IAES International Journal of Robotics and Automation (IJRA)
Vol. 13, No. 1, March 2024, pp. 105~112
ISSN: 2722-2586, DOI: 10.11591/ijra.v13i1.pp105-112

A 2D path-planning performance comparison of RRT and RRT* for unmanned ground vehicle

Shokufeh Davarzani, Muhammad Talha Ejaz
Department of Earth and Space Sciences, Columbus State University, Georgia, United State

Article Info

Article history:

Received Sep 30, 2023
Revised Jan 12, 2024
Accepted Jan 18, 2024

Keywords:

Localization
Mobile robots
Optimal path
Path planning
RRT
RRT*
Unmanned ground vehicles

ABSTRACT

In recent years, path-planning has gained significant attention as mobile robots are used in various applications. Several factors determine the optimal path for a mobile robot, including accuracy, length of path, execution time, and turns. Among all planners, sampling-based planners such as rapidly exploring random trees (RRT) and rapidly exploring random trees-star (RRT*) are extensively used for mobile robots. The aim of this paper is the review and performance of these planners in terms of step size, execution time, and path length. All planners are implemented on the Jackal robot in a static environment cluttered with obstacles. Performance comparisons have shown that the reduction of step size results in exploring a greater number of nodes in both algorithms, increasing the probability of each extension succeeding. However, this causes the tree to become denser in both algorithms due to the more explored nodes. The RRT planner requires less execution time when the step size and iteration count are equal to RRT* planners. Moreover, performance plots of both algorithms show that RRT* provides an optimal and smooth path than RRT.

Corresponding Author:

Shokoufeh Davarzani
Department of Earth and Space Sciences, Columbus State University
Georgia, United State
Email: davarzani_shokoufeh@students.columbusstate.edu

1. INTRODUCTION

The advent of mobile robots and their application to diverse fields, such as industry, services, medical care, and agriculture, has made robot path planning a significant challenge [1]–[3] . The goal of path planning for robots is to find the best path between a given start point and a desired goal point based on certain evaluation criteria, including path length, planning time, and so on.

There are various planners for mobile robots, which can be divided into two main categories, namely, graph-based and sampling-based algorithms [4], [5]. In graph-based search methods, obstacles are defined as inaccessible grid points in the state space. Based on the available grid points, the algorithm constructs a path from the start point to the goal point if possible. The sampling-based method selects several number of points in the configuration space and establishes connections between them [6]. These methods are straightforward and efficient and ensure a solution in an infinite time. Among the sampling-based methods, rapidly exploring random trees (RRT) has been extensively applied in different applications [7]. It was introduced in 1998 as a solution to resolve the path planning problem for robots in complex environments with high dimensions [8]. In fact, this algorithm does not require any model of the space, and it performs searches at a fast rate. However, due to the random sampling of space by RRT, the generated paths often are not optimal or suboptimal [9]. In order to tackle this limitation, a variation of RRT named rapidly exploring random trees-star (RRT*) was proposed in [10]. In fact, an asymptotic optimal path is established by using tree rewiring and best neighbor search. Path planning in robotics has evolved over the past decades.

It is still active research that involves finding a collision-free path and continuous path for robotics to move from the initial state to the target state while satisfying multiple constraints such as velocity and acceleration [11], [12] .

RRT utilize the Voronoi bias strategy based on the Voronoi diagram. It is a geometric structure that makes region space based on the proximity sets of points from partition space. Voronoi is a powerful technique that is used to guide its expansion towards unexplored areas with a higher probability of exploration proportional to their proximity to the robot's current position [13]. The algorithm employs a tree data structure that is easier to implement than a graph-based method. RRT explores locally first, which means that if the starting and goal positions are nearby, it can quickly reach the goal position, making it faster than probabilistic roadmap (PRM) [14]. PRM utilizes a graph-based approach where it constructs a graph in configuration space and creates sample points in the space. Once the roadmap is constructed by connecting all the points, the graph search algorithm is used to create a feasible path from starting point to the goal points in the space without any collision of an obstacle. PRM is a global algorithm that performs well in a highly complex environment with multiple obstacles. It is highly computational as compared to RRT. Once the graph is created, it can be used repeatedly with different start and goal points without constructing a new graph. RRT* is also a sampling-based algorithm that constructs trees and constructs a feasible path by random sampling. Despite RRT, RRT* adapts two optimization procedures: choose parent and rewire [15]. RRT* connects to the nearest node just like RRT but then rewires it for optimization by checking nearby nodes which can be reached more efficiently. The rewiring radius is the user-defined parameter according to robot properties, and the environment beyond the nearby node is not considered for rewiring. A path that has the lowest cost will be added to the tree and updated the tree at each iteration.

