Advances in Robotics & Automation
ISSN: 2168-9695

Kamkarian and Hexmoor, Adv Robot Autom 2015, 4:2
DOI: 10.4172/2168-9695.1000130

Research Article Open Access

An Analysis for a Novel Path Planning Method

Kamkarian P[1]* and Hexmoor H[2]

[1]Department of Electrical and Computer Engineering, Southern Illinois University, Carbondale, USA
[2]Department of Computer Science, Southern Illinois University, Carbondale, USA

Abstract

The aim of disseminating this research article is to introduce a novel path planner method in detail and argue that it performs as well as other offline path planners in terms of analyzing and constructing collision-free trajectories in addition to shortest possible path from start to goal configurations. Our novel path planner is able to build optimal trajectories in terms of the shortest length as well as near miss avoidance route from the initial to the goal configuration. It allows a moving point robot to make a proper transition from its assigned goal toward a collision-free path in the reasonable time frame, successfully. As an offline path planner, our planner performs the operation of analyzing the environment on static workspaces with the fixed and known initial and the goal configurations and computing optimal trajectories, using global information about the environment. As a feature of novelty, our planner benefits of using a limited amount of global knowledge, however. This helps moving robot to allocate less system resources such as memory which leads the robot to perform the process of building optimal trajectories more efficiently. The shortest route is considered to be secure enough to enable mobile robot to maneuver among obstacles in workspace without involving any near misses. Our novel path planner is able to process any types of obstacles in terms of shapes and edges flawlessly. For instance, it can be applied on any spiral or curved obstacles successfully. This novelty feature distinguishes our offline path planner from the majority of other planners that are solely able to compute the optimal trajectories for certain obstacle shapes such as polygonal obstacles. Moreover, this paper attempts to evaluate our novel path planner algorithm abilities and skills by examining it on selected workspaces. We also assess our novel path planner through comparing it with other planners to reveal its efficiency in terms of ability to route optimal trajectories in regards to minimizing trajectory distances from initial to the goal configurations as well as the reliability and safety of the optimal processed path.

Keywords: Path planning; Rapidly optimizing mapper; Robot trajectory builder; Robot safety path

Introduction

A robot is basically a collection of many different units, with the purpose of achieving desirable goals assigned by human users. Depending on the robot complexity, it is able to perform a broad range and variety of missions with minimizing control from a human user. Robots, as important entities, are going to find their place in different manmade constructions in both industrial and everyday lives, which soon, by their nature, are expected to become necessary tools to help facilitate performing tasks as well as elevating the accuracy rate of productions. Based on the specifications of the environment, along with the type of duty that a robot is supposed to perform, they are categorized in a variety types and in different groupings. Offline moving robots is a robot category that consists of many components such as movement unit, which rely on a form of mechanical system to enable them to move in workspaces, with the sole intention of reaching from an initial point to the goal configuration, successfully. An important computational unit that plays a vital role for a robot to service its targets is the path planner. Planner assignment is to analyze the surrounding world around the robot and plan a reliable trajectory in terms of enabling the robot to reach its goal through a reasonable path length as well as a secure route. In other terms, the planner's primitive objective is to route a collision-less path among obstacles in the environment. Since offline robots are emerged and employed, there are several path planners have proposed based on the variety of different aspects and considerations. A planner can be constructed based on adopting scientific rules and regulations or it can be the result of a combination of many approaches combined into a unique solution to plan a sufficient path. As a primitive offline path planner that is generated to suggest a collision-fewer trajectories for an offline moving robot, potential filed planner [1], is a prime example. The Potential Planner works based on emulating the electromagnetic facts among same and opposite electric charges. Two entities with the same charge, when close enough, develop repulsive force, which causes them to push each other away, whereas two objects with opposite charges; try to absorb one another through attracting forces. The more amounts of charge for close objects, the higher volume of force between them. Although adopting potential field planner can guarantee a safe path toward the goal in terms of collision-free trajectory; however, has its own constrains. Simulated obstacle repulsion charges, as an instance, can be close and powerful enough to cause the local minima problem [2,3]. Hence, it suffers from computing a proper path. The latter problem is resolved by further researches that are mainly focused on either optimizing the planner algorithm [4,5], or updating the general construction of the planner [6-8]. As for leading types of planners, evolutionary path planner algorithm [9], genetic algorithm [10,11] ant colony planner approach, [12,13] and particle swarm optimization planner [14], are remarkable. Using methods that randomly generate paths for offline robot is another popular idea that is employed by several researchers during the last few decades. For instance probabilistic roadmap method benefits from connecting randomly generated points in free spaces of the workspace

***Corresponding author:** Kamkarian P, Department of Electrical and Computer Engineering, Southern Illinois University, Carbondale, USA, Tel: +1618-453-2121; E-mail: pejman@siu.edu

