15-381 Spring 2007 Assignment 3: Robot Motion Planning and Game Theory

Out: February 20th, 2007
Due: March 20th, 1:30pm Tuesday
Questions to Ellie Lin (elliel+15381@cs.cmu.edu)

The written portion of this assignment must be turned in at the beginning of class at 1:30pm on March 20th. Type or write neatly; illegible submissions will not receive credit. Write your name and andrew id clearly at the top of your assignment. If you do not write your andrew id on your assignment, you will lose 5 points.

The code portion of this assignment must be submitted electronically by 1:30pm on March 20th. To submit your code, please copy all of the necessary files to the following directory:

/afs/andrew.cmu.edu/course/15/381/hw3_submit_directory/yourandrewid

replacing yourandrewid with your Andrew ID. The TAs grading this assignment would prefer that you use C/C++, Java, or Matlab. If you would like to use a different programming language, please check with us first. All code will be tested on a Linux system, we will not accept Windows binaries. No matter what language you use, you must ensure that the code compiles and runs in the afs submission directory. Clearly document your program.

Late Policy. Both your written work and code are due at 1:30pm on 3/20. Submitting your work late will affect its score as follows:

- If you submit it after 1:30pm on 3/20 but before 1:30pm on 3/21, it will receive 90% of its score.
- If you submit it after 1:30pm on 3/21 but before 1:30pm on 3/22, it will receive 50% of its score.
- If you submit it after 1:30pm on 3/22, it will receive no score.

Collaboration Policy. You are to complete this assignment individually. However, you are encouraged to discuss the general algorithms and ideas in the class in order to help each other answer homework questions. You are also welcome to give each other examples that are not on the assignment in order to demonstrate how to solve problems. But we require you to:

- not explicitly tell each other the answers
- not to copy answers
- not to allow your answers to be copied

In those cases where you work with one or more other people on the general discussion of the assignment and surrounding topics, we ask that you specifically record on the assigment the names of the people you were in discussion with (or "none" if you did not talk with anyone else). This is worth five points: for each problem, you solution should either contained the names of people you talked to about it, or "none." If you do not give references for each problem, you will lose five points. This will help resolve the situation where a mistake in general discussion led to a replicated weird error among multiple solutions. This policy has been established in order to be fair to everyone in the class. We have a grading policy of watching for cheating and we will follow up if it is detected.

1. Robot Motion Planning

(a) (5 points) Draw (approximately) the Voronoi edges in this space assuming that A, B, C, and D are obstacles (ignoring S and G). Now draw the path from start (S) to goal (G) found by using the Voronoi diagram.

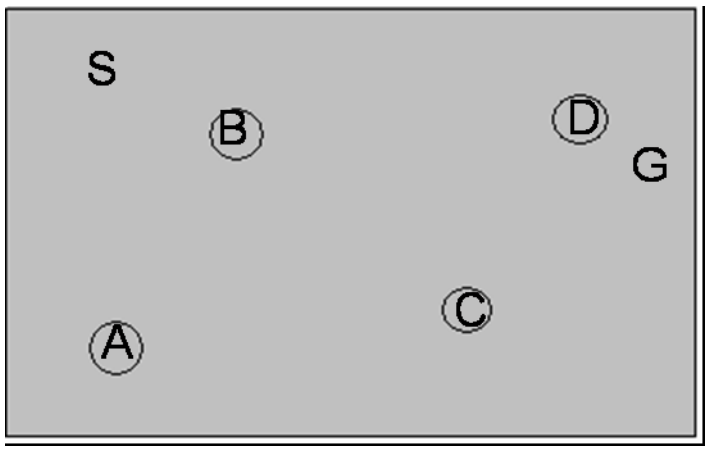

Figure 1: Voronoi

(b) (5 points) Draw (approximately) the path found by using the visibility graph technique from start (S) to goal (G). Draw the initial visibility graph.

