Improved RRT-Connect Based Path Planning Algorithm for Mobile Robots

ENPM661 - Project 5

Ishan Kharat
University of Maryland College Park
ishanmk@umd.edu

Abhey Sharma
University of Maryland College Park
abheys16@umd.edu

***Abstract*—Path planning is essential for mobile robots, enabling them to navigate intelligently in diverse environments. Traditional algorithms face challenges in modeling deterministic spaces with obstacles, often leading to entrapment in local minima. In contrast, sampling-based approaches like Rapidly Exploring Random Trees (RRT) offer rapid path generation through collision detection.**

However, the RRT-Connect algorithm, while efficient, can suffer from suboptimal search behavior. To address this, we propose an enhanced version termed IRRT-Connect. This algorithm introduces a novel approach to expedite search by incorporating a straightforward yet powerful third node generation mechanism. By extending the algorithm with a quadruple tree structure, search efficiency is further enhanced. Moreover, we introduce a guidance mechanism to bias the expansion of the tree towards the target point. This feature significantly improves the exploration efficiency of the algorithm, enabling it to find more optimal paths in complex environments.

To evaluate the effectiveness of our proposed algorithm, we conducted [30] extensive simulation experiments across in an environment of varying complexity. The results demonstrate that IRRT-Connect outperforms traditional RRT, RRT-Connect, and RRT algorithms in terms of algorithm iterations, planning time, and final path length is fully detailed on our https://github.com/IshanMahesh/Improved-RRT-Connect-for-Mobile-RobotsGitHubfor further research engagement.

***Index Terms*—Path planning, RRT-connect, target bias, dichotomous method.**

I. Introduction

In recent decades, the proliferation of advanced hardware technologies has facilitated the widespread adoption of mobile robots across diverse sectors such as manufacturing, healthcare, agriculture, and services. Within this context, path planning emerges as a pivotal technology and a formidable challenge for mobile robots. Path planning involves the intricate task of determining an optimal or sub-optimal route from a specified starting point to a predefined goal, taking into account various evaluation criteria such as planning time, path length, and obstacle avoidance. This process is typically categorized into two main types: global path planning, where the environment's information is fully known in advance, and local path planning, which operates under the assumption of incomplete or dynamically changing environmental information.

A plethora of algorithms has been developed to address the complexities of path planning in robotic systems. These algorithms encompass a wide spectrum of approaches, including classical methods such as the Dijkstra algorithm, which prioritizes greed-based exploration, and heuristic algorithms like A* and artificial potential field, which leverage heuristic estimates to guide search processes more efficiently. Additionally, bio-inspired algorithms such as the bionic ant colony algorithm draw inspiration from natural phenomena to optimize path finding in complex environments. Over time, researchers have fine-tuned these foundational algorithms and adapted them to suit specific application scenarios. For instance, enhancements to the A* algorithm have been proposed by refining heuristic functions and introducing novel path smoothing strategies to generate safer routes that steer clear of obstacles. Moreover, in environments with dynamic factors or safety concerns, algorithms often convert spatial distances and safety considerations into temporal costs to facilitate more informed decision-making during path planning. Despite these advancements,

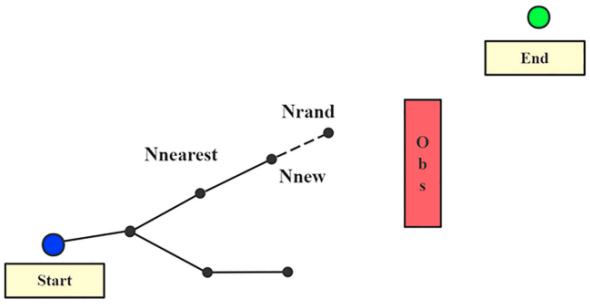

Fig. 1. RRT algorithm tree expansion process.

traditional path planning algorithms face significant challenges when navigating complex environments. The computational complexity of these algorithms escalates exponentially in intricate scenarios, and they frequently encounter the issue of getting trapped in local minima, where suboptimal solutions are reached prematurely. To address these limitations and improve the efficiency and practicality of path planning algorithms, researchers have turned to randomized sampling-based

approaches. These algorithms, such as Rapidly Exploring Random Trees (RRT), are characterized by their probabilistic completeness and resolution completeness, which contribute to reducing the computational burden and increasing efficiency. However, the inherent randomness in these algorithms may lead to the generation of non-optimal paths.

