HASIL ANALISIS

Chapter 1 – Introduction to ROS

ROS atau Robot Operating System adalah kerangka kerja perangkat lunak open-source yang digunakan untuk mengembangkan aplikasi robotika. ROS membantu pengembang mengontrol robot dengan lebih mudah, menyediakan tools dan library untuk berbagai keperluan seperti pemetaan, navigasi, dan manipulasi objek

Kelebihan ROS:

-Mendukung berbagai jenis robot.

-Komunitas global yang besar dan aktif.

-Banyak library siap pakai.

-Mudah diintegrasikan dengan sensor, aktuator, dan algoritma AI.

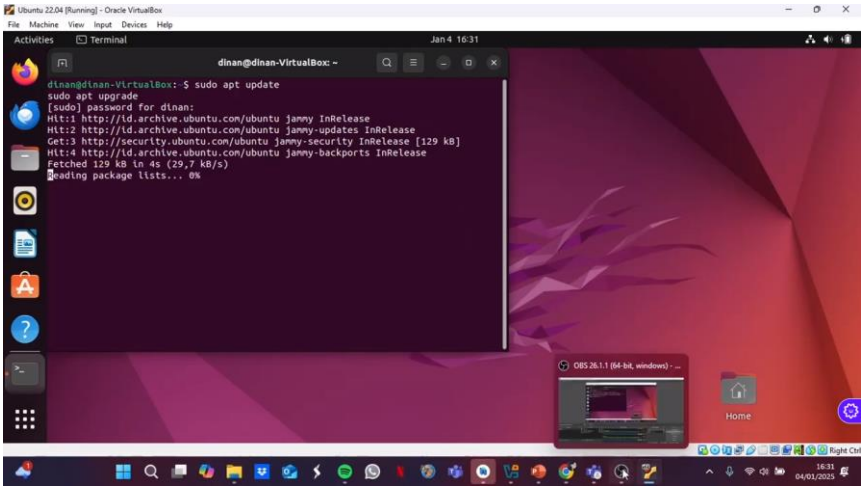
Distribusi ROS:

-ROS 1: Melodic, Noetic.

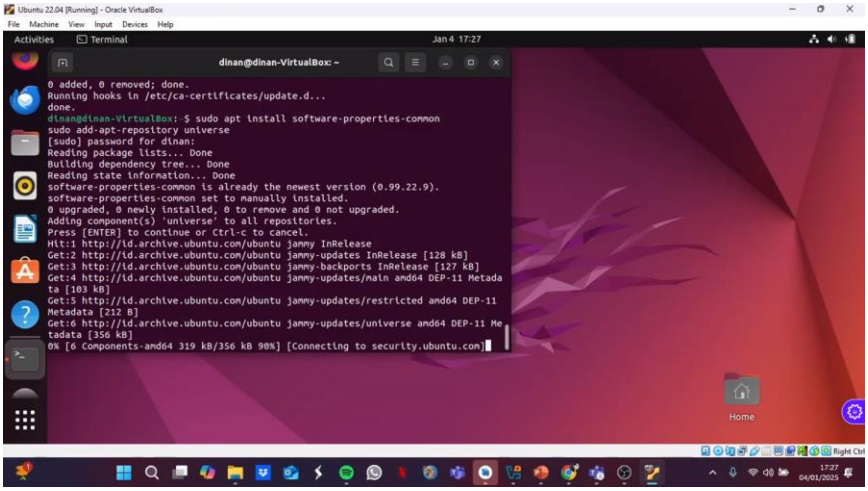
-ROS 2: Foxy, Galactic, Humble, dan Iron.

CARA INSTALL ROS :

