PERANCANGAN ROBOT PEMINDAH BARANG *LINE FOLLOWER* DENGAN *PATH PLANNING* BERBASIS APLIKASI *MOBILE*

PROYEK AKHIR

Laporan akhir ini dibuat dan diajukan untuk memenuhi salah satu syarat kelulusan Diploma III Politeknik Manufaktur Negeri Bangka Belitung

Disusun Oleh:

Dilah Andini	NIRM:0032107
Sahrul Ramadhan	NIRM:0032125

POLITEKNIK MANUFAKTUR NEGERI BANGKA BELITUNG

TAHUN 2024

LEMBAR PENGESAHAN

LEMBAR PENGESAHAN

PERANCANGAN ROBOT PEMINDAH BARANG *LINE FOLLOWER* DENGAN *PATH PLANNING* BERBASIS APLIKASI *MOBILE*

Oleh :

Dilah Andini	/NIRM0032107
Sahrul Ramadhan	/NIRM0032125

Laporan akhir ini telah disetujui dan disahkan sebagai salah satu syarat kelulusan Program Diploma III Politeknik Manufaktur Negeri Bangka Belitung

Menyetujui,

Pembimbing 1	Pembimbing 2
(Ocsirendi, M.T)	(Zanu Saputra, S.ST., M.Tr.T.)

Penguji 1	Penguji 2
(Indra Dwisaputra, S.ST., M.T.)	(Yang Agita Rindri, S.Kom., M.Eng.)

ii

PERNYATAAN BUKAN PLAGIAT

PERNYATAAN BUKAN PLAGIAT

Yang bertanda tangan di bawah ini:
Nama Mahasiswa 1: Dilah Andini NIRM:0032107
Nama Mahasiswa 2: Sahrul Ramadhan NIRM:0032125

Dengan Judul : *PERANCANGAN ROBOT PEMINDAH BARANG LINE FOLLOWER DENGAN PATH PLANNING BERBASIS APLIKASI MOBILE*

Menyatakan bahwa laporan akhir ini adalah hasil kerja kami sendiri dan bukan merupakan plagiat. Pernyataan ini kami buat dengan sebenarnya dan bila ternyata dikemudian hari ternyata melanggar pernyataan ini, kami bersedia menerima sanksi yang berlaku.

Sungailiat, 16 Juli 2024

Nama Mahasiswa	Tanda Tangan
1. Dilah Andini	
2. Sahrul Ramadhan	

ABSTRAK

Dalam bidang industri, pendidikan, pemerintahan, maupun kehidupan sehari – hari, kebutuhan untuk memindahkan barang secara efisien sangat penting. Solusi yang tepat dalam membantu hal ini yaitu, Robot line follower. Robot line follwer dapat digunakan untuk berbagai tujuan termasuk pemindahan barang secara otomatis. Proyek akhir ini bertujuan untuk mengembangkan sebuah robot line follower yang mampu mengikuti jalur, mendeteksi objek dengan sensor ultrasonik, melakukan kontrol gerakan menggunakan algoritma PID, serta implementasi dengan path planning berbasis aplikasi mobile. Pemakaian robot line follower bertujuan untuk memindahkan barang lebih mudah dan menghemat waktu, karena robot ini di program untuk memindahkan barang dengan titik koordinat jalur tercepat. Robot ini juga dilengkapi dengan gripper untuk mengambill dan memindahkan benda. Robot yang dibuat menggunakan sensor garis dengan 12 kanal multiplekser. Pergerakan robot mengikuti garis menggunakan sistem kontrol PID dengan performa terbaik diperoleh nilai Kp = 8, Ki = 0 dan Kd = 2,5. Berdasarkan hasil percobaan secara menyeluruh, robot berhasil memindahkan benda dari object coordinates menuju destination coordinates.

Kata kunci : Robot Line Follower, Kontrol PID, Path Planning, Aplikasi Mobile, Pemindahan barang otomatis

ABSTRACT

In industry, education, government, and daily life, the need to move goods efficiently is very important. The right solution to help with this is the line-follower robot. Line-follower robots can be used for various purposes, including moving goods automatically. This final project aims to develop a line-follower robot that can follow a path, detect object s with ultrasonic sensors, perform motion control using the PID algorithm, and implement mobile application-based path planning. The use of a line-follower robot aims to move goods more easily and save time because this robot is programmed to move goods with the coordinate points of the fastest path. This robot is also equipped with a gripper to pick up and move object s. The robot uses a line sensor with 12 multiplexer channels. The movement of the robot follows the line using a PID control system, with the best performance obtained at Kp = 8, Ki = 0, and Kd = 2.5. Based on the overall experimental results, the robot successfully moves object s from object coordinates to destination coordinates.

