ROB521 – Assignment 1 Report

In this assignment, we implement the Probabilistic roadmap (PRM) algorithm to construct a graph connecting start to finish nodes for a maze. The initial maze is in a shape of a 7 x 5 units' rectangle. Subsequently, we find the shortest path from the graph using the A* algorithm. Finally, we modify the sampling method from random to uniform to reduce the runtime. Using the optimized method, we could find the shortest path for mazes larger than 40 x 40 units in under 20 seconds.

Question 1: Implement the PRM algorithm to construct a graph connecting start to finish nodes

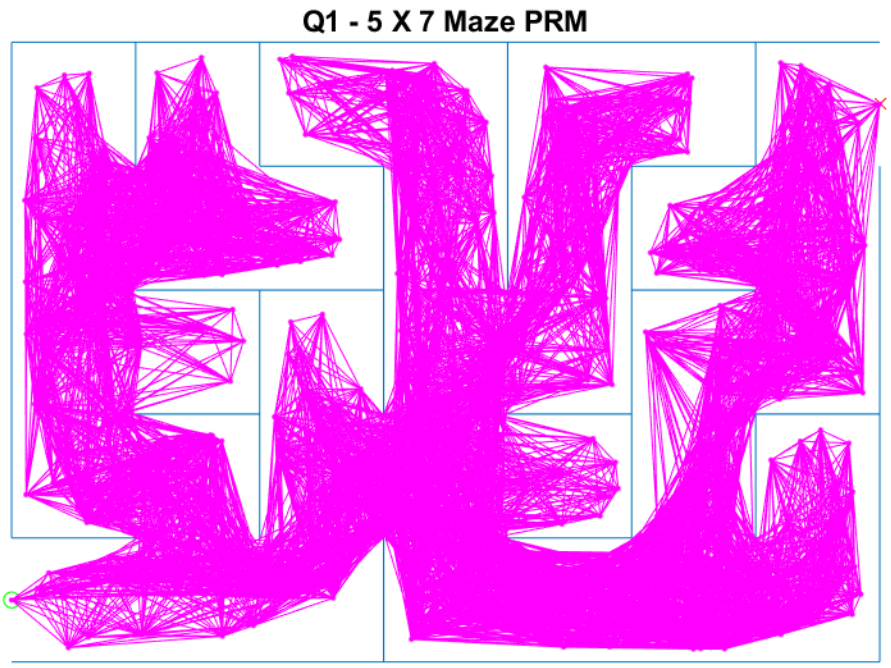

Figure 1: PRM Alogorithm Constructed Graph

Observation: Since we are using 500 samples in a relatively small maze of 7 x 5 units, the graph we construct using the PRM algorithm has a fairly similar shape to the actual maze walls. However, the payoff for having a large number of nodes in close proximity of each other will increase the runtime to construct edges since we are using the nearest neighbour connection strategy that results in lots of edges. Collision checking is

very time-consuming, and each edge would need to be checked against the maze walls and removed if it collides with them. As a result, the runtime is long, and the same method would be hard to scale to bigger mazes.

Question 2: Find the shortest path over the graph by implementing the Dijkstra's or A* algorithm

