International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Performance Analysis of Probabilistic Roadmaps in Simulation for Mobile Robot Path Planning

Karthik Krishna Srivatsava Kondapally[1], Hari Deepak N[2], Anbarasi MP[3]

[1,2]Undergraduate Student, PSG Institue of Advanced Studies, Coimbatore, India
[3]Department of Robotics and Automation, PSG College of Technology, Coimbatore, India

Abstract
Many challenges, including path-planning for a mobile robot, revolve around finding the lowest-cost path through a graph. A quick path-planning algorithm for a mobile robot in a dynamic environment is proposed by integrating Dijkstra's algorithm. Dijkstra's method and Probabilistic Roadmaps are used to determine an initial path from the starting point to the goal. This method searches for an optimal path from the robot's current location to this local target, and as the number of nodes increases, a new path that brings the robot from the current place to the goal is obtained. Meanwhile, the re-planning time is substantially reduced. The simulation results validate the suggested algorithm's feasibility and effectiveness.

Keywords: Path planning, dynamic environment, Dijkstra's algorithm, Probabilistic Roadmaps

I. INTRODUCTION

In robotics, path planning is an important part of self-navigation. It involves a complex process that allows the robot to determine the best part from its current location to a desired location, avoiding obstacles and complying with various constraints. This process is important for many robotics applications, such as industrial automation, driverless cars, and drones, because it allows these machines to make quick decisions about how to move safely and is well decisions about how to move safely and is well designed and implemented in a real space. There are other sampling-based algorithms, including Probabilistic Roadmaps (PRM), A*, Dijkstra algorithm, and RRT-based [1] methods are used to solve this challenge, and their selection depends on the specific application, including requirement efficiency, on-the-fly scheduling, and avoiding outages. By knowing the plan, robots can work more effectively in different situations, change jobs and improve our daily lives thanks to technology electricity and smart solutions.

A network graph of potential routes in a given map based on open and occupied spaces is known as a probabilistic roadmap (PRM) [2]. Based on the PRM algorithm parameters, the mobile robot object generates nodes at random and connects these nodes. Nodes are connected based on the connection distance and the locations of the obstacles shown on the map. The number of nodes can be altered to suit the complexity of the map and your goal to discover the quickest route. To discover a path free of obstacles from a starting point to an ending point, the PRM method makes use of a network of connected nodes. A path through an environment can be properly planned by adjusting the connection distance and node attributes. The node locations are created at random when creating or changing the mobile robot class, which may have an impact on your final path after several iterations.

II. METHODOLOGY

The insights about path planning with Dijkstra's and probabilistic roadmap-based search algorithms will be discussed in this section. Initially, designed an environment for the mobile robot, which is crucial for the robot to travel, and then converted the image to a grayscale image, which helped us to convert it into the binary occupancy grid [3] image, i.e., in binary bits, helps in designing autonomous navigation systems.

The model considered when evaluating the surroundings is grid-based because the algorithms utilized are likewise based on it [4][5]. Then, a highly effective method is suggested for creating the best routes for mobile robots in a challenging dynamic environment. By using local re-planning, the algorithm can always determine the best route from the robot's current location to the target.

A. Introduction to Dijkstra's algorithm

A crucial technique for route planning is Dijkstra's shortest path algorithm, particularly when it comes to mobile robots and navigation systems. The algorithm to find the optimal path between any two nodes or points in a tree or graph was created by Edger W. Dijkstra. In robotics, it is used to determine the most efficient path from one place to another. The benefits of this algorithm for route planning, its temporal characteristics as the number of nodes rises, and its use in mobile robots and navigation systems will all be covered.

B. Dijkstra's algorithm's basic steps

Various maps, resembling warehouse environments, were employed for the calculations. Dijkstra's algorithm with PRM, a global path planning method, was applied in scenarios where the maps were already known to the mobile robot, without considering dynamic robot behaviour. The algorithm followed these steps:

1. Each node initially had a distance value set to it, with the starting node's value being zero and all other nodes' values set to infinity.
2. The starting node was designated as the current node, and all remaining unvisited nodes were flagged.
3. For all unvisited neighbours of the current node, approximate distances were computed.
4. Once all other nodes had been considered, the current node was marked as visited.

