

**Technical Report UAS Robotika
Mastering ROS**



Oleh :
Imam Algiza
1103200077

**PROGRAM STUDI S1 TEKNIK KOMPUTER
FAKULTAS TEKNIK ELEKTRO
UNIVERSITAS TELKOM
2024**

CHAPTER 1

ROS (Robot Operating System) adalah sebuah framework open-source yang dirancang untuk membangun aplikasi perangkat lunak untuk robot. Meskipun namanya "Operating System," ROS sebenarnya bukanlah sistem operasi dalam arti konvensional seperti Windows atau Linux, melainkan sebuah platform pengembangan yang berjalan di atas sistem operasi Linux.

1. Melakukan instalasi ROS Noetic

- `sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'`

```
imamalgiza@ImamAlgiza:~$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

- `sudo apt install curl`
- `curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -`
- `sudo apt install ros-noetic-desktop-full`

```
imamalgiza@ImamAlgiza:~$ sudo apt install ros-noetic-desktop-full
```

md

- `source /opt/ros/noetic/setup.bash`

2. Menjalankan command roscore

`roscore`

fungsi dari roscore ini adalah :

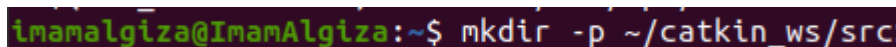
- Menyediakan Parameter Server yang merupakan penyimpanan data yang digunakan untuk menyimpan konfigurasi dan informasi parameter yang dibutuhkan oleh berbagai node dalam sistem ROS.
- Registrar dan penyedia nama (name service) untuk node-node ROS. Ketika node-node mulai berjalan, mereka mendaftarkan diri dengan roscore untuk mendapatkan alamat dan informasi lain yang diperlukan untuk komunikasi antar node.
- Sebagai "master" dalam arsitektur ROS, roscore bertanggung jawab untuk memfasilitasi komunikasi antar node. Node-node mengirimkan permintaan (requests) kepada roscore untuk menemukan node lain, mempublikasikan topik (topics), atau menggunakan layanan (services). roscore mengelola semua permintaan ini dan memberikan informasi yang diperlukan kepada node-node yang terlibat.
- Memfasilitasi sinkronisasi waktu (time synchronization) antara node-node ROS yang berjalan di sistem yang sama. Ini penting untuk aplikasi yang memerlukan waktu yang sinkron.
- Memungkinkan node-node ROS untuk menemukan, mendaftar, dan berkomunikasi satu sama lain melalui topik (topics) dan layanan (services). Node-node ini dapat berada di sistem yang sama atau di jaringan yang berbeda, asalkan mereka terhubung melalui ROS network.

CHAPTER 2

Pada bab 2 ini membahas dasar pemrograman ROS, lebih spesifiknya membahas pembuatan package ROS menggunakan catkin, disini belajar menginisialisasi catkin, membuat paket baru dan menambahkan dependensi yang diperlukan. Langkah - langkah yang harus dilakukan antara lain :

1. membuat workspace ROS

```
mkdir -p ~/catkin_ws/src
```



```
imamalgiza@ImamAlgiza:~$ mkdir -p ~/catkin_ws/src
```

2. source ROS environment

```
source /opt/ros/noetic/setup.bash
```

3. membuat workspace catkin

```
cd ~/catkin_ws/src
```

```
catkin_init_workspace
```

```
cd ~/catkin_ws
```

```
catkin_make
```

4. membuat package catkin

```
catkin_create_pkg package_name [dependency1] [dependency2]
```

5. membuat sample ROS package

```
catkin_create_pkg mastering_ros_demo_pkg roscpp std_msgs
```

```
actionlib actionlib_msgs
```

Selain itu pada bab 2 ini juga membahas pembuatan nodes, nodes adalah unit dasar dari proses komputasi. Node-node ini adalah program-program yang berjalan secara mandiri dan berkomunikasi satu sama lain dalam sistem yang menggunakan ROS. Setiap node menjalankan tugas tertentu dalam aplikasi robotika atau sistem yang menggunakan ROS, dan mereka berinteraksi melalui mekanisme komunikasi yang disediakan oleh ROS. Langkah - langkah pembuatan nodes adalah :

1. lakukan langkah - langkah pembuatan workspace dan package seperti diatas
2. membuat node