Received July 15, 2015; **Accepted** July 27, 2015; **Published** July 29, 2015

Citation: Kamkarian P, Hexmoor H (2015) An Analysis for a Novel Path Planning Method. Adv Robot Autom 4: 130. doi: 10.4172/2168-9695.1000130

to generate a graph with the sole intention of analyzing paths from start to goal points and hence, route the desire trajectories. In other words, the basic PRM approach builds a roadmap consisting of collision-free samples and later, it tries to find the optimal path from the generated roadmap. The PRM method has developed in many different sub-approaches such as lazy-PRM [15], which works based on choosing a random path from the start to the goal configuration and examines it to assure that it is collision-less and optimized. It repeats the process of random path planning until reaches the best possible results. The PRM has developed by different researchers with some variations [16-21]. It is able to produce remarkable results in the form of optimal paths in different scenarios, in terms of efficiency of the method as well as collision-free trajectories [21-31]. Both PRM and RRT methods have an initial phase of selecting a set of free configuration points before a traversal graph is constructed. Our discussion of these techniques did not separately outline the exploration phase of each method. Instead, we treated the methods in entirety and performed comparison with our method on exactly the same environments and identical conditions. Adjusting the exploration phases does not change reported results. The primitive probabilistic roadmap method is subject to be improved within last decade due to having many constraints that were significantly reduced its performance to route optimal trajectories. For example, the way of electing decent pairs of collision-free nodes, the lacking of a suitable method for sampling workspace, and the confusion about using a certain local planner are samples of different issues involving the RPM that were consequently addressed and resolved by several researchers [15,26,28,30,32-36]. Employing proper path planner on environment with high complexity and constraints led researchers to consider proposing the next generation of path planners including a group of many single path planners in the form of a unique planner known as hybrid path planners [37-43]. A hybrid planner usually benefits from the advantages of more than a single planner strategy by averting the limitations of single style planners which leads to elevating the efficiency of the hybrid planner to overcome a wider range of environment complexities and variety of constraints with the purpose of being able to route optimal trajectories, flawlessly.

This paper aims to disseminate the efficiency of our planner called rapidly optimizing mapper (ROM) through comparing it with several other path planners, with the purpose of assuring our planner performance in planning reliable trajectories in regards to minimizing the length of the path as well as determining a collision-less trajectory which leads computing a secure path. In the next section, we illustrate the construction of our path planner as well as its key features and the related properties. Section 3 presents the evaluation of our path planner as well as other path planners by examining them on many scenarios. A final report will be concluded from comparisons and obtained results in section 4.

The Rom Theoritical Framework

Our planner fabrication is based on a multi-layer approach in the form of a unique algorithm that each layer uses data provided from the prior layer and is responsible to generate the needed information for the next level. In order to analyze the environment and achieve to the proper collision-free trajectory, our planner performs the following steps, in general.

Step 1: Analysing the workspace to find roadblock obstacles and then investigating over each roadblock to determine side edge nodes for traversal,

Step 2: Evaluating side edge nodes and performing simplification operations to achieve optimal graph for pathways in the form of a lattice of side edge nodes consisting of the start and goal configurations.

Step 3: Interpreting the optimal graph to determining the proper trajectory from initial to the goal configuration.

All concepts mentioned on the above steps will be explained in detail consequently. Initially, the planner benefits from the information about variables that have to be predetermined based on both environment and robot specifications along with the workspace constraint. We have considered the following features to be predetermined at the initial phase for the planner: standoff distance (SD), degree of traverse (DT), degree of surface traversal (DST), start and goal configurations. Surpassing almost all other offline path planners, our planner is able to build the optimal trajectory without having the scale and sizes of the workspace as well as the specifications of obstacles in the environment such as their number, sizes, shapes and locations. We are not limited to polyhedral shaped obstacles. Each robot, based on the vision equipment sensitivity, is able to detect objects and safely maneuver around them in a specific range. The Standoff distance is measured and determined based on the robot skills in analyzing, detecting, and the time needed to make proper decision and eventually reaction time as well as motion equipment strength to help robot to maneuver and adjusts its path. The SD is a scalar value and depending on the granularity size of the environment and it can be measured based on millimeters, centimeters, meters, etc. SD can be fixed at zero for the touch sensors which detect objects from surface sensing technology and higher for other types of vision sensors. Equation 1 captures this idea.

$$SD \subset \mathbb{N}, SD = \begin{cases} 0 & touch\,sensors \\ > 0 & otherwise \end{cases} \quad (1)$$

The planner considers the SD value as a pathway around obstacles and avoids building the trajectory through this boundary area. In other terms, the planner acknowledges each obstacles diameter based on the obstacle's real diameter plus the SD value as it is indicated in the following mathematical equation 2.

$$\forall w_{o_i} \in W_o,\ D_{wo_i} = D_{wo_i} + SD \quad (2)$$

In equation 2, w_{o_i} indicates the ith obstacle and D_{wo_i} represents its diameter.

At the beginning of the optimal path building operation, the planner examines the workspace with the purpose of recognizing roadblock obstacles. An obstacle categorizes as a Roadblock obstacle (RO), if it at least falls under one of the following two criteria.

1) Obstacles that have at least a single hit point (i.e., intersection) with the path crossing from start point to the goal configuration through a straight line.

2) Any hit point from the initial point, which is recognized as the latest position of the trajectory builder while the processing of the optimal trajectory with any obstacles in the workspace.

Generally, roadblock obstacles will be recognized through the following equation 3.

$$\forall n \subset IG, \forall w_{o_i} \subset W_O \quad if\ (n \cap \bigcup_{i=1}^{p} w_{o_i} \neq \phi) \Rightarrow w_{o_i} \subset RO \quad (3)$$

In the equation 3, n indicates an arbitrary locus points located on the straight line from Initial point toward the Goal configuration, $IG.\ RO$ also represents the roadblock group.

As the next step of analyzing and building the optimal trajectory, our planner considers roadblock obstacles as the candidates to route

trajectory. In order to analyze roadblock obstacles, our planner uses degree of traverse (DT), and degree of surface traversal (DST) concepts. The Degree of Traverse is defined based on angle in radians measurement which is used to scan the surface of the roadblock obstacle for the purpose of determining roadblock side edge nodes. Depending on the size and specifications of the workspace, the DT value takes different values. Side edge nodes are special locations on the workspace that our path planner routes the trajectory through them. A side edge node has the following specifications.

