Implementing RRT-Connect and GT-RRT for Motion Planning and Comparing the Results with Standard RRT

*

1st Divyansh Agrawal
A. James Clark School of Engineering
University of Maryland
College Park, MD, USA
dagrawa1@umd.edu

2nd Jai Sharma
A. James Clark School of Engineering
University of Maryland
College Park, MD, USA
jaisharm@umd.edu

***Abstract*—In order to enhance the performance of robot systems in the manufacturing industry, it is essential to develop motion and task planning algorithms. Especially, it is important for the motion plan to be generated automatically in order to deal with various working environments. Our work aims at exploring improved RRT algorithms in greater detail by leveraging the work done by established researchers in the field. We look at two variants: Improved RRT-Connect and GT-RRT. Through our work, we will implement these algorithms on a point robot in a 2D obstacle space. We will make observations and test the efficiency and reliability of the two variants. We will also compare the advantage of using variants of RRT over the standard RRT algorithm.**

***Index Terms*—RRT, RRT-Connect, Node, Path Planning**

==

I. Introduction

Sample Based Planning algorithms are extremely useful in addressing motion planning problems. Steven M. LaValle and James J. Kuffner Jr initially developed the Rapidly Exploring Random Tree (RRT) algorithm and since then, the algorithm has been widely studied by researchers. There are still some problems like slow convergence speed and low efficiency that researchers aim to address. Some researchers have been successful in enhancing the basic concept by developing variants to tackle these shortcomings. Through this project, we aim to understand and implement a few of these enhanced variants of RRT. We explore two specific variants that are inspired by work done in [1] and [2]. They are as follows:

- Improved RRT-Connect: looks at the result of RRT and enhances it to get optimal path [1]
- GT-RRT: couples three source fast expansion technique with a superimposed potential field [2]

To make better observations, we compare the results of the aforementioned algorithms with the standard RRT, RRT-Connect and Biased RRT-Connect results. This gives us an entire suite of RRT algorithms that spread from traditional to the improved ones resulting in an optimal path.

II. Literature Review

A. Improved RRT - Connect

Shuyu Li et al. in their paper titled ”Path Planning Algorithm Based on the Improved RRT-Connect for Home Service Robot Arms” discuss the concept for improving the RRT-Connect algorithm. The idea of the improved algorithm is to delete redundant nodes to simplify the path.

B. GT - RRT

C. Zhao et al. in their paper titled ”Research on Path Planning of Robot Arm Based on RRT-connect Algorithm” address the shortcomings of the RRT-connect algorithm like low efficiency of path planning in sampling space and high randomness of node sampling. The paper explains how it incorporates a third exploration node and a potential field to make the algorithm faster. The planning algorithm is tested using a 2D map and a point robot. The method is also tested on a Baxter model on ROS as a 3D model.

III. Background

A. Why Path Planning is Important?

Path planning for industrial robots is an essential aspect of the overall performance of automation systems. Essentially, path planning algorithms determine how an industrial robot arm should approach a part, how it should process a part, and how it should orient itself for optimal productivity and to avoid collisions. Industrial robot path planning is a necessary component of a productive automation system, making it vitally important to ensure path planning is accurate, safe, and efficient [5]. Robot path planning plays an important part in:

- **Robot Accuracy:** a robot’s path needs to be meticulously planned in order for it to productively process a part with little or no error.
- **Task Repeatability:** once a robot’s path is well-defined it can repeat the same task thousands of times without variation to help accelerate throughput.

- **Product Quality:** when products are created with a high degree of accuracy and repeatability, there are fewer mistakes and higher consistency, leading to higher overall quality products.

B. What is RRT?

RRT generates nodes randomly and then connects each node to the closest available node. Each time a node is created, a check must be made that the node lies outside of an obstacle. Furthermore, chaining the node to its closest neighbor must also avoid obstacles. The algorithm ends when a node is generated within the goal region, or a limit to the number of iterations is hit.

