CSE 276A - Homework 5

Chhabra, Rachit (rachhabra@ucsd.edu)
Department of Mechanical and Aerospace Engineering, University of California San Diego

I. PROBLEM STATEMENT

The objective of Homework 5 is to implement area coverage of the environment using Coverage Algorithms. The goal of the algorithm is to cover all the free area in the environement and provide performance guarantees for the same. Also, these algorithms would be run on pre defined environments to test the algorithm, and then on maps generated by the SLAM algorithm written in HW3.

II. SETUP

The setup of the environment is follows:

1. We have 12 April Tags which we considered as known landmarks during the Area Coverage problem at first. Afterwards, it is assumed that we do not know the position and number of any landmarks. The landmarks can be repeated and the ground truth is shown below.

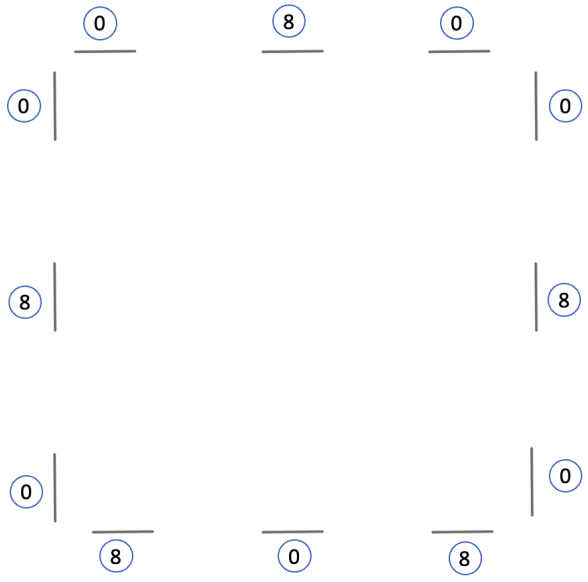

Fig. 1. Setup

2. Here, each landmark does not have a distinct ID (April Tag id) and thus we need to create an algorithm to solve this ambiguity as well while localising the robot.
3. The initial starting position of the robot is $(1.225, 0.5, 0)$ meters.
4. The overall width of the area is 8 feet and there is no obstacle in between but the algorithm we create should be generic such that it takes into account the obstacles as well.

III. TECHNICAL APPROACH

Following is the diagram which explains the control flow of the system. The problem statement

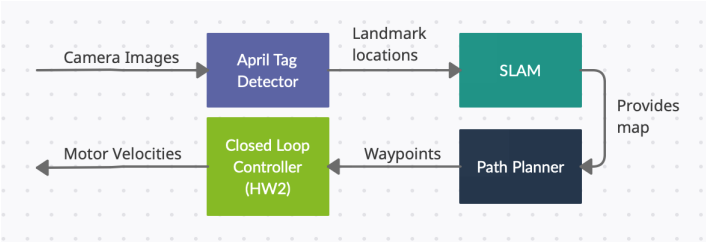

Fig. 2. Control Flowchart

is a mixture of HW3 and HW4 where, both the SLAM node and the Motion Planning node are involved. First, the April tag detector send the landmark images to SLAM function, where a map is generated using the Kalman Filter Algorithm written in HW3. Next, the map is passed on to the planning function, where the coverage algorithm is written. It outputs the required waypoints to cover the whole empty area. Finally, the controller iterates over the waypoints, and using the PID controller, sends it to motors for the robot to move.

Coverage Algorithm

Firstly, a very simple algorithm was written to solve the problem. Wherever we start in the environment, first step for the robot is to go to the start position $(0, 0, 0)$. Then, the robot travels in the x axis, until a wall is reached (1 in the map). Then it rotates towards the y axis, move the length of the robot, rotates towards the -x axis, moves until the wall is reached again and then moves the length of the robot towards y axis. This motion is repeated

until the map limit is reached in the y axis(1 is encountered in the map). This algorithm is very simple and works on the given environement. The result is shown in the next section. But it does not provide and performance guarantees as if we have an irregular shaped polygon as the environement or have an obstacle in the center, it would not cover close to 100% of the area.

The second algorithm was made to consider the random shape of the environement and addition of an obstacle.

1. First step is to detect the corners in the given environement. This is done by iterating over all the cells and looking at the neighbouring cells. Depending upon the number of cells with 1s and 0s in it, we can detect whether it's a corner. Also, from the detected corners, if the value of the corner in the map is 1, it's the corner of the obstacle, and if 0, it's the boundary corner.
2. If no obstacle is detected, A star is run firstly from the start position to corner 1, and then from corner 1 to corner 2, and so on. Finally when we reach to corner 1, steps 1 and 2 are repeated until no corners are found.
3. If an obstacle is detected, the obstacle corners and boundary corners are joined together to form smaller polygons, where step 1 and step 2 are repeated for each of the area, and once an area is fully covered, it moves onto the next area until all the areas are iterated over.

