Project 1
Learning Robot Operating System (ROS)

Assigned: January 29, 2019
Deliverable 1 (Task 1) Due Date: February 5, 23:59:59
Deliverable 2 (Tasks 2 & 3) Due Date: February 15, 23:59:59

Note: Project 1 has a four-day grace period for each deliverable. Late submissions will lose 25% points per day of that specific deliverable.

Introduction

In this project, the objective is to set up the ROS development environment on your computer, write a “Hello World” program using ROS, get the Turtlebot simulation working, and test the mapping capability provided by ROS. This project is an opportunity for you to get ready for the final project using the Turtlebot 2 robot and understand open-source software for robotics applications.

Project 1 must be done individually, although group discussion is encouraged.

Task 1: Setup Development Environment

You are required to finish the following steps:
1. Install Ubuntu 16.04 LTS:
   http://releases.ubuntu.com/16.04
   Search the ISO file named: ubuntu-16.04.5-desktop-amd64.iso
   IMPORTANT NOTE: Please install the 16.04 version, you may have issues if other versions of Ubuntu are installed.

   Note: if you don’t want a physical installation, you can use Oracle VM VirtualBox to install a virtual machine under Windows or Mac OS X. Follow the steps in the link:

2. Install ROS Kinetic following: http://wiki.ros.org/kinetic/Installation/Ubuntu
   IMPORTANT NOTE: Please install the exact ROS version and follow the exact steps!

3. Go through ROS tutorial 1.1.1-1.1.9, i.e., “Beginner Level” from the beginning (1.1.1) to “Using rosd to edit files in ROS” (1.1.9). The tutorial is available at: http://wiki.ros.org/ROS/Tutorials. Follow the
tutorial: http://wiki.ros.org/turtlesim to play with the Turtlesim tool and understand how ROS works (to some extent). For Turtlesim, make sure you follow the tutorial under ROS Kinetic.

What to submit:

You are required to submit a screenshot of a working turtlesim window for grading, similar to Figure 1. Name the figure T1_firstname_lastname.png (or jpg) and submit to the Canvas portal named P1-T1.

Task 2: Understand the Turtlebot Simulator

Go through the following TurtleBot Simulation tutorial using Gazebo in ROS:
“Learning with the TurtleBot in Simulation”: 1-6 and 8-9 (no need to do 7)
http://learn.turtlebot.com/

(Updated on Feb. 5) For Ubuntu 16.04 + Kinetic:

When following the Turtlebot software installation step, you will be linked to http://wiki.ros.org/turtlebot/Tutorials/indigo/Turtlebot%20Installation.

1. Run the command in Section 1.2.2, replacing ‘indigo’ with ‘kinetic’ and omitting ‘ros-indigo-rocon-remocon’ and ‘ros-indigo-rocon-qt-library’.
2. Run ‘sudo apt-get install pyqt5-dev-tools’.
3. Then go to section 1.3.2 and execute those commands, but in the ‘rocon’ section replace the word ‘indigo’ with ‘kinetic’.
4. Run the remainder of the commands as they are written. (Updated on Feb. 6: You may get build errors near the end of the catkin_make processes. But it will not affect Task 2; all needed nodes for Task 2 should have been built before the errors happen.)

Ubuntu 14.04 + Indigo does not have the problem.

You will need to (1) customize a simulated world (tutorial 6), build a map of your simulated world (tutorial 8), and navigate the robot in the world (tutorial 9).

FYI, additional information of the simulator can be found: http://wiki.ros.org/turtlebot_simulator

What to submit:

You are required to submit snapshots of (1) your customized simulation world, (2) simulated 3D scene, and (3) simulated mapping result, as shown in Figures 2-4, respectively. Include all files in a tar or zip file named T2_firstname_lastname.tar (or zip) and submit to the Canvas portal named P1-T2.
Task 3: Write a “Hello World” Program

You need to go through 1.1.11 (if you use C++) or 1.1.12 (if you use Python), as well as finish 1.1.13.

Write a ROS node that uses the turtlesim simulator to draw the **inner-boundary of the Mines’ logo** shown by the **RED CURVE** in Figure 5:

- Create a new package called `mines_lastname_firstname` (e.g., `mines_zhang_hao`). Your package should contain an executable with the same name as the package.
- Your program should assume that `roscore` and `turtlesim_node` have been started independently.
- The drawing should not take longer than two minutes to complete.
- You should NOT use the teleport services offered by turtlesim.

**For CSCI 473:**
- An open-loop control can be implemented (i.e., only tell the turtle the driving directions without pose feedback).
For CSCI 573:
- A close-loop control is required, i.e., designing a subscriber to receive turtle’s pose data and use it to improve navigation accuracy.

What to submit

You are required to submit (1) your code, (2) a README to detail how to compile and run your code (and other information), and (3) a snapshot of the result. Include all files in a tar or zip file named T3_firstname_lastname.tar (or zip) and submit to the Canvas portal named P1-T3.

Submission and Grading

Project 1 has two due dates, with a four-day grace period for each of the two deadlines. The project deliverables must be submitted to the Canvas, but you are encouraged to use version control systems (GitHub or GitLab) to manage your code.

Grading: Your report will be graded as follows:

- 2/10: Task 1.
- 3/10: Task 2.

Useful Links

A comprehensive (free) book of ROS: https://cse.sc.edu/~jokane/agitr/