Daffa Fauzan A (19507334009)
Andi Prasetyo (19507334026)
Hafizal P.S (19507334028)

A. Judul

Laporan Praktik Robotika tentang *Line Follower* dan *wall following robot*

B. Hari/Tanggal Praktikum

Praktikum dilakukan pada hari Selasa, 28 September 2021 dan laporan ini ditulis pada hari Kamis, 7 Oktober 2021

C. Tujuan Praktikum

Adapun tujuan dari praktikum ini adalah:

a. Mahasiswa dapat mengakses sensor-sensor yang ada pada robot
b. Mahasiswa dapat membuat program robot yang dapat menjalankan tugas seperti perintah yang ada di GitHub, yaitu: Robot mulai dari titik Start, Robot dapat mengikuti garis (line follow), Robot dapat mengikuti dinding ketika tidak ada garis (wall follow), Robot dapat mengambil dan membawa box menggunakan lengan, dan Robot berhenti pada titik Finish
c. Mahasiswa dapat menerapkan *PID Controller* untuk menjalankan robot

D. Alat dan Bahan Praktikum

Alat dan Bahan praktikum adalah:

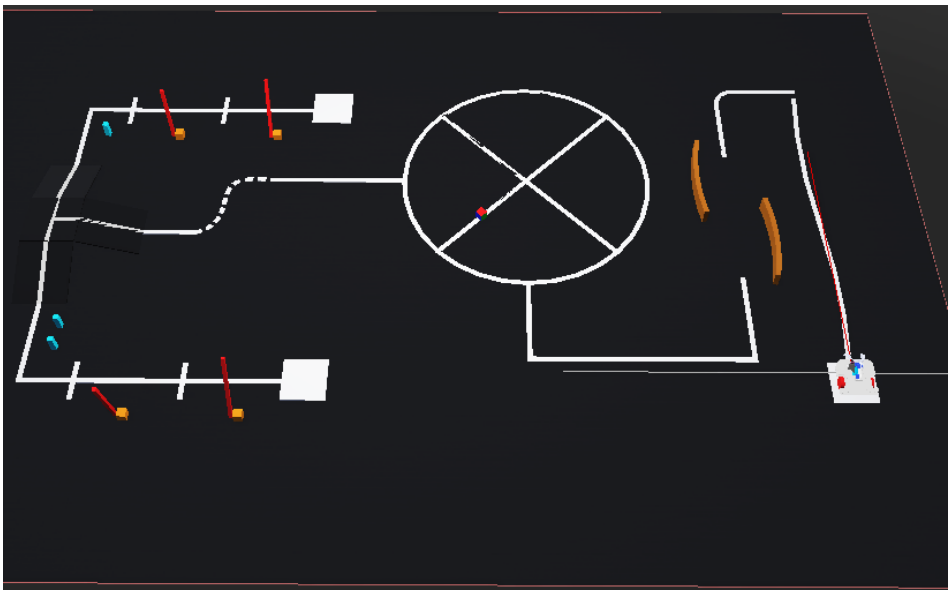
a. Laptop (OS Windows)
b. Software Webots https://cyberbotics.com/

E. Prosedur Kerja

Langkah kerja yang dilakukan adalah sebagai berikut:

a. Mendownload World dan Robot
b. Mengakses sensor-sensor yang ada pada robot
c. Membuat program *PID* untuk menjalankan *Line Follower* dan *wall following robot* yang menggunakan robot *custom*.

