

Chapter 3 Working with ROS for 3D Modeling

Nama : Al Ghifary Akmal Nasheeri

NIM : 1103201242

Kelas : TK-44-06

Pada tahap awal pembuatan robot, terlibat dalam proses desain dan pemodelan. Desain dan pemodelan robot dapat dilakukan menggunakan alat CAD seperti Autodesk Fusion 360, SolidWorks, Blender, dan lainnya. Salah satu tujuan utama dari pemodelan robot adalah simulasi.

Alat simulasi robot dapat memeriksa cacat kritis dalam desain robot dan memastikan bahwa robot akan berfungsi sebelum memasuki fase manufaktur. Dalam bab ini, kita akan membahas proses desain dua robot, yaitu manipulator dengan tujuh Derajat Kebebasan (DOF) dan robot penggerak roda differential.

Di bab-bab berikutnya, kita akan melihat simulasi, mempelajari cara membangun perangkat keras nyata, dan membahas antarmuka dengan ROS. Jika Anda berencana membuat model 3D robot dan mensimulasikannya menggunakan ROS, Anda perlu mempelajari beberapa package ROS yang dapat membantu dalam desain robot.

Menciptakan model untuk robot dalam ROS penting karena berbagai alasan. Misalnya, model ini dapat digunakan untuk mensimulasikan dan mengendalikan robot, memvisualisasikannya, atau menggunakan alat ROS untuk mendapatkan informasi tentang struktur kinematika robot.

ROS menyediakan beberapa package untuk merancang dan membuat model robot, seperti `urdf`, `kdl_parser`, `robot_state_publisher`, dan `collada_urdf`. Package-package ini akan membantu kita membuat deskripsi model robot 3D dengan karakteristik yang tepat dari perangkat keras nyata.

ROS packages for robot modelling

ROS menyediakan beberapa package yang sangat berguna untuk membangun model robot 3D. Dalam bagian ini, kita akan membahas beberapa package ROS penting yang umum digunakan untuk membangun dan memodelkan robot:

- **urdf**: Package ROS paling penting untuk memodelkan robot adalah package `urdf`. Package ini berisi parser C++ untuk URDF, yang merupakan file XML yang mewakili model robot. Beberapa komponen berbeda membentuk `urdf`, seperti `urdf_parser_plugin`, `urdfdom_headers`, `collada_parser`, dan `urdfdom`.
- **joint_state_publisher**: Alat ini sangat berguna saat merancang model robot menggunakan URDF. Package ini berisi sebuah node bernama `joint_state_publisher`, yang membaca deskripsi model robot, menemukan semua sendi, dan menerbitkan nilai sendi ke semua sendi yang tidak tetap.

- **joint_state_publisher_gui**: Alat ini mirip dengan package `joint_state_publisher`. Ini menawarkan fungsionalitas yang sama dan, selain itu, mengimplementasikan serangkaian slider yang dapat digunakan oleh pengguna untuk berinteraksi dengan setiap sendi robot dan memvisualisasikan output menggunakan RViz.
- **kdl_parser**: Package ini berisi alat parser untuk membangun pohon Kinematika dan Dinamika (KDL) dari model URDF robot. KDL adalah perpustakaan yang digunakan untuk menyelesaikan masalah kinematika dan dinamika.
- **robot_state_publisher**: Package ini membaca status sendi robot saat ini dan menerbitkan posisi 3D dari setiap tautan robot menggunakan pohon kinematika yang dibangun dari URDF.
- **xacro**: Xacro adalah singkatan dari XML Macros, dan kita dapat menganggap file xacro sebagai file URDF dengan beberapa tambahan. Ini berisi beberapa tambahan untuk membuat URDF lebih pendek dan lebih mudah dibaca dan dapat digunakan untuk membangun deskripsi robot yang kompleks. Kita dapat mengonversi xacro menjadi URDF kapan saja menggunakan alat ROS.

Understanding robot modeling using URDF

Dalam bagian sebelumnya, kita telah mencantumkan beberapa package penting yang menggunakan format file urdf. Pada bagian ini, kita akan melihat lebih jauh tentang tag XML URDF, yang membantu dalam memodelkan robot. Kita perlu membuat file dan menuliskan hubungan antara setiap tautan dan sendi dalam robot serta menyimpan file dengan ekstensi .urdf.

URDF dapat merepresentasikan deskripsi kinematika dan dinamika robot, representasi visual robot, dan model tumbukan robot. Tag-tag URDF yang umum digunakan untuk menyusun model robot URDF adalah:

- **link**: Tag ini merepresentasikan satu tautan robot. Dengan tag ini, kita dapat memodelkan tautan robot beserta sifatnya, seperti ukuran, bentuk, dan warna; bahkan dapat mengimpor mesh 3D untuk merepresentasikan tautan robot. Tag ini juga dapat menyediakan properti dinamis tautan, seperti matriks inersia dan properti tumbukan.
- **joint**: Tag ini merepresentasikan sendi robot. Kita dapat menentukan kinematika dan dinamika sendi serta mengatur batasan pergerakan dan kecepatannya. Tag ini mendukung berbagai jenis sendi, seperti revolute, continuous, prismatic, fixed, floating, dan planar.
- **robot**: Tag ini menggabungkan seluruh model robot yang dapat direpresentasikan menggunakan URDF. Di dalam tag robot, kita dapat mendefinisikan nama robot, tautan, dan sendi robot.
- **gazebo**: Tag ini digunakan ketika kita menyertakan parameter simulasi simulator Gazebo di dalam URDF. Kita dapat menggunakan tag ini untuk menyertakan plugin gazebo, properti material gazebo, dan lainnya.

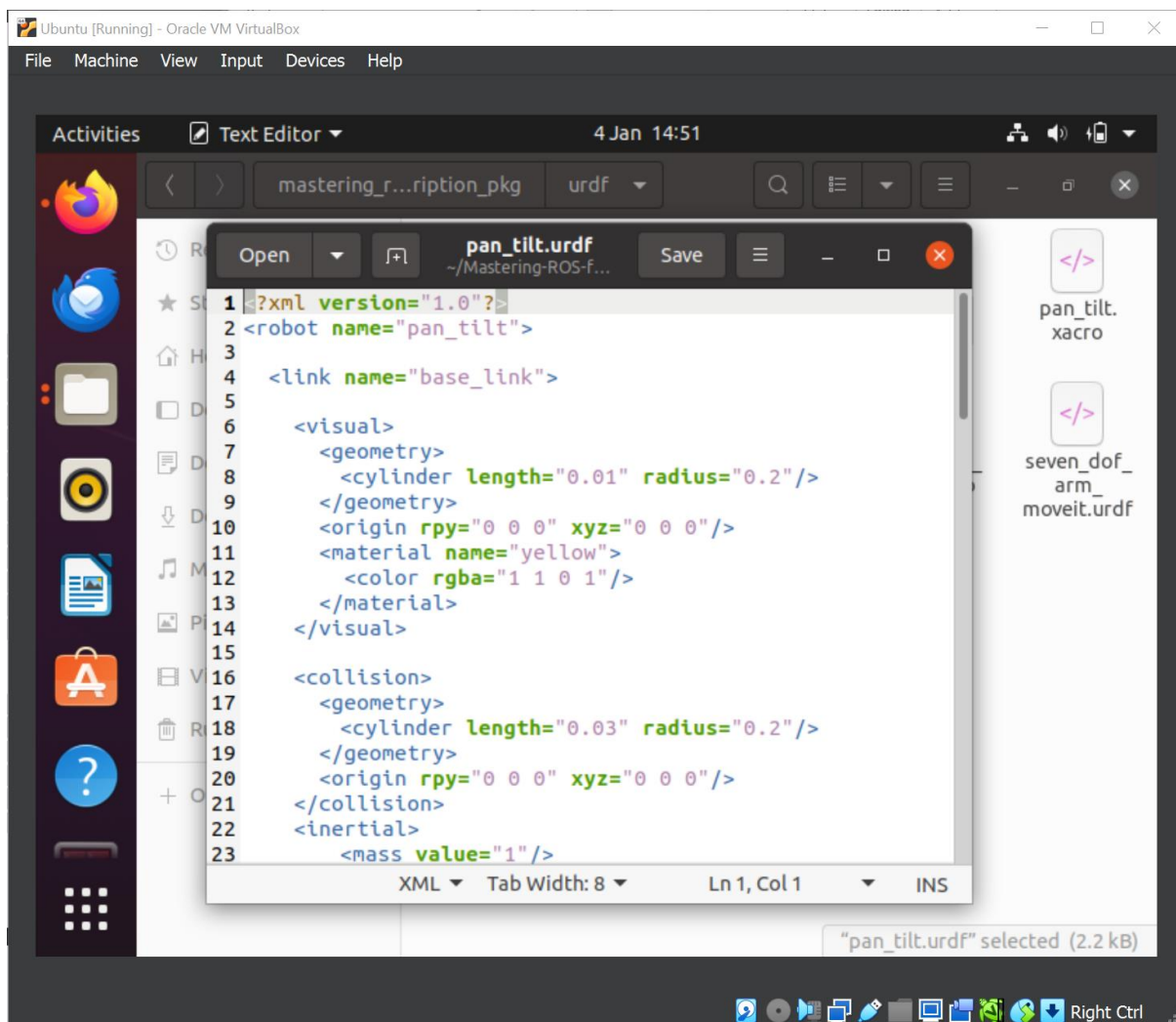
Creating the ROS package for the robot description

