Purpose

In this project, you will use the robot's sensors to find a nearby wall, follow it by keeping it on the robot's right side. On reaching the next perpendicular wall, your robot must follow a path back, like mowing a lawn.

Task

You must create a small Python script that publishes and subscribes to topics generating ROS2 messages to drive a simulated differential-drive robot (the iRobot Create 3). The robot's behavior must respond to simulated button presses as follows:

→ When the "Power" button is pressed, the robot must begin execution. Execution consists of the following:

1. Create begins a spiral search for the nearest wall (the wall where the Creates are usually charging).
2. On finding the wall, the Create aligns itself parallel, with the wall on its right side.
3. Once parallel, the Create translates until it finds the next perpendicular wall in a CCW ordering (the lab wall containing the hall door).
4. On finding that wall, the robot
   (a) Drives backwards $\approx$ 5cm and
   (b) Executes a U-turn:
      i. Rotates $+\frac{\pi}{2}$ rad,
      ii. Drives forward $\approx$ 40cm,
      iii. Rotates $+\frac{\pi}{2}$ rad,
      iv. Begins driving forward; translating back, in a "Boustrophedon" pattern.

→ If the robot comes into contact with any object with its bump sensor, it must immediately stop, drive backwards $\approx$ 5cm, and

- If executing spiral search, begin aligning parallel.
- If executing wall-following before the second wall is reached, attempt to recover and resume.
- If executing end turn, attempt to recover and resume.
- If executing second pass, resets i.e., ready to execute spiral search again.

If the robot does not immediately stop and cease hitting any obstacle, a 20% penally will be applied.

Deliverables

You may provide the following files and are required to provide `main.py`. These are the same as provided from:

```
boustrophedon/
+-- boustrophedon/
|   +-- main.py
|   +-- button_node.py
|   +-- hazard_node.py
|   +-- ir_node.py
|   +-- odom_node.py
```

Any file (other than `main.py`) that you do not provide, will be provided for you.

Points

Your code will be tested using the `create3_gazebo` simulation using the R-Viz application and with the dock node deleted/removed.

Note that your robot will be given a random start location and heading. It will be located somewhere on or to the right of the line $x = y$(slope $m = 1$).

Correctness

`boustrophedon/boustrophedon/main.py` 20%

10% for correctly responding to a "Power Button" press.

10% for correctly publishing Twist messages to `cmd_vel` in response.

-100% if it crashes; ever.

Performance

Correctly spiraling into the wall. 20%

10% for spiral

10% for finding wall

Correctly aligning with the wall. 20%

10% for correctly making an attempt

10% for correctly aligning

Correctly following the wall. 20%

10% for correctly making an attempt

10% for correctly finding second wall

Correctly performing U-Turn at the second wall. 20%

5% for correctly making an attempt

5% for correctly completing U-turn

10% for correctly returning along the indicated path

Appendix

- IR Sensor: Note that the IR sensor node as provided assumes that sensor returns:

  [left, left-front, center-left-front, center, center-right-front, right-front, right]

  The order of the Create 3. The simulation returns a different order. I would recommend looking into how to map them to the same ordering, *see* ***`boustrophedon/boustrophedon/ir_node.py`*** for details.

- State-ful Design

  It is highly recommended that you employ a state-ful design including the states:

  1. `SPIRAL_SEARCH`
  2. `ALIGN_WALL`
  3. `FOLLOW_WALL`
  4. `MAKE_UTURN`

Cont.

5. `RETURN`

This will allow you to respond to different sensor input in different ways.

- Align with wall

  The goal here, essentially, is to maximize your right IR beam.

  If you find the 5cm backup guide too far, you may reduce it. Be careful. If you do not back up enough and fast enough to turn off your bump sensor, you may have mixed results.

- Wall-following

  You may use a PD-controller here to maintain a given IR intensity.

The End.