AER 1516: Robot Motion Planning

Assignment # 2 Planning in Practice

By: Alexis Bruneau

Presented to Dr.Jonathan Kelly

March 15th, 2023

Contents

Table of Figures

Question 3 Discussion

In this assignment, two motion planning algorithms, RRT (Rapidly-exploring Random Tree) and RRT* (RRT with optimization), were implemented to navigate from a start node to a goal node while avoiding obstacles, following Dubins paths. To assess the performance of both algorithms, they were simulated 1000 times in a challenging environment.

The analysis revealed that RRT* has a lower overall path cost compared to RRT, but at the expense of increased computational time. The average cost for RRT was approximately 50, while the average cost for RRT* was 32, indicating a more optimal path. However, the average computational time for a single RRT* simulation was around 0.3 seconds, whereas the average time for an RRT simulation was approximately 0.06 seconds, making RRT* about five times slower.

This performance difference can be attributed to the additional steps in the RRT* algorithm, which involves rewiring the tree to find more optimal paths. The rewiring process requires updating parent nodes, reconfiguring the tree structure, and recalculating the costs of children nodes within a defined neighborhood. While this leads to more optimal solutions, it also results in a higher computational cost.

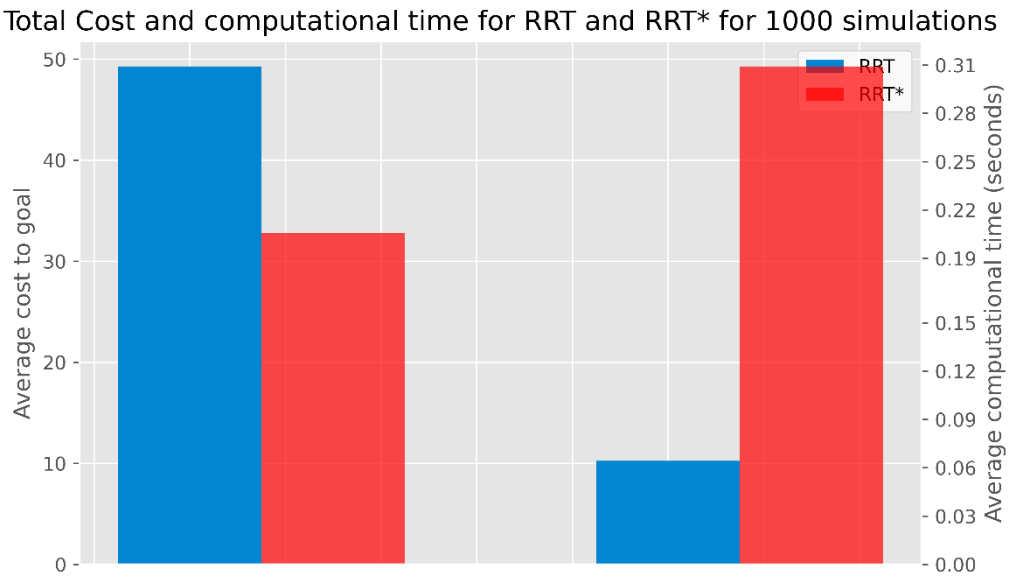

Figure 1 Total Cost and Computational time for RRT and RRT* for 1000 simulations

The subsequent figures illustrate the paths generated by RRT and RRT* algorithms from the start to the goal for the last five simulations. Upon visual inspection, it is evident that the paths generated by RRT* are shorter compared to those generated by RRT, as the former avoids unnecessary loops and follows a more optimal route.

