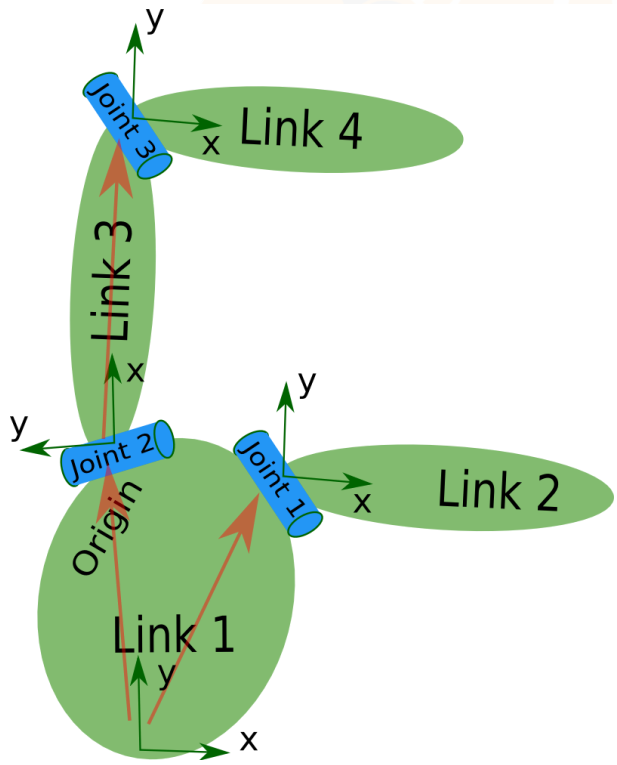
MOBILE ROBOTS (01416511)

Mobile Robot Chapter 5 : URDF

XML Robot Description Format (URDF)

The **Unified Robot Description Format (URDF)** is an XML specification for describing a robot. We aim to keep this specification as general as possible, but obviously, it cannot describe all robots. The main limitation is that only tree structures can be represented, ruling out all parallel robots. Additionally, the specification assumes the robot consists of rigid links connected by joints; flexible elements are not supported. The specification covers:

- Kinematic and dynamic description of the robot.
- Visual representation of the robot.
- Collision model of the robot.



The description of a robot consists of a set of **link elements**, and a set of **joint elements** connecting the links together. So a typical robot description looks something like this:

```
<?xml version="1.0"?>
<?xml-model
href="https://raw.githubusercontent.com/ros/urdfdom/master/xsd/urdf.xsd" ?>
<robot name="pr2" xmlns="http://www.ros.org">
  <link> ... </link>
  <link> ... </link>
  <link> ... </link>

  <joint>  ....  </joint>
  <joint>  ....  </joint>
  <joint>  ....  </joint>
</robot>
```

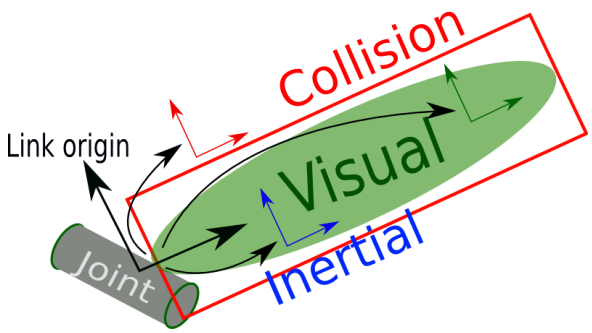
Link Element

The link element describes a rigid body with inertia, visual features, and collision properties. Here is an example of a link element:

```
<link name="my_link">
   <inertial>
     <origin xyz="0 0 0.5" rpy="0 0 0"/>
     <mass value="1"/>
     <inertia ixx="100"  ixy="0"  ixz="0" iyy="100" iyz="0" izz="100" />
   </inertial>

   <visual>
     <origin xyz="0 0 0" rpy="0 0 0" />
     <geometry>
       <box size="1 1 1" />
     </geometry>
     <material name="Cyan">
       <color rgba="0 1.0 1.0 1.0"/>
     </material>
   </visual>

   <collision>
     <origin xyz="0 0 0" rpy="0 0 0"/>
     <geometry>
       <cylinder radius="1" length="0.5"/>
     </geometry>
   </collision>
</link>
```



Attributes

name (required): The name of the link itself.

