

Agung Aji Saputra

1103202114

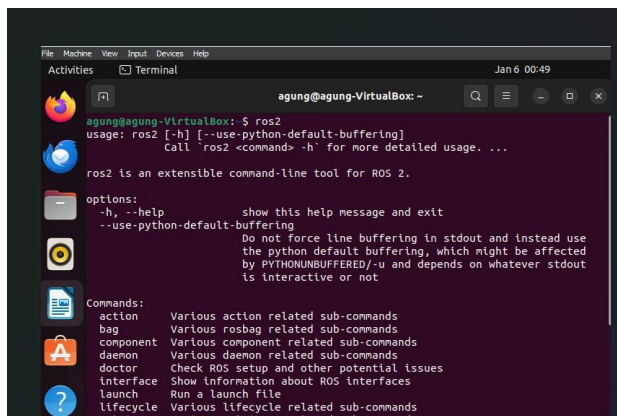
## UAS Robotika\_Technical Report

### Introduction to ROS

Robot Operating System (ROS) adalah sebuah kerangka kerja perangkat lunak yang telah menjadi landasan utama dalam dunia robotika modern. Dirancang untuk memudahkan dan mempercepat pengembangan, pengujian, serta pengoperasian robotika, ROS menyediakan lingkungan yang terstruktur dan konsisten bagi pengembang robot.

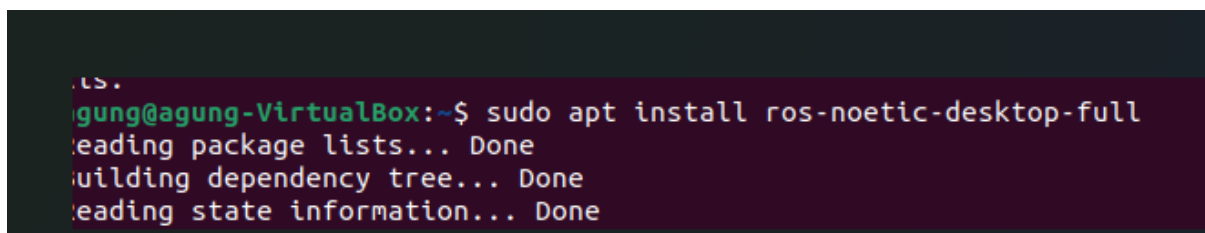
Dengan memanfaatkan ROS, pengembang dapat merancang dan mengintegrasikan berbagai komponen perangkat lunak dengan lebih mudah, mulai dari sensor hingga aktuator, kontroler, dan sistem pemrosesan data.

Hal ini memungkinkan robot berkomunikasi, berkoordinasi, dan beroperasi secara efisien dalam berbagai tugas yang kompleks. ROS tidak hanya memberikan perpustakaan perangkat lunak yang luas, tetapi juga menyediakan berbagai alat pengembangan yang mendukung simulasi, visualisasi, dan analisis data. Komunikasi antar-proses, manajemen paket, serta arsitektur modular ROS menjadikannya sangat populer di kalangan peneliti, pengembang, dan industri robotika.



```
agung@agung-VirtualBox: ~  
agung@agung-VirtualBox: $ ros2  
usage: ros2 [-h] [--use-python-default-buffering]  
          Call 'ros2 <command> -h' for more detailed usage. ...  
  
ros2 is an extensible command-line tool for ROS 2.  
  
options:  
  -h, --help            show this help message and exit  
  --use-python-default-buffering  
                        Do not force line buffering in stdout and instead use  
                        the python default buffering, which might be affected  
                        by PYTHONUNBUFFERED/-u and depends on whatever stdout  
                        is interactive or not  
  
Commands:  
  action                Various action related sub-commands  
  bag                   Various rosbag related sub-commands  
  component             Various component related sub-commands  
  daemon               Various daemon related sub-commands  
  doctor                Check ROS setup and other potential issues  
  interface             Show information about ROS interfaces  
  launch                Run a launch file  
  lifecycle              Various lifecycle related sub-commands  
  multiplatform         Various multiplatform related sub-commands
```

Sebelum memasuki modul ini di pastikan sudah menginstall ubuntu 20.40, hanya ubuntu 20.04 yang mendukung ros neotic



```
agung@agung-VirtualBox:~$ sudo apt install ros-noetic-desktop-full  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done
```

