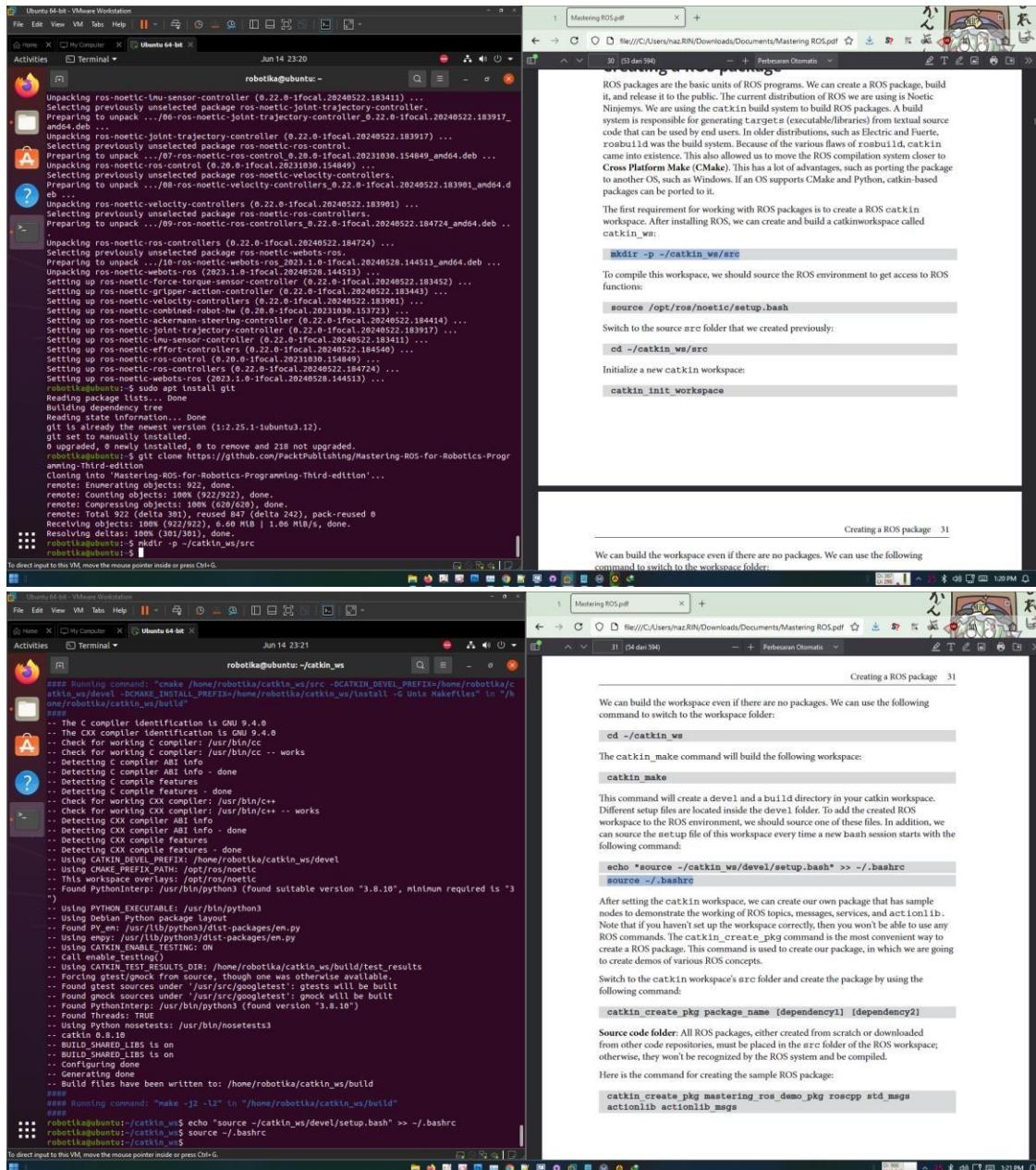
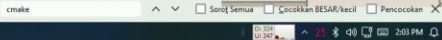


Bab 1 dan 2 mempelajari tentang apa itu ROS serta mempelajari bagaimana cara instalasi, konfigurasi, memahami server dan client ROS, memahami publisher dan subscriber dari ROS





Bab 3

Bab 3 mempelajari cara modifikasi file URDF dan juga mencoba Rviz untuk menampilkan demo robot

The image is a composite of four screenshots illustrating the process of visualizing a 3D robot model in RViz.