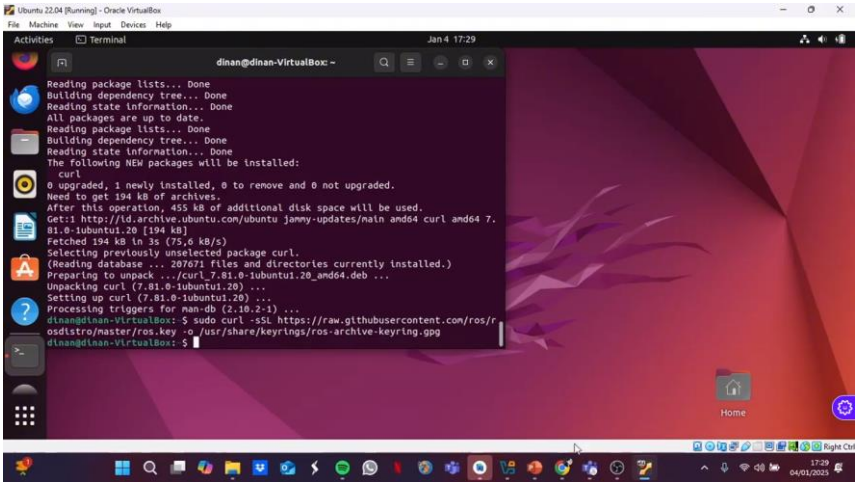
- Upgrade sistem



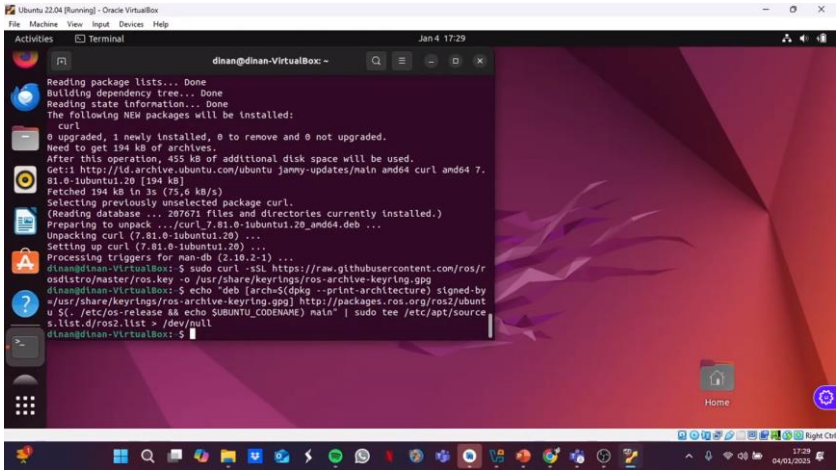
- Tambahkan repository ROS2



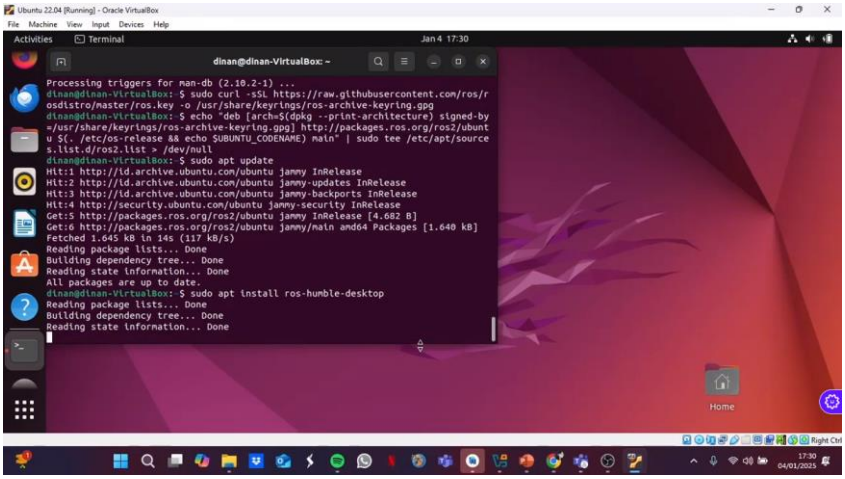
- Tambahkan Kunci GPG ROS2



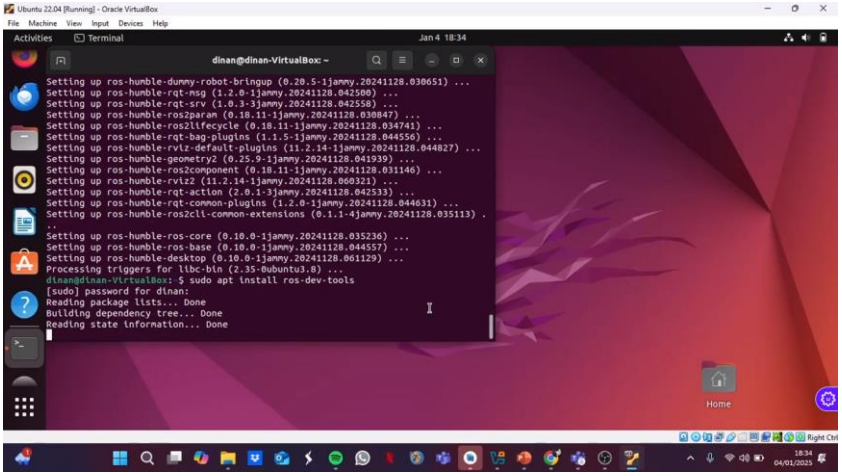
- Tambahkan repository ROS2 ke sourcelist