C. What is RRT Connect?

RRT Connect is a bi-directional algorithm that grows two fast extended random trees from the initial state point q_{init} and the destination state point q_{goal} at the beginning of operation to search the state space. In each iteration, the sampling random point is extended, the first tree gets the new node q_{new} and takes the nearest q_{new} as the target state point of the second tree q_{near}, the second tree expands in the direction of q_{near} to get the next node q_{target}. The first tree continues to expand with q_{target} as the new destination state point, and repeat iteration of two trees. If no collision occurs, proceed to the next extension until the extension fails or two trees are connected $q_{new} = q_{target}$ the whole algorithm ends. This has been explained in figure 1. [3] [4]

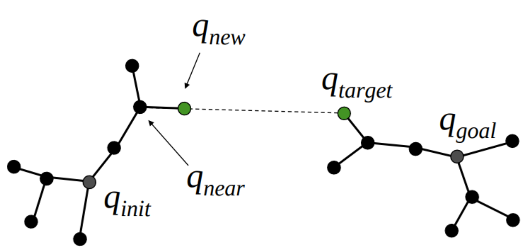


Fig. 1. RRT Connect

IV. Improved RRT Variants

A. What is Improved RRT Connect?

(Refer figure 1) Once RRT finds a path from start to goal (considering this is not the optimal path), then obstacle presence is checked between the initial node q_{init} and the next to next node q_{near}. If there is no obstacle then q_{init} is connected to the q_{near} (next to next node) and the nodes in between are deleted as they are redundant. The loop continues until the goal is reached. If an obstacle is found between q_{init} and q_{near} then the first path is the connection path between q_{init} and the node before q_{near}, called as $q^{'}_{near}$ and the loop operation continues with $q^{'}_{near}$ as the new starting node for that specific iteration. Figure 2 shows the algorithm for Improved RRT Connect. [1]

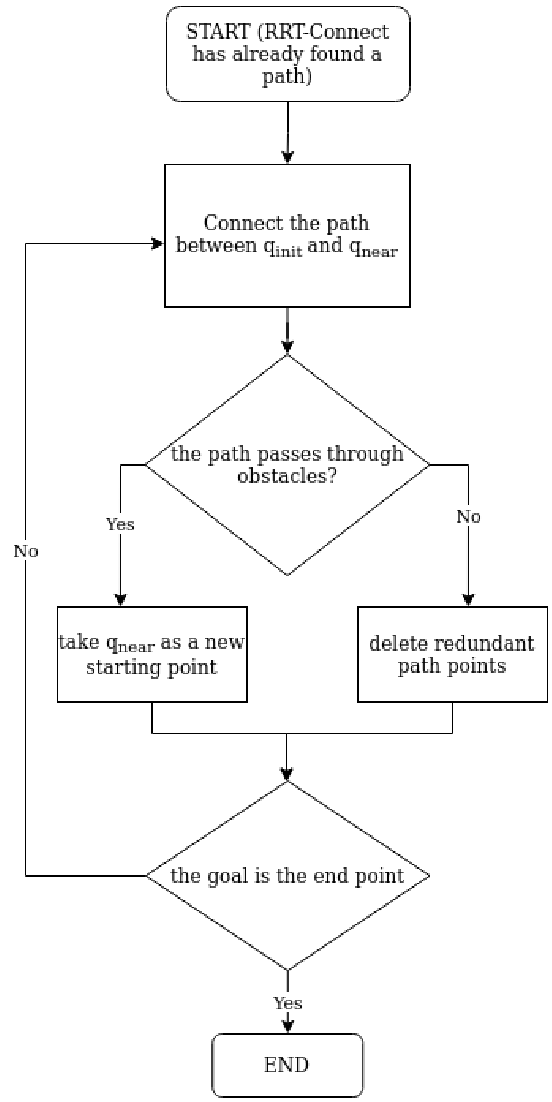


Fig. 2. Algorithm for Improved RRT Connect

B. What is GT - RRT?

As mentioned earlier, GT-RRT [2] is an improved path planning variant of the renowned RRT and RRT-Connect method. It uses two techniques to reduce the randomness of sampling and increase the efficiency of search. They are elaborated below:

- **Introduction of a 3rd Node:** The code introduces a third node, strategically placed in between the start node and goal node. So, as the start and end nodes build branches in random directions, the third node simultaneously expands it's two random trees to cover greater are in the same amount of time. As soon as the start node and goal node branch connect to the nodes of the third node branches, the solution is found. An object on the start node can traverse through the start node branch, through the third node branches and the end node branch to ultimately reach the goal node. There is a possibility that the third node lies in Obstacle space. To counter that, node selection is iterated right in the beginning of the algorithm until we find an acceptable node. The new node can be translated up or down in the vertical direction of the line by one or more units distance as shown below:

$$x_3 = \frac{x_1 + x_2}{2} + n * \frac{\sqrt{2}}{2} \quad (1)$$

$$y_3 = \frac{y_1 + y_2}{2} + n * \frac{\sqrt{2}}{2} \quad (2)$$

Here, n represents n units of distance moved along the vertical line. In total, there are 4 random trees expanding at any given point. More area is being explored in each iteration and hence the exploration technique is more effective.

- **Superimposed Gravitational Field**: The concept of superimposing a gravitational field is similar to the idea of introducing a potential field to an algorithm. The goal is to integrate a function that encourages the algorithm to choose the node that is closer to a target node. In this algorithm, the researchers introduce the bias by adding a target gravitational function $G(n)$ to each of the three nodes: Start Node, Goal Node and End Node. The overall equation for finding new node is:

$$F(n) = R(n) + G(n) \quad (3)$$

$F(n)$ denotes the expansion function of the new node to the target point and (n) is the random expansion function of the new node. The random expansion function $R(n)$ of the new node is the same as that used for any RRT-Connect Algorithm:

$$R(n) = u * \frac{X_{rand} - X_{near}}{\|X_{rand} - X_{near}\|} \quad (4)$$

The gravitational function is constructed using the square of the Euclidean distance from the target point to the nearest point in random tree.

$$G(n) = u * \varepsilon * \|x_{goal} - x_{near}\|^2 \quad (5)$$

By this logic, the gravitational force is strongest at a node furthest away from the target node. Hence, the two bidirectional random trees are expanded in the direction of of each other, guided by their respective gravitational components.

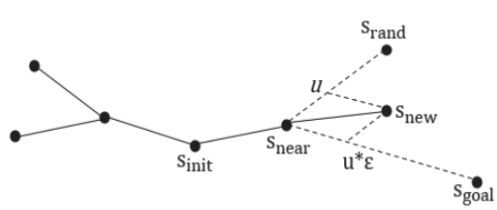


Fig. 3. Expansion Step

The expansion step in Figure 3 shows how the New node falls somewhere between the Random Node and biased Goal Node. The flowchart can be referred to, to understand the detail chronology of this RRT variant. Figure 4 shows the algorithm for GT-RRT.

