URDF Model

This lesson uses the Raspberry Pi as an example. For Raspberry Pi and Jetson-Nano boards, you need to open a terminal on the host computer and enter the command to enter the Docker container. Once inside the Docker container, enter the commands mentioned in this lesson in the terminal. For instructions on entering the Docker container from the host computer, refer to **[01. Robot Configuration and Operation Guide] -- [4.Enter Docker (For JETSON Nano and RPi 5)]**. For RDKX5 and Orin boards, simply open a terminal and enter the commands mentioned in this lesson.

1. Program Function Description

After the program starts, the URDF model file of the vehicle model will be displayed in rviz. There will also be a GUI debugging tool window; by sliding the progress bar, you can animate the model file.

2. Program Code Reference Path

For Raspberry Pi and Jetson-Nano boards, you need to enter the Docker container first. For RDKX5 and Orin controllers, this is not necessary.

Enter the Docker container (see [Docker course] --- [4. Docker Startup Script] for steps).

All the following commands must be executed within the same Docker container **(see [Docker course] --- [3. Docker Submission and Multi-Terminal Access] for steps).**

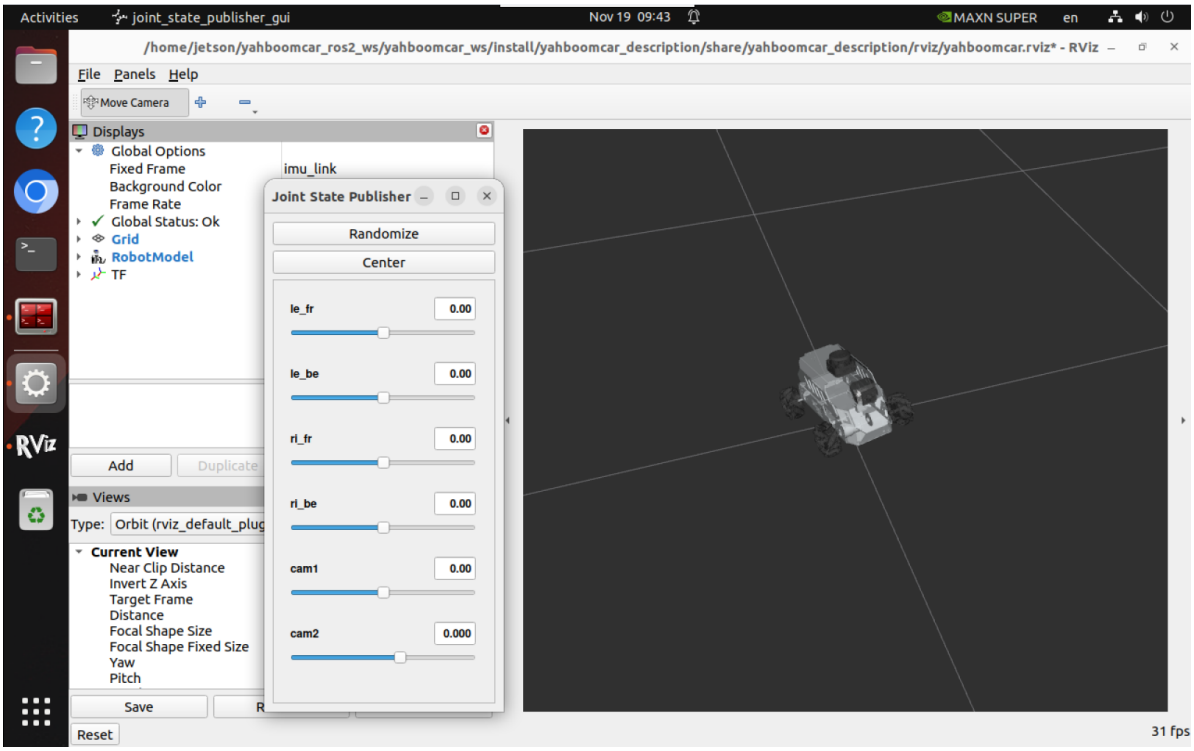
The source code for this function is located at:

```
~/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_description/launch/display_M1.
launch.py
```

3. Program Startup

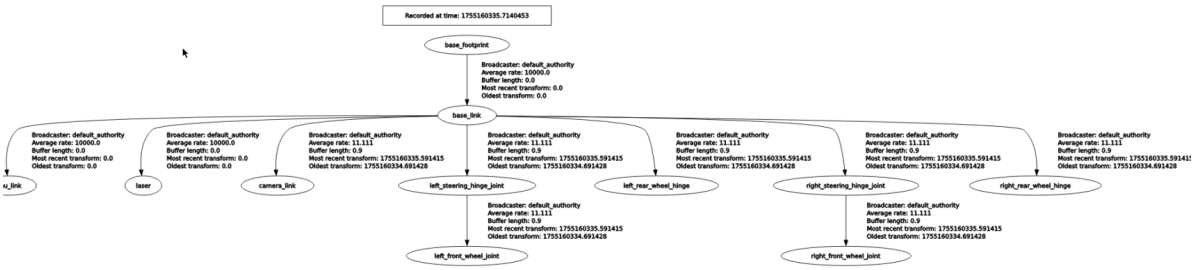
Enter the following in the terminal:

```
ros2 launch yahboomcar_description display_M1.launch.py
```



You can view the TF tree by entering the following in the Docker terminal:

```
ros2 run rqt_tf_tree rqt_tf_tree
```



4. Introduction to URDF

URDF, short for Unified Robot Description Format, is a robot model file described in XML format, similar to D-H parameters.

```
<?xml version="1.0" encoding="utf-8"?>
<robot name="yahboomcar_M1"
xmlns:xacro="http://ros.org/wiki/xacro">
```

The first line is a required field for XML, describing the XML version information.