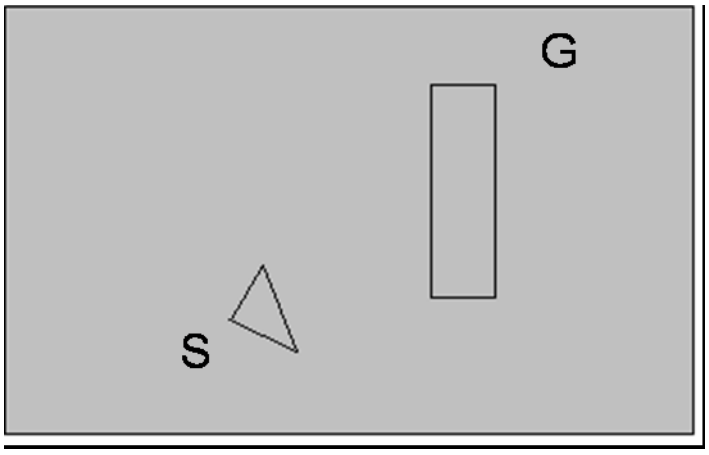

Figure 2: Visibility graph

(c) (5 points) Draw the path found by using approximate cell decomposition from start (S) to goal (G). Begin with an initial grid spacing of (approximately) 3 cm. Draw the cells that partition this space.

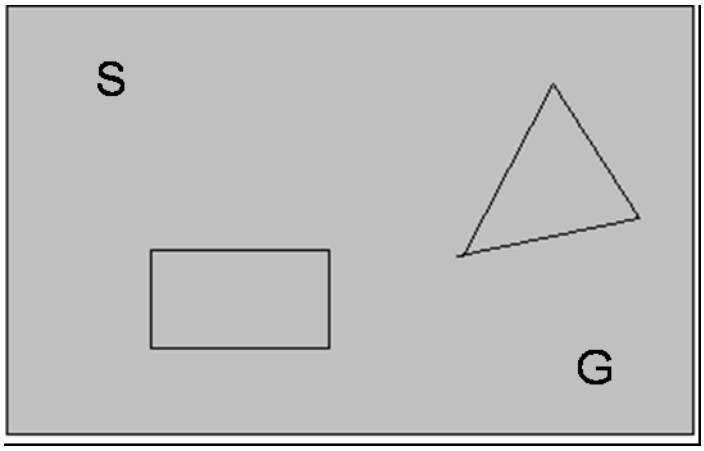

Figure 3: Approximate Cell Decomposition

(d) (5 points) Draw (approximately) the edges created with Rapidly Exploring Random Trees when the random new sample nodes appear in the order specified on the diagram. The initial tree is just the node S. Use the second way of determining ϵ as discussed in class. That is: if the line between q_{near} and q_{rand} intersects an obstacle, q_{new} is the closest point to q_{rand} in front of (on the q_{near} side of) the obstacle. Otherwise, q_{new} is q_{rand}.

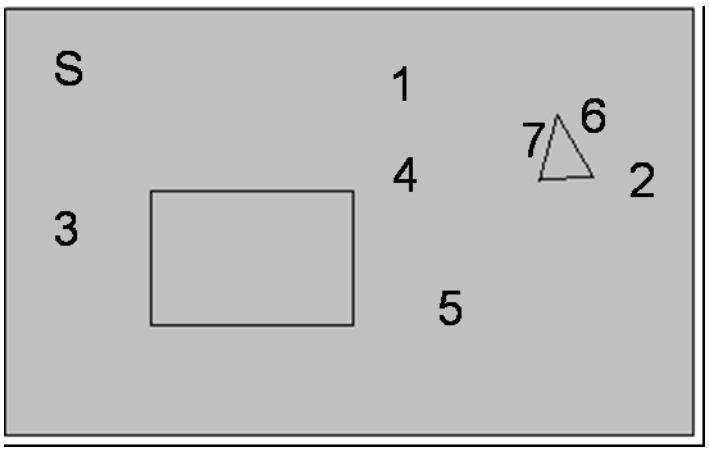

Figure 4: RRT

2. Probabilistic Roadmap

Roboto the robot needs to find a path from his start point to his goal point in his discretized, 2D world while avoiding obstacles. Roboto is given the location of his start point, his goal point, and every obstacle in his world. Roboto is also given locations of sample points to use for building a probabilistic roadmap.

Assume that Roboto can move in straight lines between sample points (thus a Euclidean distance metric is appropriate).

For this problem, you will be using A-star search to find the shortest path between the start and goal along a roadmap created from sample points.

