# Control of Mobile Robotics

# CDA4621

# Spring 2019

# Lab 2

# Wall Following

# (Closed Loop Control)

# Total: 100 points

### Due Date: 2-28-19 by 8am

The assignment is organized according to the following sections: (A) Lab Requirements, (B) Task Description, and (C) Task Evaluation.

## A. Lab Requirements

The lab requires use of the "Robobulls-2018" robot hardware provided at no charge for the duration of the semester. Required software can be downloaded free of charge from the web. All labs are to be done by teams of one or two students depending on robot availability. Note that no diagrams or descriptions by hand will be accepted. Each group is required to submit its report through Canvas. Penalties will be applied for late submission (see syllabus). All document need to be in PDF while a single "zip" file must be uploaded to Canvas containing all requested files.

### A.1 Hardware Requirements

The "Robobulls-2018" is the main robot hardware used for the course.

### A.2 Software Requirements

Python

## B. Sensor and PID Descriptions

The main components for this lab consist primarily of "Distance Sensors" and "PID Control" to enable a robot to do "wall following".

### B.1 Distance Sensors

This section will provide you with necessary information to use the distance sensors and it will introduce you to some usual issues and pitfalls encountered. Understanding everything in this section is a prerequisite for the rest of the lab.

Read the datasheet for the distance sensors[1]. Make sure you are able to answer the following questions.

1. What is the distance measuring range of the distance sensors?
2. After the first measurement is available, what is the minimum time it takes the sensor to produce a new value?
3. Think of methods that would allow you to make sure that a distance sensor always works in its specified distance range.

---

[1] https://cdn-learn.adafruit.com/downloads/pdf/adafruit-vl53l0x-micro-lidar-distance-sensor-breakout.pdf

**B.2 PID Control**

This section describes a basic robot software closed loop control using distance sensors and motor actuators. Figure 1 presents a diagram for closed loop control, where robot velocity is set to be proportional to its desired distance to a goal, $r(t)$, and where the goal may be any object such as a wall or an obstacle. As the robot gets closer to the desired distance to the goal, the error $e(t)$ will modify the control signal $u(t)$ until such error becomes 0. In our case the control signal corresponds to robot motor velocity until it becomes 0, when the robot reaches a specific distance to the goal.

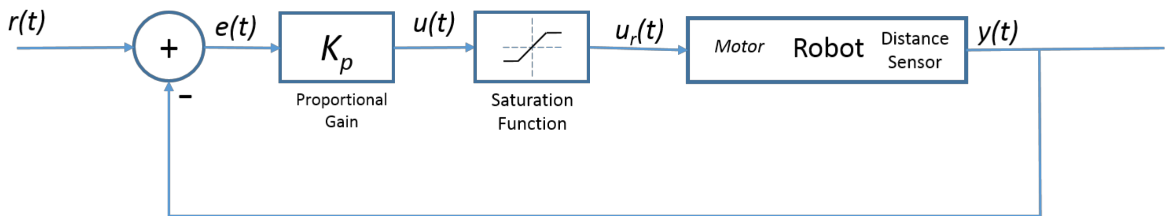


Figure 1: Close loop control of the robot velocity proportional to its distance to a goal.

Eq. 1 summarizes the diagram in Figure 1:

$$e(t) = r(t) - y(t) \qquad \text{(Eq. 1)}$$
$$u(t) = K_p * e(t) \qquad \text{(Eq. 2)}$$
$$u_r(t) = f_{Sat}(\, u(t)) \qquad \text{(Eq. 3)}$$
$$u_r(t) = f_{Sat}(\, K_p * e(t)\, ) \qquad \text{(Eq. 4)}$$
$$u_r(t) = f_{Sat}(\, K_p\left(r(t) - y(t)\right)\, ) \qquad \text{(Eq. 5)}$$

where:

$r(t) = desired\ distance\ to\ the\ goal$
$y(t) = distance\ from\ robot\ to\ the\ goal$
$e(t) = distance\ error$
$K_p = proportional\ gain\ or\ correction\ error\ gain$
$u(t) = control\ signal\ corresponding\ to\ robot\ velocity$
$f_{sat} = saturation\ function\ (see\ explanation\ below)$
$u_r(t) = control\ signal\ corresponding\ to\ saturated\ robot\ velocity$
$u_r(t) \rightarrow$ setSpeedsIPS($u_r, u_r$)

Note that function "setSpeedsIPS($u_r, u_r$)" was implemented in Assignment 1.