Sebelum membuat file URDF untuk robot, kita perlu membuat package ROS di dalam workspace Catkin. Ini dapat dilakukan dengan perintah **catkin_create_pkg** dengan dependensi pada package-package seperti **roscpp**, **tf**, **geometry_msgs**, **urdf**, **rviz**, dan **xacro**. Package ini terutama bergantung pada package **urdf** dan **xacro**, yang dapat diinstal menggunakan manajer package.

Creating our first URDF model

Setelah memahami tag-tag URDF, kita dapat mulai membuat model dasar menggunakan URDF. Robot pertama yang akan kita rancang adalah mekanisme pan-and-tilt dengan tiga tautan dan dua sendi revolute. Kita dapat melihat dan menjelaskan kode URDF untuk mekanisme ini. File URDF disimpan dalam folder **urdf**, dengan tambahan folder **meshes** dan **launch** untuk menyimpan file mesh dan file peluncuran ROS.

pan_tilt.urdf



The screenshot shows a text editor window titled "pan_tilt.urdf" with the following XML content:

```
1 <?xml version="1.0"?>
2 <robot name="pan_tilt">
3
4   <link name="base_link">
5
6     <visual>
7       <geometry>
8         <cylinder length="0.01" radius="0.2"/>
9       </geometry>
10      <origin rpy="0 0 0" xyz="0 0 0"/>
11      <material name="yellow">
12        <color rgba="1 1 0 1"/>
13      </material>
14    </visual>
15
16    <collision>
17      <geometry>
18        <cylinder length="0.03" radius="0.2"/>
19      </geometry>
20      <origin rpy="0 0 0" xyz="0 0 0"/>
21    </collision>
22    <inertial>
23      <mass value="1"/>
```

The editor interface includes a sidebar with file explorer icons, a top menu bar with "File", "Machine", "View", "Input", "Devices", and "Help", and a status bar at the bottom indicating "Ln 1, Col 1" and "INS". A notification at the bottom right states "pan_tilt.urdf selected (2.2 kB)".

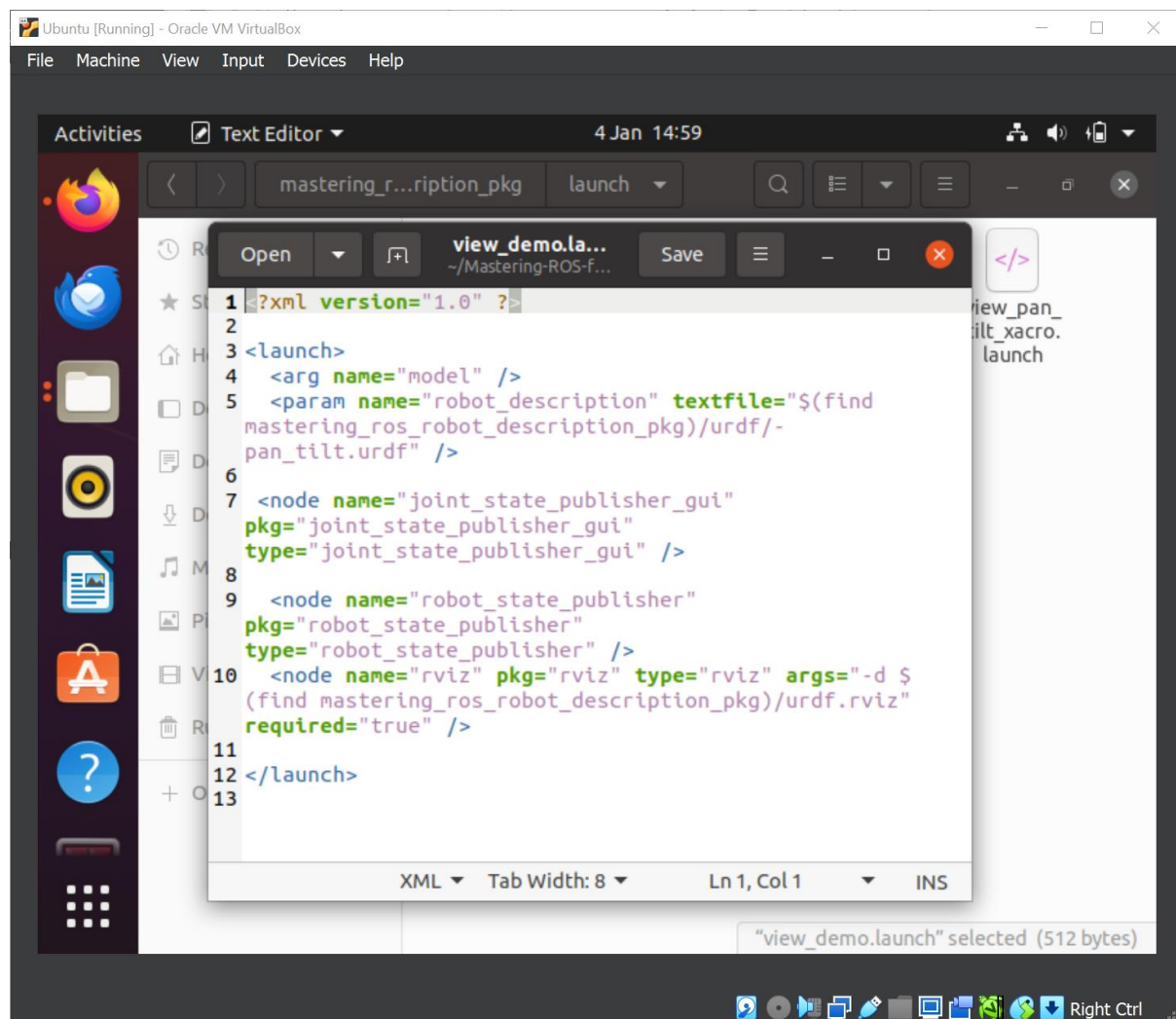
Explaining the URDF file

File URDF memiliki struktur yang terdiri dari tag-tag XML seperti `<link>`, `<joint>`, dan `<robot>`. Setiap tag mendefinisikan tautan, sendi, atau model robot secara keseluruhan. Contoh mekanisme pan-and-tilt dijelaskan dengan menggunakan tag `<link>` untuk mendefinisikan properti visual dan dinamis tautan, dan tag `<joint>` untuk mendefinisikan tautan antara dua link. Pengecekan file URDF dilakukan dengan menggunakan perintah `check_urdf`, dan visualisasi grafik menggunakan perintah `urdf_to_graphviz`. Visualisasi 3D dari model di RViz membantu pemahaman posisi dan hubungan setiap sendi robot.