F. Hasil Pengamatan

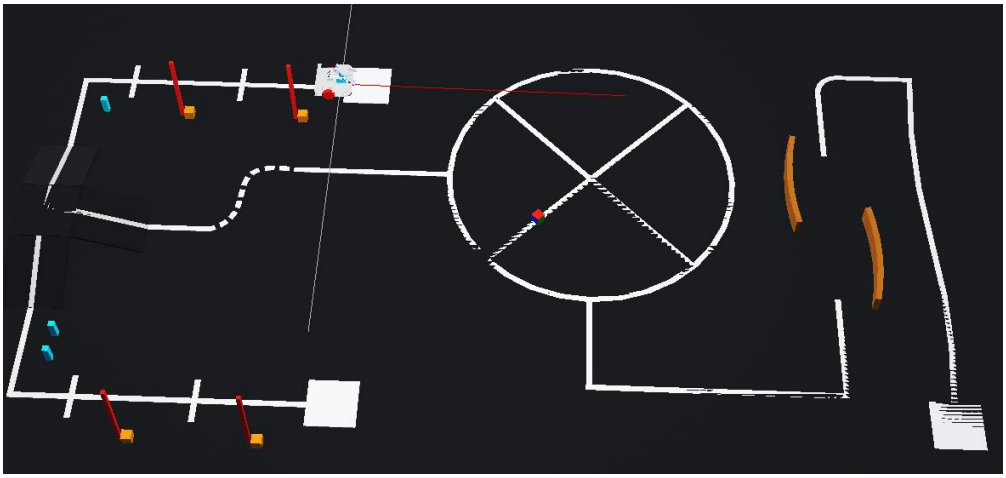


Gambar 1. *Line Follower* dan *wall following robot*

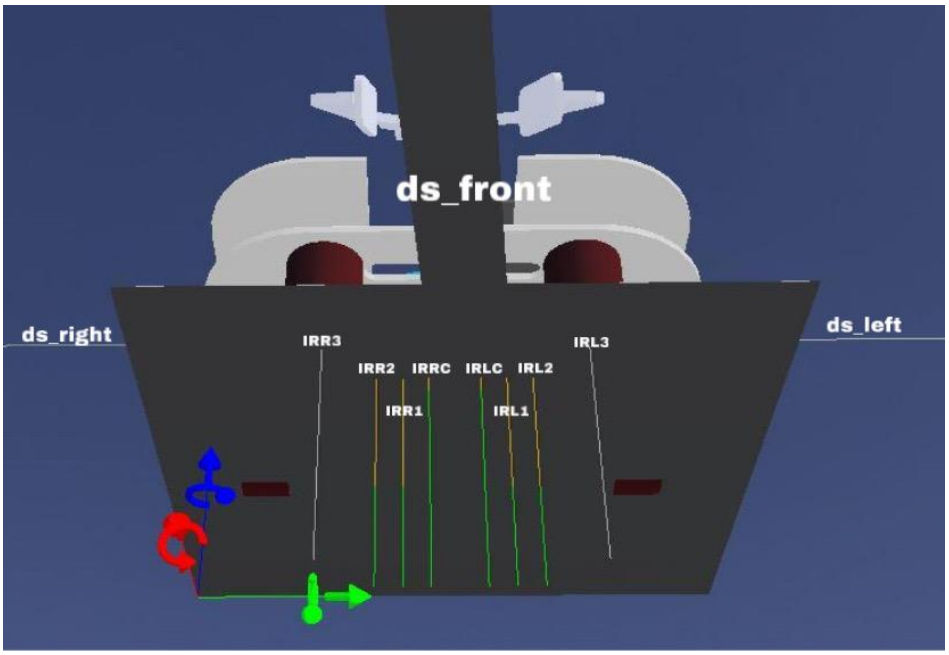
Daffa Fauzan A (19507334009)
Andi Prasetyo (19507334026)
Hafizal P.S (19507334028)

Pada pertemuan *Mobile Robots* ini, kelompok kami mencoba untuk membuat program untuk *Line Follower* dan *wall following robot* yang sebelumnya sudah dibuat dan juga sudah di download melalui github. Robot ini merupakan kombinasi antara *Line Follower* dan *wall following robot*. Jadi selain program untuk *Line Follower,* ada juga arena yang memerlukan program untuk bisa melakukan wall following.

