CDA 4621 / CDA 6626

Grad Total: 110 points + Extra Credit

Undergrad Total: 100 points + Extra Credit

Spring 2024

Lab 2

PID & Wall Following

Due Date: 2-20-2024 by 11:59pm

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

A. Objective

This lab will teach you how to apply a PID controller to navigate parallel to a wall and stop at a desired distance from an end wall. The lab will also teach you about: (1) Distance Sensors, (2) Lidars, and (3) PID.

A.1 Distance Sensors

Distance sensors use light to detect objects. These sensors are called TOF (Time Of Flight) measuring return time values that may vary depending on type of surface and angles.

A.2 Lidar

The Lidar ("Light Detection and Ranging") is a range sensor that sends multiple laser beams (pulsed light waves) at different orientations to detect distances to objects (the traditional distance sensor sends a single beam). The Lidar computes the time it takes each pulse to return a measurement value.

A.3 PID

A PID controller uses distance sensors and Lidar to control navigation during wall following. While the Lidar provides 360^0 distance readings, robot front and rear distance sensors can directly compute distances to side and end walls. You will use only the "P" component in the PID controller. Figure 1 shows a PID controller controlling distances to walls by use of distance sensors where velocities are set proportional to wall distance error, i.e. difference in distance between current and desired.

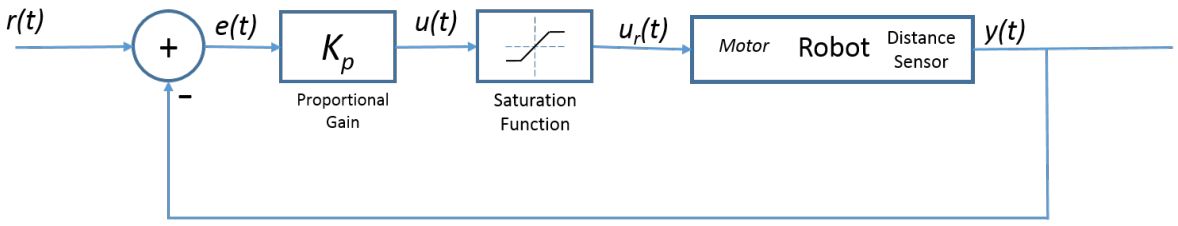

Figure 1. PID control of the robot velocity proportional to its desired distance to the wall.

The following equations summarize the diagram in Figure 2:

$$e(t) = r(t) - y(t) \quad \text{(Eq. 1)}$$

$$u(t) = K_p * e(t) \quad \text{(Eq. 2)}$$

$$u_r(t) = f_{sat}(u(t)) \quad \text{(Eq. 3)}$$

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

where:

$r(t) =$ desired distance to the wall
$y(t) =$ current distance from robot to the wall
$e(t) =$ 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 the saturated robot velocity

B. Requirements

Programming: Python

B.1 Physical Robot

See Appendix B.1.1 Robobulls-2023 or Appendix B.1.2 Yahboom 2024.

B.2 Robot Simulator

See Appendix B.2.1 Webots "RosBot".

Github Server: https://github.com/biorobaw/FAIRIS-Lite (see "README.md")

C. Task Description

The lab consists of the following tasks:

- Task 1: Motion with PID
- Task 2: Wall following with PID
- Statistic Task (Required only Grads, Extra Credit Undergrads)
- Extra Credit Task: Object Following

C.1 Task 1 – Motion with PID

The robot should use K_{pf} proportional gain control applied to its front sensor to move towards the end wall and control its stopping distance. Robot should use K_{ps} proportional gain control applied to its side distance sensors (left and right) to avoid hitting the side walls. Note that you can use either or both the distance sensor and Lidar. Robot should keep a min distance of 30cm from any of the side walls.

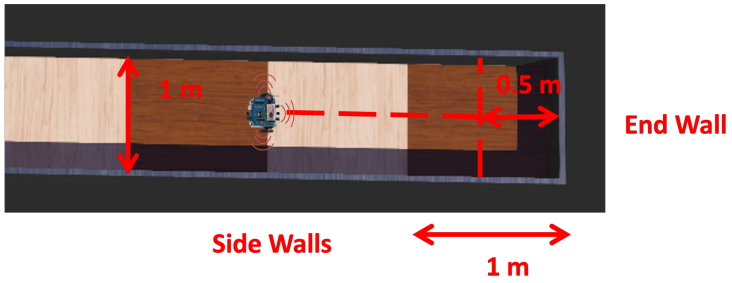

Figure 2: Robot starts 2m away from the end wall and should stop at the 0.5m mark. Note that in Webots each colored square measures 1m width, where 4 colored squares comprise a tile.

