RBE 500 — Foundations of Robotics

Instructor: Siavash Farzan

Spring 2022

ROS Assignment 1: Introduction to Robot Operating System (ROS)

Contents

1.1 Objectives

By the end of this assignment*, you should be able to:

- Set up a new ROS environment, including creating a new workspace and creating a package with the appropriate dependencies specified
- Use the catkin tool to build the packages contained in a ROS workspace
- Explain basic ROS concepts, including ROS Node, ROS Topic, ROS Subscriber, ROS Publisher, ROS Service, and ROS Client
- Run nodes using `rosrun`
- Use ROS's built-in tools to examine the topics and services used by a given node
- Create and run ROS service and client nodes using Python or C++

*Parts of this assignment are adapted from the course materials by Professor K. Karydis at UC Riverside.

1.2 Problem Statement

a) Complete the ROS Tutorials Beginner Level 1-16.

b) Develop a ROS package and a shell script to move the Turtlebot using open-loop control (i.e. sending commands only through bash scripts; no feedback) to draw the first letter of your first name.

c) Develop a second script to draw the first letter of your first name and the first letter of your last name in two different colors using two Turtlebots.

1.3 Programming Tips

i) Python is an indent-sensitive programming language, as opposed to C/C++.

ii) You can use either `space` or `Tab` for indent, but please do not mix them in one file. Otherwise you will see lingering syntax errors. (`space` is recommended)

iii) If you don't have a preferred code editor in Ubuntu, we recommend Sublime Text: `https://www.sublimetext.com`
You can follow the instructions in the following to install it on your Ubuntu machine:

```
wget -qO - https://download.sublimetext.com/sublimehq-pub.gpg | sudo apt-
    key add -
sudo apt-get install apt-transport-https
echo "deb http://download.sublimetext.com/ apt/stable/" | sudo tee /etc/
    apt/sources.list.d/sublime-text.list
sudo apt-get update
sudo apt-get install sublime-text
```

After you installed Sublime, you can open any source file from terminal by using the `subl` command. For example:

```
subl new.py
```

1.4 ROS Basics

From now on, we assume that you have already installed Ubuntu 20.04 and ROS Noetic.

Open a new terminal, and create a new ROS workspace by the following commands (run them line by line).

```
mkdir -p ~/rbe500_ros/src
cd ~/rbe500_ros
catkin_make
echo "source ~/rbe500_ros/devel/setup.bash" >> ~/.bashrc
source ~/rbe500_ros/devel/setup.bash
```

Take a look at `rbe500_ros` directory and see what happens. You can use `ls` command to see the files in this directory, or use `ls -a` to see all files including hidden files. The file `.catkin_workspace` was created to tell the system that this current directory is a ROS workspace.

Note: If you are fresh new to Linux, the instructions might be a bit hard to understand at this moment. No worries. Please just try it for the time being and you will have a better understanding as we move on.

Next let's create a new ROS package.

```
cd ~/rbe500_ros/src
catkin_create_pkg intro_turtle rospy
```

Take a look at your new package `intro_turtle` and see what happens. You should be able to see a `package.xml` file and a `CMakeLists.txt` file. Open them and take a quick look. You may use Google to help you build up a high-level understanding.

After creating a new package, we can go back to our workspace and build this package. This is to tell ROS that "Hey, we have a new package here. Please register it into the system."

```
cd ~/rbe500_ros
catkin_make
```

Now the system knows this ROS package, so that you can have access to it anywhere. Try navigating to different directories first, and then go back to this ROS package by `roscd` command. See what happens when running the following commands.

```
cd
roscd intro_turtle

cd ~/rbe500_ros
roscd intro_turtle

cd ~/Documents
roscd intro_turtle
```

Congratulations. You have completed the basic ROS tutorials. Take some time to think about how the above steps work.

1.5 Requirements

Step 1: Complete "Beginner Level" ROS tutorials 1-16 (until and including "Examining the Simple Service and Client"), available at:
`http://wiki.ros.org/ROS/Tutorials`

While following the step-by-step tutorials, please take your time to think about what you are doing and what happens in each step, with the help of Google if necessary.

For the tutorials with C++ and Python option, you can choose the programming language you want. For example, you can choose either tutorial 11 or tutorial 12. However, it is highly recommended to use the Python language, as all the sample codes in this course are written in Python. Note that the example in tutorials 14, 15, 16 develops a simple "service and client" for summing two integers.

