

Mobile robot path planning in global environment using MATLAB simulation

BBVL Deepak¹, D Zahid Ahmed¹, Dulari Hansdah² Ashish Verma¹, Manoj Sangtani¹

¹ Department of Industrial Design, National Institute of Technology, Rourkela, India

² Department of Mechanical Engineering, National Institute of Technology, Jamshedpur, India

[bbv, 222id1315@nitrkl.ac.in](mailto:bbv,222id1315@nitrkl.ac.in), dhansdah.me@nitjsr.ac.in,

[222id1456, 222id1454@nitrkl.ac.in](mailto:222id1456,222id1454@nitrkl.ac.in)

Abstract

In this paper, a path planning algorithm named as Waypoint Algorithm (WA) is developed for an unmanned vehicle in a known environment using the general search for minimum distance or shortest path. The code for the algorithm is developed using MATLAB software. The study focussed on developing an algorithm to create a virtual simulation of an unmanned vehicle in a known or static environment like that of shop floor of a manufacturing cell layout with ten, five or three cells having a network of paths connected with each cell. The main purpose of this study is to find a shortest path between the cells when an input command is given to the unmanned vehicle to move from one cell to another. The algorithm also takes care of the command where the unmanned vehicle is asked to visit multiple cells one after another sequentially. Thus, the algorithm is robust enough to execute the command for making the unmanned vehicle move from source position to either one cell or multiple cells in a chronological manner.

Keywords: Path planning, mobile robot, fitness function, unmanned vehicle.

1 Introduction

In this modern era of industrial revolution 4.0, the technology is more focussed on developing systems which are more automated and intelligent. One of the most preferred topics for research in this domain is the path planning of autonomous mobile robots or unmanned vehicle [8]. Industries which are looking for automation of their material handling and transportation for reducing the manual labour cost, needs to be using the autonomous mobile robot or unmanned vehicle. One of the main problems of this autonomous vehicle is to determine the path which is optimal and shortest for moving the unmanned vehicle without hitting any obstacles or does not deviate from the predefined path of the environment of the workshop or manufacturing cell layout floor. Path planning of autonomous vehicle is divided into global path planning and local path planning [3]. In global path planning, the information of the environment is known, and all the obstacles and path are static while in case of local path planning, the information of the environment is unknown, and all the obstacles are dynamic [6]. There is various method for solving the path search for both the categories environment. For global path planning, the algorithms used for path search are heuristic approach which includes Dijkstra, A* and D* algorithms and artificial intelligence approach which includes ANN, GA, ACO, PSO and SA [4][7]. For local path planning, the algorithms used for path search are artificial intelligence approach which is same as global path planning as well as artificial potential field (APF), behaviour decomposition method, cased-based learning method, rolling windows algorithms [5]. In this study, the focus is given on static known environment and a novel path search algorithm coined as **Waypoint Algorithm (WA)**, is generated in virtual simulation using MATLAB software. The name of the algorithm is given because the algorithm takes a sequence of the initial and target destination waypoints and uses a fitness function which compares the length of the available connected paths from a particular source waypoint to all the connected target waypoints and finally outputs an optimal path which is the shortest of all the compared paths. The input sequence may be a single source to single target or single source to multi-target destinations.

2 Methodology

The first step in the algorithm was to develop the static or global map of the environment using the coordinates of the road and cells of the workshop which were also to be numerically labelled. This static environment map was to be displayed at the start of the algorithm. Then the algorithm also provided a way for the user to give the input

sequence i.e., robot's initial source cell position and the subsequent final cells. The algorithm provided a flexibility for the user to make the robot move either from one cell to another cell directly or via multiple cells. The algorithm was further developed using some critical waypoints on the static environment which were used as the source and the target. These critical waypoints are then mapped to their coordinates in the map and then the distances between all source waypoints and their corresponding connected target waypoints were calculated and stored in a weight matrix. The sequence given by user input was then mapped to the critical points and the sequence was updated accordingly. Then the updated sequence was used for determining the shortest path for each of the two waypoints in the sequence where the algorithm search for all the waypoints connected to a particular start/source waypoint (or cell position) and then finds the shortest waypoint connected to that source. Finally, the discrete points were plotted between the stored waypoints for giving an animation view of moving robot. The loop for plotting the discrete points was continued until the sequence was completed. For each loop of sequence, the distance covered between two cells was also displayed and the total distance was also displayed at the end of the loop i.e., after the robot has covered all the cells of the manufacturing workshop. The flow diagram of the algorithm is shown in figure 1 below.