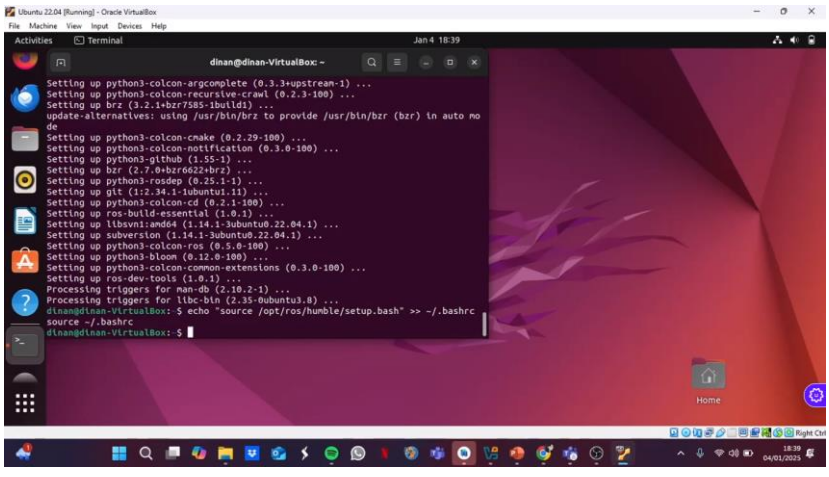
- Install ROS2 Humble



- Install Development Tools



- Tambahkan Setup ROS2 ke ./bashrc



Chapter 2 – Getting Started With ROS Programming

Simulasi Pertama : Publisher dan Subscriber dengan Topic

- Terminal 1: ros2 run mastering_ros_demo_pkg demo_topic_publisher

  Tujuan: Menjalankan node yang mempublikasikan pesan ke sebuah topic tertentu.

- Terminal 2: ros2 run mastering_ros_demo_pkg demo_topic_subscriber
  Tujuan: Menjalankan node yang berlangganan ke topic yang sama untuk menerima dan memproses pesan.

Hasil: Subscriber akan mencetak atau memproses pesan yang diterima dari publisher.

Simulasi Kedua : Publisher dan Subscriber dengan Custom Message

- Terminal 1: ros2 run mastering_ros_demo_pkg demo_msg_publisher
  Tujuan: Memublikasikan pesan khusus (custom message) yang didefinisikan dalam package ini ke sebuah topic.
- Terminal 2: ros2 run mastering_ros_demo_pkg demo_msg_subscriber
  Tujuan: Berlangganan ke topic yang sama untuk menerima dan memproses pesan khusus tersebut.

Hasil: Anda dapat melihat demonstrasi penggunaan custom message di ROS 2.

Simulasi Ketiga : Service Server dan Client

- Terminal 1: ros2 run mastering_ros_demo_pkg demo_service_server
  Tujuan: Menjalankan node service server yang menyediakan sebuah layanan (service).
- Terminal 2: ros2 run mastering_ros_demo_pkg demo_service_client
  Tujuan: Menjalankan node service client untuk memanggil layanan dari server.

Hasil: Client akan mengirim permintaan ke server, dan server akan merespons sesuai dengan logika yang telah didefinisikan.

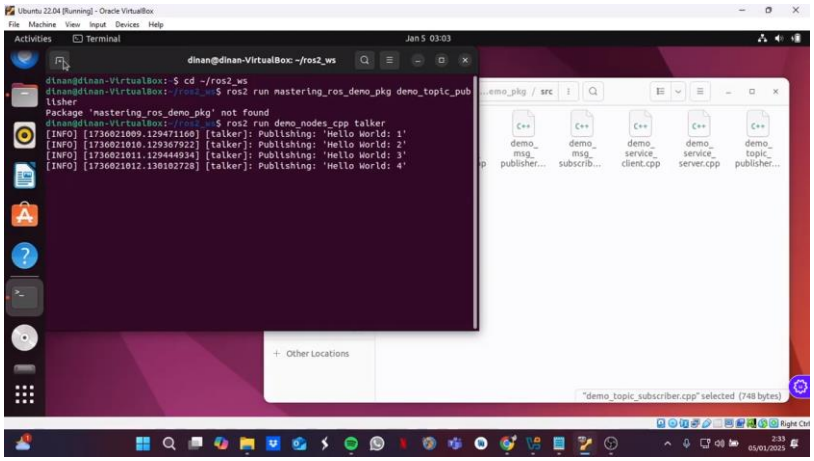
Simulasi Keempat : Action Client dan Server

- Terminal 1: ros2 run mastering_ros_demo_pkg demo_action_client 10 1
  Tujuan: Menjalankan action client yang meminta sebuah aksi dengan parameter (misalnya 10 dan 1 sebagai input).
- Terminal 2: ros2 run mastering_ros_demo_pkg demo_action_server
  Tujuan: Menjalankan action server yang menerima permintaan dari client dan mengeksekusi aksi sesuai logika yang telah didefinisikan.

