SDF and Gazebo

R Prasanth Kumar, IIT Hyderabad

26/02/2025

1 Introduction

Gazebo Classic (often referred to simply as "Gazebo") is a widely used open-source robotics simulator that provides a 3D environment for testing and development, offering physics-based simulation of robot models, sensors, and complex environments. It uses the Simulation Description Format (SDF) to describe both the world and robots, although it can also accept robot models in URDF via conversion. Gazebo Classic integrates seamlessly with ROS2 (such as in the Humble release), allowing users to simulate robots, run navigation or manipulation algorithms, and develop complex multi-robot systems without the risk of real-world hardware damage. This "classic" version (up to Gazebo 11) has a mature ecosystem and remains the standard in many ROS tutorials, even as newer versions of Gazebo (Ignition/Gazebo Sim) continue to evolve. To install Gazebo, execute from the command line:

```
sudo apt install ros-humble-gazebo-ros
```

Related to this Gazebo simulator, two executables are installed in `/usr/bin`. These are `gz` and `gazebo`, which can be executed from the command line. `gazebo` is used to load the simulator with a particular world, whereas `gz` is used from command line to alter the properties of the world that is running on gazebo simulator.

2 Simulation Description Format (SDF)

A typical SDF file looks like:

```
<?xml version="1.0" ?>
<sdf version="1.6">
    <model name="myrobot">

    </model>
</sdf>
```

The robot tag in URDF is now called as model in SDF. Let us consider a simple sphere for this example.

```
<?xml version="1.0"?>
<sdf version="1.6">
<world name="worldname">
  <model name="simple_sphere" >
    <static>true</static>
    <pose>0 0 3 0 0 0</pose>
    <link name="spherelink" >
      <visual name="visual" >
```

```
        <geometry>
          <sphere>
            <radius>0.5</radius>
          </sphere>
        </geometry>
      </visual>
    </link>
  </model>
</world>
</sdf>
```

Save it as `test1.world` and open from command line using:

```
gazebo test1.world
```

With `static` as true, the model remains static. Make this false, and include inertia tag as follows:

```
<?xml version="1.0"?>
<sdf version="1.6">
    <world name="worldname">
    <gravity>0 0 0</gravity>
        <model name="simple_sphere" >
            <static>false</static>
            <pose>0 0 3 0 0 0</pose>
            <link name="spherelink" >
            <visual name="visual" >
                <geometry>
                    <sphere>
                        <radius>0.5</radius>
                    </sphere>
                </geometry>
            </visual>
            <inertial>
                <mass>1.0</mass>
                <inertia>  <ixx>0.1</ixx> <iyy>0.1</iyy> <izz>0.1</izz>
                    <ixy>0.0</ixy> <ixz>0.0</ixz> <iyz>0.0</iyz>
                </inertia>
            </inertial>
            </link>
        </model>
    </world>
</sdf>
```

When the simulation is running, the properties can be changed with `gz` from command line. To include gravity for example:

```
gz physics test1.world -g 0,0,-9.81
```

The sphere passes through the ground because neither the sphere has collision property defined nor a ground is define in the world. Include the following below gravity tag in the file:

```
    <include>
      <uri>model://ground_plane</uri>
    </include>
```

and define collision in visual tag.

```
<collision name="visual" >
    <geometry>
        <sphere>
            <radius>0.5</radius>
        </sphere>
    </geometry>
</collision>
```

3 Exercises

1. Create test2.world file containing a support structure to which a pendulum is attached. With initial condition displaced from equilibrium position, start the simulation in Gazebo.