Adapun cara kerja dari *Line Follower* dan *wall following robot* menggunakan *PID Controller* ini adalah menggunakan sensor-sensor yang ada pada robot *custom*, sensornya adalah IRL2 (kiri terluar), IRL1 (kiri), IRCL (tengah kiri), IRCR (tengah kanan), IRR1 (kanan), IRR2 (kanan terluar), lalu IRL3 (kiri pojok), IRR3 (kanan pojok) yang fungsinya untuk mendeteksi persimpangan dan ds_left (dinding kiri), ds_front (portal di depan), ds_right (dinding kanan). Sensor IR kanan dan kiri dari robot inilah yang mendeteksi adanya garis atau tidak. Untuk cara mendeteksinya disini hasil pembacaan nilai sensor IR harus dikonversi ke bilangan biner terlebih dahulu, jadi jika mendeteksi adanya garis binernya adalah 1 dan jika tidak ada garis binernya adalah 0. Sedangkan untuk yang ds sensor masih menggunakan pembacaan menggunakan hasil pembacaan dari sensor IR tanpa harus dikonversi. Disini kami menambahkan 2 sensor IR lagi yaitu sensor IRL3 dan IRR3 yang fungsinya adalah untuk mendeteksi persimpangan. Ketika simulasi dijalankan, robot sudah berhasil berjalan mengikuti garis yang ada dan ketika melewati dinding, robot juga sudah berhasil berjalan menyusuri dinding sesuai dengan program wall followingnya. Dan ketika ada persimpangan, robot pun berjalan dengan menggunakan program persimpangan yang sudah dibuat sehingga robot tidak bingung ketika ada persimpangan. Dan ketika ada portal, robot menjalankan program yang mendeteksi adanya portal dimana sensor ds_front yang mendeteksi. Disini kami menggunakan *Stage* agar robot bisa berjalan berurutan sesuai dengan algoritma yang sudah kami buat. Sehingga robot berhasil mencapai garis finish.



Gambar 2. Line Follower dan Wall Following robot sudah finish



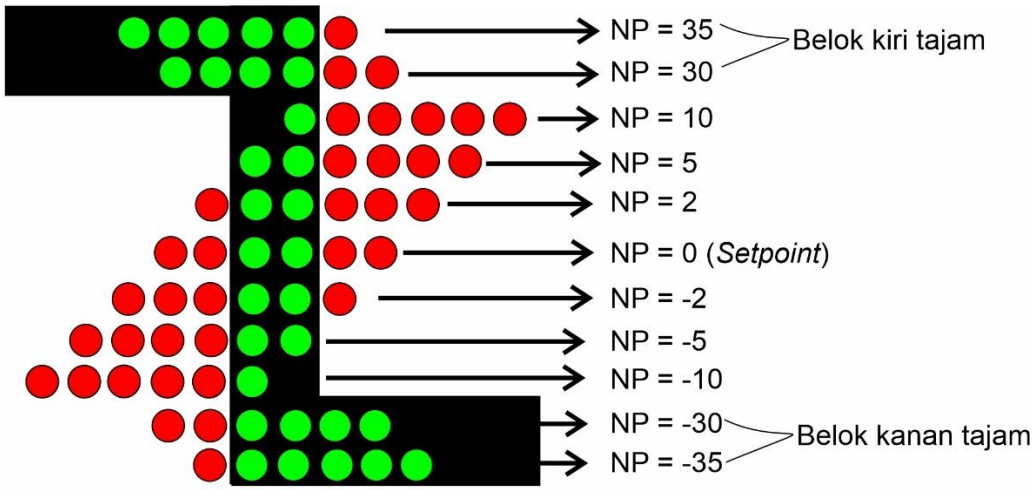
Gambar 3. Robot custom yang digunakan beserta sensornya

Gambar diatas adalah gambar robot custom beserta sensornya yang digunakan dalam praktikum kali ini, jadi letak sensor yang digunakan untuk mendeteksi adanya garis ditunjukkan pada gambar yaitu sensor IR. Sedangkan sensor yang digunakan untuk mendeteksi adanya halangan portal adalah ds_front dan sensor yang mendeteksi adanya dinding adalah ds_left, ds_right.

G. Pembahasan

1. Pembahasan konsep *Line Follower Robot*

   Pada program untuk *Line Follower Robot* disini menggunakan PID Controller, untuk *setpoint*nya di setting 0, karena saat kondisi dimana sensor IR berada pada posisi 001100 (nilai 1 disini adalah ketika sensor IRLC dan IRRC mengenai garis) itu adalah posisi dimana robot mencapai kondisi idealnya yaitu *setpoint* 0 dan juga nilai posisinya 0. Dan ketika ada error atau jika nilai posisi tidak = 0, baik itu yang bernilai positif ataupun negatif, maka robot akan dengan cepat mengubah posisinya agar bisa mencapai nilai posisi = 0.



