Homework 2
Robot Autonomy CMU 16-662

Team members: Bhatia Abhishek (abhatia1), Babcock Eitan (eitanb), Zhang Letian (letianz)

Note: In the code for RRTPlanner and RRTConnectPlanner, there is a variable self.plot that needs to be set to '1' when the code is being run with robot as 'simple' and set to '0' when the code is being run with robot as 'herb'. This variable ensures plotting happens with simple environment.

1. **Results for 2D configuration space using RRT planner:**

	Path length	**Plan time**	**Number of vertices**
1	12.598	2.667	28
2	12.475	1.27	10
3	15.15	3.46	36
4	13.08	1.664	16
5	12.88	4.102	45
6	16.676	1.932	20
7	10.68	1.259	12
8	12.02	4.13	45
9	9.077	1.358	14
10	13.854	1.687	16
Average	**12.849**	**2.3529**	**24.2**

The goal sampling probability used was 0.2 (20%).

2. **Result for 2D configuration space using bidirectional RRTConnect planner**

	Path length	**Plan time**	**Number of vertices**
1	12.19	0.926	7
2	12.415	2.055	21
3	8.258	1.13	9

4	8.909	1.367	15
5	10.721	1.28	12
6	18.39	1.573	12
7	15.148	1.38	12
8	20.000	1.713	16
9	18.083	1.262	11
10	13.89	0.985	9
Average	**13.8004**	**1.3671**	**12.4**

The bidirectional RRT Connect planner produced similar path lengths as the regular RRT planner, but did it in about half the time with about half the number of vertices in the tree.

3. **Result for WAM arm by using the RRT planner**

	Path length	**Plan time**	**Number of vertices**
1	29.514	0.8515	158
2	17.619	2.53	324
3	23.184	0.453	61
4	16.386	5.303	471
5	19.941	0.489	82
6	31.87	1.98	204
7	19.077	1.275	229
8	10.04	0.151	33
9	12.002	1.46	236
10	12.345	1.809	39
Average	**19.1978**	**1.63015**	**183.7**

Result for WAM arm by using the bidirectional RRTConnect planner

	Path length	**Plan time**	**Number of vertices**

1	5.25	1.35	28
2	5.88	0.034	6
3	15.319	0.642	98
4	18.82	1.058	32
5	10.86	0.13	25
6	9.597	1.355	43
7	13.793	0.644	116
8	32.633	0.262	48
9	8.261	1.57	4
10	16.195	1.965	125
Average	**13.6608**	**0.901**	**52.5**

The RRT Connect planner is a significant improvement compared to the RRT planner for the WAM arm. The path length is about 71%, the plan time is about 55%, and the number of tree vertices is about 28% as compared to the RRT planner. It is more useful in higher dimensional configuration space to use the bidirectional RRT Connect planner.

