6.S078 Planning Algorithms, Fall 2014

October 2, 2014

Assignment 3 (due Wed. Oct 15 before class)

Implement an RRT planner, test it for planning the motions of polygons in the plane with rotation (same setup as Assignment 2) and also planning the motion for an n-link planar robot arm. You should have a complete separation between the RRT code and the forward-kinematics and collision-checking code that's specific to the robot.

Part 0. Write the forward-kinematics and collision-checking interface for the robots. You should have all the code for the polygonal case already. For the n-link robot, define it as a list of convex polygons (one for each link), collision checking is then simply placing each link polygon given a configuration (a vector of joint-angles) and then doing polygonal collision checking. Define the range of legal joint angles for each joint to be less that the full 2π range.

Part 1. Implement the one-directional RRT, as described in LaValle. You can use a "brute force" nearest neighbor method or (if you can install `scipy`) look at the k-d tree implementation in `scipy.spatial`). You will need to have a "goal bias" (sample the goal 5-10% of the time). Run experiments for both robots.

Part 2. Implement the bi-directional RRT (be careful putting the paths from each tree together). Run experiments for both robots. Compare performance with the one-directional RRT.

Part 3. Implement "random short-cut" smoothing for the paths, illustrate the effect (and the cost). Compare to paths from the visibility graph method for the polygonal robot case. Make sure you have experiments where there are two ways to reach a goal (going to the left and right of one or more obstacles).

Please send email to ask for any clarifications.

You should uplaad a zip file with your code to the Stellar site. Also write a brief description of your approach in a PDF file and include some runs of your algorithm, with tables showing performance, as you did for assignments 1 and 2.