ENPM 662
Project 1: CAD Modeling and Simulation Using Gazebo

Name: Chaitanya Kulkarni
UID: 119183502
Directory ID: chaitkul

1 Introduction

In this project, our main goal is to create a robot model in SolidWorks and move the robot using a closed loop controller. The objectives of this project are:

1. Build a robot model on SolidWorks and export it as URDF
2. Add LiDAR Sensor on the robot and show the LIDAR points in RViz.
3. Move the robot using teleop in the competition environment.
4. Write a closed loop controller to move the robot from point A(0,0) to B(10,10) using the IMU sensor data.

2 Steps

2.1 Building the Model in SolidWorks

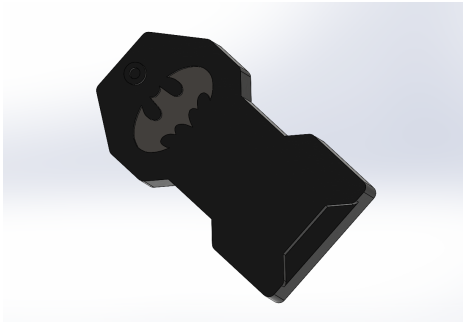

Figure 1: The main body of the robot

The main body of the robot in 38 inches long, 20 inches wide and the height of the body is 4.5 inches. The wheelbase of the robot is 20 inches.

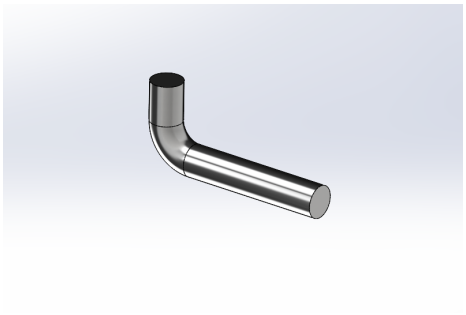

Figure 2: Axle for the wheels

The front two axles are joined to the body using revolute joints which is essential to steer the car, where as the two rear axles are joined to the main body using a fixed joint.

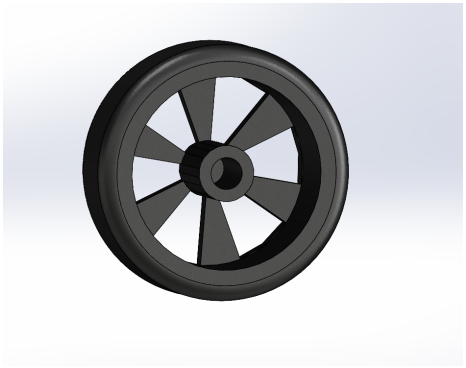

Figure 3: Wheels

This robot is a rear wheel drive where the velocity is give to the rear wheels. All the wheels are joined to the axles using continuous joints.

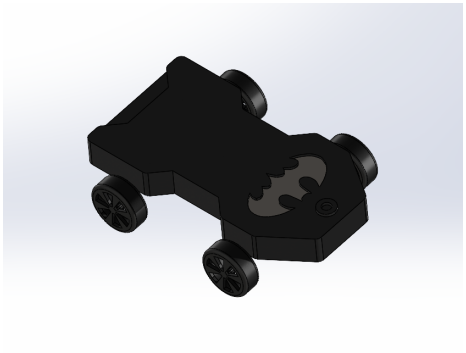

Figure 4: Robot Model

2.2 Exporting the model as URDF

After assembling the model in Solidworks, we define the relationships between the links according to the parent-child hierarchy, and then export the SolidWorks model as a URDF.

2.3 Adding Controllers to the Robot

Joint Group Position Controller has been used too control the front two wheels for the purpose of steering the robot. The Joint Group Velocity Controller has been used to give velocity to the two rear wheels which will drive the robot forward.

2.4 Adding the LIDAR and IMU sensors

First, we add a dummy link to the robot which a parent of the base link. We then add the LIDAR and the IMU sensors to the robot. These links are added with respect to the base link and their plugins are added to the urdf for getting the sensor data.

2.5 Driving the robot using teleop in competition setup

Controlling the robot using keyboard to complete one lap of in the competition environment.

2.6 Writing a closed loop controller

The objective is to move the robot from point A(0,0) to B(10,10). Hence we get the orientation and the velocity from the IMU sensors and use it to design a proportionalc ontroller for the robot. We get the difference between the desired orientation and the current orientation which is error and use propor-

tional gain to minimize this error. The steer angle that we publish to the robot is proportional to this error.

