CS593
Shivam Bhat
bhat41@purdue.edu
PUID:0033760929

Instructions:

Part 2
For **rectangle the dof is to be set to** 3 -> X,Y,Theta

Part 1
For RRT the stopping condition is when it reaches the goal region which is <0.5 angle in C-Space. I was informed on Piazza that This is a hyper-parameter and needs to be set by us. With 0. It converges in 80% runs.

It stops when inside the Goal Region and not necessarily exactly at goal as stated in Pseudo code. I haven't added the final goal into my path if it finds the path till goal region but not goal to show that exactly(Has reached goal region and not goal, if and when applicable)

BiRRT Smoothing - Code is implemented and correct but while running the visualizer seems to get stuck

Q1,2,3)

RRT:
Pros:

1. We can see that it's capable of finding a solution in a probabilistic sense, even for very complex problems
2. Its relatively simple and intuitive to understand
3. Canefficiently explore high-dimensional configuration spaces

Cons:

1. The SteerTo function currently used as described in the pseudocode just returns true/false and not the last collision free point along the path so it loses out on some potential progress
2. The current pseudocode doesn't move ahead in small step sizes since it inserts Qrand into the tree, not the point lying step_size ahead along the collision free SteerTo path. Hence the transitions ae very abrupt and not smooth

BiRRT
Pros

1. The SteerToUntil Function used in the pseudocode doesn't lose its progress as it also returns the last collision free node along the part from Qnearest to Qrand.

2. By Bi-directionally exploring the configuration space, BiRRT is more likely to find a shorter and smoother path than RRT.

3. By exploring both sides of the problem simultaneously, BiRRT is more efficient in terms of computational resources compared to other path planning algorithms.

Cons

1. BiRRT is more computationally resource intensive compared to traditional RRT algorithms, as it requires searching in both directions and maintaining two separate trees.
2. The current SteertToUntil used in pseudocode has higher complexity than SteerTo used in Also BiRRT's SteertToUntil steers in steps size of 0.05 which requires more computations and time, although may result in more precise result.

BiRRT Smoothing:

Pros

1. It gives a better and smoother path by reducing jaggedness, sharp turns, and making.

2. Smooth paths can be executed faster and more efficiently, reducing energy consumption and increasing the speed of motion. Since there are less decisions to make, there are in turn less jerks.
3. Better motion planning: Path smoothing can help improve the overall motion planning process by reducing the number of redundant or unnecessary points in the path, and making it easier to control and execute the path.