Visualizing the 3D robot model in Rviz

Setelah merancang URDF, kita dapat melihatnya di RViz. Kita dapat membuat file peluncuran `view_demo.launch` dan menempatkannya di folder `launch`. File ini berisi kode XML yang menyertakan parameter deskripsi robot, node pengelolaan status sendi, dan node pengelolaan status robot. Jika semuanya berfungsi dengan benar, kita dapat meluncurkan model menggunakan perintah `roslaunch` dan melihat mekanisme pan-and-tilt di RViz.

view_demo.launch



The screenshot shows a text editor window titled "view_demo.la..." with the following XML code:

```
1 <?xml version="1.0" ?>
2
3 <launch>
4   <arg name="model" />
5   <param name="robot_description" textfile="$(find
mastering_ros_robot_description_pkg)/urdf/-
pan_tilt.urdf" />
6
7   <node name="joint_state_publisher_gui"
pkg="joint_state_publisher_gui"
type="joint_state_publisher_gui" />
8
9   <node name="robot_state_publisher"
pkg="robot_state_publisher"
type="robot_state_publisher" />
10  <node name="rviz" pkg="rviz" type="rviz" args="-d $(
(find mastering_ros_robot_description_pkg)/urdf.rviz"
required="true" />
11
12 </launch>
13
```

The status bar at the bottom indicates "view_demo.launch" selected (512 bytes).

Interacting with pan-and-tilt joints

RViz dilengkapi dengan GUI yang memungkinkan pengguna mengontrol sendi pan dan tilt menggunakan slider. Node `joint_state_publisher_gui` digunakan untuk memasukkan GUI ini ke dalam file peluncuran. Batasan dari setiap sendi juga dapat diatur dalam tag `<limit>` di dalam tag `<joint>`.

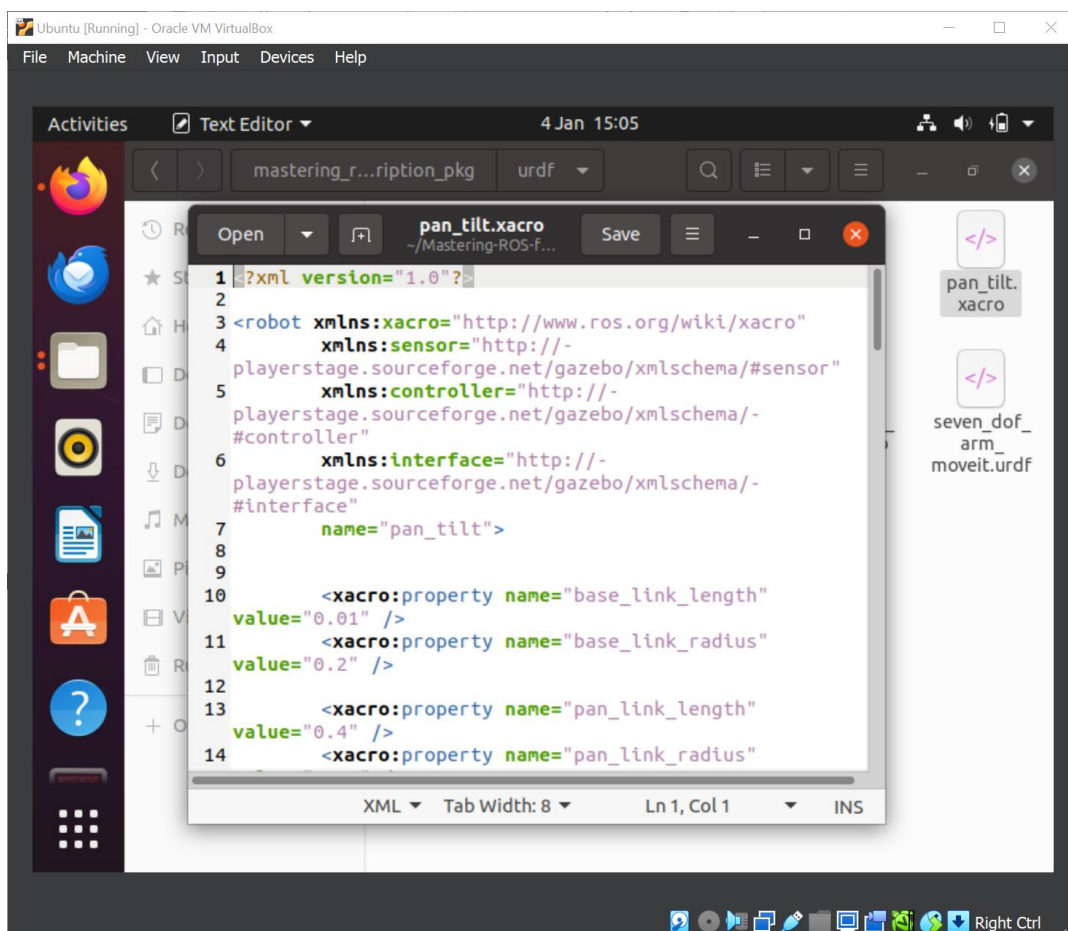
Adding physical and collision properties to a URDF model

Sebelum mensimulasikan robot di simulator seperti Gazebo atau CoppeliaSim, properti fisik dan collision dari setiap tautan robot perlu ditentukan dalam file URDF. Informasi seperti geometri, warna, massa, dan inersia didefinisikan dalam tag `<collision>` dan `<inertial>`.

Understanding robot modeling using xacro

URDF memiliki keterbatasan dalam hal kejelasan, kegunaan kembali, modularitas, dan pemrograman. Pemodelan menggunakan xacro memenuhi kebutuhan ini dengan menyederhanakan URDF, memberikan dukungan pemrograman, mendeklarasikan properti, menggunakan ekspresi matematika, dan menggunakan makro untuk mengurangi panjang kode. File xacro dapat dikonversi menjadi URDF untuk digunakan dalam sistem ROS.