There are three map files associated with this exercise (map1.txt, map2.txt, map3.txt). Each map file specifies the start point, goal point, free space, and obstacles in the world. The map file is a 7×7 grid of ints. The start point is 1, the goal point is 9, free space is 0, and obstacles are 5.
This is maptest.txt:

0 0 0 0 0 0 0
1 0 0 0 0 0 0
0 0 5 5 5 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 9 0
0 0 0 0 0 0 0

There are two sample files associated with this exercise (samples1.txt, samples2.txt). Each sample file specifies the sample points to use for building the probabilistic roadmap. The sample file is a 7×7 grid of ints. Samples are 1. Non-samples are 0. This is samples2.txt:

0 0 0 1 0 0 0
0 1 0 0 0 1 0
0 0 0 0 0 0 0
0 0 1 0 0 1 0
1 0 0 0 0 0 0
0 0 0 1 0 0 0
0 0 0 0 0 0 1

(a) (40 points) For this question, you will implement A-star search on a probabilistic roadmap using the samples provided. Your roadmap will use only samples in the free space specified by the map and consider the $K = 4$ nearest neighbors of each sample for creating edges. Of the $K = 4$ possible edges, only create those that do not intersect obstacles. The start and goal points can effectively be considered samples when creating edges.

When implementing A-star search, make sure your successor function is aware that it would be counterproductive to visit a location in the path more than once.

In addition to submitting your code, please submit an executable called $findPath$ that takes a map file as its first input and a sample file as its second input. Output the number of expansions before achieving your goal, the locations of each sample visited along the path from start to goal (include the start and goal locations, as well), and the total distance along the roadmap from start to goal. If no solution exists, output the number of expansions. If your program requires doing something other than
($./ < executablename > \ < argument1\_name > \ < argument2\_name >$) at the command line, you must specify exactly what we should run from the command line to get your program to take the scenario input files and produce the required output.

The grid locations follow the (row,col) convention where the top left corner is (1,1):

(1,1) (1,2) (1,3) (1,4)
(2,1) (2,2) (2,3) (2,4)
(3,1) (3,2) (3,3) (3,4)
(4,1) (4,2) (4,3) (4,4)

Here is example output:

% *./findPath maptest.txt samples2.txt*
Number of expansions: 5
(2, 1)
(5, 1)
(6, 4)
(6, 6)
Total distance: 8.16228
%

Note on distances: The cells are 1 unit × 1 unit. I.e., the distance between two rectilinear neighboring cells is 1 unit while the distance between two diagonal neighboring cells is $\sqrt{2}$ units.

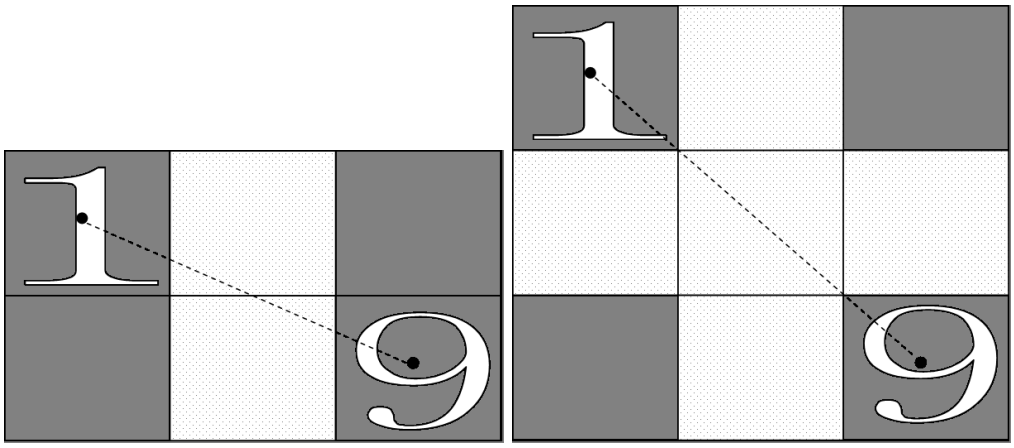

Figure 5: Obstacle detection: patterned cells show where to check for obstacles.

