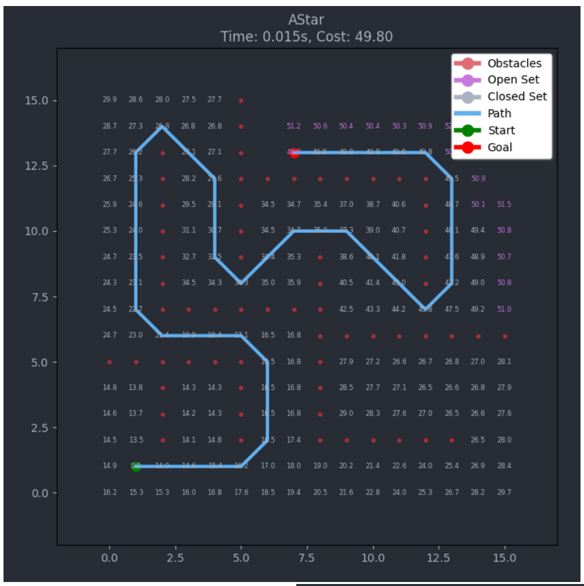
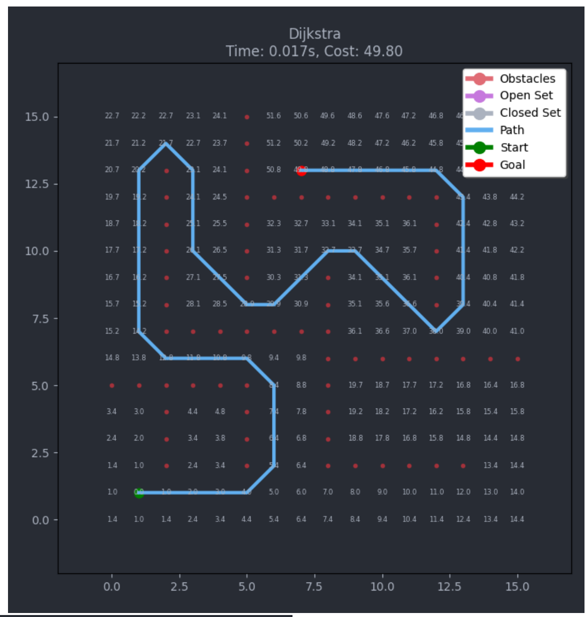
Homework 5
Christopher Seagraves