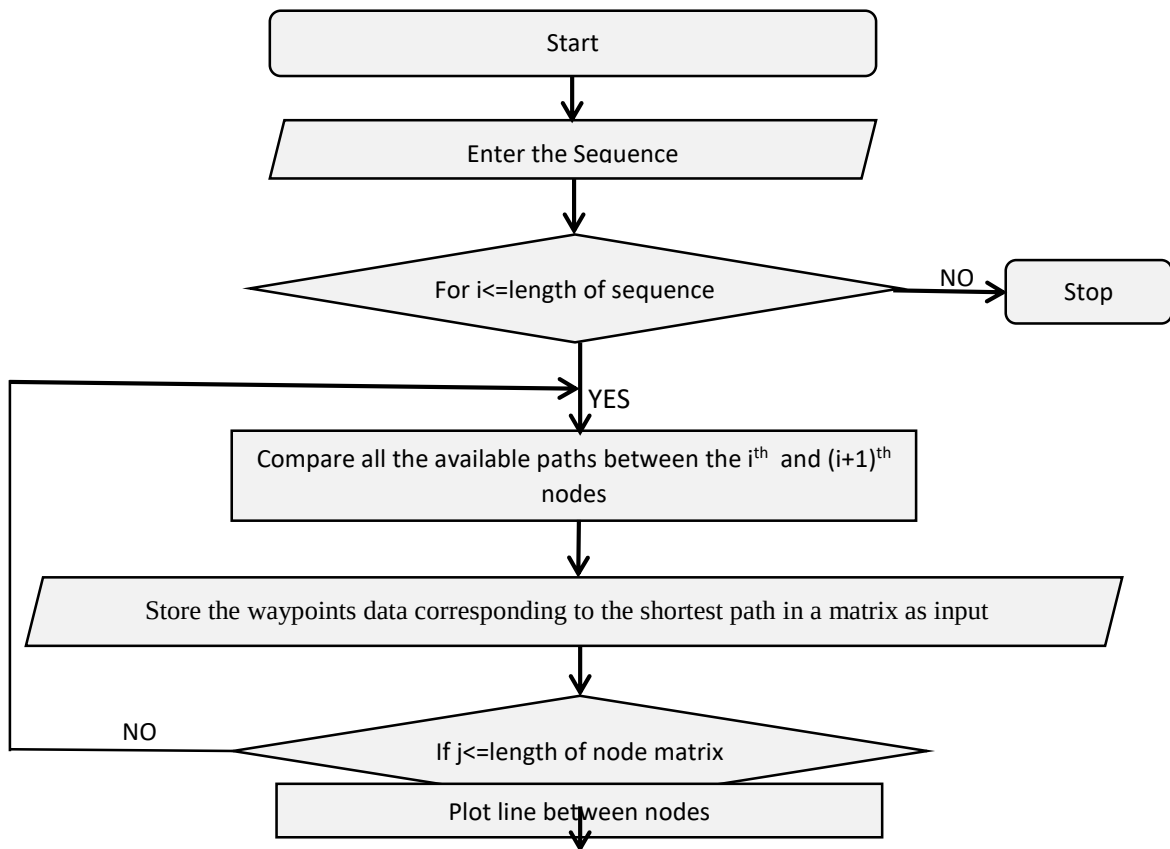


Figure 1: Flow chart of the Waypoint Algorithm (WA)

The data about the source and target waypoints and the probable roads which can be taken by the robot was generated using a global MATLAB function 'graph (s, t, w)' where **s** is the source points matrix, **t** is the target points matrix and **w** is the weight matrix. This graph was stored in the variable **G**. Then for each two waypoints from the sequence was considered and then the shortest path between them was calculated using the MATLAB function 'shortestpath (G, S1, S2)' where **S1** is the start node and **S2** is the destination node of the sequence. The function returned the results in the form of shortest path sequence matrix and the distance. Then the shortest path sequence was used for plotting the path of the robot and the distance was updated by adding the new distance for each loop of the next sequence and finally the algorithm ended with the displaying of the total distance covered.

2.1 Development of the Fitness function

As the input sequence was entered, a fitness function (Z) was calculated as given below.

$$Z = \sum_{i=1}^{n-1} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$$

Where n is the number of input cells, i represents the source and $i+1$ represents the destination. For every iteration of the sequence the value of i changes and thus when $i=1$, then the source is the first point and $i+1=2$ is the target point and thus after updating i becomes 2 where the previous target point becomes source, and a third point becomes the destination and so on until the sequence is completed. The main objective of the algorithm is to minimize the fitness function. The sequence which gave the minimum value of the fitness function Z is taken as the optimal sequence path and the robot moves through that path.

2.2 Representation of the manufacturing workshop

There are different types of manufacturing layouts with different number of manufacturing cells. Typical manufacturing workshop layout with three, five and ten cells are shown in the following figures 2.1, 2.2 and 2.3.

