ABA AIOT PVT LTD

Pratik Rohane

Import Model in Gazebo and Simulate

OBJECTIVE

1. Import a Robot Model URDF package
2. Simulate Robot Model in Gazebo

REQUIREMENT

Software Required

1. Fusion360
2. VS Code

PlugIns Required

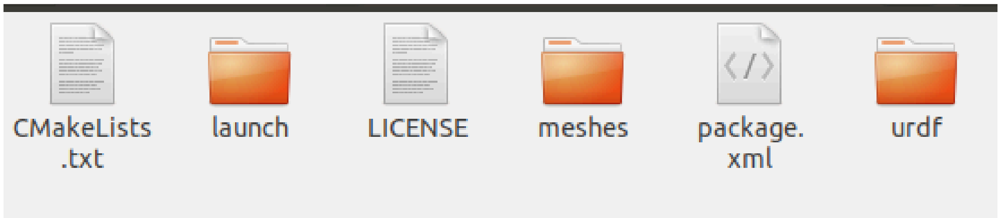
1. Fusion2URDF (Fusion360 Plugin)
2. Planer Move (Gazebo Plugin)

PLUGINS DESCRIPTION

Fusion2URDF Plugin

This is a fusion 360 script to export the URDF package of the robot model from the fusion 360. This package contains

- .urdf file of the robot model
- .launch and .yaml files to simulate robot on Gazebo
- .stl files of the robot model



Planar Move Plugin

This plugin allows the robot model to move along the horizontal plane using a geometry_msgs/Twist message of ROS. The plugin works by imparting a linear velocity in x and y and angular velocity in z to the robot.

The thing to be noted here is that the robot must have sufficient inertia to undesirable motions which occur as a reaction to applied velocity.

```
<gazebo>
    <plugin name="object_controller"
filename="libgazebo_ros_planar_move.so">
      <commandTopic>cmd_vel</commandTopic>
      <odometryTopic>odom</odometryTopic>
      <odometryFrame>odom</odometryFrame>
      <odometryRate>20.0</odometryRate>
      <robotBaseFrame>base_footprint</robotBaseFrame>
    </plugin>
  </gazebo>
```

Import a Robot Model URDF package

OVERVIEW

First, create a robot model into the Fusion360. Once the model is ready we can create a URDF package from this with the Fusion2URDF script plugin.

Note that this script may change the robot model, so before exporting the URDF make a copy as a backup.

PROCEDURE

Installation

Download a GitHub Repository

Step1: Download a .zip from the GitHub Repository and Extract that into a folder.

Fusion2urdf

Step2: Open the Windows Powershell and navigate into the same folder where we extract the zip file

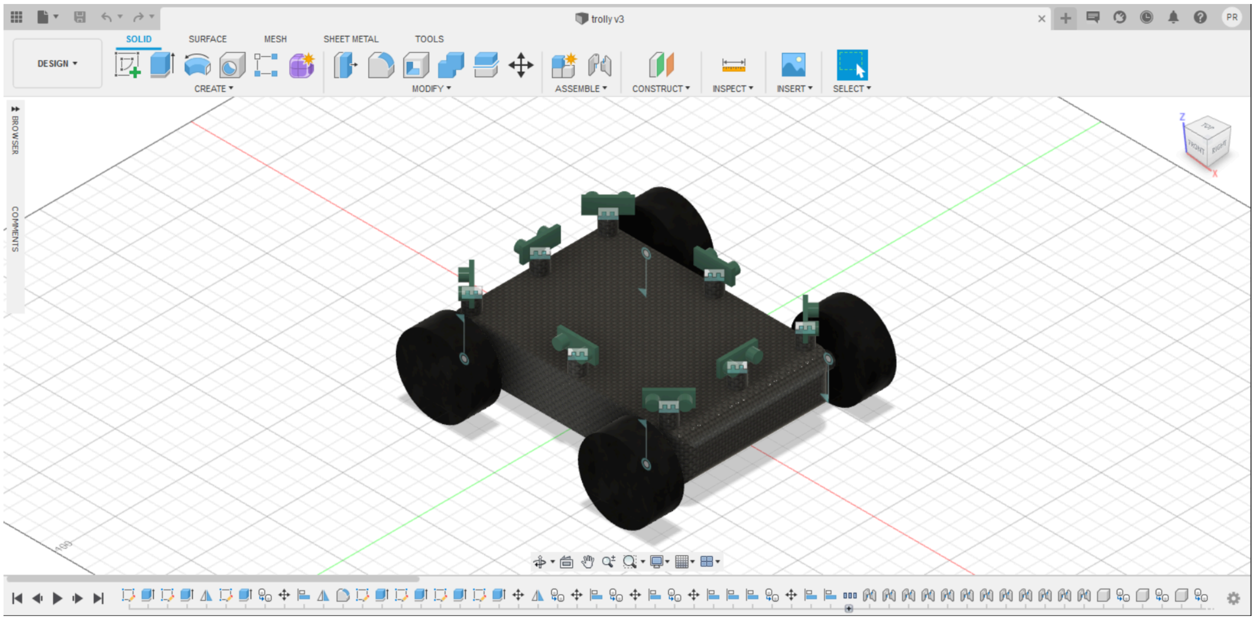
Step3: and Copy this package into the fusion360 scripts folder using the below command

```
Copy-Item ".\URDF_Exporter\" -Destination "${env:APPDATA}\Autodesk\Autodesk Fusion 360\API\Scripts\" -Recurse
```