Elements

<inertial> (optional: defaults to a zero mass and zero inertia if not specified): The link's mass, position of its center of mass, and its central inertia properties.

- **<origin>** (optional: defaults to identity if not specified): This pose (translation, rotation) describes the position and orientation of the link's center of mass frame CCC relative to the link-frame LLL.
    - **xyz** (optional: defaults to zero vector) Represents the position vector from Lo (the link-frame origin) to Co (the link's center of mass) as x L̂x + y L̂y + z L̂z, where L̂x, L̂y, L̂z are link-frame L's orthogonal unit vectors
    - **rpy** (optional: defaults to identity if not specified) Represents the orientation of C's unit vectors Ĉx, Ĉy, Ĉz relative to link-frame L as a sequence of Euler rotations (r p y) in radians. Note: Ĉx, Ĉy, Ĉz do not need to be aligned with the link's principal axes of inertia.
- **<mass>** The mass of the link is represented by the value attribute of this element.
- **<inertia>** This link's moments of inertia **ixx, iyy, izz** and products of inertia **ixy, ixz, iyz** about Co (the link's center of mass) for the unit vectors **Ĉx, Ĉy, Ĉz** fixed in the center-of-mass frame C.

<visual> (optional) The visual properties of the link. This element specifies the shape of the object (box, cylinder, etc.) for visualization purposes. Note: multiple instances of <visual> tags can exist for the same link. The union of the geometry they define forms the visual representation of the link.

- **name** (optional): Specifies a name for a part of a link's geometry. This is useful to be able to refer to specific bits of the geometry of a link.
- **<origin>** (optional: defaults to identity if not specified): The reference frame of the visual element with respect to the reference frame of the link.
    - **xyz** (optional: defaults to zero vector): Represents the x, y, z offset.
    - **rpy** (optional: defaults to identity if not specified): Represents the fixed axis roll, pitch and yaw angles in radians.
- **<geometry>** (required): The shape of the visual object. This can be one of the following:
    - **<box>**: size attribute contains the three side lengths of the box. The origin of the box is in its center.
    - **<cylinder>**: Specify the radius and length. The origin of the cylinder is in its center.

    - **<sphere>**: Specify the radius. The origin of the sphere is in its center.
    - **<mesh>**: A trimesh element specified by a filename, and an optional scale that scales the mesh's axis-aligned-bounding-box. Any geometry format is acceptable but specific application compatibility is dependent on implementation. The recommended format for best texture and color support is Collada .dae files. The mesh file is not transferred between machines referencing the same model. It must be a local file. Prefix the filename with package://<packagename>/<path> to make the path to the mesh file relative to the package <packagename>.
- **<material>** (optional) The material of the visual element. It is allowed to specify a material element outside of the 'link' object, in the top level 'robot' element. From within a link element you can then reference the material by name.
    - **name** name of the material
    - **<color>** (optional) rgba The color of a material specified by a set of four numbers representing red/green/blue/alpha, each in the range of [0,1].
    - **<texture>** (optional) The texture of a material is specified by a filename

<collision> (optional) The collision properties of a link. Note that this can be different from the visual properties of a link, for example, simpler collision models are often used to reduce computation time. Note: multiple instances of <collision> tags can exist for the same link. The union of the geometry they define forms the collision representation of the link.

- **name** (optional) Specifies a name for a part of a link's geometry. This is useful to be able to refer to specific bits of the geometry of a link.
- **<origin>** (optional: defaults to identity if not specified) The reference frame of the collision element, relative to the reference frame of the link.
    - **xyz** (optional: defaults to zero vector) Represents the x, y, z offset.
    - **rpy** (optional: defaults to identity if not specified) Represents the fixed axis roll, pitch and yaw angles in radians.
- **<geometry>** See the geometry description in the above visual element.

Joint Element

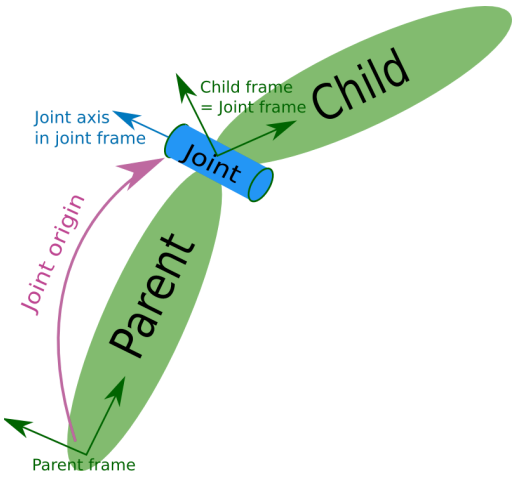
The joint element describes the kinematics and dynamics of the joint and also specifies the safety limits of the joint.

