Anthony Le
ROB 534: SDM
Winter 2017
02-01-2017 (4 PM)

HW #1: Discrete and Sampling-based Planning

Questions

1. Search-based planning
   (a) X={a,b,c,d,e}; Let K=4, x_I=a, X_G={e}
      i. Cost-to-go functions

	a	b	c	d	e
G_5*	∞	∞	∞	∞	**0**
G_4*	∞	∞	7	**1**	∞
G_3*	∞	11	**4**	2	∞
G_2*	13	**8**	5	3	∞
G_1*	**10**	9	6	6	∞

      ii. Cost-to-come functions

	a	b	c	d	e
C_1*	**0**	∞	∞	∞	∞
C_2*	∞	**2**	∞	∞	∞
C_3*	3	∞	**6**	∞	∞
C_4*	∞	5	∞	**9**	13
C_5*	6	∞	9	10	**10**

   (b)
      i. Admissible → never overestimates the optimal cost to reach the goal state
      ii. Not Admissible → overestimates the optimal cost to reach the goal state by multiplying the heuristics by 2
      iii. Admissible → never overestimates the optimal cost to reach the goal state
      iv. Admissible → least informed but never overestimates the optimal cost to reach the goal state
      v. Admissible → most informed; equal to optimal cost to reach the goal state
   (c) iv. < iii. < i. < v. → Number of tiles not in their goal state is least informed and number of moves remaining in optimal solution to reach goal state is most informed (dominate).

2. Configuration Spaces and RRTs
   (a) The space in Figure 2 can pose problems for RRTs. Explain why.
   The narrow passage in Figure 2 pose problems for RRTs because these planning algorithms involve finding paths by random sampling in continuous space, specifically in obstacle-free

configuration space (C_{free} space), but C_{free} space is limited in the passage so collision with the obstacle will most likely occur. Furthermore, random samples are more likely to sample in an obstacle which are rejected so RRTs fail to expand the search tree through the narrow passage. Thus, RRTs are less likely to efficiently and/or completely explore and expand the search tree through the passage to establish a feasible path to the goal.

(b) What would be the advantages and disadvantages of A* over RRTs?

Advantages: In discrete space, A* is guaranteed to find the optimal solution using admissible heuristics and is complete provided a finite boundary condition. Especially in a reasonably sized discrete space, A* can efficiently find the solution with the optimal path cost by evaluating the function of the cost-to-go and cost-to-come (admissible heuristics) in order to avoid expanding paths that are already expensive. On the other hand, RRTs are only guaranteed to find suboptimal solutions where random sampling is the driving method of exploring/searching the space at hand. RRTs do not pay attention to solution costs and is only concerned with providing "feasible" solutions, but is complete, can search in continuous space, and works well in complex configuration spaces. A* has control over solution quality while RRTs do not.

Disadvantages: A* cannot search in continuous space. In discrete space, A* may try all edges to guarantee optimality which can be time consuming and computationally expensive in more complex spaces while RRTs can probabilistically subsample all edges in both discrete and continuous spaces. Given a large, complex, high-dimensional configuration space, A* is especially constrained by memory as A* will have to store all nodes of the C space in an "open" list to search/plan over or even when dynamically planning, expanding neighboring nodes to evaluate costs in a high-dimensional space requires exhaustive computational efforts and an enormous amount of memory storage since A* will have consider all expanded nodes from the current node to guarantee optimality and completeness. In contrast, RRTs plan forward by randomly sampling in continuous space and store vertices and edges to build a search tree that consist of "feasible" solutions.