This paper presents a comprehensive analysis of the 2D path-planning performance of RRT and its enhanced version, RRT*, for Jackal unmanned ground vehicles (UGV) from ClearPath robotics. In fact, this study aims to address the critical importance of efficient path planning within the domain of autonomous navigation, providing insight into the challenges and proposing a comparative study between the traditional RRT and the more sophisticated RRT* algorithm with considering different parameters. The main contribution of this paper is implementing both these algorithms on Jackal UGV, which serves as a practical application of the theoretical concepts explored. This hands-on implementation not only validates the effectiveness of the proposed algorithms in a real-world scenario but also provides valuable insights into their performance under practical constraints and environmental considerations. The paper is organized into the following parts. In section 2, we provide a concise review of the definitions related to the general optimal path problem and the methodologies employed in this study. These methods are discussed concerning the effects of different parameters. Section 3 outlines the implementation scheme and presents the results obtained from the real-time experiment. Lastly, section 4 encompasses the conclusions and a thorough discussion of the statistical analysis.

2. METHODOLOGY

2.1. Rapidly exploring random trees (RRT)

RRT relies on building a tree structure containing the start and goal points as its roots and leaves. Random extension trees are created by sampling from the environment, adding the nearest leaf nodes to an initial root point, and moving toward that point with an incremental distance. It should be noted that adding the point as a leaf node in the tree is possible if the movement does not encounter any obstacles. Moving across the tree until the root node is reached allows you to create the path from the goal to the start point. Due to the fact that RRT does not require modeling of space and its fast search speed, it can be considered an appropriate path-planning method, especially in high-dimensional and complex environments [16], [17]. However, because of its randomness, it has only a single probability complete, resulting in several limitations [18], including instability, non-optimal path, and low convergence rate. As a general rule, RRT involves the following steps:

$RRT_{Path}(q_{init}, j, \Delta t)$ (1)
1. $Time_{init}\ (q_{init})$;
2. $while\ j\ =\ 1\ <\ N$
3. $q_{rand} \leftarrow RANDOM()$; $j \leftarrow j + 1$;
4. $q_{near} \leftarrow NEAREST\ (q_{rand}, Time\ )$;
5. $K \leftarrow NEW\_NODE\ (q_{rand}, q_{near})$
6. $q_{near} \leftarrow NEW\_CONFIG\ (q_{near}, K, \Delta t)$
7. $Time.add\_vertex\ (q_{new})$;
8. $Time.add\_edge\ (q_{new}, q_{near}, K)$

9. $q_{goal} \leftarrow If\ q_{goal} reached, end.$
$Return\ T$;

This algorithm takes the initial state, the goal state (region), and the environment as inputs and produces a feasible path as an output. As each iteration proceeds, a sample q_{rand} is generated, and the closest vertex q_{new} is determined based on distance. Then q_{rand} calculates all the distance measured in the tree between is set. A new node $q_{new}$is created between $q_{nearest}$ and q_{rand}, and the Euclidean distance from $q_{neares_t}$and q_{new} is calculated. In the absence of obstacles, a new node will be added to expand the tree. Otherwise, it will repeat the process of generating a brand until a feasible path is found. An illustration of the random tree expansion process can be found in Figure 1.

