Optimization of Area Coverage for Field Exploration

Ali Imran, Wenye Ouyang, Ranvir Bhogal, Arief Ramadhan

I. Introduction

Exploration is one of the domains where robotics has a very promising future. There are a number of issues inherent to this task. Exploration is intensive and time-consuming. Using humans to explore such large areas is infeasible. The level of difficulty of this task is proportional to the nature of the terrain. Moreover, if the exploration operation is to be carried out underwater or in outer-space, using autonomous agents is the only option. Also, there is a greater amount of inaccuracy associated with the mapping techniques and results collected by humans perform the exploration operation.

In this experiment, an underwater exploration operation has been emulated. The task of the robot is to explore an undersea area which is believed to contain a sunken ship loaded with treasure. The location of the sunken ship has been narrowed down to an uncharted area within a radius. The robot is not capable of retrieving the treasure; however, it must effectively inform a subsequent recovery team. Recording undersea features, such a prominent rock, undersea pipelines, and scattered ship debris may assist in effectively describing the area and the treasure location. Since the tests are being in a controlled environment, inside a lab, the aforementioned real objects are represented by obstacles, lines and RFID tags respectively.

II. Aims and Objectives

A. Aims

Evaluating exploration methods is difficult because the exploration problem is comprised of many problems that need to be addressed. Multiple subsystems work in parallel to ensure that the task of exploration is performed efficiently and accurately. Moreover, exploration is an open-ended problem with no generalized way of achieving the goal. Therefore, this experiment focuses on one approach to accomplish the goal. We believe that aiming at achieving efficient area coverage would lead to the desired results. Successful area coverage would ensure that all the places on the map have been visited, already visited block are not re-visited. Also, if the complete area is covered, it will ensure the detection of the RFID tags, leading to the treasure we want to find.

B. Objectives

1) To find and implement motion planning and the pattern which the robot follows to explore the environment.
2) To implement accurate kinematics for the robot in order to achieve accurate mapping.
3) To detect and map obstacles and other features on map.

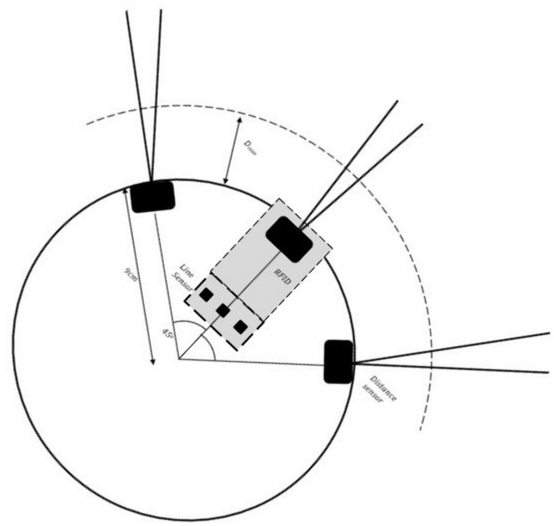

Fig. 1: Physical Model

III. Methodology

This section enlists all design parameters and methods taken into account for optimisation of demo 2. Details of demo 1 are discussed in detail in the results section.

A. Physical Model

Fig. 1 shows the physical model of the robot. The system is equipped with three proximity sensors, placed at an angle of 55 degrees from the axis. Line sensors and an RFID tag sensor are mounted on the lower side of the robot to map the lines and detect and identify RFID tags respectively. Calculations for the kinematics are done with respect to the center of the robot. Therefore, for accurate modeling of the systems, the sensors on the boundary of the robot are transformed using the following transformation equations:

$$Xl = \begin{bmatrix} cos(55) & -sin(55) \\ sin(55) & cos(55) \end{bmatrix} X' + \begin{bmatrix} 5.16 \\ 7.37 \end{bmatrix} \tag{1}$$

$$Xr = \begin{bmatrix} cos(55) & -sin(55) \\ sin(55) & cos(55) \end{bmatrix} X' + \begin{bmatrix} 5.16 \\ -7.37 \end{bmatrix} \tag{2}$$

$$Xf = X' + \begin{bmatrix} 9 \\ 0 \end{bmatrix} \tag{3}$$

The line sensor module has three sensors. The center line sensor was used to detect lines for mapping purposes. Only one sensor was utilized as we did not need to follow the lines and only detection of their locations was required.