Keywords: Line Follower Robot, PID Control, Path Planning, Mobile Application, Automated Goods Transfer.

KATA PENGANTAR

Puji syukur atas kehadiran Tuhan Yang Maha Esa atas berkat rahmat danhidayah-Nya, penulis dapat menyelesaikan makalah Proyek Akhir yang berjudul "Perancangan Robot Pemindah Barang *Line Follower* Dengan *Path Planning* berbasis Aplikasi *Mobile*" ini dengan baik. Adapun laporan Proyek Akhir ini disusun sebagaisalah satu persyaratan dan kewajiban mahasiswa untuk menyelesaikan kurikulum program Pendidikan Diploma III (DIII) di Politeknik Manufaktur Negeri Bangka Belitung.

Penulis mengakui bahwa selesainya Proyek Akhir ini tidak lepas dari bantuan banyak pihak yang telah membantu dan memberi dukungan dalam membuat alat maupun dalam menyelesaikan laporan Proyek Akhir ini. Untuk itu penulis mengucapkan terima kasih kepada:

1. Orang tua serta keluarga yang selalu memberikan motivasi, dukungan, semangat, serta doa sehingga penulis dapat menyelesaikan makalah ini sesuai dengan waktu yang telah ditentukan.
2. Bapak I Made Andik Setiawan, M.Eng., Ph.D selaku Direktur Politeknik Manufaktur Negeri BangkaBelitung.
3. Bapak Ocsirendi, S.ST, M.T. selaku pembimbing 1 yang telah memberikan bimbingan, saran, dan solusi dari masalah-masalah yang penulis hadapi dalam menyelesaikan Proyek Akhir ini.
4. Bapak Zanu Saputra, M.Tr.T selaku pembimbing 2 yang telah memberikan pengarahan dalam pengerjaan makalah dan pengerjaan ProyekAkhir ini.
5. Seluruh staff pengajar dan karyawan Politeknik Manufaktur Negeri Bangka Belitung.
6. Seluruh teman-teman seperjuangan yang telah bersedia membantu dan memberi dukungan dalam penyelesaian Proyek Akhir ini.
7. Pihak- pihak lain yang telah memberi bantuan baik secara langsung maupun tidak langsung yang tidak dapat disebutkan satu persatu.Penulis menyadari bahwa dalam Laporan Proyek Akhir ini masih jauh dari kata sempurna, untuk itu penulis sangat mengharapkan semua jenis saran, kritik, dan masukan yang bersifat membangun dalam

perbaikan laporan ini. Demikian laporan ini dibuat dan semoga laporan ini dapat bermanfaat dan menambah wawasan bagi pembaca. Akhir kata penulis ucapkan terimakasih.

Sungailiat, 16 Juli 2024

Penulis

DAFTAR ISI

LEMBAR PENGESAHAN ii
PERNYATAAN BUKAN PLAGIAT ii
ABSTRAK iv
KATA PENGANTAR vi
DAFTAR TABEL xi
DAFTAR GAMBAR xii
DAFTAR LAMPIRAN xiv
BAB I
PENDAHULUAN
1.1. Latar Belakang Masalah 15
1.2. Perumusan Masalah 16
1.3. Tujuan Proyek Akhir 17
1.4. Batasan Masalah 17
BAB II
DASAR TEORI
2.1. Robot *Line Follower* 18
2.2. ESP 32 20
2.3. PID (Proportional Intergral Derivative) 21
2.4. Sensor *Ultrasonic* 24
2.5. Photodioda 26
2.6. *Path Planning* 28
2.7. *Gripper* 29
BAB III
METODE PELAKSANAAN
3.1. Studi Literatur 32

3.2. Desain 3D *Hardware* 33
3.3. Desain Rangkaian Elektrik 36
3.4. Desain *Software* 37
3.5. Pembuatan Konstruksi Alat 39
3.6. Pembuatan Rangkaian Elektrik dan PCB 39
3.7. Pembuatan Program 41
3.8. Pengujian Keseluruhan Alat 41
1.1.1. Pengujian Hardware *41*
1.1.2. Pengujian Software *41*
3.9. Pengambilan dan Analisis Data 42
3.10. Pembuatan Laporan Proyek Akhir 43

