COM S 476/576 Midterm Exam

Important Note: For any problem that asks for formal or mathematical definition, a rigorous mathematical description is required to receive full credit. Verbal explanations or examples will not be sufficient for full marks.

Let's revisit the motion planning problem for a mobile planar robot with a circular body of radius r operating in a 2D rectangular workspace, as previously described in Homework 3.

As before, the robot is modeled as a rigid body with a circular base of radius r. The robot can move in two ways:

- Translation $T(d)$: The robot can move along the direction specified by its current orientation, where $d \in \mathbb{R}$ is the distance parameter. A positive distance $d > 0$ indicates forward movement, while a negative distance $d < 0$ indicates backward movement.

- Rotation $R(\theta)$: The robot can rotate in place by an angle θ around its center, where $\theta \in \mathbb{R}$ is the rotation parameter.

The workspace, where the robot operates, is a rectangular area defined by the set $[0, X_{max}] \times [0, Y_{max}]$, with the bottom-left corner at $(0, 0)$ and the top-right corner at (X_{max}, Y_{max}). The world contains a set O of circular obstacles, each defined by a tuple $o = (x_o, y_o, r_o) \in \mathbb{R}^2 \times \mathbb{R}_{>0}$, where (x_o, y_o) is the position of the obstacle's center and r_o is its radius.

The robot initial configuration is $q_I = (x_I, y_I, \theta_I)$. Unlike in Problem 3 of Homework 3, the robot's initial position (x_I, y_I) is not restricted to grid points; it can be located anywhere within the workspace. Additionally, collisions must be considered at all times during the robot's movement.

The goal configuration $q_G = (x_G, y_G, \theta_G)$ includes both position and orientation, and the robot is considered to have reached the goal only when its configuration exactly matches q_G, not just when its center is at (x_G, y_G).

The objective of this problem is to determine a sequence of robot's center positions $p = (x_0, y_0)(x_1, y_1) \ldots (x_n, y_n)$, where $(x_0, y_0) = (x_I, y_I)$ and $(x_n, y_n) = (x_G, y_G)$, along a path $\tau : [0, 1] \to \mathcal{C}_{free}$ such that when projecting τ onto the xy-plane, it forms a sequence of line segments $l_1 l_2 \ldots l_n$ with each segment l_i connecting (x_{i-1}, y_{i-1}) and (x_i, y_i). Here, $\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs}$, where $\mathcal{C} = \mathbb{R}^2 \times \mathbb{S}$ is the configuration space and $\mathcal{C}_{obs}$ is the configuration space obstacle as defined in class. In other words, the sequence of line segments corresponding to p must ensure that the robot remains entirely within the workspace and avoid collisions with any obstacle through its motion.

1. (15 points for COM S 4760, 10 points for COM S 5760) Can the problem of computing the sequence of positions p be formulated as a discrete planning problem?

   Explain your reasoning by addressing the following points:

- Provide a formal, mathematical definition of the free configuration space $\mathcal{C}^p_{free}$. Here, the superscript p indicates that the focus is solely on computing the sequence p of positions rather than the full path τ. So $\mathcal{C}^p_{free}$ is not necessarily the same as $\mathcal{C}_{free}$ defined earlier. If your definition of $\mathcal{C}^p_{free}$ differs from $\mathcal{C}_{free}$, explain the reasons for this difference.
- Specify whether $\mathcal{C}^p_{free}$ is countable, and if so, whether they are finite.
- Describe how to obtain the sequence $p = (x_0, y_0)(x_1, y_1) \ldots (x_n, y_n)$ once a path $\tau^p : [0, 1] \to \mathcal{C}^p_{free}$ is found, assuming that when projecting τ^p onto the xy-plane, it corresponds to a sequence of line segments.

2. (15 points for COM S 4760, 10 points for COM S 5760) Solve the planning problem using RRT:

   Use the single-tree search outlined in Section 5.5.3 to compute a path p within $\mathcal{C}^p_{free}$, starting from q^p_I and ending at q^p_G. Here, q^p_I and q^p_G are the projection of q_I and q_G, respectively, on the xy-plane.

   You should check periodically if the tree can be connected to q^p_G. This can be easily done by setting $\alpha(i)$ as q^p_G with a certain probability pr. For example, the book recommends $pr = 0.01$. You should have this as a parameter in your code. Once q^p_G is successfully added to the tree, quit the loop and compute the path p from q^p_I to q^p_G.

   Implementation Requirements: You are required to implemented a class named `CircularMobileRobotPathPlanning` with the following methods in a file called `midterm.py`.

