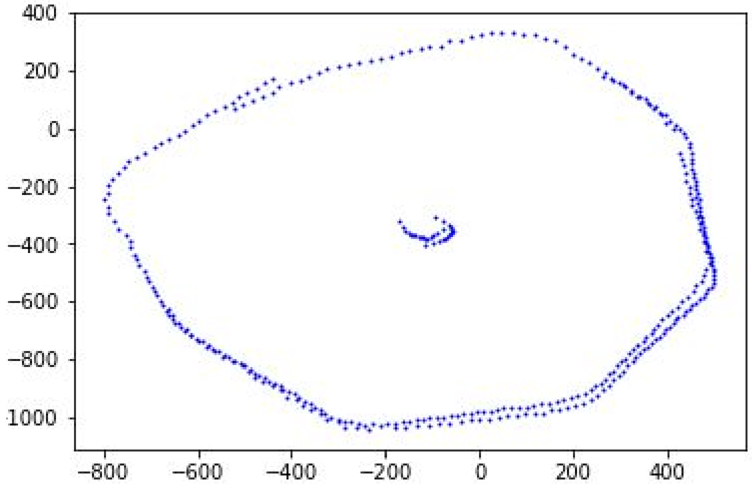

Figure 21: Obtained measurements before offset correction

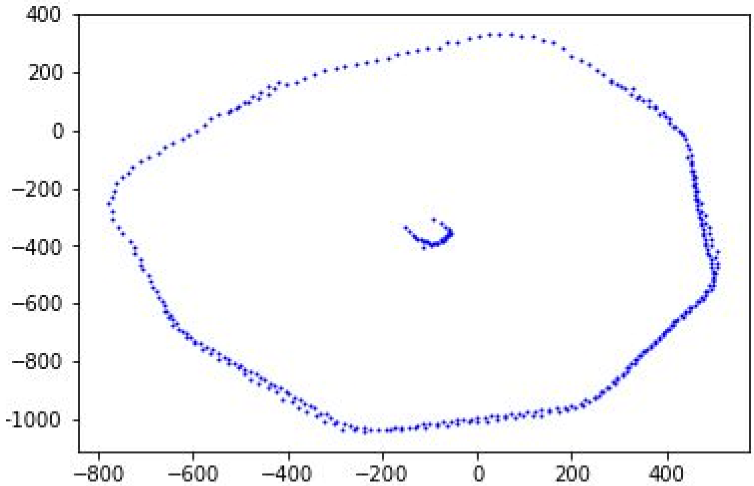

Figure 22: Obtained measurements after offset correction by error correction algorithm

As it can be seen from Figure 21 there is an offset error between two measurements. It is not much in this case but sometimes bigger errors occurs. After using the error corrector algorithm the offset correction is done both in rotation and distance and the Figure 22 is obtained. By examining the walls the error correction accuracy can be seen. The directional error is less than 0.5cm in the x and y axises even in the worst case. The rotation error is less than 3 degree.

4.6. VISUALIZATION UNIT

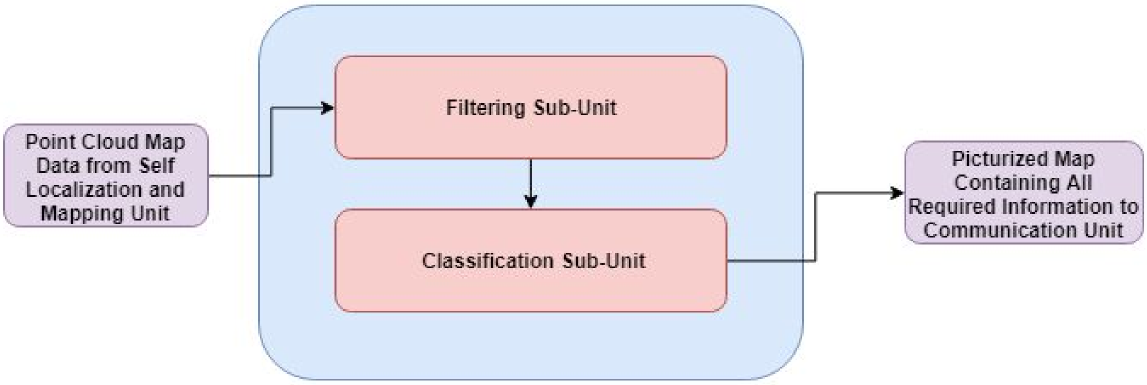

Figure 23: Block diagram of visualization unit

Visualization unit forms the fifth main unit of the system. Its mission is to take the point cloud data as input after each scan and turn it into the final version of the map in the end. To do this, it uses its two subunits which are filtering and classifications subunits. Filtering subunit simply filters the point cloud data and classification unit decides the desired properties of the objects such as their centers and shapes by using the K-mean clustering algorithm.

4.6.1. Filtering Sub-Unit

Because of the measurement characteristics of the ToF sensor, some undesired distance measurements occurs in the point cloud data. Filtering sub-unit is used in filtering these measurements in the point cloud data. In this unit, 1 millimeter is modeled as 1 index in the point cloud data. For instance, if 1 meter length distance is measured, a list with 1000 element is initialized. For a square map with 1 meter width and 1 meter height 1000x1000 sized object is created. When dimensions of a possible map is considered, Raspberry Pi can easily handle such sized lists. Since the ToF sensor gives noisy data which stems from the fact that it cannot detect sharp edges precisely and gives average distance data, the need of eliminating the noisy data occurs. To obtain a final clear map, morphology method is used to filter the data and get more accurate results. The method works in this way: A kernel is convolved with the point cloud data and a threshold value is determined. If the kernel has a value above the threshold, the data is checked by a second filter which has a wider dimension. If the points can also pass the second check, they are kept on the dataset. Unlike the previous filtering, using the second check removes almost all the noisy points (up to %97, 97 noise points out of 100 when the noise points between the borders and the objects are considered). If kernel has a value below the threshold the data is removed from the dataset. In this way, undesired noise can be eliminated. This morphology operation is brute coded and cv2 library of Python is not used.