Similarly, we estimate the current position of the robot by integrating the velocity of the robot and the distance between the current position and the goal position is the error. We use another proportional gain to minimize this error. The linear velocity that we publish to the robot is proportional to this error.

3 Problems Faced

1. Faced issues while exporting the SolidWorks model as urdf where there was error of 'nan' values in transform due to missing links. Solved this issue by going back to SolidWorks and defining all the axis of rotations properly.
2. The robot was getting spawned in Gazebo, even the controllers were working fine but the front two wheels were missing in Rviz. Solved this issue by assigning velocity controllers to the front wheels in the urdf and control.yaml file
3. Estimating the robot position using the imu sensor data. Tackled this by getting the velocity by subscribing to the imu and integrating it.

4 Project Deliverables

4.1 Debug Launch

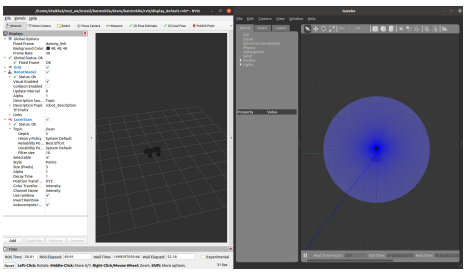

Figure 5: Robot spawned in Gazebo and Rviz

4.2 Gazebo Launch

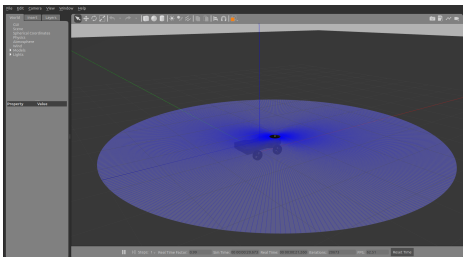

Figure 6: Robot spawned in empty world

4.3 Display Launch

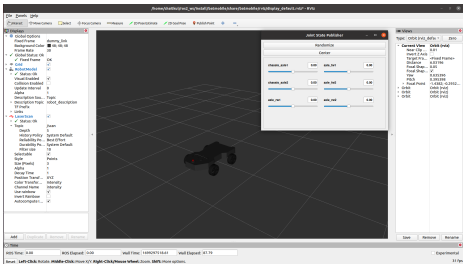

Figure 7: Enter Caption

4.4 Rviz LIDAR Visualization

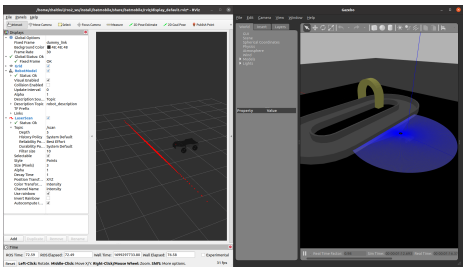

Figure 8: LIDAR data visualization

4.5 IMU plugin topic visualization

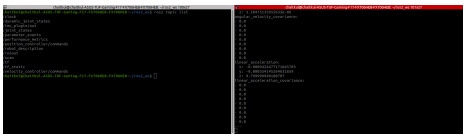

Figure 9: IMU plugin topic visualization

4.6 Error and control vs time graph

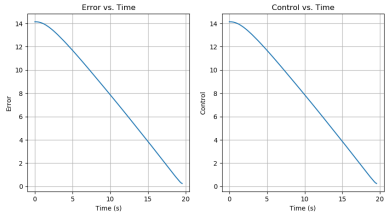

Figure 10: Error and control vs time graph

4.7 Teleop Video Link

Teleop Video

4.8 Closed Loop Controller

Closed Loop Controller

5 Personal Contribution

1. Building the SolidWorks parts and the assembly.
2. Exporting the assembly as URDF
3. Updating the URDF, launch and the control.yaml files and adding position and velocity and position controllers to the robot.
4. Adding LIDAR and IMU sensors to the robot, and getting the Rviz data visualization
5. Assistance in writing and debugging the controller script.

6 Improvements

The project was indeed well-documented, providing clear and precise instructions that were very helpful. However, it would have been even more valuable

if there was a simple tutorial explaining how to create a closed-loop controller using sensor data to obtain the required parameters.