pan_tilt.xacro



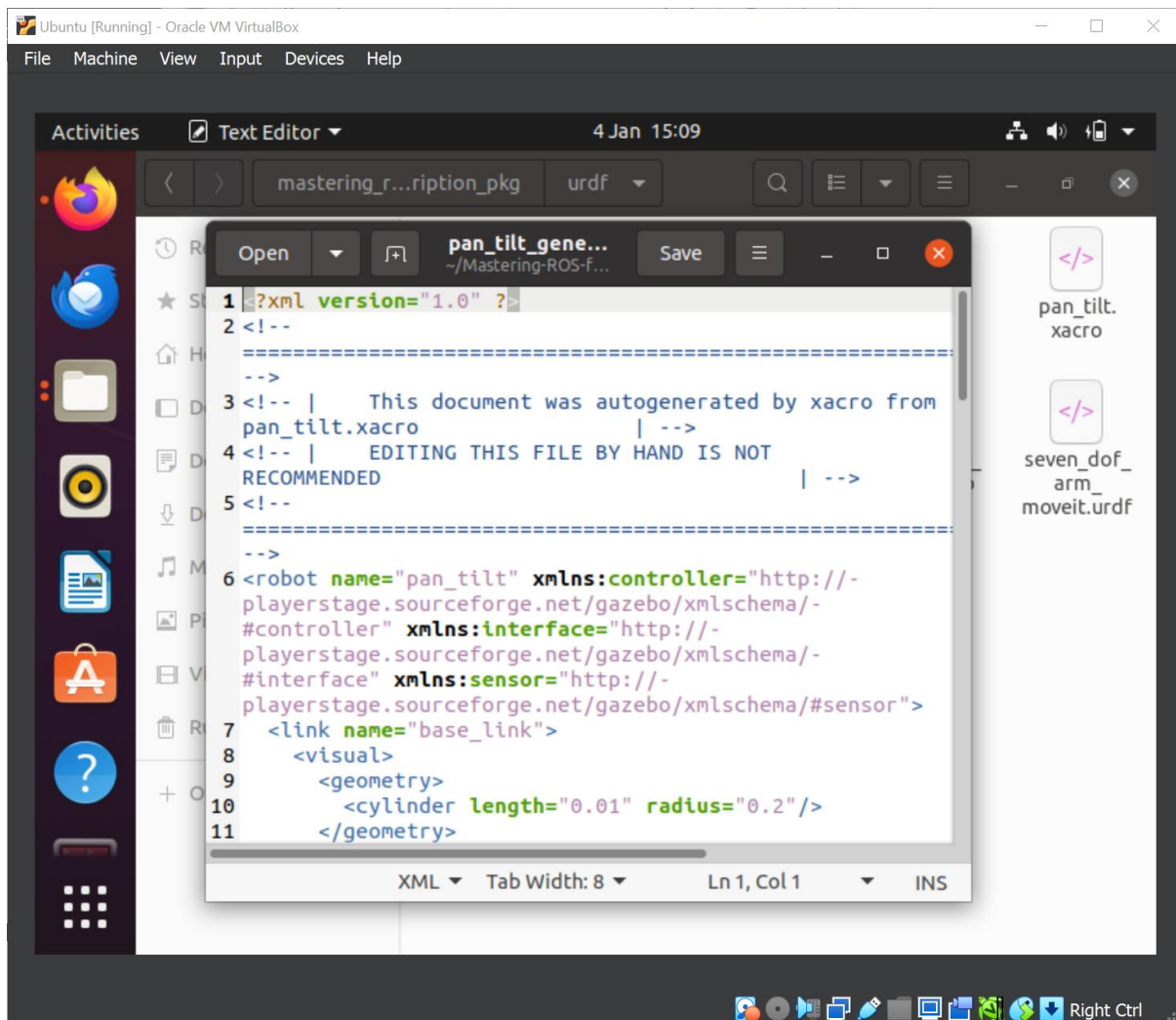
```
1 <?xml version="1.0"?>
2
3 <robot xmlns:xacro="http://www.ros.org/wiki/xacro"
4       xmlns:sensor="http://-
5       playerstage.sourceforge.net/gazebo/xmlschema/#sensor"
6       xmlns:controller="http://-
7       playerstage.sourceforge.net/gazebo/xmlschema/-
8       #controller"
9       xmlns:interface="http://-
10      playerstage.sourceforge.net/gazebo/xmlschema/-
11      #interface"
12      name="pan_tilt">
13
14      <xacro:property name="base_link_length"
15      value="0.01" />
16      <xacro:property name="base_link_radius"
17      value="0.2" />
18
19      <xacro:property name="pan_link_length"
20      value="0.4" />
21      <xacro:property name="pan_link_radius"
```

Converting xacro to URDF

File xacro dapat dikonversi menjadi file URDF setiap saat. Setelah merancang file xacro, kita dapat menggunakan perintah **roswin xacro** untuk mengonversinya menjadi file URDF. Perintah ini digunakan untuk file **pan_tilt.xacro**, menghasilkan **pan_tilt_generated.urdf**. Dalam file peluncuran ROS, kita dapat menggunakan perintah **<param>** untuk mengonversi xacro ke URDF dan menggunakan parameter **robot_description**.

Kita dapat melihat isi file xacro dari robot pan-and-tilt dengan membuat file peluncuran khusus dan menggunakan perintah **roslaunch**. Hasil visualisasi dari file xacro ini seharusnya sama dengan hasil visualisasi dari file URDF.

pan_tilt_generated.urdf



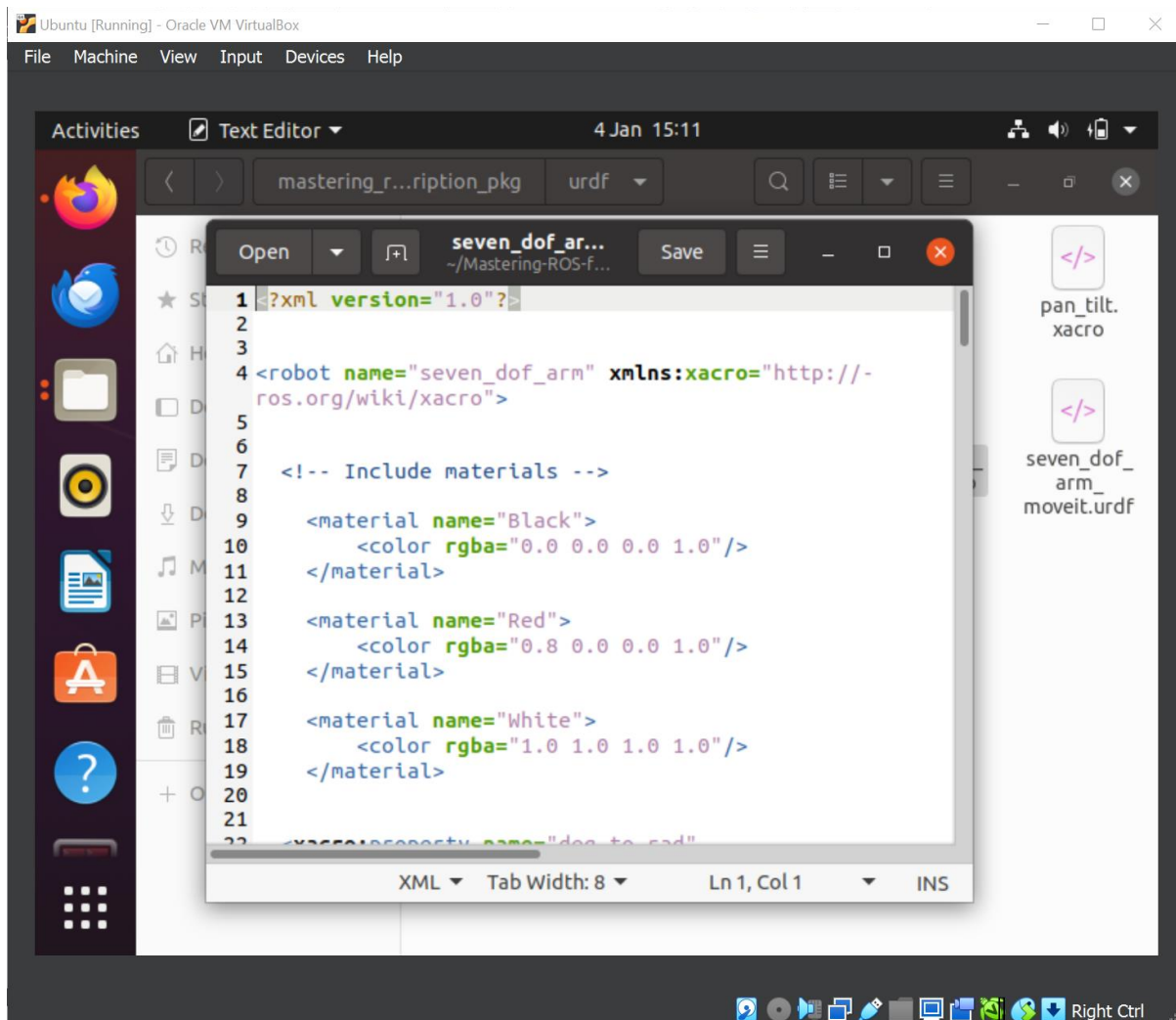
```
1 <?xml version="1.0" ?>
2 <!--
3 =====
4 <!-- | This document was autogenerated by xacro from
5 pan_tilt.xacro | -->
6 <!-- | EDITING THIS FILE BY HAND IS NOT
7 RECOMMENDED | -->
8 <!--
9 =====
10 -->
11 <robot name="pan_tilt" xmlns:controller="http://-
12 playerstage.sourceforge.net/gazebo/xmlschema/-
13 #controller" xmlns:interface="http://-
14 playerstage.sourceforge.net/gazebo/xmlschema/-
15 #interface" xmlns:sensor="http://-
16 playerstage.sourceforge.net/gazebo/xmlschema/#sensor">
17 <link name="base_link">
18 <visual>
19 <geometry>
20 <cylinder length="0.01" radius="0.2"/>
21 </geometry>
```

