ECE 4984 & 5984: (Advanced) Robot Motion Planning (Spring 2017)

Homework 4

Due: Tuesday March 28th, 9PM

March 17, 2017

Instructions. This homework constitutes 10% of the course grade. You must work individually on all problems. It is okay to discuss the problems with other students in class. However, your submission must be your original work. This implies:

- If you discuss the problems with other students, you should write down their names in the pdf report.
- If you refer to any other source (textbook, websites, etc.), please cite them in the report at the relevant places.
- The answers in the pdf report must be written entirely by you from scratch. No verbatim copy-paste allowed without citations.
- Any software you submit must be written entirely by you and your partner with no copy-pasting of significant portions of code from other sources.

Please follow the submission instructions posted on canvas exactly. You must submit your assignment online on canvas by the due date. Your submission must include one pdf file with the answers to all the problems and one or more files containing your code. It is okay to scan your answers and create the pdf submission.

Problem 1 — 5+5+5+5 points

Consider a four wheeled cart whose kinematic model is similar to the four wheeled car-like robot studied in class with the exception that the front two wheels cannot be steered. That is, all four wheels always point in the same fixed direction. We describe the configuration of the cart by (x, y, θ) where (x, y) are the coordinates of the center of the cart's rear axle. θ is the angle made by the normal to the axle with the X–axis. If needed, assume that initially the cart is at the configuration (x_0, y_0, θ_0).

(a) Derive the equation for the kinematic constraint on the motion of the cart. The kinematic constraint is the same as that of a car-like robot – the vehicle must move only in the direction that the rear wheels are pointing.

(b) Write down the configuration transition equation, $\dot{x} = f_1(x, y, \theta, t), \dot{y} = f_2(x, y, \theta, t), \dot{\theta} = f_3(x, y, \theta, t)$ for the cart.

(c) Show that the kinematic constraint is a holonomic one.

(d) Holonomic constraints often help in reducing the dimensionality of the configuration space to the actual degrees-of-freedom. For instance, in the example we covered in class (also given in Section 13.1.3.4 of the text), we were able to reduce the configuration space from $\mathbb{R}^2$ to a circle $\mathbb{S}^1 = [0, 2\pi)$. Can you reduce the configuration space for the cart from $\mathbb{R}^2 \times \mathbb{S}$ to a smaller dimensional space? If so, what is the smaller dimensional configuration space.

Hint: Refer to section 13.1.2.1 in the text and make suitable modifications for the case of fixed front wheels.

Problem 2 — 10+5 points

Suppose we have a triangular robot operating in a 2D workspace. The robot can translate in any direction but cannot rotate. The configuration of the robot is specified by the position leftmost vertex of the triangle (marked by a circle in the Figure 1). Since the robot can only translate and not rotate, the configuration space is $\mathbb{R}^2$.

The workspace has three polygonal obstacles as shown in Figure 1. Our goal is to find the shortest path for the robot from its initial configuration (marked in red) of $(0, 0)$ to the goal configuration (marked in green) of $(10, 10)$ while avoiding the obstacles. We will do it by hand.

(a) Neatly draw the configuration space and mark the configuration space obstacles by correctly inflating the workspace obstacles. You can draw this figure by hand, or write a program to do so. If you are drawing it by hand, it may be easier to print a high resolution picture of the workspace first (`workspace.svg` file on canvas). Also mark the starting and the goal configuration in the figure.

(b) Now find the minimum length path from the starting configuration to the goal configuration while avoiding the C-space obstacles. You can do this by hand or write a program to do so. Include a scanned copy of your C-space obstacles and the path in the pdf report.

Problem 3 — 65 points

Write a program (in C/C++/Python/MATLAB) to find the shortest length path for the robot using Rapidly-Exploring Random Trees (RRTs). You are allowed to use any version of RRT or RRT* that we have covered in class. We will restrict our attention to robots whose configuration space is $\mathbb{R}^2$ and the robot can move in any direction instantaneously.