To overcome these shortcomings, researchers have proposed enhancements to randomized sampling-based algorithms. These improvements aim to strike a balance between exploration and exploitation, enabling more effective path planning in complex environments. For instance, Zhang et al. introduced a regression mechanism to RRT to mitigate its tendency to converge to local minima in challenging environments. Similarly, Wei and Ren implemented a directional expansion strategy in RRT, enhancing its efficiency and incorporating curvature constraints for smoother path trajectories. Moreover, researchers have explored the integration of RRT with other path planning techniques, such as the artificial potential field algorithm, to address specific challenges like oscillations and ensure more stable robot navigation.

Among the bidirectional exploration strategies, RRT-Connect stands out for its capability to expedite search efficiency by simultaneously generating two random trees from the start and end points. However, while RRT-Connect accelerates tree expansion, resulting paths may still fall short of optimality. To address this, Kang et al. devised a triangular inequality-based variant of RRT-Connect, achieving notable improvements in path length reduction compared to the original algorithm. Additionally, efforts have been made to optimize parent node selection within the RRT framework, leveraging cost functions to prioritize nodes with the smallest cost in the extended neighborhood for enhanced convergence towards optimal solutions.

In light of the ongoing advancements and challenges in path planning algorithms, this paper proposes an enhanced version of the RRT-Connect algorithm, denoted as IRRT-Connect. This novel approach aims to further expedite search efficiency in sampling-based algorithms, offering promising prospects for more efficient and reliable robotic navigation in diverse real-world scenarios.In general, the contributions to this paper are as follows:

- **Generation of a Third Node**: The algorithm generates a third node in the configuration space, in addition to the start and end points, to further optimize the search speed of the RRT-Connect algorithm. The third node is generated as the midpoint of the line connecting the start and end points. If this midpoint is on an obstacle, a dichotomy-based approach is used to find a valid third node. This third node allows the algorithm to expand four random trees instead of just two, significantly improving the search efficiency.
- **Increasing the Power of Guidance**: The algorithm modifies the generation of new nodes to be more biased towards the target point. The coordinates of the new node are calculated based on the nearest node on the tree, the sampling point, and the angle between the nearest node and the target point. This increases the guiding force towards the target, improving the exploration efficiency of the algorithm.

II. Related Work

A. Rapidly Exploring Randon Trees [RRT]

The Rapidly Exploring Random Trees (RRT) algorithm begins by randomly selecting a configuration node within the free space of the environment, ensuring exploration of diverse areas. It then identifies the nearest node in the existing tree structure and attempts to connect it to the randomly sampled node by steering towards it with a specified step size. If an obstacle is encountered during this process, the corresponding edge is considered invalid to maintain a collision-free path. This iterative process continues until a termination condition,

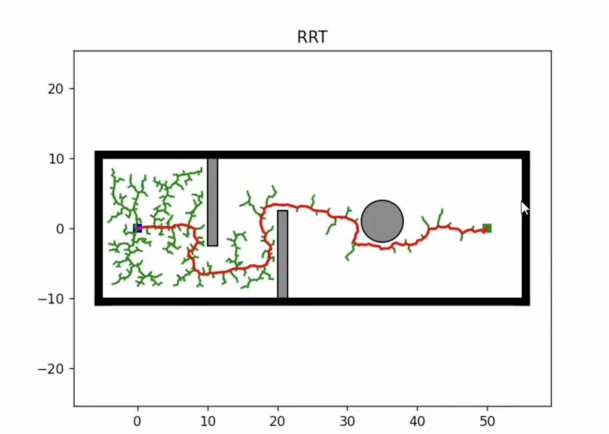

Fig. 2. RRT

such as reaching the goal region or a maximum number of iterations, is met. While RRT does not guarantee optimal paths, it efficiently generates feasible paths, making it valuable for mobile robot navigation, particularly in complex environments where exact optimality is not crucial.

