EE3305/ME3243
Robotic System Design
Exercise with Ubuntu and ROS

Teaching Assistant: Guo Haoren haorenguo_06@u.nus.edu
Instructor 1: Dr. Andi Sudjana Putra andi_sp@nus.edu.sg
Instructor 2: A/Prof. Prahlad Vadakkepat prahlad@nus.edu.sg

©National University of Singapore

AY 2023/2024 Semester 1
October 4, 2023

Contents

1 About the Exercise

Students will practice launching a ROS program. The package is provided. Using the ROS program that will be developed, students will practice navigating files and folders in Ubuntu.

The program is based on the "Talker-Listener" tutorial (http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29). An analogy is presented below.

1. Student A wants to send information to Student B. Student A is a *node* , student B is another *node*. Student A sends the information using an email and a telegram. The email is a *topic*, the telegram is another *topic*.
2. Student A is a *publisher node*. Student B is a *subscriber node*. Node B needs to subscribe to a *topic*, in the same way as student B needs to have an email account if student A sends information via an email.
3. The information convey by the *topic* is called a *message*.
4. The content of the information may be "I saw 3 students in location 1", which can be the *message*.
5. The pattern of the information may be repeated. In the above example, if the bold message in "**I saw** 3 **students in location** 1" is repeated, as in the next piece of information is "**I saw** 4 **students in location** 2", the text is redundant. The published *message* may just be 3 and 1, and then 4 and 2, and so on.
6. `msg` file defines the meaning of 3 and 1 in the above example. `msg` file is a text file that describes the field of a *message*, i.e., it defines the data structure of the *message* contained in a *topic*.

2 Objectives

At the end of the project, students are expected to demonstrate their ability to:

1. Create a workspace
2. Make a package
3. Navigate files and folders
4. (Optional) Create a launch file

3 Preparation (for personal machines, not applicable to machines in the lab)

1. Install ROS Noetic.

2. Install the standard tutorials using the command

   `$ sudo apt-get install ros-noetic-ros-tutorials`.

3. Install Sublime Text editor using the command `$ sudo snap install sublime-text --classic`.

4 Steps

1. Create a workspace and build it. Create a workspace called `a345b_exercise_ws` using the following command:

```
$ cd ~
$ mkdir a345b_exercise_ws
$ cd a345b_exercise_ws
$ mkdir src
$ cd ~/a345b_exercise_ws
$ catkin_make
```

2. Download and copy the package as provided.

   Download the zipped package `a345b_talker_listener.zip`. Unzip it. Copy the entire **unzipped folder** to `~/a345b_exercise_ws/src`.

3. Create the package. In `~/a345b_exercise_ws/`, make the package using the following command:

```
# In ~/a345b_exercise_ws/
$ catkin_make
```

4. Source the workspace. Source the workspace using the following command:

```
# In ~/a345b_exercise_ws
$ source ./devel/setup.bash
```

5. Run the publisher (talker) and the subscriber (listener). Simulate the talker and listener communicating as follows:

   a) Open a terminal and run `$ roscore`. This is the master node.

   b) Open another terminal. Go to `~/a345b_exercise_ws`. Source it using the command

      `$ source ./devel/setup.bash`.

      Run the command `$ rosrun a345b_talker_listener talker`.

   c) Open another terminal. Go to `~/a345b_exercise_ws`. Source it using the command

      `$ source ./devel/setup.bash`.

      Run the command `$ rosrun a345b_talker_listener listener`.

   You should see the terminal running the talker node publishing a series of messages and the terminal running the listener node repeating those messages. You can try out a few things as follows:

a) Stop the listener node (using `Ctrl+C`) while the talker node is kept running. **What do you observe?**

b) Stop the talker node (using `Ctrl+C`) while the listener node is kept running. **What do you observe?**

5 Navigate files and folder.

Refer to *"Frequently Used Ubuntu Commands and Shortcuts"*.

1. Go to `Home` folder. Use the command `$ cd ~`.
2. From `Home` folder, go to `~/a345b_exercise_ws`. Use the command `cd a345b_exercise_ws`.
3. From the folder `~/a345b_exercise_ws`, go up one folder using the command `$ cd ..`. **What folder are you in now?**
4. Run the command `$ pwd`. **What is the result? What does it mean?**
5. From the current folder, go to `~/a345b_exercise_ws/src`. **What command(s) do you use?**
6. List the folders and files under `~/a345b_exercise_ws/src`. Use the command `ls`. **What are the folders and files?**
7. You can continue practicing.

6 (Optional) Creating a launch file.

In Section 4, three terminals are opened separately to launch multiple nodes. Creating a *launch* file allows all nodes to run using one command, i.e., `roslaunch`.

1. Inside `~/a345b_exercise_ws/src/a345b_talker_listener`, create a *launch* folder as follows:

```
# In ~/a345b_exercise_ws/src/a345b_talker_listener
$ mkdir launch
```

2. Inside `~/a345b_exercise_ws/src/a345b_talker_listener/launch`, create a *launch* file `a345b_talker_listener.launch` and open it as follows:

```
# In ~/a345b_exercise_ws/src/a345b_talker_listener/launch
$ touch a345b_talker_listener.launch
$ subl a345b_talker_listener.launch
```

3. Write the following statements:

```
<?xml version="1.0"?>
<launch>
  <node pkg="a345b_talker_listener" type="talker" name="talker"
  ↪  output="screen" launch-prefix="gnome-terminal --command"/>
  <node pkg="a345b_talker_listener" type="listener" name="listener"
  ↪  output="screen" launch-prefix="gnome-terminal --command"/>
</launch>
```

4. Save the `launch` file.

5. Build the workspace and source it. Go to `~/a345b_exercise_ws`. Use the command `$ catkin_make` to build the workspace. Use the command `$ source ./devel/setup.bash` to source the workspace.

6. Launch the file. In `~/a345b_exercise_ws`, launch the nodes using the command `$ roslaunch package_name launch_file_name.launch` as follows:

```
# In ~/a345b_exercise_ws
$ roslaunch a345b_talker_listener a345b_talker_listener.launch
```

   You should see the same results by running only one command.