Top-Left Screenshot: Shows the RViz interface. The main 3D view displays a robot model with a blue base link, a red pan link, a green tilt link, and a pan-tilt mechanism. The left sidebar shows the 'Displays' panel with 'Global Options' and 'RobotModel' settings. The bottom status bar shows ROS Time: 1718436540.18, ROS Elapsed: 512.84, and Wall Time: 1718436540.21.

Top-Right Screenshot: Shows a PDF document titled 'Mastering ROS.pdf'. The section 'Visualizing the 3D robot model in RViz' explains that after designing the URDF, the GUI of joint_state_publisher was enabled. It provides a code snippet for the launch file:

```
<?xml version="1.0"?>
<launch>
  <arg name="model" />
  <param name="robot_description" textfile="$(find mastering_ros_robot_description_pkg)/urdf/pan_tilt.urdf" />
  <node name="joint_state_publisher_gui" pkg="joint_state_publisher_gui" type="joint_state_publisher_gui" />
  <node name="robot_state_publisher" pkg="robot_state_publisher" type="robot_state_publisher" />
</launch>
```

Bottom-Left Screenshot: Shows a terminal window and a file manager. The terminal displays the command `roslaunch mastering_ros_robot_description_pkg view_demo.launch` and its output. The file manager shows the directory `mastering_ros_robot_description_pkg/urdf` containing files like `pan_tilt.urdf` and `pan_tilt.xacro`.

Bottom-Right Screenshot: Shows a PDF document titled 'Working with ROS for 3D Modeling'. The section 'Visualizing the 3D robot model in RViz' explains that the joint type used here is a revolute, and the parent and child links are `base_link` and `pan_link`, respectively. It provides a code snippet for the URDF file:

```
robot name is: pan_tilt
----- Successfully parsed XML -----
root link: base_link has 1 child(ren)
child(1): pan_link
child(1): tilt_link
```

The document also mentions the `urdf_to_graphviz` command to generate a graphical representation of the URDF file.

Ubuntu 64-bit - VMware Workstation

File Edit View VM Help

Jun 15 00:40

urdfviz - RViz

File Panels Help

Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point

Displays

Global Options

Fixed Frame base_link

Background Color 122; 122; 122

Frame Rate 30

Default Light

Global Status: OK

Fixed Frame OK

Grid

Status: OK

Reference Frame <Fixed Frame>

Plane Cell Count 10

Normal Cell Count 0

Cell Size 1

Line Style Lines

Color 0; 0; 0

Alpha 0.5

Plane XY

Offset 0; 0; 0

Add Duplicate Remove Rename

Time

Pause Synchronization: Off ROS Time: 1718437210.72 ROS Elapsed: 33.42 Wall Time: 1718437210.76 Wall Elapsed: 33.39