BAB IV

PEMBAHASAN

4.1. Keterangan Alat 44
4.2. Perencanaan dan Pembuatan Robot 45
a. Instalasi Sensor Garis 45
4.3. Pengujian Elektrik Robot 47
4.3.1. Pengujian Sensor Garis *47*
4.3.2. Pengujian Sensor Ultrasonic *50*
4.3.3. Pengujian Linearitas antara PWM *dan RPM* *53*
4.3.4. Blok Diagram Sistem *56*
4.3.5. Pengujian Aplikasi *57*
4.3.6. Pengujian Kontrol PID Terhadap Sensor Garis *61*
4.3.7. Pengujian Keseluruhan *63*

BAB V

KESIMPULAN DAN SARAN

5.1. KESIMPULAN 67

5.2. SARAN 67

DAFTAR PUSTAKA 68

LAMPIRAN 1 71

LAMPIRAN 2 77

DAFTAR TABEL

Tabel 3. 1 Komponen yang Digunakan 39
Tabel 4. 1 Hasil Pengujian Sensor Garis……49
Tabel 4. 2 Hasil Pengujian Sensor Ultrasonik..... 51
Tabel 4. 3 Pengujian *Line*aritas antara PWM dan RPM 55
Tabel 4. 4 Pengujian Aplikasi 60
Tabel 4. 5 Hasil Pengujian Robot Tanpa Beban..... 63
Tabel 4. 6 Hasil Pengujian Robot Dengan Beban 64
Tabel 4. 7 Hasil Waktu Tempuh Robot Pengantar Barang 64

DAFTAR GAMBAR

Gambar 2. 1 ESP 32[12] 21
Gambar 2. 2 Blok Diagram 21
Gambar 2. 3 Respon sistem dengan Kp= 8, Ki= 0,1, dan Kd=2 23
Gambar 2. 4 Sensor *Ultrasonic*[19] 26
Gambar 2. 5 Photodioda[23] 28
Gambar 2. 6 Desain 3D *Gripper* 30
Gambar 3. 1 Diagram Alir Tahapan Pelaksanaan Proyek Akhir……...31
Gambar 3. 2 Roda (a). Roda Bagian Dalam (b). Roda Bagian Luar 33
Gambar 3. 3 Tempat Baterai 33
Gambar 3. 4 Tempat PCB 33
Gambar 3. 5 *Body* Utama 34
Gambar 3. 6 Sensor *Line Follower* 34
Gambar 3. 7 Mountin (Tempat Motor Dc) 34
Gambar 3. 8 Motor DC 35
Gambar 3. 9 *Gripper* 35
Gambar 3. 10 Desain 3D *Line Follower* Tampak Atas 35
Gambar 3. 11 Rangkaian Elektrik 36
Gambar 3. 12 *Flowchart* Desain *Software* 37
Gambar 3. 13 *Schematic* Kontrol *Line Follower* 40
Gambar 3. 14 *Schematic* Sensor *Line Follower* 40
Gambar 3. 15 *Flowchart* Pengujian Keseluruhan Alat 42
Gambar 4. 1 Gambar Keseluruhan Robot *Line Follower* ……44
Gambar 4. 2 Jalur Robot *Line Follower* 44
Gambar 4. 3 Instalasi Sensor Garis 45
Gambar 4. 4 Pemasangan Roda 46
Gambar 4. 5 Pemasangan sensor ultrasonik 46
Gambar 4. 6 Pemasangan Modul ESP 46
Gambar 4. 7 Pemasangan *LCD* 47
Gambar 4. 8 Rangkaian Elektrik Komponen 47
Gambar 4. 9 *Wiring* Sensor Garis 48

Gambar 4. 10 Hasil Pengujian Sensor Garis (a). Bit [1,2] (b). Garis Hitam..... 49
Gambar 4. 11 *Wiring* Sensor *Ultrasonic*..... 50
Gambar 4. 12 Hasil Pengujian Sensor *Ultrasonic* (a). Hasil Pengujian Menggunakan Penggaris (b). Hasil Pengujian Menggunakan Sensor *Ultrasonic* 51
Gambar 4. 13 *Wiring* Motor Driver..... 53
Gambar 4. 14 Hasil Pengujian Persamaan *Line*ar antara PWM dan RPM 55
Gambar 4. 15 Grafik Pengujian Persamaan *Line*ar antara PWM dan RPM 55
Gambar 4. 16 Blok Diagram Sistem..... 56
Gambar 4. 17 Pengujian Aplikasi..... 57
Gambar 4. 18 Simulasi Pengujian Aplikasi 60
Gambar 4. 19 Grafik Pengujian PID Kontrol terhadap Sensor Garis..... 62
Gambar 4. 20 Robot Berhasil Angkat Beban (a). Robot Berhasil Angkat Beban teringan (b). Robot Berhasil Angkat Beban Terberat 68

