CSCE 439/839: Robotics: Algorithms and Applications, Spring 2020
Homework 1

Started: Monday, Jan 13th
Checkoff 1: Tuesday, Jan 21st
Due: Beginning of class Friday, Jan 31st

Instructions: This homework is an individual assignment, collaboration is not allowed. If you discuss any problems with others, please note this on the assignment as described in the syllabus. Also note any materials outside of lecture notes, course textbooks, and datasheets that you used. Show your work and describe your reasoning to get partial credit if your solution is incorrect.

You should also make sure that you properly label and **describe** any figures or plots that you include in your writeup. You will not receive full credit if you do not explain these. In addition, you should refer to your code where needed to answer the questions (e.g. say "See file mynode/launch/test.launch for this problem, which ...").

You must turn in a pdf of your assignment on handin. Make sure to answer questions in complete sentences and explain answers as needed. **You must also turn in your code for all parts of this problem and a PDF version of your report by visiting** `http://cse.unl.edu/handin/.` Failing to electronically turn in your code will result in a 10 point penalty on this assignment. Points may also be deducted for coding errors, poor style, or poor commenting.

If you use any code from an online or other source you must cite the source in the comments. Otherwise it is considered plagiarism, which we check for!!

Note regarding checkoffs: There are a number of checkoffs associated with this assignment, you can wait and get them all checked off at once or as you complete them unless a due date is associated with a checkoff.

Name:

Problem 1. *(5 pts)*[1] *(To be completed at end of assignment) Approximately how much time did the total assignment take? Which sub-problem took longest and how much time did it take? Are there any questions that need clarification?*

Problem 2. *(10 pts) Installing ROS*[2]*. A VMware image of Ubuntu 18.04 and ROS Melodic already setup is available here:* `https://unl.box.com/s/sdv28zs7c12crbdpxalmt0t6ds2gbrve`*. This is a 6GB file zip file that extracts to a 16GB image. You can download a copy of VMware by going to* `https://unlengineering.onthehub.com` *or download the free VMware Player software. You can login to the* `CSCE439` *account with the password* `robotsrule!` *(please change it).*

As an alternative, you can manually install **ROS Melodic** *(one of the ROS releases). I would either recommend installing Ubuntu 18.04 natively on your computer and then install ROS or you can do all of this inside of a virtual machine such as VirtualBox (open source) or VMWare (free to CoE students or the free vmware-player can be used). VMWare is slightly preferred as VitualBox occasionally has had bugs that make it difficult to use with the radios we will use in this class.* **Make sure that you install Melodic as I will assume you are using this version of ROS throughout this assignment and course.**

Checkoff: This checkoff is due by Tuesday, Jan 21st. Bring your computer[3] to office hours and have the TA or instructor sign below to indicate that you have completed this step.

[1] Each HW counts equally in your overall grade, even if homeworks have different point totals. This one is out of 80 points for 439 and 80+20 points for 839.

[2] You should explore `http://www.ros.org` as there are many more helpful hints and documentation that are not referenced in this homework.

[3] If you do not have a laptop, please talk to me now.

Problem 3. *Turtle Sim. Complete section 1.1 the beginner tutorials found at* `http://www.ros.org/wiki/ROS/Tutorials`. *This goes through a "turtle" tutorial that you will need to use to answer the below questions.* **Note:** *If you downloaded the VMWare image step 1.1.1 is complete and there is already a catkin_ws directory with other source files in there (you can ignore these for now, we will use them later in the semester).*

a). *(5 pts) Use* `rostopic pub` *from the command line to write "CS" with your turtle. Attach a picture showing your final "CS" as drawn in TurtleSim.*

b). *(5 pts) Give all the commands you used to write CS in TurtleSim.*

c). *(5 pts) Show the* `rqt_plot` *of the x and y position, speed, and angle (so four lines) of your turtle as it writes CS. Augment the plot to show where drawing of each of the letters occurred.*

Problem 4. *Creating New ROS Nodes*

a). *(10 pts) Now create a new ROS node (see* `http://www.ros.org/wiki/ROS/Tutorials` *for details, you can use either C++ or Python) that publishes messages to write CS (similar to the manual* `rostopic pub` *from problem 3a). Include a picture of the results. Also, describe how you dealt with the delays between sending messages. Remember that you need to comment the code appropriately for full credit (and turn it in using* `handin`*).*

b). *(5 pts) Create a launch file that starts both the TurtleSim node and your new node. Include a copy of the launch file in your report.*

c). *(5 pts)* **839 Only:** *Create* **one** *new node that writes "CSE" using three different turtles. Include a picture of the results.*

d). *(5 pts)* **839 Only:** *Create three different nodes (or they could be one node instantiated three different times) that each control a different turtle to write CSE (one turtle per letter). Include a picture of the results. The result should be the same as for the prior question.*

Problem 5. *(This will be distributed during the first lab) In the kit I gave you in lab you have an Arduino and a breadboard with some buttons. In this problem, you will program the Arduino to read the buttons and then send this data to a ROS node you will create to ultimately start moving your turtle. First start out by installing the Aruino software (also available in the VMware image).*

a). *(5 pts) Program your Arduino to read the inputs of the buttons (you can refer/use some of the referenced libraries in the Arduino software). Make sure to include your code in* `handin` *and tell me here if you wrote your own code or if you used an existing library (include a link to it in your report if you used an existing library). You should print out the values of the buttons over the serial port in an easy to read format (e.g. labeled "button1: on" etc.). Include a few lines of this output in your writeup and annotate it to indicate what is going on.*
Checkoff: Show the output of this code to the instructor.

Problem 6. *Creating a ROS interface to the Arduino.*

a). *(10 pts) Create a binary protocol that includes at least a start byte (a unique byte at the start of each packet) and a checksum to transmit all of the button information from the Arduino. Note that by binary protocol, I mean that you should transmit the byte 0x01 instead of the string 'on' or the number "1". You can use any type of checksum you would like (e.g. just the sum of the bytes, XOR, etc).* **Describe** *your protocol in this report.*

b). *(5 pts) Create a ROS node to process the data sent from the Arduino. You can use the* `read_serial.py` *python script (on the course website). This creates a node that publishes each byte it receives on a topic called* `\rx`*. You can subscribe to this topic to get the data, but you should create a new node (in C++ or python) to do the processing of the data received on this topic. Make your node compatible with the* `turtlesim_node` *ROS node. By this I mean that your node should publish the same topics that can be consumed by the* `turtlesim_node` *node does.*

c). *(5 pts) Figure out how to use the buttons to drive around the turtle in TurtleSim. Include a picture drawn by you while controlling the turtle with your joystick. Be creative in the picture you draw.*

d). *(5 pts) Now create a launch file to launch all the nodes to drive around the turtle with your Arduino.*

e). *(5 pts) Include a printout of the* `rqt_graph` *of your overall system.* **Remember, that all problems that ask for figures or launch files should also include a description to receive full credit.**

f). *(10 pts)* **839 Only:** *Write additional ROS and Arduino code to enable sending a boolean value on the topic* `ArduinoLED` *that will cause the LED on the Arduino to turn on or off based on the value. This will require modifying your ROS node, sending a serial command (modification of the* `read_serial.py` *code), and processing this command on the Arduino. Make sure to include a header and checksum on the binary packet you create (to allow for future expansion). Configure one of the buttons to turn on/off the LED.*
Checkoff: Show turning on and off the LED with the buttons to the instructor.

Do not forget to fill in the amount of time you spent on this assignment in Question 1.