Homework 1

CMU 16-782 : Planning and Decision Making in Robotics

Implementation details

1. I have implemented a simple Dijkstra planner for this homework. The planner starts from the current position of the robot and calculates the optimum g as well as the path steps to every node in the map. After this, I calculate the target node where my robot need to go and backtrack to calculate the path. All of this is done in the first time step only.

2. After the g value and the path time to every node in the map is calculated, the algorithm loops over every point in the target trajectory and picks up the point where the total cost of reaching plus waiting is minimized.

3. I have used a priority queue(open_list) of pointers to the nodes for my open list, an unordered map(openPtrList) for maintaining the pointers to all the nodes which have been visited(the key in this unordered map is obtained from GETMAPINDEX), an unordered map(closedPtrList) to maintain if I have expanded a node or not(the key in this unordered map is obtained from GETMAPINDEX).

4. **Open list** : The data structure I have used for maintaining my open_list is a priority_queue. I maintain a priority queue of pointers to the nodes, arranged according to their decreasing g-value. Since we can not access any pointer in a priority queue, after popping a pointer off the open list, I check if it has been closed or not. The node is expanded only if it has not been expanded before.
   Open list is open_list in my code.

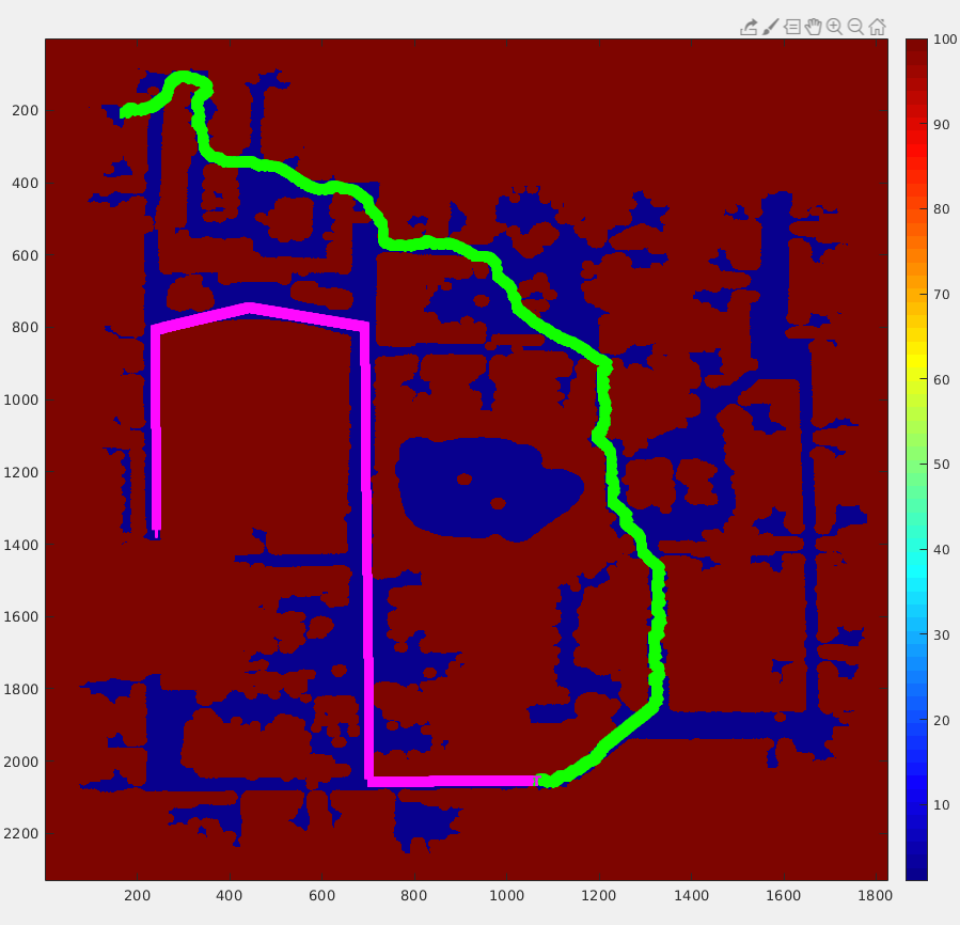
5. **Closed list** : I have used an unordered map for maintaining my closed list. This unordered map uses the GETMAPINDEX as it's key and a boolean as it's value.
   Closed list is closedPtrList in my code.