A node that has been visited was not verified again; the distance recorded is definitive and minimal

Path planning is a specialized field of path planning that determines Uncertainty and consequences when walking around. This is especially important in mobile robots when dealing with unstable and uncertain environments where objects may move or the environment may change. Dijkstra's algorithm can be adapted to the planning method by integrating a probabilistic model into the edge weights of the graph. The flow diagram shown in Figure 1 will help to understand the algorithm. For example, edge weights can represent the probability of completing a path based on sensor data or historical observations rather than normal distance. This approach allows mobile robots to make decisions in an environment of uncertainty, increasing their flexibility and safety.

As the number of nodes in the graph increases, Dijkstra's algorithm becomes important. Real-time or near-real-time planning is often needed in mobile robots and navigation systems, so algorithm efficiency is very important. The time complexity of Dijkstra's algorithm depends on the data chosen. When a priority queue or minimum stack is used, it runs in O time,

$$(E + V * \log (V))$$

where E is the edge count and V is the vertices count. The time required to plan the path grows as the number of nodes increases. This results in a delay in designing the best route. So solving scalability problems involves optimization techniques like transforms like A* or D*Lite.

C. Algorithm description

To plan the best routes from the goal state to every state that is free of obstacles, this path-planning method first adopts the Dijkstra algorithm. The accessible area is the space the robot can move through, and the inaccessible area is the off-limits space. Based on the map and its binary occupancy grid [7], nodes are produced at random in the reachable area.

Additionally, the map is enlarged., which is a process of enlarging the occupied areas in a map representation to consider the size of the robot. Inflation works to make sure that the robot plans and follows its journey without running into any barriers. The inflation radius is taken into account based on the robot workspace. The nodes are later connected based on the distance decided. Based on the start and end points, the robot then determines the optimal route for it to take. Once on the best path, the robot proceeds along it until it either accomplishes the goal or returns to the beginning condition.

This is an interactive map by selecting the start and the goal positions in the map, one can observe the different paths as the node increases.

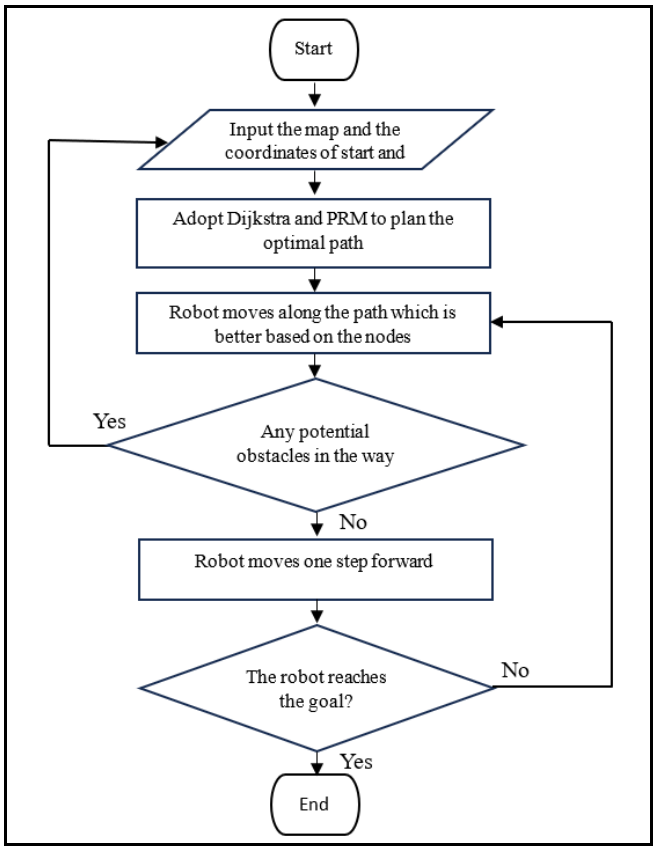

Fig. 1. Flow chart for the algorithm

III. IMPLEMENTATION

At first, it was difficult to construct the map, turn it into a binary image, and then access the nodes in the accessible area. It was a simple assignment to do by utilizing the MATLAB function. Later, making a few assumptions like not taking the mobile robot's dynamic behaviour into account.

A. Simulation Setup

MATLAB software was used to simulate this method. Figure 2 shows the original map, which was essentially constructed in an inverted color format. Figure 3 shows how the map changed when converted to grayscale.