## Getting started With ROS Programming

Yang akan yang akan mempelajari perintah ros yang disebut catkin create package untuk membuat beberapa paket yang berguna dan paket yang tidak perlu di ruang kerja.

```
$ roscd
```

dokumen desktop mengunduh gambar publik dan direktori yang tersisa, kemudian jalankan perintah roscd yang merupakan perintah ros

```
ls
lib local_setup.bash local_setup.sh local_setup.zsh setup.bash setup.sh _setup_util.py setup.zsh share
sudo mkdir ros_ws
```

kemudian buat perintah direktori mkdir ros\_ws

```
cd ros_ws
```

```
$ ls
$ catkin_make
```

gunakan perintah ubah direktori cd\_ ws dan tekan tombol enter dan perintah pengguna ls untuk mencantumkan direktori tidak ada apa pun di dalamnya .

ketik ls list list ini dan menggunakan perintah ini kita bisa daftar semua file dan direktori di bawah, catkin\_make adalah perintah yang sering digunakan dalam Robot Operating System (ROS) untuk membangun paket-paket di dalam workspace.

```
$ sudo mkdir src
$ ls
```

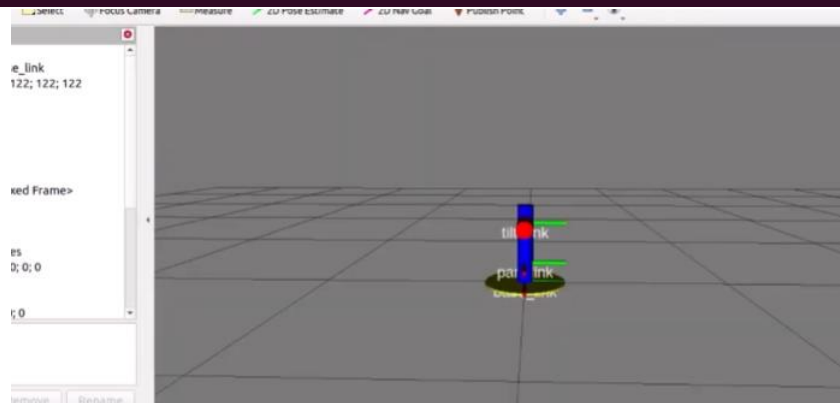
Kemudian jalankan perintah sudo mkdir src lalu perintah ls

Dalam konteks Robot Operating System (ROS), terdapat perintah-perintah penting yang mendukung pengembangan dan manajemen paket. Pertama, catkin\_make digunakan untuk membangun paket-paket ROS di dalam workspace. Workspace sendiri adalah struktur direktori yang mengelola berbagai paket ROS yang sedang Anda kembangkan. Sebagai contoh, dengan menjalankan catkin\_make di direktori workspace, kita dapat membangun semua paket yang ada di dalamnya.

Dengan menggunakan perintah-perintah ini, pengembang dapat membangun, membuat, dan mengelola paket ROS dengan lebih efisien, mempercepat proses pengembangan robotika di dalam lingkungan ROS

## Working with ROS for 3D Modeling

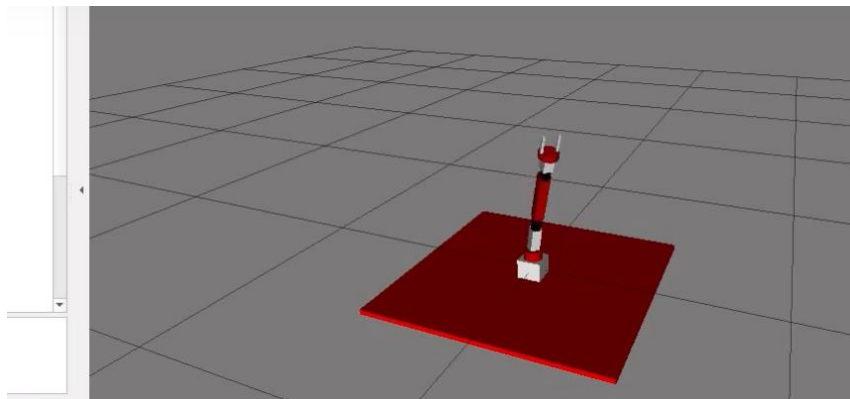
```
agung@agung-VirtualBox:~$ roslaunch mastering_ros_robot_description_pkg view_demo.launch
```



