Individual Assignment: Differential Drive and LiDAR

Start Assignment

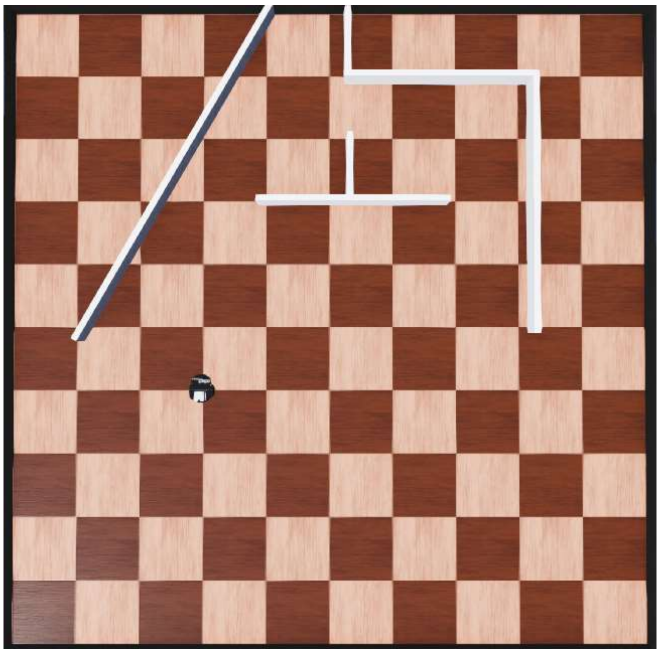
Due Friday by 11:59pm **Points** 20 **Submitting** a file upload **Attempts** 0
Allowed Attempts 1

Assignment Overview

In this assignment, you will use the Webots simulator to control a Pioneer 3-DX robot. First, you are asked to implement a differential drive controller using the formulas you learned in this module. Second, you have to use the robot's LiDAR to find the closest surface and navigate to it. Finally, you will devise and implement a wall-following algorithm. Before starting with the actual assignment, you should read and follow the instructions in the Preliminaries section in order to familiarize yourself with the simulator.

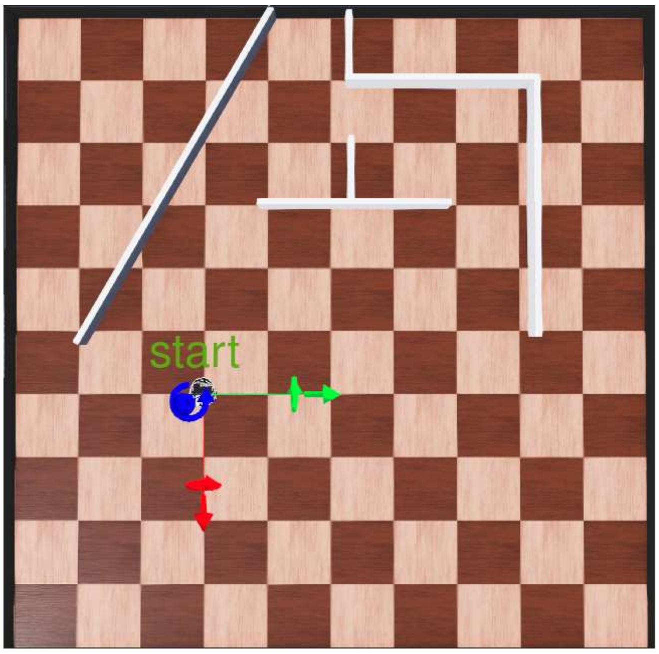
Preliminaries (1 hour)