How to export URDF

Note: You must have ‘base_link’ component in robot model

Step1: Open the Robot Model Project in the Fusion360



Step2: Click on TOOLS >> ADD-INS

Step3:

Step4: Run the script, it will take some time, and ask for the folder where to create a package.
Step5: Once done click ok and can check the folder

Problem

After running the fusion2urdf script, if it threw any error that was because the robot model was not created properly. All components must have joints to each other.

Import a Robot Model URDF package

OVERVIEW

Secondly, once the robot model URDF package copies that into the src folder of the catkin workspace. Then it's ready to simulate in Gazebo just by adding some plugin and changing the parameters

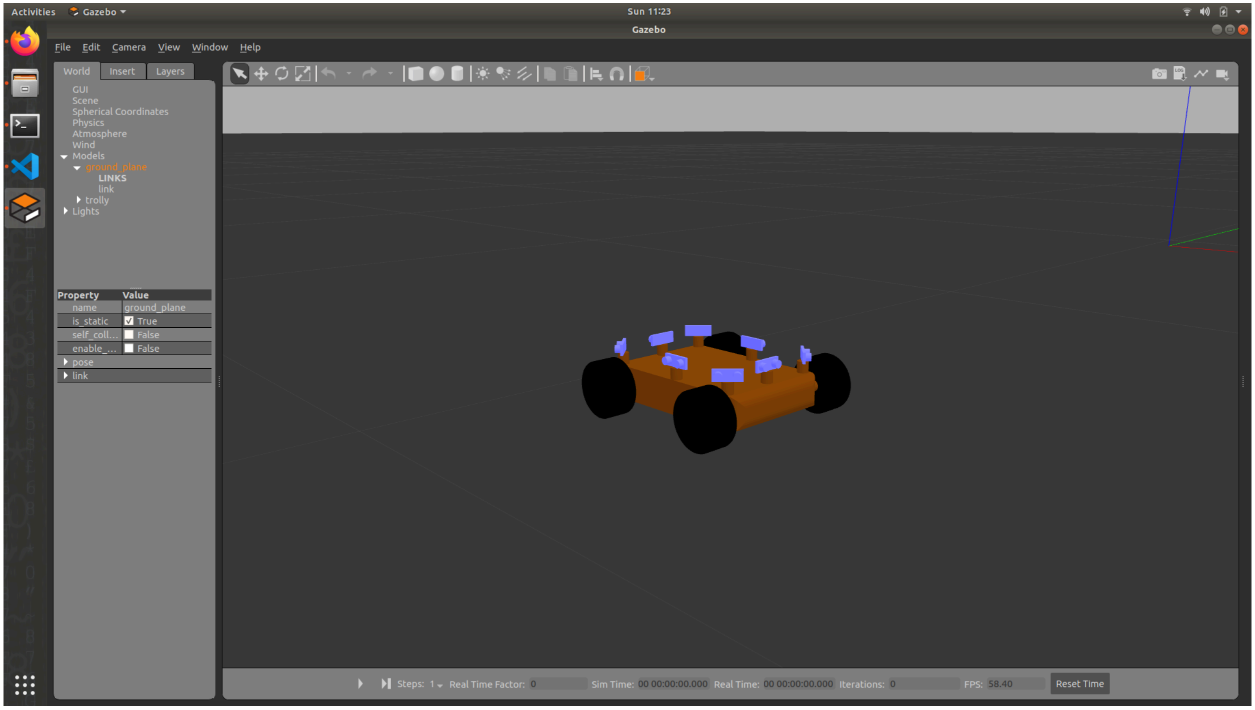
PROCEDURE

Run the following commands

```
cd ~/catkin_ws/
catkin_make
source devel/setup.bash
```

