EE 144 - Lab 4:
Working with the physical robots

Yecheng Xiang X640545

Jianqing wu X641222

1.1 Lab assignments

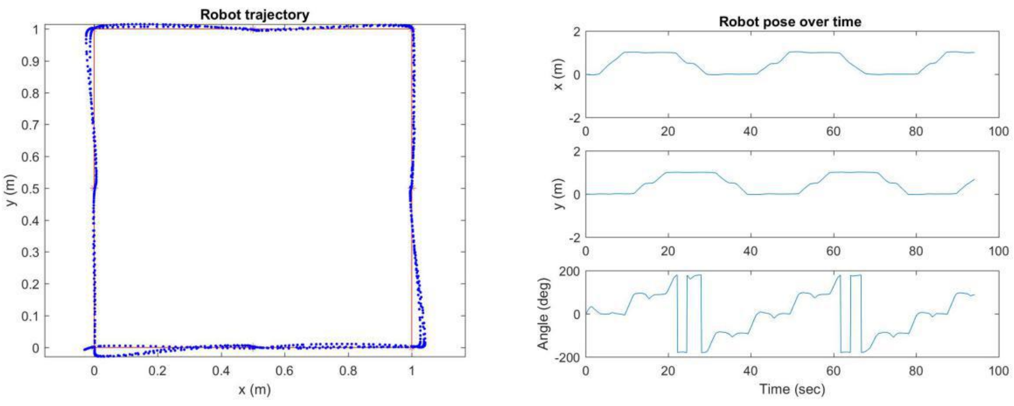
The following are the assignments for the lab. You should ideally be able to complete the assignments in the 3-hour period of the lab session, but you may also continue your work during the following days if needed. For this lab, you will be working in two-person teams, and you should report your work in a team lab report, due one week after the start of your lab session. In your submission you should also include your commented code, as described in the instructions.

Assignment 1: Waypoint navigation

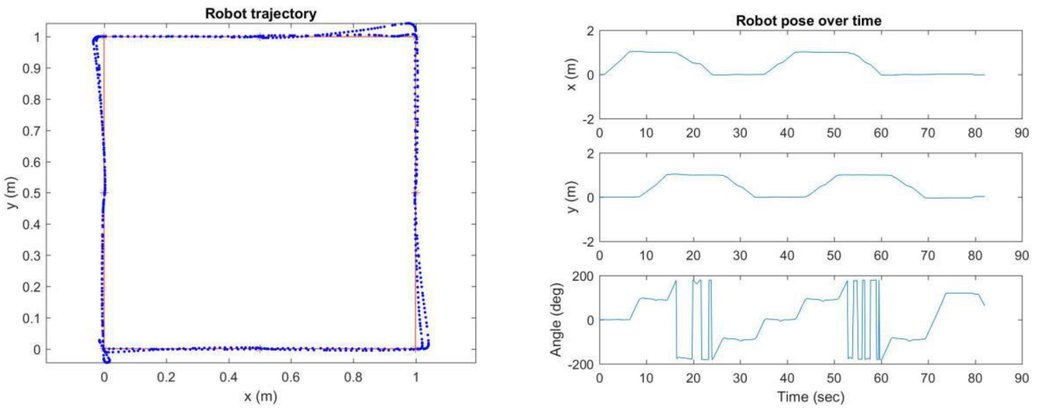
You are provided with an implementation of waypoint navigation, which will command the robot to move continuously on a square (see Lab 2 for a description). To run this code with the real robot, please follow the instructions for setting up the robot, posted on iLearn.

• Try different values of the control gain kp and the distance threshold dthresh, and observe the behavior of the robot when it tries to move through the waypoints. Select values that give good performance, and compare these values to the values you used in the simulator in Lab 2. If they are different, what may explain the difference?

