



ASRL ROS 2 Tutorial - ROS 2 Basics

Created By Bastian Weiss

Tutorial Contents

- ROS 2 Installation
- How to create a ROS 2 Cmake Package
- How to create a URDF File
- How to create a Launch File
- Launching Rviz
- Checking for Joint Control

Table of Contents

- [Downloading Required Programs](#)
- [Create a ROS 2 Package](#)
- [Creating The URDF](#)
- [Creating The Launch File](#)
- [Rviz Configs](#)
- [Updating CMakeLists.txt](#)
- [Updating package.xml](#)
- [Launching our Package](#)

Downloading Required Programs

- If on ubuntu computer, Install ROS 2 Jazzy, Follow steps for Ubuntu Installation
 - <https://docs.ros.org/en/jazzy/Installation/Ubuntu-Install-Debs.html>
- If on Windows computer, follow all steps above but first install the Ubuntu 24.04 VM.
 - Go to Microsoft Store, Search Ubuntu, Install Ubuntu 24.04.1 LTS.
 - IMPORTANT: It has to be the 24.04.1 LTS Version, Regular Ubuntu will not work.

Downloading Required Programs

- Optional Programs:
 - Visual Studio Code: Highly recommended
 - <https://code.visualstudio.com/Download>

General Ubuntu Info

- **Make sure you update packages before starting**
 - sudo apt update
 - sudo apt upgrade
- **Change working directory through cd**
 - Example: cd ws_tutorial
- **Make a new directory through mkdir**
 - Example: mkdir model
- **Make a new file through touch**
 - Example: touch main.xacro

Create a ROS 2 Package

- **Source ROS 2**

- source /opt/ros/jazzy/setup.bash

- **Create the Workspace**

- mkdir ws_tutorial/src

- **Create the C++ Package**

- ros2 pkg create --build-type ament_cmake asrl_tutorial

Your structure should look like this

```
└── src
    └── asrl_tutorial
        ├── CMakeLists.txt
        ├── include
        │   └── asrl_tutorial
        ├── package.xml
        └── src
```

Creating The URDF

- Rename the `src` folder inside `asrl_tutorial` to `model`
- Install Xacro, this will help with the creation of our model
 - `sudo apt install ros-jazzy-xacro`
- Create the model files
 - Touch `main.xacro` `chassis.xacro` `connector.xacro` `rim.xacro` `roller.xacro`
- Copy upcoming code into each file

URDF Code Breakdown - Main

```
<?xml version='1.0'?>  
  
<robot name="rome_urdf" xmlns:xacro="http://www.ros.org/wiki/xacro">  
  
  <!-- Base -->  
  
  <xacro:include filename="$(find asrl_tutorial)/model/chassis.xacro"/>  
  
  <!-- World Link -->  
  
  <link name="base_link" />  
  
  <joint name="base_to_chassis" type="fixed">  
    <parent link="base_link"/>  
    <child link="chassis"/>  
    <origin xyz="0 0 0" rpy="0 0 0" />  
  </joint>  
  
</robot>
```

URDF Code Breakdown - Chassis

```
<?xml version='1.0'?>
<robot xmlns:xacro="http://www.ros.org/wiki/xacro">
<xacro:include filename="$(find asrl_tutorial)/model/connector.xacro"/>

<!-- Chassis -->
<link name="chassis">
<visual>
<origin xyz="0 0 0" rpy="0 0 0" />
<geometry>
<mesh filename="file://${find asrl_tutorial}/meshes/chassis.stl" />
</geometry>
<material name="">
<color rgba="1 1 0 1" />
</material>
</visual>
<inertial>
<origin xyz="0 0 0" rpy="0 0 0" />
<mass value="16.004"/>
<inertia ixx="0.3610" ixy="0.0" ixz="0.0" iyy="0.3610" iyz="0.0" izz="0.6749"/>
</inertial>
</inertial>
</link>

<gazebo reference="chassis">
<visual>
<material>
<ambient>0.9 0.9 0 1</ambient>
<diffuse>0.9 0.9 0 1</diffuse>
<specular>0.9 0.9 0 1</specular>
</material>
</visual>
</gazebo>

<!--Wheel Rims -->
<xacro:connector name="1" parent="chassis" xyz="0.228 0.228 0.01" rpy="1.57 0 2.3562"/> <!--Front Left-->
<xacro:connector name="2" parent="chassis" xyz="0.228 -0.228 0.01" rpy="1.57 0 0.7838"/> <!--Front Right-->
<xacro:connector name="3" parent="chassis" xyz="-0.228 -0.228 0.01" rpy="1.57 0 -0.7838"/> <!--Back Right-->
<xacro:connector name="4" parent="chassis" xyz="-0.228 0.228 0.01" rpy="1.57 0 -2.3562"/> <!--Back Left -->

</robot>
```

