HW 1 Report

William Hubert

February 11, 2025

1 Question 1 3D RRT

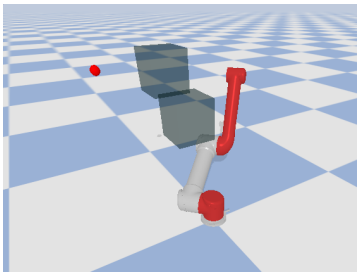

Figure 1: Robot Initial Position

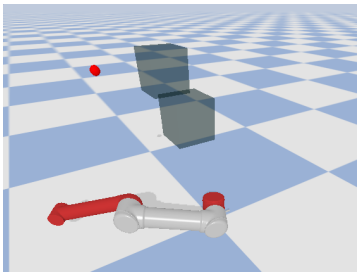

Figure 2: Robot Intermediate Planned Position

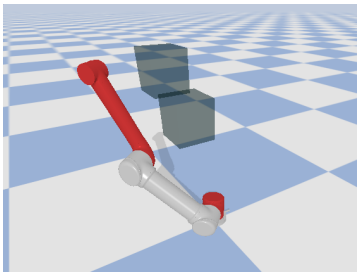

Figure 3: Robot Final Goal Position

The above is one of the paths planned by 3D RRT, in this case it planned a pretty optimal route, but often the route is very odd with some meandering intermediate steps.

2 Question 2 RRT* in 2D Environment

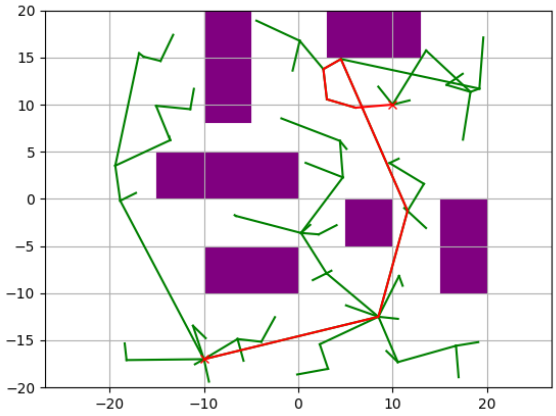

Figure 4: RRT

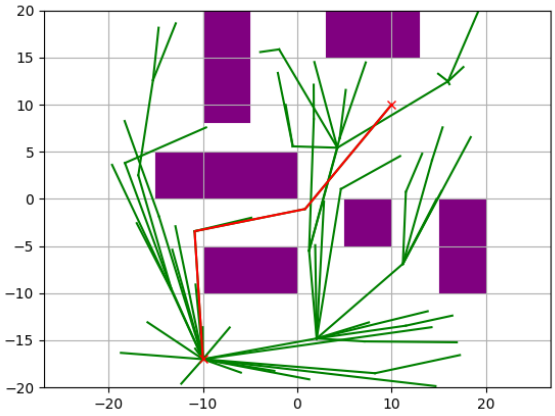

Figure 5: RRT*

Table 1: Comparison of RRT and RRT*

Algorithm	Time per Iteration(s)	Avg Cost
RRT	0.024	53.86
RRT*	0.612	38.65

As seen in the figures above, both algorithms successfully find paths to the final destination. However, the RRT algorithm simply finds a path without any post-processing optimization, unless a more optimal path is randomly selected. In contrast, the RRT* method produces a much more direct path by incorporating the ability to search for a more optimal route. However, this improvement comes at the cost of significantly increased computational complexity.

3 Question 3 RRT* in 2D Environment with Circular Robot

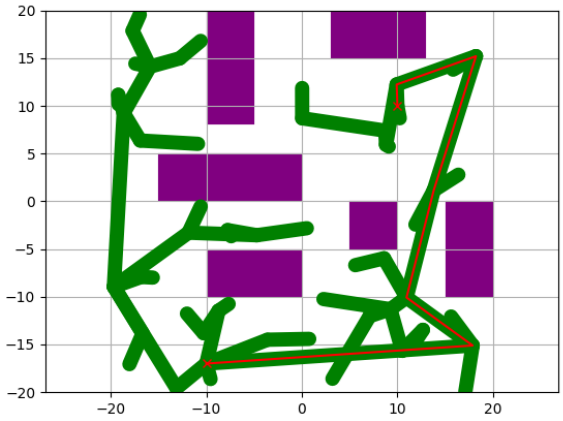

Figure 6: RRT

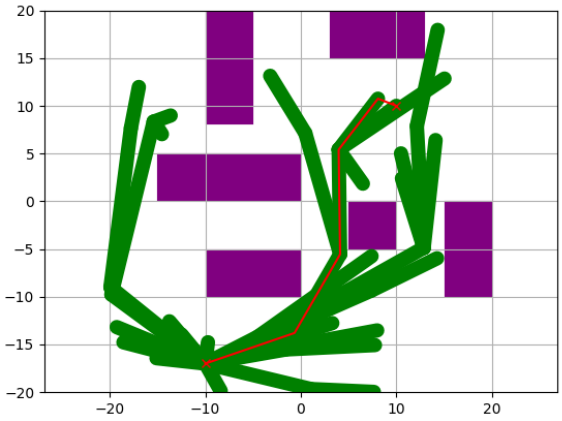

Figure 7: RRT*

Table 2: Comparison of RRT and RRT* Performance with Circular Robot

Algorithm	Time per Iteration(s)	Avg Cost
RRT	0.024	54.49
RRT*	0.502	39.55

When we change the geometry of the robot in path planning, the performance values remain relatively similar to those obtained with a point mass moving through the space. If we visualize this as a physical robot, we can see that RRT may generate unusual paths, leading to strange robot movements that a human might not typically perform. While this is not always a disadvantage, it can be considered inefficient when trying to find the most optimal path. RRT* produces a much smoother and more efficient path but requires significantly more computation time.

4 Code Running

4.1 Part 1

-The code for Part1 runs exactly as described in the document, simply run:

```
$ python3 assignment1_part1_3d.py
```

4.2 Part 2

-The Code For Part2 has been modified with the code commented out that allows it to run 30 times and calculate the table values, right now it is setup in the original configuration.

For RRT

```
$ python3 assignment1_part2_2d.py -alg rrt
```

For RRT*

```
$ python3 assignment1_part2_2d.py -alg rrtstar
```

4.3 Part 3

-This code is in a seperate file compared to the part 2 version under a new folder labeled:

```
assignment1_part3_2d
```

-Similar to Part 2 the code was modified with the code commented out to produce the table values, running, otherwise it was modified to visualize a larger moving path and greater collision radius.

For RRT

```
$ python3 assignment1_part3_2d.py -alg rrt
```

For RRT*

```
$ python3 assignment1_part3_2d.py -alg rrtstar
```