Path Planning Algorithm for Mobile Robots in Static Environment

Dhaval Patel, Jashkumar Diyora, Trushar Shah

Mechanical and Mechatronics Department, U.V. Patel College of Engineering, Ganpat University, GJ, India

***Abstract*—Path planning for mobile robots is to find optimum path for robot to move at required location in its environment. In this Literature, a new approach to find the shortest path with less response time for mobile robot is discussed. The core concept lying in this procedure is discussed with simple illustrative example for various cases. For a fixed environment the general procedure of this approach is discussed in detail. Experimental results obtained by this method are compared with already implemented successful approaches and limitations of others approaches overcome by this method are discussed.**

***Keywords*—Path Planning, Optimisation, Mobile robot, Static Environment, AI Techniques, Obstacle avoidance**

I. INTRODUCTION

Path planning is one of the most important elements for Autonomous Mobile Robots. Path Planning is determination of optimum path for a mobile robot in order to change its current position (source point) to the desired location (destination point) [1, 2, 3, 7, 8, 9, 10, 11]. Path is geometric locus of points on which a mobile robot should move to reach the destination point by covering minimum distance [5, 7, 9,10].

Many techniques have been implemented for Path Planning in static environment for Autonomous Ground Mobile Robots [1, 2, 3, 5, 6, 9]. Such techniques include algorithms like NNA (Nearest Neighbour Algorithm), BFS (Breadth First Search) Algorithm, ACO (Ant Colony Optimisation), A* or Best First Search Technique [1, 2, 3, 5, 6,]. These algorithms provide optimum paths in different cases [1, 3, 5, 7, 9, 8].

For Static Obstacle Environment these methods have been proven efficient. Some of these approaches while determining shortest path may take more processing and computing time whereas some take less computing time but may not guarantee the shortest path.

In this Literature a new algorithm is introduced confirming the shortest path and less computing time in most of the cases for orientation of obstacles in a fixed environment.

For simplicity it is assumed that obstacles are already mapped (detected) in a rectangular or square boundary and there is a space (enough for robot to pass through) in all directions between any two distinct objects.

II. METHODOLOGY

A. Concept

In this approach the concept behind finding shortest path for robot is to record number of obstacles coming in way as seen from robot's source point to its destination point. Then to overcome only these obstacles one by one in order, while ignoring other obstacles in the static environment, a simple method is used to determine robot's next position in every step of algorithm to find the shortest path for robot.

B. Theory

To understand the method for overcoming the obstacles coming in way as seen from source to destination it is required to have a method for overcoming one obstacle between source and destination. This method is the core method of the whole procedure.

Obstacles are considered to be detected in a square or rectangular polygon. First, all points on the edge of this polygon are obtained. These points are called as nodes[5,7].

The nodes which are directly seen from source are termed as visible source nodes (n_s). The nodes which are directly seen from destination are termed as visible destination nodes (n_d). Depending on the number of visible source and destination nodes in a particular configuration there are five cases as follows:

Case 1: $n_s = 2$ & $n_d = 3$ OR $n_s = 3$ & $n_d = 2$

In this case there is always one common node visible to both source and destination as shown in fig. 1. There are two optimum paths possible in this case. One path is from source to destination via common node and other via node diagonally oppositetocommonnodeandnodenexttoitasshowninfig.1Then lengths of both paths are measured and path having shorter length is considered to be the shortest path of all possible paths.

For example, in fig. 1 visible source nodes are 1, 2 & 4 and visible destination nodes are 3 & 4 on obstacle's Polygon. Common node visible to source and destination is node 4 (marked yellow) on Polygon. Node diagonally opposite to node 4 is node 2. Thus there are two possible optimum paths S-2-3- D & S-4-D. By comparing lengths of both paths, the shortest path from source to destination is S-4-D.

Case 2: $n_s = n_d = 2$ & there is no common visible node

In this case for each visible destination node, a visible source node nearest to it is found out. For each of visible destination node one nearest visible source node is obtained. Thus, there are two possible paths from source to destination viaonen$_s$ and one n$_d$. Now lengths of both paths are measured and the path having smaller length is considered to be the shortest path fall possible paths.