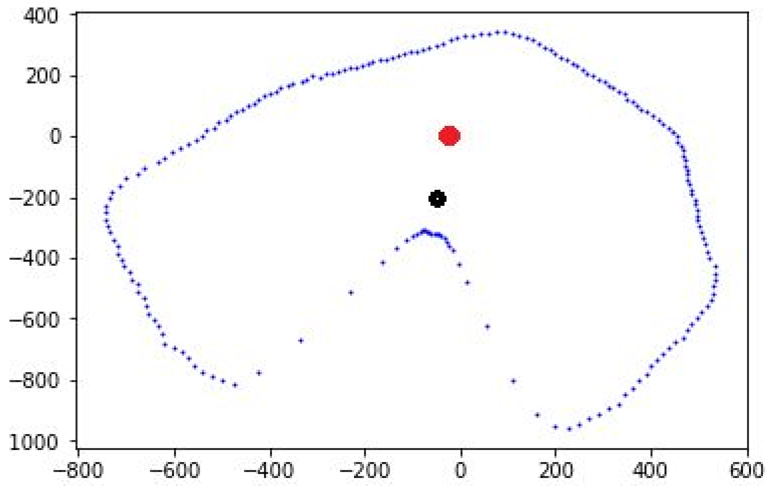

Figure 24: The point cloud obtained after first measurement

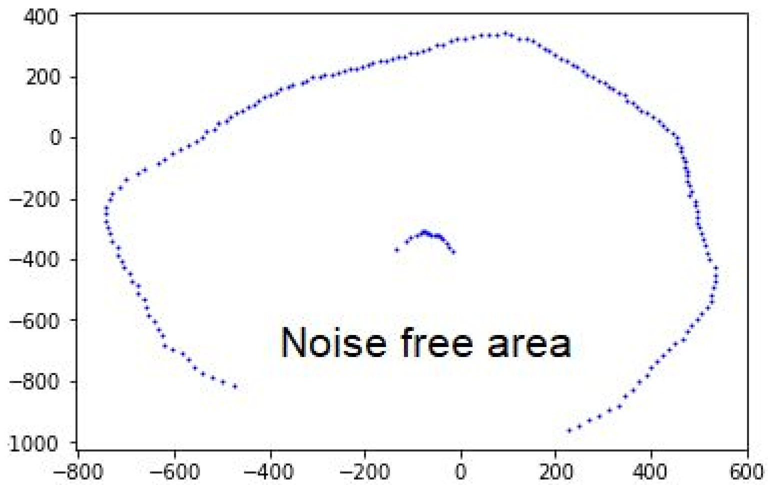

Figure 25: Filtered version of Figure 24

The red dot represents the initialization point of the robot which is 0,0 point. Then the robot moves to a closer point which is shown by the black dot. Then The measurement is done. The object is square. After measurement a noisy result is obtained. When the filtering algorithm is used, the noises are eliminated and the result given by 25 is obtained. It shows a single side of the square object.

