

Technical Report from Mastering ROS
Final Examination Robot Autonomy



Oleh:

Nama : Alifia Mutiara Rahma
NIM : 1103200025

PROGRAM STUDI TEKNIK KOMPUTER
FAKULTAS TEKNIK ELEKTRO
UNIVERSITAS TELKOM
2023

Chapter 1 – Introduction to ROS

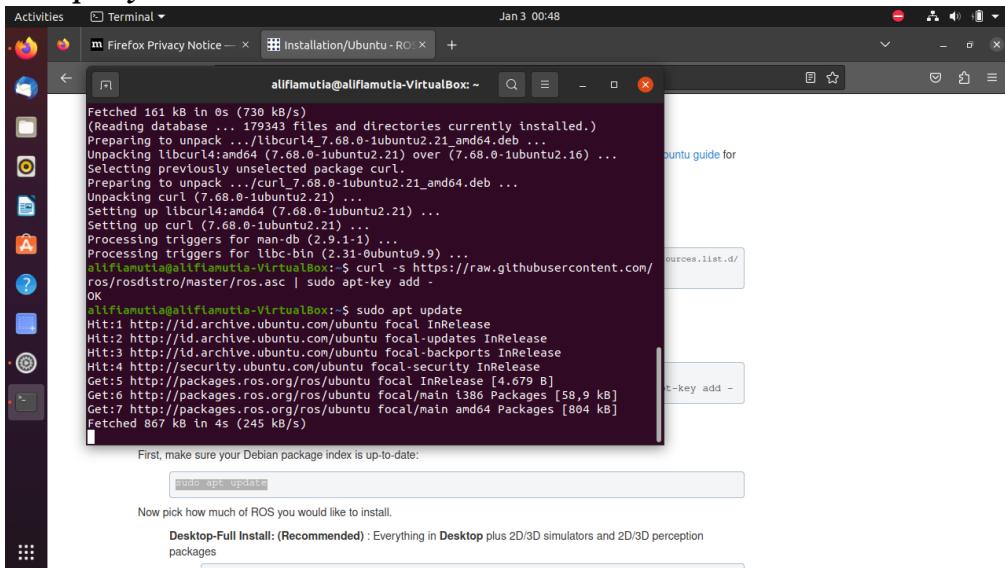
Persiapan Awal

Sebelum memulai dengan ROS dan mencoba kode dalam buku ini, berikut adalah persyaratan awal yang harus dipenuhi:

- Ubuntu 20.04 LTS/Debian 10: ROS secara resmi didukung oleh sistem operasi Ubuntu dan Debian. Kami lebih suka menggunakan versi LTS dari Ubuntu; yaitu, Ubuntu 20.04.
- Instalasi Penuh Desktop ROS Noetic: Instalasi lengkap desktop ROS harus dipasang. Versi yang kami pilih adalah ROS Noetic, yang merupakan versi stabil terbaru. Petunjuk instalasi untuk distribusi ROS terbaru dapat ditemukan di tautan berikut: <http://wiki.ros.org/noetic/Installation/Ubuntu>. Paket ros-noetic-desktop-full dari daftar repositori harus dipilih.

Berikut urutan yang dilakukan:

- **Set-up keys**

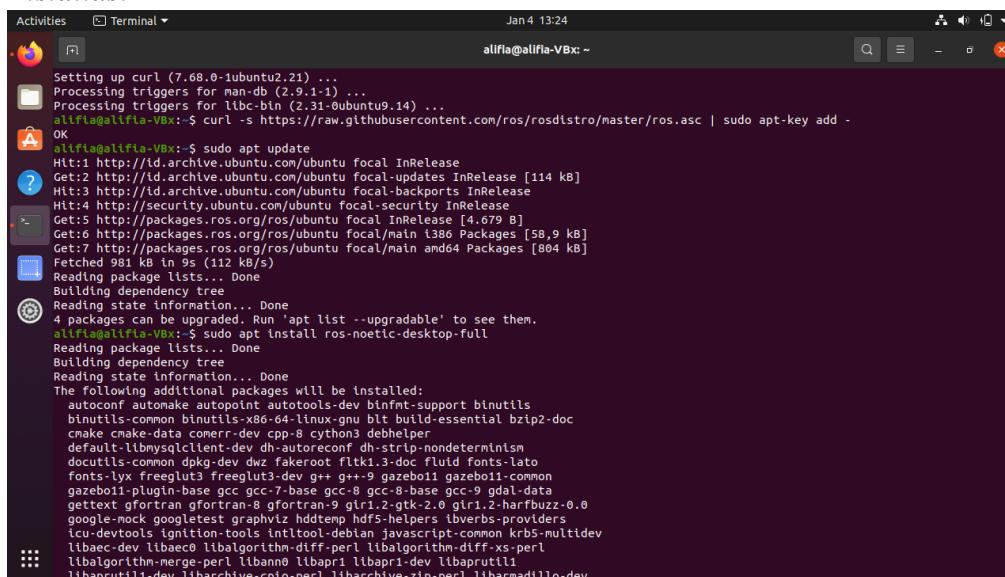


```
Fetched 161 kB in 0s (730 kB/s)
(Reading database ... 179343 files and directories currently installed.)
Preparing to unpack .../libcurl4_7.68.0-1ubuntu2.21_amd64.deb ...
Unpacking libcurl4:amd64 (7.68.0-1ubuntu2.21) over (7.68.0-1ubuntu2.16) ...
Selecting previously unselected package curl.
Preparing to unpack .../curl_7.68.0-1ubuntu2.21_amd64.deb ...
Unpacking curl (7.68.0-1ubuntu2.21) ...
Setting up libcurl4:amd64 (7.68.0-1ubuntu2.21) ...
Setting up curl (7.68.0-1ubuntu2.21) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.9) ...
alifl@alifl-VirtualBox:~$ curl -s https://raw.githubusercontent.com/
ros/rosdistro/master/ros.asc | sudo apt-key add -
OK
alifl@alifl-VirtualBox:~$ sudo apt update
Hit:1 http://id.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://id.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://id.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:5 http://packages.ros.org/ros/ubuntu focal InRelease [4.679 B]
Get:6 http://packages.ros.org/ros/ubuntu focal/main i386 Packages [58,9 kB]
Get:7 http://packages.ros.org/ros/ubuntu focal/main amd64 Packages [804 kB]
Fetched 867 kB in 4s (245 kB/s)

First, make sure your Debian package index is up-to-date:
sudo apt update

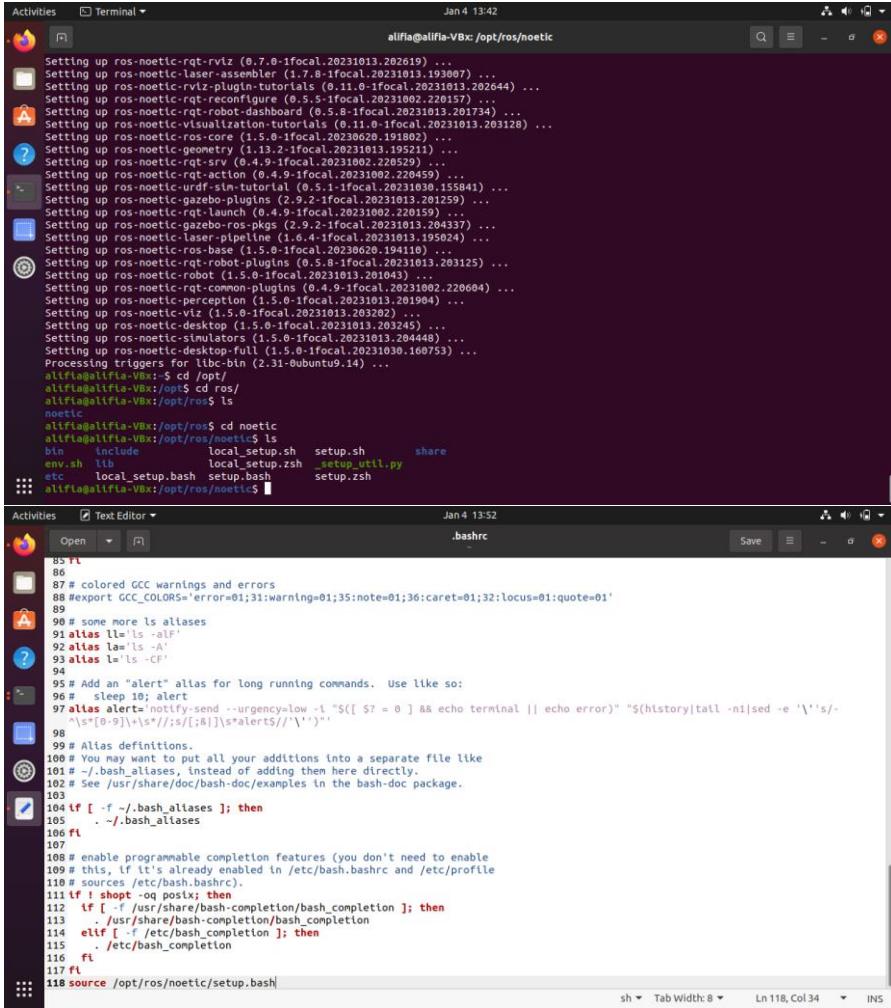
Now pick how much of ROS you would like to install.
Desktop-Full Install: (Recommended) : Everything in Desktop plus 2D/3D simulators and 2D/3D perception
packages
```

- **Installasi**



```
Setting up curl (7.68.0-1ubuntu2.21) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.14) ...
alifl@alifl-VBX:~$ curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
OK
alifl@alifl-VBX:~$ sudo apt update
Hit:1 http://id.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://id.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Hit:3 http://id.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:5 http://packages.ros.org/ros/ubuntu focal InRelease [4.679 B]
Get:6 http://packages.ros.org/ros/ubuntu focal/main i386 Packages [58,9 kB]
Get:7 http://packages.ros.org/ros/ubuntu focal/main amd64 Packages [804 kB]
Fetched 981 kB in 9s (112 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
4 packages can be upgraded. Run 'apt list --upgradable' to see them.
alifl@alifl-VBX:~$ sudo apt install ros-noetic-desktop-full
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  autoconf automake autopoint autotools-dev binfmt-support binutils
  binutils-common binutils-x86_64-linux-gnu bitt build-essential bztp2-doc
  cmake cmake-data comerr-dev cpp-8 cython3 debhelper
  default-libmysqlclient-dev dh-autoreconf dh-strip-nondeterminism
  docutils-common dpkg-dev dwz fakeroot fltk1.3-doc fluid fonts-lato
  fonts-lyx freetype3-dev g++ g++-9 gazebo11 gazebo11-common
  gazebo11-plugin-base gcc gcc-7-base gcc-8 gcc-8-base gcc-9 galil-data
  gettext gfortran gfortran-8 gfortran-9 gir1.2-gtk-2.0 gir1.2-harfbuzz-0.0
  google-mock googletest graphviz hdf5-helpers iverbs-providers
  iuc-devtools ignition-tools intltool-debian javascript-common krb5-multidev
  libaec-dev libaec0 libalgorthm-diff-perl libalgorthm-diff-xs-perl
  libalgorthm-merge-perl libbanjo libbapr1 libbapr1-dev libbaprutil1
  libbaprutil1-dev libarchive-cpio-perl libarchive-zip-perl libarmadillo-dev
```

- **Environment Setup**



```

Setting up ros-noetic-rqt-rviz (0.7.0-1focal.20231013.202619) ...
Setting up ros-noetic-laser-assembler (1.7.8-1focal.20231013.193007) ...
Setting up ros-noetic-rviz-plugin-tutorials (0.11.0-1focal.20231013.202644) ...
Setting up ros-noetic-rqt-reconfigure (0.5.5-1focal.20231002.220157) ...
Setting up ros-noetic-rqt-viz-visualization (0.5.2-1focal.20231013.2023128) ...
Setting up ros-noetic-ros-control (0.5.0-1focal.20230628.191802) ...
Setting up ros-noetic-geometry (1.13.2-1focal.20231013.195211) ...
Setting up ros-noetic-rqt-srv (0.4.9-1focal.20231002.220529) ...
Setting up ros-noetic-rqt-action (0.4.9-1focal.20231002.220459) ...
Setting up ros-noetic-urdf-sim-tutorial (0.5.1-1focal.20231030.155841) ...
Setting up ros-noetic-gazebo-plugins (2.9.2-1focal.20231013.201259) ...
Setting up ros-noetic-rqt-launch (0.4.9-1focal.20231002.220159) ...
Setting up ros-noetic-gazebo-ros-pkgs (2.9.2-1focal.20231013.204337) ...
Setting up ros-noetic-laser-pipeline (1.6.0-1focal.20231013.195024) ...
Setting up ros-noetic-robot (1.5.0-1focal.20231002.220459) ...
Setting up ros-noetic-rqt-common-plugins (0.4.9-1focal.20231002.220604) ...
Setting up ros-noetic-perception (1.5.0-1focal.20231013.201964) ...
Setting up ros-noetic-viz (1.5.0-1focal.20231013.203202) ...
Setting up ros-noetic-desktop (1.5.0-1focal.20231013.203245) ...
Setting up ros-noetic-simulators (1.5.0-1focal.20231013.204448) ...
Setting up ros-noetic-desktop-full (1.5.0-1focal.20231030.166753) ...
Processing triggers for libc-bin (2.31-0ubuntu9.14) ...
alifia@alifia-VBx:~$ cd /opt/
alifia@alifia-VBx:~/opt$ cd ros/noetic/
alifia@alifia-VBx:~/opt/ros/noetic$ ls
noetic
alifia@alifia-VBx:~/opt/ros$ cd noetic
alifia@alifia-VBx:~/opt/ros/noetic$ ls
bin   include    local_setup.sh  setup.sh      share
env.sh  lib       local_setup.zsh  _setup_util.py
etc   local_setup.bash  setup.bash  setup.zsh
alifia@alifia-VBx:~/opt/ros/noetic$ 
```

```

87 # colored GCC warnings and errors
88 #export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'
89
90 # some more ls aliases
91 alias ll='ls -alF'
92 alias la='ls -A'
93 alias l='ls -CF'
94
95 # Add an "alert" alias for long running commands.  Use like so:
96 #   sleep 10; alert
97 alias alert='notify-send --urgency=low -L "$([ ${+_RETVAL_} = 0 ] && echo terminal || echo error)" "$(history|tail -n1|sed -e '\''s/^\s*[0-9]+\+\s*//;s/[[:digit:]]*\s*alert$/\''')"
98
99 # Alias definitions.
100 # You may want to put all your additions into a separate file like
101 #   ~/.bash_aliases, instead of adding them here directly.
102 # See /usr/share/doc/bash-doc/examples in the bash-doc package.
103
104 if [ -f ./bash_aliases ]; then
105     . ./bash_aliases
106 fi
107
108 # enable programmable completion features (you don't need to enable
109 # this, if it's already enabled in /etc/bashrc and /etc/profile
110 if [ -z "${BASH_SOURCE:-}" ]; then
111     if [ -f /usr/share/bash-completion/bash_completion ]; then
112         . /usr/share/bash-completion/bash_completion
113     elif [ -f /etc/bash_completion ]; then
114         . /etc/bash_completion
115     fi
116 fi
117
118 source /opt/ros/noetic/setup.bash
```

ROS distributions

Pembaruan ROS dirilis bersama dengan distribusi ROS yang baru. Distribusi ROS yang baru terdiri dari versi terbaru perangkat lunak intinya dan sejumlah paket ROS baru/diperbarui. ROS mengikuti siklus rilis yang sama dengan distribusi Linux Ubuntu: versi baru ROS dirilis setiap 6 bulan. Umumnya, untuk setiap versi Ubuntu LTS, versi ROS LTS dirilis.

