AuE 823: Sp'23: Autonomy Science and Systems

Department of Automotive Engineering

Venkat Krovi, Dhruv Mehta & Sumedh Sathe

ASSIGNMENT 3A

(Due: Tuesday, 7th March, 2023. 12:59PM)

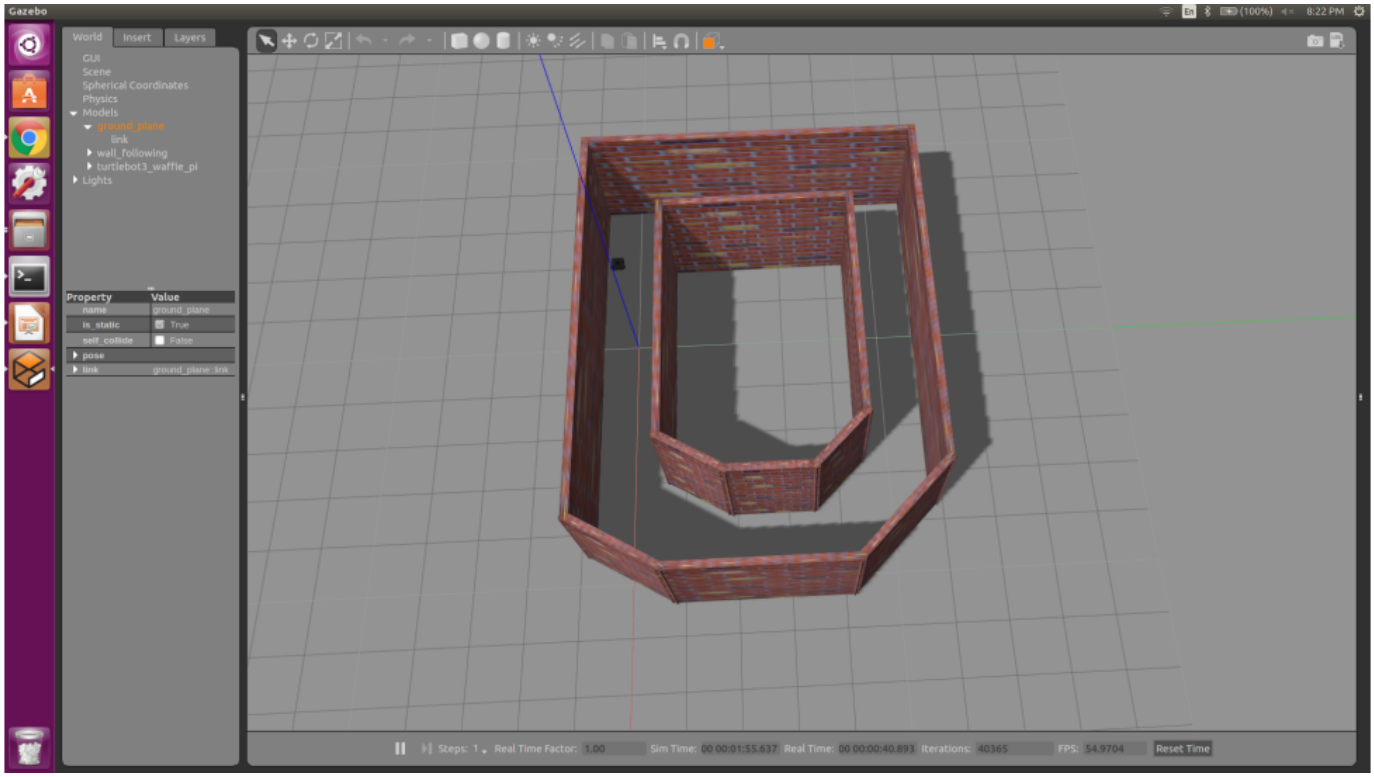
Learning outcomes:

1. Manipulating scan data for navigation
2. Implementing P or PD controllers in Python

Setup : This assignment builds on the last assignment. There are three parts. To get started create a new ROS package called "assignment3a_wallfollowingandobstacleavoidance" in your catkin workspace.