1) It belongs to obstacles and is located on a roadblock obstacle;

2) Leave (i.e., departure) points located on the roadblock obstacles are considered as side edge nodes;

3) The trajectory changes direction on side edge nodes.

Figure 1 shows a sample case of roadblock obstacle scanned through the process of building trajectory.

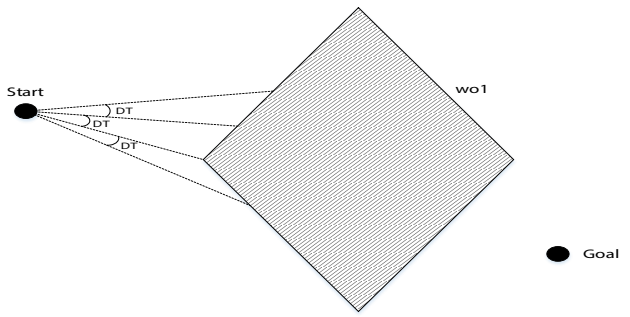

Figure 1: Sample roadblock obstacle surface scanning.

The process of analyzing roadblock obstacles falls under two different operations. Scanning through the DT process which is lately explained in detail, and routing the trajectory through DST. In case of scanning the surface of a roadblock obstacle, due to its specific shape, is impossible; the DST operation will be performed by the planner to handle the process of trajectory routing. For example, in the event oflocating either start or goal configuration located inside a spiral shape roadblock obstacle, the planner performs a DST operation to achieve the best results in building optimal trajectory. Figure 2 is a showcase of the mentioned situation:

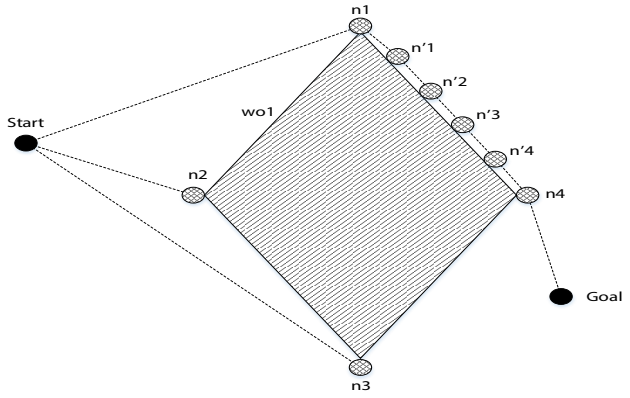

Figure 2: The situation of locating the goal configuration inside a spiral roadblock obstacle in a sample workspace.

Nodes n ′ 1 through n ′ 4 on the Figure 2 are indicating the resultant of surface scanning through the process of DST.

The procedure of analyzing the workspace and hence, achieving side edge nodes are outlined in the following algorithm 1.

Algorithm 1: The algorithm of the first stage of ROM planner

1) Draw virtual straight line from initial point to the goal configuration and collect obstacles that have at least one hit point with the mentioned line, in the roadblock obstacles;

2) Process roadblock obstacles to compute side edge nodes using initial variables for DT and DST;

3) If the goal configuration has met, END.

4) Jump to the step 1 and repeat the process for the latest located side edge nodes;

The second level of the trajectory construction procedure is to build and optimize a lattice of side edge nodes in the form of a graph consisting all possible pathways from start to goal configuration. In order to achieve the objectives of this level, our path planner performs the following steps:

Node visibility simplification

In order to achieve the optimal trajectory, the planner examines each side edge node with other adjacent nodes and pursues to find edge with maximum length possible of side edge nodes that are not intersecting with any obstacles through a straight line connecting them to each other. In order to interpret the mentioned strategy in math relations, we formulated it as the following equation 4:

$$\begin{aligned} &\forall\left((w_{ot}\in W_O)\wedge(initial\in W)\wedge\left((n_i,n_j)\in w_{ot}\right)\right),\\ &if\left((initial,n_j)\right)\cap\cup_{v=1}^{p}\, w_{o_v}=\phi)\,\&\,\&\left(E_{(initial,nj)}>E_{(initial,nj)}\right)\\ &then\ (n_i\in SN)\,\&\,\&\,(\mathrm{n}_j\notin SN) \end{aligned} \quad (4)$$

In the equation 4, initial, n_i and n_j are configurations on the workspace and obstacle respectively. *SN* indicates the group of side edge nodes. Figure 3 shows a sample of node visibilities in a workspace:

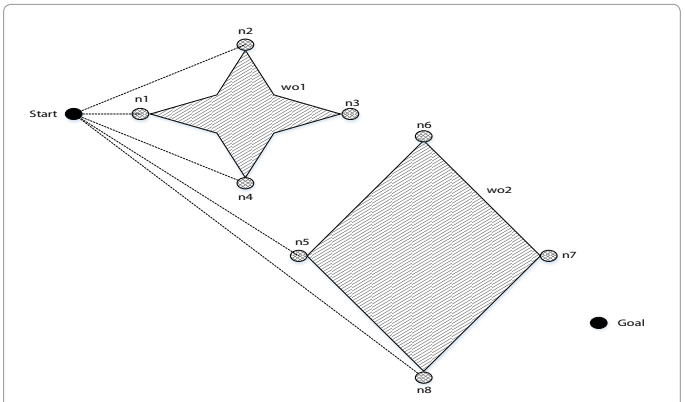

Figure 3: Node visibility samples in a workspace.

