WPI

RBE595 – Advance Robot Navigation
Assignment 01

Kartik Patath, kpatath@wpi.edu
Sunday 3 February 2020

1 Project Objective

Write a computer program that simulates the motion of a robot in a 2D environment. The robot should be able to move in any direction with a constant velocity, while avoiding obstacles. (Random motion)

2 Environment Setup

For this task I used GAZEBO-2 with ROS as the simulator, as it fits perfectly with the given task description. I used a differential drive system with one castor wheel as support. The robot looks as follows

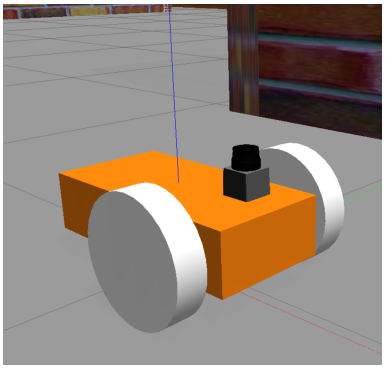

Figure 1: Differential drive robot with two drive wheels (in front) and. a castor wheel (in the rear) along with a Hokuyo Lidar (on the top)

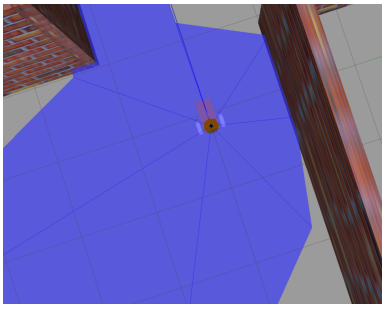

Figure 2: Sensor output of the Lidar

The robot is mounted with a Hokuyo lidar which is our sensor used for navigation. For this assignment we were asked to check for only 8 directions and hence the sensor was customized accordingly

so that it samples only in 8 directions N, NE, E, SE, S, SW, W and NW (in 8 directions, *i.e.* 45 degrees separation) and it looks as shown in Fig. 2. The complete sensor output is as follows

```
header:
seq: 49
stamp:
    secs: 208
    nsecs: 926000000
    frame_id: "hokuyo"
angle_min: -3.14159011841
angle_max: 3.14159011841
angle_increment: 0.897597134113
time_increment: 0.0
scan_time: 0.0
range_min: 0.10000000149
range_max: 30.0
ranges: [13.228289, 1.756185, 6.927513, 5.431076, 1.812129, 1.042050,
         1.349644, 1.326585]
intensities: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

As we can see, the "*ranges*" gives us 8 different values which correspond to the eucidian distance in 8 different directions.

3 Experimental Setup

Here are some parameters and constraints that are imposed on the robot

- collision threshold = 1 meter
- Robot velocity is 0.1 m/s in x-direction
- Laser scan range is 30 meters at max
- Walls of the environment are 20 meters high

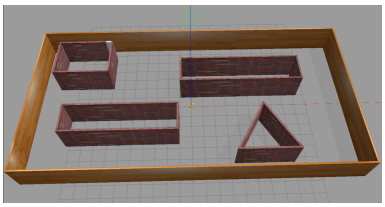

Figure 3: Gazebo world (3D)

4 Critical Questions

4.1 How well it avoids running into obstacles?

The algorithm that I used to avoid obstacles is, first i read all the 8 distances from the sensor and check if there is a collision. If there is no collision, the robot is asked to move in x-direction with a constant velocity of 0.1 m/s. If there is an obstacle, I send a halt command which stops the robot and then I make it change its direction with an angular velocity of 0.05 rad/s and now it resumes the constant velocity command.

Particularly if it is a sharp object with a 30 degree edge, there are two conditions that can occur. One, when the sharp edge (30 deg) comes in between two directions which are separated with 45 degrees. In such case the robot cannot detect these edges until they are at a distance of 1 meter. But, as soon as it reaches the 1 meter distance, the robot will be aware of two different walls of the edge (walls that contribute to the sharp 30 degree point) and hence the robot starts to make a turn to avoid it, as shown in the video. Second case is in which the 30 degree edges are detected by atleast one of the sensors and the robot starts rotating in a different direction to avoid it.

4.2 How well it recovers if it does run into an obstacle?

If the robot accidentally runs into an obstacle, as explained in the obstacle avoidance section it turns with an angular velocity of 0.05 rad/s and it keeps on doing that until it clears the obstacle from its field of view.

4.3 How well it evaluates the collision avoidance strategy?

The current setup evaluates collision avoidance based on sensor resolution. The more the sensor resolution the higher will be the chance to detect sharp obstacles and thus the robot will be able to navigate properly.

4.4 How fast it is able to move in general?

In general it moves at a constant velocity of 0.1 m/s.

4.5 Can we stream position / translation of the robot ?

Yes, It can be done using the following commands :
" rostopic echo /gazebo/modelstates "
" rostopic echo /odom"

5 How to run the code?

Please open the following URL and follow the README instructions to run the simulation

`https://github.com/kartiksonu/Obstacle_avoidance_randomwalk`