ECET 499/581 - Programming Robots with ROS

Lab 1a – Due Sept 13, 2019

1. In groups of two or three, you will set up your linux workspace to run ROS python scripts in a catkin workspace called "ros_ws" (not "catkin_ws"). Follow the non-sudo procedures in the links on the website for setting up the workstation, Hello, Baxter, and the Simulator starting with the "rosdep" line. The links are copied here:

http://sdk.rethinkrobotics.com/wiki/Workstation_Setup

http://sdk.rethinkrobotics.com/wiki/Simulator_Installation

Use the name "crlBaxter" for the robot, with no ".local".

Some basic Linux commands to be aware of:

Process status: "ps" and "ps –a" These commands list the status of processes.

Killing a process: "kill -9 <PID> where "<PID>" is replaced by the process ID found using the "ps" command. This kills any pesky processes that did not exit properly.

Running a program in the background: append the " &" character to the end: "gedit &" will run gedit in the background so you get access to the command shell. Also, if a process is already running (e.g. "gedit"), you can press ctrl-Z to stop the process. Then, you can use the background command, "bg", to re-start the process in the background. (So, the sequence, "gedit", "ctrl-Z", "bg" will also run "gedit" in the background.)

Nothing to turn in for this one.

2. You are using catkin_make to build executables. It is beneficial to go through the catkin_make tutorial on "Creating packages" and "Building a ROS package" starting at:

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

   Then, follow the instructions to build the talker and listener under the basic tutorial:

http://wiki.ros.org/rospy_tutorials/Tutorials/WritingPublisherSubscriber

Upload your talker.py and listerner.py files to BlackBoard to Problem 2.

Make sure your names AND COURSE NUMBER (e.g. "ECET 499") are in the comment line at the top of the files and follow the coding and commenting conventions on the website. Your grade will be impacted by comments!!

3. Modify your talker and listener to implement a simple menu on the talker side that sends joint "jog" commands to move a particular joint by a static increment for each key press. (Use the keys 1-7 to control joints 1-7 on the left arm, with the corresponding shift keys causing negative jogs, and keys q-u to control the right arm, with shift-Q – shift-U causing negative jogs.) Modify your listener to receive these menu commands and move Baxter in response. Be careful to implement good clean-up procedures, as noted in the tutorials. Demonstrate your program with the Baxter simulator implemented with Gazebo.

You may look at `ros_ws/src/baxter_examples/joint_position_keyboard.py`, but do not copy it. Implement your own menu code. (The “bindings” approach is overkill, but interesting, nonetheless.)

Set up a time to demonstrate your menu-based programs and upload the files to Problem 3. In the comments, provide a detailed explanation of how your programs work (in particular, the menu process) and follow coding conventions on the website.