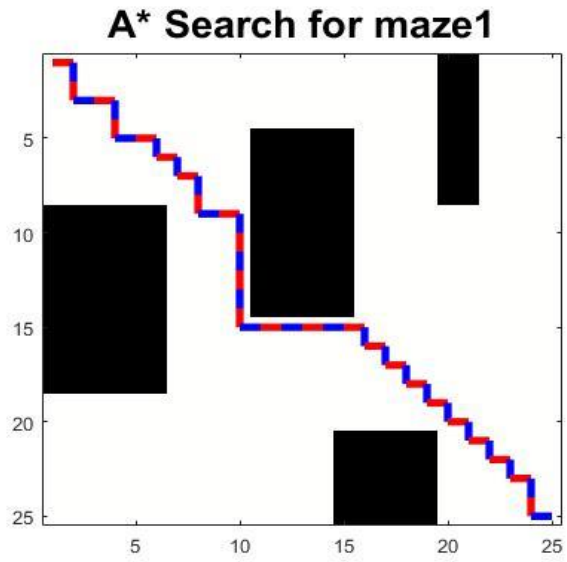
(c) How could you get around some of the disadvantages of using A*?

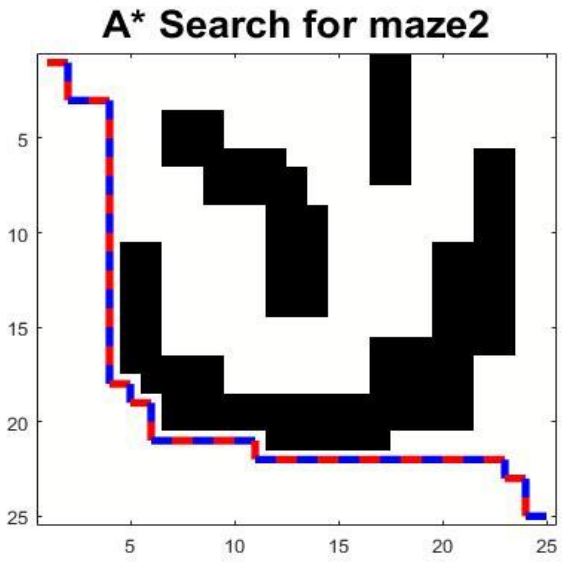
Since A* cannot search in continuous space, you can discretize the space when given a list of coordinates of the corner points of each obstacle and of the workspace as a whole for A* to plan over, avoid the obstacles, and find the optimal solution to the goal. In regards to searching in a large, high-dimensional space, A* can overcome the working memory constraint by inflating the heuristic in order to expand less nodes and only consider a narrower list of best possible actions to take and nodes to go to. However, when inflating the heuristic, A* does not guarantee optimality anymore since the heuristic is less likely admissible now and overestimates the optimal cost of reaching goal from the current node in the path.

Programming Assignment (cont'd)

