URDF: Unified Robot Description Format

File format for specifying the geometry and organization of robots in ROS

main tags inside a link are <visual> <collision> and <inertial>
visual-used for link geometry.
Collision-bounding shape around the link for collision detection
inertial-describes mass and inertia

geometry-metres for all lengths. Types are box, cylinder, sphere, mesh
mesh format: <mesh filename="package://package_name/meshes/l_finger.dae"/>
origin-used to define translation and rotation using xyz and rpy
material-defines color using rgba
axis-axis of rotation
dynamics-damping and friction

joint types: fixed (no motion), continuous (rotation about axis), revolute (rotation about axis with limits in radians), prismatic (translation about axis), floating (6 dof), planar (xy axis and yaw)

axis colors=xyz=red green blue

R2D2:
Resources: ROS2 Docs and Kevin Wood Yt

A major error I came across was that using robot state publisher with fixed joints was working but its said no tranform from various links to base when the joint types were changed. This can be resolved be adding a joint_state_publisher_gui node

First the GUI parses the URDF and finds all the non-fixed joints and their limits. Then, it uses the values of the sliders to publish sensor_msgs/msg/JointState messages. Those are then used by robot_state_publisher to calculate all of transforms between the different parts. The resulting transform tree is then used to display all of the shapes in Rviz

Xacro:

xacro model.xacro > model.urdf: converts a .xacro file to a .urdf file

You can either convert xacro to urdf and have robot_state_publisher read it:

```
path_to_urdf = get_package_share_path('turtlebot3_description') / 'urdf' / 'turtlebot3_burger.urdf'
robot_state_publisher_node = launch_ros.actions.Node(
    package='robot_state_publisher',
    executable='robot_state_publisher',
    parameters=[{
        'robot_description': ParameterValue(
            Command(['xacro ', str(path_to_urdf)]), value_type=str
        )
    }]
)
```

or use urdf_launch package to load the urdf or xacro file

```
from launch import LaunchDescription
from launch.actions import IncludeLaunchDescription
from launch.substitutions import PathJoinSubstitution
from launch_ros.substitutions import FindPackageShare


def generate_launch_description():
    return LaunchDescription([
        IncludeLaunchDescription(
            PathJoinSubstitution([FindPackageShare('urdf_launch'), 'launch', 'display.launch.py']),
            launch_arguments={
                'urdf_package': 'turtlebot3_description',
                'urdf_package_path': PathJoinSubstitution(['urdf', 'turtlebot3_burger.urdf'])
            }.items()
        )
    ])
```

Xacro property: used when same thing is repeated

<xacro:property name=”property_name” value-”...”/>

you can use it like:

<… variable=”${property_name}” …/>

Macro: replaces many lines of code and can have input params

<xacro:macro name-”macro_name” params=”param_1 param_2….param_n”>
…
</xacro:macro>

you can use it like:
<xacro:macro_name param_1=”val”/>

Making separate xacro files:

file name is filename.xacro

<?xml version=”1.0”?>
<robot xmlns:xacro=”https://www.ros.org/wiki/xacro”>
<xacro:property name=” ” value=” “/>
<xacro:macro …...>
</xacro:macro>
</robot>

to use it in a urdf file:

<?xml version=”1.0”?>
<robot name=” “ xmlns:xacro=”https://www.ros.org/wiki/xacro”>
….
<xacro:include filename=”filename.xacro”/>

….
</robot>