Here is an example of a joint element:

```
<joint name="my_joint" type="floating">
    <origin xyz="0 0 1" rpy="0 0 3.1416"/>
    <parent link="link1"/>
    <child link="link2"/>

    <calibration rising="0.0"/>
    <dynamics damping="0.0" friction="0.0"/>
    <limit effort="30" velocity="1.0" lower="-2.2" upper="0.7" />
    <safety_controller k_velocity="10" k_position="15" soft_lower_limit="-2.0"
soft_upper_limit="0.5" />
```

```
</joint>
```



Attributes

The joint element has two attributes:

- **name** (required) Specifies a unique name of the joint
- **type** (required) Specifies the type of joint, where type can be one of the following:
    - **revolute** — a hinge joint that rotates along the axis and has a limited range specified by the upper and lower limits.
    - **continuous** — a continuous hinge joint that rotates around the axis and has no upper and lower limits.
    - **prismatic** — a sliding joint that slides along the axis, and has a limited range specified by the upper and lower limits.
    - **fixed** — this is not really a joint because it cannot move. All degrees of freedom are locked. This type of joint does not require the <axis>, <calibration>, <dynamics>, <limits> or <safety_controller>.
    - **floating** — this joint allows motion for all 6 degrees of freedom.
    - planar — this joint allows motion in a plane perpendicular to the axis.

Elements

The joint element has following elements:

- **<origin>** (optional: defaults to identity if not specified) This is the transform from the parent link to the child link. The joint is located at the origin of the child link, as shown in the figure above.
    - **xyz** (optional: defaults to zero vector) Represents the x, y, z offset. All positions are specified in metres.
    - **rpy** (optional: defaults to zero vector) Represents the rotation around fixed axis: first roll around x, then pitch around y and finally yaw around z. All angles are specified in radians.
- **<parent>** (required) Parent link name with mandatory attribute:

    - **link** The name of the link that is the parent of this link in the robot tree structure.
- **<child>** (required) Child link name with mandatory attribute:
    - **link** The name of the link that is the child link.
- **<axis>** (optional: defaults to (1,0,0)) The joint axis specified in the joint frame. This is the axis of rotation for revolute joints, the axis of translation for prismatic joints, and the surface normal for planar joints. The axis is specified in the joint frame of reference. Fixed and floating joints do not use the axis field.
    - **xyz** (required) Represents the (x, y, z) components of a vector. The vector should be normalized.
- **<calibration>** (optional) The reference positions of the joint, used to calibrate the absolute position of the joint.
    - **rising** (optional) When the joint moves in a positive direction, this reference position will trigger a rising edge.
    - **falling** (optional) When the joint moves in a positive direction, this reference position will trigger a falling edge.
- **<dynamics>** (optional) An element specifying physical properties of the joint. These values are used to specify modeling properties of the joint, particularly useful for simulation.
    - **damping** (optional, defaults to 0) The physical damping value of the joint (in newton-seconds per metre [N·s/m] for prismatic joints, in newton-metre-seconds per radian [N·m·s/rad] for revolute joints).
    - **friction** (optional, defaults to 0) The physical static friction value of the joint (in newtons [N] for prismatic joints, in newton-metres [N·m] for revolute joints).
- **<limit>** (required only for revolute and prismatic joint) An element can contain the following attributes:
    - **lower** (optional, defaults to 0) An attribute specifying the lower joint limit (in radians for revolute joints, in metres for prismatic joints). Omit if joint is continuous.
    - **upper** (optional, defaults to 0) An attribute specifying the upper joint limit (in radians for revolute joints, in metres for prismatic joints). Omit if joint is continuous.
    - **effort** (required) An attribute for enforcing the maximum joint effort (|applied effort| < |effort|).
    - **velocity** (required) An attribute for enforcing the maximum joint velocity (in radians per second [rad/s] for revolute joints, in metres per second [m/s] for prismatic joints).
- **<mimic>** (optional) This tag is used to specify that the defined joint mimics another existing joint. The value of this joint can be computed as value = multiplier * other_joint_value + offset.