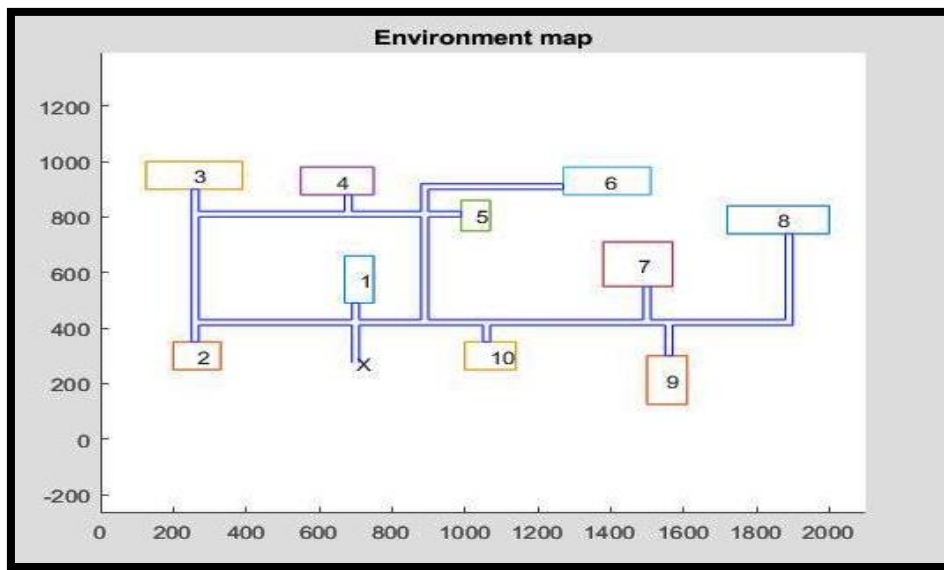


Figure 2.1: Workshop environment with ten cells

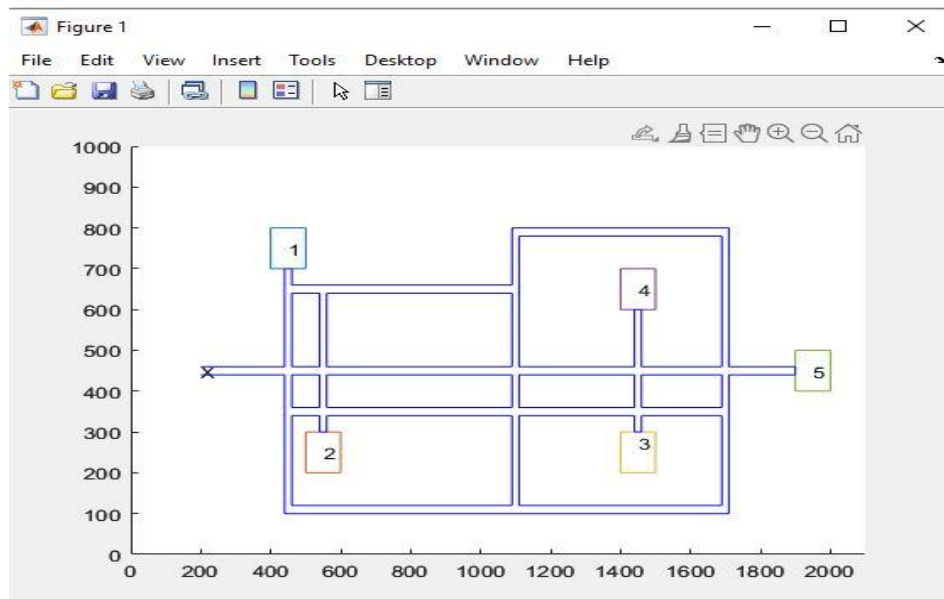


Figure 2.2: Workshop environment with five cells

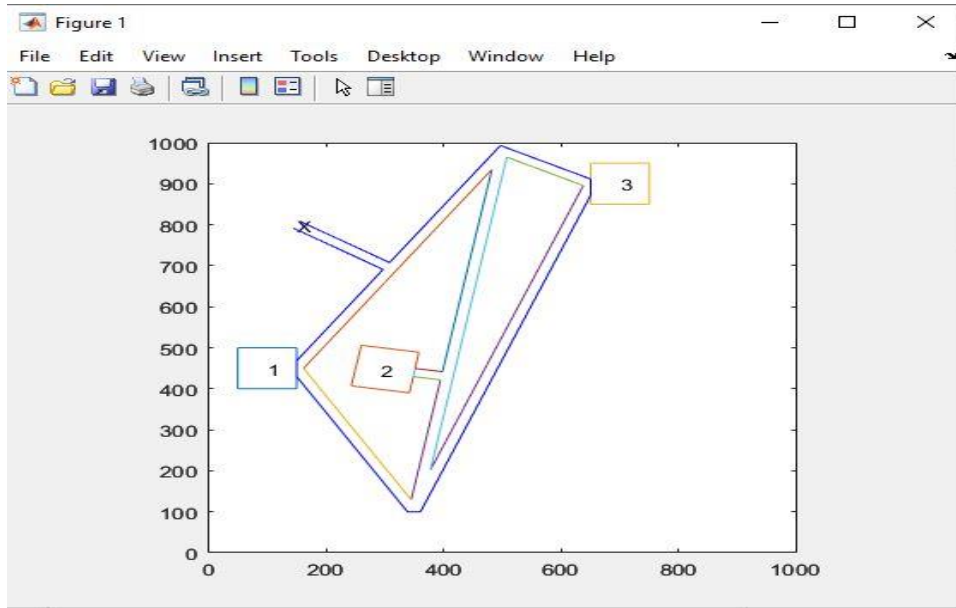


Figure 2.3: Workshop environment with three cells

The above manufacturing layouts are just assumptions, but the algorithm is robust enough to accommodate any type of manufacturing layout with any number of cells.

3 Results and Discussion

In this section the results of the code developed for the algorithm is discussed in detail. The MATLAB code developed first display the known environment in a popup window. The codes are developed for three different known environment scenarios with three cells workshop, five cells workshop and ten cells workshop. The cells are connected with multiple number of roads or paths.