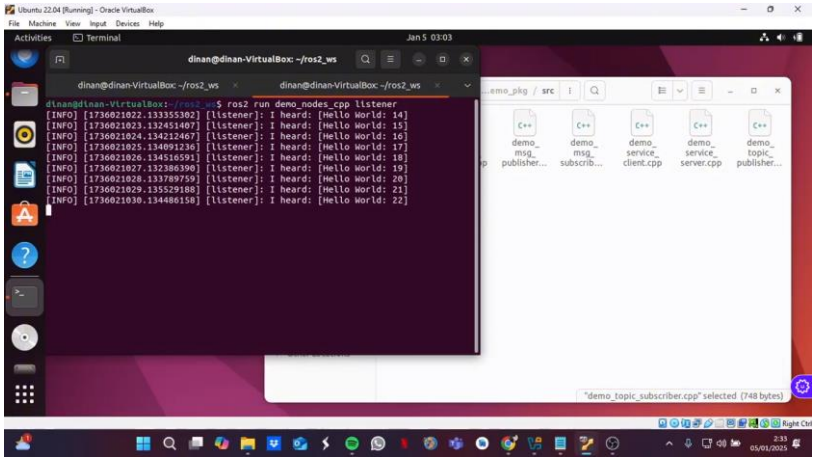
Parameter 10 dan 1 mungkin mengacu pada jumlah tugas dan interval waktu eksekusi. Client mengirimkan permintaan ke server, yang kemudian memproses aksi dan memberikan hasil atau pembaruan status.

Hasil: Action server akan melaporkan status eksekusi atau hasil akhir kepada client.

Talker :



Listener :



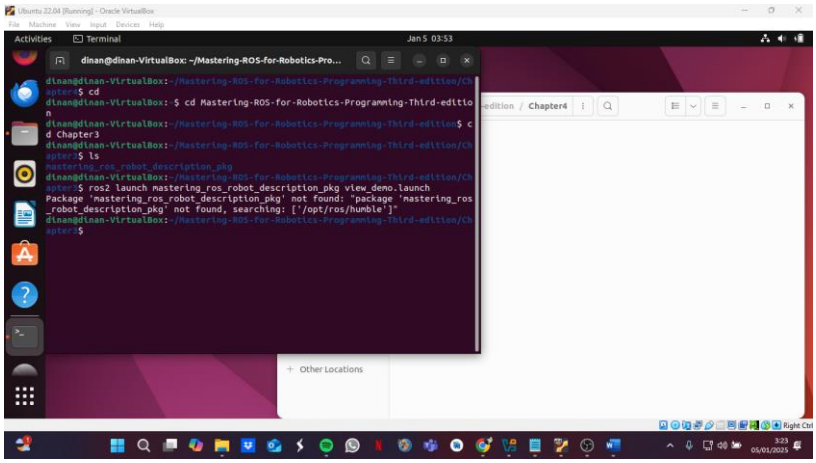
Chapter 3 – Working with ROS for 3D Modeling

Simulasi Pertama :

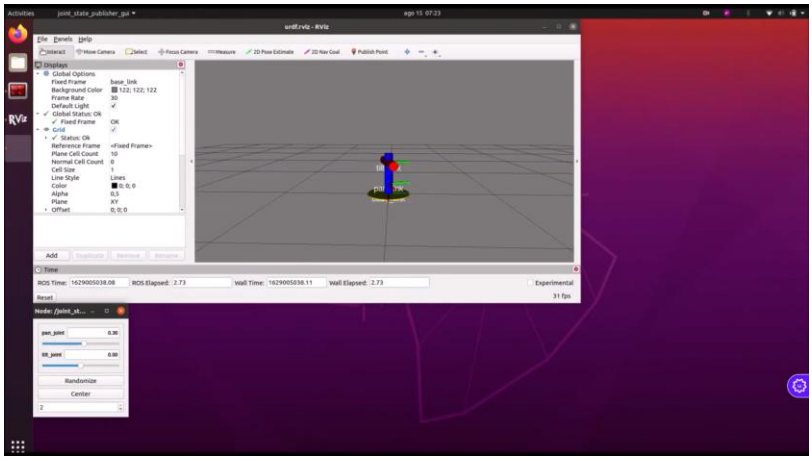
Buka Terminal , Jalankan Perintah :

ros2 launch mastering_ros_robot_description_pkg view_demo.launch

Output Saya (Belum Berhasil) :



Output Seharusnya :

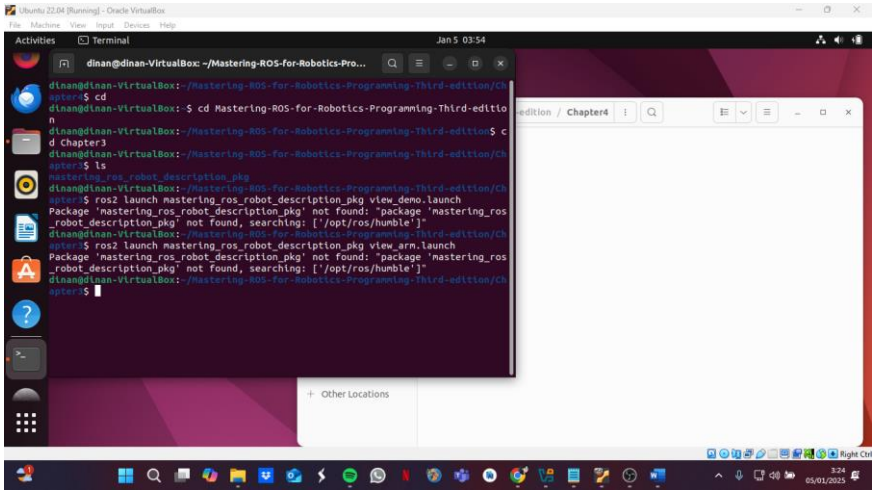


Simulasi Kedua :