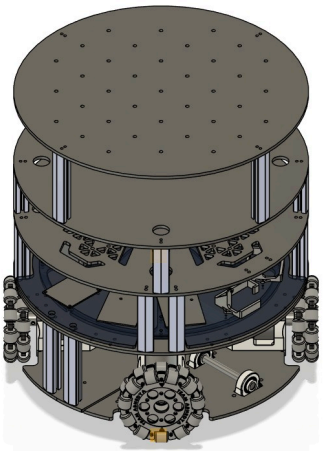
    Expected and optional attributes:

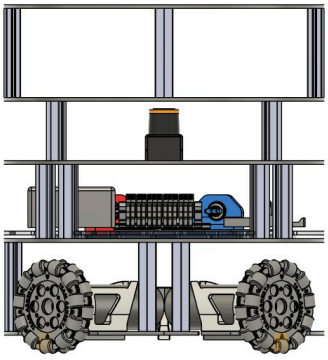
    - **joint** (required) This specifies the name of the joint to mimic.
    - **multiplier** (optional) Specifies the multiplicative factor in the formula above.
    - **offset** (optional) Specifies the offset to add in the formula above. Defaults to 0 (radians for revolute joints, meters for prismatic joints)
- **<safety_controller>** (optional) An element can contain the following attributes:
    - **soft_lower_limit** (optional, defaults to 0) An attribute specifying the lower joint boundary where the safety controller starts limiting the position of the joint. This limit needs to be larger than the lower joint limit (see above).

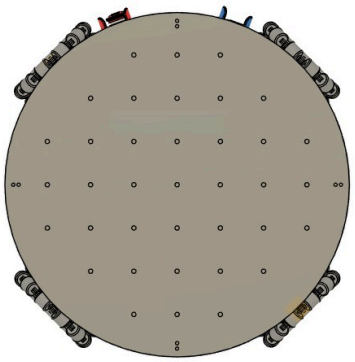
- **soft_upper_limit** (optional, defaults to 0) An attribute specifying the upper joint boundary where the safety controller starts limiting the position of the joint. This limit needs to be smaller than the upper joint limit.
- **k_position** (optional, defaults to 0) An attribute specifying the relation between position and velocity limits.
- **k_velocity** (required) An attribute specifying the relation between effort and velocity limits.