One of the major tasks before carrying out the experiment was to set up the distance sensors. The output and input value of the sensor have a non-linear relation between them, as shown in Eq.4. The effective range of he sensor is between 6.5cm-15cm (approx.). The output values from the sensors are of no use when it is closer than 6cm from the obstacle.

$$x = (y/12.494)^{(1/-0.66)} \tag{4}$$

B. Experimental Setup

Testing has been done on a 1.8 x 1.8 m arena, appropriate for the size of the robot. Stacks of rectangular shaped acrylic sheets were available to act as obstacles within the environment. The base of the arena was white with a matte finish, making it easier for the robot to detect lines. Fig- shows the picture of the arena with the robot and obstacles inside. To simplify the problem of exploration, the arena has been divided into a grid, making it easier to evaluate the performance of the overall system.

In the first experiment, the arena was divided into 25 by 25 grid. However, due to the size of the Romi, which length and width was twice the size of that grid, some of the explored grids were not written into the map. We divided the arena into 18 by 18 grid in the second demo to eliminate this problem.

C. Motion Planning and Kinematics

Motion planning is one of the crucial aspects of exploration. Designing a behaviour which lets the robot explore the area in less amount of time in an efficient manner is important. Existing literature enlists a number of ways to address this issue. However, the robots being used in those experiments are different and have far greater sensory capabilities than that of Romi. For example, room cleaning robot, Roomba, uses real-time image processing techniques to map the environment [1]. The older versions of the robot did not have cameras integrated into the system, however, they had mechanical bumpers which were used to detect collision with an obstacle [2].

The robot used in this experiment implements a zigzag motion, going to and from between the boundaries of the arena. When the robot detects the boundary, it rotates 90 degrees, move forward for a fixed pre-defined distance and then turns 90 degrees again to continue its motion. During this operation, the robot keeps a track of its position using a set of kinematic equations. Data obtained from the encoders in the form of encoder counts is converted to distances and used to estimate the position of the robot. Following equations have been used for his robot:

This set of equations serve the following purpose:

1. To track the position of Romi
2. To make sure Romi will not go outside of the arena
3. To generate a map of the arena

Fig. 2 illustrates how the zigzag movement works. The steps of this movement, as indicated by number 1, 2, 3, and 4 in Fig. 2 are:

1. Romi moves straight until it reaches the boundary
2. It rotates 90 degree and moves straight for a particular distance
3. Romi rotates another 90 degrees and moves straight
4. Starts over

D. Mapping

One of the main goals of this experiment was to explore the arena and generate its map in the serial plotter of Arduino IDE after the exploration completed. The arena features were classified into empty space traversed which is characterized as M, line as L, obstacle as O and RFID tag as R.

At first, Romi creates an initial map of unexplored area which characterized as "#". As it moved, it updated the map based on what it found. The priority of mapping consecutively from the highest to lowest was RFID, obstacles, lines, empty spaces, and unexplored area. This means that if we find an obstacle in the place where we previously detected as empty space, the program will overwrite the M with an O, while if the opposite occurs (we detect empty space where previously obstacle is detected in that space) it will not be overwritten.

E. Performance Metrics

Three performance metrics were used to determine the effectiveness and efficiency of the algorithm implemented for area coverage.

- The internal mapping function of the Romi generated a map of the arena which was stored in the EEPROM of the system. This map was used as the first performance metric by summing everything detected and dividing it by the grid area:

$$\sum(O + L + R + M)/gridArea \tag{5}$$

  where O is an obstacle, L line, R RFID tag and M Romi's position. The grid area is 25*25=625 for demo 1 and 18*18 =324 for demo 2.
- Based on the area covered in metric one, we were able to calculate the accuracy of Romi's performance. By dividing an image of the map into the same sized grid. If each block in Romi's internal map (Fig. 4) matched the same block in the image of the map (Fig. 3) we incremented. We then divided the total number of matched blocks by the area of the grid.

$$\sum(if(block_R == block_M))/gridArea \tag{6}$$

- The whole experiment was being recorded from a handheld camera, serving the purpose of validating the map obtained from Romi's internal memory. Kinematic equations were implemented to keep a track of the motion of the robot. These equations were used to calculate the total distance travelled by the robot.

