Homework 3

ENPM690 - ROBOT LEARNING

Submitted by
Nalin Das
UID - 116698290
M.Eng. Robotics

Contents

Introduction

Problem Statement

The Problem statement is divided into two parts as follows:

Part 1. Program a simple robot vehicle in a simulated environment (robot simulation tools and libraries may be used). Your simulated robot should exhibit at least one sensor input (e.g., forward-looking range sensor that returns the distance to the nearest obstacle) and two control outputs (e.g., left and right wheels, or speed and direction of vehicle motion). Show that you can drive your robot around through mouse or keyboard Inputs.

Part 2. Add a programmed behavior to your robot, such as following (or avoiding) a light, or wandering, while avoiding collisions with obstacles.

Robot Description

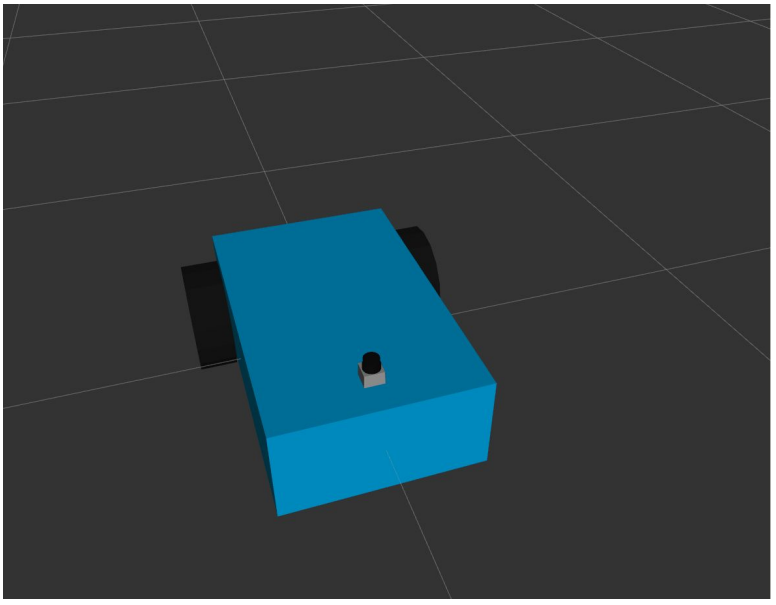

Fig. 1.0: Shady Robot

Robot Specifications

Robot Name: Shady
Drive Mechanism: Differential Drive
Model Description:

1. Two wheels with continuous joints
2. One Caster
3. Hokuyo Laser Rangefinder sensor mounted on top

PART 1

Modeling Steps

1. Created a URDF file for the Shady Robot and wrote the XML code for the Boxed Body and wheel's attached to the robot.

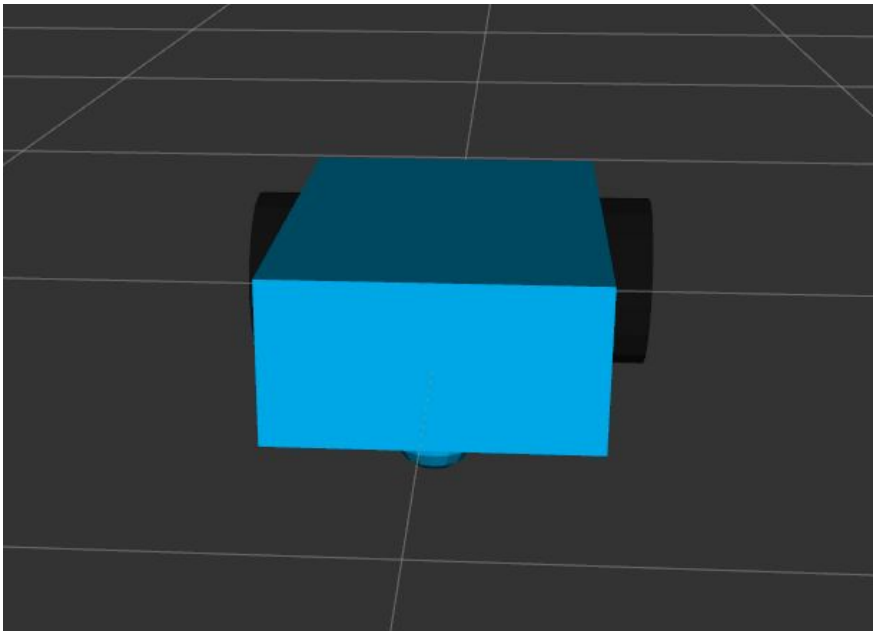

Fig. 1.1: Shady Robot Basic Design (RVIZ)

2. Added Gazebo Specific Tags in the URDF - Collision and Inertia, to the Robot URDF. This was done to render the model in Gazebo correctly.