6. **Open pointer list** : I have used an unordered map for maintaining a directory of pointer to the nodes which have been visited before. This is important so that we can retrieve the pointer to nodes which have been visited. This unordered map uses GETMAPINDEX as it's key with pointer to a node as it's value.
   Open pointer list is openPtrList in my code.

7. **Actions** : Since the path is generated only once during the first time step, I have setup two static unordered map which use GETMAPINDEX as their key and corresponding action as their value. I have setup two unordered maps to correspond to x and y steps.
   Actions are actions_x and actions_y in my code.

Results

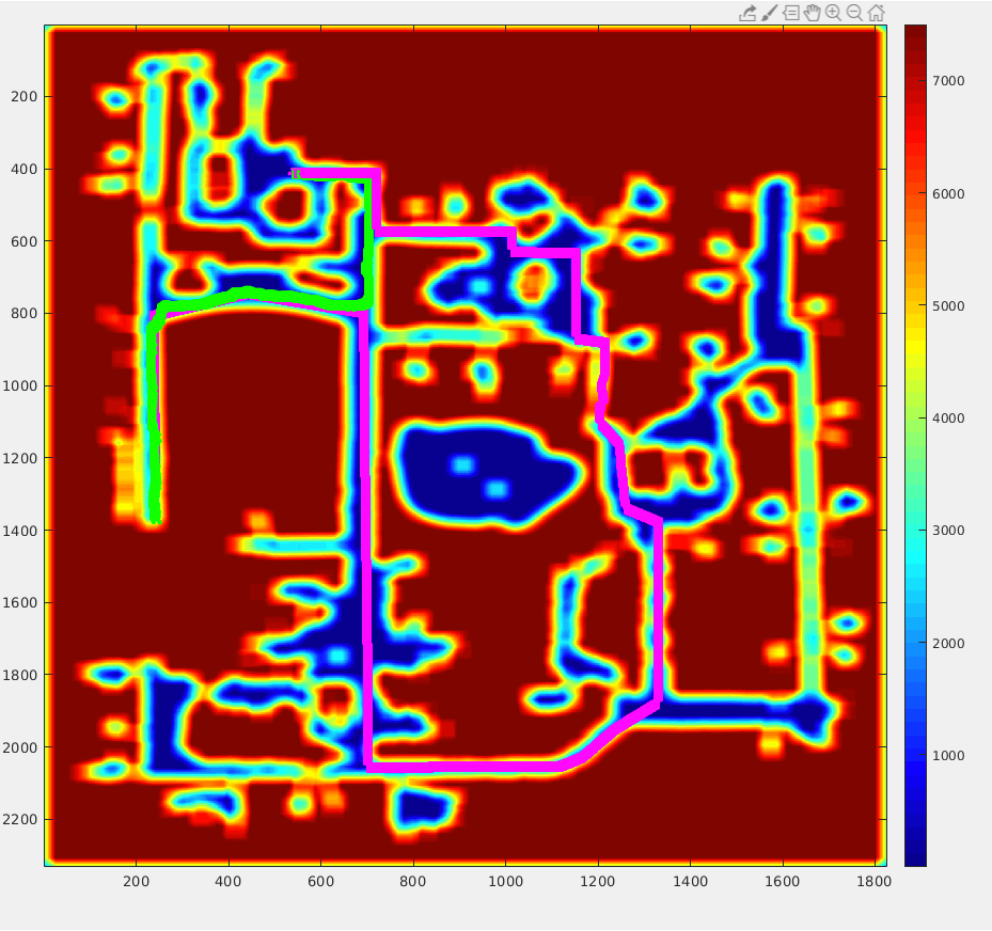
1. Map 1 :



Cost Details:

```
RESULT:
        target caught = 1
        time taken (s) = 2641
        moves made = 2639
        path cost = 2641

ans =

  logical

   1
```