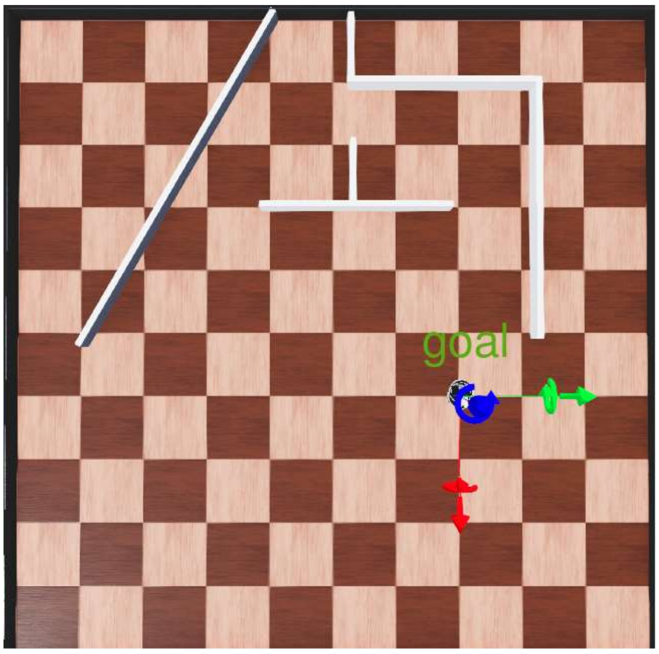
If you have not yet installed the **Webots simulator** (https://cyberbotics.com/), please do so. We recommend the latest version, 2022b, however, you can also use version 2022a if that works better on your system. Then, follow the **official tutorial** (https://cyberbotics.com/doc/guide/tutorial-1-your-first-simulation-in-webots?tab-language=python) (only Tutorial 1) to learn the basics of the simulator. Since you are supposed to use Python for implementing the assignment, please select the corresponding code option in the tutorial. Then, download **module2_assignment.zip** (https://utn.instructure.com/courses/111/files/996?wrap=1) (https://utn.instructure.com/courses/111/files/996/download?download_frd=1) (or **module2_assignment_2022a.zip** (https://utn.instructure.com/courses/111/files/997?wrap=1) (https://utn.instructure.com/courses/111/files/997/download?download_frd=1) if you are using 2022a), unpack it, and open module2_assignment/worlds/module2_assignment.wbt via File->Open World. You should see a rectangular arena with a Pioneer robot and several walls as shown below:



Differential drive (3 hours, 10 points)

Implement a differential drive controller to move the robot from the start pose ($x_s = -1m$, $y_s = 2m$, $\theta_s = \pi$) to the goal pose ($x_g = -1m$, $y_g = -2m$, $\theta_g = \pi$). In other words, the robot should be translated by 4 meters along the y-axis in world coordinates, as shown below.



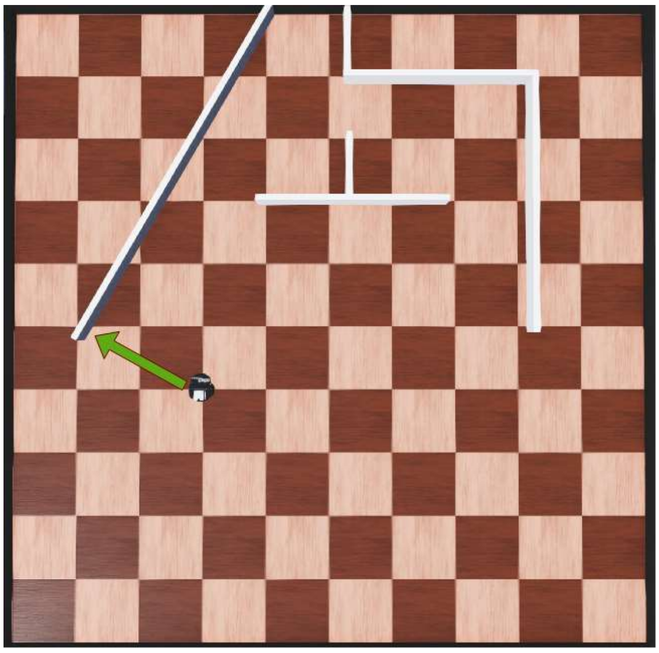


A trivial solution requires three steps to reach the goal, and one step to stop the robot once the goal is reached. Here, a step means setting or changing the pair of wheel velocities. However, it is also possible to reach the goal pose using two steps, and one step for halting. Just reaching the goal will grant you 7 points, by reaching it in two steps you will earn 10 points.

For your code, please create a new controller called "diff_drive", you can use the provided "proxy_controller" as a template. Note that the wheel speed is measured in radians per second, i.e., a velocity of 2π means that the wheel will make one turn per second. The maximal wheel velocity of the Pioneer 3-DX is 12.3, the wheel diameter is $0.195m$, and the distance between the wheels is $l = 0.325$.