3. Then, added the Gazebo Differential Drive ROS plugin in the URDF. This was done to interface the Robot with the Teleop Keyboard Controller. It uses the ros messages being published over the topic “/cmd_vel” . Thus we can control this Robot using the following keyboard controls:

```
Reading from the keyboard  and Publishing to Twist
!
---------------------------
Moving around:
   u    i    o
   j    k    l
   m    ,    .

For Holonomic mode (strafing), hold down the shift
 key:
---------------------------
   U    I    O
   J    K    L
   M    <    >

t : up (+z)
b : down (-z)

anything else : stop

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
```

Fig. 1.2: Teleop Keyboard Controls

4. Added the Hokuyo Laser RangeFinder to the Shady Robot’s URDF. Add all the relevant Gazebo Specific tags and Sensor Plugin. The sensors give distance measurements from any obstacle in its range, which is from -90 to 90 degree FOV. This FOV has a resolution of 0 to

720 points mapping -90 to 90 range. So, 0 corresponds to measurement at -90 degrees, 360 corresponds to 0 degrees and 720 corresponds to 90 degrees.

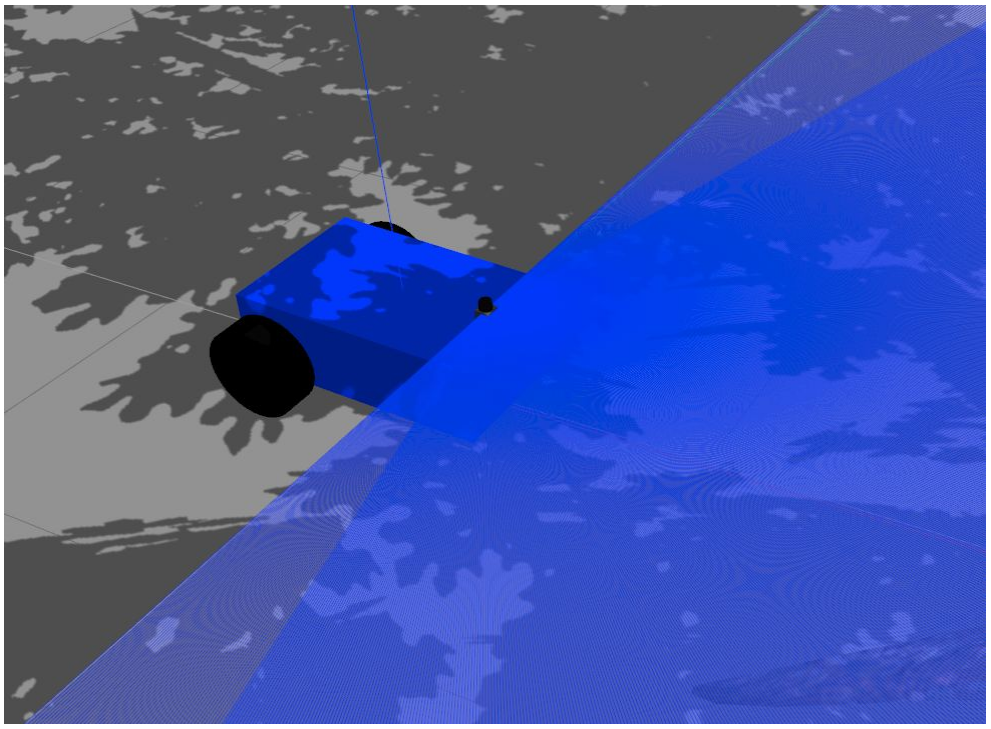

Fig. 1.3: Hokuyo Rangefinder Visualization

PART 2

Obstacle Avoidance

For the second part of the problem statement, I made a python script to design a basic velocity controller for the Robot to only move forward. The velocity controller was designed to publish linear and angular velocities as messages to the “/cmd_vel” topic.

After this, I modified the python script to add a rospy subscriber which subscribes to the ROS topic “/shady/laser/scan” where the Laser Scanned data is being published.

This data is then being used to set a condition on the controller. If the distance reading is less than 6 meters, the Robot moves backward and turns.

If the distance reading from the obstacle is greater than 6 meters, than the robot moves forward.

Github Link

The Github Repository Link for the Project is as follows:

https://github.com/nalindas9/enpm690/tree/master/Project_2/shady_ws

The Step by Step instructions to run the code and simulation is provided in the Readme file in the Repository itself.

The Simulation Videos are also uploaded to Github.

REFERENCES

1. ROS Robotics By Example - Carol Fairchild, Dr. Thomas L. Harman
2. Official ROS Documentation