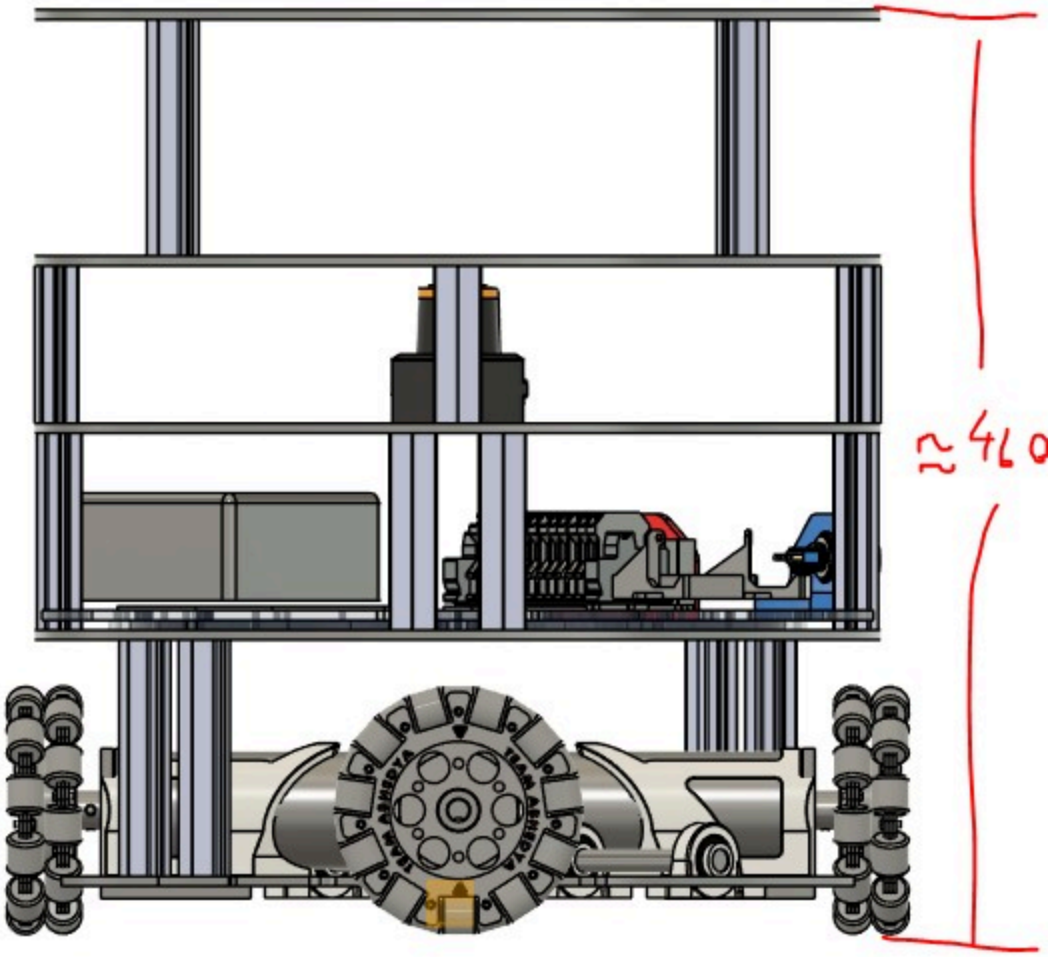
Robot URDF

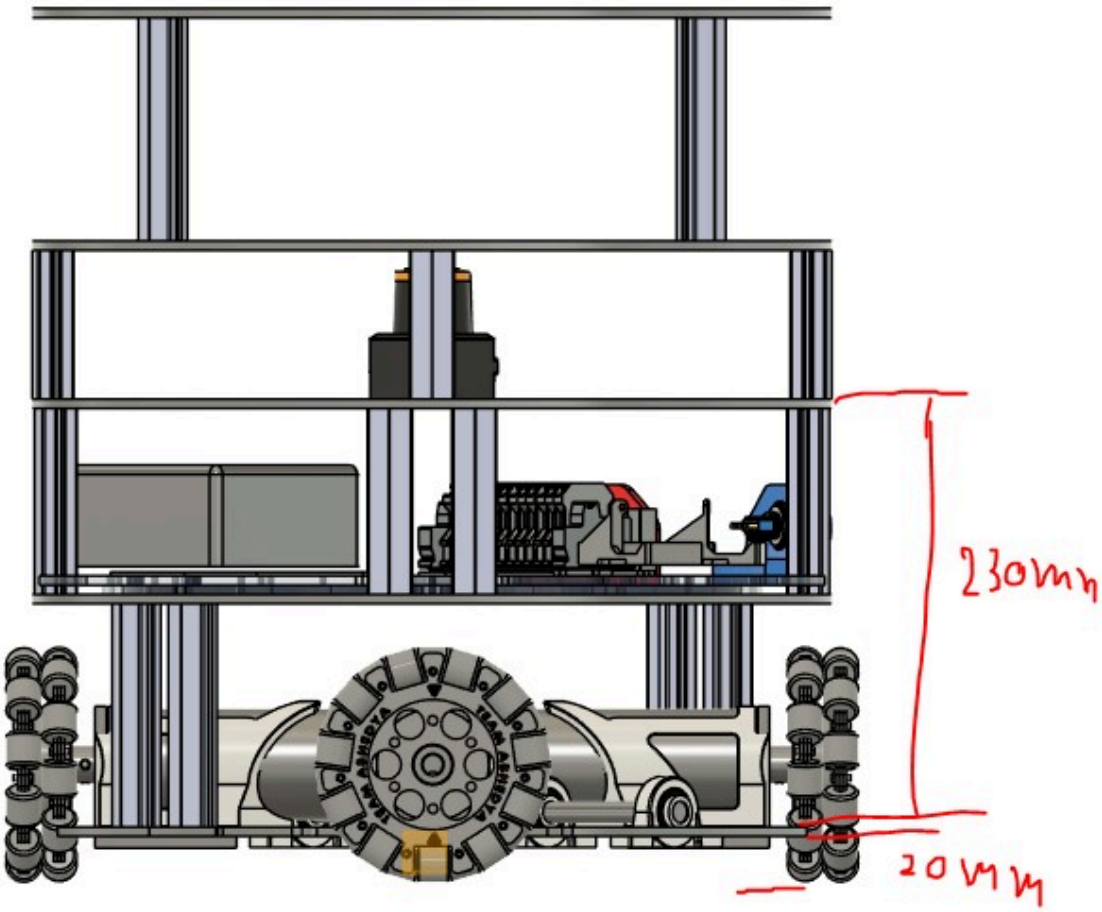
Robot Model

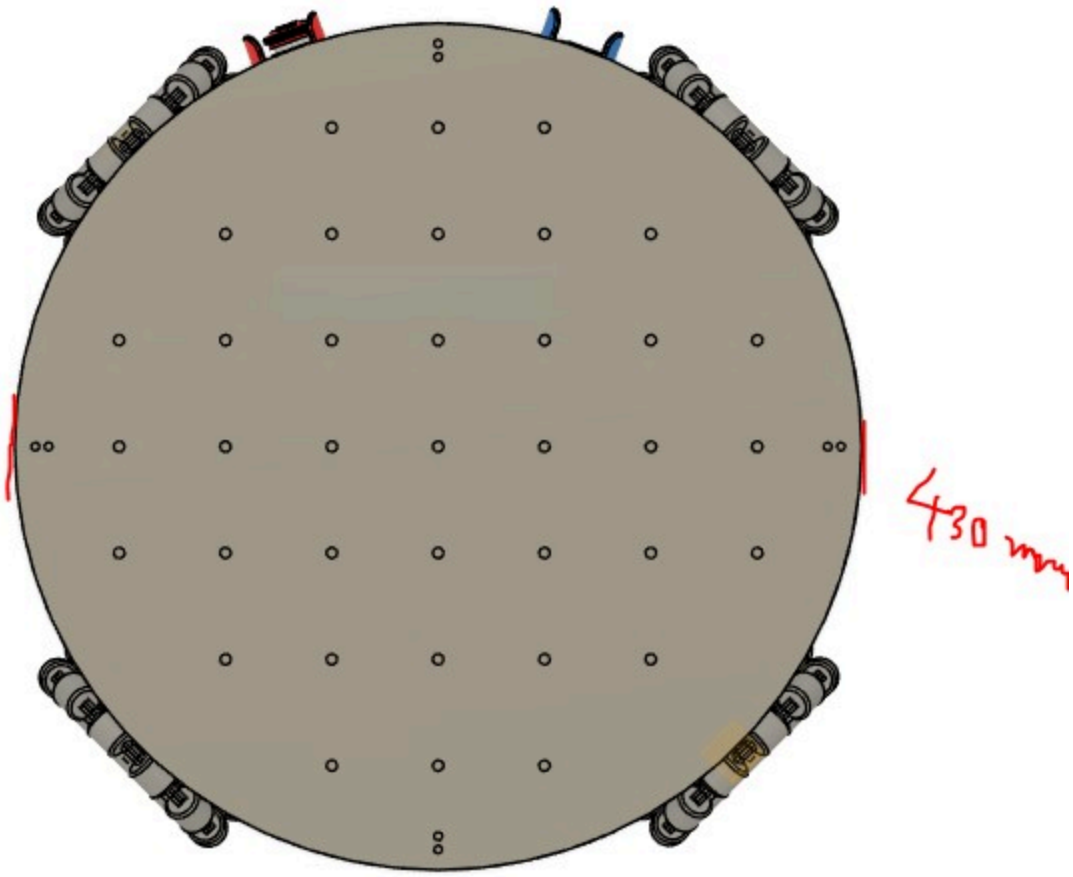







≃ 460
TEAM ASHREYA
TEAM ASHREYA


230mm
20mm
base footprint → base_link



Example Robot URDF file

Create ***ired_bringup*** package for add file robot.urdf.xacro in folder ***urdf***.

```
$ cd ~/catkin_ws/src
$ catkin_create_pkg ired_bringup roscpp rospy std_msgs
$ cd ~/catkin_ws
$ catkin_make
$ source ~/.bashrc
```

Create ***robot.urdf.xacro*** in the package folder.

```
$ roscd ired_bringup
$ mkdir urdf
$ cd urdf
$ touch robot.urdf.xacro
$ code robot.urdf.xacro
```