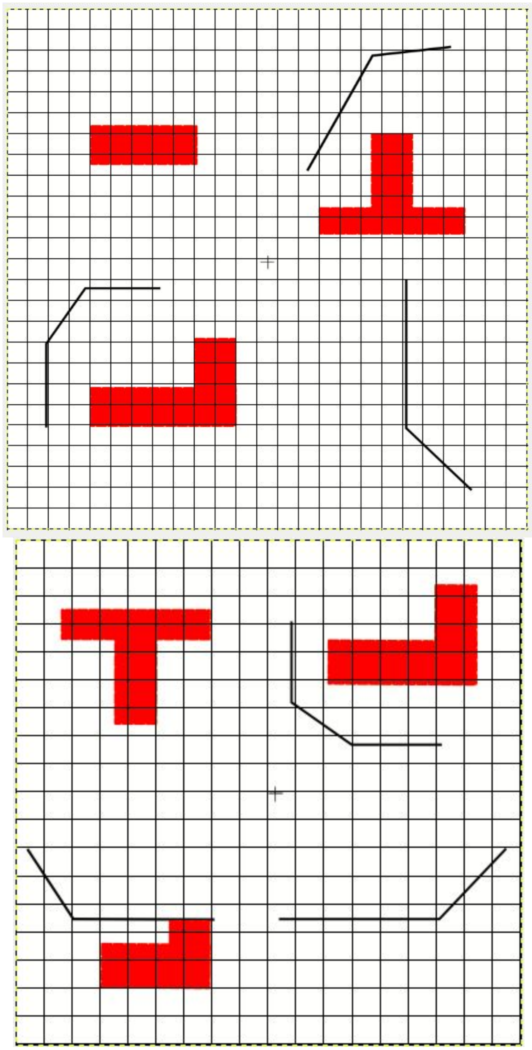

Fig. 3: Demo 1(25*25) and demo 2(18*18)

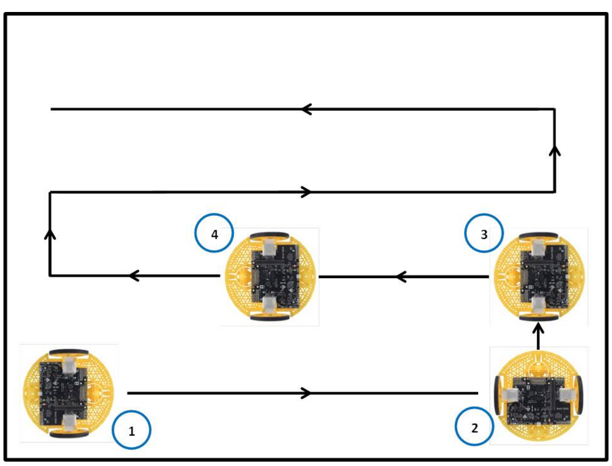

Fig. 2: Zigzag Movement

F. Evaluation

In addition to the performance metrics defined above, which emphasize on the quality of the map generated, three additional parameters were taken onto account in order to get the most efficient solution.

*1) **Robustness evaluation**:* It is possible for the robot to get stuck in a local minima and keep roaming around in the same area. In addition to the speed of the robot, it is important to take into account the robustness of the system to ensure that the ultimate goal of area coverage is achieved.

The zigzag motion of the robot makes it relatively more vulnerable to getting stuck in local minimas.To address this issue, we have introduced a random aspect to the turning action of the robot. Whenever the robot detects an obstacle or detects the boundary of the arena, it turns at an angle 90 degrees with a 50 percent probability of the direction being right or left. As a result, even if the robot is stuck, it gets out after a couple of trials.