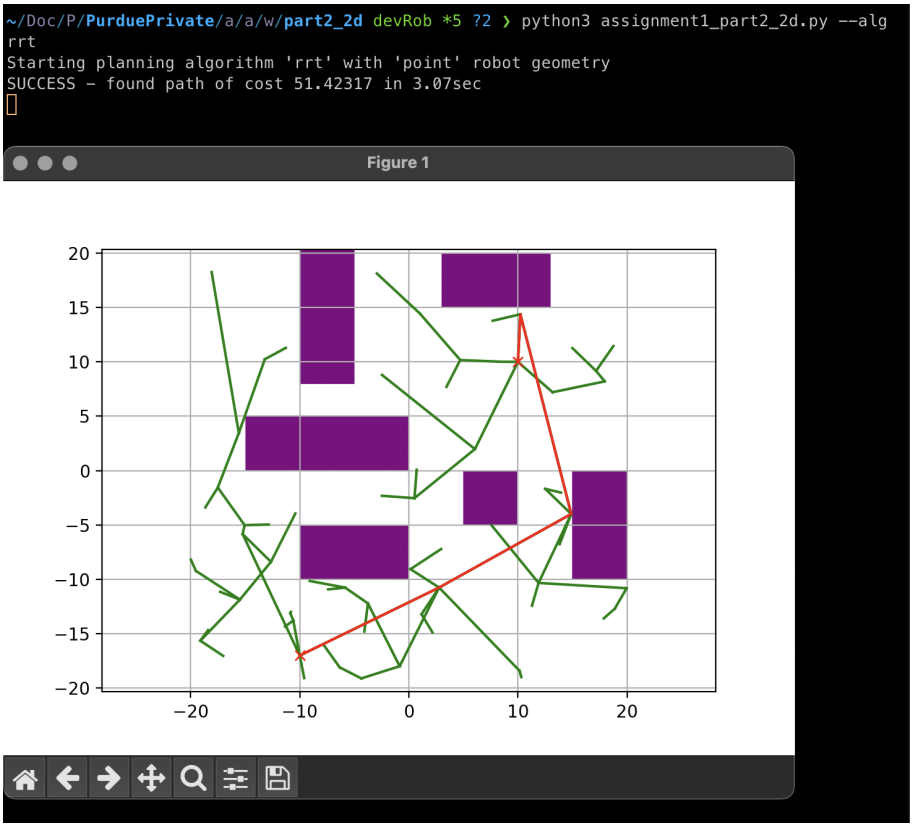
Cons

1. Smoothing requires one to again process and perform computations on already found collision free paths hence it can be computationally intensive, especially for large and complex paths, requiring significant computing resources and processing time.

2. Path smoothing can sometimes remove important information or details from the original path, leading to a loss of information or accuracy.

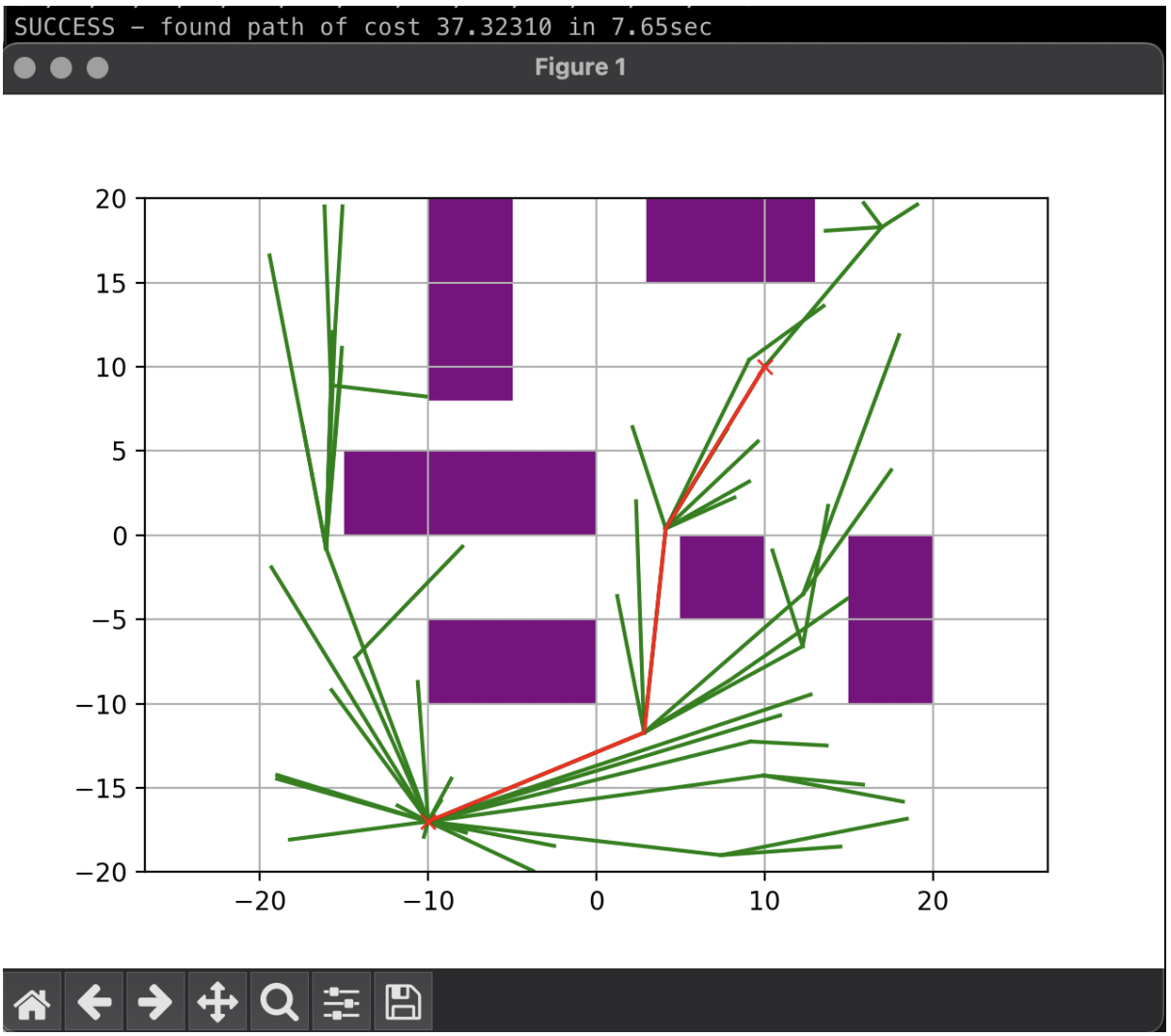
Q4)

