CS/ME/ECE/AE/BME 7785

Lab 1

Due: September 07, 2018 at 3 pm

Overview

The objective of this lab is to get you familiar with the robot and to get started writing code. You will accomplish two things: 1) run provided Turtlebot3 code to get familiar with running the robot, 2) create your own image processing script that enables the robot to identify the location of a ball in the frame. Note that the first part will take far less time than the second, so budget your time wisely.

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 pieces of code from online tutorials, and talking to classmates. You may discuss solutions and problem solve with others in the class, but this remains an individual assignment and each person must submit their own solution. Multiple people should not jointly write the same program and each submit a copy, this will not be considered a valid submission.

Lab

First, make sure your ROS setup is working and you are able to run code on the robot. All instructions for the Turtlebot3 are available online at

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

To test your setup:

1. First 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 should be "burger" for all the robots. Once entering the password you should see the terminal environment on the Raspberry Pi of your Turtlebot3!

2. Run the Bringup instructions (listed as "7. Bringup" on the left side of the above website) Note that you may have to refer to parts of "6.1 PC Software Setup" the first time you do this. On the robot itself, edit the .bashrc file such that the IP address listed under ROS_MASTER_URI is the IP of the laptop where you are running roscore. Don't forget to source ∼/.bashrc whenever you edit that file to make the changes take effect.

3. Once you run Bringup, run the "8.2.1 Teleoperation Keyboard" demo using your keyboard to control the robot.

4. (Optional) Run any other examples like Turtlebot Follower.

For the second half of the lab, create a new python script called `ball_follower`. Some potentially useful imports would be cv2 (OpenCV) and numpy :

find_ball.py: This script should receive images from the webcam on your laptop and track the location of a specific ball in the frame. 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 ball_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,

```
./ball_follower.py
```

Some 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 circles. An easy place to start is to use the HoughCircles() functionality within OpenCV:

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

As described in class, this alone will likely not be enough. Experiment with ranking the returned circles based on color or other properties. Normalizing and slightly blurring the image can also improve the performance of HoughCircles().

Once you've located the ball in an image, this script should print the pixel coordinate of the ball and display some sort of marker (or text pixel location) 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 ball of your choosing. For this lab, the ball can be at any distance from the robot that you choose.

Grading Rubric

Successfully run example code on the robot	25%
Find specific ball in image >50% of the time*	65%
Print the pixel location of the ball	5%
Display the pixel location of the ball on the image (through a marker or text)	5%

*We're setting a pretty low bar for ball detection here because there's already a lot going on in this lab and because we'll talk about ways to improve detection later in the semester.

Submission

Lab submission consists of two parts:

1. Your python script and any supplimentary files, in a single zip file called YourLastName_YourFirstName.zip uploaded on Canvas under Assignments–Lab 1.

2. A live demonstration of your code to one of the course staff by the date listed at the top of this page. We will set aside class time and office hours on the sue date for these demos, but if your code is ready ahead of time we encourage you to demo earlier (you can then skip class on the due date).