Wall Following Robot Using ROS

Chaitanya Krishna

1. Introduction:

1.1. Overview

Robotics is part of today’s technological world. It is one of the fastest growing and interesting field. Robots are meant to simplify our daily work. They are designed to make human life less complex with sophisticated features. Autonomous Intelligent Robots are robots that can perform desired tasks in unstructured environments without continuous human guidance. The obstacle avoidance is primary requirement of this autonomous robot. The robot gets the information from surrounding area through mounted sensors on the robot. One of the popular robots is the obstacle or wall following robots.

Obstacles/wall following robots are widely used in industrial applications. A wall following robot is also an obstacle avoidance robot which follows the wall. If any obstacle comes in between, the robot will turn right or left depending upon the position of obstacle. An obstacle robot is an intelligent device which can automatically sense and overcome obstacles on its path. It is part of a robotic discipline with the objective of moving vehicles with the help of information from external devices such as sensors, lasers etc.

1.2. Purpose:

The main purpose of building this robot is to gain experience with Robotic Operating System (ROS) and its working. Here we are going to build a robot application that follows the walls in the simulator. It recognizes the one laser distance and it follows along the wall. The whole project is developed with the ROS framework. The final robot is tested on the Gazebo simulator.

Build a robot application that identifies the wall/obstacles and the robot has to follow along the identified wall for a certain distance. It should stop whenever it doesn't recognize the wall/obstacle. The robot will collect the wall/obstacle info from the laser distance sensor.

2. Literature Survey:

2.1. Existing Problem

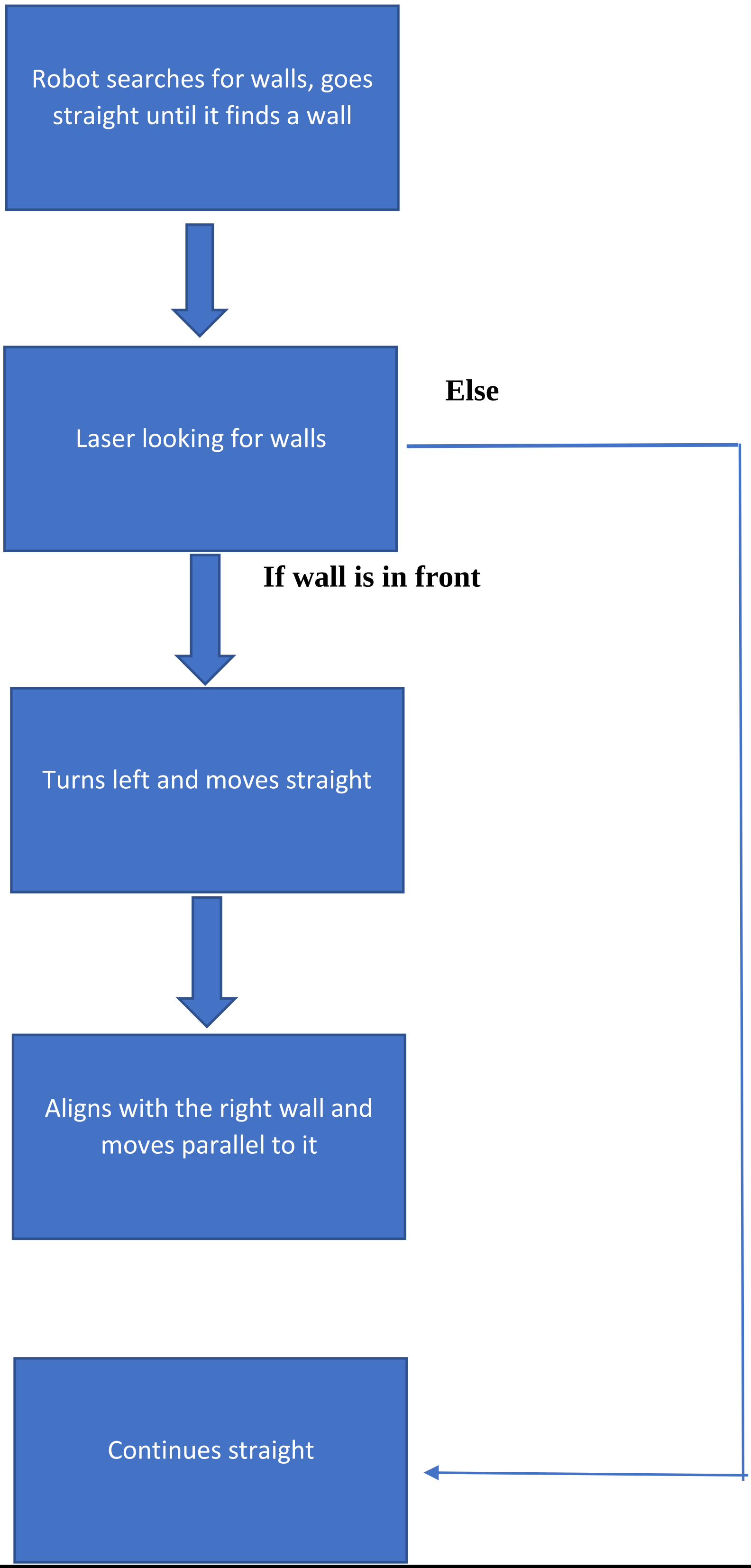
Obstacle detection can be considered as the central issue in designing mobile robots. Without the help of sensor or lasers it is impossible for a robot to traverse in unfamiliar environments without damaging itself.

2.2. Proposed Solution

A wall following Robot is designed which can detect obstacles in its path and maneuver along them without making any collision. The robot has the ability to follow along a wall and avoid any obstacles in front of it. The robot is designed and visualized using ROS and Gazebo. Being a fully autonomous robot, it successfully maneuvered in unknown environments without any collision.