Robot should start at 2m away front the front wall and should stop at the 0.5m using the front sensor, as shown in Figure 2. Test the program using 6 different values of K_{pf} and K_{ps}, (0.1, 1.0, 2.0, 4.0, 8.0, 10.0). Find the one K_{pf} and K_{ps} values that you think works best. These values may be different from the ones previously specified. You can use the same K_p for all PID computations. The task should run for no more than 30 sec. Test the task starting at different

distances and orientations from the walls. You will need to continuously print distance measurements from the front and side sensor readings.

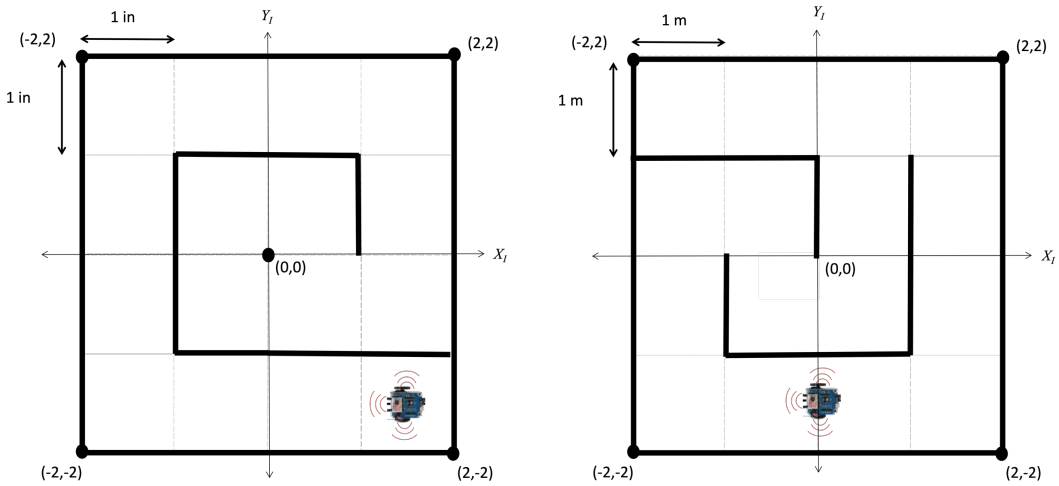

Figure 3. Robot mazes for wall following task.

C.2 Task 2 – Wall Following with PID

The robot should apply the best K_p proportional gain control, and you may use different K_{pf} and K_{ps} values. Consider the following:

- The robot does not need to use the front sensor controller for this task.
- If the robot reaches any end wall it should make a 90-degree and continue navigation for this task.
- During navigation, if no 90-degree turns are possible, the robot should make a 180-degree turn, and continue wall following in the opposite direction.
- Depending on the original orientation, the robot my end up following different paths. Also note that navigation does not require visiting every single square.
- The robot can start at any grid cell with any.
- The program should be able to perform wall following on either left or right wall relative to the robot.
- Your solution should take a parameter "left" or "right" to specify which relative wall the robot is following.
- Test the task starting at different positions and orientations.
- Two different world mazes should be tested using the same robot controller, as shown in Figure 3.

C.3 Statistic Task – Statistics on Wall Following

Perform statistics on the following:

- Run wall following at 3 different speed combinations while trying to minimize overall travel time.
- Analyze and graph min, max and variance of velocities, orientations, and distances to walls.
- Discuss your results including any navigation errors.

C.4 Extra Credit Task – Object Following

Follow the boundaries of the larger object shown in Figure 4, from the "S" starting location until completing a full cycle around the object.

