CS-475 Assignment 0
Introduction to ROS

Michael Maravgakis
maravgakis@csd.uoc.gr

Release date: 17/02/25
Deadline: 28/02/25

1 Introduction

This assignment is a tutorial on how to setup and use the Robot Operating System *ROS*. You will install ROS Noetic and then you will add the turtlebot3 robot to the built-in simulated environment, *Gazebo*. The goal is to get familiar with the core concepts and tools of ROS.

2 Installation

In order for ROS Noetic to work, it needs to be installed on top of Ubuntu 20.04. So you need to install Ubuntu 20.04 first. Choose one of the following options :

1. (Recommended) Install Ubuntu 20.04 as your main operating system (or dual boot). Follow these instructions:
   https://phoenixnap.com/kb/install-ubuntu-20-04
2. (OK) Install Ubuntu 20.04 in WSL (prefered) or VM. Download the Desktop image (.iso) from https://releases.ubuntu.com/focal/ and then follow the tutorial tutorial: https://linuxhint.com/install_ubuntu_virtualbox_2004/.
3. (Not recommended) You can skip the installation of Ubuntu and work with ROS online.Use *the contrust*, online ROS platform
   The online ROS platform is https://www.theconstructsim.com/ where you are required to create a free account.

Now that you have installed Ubuntu 20.04, follow the ROS instructions to install ROS noetic, http://wiki.ros.org/noetic/Installation/Ubuntu (Install the full desktop version). After the installation, it is highly suggested to practice on the beginner tutorials from: http://wiki.ros.org/ROS/Tutorials especially the "Writing a Simple Publisher and Subscriber (Python)"

3 Setup turtlebot3

Assuming that you have installed ROS and followed some of the tutorials, now you should be a bit familiar with the file structure and some of the key concepts of ROS(nodes,topics,msg, etc.)

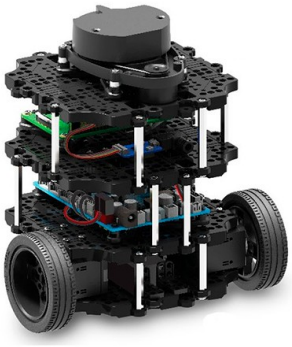

Figure 1: turtlebot 3

Follow the instructions bellow in order to create a fresh workspace and install the turtlebot there.

Use a terminal to navigate to your home directory and then:

```
$ mkdir -p 475_ws/src
$ cd 475_ws/src
```

If you already do not have *git* installed, you can use:

```
$ sudo apt install git
```

Clone the turtlebot repositories to your workspace:

```
$ git clone --branch noetic --single-branch
    https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git
```

```
$ git clone --branch noetic --single-branch
    https://github.com/ROBOTIS-GIT/turtlebot3.git
```

```
$ sudo apt-get install ros-noetic-turtlebot3-msgs
```

Compile the project:

```
$ cd ..
$ catkin_make
```

In order for your system to see the new ROS workspace, you need to source the devel/setup.bash file every time you open a new terminal window by using:
`$ source /home/<usr_name>/475_ws/devel/setup.bash`

I suggest to just add this line at the bottom of your .bashrc and never have to deal with it again. The .bashrc file is located at your home directory, you need to modify it as a superuser:

```
$ sudo gedit .bashrc
```

Add the following lines at the and of the file:

```
source /home/<usr_name>/475_ws/devel/setup.bash
export TURTLEBOT3_MODEL=burger
```

The latter is just needed in order to specify the robot we are going to use. Save and exit!
Open a new terminal window and run `$ rospack profile`, this practically updates the packages for ROS. Now that everything is in place, you can spawn the robot inside the simulation environment by using:

```
$ roslaunch turtlebot3_gazebo turtlebot3_empty_world.launch
```

You should be able to see the turtlebot located at the origin (0,0,0) inside an empty world. In general you can use the `roslaunch` command to run any `.launch` files. The general use is:
`$ roslaunch <package_name> <name_of_launchfile>`
When you type `$roslaunch <package_name>$` if you press the TAB button twice, it displays all the launch files contained in the package. In genera, the names are pretty descriptive on what each file actually does when you launch it so you can decide on the spot what would you like to run. I suggest to navigate

and test different environments (e.g. `turtlebot3_house.launch`). Note that the more complicated the environment is, the more time you have to wait in order to load it.

4 Create a package

Finally, the only thing that remains is for you to create a package and then a node that subscribes and publishes to a topic.
Use this to create a package after you've navigated to the src/ folder of your workspace:
`$ catkin_create_pkg <Package Name> rospy nav_msgs std_msgs`
The rospy, nav_msgs, std_msgs arguments are the dependencies of your package. Navigate to your new package/src/ and there create a `<name>.py` file, where `<name>` is your registration number, this will be the name of your node. In order for a python script to get installed properly and use the right python interpreter, you need to add to the end of the `CMakeLists.txt` the following lines:

```
catkin_install_python(PROGRAMS src/<name>.py
         DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
)
```

Replace the `<name>` with whatever you've chosen as your node name. This makes sure that python scripts get installed properly, and use the right python interpreter.

5 Compute Distance

The purpose of your node is to extract the necessary information from the `/odom` topic and calculate the 2D distance between the robot and the world frame (0,0,0). Then, you will need to publish the calculated distance to another topic with a name of your choice.

To run your node you will need to use the `rosrun` command:
`$ rosrun <name_of_your_package> <name_of_your_node>`
Move the robot around by using the teleop launch file and then subscribe to your created topic and see if you get the expected results. You can move the robot by running:
`$ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch`
(Follow the prompt instruction on how to move it)

6 Tips

1. Use: `rostopic type <topic_name>` to find out what type of message a topic uses
2. Run `$ rosmsg show <general_class/specific_type>` to easily see the data structure of the message.(e.g. `rosmsg show std_msgs/Float32`)
3. To get only one message from a topic you can use the `rospy.wait_for_message('topic_name',msg_type,timeout=None)`

7 Submission

Send your node (`<registration number>.py`) attached via email at: **maravgakis@csd.uoc.gr** Don't forget to mention your name and registration number. The deadline is at **28/02/25**