Sensing

Estimated time to completion: **25 minutes**

6.3 Hands-on Practice!

Now that you know how to add a lidar sensor to your robot, let's practice further by adding two more sensors: an RGB camera and a PointCloud camera.

- Exercise 6.1.1 -

- Add two more sensors to the box_bot:
  - RGB wide-angle camera
  - PointCloud camera
- Create a new URDF file called **`box_bot_final.urdf`** , which will be the final version of the box_bot.
- Create also the corresponding new launch files to have everything up and running:
  - **`spawn_robot_ros2_final.launch.xml`**
  - **`urdf_visualize_final.launch.py`**

- Notes -

This is the structure for adding both sensors:

In [ ]:

```
<gazebo reference="pointcloud_link">
    <sensor type="ray" name="pointcloud_sensor">
        <ray>
        <scan>
            <horizontal>
                <samples>50</samples>
                <resolution>1.0</resolution>
                <min_angle>-1.0</min_angle>
                <max_angle>1.0</max_angle>
            </horizontal>
            <vertical>
                <samples>50</samples>
                <resolution>1.0</resolution>
                <min_angle>-1.0</min_angle>
                <max_angle>1.0</max_angle>
            </vertical>
        </scan>
        <range>
            <min>0.10</min>
            <max>5.0</max>
            <resolution>0.01</resolution>
        </range>
        <!-- Using gazebo's noise instead of plugin's -->
        <noise>
            <type>gaussian</type>
            <mean>0.0</mean>
            <stddev>0.01</stddev>
        </noise>
        </ray>
        <!-- Using gazebo's update rate instead of plugin's -->
        <update_rate>30</update_rate>
        <plugin name="gazebo_ros_block_laser_controller" filename="libgazebo_ros_ray_sensor.so">
        <!-- Change namespace and output topic so published topic is /rrbot/laser/pointcloud -->
        <ros>
            <namespace>box_bot</namespace>
            <argument>~/out:=pointcloud</argument>
        </ros>
        <!-- Set output to sensor_msgs/PointCloud to get same output type as gazebo_ros_block_laser -->
        <output_type>sensor_msgs/PointCloud</output_type>
```

```
        <frame_name>pointcloud_link</frame_name>

        <!-- min_intensity instead of hokuyoMinIntensity -->
        <min_intensity>100.0</min_intensity>
        </plugin>
    </sensor>
</gazebo>


<!-- RGB CAMERA -->

  <gazebo reference="rgb_camera_link_frame">
      <sensor name="camera" type="wideanglecamera">
        <camera>
          <horizontal_fov>6.283</horizontal_fov>
          <image>
            <width>320</width>
            <height>240</height>
          </image>
          <clip>
            <near>0.1</near>
            <far>100</far>
          </clip>
          <lens>
            <type>custom</type>
            <custom_function>
              <c1>1.05</c1>
              <c2>4</c2>
              <f>1.0</f>
              <fun>tan</fun>
            </custom_function>
            <scale_to_hfov>true</scale_to_hfov>
            <cutoff_angle>3.1415</cutoff_angle>
            <env_texture_size>512</env_texture_size>
          </lens>
          <always_on>1</always_on>
          <update_rate>30</update_rate>
        </camera>
        <plugin name="camera_controller" filename="libgazebo_ros_camera.so">
          <cameraName>rgb_camera</cameraName>
```

```
            <imageTopicName>image_raw</imageTopicName>
            <cameraInfoTopicName>camera_info</cameraInfoTopicName>
            <frameName>rgb_camera_link_frame</frameName>
            <hackBaseline>0.07</hackBaseline>
          </plugin>
        </sensor>
    </gazebo>
```

You will need to create two links and their corresponding joints connected to the `chassis` link:

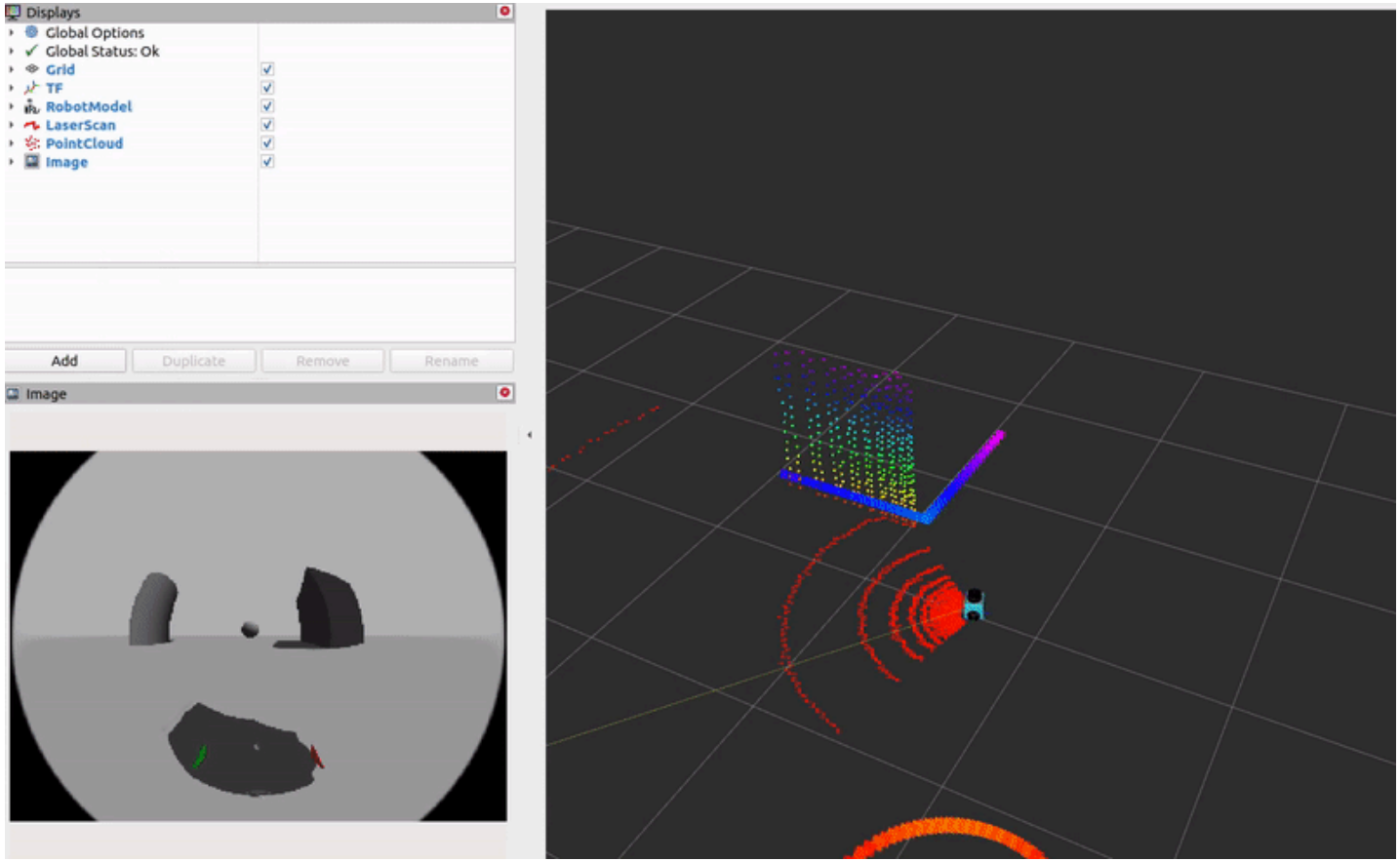
- rgb_camera_link_frame
- pointcloud_link