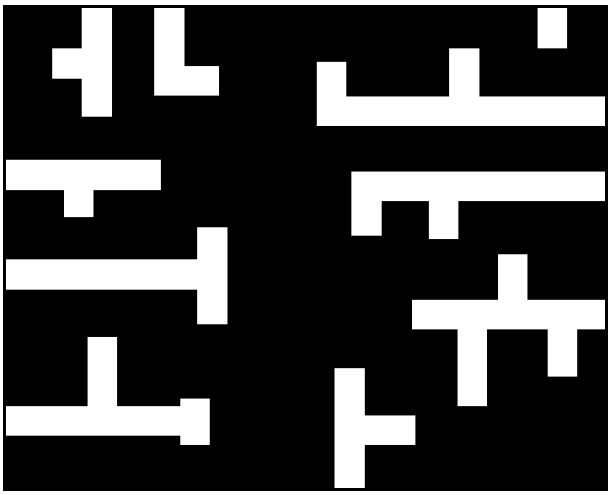

Fig. 2. Original Design of the Map

B. Assumptions

Considering this is merely a simulation of a model, it only shows the potential paths a mobile robot might take when given start and finish points for a task. The robot's dynamic behaviour was not taken into account because it depends on the application and the workplace. It was assumed additionally that this primarily operates in warehouses with minimal human interaction.

C. Map Selection

Selecting a map serves as a fundamental criterion, offering insights into the dynamic environment. The subsequent task of designing this map presents a substantial challenge, contingent on the specific requirements and applications of the mobile robot. Maps hold pivotal significance in the context of mobile robots, as they underpin the core functionality within the environment. These maps must be thoughtfully designed to be inherently conducive to the robot's operations while ensuring safety for both humans and robots. Inflating the map helps the robot to avoid collisions with the walls. The inflated map can be seen in Figure 4.

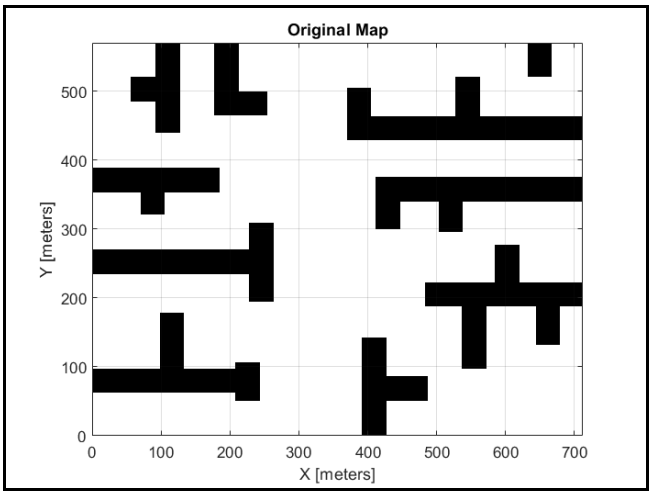

Fig. 3. Map after grayscale

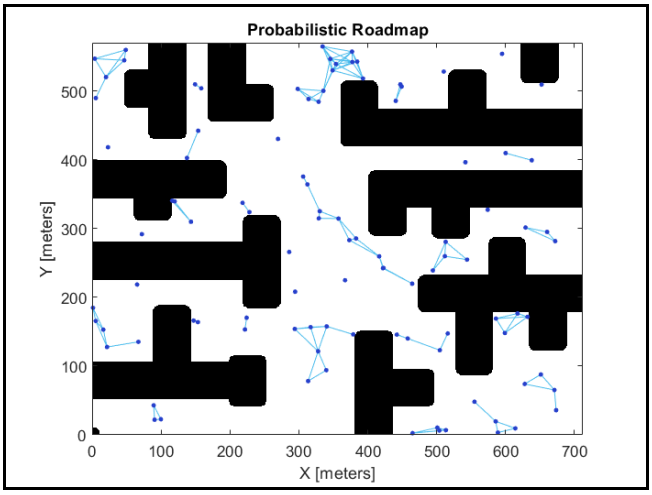

Fig. 4. Infalted Map

D. Algorithm Procedure

The procedure of the path planning algorithm is given below

1) Load the map image and create a binary occupancy map from it.
2) Display both the original map and an inflated version.
3) Initialize parameters for path planning, such as the number of nodes and connection distance.
4) Create a probabilistic roadmap (PRM) with the inflated map.
5) Capture the start and end locations interactively from the user.
6) For each path to calculate:
   a. Incrementally increase the number of nodes in the PRM.
   b. Use PRM to find a path between the selected start and end locations.
   c. Measure the time taken for path calculation.
   d. Store the path and elapsed time for analysis.
   e. Display path details, including elapsed time.
7) Plot a bar chart to compare elapsed times for different optimal paths.
8) Create a plot for displaying path planning results.
9) Plot all calculated paths on the map.