RRT



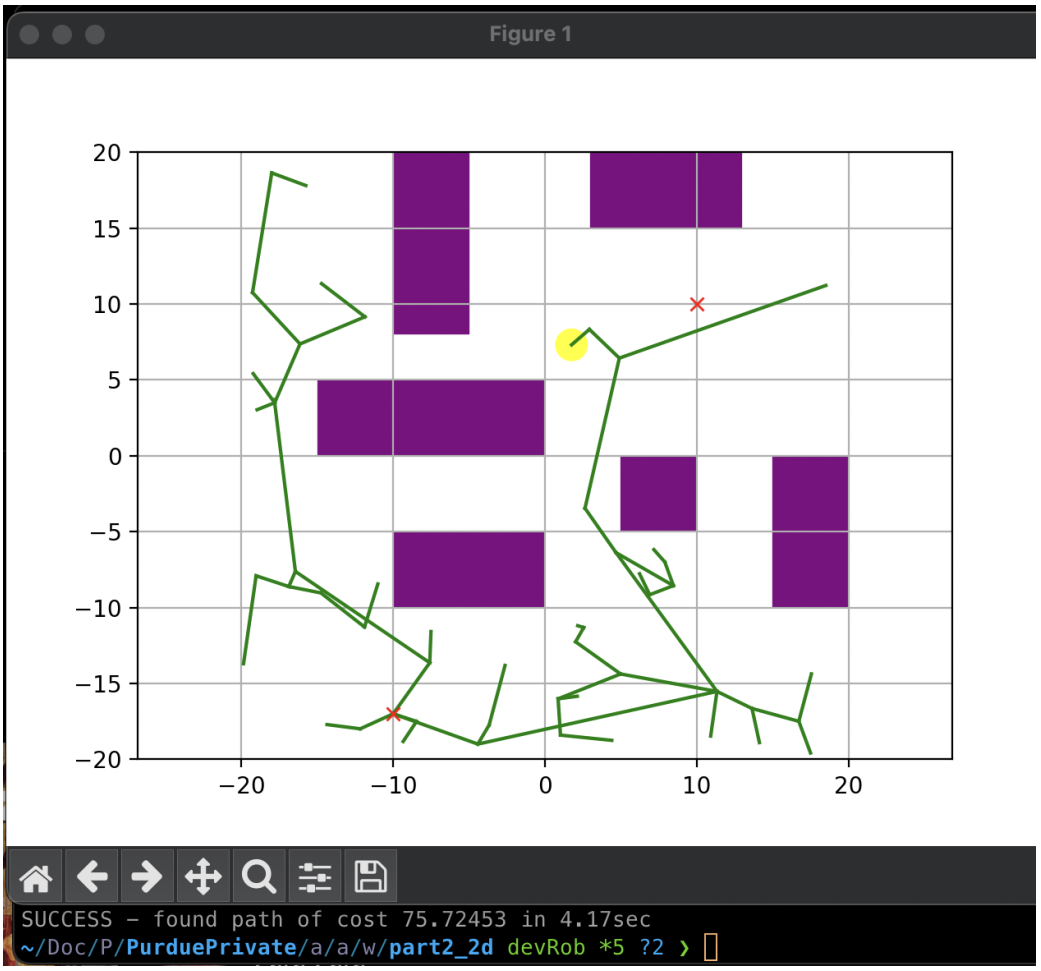
Path Cost: 51.28
Computational Time: 2.85