Hints on obstacle detection: When performing obstacle detection between two cells, draw a line between the centers of both cells. If the line falls within any other cells along the way, check those cells for obstacles. The corners of cells also count! The figure above left shows the line between cell (1,1) and cell (2,3). If an obstacle exists in either (1,2) or (2,2), the path has been obstructed. The figure above right shows the line between cell (1,1) and cell (3,3). If an obstacle exists in (1,2), (2,1), (2,2), (2,3) or (3,2), the path has been obstructed.

(b) (2 pts) Describe the advantages and disadvantages of using sampling.

(c) (6 pts) Report the results of running each of the three map (map1, map2, map3) scenarios with the two sample (samples1, samples2) files. Be sure to report whether or not a solution was found and the number of expansions. Include all the output requested when running the executable. Also discuss the effectiveness of the two different sampling strategies used on the map scenarios.

(d) (2 pts) Describe how you detected the intersection of an edge of your roadmap with an obstacle. Also describe the heuristic you used with A-star.

3. (5 points) Chomp is a 2-player game played on a rectangular "chocolate bar" made up of smaller rectangular cells. The players take it in turns to choose one block and "eat it" (remove from the board), together with those that are below it and to its right. The top left block is "poisoned" and the player who eats this loses. Please see http://en.wikipedia.org/wiki/Chomp for more details about this game. This game is an example of a two-player zero-sum deterministic game of perfect information.

   Any two-player zero-sum deterministic game can be represented by the following quintuple: $(S,I,Succ,T,V)$, where S is the entire space of game states, I is the initial state, $Succ$ is the successor function, T are the terminal states, and V maps from terminal states to its payoff/utility. Please describe each element of the quintuple in the game of Chomp.

4. (5 points) A vendor at a street carnival offers the following games that involve rolling a single 6-sided die:

   Game 1: You pay 3 dollars and win the dollar amount on the roll.
   Game 2: You pay 1 dollar and win 2 dollars if the roll is odd.
   Game 3: You pay 2 dollars and win 8 dollars if you roll at least a 5.

   Which of these games would you play? Rank the games in order of which you would play them. Explain your reasoning in terms of expected payoff.

5. The Even versus Odd game

   A game is played as follows: The two players, A and B, simultaneously hold up one or two sticks. A wins if the total number of sticks is odd while B wins otherwise. The amount won is the total number of sticks held. It is paid to the winner by the loser.

   (a) (5 points) Write down the matrix form of the game. Is there a pure strategy solution in this game? Explain why or why not.

   (b) (5 points) Assume B holds up one stick $\frac{1}{2}$ the time and two sticks the other $\frac{1}{2}$ of the time. What is the expected payoff for player A if A also chooses one stick $\frac{1}{2}$ the time and two sticks the other $\frac{1}{2}$ of the time? What is the expected payoff for player A if A chooses one stick $\frac{3}{4}$ths of the time and two sticks the remaining $\frac{1}{4}$th?

6. Cing and Veri were best friends at CMU, but Cing recently graduated and moved to another state. Wanting to stay connected, Cing and Veri both decide to sign up for cell phone service. However, they both have different preferences:

- Cing travels a lot and likes Cingular above all else for its extensive network.
- Veri likes Verizon above all else because of its cell tower conveniently placed above Wean Hall.
- If at least one of them gets T-mobile, the two could talk for free with the T-mobile Fave-5 plan.
- If both of them get Cingular, the two could talk for free with Cingular's in-Network plan.
- If both of them get Verizon, the two could talk for free with Verizon's in-Network plan.
- If they cannot talk for free, they would rather not have cell phone service at all.

(a) (5 points) We can model their preferences as a non-zero-sum game. Fill in the payoff matrix for the two of them that reflects their preferences (there are many ways to do this). State your assumptions (e.g. No cell phones pays 0).

		Veri		
		Cingular	Verizon	T-mobile
Cing	Cingular	,	,	,
	Verizon	,	,	,
	T-mobile	,	,	,

Figure 6: Payoff matrix for Cing, Veri

Assumptions:

(b) (2 points) Are there any strictly dominating strategies for either of them? If so, which are they?

(c) (3 points) Which strategy should they take? Please explain your reasoning.