Open a model in Gazebo

```
roslaunch trolly_description gazebo.launch
```



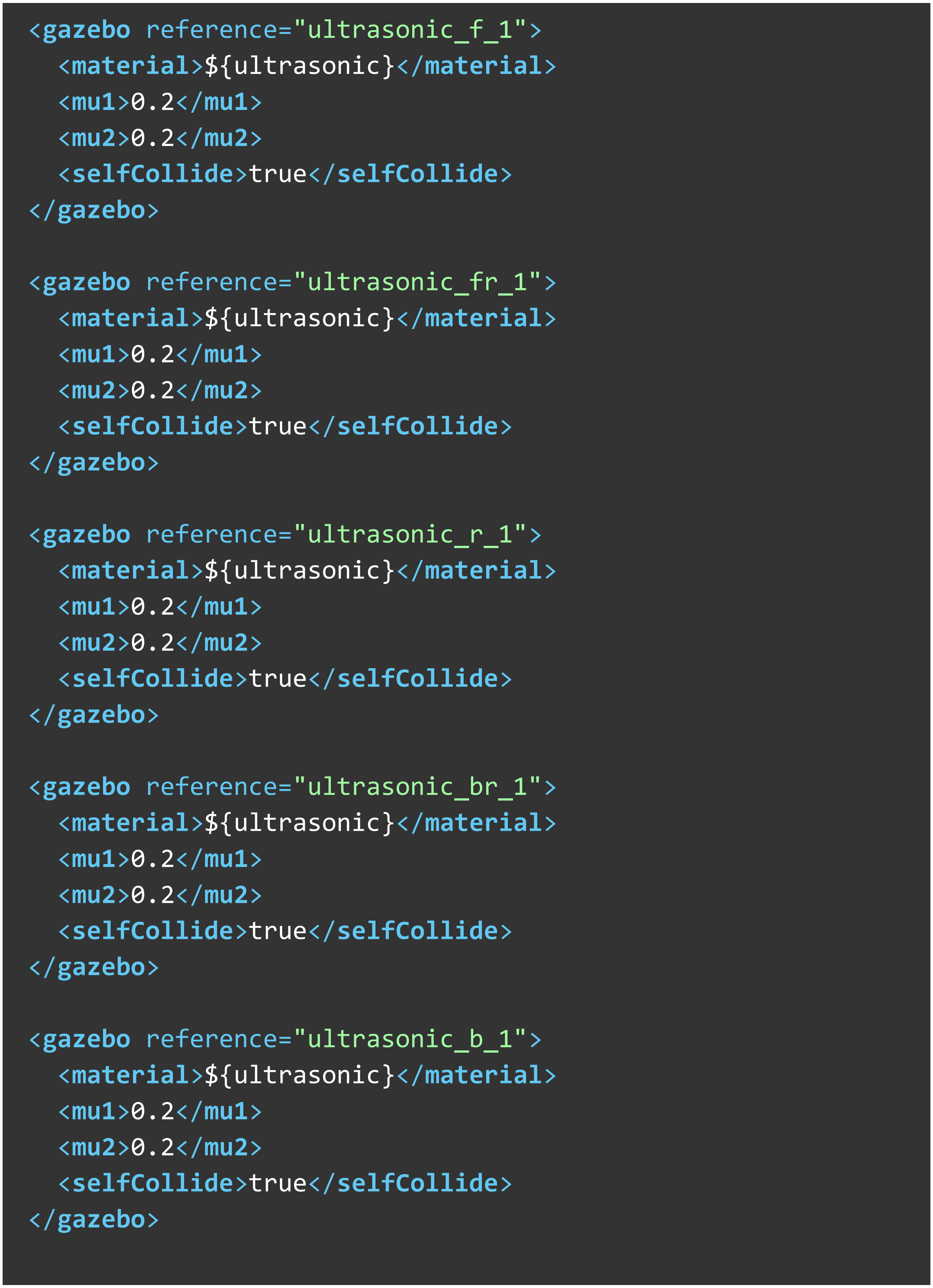
Property	Value
name	ground_plane
is_static	☑ True
self_coll...	□ False
enable_...	□ False