B. RRT Star

In the RRT-star algorithm, an enhancement over RRT, the utilization of cost-to-come information for each node relative to the start node is introduced, marking a departure from RRT's lack of map information utilization. Upon random node selection within free space, the algorithm evaluates cost-to-come for all nodes within a specified radius around the sampled node, prioritizing nodes with the lowest cost for further expansion towards the sampled node with a predetermined step size.

Additionally, two new steps are integrated into the algorithm: the selection of the lowest cost node for expansion and subsequent rewiring to minimize cost-to-come for neighboring nodes, ensuring path optimization after each iteration. This iterative refinement process distinguishes RRT-star, fostering the generation of solutions closer to optimality while retaining the fundamental structure and exploration capability of the RRT algorithm.

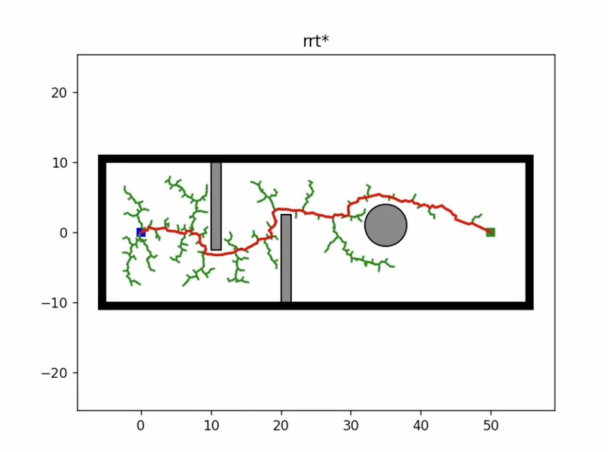

Fig. 3. RRT Star

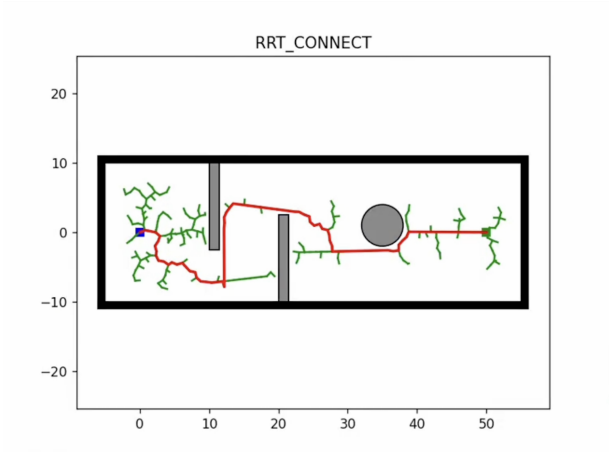

Fig. 5. RRT Connect

C. RRT Connect

In the realm of path planning for mobile robots, both the Rapidly Exploring Random Trees (RRT) and RRT-star algorithms traditionally employ random sampling within the free space without regard for the goal point's position. This indiscriminate sampling strategy, while straightforward, often leads to a significant increase in computational time and the number of iterations required for path planning. However, the RRT Connect algorithm introduces two crucial modifications aimed at mitigating these inefficiencies. Firstly, unlike RRT

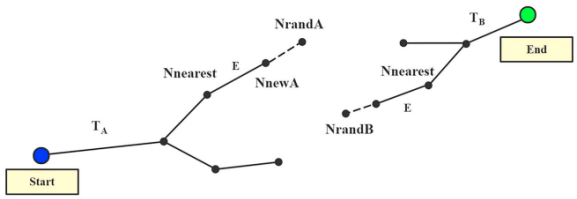

Fig. 4. The position of the generated third node

and RRT-star, RRT Connect simultaneously grows trees from both the start and goal nodes, halting planning once these trees intersect, thereby reducing the number of iterations and computational overhead. Secondly, to expedite the convergence of the trees, the node selection process in RRT Connect is biased towards nodes proximate to the opposing tree. This bias accelerates the convergence of the trees by favoring connections between nodes from different trees, further reducing the time required for planning. To implement the RRT Connect algorithm, the initial steps mirror those of RRT: a node is randomly sampled within the free space, and it is then connected to the nearest node on the starting tree. Subsequently, the algorithm attempts to establish a direct connection between the newly generated node and the nearest node on the opposing tree. If this connection is unobstructed by obstacles, signifying a successful path to the goal, the planning process terminates. Conversely, if an obstacle obstructs the direct connection, the node is extended towards the obstacle until it is reached, at which point the algorithm switches to the other tree and repeats the process. This alternating sequence continues until the trees intersect, indicating that a feasible path from the start to the goal has been found.