The easiest way to quantify the robustness of the system is to monitor the robot throughout the operation ans see how many times does the robot escape the local minimas. This strategy has been used in our system and is discussed in the following sections.

*2) **Computational evaluation**:* The right choice of algorithm is also important for efficiency of the system. Keeping in mind the limited computational power of the controller used for Romi, higher complexity algorithms cannot be implemented. Thus, a fair trade-off between the computational complexity and performance had to made to get optimal results.

*3) **Experimental evaluation**:* Another important parameter is the total time that the robot takes to explore the area. Since most of the exploration tasks have time constraints, the algorithm for the robot had to be designed such that it took minimum amount of time to explore the area, without compromising the quality of the map. However, certain trade-offs between performance and time have to be made.

IV. Results and Discussion

*A. **Demo 1: Results and Analysis***

A very simple and computationally inexpensive strategy was adopted for Demo 1. Table I displays the performance of each test of the first demo. It can be seen that the internal coverage of the robot based on total area covered (M1) and the accuracy (M2) is very low.

The first three tests were carried out on the experiment day, while the rest were after. In the first two tests, failure of the line sensor's reading affected the map generation; thus only 1.6% and 1.76% of the map were correct.

The random movement made the exploration time to be varied among the tests. In test 16 Romi moved inside a small area for a long time before it stuck in an obstacle. In test 13 it went outside the boundaries when only 7.52% was explored. The best performance was acquired in test 15, where 124 blocks were covered, and 15.52% of the area was able to be mapped correctly.

Analysis of the major causes of the issues that occurred in the first demo is presented below:

*1) **Random Motion**:* Starting position for the robot was set as the center of the arena, which had the coordinates of (900,900). Instead of implementing a specific pattern for the motion, random motion was implemented. The robot traversed in a straight line until it either reached the end of the arena or detected an obstacle in its way. As a result, the robot turned at a random angle, avoiding the obstacle. On detecting the boundary of the arena, the robot was programmed to move in reverse and change its path by a random angle.

The results obtained showed that, with no goal position, the robot failed to explore the whole area. The robot also went out of the arena in a couple of the experiments.

*2) **Collision with Obstacles**:* For the first demo, sensors were mounted in a different configuration. One of three sensors was pointing in the forward direction with an angle of zero degrees with the x axis. The remaining two sensors were mounted at angle of 90 degrees and -90 degrees. Although this configuration had several advantages, it posed some serious issues during the process of exploration. Since there was a large gap between the sensors and the sensors have a narrow search space, the robot failed to detect obstacles which were between the sensor-gap. As a result, the robot got stuck there and required external assistance to continue its operation. Moreover, some obstacles were placed in a way that it formed a 'V' shape. The robot got trapped and failed to come out without human intervention. Fig.1 shows the placement of sensors for demo 1.

*3) **The map and Kinematics**:* As seen in Table VIII, the internal map generated by Romi shows very little area coverage.The table shows the result of three experiment runs, all done on the same day. Fig.4 shows the maps generated. The results obtained were not very promising. However, after comparing the results with the videos recorded for the same experiment runs, it was seen that the robot had actually covered more area than that mapped by the robot. It was later found that the random motion of the robot was causing problems with the kinematics of the system and robot was unable to update its position accurately after regular intervals of time. Also, it can be seen that the robot was unable to detect the RFID tags.

*B. **Demo 2: Results and analysis***

Over the period of one month, some drastic changes were made in the system. The results of demo 2 can be seen in Table 2. As it can be seen that the efficiency of the system has improved significantly, however, the results are not yet perfect.

Similar to the first demo, three tests were carried out in the second demo day with our optimised solution. The average of the robot's internal mapping coverage, M1, in the second demo was 51.16% compared to 12.26% in the first demo.

On average 37.32% of the arena was accurately measured (M2), meaning that 72.94% of the 51.16% covered area was correct. It was an improvement of the first demo in which 67.31% of the covered area was accurately measured.