Remember to change the **QoS** of the sensor topics in RVIZ to their correct configuration. Know that by asking for the information for each topic with the verbose tag:

In [ ]:
```
ros2 topic info /some_topic --verbose
```

- End Notes -

- Expected Behavior for Exercise 6.1.1 -



- End Expected Behavior for Exercise 6.1.1 -

- End Exercise 6.1.1 -

Please try to complete the exercise on your own before checking the solution. You will learn much more from your mistakes.

- Solution for Exercise 6.1.1 -

▶ **Execute in Terminal 1**

In [ ]:

```
cd ~/ros2_ws/src

touch my_box_bot_gazebo/launch/spawn_robot_ros2_final.launch.xml
touch my_box_bot_description/launch/urdf_visualize_final.launch.py
touch my_box_bot_description/urdf/box_bot_final.urdf
```

box_bot_final.urdf

In [ ]:

```
<?xml version="1.0"?>
<robot name="box_bot">

  <material name="red">
      <color rgba="1.0 0.0 0.0 1"/>
  </material>

  <material name="green_light">
      <color rgba="0.0 1.0 0.0 1"/>
  </material>

  <material name="green_dark">
    <color rgba="0.0 0.5 0.0 1"/>
  </material>

  <material name="blue">
      <color rgba="0.0 0.0 1.0 1"/>
  </material>

  <link name="base_link">
  </link>


  <!-- Body -->
  <link name="chassis">
    <visual>
      <geometry>
        <mesh filename="package://my_box_bot_description/meshes/cute_cube.dae" scale="0.1 0.1 0.1"/>
      </geometry>
    </visual>

    <collision>
      <geometry>
        <box size="0.1 0.1 0.1"/>
      </geometry>
    </collision>

    <inertial>
      <mass value="0.5"/>
```

```
    <origin rpy="0 0 0" xyz="0 0 0"/>
    <inertia ixx="0.0008333333333333335" ixy="0" ixz="0" iyy="0.0008333333333333335" iyz="0" izz="0.0008333333333333335"/>
  </inertial>

</link>

<joint name="base_link_joint" type="fixed">
  <origin rpy="0 0 0" xyz="0 0 0" />
  <parent link="base_link" />
  <child link="chassis" />
</joint>

<!-- Wheel Left -->
<link name="left_wheel">
    <visual>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.035"/>
      </geometry>
      <material name="red"/>
    </visual>

    <collision>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.035"/>
      </geometry>
    </collision>

    <inertial>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <mass value="0.05"/>
      <inertia ixx="1.5316666666666667e-05" ixy="0" ixz="0" iyy="1.5316666666666667e-05" iyz="0" izz="3.0625000000000006e-05"/>
    </inertial>

</link>

<gazebo reference="left_wheel">
  <kp>10000000000000000000000000.0</kp>
  <kd>10000000000000000000000000.0</kd>
```

```
  <mu1>10.0</mu1>
  <mu2>10.0</mu2>
  <material>Gazebo/Green</material>
</gazebo>


<joint name="joint_left_wheel" type="continuous">
  <origin rpy="0 0 0" xyz="0 0.05 -0.025"/>
  <child link="left_wheel"/>
  <parent link="chassis"/>
  <axis rpy="0 0 0" xyz="0 1 0"/>
  <limit effort="10000" velocity="1000"/>
  <joint_properties damping="1.0" friction="1.0"/>
</joint>

<!-- Wheel Right -->
<link name="right_wheel">
    <visual>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.035"/>
      </geometry>
      <material name="green"/>
    </visual>

    <collision>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.035"/>
      </geometry>
    </collision>

    <inertial>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <mass value="0.05"/>
      <inertia ixx="1.5316666666666667e-05" ixy="0" ixz="0" iyy="1.5316666666666667e-05" iyz="0" izz="3.0625000000000006e-05"/>
    </inertial>
</link>

<gazebo reference="right_wheel">
```

```
  <kp>10000000000000000000000000.0</kp>
  <kd>10000000000000000000000000.0</kd>
  <mu1>10.0</mu1>
  <mu2>10.0</mu2>
  <material>Gazebo/Orange</material>
</gazebo>

<joint name="joint_right_wheel" type="continuous">
  <origin rpy="0 0 0" xyz="0 -0.05 -0.025"/>
  <child link="right_wheel"/>
  <parent link="chassis"/>
  <axis rpy="0 0 0" xyz="0 1 0"/>
  <limit effort="10000" velocity="1000"/>
  <joint_properties damping="1.0" friction="1.0"/>
</joint>


<!-- Caster Wheel Front -->
<link name="front_yaw_link">
    <visual>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.0045000000000000005"/>
      </geometry>
      <material name="blue"/>
    </visual>

    <collision>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.0045000000000000005"/>
      </geometry>
    </collision>

    <inertial>
        <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
        <mass value="0.001"/>
        <inertia ixx="5.145833333333334e-09" ixy="0" ixz="0" iyy="5.145833333333334e-09" iyz="0" izz="1.0125000000000003e-08"/>
    </inertial>
```

```
</link>

<joint name="front_yaw_joint" type="continuous">
  <origin rpy="0 0 0" xyz="0.04 0 -0.05" />
  <parent link="chassis" />
  <child link="front_yaw_link" />
  <axis xyz="0 0 1" />
  <limit effort="1000.0" velocity="100.0" />
