Path Planning, Multi-Robot Systems 6

(Max score: 60 + 10 bonus points)

16-311: Introduction to Robotics (Fall 2017)

OUT: November 24, 2017, at 3:00pm

DUE: November 30, 2017 at 9:pm - Available late days: 1

Instructions

Homework Policy

Homework is due on autolab by the posted deadline. As a general rule, you have a total of 8 late days. For this homework you cannot use more than 1 late day. No credit will be given for homework submitted after the late day. After your 8 late days have been used you will receive 20% off for each additional day late.

You can discuss the exercises with your classmates, but you should write up your own solutions. If you find a solution in any source other than the material provided in the course, you must mention the source.

Submission

Create a tar archive of the folder with your ROS packages/nodes and submit it to Homework 6 on autolab. You should also have one PDF file in your archive, with an explaination of your results regarding the experiments with the robot, and the instructions for running the nodes, if any.

Contents

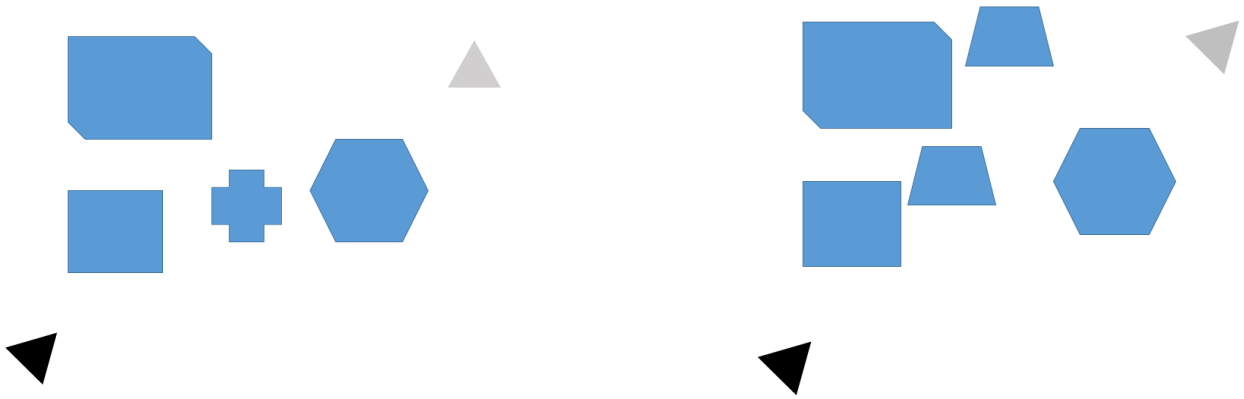
Part I

Pen and paper questions

1 Polygonal robot in a polygonal obstacle world (5 points)

The given robot has a triangular shape (as shown in the figures) and can only translate. All obstacles in the world are polygons. The goal is to plan the path for the robot from the initial configuration (bottom-left, black triangle) to a final configuration (top-right, gray triangle) for the two scenarios in the figures.

