ME 4451 Lab 7 Handout

Anirban Mazumdar, Bryan Blaise, Nikhil Nandikanti

Spring 2024

1 Objectives

1. Learn how to implement motion control of mobile robots to execute a straight path
2. Perform object detection and avoidance
3. Execute a navigation task from one end of the lab to another
4. Experiment with advanced path planning algorithms for a mobile robot.

2 Prelab Assignment

- Read Lab 7 Handout
- Write the `turtlebotGoToWaypoint function`
- Write the `turtlebotfaceDirection function`

3 Straight Line Motion

In lab 6, you were tasked with executing square path with the Turtlebot. Chances are, you noticed the default functions `turtlebotGoDistance()` and `turtlebotTurnAngle()` performed poorly and could be improved upon. But even if finely tuned, these methods focus on executing linear or angular motions in isolation. For this lab, you will generate a function that allows you to both travel a particular distance while maintaining a specified heading. This will be necessary when trying to navigate across long distances. You will need to make use of odometry to perform this.

Generate a function that commands your Turtlebot to execute a straight path of 10 ft. This is outlined by the yellow tape in lab. Your robot must stay between the tape and is not allowed to touch it. The robot must start with its wheels on the starting line and end with at least one part of the robot on the line at the end of the tape. Demonstrate your robot performing this task to your TA.

Figure 1: Straight path to be executed by Turtlebot. Maintain your robot's heading throughout the path and stay between the lines

4 Obstacle Detection and Avoidance

In lab 6, you learned how to use a subscriber to read the lidar data transmitted along the `scan` topic. You will now expand upon this code to learn how to detect the presence of an object impeding the robot's forward progress and circumnavigate it. You should eventually incorporate this ability into your full navigation task, but as a preliminary task, you will perform obstacle avoidance in isolation.

Place an object one or two meters in front of the robot. This object needs to be at least as tall as the spinning disk (the lidar sensor) is on the robot. Program your robot to advance forward until it detects the object. It is up to you to decide how close you want the robot to get before it stops but keep in mind the minimal detection distance of the lidar is 12 cm. After the robot has detected the object, program it to get around the object and end up on the other side of the object, facing the initial forward direction. You are free to use any strategy besides hard coding specific distances to travel based on the dimensions of the object. Your method should be independent of the width and depth of the obstacle since the TA will choose an object at random for the demonstration. Moreover, do not assume the obstacle is rectangular. Once you can reliably perform this task, demonstrate this to your TA and walk him/her through your code.

5 Navigation

Your final objective will be to generate a program that executes a navigation task on the robot. The robot will have to navigate from one end of the room to the other. Here is the map of the room showing the beginning and ending location of the path:

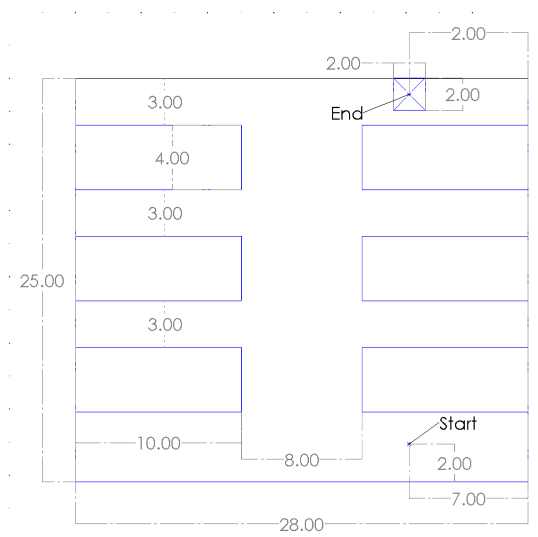

Figure 2: Map of lab. Note the starting pose includes both position and orientation. Dimensions are in ft

In the Lab 7 folder under the "Files" tab, you can access this map. Dimensions are in feet. The entrance of the lab is at the bottom of the map. Note that the starting pose includes both the position and orientation of the robot. The goal position is a square with 2 ft sides. Ending up anywhere in that square is considered a success. At lab, you will see the starting and ending locations marked in tape on the floor. Start with your wheels on the tape line and your robot facing the direction of the arrow. Recall that the Turtlebot's odometry functions consider the origin of the global frame to be the pose of the robot at the robot's startup. You will need to perform transformations to create a new global origin that matches those in the map. Once you can reliably execute the navigation task, demonstrate it to your TA for credit.

5.1 Extra Credit: Final Orientation

The navigation task only requires your robot to end inside the final square in an unspecified direction. For extra credit, program your robot to end its path facing a particular direction. In this case, program the robot to face the wall containing the entrance of the lab.

5.2 Extra Credit: Obstacle Avoidance During Navigation

For additional extra credit, program your robot to perform obstacle avoidance during the navigation task. While your robot is performing its navigation task, the TA will suddenly place an object in front of your robot at his discretion. Your robot should detect the object, avoid it, and resume its path and proceed to its goal. Your robot must both avoid the obstacle and complete the path to receive credit.