E. Results

As the output, one can observe the Probabilistic roadmap in Figure 5 for different paths in the inflated map with the nodes and the nodes connected based on the distance specified. The distance is set to 50 meters, so any node within 50 meters is connected. Also, consider the number of nodes randomly generated to be 100. The higher the nodes present the more accurate path is generated.

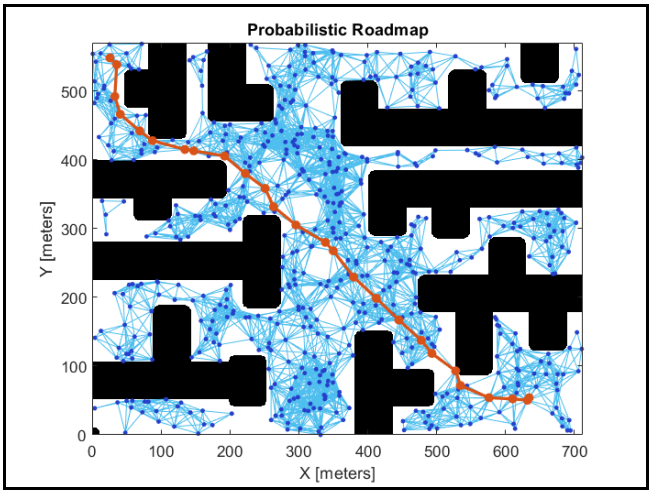

Fig. 5. Probabilistic Roadmap of the Map

F. Statistical Analysis

Case 1: Where considered the start and end location points to be the same for 5 different times as shown in the figure 6, to acquire different times required to reach figure 7, based on the nodes being added.

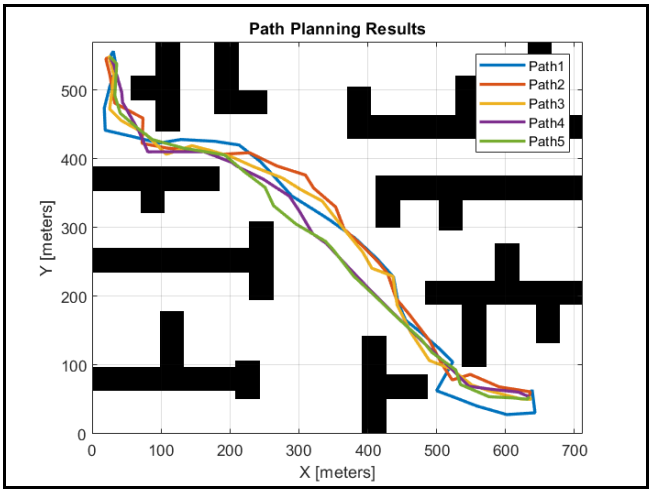

Fig. 6. Path with same start and end points

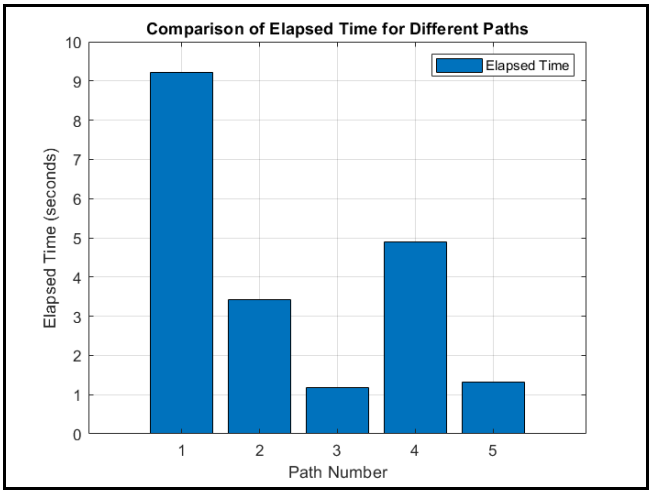

Fig. 7. Elapsed Time for the same start and end points

TABLE I. PARAMETERS CONFIGURATION

Path	*Calculation Time (seconds)*	*Number of Paths Calculated*	*Number of Nodes in the Path*
1	9.23	8	31
2	3.42	3	27
3	1.18	1	27
4	4.89	4	25
5	1.32	1	26

Case 2: In this case, consider the start and end location points to be different but extreme points as shown in figure 8 (random based on the accessible area in the map) for all 5 paths to compare the time taken in figure 9.