Steps: 1 Real Time Factor: 0 Sim Time: 00 00:00:00.000 Real Time: 00 00:00:00.000 Iterations: 0 FPS: 58.40

To simulate this model we need to add the controller plugin in the .gazebo file.
Step1: Open trolly.gazebo file from urdf folder and add the above plugin

```
<?xml version="1.0" ?>
<robot name="trolly"
xmlns:xacro="http://www.ros.org/wiki/xacro" >

 <xacro:property name="body_color" value="Gazebo/Orange" />
 <xacro:property name="wheels" value="Gazebo/Black" />
 <xacro:property name="ultrasonic" value="Gazebo/BlueGlow" />

 <gazebo reference="base_link">
   <material>${body_color}</material>
   <mu1>0.2</mu1>
   <mu2>0.2</mu2>
   <selfCollide>true</selfCollide>
   <gravity>true</gravity>
 </gazebo>
```

```
<gazebo reference="wheel_fr_1">
  <material>${wheels}</material>
  <mu1>1.0</mu1>
  <mu2>1.0</mu2>
  <selfCollide>true</selfCollide>
</gazebo>

<gazebo reference="wheel_fl_1">
  <material>${wheels}</material>
  <mu1>1.0</mu1>
  <mu2>1.0</mu2>
  <selfCollide>true</selfCollide>
</gazebo>

<gazebo reference="wheel_br_1">
  <material>${wheels}</material>
  <mu1>1.0</mu1>
  <mu2>1.0</mu2>
  <selfCollide>true</selfCollide>
</gazebo>

<gazebo reference="wheel_bl_1">
  <material>${wheels}</material>
  <mu1>1.0</mu1>
  <mu2>1.0</mu2>
  <selfCollide>true</selfCollide>
</gazebo>

<gazebo reference="ultrasonic_fl_1">
  <material>${ultrasonic}</material>
  <mu1>0.2</mu1>
  <mu2>0.2</mu2>
  <selfCollide>true</selfCollide>
</gazebo>
```



```
<gazebo reference="ultrasonic_f_1">
  <material>${ultrasonic}</material>
  <mu1>0.2</mu1>
  <mu2>0.2</mu2>
  <selfCollide>true</selfCollide>
</gazebo>

<gazebo reference="ultrasonic_fr_1">
  <material>${ultrasonic}</material>
  <mu1>0.2</mu1>
  <mu2>0.2</mu2>
  <selfCollide>true</selfCollide>
</gazebo>

<gazebo reference="ultrasonic_r_1">
  <material>${ultrasonic}</material>
  <mu1>0.2</mu1>
  <mu2>0.2</mu2>
  <selfCollide>true</selfCollide>
</gazebo>

<gazebo reference="ultrasonic_br_1">
  <material>${ultrasonic}</material>
  <mu1>0.2</mu1>
  <mu2>0.2</mu2>
  <selfCollide>true</selfCollide>
</gazebo>

<gazebo reference="ultrasonic_b_1">
  <material>${ultrasonic}</material>
  <mu1>0.2</mu1>
  <mu2>0.2</mu2>
  <selfCollide>true</selfCollide>
</gazebo>
```

```
 <gazebo reference="ultrasonic_bl_1">
   <material>${ultrasonic}</material>
   <mu1>0.2</mu1>
   <mu2>0.2</mu2>
   <selfCollide>true</selfCollide>
 </gazebo>

 <gazebo reference="ultrasonic_l_1">
   <material>${ultrasonic}</material>
   <mu1>0.2</mu1>
   <mu2>0.2</mu2>
   <selfCollide>true</selfCollide>
 </gazebo>
<!-- Controller  -->
 <gazebo>
     <plugin name="object_controller"
filename="libgazebo_ros_planar_move.so">
       <commandTopic>cmd_vel</commandTopic>
       <odometryTopic>odom</odometryTopic>
       <odometryFrame>odom</odometryFrame>
       <odometryRate>10.0</odometryRate>
       <robotBaseFrame>base_link</robotBaseFrame>
     </plugin>
 </gazebo>

</robot>
```