URDF Code Breakdown - Connector

```
<?xml version='1.0'?>
<robot xmlns:xacro="http://www.ros.org/wiki/xacro">
<xacro:include filename="$(find asrl_tutorial)/model/rim.xacro"/>
<xacro:macro name="connector" params="name parent xyz rpy">
<link name="C${name}">
<inertial>
<origin xyz="0 0 0.0286" rpy="0 0 0" />
<mass value="0.092"/>
<inertia ixx="4.8343e-05" ixy="0.0" ixz="0.0" iyy="4.8343e-05" iyz="0.0" izz="4.6517e-05"/>
</inertial>

<visual>
<origin xyz="0 0 0.0286" rpy="0 0 0" />
<geometry>
<cylinder radius="0.0318" length="0.0572"/>
</geometry>
</visual>

</link>

<joint name="${parent}_C${name}_joint" type="continuous">
<parent link="${parent}"/>
<child link="C${name}"/>
<origin xyz = "${xyz}" rpy = "${rpy}"/>
<axis xyz = "0 0 1"/>
</joint>

<!-- Wheel Rims -->
<xacro:rim name="wheel_${name}" parent="C${name}" xyz="0 0 0.0572" rpy="0 0 0"/>

<gazebo reference="C${name}">
<visual>
<material>
<ambient>0.7 0.7 0.7 1</ambient>
<diffuse>0.7 0.7 0.7 1</diffuse>
<specular>0.7 0.7 0.7 1</specular>
</material>
</visual>
</gazebo>

</xacro:macro>
</robot>
```

