https://doi.org/10.31256/Ye1Dq6J

Visibility Graph (VG) - based path planning method for multi-robot systems

Fatma Alwafi
Material and Engineering Research Institute
Sheffield Hallam University
Sheffield, U.K.
b3046884@my.shu.ac.uk

Xu Xu
Material and Engineering Research Institute
Sheffield Hallam University
Sheffield, U.K.
xu.xu@shu.ac.uk

Lyuba Alboul
Jacques Penders[1]
Material and Engineering Research Institute
Sheffield Hallam University
Sheffield, U.K.
l.Alboul@shu.ac.uk

Abstract- This paper addresses path planning algorithms for a multi-robot system based on the Visibility Graph (VG) method.VG has been selected because it produces solutions with optimal path lengths from a start position to a target. However, the main disadvantages of VG are: (1) it plans paths which may go through the vertices of obstacles; (2) the computation time increases with the increased number of obstacles. Therefore, two new algorithms have been developed: Central Algorithm (CA) and its associate, Optimisation Central Algorithm (OCA). Simulation results show that the proposed algorithms can find optimal paths in 2D environments.

Keywords— Multi-robot path planning, Visibility graph, Central baseline, Optimisation, Optimal path

I. INTRODUCTION

The path planning problem for multi-robot teams has gained extensive attention in robotics research. The path planning problem is to develop a strategy that allows the robots to reach their targets over short, collision-free paths [1]. One of the path planning approaches is a VG, which is one of the roadmap methods [1][2]. The VG considers the vertices of the obstacles in the environment, to be vertices through which robots can arrive at their desired locations [1][3]. The VG satisfies two of the path planning criteria: (i) it provides the optimal path, (ii) it is complete which means it always produces a path if one exists, guaranteeing that the robots will find the shortest paths to their targets [1], especially if it is combined with Dijkstra's algorithm. Dijkstra's algorithm is one of the algorithms that guarantee to provide an optimal path. So, we have selected a VG-based method to create paths in an environment in conjunction with Dijkstra's algorithm [4][5][6]. However, the main drawback of this method is that it plans a path that forces the robots to pass as close as possible to the obstacles, and its calculation time increases with the increase of the number of obstacles in the environment [1][3]. Therefore, we have developed the CA coupled with OCA, which is based on the VG method. The main idea of these algorithms is that the obstacles associated with Central Baseline (CB) are only considered whilst the rest are discarded during the path calculation. Thus, it can create paths relatively fast and is convenient for path planning applications in obstacle-rich environments because it uses a small set of obstacles, whilst retaining the advantages of the VG [1][3][6].

II. METHDOLOGY

We have developed path planning algorithms based on the VG called CA and OCA. The first algorithm is called CA because the straight line that links the starting positions and the goal positions, which we call CB, is the main factor that defines which obstacles are to be involved in establishing the VG. The first step in the algorithms is to find the obstacles that intersect with CB, and the second one is to generate a set of waypoints around obstacles.

A. Central Algorithm

Considering the environment in Fig.1, we first create the CB that is connecting the start and goal positions. The obstacles that overlap with CB are determined, and the intersection points between the obstacles and CB are highlighted in grey. The waypoints are generated from each vertex of each obstacle intersect with CB, and calculated for each half of the map, and the lines linking waypoints and their intersection points (vertex of obstacle) are orthogonal to the original CB. There are two intersection points lying along each CB, each point having two waypoints, so the total of four waypoints are computed, which are marked in blue colour. Successive waypoints are related to each other to create multiple possible collision-free paths for robots around obstacles.