```
roslaunch mastering_ros_demo_package demo_topic_publisher
```

node ini menjalankan program demo_topic_publisher dari ROS

```
roslaunch mastering_ros_demo_pkg demo_msg_subscriber
```

node ini menjalankan program demo_service_client dari ROS

CHAPTER 3

Pada bab 3 mensimulasikan 3D modelling menggunakan ROS, ROS terdapat beberapa paket utama untuk membangun model 3D dari sebuah robot yaitu, urdf yang berisikan C++ parser, joint_state_publisher yang digunakan untuk membaca deskripsi model robot, joint_state_publisher_gui, yang memberikan user interface untuk joint_state_publisher, kdl_parser, untuk membangun KDL tree dari model robot URDF, robot_state_publisher untuk membaca status joint dan pose dari robot dan xacro untuk membuat file URDF lebih mudah dibaca.

ROS dapat integrasi dengan berbagai sensor 3D seperti lidar, kinect, atau kamera stereovision, yang memungkinkan akses, pemrosesan, dan manipulasi data sensor menggunakan nodes ROS. Selain itu, ROS kompatibel dengan simulator 3D seperti Gazebo, yang memfasilitasi simulasi robotika dalam lingkungan yang realistis sebelum implementasi pada perangkat keras. ROS juga mendukung paket untuk manipulasi objek 3D seperti moveit, serta integrasi dengan alat pemodelan 3D seperti Blender atau CAD tools untuk pengembangan aplikasi yang melibatkan objek atau lingkungan 3D. Kemampuan ROS untuk berkomunikasi dengan perangkat lunak pemodelan 3D lainnya juga memungkinkan penggunaan data yang dihasilkan atau pengembangan algoritma tambahan untuk pengolahan data.

Langkah langkah 3D model di ROS antara lain :

1. membuat paket ROS dalam workspace catkin
catkin_create_pkg mastering_ros_robot_description_pkg roscpp tf geometry_msgs urdf rviz xacro
2. install urdf dan xacro (hanya jika belum terinstall)
sudo apt-get install ros-noetic-urdf
sudo apt-get install ros-noetic-xacro
3. membuat model robot dengan RViz
'roslaunch mastering_ros_robot_description_pkg view_demo.launch'
'roslaunch mastering_ros_robot_description_pkg view_arm.launch'
'roslaunch mastering_ros_robot_description_pkg view_mobile_robot.launch'

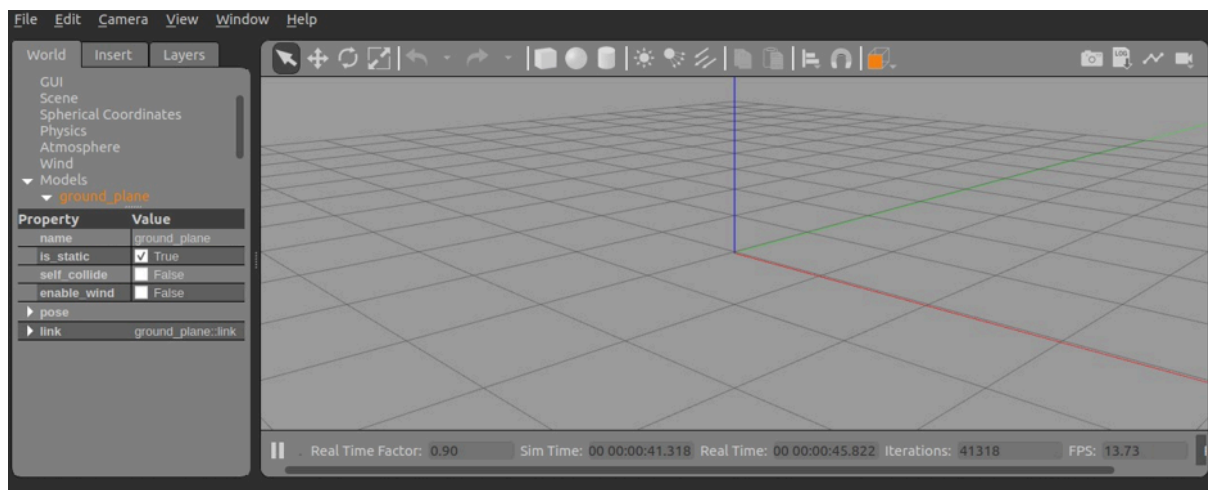
CHAPTER 4

Simulasi robot menggunakan ROS dan Gazebo merupakan pendekatan yang umum dan efektif dalam pengembangan robotika modern. Gazebo berperan sebagai simulator fisika 3D yang menyediakan lingkungan simulasi realistis untuk sistem robotik, memungkinkan pengembang untuk mensimulasikan robot, menguji algoritma, dan memvalidasi strategi kontrol dalam lingkungan virtual sebelum implementasi pada robot fisik. Integrasi antara ROS dan Gazebo memanfaatkan middleware ROS untuk mengatur komunikasi antar komponen robot, sementara Gazebo mengelola simulasi fisika dan lingkungan.