URDF Code Breakdown - Rim

```
<?xml version='1.0'?>
<robot xmlns:xacro="http://www.ros.org/wiki/xacro">
<xacro:include filename="$(find asrl_tutorial)/model/roller.xacro"/>
<xacro:macro name="rim" params="name parent xyz rpy">
<link name="rim_${name}_link">
<inertial>
<origin xyz="0 0 0.0182" rpy="0 0 0" />
<mass value="0.20"/>
<inertia ixz="6.3068e-04" ixy="0.0" ixz="0.0" iyy="6.3068e-04" iyz="0.0" izz="0.0012"/>
</inertial>

<visual>
<origin xyz="0 0 0.0182" rpy="0 0 0" />
<geometry>
<cylinder radius="0.076" length="0.0381"/>
</geometry>
</visual>
</link>

<joint name="${parent}_rim_${name}_joint" type="fixed">
<parent link="${parent}"/>
<child link="rim_${name}_link"/>
<axis xyz = "0 0 1"/>
<origin xyz = "${xyz}" rpy = "${rpy}"/>
</joint>

<xacro:roller name="1" parent="${name}" xyz="0.068 0 0.0091" rpy="1.57 0 0"/>
<xacro:roller name="2" parent="${name}" xyz="0.0647 0.021 0.0291" rpy="1.57 0 0.3142"/>
<xacro:roller name="3" parent="${name}" xyz="0.055 0.04 0.0091" rpy="1.57 0 0.6283"/>
<xacro:roller name="4" parent="${name}" xyz="0.04 0.055 0.0291" rpy="1.57 0 0.9425"/>
<xacro:roller name="5" parent="${name}" xyz="0.021 0.0647 0.0091" rpy="1.57 0 1.2566"/>
<xacro:roller name="6" parent="${name}" xyz="0.068 0.0291" rpy="1.57 0 1.5708"/>
<xacro:roller name="7" parent="${name}" xyz="-0.021 0.0647 0.0091" rpy="1.57 0 1.8850"/>
<xacro:roller name="8" parent="${name}" xyz="-0.04 0.055 0.0291" rpy="1.57 0 2.1991"/>
<xacro:roller name="9" parent="${name}" xyz="-0.055 0.04 0.0091" rpy="1.57 0 2.5133"/>
<xacro:roller name="10" parent="${name}" xyz="-0.0647 0.021 0.0291" rpy="1.57 0 2.8274"/>
<xacro:roller name="11" parent="${name}" xyz="-0.068 0 0.0091" rpy="1.57 0 3.1416"/>
<xacro:roller name="12" parent="${name}" xyz="-0.0647 -0.021 0.0291" rpy="1.57 0 3.4558"/>
<xacro:roller name="13" parent="${name}" xyz="-0.055 -0.04 0.0091" rpy="1.57 0 3.7699"/>
<xacro:roller name="14" parent="${name}" xyz="-0.04 -0.055 0.0291" rpy="1.57 0 4.0841"/>
<xacro:roller name="15" parent="${name}" xyz="-0.021 -0.0647 0.0091" rpy="1.57 0 4.3982"/>
<xacro:roller name="16" parent="${name}" xyz="0 -0.068 0.0291" rpy="1.57 0 4.7124"/>
<xacro:roller name="17" parent="${name}" xyz="0.021 -0.0647 0.0091" rpy="1.57 0 5.0265"/>
<xacro:roller name="18" parent="${name}" xyz="0.04 -0.055 0.0291" rpy="1.57 0 5.3407"/>
<xacro:roller name="19" parent="${name}" xyz="0.055 -0.04 0.0091" rpy="1.57 0 5.6549"/>
<xacro:roller name="20" parent="${name}" xyz="0.0647 -0.021 0.0291" rpy="1.57 0 5.9690"/>

<gazebo reference="rim_${name}_link">
<visual>
<material>
<ambient>0.2 0.2 0.2 1</ambient>
<diffuse>0.2 0.2 0.2 1</diffuse>
<specular>0.2 0.2 0.2 1</specular>
</material>
</visual>
</gazebo>
</xacro:macro>
</robot>
```

URDF Code Breakdown - Roller

```
<?xml version='1.0'?>
<robot xmlns:xacro="http://www.ros.org/wiki/xacro">
  <xacro:macro name="roller" params="name parent xyz rpy">
    <link name="roller_${name}_rim_${parent}_link">
      <inertial>
        <origin xyz="0 0 0" rpy="0 0 0" />
        <inertia ixx="1e-6" ixy="0.0" ixz="0.0" iyy="1e-6" iyz="0.0" izz="1e-6"/>
        <mass value="0.01"/>
      </inertial>
      <visual>
        <origin xyz="0 0 0" rpy="0 0 0" />
        <geometry>
          <cylinder radius="0.01" length="0.03"/>
        </geometry>
      </visual>

      <collision>
        <origin xyz="0 0 0" rpy="0 0 0" />
        <geometry>
          <cylinder radius="0.01" length="0.03"/>
        </geometry>
      </collision>
    </link>

    <joint name="roller_${name}_rim_${parent}_joint" type="continuous">
      <parent link="rim_${parent}_link"/>
      <child link="roller_${name}_rim_${parent}_link"/>
      <axis xyz="0 0 1"/>
      <origin xyz="${xyz}" rpy="${rpy}"/>
    </joint>

    <gazebo reference="roller_${name}_rim_${parent}_link">
      <mu1 value="100"/>
      <mu2 value="100"/>
      <visual>
        <material>
          <ambient>0.6 0.6 0.6 1</ambient>
          <diffuse>0.6 0.6 0.6 1</diffuse>
          <specular>0.6 0.6 0.6 1</specular>
        </material>
      </visual>
    </gazebo>
  </xacro:macro>
</robot>
```