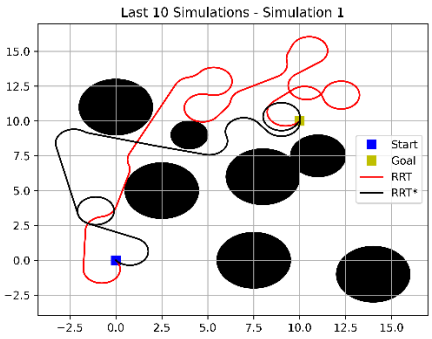

Figure 2 RRT vs RRT simulation 1*

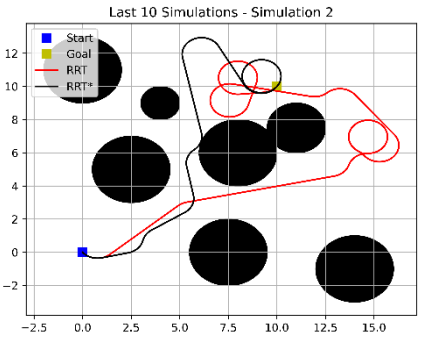

Figure 3 RRT vs RRT Simulation 2*

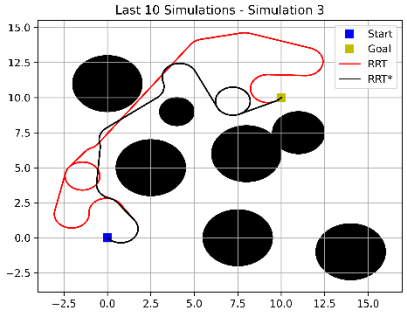

Figure 4 RRT vs RRT Simulation 3*

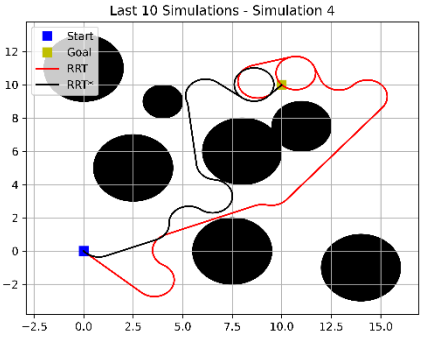

Figure 5 RRT vs RRT Simulation 4*

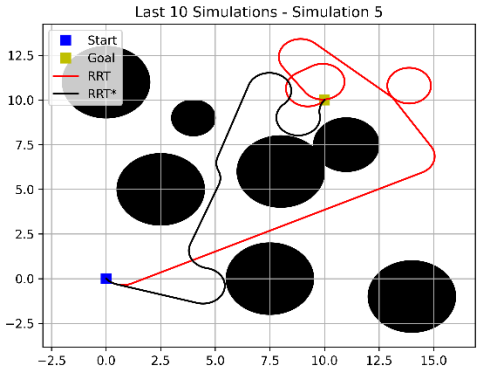

Figure 6 RRT vs RRT Simulation 5*

The next image displays the results of RRT and RRT* for a single simulation, showcasing the nodes generated during the process. Figure 8 highlights that, although not being the most optimal path, RRT* still outperforms RRT in terms of path optimality.