In this plugin code we need to edit some parameters
commandTopic - name of the topic to publish velocity messages(cmd_vel)
robotBaseFrame- base_link

Now run the gazebo.launch and teleop twist keyboard to control the robot

Gazebo

```
roslaunch trolly_description gazebo.launch
```

Teleop Twist Keyboard

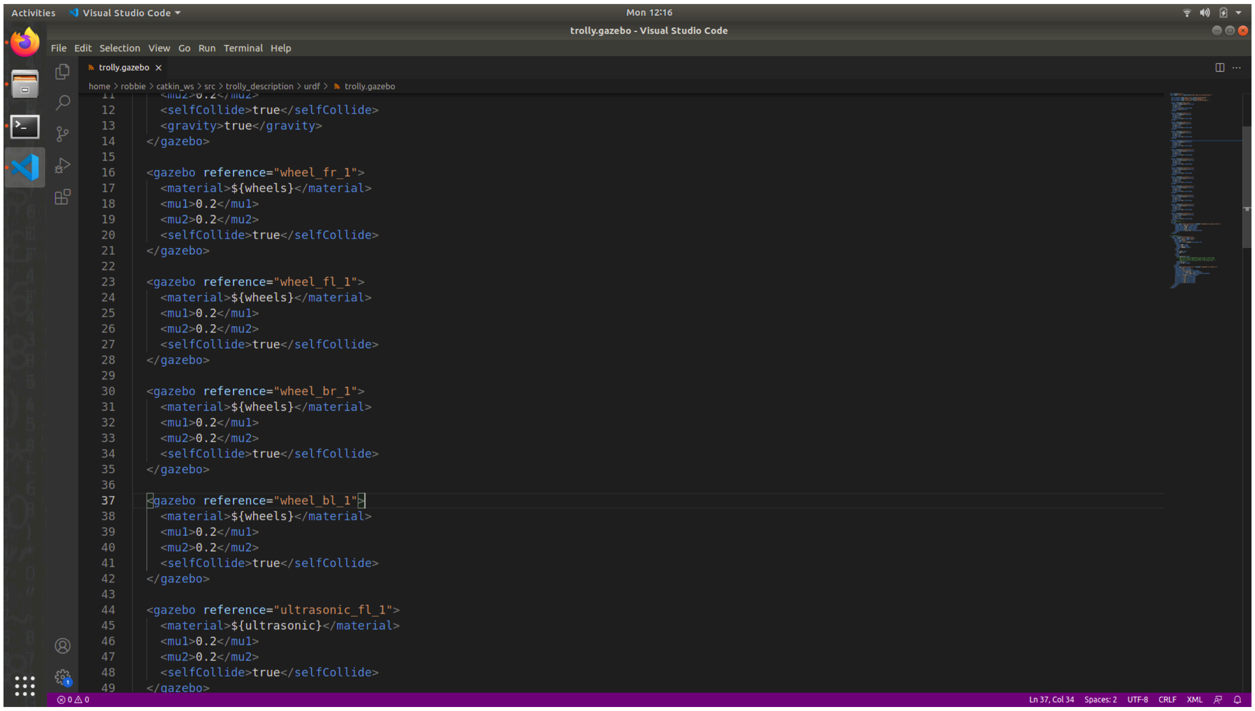
```
rosrun teleop_twist_keyboard teleop_twist_keyboard.py
```

Problem

If we try to control the robot, the robot moves very fast, and will go out off screen in very less time.