As it illustrated in the Figure 3, nodes n_1, n_2 and n_4 located on the obstacle 1 are accessible through start point. This is because there is no any intersections between (start, n_1), (start, n_2), and (start, n_4) edges and other obstacles in the workspace. In the sample figure, nodes n_5 and n_8 are also belong to obstacle 2 and are visible from start configuration. For the same reason as mentioned lately, the reason of considering (start,

n_5) and (start, n_8) edges as visible is that there is no any intersections between them and any obstacles presents in the environment. Our planner, therefore, removes edges (start, n_1) related to the first obstacle and also (start, n_5) from the second obstacle to achieve the optimal trajectory at the end of building trajectory. We have adapted visibility among obstacle side nodes only as a subcomponent inside our method. Visibility method for map generation is among of the earliest techniques. The original visibility based path planning was limited to vertices of polyhedral obstacles. It was never applied to obstacles that are not polyhedral. We are relying on use of visibility to complete a traversal graph but we can handle non-polyhedral obstacles. The evidence is application on spiral shaped obstacles.

Visibility pathway optimization

Our planner objective is to discover the shortest collision-free available trajectory from the start to the goal conjurations. To reach this target, at this level of analyzing and optimization process, it considers removing edges formed from a set of side edge nodes that belong to the same roadblock obstacles that share no intersections with other obstacles. To clarify this strategy, Figure 4 is shown.

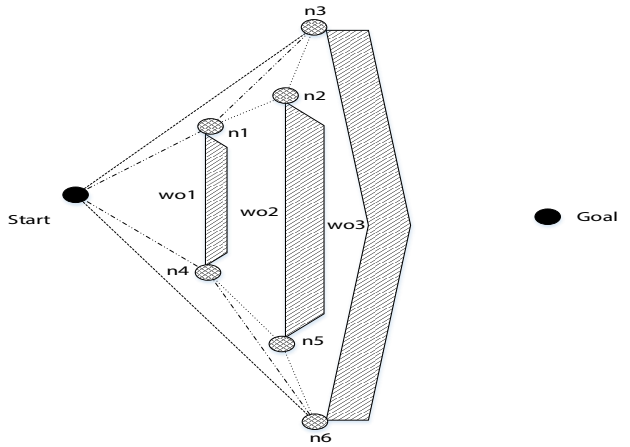

Figure 4: Visibility pathways in a sample roadblockobstacle located in a sample environment.

In Figure 4, the planner starts analyzing nodes 1 and 3. Both nodes 1 and 3 are visible from node 2 through straight lines without intersecting any other obstacles, so based on the visibility pathway strategy our planner keeps nodes 1 and 3 and removes node 2 from the group of side edge nodes. In other terms, the planner keeps nodes that are closer to the start point and removes the ones that have larger Euclidean distances from start point. The same process occurs for the nodes 4, 5 and 6. The second level of visibility pathways optimization is also applied for the nodes Start, n_1 and n_3. The result is to maintain nodes start and n_3 and removing node 2 from the final constructed graph by our planner. Node 5 will be also removed from the final graph because of applying the same strategy indicated above. The visibility pathways optimization can be expressed as the following criteria in equation 5:

$$\forall\Big(\big((n_i,n_j,n_k)\in SN\big)\wedge(initial\in W)\wedge(w_{o_v}\in W_O)\Big),$$
$$if\ \Big((n_i,n_j,n_k)\in w_{o_v}\Big)\&\&\Big((n_i\leftrightarrow n_j)\cup(n_i\leftrightarrow n_k)\cup(n_j\leftrightarrow n_k)\Big)\cup\cap \quad (5)$$
$$W_o=\phi)then\ \max_{initial} E_{(n_j,n_j,n_j)}\notin SN$$

Where *SN* indicates the set of side edge nodes and *E* illustrating the Euclidean distance.

Isolated nodes completion

In order to examine all possible pathways between side edge nodes, our planner at this level of obtaining the optimal graph, examines all remaining side edge nodes in sole purpose of recognizing nodes that are belong to same roadblock obstacles and are not connected to one another from either direction. In case of detecting any isolated nodes, they will be completed by considering a route that connects them through the missing direction. Figure 5 shows a sample of isolated nodes and the resultant of completions:

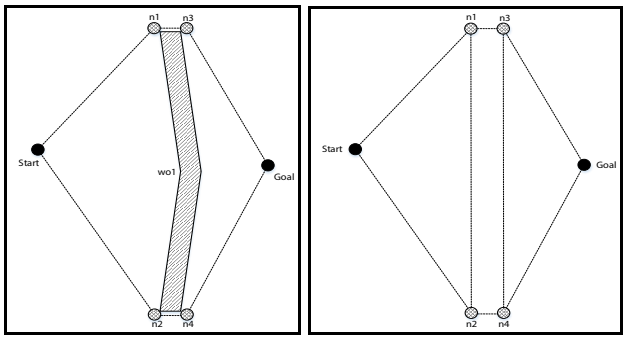

Figure 5: Left: A sample roadblock obstacle including a few isolated nodes. **Right:** The resultant graph from applying isolated node completion.

As it indicates in the Figure 5, left, nodes 1 through 4 are considered as isolated nodes. This is because connections from node n1 to the n_2 and also from node n_3 to the n_4 are missing. In the Figure 5, right, the resultant graph from recognizing and completion the detected isolated nodes are revealing. In other terms, our planner has completed nodes n_1 through n_4 by considering connections from node n1 to the node n_2 and also from node n3 to the node n_4, as indicated in the Figure 5, right.