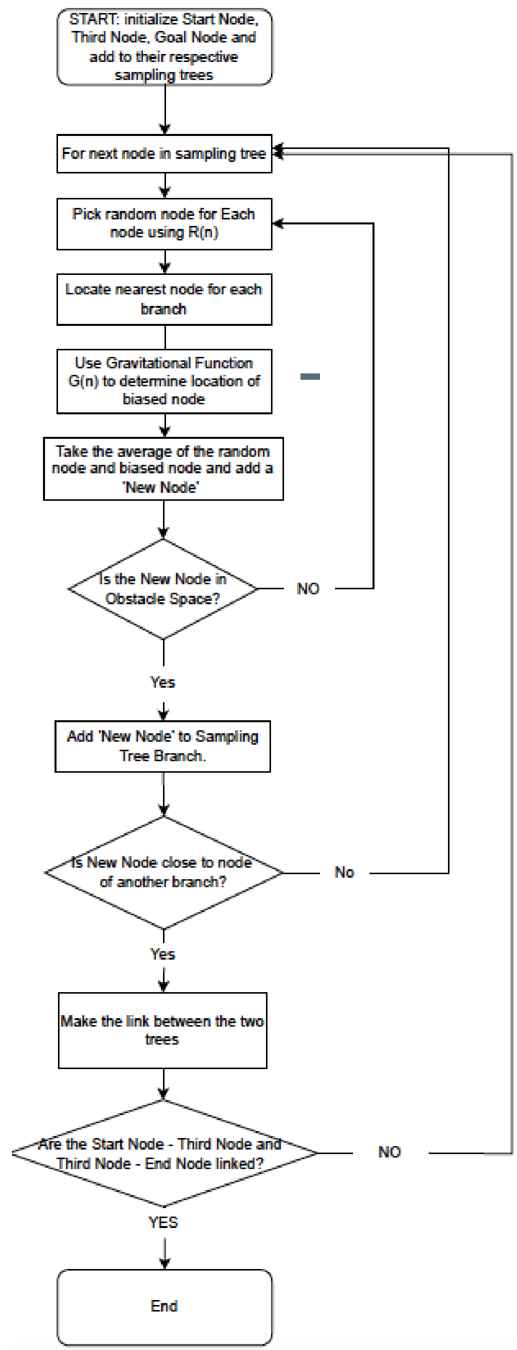


Fig. 4. GT-RRT Algorithm

C. *What is RRT Connect - Biased?*

This is a biased version of RRT-Connect so the sampling method is not Random. In this version, the start node generates the child nodes in the direction of the goal. There is a check to make sure the new node is valid and not in obstacle space. The branch keeps traversing until an object is detected, only then, the branch will make nodes in a random direction. The extreme bias causes the exploration to be limited in the direction of the other branch, hence being efficient. It does not always give the most optimized path but it will definitely find the goal swiftly.

V. Simulation Results

We have considered a map of size 10*10 with obstacles. The images below show the simulation results for different variants of RRT. The table below shows the time taken and number of nodes used to reach the goal in all the different RRT Variants.

Algorithm	**Time Taken (in seconds)**	**Number of Nodes in Optimised Path**
RRT	25.09	155
RRT Connect	21.94	198
RRT Connect Biased	9.26	128
Improved RRT Connect	26.68	198
RRT Connect - Second Map	27.78	212
GT-RRT - Second Map	20.79	175

- **RRT:** Figure 5 shows the simulation for standard RRT.

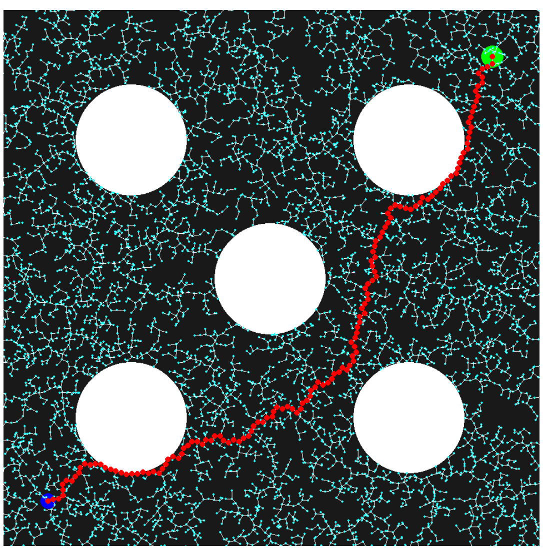


Fig. 5. Standard RRT

- **RRT Connect:** Figure 6 shows the path formed by RRT Connect. Since this is a bi-directional algorithm, nodes are generated from start as well as goal points. It can be seen in the figure that the nodes in blue color are the ones generated by the start node and the nodes in green are the ones generated by the goal node.