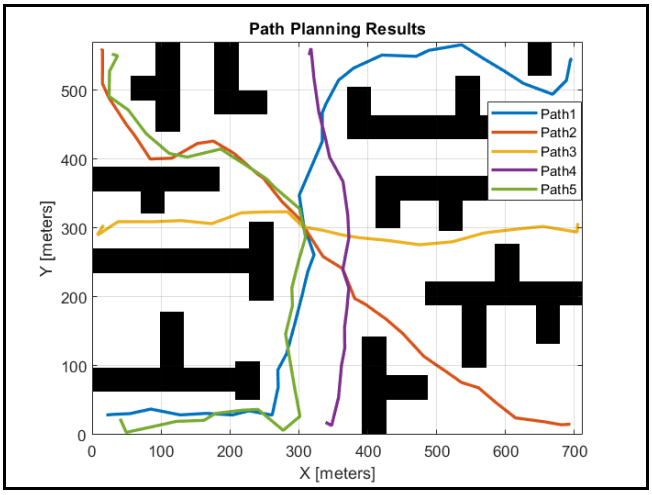

Fig. 8. Path with extreme start and end points

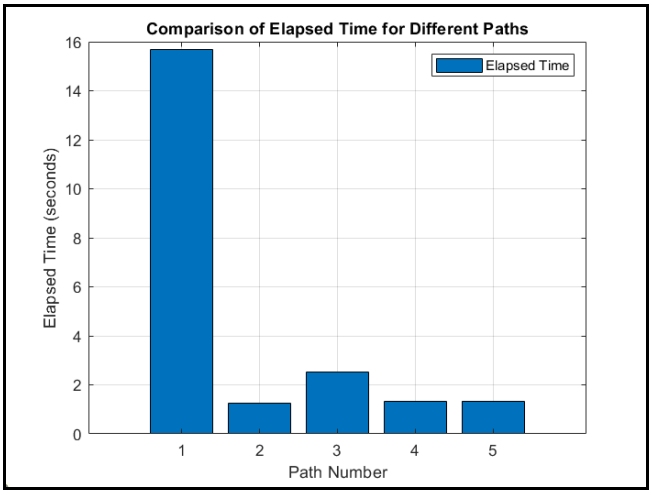

Fig. 9. Elapsed Time for the extreme start and end points

TABLE II. PARAMETERS CONFIGURATION

Path	*Calculation Time (seconds)*	*Number of Paths Calculated*	*Number of Nodes in the Path*
1	15.70	14	35
2	1.24	1	31
3	2.51	2	22
4	1.31	1	18
5	1.34	1	30

Case 3: In this case, the path was initially made smaller while the start and finish locations were assumed to be different as shown in figure 10 and later increase the path lengths, which were then compared to the time taken in figure 11.