3. Theoretical Analysis

3.1. Block Diagram



3.2. Software Designing

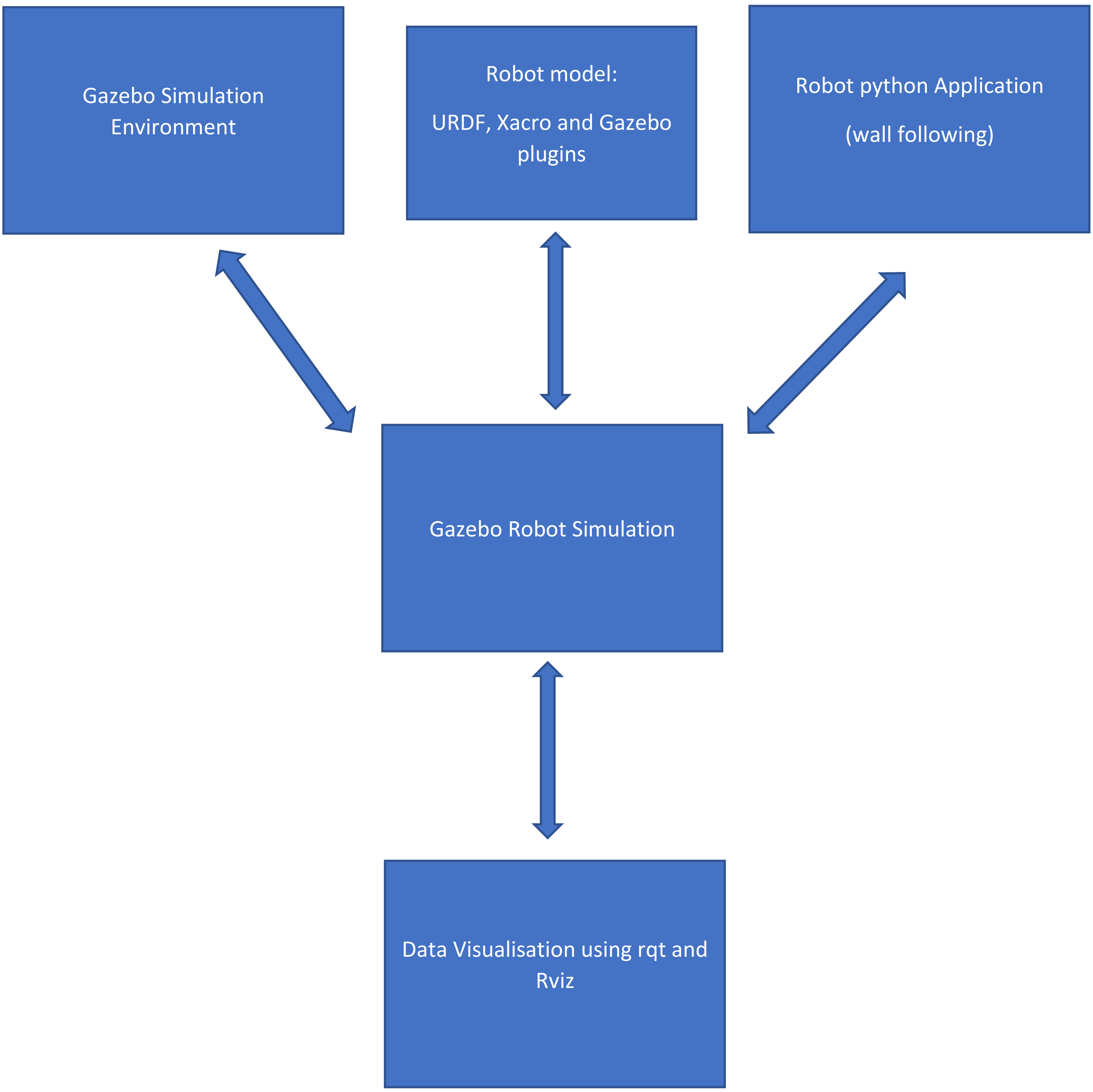
- For designing the robot, Gazebo and ROS are the primary softwares used.
- The entire coding is done in visual code studio and execution is performed in the linux terminal.
- To build a robot we need URDF, Xacro and Gazebo simulator. The URDF and Xacro files will help to integrate essential elements of the robot such as wheels, chassis and its features.
- The gazebo simulator is used to simulate the robot and the environment which is walls in our case.
- Rviz and rqt are additional visualization tools which help us visualize the robot and the environment. Also, the output of the laser device and camera can also be visualized using the above tools.

4. Experimental Investigation and Procedure

Our objective is to build a wall following robot from scratch using ROS and Gazebo simulator. So we follow the given steps in order to build the robot and apply wall following algorithm to the robot.

1. To build a robot, we first need to create a workspace. After the successful completion of the project workspace creation, we built a Robot model using URDF and Xacro techniques in the ROS, that should be interacting with the robot application in the Gazebo Simulator.
2. The robot consists of a hokuyo laser which is essential to detect the wall. It also consists of a camera that is mounted on top of the robot to visualize the surrounding environment.
3. For a robot to work, it needs to be placed in an environment. So we created an environment or a world for the robot using Gazebo simulator. We will create a simulation environment and visualize that in the Gazebo simulator.
4. In Gazebo we built walls so that the robot can sense and work alongside the walls when the wall following algorithm is embedded into the robot.
5. Now after coding the URDF, Xacro and launch files required for the robot to function, we used catkin_make to compile the workspace.
6. Using python language, an algorithm for wall following robot is coded and is moved to the workspace in which the robot features were developed. The python file is changed to executable mode for compiling and running the file.
7. After building all the files, the gazebo, rviz, python and rqt files are launched in the terminal one by one. The gazebo simulates the built robot along with the environment which consisted of walls. The rviz and rqt are used as simulation tools to visualize the robot movement and imagery. The python file which consists of wall following algorithm is executed which enables the robot to follow the algorithm and follow the walls.