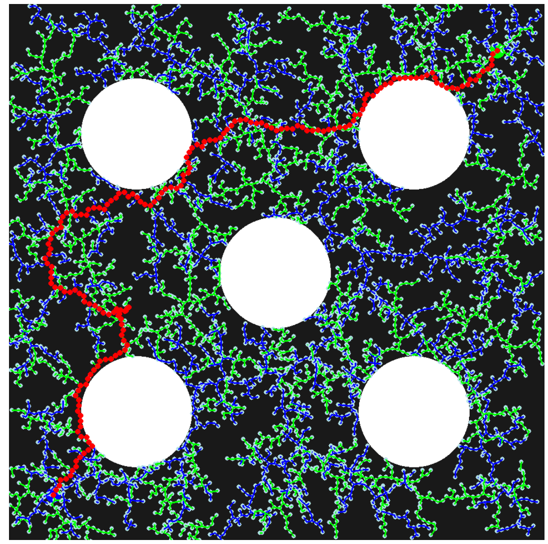


Fig. 6. RRT Connect

- **Improved RRT Connect:** This algorithm gives an optimised path and this starts once RRT has already found the path. Due to this, the time taken by Improved RRT Connect is much more than RRT and RRT Connect. In figure 7, the yellow coloured line shows the path formed by Improved RRT Connect and the red coloured line shows the path formed by RRT Connect. Figure 8 shows the results from [1]. On comparing, it can be seen the our results follow a similar pattern. We were able to achieve an optimised path using Improved RRT Connect.

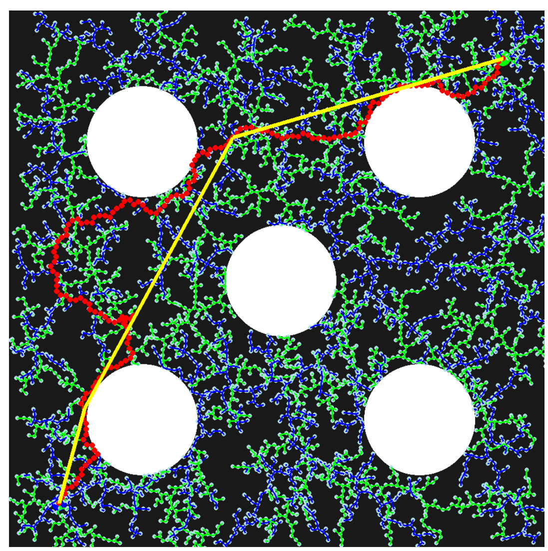


Fig. 7. Improved RRT Connect

- **RRT Connect Biased:** Since this variant is extremely biased, it consistently generates nodes in the direction of the other branch, until an obstacle comes in the way. Hence, the exploration is efficient and it can quickly find the solution path between the Start and Goal nodes.

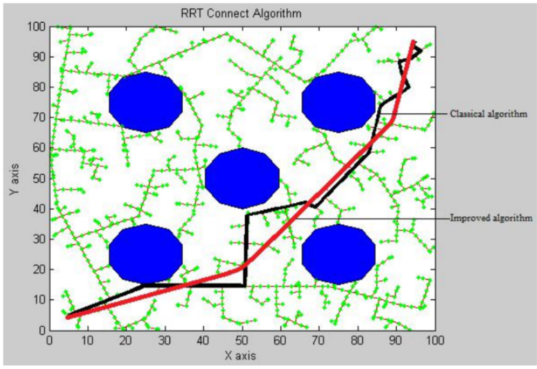


Fig. 8. Improved RRT Connect Simulation Results from [1]

Figure 9 shows the path generated by RRT Connect Biased.