Step 2: Inside your `intro_turtle` ROS package, create a new directory named `scripts`. Inside this directory, create a new shell script file named `moveturtle.sh`, and open it using Sublime or your preferred code editor.

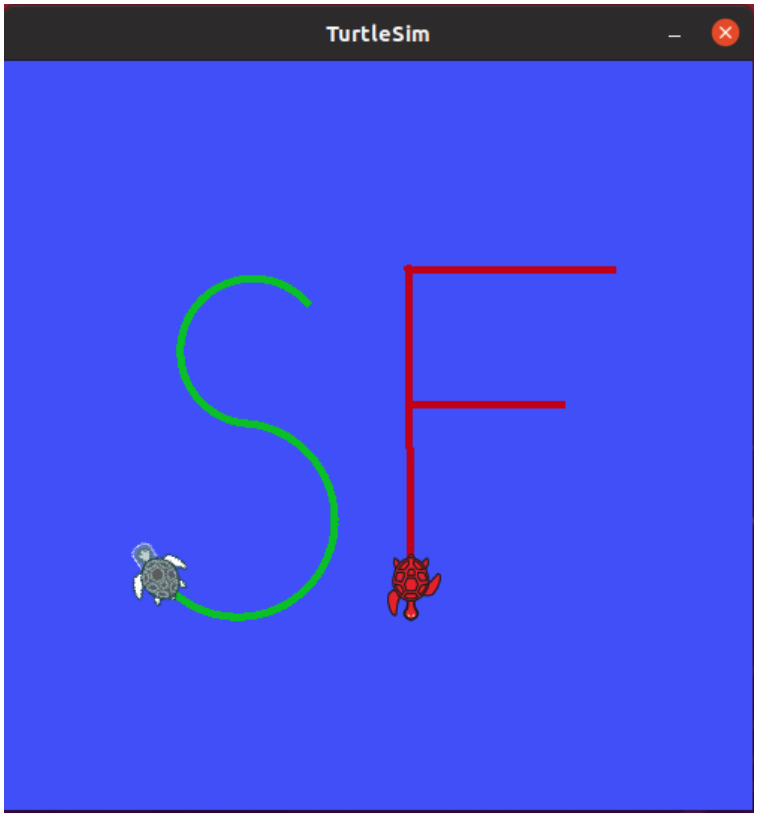

Figure 1: An example for Step 3.

```
touch moveturtle.sh
chmod +x moveturtle.sh
subl moveturtle.sh
```

Write shell scripts inside the file to move the Turtlebot to draw the first letter of your first name (in UPPERCASE). Note that the robot is moved using open-loop control (i.e. sending commands only; no feedback). Include the screenshot of the final Turtlebot window, showing the performance of your script.
Hint: You can use `rostopic pub` for this task. See Section 4.1 of ROS Tutorial 6.

Step 3: Again, inside the `scripts` directory, create a new shell script file named `movetwoturtles.sh`. Write shell scripts inside to spawn two Turtlebots, one drawing the first letter of your first name and the other one the first letter of your last name (both in UPPERCASE, in two different colors). An example is shown in Figure 1. Include the screenshot of the final Turtlebots window, showing the performance of your script.
Hint: This is an extension to Step 2. You can use ROS Services and Parameters for the new tasks. Specifically, see ROS Tutorial 7.

Step 4: Consider the "talker/listener" Publisher and Subscriber example developed in Tutorials 11-13. In your report, briefly explain how the same functionality could be implemented using a Service and Client (similar to the example in Tutorials 14-16), and what the structure of `.srv` file would be in that case.

1.6 Performance Testing

First, start `roscore` in the background. Then open a new terminal, and run the `turtlesim_node` in the turtlesim package.

```
rosrun turtlesim turtlesim_node
```

Finally, in a third terminal, execute the script under testing by using the `bash` command, for example:

```
bash movetwoturtles.sh
```

1.7 Submission

- Submission: individual submission via Gradescope.
- Due time: as specified on Canvas
- Files to submit: (please use exactly the same filename)
  - `moveturtle.sh`
  - `movetwoturtles.sh`
  - `ros1_report.pdf`
- Grading rubric:
  - 25%: submit the correct `.sh` scripts developed in step 2 and step 3.
  - 25%: clearly describe your approach and explain your scripts (i.e. the topics, services, and messages used in the scripts) in the report.
  - 25%: include the screenshots of the Turtulebot(s) performance in step 2 and step 3 in the report.
  - 25%: include the answer to step 4 (concerning the Publisher/Subscriber vs. Service/-Client) in the report.