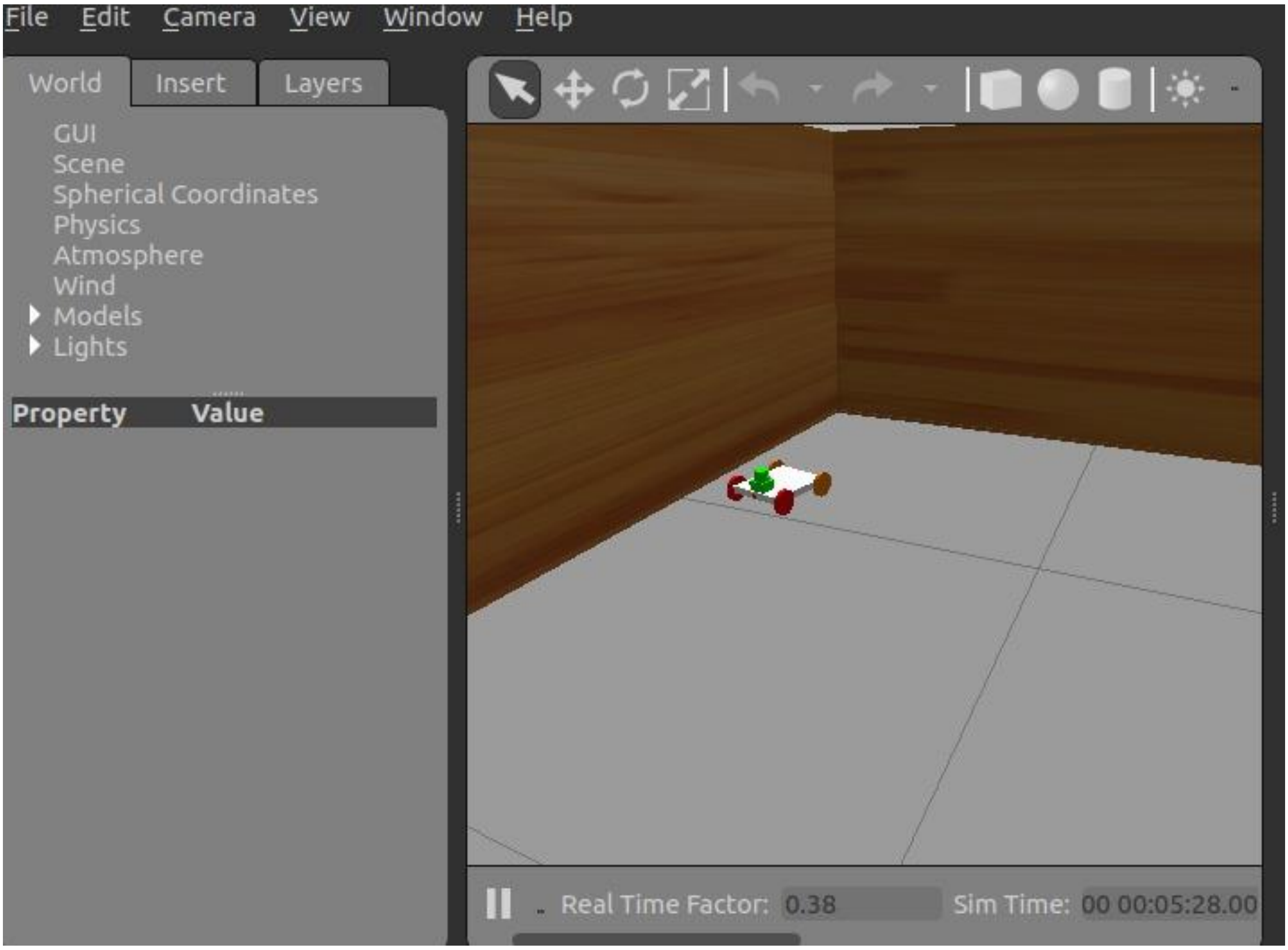
5. Flowchart

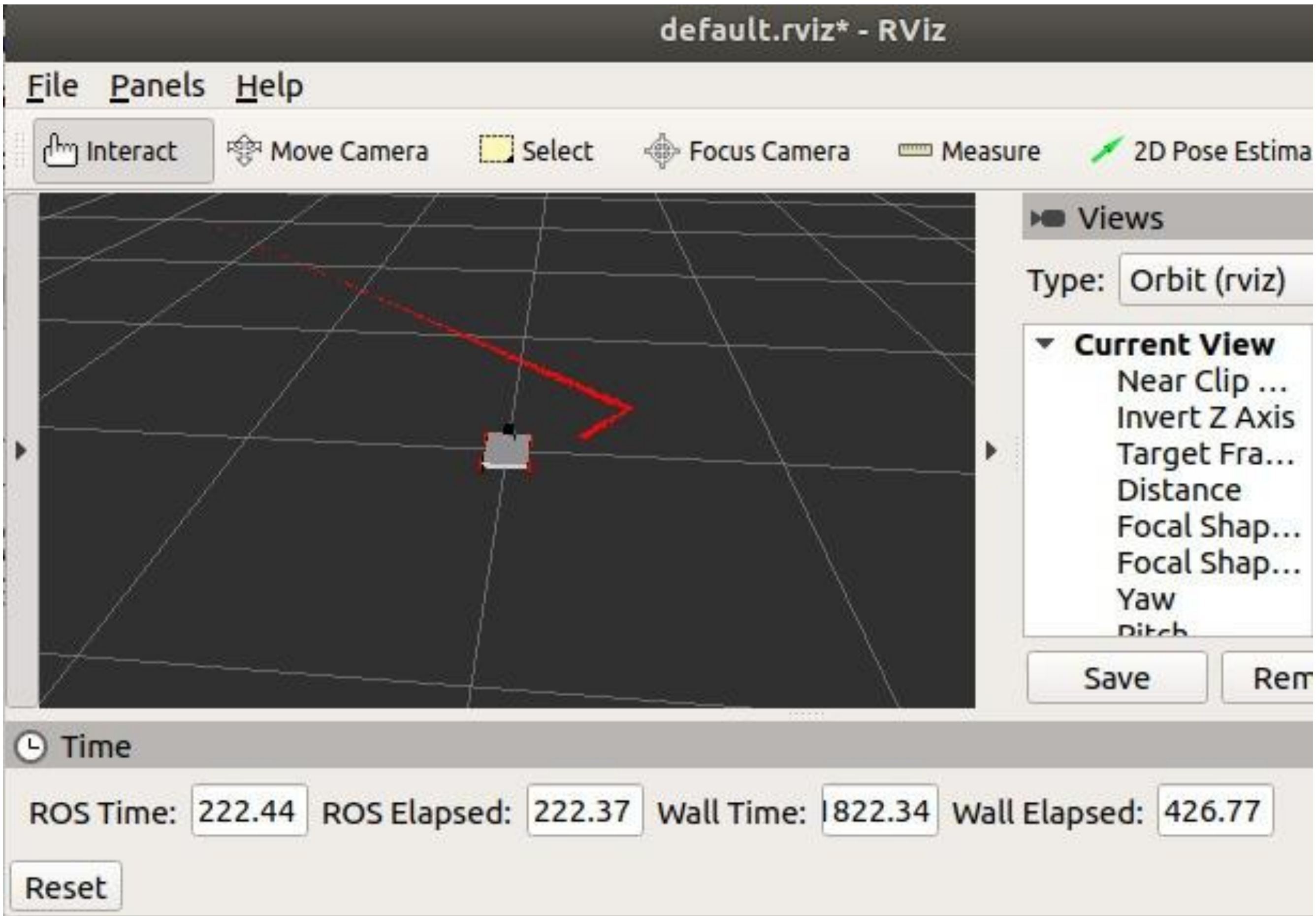


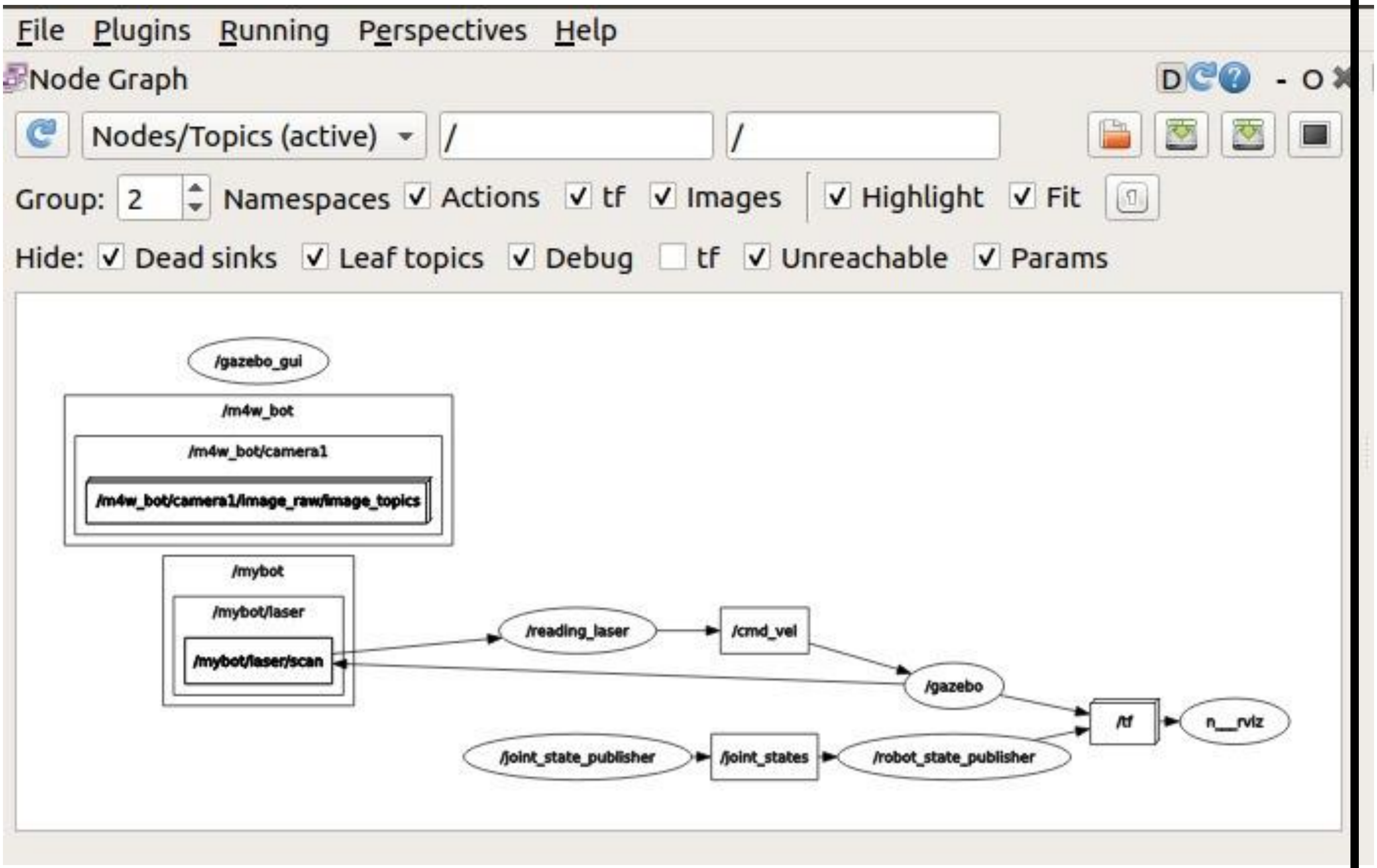
6. Result

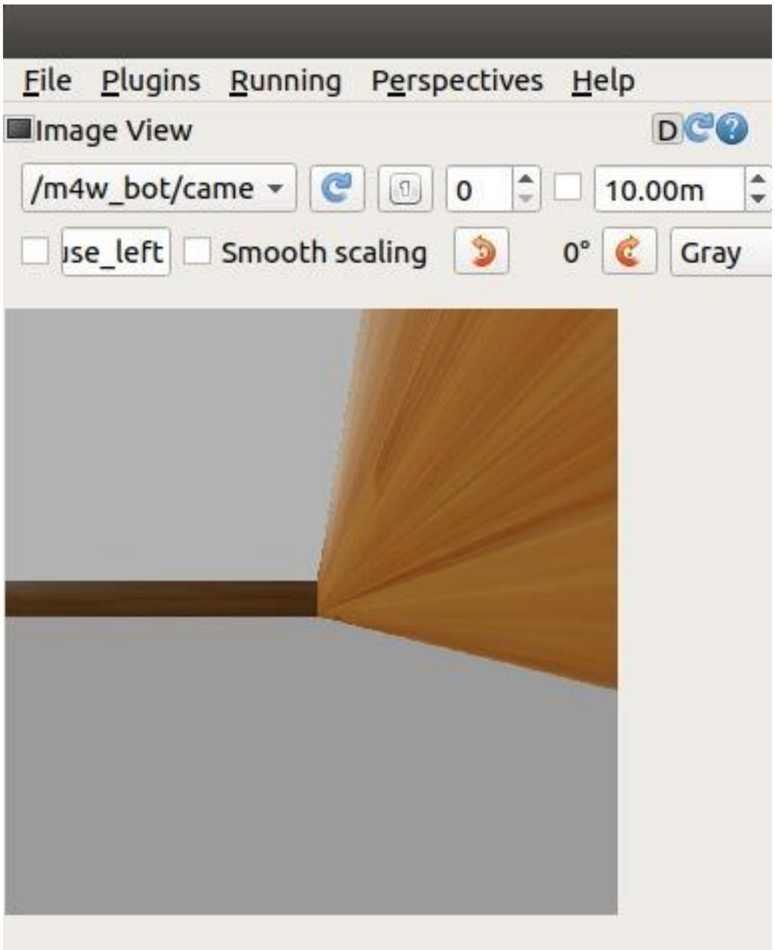
The wall following robot is successfully simulated using ROS and gazebo simulator. The surroundings of the robot is visualized using rviz and rqt.

The following are images of the moving robot and the camera images captured using rqt and rviz.


File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
Property Value
Real Time Factor: 0.38
Sim Time: 00 00:05:28.00


default.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estima
Views
Type: Orbit (rviz)
Current View
Near Clip ...
Invert Z Axis
Target Fra...
Distance
Focal Shap...
Focal Shap...
Yaw
Save
Time
ROS Time: 222.44
ROS Elapsed: 222.37
Wall Time: 1822.34
Wall Elapsed: 426.77
Reset


File Plugins Running Perspectives Help
Node Graph
Nodes/Topics (active)
Group: 2 Namespaces Actions tf Images Highlight Fit
Hide: Dead sinks Leaf topics Debug tf Unreachable Params
/gazebo_gui
/m4w_bot
/m4w_bot/camera1
/m4w_bot/camera1/image_raw/image_topics
/mybot
/mybot/laser
/mybot/laser/scan
/reading_laser
/cmd_vel
/gazebo
/tf
n__rviz
/joint_state_publisher
/joint_states
/robot_state_publisher


File Plugins Running Perspectives Help
Image View
/m4w_bot/came
0
10.00m
ıse_left
Smooth scaling
0°
Gray

7. Advantages and Disadvantages