The following algorithm explained different operations related to the second step of refining optimal trajectory performs via our path planner:

Algorithm 2: The algorithm of the second stage of ROM planner

1) Build the primitive graph including all side edge nodes as well as start and goal configurations;

2) Optimizing the graph by performing node visibility simplification process;

3) Conduct further optimization through applying visibility pathways optimization operation;

4) Obtain the final graph by detecting and completion isolated nodes.

Our path planner, as the final step to refine the ultimate trajectory, performs a search on the resultant graph including remaining side edge nodes as well as the start and goal configuration along with all possible routes connecting them to one another, with the purpose of finding the shortest available path from start to the goal point. In order compute the proper trajectory, we benefited using Dijkstra's shortest path finder algorithm for this level of our planner. The resultant path obtained from applying Dijkstra's shortest path algorithm will be recognized as the ultimate trajectory which brings the robot a collision-free path from start to the goal configuration. In the following section, we evaluate the performance of our path planner through applying it on two different scenarios in form of workspaces along with comparing the results

obtained from our planner with similar results from applying other path planner candidates.

Experiments and Evaluation of ROM

The process of evaluation for our path planner ROM consists of comparing it with three offline path planners. In order to perform the comparison, we applied our planner along with other path planners to sample, nontrivial environments. We have employed the following path planners for the assessment purposes: A* path planner algorithm, bidirectional rapidly-exploring random tree approach, and probabilistic roadmap path planner. The following figures are sample workspaces that we used to apply our path planner along with other candidate path planners in sole purpose of evaluating performances:

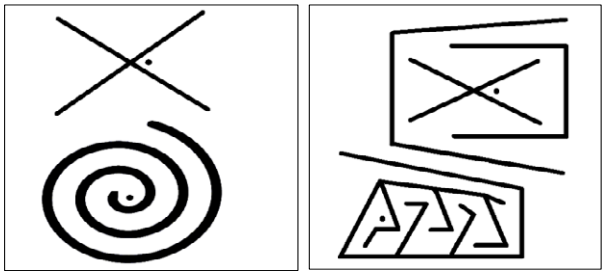

Figure 6: Left: The first sample environment map. **Right:** the second workspace showcase.

As it shown in the Figure 6, we used various obstacles with different specifications such as shape, sizes and locations. Each path planner, therefore, has to use a wide range of attempts to analyze and hence, building optimal trajectories in terms of achieving collision-free paths from start point to the goal configurations. The start point is located at (244, 365) and the goal configuration is at (295, 108) coordinates of the two dimensional space. We have categorized the process of applying each path planner candidate on the sample workspaces in separate case studies. At the end of each case study, we compared the performances of our path planner with the case study path planner. A final conclusion in form of a unique chart consisting of the results from our planner along with all other path planner candidates will be provided consequently. It helps to reveal the ultimate path planner among the group of all path planners used in this research article. Figure 7 illustrate the results of applying our path planner on sample workspaces:

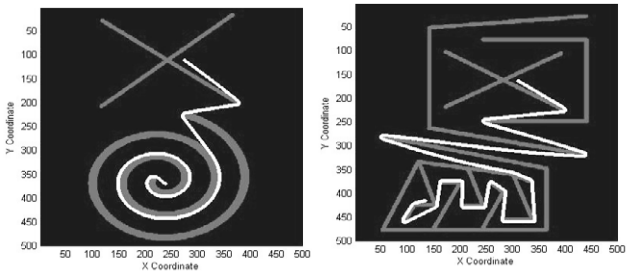

Figure 7: Left: The resultant trajectory from applying our path planner on the first workspace. **Right:** The trajectory constructed from applying our path planner on the second sample of workspace.

As it illustrate in the Figure 7, the planner was able to build a collision-free trajectory from start to goal configuration successfully. It also determined the Euclidean distance from start point to the goal configuration equaling 1983 for the first workspace, (left), and 2148 for the second environment, (right). Because of the workspace constraints and specifications, the planner used various strategies such as roadblock surface scanning while building the optimal trajectory.

The process of applying path planner candidates on the sample workspaces are illustrated as the following case studies:

Case study 1

Throughout this case study, we applied the A* path planner algorithm on the sample workspaces and will compare the resultant trajectories with our path planner consequently. The following figures indicate the resultant trajectories (Figure 8).

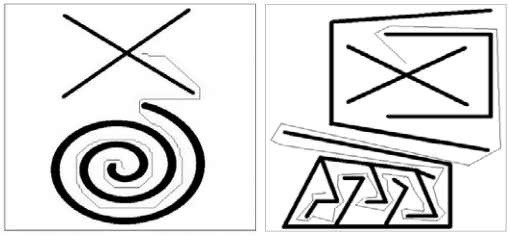

Figure 8: Left: The trajectory built from applying *A**path planner algorithm on the first workspace. **Right:** The collision-free path processes from applying the *A**path planner algorithm on the second sample environmental map.

For clarification purposes, we have collected the Euclidean distance results obtained from applying our path planner along with the results attained from A* path planner algorithm on two sample workspaces in one chart illustrated as Figure 9.