Example XML format for URDF file

```
<?xml version="1.0"?>
<robot name="IRED">
   <!-- link -->
   <link name="base_footprint"/>
   <link name="base_link"/>
   <link name="wheel_front_left_link"/>
   <link name="wheel_front_right_link"/>
   <link name="wheel_rear_left_link"/>
   <link name="wheel_rear_right_link"/>
   <link name="base_scan"/>
   <link name="aruco_camera_link"/>


  <!-- joint -->
   <joint name="base_joint" type="fixed">
      <parent link="base_footprint"/>
      <child link="base_link"/>
      <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
   </joint>

   <joint name="scan_joint" type="fixed">
      <parent link="base_link"/>
      <child link="base_scan"/>
      <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
   </joint>

   <joint name="aruco_camera_joint" type="fixed">
      <parent link="base_link"/>
      <child link="aruco_camera_link"/>
      <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
   </joint>


   <joint name="wheel_front_right_joint" type="continuous">
      <parent link="base_link"/>
      <child link="wheel_front_right_link"/>
      <axis xyz="0 0 1"/>
      <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
   </joint>

   <joint name="wheel_front_left_joint" type="continuous">
      <parent link="base_link"/>
      <child link="wheel_front_left_link"/>
      <axis xyz="0 0 1"/>
      <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
   </joint>

   <joint name="wheel_rear_right_joint" type="continuous">
      <parent link="base_link"/>
      <child link="wheel_rear_right_link"/>
```

```
    <axis xyz="0 0 1"/>
    <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
  </joint>

  <joint name="wheel_rear_left_joint" type="continuous">
    <parent link="base_link"/>
    <child link="wheel_rear_left_link"/>
    <axis xyz="0 0 1"/>
    <origin xyz="0.0 0.0 0.0" rpy="0.0 0.0 0.0"/>
  </joint>
</robot>
```

Create a ***launch*** file for testing your robot ***URDF***.

```
$ roscd ired_bringup
$ mkdir launch
$ touch bringup.launch
$ code bringup.launch
```

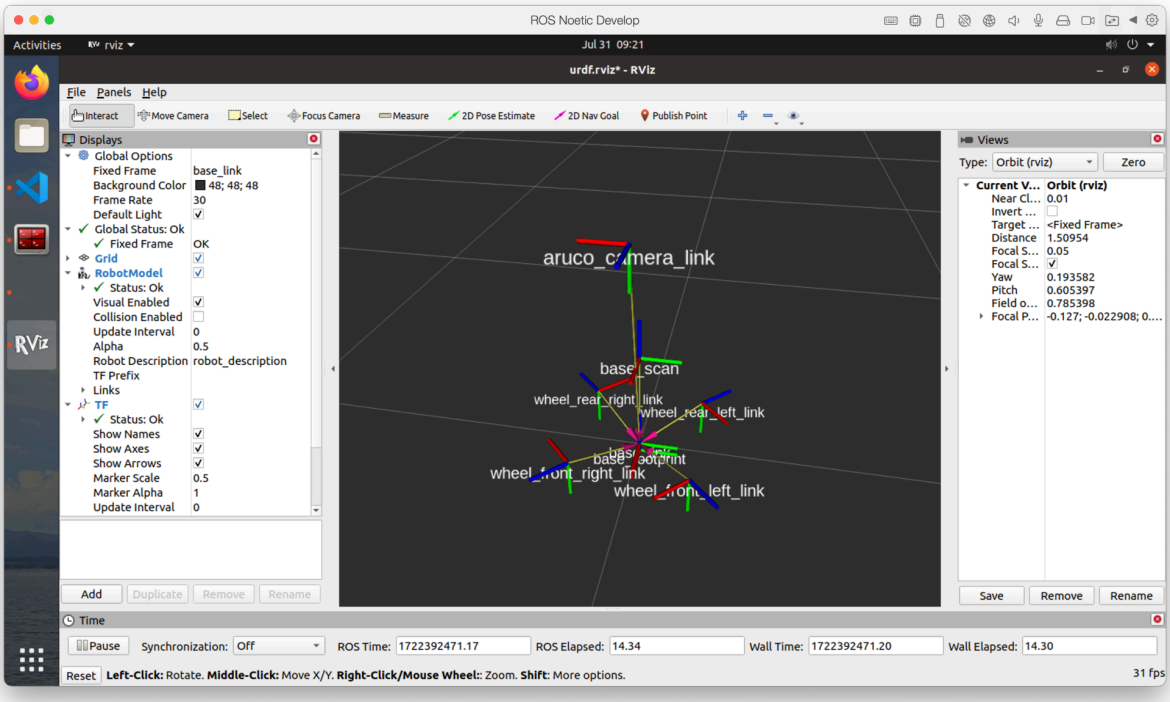
Example ***launch*** file for testing your robot ***URDF***.