Creating The URDF

- Create the Meshes folder in correct directory
 - cd ws_tutorial/src/asrl_tutorial
 - mkdir meshes
- Download this STL file and place in folder
 - https://github.com/UCF-ASRL/ROS-2-Tutorial/blob/main/asrl_tutorial/meshes/chassis.stl

Your structure should look like this

```
└── src
    └── asrl_tutorial
        ├── CMakeLists.txt
        ├── include
        │   └── asrl_tutorial
        ├── meshes
        └── model
            ├── chassis.xacro
            ├── connector.xacro
            ├── main.xacro
            ├── rim.xacro
            └── roller.xacro
        └── package.xml
```

Creating The Launch File

- **Create a new folder**

- cd ws_tutorial/src/asrl_tutorial
 - mkdir launch

- **Create the Launch File**

- touch tutorial.launch.py

Your structure should look like this

```
└── src
    └── asrl_tutorial
        ├── CMakeLists.txt
        ├── include
        │   └── asrl_tutorial
        ├── launch
        │   └── tutorial.launch.py
        ├── meshes
        │   └── chassis.stl
        ├── model
        │   ├── chassis.xacro
        │   ├── connector.xacro
        │   ├── main.xacro
        │   ├── rim.xacro
        │   └── roller.xacro
        └── package.xml
```

Launch File Code Breakdown

- Install Packages used in the launch file
 - sudo apt install ros-jazzy-joint-state-publisher-gui
 - sudo apt install ros-jazzy-rviz2

```
# Imports
# For os path commands
import os

# Importing Model and world and launch files
from ament_index_python.packages import get_package_share_directory # type: ignore

# Import Ros Launch
import launch_ros # type: ignore
from launch_ros.actions import Node # type: ignore
from launch_ros.substitutions import FindPackageShare # type: ignore

# Core structure
from launch import LaunchDescription
from launch.actions import DeclareLaunchArgument, IncludeLaunchDescription, RegisterEventHandler # type: ignore
from launch.event_handlers import OnProcessExit # type: ignore
from launch.launch_description_sources import PythonLaunchDescriptionSource # type: ignore
from launch.substitutions import LaunchConfiguration, PathJoinSubstitution, TextSubstitution # type: ignore

# Xacro
import xacro # type: ignore

#-----
# Launch
def generate_launch_description():

    # Names
    # Base Files
    package_name = 'asrl_tutorial'
    robot_name = 'rome_urdf'
    urdf_name = 'main.xacro'
    urdf_folder_name = 'model'
    rviz_param_file = 'rviz_config.rviz'

    # Paths
    path_to_urdf = os.path.join(get_package_share_directory(package_name),urdf_folder_name,urdf_name)
    path_to_rviz_params = os.path.join(get_package_share_directory(package_name),'config',rviz_param_file)

    # Robot Description
    robot_description = xacro.process_file(path_to_urdf).toxml()

    # Publishers
    # Robot State Publisher
    robot_state_publisher_node = Node(
        package='robot_state_publisher',
        executable='robot_state_publisher',
        output='screen',
        parameters=[{'robot_description': robot_description}],
    )

    joint_state_publisher_gui_node = Node(
        package='joint_state_publisher_gui',
        executable='joint_state_publisher_gui',
        name='joint_state_publisher_gui',
    )

    # RVIZ
    rviz_node = Node(
        package='rviz2',
        executable='rviz2',
        name='rviz2',
        output='screen',
        arguments=[ '-d',path_to_rviz_params],
    )

    # here we create an empty launch description object
    ld = LaunchDescription()

    # Add Launch Nodes
    ld.add_action(robot_state_publisher_node)
    ld.add_action(joint_state_publisher_gui_node)
    ld.add_action(rviz_node)

    return ld
```