The Romi explored the most area of the map in test 20, with 60.8% area coverage(M1) and 49.69% of correct map generation(M2). The least coverage occurred in test 3 with only 19.44%, in this test the Romi predicted only 14.19% of the whole arena correctly. The Romi went outside the Arena due to failure in its kinematics.

V. Improvements over Baseline Experiment

Improvement over demo 1 is explained in detail in the following sections:

*A. **Grid Size***

It was later realized that the size of the grid did not match the dimensions of the robot. Initially, the arena was divided into a grid of 25x25 blocks, implying that the length of one block is 7.2cm. However, the diameter of Romi is more than 14cm. Therefore, changes had to be made in the system to remove this inconsistency. There were two possibilities. One possible solution was to keep the size of the grid same, and increase the footprint of the robot. This would have meant that the robot would have mapped more than one block at a time, number of blocks corresponding to the size of the robot.

This idea was implemented initially but it had some issues. The accuracy of the map generated was being compromised. Since the robot had only one line-sensor, it could detect a line only when it was exactly below the center point of the robot. A line which was way from the center of the robot but still under the footprint of the robot was mapped as an empty slot. On the positive side, the efficiency of the system increased as the robot was covering double the amount of space it was covering before.

To resolve this issue, the size of the grid was changed. The size of the grid was changed from 25x25 to 18x18. This makes changes the length of every block to 10cm. Although the size of the block is not consistent with the dimensions of he robot, it has some added advantages. Firstly, the size of the area being mapped was very close to the actual size of the robot, making the overall result realistic. Secondly, it created a relatively more accurate map as compared to the map created using the strategy used before, although the results were not perfect.

*B. **Placement of sensors***

To resolve the issue of collisions with the obstacles (blind side) in the first demo, the gap between the sensors was decreased from 90 degrees to 55 degrees. This value was chosen after extensive testing with different range of angles. We settles for this value as the robot was able to detect the obstacles reasonably well and also avoid collisions.

Optimizing the position of the sensors had a great effect on the performance of the system. During demo 1, the run time of the experiment was very short, reason being that the robot used to get stuck after colliding with the obstacle and would not get out without external intervention. Intervening externally would have caused inaccuracies in the kinematics of the system. Therefore, the experiment had to be stopped every time the robot got stuck. Using hit and trial, placing the obstacles in different configurations, the position of the sensors was optimized.

C. *Optimizing Coverage Path Planning*

Choosing the correct motion planning strategy was the most crucial part of this experiment. After reviewing the behavior of the robot when random motion was implemented, it was certain that a more systematic approach is needed. Although random motion had a certain degree of robustness but it also had some uncertainty regarding the completion of its goal. It was impossible to predict whether the robot will be able to perform complete area coverage or not. To remove this uncertainty, we had to choose a coverage path planning algorithm.

During this process, two algorithms were taken into consideration. One was the conventional spiral motion, which was implemented in older versions of conventional room cleaning robots and a zig-zag motion. After conducting several tests in which spiral path algorithm was implemented on the robot, the results obtained were not satisfactory. Firstly, the arena is divided in a rectangular grid. The accuracy of the generated map was being compromised by using a spiral path. Secondly, it is relatively difficult to implement curved path kinematics as compared to straight path kinematics. In addition to the difficulty, there were inaccuracies in the kinematics due to the issues in the mechanical structure of the robot (play in the shaft of the motor etc.). The results of this experimented were not recorded as this did not prove to have promising effects on the improvement of the system and was avoided to spend time on other, more useful path planning algorithms.