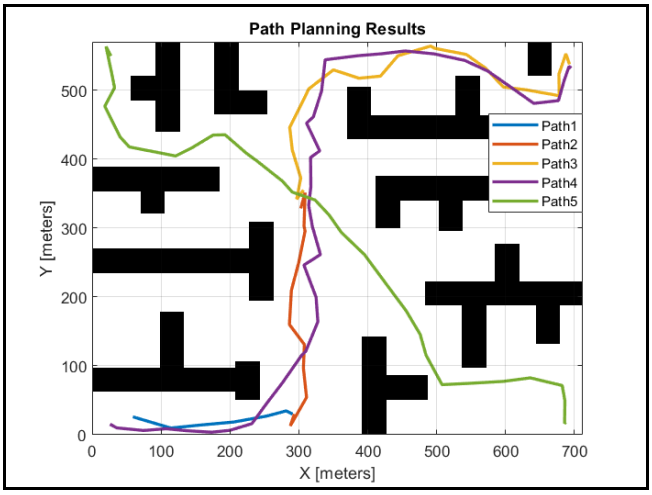

Fig. 10. Path with incremental start and end points

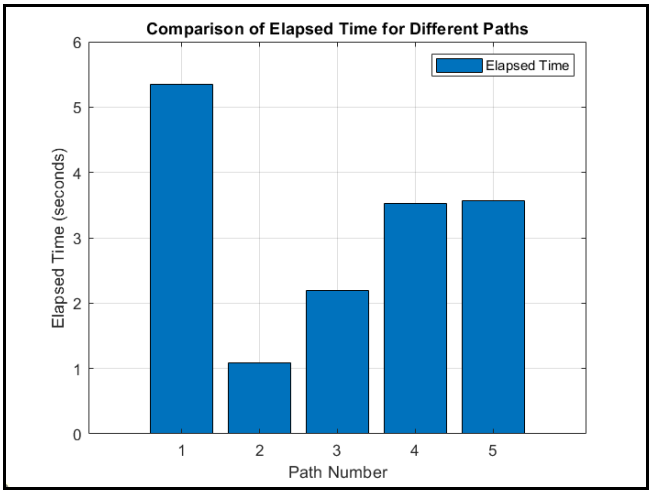

Fig. 11. Elapsed Time for the incremental start and end points

TABLE III. PARAMETERS CONFIGURATION

Path	*Calculation Time (seconds)*	*Number of Paths Calculated*	*Number of Nodes in the Path*
1	5.34	5	7
2	1.09	1	12
3	2.20	2	21
4	3.53	3	37
5	3.57	3	33

Case 4: Comparison of the 2 different cases i.e., Case 2 and Case 3. The Table IV is the comparison of the respective cases and the figure 12 is the graphical representation. Through this comparison, the graphical representation's numerous parameters such as the calculation of the time and number of paths and nodes in the path make sense to us.

TABLE IV. PARAMETERS CONFIGURATION

Case and Path	*Calculation Time (seconds)*	*Number of Paths Calculated*	*Number of Nodes in the Path*
Case2_Path1	15.7	14	35
Case2_Path2	1.24	1	31
Case2_Path3	2.51	2	22
Case2_Path4	1.31	1	18
Case2_Path5	1.34	1	30
Case3_Path1	5.34	5	7
Case3_Path2	1.09	1	12
Case3_Path3	2.2	2	21
Case3_Path4	3.53	3	37
Case3_Path5	3.57	3	33

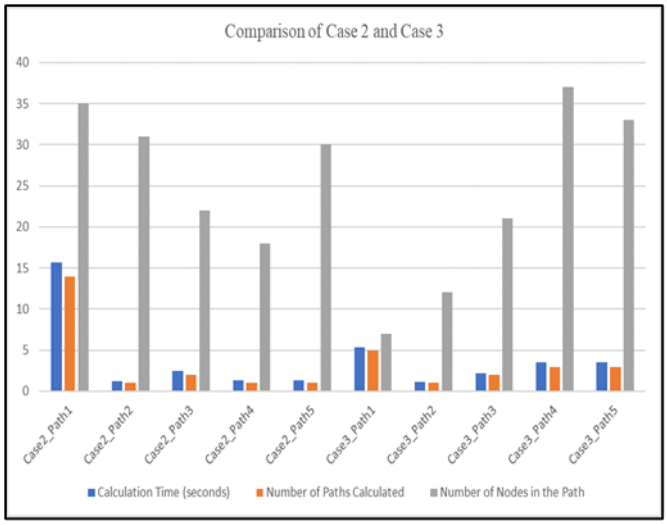

Fig. 12. Comparison of Case 2 and Case 3

IV. Related Insights

In mobile robots, Dijkstra's algorithm is frequently utilized. In order to see their surroundings and choose the safest and most effective path to their destination, mobile robots frequently rely on sensor data. The method can be used to find the fastest route and how long it takes. Moreover, the efficiency can be used to choose the path.

A. Application in navigation systems

In the field of navigation systems [8], Dijkstra's algorithm is indispensable for creating a way to display used contacts. GPS devices, registration cards, and vehicle navigation software use this technique to calculate the shortest or fastest route from the user's current location to the desired destination.

In the context of mobile robots and navigation systems, just-in-time planning is still quite a concern. To meet urgent requirements, engineers and scientists use a combination of techniques such as optimization algorithms, use of high-quality data, and sometimes compromise performance by using algorithms such as D* Lite Faster calculations. Balancing the need for improved visibility with immediate operation is an important consideration in the design of navigation systems and mobile robot controllers.