4.6.2. Classification Sub-Unit

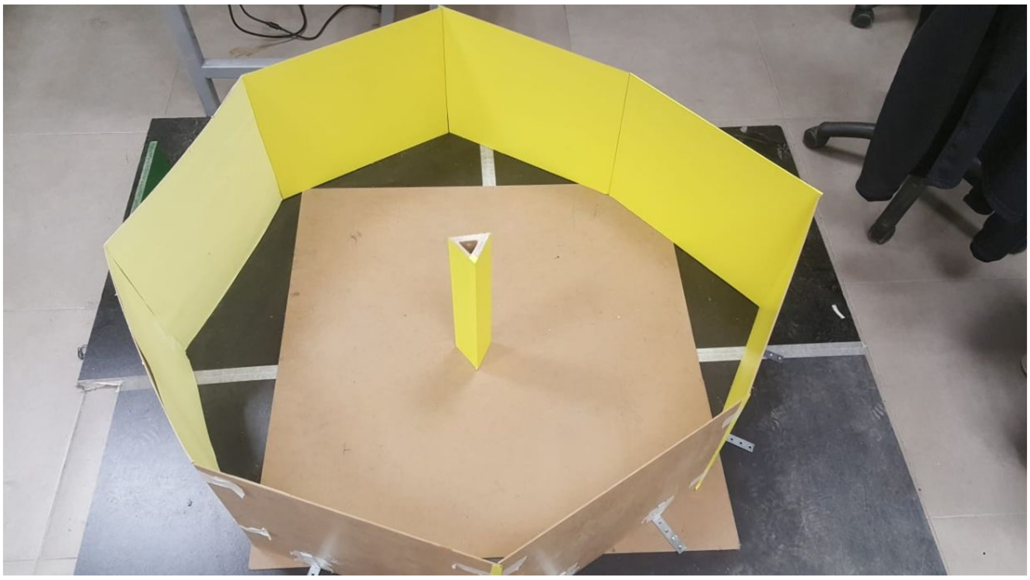

Figure 26: Test environment picture

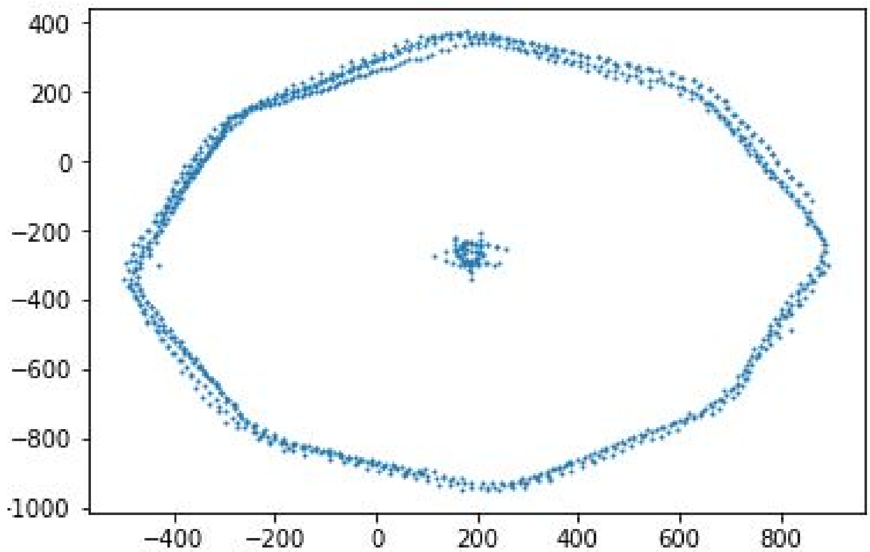

Figure 27: The raw point cloud data of for single triangular object

Classification sub-unit uses K-mean clustering algorithm on the filtered point cloud data by separating it into several sections to determine the centers of the objects precisely after the object locations and total number of objects are determined. To do so, the algorithm excludes the wall data from the point cloud and only keeps the objects' data first. With the help of Hough transform, the outer borders are found and separated from the rest of the data by giving proper threshold values.

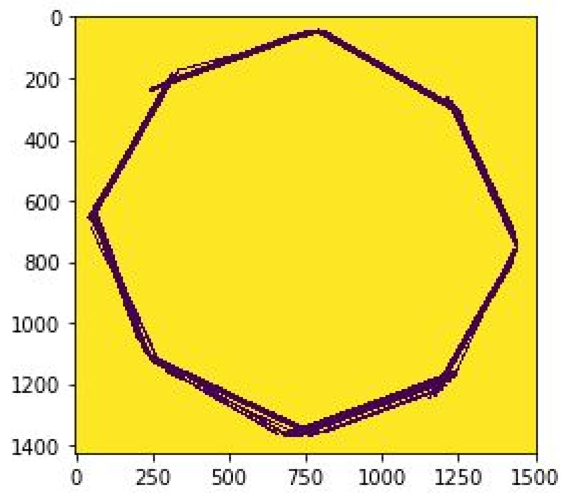

Figure 28: Obtained outer borders (walls) of the map

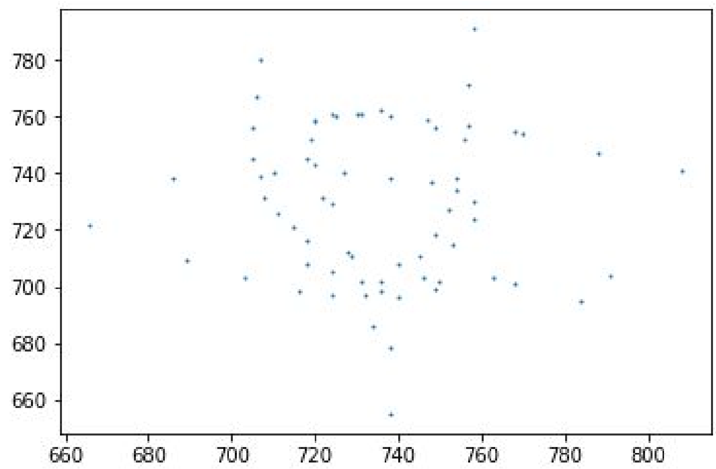

Figure 29: The object points after they are seperated from the outer borders in point cloud form

After separation, K-means clustering algorithm is used to find the centers of the objects from the data clusters that forms them by using the total number of objects that is taken as input by the algorithm. In addition a method called as elbow method is used to double check this input. Result of elbow method can be seen in Figure 30. It can be seen there is a sharp fall and no breaking point at all since there only a single object. It basically finds the number of object by initialization clusters with several numbers and picks the value when there is a big gap between the converged centroids. For this purpose sklearn library of Python is used because of its speed and reliability. Since the distance between object points and its centroids varies from object to object, each distance data has a different mean and variance. By using that information, objects can be distinguished from each other.

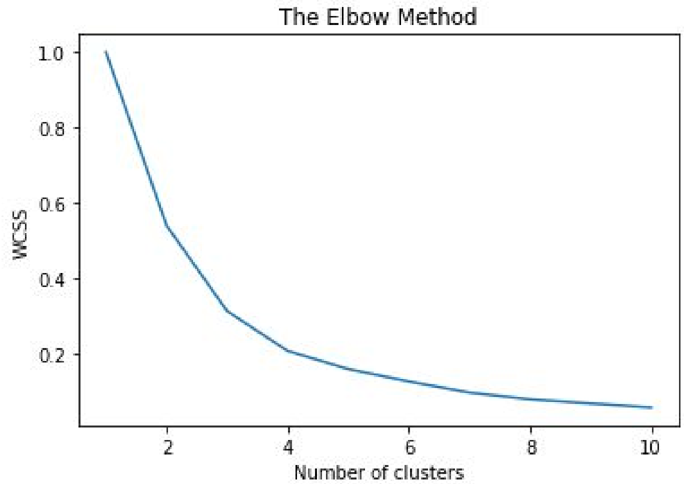

Figure 30: The result of elbow method

Since we already know the object perimeters, after finding the centers of objects, the center location can be double-checked if it matches with the found result. For instance if a cylinder and its centre is found, it can be convolved with a foreknown cylinder kernel to check if it returns a high value. A Priori knowledge of the object types is the main advantage for this type of an algorithm since the produces algorithm only needs to do matching of the scanned objects. For a more developed version of our product, object classification manner without any foreknown information should be implemented.