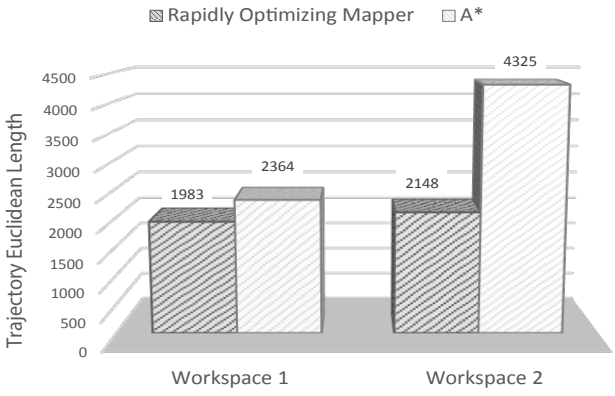

Figure 9: Euclidean distances chart collected from the results obtained from applying our path planner along with the *A**path planner algorithm.

Comparing the results achieved from both our planner and A* path planner reveals that our planner was able to build the collision-free trajectory in shorter path which indicates a higher performance for our planner. As it shown in the Figure 9, the shortest collision-less trajectory obtained from A* planner for the first map is equal 2364 and 4325 for the second workspace.

Case study 2

This case study is dedicated to study on the bidirectional rapidly-exploring random tree path planner. We will apply this path planner on

two sample workspaces and compare it with our planner consequently. The following figure indicates the conclusions gathered from applying bidirectional rapidly-exploring random tree path planner on the environments (Figure 10).

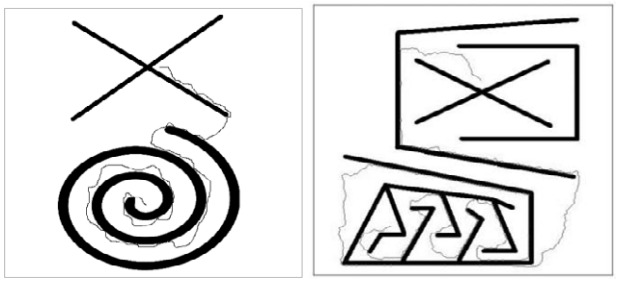

Figure 10: Left: The collision-free path built from applying thebidirectional rapidly-exploring random tree path planner on the first workspace. **Right:** The optimal path obtained from bidirectional rapidly-exploring random tree path planner on the second environment.

The following figure reflects the distances from start to the goal configurations in the Euclidean measurements obtained from bidirectional rapidly-exploring random tree path planner along with our path planner in a same chart.

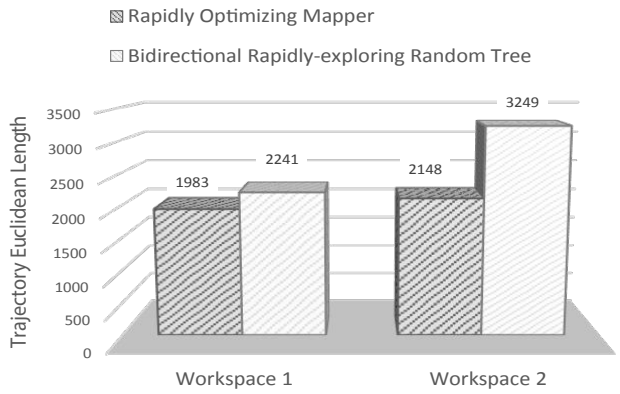

Figure 11: Euclidean distances from start point into the goal configurations gained from applying bidirectional rapidly-exploring random tree path planner along with our pathplanner on the sample workspaces.

As it indicates in the Figure 11, our path planner is able to build a collision-free trajectory more efficient in terms of the length of the path. The bidirectional rapidly-exploring random tree planner computed the Euclidean length for the first workspace as 2241 and 3249 for the second sample environment.

Case study 3

The aim of this case study is to apply the probabilistic roadmap planner on the sample workspaces and compare results with our path planner in sole purpose of evaluating performances. We also considered 1500 sample points to be located randomly on the free spaces of the environment. The following figure explains the results obtained from applying the probabilistic roadmap planner on the sample environments (Figure 12).

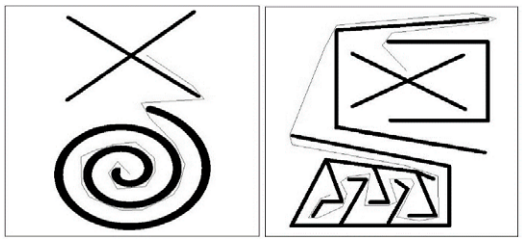

Figure 12: Left: The resultant trajectory obtained from applying probabilistic roadmapplanner on the first workspace. **Right:** The collision-less path calculated from probabilistic roadmap planner operation on the secondsample environment.

Figure 13 shows results in Euclidean distances achieved from applying probabilistic roadmap planner along with our path planner on the sample workspaces in one chart.

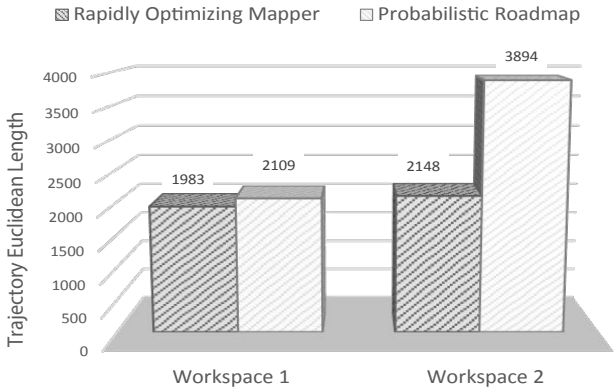

Figure 13: Euclidean chart including distances from start to the goal configuration calculated from probabilistic roadmap planner and our path planner operation on the sample environments.

The chart shown in Figure 13 indicates that the probabilistic roadmap planner calculated Euclidean lengths for the trajectories belong to the first and the second sample workspaces for 2109 and 3894 relatively which reveals a lower efficiency compare to our path planner.