RRT-STAR



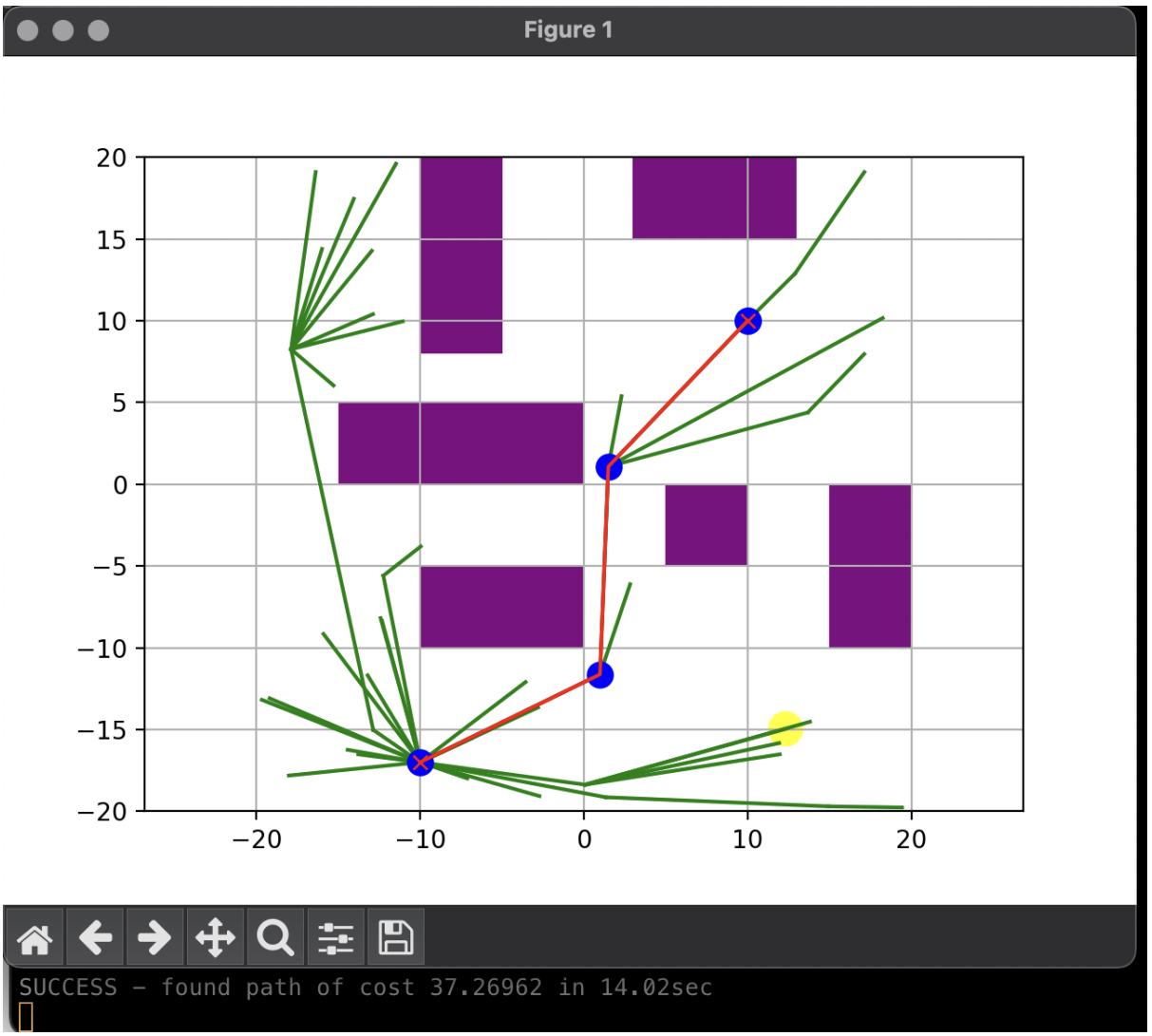
Path Cost: 38.187
Computational Time: 8.66