For the final version, the number of objects, their perimeters and centers will be shown on the map.

This is triangle, coordinates of the centre are x=185.0 , y=-266.0

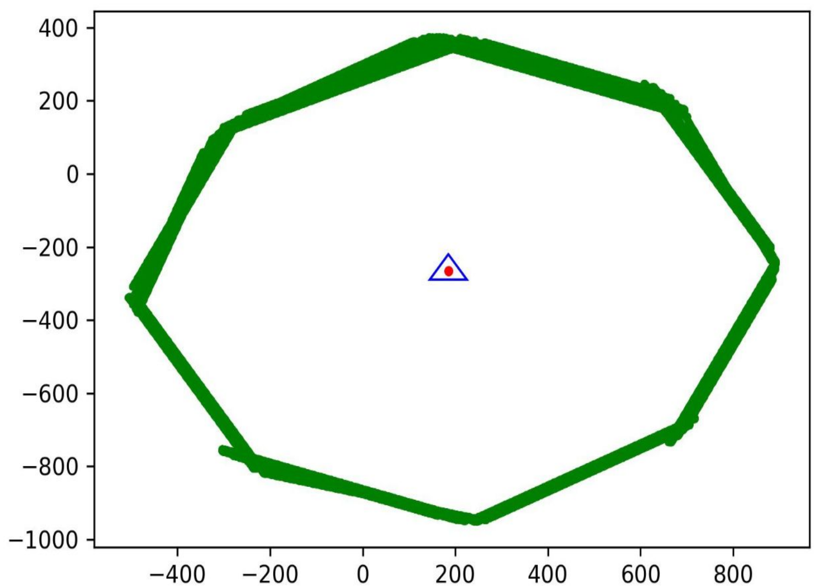

Figure 31: The final result of the scanning of a map with a single triangular object

4.7. COMMUNICATION UNIT

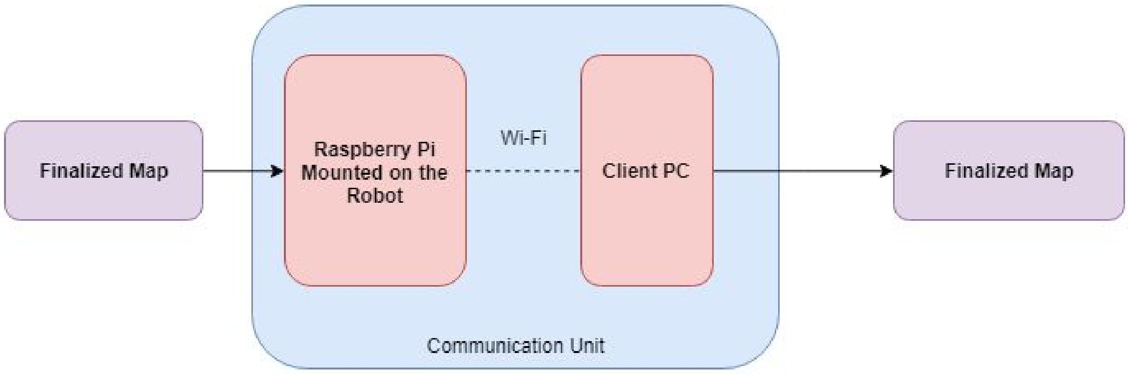

Figure 32: Block diagram of communication unit

Communication unit forms the fifth and last main unit of the system. It enables the robot to send the final version of the map of the environment to a computer screen. This map is the only and final product of the whole process, hence the proper display of it is very crucial since displayed result is the reflection of all the complicated process that is being held.

After the communication unit completes its mission, the user can see the final map with the related information about objects such as their types and centers' location written on the top.

As it will be mentioned in the user manual in Appendix, only user interaction will be in the beginning by pressing a button to start the process. After the scanning process has ended, the obtained map in JPEG form will displayed in the screen for the user.

The communication process is held through by Secure Shell (SSH) protocol.