Reset 31 fps

```
[WARN] [1718437176.551215193]: link 'front_left_wheel' material 'DarkGray' undefined.
[WARN] [1718437176.551363393]: The root link base_footprint has an inertia specified in the URDF, but KDL does not support a r
oot link with an inertia. As a workaround, you can add an extra dummy link to your URDF.
[WARN] [1718437177.110222924]: link 'front_right_wheel' material 'DarkGray' undefined.
[WARN] [1718437177.110253417]: link 'front_right_wheel' material 'DarkGray' undefined.
[WARN] [1718437177.110281964]: link 'front_left_wheel' material 'DarkGray' undefined.
[WARN] [1718437177.110289142]: link 'front_left_wheel' material 'DarkGray' undefined.
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Mastering ROS.pdf

File:///C:/Users/naz.RIN/Do...

1 (113 dca 394)

Perfexaram Chromis

mobile robot with differential wheeled mechanisms.

Creating a robot model for the differential drive mobile robot

A differential wheeled robot will have two wheels connected to opposite sides of the robot chassis, which is supported by one or two caster wheels. The wheels will control the speed of the robot by regulating the velocity of the single wheels. If the two motors are running at the same speed, the wheels will move forward or backward. If one wheel is running slower than the other, the robot will turn to the side of the lower speed. If we want to turn the robot to the left side, we reduce the velocity of the left wheel and vice versa.

There are two supporting wheels, called **caster wheels**, that will support the robot and rotate freely based on the movement of the main wheels.

90 Working with ROS for 3D Modeling

The URDF model of this robot is present in the cloned ROS package. The final robot model is shown here:

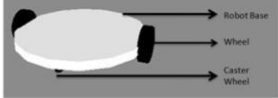



Figure 3.12 - The differential drive mobile robot

The preceding robot has five joints and links. The two main joints connect the wheels to the robot, while the others are fixed joints connecting the caster wheels and the base footprint to the body of the robot.

The preceding robot has five joints and five links. The two main joints connect the wheels with the base of the robot. The other joints are fixed and are used to link the caster wheels and the base footprint of the robot with its base link, respectively. Here is the connection graph of this robot.



Bab 4

Bab 4 mempelajari tentang menjalankan demo robot menggunakan Gazebo dan juga mengontrolnya dengan menggunakan teleop dan menampilkan gambar sensor menggunakan `image_view`

The image displays two screenshots of a Gazebo simulation environment. The left screenshot shows a 3D model of a robotic arm on a white platform. The right screenshot shows a terminal window with ROS commands and a document window with a PDF file.

Top Screenshot:

- Gazebo Window:** Shows a 3D simulation of a robotic arm on a white platform. The interface includes a menu bar (File, Edit, Camera, View, Window, Help), a toolbar, and a sidebar with options like World, Insert, Layers, GUI, Scene, Spherical Coordinates, Physics, Atmosphere, Wind, Models, and Lights. A status bar at the bottom indicates Real Time Factor: 0.89, Sim Time: 00:00:03.35.978, Real Time: 00:00:03.37.667, Iterations: 215978, and FPS.
- Terminal Window:** Displays the following commands and output:

```
roslaunch seven_dof_arm_gazebo seven_dof_arm_world.launch
```
- PDF Window:** Shows a document titled "Mastering ROS.pdf" with the following text:

Run a python script to send a service call to gazebo_ros to spawn a URDF robot. ...

```
roslaunch seven_dof_arm_gazebo seven_dof_arm_world.launch
```

Bottom Screenshot:

- Gazebo Window:** Shows the same 3D simulation of the robotic arm. The status bar at the bottom indicates Real Time Factor: 0.97, Sim Time: 00:00:00:15.880, Real Time: 00:00:00:18.017, Iterations: 15880, and FPS.
- Terminal Window:** Displays the following commands and output:

```
roslaunch seven_dof_arm_gazebo seven_dof_arm_world.launch
```
- PDF Window:** Shows a document titled "Mastering ROS.pdf" with the following text:

The plugin filename of Xtion Pro is libgazebo_ros_opengl_xi10ect.so, and we can define the plugin parameters, such as the camera name and image topic.

Simulating the robotic arm with Xtion Pro

Now that we have learned about the camera plugin definition in Gazebo, we can launch our complete simulation using the following command:

```
roslaunch seven_dof_arm_gazebo seven_dof_arm_world.launch
```

We can use the robot model with a sensor on the top of the arm, as shown here:

Figure 4-2: Simulation of a seven DOF arm with Asus Xtion Pro in Gazebo

We can now work with the simulated `rgb-d` sensor as if it were directly plugged into our computer. So, we can check whether it provides the correct image output.

106 Simulating Robots Using ROS and Gazebo

Visualizing the 3D sensor data

After launching the simulation using the preceding command, we can check the topics generated by the sensor plugin:

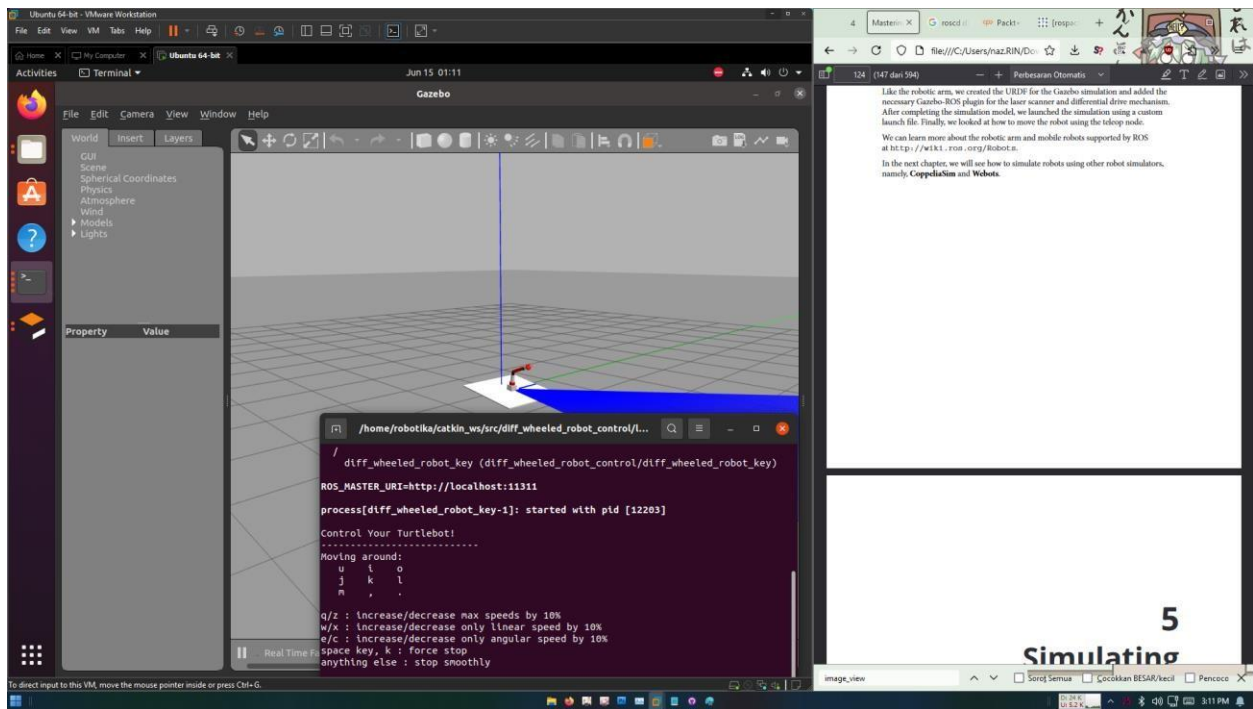
```
rostopic list
```

```
/gazebo/robot_1/depth/image_raw
```

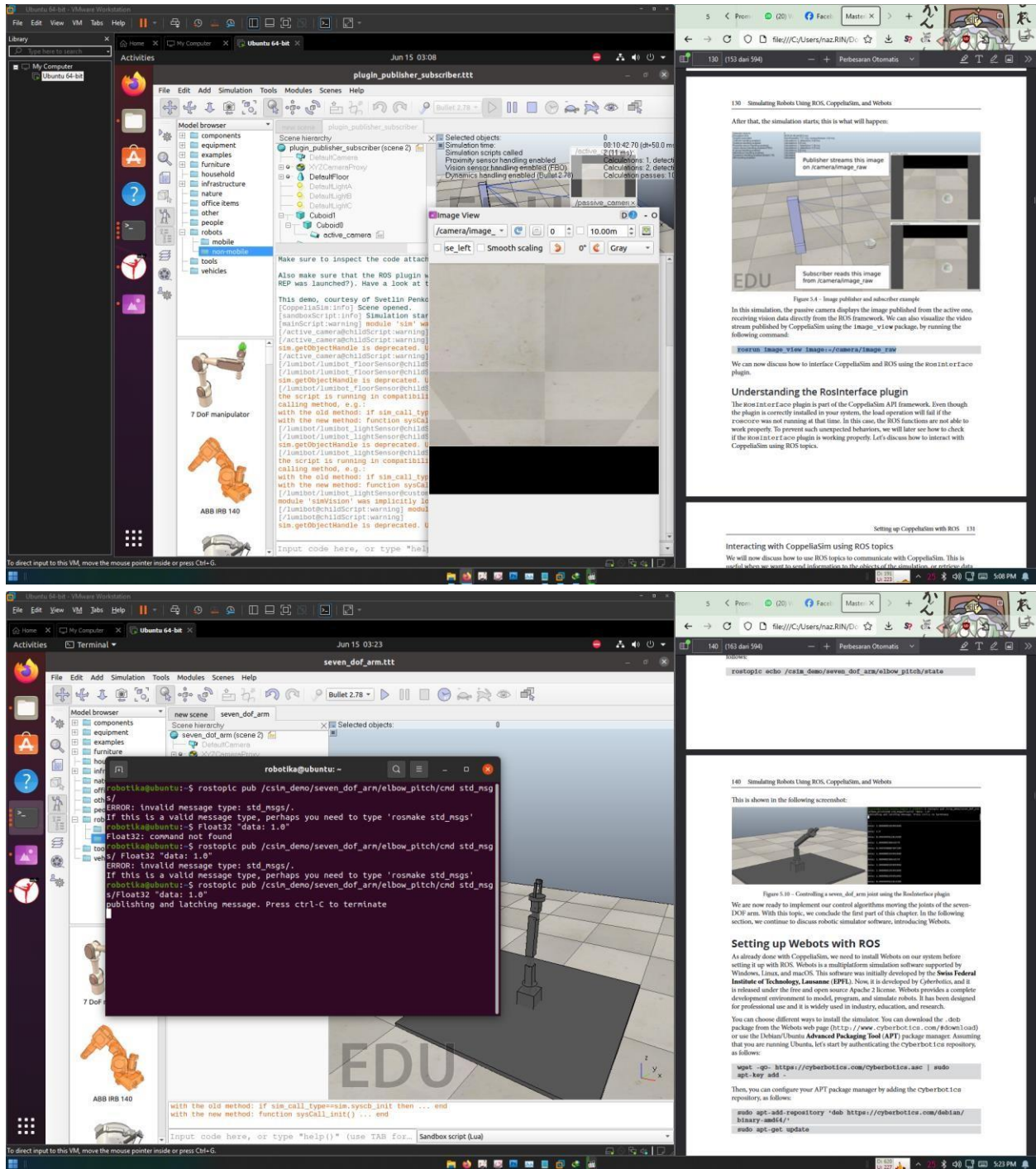
```
/gazebo/robot_1/depth/image_raw
```

