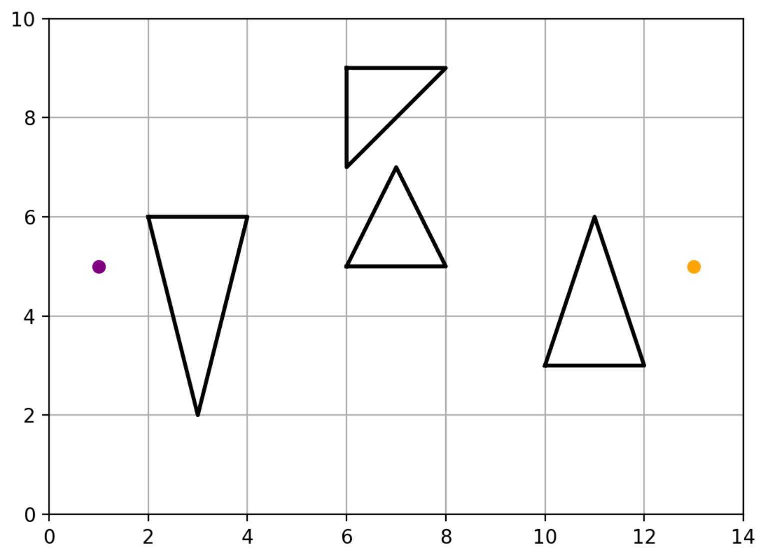
Homework #4

due Wednesday 1/31/24 11:59pm

This week, overlapping with the quiz, I would like to keep things a little lighter. We only have two problems, exploring the single query RRT and EST planners. The former is one of the main tools in robotics and often a little easier to implement. But the latter has some interesting behaviors and can be useful too.

To keep things simple, both problems revisit the the 2D space from last week's Problem 1. The dimensions and object locations are the same, but the start/goal are reversed:



You are, as always, welcome to code however you like, but the programs will share a lot of pieces with the PRM implementation from last week. So please review the skeleton code posted in `hw4democode.zip`. Indeed, relative to last week, you'll notice

- the `class Node` definition is almost the same, mostly adding a `self.parent` link (for the tree structure) and removing the references to and utilities for A*.
- the visualization is exactly the same.
- the `main()` code is greatly reduced: instead of sampling nodes, connecting nodes, and running A*, it simply calls `rrt()` or `est()` respectively.
- the path post-processing is exactly the same.

To edit/update the skeleton code, please look for the "FIXME" markings.

And please submit the critical code pieces, especially the pieces that you added/changed relative to the skeleton code. Also submit screen shots of the paths for the various configurations.

Problem 1 (Rapidly-exploring Random Tree Planner) - 48 points:

I would like to first play with an RRT planner. This is slightly easier to implement and fun to watch.

(a) Please code and run the planner with nominal parameters. This requires three code pieces:

  (i) Determining the target node. For now, select a target uniformly in the entire space (regardless with whether inside an obstacle or not).

  (ii) Determine the next node to try to add to the tree. For this, advance the nearest node in the tree toward target by a step of size 0.25 (or less if it reaches the target).

  (iii) If this next node is added to the tree, also check whether it lies within the step size of the goal and try to make that connection too.

(b) Now let's try directing the tree. Use the goal as target for the tree's growth 5% of the time. What happens if you use it more frequently, 50% or even 99% or 100% of the time.

(c) Finally, using the goal 5% of the time, let's try to increase the step size. What happens for a size of 1.0 or even 5.0?

Problem 2 (Expansive-Spaces Tree Planner) - 48 points:

Leveraging the RRT code, and working with the EST skeleton code, let us examine the EST planner.

(a) Again, first code/run the nominal planner. Here we are asking you to update the code to:

  (i) Select a step size for the below growth of 0.5.

  (ii) Determine the node in tree from which to grow. For now, select the node with the fewest neighbor nodes within a radius of 150% of the step size. Note, as this metric uses integer values, you may want to randomly select from all nodes that have the (same) smallest number of neighbors.

  (iii) Find the next node to add to the tree, randomly selecting a node that lies exactly the step size away from the growth node (on a circle around the growth node). That is, randomize the heading (angle) in which the tree should grow (uniformly from $-\pi$ to π).
  Keep selecting from that circle until you find a connectable node. If you don't add a node the above metric stays the same and the same growth node will be used again anyway.

  (iv) Once added to the tree, also check whether this node lies within the step size of the goal and try to connect (same as in RRT).

(b) Once the nominal planner looks good, let's play a little with how the tree grows. In place of a uniform distribution, select the growth's heading from a Gaussian distribution centered around the heading of the underlying branch. That is, the tree will grow a little "straighter". Try a standard deviation of $\pi/2$ or even $\pi/8$. What do the branches look like?

(c) For a standard deviation of $\pi/2$, also try growing toward with goal. To do so, update the metric for selecting the growth node to minimize (#Neighbors + scale * Distance-to-goal). Try a scale of 1.0 or even 5.0.

Problem 3 (Time Spent) - 4 points:

Approximately how much time did you spend on this homework? Any particular bottlenecks? And any late submission information?