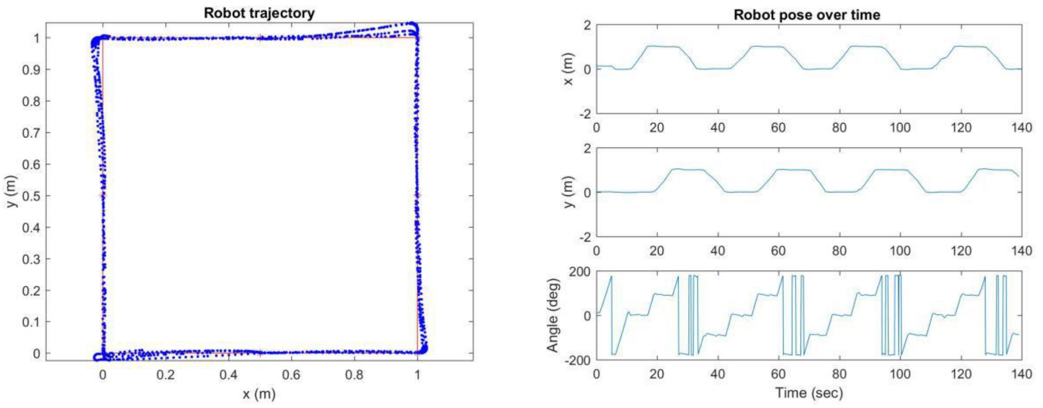
Here are the results from different kp and distance threshold.



Kp=2, dthresh=0.01



Kp=3, dthresh=0.01



Kp=3, dthresh=0.03

From the result we can see that with the same dthresh, the larger kp we choose, the faster the robot can adjust its orientation. But if the kp is too large, there will be oscillation and the system would became unstable. When the dthresh became larger, the overshoot would be smaller, but the dthresh cannot be too small, or it will affect the accuracy. Comparing to the simulation, the robot in the real world has more random disturbance. That's why the trajectory of the simulation seems the same every loop while the trajectory for real robot between two loops are more likely to be different. In real world, we are more likely to choose larger kp or dthresh than in the simulation.

• After choosing values for the parameters that will allow the robot to closely follow the desired square trajectory, let the robot complete the square several times. Compare the trajectory plot that you see in Matlab with the actual trajectory of the robot in the world. What do you observe? What explains the difference?

After several loops the square in the real world will slightly move from the original square, meanwhile the trajectory in Matlab doesn't change much. I figure this is because the trajectory was plotted based on counting the loops of the wheel. But in real world the real distance the robot goes may have slight difference from the measurement from the wheel. So after several loops, the slight difference will become observable.

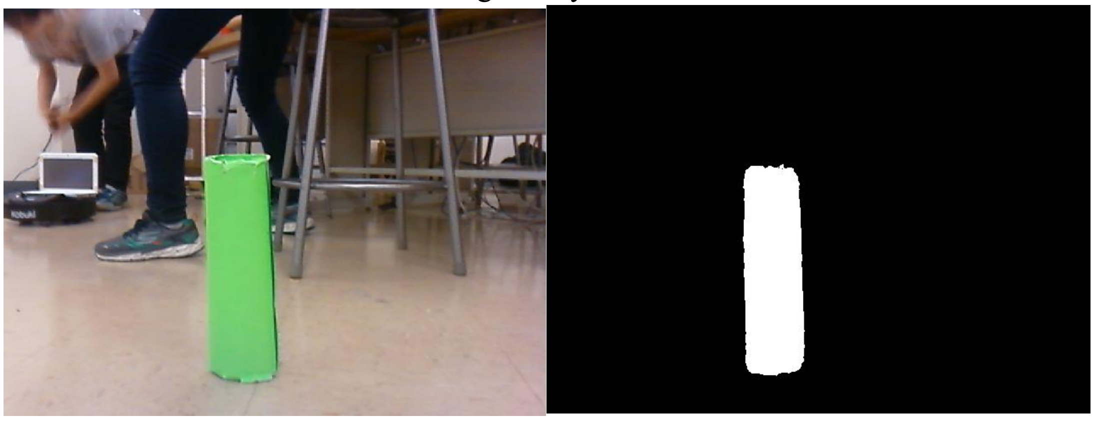
Assignment 2: Color-based tracking

In this assignment, your goal is to implement color-based tracking, as in Assignment 1 of Lab 3. We will here use colored cylinders, instead of the colored balls used in the simulator.