  <dynamics damping="0.0" friction="0.1"/>
</joint>

  <gazebo reference="front_yaw_link">
      <material>Gazebo/Blue</material>
  </gazebo>



<link name="front_roll_link">
    <visual>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.0045000000000000005"/>
      </geometry>
      <material name="red"/>
    </visual>

    <collision>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.0045000000000000005"/>
      </geometry>
    </collision>

    <inertial>
        <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
        <mass value="0.001"/>
        <inertia ixx="5.145833333333334e-09" ixy="0" ixz="0" iyy="5.145833333333334e-09" iyz="0" izz="1.0125000000000003e-08"/>
    </inertial>
</link>
```

```
<joint name="front_roll_joint" type="continuous">
  <origin rpy="0 0 0" xyz="0 0 0" />
  <parent link="front_yaw_link" />
  <child link="front_roll_link" />
  <axis xyz="1 0 0" />
  <limit effort="1000.0" velocity="100.0" />
  <dynamics damping="0.0" friction="0.1"/>
</joint>

  <gazebo reference="front_roll_link">
      <material>Gazebo/Red</material>
  </gazebo>


<link name="front_pitch_link">
  <visual>
    <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
    <geometry>
      <sphere radius="0.010"/>
    </geometry>
    <material name="green_dark"/>
  </visual>

  <collision>
    <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
    <geometry>
      <sphere radius="0.010"/>
    </geometry>
  </collision>

  <inertial>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <mass value="0.001"/>
      <inertia ixx="4e-08" ixy="0" ixz="0" iyy="4e-08" iyz="0" izz="4e-08"/>
  </inertial>
</link>

<gazebo reference="front_pitch_link">
  <kp>10000000000000000000000000.0</kp>
  <kd>10000000000000000000000000.0</kd>
```

```
    <mu1>0.5</mu1>
    <mu2>0.5</mu2>
    <material>Gazebo/Purple</material>
  </gazebo>

  <joint name="front_pitch_joint" type="continuous">
    <origin rpy="0 0 0" xyz="0 0 0" />
    <parent link="front_roll_link" />
    <child link="front_pitch_link" />
    <axis xyz="0 1 0" />
    <limit effort="1000.0" velocity="100.0" />
    <dynamics damping="0.0" friction="0.1"/>
  </joint>

<!-- Caster Wheel Back -->
  <link name="back_yaw_link">
    <visual>
        <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
        <geometry>
          <cylinder length="0.001" radius="0.0045000000000000005"/>
        </geometry>
        <material name="blue"/>
      </visual>

      <collision>
        <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
        <geometry>
          <cylinder length="0.001" radius="0.0045000000000000005"/>
        </geometry>
      </collision>

      <inertial>
          <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
          <mass value="0.001"/>
          <inertia ixx="5.145833333333334e-09" ixy="0" ixz="0" iyy="5.145833333333334e-09" iyz="0" izz="1.0125000000000003e-08"/>
      </inertial>
  </link>

  <joint name="back_yaw_joint" type="continuous">
    <origin rpy="0 0 0" xyz="-0.04 0 -0.05" />
```

```
  <parent link="chassis" />
  <child link="back_yaw_link" />
  <axis xyz="0 0 1" />
  <limit effort="1000.0" velocity="100.0" />
  <dynamics damping="0.0" friction="0.1"/>
</joint>

  <gazebo reference="back_yaw_link">
      <material>Gazebo/Blue</material>
  </gazebo>



<link name="back_roll_link">
    <visual>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.0045000000000000005"/>
      </geometry>
      <material name="red"/>
    </visual>

    <collision>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <geometry>
        <cylinder length="0.001" radius="0.0045000000000000005"/>
      </geometry>
    </collision>

    <inertial>
        <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
        <mass value="0.001"/>
        <inertia ixx="5.145833333333334e-09" ixy="0" ixz="0" iyy="5.145833333333334e-09" iyz="0" izz="1.0125000000000003e-08"/>
    </inertial>
</link>

<joint name="back_roll_joint" type="continuous">
  <origin rpy="0 0 0" xyz="0 0 0" />
  <parent link="back_yaw_link" />
  <child link="back_roll_link" />
```

```
  <axis xyz="1 0 0" />
  <limit effort="1000.0" velocity="100.0" />
  <dynamics damping="0.0" friction="0.1"/>
</joint>

  <gazebo reference="back_roll_link">
      <material>Gazebo/Red</material>
  </gazebo>



<link name="back_pitch_link">
  <visual>
    <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
    <geometry>
      <sphere radius="0.010"/>
    </geometry>
    <material name="green_light"/>
  </visual>

  <collision>
    <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
    <geometry>
      <sphere radius="0.010"/>
    </geometry>
  </collision>

  <inertial>
      <origin rpy="0 1.5707 1.5707" xyz="0 0 0"/>
      <mass value="0.001"/>