7.1. Advantages:

- It can be used a surveillance system
- It doesn’t require man power
- Can be used for critical applications such as disasters, floods, fire attack etc.
- Easy to operate and perform

7.2. Disadvantages:

- Requires good amount of technical knowledge in order to build such kind of sophisticated robot.
- Slow paced robot. Not good for immediate responses.

8. Applications

- Can be used in military applications where the robot needs to spy along border areas.
- Can be used in abandoned mines to map the mine structure.
- Can be used as a cleaning robot which cleans the surroundings along walls.
- Can be used for emergency responses during disaster or accidents and get information about the accident site.
- Can be used as a surveillance system

9. Conclusion:

The obstacle/ wall following robot is successfully designed using ROS and Gazebo simulator. The robot uses laser to detect obstacles and walls. The algorithm is written in such as way that it can follow along the obstacles and follow along the wall with ease. The robot has a camera mounted on it for more visualisation. This robot can be used for many applications such as military purposes, abandoned mines, for emergency responses during disaster or accidents, can be used as cleaning robot.

10. Future Scope

The current version of the robot can avoid obstacles and follow the wall using information obtained from the laser and can visualize using a camera mounted on the robot. In further improvisation of the robot features, we can add path planning and mapping using SLAM which can get a map of the surrounding based on the obstacles. This will help in getting to know about the surroundings of an unknown location.

11. Bibliography

[1] R. Chinmayi et al., "Obstacle Detection and Avoidance Robot," 2018 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 2018, pp. 1-6, doi: 10.1109/ICCIC.2018.8782344.

[2] https://globaljournals.org/GJRE_Volume17/2-Obstacle-Avoiding-Robot.pdf

[3] https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS

[4] https://www.engineersgarage.com/efficient-wall-following-robot-with-ultrasonic-sensor-that-works-in-both-indoor-and-outdoor environments

[5] http://faculty.salina.k-state.edu/tim/robot_prog/MobileBot/Algorithms/WallFollow.html

[6] A. Imhof, M. Oetiker and B. Jensen, "Wall following for autonomous robot navigation," 2012 2nd International Conference on Applied Robotics for the Power Industry (CARPI), 2012, pp. 1-4, doi: 10.1109/CARPI.2012.6473370.

[7] M. Hashimoto, Kazuhiko Takahashi and Yosuke Matsui, "Moving-object tracking with multi-laser range sensors for mobile robot navigation," 2007 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2007, pp. 399-404, doi: 10.1109/ROBIO.2007.4522195.