In fig. 2, nodes 1&2 (marked green) are visible source nodes and 3&4 (marked red) are visible destination nodes on obstacle's Polygon. As discussed above, there are two possible optimum paths S-1-4-D and S-2-3-D. Comparing lengths of bothpathstheshorterpathtoovercomethisobstacleisS-2-3-

D. Hence this is the shortest path of all possible paths to overcome the obstacle

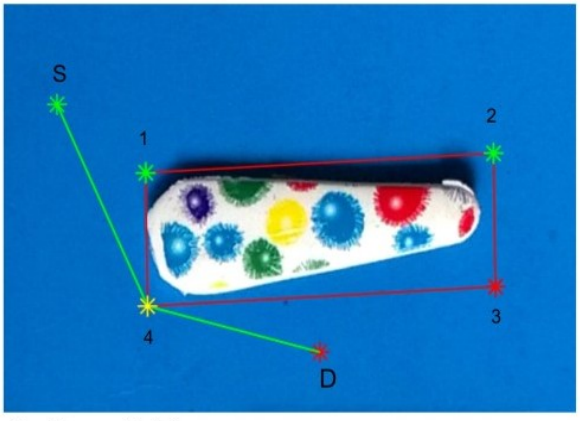
Case 3: $n_s = n_d = 2$ and there is one common visible node.

In this case, the shortest path is path joining source and destination via common visible node. Fig. 3 shows this path (green).

Case 4: $n_s = 3$ & $n_d = 3$

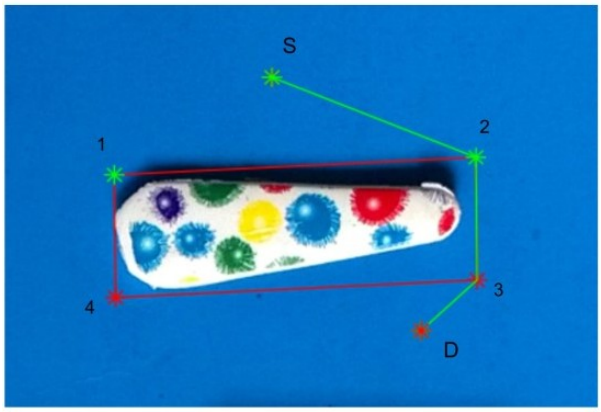
In this case there are two common nodes each visible to source and destination. So there are two optimum paths possible that join source and destination. Again lengths of both paths are measured. Path having shorter length is considered to be the shortest path of all possible paths.

In fig. 4, nodes 2 and 3 are common visible nodes to source and destination. So there are two possible optimum paths S-2-DandS-3-D. Lengths of both paths are measured with the help of Euclidean distance in 2D space. Path S-2-Disshorter among both paths and hence this path is the shortest of all possible paths.



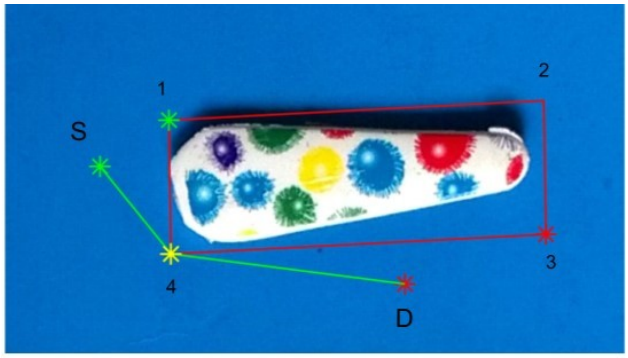
S = Source Point
D = Destination Point
1, 2, 3 & 4 are Obtacle Polygon's nodes

Fig.1. Path determined from S-D when there are three visible source nodes and two visible destination nodes. (Case 1)



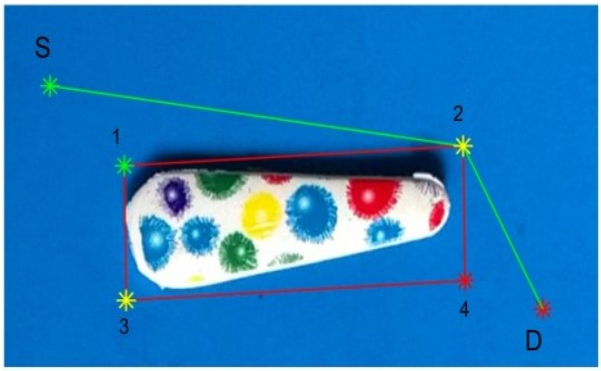
S = Source Point
D = Destination Point
1, 2, 3 & 4 are Obtacle Polygon's nodes