The 'X' position in the map shows the home location of the robot. Now as the workshop environment is displayed by the algorithm the user can give the required input sequence. So, each case of the above three scenarios is discussed for different input sequences in the following subsections.

3.1 Case 1: Workshop environment with three cells

As seen from *figure 2.3*, the manufacturing layout with three cells is about 1000m×1000m size. The home position of the robot is at X which is represented as '0' for input sequence. Now a sequence **0-1-2-3** is taken. The output for each segment of the sequence is shown in *figures 3.1, 3.2 and 3.3*.

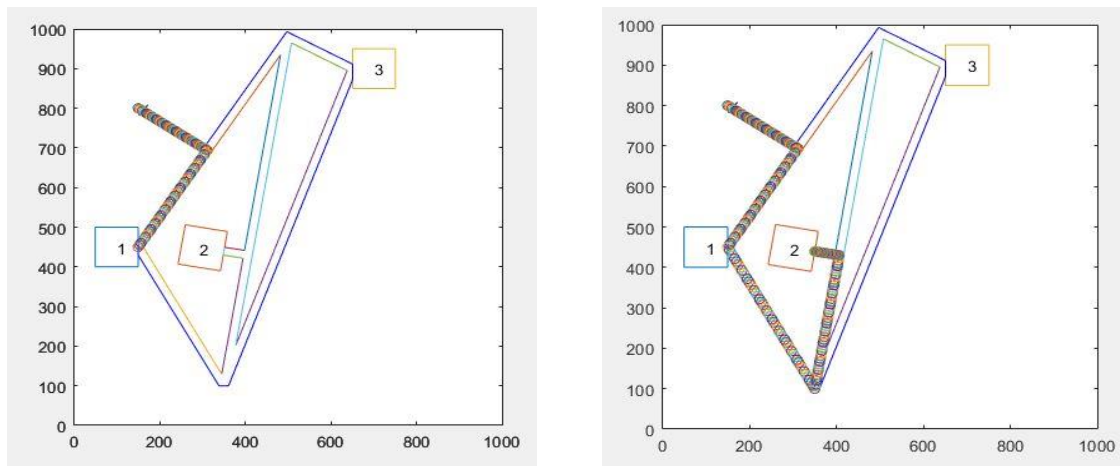


Figure 3.1: For start (0) to cell 1

Figure 3.2: For cell 1 to cell 2

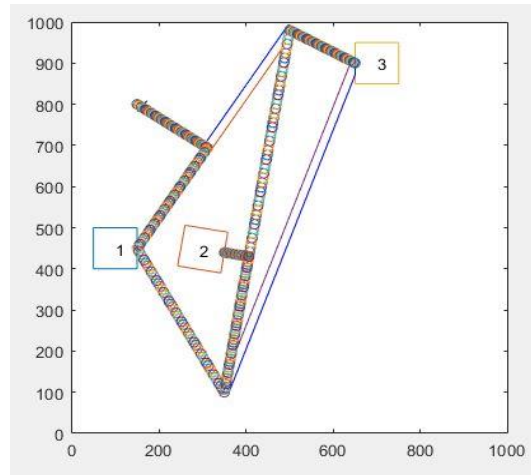


Figure 3.3: For cell 2 to cell 3

Now the final output for the whole sequence 0-1-2-3 and the corresponding distances as well as total distance covered is shown in *figure 3.4*.

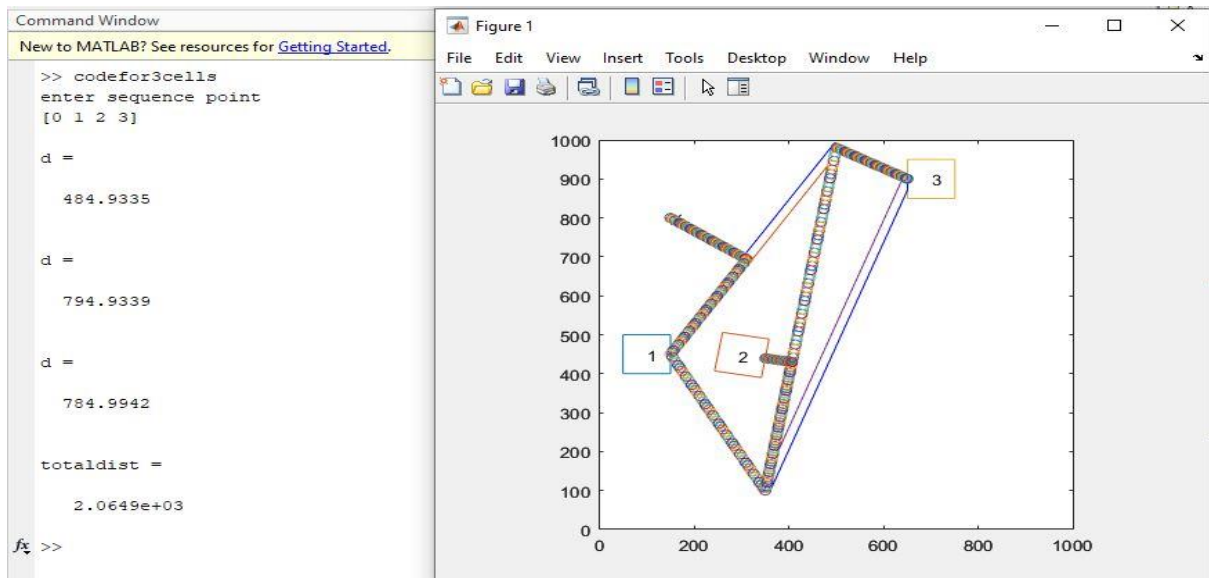


Figure 3.4: For start (0) to cell 3 with distances and total distance.

