**Fall 2019**
**CS 480/580: Intelligent Mobile Robotics**
**Assignment 1 (5% of total grade)**
**Due time: Thursday 9/12 11:59 PM**
**(Updated on 9/7/2019)**

**Notes:**

- If you encounter errors, you should first try to solve it on your own (e.g., using Google). Solving problems on your own is an important skill for computer scientists and software engineers. Our Piazza forum is also a good place to look for help. If you still cannot find the answer easily, then please email the TA/instructor or come to office hours.
- Sometimes, restarting your program or rebooting your machines just magically fixes problems.
- The rest of this course will assume you know how to install Ubuntu, ROS, and other required tools, so it's important that you finish this assignment independently.

**Goal**

The goal of this assignment is to let you first install, and then get used to the ROS+Ubuntu working environment. You also need to run a 2D simulator in ROS and look at the messages from your control commands.

**Instructions**

1. Find a machine with Ubuntu 16.04 and ROS Kinetic

   You can either use the machines in the public computer labs (if the software versions meet our requirements) or you can install ROS on your own machine.

   To check the release of a Ubuntu system, you can use the following command:

   *lsb_release -d*

   To test if your machine has ROS installed, type the following in the terminal:

   *roscore*

   If the printout is "roscore: command not found", you will need to following this instruction to install ROS Indigo on the machine:

   http://wiki.ros.org/indigo/Installation/Ubuntu

   After the installation, again, you can test it by running "roscore", and check the printout to make sure your ROS has been installed and setup properly.

2. Compile your own workspace (Core ROS Tutorials 1)

   Now we can move on to ROS Tutorials. You will need to finish Tutorials 1-6 in this assignment. http://wiki.ros.org/ROS/Tutorials

   Let us start with Tutorial 1:

http://wiki.ros.org/ROS/Tutorials/InstallingandConfiguringROSEnvironment

You can directly jump to Step 3 (Create a ROS Workspace). Make sure that you are following the "catkin" subpage (not "rosbuild", which was created for a few very old versions of ROS).

Following the instructions to crate your "catkin workspace" and "build" the workspace. If everything works well, the following command should not produce any warnings or errors:

```
$ catkin_make
```

Now you have your empty workspace compiled. If you run "roscd", your current directory will be changed to "/opt/ros/indigo". Now go back to your workspace "cd ~/catkin_ws", and then run the following command to "source" your own workspace:

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

If you run "roscd" now, you will be under the path of "~/catkin_ws/devel". Note that you may need to run the above "source" command every time a new terminal is opened. Alternatively, you can add it into "~/.bashrc", enabling to automatically run this command for each new terminal.

3. Continue to finish Core ROS Tutorials 2-6.

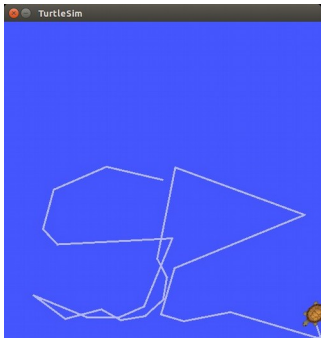
   Following the instructions in Tutorials 2-6, you will be able to run the "turtlesim" 2D simulation environment. You can skip the "rqt_graph" section.

**What to turn in**

**PART A:**

You will need to ***write code to automatically drive the turtle*** to "draw" two letters: the initials of your first name and your last name (the image below is an example one generated by the instructor). While the turtle is moving, you will use the **rostopic** tool to print out all **messages** of your control signals, and save the messages in plain text.

**Students will create a ROS package that includes a single launch file in such a way that running the following command will bring up the demonstration of your program.**




You need to create a single file (in tar/zip/rar format), and name it as your last name (initial in uppercase) followed by the initial of your first name (uppercase). For example, the file name should be "ZhangS.tar" for the instructor. Make sure that extracting this file generates a folder of the same name that contains the following subfolders

- "launch", where your launch file locates
- "src", where saves your source code
- "others", where saves your "message" text, and the image of your name initials. Please include a readme file in this subfolder to list your collaborator's name (if any).

and the following files as necessary:

- CMakeLists.txt
- package.xml

We will use the following command to start the demonstration of your program, and the instructor might request live demonstrations as needed, which (if happens) should not be taken personally.

roslaunch ZhangS assign1.launch

~~Send an email, with your submission in the attachment (less than 1M), to both the instructor's and the TA's email addresses with subject line: **"class assignment for yymmdd"**, where yymmdd is the ***original*** due date. You can leave the body of the email empty.~~

Updated on 9/7/2019: please upload your submission to Blackboard (less than 1M). Your submission can be late by at most 2 days (48 hours) without penalty. Throughout the semester (in all programming assignments), each student has 4 "late" days allowed. After that, submissions will not be graded.

**PART B (optional, but highly suggested, and will be considered in participation evaluation):**

Each student will post a new thread on Piazza, introducing yourself (programming, research experiences, or anything related) – just a few sentences will be sufficient. It will serve as the student's self-introduction and be used later for forming final project teams.

If you have any questions about this assignment, feel free to ask the instructor, the TA, or both.