12. Appendix

12.1. Source code

12.1.1. Xacro file of the robot

https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS/blob/master/src/mybot_description/urdf/m4w_robot.xacro

12.1.2. Gazebo file of the robot

https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS/blob/master/src/mybot_description/urdf/m4w_robot.gazebo

12.1.3. URDF materials xacro file

https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS/blob/master/src/mybot_description/urdf/materials.xacro

12.1.4. hokuyo file

https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS/blob/master/src/mybot_description/meshes/hokuyo.dae

12.1.5. rviz launch file

https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS/blob/master/src/mybot_description/launch/rviz.launch

12.1.6. spawn launch file

https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS/blob/master/src/mybot_description/launch/spawn.launch

12.1.7. world file for walls

https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS/blob/master/src/mybot_gazebo/worlds/wall.world

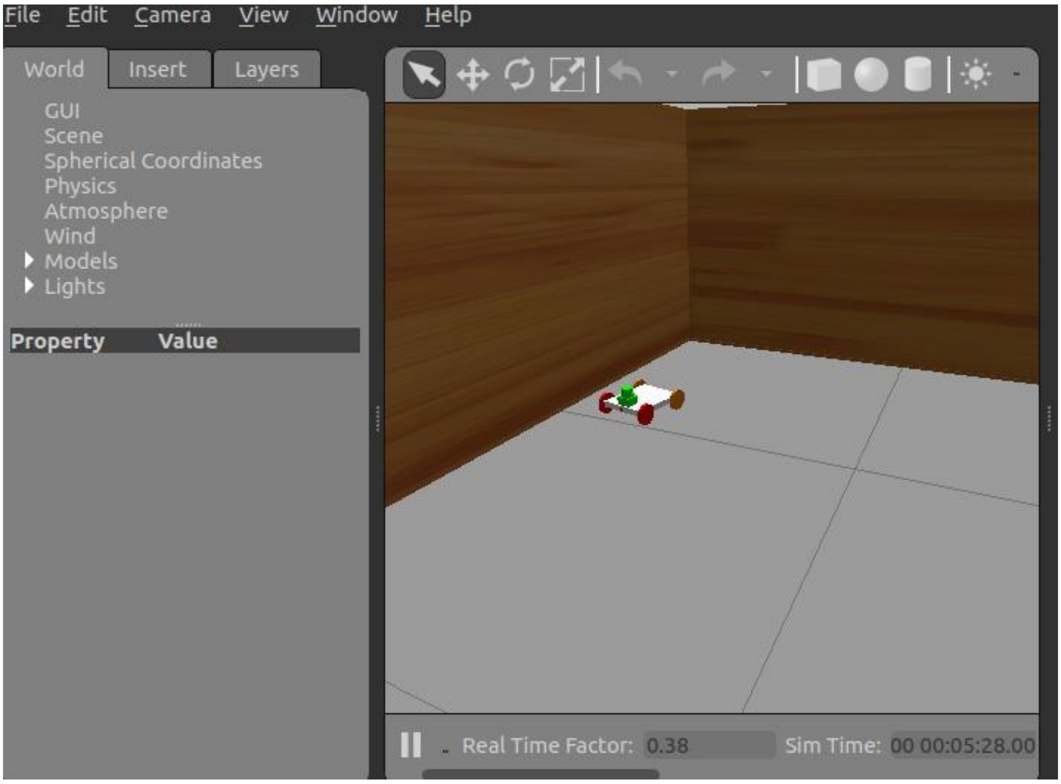
12.1.8. launch file of world

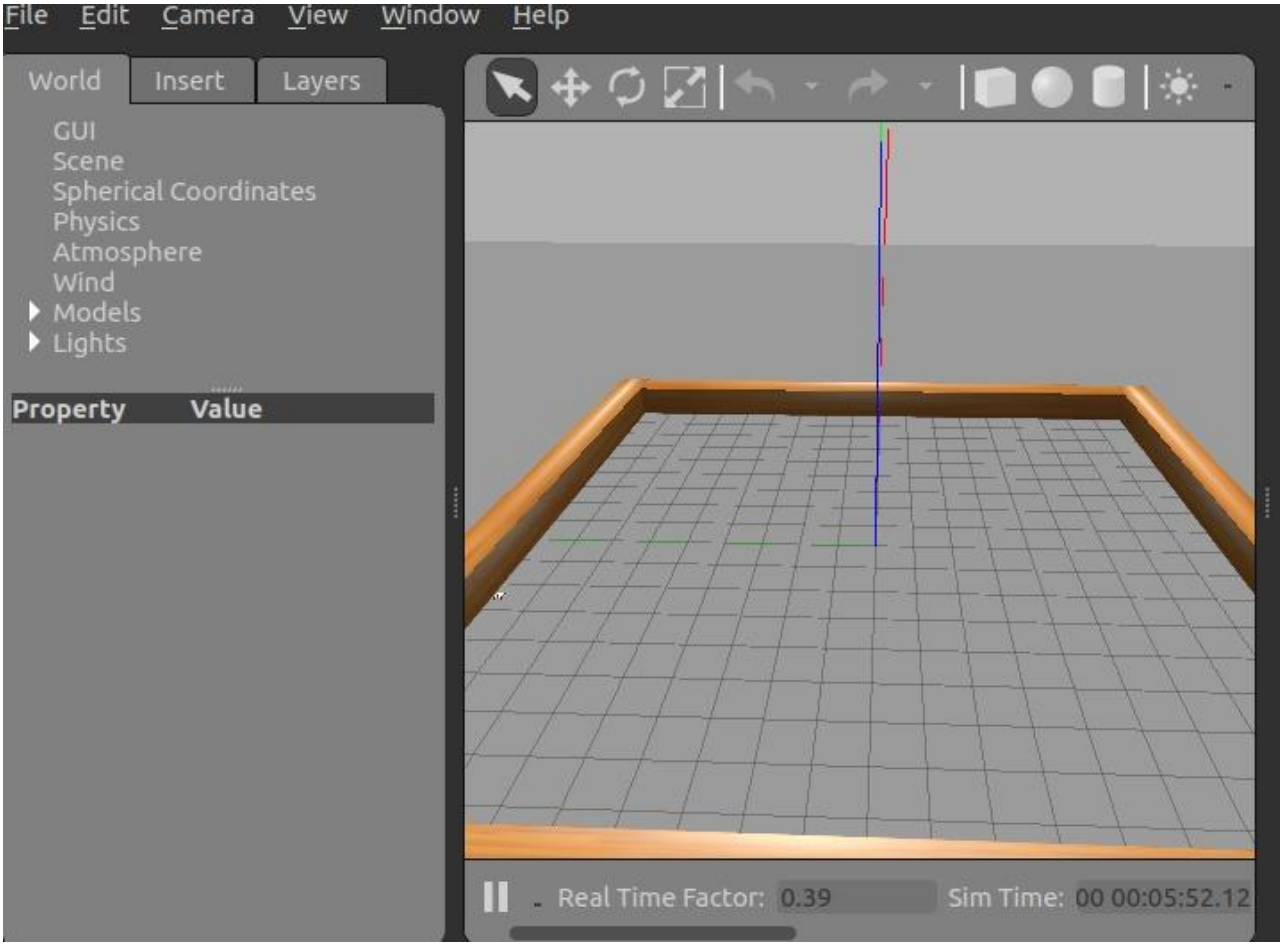
https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS/blob/master/src/mybot_gazebo/launch/mybot_world.launch