RRT Connect enhances upon RRT in three key aspects: it typically yields a lower-cost final path, reduces the number of iterations required for planning, and decreases the computational time needed to find a feasible path. By strategically growing trees from both the start and goal nodes and biasing node selection towards connections with the opposing tree, RRT Connect offers a more efficient and effective approach to path planning for mobile robots, particularly in scenarios where computational resources or time constraints are paramount.

D. Generation of a Third Node

Building upon the foundational idea of RRT-Connect, which enhances path planning efficiency by constructing random trees from both the start and end points, this paper proposes a refinement by introducing a third node to further optimize the iteration process of the original RRT-Connect algorithm. Initially, within the configuration space, the coordinates of the start and end points, denoted as (x1, y1) and (x2, y2) respectively, are established. To expedite the generation of a suitable third node, the algorithm calculates the midpoint of the line connecting the start and end points. This midpoint, termed as Xmid (xmid, ymid), is computed using the following equations:

$$x_{\text{mid}} = \frac{x_1 + x_2}{2}$$

$$y_{\text{mid}} = \frac{y_1 + y_2}{2}$$

where x_1, x_2, y_1, y_2 represent the horizontal and vertical coordinates of the start and end points in the configuration space, respectively.

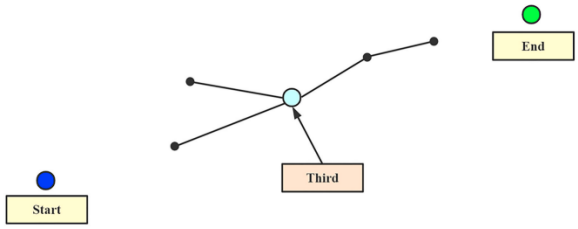

Fig. 6. The position of the generated third node

Subsequently, two scenarios may unfold in the generation of this third node. In the first scenario, if the calculated midpoint falls within an obstacle-free region, a single iteration is sufficient to locate the third node, thereby markedly expediting the search process. Conversely, if the midpoint coincides with an obstacle, a dichotomy approach is employed to iteratively search for a valid third node. This iterative process involves selecting midpoints between the start point and the obstacle-pointed midpoint, as well as between the obstacle-pointed midpoint and the end point, which serve as alternative valid nodes. The selection process continues iteratively until a valid

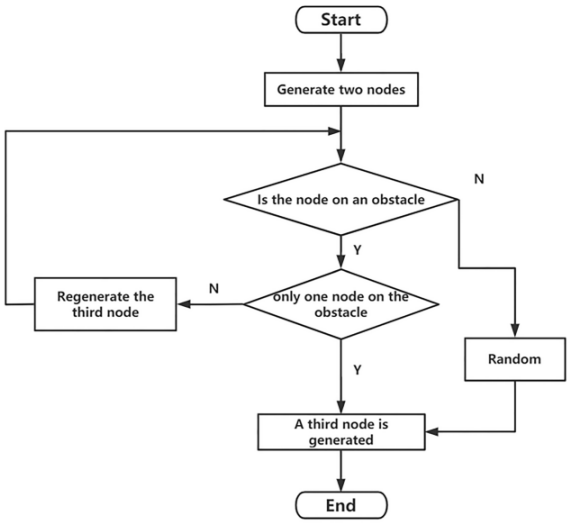

Fig. 7. Flowchart for generating the third node.

third node is successfully identified. To illustrate, the process begins by calculating the midpoint between the start point (Xstart) and the midpoint on the obstacle, referred to as Xstartmid. This midpoint's coordinates, denoted as (x, y), are computed as follows:

$$X_{\text{startmid}}(x, y) = \left(\frac{x_1 + x_{\text{mid}}}{2}, \frac{y_1 + y_{\text{mid}}}{2} \right)$$

Similarly, the midpoint between the obstacle-pointed midpoint and the end point (Xendmid) is computed using the following equations:

$$X_{\text{endmid}}(x, y) = \left(\frac{x_2 + x_{\text{mid}}}{2}, \frac{y_2 + y_{\text{mid}}}{2} \right)$$

If both candidate third nodes lie within obstacle-free regions, one is randomly selected as the valid third node. Conversely, if one candidate node lies on an obstacle, the other non-obstructed node is chosen as the valid third node. If both generated nodes coincide with obstacles, the dichotomy process is reiterated towards the midpoint Xmid, based on these alternative third nodes. This iterative search continues until a valid third node is successfully identified.