   - constructor:

     ```
     __init__(
         self,
         robot_radius: float,
         Xmax: float,
         Ymax: float,
         circular_obstacles: list
     )
     ```

     – `robot_radius` is the radius r of the robot.
     – `Xmax` and `Ymax` define the boundaries of the workspace, corresponding to X_{max} and Y_{max}, respectively.
     – `circular_obstacles` is a list of circular obstacles, each specified as a tuple `(xo, yo, ro)`, where `xo`, `yo`, `ro` represent the obstacle's center coordinates x_o, y_o and radius r_o, respectively.

   - `plan` method:

     ```
     plan(
         self,
     ```

```
    qI: tuple,
    qG: tuple
)
```

Here, `qI` and `qG` are tuples of three elements representing the initial and goal configurations, i.e., `qI = (xI, yI, tI)` and `qG = (xG, yG, tG)` where `xI`, `yI`, `tI`, `xG`, `yG`, `tG` are floats corresponding to x_I, y_I, θ_I, x_G, y_G, θ_G, respectively.

This method should return a tuple `(G, p)` where:

- `G` is a graph with a `draw(self, ax)` function that draws the graph on the specified axis `ax`
- `p` is the computed path $p = (x_0, y_0)(x_1, y_1) \ldots (x_n, y_n)$, represented as a list of tuples `[p[0], p[1], ..., p[n]]`, where each `p[i]` is a tuple `(xi, yi)`, corresponding to the position (x_i, y_i) in p.

An example `main.py` file is provided, along with a file `draw_cspace.py` containing the plotting functionalities. Your implementation should be compatible with these files, so that when running

```
python main.py
```

a plot similar to that in Figure 1 is generated. Additionally, your implementation will be tested with different obstacle configurations and varying values of r, X_{max}, Y_{max}, q_I, q_G.

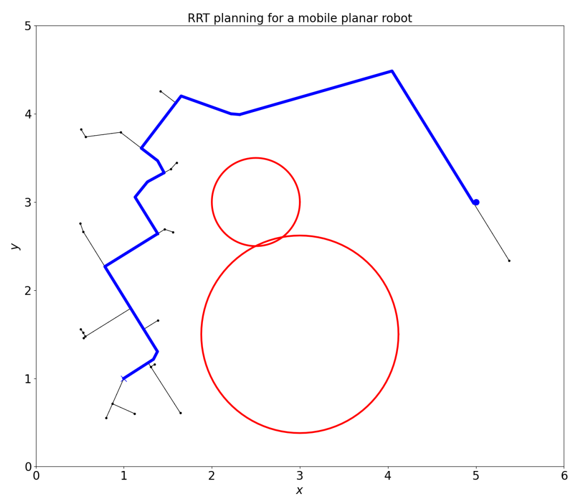

Figure 1: The result of RRT planning with $p = 0.1$, showing the tree and the path from x_I to x_G.

3. (COM S 5760 only, 10 points) Consider a scenario where two robots b_1 and b_2, each with a radius r, operates within this 2D rectangular workspace.

   In addition to finding individual paths for each robot, we must ensure that their paths are collision-free, meaning that the robots must not collide with each other at any point during their execution.

   Discuss the potential coordination challenges that arise in multi-robot motion planning by addressing the following points:

   - Provide a formal, mathematical definition of the configuration space $\mathcal{C}_2$ for this multi-robot motion planning problem, where the subscript 2 indicates the setup involving two robots.
   - Provide a formal, mathematical definition of the free configuration space $\mathcal{C}_{free,2} \subseteq \mathcal{C}_2$ that excludes the configurations where any robot collides with circular obstacles, workspace boundaries, or the other robot.
   - Identify assumptions that enable the formulation of a discrete motion planning problem for this two-robot setup. Describe the resulting state space X, the set of actions U, the action space $U(x)$ for each state $x \in X$, and the state transition function $f : X \times U \to X$.

     Unlike in Problem 3 of Homework 3, you are explicitly **NOT** allowed to assumed that collisions—whether with the workspace boundaries, circular obstacles, or the other robots—can only occur when the robot's center is precisely at a grid point. Collisions must be checked continuously throughout the robot's execution, including when the robots move between grid points.

   Note: A path for a robot may be represented either as a full path $\tau : [0, 1] \to \mathcal{C}_{free,1}$, where $\mathcal{C}_{free,1} \subset \mathbb{R}^2 \times \mathbb{S}^2$ is the free configuration space for a scenario where there is a single robot within this 2D rectangular workspace, or as a sequence of robot's center positions, as described before. When answering the above points, specify how you will represent the paths.