COMP 581: Introduction to Robotics

Fall 2020
Instructor: Prof. Ron Alterovitz
Teaching Assistant: Sandeep Kumar <sandeep@cs.unc.edu>

Assignment 4: Robot Controller for Wall Following

Deadline

October 14, 2020 at 11:30am. For the late policy, see the course syllabus. **Before the due time, every student should individually submit their controller code in a single file on Sakai**; see specific instructions in the Code section below**.**

The Objective

The primary purpose of this lab is to implement forward kinematics and wall following for the Pioneer Robot.

Outline

This lab will be completed using the Webots simulator. You are provided with the Pioneer Robot in a few sample environments (see Sakai for the .wbt world files) and your job is to make it complete the tasks mentioned in the following section using a controller. You should design your controller such that it is able to complete the same task even in a different environment. We will be using different test environments (following the same basic structure) to run your controller and you will be evaluated on the basis of the performance of your robot in those environments. All measurements will be made with respect to the **frontmost center point** of the Pioneer Robot. Your robot should be designed to accomplish the task autonomously without any human intervention.

The Task

The task in this lab is to be completed in a single run.

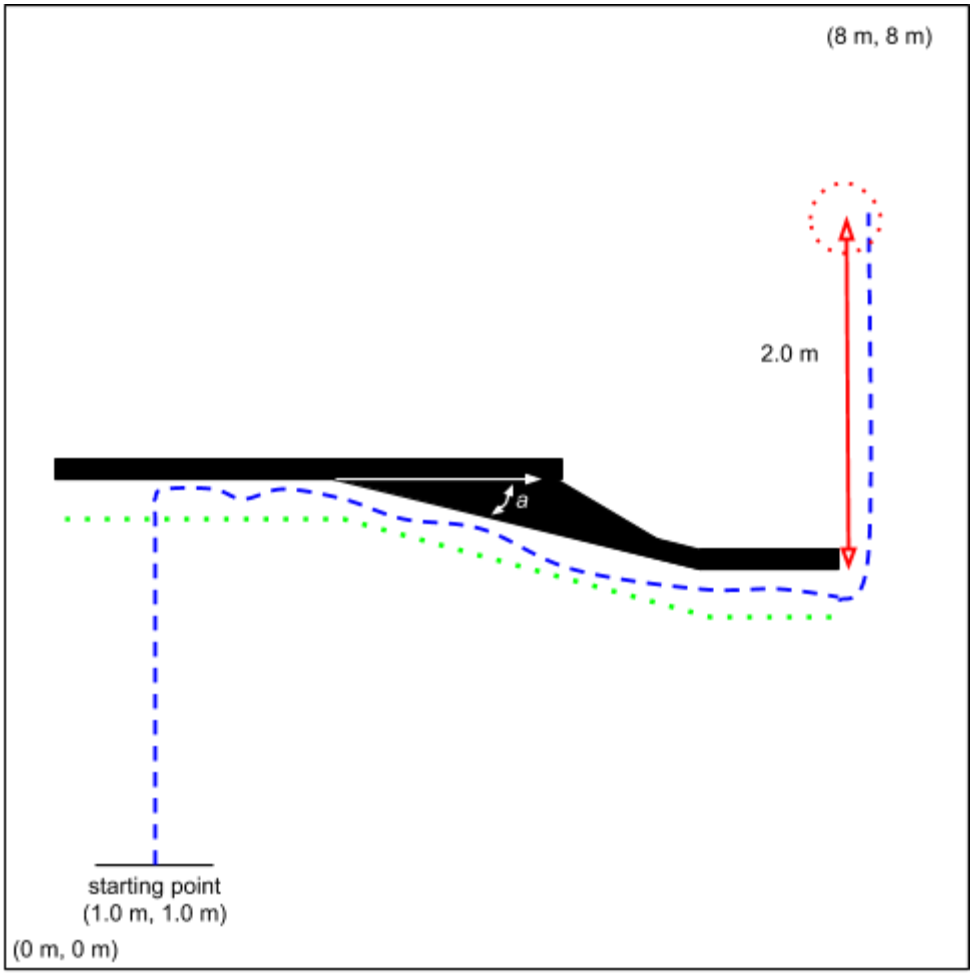
The Pioneer Robot will be placed at (1m, 1m) facing the wall (minor modification from the figure below). On running the simulation, your robot should then move straight forward (in the positive *y* direction, *according to the coordinate system defined in the figure*). Note that the coordinates in Webots are different and you should consider the one defined in the figure below for the task. Straight ahead, at a distance of somewhere between 150 and 250 cm, will be a wall that is perpendicular to the robot's forward motion. Once your robot is less than 50 centimeters from the wall (this is indicated by the wooden tiles on the floor in the environment), your robot should turn right and follow the wall. Once starting to follow the wall, your robot should never be more

than 50 cm from the wall at all times while following the wall (Some part of the Pioneer robot has to be always on the wooden tiles).