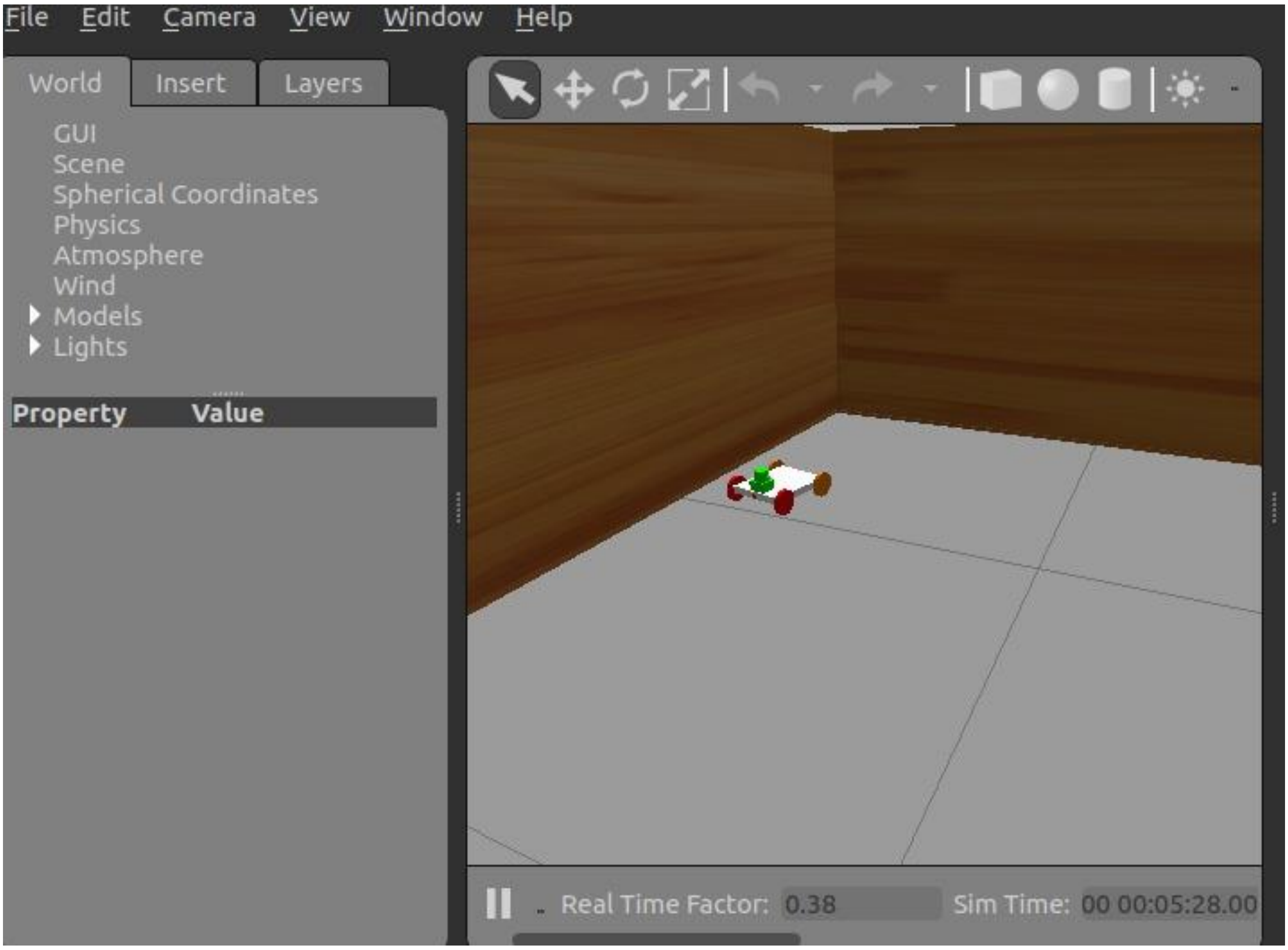
12.1.9. Python file for wall following algorithm

