Course › Week 8 › Proble... › RRT

RRT

RRT (Part A)

0.0/10.0 points (graded)

In this problem, we will implement a Rapidly Exploring Randomized Tree (RRT). For easy visualization, we will consider a two dimensional example. However, it will be relatively straightforward to extend your code to handle higher dimensional environments.

(a) The most computationally expensive portion of the algorithm is typically the collision checker. Write code that will take a set of points $\{x_i\}$ in two dimensions and determine if a given set of points $\{x_{0,i}\}$ **are in the convex hull** of the points $\{x_i\}$. The output of your code should be a row vector of boolean variables ("collisionFree" in the code below) whose i^{th} element is "true" if and only if $x_{0,i}$ is **not** in the convex hull.

Hint: The Matlab functions convhull and inpolygon may be useful to you here (but of course, you don't have to use them). Also, check your code by making plots a few times before submitting it.

```
%%% Problem Setup (DO NOT MODIFY) %%%
xi = randn(2,10); % Each column is a point (x,y)
x0 = randn(2,10);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% Your code here %%%%%%%%%%%%%%%%%


collisionFree = ;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



```

Unanswered

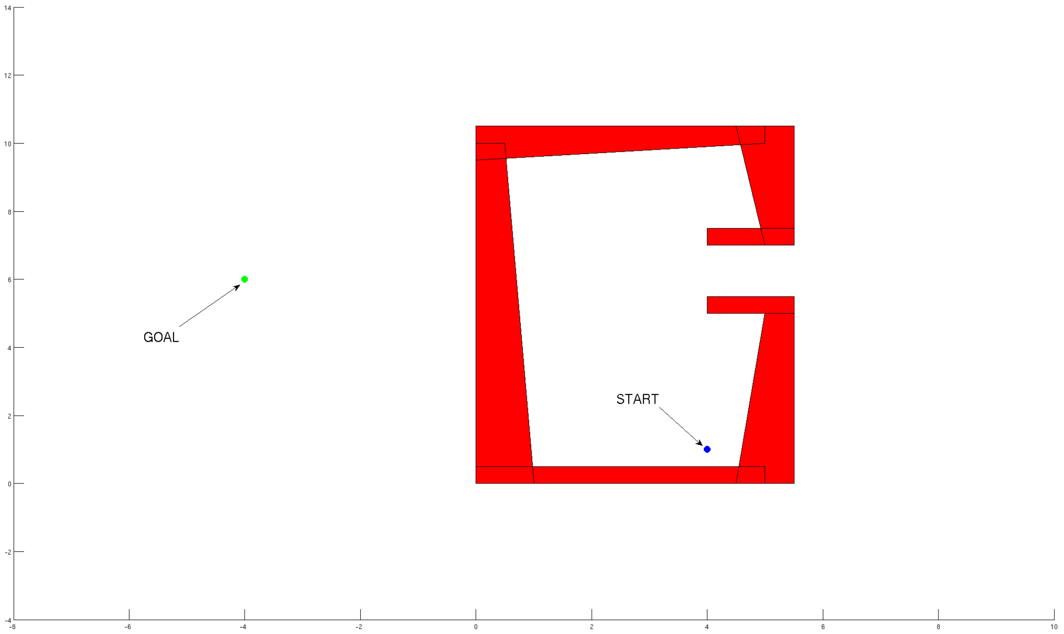
Run Code

Submit You have used 0 of 3 attempts

RRT (Part B)

0.0/20.0 points (graded)

(b) Next, we will implement the entire RRT algorithm and test it out on a "bug trap" environment (shown in the figure below). This kind of environment is typically quite challenging for motion planning algorithms since it requires discovering and navigating through a narrow passage in order to get out of the "trap". However, hopefully you'll see that the RRT algorithm is able to do this with relative ease.



We have provided some stub code here. Fill in the portions that say "FILL ME IN". Run your code a few times to get intuition for how the RRT grows. Once you are satisfied that the code is running correctly, paste in the RRT your code found below (this is the variable "rrt_verts_grade" in the code). The first row should be the start state and the last row should be (close to) the goal state. You may find the "clipboard" Matlab function useful for copying data in variables.

```
rrt_verts_grade = [];

```

Unanswered

Run Code

Submit You have used 0 of 3 attempts