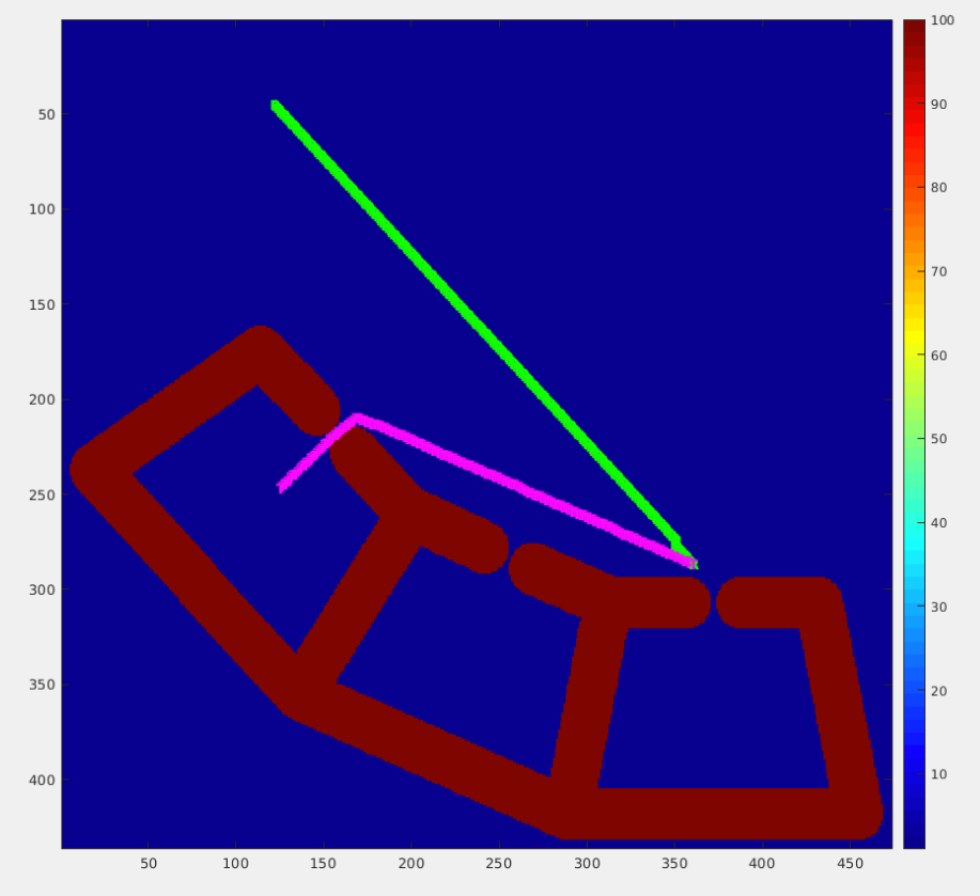
2. Map 2 :



Cost Details:

```
RESULT:
        target caught = 1
        time taken (s) = 5013
        moves made = 1528
        path cost = 2010744

ans =

  logical

   1
```

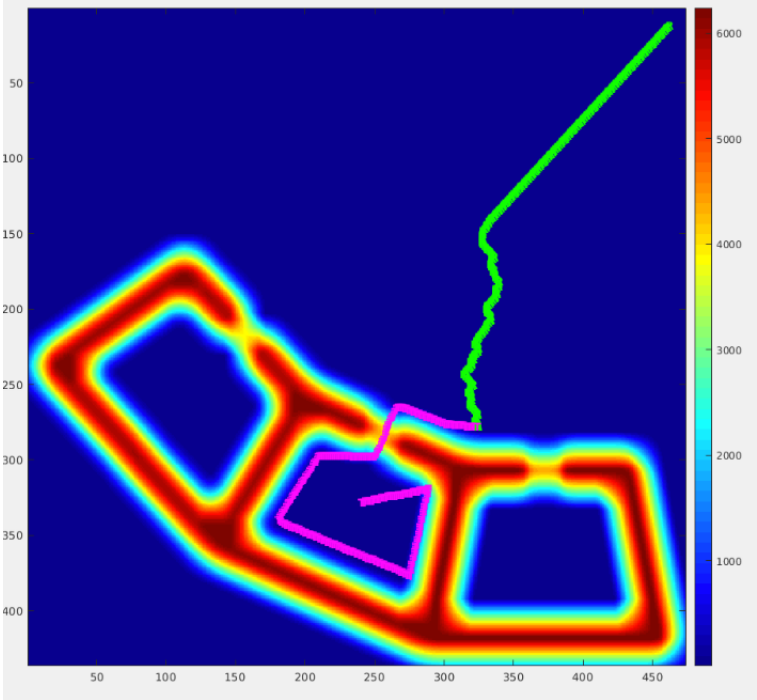
3. Map 3 :



Cost Details:

```
>> runtest('map3.txt')

RESULT:
         target caught = 1
         time taken (s) = 244
         moves made = 242
         path cost = 244

ans =

  logical

    1
```

4. Map 4 :



Cost Details:

```
RESULT:
        target caught = 1
        time taken (s) = 380
        moves made = 266
        path cost = 380

ans =

  logical

   1
```