Fig.2. Path determined from S-D when there are two visible source and destination nodes with no common node. (Case 2)



S = Source Point
D = Destination Point
1, 2, 3 & 4 are Obtacle Polygon's nodes

Fig.3. Path determined from S-D when there are „two" visible source and destination nodes with one common node. Node 4 is common node. (Case 3)

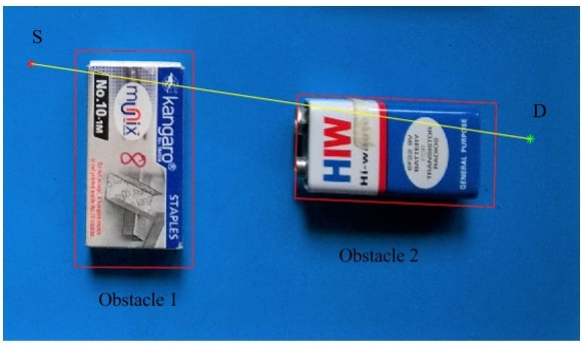


S = Source Point
D = Destination Point
1, 2, 3 & 4 are Obtacle Polygon's nodes

Fig.4. Path determined from S-D when there are two visible source/destination nodes and three visible destination/source nodes. (Case 1)

III. PROCEDURE FOR OBTAINING THE SHORTESTPATH

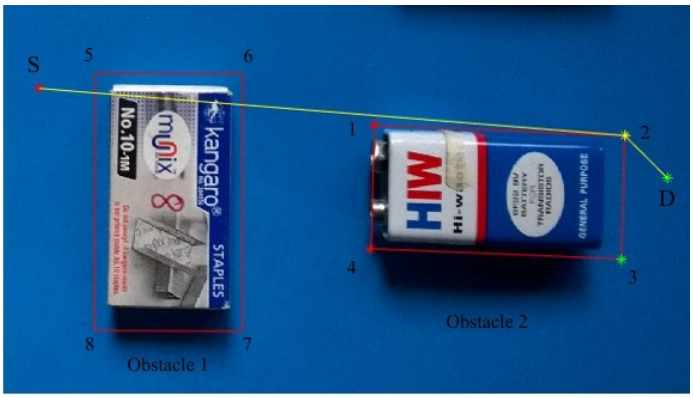
For simplicity, consider a case as shown in fig. 5 where two obstacles are coming in way as seen from source to destination.



S = Source Point
D = Destination Point

Fig.5. Two obstacles coming in way as seen (yellow line) from Source(S) to Destination (D)

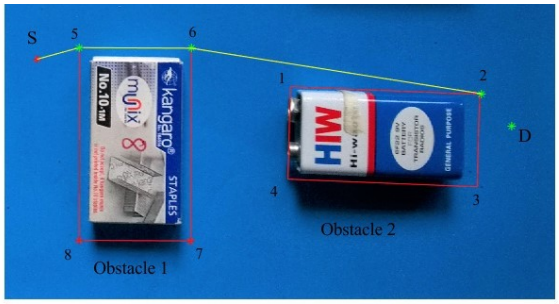
Considering only the nearest obstacle (obstacle 2) from Destination(D), a shortest path from Source(S) to Destination (D) is obtained as shown in fig. 6.



S= Source Point
D = Destination Point
1, 2, 3 & 4 are nodes on Obstacle 2
5, 6, 7 & 8 are nodes on Obstacle 1

Fig.6. Determining the shortest path (yellow path) from S-D by only considering the nearest obstacle (Obstacle 2) to Destination (D).