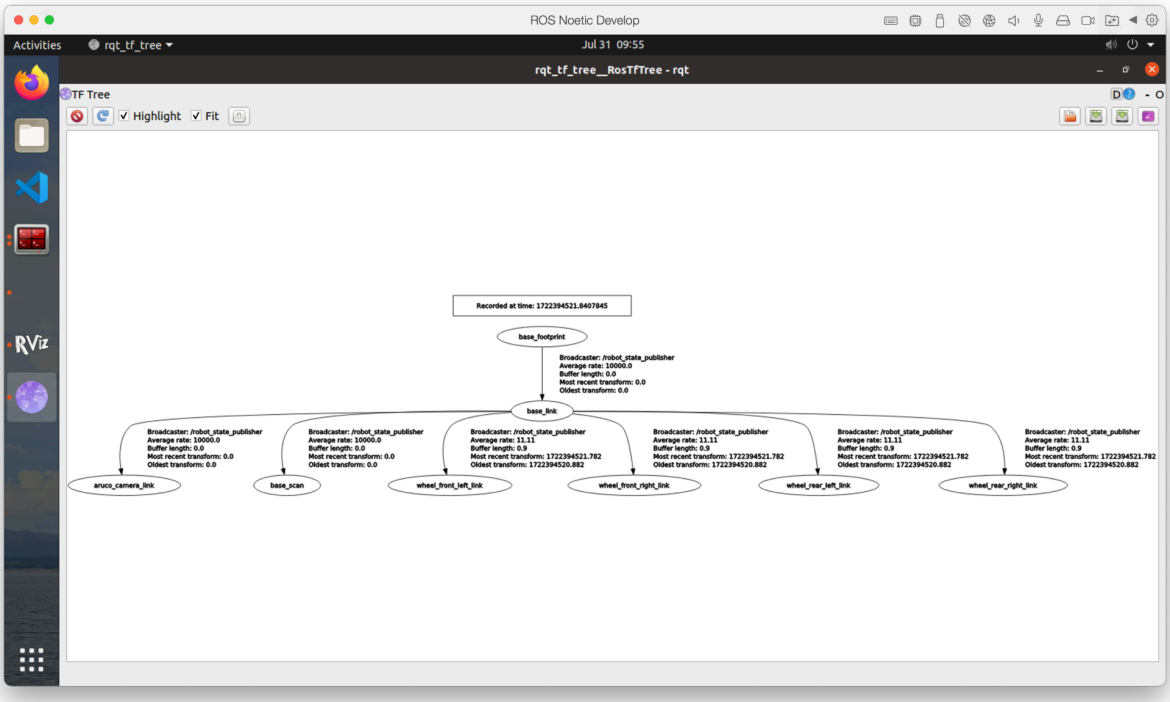
```
<?xml version="1.0"?>
<launch>
  <arg name="urdf_file" default="$(find xacro)/xacro --inorder '$(find
ired_bringup)/urdf/robot.urdf.xacro'"/>
  <param name="robot_description" command="$(arg urdf_file)" />

  <node name="robot_state_publisher" pkg="robot_state_publisher" type="robot_state_publisher"
output="screen">
    <param name="publish_frequency" type="double" value="50.0" />
    <param name="tf_prefix" value=""/>
  </node>
</launch>
```

Command for testing your robot URDF with RViz.

```
$ roslaunch urdf_tutorial display.launch model:=<Your URDF file path>
$ rosrun rqt_tf_tree rqt_tf_tree
```





Add this launch code to run the LiDAR in bringup.launch:

```
<!-- LiDAR -->
```

```
<arg name="lidar_frame" default="base_scan"/>
<arg name="lidar_port" default="/dev/ttyUSB0"/>
<include file="$(find rplidar_ros)/launch/rplidar_ros.launch">
    <arg name="frame_id" value="$(arg lidar_frame)"/>
    <arg name="serial_port" value="$(arg lidar_port)"/>
</include>
```

Assignment

Group Assignment: Create your robot URDF file, open it in RViz and display the data with the transform from the URDF file. Take a screenshot and submit it to Classroom. The group representative should submit the file to the Classroom.