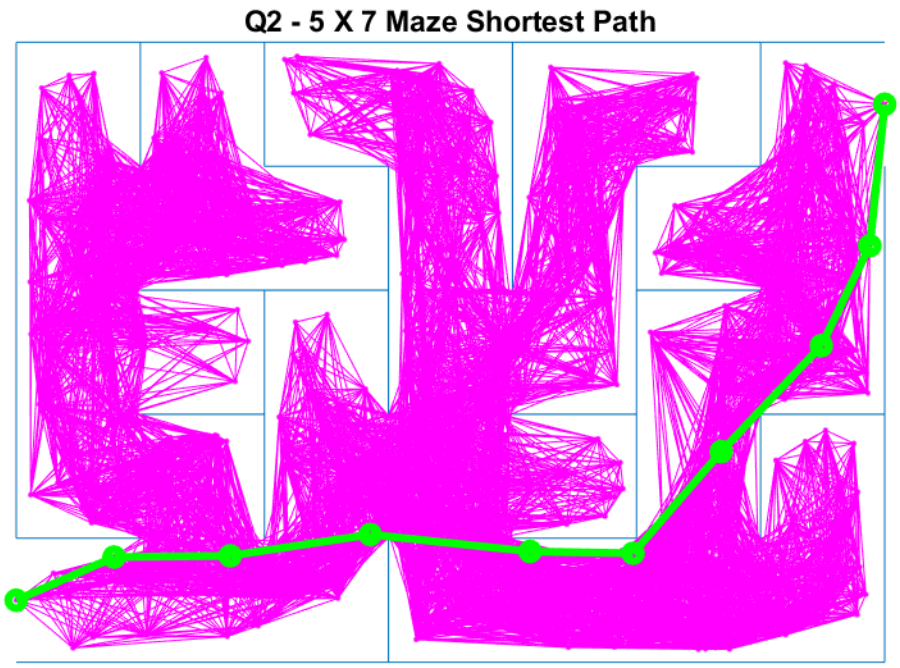

Figure 2: Shortest Path for 7 x 5 Maze Using A-Star Algorithm

Observation: As mentioned in the observation of question 1, the abundance of nodes in close proximity allows us to have a fairly accurate representation of the maze. This supports the algorithm that generates the shortest path to cut corners and generate a path that is close to the maze walls which would reduce the overall length of the shortest path found. The shortest path only takes around 0.2 seconds to find after the PRM graph is constructed, while it takes approximately 8 seconds to construct the graph, confirming that most of the time is used to generate the graph.

Question 3: Identify sampling, connection or collision checking strategies that can reduce runtime for mazes

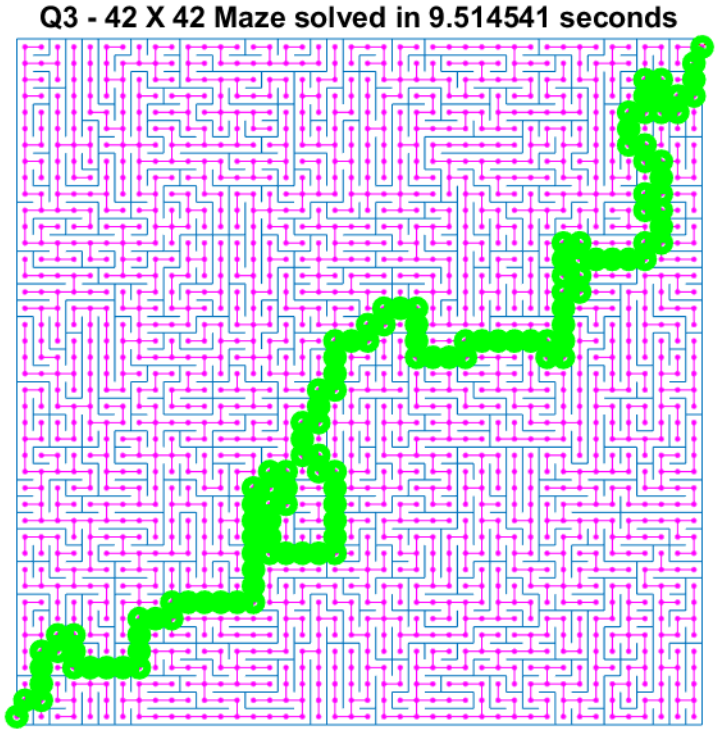

Figure 3: Optimized Algorithm Used for 42 x 42 Maze

Observation: By simply modifying the sampling method, we are able to greatly decrease the runtime, allowing the generation of the shortest path for a maze larger than 40 x 40 units in under 20 seconds. Since the maze and the maze walls are uniformly shaped, we take advantage of this setup and use uniform sampling instead of random sampling. Uniformly spaced-out nodes with our original nearest neighbour connections generation of a neighbour distance of 2 units increases the path length. Nonetheless, it greatly reduces the runtime. The uniform sampling method takes around 9 seconds for a 42 x 42 units maze, so the trade-off is reasonable.

---------- Matlab code is attached below ----------

****Note: The written helper functions start from page 10 of the pdf.**