In order to determine the best path planner among the group of our planner along with other planner candidates, we have collected all results in one chart including all distances from start point to the goal configurations in Euclidean measurement as illustrated in Figure 14.

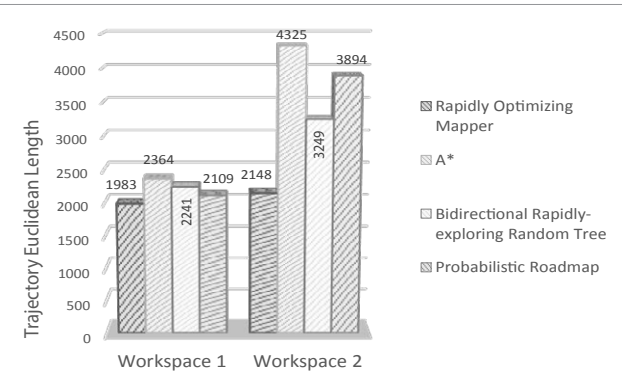

Figure 14: Resultant Euclidean distances from start to the goal configuration collected from applying our path planner along with other candidates on the sample workspaces.

The chart illustrated in Figure 14 reveals that our planner is able to build a collision-less trajectory in a higher efficient path in terms of the route length. The Euclidean distances computed for the sample workspaces by ROM are 1983 and 2148 which is the smallest lengths comparing other path planner candidates. The highest Euclidean lengths belong to the A* path planner with the values of 2364 and 4325 in Euclidean measurements which causes this planner to maintain the least performance to build collision-less trajectory among the group of planner candidates.

Conclusions and Perspectives

A novel path planner dubbed ROM is introduced in this research article. We demonstrated ROM as a solution for a holonomic offline point robot in a static environment to pursue its goal toward the shortest collision-free trajectories. ROM is able to be applied on many different workspaces having various arrangements and constraints successfully. Moreover, ROM is able to perform the process of optimal path construction more efficiently which is the resultant of processing a limited based of global knowledge. In other terms, in order to achieve a better performance and preserve more system resources such as memory, ROM, instead of processing all obstacles in the workspace, solely focuses on roadblock obstacles. The workspace including start and goal configurations are considered to be fixed during the goal pursuant from offline robot. The optimal trajectory refers to the shortest possible path from start point to the goal configuration which allows the robot to perform a near miss avoidance transition toward its assigned goal flawlessly. In order to clarify ROM's performance, we compared our planner with other path planners through applying them on selected scenarios. Our planner was able to compute optimal trajectories. This is because we adopted multiple techniques to furnish our planner algorithm which elevates its ability to operate on a variety of different workspaces with different limitations. Each technique, along with the proper parameters helps the ROM planner to achieve optimal results and also allows the planner to handle a verity of different workspaces with various constraints successfully.