## C. Tasks Descriptions

This lab requires you to build a library to use the distance sensors. The main task is to implement a closed loop control for wall following.

**C.1 Task 1 – Distance Sensor Experiments**

Use the functions provided on Canvas under "Chapters/Sample Code.zip" in the "TOFSensors" folder to make two **scatter plots** of "real distance" vs "measured distance". "Real distance" refers to the actual distance in inches from the robot to the wall (measured with a tape). "Measured distance" refers to the distance also in inches but measured by the robot using the sensors. Make sure "real distance" is on the x-axis (independent variable) and "measured distance" is on the y-axis (dependent variable).

For both plots, take measurements from 20 centimeters to 220 centimeters in steps of 20 centimeters. In the first plot, for each real distance, take and plot at least 10 single measurements.

In the second plot, for each real distance, take and plot at least 10 averaged measurements of 7 samples each. In both plots include the line $y = x$. Include both plots in the report along with a brief comparison.

### C.2 Task 2 – Proportional Control

Implement a program called "wallDistance.py". The program should use proportional control (above equations 1-5) to control the distance of the robot to the wall. The control signal $u(t)$, corresponding to robot motor velocity, will move the robot either forward or backward from a random starting position perpendicular to the wall, until the robot reaches a distance of 5 inches, i.e. $r(t)=5$, as shown in Figure 2. The distance should be measured using the front distance sensor.

The day of the presentation, the robot will be placed at different distances from the wall (either closer than 5 inches or farther away). The robot should react accordingly. Also, after reaching the goal, the robot will be moved away from the equilibrium position. The program should automatically detect this change and go back to the goal position. Use only proportional control to control the speeds of the wheels. Do not add any extra stopping conditions.

For the report, you will be required to test 6 different values of $K_p$(0.2, 0.6, 0.9, 1.2, 1.5, 5.0). For each value you will need to provide a plot showing Distance from Wall vs. Time. On the day of the presentation, you will have to choose the value that you think is best.

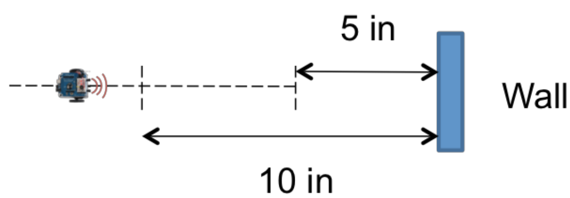


Figure 2: Wall Distance Task – starting at 10 inches

### C.3 Task 3 – Wall Following

Implement a program called "wallFollowing.py". The program should make the robot follow a wall while keeping a distance of 5 inches to the side (as shown in Figure 3). Use the same closed loop control implementation as in Task 2 with appropriate proportional gains to control front distance to the wall where the robot needs to turn. That is, as long as there is nothing in its front, the robot should move at full speed forwards (while following the wall in parallel). When there is a wall in its front, the robot should reduce its linear speed and it should not approach closer than 5 inches. On the day of the presentation, the program will be tested using multiple types of walls (both straight and curved walls).

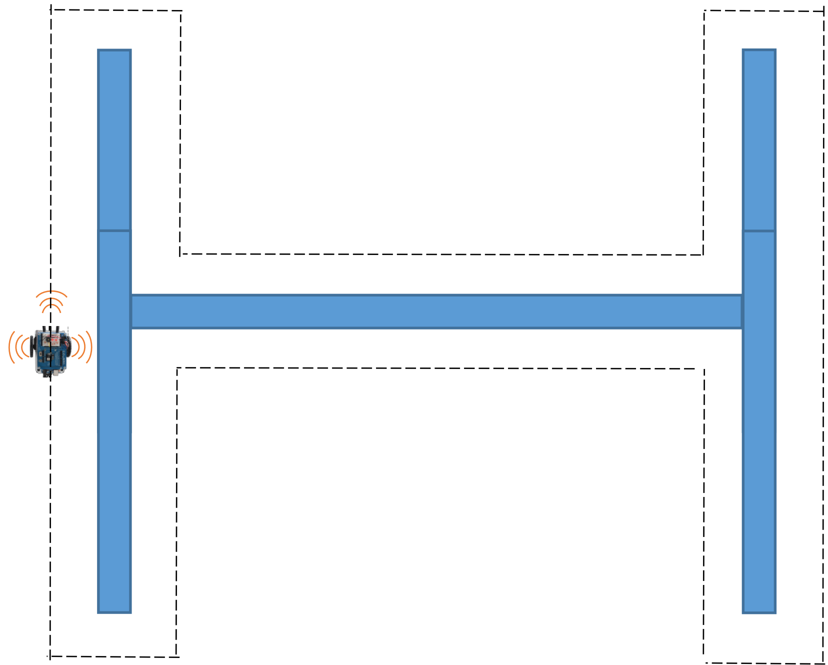

Figure 3: Wall following task

## D. Task Evaluation

Task evaluation involves: (1) a task presentation of robot navigation with the TA, (2) the accompanying code to be uploaded to Canvas, and (3) task report to be uploaded to Canvas. All uploaded documents will be scanned through copy detection software.