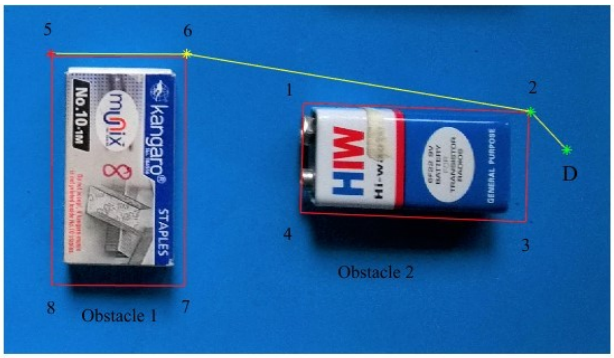
On this path node nearest to destination is now taken as temporary destination. Node 2 is taken as "temporary destination". Now considering this node and Source (S), all obstacles as seen from Source to node 2 are recorded. Here obstacle 1 comes in way. Again the shortest path from S to node2 is obtained to overcome obstacle (Obstacle1) nearest to node 2 as shown in fig. 7. This temporary path obtained is S-5- 6-2. Now again node nearest to earlier temporary destination (node 2) on this path is taken as new temporary destination.
Thisnewtemporarydestinationisnode6.Allobstaclesasseen from Source to node 6 are recorded and similarly next temporary destination is obtained till the latest one directly connects the Source (S). This set of all temporary destinations obtained is node 2, node 6 and node 5. The path S-5-6-2-D in fig7 is obtained "temporary destination path".



S= Source Point
D = Destination Point
1, 2, 3 & 4 are nodes on Obstacle 2
5, 6, 7 & 8 are nodes on Obstacle 1

Fig.7. Determining the shortest path (yellow) from Source (S) to node 2 to overcome nearest obstacle to node 2. The nearest obstacle to node 2 is Obstacle1.

On this temporary destination path, the node nearest to S is chosen as "temporarysource".Thistemporarysourceisnode5. Now considering node 5 and original Destination (D), whole procedure is again repeated and a temporary destination path from node 5 to D is obtained as shown in fig.8

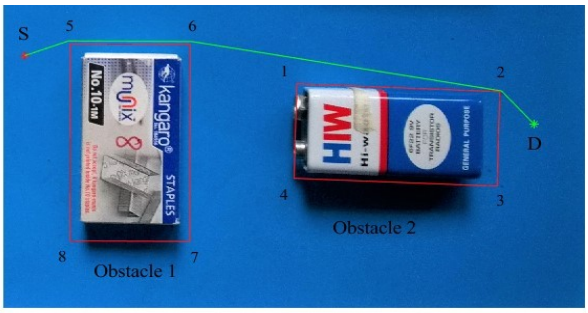


S= Source Point
D = Destination Point
1, 2, 3 & 4 are nodes on Obstacle 2
5, 6, 7 & 8 are nodes on Obstacle 1

Fig. 8 Taking node 5 as temporary source and Destination D, temporary destination path (yellow path) 5-6-2-D is obtained.

Again a new temporary destination path 5-6-2-D is obtained and on this path node nearest to node 5 is taken as next temporary source. Thus, whole procedure is repeated to obtain a set of all temporary source nodes in order from S to D till a node comes that connects directly to Destination (D). In this case node 5, node 6 and node 2 are obtained temporary sources.

Finally, as shown in fig.9, the path obtained by connecting Source and these nodes in order to destination as S-5-6-2-D is found to be the shortest possible path from Source (S) to Destination (D).



S= Source Point
D = Destination Point
1, 2, 3 & 4 are nodes on Obstacle 2
5, 6, 7 & 8 are nodes on Obstacle 1

Fig.9. Final path (green path) obtained from S to D.

In general, suppose 'n' obstacles are coming in way as seen from source to destination. So there are 'n' polygons (rectangular/square) coming in the way from source to destination. To avoid all n obstacles first focus is made to consider only nth obstacle from source and mark the points on nth polygon (of nth obstacle) to overcome this obstacle with shortest possible path to reach destination. All other n-1 obstacles are not considered. The nearest point from destination of this path on nth polygon is considered as temporary destination. Again as seen from source all obstacles coming on the way to this point (temporary destination) are recorded. If there is not any obstacle coming in way to reach this point from source this path is considered to be temporary destination path.