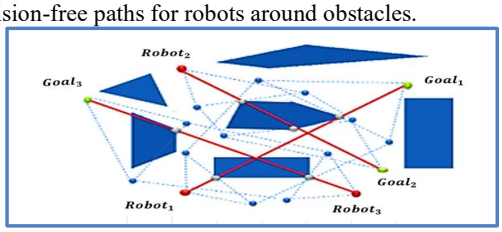

Fig. 1. Steps of CA to generate waypoints around obstacles

Fig.2 illustrates waypoint calculation, where the CB joining the start point S and goal point g intersect an obstacle, and its edges are shown as dotted lines, at point $u(x, y)$. The normal distance between the point $u'(x', y')$ and the straight line must exceed the maximum distance m between any node in that object and CB by a safe distance(δ). The waypoint u' is computed on either side of the intersection point as follows:

$$u' = u \pm (m + \delta) \begin{bmatrix} -cos(90 - \varphi) \\ sin\,(90 - \varphi) \end{bmatrix} \quad (1)$$

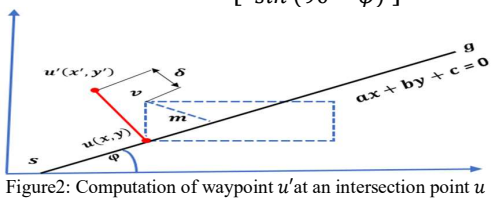

Figure2: Computation of waypoint u'at an intersection point u

B. Optimisation Central Algorithm (OCA)

Generating paths which are safer and not too close to the obstacles, an improvement to CA, is proposed in this section. To address this, the size of obstacles (i.e., the distance between them) can be expanded in the environment by a certain distance before the paths are planned. This distance is called the Safety Distance (D_s), which could be used to optimize the drawback of the CA. D_s is important for the robots to be capable to traverse the planned paths without collision with any obstacle and to ensure these paths are safe for the robots so that they can traverse through the vertices of obstacles in different workspaces, even with added new

[1] The author died prior to the submission of this paper. This is one of the last works of him

obstacles, the paths of robots will not change, because with D_s the placing of obstacles will not affect the generated paths.

III. Simulation Results

We have developed a path planning software package to run the VG, CA, and OCA, by means of extensive MATLAB/Simulink simulations, as shown in Figs.4,5 and 6 respectively. We have 3 robots, 3 goals, and 6 obstacles.

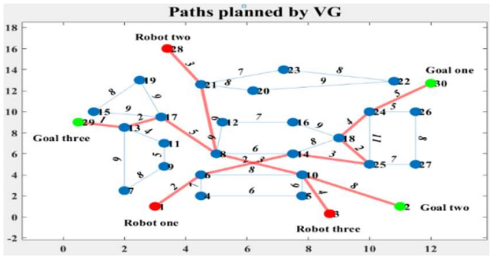

Fig.4. Paths planned by VG using MATLAB

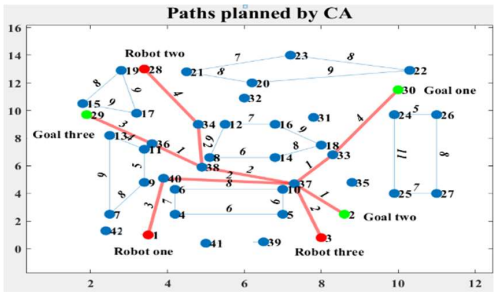

Fig.5.Paths planned by CA using MATLAB

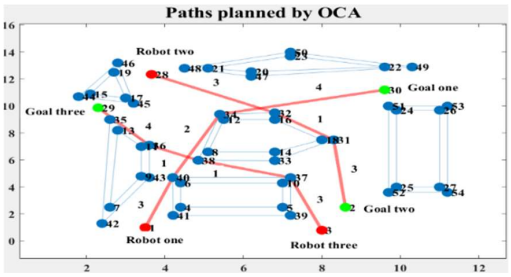

Fig.6. Paths planned by OCA using MATLAB

TABAL1. Comparison path planned in Fig.4,5, and 6

	Shortest Path for Robot$_1$	***Total distance***
VG	$R_1 \rightarrow v_6 \rightarrow v_{14} \rightarrow v_{25} \rightarrow v_{18} \rightarrow v_{24} \rightarrow v_{30}$	$\boldsymbol{P_1}$=2+2+3+2+4+5=18
CA	$R_1 \rightarrow v_{40} \rightarrow v_{37} \rightarrow v_{33} \rightarrow v_{30}$	$\boldsymbol{P_1}$=3+2+1+4=10
OCA	$R_1 \rightarrow v_{40} \rightarrow v_{34} \rightarrow v_{30}$	$\boldsymbol{P_1}$=2+3+4= 9

