

ROS Workshop - Tutorial 3 - Turtlesim

ME 4140 - Introduction to Robotics - Fall 2020

Overview:

After completing *Tutorial 2 - Install ROS*, your system is setup. You are ready to begin with Turtlesim, a simplistic robot model and simulator that serves as the *Hello World of ROS*. You can read more about turtlesim [here](#) on the wiki.

System Requirements:

- **ROS+OS:** This tutorial is intended for a system with ROS Melodic installed on the Ubuntu 18.04 LTS operating system. Alternate versions of ROS (i.e. - Kinetic, Noetic, etc.) may work but have not been tested. Versions of ROS are tied to versions of Ubuntu.
- **Internet:** Your computer must be connected to the internet to proceed. Downloading and installing *turtlesim* will only take a few minutes.

Disclaimer:

- **Copy and Paste Errors:** The ilearn PDF viewer does not allow the commands to be copied properly. Download the PDF if you want to copy and paste commands.
- **Learn the Terminal:** The commands in this tutorial are relatively short, and it may help improve understanding to type them manually. Press `Tab` for auto-completion!

Turtlesim Installation Instructions:

Press `Ctrl` + `Alt` + `T` to open a new terminal, then carefully copy each command and paste it into the terminal then press `Enter`. The terminal commands are shown in gray boxes, and you will have multiple terminals open at once during this tutorial.

1. Update your Ubuntu packages. It is recommended to do this before you begin something new.

```
sudo apt update
```

2. Install [turtlesim](#) for ROS Melodic from the pre-built repositories. This will take a few moments. Also, install a keyboard controller node.

```
sudo apt install ros-melodic-turtlesim
```

```
sudo apt install ros-melodic-teleop-twist-keyboard
```

The terminal will show if the installations were successfully, and it will indicate if the packages were previously installed (turtlesim came with ros-melodic-desktop-full).

Turtlesim Testdrive:

Now, test the newly installed simulator. This exercise is simple, but the process is important.

1. Start the roscore in a terminal. Leave this process running and this window open.

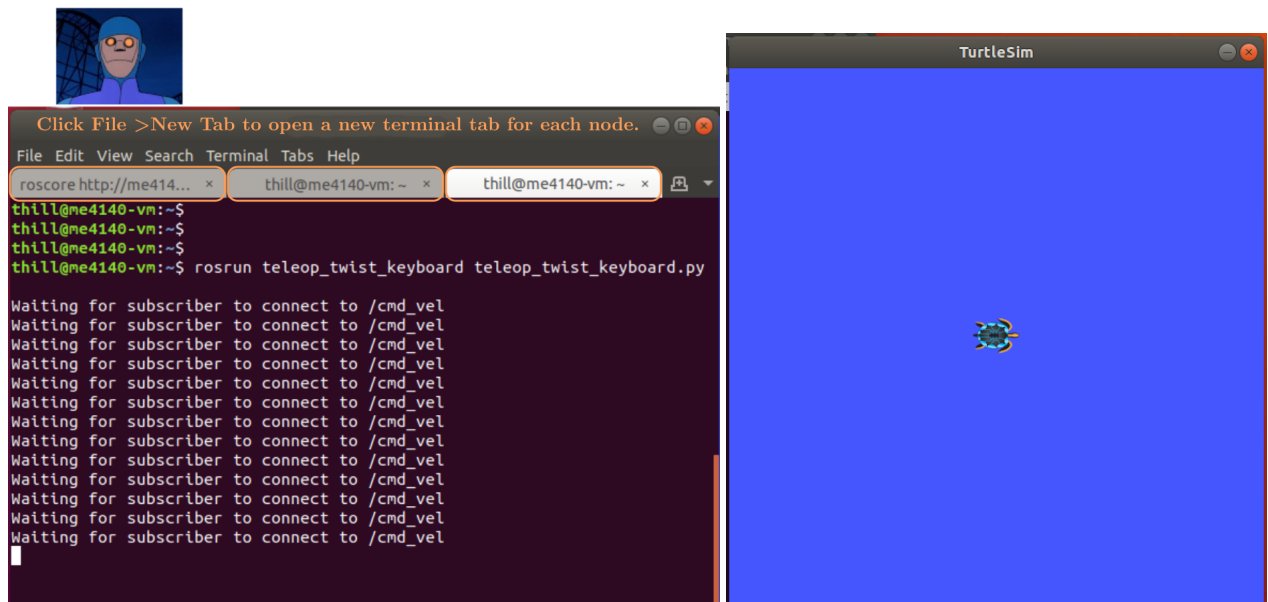
```
roscore
```

2. Open a second tab (file>new tab), and start a turtlesim_node in the new terminal tab.

```
roslaunch turtlesim turtlesim_node
```

