

Agung Aji Saputra

1103202114

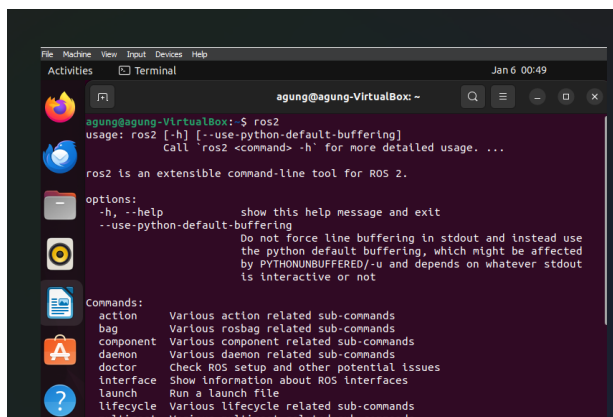
## UAS Robotika\_Technical Report

### 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 pengembangan robot.

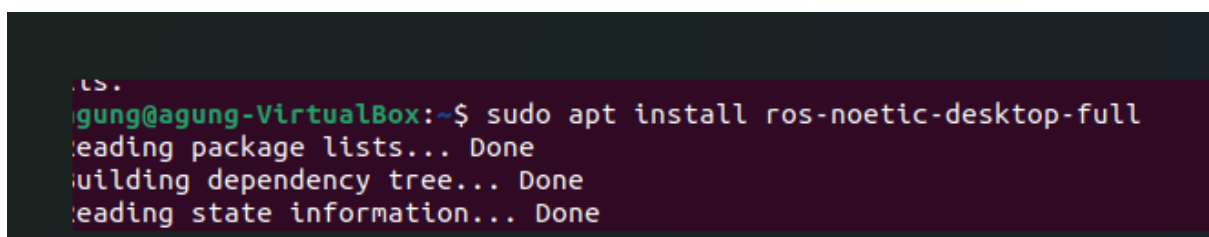
Dengan memanfaatkan ROS, pengembang dapat merancang dan mengintegrasikan berbagai komponen perangkat lunak dengan lebih mudah, mulai dari sensor hingga aktuator, kontroler, dan sistem pemrosesan data.

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: ~  
$ 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
```

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
```

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

## SEVEN DOF ARM GAZEBO

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, dan

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.