Step 2

i. A* Search – Admissible heuristic: Euclidean distance

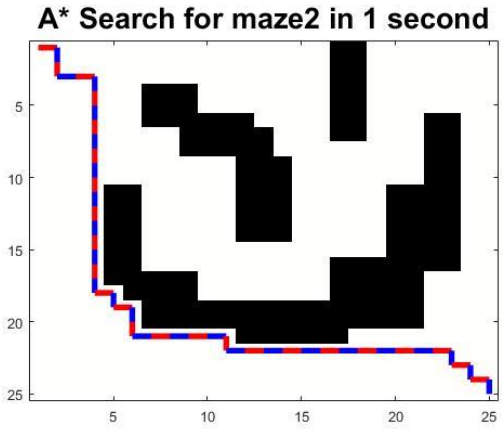


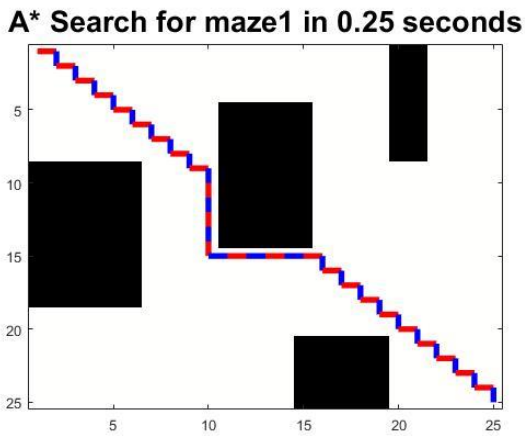


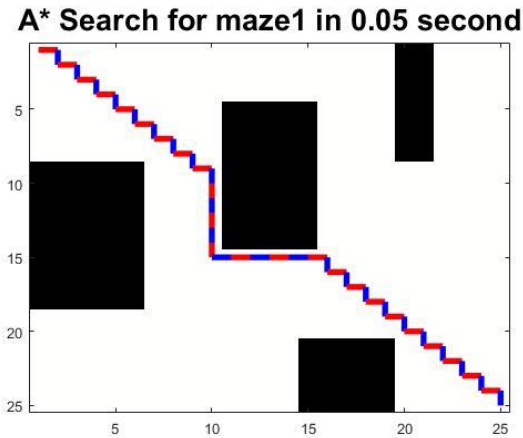
ii. A* Search in 2D – Deflating epsilon

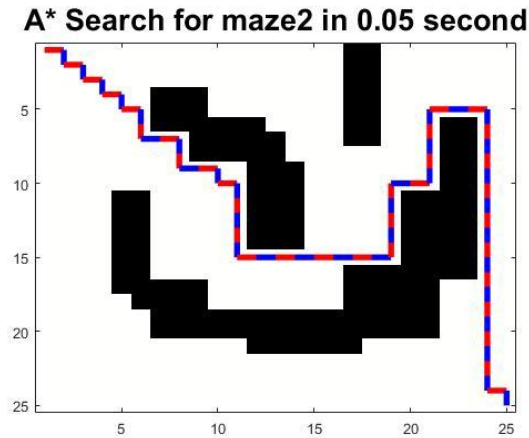
Maze 1									
	1 second			0.25 seconds			0.05 seconds		
	Epsilon	Path Length	Nodes Expanded	Epsilon	Path Length	Nodes Expanded	Epsilon	Path Length	Nodes Expanded
1	10	48	49	10	48	49	10	48	49
2	5.5	48	49	5.5	48	49			
3	3.25	48	49						
4	2.12	48	49						
5	1.5625	48	50						
6	1.2813	48	289						
7	1.1406	48	366						
8	1.0703	48	396						
9	1.0352	48	429						
10	1.0176	48	432						

Maze 2									
	1 second			0.25 seconds			0.05 seconds		
	Epsilon	Path Length	Nodes Expanded	Epsilon	Path Length	Nodes Expanded	Epsilon	Path Length	Nodes Expanded
1	10	68	145	10	68	145	10	68	145
2	5.5	68	152	5.5	68	152			
3	3.25	68	161						
4	2.12	48	205						
5	1.5625	48	284						
6	1.2813	48	297						
7	1.1406	48	342						
8	1.0703	48	382						
9	1.0352	48	412						