Now assuming the velocity of the robot as 5 m/s, the results are summarized in below table 1.

Table 1: Results for case 1

Sequence (0-1-2-3)	Distance (m)	Time (seconds)
0-1	484.9335	96.9867
1-2	794.9339	158.9868
2-3	784.9942	156.9988
For whole 0-1-2-3	2064.8616	412.9723

3.2 Case 2: Workshop environment with five cells

As seen from *figure 2.2*, the manufacturing layout with three cells is about 2000m×1000m size. The home position of the robot is at X which is represented as ‘0’ for input sequence. Now a sequence **0-1-2-3-4-5** is taken. The output for each segment of the sequence is shown in *figure 4.1, 4.2, 4.3, 4.4* and *4.5*.

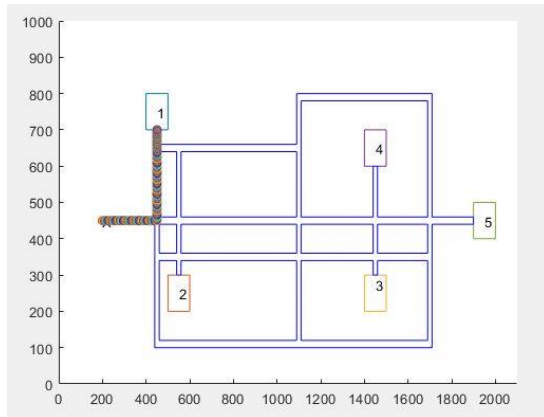


Figure 4.1: For start (0) to cell 1

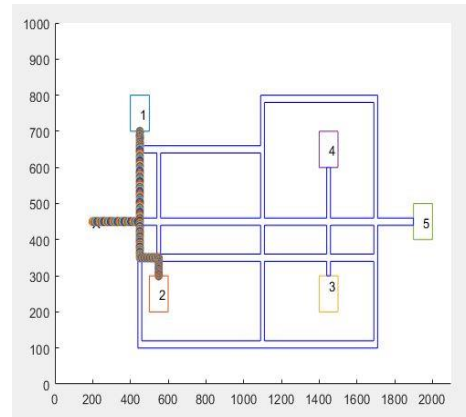


Figure 4.2: For cell 1 to cell 2

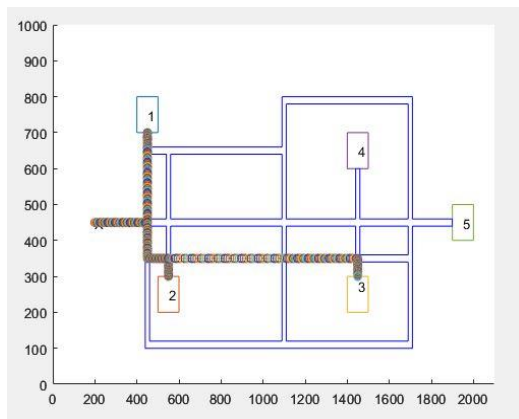


Figure 4.3: For cell 2 to cell 3

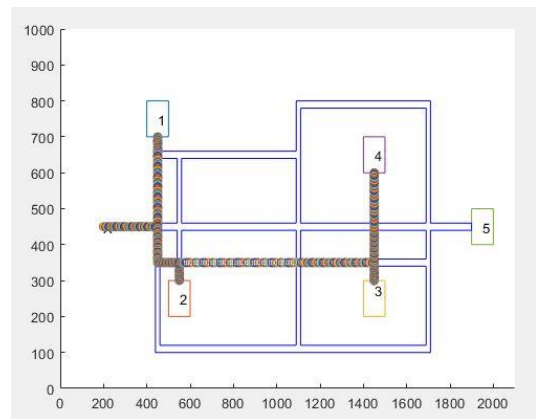


Figure 4.4: For cell 3 to cell 4

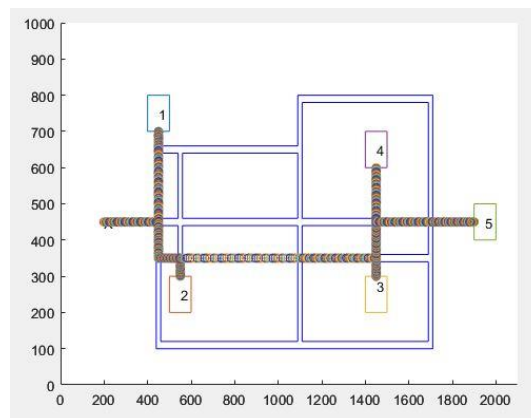


Figure 4.5: For cell 4 to cell 5

Now assuming the velocity of the robot as 5 m/s, the results are summarized in below table 2.