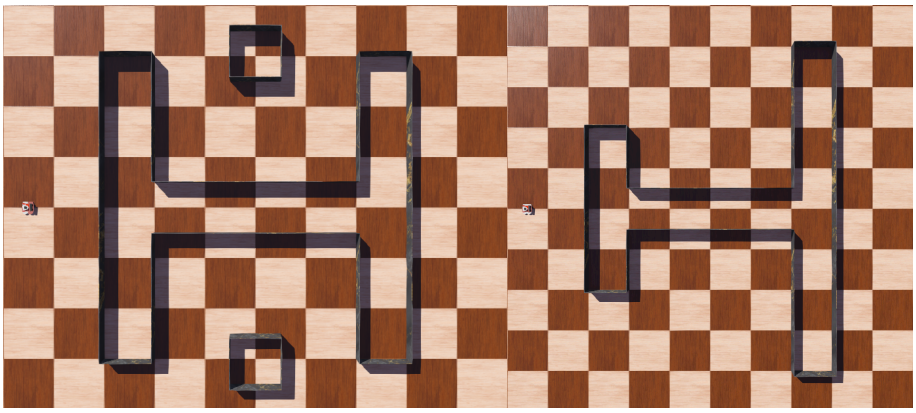

Figure 4. Different object configurations.

D. Task Evaluation

Task evaluation: (1) program code, (2) report, including a link to a video showing each different task, and (3) task presentation videos of robot navigation. Note that tasks are needed in future labs.

D.1 Task Presentation (90 points)

The following section shows the rubric for the different tasks:

- Task 1 (45 points)
  - Robot stabilizes at specified distance from end wall (10 points)
  - Robot stabilizes at specified distance from side walls (10 points)
  - Robot utilizes PID to reduce speeds proportionally to the front wall (10 points)
  - Robot utilizes PID to proportionally update distance from side walls (10 points)
  - Robot restabilizes when placed further or closer to the wall (5 points)
  - Robot hits walls (-5 points)
- Task 2 (45 points)
  - Robot performs left side wall following (10 points)
  - Robot performs right side wall following (10 points)
  - Robot makes the correct turns accordingly (15 points)
  - Robot follows end of walls accordingly (10 points)
  - Robot average travel speed less than 2 inches per second except for corners (-5 points)
  - Robot constantly exceeds 3 inches closer or further away from any wall (-5 points)
  - Robot hits walls (-5 points)
- Statistics Task (10 points) - Required for grad students. Extra credit for undergrad students.
  - Statistics are correctly computed, analyzed, and graphed (10 points)
- Extra Credit Task (20 points) – Extra credit for all students.
  - Perform correct object following for each maze (20 points)

D.2 Task Report (10 Points)

The report should include the following (points will be taken off if anything is missing):

- Mathematical computations for all kinematics. Show how you calculated the speeds of the left and right servos given the input parameters for each task. Also, show how you decide whether the movement is possible or not.
- Conclusions where you analyze any issues you encountered when running the tasks and how these 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.

- Video uploaded to Canvas showing the robot executing the different tasks. You should include in the video a description, written or voice, of each task. You can have a single or multiple videos. Note that videos will be critical in task evaluations.
- Deductions:
    - Handwritten text or images (-5 points)
    - Missing video link requires in person presentation (-20 points)
    - The submission format is not correct (see assignment details) (-10 points)
    - Failure to answer TA's email within 48 hours of the initial email (-50 points)
    - Additional deductions may apply depending on submission

Videos

Video needs to be clear and audible and must show the robot performing the correct path and showing the program outputs printed to the console, as outlined in the assignment. TA may ask you additional questions to gauge your understanding via Canvas Message or MS Teams. Failure to reply may result in point deduction.

- Task 1 Video
    - Record a single video of the robot preforming the PID task.
- Task 2 Video
    - Record a single video of the robot preforming PID task of wall following using the left side of the robot.
    - Record a single video of the robot preforming PID task of wall following using the right side of the robot.
- Extra Credit Video
    - Record a single video of the robot preforming the wall following with functions task.
    - Must show the use of the functions created and explain how they work at a high-level.

E. Lab Submission

Each student needs to submit the programs and report through Canvas under the correct assignment. Submissions should have multiple file uploads containing the following:

- The “zip” file should be named “yourname_studentidnumber_labnumber.zip” and should contain the Python file containing your controller. Controllers should be named as “Lab2_Task*X*.py”, where *X* is the task number. The zip file should contain any supporting python files needed to run your code. For example, if you created functions to perform actions and these are kept in a file, this file needs to be included. At the top of each python file include a comment with the relative path from FAIRIS_Lite that this file needs to be placed in. Example #WebotsSim/libraries/my_functions.py.
- Videos should be contained in a separate zip folder with the name “yourname_studentidnumber_labnumber_videos.zip” and added as a separate file upload to Canvas.
- The report should be in PDF and should be uploaded to Canvas as a separate file.
- All zip files must be in .zip format. We will not accept RAR, 7z, tar, or other.