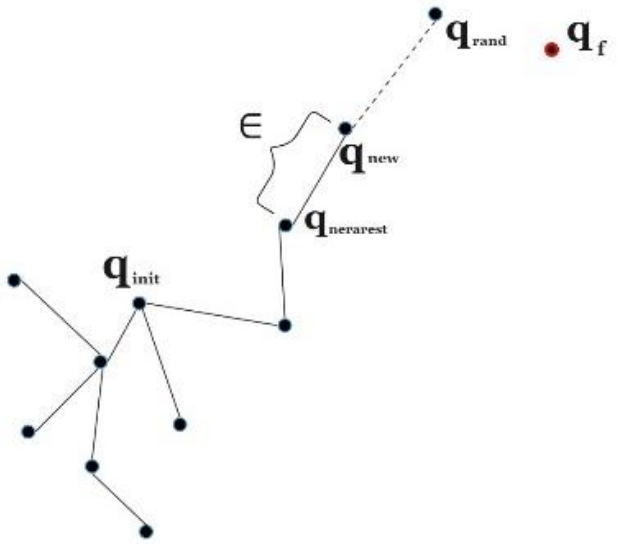
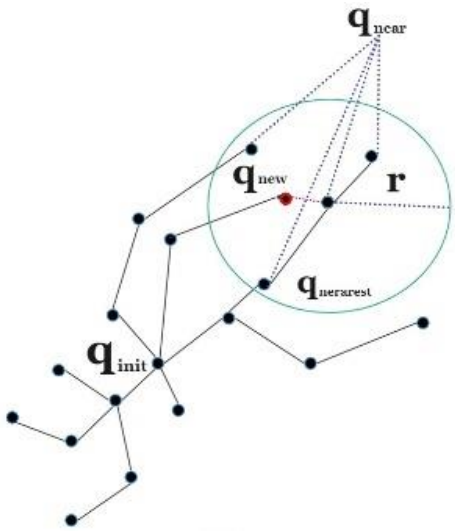


Figure 1. Illustration of RRT algorithm

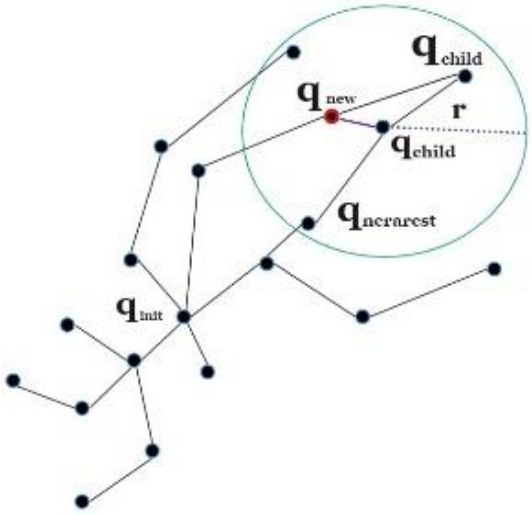
2.2. RRT*

RRT*, as an enhanced version of RRT has a similar function to RRT, but it employs two optimization functions, namely near neighbor search and rewiring the tree [19], [20] . In fact, the near neighbor function determines the most appropriate parent node for the newly created node prior to insertion in the tree, and the rewiring operation restructures the tree so that minimal cost is required between tree connections. The following steps are involved in RRT* and illustrated in Figure 2. RRT* sets the q_{neares_t}, which is closer to $q_{rand}$and q_{new}creates with the Euclidean distance away. Then $q_{new}$is generated, which is also known as the best parent process, as shown in Figure 2(a). The nodes in a certain radius around q_{new}that do not collide with an obstacle are created as q_{near}. If path costs between $q_{new}$and $q_{near}$are minimal as compared to previously connected to q_{neares_t}, it disconnects the node from $q_{neares_t}$and sets a new node with a lower path cost as the parent node. This process is also known as rewiring, as shown in Figure 2(b).

$RRT\ Star_{Path}(q_{int}, j, \Delta t)$ (2)
1. $Time_{init}$ (q_{init})
2. while $j = 1 < N$
3. q_{rand}← RANDOM( ); $j \leftarrow j + 1$;
4. q_{near}← NEAREST (q_{rand}, Time);
5 . K ← NEW-NODE (q_{rand}, q_{near})
6. q_{near}← NEW_W-CONFIG (q_{near}, K, Δt);
7. q_{goal}← If q_{goal}reached, then
8. $Time.add_{vertex}(q_{new})$;
9. $Time.add_{edge}(q_{new}, q_{near}, K)$;
10. G ← REWIRE (G, q_{new}, q_{near});
end if
Return T;



(a)



(b)

Figure 2. Illustration of RRT* algorithm (a) the best parent selection process (b) rewiring process

3. RESULTS AND DISCUSSION

The performance of algorithms is compared based on the several properties, including step size, iteration number, and execution time. The algorithms are simulated using 64-bit MATLAB version 15.