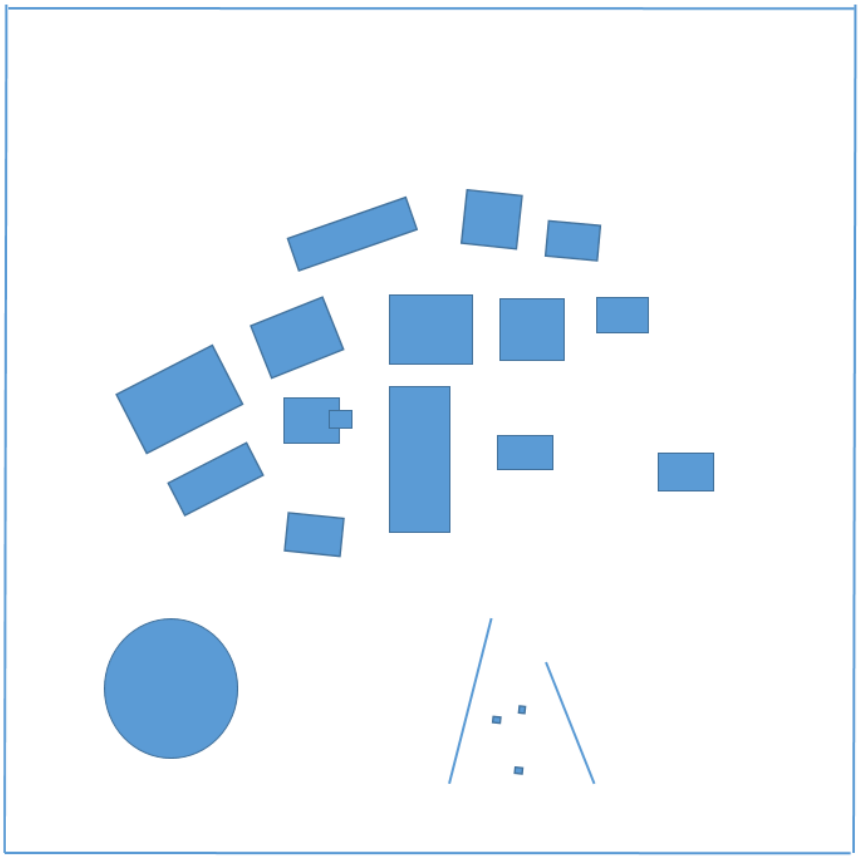
Solve the problem by working in the configuration space and constructing visibility graphs. Show the (useful parts of the) configuration space and of the visibility graphs, and the optimal path.



2 Path planning using Voronoi diagrams (5 points)

In the scenario in the figure, the blue elements represent obstacles.

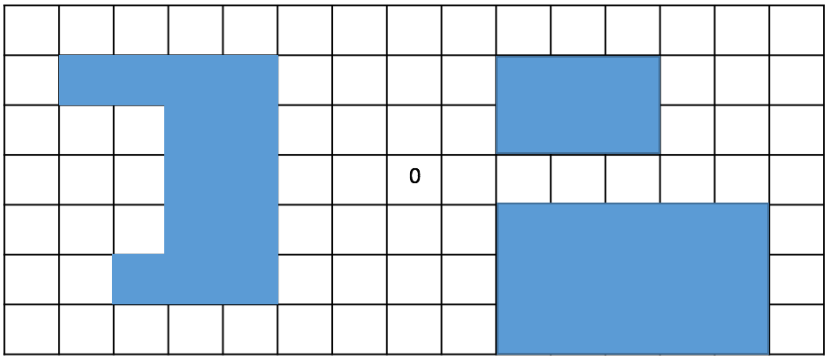
1. Draw the Voronoi diagram, and from the diagram define a road map graph that can be used for navigation;
2. Find the maximum obstacle clearance path and the shortest path for a point robot moving from the start to the goal positions.



3 Approximate cell decomposition (10 points)

3.1 Wavefront planner (3 points)

A mobile robot has a two dimensional configuration space $\mathcal{C} \subseteq [0, 15] \times [0, 7]$. After accounting for the presence and shape of both the robots and the obstacles, the resulting configuration space $\mathcal{C} = \mathcal{C}_{free} \bigcup \mathcal{C}_{obs}$ is that shown in the figure. Since we don't need much precision for moving the robot, and the obstacles are assumed to be quite regular in terms of their shape, a (uniform) grid decomposition of the configuration space is defined, where each (squared) cell is 1×1, as shown in the figure. That is, the x dimension in the C-space has a resolution $k = 15$, while the y dimension has a resolution $k = 7$.



A road map graph is defined by placing graph vertices in the middle of the cells and considering 4-connected vertices (i.e., corresponding to the N-E-S-W edges, diagonal connections/movement are not considered).

A wavefront planner (which is basically BFS) is then employed to find on the resulting graph the shortest path to the goal configuration, which is the cell with 0, of cell coordinates $(8, 4)$ (assuming that the bottom leftmost cell has cell coordinates $(1, 1)$).