A detailed flowchart, as depicted in Fig. 8, elucidates this iterative process, particularly when the initially generated third node lies on an obstacle. This meticulous approach to refining the RRT-Connect algorithm aims to expedite path planning efficiency and enhance robustness in complex environments.

E. Improved Algorithm Implementation Process and Increasing the Power of Guidance

To enhance the efficiency and effectiveness of the RRT-Connect path planning algorithm, we devised a strategy involving the introduction of a third node and the subsequent adaptation of the replanning process for new nodes. The detailed implementation of this strategy is encapsulated in the ImproveExtend function, delineated through Algorithm 1.
This function is pivotal in expanding random trees to generate new nodes, incorporating several key auxiliary functions: WithoutObstacle, which discerns whether a point in the configuration space intersects with an obstacle; Nearest, tasked with identifying the node closest to a given sampling point within the spanning tree; SampleFree, responsible for generating random points within the configuration space for sampling purposes; and Distance, which computes the Euclidean distance between two nodes. Unlike the conventional RRT-Connect algorithm, our improved approach integrates the generation of a third node within the configuration space, thereby necessitating the expansion of four trees to establish the overall path. This comprehensive path planning process is delineated in Algorithm 2, designated as ImproveRRT-Connect. Initially, the algorithm identifies a third node utilizing the SearchThirdNode function, which is implemented through equations (1), (2), or (3), (4) to derive the midpoint between the start and end points or their respective midpoints. Subsequently, four random spanning trees are initialized, denoted as T1 and T2 spanning from the start point to the third node, and T3 and T4 spanning from the third node to the end point.

During the expansion phase, the Connect Without Obstacle function plays a pivotal role in assessing the presence of obstacles between the nodes within the spanning tree and the newly generated node. If an obstacle is detected, the function regenerates the new node to circumvent the obstruction. Employing a prudent expansion strategy, the algorithm continues expansion in a given direction if no obstacles are encountered until either an obstacle is reached or two trees are successfully connected. Furthermore, to augment exploration efficiency, the algorithm initiates a swap expansion using the Swap function

Algorithm 2 ImproveRRT-Connect (X_{start}, X_{end})

Input: X_{start}, X_{end}
Output: Path

$X_{mid} \leftarrow$ SearchThirdNode();
T_1.init (X_{start}), T_2.init (X_{mid}), T_3.init (X_{mid}), T_4.init (X_{end});
for $i \leftarrow 1$ to N **do**
  $N_{rand1} \leftarrow$ SampleFree () $N_{rand2} \leftarrow$ SampleFree ();
  $N_{nearest1} \leftarrow$ Nearest (N_{rand}, T_1) $N_{nearest2} \leftarrow$ Nearest (N_{rand}, T_4);
  $N_{new1} \leftarrow$ ImproveExtend (N_{rand}, T_1) $N_{new2} \leftarrow$ ImproveExtend (N_{rand}, T_4)
  if ConnectWithoutObstacle ($N_{new1}, N_{nearest1}$), ConnectWithoutObstacle ($N_{new2}, N_{nearest2}$), **then**
    $T_1 \leftarrow N_{new1}$
    $T_4 \leftarrow N_{new2}$
    $N_{new3} \leftarrow$ ImproveExtend (N_{new1}, T_2)
    $N_{new4} \leftarrow$ ImproveExtend (N_{new2}, T_3)
    $T_2 \leftarrow N_{new3}$
    $T_3 \leftarrow N_{new4}$
    for $N_{new1} \neq N_{new3}, N_{new2} \neq N_{new4}$
      do $N_{new3\text{-}temp} \leftarrow$ ImproveExtend (N_{new3}, T_2)
      if ConnectWithoutObstacle ($N_{new3\text{-}temp}, N_{new3}$)
        $N_{new3} = N_{new3\text{-}temp}$
      do $N_{new4\text{-}temp} \leftarrow$ ImproveExtend (N_{new4}, T_3)
      if ConnectWithoutObstacle ($N_{new4\text{-}temp}, N_{new3}$)
        $N_{new4} = N_{new4\text{-}temp}$
      else break;
    if $N_{new1} = N_{new3}, N_{new2} = N_{new4}$
      Return path (T_1, T_2) + path (T_3, T_4);
    else Swap (T_1, T_2), Swap (T_1, T_2);
return null;