      <inertia ixx="4e-08" ixy="0" ixz="0" iyy="4e-08" iyz="0" izz="4e-08"/>
  </inertial>
</link>

<gazebo reference="back_pitch_link">
  <kp>10000000000000000000000000000.0</kp>
  <kd>10000000000000000000000000000.0</kd>
  <mu1>0.5</mu1>
  <mu2>0.5</mu2>
  <material>Gazebo/Yellow</material>
```

```
</gazebo>

<joint name="back_pitch_joint" type="continuous">
  <origin rpy="0 0 0" xyz="0 0 0" />
  <parent link="back_roll_link" />
  <child link="back_pitch_link" />
  <axis xyz="0 1 0" />
  <limit effort="1000.0" velocity="100.0" />
  <dynamics damping="0.0" friction="0.1"/>
</joint>

<!-- PLUGINS -->

<!-- JOINT PUBLISHER -->
<gazebo>
  <plugin name="box_bot_joint_state" filename="libgazebo_ros_joint_state_publisher.so">
    <ros>
        <remapping>~/out:=joint_states</remapping>
    </ros>
    <update_rate>30</update_rate>

    <joint_name>joint_left_wheel</joint_name>
    <joint_name>joint_right_wheel</joint_name>
    <joint_name>front_yaw_joint</joint_name>
    <joint_name>back_yaw_joint</joint_name>
    <joint_name>front_roll_joint</joint_name>
    <joint_name>back_roll_joint</joint_name>
    <joint_name>front_pitch_joint</joint_name>
    <joint_name>back_pitch_joint</joint_name>

  </plugin>
</gazebo>

<!-- Differential drive -->
<gazebo>
  <plugin filename="libgazebo_ros_diff_drive.so" name="differential_drive_controller">

    <!-- wheels -->
    <left_joint>joint_left_wheel</left_joint>
    <right_joint>joint_right_wheel</right_joint>
```

```
    <!-- kinematics -->
    <wheel_separation>0.1</wheel_separation>
    <wheel_diameter>0.07</wheel_diameter>

    <!-- limits -->
    <max_wheel_torque>1.0</max_wheel_torque>
    <max_wheel_acceleration>2.0</max_wheel_acceleration>

    <!-- output -->
    <publish_odom>true</publish_odom>
    <publish_odom_tf>true</publish_odom_tf>

    <odometry_frame>odom</odometry_frame>
    <robot_base_frame>base_link</robot_base_frame>

  </plugin>
</gazebo>


<!-- Laser Position Control-->

<link name="laser_scan_link">
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0"/>
    <geometry>
      <box size="0.02 0.02 0.02"/>
    </geometry>
  </visual>

  <collision>
     <origin rpy="0 0 0" xyz="0 0 0.0204"/>
    <geometry>
      <box size="0.02 0.02 0.02"/>
    </geometry>
  </collision>

  <inertial>

    <mass value="0.01"/>
```

```
    <origin rpy="0 0 0" xyz="0 0 0.0204"/>
    <inertia ixx="6.066578520833334e-06" ixy="0" ixz="0" iyy="6.072950163333333e-06" iyz="0" izz="9.365128684166666e-06"/>
  </inertial>
</link>

<joint name="laser_scan_link_joint" type="prismatic">
  <origin rpy="0 0 0" xyz="0.0 0.0 0.05"/>
  <parent link="chassis"/>
  <child link="laser_scan_link"/>
  <axis xyz="0 0 1"/>
  <limit lower="-0.1" upper="0.0" effort="20.0" velocity="2.0"/>
  <dynamics damping="0.1" friction="1.0"/>
</joint>

<link name="laser_scan_frame">
</link>

<joint name="laser_scan_frame_joint" type="fixed">
  <origin rpy="0 0 0" xyz="0 0 0.03"/>
  <parent link="laser_scan_link"/>
  <child link="laser_scan_frame"/>
  <axis xyz="0 0 0"/>
</joint>

<!-- Visual Laser Model to be rotated -->
<link name="laser_scan_model_link">
  <visual>
    <origin rpy="0 0 0" xyz="0 0 0"/>
    <geometry>
      <mesh filename="package://my_box_bot_description/meshes/sensors/rplidar.dae" scale="1.0 1.0 1.0"/>
    </geometry>
  </visual>

  <collision>
     <origin rpy="0 0 0" xyz="0 0 0.0204"/>
    <geometry>
      <cylinder length="0.0408" radius="0.037493"/>
    </geometry>
  </collision>
```

```
  <inertial>

    <mass value="0.01"/>
    <origin rpy="0 0 0" xyz="0 0 0.0204"/>
    <inertia ixx="6.066578520833334e-06" ixy="0" ixz="0" iyy="6.072950163333333e-06" iyz="0" izz="9.365128684166666e-06"/>
  </inertial>
</link>

<joint name="laser_scan_model_link_joint" type="continuous">
  <origin rpy="0 0 0" xyz="0.0 0.0 0.0"/>
  <parent link="laser_scan_link"/>
  <child link="laser_scan_model_link"/>
  <axis xyz="0 0 1"/>
  <limit effort="10.0" velocity="2.0"/>
  <dynamics friction="0.01"/>
</joint>

<!-- Position Config -->
  <ros2_control name="GazeboSystem" type="system">
    <hardware>
      <plugin>gazebo_ros2_control/GazeboSystem</plugin>
    </hardware>

    <joint name="laser_scan_link_joint">
      <command_interface name="position">
        <param name="min">-0.05</param>
        <param name="max">0.0</param>
      </command_interface>
      <state_interface name="position"/>
      <state_interface name="velocity"/>
      <state_interface name="effort"/>
    </joint>

    <joint name="laser_scan_model_link_joint">
      <command_interface name="velocity">
        <param name="min">0.0</param>
        <param name="max">2.0</param>
      </command_interface>
      <state_interface name="position"/>
      <state_interface name="velocity"/>
```

```
      <state_interface name="effort"/>