Table 2: Results for case 2

Sequence (0-1-2-3-4-5)	Distance (m)	Time (seconds)
0-1	500	100
1-2	500	100

2-3	1000	200
3-4	300	60
4-5	600	120
For whole 0-1-2-3-4-5	2900	580

3.3 Workshop environment with ten cells

In this subsection, the output of various input sequence for the workshop environment with ten cells are discussed in detail. The results of various input are shown in *figures 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8 5.9, 5.10* below.

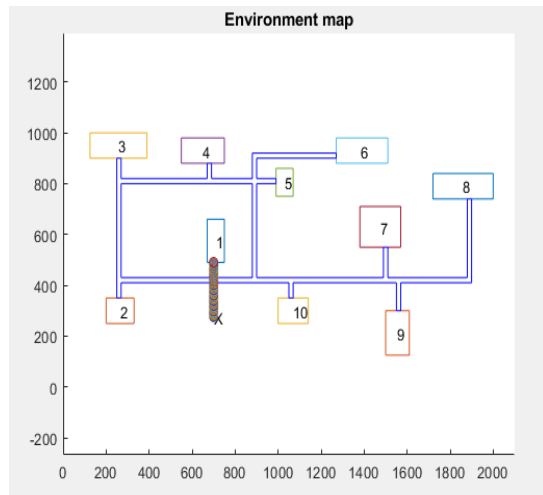


Figure 5.1: For start (0) to cell 1

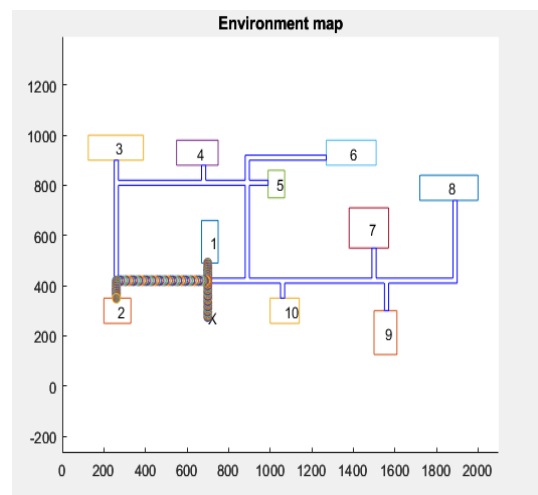


Figure 5.2: For cell 1 to cell 2

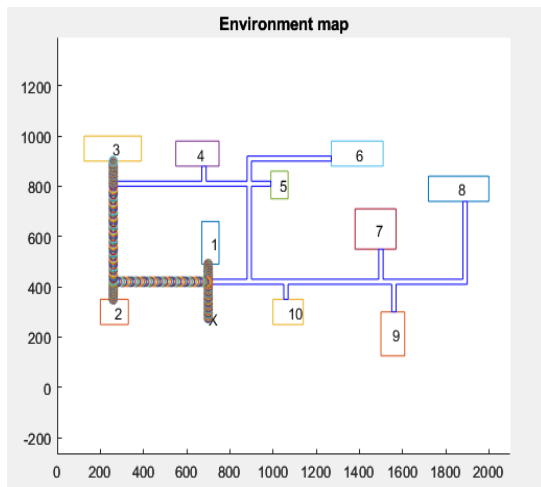