Creating the robot description for a seven-DOF robot manipulator

Selanjutnya, kita dapat membuat robot yang lebih kompleks menggunakan URDF dan xacro. Robot pertama yang akan dibahas adalah manipulator 7-DOF, yang memiliki tujuh sendi. Manipulator ini memiliki kelebihan sendi dibanding yang diperlukan untuk mencapai posisi

dan orientasi tujuan. Keuntungan manipulator yang redundan adalah fleksibilitas dan keberagaman konfigurasi sendi untuk mencapai posisi dan orientasi tujuan tertentu.

seven_dof_arm.xacro



```
1 <?xml version="1.0"?>
2
3
4 <robot name="seven_dof_arm" xmlns:xacro="http://ros.org/wiki/xacro">
5
6
7 <!-- Include materials -->
8
9 <material name="Black">
10   <color rgba="0.0 0.0 0.0 1.0"/>
11 </material>
12
13 <material name="Red">
14   <color rgba="0.8 0.0 0.0 1.0"/>
15 </material>
16
17 <material name="White">
18   <color rgba="1.0 1.0 1.0 1.0"/>
19 </material>
20
21
22 <xacro:property name="deg_to_rad" value="3.141592653589793/180"/>
```

Arm specification

Manipulator 7-DOF memiliki spesifikasi tertentu, termasuk jumlah derajat kebebasan, panjang lengan, jangkauan, jumlah tautan, dan jumlah sendi. Terdapat berbagai jenis sendi, seperti sendi revolute dan prismatic, yang digunakan dalam deskripsi manipulator ini. Deskripsi manipulator ini ditulis dalam file xacro, yang akan dijelaskan pada bagian selanjutnya.

Explaining the xacro model of the seven-DOF arm

Setelah mendefinisikan elemen-elemen yang harus dimasukkan dalam file model robot, kita siap menyertakan 10 tautan dan 9 sendi (7 untuk lengan dan 2 untuk gripper) pada robot ini, serta 2 tautan dan 2 sendi pada gripper robot.

Using constants

Konstanta digunakan dalam file xacro untuk membuat deskripsi robot lebih pendek dan mudah dibaca. Konstanta seperti faktor konversi derajat ke radian, nilai PI, panjang, lebar, dan tinggi setiap tautan didefinisikan di sini.

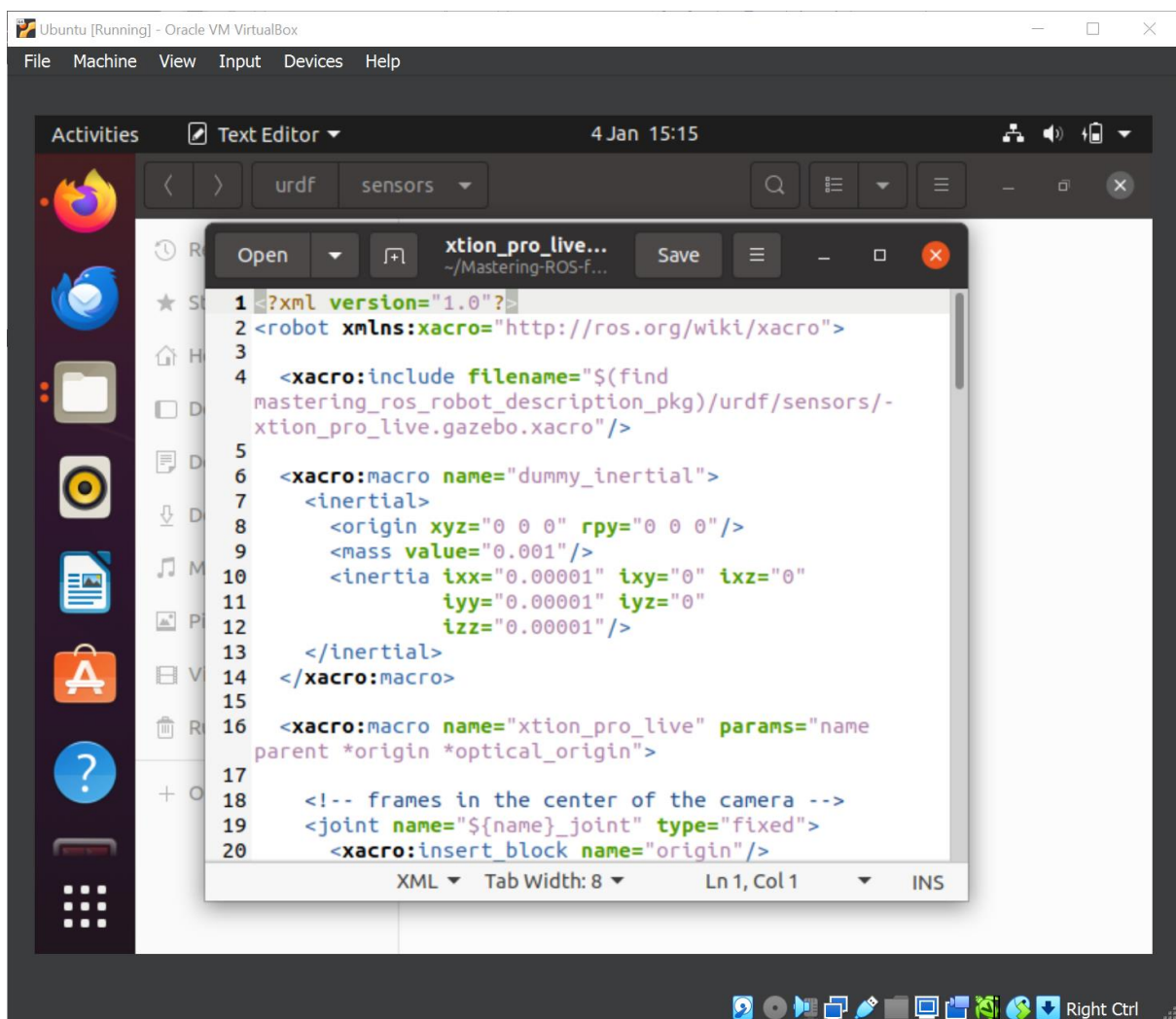
Using macro

Macro digunakan untuk menghindari repetisi dan membuat kode menjadi lebih singkat. Dua macronya adalah "inertial_matrix" untuk matriks inersia dan "transmission_block" untuk blok transmisi. Ini membantu menggambarkan elemen yang sama berkali-kali dengan lebih singkat.