3. In a third terminal tab run the keyboard controller node.

```
roslaunch teleop_twist_keyboard teleop_twist_keyboard.py
```



There is a problem, the nodes are not communicating. The turtlesim node is not subscribing to the topic published by the keyboard node.

4. **rostopic** is a tool in ROS for this type of problem. Try this in a new terminal with your other nodes still running.

List all available topics under master

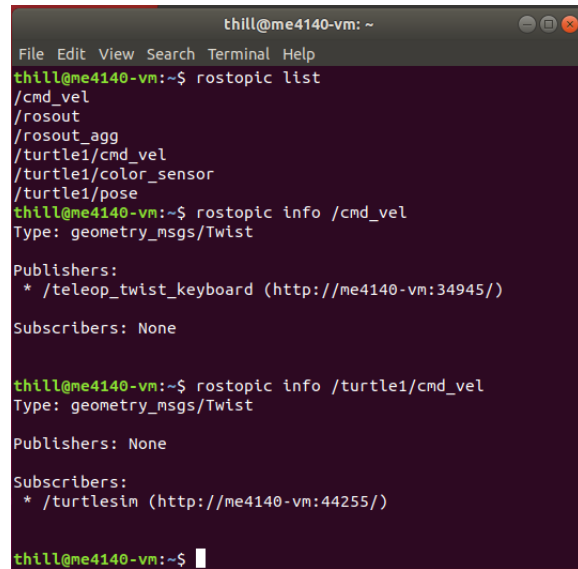
```
rostopic list
```

Get info about topic: `/cmd_vel`

```
rostopic info /cmd_vel
```

Get info about topic: `/turtle1/cmd_vel`

```
rostopic info /turtle1/cmd_vel
```



```
thill@me4140-vm: ~
File Edit View Search Terminal Help
thill@me4140-vm:~$ rostopic list
/cmd_vel
/rosout
/rosout_agg
/turtle1/cmd_vel
/turtle1/color_sensor
/turtle1/pose
thill@me4140-vm:~$ rostopic info /cmd_vel
Type: geometry_msgs/Twist
Publishers:
 * /teleop_twist_keyboard (http://me4140-vm:34945/)
Subscribers: None
thill@me4140-vm:~$ rostopic info /turtle1/cmd_vel
Type: geometry_msgs/Twist
Publishers: None
Subscribers:
 * /turtlesim (http://me4140-vm:44255/)
thill@me4140-vm:~$
```

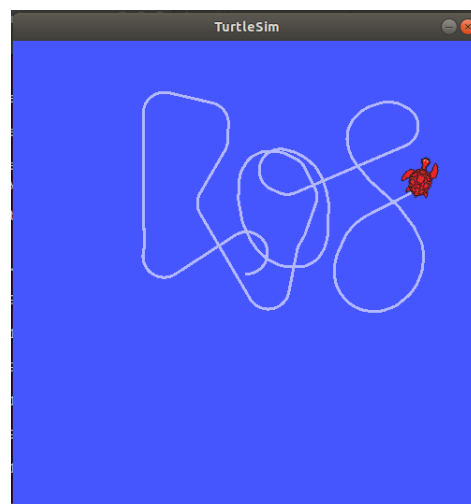
5. Abort the keyboard node by clicking in the third tab and pressing `Ctrl` + `C`. Then **append the following option** to the end of the keyboard node command and rerun the node. Now, the keyboard node can communicate with the turtle.

```
cmd_vel:=turtle1/cmd_vel
```

Move the turtlesim window to the side, and select the keyboard terminal to drive using the following keys.

`I`, `J`, `K`, `L`, and `,` (comma)

Drive your turtle around the window and save an image of the window showing the turtle and the path. Can you drive a better ROS than shown in this picture?



Tutorial Complete:

After completing *Tutorial 3 - Turtlesim*, you are ready to learn how to create a custom package.