References

1. Khatib O (1985) Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. IEEE International Conference on Robotics and Automation St Louis Missouri 2: 500-505.
2. Borenstein J, Koren Y (1991) The Vector Field Histogram-fast Obstacle Avoidance for Mobile Robots. IEEE Transactions on Robotics and Automation 7: 278-288.
3. Grefenstette J, Schultz AC (1994) An Evolutionary Approach to Learning in Robots. Proceedings of the Machine Learning Workshop on Robot Learning International Conference on Robot Learning. New Brunswick, NJ, pp. 65-72.
4. Arámbula Cosío F, Padilla Castañeda MA (2004) Autonomous Robot Navigation using Adaptive Potential Fields. Mathematical and Computer Modelling 40: 1141-1156.
5. Ge SS, Cui YJ (2000) New Potential Functions for Mobile Robot Path Planning. IEEE Transactions on Robotics and Automation 6: 615-620.
6. Connolly CI, Burns JB, Weiss R (1990) Path Planning using Laplace's Equation. Proceedings of the IEEE Conference On Robotics and Automation Cincinnati OH 3: 2102-2106.
7. Guldner J, Utkin V, Hashimoto H (1997) Robot Obstacle Avoidance in N-Dimensional Space using Planar Harmonic Artificial Potential Fields. Journal of Dynamic Systems Measurement and Control 119: 160-166.
8. Utkin VI, Drakunov S, Hashimoto H, Harashima F (1991) Robot Path Obstacle Avoidance Control via Sliding Mode Approach. Proceedings of the IEEE/RSJ International Workshop on Intelligent Robots and Systems Osaka Japan 3: 1287-1290.
9. Hocaoglu C, Sanderson AC (2001) Planning multiple paths with evolutionary speciation. IEEE Transactions on Evolutionary Computation 5: 169-191.
10. Li Q, Zhang W, Yin Y, Wang Z, Liu G (2006) An improved genetic algorithm of optimum path planning for mobile robots. Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications 2: 637-642.
11. Tu J, Yang SX (2003) Genetic algorithm based path planning for a mobile robot. Proceedings of IEEE International conference on Robotics and Automation 1: 1221-1226.
12. Tan G-Z, He H, Sloman A (2007) Ant colony system algorithm for real-time globally optimal path planning of mobile robots. Acta Automatica Sinica 33: 279-285.
13. Tewolde Girma S, Sheng W (2008) Robot path integration in manufacturing processes: Genetic algorithm versus ant colony optimization. IEEE Transactions on Systems Man and Cybernetics 38: 278-287.
14. Lei K, Qiu Y, He Y (2006) A novel path planning for mobile robots using modified particle swarm optimizer. 1st International Symposium on Systems and Control in Aerospace and Astronautics 3: 981-984.
15. Bohlin R, Kavraki LE (2000) Path planning using lazy PRM. Proceedings of the IEEE International Conference on Robotics and Automation 1: 521-528.
16. Amato N, Wu YA (1996) Randomized roadmap method for path and manipulation planning. Proceedings of IEEE International Conference on Robotics and Automation.
17. Barraquand J, Kavraki L, Latombe J-C, Li TY, Motwani R, et al. (1997) A Random Sampling Scheme for Path Planning. International Journal of Robotics Research 16: 759-774.
18. Kavraki L (1995) Random networks in configuration space for fast path planning. PhD thesis, Stanford University.
19. Kavraki L, Latombe JC (1994) Randomized preprocessing of configuration space for fast path planning. Proceedings of the IEEE International Conference on Robotics and Automation 3: 2138-2145.
20. Overmars MH (1992) A random approach to motion planning. Technical Report RUU-CS-92-32 Department of Computer Science Utrecht University Utrecht the Netherlands.
21. Svestka P (1997) Robot motion planning using probabilistic roadmaps. PhD thesis, Utrecht University.
22. Cortes J, Simeon T, Laumond JP (2002) A random loop generator for planning the motions of closed kinematic chains using PRM methods. Proceedings of the IEEE International Conference on Robotics and Automation 2: 2141-2146.
23. Han L, Amato N (2000) A kinematics-based probabilistic roadmap method for closed chain systems. Proceedings of the Workshop on Algorithmic Foundations of Robotics.
24. Holleman C, Kavraki L, Warren J (1998) Planning paths for a exible surface patch. Proceedings of the IEEE International Conference on Robotics and Automation 1: 21-26.
25. Lamiraux F, Kavraki LE (2001) Planning paths for elastic objects under manipulation constraints. International Journal of Robotics Research 20: 188-208.
26. Kavraki L, Svestka P, Latombe JC, Overmars MH (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions On Robotics and Automation 12: 566-580.
27. Sanchez G, Latombe JC (2003) A single-query bi-directional probabilistic roadmap planner with lazy collision checking. International Journal of Robotics Research 6: 403-417.
28. Sekhavat S, Svestka P, Laumond JP, Overmars MH (1998) Multilevel path planning for nonholonomic robots using semiholonomic subsystems. International Journal of Robotics Research 17: 840-857.
29. Simeon T, Cortes J, Sahbani A Laumond JP (2002) A manipulation planner for pick and place operations under continuous grasps and placements. Proceedings of the IEEE International Conference on Robotics and Automation 2: 2022-2027.
30. Svestka P, Overmars MH (1997) Motion planning for carlike robots a probabilistic learning approach. International Journal of Robotics Research 16: 119-143.
31. Svestka P, Overmars MH (1998) Coordinated path planning for multiple robots.

Robotics and Autonomous Systems 23: 125-152.

32. Amato N, Bayazit O, Dale L, Jones C, Vallejo D (1998) OBPRM: An obstacle-based PRM for 3D workspaces In Proceedings of the third workshop on the algorithmic foundations of robotics on Robotics.

33. Boor V, Overmars MH, Van der Stappen AF (1999) The Gaussian sampling strategy for probabilistic roadmap planners. Proceedings of the IEEE International Conference on Robotics and Automation 2: 1018-1023.

34. Hsu D, Kavraki L, Latombe JC, Motwani R, Sorkin S (1998) On Finding narrow passages with probabilistic roadmap planners. Proceedings of the third workshop on algorithmic foundations of robotics.

35. Nissoux C, Simeon T, Laumond JP (1999) Visibility based probabilistic roadmaps. Proceedings of the IEEE International Conference on Intelligent Robots and Systems 3: 1316-1321.

36. Wilmarth SA, Amato NM, Stiller PF (1999) MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. Proceedings of the IEEE International Conference on Robotics and Automation 2: 1024-1031.

37. Abinaya S, Hemanth KV, Srinivasa KP, Tamilselvi D, Mercy SS (2014) Hybrid Genetic Algorithm Approach for Mobile Robot Path Planning. Journal of Advances in Natural and Applied Sciences 8: 41-47.

38. Bashra K, Roth O, Roth H (2014) Modified Genetic Algorithm Based on A* Algorithm of Multi Objective Optimizationfor Path Planning. Journal of Automation and Control Engineering 2: 357-362.

39. Chaari I, Koubaa A, Trigui S, Bennaceur H, Ammar A, et al. (2014) An Efficient Hybrid ACO-GA Algorithm for Solving the Global Path Planning Problem of Mobile Robots. International Jounal of Advanced Robotics Systems: 1-15.

40. Ju M, Wang S, Guo J (2014) Path Planning using a Hybrid Evolutionary Algorithm Based on Tree Structure Encoding. The Scientific World Journal.

41. Loo CK, Rajeswari M, Wong EK, Rao MVC (2004) Mobile Robot Path Planning using Hybrid Genetic Algorithm and Traversability Vectors Method. Intelligent Automation and Soft Computing 10: 51-64.

42. McFetridge L, Ibrahim MY (1998) New Technique of Mobile Robot Navigation using a Hybrid Adaptive Fuzzy-Potential Field Approach. Computers and Industrial Engineering 35: 471-474.

43. Yao Z, Ren Z (2014) Path Planning for Coalmine Rescue Robot based on Hybrid Adaptive Artificial Fish Swarm Algorithm. International Journal of Control and Automation 7: 1-12.