Including other xacro files

Kemampuan xacro robot dapat diperluas dengan menyertakan definisi xacro sensor. Contoh penggunaan tag `<xacro:include>` untuk menyertakan definisi sensor visi.

xtion_pro_live.urdf.xacro



```
1 <?xml version="1.0"?>
2 <robot xmlns:xacro="http://ros.org/wiki/xacro">
3
4   <xacro:include filename="$(find
mastering_ros_robot_description_pkg)/urdf/sensors/-
xtion_pro_live.gazebo.xacro"/>
5
6   <xacro:macro name="dummy_inertial">
7     <inertial>
8       <origin xyz="0 0 0" rpy="0 0 0"/>
9       <mass value="0.001"/>
10      <inertia ixx="0.00001" ixy="0" ixz="0"
11              iyy="0.00001" iyz="0"
12              izz="0.00001"/>
13    </inertial>
14  </xacro:macro>
15
16  <xacro:macro name="xtion_pro_live" params="name
parent *origin *optical_origin">
17
18    <!-- frames in the center of the camera -->
19    <joint name="${name}_joint" type="fixed">
20      <xacro:insert_block name="origin"/>
```


Using meshes in the link

Dalam deskripsi link, kita dapat menyisipkan bentuk primitif atau file mesh menggunakan tag `<mesh>`. Dengan contoh penggunaan mesh pada sensor vision, tampilan visual robot menjadi lebih realistis dan akurat. Hal ini mempermudah simulasi robot dalam lingkungan 3D.

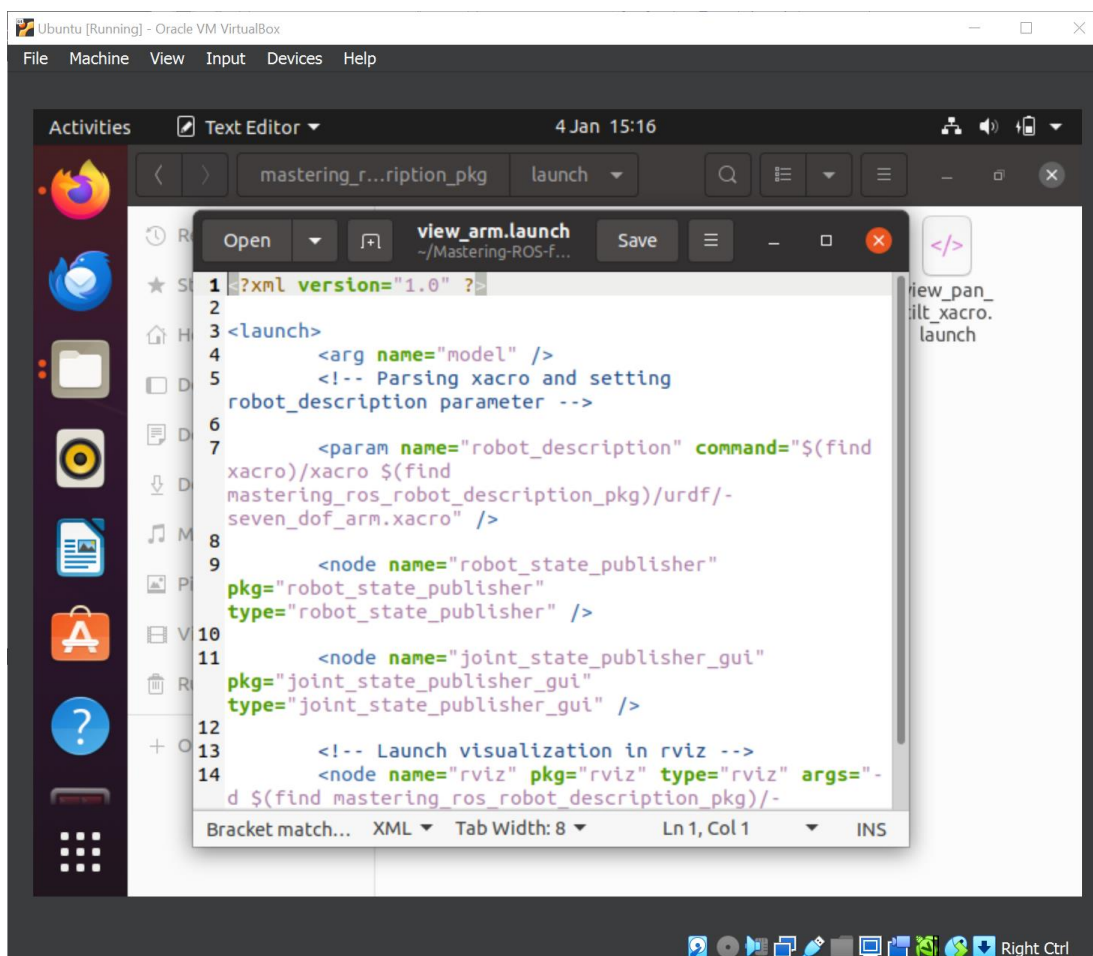
Working with the robot gripper

Gripper robot dirancang untuk mengambil dan meletakkan blok, memiliki dua sendi prismatic pada gripper. Definisi sendi mencakup parameter seperti batasan, kontrol keselamatan, dan karakteristik dinamika. Konfigurasi sendi-sendi ini memberikan fleksibilitas dalam mengendalikan gripper sesuai dengan kebutuhan aplikasi.

Viewing the seven-DOF arm in Rviz

Setelah mendefinisikan semua elemen, termasuk konstanta, makro, dan penggunaan mesh atau sensor vision, model lengan robot 7-DOF ini siap untuk divisualisasikan menggunakan RViz. RViz memungkinkan untuk memeriksa dan memvalidasi model robot secara grafis, memastikan bahwa elemen-elemen yang didefinisikan berfungsi dengan baik dalam representasi visual 3D.

view_arm.launch



```
1 <?xml version="1.0" ?>
2
3 <launch>
4   <arg name="model" />
5   <!-- Parsing xacro and setting
robot_description parameter -->
6
7   <param name="robot_description" command="$(find
xacro)/xacro $(find
mastering_ros_robot_description_pkg)/urdf/-
seven_dof_arm.xacro" />
8
9   <node name="robot_state_publisher"
pkg="robot_state_publisher"
type="robot_state_publisher" />
10
11  <node name="joint_state_publisher_gui"
pkg="joint_state_publisher_gui"
type="joint_state_publisher_gui" />
12
13  <!-- Launch visualization in rviz -->
14  <node name="rviz" pkg="rviz" type="rviz" args="-
d $(find mastering_ros_robot_description_pkg)/-
```

Understanding joint state publisher

Package joint state publisher adalah salah satu package ROS yang umumnya digunakan untuk berinteraksi dengan setiap sendi robot. Package ini mencakup node joint_state_publisher yang mengidentifikasi sendi yang tidak tetap dari model URDF dan menerbitkan nilai keadaan sendi setiap sendi dalam format pesan sensor_msgs/JointState. Package ini dapat digunakan bersama dengan package robot_state_publisher untuk menerbitkan posisi semua sendi. Berbagai sumber dapat digunakan untuk mengatur nilai setiap sendi, termasuk penggunaan GUI slider untuk pengujian atau menggunakan topik JointState yang di-subscribe oleh node.