B. Complexity and Scalability

As the complexity of the environment increases, scalability will become a challenge for Dijkstra's algorithm. In cases where there are many nodes and edges, such as city-scale navigation or complex environments [9], the needs of the algorithm will be limited. This is especially true in mobile robotics, where robots must navigate large and complex spaces. Since the image represents the condensing medium, the time complexity of the algorithm will cause the plan to take a long time. For mobile robots operating in dynamic environments with dynamic paths [10] may vary, making quick decisions to avoid collisions and adapt to changes is crucial.

C. Optimization Technique

Efforts to solve the scalability problem of Dijkstra's algorithm in a large environment involved the development of various techniques to get good at it. Additionally, communication networks and distribution systems are used to speed up the planning process, especially in complex and time-critical applications.

V. Conclusion

In summary, Dijkstra's shortest path algorithm is still the basis of the scheme in the field of mobile robots and navigation systems. Its adaptability, versatility and reliability have made it widely used to find the best path in many situations. Whether it's guiding a mobile robot through a dynamic environment, helping users navigate using a GPS app, or assisting with complex GIS analysis, Dijkstra's algorithms play an important role. But as the complexity of applications and environments continues to increase, so does the challenge of maximizing capabilities and real-time performance. Scientists and engineers continue to innovate, develop efficient systems, integrate sensor data and discover new methods to meet changing needs, changing the competition.

As it traverses, the method efficiently adjusts the initial path computed from the initial state to the goal state. The suggested approach, however, can only be used to solve the path planning problem with a certain goal state. Because if the aim is altered while the robot is going, later pre-process the map data

once more and consider the algorithm's effectiveness as well. In addition, would like to add a few dynamic capabilities to the robot in the future that will make it easier to guide it around obstacles while maintaining refined curves and retracing its path when necessary.

REFERENCES

1. Nasir J, Islam F, Malik U, et al. RRT*-SMART: A Rapid Convergence Implementation of RRT*. International Journal of Advanced Robotic Systems. 2013;10(7). doi:10.5772/56718
2. Kavraki, L.E., P. Svestka, J.-C. Latombe, and M.H. Overmars. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," IEEE Transactions on Robotics and Automation. Vol. 12, No. 4, Aug 1996 pp. 566—580.
3. “Occupancy Grids - MATLAB & Simulink - MathWorks India.” https://in.mathworks.com/help/nav/ug/occupancy-grids.html
4. Hongmei Zhang, & Minglong Li. (2017). Rapid path planning algorithm for mobile robot in a dynamic environment. Advances in Mechanical Engineering, 9(12). doi:10.1177/1687814017747400
5. Abhipsa Kar, & Rati Ranjan Dash. (2022). SIMULATION PERFORMANCE COMPARISON OF DIJKSTRA’S ALGORITHM, A *ALGORITHM, AND DYNAMIC WINDOW APPROACH FOR MOBILE ROBOT PATH PLANNING. Zenodo (Cern European Organization for Nuclear Research). doi:10.5281/zenodo.6952222
6. Y. Deng, Y. Chen, Y. Zhang, and S. Mahadevan, “Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment,” Appl. Soft Comput., vol. 12, no. 3, pp. 1231–1237, Mar. 2012, doi: 10.1016/j.asoc.2011.11.011.
7. Elfes, Alberto. "Using occupancy grids for mobile robot perception and navigation." Computer 22.6 (1989): 46-57.
8. Michael Kneebone, & Richard Dearden. (2009). Navigation Planning in Probabilistic Roadmaps with Uncertainty. Proceedings of the International Conference on Automated Planning and Scheduling, 19, 209–216. doi:10.1609/icaps.v19i1.13359
9. Savva, M., Chang, A.X., Dosovitskiy, A., Funkhouser, T. and Koltun, V., 2017. MINOS: Multimodal indoor simulator for navigation in complex environments. arXiv preprint arXiv:1712.03931.
10. Youyu Liu, Bo Chen, Xuyou Zhang, & Renjun Li. (2021). Research on the Dynamic Path Planning of Manipulators Based on a Grid-Local Probability Road Map Method. Ieee Access, 9, 101186–101196. doi:10.1109/access.2021.3098044