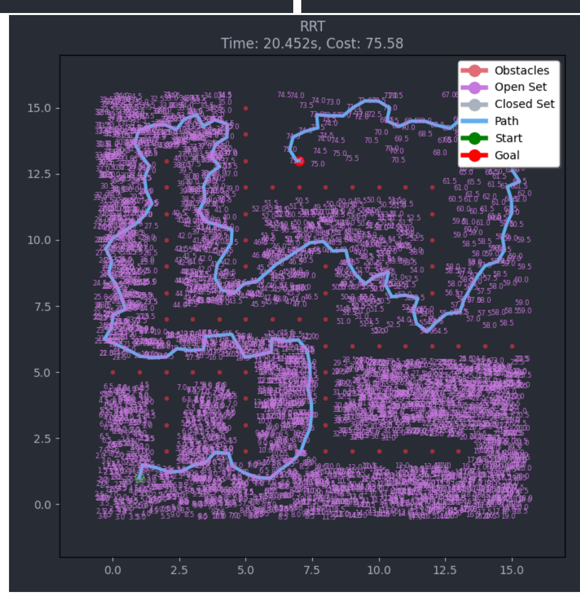
Problem 1

https://github.com/nosv1/seagraves_unmanned_systems/tree/main/SearchAlgorithms

	Time	Travel Cost
AStar	0.015	49.8
Dijkstra	0.017	49.8
RRT	20.452	75.58







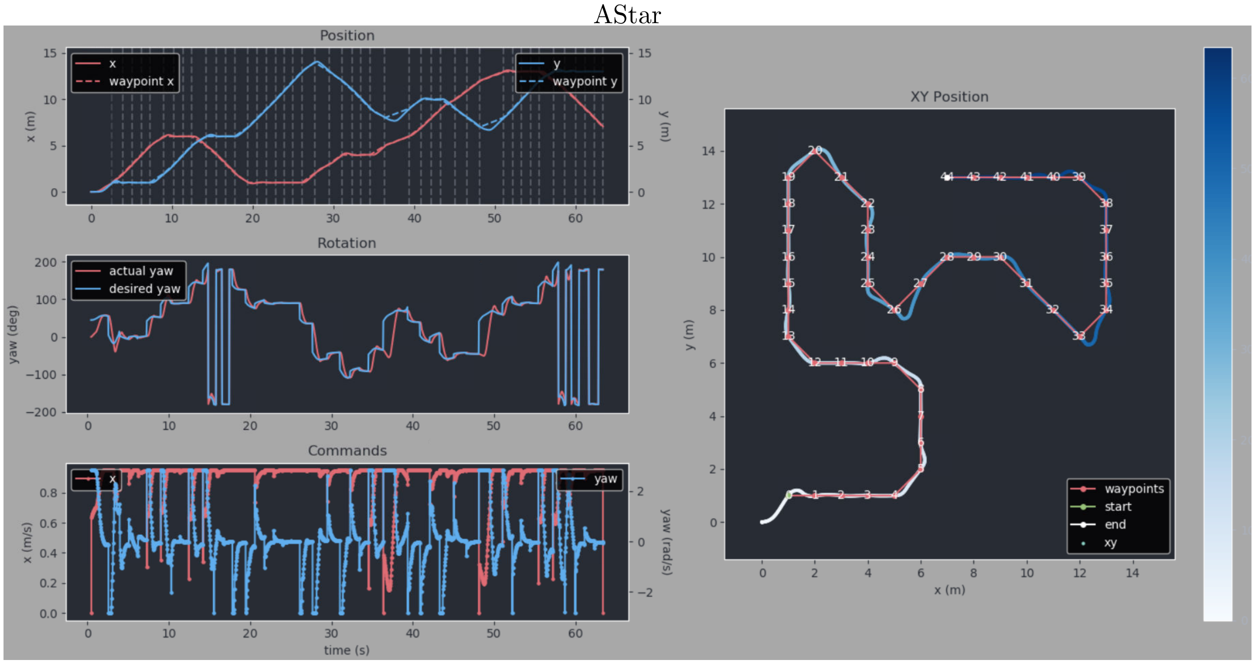
Yes, this is what I expected in terms of times and costs. AStar on a 'complex' map compard with Dijkstra will complete at similar times, and find the most effiient paths. However, RRT is still terrible on small maps in terms of time and cost.

Problem 2

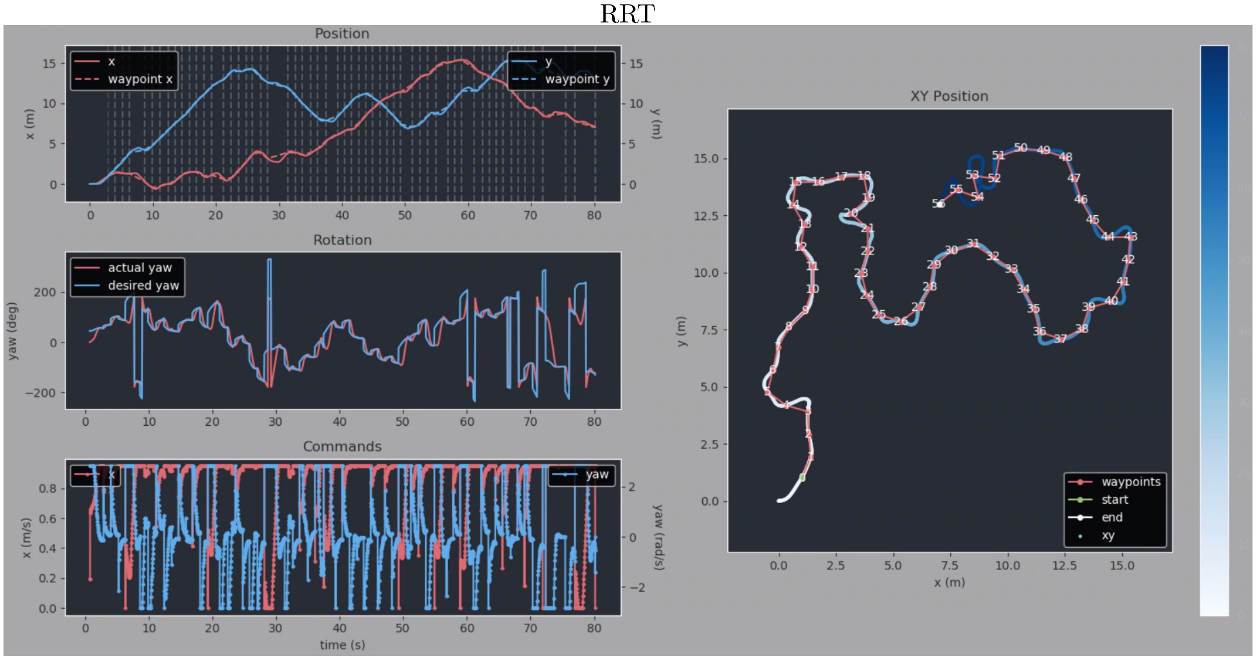
https://github.com/nosv1/seagraves_unmanned_systems_pkg/blob/master/seagraves_unmanned_systems_pkg/PathFollower/path_follower.py