The input to your program is a text file called `input.txt`. The first two lines in this file specify the coordinates of the starting and the goal configuration respectively. Each line after the second will contain the coordinates of the configuration space obstacles specified as $x_1\ y_1, x_2\ y_2, \ldots$. That is, the X and Y coordinate of a C-space obstacle corner are separated by a space, and the coordinates of consecutive corners are separated by a comma. The corners are listed in the order they appear on the boundary. You will have to implement a subroutine to check if a configuration or a path in the configuration space is in collision with the configuration space obstacles.

Run your implementation on the maze configuration space shown in Figure 2. The start and goal configurations are shown in red and green respectively. The corresponding input file, `input_maze.txt` is given on canvas. You should also create another input file, named `input_triangle.txt` corresponding to your solution to Problem 2. Compare the cost of the shortest path found by hand in Problem 2 with the cost of the path returned by RRT.

Your submission must contain three parts: (1) submit the source code containing your implementation along with a README describing the compilation setup; (2) In the pdf report, write brief description of your implementation. In particular, explain your choice of algorithm, collision check subroutine, distance function used, nearest neighbor calculation, etc. Plot the tree generated during the algorithm and the path found. Include the two plots in the pdf report; (3) Submit the input file `input_triangle.txt`.

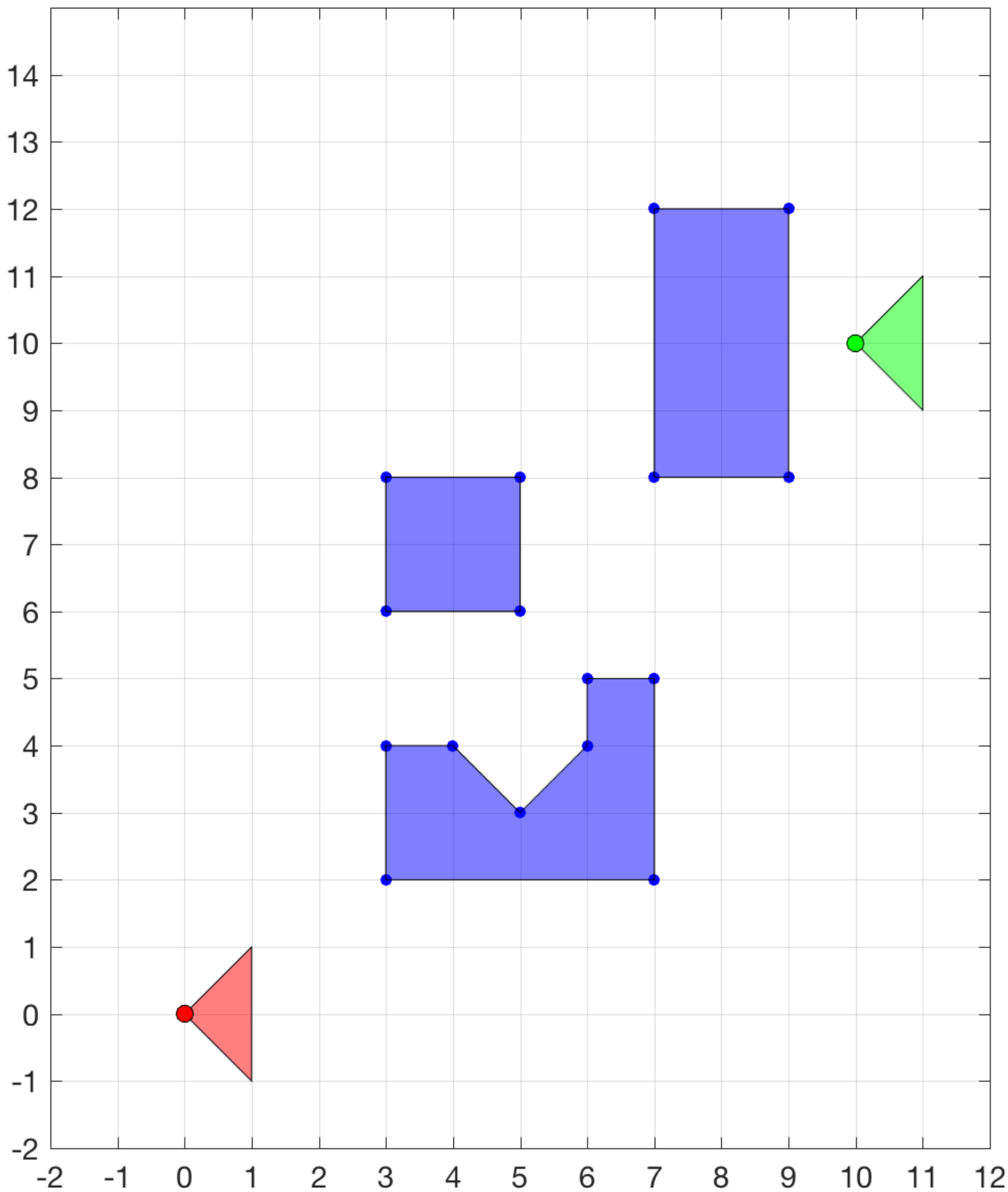

Figure 1: Workspace for a translating triangular robot. The start position is shown in red and the goal position is shown in green. The three workspace obstacles are shown in blue.

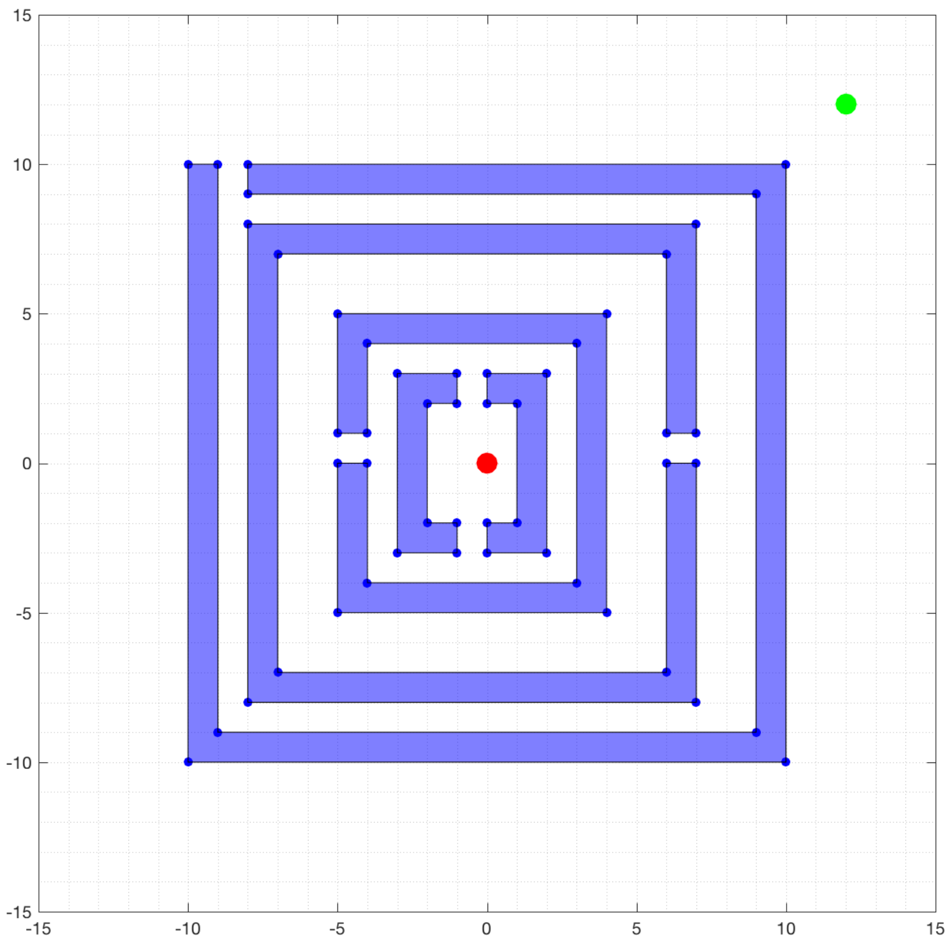

Figure 2: Configuration space for the maze.