Rviz Configs

- **Create a new folder**
 - cd ws_tutorial/src/asrl_tutorial
 - mkdir config
- **Download this file into the config folder**
 - https://github.com/UCF-ASRL/ROS-2-Tutorial/blob/main/asrl_tutorial/config/rviz_config.rviz

Your structure should look like this

```
└── src
    └── asrl_tutorial
        ├── CMakeLists.txt
        ├── config
        │   └── rviz_config.rviz
        ├── include
        │   └── asrl_tutorial
        ├── launch
        │   └── tutorial.launch.py
        ├── meshes
        │   └── chassis.stl
        ├── model
        │   ├── chassis.xacro
        │   ├── connector.xacro
        │   ├── main.xacro
        │   ├── rim.xacro
        └── roller.xacro
    └── package.xml
```

Updating CMakeLists.txt

- Update the CMakeLists.txt file to include your folders

```
cmake_minimum_required(VERSION 3.8)
project(asrl_tutorial)

if(CMAKE_COMPILER_IS_GNUCXX OR CMAKE_CXX_COMPILER_ID MATCHES "Clang")
  add_compile_options(-Wall -Wextra -Wpedantic)
endif()

# find dependencies
find_packageament_cmake REQUIRED)
# uncomment the following section in order to fill in
# further dependencies manually.
# find_package(<dependency> REQUIRED)

#Install file to share
install(
  DIRECTORY model meshes launch config
  DESTINATION share/${PROJECT_NAME}
)

if(BUILD_TESTING)
  find_packageament_lint_auto REQUIRED)
  # the following line skips the linter which checks for copyrights
  # comment the line when a copyright and license is added to all source files
  set(ament_cmake_copyright_FOUND TRUE)
  # the following line skips cpplint (only works in a git repo)
  # comment the line when this package is in a git repo and when
  # a copyright and license is added to all source files
  set(ament_cmake_cpplint_FOUND TRUE)
  ament_lint_auto_find_test_dependencies()
endif()

ament_package()
```

Updating package.xml

- Update package.xml to include important packages

```
<?xml version="1.0"?>
<?xml-model href="http://download.ros.org/schema/package_format3.xsd" schematypens="http://www.w3.org/2001/XMLSchema"?>
<package format="3">
  <name>asrl_tutorial</name>
  <version>0.0.0</version>
  <description>ASRL Tutorial ROS 2 Basics</description>
  <maintainer email="bastian.weiss@ucf.edu">Bastian Weiss</maintainer>
  <license>TODO: License declaration</license>

  <buildtool_depend>ament_cmake</buildtool_depend>

  <test_depend>ament_lint_auto</test_depend>
  <test_depend>ament_lint_common</test_depend>

  <depend>xacro</depend>
  <depend>robot_state_publisher</depend>
  <depend>joint_state_publisher_gui</depend>

  <export>
    <build_type>ament_cmake</build_type>
  </export>
</package>
```