Q5)
RRT



Path Cost: 47.15
Computational Time: 4.35

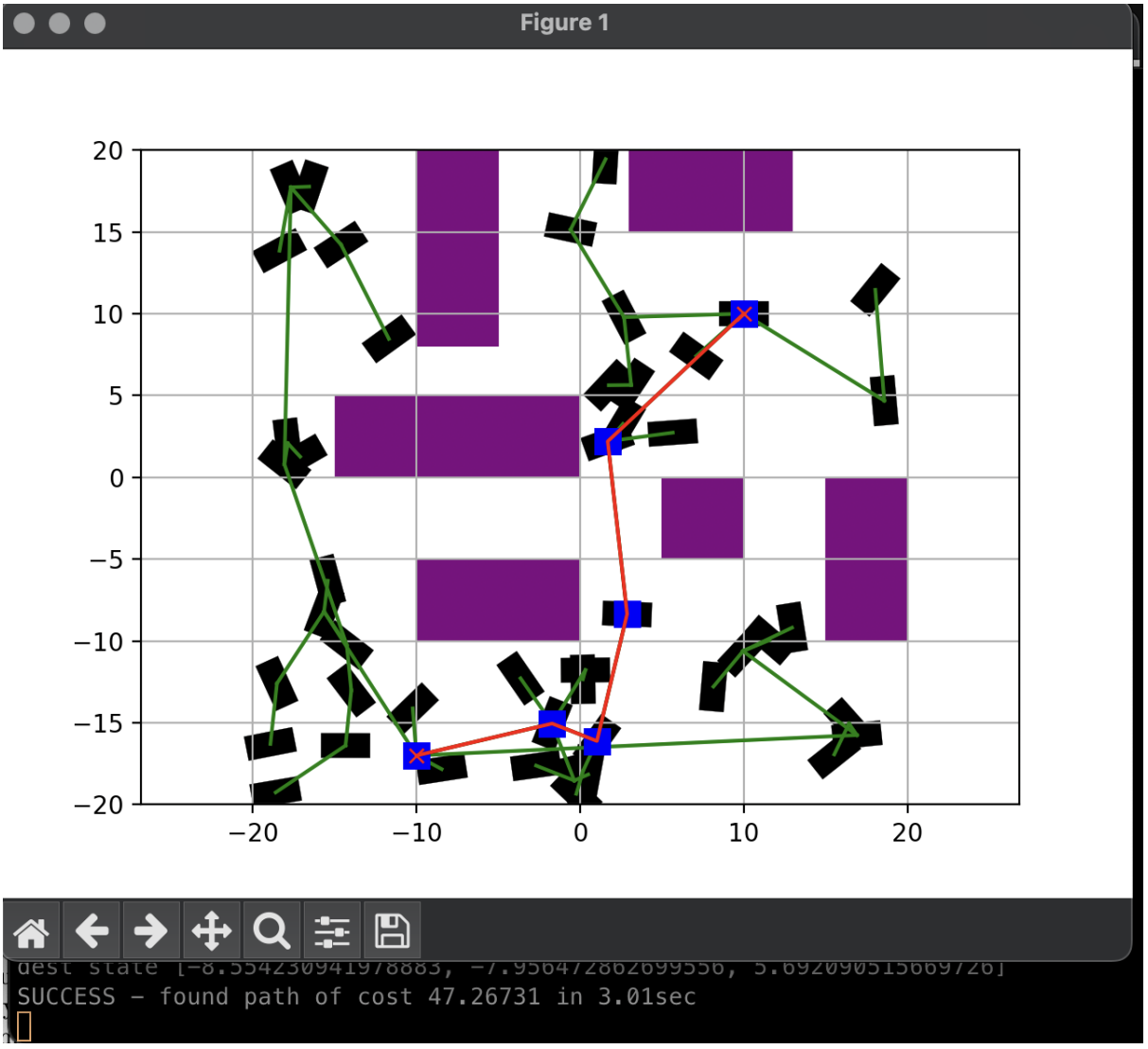
RRT-STAR



Path Cost: 39.118
Computational Time: 19.85