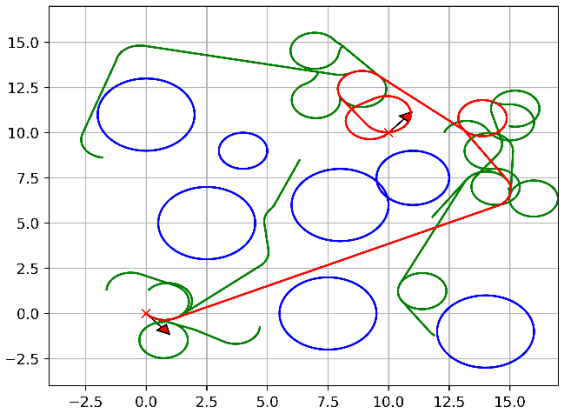

Figure 7 RRT with all nodes

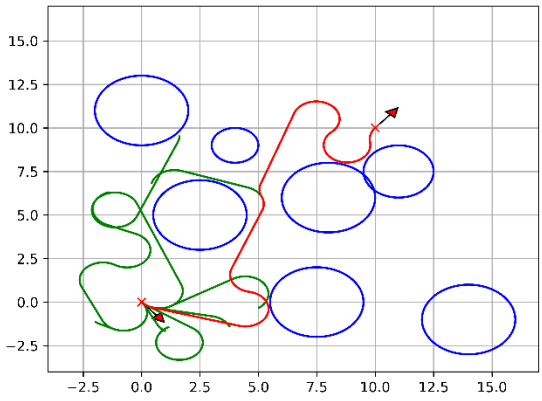

Figure 8 RRT with all nodes*

As previously mentioned, RRT* produces more optimal paths than RRT but does not guarantee the most optimal solution. Increasing the number of nodes should theoretically improve the trajectory cost, but it would also raise the computational cost due to the increased number of neighboring nodes to process. In an ideal scenario with an infinite number of points, the path would be optimal. In the following example, 30 iterations were executed without clearing the node list. This experiment led to a reduced total cost for RRT*, albeit with a significant increase in computational time. The subsequent images present the results of the average cost and computational time for RRT versus RRT* when the Dubins path list was not cleared after simulations. The average cost of RRT* is now approximately half of that of RRT, but the computational time has increased nearly tenfold compared to previous results. From Figure 11, it can be observed that RRT* produces a path that is almost optimal.

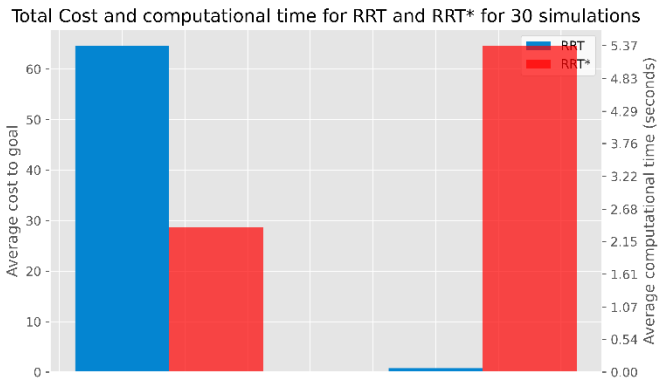

Figure 9 Total cost and computational time RRT vs RRT when dubbins list is not cleared*

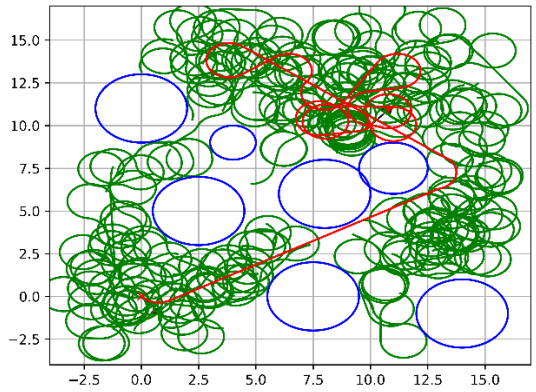

Figure 10 RRT (30 simulations without clearing dubins_list)

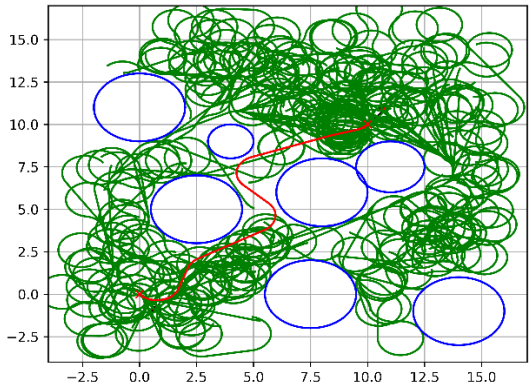

Figure 11 RRT (30 simulations without clearing dubins_list)*

In conclusion, this study has demonstrated that both RRT and RRT* algorithms exhibit distinct advantages and disadvantages. On average, RRT finds a solution more quickly, but the resulting path is longer. In contrast, RRT* takes longer to compute but yields a less costly and more optimal path.

Potential applications where RRT may prove advantageous include scenarios where quick and approximate solutions are desirable, such as exploratory robotics, navigation in dynamic environments, or preliminary path planning. On the other hand, RRT* may be more suitable for applications requiring optimized and efficient paths, such as automated vehicles, precision robotics, and long-term path planning for energy conservation. Ultimately, the choice between RRT and RRT* depends on the specific requirements and constraints of the application in question.