Being the major subsystem under consideration, a lot of improvements have been made to it over the passage of time. Following are the major improvements made to the algorithm:

1. Initially, the zigzag motion was implemented in a rather hard-coded manner. The robot was programmed to move in a straight line and at the end of the boundary, the robot did a 90 degree turn, traversed for a small distance and then turned 90 degrees again. The first issue encountered here was that the robot tend to divert from its straight line motion after covering have the length of the arena. This required the optimization / tuning of the PID and the position controller. Since, there was no goal position in this task, we had to implement speed controller instead of position controller to get more accurate results. Therefore, the first optimization task was fine tuning of the speed controller to ensure that the robot moves in a straight line even after travelling for longer distances.
2. The second challenge was to integrate an obstacle avoidance algorithm into the existing system. When an obstacle was detected by the front proximity sensor of the robot, same strategy was adopted as that adopted for the boundary. The robot turned 90 degrees twice and continue the operation. When an obstacle was detected by a side facing sensor, the robot made a 45 degrees turn in the opposite direction and after traversing for a specific predefined distance, performed a 45 degree turn again to continue its zigzag motion.This action was possible only because the sensors were mounted at an angle of 55 degrees. It ensured that the strategy of making a 45 degree turn was safe if an obstacle is detected at an angle of 55 degrees. However, there were some limitations of this strategy, which is discussed later in this section.

D. *Mapping*

Our robot uses a slightly modified version of the mapping function. The mapping function has been improved and tailored according to our needs. After modifications in the mapping function, whenever a moves over an empty place, it places a letter ”M” at its place, marking it as mapped. Similarly, whenever the line sensor detects a line, ”L” is replaced at that point and it places an ”O” when an obstacle is detected.

Moreover, effective area coverage implies that the robot does not go visit the same block twice. Labelling visited blocks as M had an added advantage related to this. Every time the robot visited a new block, it checked whether the block has been marked as M, L or O. If it had already been marked, the robot would not rewrite over the block again.

We also tested that Romi was placing obstacles in the correct place by using a measuring tape and adjusting the boundaries of the IR sensor readings in the mapping function.

Mapping results are shown in Fig.4 and Fig.5. For the first and second maps in both figures the threshold for line sensor was too low so M was replaced with L. In those images L represents the Romis movement M.

In Table 1 M1 metric 1 and M2 metric 2 are included. M1 is Romi’s internal mapping (Eq. 5) and M2 is actual accuracy (Eq. 6) based on Fig.3 maps. Please note that the maps generated by Romi in Fig.4 and Fig. 5 are inverted in the vertical axis.

VI. Overall Evaluation

This section describes the improvements done over the baseline experiments in a more compact and quantitative form. Following three parameters were mentioned during the start of the report and are discussed now.

A. *Robustness Evaluation*

It is evident from table VIII and VIII, that the robot spent more time in the arena in demo 2 as compared to demo 1. As the numbers show, in comparative terms, the robot spent approximately 50 percent more time in the arena as compared to demo 1. It implies that the robot was robust enough to escape local minimas and continue its operation. Thus, there is a drastic improvement in the robustness of the system compared to the baseline experiment.

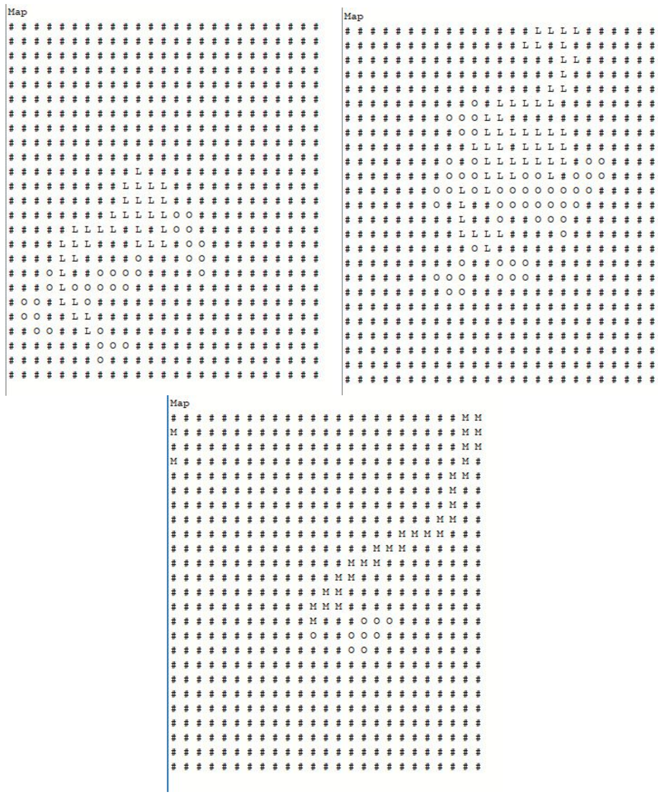

Fig. 4: Maps generated from demo-1 (25*25 grid)