```
/gazebo/robot_1/rgb/image_raw
```

```
/gazebo/robot_1/rgb/image_raw
```

Bab 5 Mencoba menjalankan demo robot menggunakan CoppeliaSim dan juga mengontrolnya menggunakan ROS



Bab 6 mempelajari tentang bagaimana membuat konfigurasi robot baru dengan menggunakan moveit dan juga bagaimana cara melakukan control dengan ROS. Hanya saja penulis menemukan kendala tidak bisa menjalankan MoveIt

The image is a collage of four screenshots illustrating the process of setting up ROS MoveIt for a robot arm.

- Top Left:** A screenshot of the 'MoveIt Setup Assistant' GUI. The 'Define Planning Groups' window is open, showing the 'Create New Planning Group' dialog. The 'Kinematics' section is filled out with 'Groun Name: kinova', 'Kinematic Solver: IK Solver', 'Kin. Search Resolution: 0.005', 'Kin. Search Timeout (sec): 0.005', 'Goal Joint Tolerance (mirad): 0.000001', 'Goal Position Tolerance (mm): 0.000001', 'Goal Orientation Tolerance (rad): 0.000001', and 'Kin. parameters file:'. The 'OMPL Planning' section shows 'Group Default Planner: None'. The 'Next, Add Components To Group:' section has 'Recommended: Add Kin, Chain' and 'Advanced Options: Add Subgroups'. A 3D model of a robot arm is visible in the background.
- Top Right:** A screenshot of a web browser displaying a PDF document titled 'Mastering ROS.pdf'. The document is open to a page discussing the 'Rapidly Exploring Random Tree (RRT)' algorithm.
- Bottom Left:** A screenshot of a terminal window in a VM. The user is logging into a robotika VM and running ROS launch commands. The output shows the user is in the 'robotika@ubuntu:~' directory and has run 'rosrun moveit_setup_assistant moveit_setup_assistant.launch http://localhost:11311'. The terminal also shows the user running 'roslaunch seven_dof_arm_gazebo seven_dof_arm_bringup_moveit.l' and 'roslaunch seven_dof_arm_bringup_moveit.launch'.
- Bottom Right:** A screenshot of a web browser displaying a PDF document titled 'Motion planning of a robot in RViz using the MoveIt configuration package'. The document is open to a page discussing the 'Motion planning of a robot in RViz using the MoveIt configuration package'.