    </joint>

</ros2_control>


<gazebo>
  <plugin filename="libgazebo_ros2_control.so" name="gazebo_ros2_control">
    <parameters>$(find my_box_bot_description)/config/controller_position_velocity.yaml</parameters>
    <robot_param_node>/my_robot_state_publisher_node</robot_param_node>
  </plugin>
</gazebo>

<!-- Sensors -->
<gazebo reference="laser_scan_frame">
  <sensor name="sensor_ray" type="ray">
      <pose>0 0 0 0 0 0</pose>
      <ray>
        <scan>
          <horizontal>
            <samples>200</samples>
            <resolution>1.0</resolution>
            <min_angle>-3.14</min_angle>
            <max_angle>3.14</max_angle>
          </horizontal>
        </scan>
        <range>
          <min>0.1</min>
          <max>5.0</max>
        </range>
      </ray>
      <always_on>true</always_on>
      <visualize>true</visualize>
      <update_rate>100.0</update_rate>
      <plugin name="laser" filename="libgazebo_ros_ray_sensor.so">
        <ros>
          <namespace>/box_bot</namespace>
          <remapping>~/out:=laser_scan</remapping>
        </ros>
        <output_type>sensor_msgs/LaserScan</output_type>
```

```
      </plugin>
    </sensor>
      </gazebo>

<link name="pointcloud_link">
</link>

<joint name="pointcloud_link_joint" type="fixed">
  <origin rpy="0 0 0" xyz="0.051 0.0 0"/>
  <parent link="chassis"/>
  <child link="pointcloud_link"/>
  <axis xyz="0 0 0"/>
</joint>

<gazebo reference="pointcloud_link">
    <sensor type="ray" name="pointcloud_sensor">
        <ray>
        <scan>
            <horizontal>
                <samples>50</samples>
                <resolution>1.0</resolution>
                <min_angle>-1.0</min_angle>
                <max_angle>1.0</max_angle>
            </horizontal>
            <vertical>
                <samples>50</samples>
                <resolution>1.0</resolution>
                <min_angle>-1.0</min_angle>
                <max_angle>1.0</max_angle>
            </vertical>
        </scan>
        <range>
            <min>0.10</min>
            <max>5.0</max>
            <resolution>0.01</resolution>
        </range>
        <!-- Using gazebo's noise instead of plugin's -->
        <noise>
            <type>gaussian</type>
            <mean>0.0</mean>
```

```
            <stddev>0.01</stddev>
          </noise>
          </ray>
          <!-- Using gazebo's update rate instead of plugin's -->
          <update_rate>30</update_rate>
          <plugin name="gazebo_ros_block_laser_controller" filename="libgazebo_ros_ray_sensor.so">
          <!-- Change namespace and output topic so published topic is /rrbot/laser/pointcloud -->
          <ros>
              <namespace>box_bot</namespace>
              <argument>~/out:=pointcloud</argument>
          </ros>
          <!-- Set output to sensor_msgs/PointCloud to get same output type as gazebo_ros_block_laser -->
          <output_type>sensor_msgs/PointCloud</output_type>
          <frame_name>pointcloud_link</frame_name>

          <!-- min_intensity instead of hokuyoMinIntensity -->
          <min_intensity>100.0</min_intensity>
          </plugin>
      </sensor>
  </gazebo>


  <!-- RGB CAMERA -->
  <link name="rgb_camera_link_frame">
  </link>

  <joint name="rgb_camera_link_frame_joint" type="fixed">
      <origin rpy="0 0 0" xyz="0.051 0.0 0.05"/>
      <parent link="chassis" />
      <child link="rgb_camera_link_frame" />
      <axis xyz="0 0 0"/>
  </joint>

    <gazebo reference="rgb_camera_link_frame">
        <sensor name="camera" type="wideanglecamera">
          <camera>
            <horizontal_fov>6.283</horizontal_fov>
            <image>
              <width>320</width>
              <height>240</height>
```

```
            </image>
            <clip>
              <near>0.1</near>
              <far>100</far>
            </clip>
            <lens>
              <type>custom</type>
              <custom_function>
                <c1>1.05</c1>
                <c2>4</c2>
                <f>1.0</f>
                <fun>tan</fun>
              </custom_function>
              <scale_to_hfov>true</scale_to_hfov>
              <cutoff_angle>3.1415</cutoff_angle>
              <env_texture_size>512</env_texture_size>
            </lens>
            <always_on>1</always_on>
            <update_rate>30</update_rate>
          </camera>
          <plugin name="camera_controller" filename="libgazebo_ros_camera.so">
            <cameraName>rgb_camera</cameraName>
            <imageTopicName>image_raw</imageTopicName>
            <cameraInfoTopicName>camera_info</cameraInfoTopicName>
            <frameName>rgb_camera_link_frame</frameName>
            <hackBaseline>0.07</hackBaseline>
          </plugin>
        </sensor>
    </gazebo>


</robot>
```

