XML Launch Files

R Prasanth Kumar, IIT Hyderabad

27/02/2025

1 Introduction

It is possible to launch multiple nodes using xml files rather than Python files. We can do this for both rviz2 as well as gazebo. In order to launch xml launch file, we will create the xml file in `launch` folder under the folder `my_robot_description` (which contains `urdf` folder with our robot urdf model file). We will start with the following sample URDF file under `urdf` folder:

```
<?xml version="1.0"?>
<robot name="twddr">
    <link name="base_link" />
    <link name="chassis">
        <inertial>
            <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
            <mass value="0.0"/>
            <inertia ixx="0.0" ixy="0.0" ixz="0.0" iyy="0.0" iyz="0.0" izz="0.0"/>
        </inertial>
        <visual name="">
            <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
            <geometry>
                <box size="0.5 0.3 0.1"/>
            </geometry>
            <material name="">
                <color rgba="0.0 1.0 0.0 0.4"/>
                <texture filename=""/>
            </material>
        </visual>
        <collision>
            <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
            <geometry>
                <box size="0.5 0.3 0.1"/>
            </geometry>
        </collision>
    </link>

    <joint name="joint_name" type="fixed">
        <origin xyz="0.0 0.0 0.3" rpy="0.0 0.0 0.0"/>
        <parent link="base_link"/>
        <child link="chassis"/>
    </joint>
</robot>
```

2 XML Launch File for rviz2

Save the following to twddr.launch.xml file in `my_robot_description/launch` folder:

```
<launch>
  <let name="urdf_path"
  value="$(find-pkg-share my_robot_description)/urdf/twddrobot.urdf" />
  <node pkg="robot_state_publisher" exec="robot_state_publisher">
        <param name="robot_description" value="$(command 'xacro $(var urdf_path)')" />
  </node>
  <node pkg="joint_state_publisher_gui" exec="joint_state_publisher_gui" />
  <node pkg="rviz2" exec="rviz2" output="screen" />
</launch>
```

In `CMakeLists.txt` of `my_robot_description` package, update the following:

```
install(
  DIRECTORY urdf launch
  DESTINATION share/${PROJECT_NAME}/
)
ament_package()
```

To launch this xml launch file, execute

```
ros2 launch my_robot_description twddr.launch.xml
```

Alternatively, you can also have this launch file in `my_robot_bringup` package (in a similar `launch` folder) with appropriate changes in the package's `CMakeLists.txt`.

3 XML Launch File for Gazebo

A typical xml launch file for Gazebo looks like:

```
<launch>
    <let name="urdf_path"
    value="$(find-pkg-share my_robot_description)/urdf/my_robot.urdf.xacro" />
    <let name="rviz_config_path"
    value="$(find-pkg-share my_robot_bringup)/rviz/urdf_config.rviz" />

    <node pkg="robot_state_publisher" exec="robot_state_publisher">
        <param name="robot_description" value="$(command 'xacro $(var urdf_path)')" />
    </node>

    <include file="$(find-pkg-share gazebo_ros)/launch/gazebo.launch.py" />

    <node pkg="gazebo_ros" exec="spawn_entity.py"
    args="-topic robot_description -entity my_robot" />

    <node pkg="rviz2" exec="rviz2" output="screen"
    args="-d $(var rviz_config_path)" />
</launch>
```

In `my_robot_bringup`, we will need the three folders `urdf`, `launch`, and `rviz`. Once you setup the RobotModel (and also TF) plug-ins in RViz, you can save the config file to the folder `rviz`. This way you don't have to add the plug-in everytime you want to visualize in RViz. In this example,

xacro is used. You can also do it without xacro. Note that `spawn_entity.py` converts URDF to SDF before spawing the robot inside Gazebo.
Note: Mass and inertia values have to filled up in URDF. Errors in mass and inertia values can make Gazebo behaviour unpredictable.

4 Exercises

1. Create an XML launch file for launching a robot (my_robot.urdf.xacro) in Gazebo. Make sure you include inertia and collision tags in the URDF file which are not required for rviz2.