In what follows note that while Task 1 is not evaluated in the presentation, it is evaluated as part of the report (see section D.2). Also, the functions that need to be implemented are required to complete future labs.

### D.1 Task Presentation (70 points)

The task presentation needs to be scheduled with the TA. Close to the project due date, a time table will be made available online from which groups will be able to select a schedule on a first come first serve basis. All team members must be present at their scheduled presentation time. On the presentation day, questions related to the project will be asked, and the robot's task performance will be evaluated. To do so, the submitted code will be downloaded at that time from Canvas and uploaded to the robot.

The following table shows the rubric for tasks presentations:

- Task 2 (30 points)
  - Robot stabilizes at 5 inches for best $k_p$ (15 points)
  - Robot re-stabilizes when moved forward (5 points)
  - Robot re-stabilizes when moved backward (4 points)
  - Robot moves as expected for low $k_p$ (3 points)
  - Robot moves as expected for high $k_p$ (3 points)
  - Robot hits the wall (-5 points)
- Task 3 (40 points)
  - Robot can follow straight wall (20 points)
  - Robot can follow 90 degree turns (3 points)
  - Robot can follow 135 degree turns (3 points)

- Robot can follow 270 degree turns (3 points)
- Robot can follow 315 degree turns (2 points)
- Robot can follow 360 degree turns (2 points)
- Robot can follow small circular walls (2 points)
- Robot can follow big circular walls (3 points)
- Robot can follow a complex path (2 points)
- Robot hits front wall once (-2 points)
- Robot hits front wall several times (-5 points)
- Robot fails to detect front walls (-10 points)

To get full credit on each item, the robot must perform the specified action successfully on the first trial. If unable to do so, 75% credit will be given, if done on second trial, 50% if done on the 3rd, and 25% if done on the fourth. No more than 4 trials will be allowed per rubric item.

**D.2 Task Report (30 Points)**

The accompanying task report needs to be uploaded to Canvas as a PDF file together with ALL the files required to run the robot navigation. Upload all files into a single "zip" file. The task report should include ALL of the following (points will be taken off if anything is missing):

1. List of all code files uploaded to Canvas. All requested code must be included as a list in the main report, adding a one-line description to each of the files being uploaded (1 point).
2. The two plots required in task 1 (*"measured distance" vs "real distance"* for both single and averaged measures (5 points) and a brief comment stating the qualitative difference between both plots (5 points).
3. The 6 plots required for task 2 (*"Distance to wall vs time"* 5 points).
4. Brief explanation of the equations used in task 2 (5 points).
5. Link to a video (2 points) where the robot is shown performing "Task 2", "Task 3".
6. Small diagram showing the implementation of the wall following algorithm implemented on "Task 3" (4 points).
7. Conclusions (3 points) where you analyze any issues you encountered when running the tasks and how they could be improved. Conclusions need to show an insight of what the group has learnt (if anything) during the project. Phrases such as "everything worked as expected" or "I enjoyed the project" will not count as conclusions.

All plots must include title, axis names, units, sufficient tick marks and legend (if plotting more than one graph). Also, each data point should clearly be marked with a symbol. Plots of a variable vs time should always place time on the x-axis. See sample plot in Figure 4. Points will be taken off for not following the specified format.

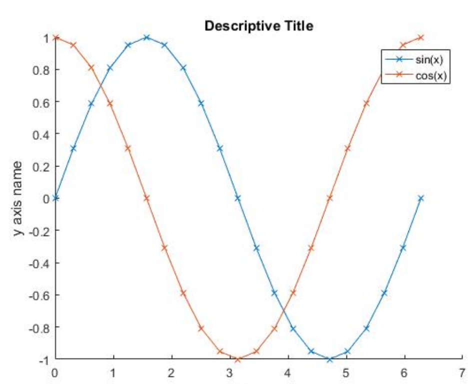

Figure 4. Sample Plot