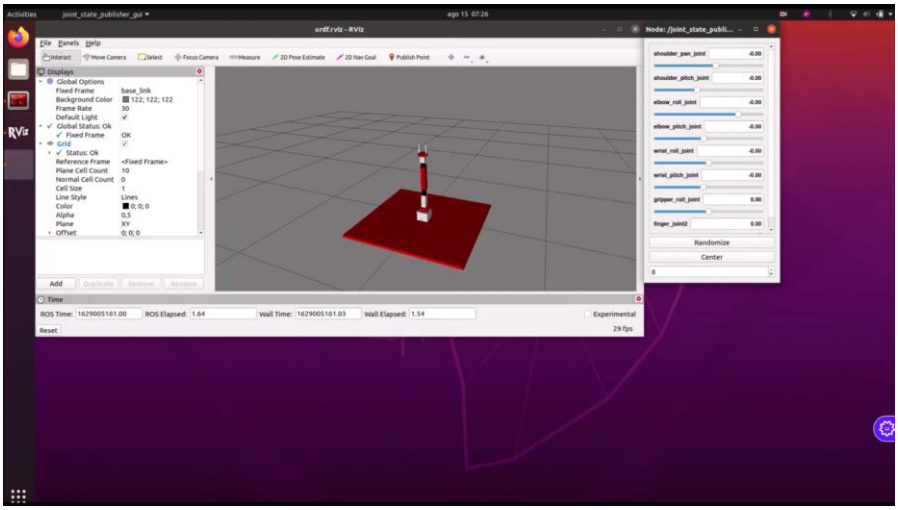
Buka Terminal , Jalankan Perintah :

ros2 launch mastering_ros_robot_description_pkg view_arm.launch

Output Saya (Belum Berhasil) :



Output Seharusnya :

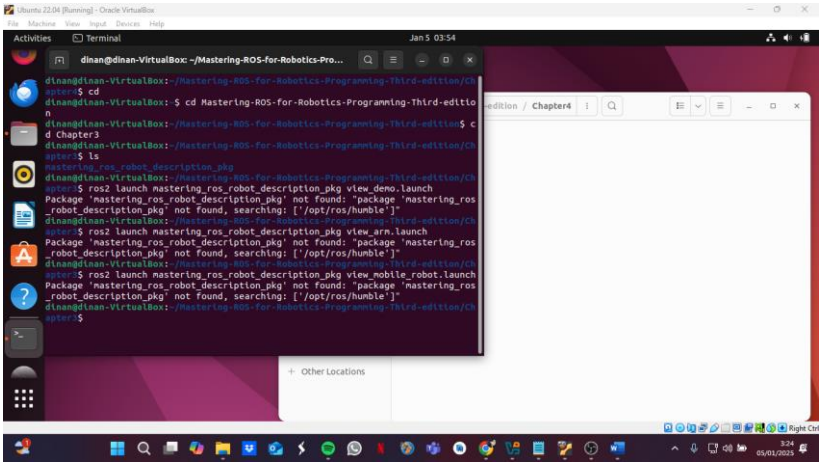


Simulasi Ketiga :

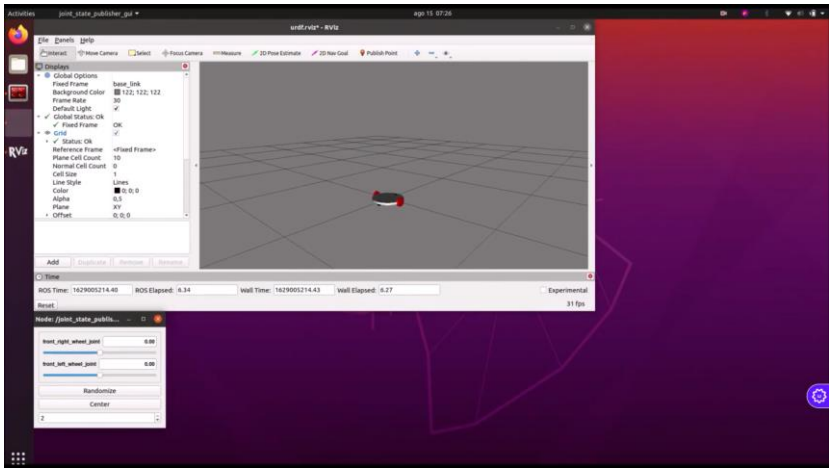
Buka Terminal , Jalankan Perintah :

ros2 launch mastering_ros_robot_description_pkg view_mobile_robot.launch

Output Saya (Belum Berhasil) :



Output Seharusnya :