3.1. Effect of step size on RRT and RRT* performance

The performance of algorithms is compared based on several properties, including step size, iteration number, and execution time. The algorithms are simulated using 64-bit MATLAB version 15. To determine how far a tree can be extended on each step, RRT and RRT* have a parameter called step size. In other words, every edge in a tree cannot extend beyond the maximum value considered for the step size. In addition, if the two extended milestones are within this range, they can be connected. As shown in Figure 3, the reduction of step size results in exploring a more significant number of nodes in both algorithms, increasing the probability of each extension succeeding. However, this causes the tree to become denser due to the more explored nodes and exploration of free space becomes slower. It should be noted that with the same step size, RRT* is less dense compared to the RRT algorithm. This is because RRT* employs two parameters, namely the Euclidean distance and cost function. In every iteration, a new random node is generated, and within a certain radius, it chooses that node that has fewer cost values than the parent node and rewires it again. By considering both the Euclidean distance and the cost function, RRT* is able to create a less dense graph and find more optimal paths.

3.2. Effect of execution time on RRT and RRT* performance

The results of one sample run with 3,000 iterations and equal step size are depicted in Figure 3. Results show that RRT* improves the initial path compared to RRT remarkably. However, the execution time is longer compared to the RRT. This is because RRT* employs additional operations than RRT, such as rewiring trees and best neighbor search. These two features improve the path cost to generate less jaggy and shorter paths. On the other hand, these features also slow down the convergence rate and increase computational time.