Distro	Release date	Poster	Tuturte, turtle in tutorial	EOL date
ROS Noetic Ninjemy	May 23rd, 2020			May, 2025 (Focal EOL)
ROS Melodic Morenia	May 23rd, 2018			May, 2023 (Bionic EOL)
ROS Lunar Loggerhead	May 23rd, 2017			May, 2019
ROS Kinetic Kame	May 23rd, 2016			April, 2021 (Xenial EOL)

Running the ROS master and the ROS parameter server

Sebelum menjalankan node-node ROS apa pun, ROS master dan parameter server ROS harus diaktifkan. ROS master dan parameter server ROS dapat diaktifkan menggunakan satu perintah yang disebut roscore, yang akan memulai program-program berikut:

- ROS master
- Parameter server ROS
- Node logging rosout

Node rosout akan mengumpulkan pesan log dari node-node ROS lainnya dan menyimpannya dalam file log, serta meneruskan pesan log yang terkumpul ke topik lain. Topik /rosout diterbitkan oleh node-node ROS menggunakan perpustakaan klien ROS seperti roscpp dan rospy, dan topik ini disubscribe oleh node rosout, yang meneruskan pesan tersebut ke topik lain yang disebut /rosout_agg. Topik ini berisi aliran pesan log yang terkumpul. Perintah roscore harus dijalankan sebagai prasyarat sebelum menjalankan node-node ROS apa pun. Tangkapan layar berikut menunjukkan pesan-pesan yang dicetak ketika perintah roscore dijalankan dalam Terminal.

Roscore

```
Activities Terminal Jan 4 13:45
noetic
alifia@alifia-VBx:/opt/ros/noetic$ cd noetic
alifia@alifia-VBx:/opt/ros/noetic$ ls
bin include local_setup.sh setup.sh share
env.sh lib local_setup.zsh _setup_util.py
etc local_setup.bash setup.bash setup.zsh
alifia@alifia-VBx:/opt/ros/noetic$ source /opt/ros/noetic/setup.bash
alifia@alifia-VBx:/opt/ros/noetic$ cd
alifia@alifia-VBx:~$ roscore
... logging to /home/alifia/.ros/log/c71ac758-aacc-11ee-8d08-7ff22a3dde95/roslaunch-alifia-VBx-33025.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

[roscore] started roslaunch server http://alifia-VBx:42619/
ros_comm version 1.16.0

SUMMARY
=====
PARAMETERS
* /rosdistro: noetic
* /rosversion: 1.16.0

NODES
auto[starting new master
process[master]: started with pid [33039]
ROS_MASTER_URI=http://alifia-VBx:11311/
setting /run_id to c71ac758-aacc-11ee-8d08-7ff22a3dde95
process[rosout-1]: started with pid [33049]
started core service [/rosout]
```

Berikut adalah penjelasan setiap bagian saat menjalankan perintah roscore di Terminal:

1. Pada bagian 1, dapat diamati bahwa sebuah file log dibuat di dalam folder `~/.ros/log` untuk mengumpulkan log dari node-node ROS. File ini dapat digunakan untuk tujuan debugging.
2. Pada bagian 2, sebuah file peluncuran ROS yang disebut `roscore.xml` dimulai oleh perintah. Ketika sebuah file peluncuran dimulai, rosmaster dan server parameter ROS secara otomatis dimulai. Perintah `roslaunch` adalah skrip Python yang dapat memulai rosmaster dan server parameter ROS setiap kali mencoba menjalankan file peluncuran. Alamat server parameter ROS di dalam port ditunjukkan oleh bagian ini.
3. Pada bagian 3, parameter seperti `rosdistro` dan `rosversion` ditampilkan di Terminal. Parameter-parameter ini ditampilkan saat menjalankan `roscore.xml`. Rincian lebih lanjut tentang `roscore.xml` akan dibahas pada bagian berikutnya.
4. Pada bagian 4, terlihat bahwa node rosmaster dimulai dengan menggunakan `ROS_MASTER_URI`, yang telah ditentukan sebelumnya sebagai variabel lingkungan.

5. Pada bagian 5, terlihat bahwa node rosout dimulai, yang akan memulai berlangganan ke topik /rosout dan mengirimkannya kembali ke /rosout_agg.

Summary

Framework perangkat lunak ROS saat ini semakin populer di kalangan ahli robotika. Pengetahuan tentang ROS menjadi semakin penting bagi mereka yang berencana membangun karir sebagai insinyur robotika dalam beberapa tahun mendatang. Dalam bab ini, dasar-dasar ROS telah dibahas, terutama dengan tujuan menyegarkan pemahaman konsep jika sebelumnya Anda sudah familiar dengan ROS. Pentingnya mempelajari ROS dan keunggulannya di antara platform perangkat lunak robotika saat ini telah dibahas. Konsep dasar seperti ROS master dan parameter server telah dijelajahi, serta penjelasan tentang cara kerja roscore telah diberikan. Pada bab berikutnya, manajemen paket ROS akan diperkenalkan dan beberapa contoh praktis dari sistem komunikasi ROS akan dibahas.

Daftar Pustaka

L. Joseph dan J. Cacace, Mastering ROS for Robotics Programming Third Edition, Birmingham: Packt Publishing Ltd, 2021.

Chapter 2 – Getting Started with ROS Programming

Creating a ROS Package

Paket ROS diciptakan untuk menjadi unit dasar dalam program-program ROS. Proses tersebut mencakup pembuatan, pembangunan, dan pelepasan paket ROS ke publik. Distribusi ROS yang sedang digunakan adalah Noetic Ninjemys, dan sistem pembangunan catkin digunakan untuk membangun paket-paket ROS. Sistem pembangunan bertanggung jawab untuk menghasilkan target (eksekutabel/perpustakaan) dari kode sumber teks yang dapat digunakan oleh pengguna akhir. Pada distribusi ROS yang lebih tua, seperti Electric dan Fuerte, sistem pembangunan yang digunakan adalah rosbuild. Karena kekurangan-kekurangan rosbuild, catkin diperkenalkan. Ini juga memungkinkan untuk mendekatkan sistem kompilasi ROS ke Cross Platform Make (CMake). Pergeseran ini menawarkan keunggulan, seperti kemudahan porting paket ke sistem operasi yang berbeda, termasuk Windows, selama sistem operasi tersebut mendukung CMake dan Python.

Langkah awal dalam bekerja dengan paket-paket ROS adalah mendirikan workspace catkin ROS. Setelah ROS terinstal, workspace catkin yang disebut `catkin_ws` dapat dibuat dan dibangun dengan langkah-langkah berikut:

```
Activities Terminal Jan 4 19:49
alifia@alifia-VBx: ~$ mkdir -p ~/catkin_ws/src
alifia@alifia-VBx: ~$ source /opt/ros/noetic/setup.bash
alifia@alifia-VBx: ~$ cd ~/catkin_ws/src
alifia@alifia-VBx: ~/catkin_ws/src$ catkin init_workspace
Creating symlink "/home/alifia/catkin_ws/src/MakeLists.txt" pointing to "/opt/ros/noetic/share/catkin/cmake/toplevel.cmake"
alifia@alifia-VBx: ~/catkin_ws/src$ catkin_make
Base path: /home/alifia/catkin_ws/src
The specified source space "/home/alifia/catkin_ws/src/src" does not exist
alifia@alifia-VBx: ~/catkin_ws/src$ cd ~/catkin_ws
alifia@alifia-VBx: ~/catkin_ws$ catkin_make
Base path: /home/alifia/catkin_ws
Source space: /home/alifia/catkin_ws/src
Build space: /home/alifia/catkin_ws/build
Devel space: /home/alifia/catkin_ws/devel
Install space: /home/alifia/catkin_ws/install
#####
#### Running command: "cmake /home/alifia/catkin_ws/src -DCATKIN_DEVEL_PREFIX=/home/alifia/catkin_ws/devel -DCMAKE_INSTALL_PREFIX=/home/alifia/catkin_ws/install -G Unix Makefiles" in "/home/alifia/catkin_ws/build"
#####
-- The C Compiler identification is GNU 9.4.0
-- The CXX Compiler identification is GNU 9.4.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Using CATKIN_DEVEL_PREFIX: /home/alifia/catkin_ws/devel
```

1. Buat direktori src dalam workspace catkin:

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

2. Agar workspace ini dapat dikompilasi, lingkungan ROS harus diaktifkan agar dapat mengakses fungsi-fungsi ROS:

source /opt/ros/noetic/setup.bash

3. Beralih ke folder src yang telah dibuat sebelumnya:

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

4. Inisialisasikan workspace catkin yang baru:

catkin init workspace

5. Workspace dapat dibangun meskipun tidak ada paket yang ada. Beralihlah ke folder workspace dengan perintah:

cd ~/catkin_ws

6. Gunakan perintah `catkin_make` untuk membangun workspace:

catkin make

Perintah ini akan membuat direktori `devel` dan `build` di dalam workspace catkin.

```

alifia@alifia-VBx: ~/catkin_ws
...
-- Using CATKIN_DEVEL_PREFIX: /home/alifia/catkin_ws/devel
-- Using CMAKE_PREFIX_PATH: /opt/ros/noetic
-- Using PythonInterp: /usr/bin/python3
-- Found PythonInterp: /usr/bin/python3 (found suitable version "3.8.10", minimum required is "3")
-- Using PYTHON_EXECUTABLE: /usr/bin/python3 (found version "3.8.10")
-- Using Debian Python package layout
-- Found PY_VENV: /usr/lib/python3/dist-packages/en.py
-- Using empty: /usr/lib/python3/dist-packages/en.py
-- Using enable_testing: ON
-- Using enable_cmake_testing: OFF
-- Using CATKIN_TEST_RESULTS_DIR: /home/alifia/catkin_ws/build/test_results
-- Forcing gtest/gmock from source, though one was otherwise available.
-- Found gtest sources under '/usr/src/googletest': gtests will be built
-- Found gmock sources under '/usr/src/googletest': gmock will be built
-- Found PythonInterp: /usr/bin/python3 (found version "3.8.10")
-- Found Threads: TRUE
-- Using Python nosetests: /usr/bin/nosetests3
catkin 0.8.10
-- BUILD_SHARED_LIBS is on
-- BUILD_SHARED_LIBS is on
-- Configuring done
-- Generating done
-- Build files have been written to: /home/alifia/catkin_ws/build
#####
## Running command: "make -j2 -l2" in "/home/alifia/catkin_ws/build"
#####
alifia@alifia-VBx:~/catkin_ws$ echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
alifia@alifia-VBx:~/catkin_ws$ source ~/.bashrc
Created file mastering_ros_demo_pkg/mastering_ros_demo_pkg.roscpp std_msgs actionlib actionlib_msgs
Created file mastering_ros_demo_pkg/CMakeLists.txt
Created folder mastering_ros_demo_pkg/include/mastering_ros_demo_pkg
Created folder mastering_ros_demo_pkg/src
Successfully created file in /home/alifia/catkin_ws/mastering_ros_demo_pkg. Please adjust the values in package.xml.
alifia@alifia-VBx:~/catkin_ws$
```

7. Berbagai file setup terletak di dalam folder `devel`. Untuk menambahkan workspace ROS yang dibuat ke lingkungan ROS, kita harus mendapatkan sumber salah satu dari file-file ini. Selain itu, kita dapat menggunakan perintah berikut untuk mendapatkan sumber file setup dari workspace ini setiap kali sesi bash baru dimulai:

```

echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

8. Setelah mengatur workspace catkin, dapat membuat paket kita sendiri yang memiliki node contoh untuk menunjukkan cara kerja topik, pesan, layanan, dan actionlib ROS. Perhatikan bahwa jika workspace tidak diatur dengan benar, maka tidak akan mungkin menggunakan perintah ROS apa pun. Gunakan perintah `catkin_create_pkg` untuk membuat paket ROS. Perintah ini digunakan untuk membuat paket kita, di mana kita akan membuat demo dari berbagai konsep ROS. Beralih ke folder `src` workspace catkin dan buat paket dengan perintah berikut:

```

catkin_create_pkg    mastering_ros_demo_pkg    roscpp    std_msgs    actionlib
actionlib_msgs
```

Folder kode sumber dari semua paket ROS, baik yang dibuat dari awal maupun diunduh dari repositori kode lain, harus ditempatkan dalam folder `src` workspace ROS. Jika tidak, paket tersebut tidak akan dikenali oleh sistem ROS dan tidak akan dikompilasi.

Setelah paket ini dibuat, bangunlah paket tersebut tanpa menambahkan node apapun dengan menggunakan perintah `catkin_make`. Perintah ini harus dijalankan dari path workspace catkin. Berikut adalah perintah untuk membangun paket ROS kosong kita:

```

cd ~/catkin_ws && catkin_make
```

```

alifia@alifia-VBx:~/catkin_ws
...
-- Found gtest sources under '/usr/src/googletest': gtests will be built
-- Found gmock sources under '/usr/src/googletest': gmock will be built
-- Found PythonInterp: /usr/bin/python3 (found version "3.8.10")
-- Using Threads: TRUE
-- Using Python nosetests: /usr/bin/nosetests3
catkin 0.8.10
-- BUILD_SHARED_LIBS is on
-- BUILD_SHARED_LIBS is on
-- Configuring done
-- Generating done
-- Build files have been written to: /home/alifia/catkin_ws/build
#####
## Running command: "make -j2 -l2" in "/home/alifia/catkin_ws/build"
#####
alifia@alifia-VBx:~/catkin_ws$ echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
alifia@alifia-VBx:~/catkin_ws$ source ~/.bashrc
alifia@alifia-VBx:~/catkin_ws$ catkin_create_pkg mastering_ros_demo_pkg roscpp std_msgs actionlib actionlib_msgs
Created file mastering_ros_demo_pkg/package.xml
Created file mastering_ros_demo_pkg/CMakeLists.txt
Created folder mastering_ros_demo_pkg/include/mastering_ros_demo_pkg
Created folder mastering_ros_demo_pkg/src
Successfully created file in /home/alifia/catkin_ws/mastering_ros_demo_pkg. Please adjust the values in package.xml.
alifia@alifia-VBx:~/catkin_ws$ cd ~/catkin_ws && catkin_make
Base path: /home/alifia/catkin_ws
Source space: /home/alifia/catkin_ws/src
Build space: /home/alifia/catkin_ws/build
Devel space: /home/alifia/catkin_ws/devel
Install space: /home/alifia/catkin_ws/install
#####
## Running command: "make cmake_check_build_system" in "/home/alifia/catkin_ws/build"
#####
#####
## Running command: "make -j2 -l2" in "/home/alifia/catkin_ws/build"
#####
alifia@alifia-VBx:~/catkin_ws$
```

Setelah berhasil dibangun, dapat dimulai penambahan node ke dalam folder `src` dari paket ini. Folder `build` dalam file CMake build umumnya berisi eksekutabel dari node yang ditempatkan di dalam folder `src` workspace catkin. Folder `devel` berisi skrip Bash, file header, dan eksekutabel dalam folder yang berbeda yang dihasilkan selama proses pembangunan. Proses pembuatan dan kompilasi node ROS menggunakan `catkin_make` telah dijelaskan. Mari sekarang bahas cara bekerja dengan topik ROS.

Building the Nodes

Langkah-langkah untuk mengedit dan membangun file CMakeLists.txt dalam paket adalah sebagai berikut:

1. File CMakeLists.txt dalam paket perlu diedit untuk mengompilasi dan membangun kode sumber. Beralihlah ke `mastering_ros_demo_pkg` untuk melihat file CMakeLists.txt yang ada.
2. Perintah `catkin_make` akan digunakan untuk membangun paket. Pertama, mari beralih ke workspace:

`cd ~/catkin_ws`

3. Bangun workspace ROS, termasuk `mastering_ros_demo_package`, dengan menjalankan perintah:

`catkin_make`

4. Dapat menggunakan perintah sebelumnya untuk membangun seluruh workspace atau menggunakan opsi `-DCATKIN_WHITELIST_PACKAGES`. Dengan opsi ini, kita dapat mengatur satu atau lebih paket untuk dikompilasi:

`catkin_make -DCATKIN_WHITELIST_PACKAGES="pkg1,pkg2,..."`

5. Perhatikan bahwa perlu untuk mengembalikan konfigurasi ini untuk mengompilasi paket atau seluruh workspace lainnya. Ini dapat dilakukan dengan menggunakan perintah berikut:

`catkin_make -DCATKIN_WHITELIST_PACKAGES=""`

6. Jika proses pembangunan telah selesai, dapat mengeksekusi node dengan `roscore`:

`roscore`

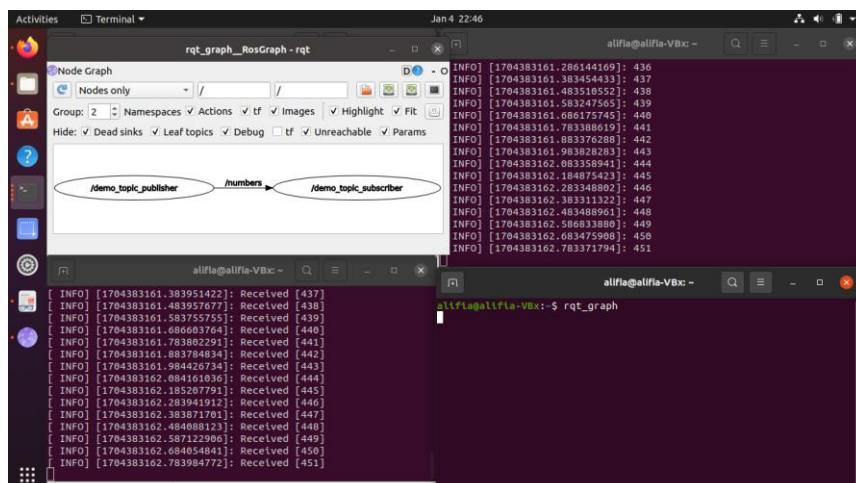
7. Selanjutnya, jalankan kedua perintah di dua shell berbeda. Di publisher yang berjalan, jalankan perintah berikut:

`rosrun mastering_ros_demo_package demo_topic_publisher`

8. Di subscriber yang berjalan, jalankan perintah berikut:

`rosrun mastering_ros_demo_package demo_topic_subscriber`

Berikut *output* yang dihasilkan dan *rqt graph*nya



Adding custom .msg and .srv files

Langkah-langkah untuk mengompilasi dan membangun paket setelah melakukan langkah-langkah sebelumnya adalah sebagai berikut:

1. Beralih ke direktori workspace catkin:

```
cd ~/catkin_ws/
```

2. Gunakan perintah `catkin_make` untuk mengompilasi dan membangun paket:

catkin make

3. Untuk memeriksa apakah pesan telah dibangun dengan benar, gunakan perintah rosmsg:

```
rosmmsg show mastering_ros_demo_pkg/demo_msg
```

Berikut *output*-nya:

```
Activities Terminal Jan 4 22:50
alifia@alifia-VBox:~/catkin_ws
```

```
[ 0%] Built target actionlib_msgs_generate_messages_cpp
[ 0%] Built target std_msgs_generate_messages_cpp
[ 0%] Built target _mastering_ros_demo_pkg_generate_messages check_deps_Demo_actionGoal
[ 0%] Built target _mastering_ros_demo_pkg_generate_messages_check_deps_Demo_actionActionResult
[ 0%] Built target _mastering_ros_demo_pkg_generate_messages_check_deps_Demo_actionFeedback
[ 0%] Built target _mastering_ros_demo_pkg_generate_messages_check_deps_Demo_actionGoalResult
[ 0%] Built target _mastering_ros_demo_msgs_generate_nodejs
[ 0%] Built target _mastering_ros_demo_pkg_generate_messages_check_deps_Demo_actionFeedback
[ 0%] Built target actionlib_msgs_generate_messages_py
[ 0%] Built target actionlib_msgs_generate_messages_nodejs
[ 0%] Built target std_msgs_generate_messages_py
[ 0%] Built target std_msgs_generate_messages_nodejs
[ 0%] Built target actionlib_msgs_generate_messages_py
[ 0%] Built target std_msgs_generate_messages_lisp
[ 0%] Built target actionlib_msgs_generate_messages_lisp
[ 0%] Built target actionlib_msgs_generate_messages_eus
[ 0%] Built target std_msgs_generate_messages_eus
[ 0%] Built target demo_action_server
[ 0%] Built target demo_topic_subscriber
[ 20%] Built target mastering_ros_demo_pkg_generate_messages_cpp
[ 37%] Built target mastering_ros_demo_pkg_generate_messages_py
[ 51%] Built target mastering_ros_demo_pkg_generate_messages_nodejs
[ 65%] Built target mastering_ros_demo_pkg_generate_messages_lisp
[ 85%] Built target mastering_ros_demo_pkg_generate_messages_eus
[ 87%] Built target demo_service_client
[ 87%] Built target demo_service_server
[ 99%] Built target demo_action_server
[ 93%] Built target demo_action_client
[ 96%] Built target demo_action_client
[100%] Built target demo_msg_publisher
[100%] Built target mastering_ros_demo_pkg_generate_messages
alifia@alifia-VBox:~/catkin_ws$ rosmsg show mastering_ros_demo_pkg/demo_nsg
string greeting
int32 number
```

Jika ingin menguji pesan khusus ini, buatlah publisher dan subscriber menggunakan tipe pesan khusus `demo_msg_publisher.cpp` dan `demo_msg_subscriber.cpp` masing-masing. Beralihlah ke folder `mastering_ros_demo_pkg` untuk mendapatkan kedua potongan kode tersebut.

Langkah-langkah untuk membangun paket menggunakan `catkin_make` dan menguji node adalah sebagai berikut:

- ### 1. Jalankan roscore:

roscore

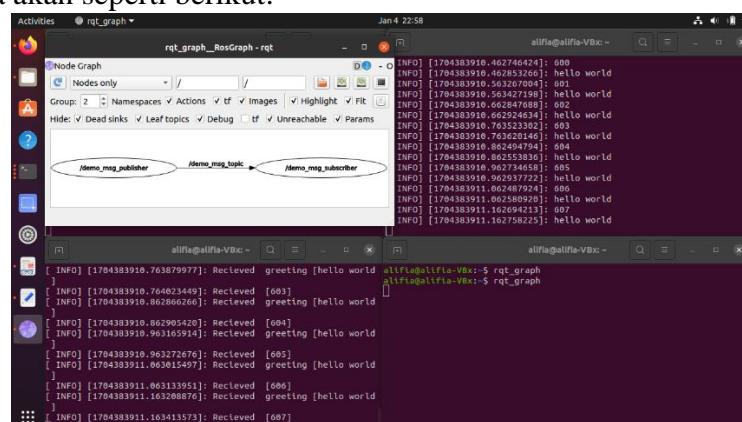
2. Mulai node penerbit pesan khusus;

```
rosrun mastering ros demo pkg demo msg publisher
```

3. Mulai node pelanggan pesan khusus:

```
roslaunch master master.launch
```

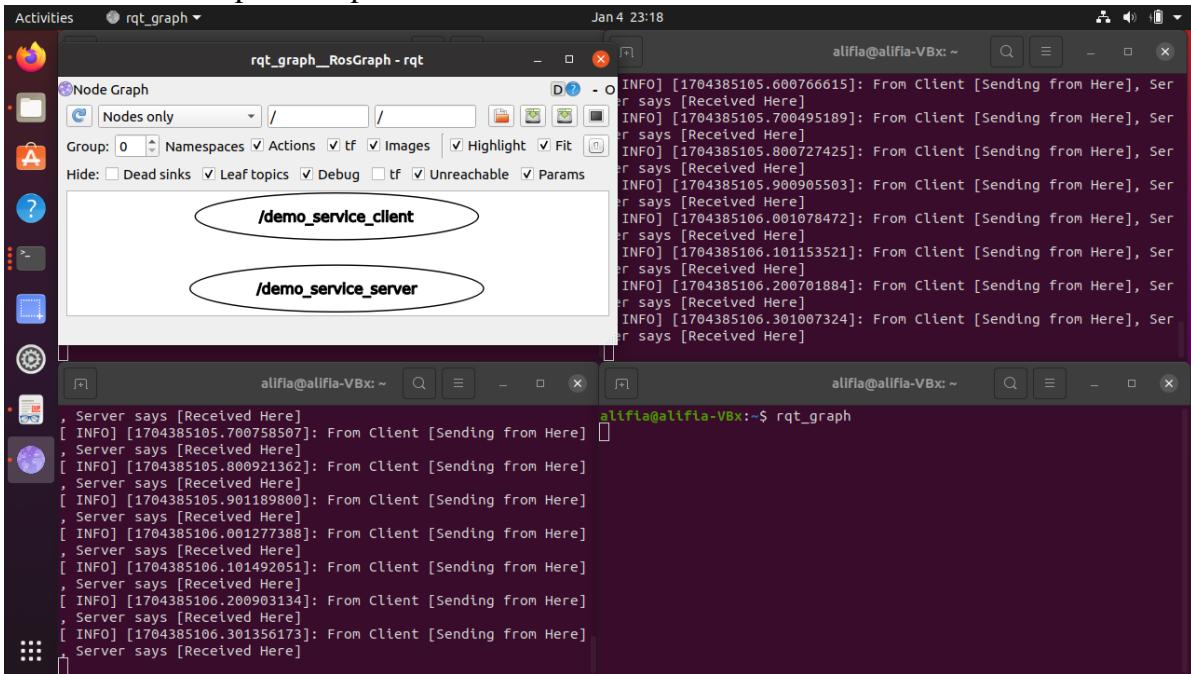
Node penerbit akan menerbitkan sebuah string bersama dengan sebuah bilangan bulat, sedangkan node pelanggan akan berlangganan ke topik tersebut dan mencetak nilai-nilainya. Hasil dan grafiknya akan seperti berikut:



Working with ROS Services

Langkah-langkah untuk menjalankan perintah-perintah dan membangun kode adalah sebagai berikut:

1. Buka terminal dan jalankan perintah berikut untuk pindah ke direktori workspace catkin:
`cd ~/catkin_ws`
2. Gunakan perintah `catkin_make` untuk membangun kode:
`catkin_make`
3. Untuk memulai node, pertama-tama jalankan roscore:
`roscore`
4. Selanjutnya, buka terminal lain dan gunakan perintah berikut untuk menjalankan node pelayan layanan (service server):
`rosrun mastering_ros_demo_pkg demo_service_server`
5. Buka terminal lainnya dan gunakan perintah berikut untuk menjalankan node pelanggan layanan (service client):
`rosrun mastering_ros_demo_pkg demo_service_client`
6. Pastikan perintah-perintah ini dijalankan setelah roscore.
7. Berikut hasil dari perintah-perintah:



Untuk berinteraksi dengan rosservice, gunakan perintah-perintah berikut:

- o `rosservice list`: Menampilkan daftar layanan ROS saat ini.
- o `rosservice type /demo_service`: Mencetak tipe pesan dari /demo_service.
- o `rosservice info /demo_service`: Mencetak informasi dari /demo_service.
- o `rosservice call /service_name service-args`: Memanggil server layanan dari baris perintah.

Aksi (action) juga merupakan elemen penting dalam ROS. Pada bagian selanjutnya, akan dipelajari cara menggunakan actionlib dalam sebuah node ROS untuk membuat node aksi/server.

Building the ROS Action Server and Client

Langkah-langkah untuk menjalankan node-node setelah menjalankan `catkin_make` adalah sebagai berikut:

1. Jalankan roscore:

roscore

2. Buka terminal baru dan jalankan node server aksi:

rosrun mastering_ros_demo_pkg demo_action_server

3. Buka terminal lainnya dan jalankan node klien aksi:

rosrun mastering_ros_demo_pkg demo_action_client 10 1

4. Pastikan perintah ini dijalankan setelah roscore.

Creating Launch Files

Langkah-langkah untuk membuat file peluncuran (launch file) `demo_topic.launch` di dalam folder paket `mastering_ros_demo_pkg` adalah sebagai berikut:

1. Terminal dibuka dan navigasi dilakukan ke direktori paket:

roscd mastering_ros_demo_pkg

2. Folder `launch` dibuat di dalam paket:

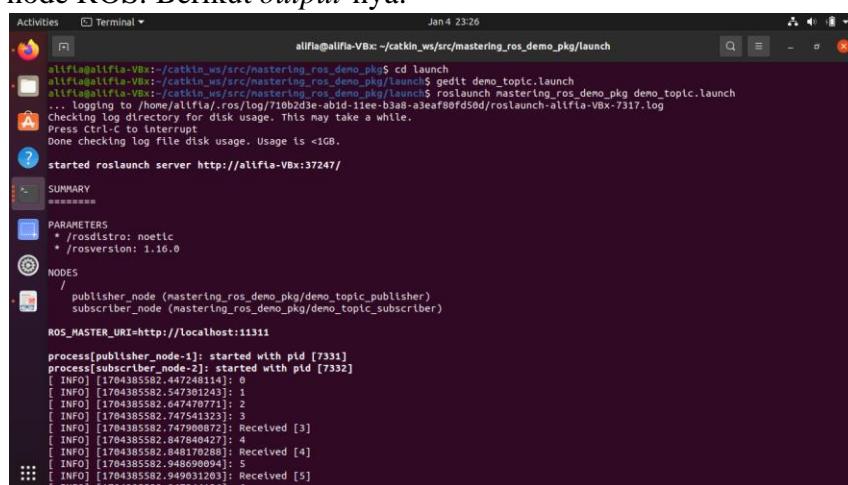
mkdir launch

3. Pindah ke dalam folder `launch`:

cd launch

4. Dengan menggunakan editor teks, seperti gedit, file `demo_topic.launch` dibuat dan dibuka:
gedit demo_topic.launch

Setelah itu, file `demo_topic.launch` dapat diisi dengan konfigurasi yang sesuai untuk meluncurkan node ROS. Berikut *output*-nya:



```
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_demo_pkg$ cd launch
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_demo_pkg$ gedit demo_topic.launch
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_demo_pkg$ roslaunch Mastering_ros_demo_pkg demo_topic.launch ...
... logging to /home/alifia/.ros/log/710b2d1e-ab1d-11ee-b1a8-a3eaef8fd5d4/roslaunch-alifia-VBx-7317.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.
started roslaunch server http://alifia-VBx:37247/
SUMMARY
========
PARAMETERS
  * /rostdistro: noetic
  * /rosversion: 1.16.0
NODES
/
  publisher_node (mastering_ros_demo_pkg/demo_topic_publisher)
  subscriber_node (mastering_ros_demo_pkg/demo_topic_subscriber)

ROS_MASTER_URI=http://localhost:11311

process[publisher_node-1]: started with pid [7331]
process[subscriber_node-2]: started with pid [7332]
[ INFO] [1704385582.447248114]: 0
[ INFO] [1704385582.547303243]: 1
[ INFO] [1704385582.647403243]: 2
[ INFO] [1704385582.747541323]: 3
[ INFO] [1704385582.747900872]: Received [3]
[ INFO] [1704385582.847840271]: 4
[ INFO] [1704385582.848170288]: Received [4]
[ INFO] [1704385582.948690094]: 5
[ INFO] [1704385582.949031203]: Received [5]
[ INFO] [1704385583.047344136]: 6
```

Berikut adalah langkah-langkah untuk memeriksa daftar node dan melihat pesan log menggunakan perintah-perintah yang disebutkan:

1. Periksa daftar node dengan perintah:

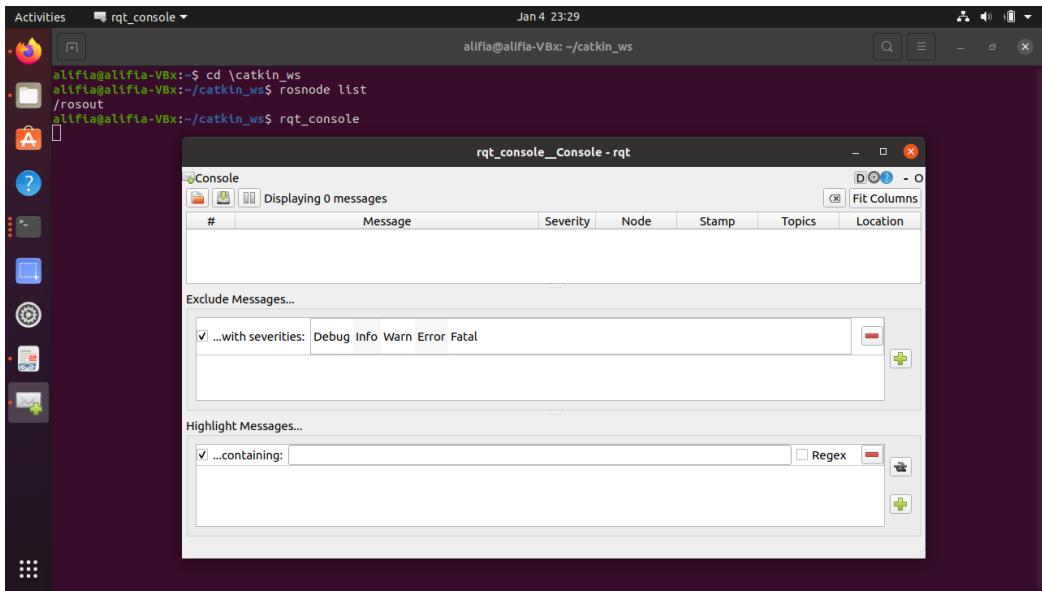
rosnode list

Perintah ini akan menampilkan daftar node yang sedang berjalan di sistem.

2. Buka GUI tool rqt_console untuk melihat pesan log:

rqt_console

Ini akan membuka antarmuka grafis yang memungkinkan untuk melihat pesan log yang dihasilkan oleh node-node yang berjalan.



Summary

Dalam bab ini, diberikan berbagai contoh dari node-node ROS di mana fitur-fitur ROS seperti topik ROS, layanan, dan aksi diimplementasikan. Alat-alat semacam itu digunakan dalam setiap paket ROS, baik yang sudah tersedia di repositori ROS maupun yang dibuat oleh Anda. Juga dibahas cara membuat dan mengompilasi paket ROS menggunakan pesan-pesan khusus dan standar. Umumnya, pesan-pesan khusus digunakan oleh paket-paket berbeda untuk mengelola data yang dihasilkan oleh node-node mereka, sehingga penting untuk dapat mengelola pesan-pesan khusus yang disediakan oleh sebuah paket.

Daftar Pustaka

L. Joseph dan J. Cacace, *Mastering ROS for Robotics Programming* Third Edition, Birmingham: Packt Publishing Ltd, 2021.

Chapter 3 – Working with ROS for 3D Modeling

Creating the ROS package for the robot description

Berikut adalah langkah-langkah dalam bentuk list untuk menciptakan paket ROS, menginstal paket yang diperlukan, dan menyiapkan struktur direktori sebelum membuat file URDF:

1. Membuat Paket ROS:

```
catkin_create_pkg mastering_ros_robot_description_pkg roscpp tf geometry_msgs  
urdf rviz xacro
```

2. Menginstal Paket yang Diperlukan:

```
sudo apt-get install ros-noetic-urdf  
sudo apt-get install ros-noetic-xacro
```

3. Clone Repositori Git untuk Paket:

```
git clone https://github.com/qboticslabs/mastering_ros_3rd_edition.git  
cd mastering_ros_robot_description_pkg/
```

4. Membuat Struktur Direktori:

```
mkdir urdf  
mkdir meshes  
mkdir launch
```

Setelah langkah-langkah ini, dapat melanjutkan dengan membuat file URDF dan file peluncuran di dalam direktori yang sesuai. Berikut *output*-nya:

The screenshot shows a terminal window titled "Terminal" with the command "alifia@alifia-VBx:~/catkin_ws/src\$". The user runs "catkin_create_pkg" to create a new package named "mastering_ros_robot_description_pkg". They then run "sudo apt-get install ros-noetic-urdf" and "sudo apt-get install ros-noetic-xacro" to install the required ROS packages. The terminal also shows the creation of directory structures for "urdf", "meshes", and "launch".

```
alifia@alifia-VBx:~/catkin_ws/src$ catkin_create_pkg mastering_ros_robot_description_pkg roscpp tf  
usage: catkin_create_pkg [-h] [--meta] [-s [SYS_DEPS [SYS_DEPS ...]]]  
                         [-b [BOOST_COMPONENTS [BOOST_COMPONENTS ...]]) [-v PKG_VERSION]  
                         [-D DESCRIPTION] [-l LICENSE] [-a AUTHOR]  
                         [-m MAINTAINER] [-r ROSDISTRO ROSDISTRO]  
                         name [dependencies [dependencies ...]]  
catkin_create_pkg: error: File exists: /home/alifia/catkin_ws/src/mastering_ros_robot_description_pkg/package.xml  
alifia@alifia-VBx:~/catkin_ws/src$ geometry_msgs urdf rviz xacro  
geometry_msgs: command not found  
alifia@alifia-VBx:~/catkin_ws/src$ sudo apt-get install ros-noetic-urdf  
[sudo] password for alifia:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
ros-noetic-urdf is already the newest version (1.13.2-1focal.20230620.185459).  
ros-noetic-urdf set to manually installed.  
0 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.  
alifia@alifia-VBx:~/catkin_ws/src$ sudo apt-get install ros-noetic-xacro  
[sudo] password for alifia:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
ros-noetic-xacro is already the newest version (1.14.16-1focal.20230620.185428).  
ros-noetic-xacro set to manually installed.  
0 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.  
alifia@alifia-VBx:~/catkin_ws/src$ cd Mastering_ros_robot_description_pkg/  
alifia@alifia-VBx:~/catkin_ws/src/Mastering_ros_robot_description_pkg$ cd src  
bash: cd: src: No such file or directory  
alifia@alifia-VBx:~/catkin_ws/src/Mastering_ros_robot_description_pkg$ cd .\.  
alifia@alifia-VBx:~/catkin_ws/src$
```

Explaining the URDF File

Berikut adalah langkah-langkah dalam bentuk list untuk mendefinisikan sambungan (joint) dalam URDF, menyimpan file URDF, dan memeriksa kemungkinan kesalahan:

1. Mendefinisikan Joint dalam URDF:

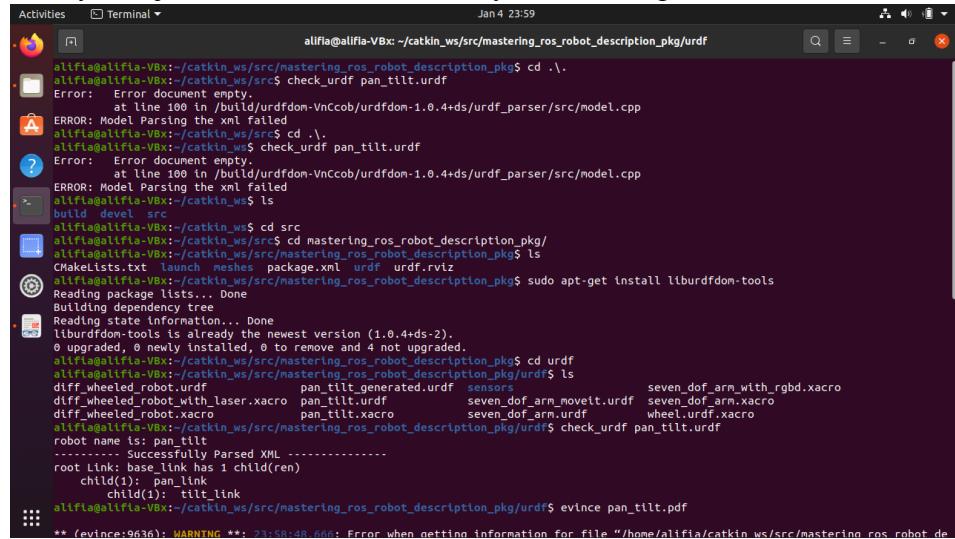
- Dalam potongan kode, tentukan sambungan dengan nama unik, jenis revolute, dan link induk serta link anaknya.
- Spesifikasikan posisi awal sambungan di dalam tag yang sesuai.

2. Simpan Kode URDF:

- Simpan kode URDF yang telah ditentukan sebagai pan_tilt.urdf.

3. Periksa Kesalahan pada File URDF:

- Gunakan perintah berikut untuk memeriksa apakah file URDF mengandung kesalahan:
check_urdf pan_tilt.urdf
- Pastikan bahwa paket liburdfdom-tools terinstal di sistem:
sudo apt-get install liburdfdom-tools
- Perintah `check_urdf` akan mem-parse tag URDF dan menunjukkan kesalahan jika ada. Jika semuanya berjalan baik, hasil keluarannya akan sebagai berikut:



```

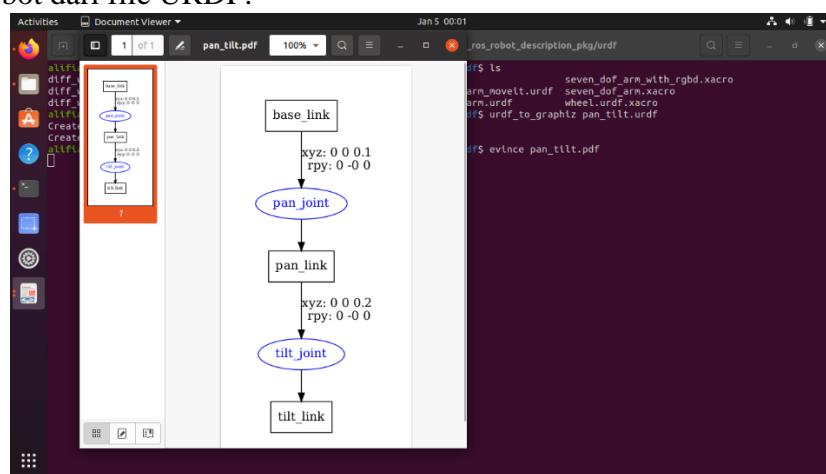
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_robot_description_pkg$ cd ..
alifia@alifia-VBx:~/catkin_ws/src$ check_urdf pan_tilt.urdf
Error: Error document empty.
at line 100 in /build/urdfdom-VnCcob/urdfdom-1.0.4+ds/urdf_parser/src/model.cpp
ERROR: Model Parsing the xml failed
alifia@alifia-VBx:~/catkin_ws/src$ cd ..
alifia@alifia-VBx:~/catkin_ws$ check_urdf pan_tilt.urdf
Error: Error document empty.
at line 100 in /build/urdfdom-VnCcob/urdfdom-1.0.4+ds/urdf_parser/src/model.cpp
ERROR: Model Parsing the xml failed
alifia@alifia-VBx:~/catkin_ws$ ls
build devel src
alifia@alifia-VBx:~/catkin_ws/src$ cd mastering_ros_robot_description_pkg/
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_robot_description_pkg$ ls
CMakeLists.txt launch meshes package.xml urdf urdf.viz
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_robot_description_pkg$ sudo apt-get install liburdfdom-tools
Reading package lists... Done
Building dependency tree
Reading state information... Done
liburdfdom-tools is already the newest version (1.0.4+ds-2).
0 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_robot_description_pkg$ cd urdf
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_robot_description_pkg/urdf$ ls
diff_wheeled_robot.urdf pan_tilt_generated.urdf sensors
diff_wheeled_robot_with_laser.xacro pan_tilt.urdf seven_dof_arm_moveit.urdf seven_dof_arm.xacro
diff_wheeled_robot.xacro pan_tilt.xacro seven_dof_arm.urdf wheel.urdf.xacro
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_robot_description_pkg/urdf$ check_urdf pan_tilt.urdf
robot name is: pan_tilt
----- Successfully Parsed XML -----
root Link: base_link has 1 child(ren)
    child(1): pan_link
        child(1): tilt_link
alifia@alifia-VBx:~/catkin_ws/src/mastering_ros_robot_description_pkg/urdf$ evince pan_tilt.pdf
** (evince:9636): WARNING **: 23:58:48.606: Error when getting information for file "/home/alifia/catkin_ws/src/mastering_ros_robot_de

```

Berikut adalah langkah-langkah dalam bentuk list untuk melihat struktur grafis link dan sambungan robot menggunakan perintah urdf_to_graphviz:

1. Gunakan Perintah urdf_to_graphviz:
 - Jalankan perintah berikut untuk melihat struktur grafis link dan sambungan robot:
urdf_to_graphviz pan_tilt.urdf
2. Perintah Menghasilkan Dua File:
 - Perintah tersebut akan menghasilkan dua file: pan_tilt.gv dan pan_tilt.pdf.
3. Lihat Struktur Robot:
 - Gunakan perintah berikut untuk melihat struktur robot menggunakan pembaca PDF, contohnya evince:
evince pan_tilt.pdf

Dengan melakukan langkah-langkah ini, dapat memvisualisasikan struktur grafis link dan sambungan robot dari file URDF:

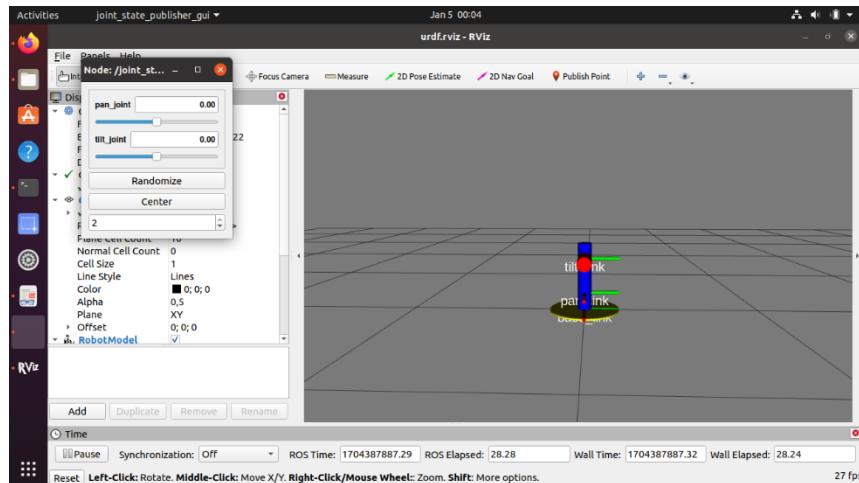


Visualizing the 3D Robot model in Rviz

Berikut adalah langkah-langkah dalam bentuk list dan kalimat pasif untuk meluncurkan model mekanisme pan dan tilt:

1. Lakukan peluncuran model dengan perintah:
`roslaunch mastering_ros_robot_description_pkg view_demo.launch`
2. Pastikan bahwa semuanya berfungsi dengan benar.
 - Jika semua berjalan dengan baik, mekanisme pan dan tilt akan terlihat di lingkungan Rviz.

Dengan melakukan langkah-langkah ini, model mekanisme pan dan tilt akan ditampilkan di Rviz:

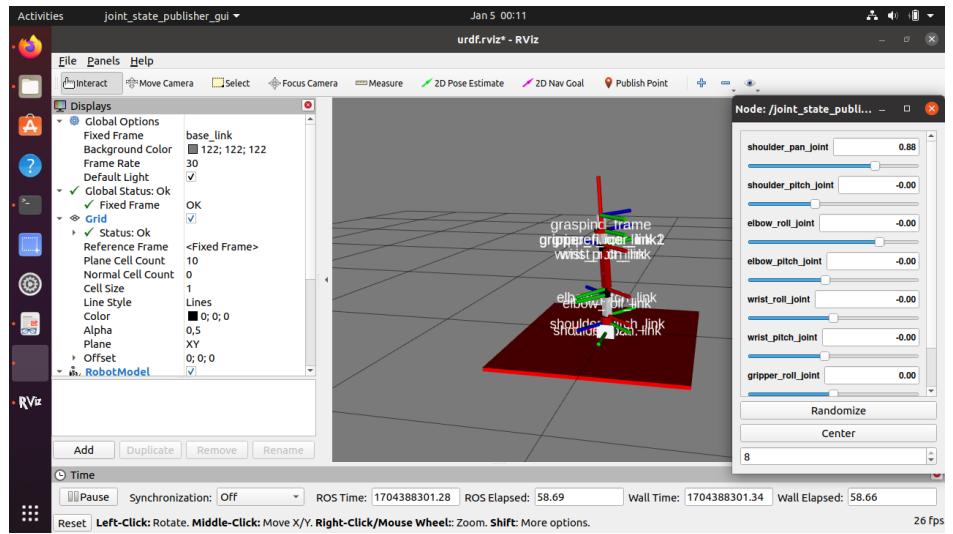


Viewing the seven-DOF arm in Rviz

Untuk mengimplementasikan petunjuk yang diberikan, langkah-langkah dapat diikuti sebagai berikut:

1. File Peluncuran Dibuat:
Sebuah file peluncuran baru dibuat dengan nama `view_arm.launch` di dalam folder `launch` dari paket `mastering_ros_robot_description_pkg`.
2. File Peluncuran Diedit:
File `view_arm.launch` yang baru dibuat dibuka dan konfigurasi peluncuran ditentukan. Kemungkinan besar, file peluncuran ini berisi konfigurasi untuk menampilkan robot di Rviz dan menggunakan GUI node `joint_state_publisher`.
3. Paket Dibangun:
Perintah berikut dijalankan untuk membangun paket setelah membuat file peluncuran:
catkin_make
4. URDF Diluncurkan:
URDF diluncurkan menggunakan file peluncuran yang baru dibuat dengan perintah berikut:
roslaunch mastering_ros_robot_description_pkg view_arm.launch
5. Robot Dilihat di Rviz:
Robot diamati yang ditampilkan di Rviz dengan menggunakan GUI node `joint_state_publisher`.

Dengan mengikuti langkah-langkah ini, seharusnya dapat dibuat dan digunakan file `view_arm.launch` untuk memvisualisasikan robot di Rviz:



Creating a robot model for the differential drive mobile robot

Perbedaan pada file URDF untuk lengan robot hanya terletak pada perubahan model robot yang dimuat; bagian lainnya tetap sama. Berikut adalah langkah-langkah untuk melihat robot mobile:

1. Edit File Peluncuran:

Buka file peluncuran, misalnya `view_mobile_robot.launch` di dalam paket `mastering_ros_robot_description_pkg`.

Perhatikan bahwa satu-satunya perbedaan adalah perubahan model robot.

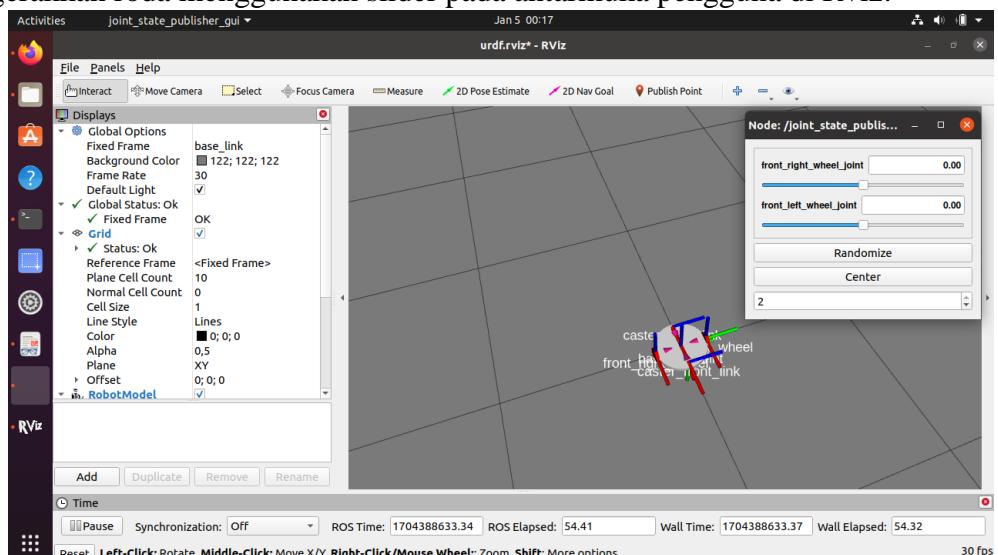
2. Lihat Robot Mobile:

Gunakan perintah berikut untuk melihat robot mobile:

roslaunch mastering_ros_robot_description_pkg view_mobile_robot.launch

3. Lihat Robot di RViz:

Lihat contoh robot di RViz, meskipun tidak dapat menggerakkan robot, dapat mencoba menggerakkan roda menggunakan slider pada antarmuka pengguna di Rviz:



Summary

Dalam bab ini, terutama dibahas pentingnya pemodelan robot dan bagaimana robot dapat dimodelkan di ROS. Pembahasan dilakukan mengenai paket-paket yang digunakan di ROS untuk memodelkan struktur robot, seperti urdf, xacro, dan joint_state_publisher beserta antarmukanya. URDF, xacro, dan tag-tag UTDF utama yang dapat digunakan juga dibahas. Model contoh dalam URDF dan xacro dibuat, dan perbedaan di antara keduanya dibahas. Selanjutnya, manipulator robotik kompleks dengan tujuh Derajat Kebebasan (DOF) dibuat, dan penggunaan paket joint_state_publisher dan robot_state_publisher dipelajari. Pada akhir bab, prosedur desain robot mobile penggerak differensial menggunakan xacro ditinjau. Pada bab berikutnya, simulasi robot ini akan dipelajari menggunakan Gazebo.

Daftar Pustaka

L. Joseph dan J. Cacace, Mastering ROS for Robotics Programming Third Edition, Birmingham: Packt Publishing Ltd, 2021.

Chapter 4 – Simulating Robots Using ROS and Gazebo

Simulating the robotic arm using Gazebo and ROS

Dalam bab sebelumnya, sebuah lengan robot dengan tujuh derajat kebebasan (DOF) telah dirancang. Pada bagian ini, simulasi robot tersebut akan dilakukan di Gazebo menggunakan ROS. Sebelum memulai dengan Gazebo dan ROS, langkah-langkah instalasi paket-paket berikut diperlukan untuk bekerja dengan Gazebo dan ROS:

1. Paket `gazebo_ros_pkgs` berisi wrapper dan alat untuk menghubungkan ROS dengan Gazebo.
2. Paket `gazebo-msgs` berisi pesan dan struktur data layanan untuk berinteraksi dengan Gazebo dari ROS.
3. Paket `gazebo-plugins` berisi plugin Gazebo untuk sensor, aktuator, dan sebagainya.
4. Paket `gazebo-ros-control` berisi pengontrol standar untuk berkomunikasi antara ROS dan Gazebo.

Setelah langkah-langkah di atas, lakukan instalasi paket-paket dengan menjalankan perintah berikut:

```
sudo apt-get install ros-noetic-gazebo-ros-pkgs ros-noetic-gazebo-msgs ros-noetic-gazebo-plugins ros-noetic-gazebo-ros-control
```

Versi default yang diinstal dari paket ROS Noetic adalah Gazebo 11.x. Penggunaan masing-masing paket adalah sebagai berikut:

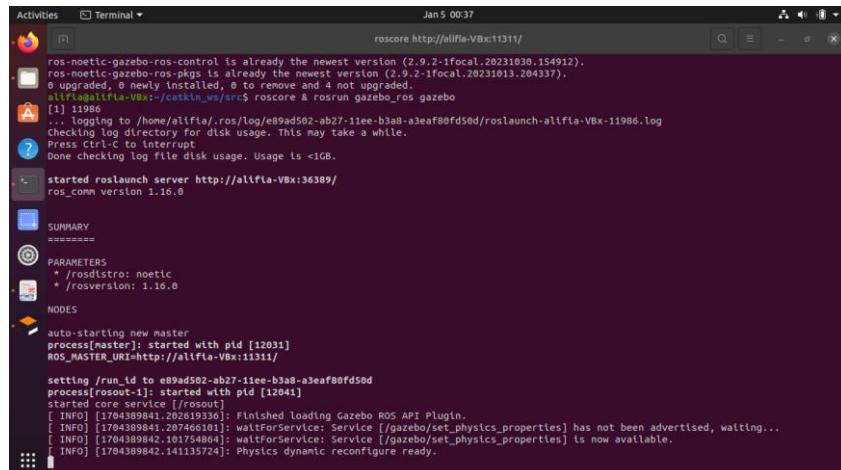
- `gazebo_ros_pkgs` digunakan untuk mengandung wrapper dan alat untuk menghubungkan ROS dengan Gazebo.
- `gazebo-msgs` berisi pesan dan struktur data layanan untuk berinteraksi dengan Gazebo dari ROS.
- `gazebo-plugins` menyediakan plugin Gazebo untuk sensor, aktuator, dan sebagainya.
- `gazebo-ros-control` berisi pengontrol standar untuk berkomunikasi antara ROS dan Gazebo.

Setelah instalasi, langkah berikutnya adalah memeriksa apakah Gazebo terinstal dengan benar.

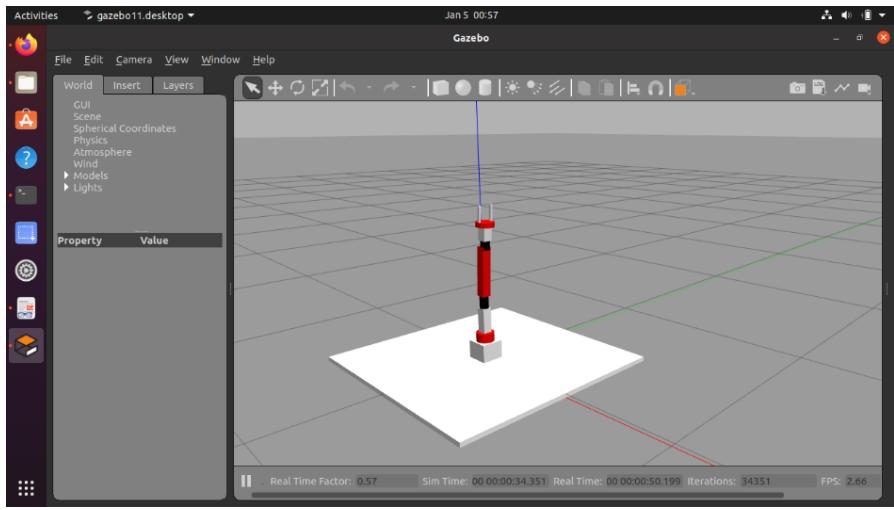
Gunakan perintah berikut untuk membuka antarmuka pengguna Gazebo:

roscore & rosrun gazebo_ros gazebo

Perintah di atas akan membuka antarmuka pengguna (GUI) Gazebo. Jika simulator Gazebo sudah terpasang, pengembangan model simulasi dari lengan robot tujuh DOF untuk Gazebo dapat dilanjutkan:



```
Activities Terminal Jan 5 00:37
roscore http://alifia-VBx:11311/
[ INFO] [1704389841.202619336]: Finished loading Gazebo ROS API Plugin.
[ INFO] [1704389841.207466181]: waitForService: Service [/gazebo/set_physics_properties] has not been advertised, waiting...
[ INFO] [1704389842.101754804]: waitForService: Service [/gazebo/set_physics_properties] is now available.
[ INFO] [1704389842.141135724]: Physics dynamic reconfigure ready.
```



Creating the robotic arm simulation model for Gazebo

Model simulasi untuk lengan robot dapat dibuat dengan memperbarui deskripsi robot yang sudah ada dan menambahkan parameter simulasi. Paket yang diperlukan untuk mensimulasikan lengan robot dapat dibuat menggunakan perintah berikut:

```
catkin_create_pkg seven_dof_arm_gazebo gazebo_msgs gazebo_plugins gazebo_ros
gazebo_ros_control mastering_ros_robot_description_pkg
```

Untuk menjalankan perintah yang diberikan dan memeriksa hasilnya, lakukan langkah-langkah berikut:

1. Jalankan perintah berikut:

```
roslaunch seven_dof_arm_gazebo seven_dof_arm_world.launch
```

2. Perintah ini akan memulai simulasi lengan robot di Gazebo. Jika semua pengaturan telah dilakukan dengan benar dan tidak ada kesalahan, dapat memvisualisasikan lengan robot pada simulator Gazebo. Pastikan bahwa hasilnya sesuai dengan harapan, seperti yang ditunjukkan dalam gambar berikut:

```

Activities Terminal Jan 5 00:56
/home/alifia/catkin_ws/src/seven_dof_arm_gazebo/launch/seven_dof_arm_world.launch http://localhost:11311
[INFO] [1704390984.834527, 0.000000]: Loading model XML from ros parameter robot_description
[INFO] [1704390984.846684, 0.000000]: Waiting for service /gazebo/spawn_urdf_model
allttab@allttab-VBx:~/catkin_ws/src$ roslaunch seven_dof_arm_gazebo seven_dof_arm_world.launch
... logging to /home/alltta/.ros/Log/926cc174-ab2a-11ee-9d6e-8366bee2068d/roslaunch-alltta-VBx-2760.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://alltta-VBx:34303

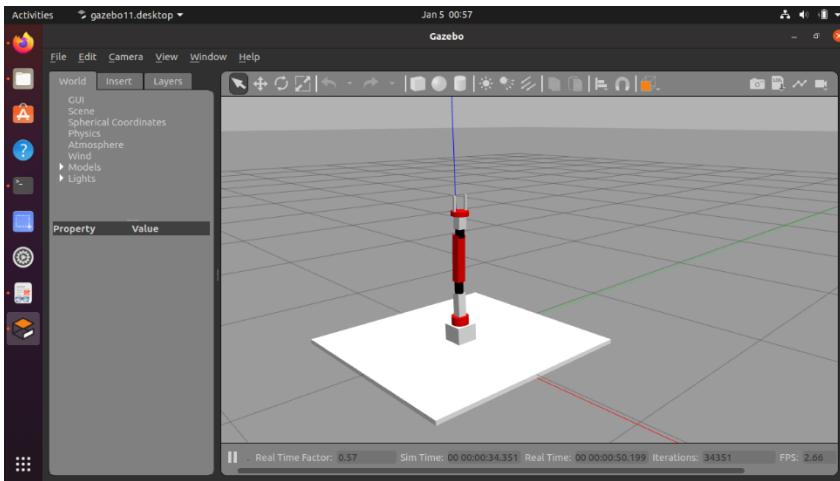
SUMMARY
=====
PARAMETERS
  * /gazebo/enable_ros_network: True
  * /robot_description: <?xml version="1.....
  * /rostdistro: noetic
  * /rosversion: 1.16.0
  * /use_sim_time: True

NODES
  /
    gazebo (gazebo_ros/gzserver)
    gazebo_gui (gazebo_ros/gzclient)
    urdf_spawner (gazebo_ros/spawn_model)

auto-starting new master
process[master]: started with pid [2772]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 926cc174-ab2a-11ee-9d6e-8366bee2068d
process[rosout-1]: started with pid [2782]
started core service [/rosout]
process[gazebo-2]: started with pid [2785]
process[gazebo_gui-3]: started with pid [2789]
process[urdf_spawner-4]: started with pid [2794]
[INFO] [1704390984.834527, 0.000000]: Loading model XML from ros parameter robot_description
[INFO] [1704390984.846684, 0.000000]: Waiting for service /gazebo/spawn_urdf_model

```

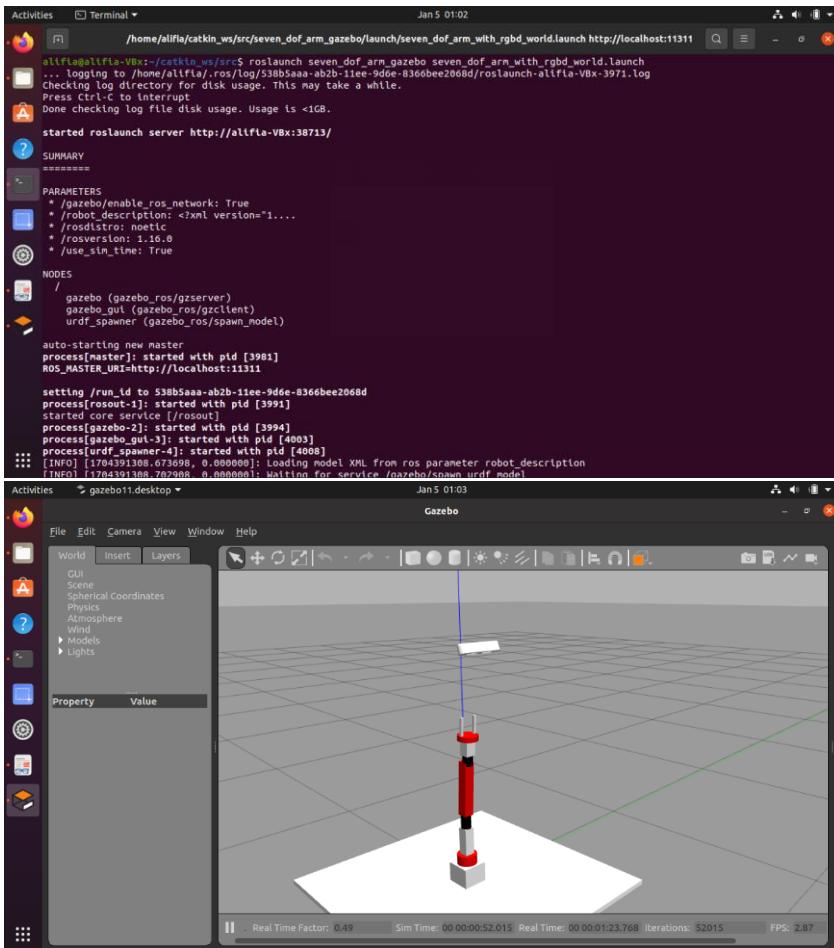


Simulating the robotic arm with Xtion Pro

Setelah definisi plugin kamera di Gazebo dipelajari, simulasi lengkap dapat diluncurkan dengan menggunakan perintah:

roslaunch seven_dof_arm_gazebo seven_dof_arm_with_rgbd_world.launch

Model robot dengan sensor di bagian atas lengan akan terlihat, sebagaimana ditunjukkan di sini:



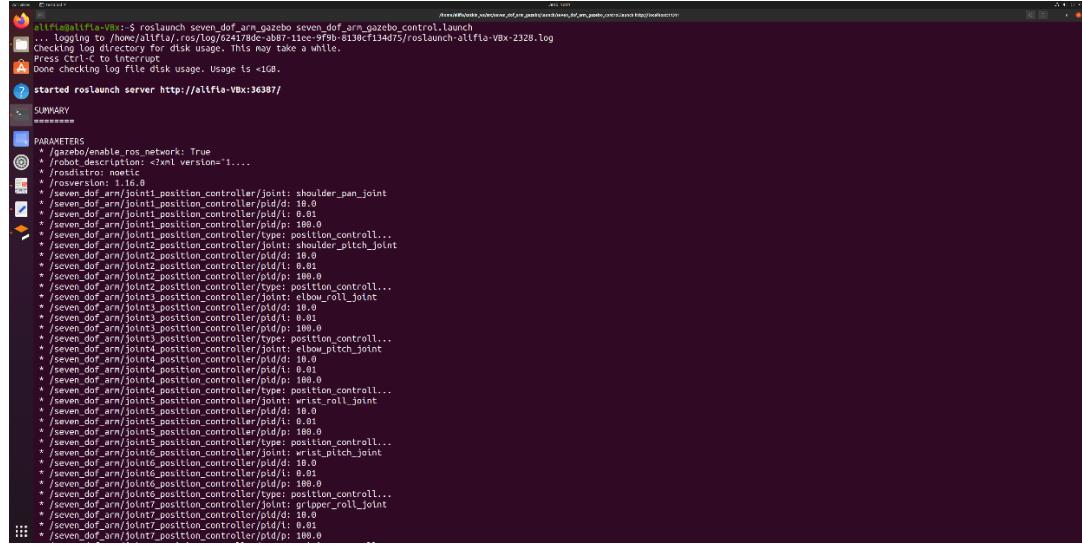
Sekarang, sensor RGB-D yang disimulasikan dapat digunakan seolah-olah terhubung langsung ke komputer. Dengan demikian, pemeriksaan apakah sensor ini memberikan output gambar yang benar dapat dilakukan.

Launching the ROS controllers with Gazebo

Simulasi lengan di Gazebo diinisiasi oleh file-file peluncuran, dan konfigurasi pengontrol, pengontrol status sendi, dan pengontrol posisi sendi dimuat. Selanjutnya, dijalankan penerbit status robot, yang bertanggung jawab untuk memublikasikan status sendi dan transformasi (TF) dengan command berikut:

