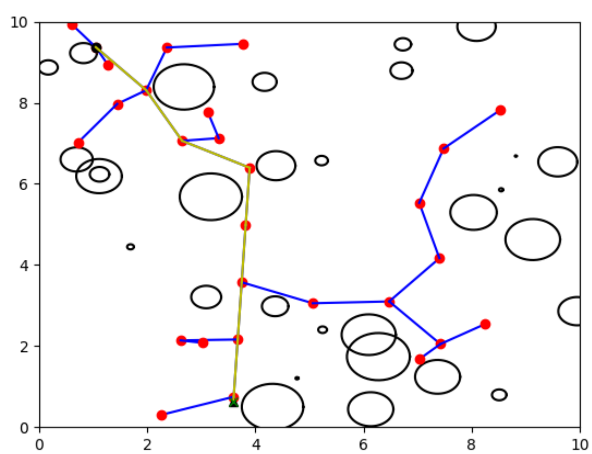
CSCI 3302: Introduction to Robotics
Homework 2: Probabilistic Motion Planning with RRT

RRT is a probabilistic search technique that can be used with rapid effectiveness in the high-dimensional, continuous environments common to robotics problems. In this assignment, you will implement the RRT algorithm for holonomic robots.

Using the provided base code [do not modify it], implement the RRT algorithm. **You are not permitted to call RRT-like functions from other packages to implement your own (i.e., you may not just write a wrapper that calls someone else's implementations)**.

The provided Python file will run your algorithm on four 2D domains: two without a goal, and two with a goal specified. The output will be saved into four plots (rrt_run1.png, rrt_run2.png, rrt_goal_run1.png, rrt_goal_run2.png).

You have to complete the ***rrt()*** function which will utilize ***get_nearest_vertex()***, ***steer()***, ***check_path_valid()*** functions which are also to be completed by you.



You are to complete this assignment on your own (without collaboration).

A successful implementation of RRT will look similar to this plot: your vertices will all be connected to each other, with no single edge being longer than Δq, a parameter passed in to your algorithm. In this example, a goal location was given (the triangle at the bottom of the plot) and the path from start vertex (black circle in the top-left) to goal vertex is shown in yellow.

Reminder:
1) If goal_point is not None, set q_rand to goal_point some small percentage of the time (e.g., if random.random() < 0.05: q_rand = goal)! This helps the tree grow towards the goal. This cannot be done all the time because there may not be a straight line path from the partial tree and the goal and thus we need the tree to grow in all directions too.

2) Make sure to keep all of your points defined as numpy arrays (e.g., cast them via np.array(my_point)) so you can perform standard mathematical operations on them.

Approximate Time to Complete: 120 - 240 minutes. Please reach out for help if you find yourself stuck or spending more time than this on the assignment!

To Submit: Turn in your fully implemented CSCI3302_hw2_rrt.py file and the four image files that it generates to Canvas. Do not change the file names.

Testing info:

- We will import the 4 functions that you have to fill from your code.
- We will test it on n-dimensional settings too. Note that the visualization and the world setup can only handle a 2D world and circular obstacles. So, you will have to test it without these functions if you wish. We highly recommend working with numpy in dealing with this general case.
- Our K would be in a range of 200 to 1000, **1e-5** for the threshold for goal, $\Delta q = \text{np.linalg.norm(bounds/10.)}$ where np is the numpy library.