You are provided with the file camera example.m which reads and stores 20 images from the netbook's webcam, and then displays the images. Run the code, after changing the IP address as required, to make sure that you can access and use the camera on the robot. Then, implement the following:

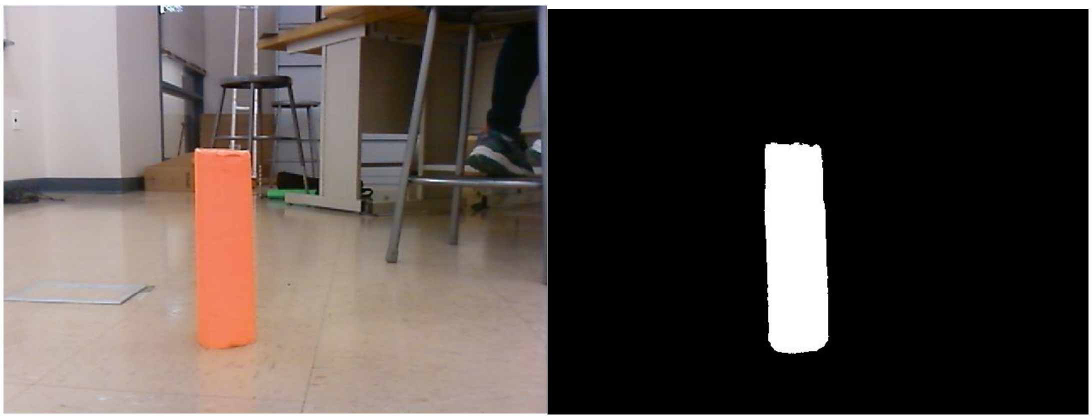
• Find appropriate thresholds on the Hue of each pixel, so that we can detect the green cylinder in the images (similarly to Lab 3). For each image, your code should create a binary image where the pixels that are deemed to belong to the green cylinder are set to one, and everything else is set to zero. Try to find threshold values that will lead to as few errors as possible (cylinder pixels that are missed, or non-cylinder pixels that are classified as belonging to the cylinder), with different cylinder orientations and distances.

The hue threshold we choose for the green cylinder is 0.2~0.35. And here is the result



• Do the same thing using an orange cylinder. Does the detection work equally well? Try using additional thresholds on the Saturation and Value of each pixel (in addition to the thresholds on the Hue). Does this improve performance?

Using hue value only will give you a lot of disturbance. So we added Saturation value as well as the value value. The threshold for saturation is 0.5~0.8. The threshold for value is 0.8~1. And here is the result.

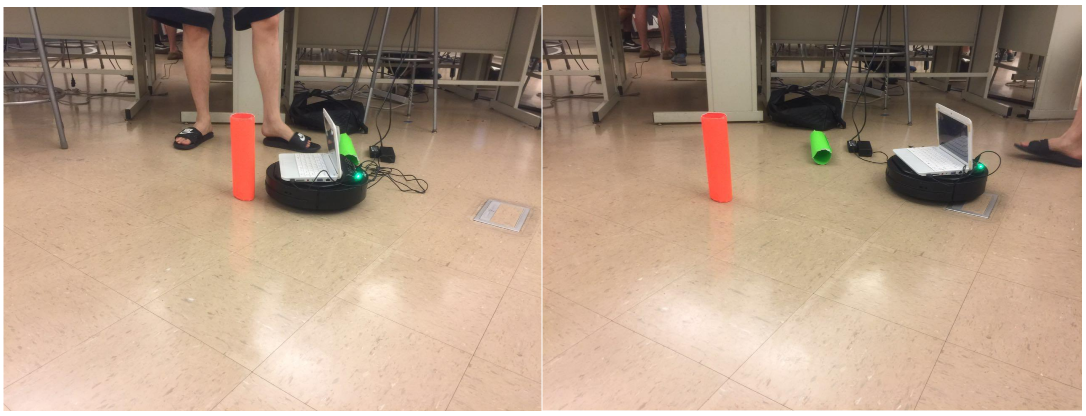


• Write a P-controller, similarly to Lab 3, to make the robot track the green cylinder. The robot should rotate in place, so that the cylinder appears in the middle of the image. Tune the controller gain so that the robot can successfully track the cylinder as it is moved within its field of view.

The kp we choose for rot_vel is 0.003, and the kp for lin_vel is 0.00002. Here is the result.

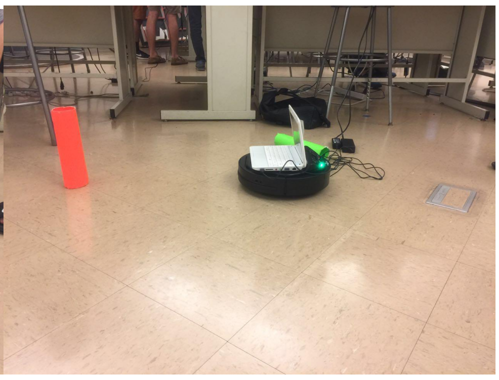
Befor After

Before After



Before After

Before


After