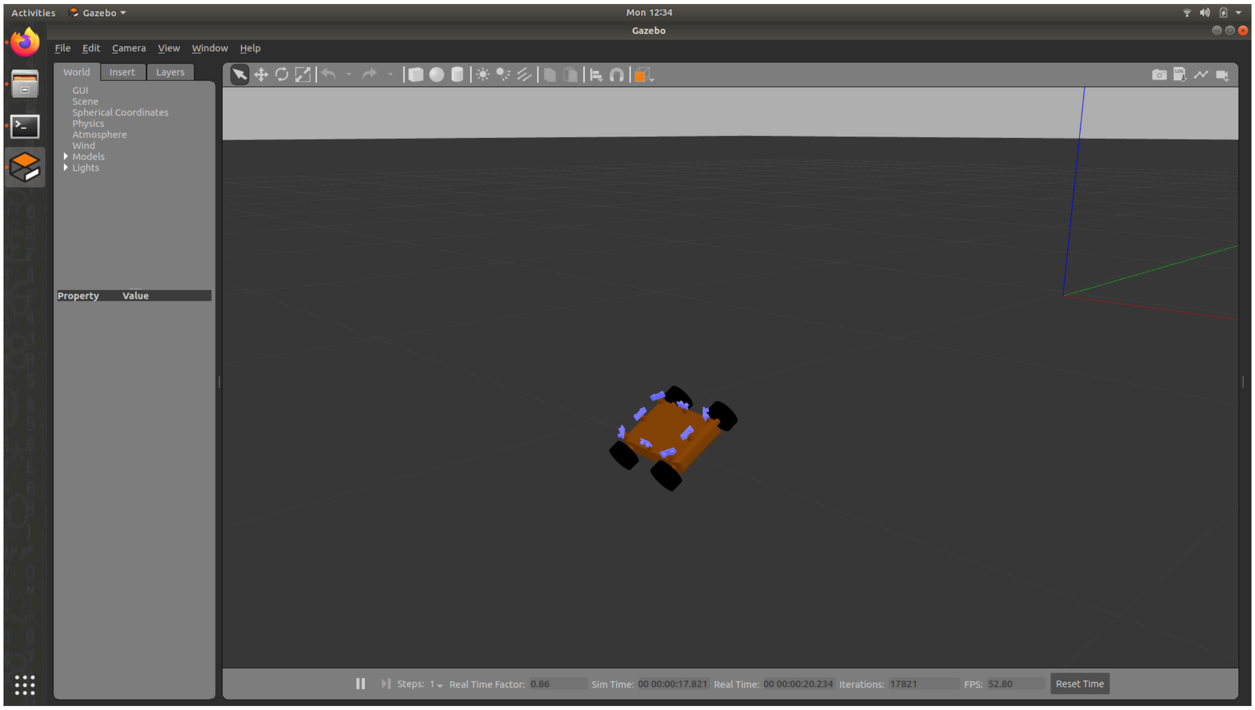
Solution

This happens because wheels have very low friction that is 0.2 for mu1 and mu2, by increasing that we can get the better result.


Activities
Visual Studio Code
Mon 12:16
trolly.gazebo - Visual Studio Code
File Edit Selection View Go Run Terminal Help
trolly.gazebo
home > robbie > catkin_ws > src > trolly_description > urdf > trolly.gazebo
Ln 37, Col 34
Spaces: 2
UTF-8
CRLF
XML

```
    <mu2>0.2</mu2>
    <selfCollide>true</selfCollide>
    <gravity>true</gravity>
  </gazebo>

  <gazebo reference="wheel_fr_1">
    <material>${wheels}</material>
    <mu1>0.2</mu1>
    <mu2>0.2</mu2>
    <selfCollide>true</selfCollide>
  </gazebo>

  <gazebo reference="wheel_fl_1">
    <material>${wheels}</material>
    <mu1>0.2</mu1>
    <mu2>0.2</mu2>
    <selfCollide>true</selfCollide>
  </gazebo>

  <gazebo reference="wheel_br_1">
    <material>${wheels}</material>
    <mu1>0.2</mu1>
    <mu2>0.2</mu2>
    <selfCollide>true</selfCollide>
  </gazebo>

  <gazebo reference="wheel_bl_1">
    <material>${wheels}</material>
    <mu1>0.2</mu1>
    <mu2>0.2</mu2>
    <selfCollide>true</selfCollide>
  </gazebo>

  <gazebo reference="ultrasonic_fl_1">
    <material>${ultrasonic}</material>
    <mu1>0.2</mu1>
    <mu2>0.2</mu2>
    <selfCollide>true</selfCollide>
  </gazebo>
```

If increated values of mu1 nad mu2 to 10, robot behave in different manner, like bellow



Steps: 1 Real Time Factor: 0.86 Sim Time: 00 00:00:17.821 Real Time: 00 00:00:20.234 Iterations: 17821 FPS: 52.80

This happens because of high friction of wheels, so you need to decrease that, check the values which work fine, in this case 1 for mu1 and mu1 work well.

CONCLUSION

As we have mecanum wheel drive we have to use the planer move plugin only to get left-right and diagonal movement of the robot.
The mu1 and mu2 values of the wheel will change from model to model as it depends on the material used and the mass of the robot.

RESULT