1. Report in each cell of the grid the distance score computed by the wavefront planner;

2. Find the shortest path defined by the wavefront planner for moving to the goal configuration from the configuration defined by cell $(3, 5)$.

3.2 Changing spatial resolution (7 points)

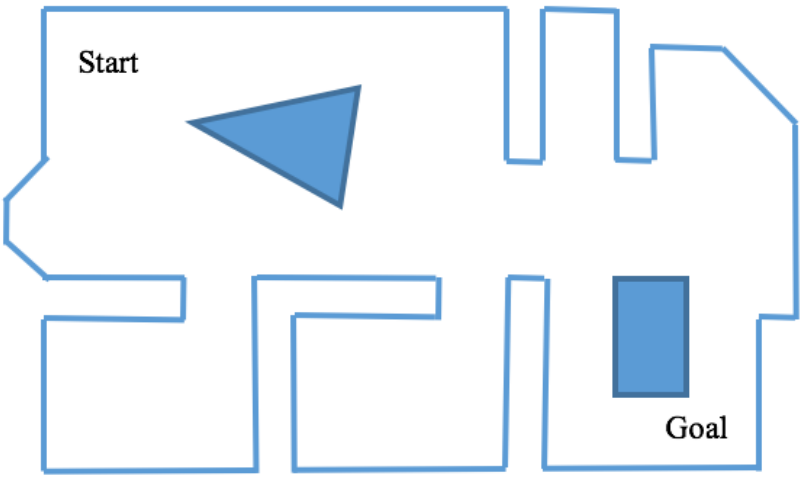
We are now given a different robot, a car-like robot, that has a 3-dimensional configuration space $\mathcal{C} \subseteq [0, 15] \times [0, 7] \times [0, 2\pi]$.

1. We want to increase the planning resolution, such that the x and y dimensions are are now discretized with a resolution respectively of $k = 1000$ and $k = 500$, while the resolution for the angular dimension is one degree. How many grid cells are needed? In the worst case (i.e., not accounting for the obstacles) how many edges has the road map graph? (define the connectivity model first).
2. Is the resulting graph "affordable" for using it with A^*? What heuristic would you use in A^*? Is the heuristic admissible?
3. Describe (operationally, report numeric values) how you would proceed in order to possibly reduce the computational complexity of the planning problem while minimizing the loss in precision.
4. A car-like robot is non-holonomic. Therefore, when we reason in the configuration space we also need to account for possible kinematic constraints that could limit the actual motion between two feasible configurations. Let's assume that from the previous question a (computationally feasible) road map graph has been computed, and now you are ready to use A^* or Dijkstra in order to find the shortest path on the graph from given initial and goal configurations. How would you proceed in order to include the non-holonomicity of the robot when you incrementally expand nodes looking for the shortest path?

   Let's assume that the car has a steering angle limited in the interval $[-\pi/3, \pi/3]$ and the rear wheels velocities are limited in the interval $[0, 50]$ (i.e., the car can only move forward (it is a *Dubin* car). Describe in detail how you would implement a check for non-holonomicity sub-paths. (Hint: you should propose something that goes in the same direction of the local planners used in sampling-based planning).

4 Exact spatial decomposition (6 points)

Given the two-dimensional configuration space scenario in the figure, where the blue line/objects represent obstacles, the task is to define a road map graph to find the shortest path from the start to the goal configurations.



1. Define the graph and find the path using an exact cell decomposition based on shooting vertical segments.
2. Define the graph and find the path using an exact cell decomposition based on a mesh.

5 Multi-robot task allocation (8 points)

The company Guber provides automatic pick-up and delivery services for goods. At any time, customers can send to a control center their requests for receiving selected goods at their homes. For instance, customer c can ask for a bundle of goods $G^c = \{g_1^c, g_2^c, \ldots, g_n^c\}$. Goods are available at known locations, where the robot can pick them up. Each good i has a capacity requirement q_i and a value v_i. The delivery of the goods in a bundle G^c requires the pick-up of the associated goods. While strictly related, these should be seen as different types of tasks.