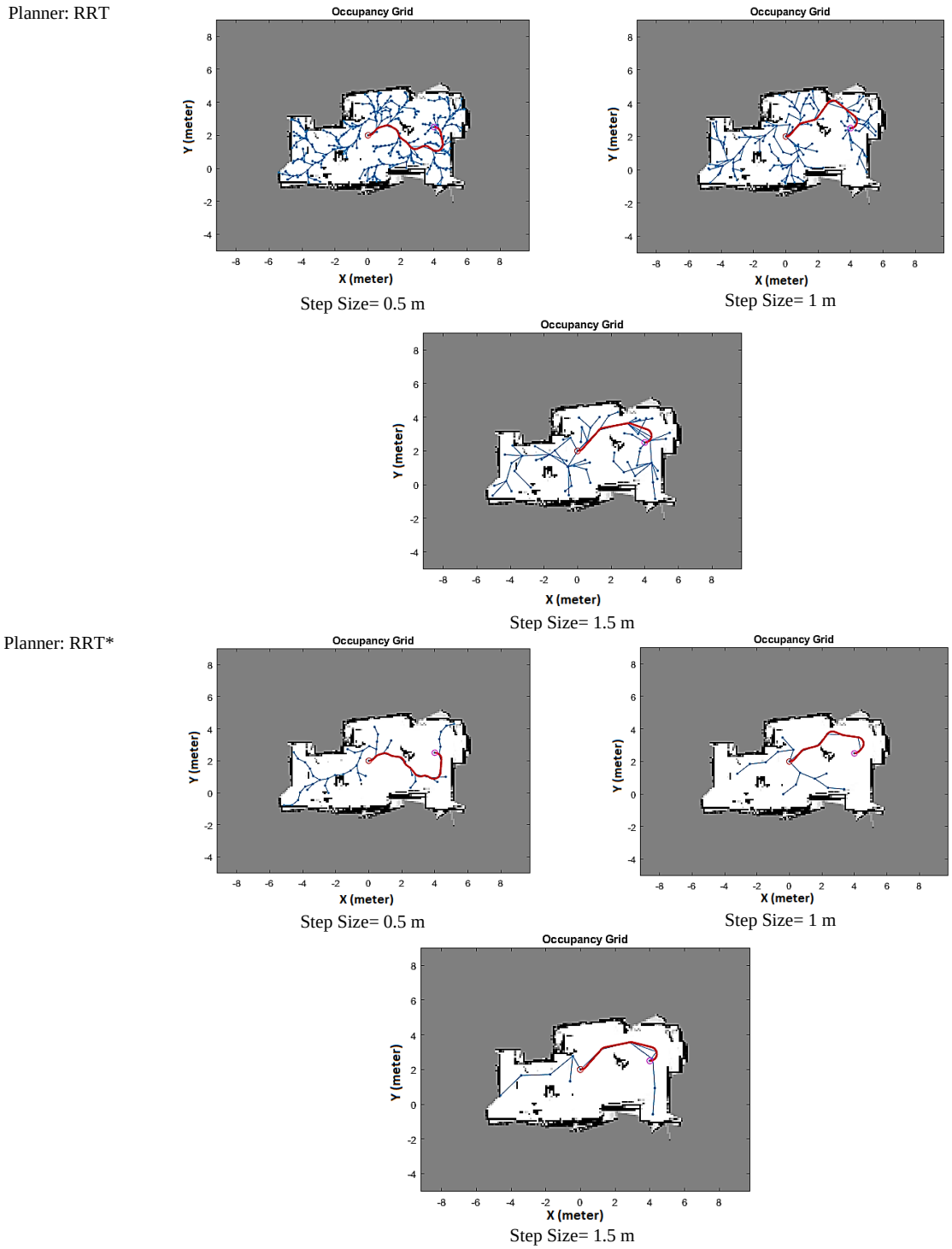


Figure 3. Performance comparison of RTT and RRT* algorithms

4. EXPERIMENTAL SETUP

This experiment is conducted utilizing the Jackal UGV from ClearPath Robotics, and a Jetson TX2 is used to control the platform. In order to acquire the 3D point cloud data, a Velodyne VLP-16 LiDAR sensor is used as the primary sensor, see in Figure 4. This LiDAR includes 16 lasers with a 30-degree vertical visual field and a 360° horizontal visual field. Both algorithms are implemented using 64-bit MATLAB version 15. The hardware and software architectures are described in Figure 5. This experiment involved creating maps from the environment, setting waypoints and goal points, and driving along the path. An environment with static obstacles is used for the experiment.

Figure 4. Components of ClearPath UGV Jackal

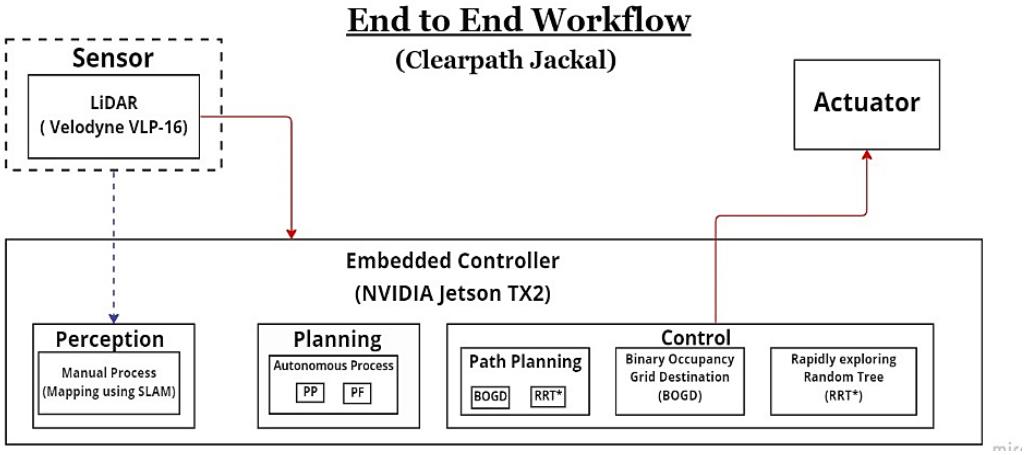


Figure 5. Workflow of implemented hardware and software architectures

In the implementation phase, the initial step involves subscribing to the ROS topic odometry/filtered to acquire data from Odom, which furnishes orientation and position values. As the Jackal operates based on position input, a comparison is made with the expected experimental outcome pertaining to position. The RRT* generated path commences from the coordinates 1,0, considering our Jackal's movement in an open-loop system. Consequently, it assumes the current position as 1,0. Notably, an observable disparity exists between the ground truth and raw data, indicating an offset in the x-direction of 1. To align the origin, we adjusted the X + 1. However, the graph becomes inverted after three steps. Subsequently, we corrected this by flipping the data, achieving partial similarity with the ground truth. Nevertheless, due to our approximations, the experimental data exhibits scaling up and offset errors in both the x and y axes. The mean absolute error (MAE) and root mean square error (RMSE) for the comparative analysis of path correction techniques in our model are 15.83 and 12.054, respectively, indicating a moderate level of deviation between the predicted and actual path data points. The comparative analysis of path correction is shown in Figure 6.