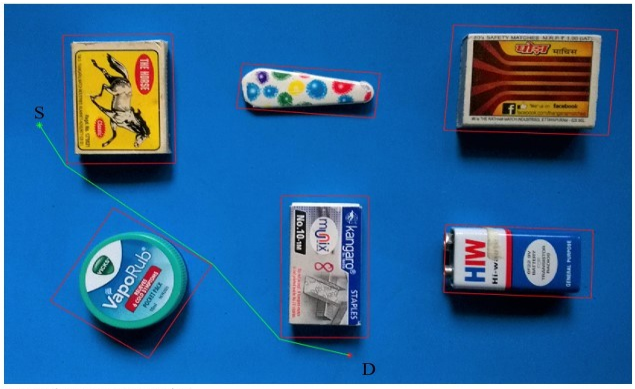
If there is/are obstacles coming in way from source to this point than same procedure is repeated considering this point as a destination and again next temporary destinations are obtained.

All the points considered to be temporary destinations are recorded until one of these points directly connects source to its destination. The obtained points in order from destination are supposed to connect source and destination with a temporary destination path.

On this path the first point from source is considered to be new temporary source. Now considering this temporary source point and original destination, from the previous procedure, again a path from this temporary source to destination is obtained by connecting all temporary destination points. On this path the point next to previous temporary source is again taken as next temporary source point.

The whole procedure is repeated until a temporary source point is obtained that connects directly with destination with no obstacle coming in way as seen from that temporary source to destination. The set of source points obtained during this procedure should connect the original source and destination with shortest possible path.

Thus, aim behind this procedure is to obtained temporary source points. In every step of each temporary source point obtained determines next position for robot to move. Hence, to obtain the shortest path by continuously determining robots next position from source point to destination this Algorithm is named as "Next First Search Algorithm"(NFS).An example of path determined by Next First Search Algorithm is shown in fig. 10.



S = Robot's Source Point
D = Robot's Destination Point

Fig.10. Path determined from Source (S) to Destination (D) (Green path)

IV. RESULTS AND DISCUSSIONS

The method introduced in this literature was successfully tested for a mapped environment with fixed and spaced obstacles. Results obtained from Next First Search (NFS) Algorithm, ACO (Ant Colony Optimization) Algorithm and BFS (Breadth First Search) Technique for same fixed and spaced obstacle environment, with same computing tools, were compared in Table. I. The results shows that in comparison to successfully implemented techniques like ACO and BFS, NFS algorithm gave minimum error in finding the shortest path from source to destination. As ACO is nonparametric in nature it required more computing time e.g. 10.2880795 seconds to find solution through search technique. Although ACO requires more time in comparison to other techniques, it gives the shortest path with zero errors in most of cases [5, 6]. Compared to ACO for the same path BFS technique gave an error of 115.7380 in finding optimal solution for path. Although it gave an error it required less computing time of 2.90304 seconds to find solution. For the same input conditions as ACO and BFS techniques NFS gave zero error and a computing time of 0.482619 seconds to find optimal solution for path which is almost 5 times less than BFS and 20 times less than ACO.

It can be seen from results of six experiments of Table I. that NFS technique requires very less time to find optimal path compared to ACO and BFS approaches and it has proved to be quick responsive. Along with quick response it also gave rms error of 0.8563 while ACO gave rms error of 4.8587 and BFS gave rms error of 98.2588. NFS algorithm gave the least rms error of 0.8563. This error is due to shifting each temporary source node obtained outside the edge of detected obstacles polygon by an acceptable amount in complex orientation of obstacles.

TABLE I. COMPARISON OF ANT COLONY INSPIRED (ACO) ALGORITHM, BREADTH FIRST SEARCH (BFS) TECHNIQUE AND NEXT FIRST SEARCH (NFS) TECHNIQUE