Periodically, customer requests are organized into a batch and dispatched from the control center to a team of n mobile robots that perform the pick-up and delivery tasks. Each robot k has a limited capacity Q_k. The goods bundle G^c requested by a customer c is assigned to a single robot (no split of requests), respecting robot's capacity constraint.

Based on the knowledge of the capacity requirements of the goods, of robots' capacity limitations and mobility skills (e.g., wheeled vs. legged), and of the locations of goods and customers, Guber's control center computes the set R_k of all routes that can be feasibly assigned to a robot k for picking-up and deliver the goods for each one of the customers. For each robot k, each feasible route r has known traveling cost c_r^k.

Given the number of requests in the batch and robots' capacity constraints, it might not be possible to satisfy all customer requests. Therefore, Guber wants to assign routes (i.e., pick-up and delivery sub-task bundles) to the available robots with the goal of maximizing the difference between the overall value obtained from the delivered goods and the costs incurred for traveling, with the additional constraint that for any robot traveling over the assigned routes doesn't exceed a maximum cost (i.e., time) T_{max}.

1. Define an optimization model for Guber's task assignment problem.

2. Can the resulting optimization problem be solved in polynomial time? (Yes/No, why)

3. Based on Gerkey and Mataric taxonomy, how can the above problem can be classified?

4. Define, with a pseudo-code, an alternative strategy for addressing Gruber's problem following a distributed approach that doesn't rely on an optimization model.

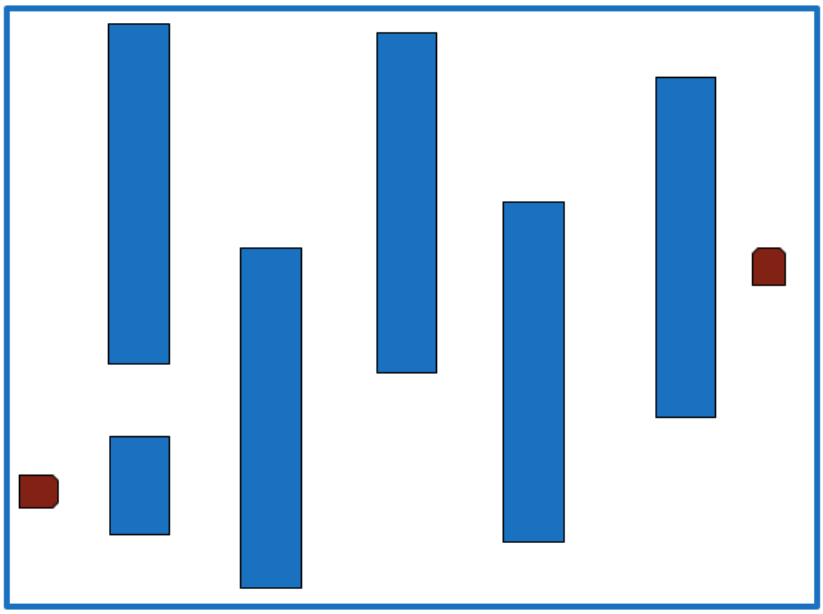
Part II

Programming questions

6 RRT (16 points)

Consider the scenario in the figure for a car-like robot with a steering angle limited between $[-\pi/3, \pi/3]$. Implement an RRT algorithm that finds a path between the given start (bottom-left) and goal (top-right) configurations . Describe the design choices of the algorithm, report the built RRT tree, and show the found path (including the orientation of the robot).

You can use any appropriate scaling for implementing the scenario, as long as proportions are respected.



7 PRM (15 points)

Consider the same scenario of the previous question. Implement a PRM algorithm that finds a path between the given start (bottom-left) and goal (top-right) configurations. Describe the design choices of the algorithm, show the built road map graph, and show the found path (including the orientation of the robot).

8 Execute the path (5 points)

Implement the scenario in Gazebo placing obstacles, and use the path found in one of the questions above. This time however, use the turtlebot to navigate the path (which will consist of a set of poses that can be achieved by the robot using a feedback-based controller developed in previous homework).