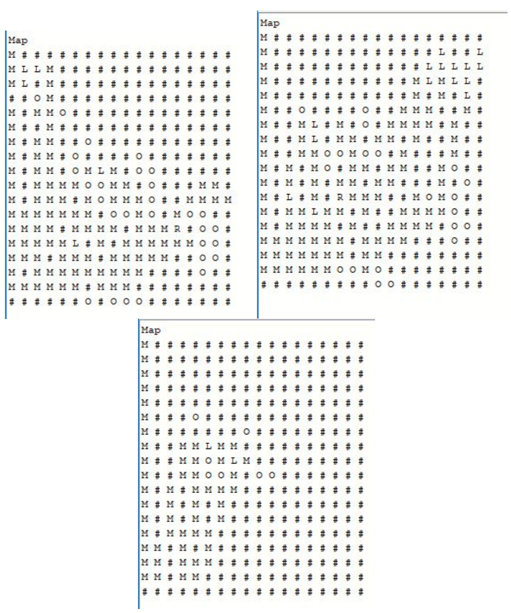

Fig. 5: Maps generated from demo-2 (18*18 grid)

B. Computational Evaluation

Due to the limited computational capabilities of the system, the algorithm had to be simple enough. Although random path algorithm was relatively simpler but it was not efficient. Therefore, transition to a slightly complex algorithm was made. Integrating the kinematics, mapping function and obstacle avoidance does have an effect on the efficiency of the algorithm, as the robot sometimes became jittery because the update cycles were out of sync.

It would be possible to generate an improved solution for this problem by implementing neural networks [3] but this would have been computationally very expensive. The algorithm which is currently implemented generates satisfactory results is computationally inexpensive at the same time.

C. Experimental Evaluation

Overall comparative experimental evaluation shows that Romi is able to explore more area in the same amount of time now as compared to the baseline experiment. Almost all of the parameters which are essential for efficient zigzag path implementation have been optimized and results show the effect of the improvements made.

VII. Limitations and Future Work

A. Hardware Differences

1. Each of the three IR sensors used in this experiment had different output values. This caused inaccuracies in our mapping. For example, in the first demo, there were cases in which obstacle was detected at about one or two blocks before the actual position. This was because we set the same threshold value for each sensor in obstacle detection. One way to eliminate this is by calibrating each sensor so that a single threshold value may apply for all or setting a different threshold for each sensor.
2. The output shaft of the gearbox attached with the motor had a certain amount of play in it. This had a significant effect on the accuracy of the kinematics of the robot. The effect of this minor play in the shaft might be insignificant for shorter distances but have a greater impact on the performance for greater distances.

B. System Failures

Even after the implementation of a fairly efficient algorithm, the overall accuracy of the maps generated is quite low. Moreover, during this time, the robot failed to cover the whole map even once (in the presence of obstacles). One of the main causes of this is the limited sensory capabilities of Romi and the placement of the obstacles in the arena. The system failed and went out of bounds when the obstacle was too close to the boundary. Whenever Romi encountered such situation, while traversing the predefined distance, it went out of bounds and failed to come back inside the arena.

The output of the line sensor changes with the change in the lighting conditions and the reflectively of the surface of the arena. Due to a preset threshold for the line sensor, it often happened that the robot wrongly mapped the lines on the arena. The solution to this problem can be the integration of more than one line sensors in the system.

C. *Future work*

Future work on the same lines would require the upgradation of the hardware of the robot. With the extension in the computational capabilities of the robot, implementation of neural networks would become possible which would have the capability of generating the best results for the system. Moreover, adding additional sensors to the system would increase the capability of the system to detect obstacles in a better way and increase the accuracy of the system.