Understanding robot state publisher

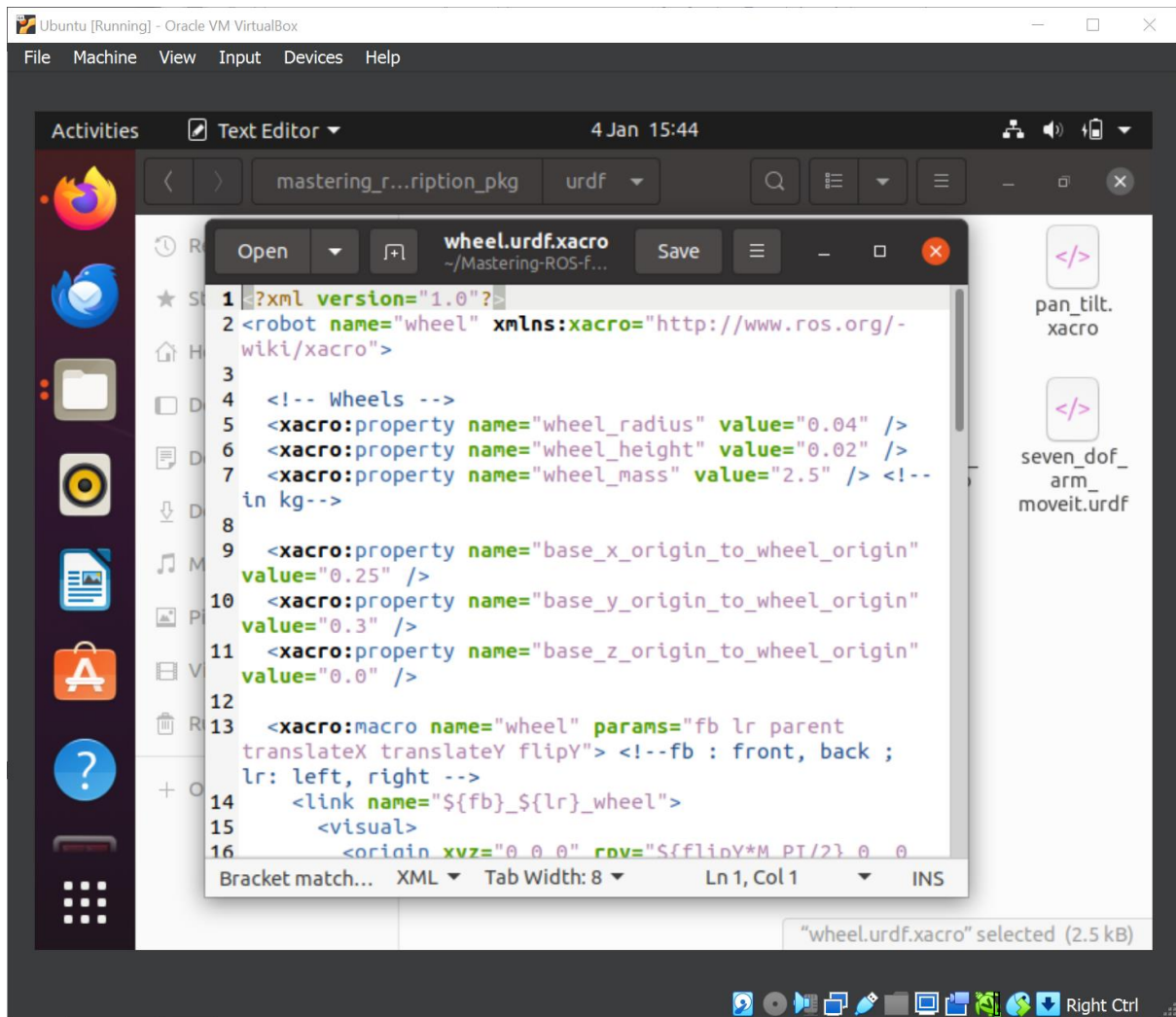
Package robot state publisher membantu dalam menerbitkan keadaan robot ke tf (transform frame). Package ini berlangganan pada keadaan sendi robot dan menerbitkan pose 3D dari setiap tautan menggunakan representasi kinematika dari model URDF. Node robot state publisher dapat diimplementasikan dalam file peluncuran dengan menambahkan baris yang sesuai. Saat package ini dijalankan, kita dapat memvisualisasikan transformasi robot dengan memilih opsi tf di RViz. Joint state publisher dan robot state publisher umumnya sudah terpasang bersama instalasi ROS desktop.

Creating a robot model for the differential drive mobile robot

Robot roda differential drive memiliki dua roda yang terhubung ke sisi berlawanan dari sasis robot, didukung oleh satu atau dua roda caster. Roda mengendalikan kecepatan robot dengan mengatur kecepatan roda tunggal. Jika dua motor berjalan dengan kecepatan yang sama, roda akan bergerak maju atau mundur. Jika satu roda berjalan lebih lambat dari yang lain, robot akan berbelok ke sisi roda yang berkecepatan lebih rendah.

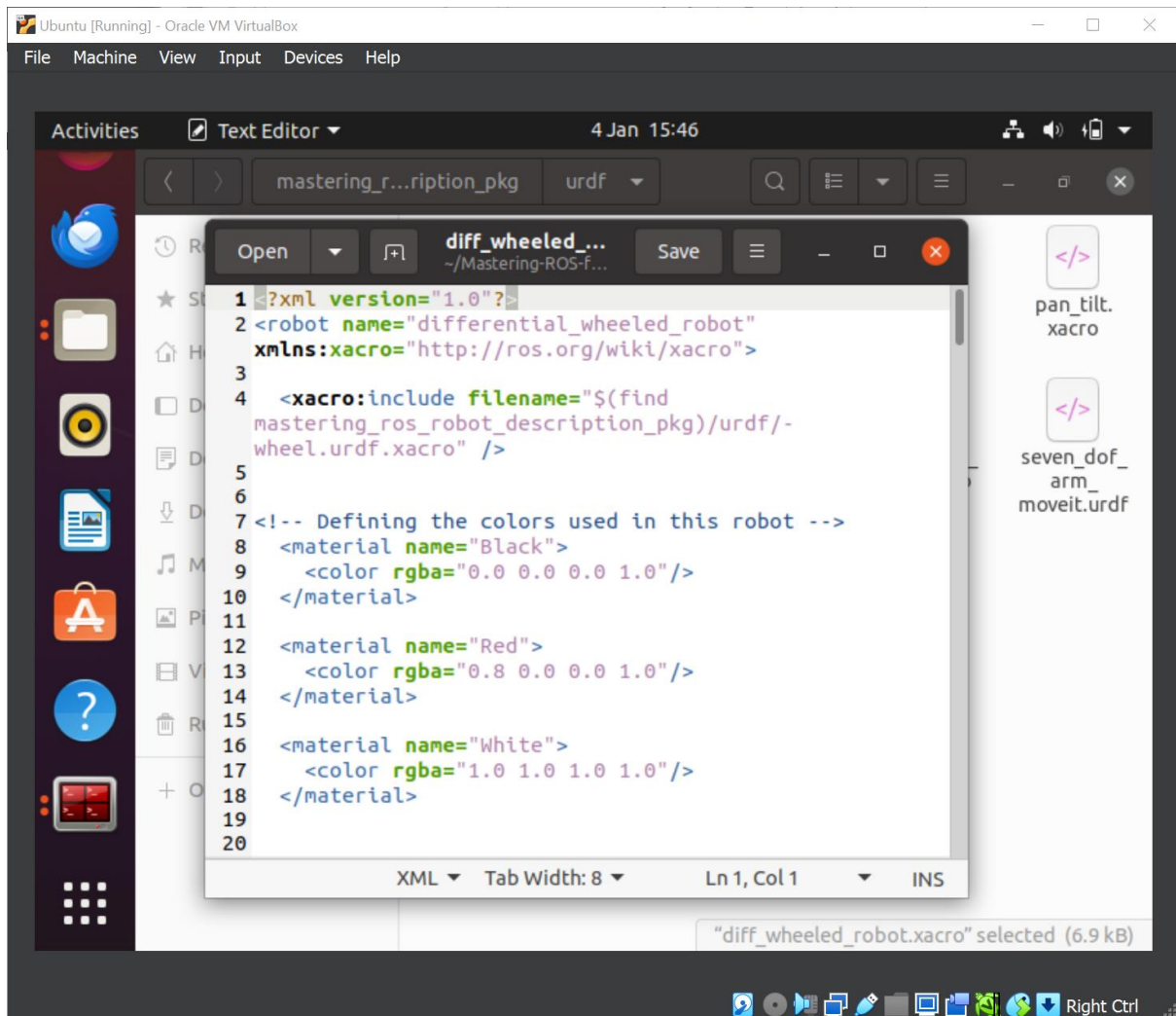
Model URDF dari robot ini hadir dalam package ROS yang telah di-clone. Robot ini memiliki lima sendi dan tautan, dengan dua sendi utama yang menghubungkan roda dengan basis robot. Tiga sendi lainnya adalah sendi tetap yang menghubungkan roda caster dan footprint basis ke badan robot.

