**Motion Planning: Discrete Planner**

The aim of the assignment is to implement two discrete planners: 1) Random Planner 2) Optimal Planner with objective of searching a path to the goal from specified start position in the given world. As per the given info we can say that the world is represented as 2D Grid with each cell representing a location. Robot is considered as a point robot. Using the info given two planners are implemented in C++ and discussed in detail below.

**Random Planner:**

Random Planner try to reach the goal by making a move randomly. It has fixed amount of time to find the goal which is max\_step\_number if the path is not found in this time it returns FAIL and prints the path it tried. If it reached the goal it prints the path it used to reach the goal.

**Optimal Planner:**

Optimal Planner unlike random planner does not try to find path by randomly moving towards goal instead it uses some intelligence to find the path to goal. This intelligence is used in terms of heuristic function. Heuristic function basically estimates the cost (distance in this assignment) from current position to the goal position. The algorithm used to implement this planner is A\* algorithm. Heuristic function used is Manhattan distance. Using this heuristic function and cost function optimal planner finds path to goal.

**Comparison:**

While comparing both the planners there are some pros and cons of each of them. Let’s start by talking about the time complexities of each planner. Time complexity of the Random Planner is O(n) (n is max number of steps) whereas time complexity of Optimal Planner is O(n logn). From the complexities it is seen that random planner works faster, but at the same time it does not provide the best path. Moving randomly it may find the path faster but also it may iterate through some of the intermediate path again and again. Although optimal planner takes time in finding solution but it gives the best solution. There is always a tradeoff present when going for optimal planner.

Also from the tests performed on both the planners it was seen that optimal planner always finds a path if there exist one while random planner was not able to find path every time. The reason behind random planner not finding path is number of steps constraint, if the number of steps is less it does not reach the goal, one of the reason for that is it gets stuck in the intermediate steps going over them again and again.

In terms of testing I have tested both the planners with different environments and both the planners have worked properly. All the constraints like orthogonal movement, max step number for random planner were implemented perfectly. I have also included 3 different input files for test purposes in the code. The code implemented takes input like world pose, robot pose and goal pose from a file.

The input from file is as follows:

First line is 2 values stating number of Rows (n) and Columns (m) (World size)

Then next n lines contain m values on each line specifying 0 (free space) or 1 (obstacle)

After n lines it has 2 values stating robot pose

After that it has 2 values stating goal pose

For Random Planner there is on more line specifying man\_step\_number

Before running the code the file path needs to be changed based on the path where you place the code on your system.