Fig. 8. Psuedo Code for Improved RRT.

if one tree encounters an obstacle while the other does not, facilitating expedited navigation around obstacles.

The planning process is deemed unsuccessful if the maximum number of iterations N is surpassed before a feasible path is identified. This meticulous attention to detail and the incorporation of iterative strategies underscores our commitment to optimizing the speed and efficacy of the RRT-Connect algorithm for robust path planning in dynamic environments.

III. Experimental Simulation and Analysis

The method iterates through a maximum number of iterations, attempting to find a feasible path from the start to the goal. The ‘planning‘ method orchestrates the iterative exploration process in the path planning algorithm. It operates within a loop, iterating through a maximum number of iterations to find a feasible path from the start to the goal. Two flags, ‘flag-1‘ and ‘flag2‘, are employed to track the progress of exploration towards the goal from different directions. Each iteration begins by generating a random node, biased towards the goal depending on the ‘goal-sample-rate‘. The algorithm then expands from two separate trees: one expanding from the start (‘V1‘) towards the goal (‘V3‘) and the other from the goal (‘V2‘) towards the start (‘V4‘). During expansion, new

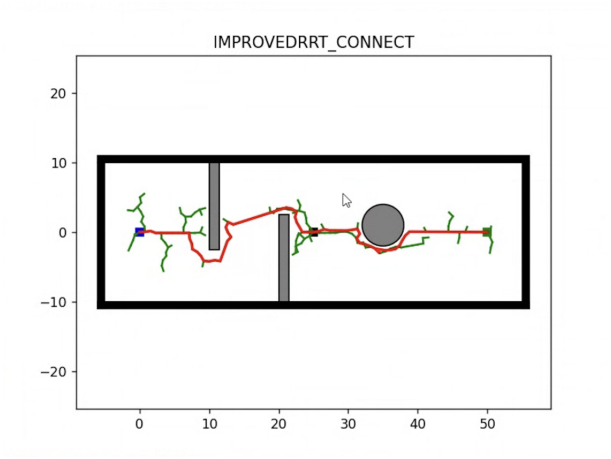

Fig. 9. Improved RRT.

nodes are generated from the nearest nodes in their respective trees to the random node. If a generated node is collision-free, it’s added to its corresponding tree. The algorithm continues expanding each tree towards the goal until further expansion is not possible or until reaching the goal node. Once both expansions towards the goal are successful, paths are extracted from the start to the goal from both trees. Additionally, the method manages node swapping between trees based on their respective node lengths to prioritize trees with more nodes for further expansion. Overall, the ‘planning‘ method effectively navigates the search space to find collision-free paths within the specified iteration limit, offering a robust approach to path planning in complex environments.

Results

The performance of four algorithms executed in an environment was compared, based on Average Time, No of iterations, and length. IRRT-Connect reduced the number of iterations by **24%** , planning time by **42%**, and path length by **11%** compared to RRT-Connect.

Refer **Table 1 for Comparative Analysis**

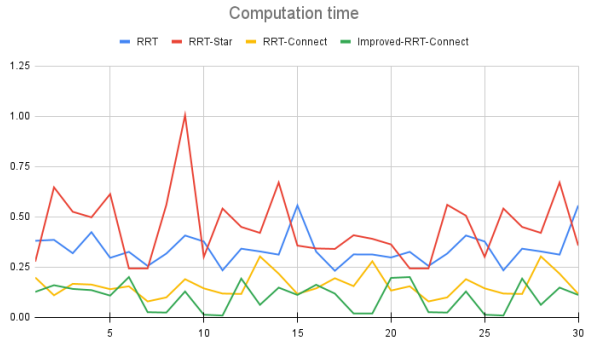

Fig. 10. Computation time.