wheel.urdf.xacro



File URDF `diff_wheeled_robot.xacro` berisi definisi untuk roda dan transmisinya, menggunakan file xacro terpisah (`wheel.urdf.xacro`) untuk mencegah duplikasi definisi. Gaya Gazebo, joint continuous, dan transmission diatur untuk masing-masing roda.

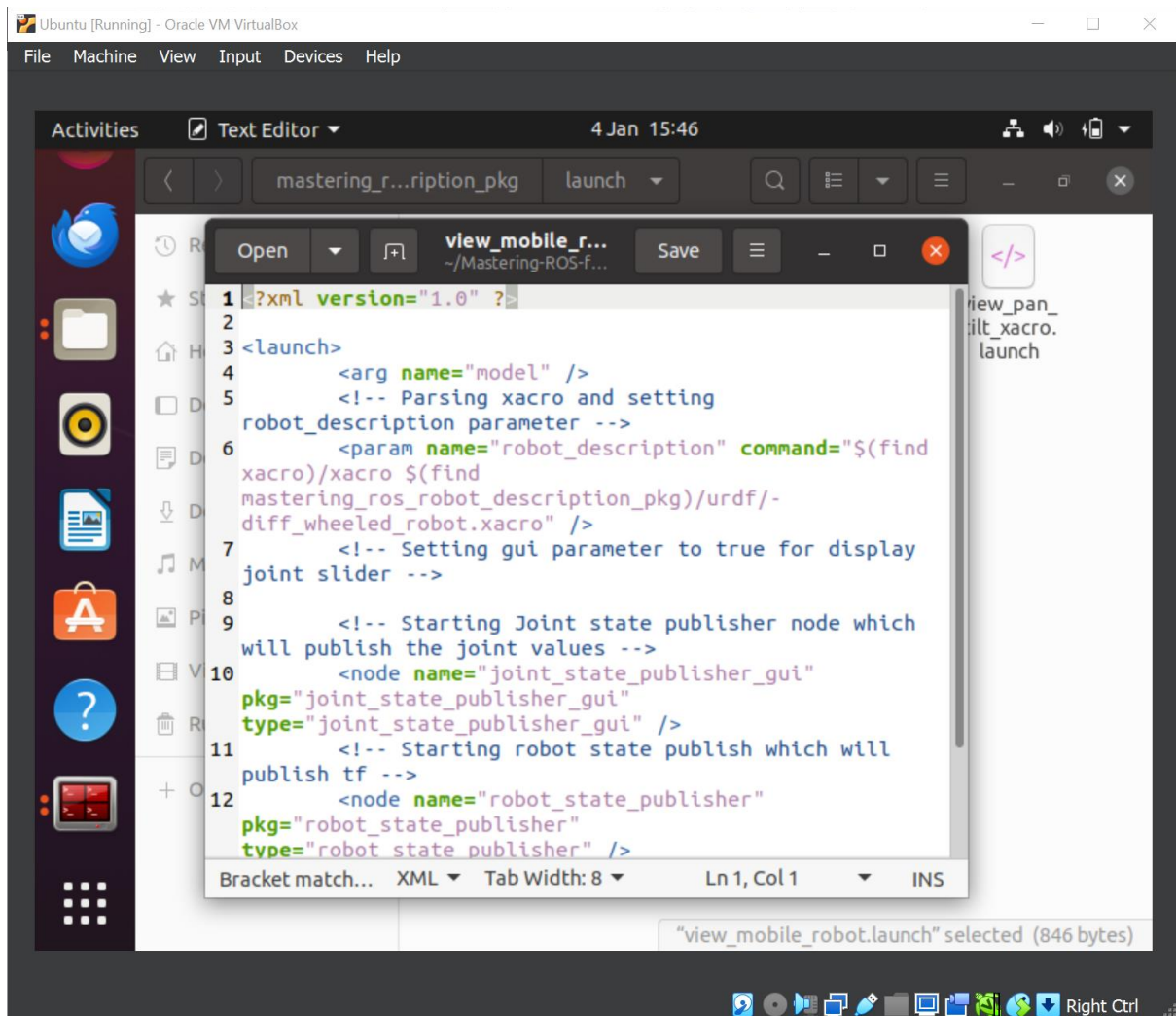
`diff_wheeled_robot.xacro`



```
1 <?xml version="1.0"?>
2 <robot name="differential_wheeled_robot"
  xmlns:xacro="http://ros.org/wiki/xacro">
3
4   <xacro:include filename="$(find
    mastering_ros_robot_description_pkg)/urdf/-
    wheel.urdf.xacro" />
5
6
7 <!-- Defining the colors used in this robot -->
8   <material name="Black">
9     <color rgba="0.0 0.0 0.0 1.0"/>
10  </material>
11
12  <material name="Red">
13    <color rgba="0.8 0.0 0.0 1.0"/>
14  </material>
15
16  <material name="White">
17    <color rgba="1.0 1.0 1.0 1.0"/>
18  </material>
19
20
```

File launch `view_mobile_robot.launch` memuat model robot ke RViz menggunakan xacro dan meluncurkan node joint state publisher, robot state publisher, dan RViz untuk visualisasi.

`view_mobile_robot.launch`



```
1 <?xml version="1.0" ?>
2
3 <launch>
4   <arg name="model" />
5   <!-- Parsing xacro and setting
robot_description parameter -->
6   <param name="robot_description" command="$(find
xacro)/xacro $(find
mastering_ros_robot_description_pkg)/urdf/-
diff_wheeled_robot.xacro" />
7   <!-- Setting gui parameter to true for display
joint slider -->
8
9   <!-- Starting Joint state publisher node which
will publish the joint values -->
10  <node name="joint_state_publisher_gui"
pkg="joint_state_publisher_gui"
type="joint_state_publisher_gui" />
11  <!-- Starting robot state publish which will
publish tf -->
12  <node name="robot_state_publisher"
pkg="robot_state_publisher"
type="robot_state_publisher" />

```

“view_mobile_robot.launch” selected (846 bytes)