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 $p = (x_0, y_0)(x_1, y_1) \dots (x_n, y_n)$, 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 $l_1 l_2 \dots l_n$ 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 $p = (x_0, y_0)(x_1, y_1)\dots(x_n, y_n)$ 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.

Solution From Homework 3, each configuration of the robot can be represented by a triplet (x, y, θ), where $(x, y) \in \mathbb{R}^2$ specifies the position of the robot's center and $\theta \in [0, 2\pi)$ specifies the robot's orientation. We have also established that if we neglect the obstacles, the robot can move from any configuration x, y, θ to any configuration (x', y', θ') by first rotating in place (i.e., keeping the position of its center fixed) so that its orientation lines up with a vector pointing from (x, y, θ) to (x', y', θ'). Then, it can move on a straight line so that its center is at (x', y'). Finally, it rotates in place so that its orientation is θ'.

Since all the rotations needed for such these transformations are in place, i.e., the position of the robot's center is fixed and whether the robot is in collision does not depend on its orientation, we can neglect the orientation when computing a sequence p of robot's center positions along a path.

With this, we can define the free configuration space as

$$\begin{aligned}\mathcal{C}^p_{free} \quad = \quad & \{(x, y) \in [r, X_{max} - r] \times [r, Y_{max} - r] \mid \\ & (x - x_o)^2 + (y - y_o)^2 > (r + r_o)^2, \forall o \in O\}\end{aligned}$$

This set is uncountable.

Therefore, τ^p corresponds to a sequence $l_1 l_2 \dots l_n$ of line segments in the xy-plane, where l_i connects (x_{i-1}, y_{i-1}) to (x_i, y_i) and $(x_0, y_0) = (x_I, y_I)$ and $(x_n, y_n) = (x_G, y_G)$. As a result, p can be simply obtained by taking the sequence of endpoints of these segments, i.e., defining $p = (x_0, y_0)(x_1, y_1)\dots(x_n, y_n)$.

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_G^p is successfully added to the tree, quit the loop and compute the path p from q_I^p to q_G^p.

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 $p = (x_0, y_0)(x_1, y_1) \dots (x_n, y_n)$, 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.

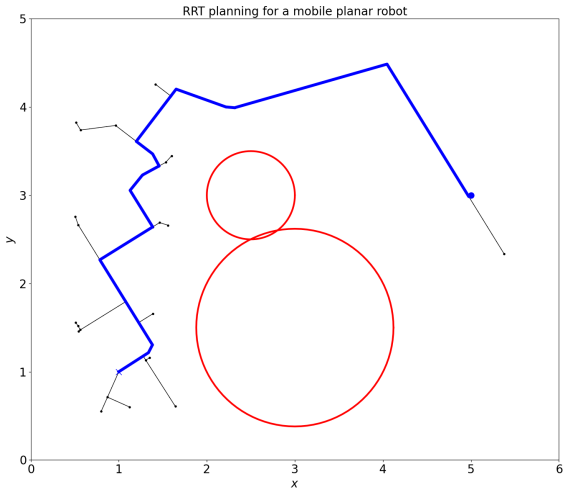

Figure 1: The result of RRT planning with $p = 0.1$, showing the tree and the path from x_I to x_G.

Solution See `midterm.py` and `obstacle.py`, which requires the following files from the solution of Homework 4.

- `edge.py`
- `geometry.py`
- `graph.py`
- `planning.py`
- `queue.py`

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.

Solution We consider representing a path of each robot b_i as a sequence of b_i's center positions.

The configuration space $\mathcal{C}_2$ can be defined as the product of the configuration spaces of the individual robots, i.e., $\mathcal{C}_2 = \mathbb{R}^2 \times \mathbb{R}^2$. A joint configuration of the two robots can be represented by a tuple (x_1, y_1, x_2, y_2), where (x_1, y_1) is the position of b_1's center and (x_2, y_2) is the position of b_2's center.

The two robots are in collision when $(x_1 - x_2)^2 + (y_1 - y_2)^2 \leq 4r^2$.

Based on $\mathcal{C}_{obs}$ computed in Problem 2 of Homework 3, the free configuration space for the two-robot setup is then given by

$$\begin{aligned}
\mathcal{C}_{free,2} \;=\; & \{(x_1, y_1, x_2, y_2) \in \mathbb{R}^4 \mid \\
& x_1, x_2 \in [r, X_{max} - r] \text{ and} \\
& y_1, y_2 \in [r, Y_{max} - r] \text{ and} \\
& (x_1 - x_2)^2 + (y_1 - y_2)^2 > 4r^2 \text{ and} \\
& (x_1 - x_o)^2 + (y_1 - y_o)^2 > (r + r_o)^2 \text{ and} \\
& (x_2 - x_o)^2 + (y_2 - y_o)^2 > (r + r_o)^2, \forall o \in O\}
\end{aligned}$$

To formulate a discrete motion planning problem, we discretize the workspace into a finite set of grid points $W = \{(x, y) \mid x \in \{X_0, X_1, \ldots, X_M\}, y \in \{Y_0, Y_1, \ldots, Y_N\}\}$. For simplicity, we assume that $X_{i+1} - X_i = \delta_x$ for all $i \in \{1, \ldots M - 1\}$ and $Y_{i+1} - Y_i = \delta_y$ for all $i \in \{1, \ldots N - 1\}$, where $\delta_x, \delta_y \in \mathbb{R}_{>0}$ are positive constant.