Algorithm	Average Time	Average No of Iterations	Average Length
RRT	503	490	485
RRT Star	13.1	13.9	13.1
RRT Connect	754	747	745
IRRT Connect	13.1	13.9	13.1

TABLE I
COMPARATIVE RESULTS BETWEEN RRT, RRT, RRT CONNECT, AND IRRT

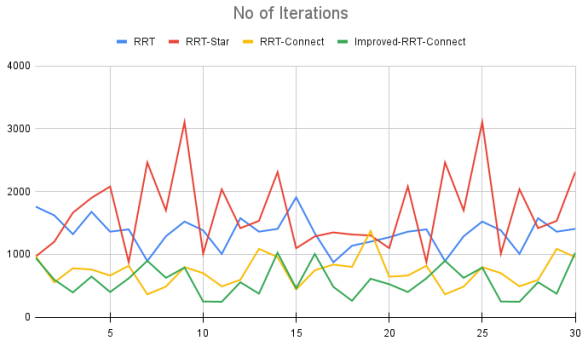

Fig. 11. No of Iterations.

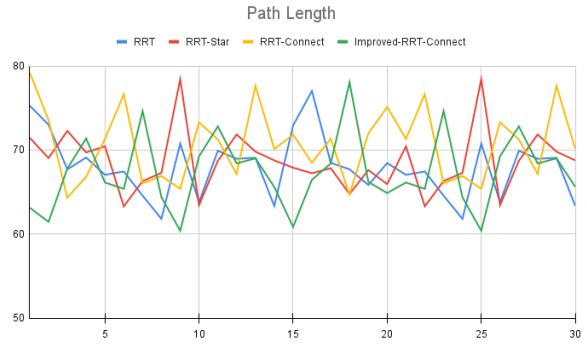

Fig. 12. Path Length.

CONCLUSION

Path planning for mobile robots within complex environments poses a significant challenge in research. Sampling-based path planning algorithms offer a solution due to their simplicity and efficient exploration of multidimensional spaces. This study focuses on enhancing the RRT-Connect algorithm. Initially, to boost exploration efficiency, the algorithm is extended using a quadtree structure by introducing a third node generation. Additionally, to address the blind search issue inherent in RRT-Connect, the node generation process is reprogrammed to favor the target point, thereby expediting search efficiency. To assess the proposed algorithm's effectiveness and applicability, six environment maps of varying complexities are constructed and evaluated across 30 experiments. In simpler environments, the enhanced algorithm substantially reduces iterations, pathfinding time, and resulting path length compared to traditional RRT and RRT-Connect algorithms. The experimental findings demonstrate the superior performance of the enhanced algorithm in various aspects within complex environments, attributed to its increased bias towards exploration efficiency, thus affirming the algorithm's versatility and effectiveness.

REFERENCES

[1] Jiagui Chen , Yun Zhao , Xing Xu, "Improved RRT-Connect Based Path Planning Algorithm for Mobile Robots," Digital Object Identifier 10.1109/ACCESS.2021.3123622.
[2] J.J. Kuffner; S.M. LaValle, RRT-connect: An efficient approach to single-query path planning, 10.1109/ROBOT.2000.844730
[3] S.M. LaValle, "Rapidly-Exploring Random Trees: A New Tool got Path Planning
[4] X. Gao, J. Li, L. Fan, Q. Zhou, K. Yin, J. Wang, C. Song, L. Huang, and Z. Wang, "Review of wheeled mobile robots' navigation problems and application prospects in agriculture," IEEE Access, vol. 6, pp. 49248–49268, 2018.
[5] O. Khatib, "Real-time obstacle avoidance system for manipulators and mobile robots," Int. J. Robot. Res., vol. 5, no. 1, pp. 90–98, 1986.
[6] A. Maoudj and A. Hentout, "Optimal path planning approach based on Q-learning algorithm for mobile robots," Appl. Soft Comput., vol. 97, Dec. 2020, Art. no. 106796.
[7] H.-M. Zhang, M.-L. Li, and L. Yang, "Safe path planning of mobile robot based on improved A* algorithm in complex terrains," Algorithms, vol. 11, no. 4, p. 44, Apr. 2018.