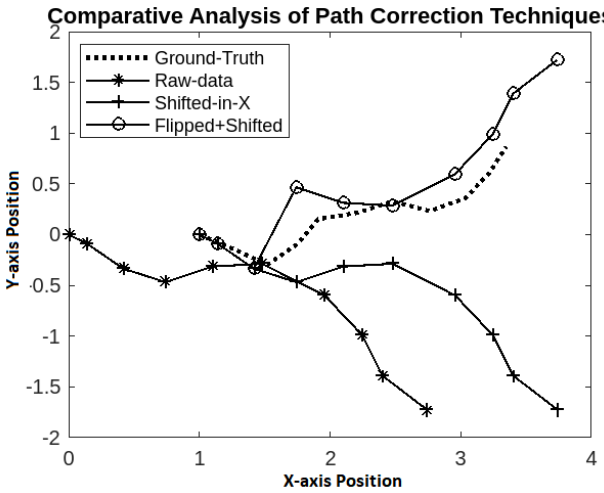


Figure 6. Comparative analysis of data transformations against ground-truth reference (linear velocity (1 unit) = 0.1 m, and angular velocity (1 unit) = 15°)

5. CONCLUSION

In conclusion, path planning for mobile robots is a critical component in ensuring the efficiency and accuracy of their operations. We reviewed and compared two popular sampling-based planners, RRT and RRT*, in terms of step size, execution time, and path length, using the Jackal robot in a static environment cluttered with obstacles. We found that RRT*, an advanced version of RRT, performs more efficiently and finds an optimal path, which plays a key role in the robot's navigation Our performance comparisons revealed that reducing step size results in exploring a greater number of nodes in both algorithms, increasing the probability of each extension succeeding, but causing the tree to become denser. Furthermore, we found that the RRT planner requires less execution time when the step size and iteration count are equal to RRT* planners. Our results also show that RRT* provides an optimal and smooth path compared to RRT. However, we encountered some challenges in our experiments, such as offset errors and scaling issues, which may affect the accuracy of the results. Overall, our study provides valuable insights into the performance of RRT and RRT* planners and highlights the importance of careful experimentation and evaluation in the development of path planning algorithms for mobile robots.

ACKNOWLEDGEMENTS

The proposed work was conducted at the Robotics Lab of Columbus State University. We are grateful to Dr. Mahmut Reyhanoglu, the Dean of the Robotics Department, for granting us the opportunity to work on the ClearPath Jackal. We would also like to express our gratitude to Dr. Mohammad Jafari for his guidance and support throughout the project.

REFERENCES

[1] A. A. A. Rasheed, M. N. Abdullah, and A. S. Al-Araji, "A review of multi-agent mobile robot systems applications," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 4, pp. 3517–3529, 2022, doi: 10.11591/ijece.v12i4.pp3517-3529.

[2] M. Cardona, F. Cortez, A. Palacios, and K. Cerros, "Mobile robots application against covid-19 pandemic," *2020 Ieee Andescon, Andescon 2020*, 2020, doi: 10.1109/ANDESCON50619.2020.9272072.

[3] N. A. K. Zghair and A. S. Al-Araji, "A one decade survey of autonomous mobile robot systems," *International Journal of Electrical and Computer Engineering*, vol. 11, no. 6, pp. 4891–4906, 2021, doi: 10.11591/ijece.v11i6.pp4891-4906.

[4] A. Pandey, "Mobile Robot Navigation and Obstacle Avoidance Techniques: A Review," *International Robotics & Automation Journal*, vol. 2, no. 3, 2017, doi: 10.15406/iratj.2017.02.00023.

[5] S. Lin, A. Liu, J. Wang, and X. Kong, "A review of path-planning approaches for multiple mobile robots," *Machines*, vol. 10, no. 9, pp. 1–27, 2022, doi: 10.3390/machines10090773.

[6] T. Sandakalum and M. H. Ang, "Motion planning for mobile manipulators—a systematic review," *Machines*, vol. 10, no. 2, 2022, doi: 10.3390/machines10020097.

