CS/ME/ECE/AE/BME 7785

Lab 1

Due: January 24, 2024 by 4 pm

Overview

The objective of this lab is to get you familiar with the robot and get you started writing code. Specifically, this lab consists of two parts:

- **Part 1a (individual)**: run provided Turtlebot3 code to get experience interfacing with the robot.
- **Part 1b (group)**: answer the ROS related questions.
- **Part 2 (group)**: create your own image processing script that enables the robot to identify a desired object in its camera image.

The code generated in Part 2 can be reused in Lab 2 to have your robot move to track whichever object you have chosen.

Note Part 1 will take far less time than Part 2, so budget your time wisely.

Part 1 is an individual assignment, each person must demonstrate ability to run the robot from a machine that they have access to. You can use your own laptop or lab machines. You can not use your partner's laptop to demo.

We strongly encourage you to use all available resources to complete this assignment. This includes looking at the sample code provided with the robot, borrowing code from online tutorials, and talking to classmates. You may discuss solutions and solve problems with other groups in the class, but each group must submit their own solution. Multiple groups should not jointly write the same program and each submit a copy, this will not be considered a valid submission.

The submission instruction and the grading details are included at the end of the document.

Handle your robot with care!

- **Battery Handling**: The robots can be powered up using 2 methods:
  1. Power the OpenCR board through the AC adapter. This is recommended if you're debugging code and the robot doesn't need to be mobile.

2. LiPo (Lithium Polymer) battery: LiPo batteries are high energy density batteries which are a popular power source for mobile robots such as UAVs, Rovers, (and Turtlebots!). They must be operated with care as they may explode and catch fire if mishandled.

Here are some tips for handling LiPo batteries:

1. Do not hot swap batteries. This means do not plug in the AC adapter and switch batteries. Always power the robot down fully before removing and plugging in batteries.
2. NEVER puncture/pierce the LiPo battery and keep it away from sharp objects.
3. Do not leave batteries plugged into the robot, always unplug them after use.
4. If you notice any swollen/puffed batteries, notify the staff at once.
5. Use the LiPo battery chargers for recharging batteries. Red LED indicates charging, Green indicates charged.
6. If you are the last one leaving the lab, please unplug batteries from the chargers.

A constant beeping sound indicates low battery. Ensure to save your work and shutdown your robot before replacing the battery.

- **Robot shutdown procedure**: Before plugging out the power supply, please run the command:

```
sudo shutdown now
```

Wait for the LIDAR to stop spinning and the Raspberry Pi's green LED to stop blinking. At this point, it is safe to remove the power supply. Abruptly powering off may lead to your SD card getting corrupted and data loss.

Part 1a: Interfacing with the Turtlebot3

All instructions for the Turtlebot3 are available online at

http://emanual.robotis.com/docs/en/platform/turtlebot3/overview/

Complete the following steps:

1. Ensure you've completed the PC Setup and Turtlebot3 package installation as outlined in Lab 0.

2. Note: Section 3.1.5 has you place the line `export ROS_DOMAIN_ID=30 #TURTLEBOT3` in the bashrc. This number should be edited depending on what robot you are using (the robot domain IDs are listed on the OLED on boot).

3. You will be able to ping, SSH, transfer files, and exchange ROS 2 messages to the robot if you are on the 7785/7785-5G network. Password is *robot-wifi*

4. Make sure you can access the Raspberry Pi on the robot. You will need to SSH (secure shell) into the robot using the command,

   `ssh burger@192.168.1.152`
   (The IP 192.168.1.152 should be replaced by your Raspberry Pi's IP or hostname)

   The password is "**burger**" for all the robots. Once entering the password you should see the terminal environment on the Raspberry Pi of your Turtlebot3!

5. Complete the Quick Start guide on the above website (listed as item 3 on the left side menu) and verify that you are able to teleoperate the robot using the keyboard (Sections 3.5 and 3.6.1.1). Optionally, feel free to run any other examples, such as Turtlebot Follower to test out the robot functionality. **Note:** For ROS2 the robot and your laptop have to be on the same network for messages to communicate (e.g. run teleop on your PC rather than on the robot). Thus, switch the network you are on to CS3660 for controlling the robot.

Part 1b: ROS Related Questions

Please answer the following questions (from Lab 0) and submit them as a PDF with your code submission from part 2.