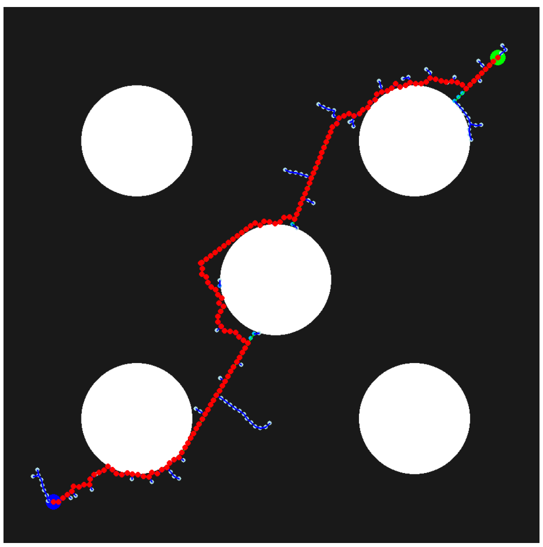
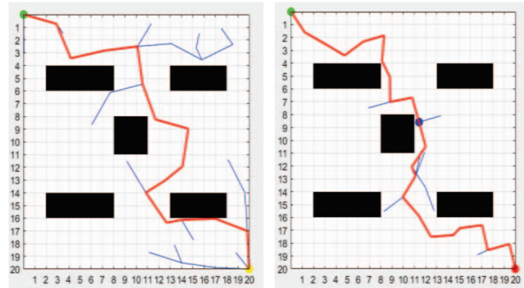


Fig. 9. RRT Connect Biased

- **RRT - Connect - New Map:** The researchers of GT-RRT implemented the code on a different map than the one used in the research of Improved RRT- Connect. We decided to use the same map to present the results of GT-RRT and RRT Connect. This can be seen in Figure 11, 12 and 13.
- **RRT - Connect - 3 Node:** A fundamental part of GT-RRT is the addition of a third exploration node. Just the implementation of this feature can be seen in figure 14. The result is already an improvement on RRT- Connect from figure 11. Although the search is still randomized, the path found is a little more optimized since the algorithm makes the path pass the estimate center of the goal and node.
- **GT - RRT:** We wanted to compare our work to the simulation results of the researchers who's work inspired us to explore the concept of GT-RRT. The results can be seen in Figure 10. Figure 13 shows our result for GT-RRT. In terms of path generated, the result was indeed



(a) RRT-connect algorithm (b) GT-RRT algorithm

Fig. 10. GT-RRT Simulation Results from [2]

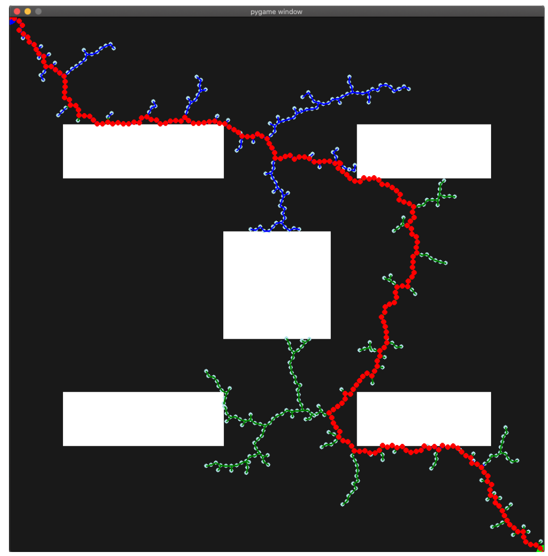


Fig. 11. RRT Connect - 2 Node

the most optimized. You can also refer to Figure12 to see how and where the start node branch (green), third node branch (yellow) and the goal node branch (purple) interact and connect. Due to the gravitational function, the node search is extremely well directed and not biased. Hence, the exploration is also efficient, making it a fairly fast algorithm. We did not have any quantifiable data to compare, but we were happy to see similar patterns between our output and the simulation results from the paper [2].

- **Potential Function:** Figure 15 and Figure 16 show how altering the epsilon (ε) in the Gravitational Function - the gravitational coefficient (ε) - affects the sampling method. We used an empty map to present these results. As the gravitational coefficient is larger, the gravitational force towards each other is higher. Hence, as can be seen in Figure 15, the exploration is extremely biased. On the other hand, however, as the gravitational coefficient is reduced, the node generation is a little more random and therefore the wavering branches can be seen in Figure 15.