Dari ketiga demo tersebut,

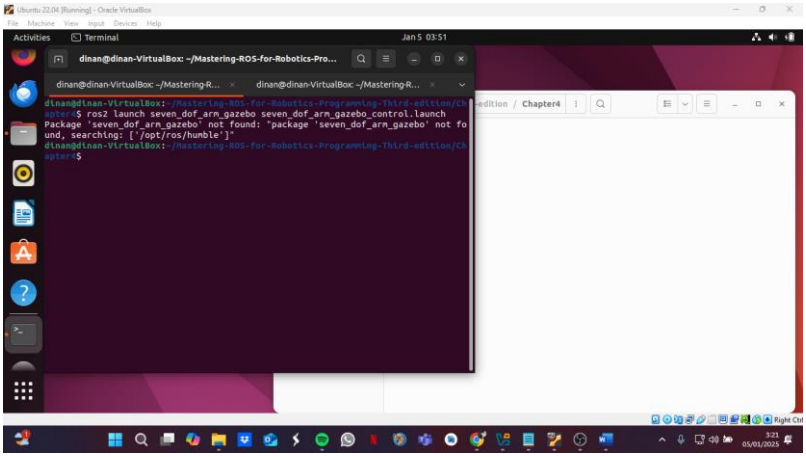
- perintah pertama menampilkan visualisasi 3D robot model di Rviz
- perintah kedua menampilkan xacro model dari lengan seven-DOF
- perintah ketiga membuat model robot untuk differential drive mobile robot

Chapter 4 – Simulating Robot Using ROS & Gazebo

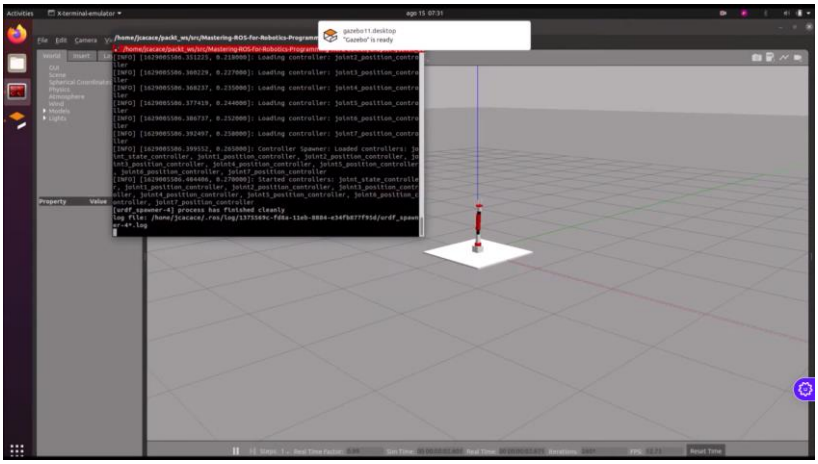
Simulasi Pertama : Menggerakkan lengan robot

- pada terminal 1 jalankan perintah : roslaunch seven_dof_arm_gazebo seven_dof_arm_gazebo_control.launch

Output Saya (Belum Berhasil) :

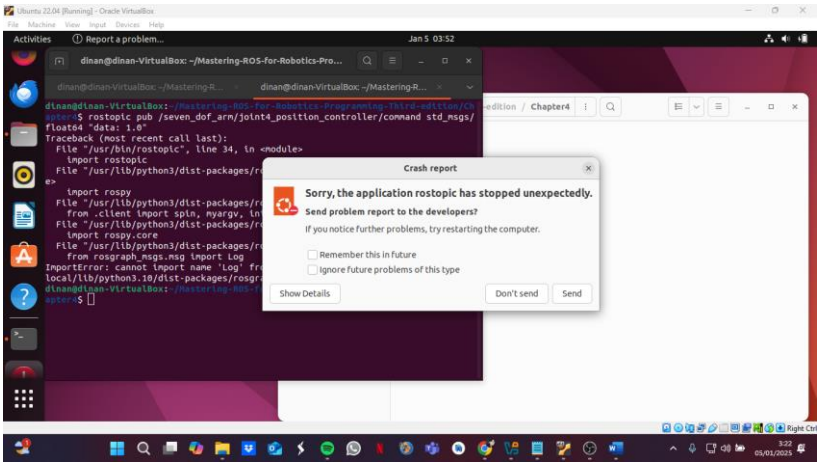


Output Seharusnya :