Launching our Package

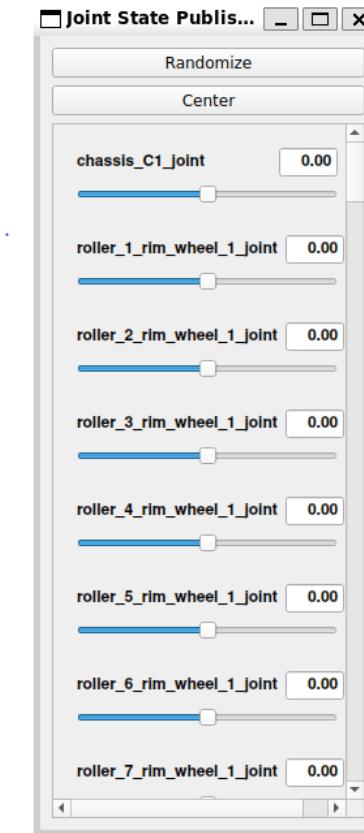
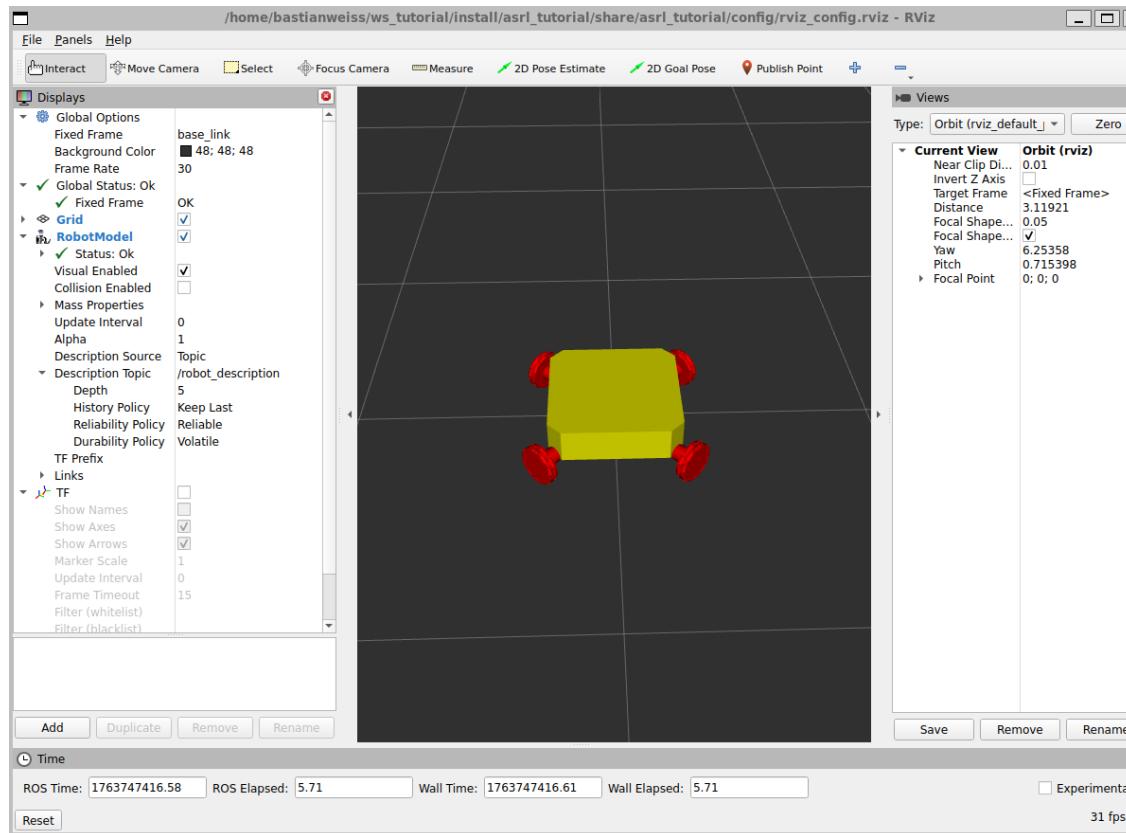
- **Make sure ROS 2 is sourced**
 - source /opt/ros/jazzy/setup.bash
- **Make sure you are in correct directory**
 - cd ws_tutorial
- **Build the package**
 - colcon build
- **Source the Install Folder**
 - source install/setup.bash
- **Launch the Package**
 - ros2 launch asrl_tutorial tutorial.launch.py

Your structure should look like this

```
└── src
    └── asrl_tutorial
        ├── CMakeLists.txt
        ├── config
        │   └── rviz_config.rviz
        ├── include
        │   └── asrl_tutorial
        ├── launch
        │   └── tutorial.launch.py
        ├── meshes
        │   └── chassis.stl
        ├── model
        │   ├── chassis.xacro
        │   ├── connector.xacro
        │   ├── main.xacro
        │   ├── rim.xacro
        │   └── roller.xacro
        └── package.xml
```

Launching our Package

- You should now see Rviz open with the Joint State Publisher GUI
- You can rotate the joints to test using the GUI





End of Tutorial