```
roslaunch seven_dof_arm_gazebo seven_dof_arm_gazebo_control.launch
```

Berikut *output*-nya:



```
alim@alim-VBox:~$ roslaunch seven_dof_arm_gazebo seven_dof_arm_gazebo_control.launch
[roslaunch-0] log.INFO: Logging to /home/alim/VBox/roslaunch/624178d6-ab87-11ee-9f9b-8130cf134d75/roslaunch-alfvio-VBX-2328.log
[roslaunch-0] log.INFO: Checking log directory for disk usage. This may take a while.
[roslaunch-0] Press Ctrl-C to interrupt
[roslaunch-0] [WARN] [roslib]: Disk usage is >1GB.
[roslaunch-0] [WARN] [roslib]: Doing checking log file disk usage. Usage is <1GB.

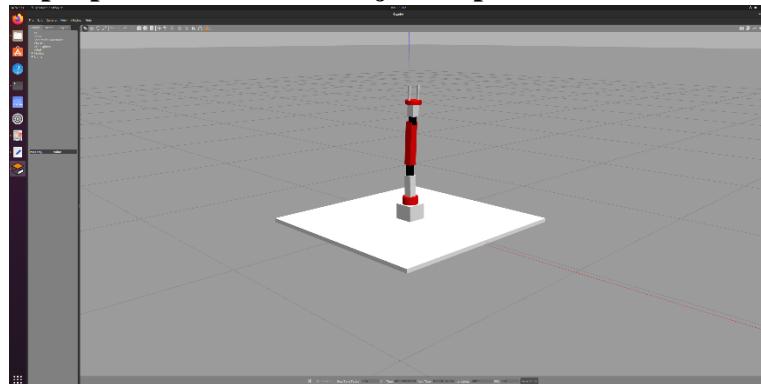
[?] started rosrun launch server http://127.0.0.1:36387/
[?] SUMMARY
[?] =====
[?] PARAMETERS
[?]   * /gazebo/enable_ros_network: True
[?]   * /robot_description: <xml version='1.0' encoding='UTF-8'><?xml version="1.0"?>
[?]   * /rosdistro: noetic
[?]   * /rosversion: 1.16.0
[?]   * /seven_dof_arm/joint1/position_controller/joint: shoulder_pan_joint
[?]   * /seven_dof_arm/joint1/position_controller/pid/i: 18.0
[?]   * /seven_dof_arm/joint1/position_controller/pid/d: 0.01
[?]   * /seven_dof_arm/joint1/position_controller/pid/p: 100.0
[?]   * /seven_dof_arm/joint2/position_controller/joint: shoulder_pitch_joint
[?]   * /seven_dof_arm/joint2/position_controller/pid/d: 18.0
[?]   * /seven_dof_arm/joint2/position_controller/pid/i: 9.0
[?]   * /seven_dof_arm/joint2/position_controller/pid/p: 100.0
[?]   * /seven_dof_arm/joint3/position_controller/joint: elbow_roll_joint
[?]   * /seven_dof_arm/joint3/position_controller/pid/d: 18.0
[?]   * /seven_dof_arm/joint3/position_controller/pid/i: 10.0
[?]   * /seven_dof_arm/joint3/position_controller/pid/p: 100.0
[?]   * /seven_dof_arm/joint4/position_controller/joint: elbow_pitch_joint
[?]   * /seven_dof_arm/joint4/position_controller/pid/d: 18.0
[?]   * /seven_dof_arm/joint4/position_controller/pid/i: 0.01
[?]   * /seven_dof_arm/joint4/position_controller/pid/p: 100.0
[?]   * /seven_dof_arm/joint5/position_controller/joint: gripper_roll_joint
[?]   * /seven_dof_arm/joint5/position_controller/pid/d: 10.0
[?]   * /seven_dof_arm/joint5/position_controller/pid/i: 0.01
[?]   * /seven_dof_arm/joint5/position_controller/pid/p: 100.0
[?]   * /seven_dof_arm/joint6/position_controller/joint: wrist_pitch_joint
[?]   * /seven_dof_arm/joint6/position_controller/pid/d: 10.0
[?]   * /seven_dof_arm/joint6/position_controller/pid/i: 0.01
[?]   * /seven_dof_arm/joint6/position_controller/pid/p: 100.0
[?]   * /seven_dof_arm/joint7/position_controller/joint: gripper_roll_joint
[?]   * /seven_dof_arm/joint7/position_controller/pid/d: 10.0
[?]   * /seven_dof_arm/joint7/position_controller/pid/i: 0.01
[?]   * /seven_dof_arm/joint7/position_controller/pid/p: 100.0
```

Moving the robot joints

Setelah menyelesaikan topik-topik sebelumnya, kita dapat mulai memberikan perintah kepada setiap sendi untuk mencapai posisi yang diinginkan.

Untuk menggerakkan sebuah sendi robot di Gazebo, kita perlu memublikasikan nilai sendi yang diinginkan dengan tipe pesan std_msgs/Float64 ke topik perintah pengontrol posisi sendi. Berikut adalah contoh untuk menggerakkan sendi keempat ke posisi 1.0 radian:

```
rostopic pub /seven_dof_arm/joint4_position_controller/command std_msgs/Float64 1.0
```



Status sendi-sendi robot juga dapat dilihat dengan menggunakan perintah berikut:

```
rostopic echo /seven_dof_arm/joint_states
```

Sekarang, semua sendi dari lengan robot tujuh DOF dapat dikendalikan, dan pada saat yang sama, nilainya dapat dibaca. Dengan cara ini, algoritma pengendalian robot kustom dapat diimplementasikan. Pada bagian berikutnya, cara mensimulasikan robot differential-drive akan dipelajari.

Simulating a differential wheeled robot in Gazebo

Untuk memulai peluncuran file ini, perintah berikut dapat digunakan:

```
roslaunch diff_wheeled_robot_gazebo diff_wheeled_gazebo.launch
```

Model robot seperti yang ditunjukkan di Gazebo akan terlihat. Jika model ini muncul, fase pertama simulasi telah berhasil diselesaikan.

Adding the ROS teleop node

Perintah Twist ROS dipublikasikan oleh node teleop ROS dengan mengambil input dari keyboard. Dari node ini, kecepatan linear dan angular dapat dihasilkan, dan sudah ada implementasi node teleop standar yang dapat digunakan ulang.

Teleop diimplementasikan dalam paket `diff_wheeled_robot_control`. Di dalam folder skrip terdapat node `diff_wheeled_robot_key`, yang merupakan node teleop. Seperti biasa, paket ini dapat diunduh dari repositori Git sebelumnya. Pada titik ini, paket ini dapat diakses dengan menggunakan perintah berikut:

roscd diff_wheeled_robot_control

Untuk berhasil mengompilasi dan menggunakan paket ini, mungkin diperlukan instalasi paket joy_node:

sudo apt-get install ros-noetic-joy

berikut *output*-nya:

```
alifia@alifia-VBox:~/catkin_ws/src$ sudo apt-get install ros-noetic-joy
[sudo] password for alifia:
Reading package lists... Done
Building dependency tree...
Reading state information...
The following additional packages will be installed:
  evemu-tools evtest joystick libeudev3
The following NEW packages will be installed:
  evemu-tools evtest joystick libeudev3 ros-noetic-joy
0 upgraded, 3 newly installed, 0 to remove and 4 not upgraded.
Need to get 152 kB of archives.
After this operation, 693 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://fd.archive.ubuntu.com/ubuntu focal/universe amd64 libeudev3 amd64 2.7.0-2 [12,5 kB]
Get:2 http://fd.archive.ubuntu.com/ubuntu focal/universe amd64 evemu-tools amd64 2.7.0-2 [13,4 kB]
Get:3 http://fd.archive.ubuntu.com/ubuntu focal/universe amd64 joystick amd64 1:1.7.0-1 [47,5 kB]
Get:4 http://fd.archive.ubuntu.com/ubuntu focal/universe amd64 evtest amd64 1:1.13.4-1 [16,9 kB]
Fetched 152 kB in 112 kB/s (Selecting previously unselected package libeudev3:amd64.
(Reading database... 278013 files and directories currently installed.)
Preparing to unpack .../libeudev3_2.7.0-2_amd64.deb ...
Unpacking libeudev3:amd64 (2.7.0-2) ...
Selecting previously unselected package evemu-tools.
Preparing to unpack .../evemu-tools_2.7.0-2_amd64.deb ...
Unpacking evemu-tools (2.7.0-2) ...
Selecting previously unselected package joystick.
Preparing to unpack .../joystick_1:3.0.1_7.0-1_amd64.deb ...
Unpacking joystick (1:3.0.1-7.0-1) ...
Selecting previously unselected package ros-noetic-joy.
Preparing to unpack .../ros-noetic-joy_1.15.1-1focal.20230620.191827_amd64.deb ...
Unpacking ros-noetic-joy (1.15.1-1focal.20230620.191827) ...
Selecting previously unselected package evtest.
Preparing to unpack .../evtest_1:3.0.1_34-1_amd64.deb ...
Unpacking evtest (1:3.0.1-34-1) ...
Setting up libeudev3:amd64 (2.7.0-2) ...
Setting up joystick (1:3.0.1-7.0-1) ...
Setting up ros-noetic-joy (1.15.1-1focal.20230620.191827) ...
Setting up evtest (1:3.0.1-34-1) ...
Setting up evemu-tools (2.7.0-2) ...
Processing triggers for liblrc-bin (2.31-0ubuntu9.14) ...
alifia@alifia-VBox:~/catkin_ws/src$ roslaunch diff_wheeled_robot_gazebo diff_wheeled_gazebo_full.launchrostopic pub /seven_dof_arm/joint4_position_controller/commandstd_msgs/Float64 1.0roslaunch seven_dof_arm_gazebo seven_dof_arm_gazebo_control.launch
RException: [diff_wheeled_gazebo_full.launchrostopic] is neither a launch file in package [diff_wheeled_robot_gazebo] nor is [diff_wheeled_robot_gazebo] a launch file name
The traceback for the exception was written to the log file
alifia@alifia-VBox:~/catkin_ws/src$ roslaunch diff_wheeled_robot_control keyboard_teleop.launch
RException: [keyboard_teleop] is neither a launch file in package [diff_wheeled_robot_control] nor is [diff_wheeled_robot_control] a launch file name
The traceback for the exception was written to the log file
alifia@alifia-VBox:~/catkin_ws/src$
```

Summary

Dalam bab ini, upaya dilakukan untuk mensimulasikan dua robot: yang pertama adalah sebuah lengan robot dengan tujuh derajat kebebasan (DOF), dan yang lainnya adalah sebuah robot beroda differential. Dimulai dengan lengan robot, diskusi dilakukan mengenai tag Gazebo tambahan yang diperlukan untuk meluncurkan robot di Gazebo. Pembahasan juga mencakup cara menambahkan sensor visi 3D ke dalam simulasi. Kemudian, sebuah file peluncuran dibuat untuk memulai Gazebo dengan lengan robot, dan dibahas bagaimana menambahkan pengontrol ke setiap sendi. Pengontrol ditambahkan dan bekerja dengan setiap sendi. Seperti halnya dengan lengan robot, URDF untuk simulasi Gazebo dibuat dan plugin Gazebo-ROS yang diperlukan untuk pemindai laser dan mekanisme penggerak roda differential ditambahkan. Setelah menyelesaikan model simulasi, simulasi diluncurkan menggunakan file

peluncuran kustom. Terakhir, dijelaskan bagaimana cara menggerakkan robot menggunakan node teleop.

Daftar Pustaka

L. Joseph dan J. Cacace, Mastering ROS for Robotics Programming Third Edition, Birmingham: Packt Publishing Ltd, 2021.

Chapter 5 – Simulating Robots Using ROS, CoppeliaSim, and Webots

Setting up CoppeliaSim with ROS

Berikut adalah langkah-langkah untuk menginstal dan mengkonfigurasi CoppeliaSim di sistem Linux, khususnya Ubuntu versi 20.04:

1. Pengunduhan CoppeliaSim:

Halaman unduhan Coppelia Robotics diakses di <http://www.coppeliarobotics.com/downloads.html>, dan versi edu untuk Linux dipilih. Dalam contoh ini, versi CoppeliaSim 4.2.0 digunakan.

2. Ekstraksi Arsip:

Setelah selesai diunduh, terminal digunakan untuk pindah ke folder tempat unduhan disimpan. Arsip diekstrak dengan perintah berikut:

```
tar vxf CoppeliaSim_Edu_V4_2_0_Ubuntu20_04.tar.xz
```

3. Perubahan Nama Folder: **

Disarankan untuk mengganti nama folder agar lebih intuitif. Misalnya:

mv CoppeliaSim_Edu_V4_2_0_Ubuntu20_04 CoppeliaSim

4. Pengaturan Variabel Lingkungan:**

Untuk kemudahan akses ke sumber daya CoppeliaSim, variabel lingkungan COPPELIASIM ROOT diatur. Perintah berikut digunakan:

```
echo "export COPPELIASIM_ROOT=/path/to/CoppeliaSim" >> ~/.bashrc
```

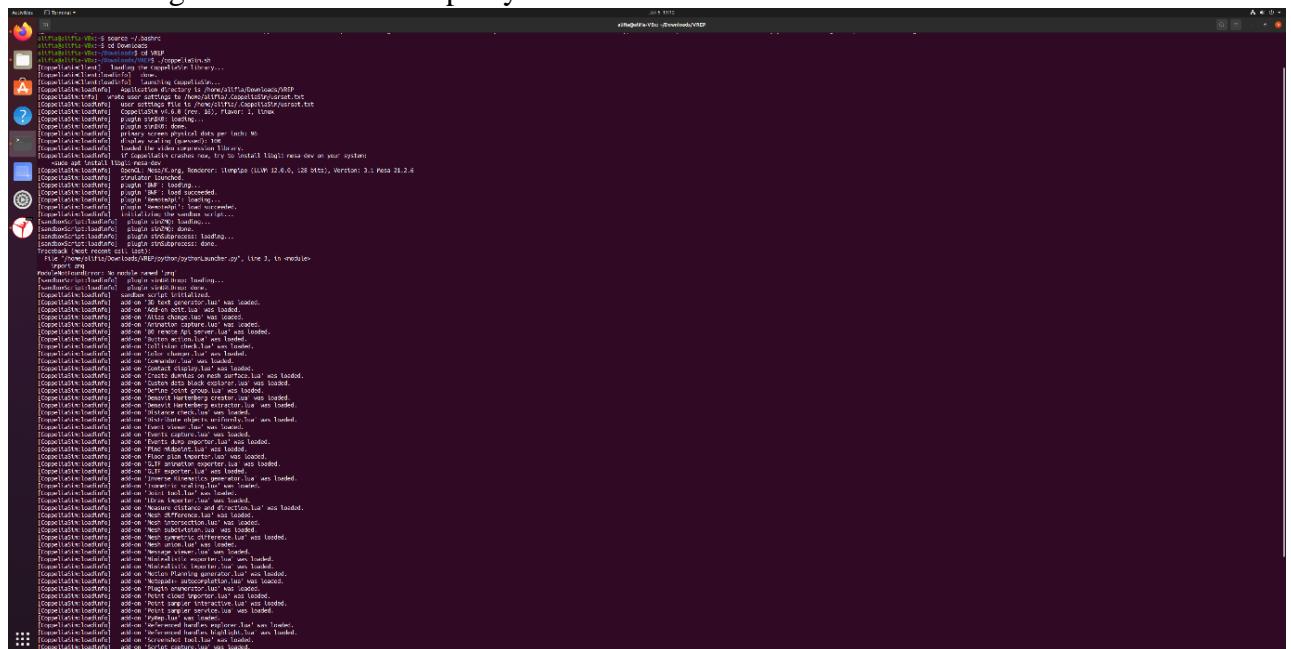
"/path/to/CoppeliaSim" diganti dengan jalur lengkap ke folder tempat CoppeliaSim diinstal.

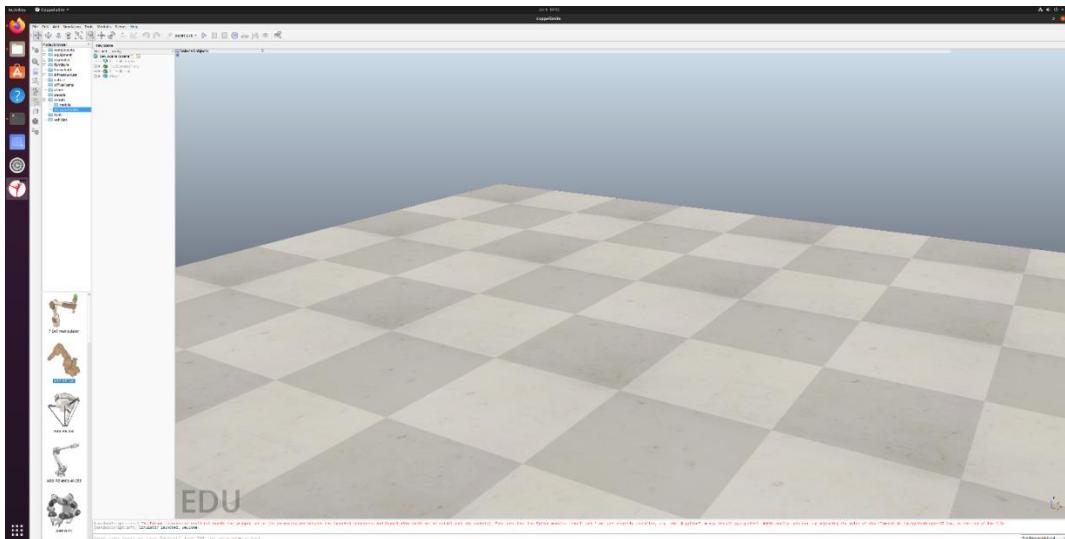
5. Pemuatan Ulang Konfigurasi Bash:**

Agar perubahan pada file `~/.bashrc` dapat diterapkan, konfigurasi bash dimuat ulang dengan perintah:

source ~/.bashrc

Dengan menyelesaikan langkah-langkah ini, CoppeliaSim sekarang diinstal dan dikonfigurasi pada sistem Linux. CoppeliaSim dapat diluncurkan, dan pengguna dapat memulai penggunaan simulasi dengan ROS. Berikut outputnya:





Working with ROS messages

Untuk mempublikasikan pesan ROS baru dalam skrip Lua, langkah-langkahnya sebagai berikut:

1. Bungkus Pesan:

Pesan baru perlu dibungkus dalam struktur data yang memiliki bidang yang sama dengan pesan ROS aslinya.

2. Periksa Struktur Pesan:

Gunakan perintah ROS berikut untuk memeriksa struktur pesan yang ingin Anda kirimkan.

Sebagai contoh, untuk pesan `std_msgs/Int32`:

rosmsg show std_msgs/Int32

Hasilnya akan menunjukkan struktur pesan, seperti:

int32 data

3. Isi Bidang Data:

Anda perlu mengisi bidang data dari struktur pesan dengan nilai yang diinginkan untuk disiarkan. Sebagai contoh, dalam skrip:

int_data['data'] = 13

```
alifia@alifia-VBx:~$ rosmsg show std_msgs/Int32
int32 data
② alifia@alifia-VBx:~$ rosrun xacro seven_dof_arm.xacro >urdf/seven_dof_arm.urdf/path/to/csim_demo_pkg/
alifia@alifia-VBx:~$
```

A screenshot of a terminal window on a Linux system. The terminal shows two commands being run. The first command is 'rosmsg show std_msgs/Int32', which outputs the message definition for 'int32 data'. The second command is 'rosrun xacro seven_dof_arm.xacro >urdf/seven_dof_arm.urdf/path/to/csim_demo_pkg/'. The terminal window has a dark background and light-colored text.

Simulating a robotic arm using CoppeliaSim and ROS

Pada bab sebelumnya, Gazebo digunakan untuk mengimpor dan mensimulasikan lengan dengan tujuh derajat kebebasan (DOF) yang dirancang pada Bab 3, Bekerja dengan ROS untuk Pemodelan 3D. Di sini, hal yang sama akan dilakukan menggunakan CoppeliaSim. Langkah pertama untuk mensimulasikan lengan tujuh-DOF adalah dengan mengimpornya ke dalam adegan simulasi. Kemampuan CoppeliaSim untuk mengimpor robot baru menggunakan file URDF memerlukan konversi model xacro lengan ke dalam file URDF, dengan menyimpan file URDF yang dihasilkan di dalam folder urdf dari paket csim_demo_pkg, sebagai berikut:

rosrun xacro seven_dof_arm.xacro > /path/to/csim_demo_pkg/urdf/seven_dof_arm.urdf

Setting up Webots with ROS

Berikut adalah langkah-langkah untuk menginstal Webots di sistem Ubuntu menggunakan Advanced Packaging Tool (APT):

1. Unduh Kunci Repotori Cyberbotics:

Jalankan perintah berikut untuk mengunduh kunci repositori Cyberbotics dan menambahkannya ke dalam sistem:

wget -qO- https://cyberbotics.com/Cyberbotics.asc | sudo apt-key add -

2. Tambahkan Repotori Cyberbotics:

Gunakan perintah berikut untuk menambahkan repositori Cyberbotics ke dalam konfigurasi APT:

sudo apt-add-repository 'deb https://cyberbotics.com/debian/ binary-amd64/'

3. Perbarui Informasi Paket:

Perbarui informasi paket APT dengan perintah berikut:

sudo apt-get update

4. Instal Webots:

Setelah repositori ditambahkan, instal Webots dengan perintah:

sudo apt-get install webots

Dengan menyelesaikan langkah-langkah ini, akan berhasil menginstal Webots di sistem Ubuntu Anda menggunakan manajer paket APT. Namun, saat menjalankan command terjadi error karena kekurangan memori, berikut outputnya:

```
Hit:1 http://ld.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://ld.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://ld.archive.ubuntu.com/ubuntu focal-backports InRelease
[...]
Hit:6 https://cyberbotics.com/debian binary-amd64/ InRelease
Reading package lists...
Reading package lists... Done
Building dependency tree...
Building dependency tree... Done
The following additional packages will be installed:
  ffmpeg libssh-dev
Suggested packages:
  libfftw3-3 libfftw3-dev
The following NEW packages will be installed:
  ffmpeg libssh-dev
0 upgraded, 3 newly installed, 0 to remove and 4 not upgraded.
Need to get 155 MB of archives.
After this operation, 456 MB of additional disk space will be used.
Do you want to continue [Y/n]? Y
Get:1 http://ld.archive.ubuntu.com/ubuntu focal-updates/universe amd64 ffmpeg amd64 7:4.2.7-0ubuntu0.1 [1.453 kB]
Get:2 http://ld.archive.ubuntu.com/ubuntu focal-updates/main amd64 libssh-dev amd64 0.9.3-2ubuntu2.4 [220 kB]
Get:3 https://id.cyberbotics.com/debian binary-amd64/ webots 2023b [153 kB]
Fetched 155 MB in 54s (2.862 kB/s)
Selecting previously unselected package ffmpeg.
(Reading database ... 27889 files and directories currently installed.)
Preparing to unpack .../ffmpeg_7:4.2.7-0ubuntu0.1_amd64.deb ...
Unpacking ffmpeg (7:4.2.7-0ubuntu0.1) ...
Selecting previously unselected package libssh-dev:amd64.
Preparing to unpack .../libssh-dev_0.9.3-2ubuntu2.4_amd64.deb ...
Unpacking libssh-dev:amd64 (0.9.3-2ubuntu2.4) ...
Selecting previously unselected package webots.
Preparing to unpack .../webots_2023b_amd64.deb ...
Unpacking webots (2023b) ...
Setting up libssh-dev:amd64 (0.9.3-2ubuntu2.4) ...
Setting up ffmpeg (7:4.2.7-0ubuntu0.1) ...
Setting up webots (2023b) ...
Processing triggers for man-db (3.6.0-1ubuntu1) ...
Processing triggers for man-db (2.9.1.1) ...
/usr/bin/mandb: can't write to '/var/cache/man/5349: No space left on device
/usr/bin/mandb: can't create index cache /var/cache/man/5349: No space left on device
Processing triggers for desktop-file-utils (0.24-1ubuntu3) ...
/usr/bin/mandb: command not found
allffmaliita-Vin:~$ webots
webots: Failed to create the Webots temporary path "/tmp/webots/allfia/1234".
:: Try 'webots --help' for more information.
```

Summary

Dalam bab ini, langkah-langkah yang telah dilakukan pada bab sebelumnya dengan Gazebo direplikasi menggunakan simulator robot lainnya, yaitu CoppeliaSim dan Webots. CoppeliaSim dan Webots adalah program perangkat lunak simulasi multiplatform yang mengintegrasikan berbagai teknologi dan sangat serbaguna. Dua robot utama disimulasikan, salah satunya diimpor menggunakan file URDF dari lengan tujuh-DOF yang dirancang sebelumnya, sementara yang lainnya adalah robot beroda differential yang populer di Webots. Pembelajaran melibatkan cara berinteraksi dan mengendalikan sendi-sendi robot dalam model menggunakan ROS, serta bagaimana menggerakkan robot beroda differential menggunakan topik. Pada bab berikutnya, akan dijelaskan bagaimana menghubungkan lengan robot dengan paket ROS MoveIt dan menghubungkan robot beroda differential dengan tumpukan Navigasi.

Daftar Pustaka

L. Joseph dan J. Cacace, Mastering ROS for Robotics Programming Third Edition, Birmingham: Packt Publishing Ltd, 2021.