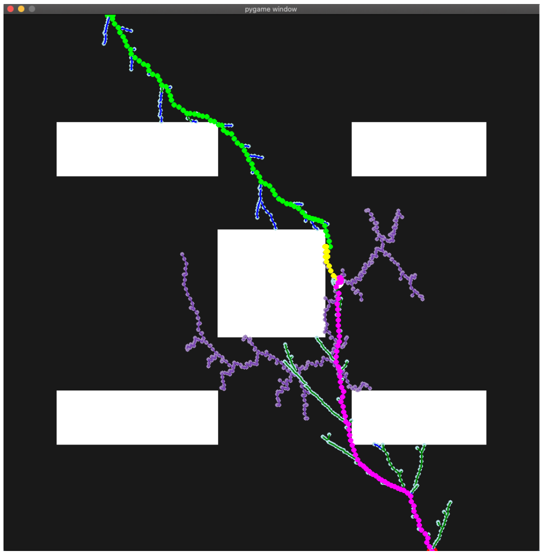


Fig. 12. GT-RRT - Individual Branch Links

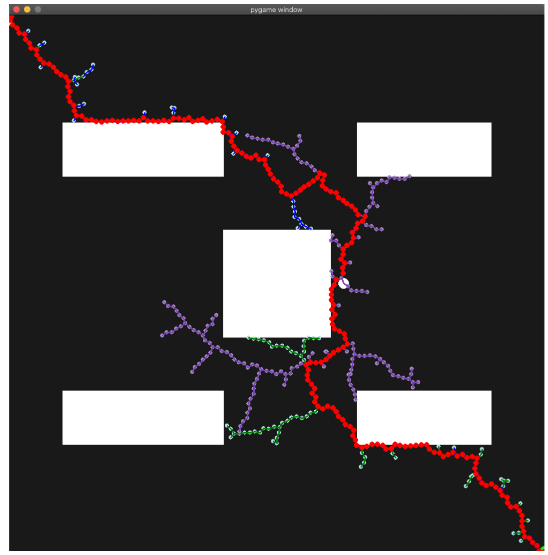


Fig. 14. RRT Connect - 3 Node

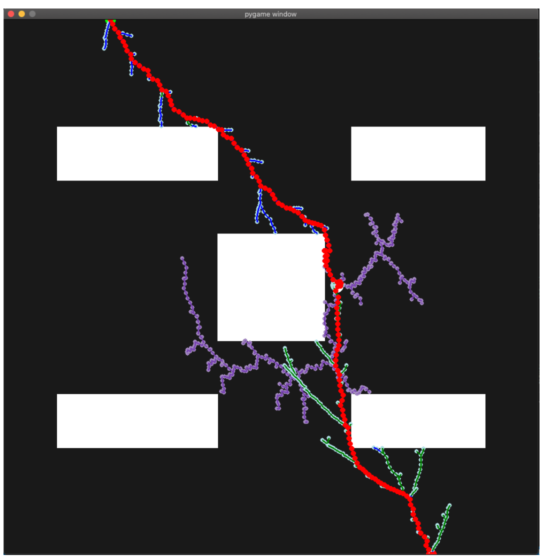


Fig. 13. Resultant Path for GT-RRT

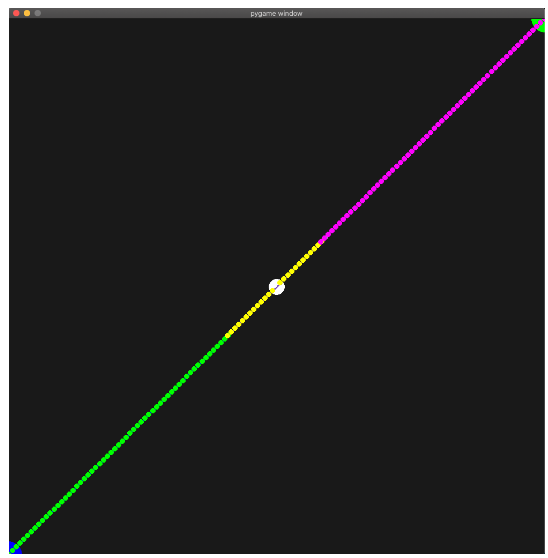


Fig. 15. GT-RRT - High Gravitational Coefficient (ε) on an Empty Map

VI. Conclusion

From the table in the section V, it can be seen that RRT-Connect Biased takes the least amount of time in reaching the goal but as it does not sample enough way points, it should not be considered as the best option. Next, RRT Connect takes lesser time as compared to the standard RRT because RRT Connect starts generating nodes from two points, start and goal, at the same time. Improved RRT Connect starts once RRT Connect has found the path so it takes more time as compared to RRT Connect. But the benefit of Improved RRT Connect is that it gives an optimized path, i.e. giving the shortest path. This leads to an interesting question of planning efficiency and path optimization. The Improved RRT-Connect will give a much shorter path than RRT-Connect given a little more time to run the algorithm.

The aforementioned trade off is well addressed by GT-RRT algorithm. The three-source fast expansion random tree strategy, combined with superimposing a gravitational field on each node, does a good job in improving both the exploration efficiency and path optimization. As can be seen