5. Circuit Diagrams and 3D Technical Drawings

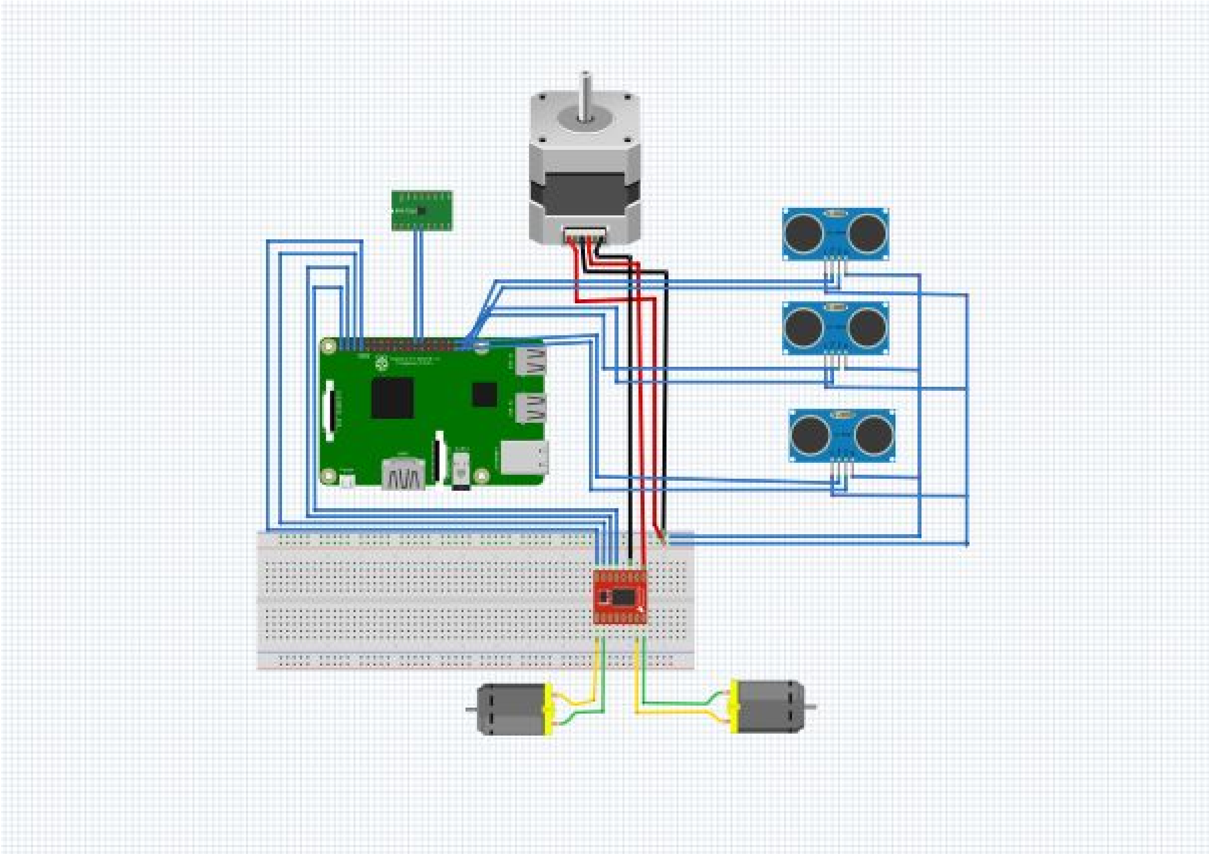

Figure 33: Circuit diagram

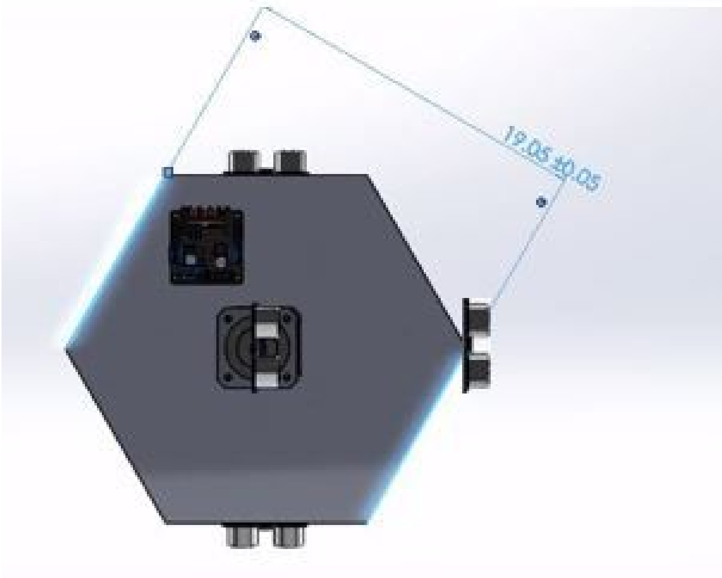

Figure 34: Technical drawing of the robot from top view

Figure 35: Technical drawing of the robot from front view

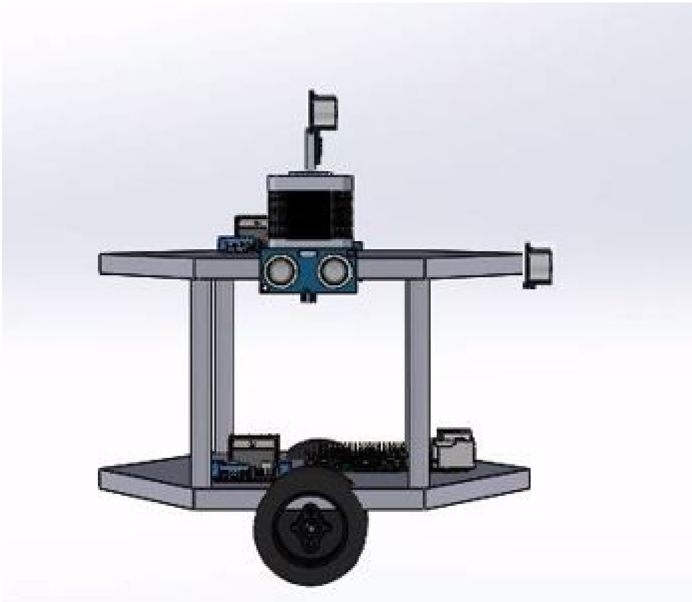

Figure 36: Technical drawing of the robot from side view

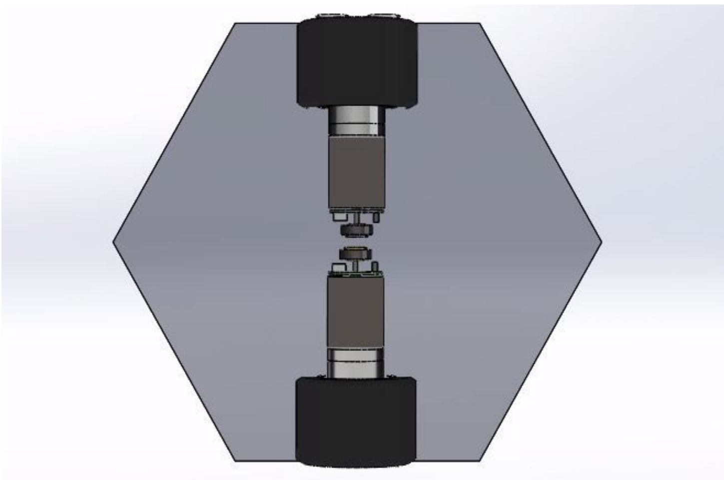

Figure 37: Technical drawing of the robot from bottom view

6. RESULTS & ANALYSES OF PERFORMANCE TESTS

When there is a single object, we have managed to distinguish each object. For 8 different trial at each case the object is distinguished correctly. In terms of centre, the numerical error results are given in the Landmark extraction part. An example of final plotted map is given in the classification sub-unit part for a single object. The error percentage varies between objects. The worst error performance is obtained for small circle.