VIII. Conclusion

In this experiment, we have designed a robotic system which can be used in an exploration scenario for the purpose of area coverage. Through an iterative development process, we have successfully developed a system which uses the limited capabilities of the robot to produce fairly satisfactory results. The robot successfully performs its operations with an accuracy of about 37.32% percent on average, which is an improvement of more than twice over that of the baseline experiment.

Through the extensive series of experiments, we have optimized the path planning subsystem, part of the greater problem of area coverage. Additionally, we have also optimized some of the aspects of odometry, e.g. position of the sensors on the robot. we can confidently say that the robot is capable of working in an environment which is bounded and has obstacles which are sparsely located. The robot would be able to detect the obstacle effectively in the range of 6cm to 25cm and would be able to map the place with a success rate of approximately 50 percent. Although, this is a fairly low success rate but it is a great improvement as compared to the starting point of the experiment. Provided with more robust hardware and computational equipment, the results can be perfected.

References

[1] P. Nattharith, "Fuzzy logic based control of mobile robot navigation: A case study on irobot roomba platform," *Scientific Research and Essays*, vol. 8, no. 2, pp. 82–94, 2013.

[2] B. L. Bailey, "User-assisted robot navigation system," Jan. 29 2013, uS Patent 8,364,309.

[3] S. X. Yang and C. Luo, "A neural network approach to complete coverage path planning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 1, pp. 718–724, 2004.

Table I. - Demo 1 Results

Run Number	Distance Travelled(m)	Boxes Visited	M1(%)	M2(%)	Time(s)
1	4.5	61	9.76	1.6	42
2	7.8	108	17.28	1.76	111
3	3.2	42	6.72	5.44	41
4	6.3	88	13.28	9.76	85
5	5.2	72	11.52	8.16	62
6	4.4	59	9.44	7.68	46
7	4.3	60	9.6	6.08	47
8	6.6	91	14.56	9.76	97
9	4.3	59	9.28	6.4	51
10	8.6	116	18.56	13.28	108
11	4.0	53	8.48	6.88	51
12	6.6	89	14.24	9.92	83
13	3.9	47	7.52	5.76	42
14	5.6	73	11.68	8.16	73
15	9.1	124	19.84	15.52	124
16	5.2	71	11.36	8.8	77
17	4.4	59	9.44	6.24	53
18	6.1	84	13.44	8.96	87
19	6.4	88	14.08	10.4	97
20	7.1	96	15.36	10.72	99
21	5.6	78	12.48	8.96	83
22	7.3	102	16.32	12.96	104
23	4.6	65	10.4	7.84	64
24	4.7	66	10.56	7.52	56
25	5.1	70	11.2	7.68	59
Average	5.6	73	12.26	8.25	73.68

Table II. - Demo 2 Results

Run Number	Distance Travelled (m)	Boxes Visited	M1(%)	M2(%)	Time(s)
1	13.8	132	40.74	28.39	189
2	15.4	142	43.82	25.93	181
3	0.62	63	19.44	14.19	59
4	17.6	174	53.7	38.27	187
5	18.2	182	56.17	42.28	176
6	18.0	181	55.86	43.51	185
7	16.3	162	50	38.58	151
8	19.4	193	59.57	44.13	194
9	18.8	188	58.02	42.9	183
10	17.2	172	53.09	37.35	177
11	16.5	166	51.23	33.02	171
12	17.6	174	53.7	33.95	180
13	13.3	133	41.04	31.48	152
14	17.1	170	52.46	39.19	184
15	18.5	186	57.41	42.59	192
16	18.6	185	57.09	36.72	175
17	13.3	132	40.74	29.93	149
18	18.9	189	58.33	43.82	197
19	17.6	175	54.01	41.67	173
20	19.7	197	60.8	49.69	213
21	18.3	184	56.79	45.06	199
22	16.8	168	51.85	31.79	170
23	19.1	192	59.26	45.68	219
24	14.3	143	44.14	35.49	157
25	16.2	161	49.69	37.04	143
Average	16.3	165.76	51.16	37.32	174.24