Implementing Path Planning with Probabilistic Roadmaps (PRM)

Shresth M Jha

***Abstract*— Here in this task, we are helping a robot to navigate through a maze in the first part through a probabilistic path planning algorithm we are given an image with the start points and the end points and the aim is to find an optimal path without touching the walls of the maze.**

I. INTRODUCTION

In short, if we want to tell the steps to be followed in path planning first we will spread the nodes across the image through uniform probability distribution join them with some constraints, and then apply some algorithm such as the Dijkstra or a star(in our case) to find the path.

II. PROBLEM STATEMENT

Explain in detail what your problem statement was with all the necessary images and equations required. the problem here which we need to solve is to first spread the nodes across the image in such a way that the complete image is covered, even the parts of the image that are hard to reach should have some nodes so that if needed we can get there. Once the nodes are generated we need to check if the nodes is white or black and if it is white add it to the nodes in the graph .now with each newly generated white node check with the previous nodes in the graph to if they are in the reachable maximum distance for connection and not too far away, we will also be needed to check if the nodes when connected are not getting a black pixel in between or to be precise an obstacle in between for this we will have to use the Bessenher's algorithm. if two nodes can be connected then we will add an edge in the graph. using the nodes and edges in the graph we will have to use a star for a graph for the final location and initial location joining point and the edges in the path will be plotted as the final path.

III. RELATED WORK

In place of Bessenher's algorithm, some other approach could have been used . also in place of the A star algorithm, some other approach could have been used.

IV. INITIAL ATTEMPTS

The main problem that I was trying to solve was how to plot the nodes which I then solved using the random int generator in Python.and at last the threshold of how many nodes to be generated was a problem which i later did with hit and trial method,similarly the location of start and end had to be done by hit and trial method.

V. FINAL APPROACH

The problem at hand revolves around efficiently traversing and exploring a given maze-like environment, represented as an image, using a graph-based approach. The primary objective is to spread nodes across the image in such a way that the entire area is covered, ensuring that even hard-to-reach regions have some nodes. Subsequently, these nodes are utilized to construct a graph where white pixels in the image represent potential node positions.

The process begins with loading the image and converting it to grayscale to facilitate analysis. Subsequently, a thresholding operation is applied to convert the grayscale image into a binary image, simplifying the identification of white pixels (representing free space) and black pixels (representing obstacles or walls).

To generate the nodes, a specified number of random positions are sampled within the image boundaries. However, it's crucial to ensure that the generated nodes do not overlap with black pixels, as they represent impassable regions. Furthermore, nodes located close to each other within a predefined maximum distance are connected by edges in the graph.

The challenge arises in determining whether a connection (edge) between two nodes is feasible, taking into account potential obstacles between them. This is addressed by employing Bresenham's algorithm, a line-drawing algorithm that efficiently determines whether there is a clear path between two points in an image grid. If no obstacles are encountered, an edge is added between the nodes in the graph.

Once the graph is constructed, the A* algorithm is utilized to find the shortest path between a specified initial location (start node) and a final destination (end node) within the graph. A* evaluates potential paths based on a combination of the actual cost of reaching a node and an estimated heuristic cost to reach the goal, guiding the search towards the goal efficiently.

The final path obtained from the A* algorithm is then visualized by plotting the edges corresponding to the optimal route on the original image. This path serves as a navigational guide, enabling traversal from the initial position to the desired destination while avoiding obstacles.

In summary, the problem involves a multi-step process that encompasses node generation, graph construction, obstacle detection, pathfinding, and visualization. By leveraging graph-based algorithms and image processing techniques, an effective strategy is devised to navigate through complex environments represented by images, facilitating efficient exploration and traversal.

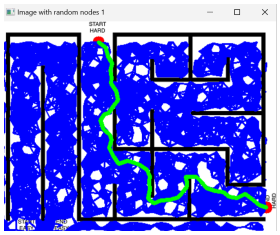

Fig. 1. Figure 1

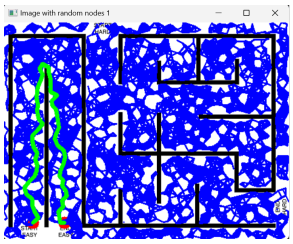

Fig. 2. Figure 1

VI. Results and Observation

the final result of the maze traversing looks as follows: Figure 1 and Figure 2.

VII. Future Work

one of the problems to be tackled was that the nodes were being generated outside the maze so to limit that some extra conditions had to be added.

CONCLUSION

This task was quite helpful to understand how path planning algorithms work and how to navigate through a maze and how to handle images in codes.