We then make similar assumptions about the robots' initial and goal positions, as well as their movements, as those outlined in Problem 3 of Homework 3. Specifically, we assume the initial and goal positions of each robot are located at one of

these grid points. Furthermore, each robot can only move between these points, transitioning only to directly neighboring points in the grid—either horizontally or vertically. Under these assumptions, the state space, the set of actions, and the state transition function can be defined as

$$\begin{aligned}
X &= \{(x_1, y_1, x_2, y_2) \in W \times W \mid \\
&\quad x_1, x_2 \in [r, X_{max} - r] \text{ and} \\
&\quad y_1, y_2 \in [r, Y_{max} - r] \text{ and} \\
&\quad (x_1 - x_2)^2 + (y_1 - y_2)^2 > 4r^2 \text{ and} \\
&\quad (x_1 - x_o)^2 + (y_1 - y_o)^2 > (r + r_o)^2 \text{ and} \\
&\quad (x_2 - x_o)^2 + (y_2 - y_o)^2 > (r + r_o)^2, \forall o \in O\} \\
U &= \{(0, \delta_y), (0, -\delta_y), (-\delta_x, 0), (\delta_x, 0)\} \times \\
&\quad \{(0, \delta_y), (0, -\delta_y), (-\delta_x, 0), (\delta_x, 0)\} \\
f((x_1, y_1, x_2, y_2), (u_{1,x}, u_{1,y}, u_{2,x}, u_{2,y})) &= (x_1 + u_{1,x}, y_1 + u_{1,y}, x_2 + u_{2,x}, y_2 + u_{2,y}), \\
&\quad \forall (x_1, y_1, x_2, y_2) \in X, (u_{1,x}, u_{1,y}, u_{2,x}, u_{2,y}) \in U
\end{aligned}$$

To define the action space $U(x) \subseteq U$ for each state $x \in X$, we introduce two distance functions: $D_{lsls} : \mathbb{R}^8 \to \mathbb{R}_{\geq 0}$ and $D_{pls} : \mathbb{R}^6 \to \mathbb{R}_{\geq 0}$.

For any $x_1, y_1, x_1', y_1', x_2, y_2, x_2', y_2' \in \mathbb{R}$, $D_{lsls}(x_1, y_1, x_1', y_1', x_2, y_2, x_2', y_2')$ is the minimum distance between two line segments: one connecting (x_1, y_1) and (x_1', y_1') and the other connecting (x_2, y_2) and (x_2', y_2').

For any $x, y, x_1, y_1, x_1', y_1' \in \mathbb{R}$, $D_{pls}(x, y, x_1, y_1, x_1', y_1')$ is the minimum distance between the point (x, y) and the line segment connecting (x_1, y_1) and (x_1', y_1').

Using these distance functions, we establish two key conditions that are sufficient to ensure collision-free movement. First, the condition

$$D_{lsls}(x_1, y_1, x_1', y_1', x_2, y_2, x_2', y_2') > 2r$$

is sufficient to ensure that as b_1 transitions from (x_1, y_1) to (x_1', y_1') and b_2 transitions from (x_2, y_2) to (x_2', y_2'), there is no collision between them.

The condition

$$D_{pls}(x_o, y_o, x_i, y_i, x_i', y_i') > r + r_o,$$

where $i \in \{1, 2\}$ and $o \in O$, is sufficient to ensure that b_i does not collide with obstacle $o \in O$.

Note that, based on the grid definition and the movement constraints for the robots, ensuring that there are no collisions with the workspace boundaries at the endpoints of each transition is sufficient to guarantee that there are no collisions with the boundaries throughout the entire transition.

Thus, we can write the action space $U(x)$ for each state $x \in X$ as

$$\begin{aligned} U(x_1, y_1, x_2, y_2) \;=\; & \{u \in U \mid f(x,u) \in X \text{ and} \\ & D_{lsls}(x_1, y_1, x_1', y_1', x_2, y_2, x_2', y_2') > 2r \text{ and} \\ & D_{pls}(x_o, y_o, x_1, y_1, x_1', y_1') > r + r_o \text{ and} \\ & D_{pls}(x_o, y_o, x_2, y_2, x_2', y_2') > r + r_o, \forall o \in O, \text{ where} \\ & (x_1', y_1', x_2', y_2') = f(x_1, y_1, x_2, y_2)\} \end{aligned}$$