The desired performance metrics are obtained in the overall if there is a single object except the rotation information of triangular and square objects.

-The centre of the objects are found with less than 8mm error

-The overall run time is less than 20 minutes.

-Robot functions without the need of recharge for 30 minutes

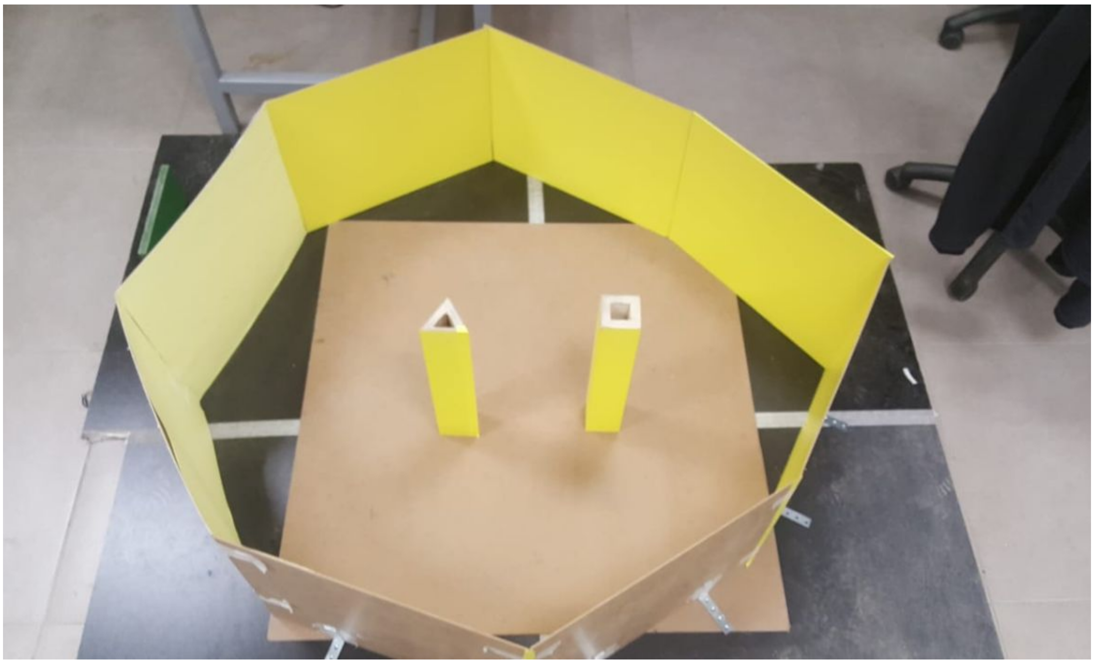

Figure 38: The testing environment for two object.