The second line describes the current robot name; all information about the current robot is contained within the `robot` tag.

4.1 Components

- Link, a connecting rod, can be imagined as a human arm.
- Joint, a joint, can be imagined as a human elbow joint.

Relationship between link and joint: Two links are connected by a joint, like an arm with a forearm (link) and upper arm (link) connected by an elbow joint (joint).

4.1.1 Link

1) Introduction

In the URDF descriptive language, a link is used to describe physical properties.

- Describes visual display, `<link>` tag.
- Describes collision properties, `<collision>` tag.
- Describes physical inertia, `<physical>` tag (less commonly used).

Links can also describe link dimensions, color, shape, inertial matrix, collision properties, etc. Each link becomes a coordinate system.

2) Example code (yahboomcar_M1.urdf.xacro)

```
<link name="imu_link"/>
  <link
    name="base_link">
    <inertial>
      <origin
        xyz="-0.015746541514953 -3.81194310309278E-06 0.0445016032560171"
        rpy="0 0 0" />
      <mass
        value="1.07553948051665" />
      <inertia
        ixx="0.00172418331028207"
        ixy="-2.35223116839001E-07"
        ixz="8.3813050960105E-05"
        iyy="0.00329378069831256"
        iyz="-2.25863771079264E-07"
        izz="0.00445732461717744" />
    </inertial>
    <visual>
      <origin
        xyz="0 0 0"
        rpy="0 0 0" />
      <geometry>
        <mesh

 filename="package://yahboomcar_description/meshes/M1Ackermann/base_link.STL" />
      </geometry>
      <material
        name="">
        <color
          rgba="0.898039215686275 0.917647058823529 0.929411764705882 1" />
      </material>
    </visual>
    <collision>
      <origin
        xyz="0 0 0"
        rpy="0 0 0" />
      <geometry>
        <mesh

 filename="package://yahboomcar_description/meshes/M1Ackermann/base_link.STL" />
      </geometry>
    </collision>
  </link>
```

```
  <link
```

3) Tag Descriptions

- origin

Describes pose information; the xyz attribute describes the coordinates in the environment, and the rpy attribute describes the pose itself.

- mess

Describes the quality of the link.

- inertia

Inertial reference frame. Due to the symmetry of the rotational inertia matrix, only six upper triangular elements ixx, ixy, ixz, iyy, iyz, and izz are needed as attributes.

- geometry

The tag describes the shape; the mesh attribute is mainly used to load texture files, and the filename attribute is the file path of the texture.

- material

The tag describes the material; the name attribute is **required**, can be empty, and can be repeated. Red, green, blue, and transparency are described by the rgba attribute in the 【color】 tag, separated by spaces. The color range is [0-1].

4.1.2 Joints

1) Introduction

Describes the relationship between two joints, their position and velocity limitations, and kinematic and dynamic properties.

Joint Types:

- Fixed: Fixed joint. Movement is not allowed; it serves as a connection.
- Continuous: Rotational joint. Can rotate continuously without rotational angle limitations.
- Revolute: Rotational joint. Similar to continuous, but with rotational angle limitations.
- Prismatic: Sliding joint. Can move along a certain axis; position is limited.
- Floating: Floating joint. Has six degrees of freedom (3T3R).
- Planar: Planar joint. Allows translation or rotation above a plane.

2) Example Code (yahboomcar_M1.urdf.xacro)

```
<joint name="front_right_joint" type="continuous">
    <origin xyz="0.08 -0.0845 -0.0389" rpy="-1.5703 0 3.14159"/>
    <parent link="base_link"/>
    <child link="front_right_wheel"/>
    <axis xyz="0 0 1" rpy="0 0 0"/>
    <limit effort="100" velocity="1"/>
</joint>
```

In the `joint` tag, the `name` attribute is **required**, describing the name of the joint and ensuring it is unique.

In the `joint` tag, the `type` attribute corresponds to one of the six major joint types.

3) Tag Introduction

- `origin`

This child tag indicates the relative position of the rotational joint to the coordinate system of the `parent`.

- `parent`, `child`

The `parent` and `child` child tags represent two links to be connected; the `parent` is the reference point, and the `child` rotates around the `parent`.

- `axis`

This child tag indicates the axis (xyz) around which the `child` link rotates and the amount of rotation around a fixed axis.

- `limit`

This child tag primarily restricts the `child`. The `lower` and `upper` attributes limit the range of radians of rotation, the `effort` attribute limits the range of forces applied during rotation (positive or negative values in Newtons or N), and the `velocity` attribute limits the speed of rotation (in meters per second or m/s).

- `mimic`

Describes the relationship between this joint and existing joints.

- safety_controller

Describes the parameters of the safety controller. Protects the movement of the robot's joints.