Airlab Assignment

Keerthana Subramanian Manivannan

November 21, 2016

Part3: Planning

3.2 Distance Transform

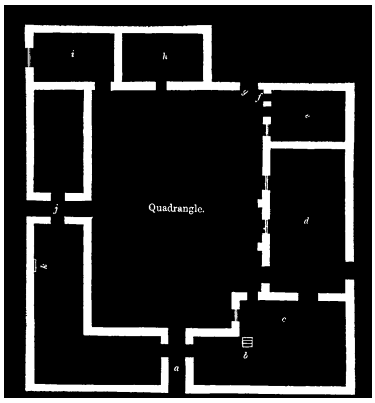

Figure 1: Given Map

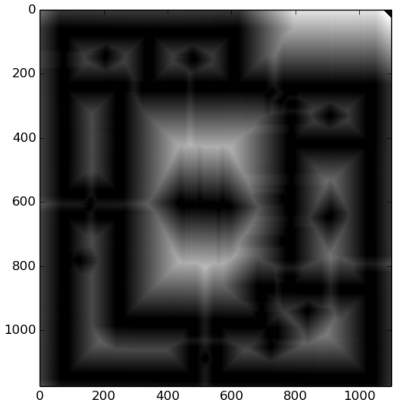

Figure 2: The Distance Transform obtained by Dijkstra's algorithm

Fig. 1 is the occupancy grid map given. I implemented Dijkstra's algorithm to compute the distance transform of the given map, and got the results shown in Fig. 2. In this Dijkstra's algorithm, there is no specified goal position. The algorithm expands till there is no state left in the open list.

3.3 Motion Planning

A* algorithm was implemented with Manhattan Distance as the heuristic. The heuristic is admissible because even if the goal position is near an obstacle, the heuristic will never overestimate the distance to go to the goal position.

The heuristic is also consistent, because its estimate is always less than or equal to the estimated distance from any neighboring state to the goal, plus the cost of reaching that neighbor.

3.3.1 Optimizing for Distance to Obstacles

Since this is assumed to be a four-connected grid, the robot takes the path shown in black to reach the goal position of (910,1035). As shown in Fig. 3, the black path taken is to minimize the cost of traversing close to an obstacle. The path length of robot in this case is: 154237

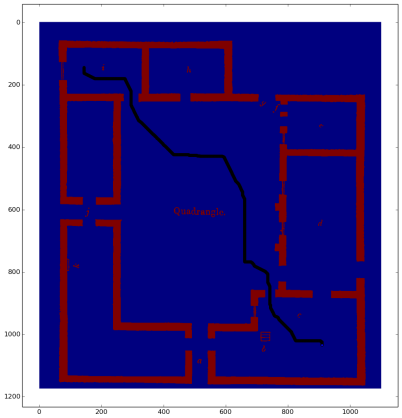

Figure 3: The path obtained by A* algorithm to optimize on the cost obtained from the obstacles from the distance transform

3.3.2 Optimizing for Distance to Goal

As shown in Fig. 4, the black path taken is to minimize the cost of traversing close to the goal. Essentially, this is the shortest path length to the goal. The path length of robot in this case is: 41375

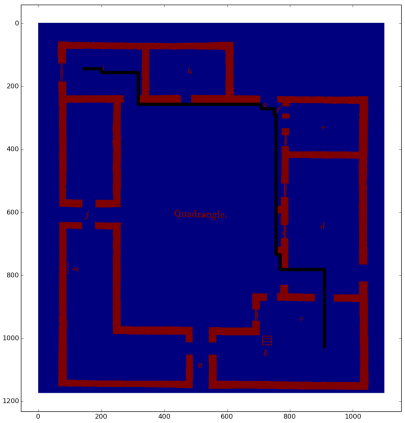

Figure 4: The Optimized path obtained by A* algorithm to minimize the path length taken by the robot

3.4 Exploration Planning

One of the naive ways of implementing exploration in an autonomous robot is the Frontier Based Planning. Frontiers are the interest points in the map which have the highest information gain. They are essentially the points in the boundary between open space and uncharted territory. The whole map is set to some prior probability, and when explored, the robot adds the new information and updates it map.

Problem: The robot oscillates from one frontier to another while selecting a frontier node to pursue.

Analysis: The frontier nodes are chosen from a list of nodes which give the maximum information gain. The oscillations occur because two frontier nodes have same information gain value, and once the robot moves to the second node, the first node now gives the maximum information gain value. Thus, the robot oscillates from one node to another, not completing the exploration.

Solution: One way of tackling this is to subtract distance from the information gain value and thus choosing the closest node. Looking at the code, I see that the distance has already been incorporated in it. But the weights to information gain and the distance were 10 and 1 respectively. Printing the information gain value and distance value to the output console gave the values of information gain in the range of 200s and distance in the range of 0.5-1.5. Thus the distance has very little effect on choosing the next frontier node to explore. I switched the weights to 1 and 100 respectively, and the robot doesn't oscillate while choosing the next frontier node and it completes exploring the whole map shown in Figures 5 and 6. In these figures, the green lines show the path taken by the robot.

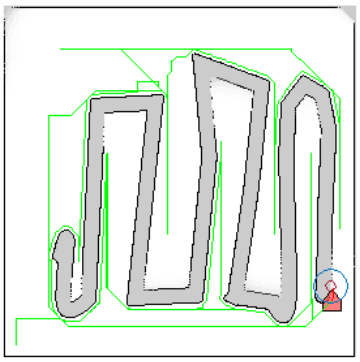

Figure 5: Fully explored map

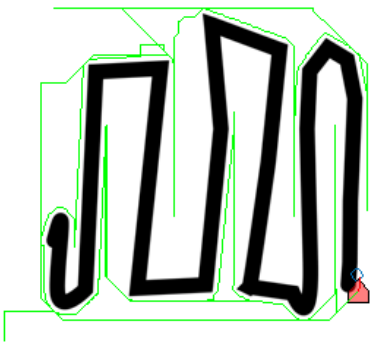

Figure 6: The traversed ground truth map