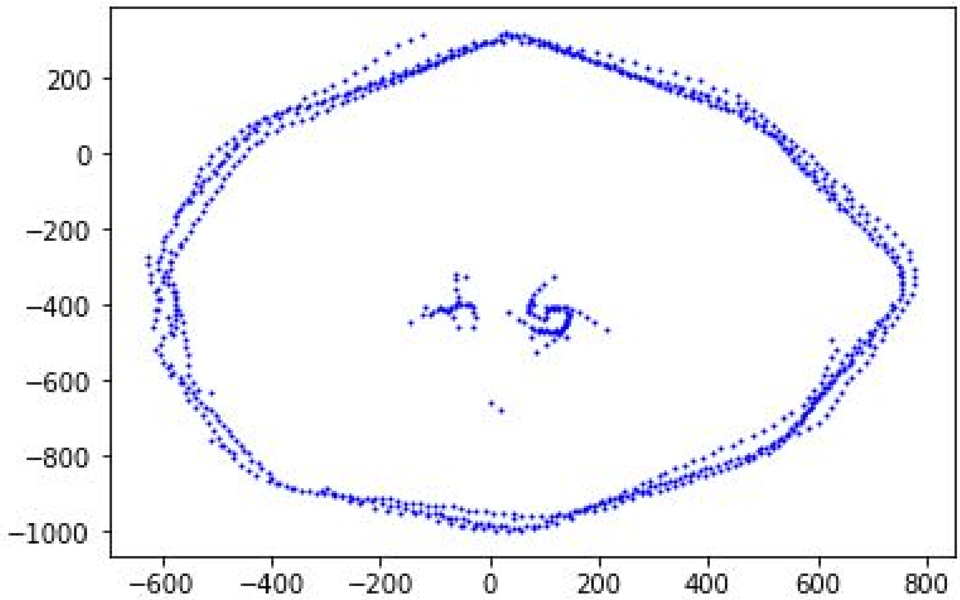

Figure 39: Raw scatter plot data after measurements

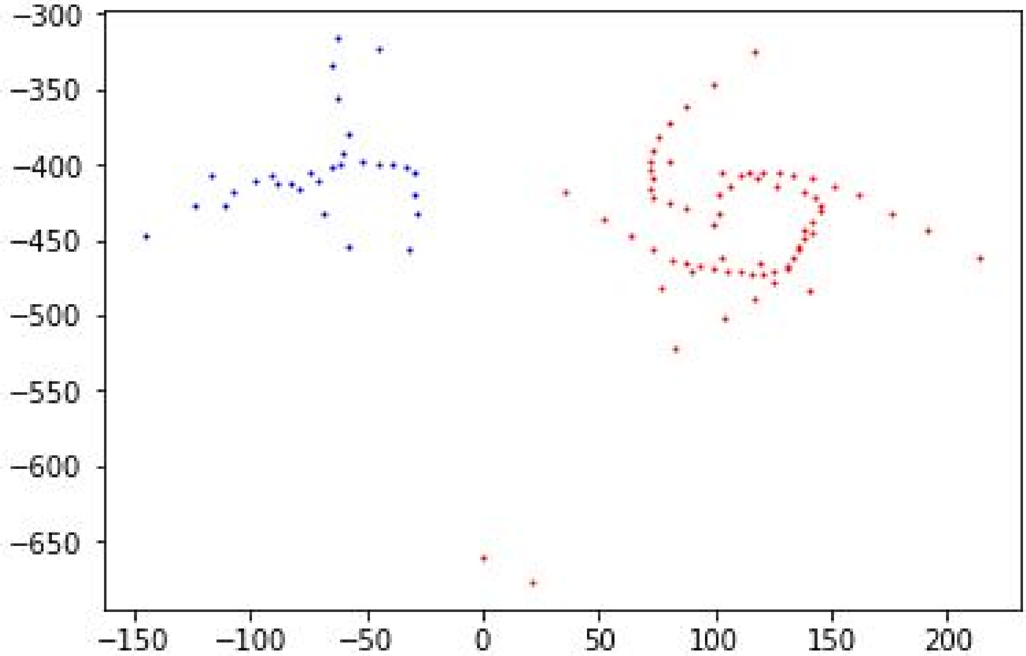

Figure 40: The Classified objects by K-means when the outer borders are seperated

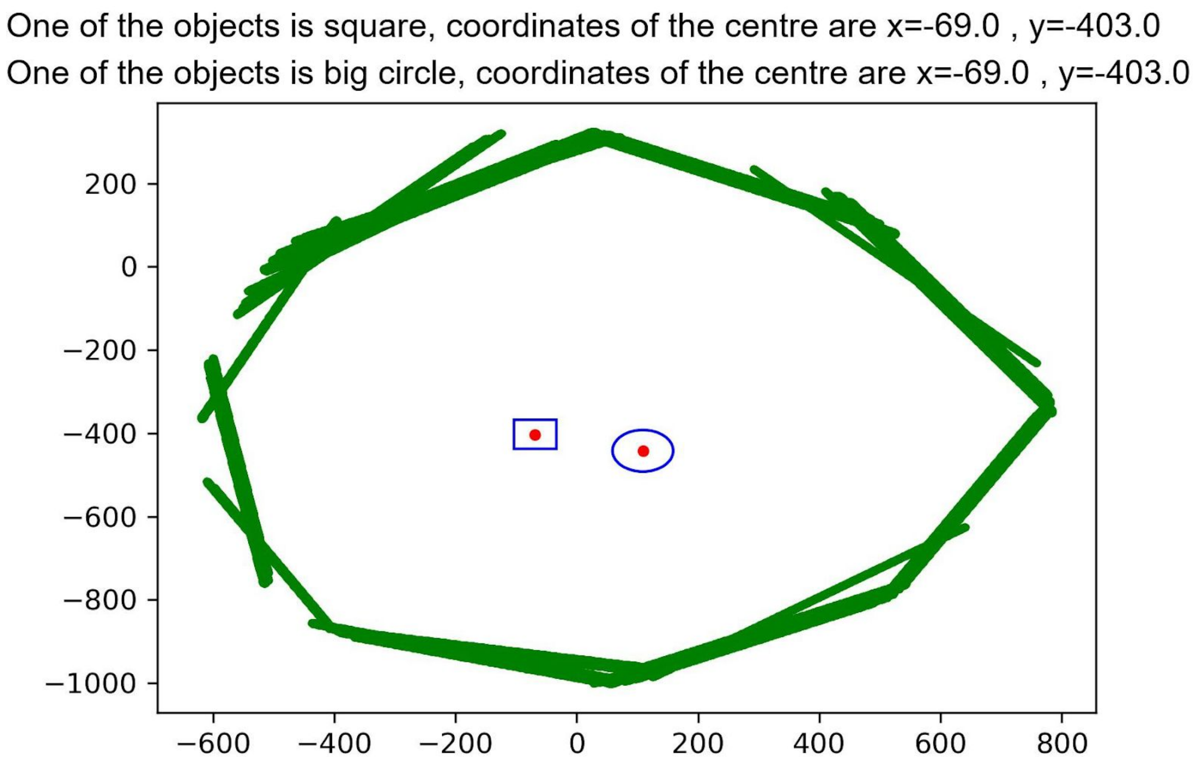

Figure 41: The final plotted map of the objects.

For this two object case, error corrector was not run because it took quite long to finish the overall measurements. As a result the raw data given in figure 39 is obtained. since the borders are not matched well, during hough transform the outer borders could not be obtained clearly. Still, the objects could be found, seperated from outer borders and each object could be distinguished from each other. One is shown by blue the other one is shown by red in figure 40. However, due to noise the classification failed. As it can be seen in the bottom part of figure 40 there are two noisy points in the bottom which results as a higher variance. For that reason the object was classified as big circle. Also in the triangle case, the measurements don't match onto each other as it can be seen in figure 40 by blues dots. Due to offset error it is found as square. As a result, centres were found quite shifted which does not meet 8mm criterion. At the end the figure 41 is plotted. Both of the objects are wrong and the centres are not correct.

To sum up for single object, the criterions are met. However when there are more than an object the desired results could not be satisfied.

7. DELIVERABLES:

List of deliverables are stated below:

- 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 (1 cylinder of 10 cm diameter, 1 cylinder of 5 cm diameter, 1 square prism with 7 cm edge length, 1 prism with an equilateral triangular base of 8 cm edge length)

8. BUDGET

8.1 Actual Expenditures

Cost of final product can be seen in Table 2.