- What is a ROS_DOMAIN_ID? (2pt)
- What is a node? (2pt)
- What is a topic? (2pt)
- What is a message? (2pt)
- What is a subscriber? Write the syntax to create a subscriber that subscribes to the topic `amazing_int`, which takes message of type `UInt64`, and uses the callback function `magic_fun`, in C++ or Python. Note: This should only be a few lines of code, not a full script. (5pt)
- What is a publisher? Write the syntax to create a publisher that publishes to the topic `amazing_bool`, which takes message of type `Bool`, in Python. Note: This should only be a few lines of code, not a full script. (5pt)
- Can a node have multiple subscribers? Can a node have multiple publishers? (2pt)

Part 2: Object Following

Objective: Create a program to reliably locate and track an object of your choice from a real-time images stream on your computer. You are encouraged to use libraries such as OpenCV to complete this assignment. Note: This cannot be done using a pre-made tag tracker package like ArUco nor using premade ML packages like ImageAI.

In this part of the lab, you will use OpenCV to locate an object from your laptop's webcam. This is a valuable step towards Lab 2, in which you will track the object from your robots camera frame and add driving capabilities to your code. Feel free to use any additional python libraries you want, at the very least you should use cv2 (OpenCV). You are not expected to create image processing algorithms from scratch.

Create a new python script called **find_object.py**. This script should receive images from the webcam on your laptop and track the location of a specific object of your choice in the frame. Note: Again, this cannot be done using a pre-made tag tracker package like ArUco nor using premade ML packages like ImageAI. Once you have made the script it is often useful to make it an *executable*. To do this you must first make sure you have the type of environment being used in the first line of your code. For python this typically is,

```
#!/usr/bin/env python
```

Then, to make the file executable, using the command line in the directory (or with the full path specified) where your file is stored type,

```
chmod +x object_follower.py
```

You may now run your python script from the command line, when in the directory where it is stored, by giving the command,

```
./object_follower.py
```

Tips, Tricks, and Suggestions

Example code to capture and display your webcam with OpenCV in python can be found at

https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_gui/py_video_display/py_video_display.html

After grabbing the image, process the image to look for your object. An easy place to start is to use either the HoughCircles() functionality within OpenCV if your object is circular

```
circles = cv2.HoughCircles(cv_image, cv2.HOUGH_GRADIENT, 1, 90, param1=70, param2=60,
                           minRadius=50, maxRadius=0)
```

or use findContours() to find blobs of the same color:

```
im2, contours, hierarchy = cv.findContours(thresh, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
```

As described in class, these alone will likely not be enough to produce a reliable tracking algorithm. Be sure to prefilter your images (changing color space, blurring, sharpening, improving contrast, thresholding, etc). You are not required to use Hough circles or contours, if you prefer to try a different method that is completely acceptable so long as the object is tracked.

Once you have located the object in an image, print the pixel coordinate of the object and display some sort of marker or bounding box on the image itself.

We will provide balls of several sizes and colors for your use in the lab. Feel free to use any of them, or a different object of your choosing. For this lab, the object can be at any distance from the webcam that you choose. However, it cannot take up the entire frame nor only be tracked at a single distance exactly (you cannot say it only works at 15cm from your monitor exactly).

Grading Rubric

The lab is graded out of 100 points, with the following point distribution:

Answer ROS questions from Part 1b	20%
Successfully run teleop or example code on the robot (individual assignment)	15%
Find your chosen object in images in >65% of the frames	55%
Print the pixel location of each identified shape	5%
Annotate output images with bounding boxes or markers	5%

Submission

Part 1a: Perform a live demonstration of you running the robot to one of the course staff by the listed deadline. You can use your own laptop or the lab machines, you can not use your partner's laptop to demo Part 1. There is no code to submit.

Part 1b and 2:

1. Perform a live demonstration of you running the robot to one of the course staff by the listed deadline.
2. Put the names of both lab partners into the header of the python script. Put your python script, any supplementary files, and a PDF with your answers to Part 1b in a single zip file called
Lab1_LastName1_LastName2.zip and upload on Canvas under Assignments–Lab 1.
3. Only one of the partners needs to upload code and answers.

We will set aside class time and office hours on the due date for these demos, but if your code is ready ahead of time we encourage you to demo earlier in any of the office hours (you can then skip class on the due date). Office hour times are listed on the Canvas homepage.