Roslaunch `mastering_ros_robot_description_pkg view_demo.launch` untuk meluncurkan sebuah paket ros menggunakan perintah `roslaunch`  
`roslaunch`: Ini adalah perintah untuk meluncurkan file launch ROS.

```
agung@agung-VirtualBox:~$ roslaunch mastering_ros_robot_description_pkg view_arm.launch
```

Jalankan perintah `roslaunch mastering_ros_robot_description_pkg view_arm.launch`  
`mastering_ros_robot_description_pkg`: Nama paket ROS yang akan diluncurkan.  
`view_arm.launch`: Nama file launch yang akan digunakan. File ini berisi konfigurasi untuk memulai satu atau lebih node ROS dan dapat menyertakan parameter serta argumen yang diperlukan.



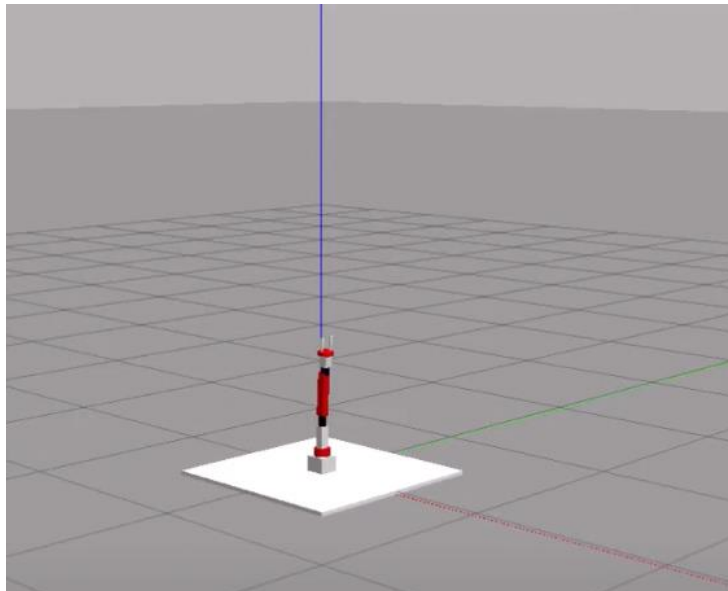
## SEVEN DOF ARM GAZEBO

```
agung@agung-VirtualBox:~$ roslaunch seven_dof_arm_gazebo seven_dof_arm_gazebo_control.launch
```

Perintah "roslaunch mastering\_ros\_robot\_description\_pkg view\_arm.launch" merupakan perintah dalam lingkungan ROS (Robot Operating System) untuk meluncurkan (launch) konfigurasi yang terdapat dalam file view\_arm.launch dari paket mastering\_ros\_robot\_description\_pkg

roslaunch: Ini adalah perintah utama untuk meluncurkan file konfigurasi ROS. Perintah ini memungkinkan Anda meluncurkan konfigurasi yang telah ditentukan sebelumnya dalam file launch.

mastering\_ros\_robot\_description\_pkg: Merupakan nama paket ROS yang berisi file launch yang akan dijalankan. Paket ROS adalah cara organisasi dalam ROS, yang mengelompokkan file dan kode terkait.



```
agung@agung-VirtualBox:~$ roslaunch diff_wheeled_robot_gazebo diff_wheeled_gazebo_full.launch
```

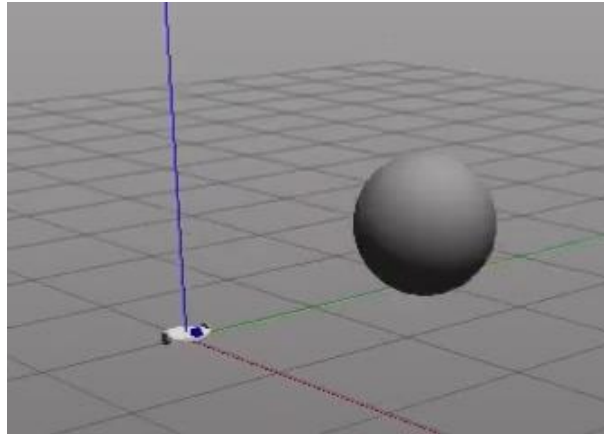
Perintah "roslaunch diff\_wheeled\_robot\_gazebo diff\_wheeled\_gazebo\_full.launch" digunakan untuk meluncurkan simulasi robot roda berbeda pada lingkungan Gazebo dalam lingkungan ROS.