Closest surface (2 hours, 7 points)

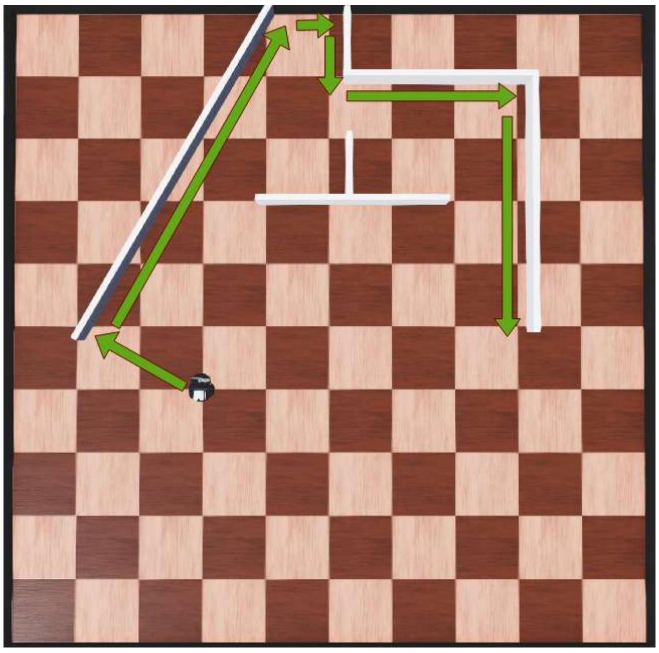
In the second part of the assignment, you should use the robot's LiDAR to find the wall point closest to the robot's start position and then move the robot to that point. For your information, the following image indicates the closest point:



However, in your code this point should be determined from LiDAR data. To that end, please create a new controller called "closest_point". As a template, you can again use the "proxy_controller" which also contains a code snippet for reading LiDAR measurements. The LiDAR has a 180° FoV, a scan contains $n = 181$ measurements, and the position of the LiDAR in the robot coordinate frame is $(x = 0.08m, y = 0m, z = 0.21m)$. The robot should halt $50cm$ in front of the wall, measured from the center of the robot. Please drive the robot directly to the goal point and do not use the two-step algorithm from the previous part.

Wall following (2 hours, 3 points)

In the last part, please implement a wall-following algorithm in a new controller called "wall_following". In the first step, the robot should drive to the closest wall, here you can use your results from the previous part. Then, it should drive along that wall as depicted below (the wall should be always left of the robot).



Guidelines

Please submit the three controller files "diff_drive", "closest_point", and "wall_following" as a single .zip file called "module2_assignment_your_name.zip", the deadline is Friday, Nov 25, 23:59 CET. The controllers should be implemented in Python.

Technical Support

For technical support please get in contact with stars@utn.de (mailto:stars@utn.de) .

For questions regarding the Webots simulator please contact Dr. Michael Krawez.

Need help using Canvas Discussions? If so, please review the following page: **[Canvas Resources for Students - Discussions](https://design.instructure.com/courses/178/pages/discussions?module_item_id=676)** (https://design.instructure.com/courses/178/pages/discussions?module_item_id=676) .

UTN

Differential Drive and LiDAR

Criteria	Ratings		Pts
Differential drive (three steps) The robot reaches the goal pose using at most three steps and halts at the goal in an additional step.	**7 to >0.0 pts** **Full Marks**	**0 pts** **No Marks**	7 pts
Differential drive (two steps) The robot reaches the goal pose using two steps and halts at the goal in an additional step.	**3 to >0.0 pts** **Full Marks**	**0 pts** **No Marks**	3 pts
Closest point (detection) The point on a wall closest to the robot's start position is determined from the LiDAR data.	**5 to >0.0 pts** **Full Marks**	**0 pts** **No Marks**	5 pts
Closest point (moving) The robot moves towards the detected point and halts 50cm before the wall (the distance is relative to the robot center).	**2 to >0.0 pts** **Full Marks**	**0 pts** **No Marks**	2 pts
Wall following The robot follows the wall as depicted in the assignment description.	**3 to >0.0 pts** **Full Marks**	**0 pts** **No Marks**	3 pts
			Total Points: 20