Behaviour to provide coverage/avoidance

As discussed above, to provide coverage, the behaviour needed and expected by the robot is to move along the edges from one corner point to the next, and repeat the same iteratively over smaller and smaller areas, until all of the area is covered in the quadilateral. Also, no or only one obstacle is assumed in the algorithm, and thus the limitation of the algorithm lies at the point when we have multiple obstacles in the environment.

IV. RESULTS AND ANALYSIS

The result for the first algorithm are shown below. As we can see the robot covers the full area perfectly. But as discussed above, it does not provide any performance guarantees so we now see the results of the second algorithm below.

Fig. 4 showing the initial map for the problem statement for HW5. Here, there are no obstacles

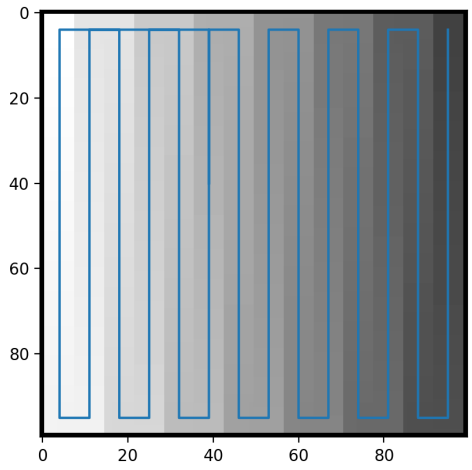

Fig. 3. Map 1

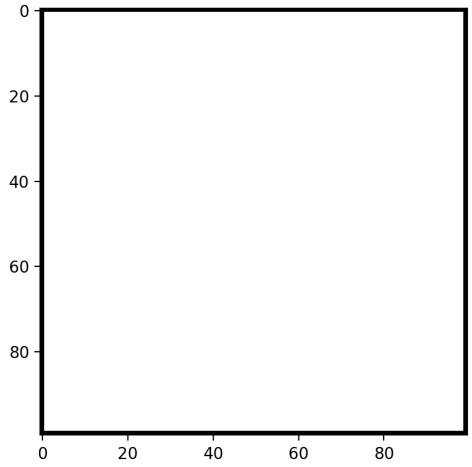

Fig. 4. Map 1

and this map is generated and used to show the performance of the coverage algorithm.

In Fig. 5, we can see that using our second algorithm, we again get 100% coverage for the environment given in HW5. And finally in Fig. 6, we can see the path taken by the robot to cover the full area. We can see it moves along the walls to reach the corners as intended.

But to guarantee the performance, we test our algorithm on other environements as well.

Now we look at Fig. 7, where we introduce a map with random boundaries. The algorithm still finds the corners for the given map and the robot