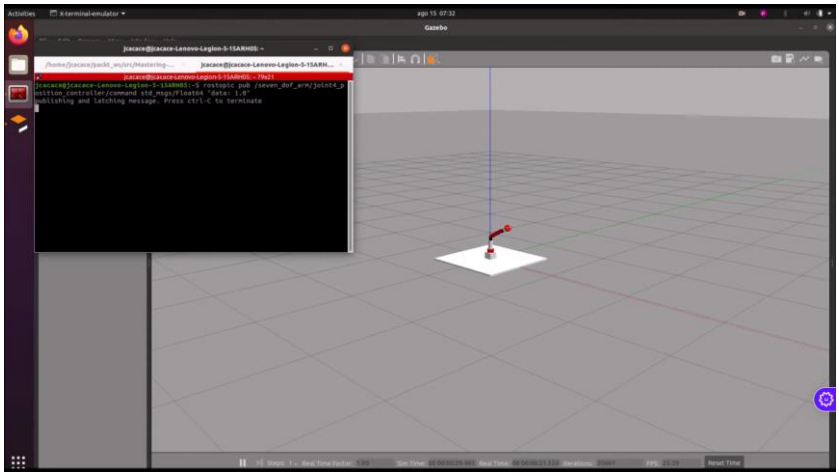


- pada terminal 2 jalankan perintah : rostopic pub /seven_dof_arm/joint4_position_controller/command std_msgs/float64 "data: 1.0"

Output Saya (Belum Berhasil) :



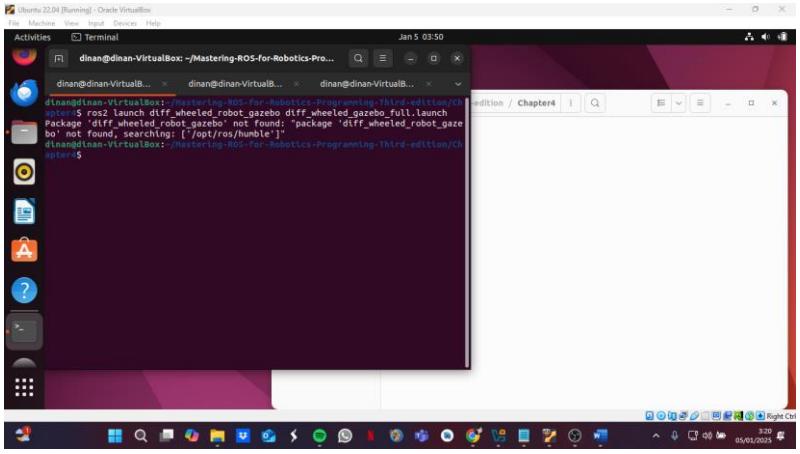
Output Seharusnya :



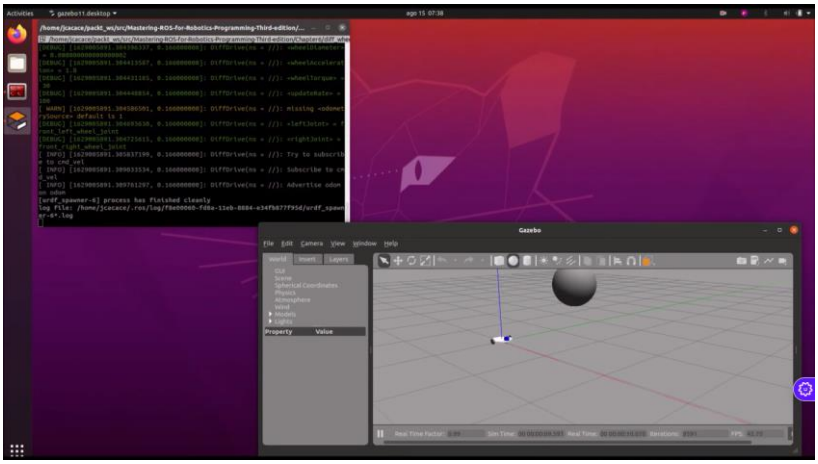
Simulasi Kedua : Menggerakkan Posisi Robot

- pada terminal 1 jalankan perintah : roslaunch diff_wheeled_robot_gazebo diff_wheeled_gazebo_full.launch

Output Saya (Belum Berhasil) :

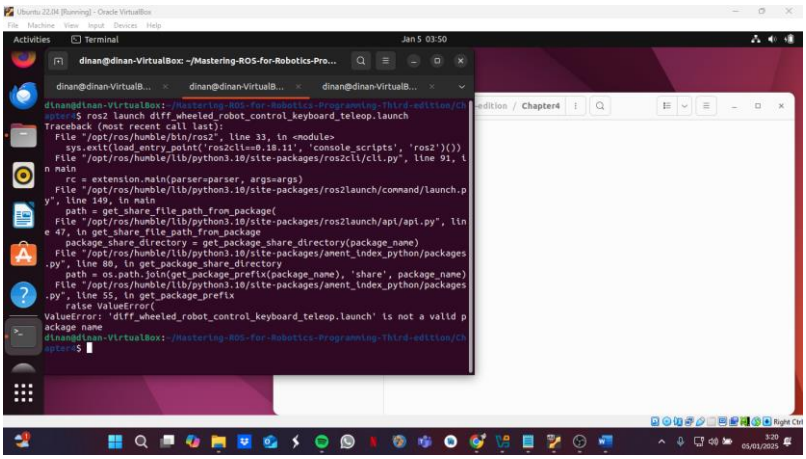


Output Seharusnya :