spawn_robot_ros2_final.launch.xml

In [ ]:

```
<?xml version='1.0' ?>
<launch>
    <!-- Publish URDF file in robot_description topic -->
    <include file="$(find-pkg-share my_box_bot_description)/launch/urdf_visualize_final.launch.py"/>
    <!-- Read robot_description and spawn in gazebo running sim -->
    <include file="$(find-pkg-share my_box_bot_gazebo)/launch/spawn_robot_description.launch.py"/>
    <!-- Load the controllers -->
    <include file="$(find-pkg-share my_box_bot_gazebo)/launch/control_position_velocity.launch.py"/>
</launch>
```

urdf_visualize_final.launch.py

In [ ]:

```
import os

from ament_index_python.packages import get_package_share_directory
from launch import LaunchDescription
from launch.substitutions import Command
from launch_ros.actions import Node

# this is the function launch  system will look for
def generate_launch_description():

    ####### DATA INPUT ##########
    urdf_file = 'box_bot_final.urdf'
    #xacro_file = "box_bot.xacro"
    package_description = "my_box_bot_description"

    ####### DATA INPUT END ##########
    print("Fetching URDF ==>")
    robot_desc_path = os.path.join(get_package_share_directory(package_description), "urdf", urdf_file)

    # Robot State Publisher

    robot_state_publisher_node = Node(
        package='robot_state_publisher',
        executable='robot_state_publisher',
        name='my_robot_state_publisher_node',
        emulate_tty=True,
        parameters=[{'use_sim_time': True, 'robot_description': Command(['xacro ', robot_desc_path])}],
        output="screen"
    )

    # RVIZ Configuration
    rviz_config_dir = os.path.join(get_package_share_directory(package_description), 'rviz', 'urdf_vis.rviz')


    rviz_node = Node(
            package='rviz2',
            executable='rviz2',
            output='screen',
            name='rviz_node',
```

```
            parameters=[{'use_sim_time': True}],
            arguments=['-d', rviz_config_dir])

    # create and return launch description object
    return LaunchDescription(
        [
            robot_state_publisher_node,
            rviz_node
        ]
    )
```

- If you have more questions, review the Bitbucket repository with a solution for the URDF ROS2 course (https://bitbucket.org/theconstructcore/course_urdf_ros2_solutions/src/update/).
- You can also ask in the forum by clicking the following icon in the bottom taskbar.

▶ Execute in Terminal 1

In [ ]:
```
cd ~/ros2_ws; colcon build; source install/setup.bash
```

In [ ]:
```
ros2 launch my_box_bot_gazebo start_world.launch.py
```

▶ Execute in Terminal 2

In [ ]:
```
cd ~/ros2_ws; source install/setup.bash
```

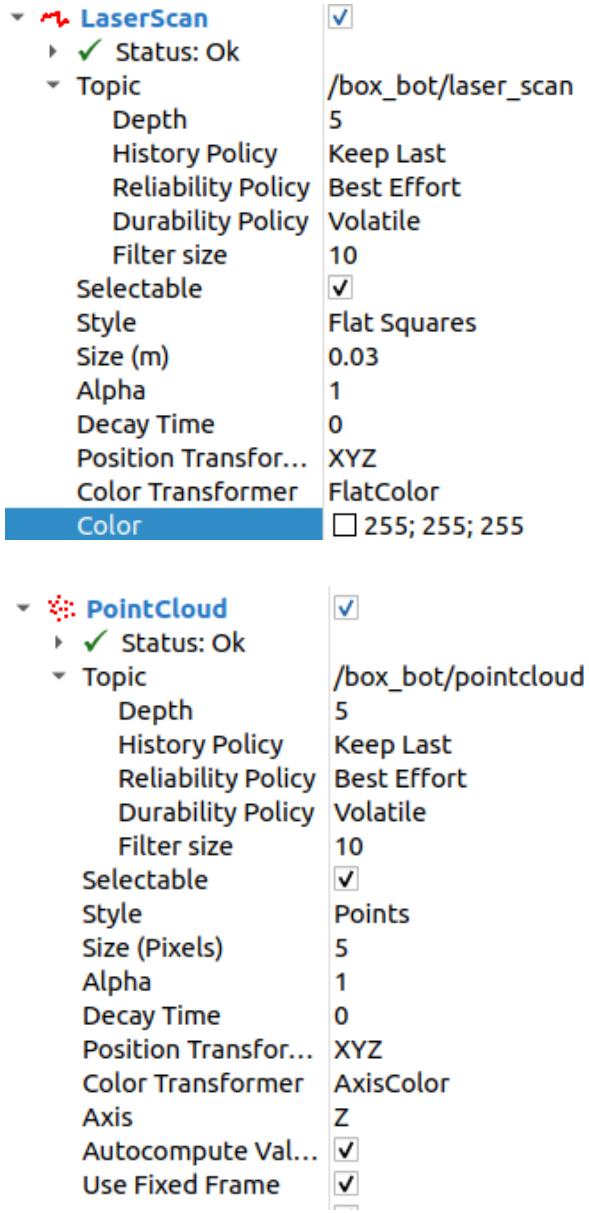
In [ ]:
```
ros2 launch my_box_bot_gazebo spawn_robot_ros2_final.launch.xml
```