Figure 5.3: For cell 2 to cell 3

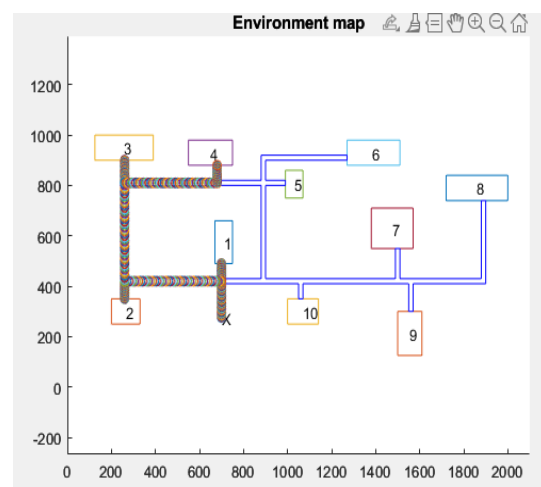


Figure 5.4: For cell 3 to cell 4

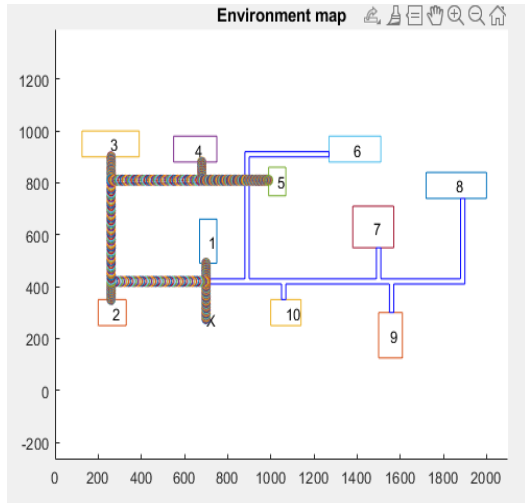


Figure 5.5: For cell 4 to cell 5

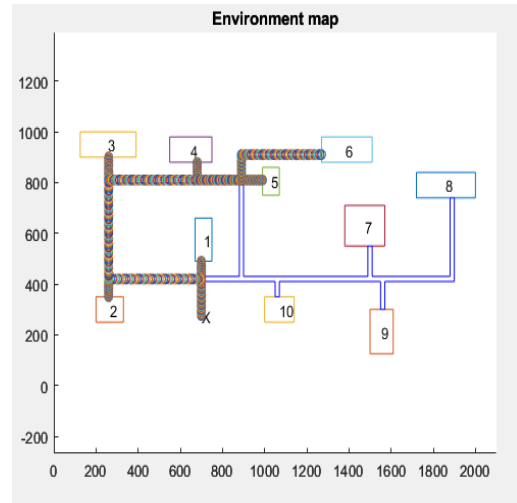


Figure 5.6: For cell 5 to cell 6

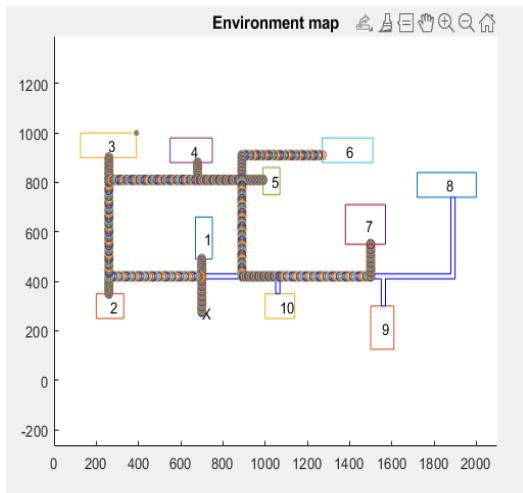


Figure 5.7: For cell 6 to cell 7

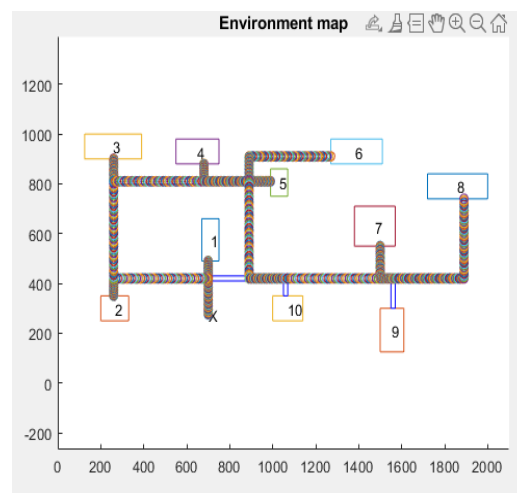


Figure 5.8: For cell 7 to cell 8

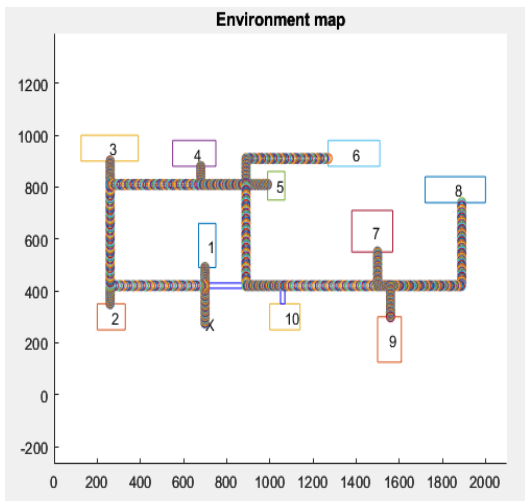


Figure 5.9: For cell 8 to cell 9

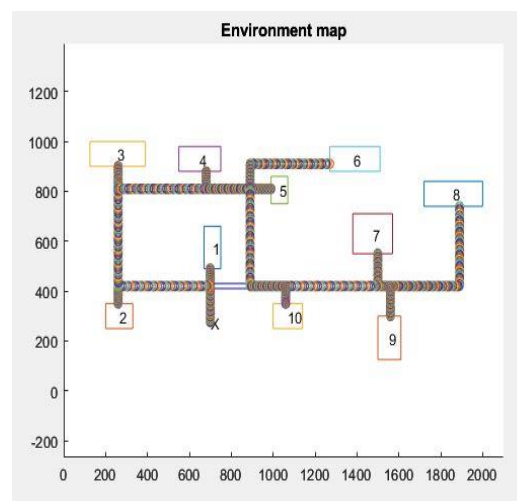


Figure 5.10: For cell 9 to cell 10

Now assuming the velocity of the robot as 5 m/s, the results are summarized in below table 3.