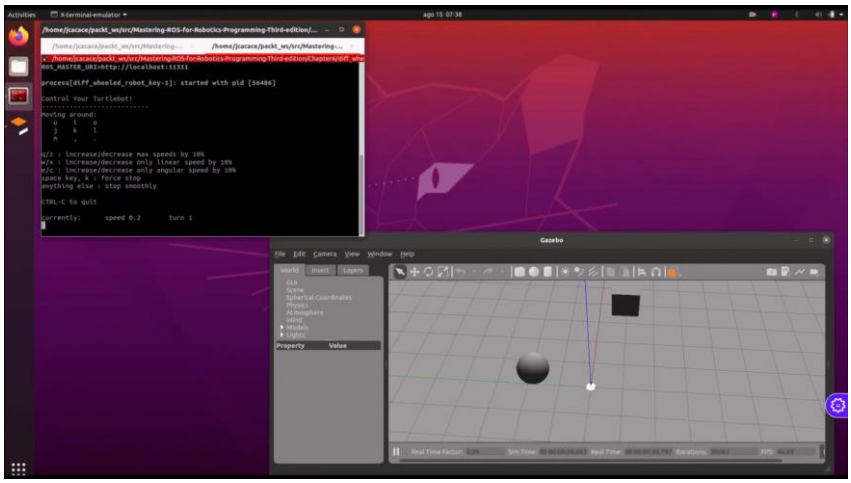


- pada terminal 2 jalankan perintah : roslaunch diff_wheeled_robot_control_keyboard_teleop.launch

Output Saya (Belum Berhasil) :



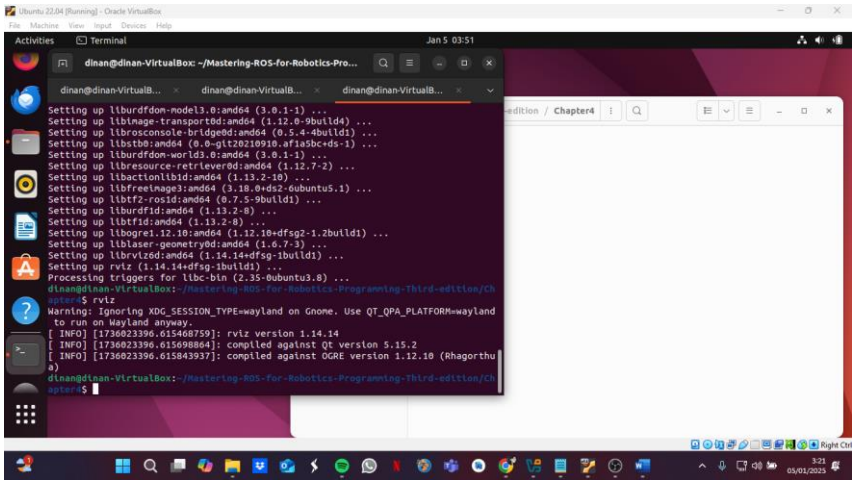
Output Seharusnya :



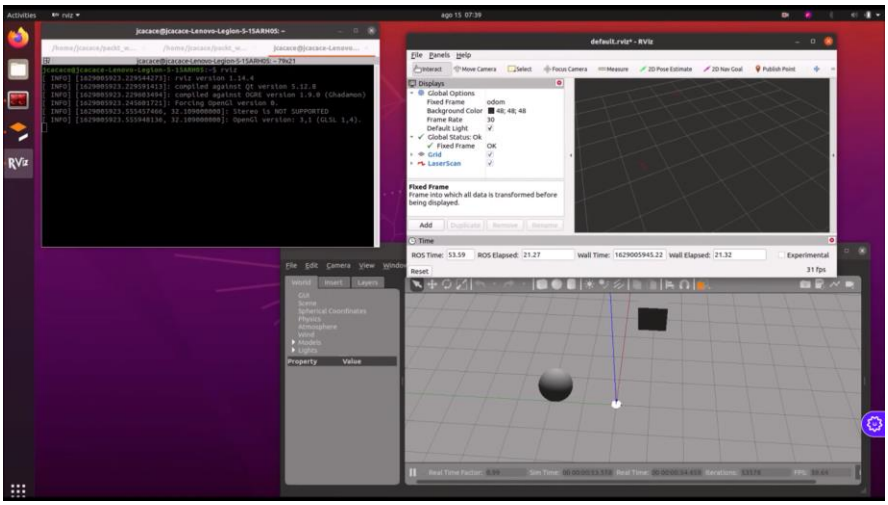
- pada terminal 3 jalankan perintah : rviz

  kemudian ubah fixed frame menjadi "odom"

Output Saya (Belum Berhasil) :



Output Seharusnya :



Dari kedua demo tersebut,

- pada simulasi pertama, terminal 1 berperan sebagai pembuka program dan terminal 2 berperan sebagai controller gerak robot
- pada simulasi kedua, terminal 1 berperan membuka program gazebo, kemudian terminal 2 berperan sebagai controller berbasis keyboard, dan terminal 3 membuka RViZ