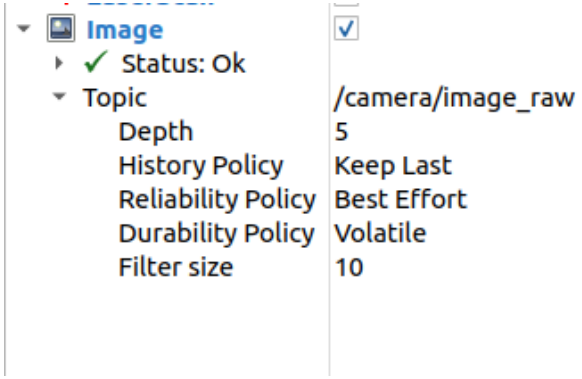
▶ Execute in Terminal 3

In [ ]:
```
cd ~/ros2_ws; source install/setup.bash
```

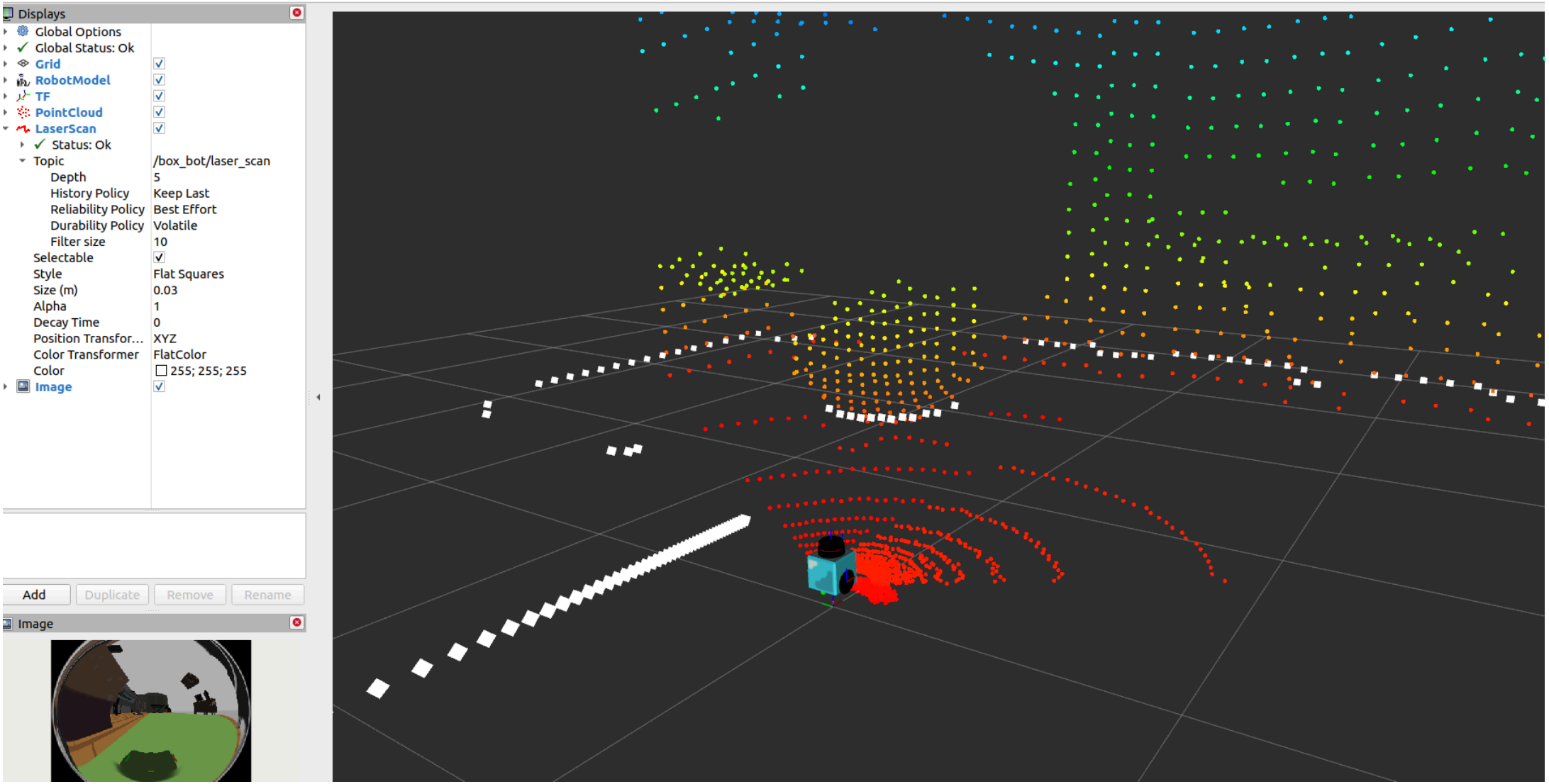
In [ ]:
```
ros2 run teleop_twist_keyboard teleop_twist_keyboard
```

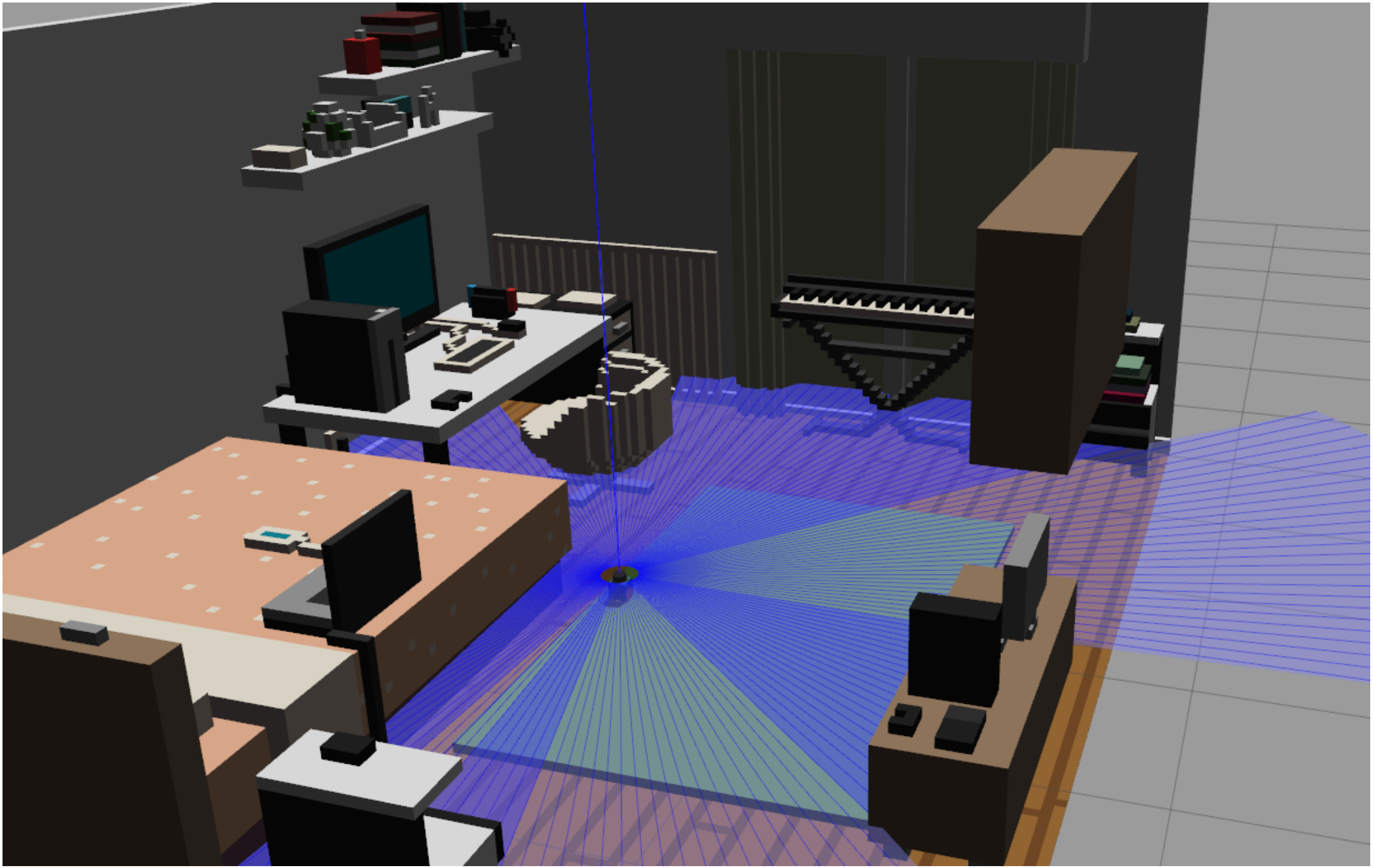
Topics **QoS** setup in RVIZ2:





As an extra, try creating a new world that has the model `box_room` in it, spawning the box bot in this world. That model was copied from the same Git repository where you got the meshes for box_bot and RPLIDAR.


Displays
Global Options
Global Status: Ok
Grid
RobotModel
TF
PointCloud
LaserScan
Status: Ok
Topic /box_bot/laser_scan
Depth 5
History Policy Keep Last
Reliability Policy Best Effort
Durability Policy Volatile
Filter size 10
Selectable
Style Flat Squares
Size (m) 0.03
Alpha 1
Decay Time 0
Position Transfor… XYZ
Color Transformer FlatColor
Color 255; 255; 255
Image
Add
Duplicate
Remove
Rename
Image



room.world

In [ ]:

```xml
<?xml version="1.0" ?>

<sdf version="1.6">
  <world name="default">

    <include>
      <uri>model://sun</uri>
    </include>
    <include>
      <uri>model://ground_plane</uri>
    </include>
    <include>
      <uri>model://box_room</uri>
      <pose>0 0 -0.095708 0 0 0</pose>
    </include>

  </world>
</sdf>
```

main.launch.xml

In [ ]:

```xml
<?xml version='1.0' ?>
<launch>
    <include file="$(find-pkg-share my_box_bot_gazebo)/launch/start_world_box_room.launch.py">
    </include>
    <include file="$(find-pkg-share my_box_bot_gazebo)/launch/spawn_robot_ros2_final.launch.xml"/>
</launch>
```

start_world_box_room.launch.py

In [ ]:

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
import os

from ament_index_python.packages import get_package_share_directory
from launch import LaunchDescription
from launch.actions import DeclareLaunchArgument
from launch.actions import IncludeLaunchDescription
from launch.launch_description_sources import PythonLaunchDescriptionSource
from ament_index_python.packages import get_package_prefix

def generate_launch_description():

    pkg_gazebo_ros = get_package_share_directory('gazebo_ros')
    pkg_box_bot_gazebo = get_package_share_directory('my_box_bot_gazebo')

    # We get the whole install dir
    # We do this to avoid having to copy or softlink manually the packages so that gazebo can find them
    description_package_name = "my_box_bot_description"
    install_dir = get_package_prefix(description_package_name)

    # Set the path to the WORLD model files. Is to find the models inside the models folder in my_box_bot_gazebo package
    gazebo_models_path = os.path.join(pkg_box_bot_gazebo, 'models')
    # os.environ["GAZEBO_MODEL_PATH"] = gazebo_models_path

    if 'GAZEBO_MODEL_PATH' in os.environ:
        os.environ['GAZEBO_MODEL_PATH'] =  os.environ['GAZEBO_MODEL_PATH'] + ':' + install_dir + '/share' + ':' + gazebo_models_pat
    else:
        os.environ['GAZEBO_MODEL_PATH'] =  install_dir + "/share" + ':' + gazebo_models_path

    if 'GAZEBO_PLUGIN_PATH' in os.environ:
        os.environ['GAZEBO_PLUGIN_PATH'] = os.environ['GAZEBO_PLUGIN_PATH'] + ':' + install_dir + '/lib'
    else:
        os.environ['GAZEBO_PLUGIN_PATH'] = install_dir + '/lib'



    print("GAZEBO MODELS PATH=="+str(os.environ["GAZEBO_MODEL_PATH"]))
    print("GAZEBO PLUGINS PATH=="+str(os.environ["GAZEBO_PLUGIN_PATH"]))
```

```
    # Gazebo launch
    gazebo = IncludeLaunchDescription(
        PythonLaunchDescriptionSource(
            os.path.join(pkg_gazebo_ros, 'launch', 'gazebo.launch.py'),
        )
    )

    return LaunchDescription([
        DeclareLaunchArgument(
          'world',
          default_value=[os.path.join(pkg_box_bot_gazebo, 'worlds', 'room.world'), ''],
          description='SDF world file'),
        gazebo
    ])
```

- End Solution for Exercise 6.1.1 -

16/11/2023