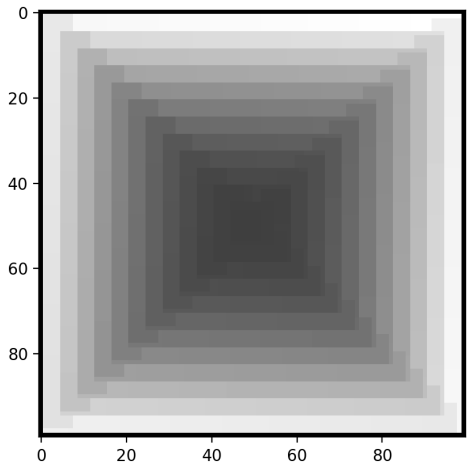

Fig. 5. Robot coverage for Map 1

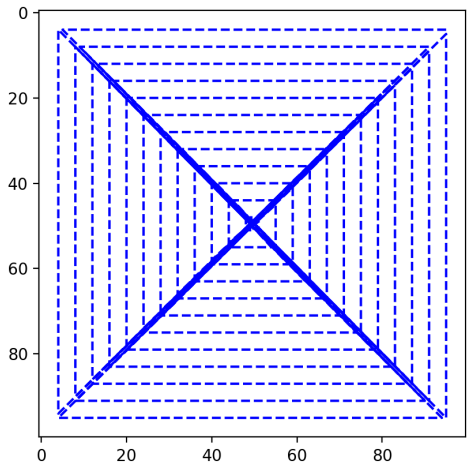

Fig. 6. Path followed for Map 1

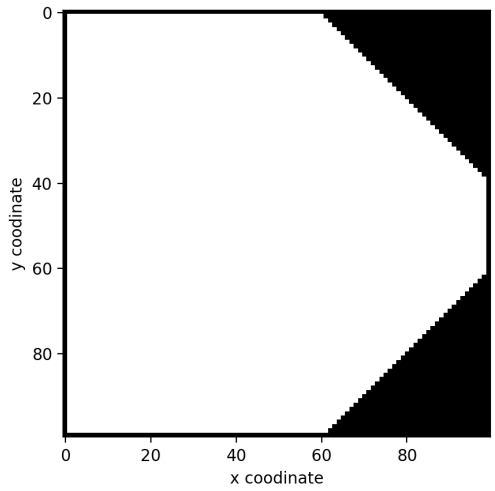

Fig. 7. Map 2

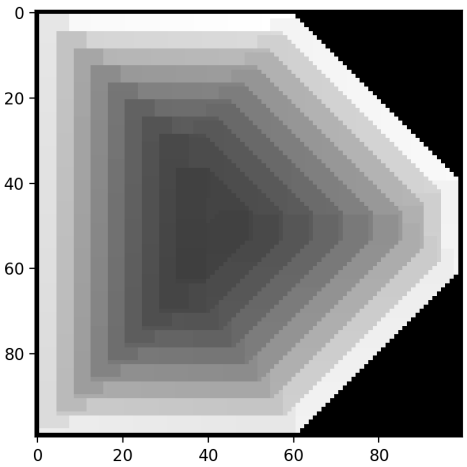

Fig. 8. Robot coverage for Map 2

follows the edges of the map as expected.

We see that in Fig. 8. The algorithm ends at the center when whole of the area is covered and no corners are left.

Now, to test our algorithm even more, we introduce an obstacle in the middle of the environement. This configuration is chosen as it was the configuration used in HW4. The map looks like the on shown in Fig. 9.

We can see that due to the obstacle being present, we no longer can use our original algorithm, so we first divide the area into quadilaterals and then run the algorithm individually and following is the results.

In Fig. 10, we can see the area covered by the algorithm. The coverage percentage is more than 99.5%. This can be increased by tweaking the minkowski sum function.

Finally in Fig. 11, we can see that each divided area is covered one by one, and once one area is covered, the robot moves to the next area depicted by the dotted lined moving from one area to another.

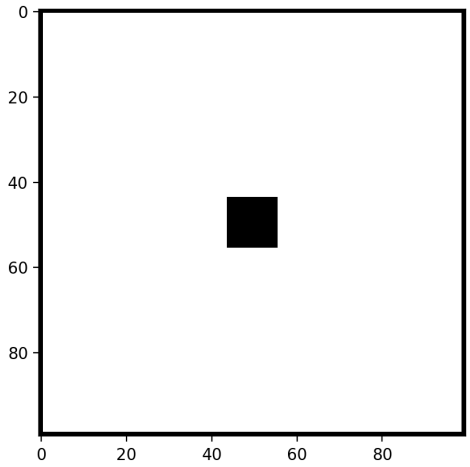

Fig. 9. Map 3

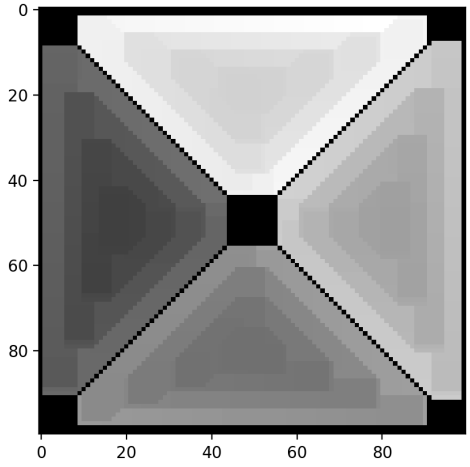

Fig. 10. Robot coverage for Map 3

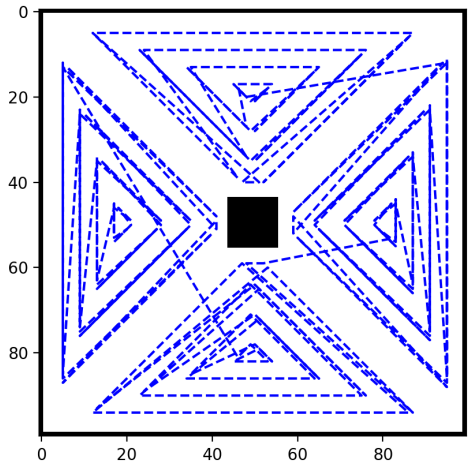

Fig. 11. Path followed for Map 3