The graphs below demonstrate a turtlebot set with a Vmax of 0.95 m/s and max turn-rate of 2.84 rad/s.
The turtlebot was tuned with a heading_pid and a throttle_pid.
Heading PID: Kp=4.5, Ki=0, Kd=0.25
Throttle PID: Kp=0.4, Ki=0, Kd=0.02

AStar

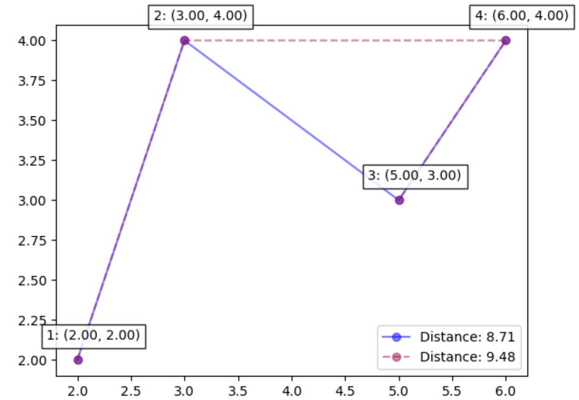


RRT

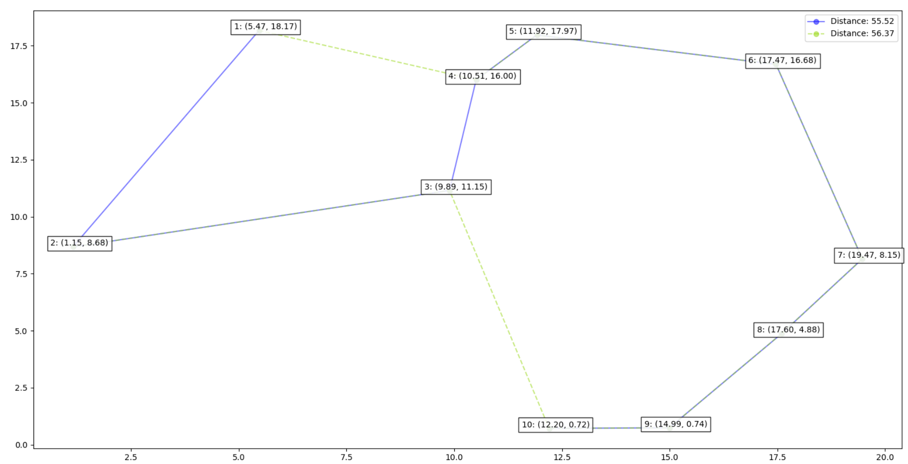


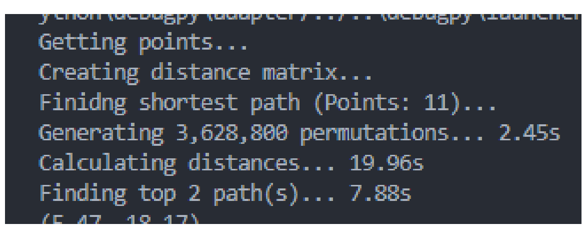
Problem 3

https://github.com/nosv1/seagraves_unmanned_systems/blob/main/HW5/TSP_main.py

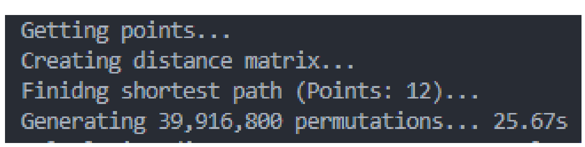


For fun, I tried to generate a more complicated path, but seemed 11 points + start was my PC's limit... Even parellizing what I could, 40 million combinations (11!) is a doozy. Given we had a non-moving start point though, a trick was to generate all the paths that don't include start, then just add the distance from start and the first point in the permutation when you calculate the distances.





```
Getting points...
Creating distance matrix...
Finidng shortest path (Points: 11)...
Generating 3,628,800 permutations... 2.45s
Calculating distances... 19.96s
Finding top 2 path(s)... 7.88s
```



```
Getting points...
Creating distance matrix...
Finidng shortest path (Points: 12)...
Generating 39,916,800 permutations... 25.67s
```

Calculating distances never completes...