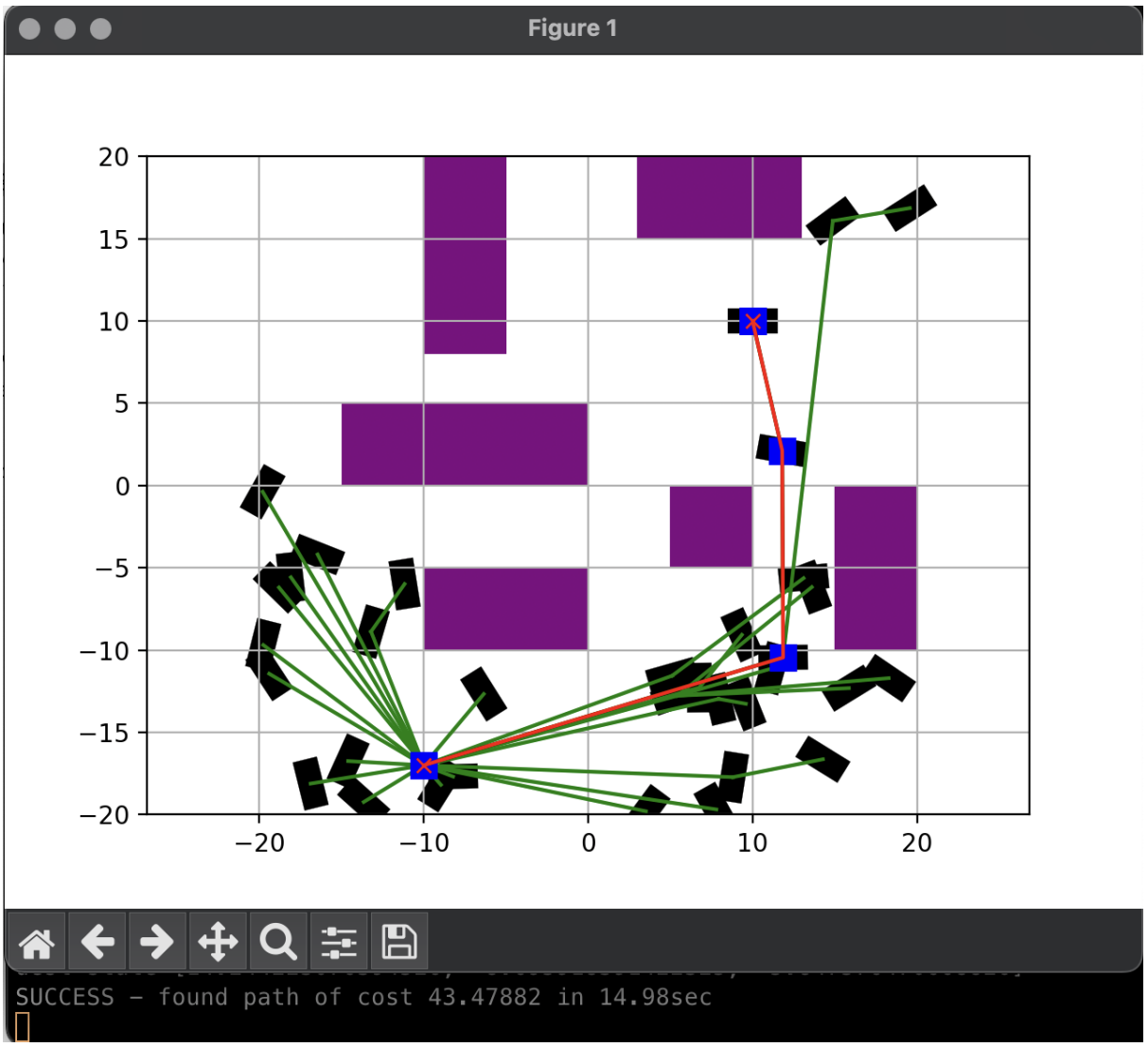
Q6)

RRT



Path Cost: 52.39
Computational Time: 2.767

RRT-STAR



Path Cost: 46.38
Computational Time: 19.9