No	Actual Distance	ACO Algorithm			BFS Technique			NFS Technique		
		Measured Distance	*Error*	*Computing Time (seconds)*	*Measured Distance*	*Error*	*Computing Time (seconds)*	*Measured Distance*	*Error*	*Computing Time (seconds)*
1	457.9060	457.9060	0.0	10.2880795	573.6441	115.7380	2.90304	457.9060	0.0	0.482619
2	499.1225	499.1225	0.0	8.596091	559.2591	60.1366	2.861058	499.1225	0.0	0.840219
3	734.6906	734.6906	0.0	17.142356	833.7403	99.05	2.739960	734.6906	0.0	0.733984
4	740.4859	752.4781	11.9922	15.93820	900.7824	160.2965	2.6542185	740.4859	0.0	0.781152
5	411.6511	411.6511	0.0	9.481088	411.6511	0.0	2.794195	411.6511	0.0	0.644712
6	569.0062	569.0062	0.0	13.39741	669.5675	73.5613	2.303755	571.1039	2.0977	0.6995675
RMS Error	-	-	4.8957	-	-	98.2588	-	-	0.8563	-

This error occurs only when orientation of obstacles is complex and obstacles polygons are inclined in complex ways. But for most of the cases there is no error in solution obtained by NFS algorithm compared to actual shortest path.

V. CONCLUSIONS

In fixed Environments where task for mobile robot is to reach a particular location quickly it requires the shortest path to overcome obstacles coming in way. Also to complete multiple tasks by moving from one position to another in given time there is a need of an approach that gives the shortest possible path in a short time. Though many successful approaches have been implemented to find shortest path for mobile robots there is either an error inpath related to original possible path or required processing time is more [3,5]. NFS algorithm eliminates both this limitations to give shortest path taking very less processing time. This technique also occupy less computing memory as it doesn't search for solution for all nodes as in case of ACO but only considers the obstacles coming in way from source to reach destination.

REFERENCES

[1] Alpa Reshamwala and Deepika P Vinchurkar "*Robot Path Planning using An Ant Colony Optimization Approach: A Survey*" . (IJARAI) International Journal of Advanced Research in Artificial Intelligence, Vol. 2, No.3,2013.

[2] Sandeep Dhakal and Raymond Chiong. (2008). "*A hybrid nearest neighbor and progressive improvement approach for travelling salesmen problem*".IEEE.

[3] Tolga Yuksel, Abdullah Sezgin. "*An implementation of path planning algorithms for mobile robots on a grid based map*". International Journal of New Technology and Research. Vol 2, Issue5.

[4] Marco Dorigo, Mauro Birattari, and Thomas Stutzle (2006), "*Ant colony optimization: Artificial ants as a computational intelligence technique*." IEEE Computational intelligence magazine. Pp.28-39.

[5] Trushar Shah, Jay Patel, Vinod P. Shingadiya, V.B Patel "*Waveguide Path Planning using Nearest Neighbour, Breadth First Search and Ant Colony Inspired Optimization*" (IJRSI) International Journal of Research and Scientific Innovation Vol III, Issue VI,June2016.

[6] M.Bala Subramanian, Dr.K.Sudhagar,G.RajaRajeswari "*Intelligent Path Planning Of Mobile Robot Agent By Using Breadth FirstSearch Algorithm" International Journal of Innovative Research inScience, Engineering and Technology. Volume 3, Special Issue 3, March 2014*

[7] Buniyamin N., Wan Ngah W.A.J., Sariff N., Mohamad Z. "*A Simple Local Path Planning Algorithm for Autonomous Mobile Robots* " International Journal of Systems and Application, Engineering and Development, Issue 2,Volume5.

[8] Mohamed EL KHAILI "*Path Planning in Dynamic Environment*" (IJACSA) International Journal of Advanced Computer Science and Applications. Vol. 5, No. 8, August2014

[9] Garrido, Moreno, Blanco & Jurewicz, "*Path Planning for Mobile Robot Navigation using Voronoi Diagram and Fast Marching,*" International Journal of Robotics and Automation (IJRA), Vol.2:Issue (1):2011

[10] Sariff, N., Buniyamin N. "*Ant Colony System For Robot Path Planning In Global Static Environment*". in 9th WSEAS International Conference on System Science and Simulation in Engineering (ICOSSSE'10),. 2010.4th -6th October, Iwate, Japan: WSEASPress.

[11] A.Oualid, Karim and Redouane, "*A Sensor Based Navigation Algorithm for a Mobile Robot using the DVFF Approach*," International Journal of Advanced Robotic Systems,Vol 6, No.2(2009) ISSN 1729-8806, pp. 97-108