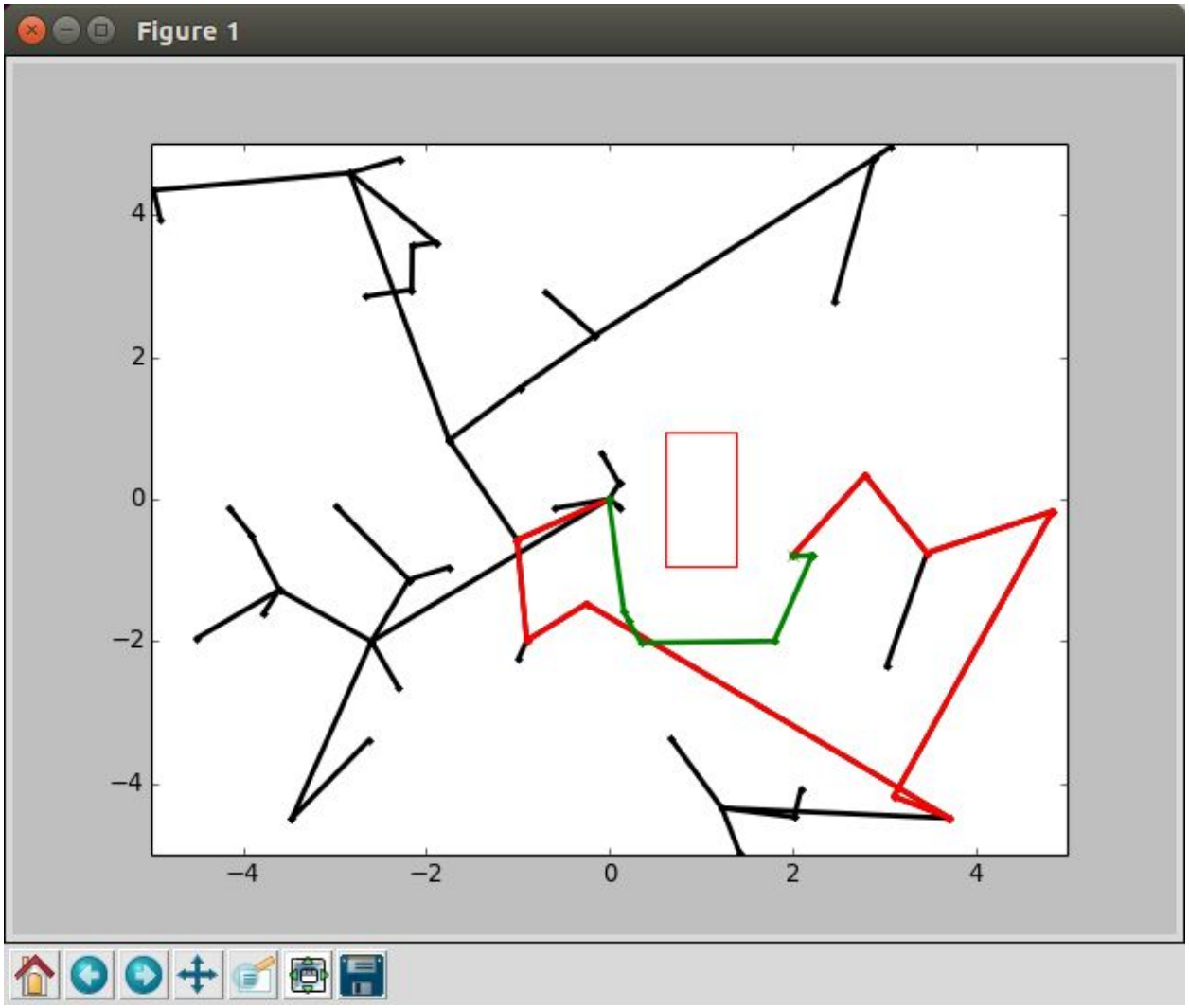
4. For shortening the path with the WAM arm, we tried three things, and none of them worked really well, we saw a lot of collision cases, and debugging them was difficult considering two things; first that the approach a) mentioned below worked really well for 2D case and we witnessed no collision, so it was strange why we saw collisions with the 7D case, and second we knew no way to visualize the output for the 7D case to see where we might be going wrong.
    a. The first thing we did was loop through the path and eliminate redundant points. If the straight line between point i and point i+2 is collision-free, then point i+1 is redundant and can be deleted. Then we added extra points along the path, and repeated the redundant point removal. When implementing this code, we saw no change in the shortened path compared to the regular path.
    b. Next, we tried moving each point a small amount closer to the goal configuration. We then checked three thing: is the new configuration in collision, is the path between this new configuration and the previous configuration in collision, and is the path between this new configuration and the next configuration in collision. If any of these three things are in collision, the point is put back to where it was. This is looped so that the path gets shortened. Again, no change was observed in the paths.
    c. Last, we tried simply checking each configuration, and if it has a linear line of sight to the goal configuration, go straight to the goal configuration. This similarly did not produce any change.

For some reason, each attempted move of a configuration thought it was in some form of collision and thus no points were moved.
Finally, we implemented b) as our final shorten path algorithm but with a minor modification of not looping it more than once, basically meant moving each point by a small amount close to the goal configuration once. This still gave us some cases with collision, but most of the cases were collision free.