Dijkstra algorithm is used to find the optimal paths. The paths are evaluated based on the edge weight (w_{ij}), which calculates the Euclidian distance between successive waypoints in the path. During path planning, vertices' weights (w_i) change and equate to the moments of time at which the robot R_i passes through these vertices. To provide collision avoidance, w_{ij} can be modified during path planning, either by path correction, or through control robot's motion time by controlling the w_{ij} between vertices. Let $T_{R_n} = w_i + w_{ij}$, be the arrival time: (i.e., a time when robot $(R_n, n = 1,2,.m)$ passes through the vertex v_i). T_{R_n} depended on the w_{ij} between the edges in the graph. If $T_{R_m} > T_{R_n}$, the distance travelled by R_n is less than the distance travelled by R_m, hence, T_{R_n} is shorter than T_{R_m}. In Fig.5 , $T_{R_2} = w_{38} + w_{38,37} = 8 > T_{R_3} = w_{37} + w_{37,38} = 4$,so no collision happens.

We have simulated a scenario of a simple workspace to test CA, see Fig.7.

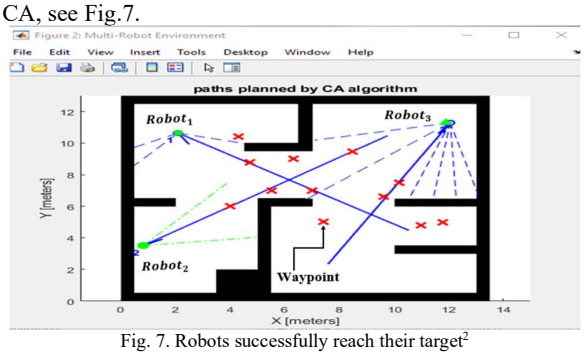

Fig. 7. Robots successfully reach their target[2]

I. Discussion and Conclusion

To sum up, we have developed the CA & OCA to address the disadvantages of the VG method. The paths planned by the VG are close to the obstacles, whereas the CA and OCA produce short collision-free paths whilst maintaining a safe distance from obstacles, which makes it safer. Both algorithms employ a smaller number of obstacles, and this reduces the computational complexity of roadmap approaches. The VG in Fig.4 considered six obstacles whilst CA and OCA in Fig.5 and 6 considered three obstacles. Thus, the calculation time to find the paths are less for computed paths. CA and OCA have made finding the shortest paths simpler because the process of path planning is equipped with pre-calculated step-by-step instructions. All these features make it more efficient than the VG. Simulation results and comparison with the VG confirmed that both the CA and OCA can find global optimal paths (short and safe) with computational efficiency.

References

[1] Omar, R. B. (2012). Path planning for unmanned aerial vehicles using visibility line-based methods (Doctoral dissertation, University of Leicester).

[2] Nilsson, N. J. (1984). Shakey the robot Technical note 323. Artificial Intelligence Center, SRI International, Menlo Park, CA.

[3] Elbanhawi, M., Simic, M., & Jazar, R. (2013). Autonomous robots path planning: An adaptive roadmap approach. In Applied Mechanics and Materials (Vol. 373, pp. 246-254). Trans Tech Publications Ltd.

[4] Roadmap Methods vs. Cell Decomposition in Robot Motion Planning.

[5] Toan, T. Q., Sorokin, A. A., & Trang, V. T. H. (2017, June). Using modification of visibility-graph in solving the problem of findishortest path for robot. In 2017 International Siberian Conference on Control and Communications (SIBCON) (pp. 1-6). IEEE.

[6] Omar, R., & Gu, D. W. (2009, August). Visibility line-based methods for UAV path planning. In 2009 ICCAS-SICE (pp. 3176-3181). IEEE.

[2] The environments presented in this figure are part of MATLAB 'exampleMaps', https://www.mathworks.com