Table 2: Detailed budget of the project

Component	**Quantity**	**Total Cost**
Raspberry Pi 3B+	1	**40$**
VL53L0X time-of-fight sensor	1	**9.95$**
HC-SR04 Sensor	3	**4.5$ (1.50$ each)**
Stepper Motor	1	**10.5$**
DC Motor with Magnetic Encoder	2	**42$ (21$ each)**
High Friction Wheels	2	**6$ (3$ each)**
Li-Po Battery(2s 1350mAh)	1	**16$**
Plexiglass robot body	1	**13$**
Jumpers		**3$**
Stripboard		**1$**
Header		**3$**
Total		**148.95$**

8.2 Total Cost

In addition to robot's construction budget, additional expenditures can be seen in Table 3. Testing kit which includes the walls and objects, materials used while covering them, broken and replaced components such as the stepper motor, burned lipo charge etc. are stated.

Table 3: Additional expenditure budget

Expense Units	Total Cost
Robot's Components	**149$**
Testing Kit (Walls & Objects)	**16.38$**
Brace Clamps	**2.95$**
Light Yellow Matte Vinyl Coating 2m^2	**9.01$**
Replaced Components (stepper motor,charger etc.)	**20$**
Software Engineering	**10$**
Hardware Engineering	**5$**
Total	**192.34$**

9. CONCLUSION

This final report is a summary of all work that our company members conducted for the last 6 months. It presents all details related to the final product of our project. Top-down system description is given with all algorithms that are created in order to meet the project requirements. Given step-by-step performance test results and analyses ensures the operating principle of our product. All of the used methods and components used during the creation process are stated clearly. Also a user manual is created considering all possible questions and problems our users might have in the future while using our product.

10. APPENDIX

USER MANUAL

Introduction

Dear customers,

Thank you for choosing the Thunderbolt Autonomous Map Extraction Robot. Please read all instructions before using your robot.

This product is a high-intelligence autonomous robot based on high precision time of flight sensor combined with ultrasonic sensors,working together through localization algorithms to realize map extraction of any unknown surrounding, display it and identify the detected objects. Please follow the safety and operating objectives carefully to have good performance results. Enjoy your robot!

Main Unit

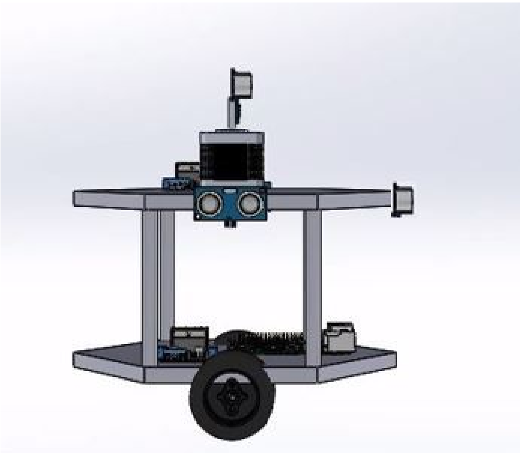

Figure A.1: Robot prototype that will be delivered

Safe Handling Informations & Usage Restrictions

Please read the following warnings before attempting to move or use your Thunderbolt AMER.

- This autonomous map extraction robot is a prototype of a larger model which is suitable for every environment, hence this delivered prototype is only suitable for indoor usage. It will provide best results on a flat and clean surface away from direct light sources. Avoid direct sunlight and light sources.
- You can extract your own maps, but using the provided set of walls and objects again recommended to obtain better results.

- The robot contains 2 power sources: 1 Li-Po battery and 1 Powerbank. For optimal performance results, use the robot when it is fully charged.
- Incorrect handling, such as dropping the robot, can cause data loss and invalidates the warranty.
- The robot is for indoor use only. Do not use this robot outdoor, non-ground and drop-off environments without physical barrier around.
- Do not use the robot when the ambient temperature is higher than 40 or lower than 0 degrees celsius or if there any liquid or viscous materials on the floor.
- Please disconnect all the cable before the robot starts running.
- Please supervise children and pets as far as possible when robot is operating.
- Do not put any objects on the robot.
- Use this product following the given instructions.
- Be sure to remove internal processor connections while charging the robot.
- If the power cord is damaged or broken, stop using it immediately and replace it with a new one.
- If the product needs to be transported please make sure the robot is turned off. It is also recommended to pack and carry the robot in its original packing box.
- If the robot is not used for a long time, please fully charge it before using it.

Thunderbolt AMER Components/Packing List

1) Main Unit x1
2) Power cable/adapter for Li-Po battery x1
3) Power cable for power bank x1
4) USB 2.0 cable x1
5) User Manual x1
6) Testing kit:
    I. 50x40 cm light yellow coated wall x6
    II. 8 cm edge triangular prism x1
    III. 7 cm edge square prism x1
    IV. 5 cm diameter cylinder x1
    V. 10 cm diameter cylinder x1

Getting Started with Your Thunderbolt AMER & Operating Instructions

You can get started with your Thunderbolt AMER in a few easy steps.

- Replace the robot in the environment whom you want to obtain the map. If the robot is facing towards a wall in the beginning, the results will be better.
- Press the button to start the scanning process and wait until the robot finishes exploring the map.
- The map will be displayed on your computer screen with proper object labels.

Technical Specifications

Name	Parameters
Dimensions	19x22x20 cm (Width x Length x Height)
Product Weight	1.650 kg
Battery Capacity	1350mAh
Rated Voltage	7.4 V
Rated Power	24,3 Watt (3.86A)

Getting Help

For additional help with AMER, its operation and the software, please contact the following number:

Thunderbolt 24/7 Customer Assistance: +90538310501

Thank you for choosing Thunderbolt Corp.!