***NP = Nilai Posisi**

Gambar 4. Nilai Posisi Robot yang didapatkan dari hasil pembacaan sensor IR

Sehingga nilai error dapat dihitung dengan menggunakan rumus sebagai berikut,

$$Error = Setpoint - Nilai\ Posisi$$

Nilai error dari pembacaan sensor diatas yang dijadikan untuk perhitungan kendali PID, lalu nilai perhitungan PID tersebut dijumlahkan dengan kecepatan untuk dijadikan sebagai nilai aktual PWM motor dari robot line follower, berikut perhitungan PWM kiri dan kanan,

$$Kiri = Kecepatan + Nilai\ Posisi$$
$$Kanan = Kecepatan - Nilai\ Posisi$$

2. Pembahasan konsep *Wall-Following Robot*

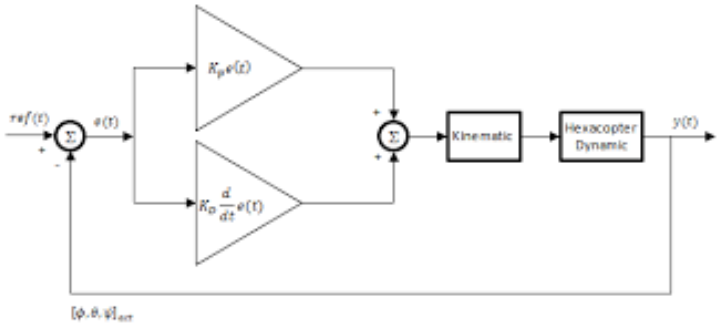
Pada program untuk *Wall-Following Robot* disini menggunakan PD Controller, telah ditentukan *setpoint* atau jarak antara sensor kanan (ds_right) dan sensor kiri (ds_left) dengan dinding yaitu 990. Jadi ketika terjadi error maka robot akan kembali membenahi posisinya agar sesuai dengan *setpoint* yang sudah ditentukan. Jika terjadi error yang besar maka pergerakan roda robot juga akan besar dengan tujuan untuk memperkecil nilai errornya. Jadi fungsi dari program *PD* adalah untuk menghitung besarnya error yang dihasilkan oleh robot dan robot pun akan membenahinya sendiri agar bisa mencapai *setpoint* untuk perhitungan *PD* pada *Wall-Following Robot* masing – masing adalah seperti berikut :

Pout = Error x Kp

Untuk perhitungan proportional diatas, Kp sudah kita tentukan yaitu sebesar 0.5.

Dout = Differensial error x Kd

Untuk perhitungan Differensial diatas, cara menghitungnya adalah diff error = error sekarang – error terakhir, error terakhir sendiri didapatkan dari error yang ada pada Pout. Dan untuk nilai Kd nya adalah 0,1.
Jadi penggabungan antara hasil output keseluruhannya adalah sebagai berikut:
P + D = Pout + Dout
Rumus PD itulah yang digunakan untuk menelusur dinding



Gambar 5. Block diagram PD

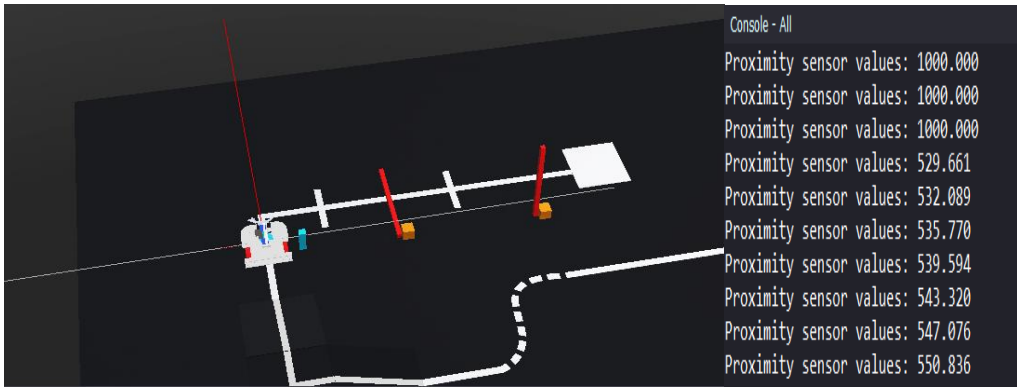
Berdasarkan gambar 4 diatas, ***Proportional controller*** hanya bergantung pada perbedaan antara set point dan variabel proses. Perbedaan ini disebut dengan Error. ***Derivative Controller*** menyebabkan output menurun jika variabel proses meningkat dengan cepat. Respon turunan sebanding dengan laju perubahan variabel proses. Meningkatkan waktu turunan *(T_d)* akan menyebabkan sistem kontrol bereaksi lebih kuat terhadap perubahan istilah kesalahan dan akan meningkatkan kecepatan respons sistem kontrol secara keseluruhan.