Pengembang menggunakan format URDF untuk mendeskripsikan model robot, mengintegrasikan plugin Gazebo untuk mensimulasikan sensor dan aktuator, serta mengonfigurasi pengontrol untuk mengatur pergerakan dan dinamika robot dalam simulasi. Melalui penggunaan topik dan layanan ROS, pengembang dapat menguji node-node dan algoritma mereka dalam lingkungan simulasi sebelum melangkah ke implementasi pada perangkat keras fisik, sambil memanfaatkan alat visualisasi seperti RViz untuk memonitor keadaan robot dan data sensor selama simulasi. Simulasi juga memungkinkan pengujian yang terkendali dan dapat diulang, serta penyetelan parameter kontrol tanpa resiko terhadap perangkat keras robot sebenarnya.

Langkah langkah pada chapter 4 ini adalah :

1. Buat workspace pada ROS
2. Mempersiapkan package ROS untuk robot
3. Mempersiapkan model URDF
4. Menjalankan gazebo
 - 'roslaunch seven_dof_arm_gazebo seven_dof_arm_world.launch'
 - 'rostopic pub /seven_dof_arm/joint4_position_controller/command std_msgs/Float64 "data :1.0"'
 - 'roslaunch diff_wheeled_robot_gazebo diff_wheeled_gazebo.launch'
 - 'roslaunch diff_wheeled_robot_control keyboard_teleop.launch'



CHAPTER 5

Pada bab ini kita mensimulasikannya menggunakan CoppeliaSim, yang merupakan sebuah perangkat lunak simulasi robotika yang kuat dan serbaguna yang dikembangkan oleh Coppelia Robotics. Perangkat lunak ini digunakan untuk mensimulasikan berbagai jenis robot dan sistem mekatronika dalam lingkungan 3D yang detail dan interaktif. CoppeliaSim (sebelumnya dikenal sebagai V-REP) menawarkan berbagai fitur seperti simulasi fisika yang realistis, integrasi dengan berbagai sensor dan aktuator, serta kemampuan untuk mengembangkan dan menguji algoritma kontrol robot sebelum penerapan pada perangkat keras fisik. CoppeliaSim juga mendukung pemrograman robot menggunakan berbagai bahasa seperti C/C++, Python, MATLAB, dan LUA, serta menyediakan antarmuka grafis yang intuitif untuk membangun dan mengkonfigurasi simulasi. Perangkat lunak ini populer di kalangan peneliti, mahasiswa, dan pengembang di bidang robotika untuk eksperimen, pengajaran, dan pengembangan aplikasi robotika yang kompleks.

Langkah langkah pada bab ini antara lain :

1. Download CoppeliaSim dengan versi yang sesuai dengan OS.
2. menjalankan program CoppeliaSim
'./coppeliaSim.sh'