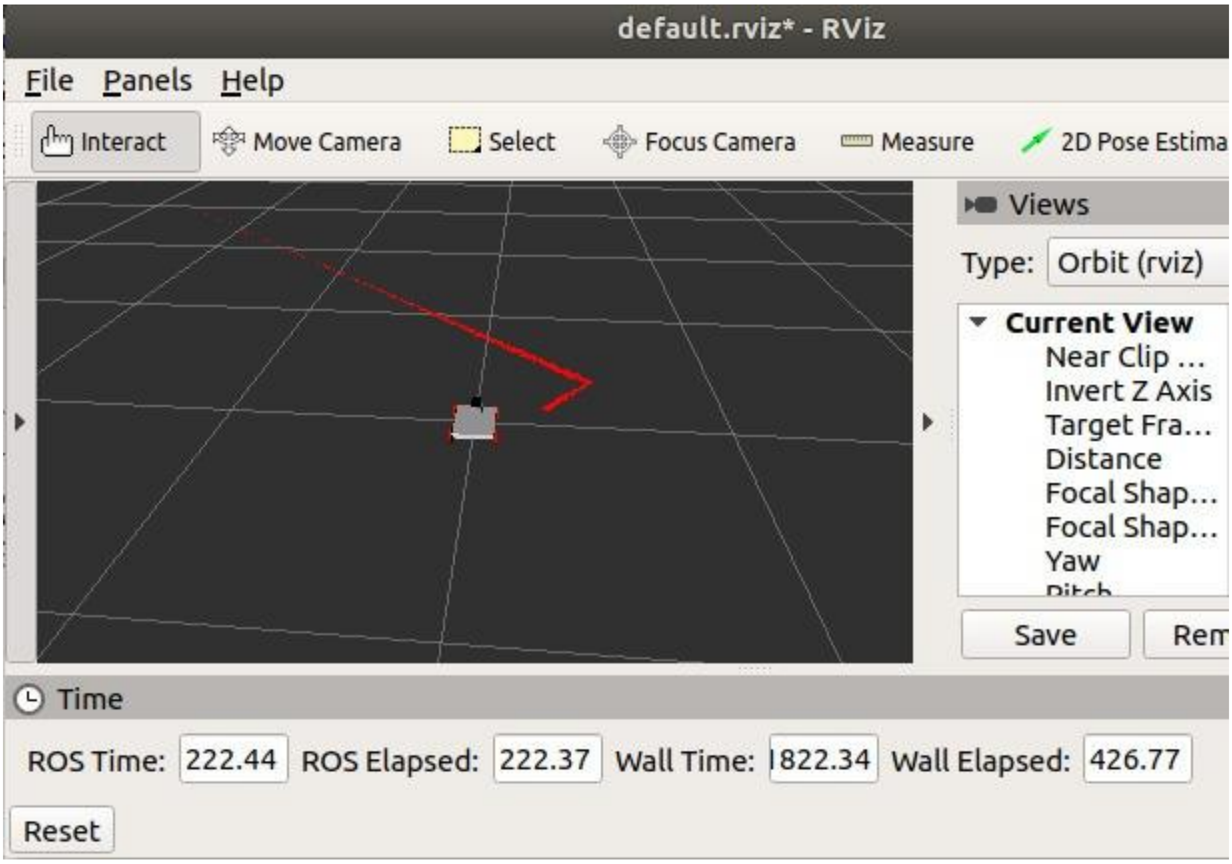
https://github.com/Durgaprasad-SB/Wall_Following_Robot_using_ROS/blob/master/src/mybot_motion/scripts/wallfollow.py

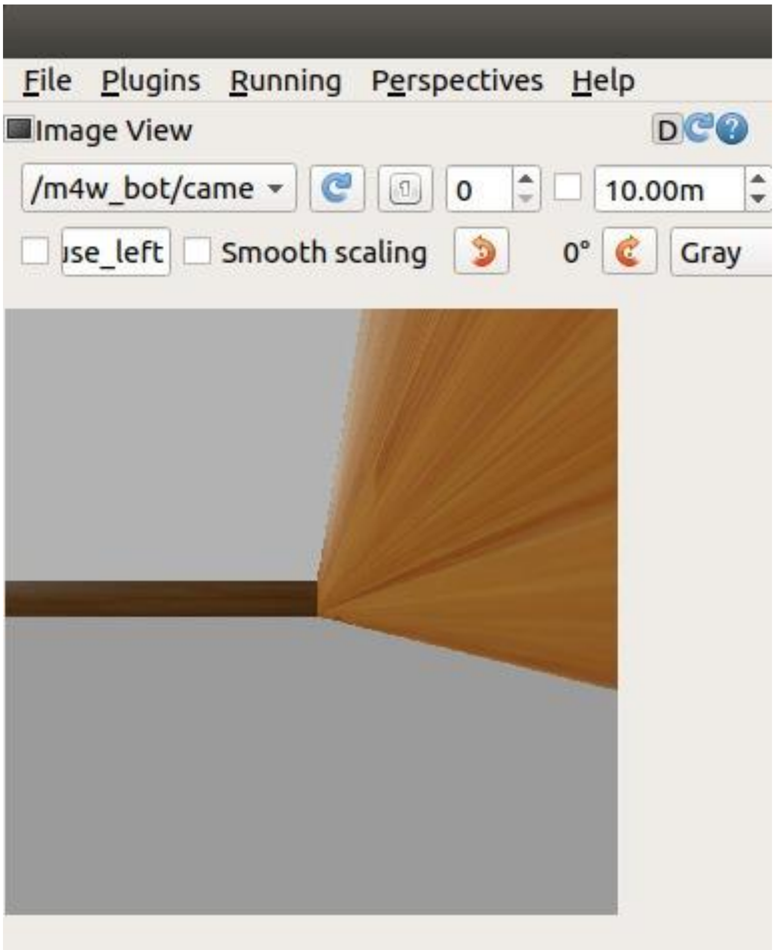
12.2. Output screenshot




File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
Property Value
Real Time Factor: 0.39
Sim Time: 00 00:05:52.12


File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
Lights
Property Value
Real Time Factor: 0.38
Sim Time: 00 00:05:28.00


default.rviz* - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estima
Views
Type: Orbit (rviz)
Current View
Near Clip ...
Invert Z Axis
Target Fra...
Distance
Focal Shap...
Focal Shap...
Yaw
Save
Time
ROS Time: 222.44 ROS Elapsed: 222.37 Wall Time: 1822.34 Wall Elapsed: 426.77
Reset


File Plugins Running Perspectives Help
Image View
/m4w_bot/came
0
10.00m
use_left Smooth scaling
0°
Gray