PART 1: Wall following

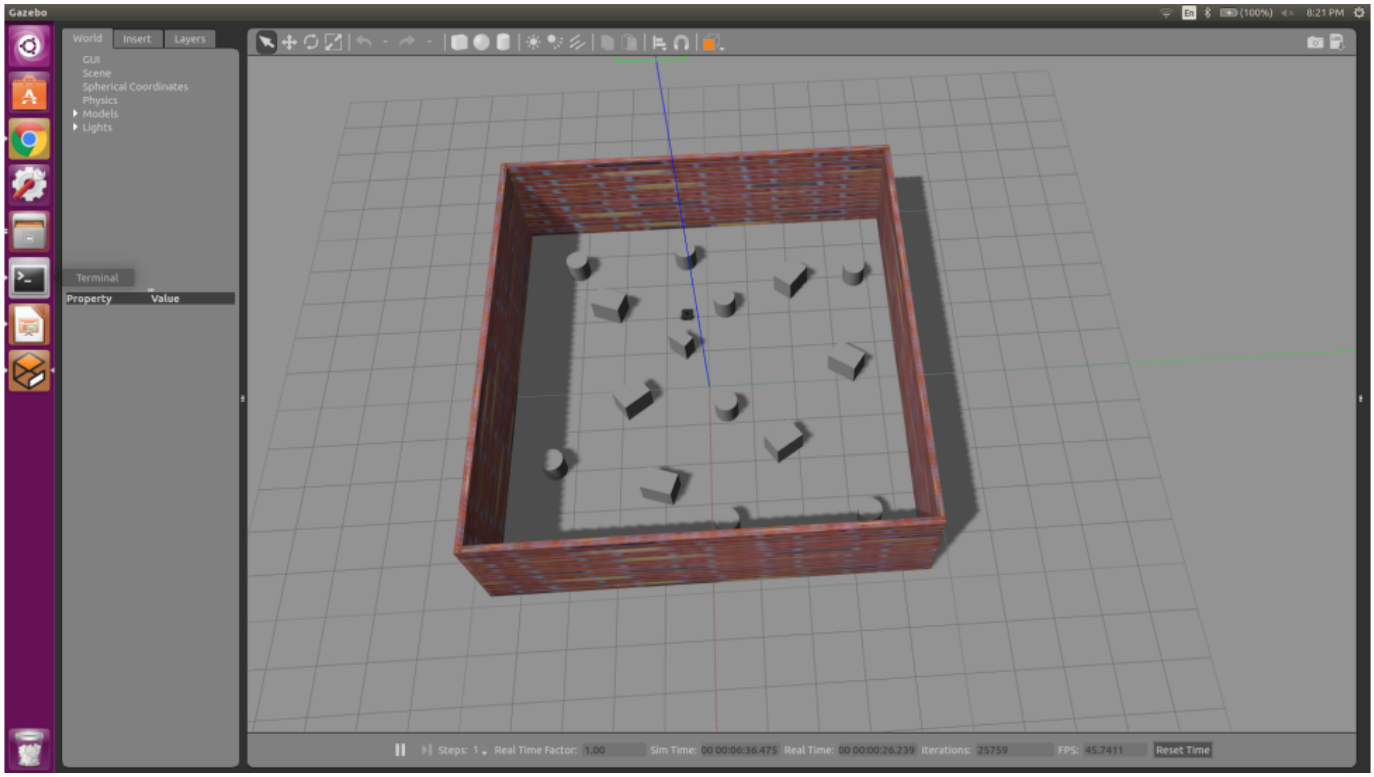
Download the world files uploaded on canvas. The world for wall following:



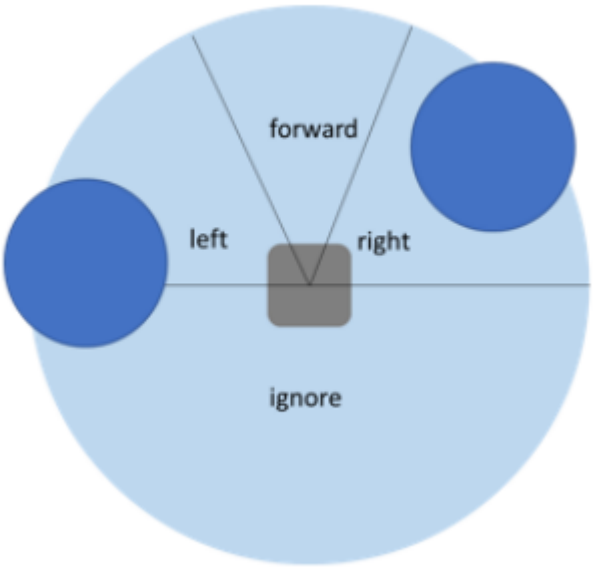
In the first part of this assignment you will create a python code wall_follower.py that extracts the /scan data and chooses data points that reflect the sides of the robot. Feed the robot a constant forward velocity and attempt to avoid the wall. You can write a simple P controller to maintain the equal distances from each face of the wall, or follow a single face while maintain a fixed (threshold) distance from it. Or you can come up with your own novel method.

PART 2: Obstacle avoidance

The world for obstacle avoidance:



In this part you will create a python code wander.py where you implement obstacle avoidance using scan data.



You can separate the scan into different sectors. For example: In the right sector, the sum of all data points, divided by the number of data points (mean data) is less than that for the left sector or the forward sector -- this implied that should move forward- left. The steering angle of the robot can be a function of these three

means. You may choose other approaches -- you may divide the data into several sectors or choose to not use sectors at all. The robot can be given a constant forward velocity. There is no goal location, so the robot will be expected to simply wander around the room avoiding obstacles.

Part 3: Obstacle avoidance in real world

Implement the obstacle avoidance code on the actual Turtlebot. There are cardboard boxes in our lab that can be used as obstacles and set up in the 4th floor mess area whenever you are testing out your code.

PART 4:

Write a python code that executes an "emergency braking" maneuver. The TurtleBot3 must move in a constant forward velocity unless it meets an obstacle. The obstacle is a wall in the path of the robot. If the robot gets too close to the wall it should execute an "emergency brake" (publish 0 to `/cmd_vel`)

1. Modify the turtlebot3 empty world to include a wall directly in the path of the robot but some distance away. Call it `turtlebot3_wall.world`
2. Write a python code which publishes a constant linear velocity to `/cmd_vel` . The code must also subscribe to the `/scan` topic and extract the center-most value of the scan data array.
3. If the extracted `/scan` data point falls below a defined threshold, then publish 0 to `/cmd_vel` , else continue.
4. Put it all in a launch file.

SUBMISSION:

1. Videos of all parts
2. Save the video in your workspace in a folder called `/videos` . (You will be graded for the video submission).
3. Launch files to run each part
4. Create a README.md with a brief explanation of the launch file and what the launch file implements. The README file should also clearly contain the command the TA should enter into the terminal to run each part of the homework!
5. Push this to your GitHub repo. Submit the link to the repo on Canvas.