[7] Y. Tian *et al.*, "Application of RRT-based local path planning algorithm in unknown environment," in *2007 International Symposium on Computational Intelligence in Robotics and Automation*, Jun. 2007, pp. 456–460, doi: 10.1109/CIRA.2007.382896.

[8] Steven M. LaValle, "Rapidly-exploring random trees : a new tool for path planning," *The annual research report*, pp. 1–4, 1998.

[9] D. Connell and H. M. La, "Dynamic path planning and replanning for mobile robots using RRT," *2017 IEEE International*

Conference on Systems, Man, and Cybernetics, SMC 2017, vol. 2017-Janua, pp. 1429–1434, 2017, doi: 10.1109/SMC.2017.8122814.
[10] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011, doi: 10.1177/0278364911406761.
[11] M. T. Ejaz, A. Zahid, and M. M. Ejaz, *International Conference on Artificial Intelligence for Smart Community*, vol. 758. Singapore: Springer Nature Singapore, 2022.
[12] J. Ni, L. Wu, X. Fan, and S. X. Yang, "Bioinspired intelligent algorithm and its applications for mobile robot control: A survey," *Computational Intelligence and Neuroscience*, vol. 2016, 2016, doi: 10.1155/2016/3810903.
[13] M. Burch, J. Van Lith, N. Van De Waterlaat, and J. Van Winden, "Voronoier: From images to Voronoi diagrams," *ACM International Conference Proceeding Series*, no. 1, 2020, doi: 10.1145/3430036.3430043.
[14] M. Korkmaz and A. Durdu, "Comparison of optimal path planning algorithms," *14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2018 - Proceedings*, vol. 2018-April, pp. 255–258, 2018, doi: 10.1109/TCSET.2018.8336197.
[15] A. Belaid, B. Mendil, and A. Djenadi, "Narrow passage RRT*: A new variant of RRT*," *International Journal of Computational Vision and Robotics*, vol. 12, no. 1, pp. 85–100, 2022, doi: 10.1504/IJCVR.2022.119270.
[16] Y. Li, W. Wei, Y. Gao, D. Wang, and Z. Fan, "PQ-RRT*: An improved path planning algorithm for mobile robots," *Expert Systems with Applications*, vol. 152, p. 113425, 2020, doi: 10.1016/j.eswa.2020.113425.
[17] Y. Yang, H. Leeghim, and D. Kim, "Dubins Path-Oriented Rapidly Exploring Random Tree* for Three-Dimensional Path Planning of Unmanned Aerial Vehicles," *Electronics (Switzerland)*, vol. 11, no. 15, 2022, doi: 10.3390/electronics11152338.
[18] M. Ye, X. Dong, J. Zhao, and Z. Huang, "Path Planning of Manipulator Based on Improved RRT* Algorithm," *Journal of Physics: Conference Series*, vol. 2365, no. 1, pp. 4741–4746, 2022, doi: 10.1088/1742-6596/2365/1/012038.
[19] I. Noreen, A. Khan, and Z. Habib, "Optimal Path Planning using RRT* based Approaches: A Survey and Future Directions," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 11, pp. 97–107, 2016, doi: 10.14569/ijacsa.2016.071114.
[20] J. Jermyn and R. Roberts, "Path Planning Algorithms: An Evaluation of Five Rapidly Exploring Random Tree Methods," 2023, doi: 10.5038/itsw6402.

BIOGRAPHIES OF AUTHORS

Shokoufeh Davarzani is a graduate student at Robotics Department of Columbus State University. She is working on some research in the field of optimization algorithms, control theory, and computational intelligence. Her main research directions include signal processing, artificial intelligence (neural networks, fuzzy logic, genetic algorithm, control intelligent systems and robotic neural, evolutionary and fuzzy computation neural networks optimization. She can be contacted at davarzani_shokoufeh@students.columbusstate.edu.

Muhammad Talha Ejaz is a graduate student in the Robotics Department at Columbus State University, USA, holding a bachelor's degree in Mechatronics Engineering from the Karachi Institute of Economic Technology (K.I.E.T). He is currently focusing on a multi-UAV-based vision system and a deep CNN-genetic hybrid system. His research interests include AI, computer Vision, power systems, neuroscience, deep learning, autonomous systems, and neural network optimization. His aims are to contribute significantly to the field of robotics and AI. He can be contacted at ejaz_muhammadtalha@students.columbusstate.edu.