Table 3: Results for case 3

Sequence (0-1-2-3-4-5-6-7-8-9-10)	Distance (m)	Time (seconds)
0-1	215	43
1-2	580	116
2-3	550	110
3-4	580	116
4-5	380	76
5-6	580	116
6-7	1610	322
7-8	840	168
8-9	770	154
9-10	690	138
For whole 0-1-2-3-4-5-6-7-8-9-10	6795	1359

4 Conclusion

Many real-world applications of the shortest path planning algorithm, including robotic path planning and routing in transportation networks, make it a basic subject in computer science. The algorithm's objective is to determine the route between two locations in a graph that has the lowest cost.

The shortest path problem can be solved using several algorithms, such as Dijkstra's algorithm, Bellman-Ford algorithm, and A* search algorithm [2]. The choice of algorithm relies on the particular problem and requirements. Every one of those algorithms has advantages and cons of its own.

In conclusion, the shortest path planning problem is a significant and difficult computer science topic with several real-world applications. The creation of effective algorithms to address this issue has made it much easier to optimise routing protocols, robotic path planning, and transportation networks [10].

5 Future Scope

With the current state of technological development and the expanding application of artificial intelligence across a range of industries, the future potential of shortest path planning is fairly bright. Future applications of shortest path planning could be found in the following areas:

1. **Autonomous Vehicles:** Shortest trajectory planning algorithms will be essential in identifying the best routes for self-driving cars as they become in popularity. To ensure safe and effective travel, these algorithms will need to take into account current traffic information, road conditions, and other factors [9].
2. **Robotics:** Shortest path planning can also be used in robotics, particularly in the autonomous navigation systems used by drones and other robotic equipment. In order to find the fastest and least complicated paths to their destination, these devices can use shortest path algorithms.
3. **Urban Planning:** Algorithms that find the shortest path can improve transportation efficiency and lessen congestion in urban areas. This can be done by analysing traffic information and figuring out the optimal paths for moving vehicles in real-time.
4. **Healthcare:** In the healthcare industry, shortest path algorithms can be used to optimise patient transit inside hospitals. The fastest and safest routes for medical staff to take when moving patients between wards, operating rooms, and other facilities can be determined by hospitals using these algorithms.

Overall, the future of shortest path planning is bright, and we can expect to see more innovative applications of these algorithms in the years to come.

Reference

- [1] Dhulkefl, Elaf & Durdu, Akif. (2019). Path Planning Algorithms for Unmanned Aerial Vehicles. *International Journal of Trend in Scientific Research and Development*. Volume-3. 359-362. 10.31142/ijtsrd23696. <https://doi.org/10.31142/ijtsrd23696>
- [2] Parimala, M. Broumi S., Prakash, K. *et al.* Bellman-Ford algorithm for solving shortest path problem of a network under picture fuzzy environment. *Complex Intell. Syst.* 7, 2373-2381 (2021). <https://doi.org/10.1007/s40747-021-00430-w>
- [3] Pablo Marin-Plaza, Ahmed Hussein, David Martin, Arturo de la Escalera, "Global and Local Path Planning Study in a ROS-Based Research Platform for Autonomous Vehicles", *Journal of Advanced Transportation*, vol. 2018, Article ID 6392697, 10 pages, 2018. <https://doi.org/10.1155/2018/6392697>
- [4] Zhang, Han-ye, Wei-ming Lin, and Ai-xia Chen. 2018. "Path Planning for the Mobile Robot: A Review" *Symmetry* 10, no. 10: 450. <https://doi.org/10.3390/sym10100450>
- [5] Zhang, Yalong & Liu, Zhenghua & Chang, Le. (2017). A new adaptive artificial potential field and rolling window method for mobile robot path planning. 7144-7148. <https://doi.org/10.1109/CCDC.2017.7978472>.
- [6] Lou, Ping & Xu, Kun & Jiang, Xuemei & Xiao, Zheng & Yan, Junwei. (2021). Path planning in an unknown environment based on deep reinforcement learning with prior knowledge. *Journal of Intelligent & Fuzzy Systems*. 41. 1-17. <https://doi.org/10.3233/JIFS-192171>.
- [7] M. K. Rath and B. B. V. L. Deepak, "PSO based system architecture for path planning of mobile robot in dynamic environment," 2015 Global Conference on Communication Technologies (GCCT), Thuckalay, India, 2015, pp. 797-801, <https://doi.org/10.1109/GCCT.2015.7342773>.
- [8] Deepak, B. & Parhi, Dayal. (2012). Innate Immune based Path Planner of an Autonomous Mobile Robot. *Central European Journal of Computer Science*. 2. <https://doi.org/10.2478/s13537-012-0009-5>.
- [9] Deepak, B. & Parhi, Dayal. (2013). Target seeking behaviour of an intelligent mobile robot using advanced particle swarm optimization. CARE 2013 - 2013 IEEE International Conference on Control, Automation, Robotics and Embedded Systems, Proceedings. 1-6. <https://doi.org/10.1109/CARE.2013.6733749>.
- [10] Abdallaoui, Sara, El-Hassane Aglzim, Ahmed Chaibet, and Ali Kribèche. 2022. "Thorough Review Analysis of Safe Control of Autonomous Vehicles: Path Planning and Navigation Techniques" *Energies* 15, no. 4: 1358. <https://doi.org/10.3390/en15041358>.