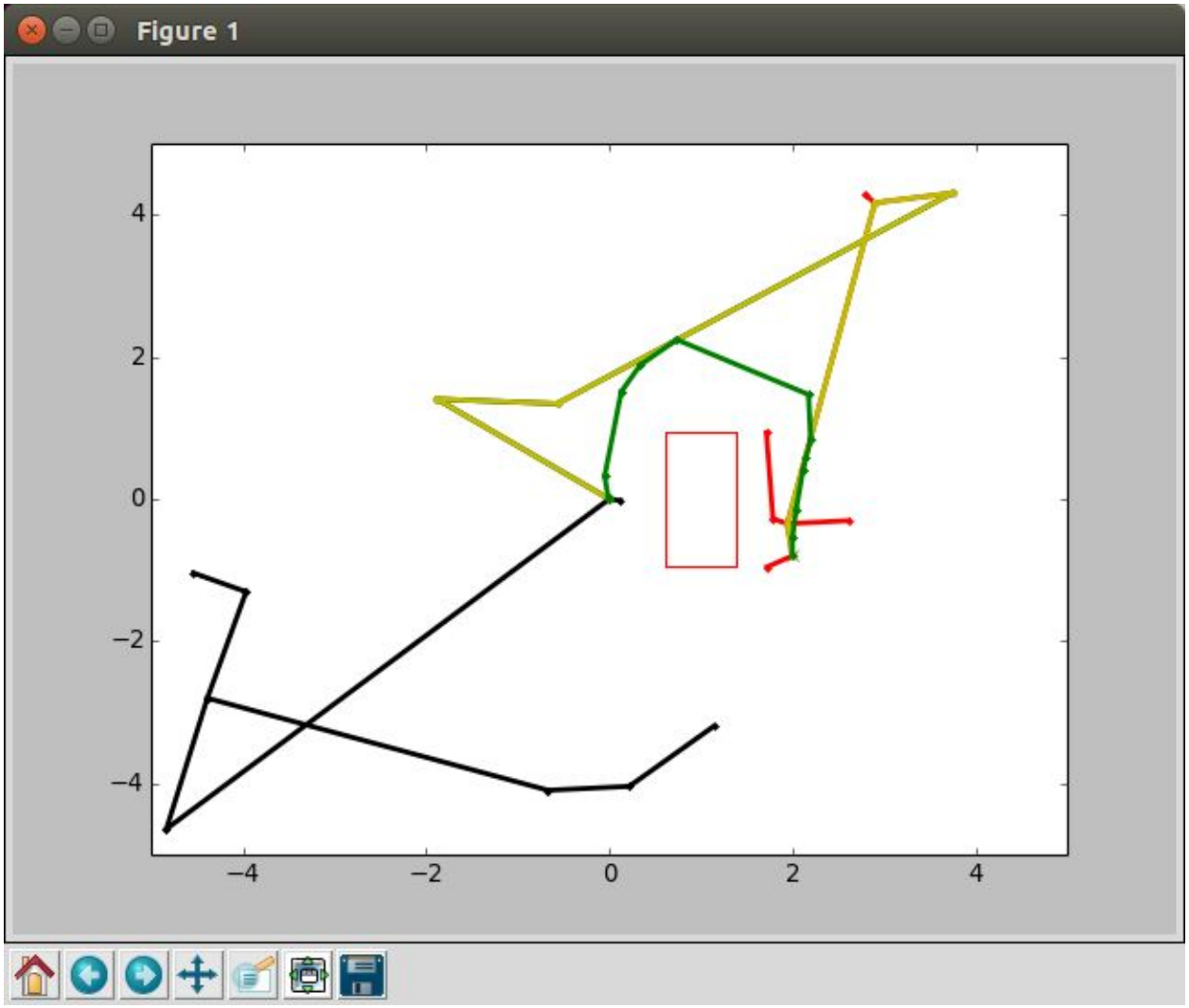
	Original Length	**Shortened length**	**Difference**
1	14.77	13.433	1.337
2	22.98	20.69	2.29
3	20.749	18.512	2.237
4	21.225	19.102	2.123
5	19.833	17.849	1.984
6	22.869	20.289	2.58
7	20.084	18.249	1.835
8	26.792	23.884	2.908
9	31.989	28.896	3.093
10	13.403	12.007	1.396
Average	**21.4694**	**19.2911**	**2.1783**

The path was shortened by an average of about 2m.

For shortening the path in the simple case, the first thing we did was loop through the path and eliminate redundant points. If the straight line between point i and point i+2 is collision-free, then point i+1 is redundant and can be deleted. Then we added extra points along the path, and repeated the redundant point removal. This smoothed out corners that were far away from the table.



The above image is for the RRT planner. The red trace is the original path calculated by RRT, and the green is the shortened path.



The above image shows similar results for the RRT Connect planner. The yellow is the path originally chosen, and the green is the shortened path.

Path Shortening for RRT Connect planner on simple robot.

	Original Length	**Shortened length**	**Difference**
1	15.247	5.65987	9.58713
2	9.0717	4.2458	4.8259
3	6.6286	4.348	2.2806
4	17.226	5.955	11.271
5	14.997	6.372	8.625
6	20.542	6.48	14.062

7	9.114	4.83	4.284
8	15.4055	5.3119	10.0936
9	16.202	5.1099	11.0921
10	10.459	6.391	4.068
Average	**13.48928**	**5.470347**	**8.018933**

The path was shortened by an average of about 8 meters.