The wall may be curved and may bend up to 45 degrees. The wall will be at least 50 cm high, of unknown width, and at most 5 cm thick near its ends (although the wall may have greater thickness at other points).

The robot should follow the wall until the wall ends. When the wall ends, your robot should turn left and move in the positive *y* direction (the same direction the robot was heading at the beginning) for **2.0 m**. Your robot must then stop as accurately as possible, **2.0 m past the wall**.

Your job is to write a controller code which makes the Pioneer Robot perform the above mentioned tasks. All of the robot motion must be completed within an 8 meter square workspace without leaving the workspace. The task (from the time the simulation starts until the robot stops moving while using the “run the simulation in real-time” option) must be completed in under 90 seconds.



In the figure above, we illustrate an example of a possible environment that complies with the

rules. The wall is shown in black. The angle of any bend (e.g., angle *a* in this environment) is unknown in advance, but it is at most 45 degrees with respect to horizontal. *Note that there may be more than one bend*. The dotted green line represents 50 cm distance to the wall. The blue dashed line shows a possible robot trajectory that would receive full credit if completed in the allotted time and assuming the robot stops inside the dotted red circle. You can also see an example environment in the .wbt file. We note that the environments in our test runs may be different. The robot starts at the starting point.

Points:

Points for tasks:

Attempt: 20 points. You will receive these points simply by having a valid controller associated with the Pioneer robot and submitting your code on Sakai on time, and having your robot attempt the course. We should be able to use the controller which you provided to make the robot move forward.

Detect the wall: 15 points. You will receive these points if your robot stops or attempts to turn right when it approaches the wall.

Turn at the wall: 5 points. You will receive these points if your robot successfully turns right and starts following the wall.

Following the wall: 30 points. You will receive 20 points if your robot is never more than the allowable maximum distance from the wall. This requirement must be satisfied 60 cm after the robot is supposed to turn right until the end of the wall. Partial credit will be awarded.

Turn at end of wall: You will receive 10 points for stopping or successfully turning left at the end of the wall.

Accuracy of reaching goal: 20 points. Points will be based on the accuracy of reaching the robot's target location when it stops at the end of the task. Full points will be awarded if the robot stops inside a circle of radius 50cm around where it ideally should end. Partial credit will be given based on distance to this circle.

Penalties:

There will be a 10 point penalty if, after starting, your robot ever exits the 8m by 8m workspace (or collides with the outer brick walls).

There will be a 10 point penalty if more than 90 seconds elapse from when the simulation is started until the robot stops.

If the robot is still moving after 2 minutes, then the instructor can stop the run and no additional points can be earned.

Code

Important: You may only use the following modules from Webots:

Robot

Motor

DistanceSensor

You may *not* import, use, or, copy other modules from Webots, or any of its included classes or functions, in your program. You can still use other basic modules to handle computations like Numpy for Python or stdio.h, stdlib.h, and math.h for C. If you would like to use any other library, please contact TA Sandeep Kumar via email and we will consider adding it to the list.

Please follow the overall structure which Webots provides for the controller when you create a controller using the Wizards menu. It would also help to evaluate your code for mistakes if you add comments in the code explaining what you are trying to do.

A robot relying on a module not allowed as stated above will be disqualified. Also, any robot using Wi-Fi or Bluetooth communications in any way will be disqualified.

You must submit on Sakai a single Python or C file for your controller (which should have either a .py or .c file extension). In the comments at the top of the file, please include your name and PID.

You should feel free to discuss concepts in the course with other students in natural language (e.g. English). You should not share any programming code with others and must write all the robot's programming code yourself. If you access any sources other than the textbook or documents on Sakai, you must cite them in comments at the top of your code and send an email to the instructor with the citation. You must fully understand your code and be able to reproduce the algorithmic approach without references if asked. The Honor Code is in effect for these policies.

Good luck!