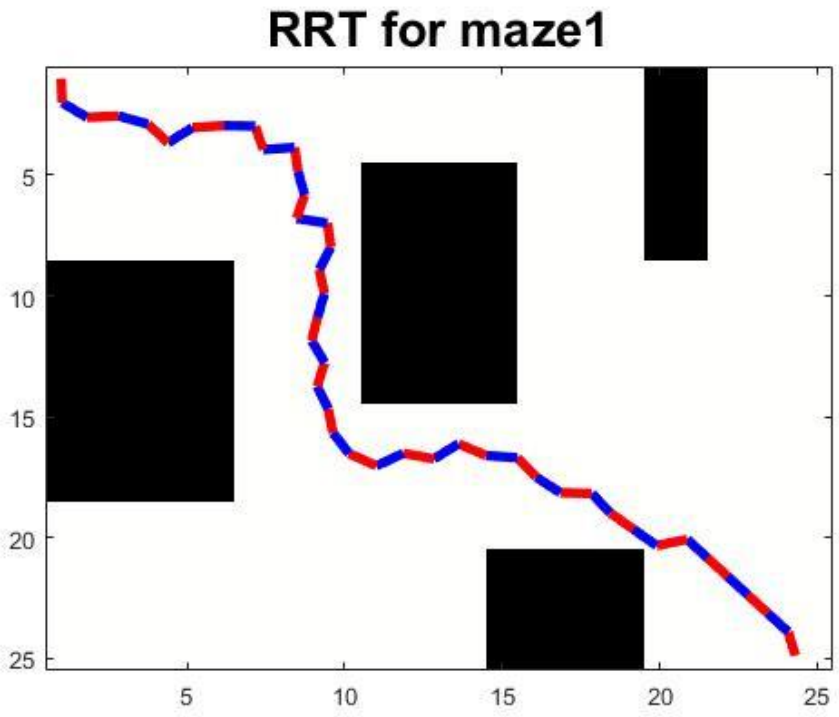
iii. RRT in 2D

Maze 1:

Run time: 0.1612 seconds

Path length: 33.3688

Nodes expanded: 120

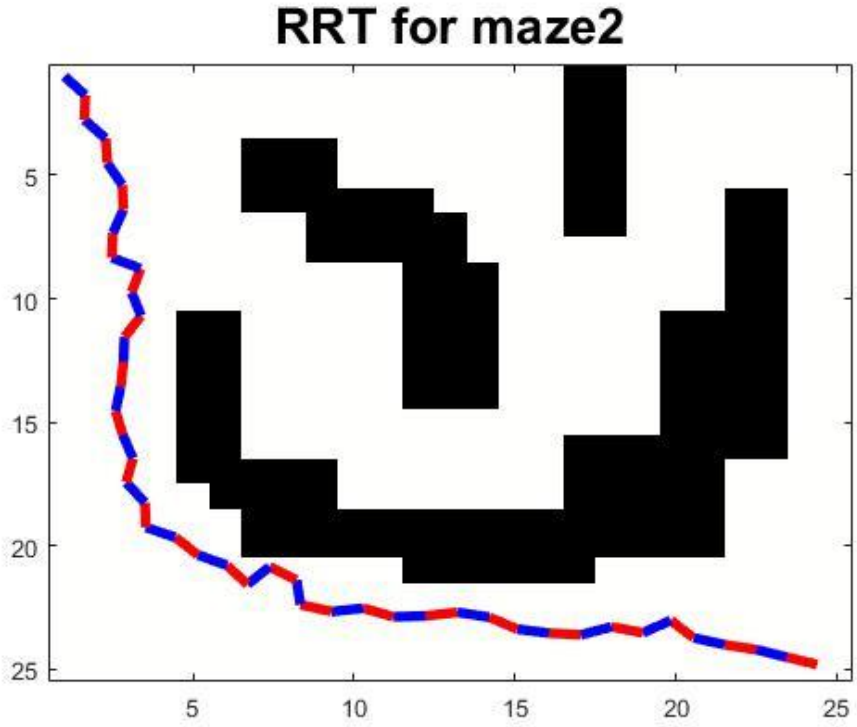


Maze 2:

Run time: 0.2644 seconds

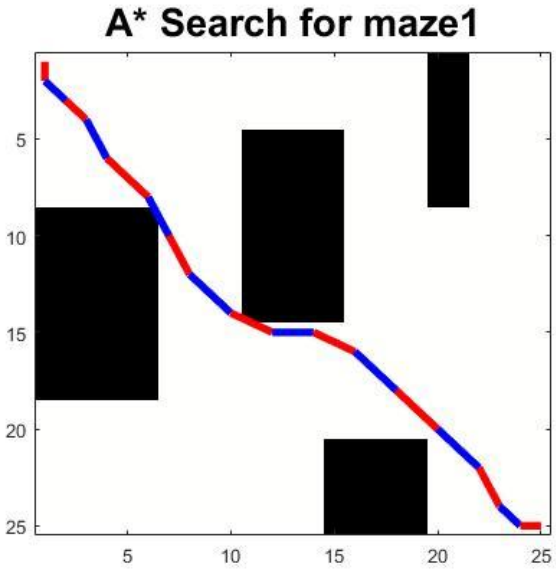
Path length: 33.823

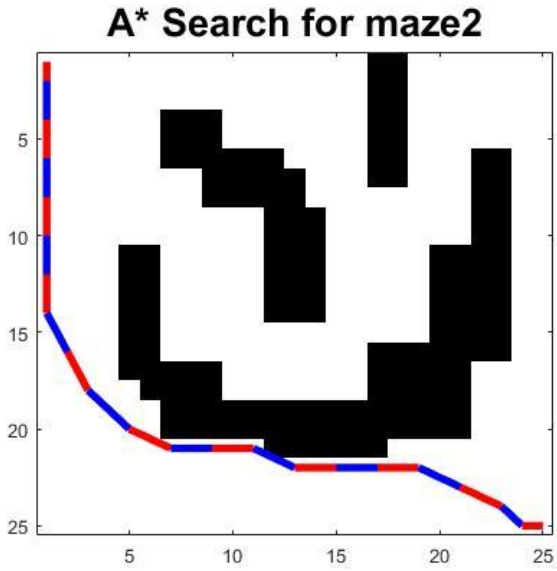
Nodes expanded: 207



Step 3

i. A* Search – Admissible heuristic: Euclidean distance divided by 2

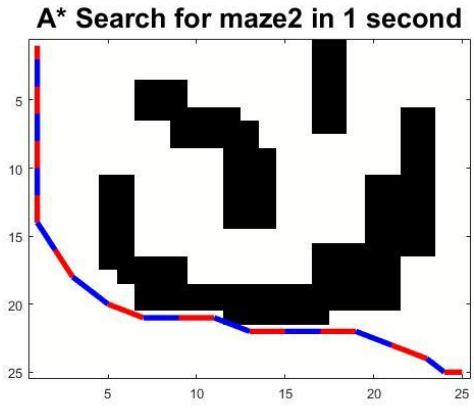




ii. A* Search in 4D – Deflating epsilon

Maze 1									
	1 second			0.25 seconds			0.05 seconds		
	Epsilon	Path Length	Nodes Expanded	Epsilon	Path Length	Nodes Expanded	Epsilon	Path Length	Nodes Expanded
1	10	33.9411	2808	Failed to Complete in Time			Failed to Complete in Time		
2	5.5	33.9411	2808						

Maze 2									
	1 second			0.25 seconds			0.05 seconds		
	Epsilon	Path Length	Nodes Expanded	Epsilon	Path Length	Nodes Expanded	Epsilon	Path Length	Nodes Expanded
1	10	33.9411	2427	Failed to Complete in Time			Failed to Complete in Time		
2	5.5	33.9411	2427						



Comparison

1) The larger the heuristic inflation factor epsilon, the less nodes that were expanded and the longer the path length was. Thus, solution quality worsens (less optimal the solution was) as epsilon was increased. The effect of the heuristic inflation factor epsilon was diminished by the complexity and dimensionality of the environment. However, this statement is consistent with my observation. The epsilon is dependent on the heuristic so you would have to determine your heuristic and then pick your epsilon and deflation rate based on how informative your heuristic is.
2) Even though 4D isn't an absurdly large and complex dimensionality, when using RRT, random sampling would be more difficult and complex since the possible combinations of states increases so there would have a need for a more directed sampling when expanding nodes for the growing tree. Also, instead of searching in the state space, you can search in the configuration space so random sampling becomes less complex and easier to direct. Each configuration contains the state in the n-dimensional state space so you would sample one point at a time without having to pick from one of the dimensions in the state space.