roslaunch: Ini adalah perintah utama untuk meluncurkan file konfigurasi ROS. Perintah ini memungkinkan Anda meluncurkan konfigurasi yang telah ditentukan sebelumnya dalam file launch.

diff\_wheeled\_robot\_gazebo: Ini adalah nama paket ROS yang berisi file launch yang akan dijalankan. Paket ini mungkin berisi konfigurasi dan model robot untuk simulasi di Gazebo.

diff\_wheeled\_gazebo\_full.launch: Merupakan nama file launch yang akan dijalankan. File ini kemungkinan besar berisi konfigurasi untuk meluncurkan simulasi Gazebo untuk robot roda berbeda. Ini termasuk mengatur model robot, lingkungan simulasi, dan node-node ROS yang diperlukan.

view\_arm.launch: Merupakan nama file launch yang akan dijalankan. File ini biasanya berisi konfigurasi untuk memulai satu atau lebih node ROS, dan mungkin termasuk parameter, argumen, atau konfigurasi lainnya yang diperlukan untuk menjalankan komponen atau robot tertentu.



Simulasi robot dengan ROS dan Gazebo adalah praktek umum dalam pengembangan robotika. Gazebo, sebagai simulator yang kuat, memungkinkan pengguna untuk mensimulasikan dinamika robot dan interaksinya dengan lingkungan sekitarnya.

Proses dimulai dengan instalasi ROS dan Gazebo, diikuti dengan pembuatan workspace dan instalasi paket-paket yang diperlukan. Setelahnya, pengguna dapat membuat paket ROS untuk robot, mendefinisikan model URDF, dan menjalankan Gazebo dengan ROS untuk memulai simulasi. Kontrol robot dapat diimplementasikan dengan ROS controllers

visualisasi tambahan dapat dilakukan menggunakan RViz. Selanjutnya, pengguna dapat berinteraksi dengan simulasi, mempublikasikan topik, dan menguji algoritma kontrol sebelum diterapkan pada robot fisik.

Keseluruhannya, penggunaan ROS dan Gazebo mempermudah pengembangan robotika dengan menyediakan lingkungan simulasi yang kuat dan efisien.

## **Simulating Robots Using ROS, CoppeliaSim, and Webots**

Simulasi Robot menggunakan Robot Operating System (ROS), CoppeliaSim, dan Webots melibatkan pendekatan komprehensif dan serbaguna dalam pengembangan dan pengujian robot. ROS, sebagai middleware framework, memfasilitasi integrasi berbagai komponen perangkat lunak, memungkinkan komunikasi yang mulus antara robot dan periferalnya. CoppeliaSim, perangkat lunak simulasi robot yang powerful, menyediakan lingkungan yang realistis dan interaktif untuk merancang, menguji, dan menyempurnakan algoritma dan strategi kontrol robot. Selain itu, Webots, platform simulasi yang cukup dikenal, menawarkan antarmuka yang ramah pengguna untuk mensimulasikan dan memvalidasi sistem robotik dalam berbagai skenario. Pemanfaatan bersama ROS, CoppeliaSim, dan Webots ini memberdayakan para peneliti dan pengembang untuk dengan efisien membuat prototipe, mendebug, dan mengoptimalkan aplikasi robotik, mendorong kemajuan di bidang robotika dan otomasi.

masukan perintah `cd dev/coppeliaSim/` lalu jalankan perintah `./coppeliaSim.sh`

Perintah "`cd dev/coppelasim/`" adalah perintah untuk berpindah ke direktori (change directory) di sistem file. Mari kita breakdown perintah tersebut:

`cd`: Ini adalah singkatan dari "change directory," yang digunakan untuk berpindah dari satu direktori ke direktori lainnya.

`dev/coppelasim/`: Merupakan path atau alamat direktori yang akan menjadi destinasi perpindahan. Dalam hal ini, perintah tersebut mencoba untuk berpindah ke direktori "coppelasim" yang berada dalam direktori "dev."

Perintah `./coppeliaSim.sh` digunakan untuk menjalankan skrip shell (bash script) bernama "coppeliaSim.sh" yang terletak dalam direktori saat ini.