in the table in the section V, compared to RRT-Connect, GT-RRT is faster and gives a shorter path. Overall, the implementation and analysis of the two enhanced variants of RRT taught us several techniques to make a basic RRT algorithm even better; this includes, adding new source for branching, superimposing gravitational field and drawing straight lines for unobstructed nodes on the path.

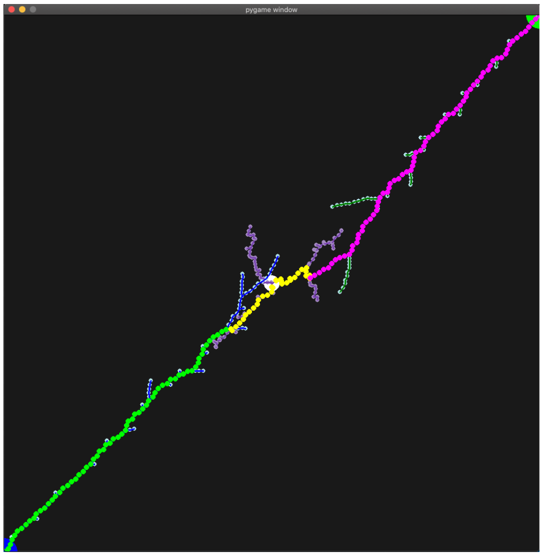


Fig. 16. GT-RRT - Low Gravitational Coefficient (ε) on an Empty Map

References

[1] S. Li, D. Zhao, Y. Sun, J. Yang and S. Wang, "Path Planning Algorithm Based on the Improved RRT-Connect for Home Service Robot Arms," 2021 IEEE International Conference on Intelligence and Safety for Robotics (ISR), 2021, pp. 403-407, doi: 10.1109/ISR50024.2021.9419385.

[2] C. Zhao, X. Ma and C. Mu, "Research on Path Planning of Robot Arm Based on RRT-connect Algorithm," 2021 33rd Chinese Control and Decision Conference (CCDC), 2021, pp. 3800-3805, doi: 10.1109/CCDC52312.2021.9602848.

[3] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), 2000, pp. 995-1001 vol.2, doi: 10.1109/ROBOT.2000.844730.

[4] Robotic Motion Planning: RRT's - Carnegie Mellon School of Computer Science (https://www.cs.cmu.edu/ motionplanning/lecture/lec20.pdf)

[5] Robotics Online Marketing Team — 12/04/2018. (n.d.). Overview of industrial robot path planning. Automate.(https://www.automate.org/blogs/introduction-to-industrial-robot-path-planning)