Jadi penggunaan *PD* Controller dalam *wall-following Robot* cukup efektif, karena kesalahan error akan dibenahi secara perhitungan dengan menggunakan *PD* yang menyebabkan pergerakan robot akan lebih halus. Ketika mendeteksi error yang sangat besar maka robot pun akan dengan cepat memperbaiki error tersebut agar bisa kembali menuju ke *setpoint* yang sudah ditentukan.

3. Pembahasan konsep Algoritma *Portal Gate Arena*

Pada program untuk *Portal Gate Arena* disini digunakan sensor depan (ds_front). Jadi Ketika robot menjalankan algoritma pada stage 3 dan kemudian kondisi sensor ds_right < 550 yang berarti robot mendeteksi ada objek disebelah kanan (objek warna biru) maka robot akan menjalankan stage 4 atau algoritma *Portal Gate Arena*. Untuk cara kerja algorimanya adalah ds_front mendeteksi adanya jarak antara portal dengan sensor < 190 maka akan membuat robot berhenti bergerak dan ketika sensor ds_front mendeteksi jarak antara portal dengan sensor > 190 maka motor akan bergerak dengan kecepatan cepat. Jika tidak dalam kondisi tersebut, maka robot akan menjalankan algoritma *line follower* seperti biasa. Lalu robot akan finish atau

berhenti di titik tujuan jika kondisi sensor IR 111111, sensor pojok (IRR3 dan IRL3) 11 dan sensor ds_front tidak mendeteksi adanya portal.



Gambar 6. Sensor ds_right mendeteksi objek berwarna biru, robot menjalankan *stage* 4

H. Kesimpulan

Kesimpulan yang dapat ditarik dari hasil pembahasan sebelumnya adalah Kita dapat membuat dan juga memprogram sebuah *Mobile Robots custom* yang merupakan kombinasi antara *wall-following robot* dan *line follower robot*. Selain itu kami juga bisa mengetahui bagaimana mengakses sensor-sensor yang ada di Robot *custom* tersebut, menambahkan sensor baru untuk robot tersebut. Selain itu kami juga bisa mengetahui bagaimana membuat *line-follower robot* yang menggunakan PID *Controller*, dimana proses yang cukup memakan waktu lama adalah ketika kami men*tunning* parameter Kp. Ki, dan juga Kd agar robot bisa mencapai posisi yang stabil, sehingga didapatkan masing-masing nilai Kp, Ki, dan juga Kd adalah 20, 0.2, 70. Sedangkan untuk *Wall-follower* robotnya kami menggunakan PD *Controller*, kami menggunakan parameter Kp = 0,5 dan Kd = 0,1. Sehingga robot kami bisa berjalan dengan halus ketika mengikuti garis dan juga menyusuri dinding. Kami juga menggunakan *stage* agar robot kami bisa berjalan mulus dan berurutan sesuai dengan lintasan yang ada. Dimana kondisi per *stage* nya dibuat berbeda sesuai dengan kondisi lintasan. Sehingga hasil dari program yang sudah kami buat, bisa membuat robot berjalan hingga garis *finish.*

I. Daftar Pustaka

a. Ir. Ardy Seto Priambodo, S.T., M.Eng. (2021).
b. *PID Theory Explained.* (2020). *Accessed on :* https://www.ni.com/en-id/innovations/white-papers/06/pid-theory-explained.html
c. *Line Follower Robot Using PID Control.* Muhammad Ilham. (2021). *Accessed on : https://www.muhilham.com/2018/01/line-follower-robot-using-pid-control.html*

J. Lampiran

a. Link Video Gdrive : https://unyku.id/Kelompok7-Prak2
b. Link GitHub : https://github.com/Hafizalps/webots-group7/tree/hafizal/competition/line-arena