DAFTAR LAMPIRAN

LAMPIRAN 1 71
LAMPIRAN 2 74

BAB I
PENDAHULUAN

1.1. Latar Belakang Masalah

Dalam bidang pendidikan, industri, pemerintahan, maupun kehidupan sehari hari, tidak bisa lepas dengan pekerjaan memindahkan barang. Untuk memudahkan dalam pengerjaan membutuhkan alat bantu seperti robot [1]. Pengertian robot ialah salah satu perangkat yang dapat membantu manusia, untuk menyelesaikan berbagai jenis pekerjaan dalam berbagai bidang. Pengaruh robot sangat besar sehingga dapat menggantikan peran manusia untuk melakukan pekerjaan yang membutuhkan ketelitian tinggi serta pekerjaan dengan tingkat resiko ini dapat mengancam keselamatan pekerja [2].

Robot *Line Follower* merupakan jenis robot yang sudah diprogram untuk bergerak mengikuti jalur tertentu yang di atur pada aplikasi *mobile*. Robot *line follower* familiar digunakan untuk kompetisi dan pembelajaran. Tidak hanya digunakan untuk kompetisi, dewasa ini robot *line follower* dapat digunakan untuk dunia terutama dalam pengerjaan pemindahan barang. Dari penjelasan tersebut, robot *line follower* pemindah barang bisa meminimalisir kesalahan yang dilakukan oleh manusia atau human *error*, karena robot di program untuk melakukan pekerjaan manusia secara otomatis. Pemakaian robot *line follower* bertujuan untuk memindahkan barang lebih mudah dan menghemat waktu, karena robot ini di program untuk memindahkan barang dengan titik koordinat jalur tercepat [3]. Penggunaan robot *line follower* dapat membantu dalam transportasi material di sepanjang jalur produksi, meningkatkan efisiensi dan kecepatan produksi. Namun, untuk meningkatkan fungsionalitas dan fleksibilitas robot, diperlukan integrasi teknologi pemetaan.

Pemetaan (*path planning*) yang dimaksud adalah pengelompokan jalur - jalur untuk mendapatkan peta jalur tersingkat. Serta kemampuan untuk membuat peta lingkungan sekitar robot, memungkinkan robot untuk menentukan posisi jalur dengan akurat. Dengan adanya pemetaan, robot dapat beroperasi dengan lebih efisien dan aman, menghindari rintangan serta merencanakan rute yang optimal. Ini

sangat bermanfaat dalam lingkungan yang dinamis dan berubah-ubah [4]. Penggunaan robot *line follower* dapat membantu dalam transportasi material sepanjang jalur produksi.

Dengan menggabungkan teknologi *line follower*, pemetaan, dan *gripper* dalam satu sistem, robot dapat menjadi solusi yang lebih serbaguna dan adaptif untuk memenuhi berbagai kebutuhan industri. Robot ini dapat digunakan untuk berbagai aplikasi seperti pemindahan material, pengemasan, bahkan pemeliharaan fasilitas dengan kemampuan untuk mengenali dan merespons perubahan dalam lingkungan kerjanya. Melalui pengembangan robot *line follower* dengan kemampuan pemetaan dan *gripper* , diharapkan dapat meningkatkan efisiensi operasional, mengurangi risiko kesalahan manusia, dan memberikan kontribusi positif terhadap kemajuan industri otomasi di masa depan, dari pemapaaran di atas maka penulis tertarik untuk mengajukan proyek akhir dengan judul **"Perancangan Robot Pemindah Barang *Line Follower* dengan Path planning Berbasis Aplikasi *Mobile*."**

1.2. Perumusan Masalah

Berdasarkan latar belakang masalah diatas, maka rumusan masalah pada penelitian ini sebagai berikut:

1. Bagaimana merancang robot *line follower* yang dapat memindahkan barang dan mengikuti *path planning* dari koordinat yang dibuat.
2. Bagaimana membuat robot yang dapat memindahkan barang sesuai dengan koordinat yang telah ditentukan.

1.3. Tujuan Proyek Akhir

Tujuan dari proyek akhir ini adalah sebagai berikut:

1. Robot berjalan sesuai koordinat yang dibuat.
2. Robot dapat memudahkan operator dalam pemindahan barang melalui aplikasi secara ototmatis.
3. Bagaimana membuat robot yang dapat memindahkan barang menggunakan *gripper* dari satu titik ke titik lain.

1.4. Batasan Masalah

Batasan masalah berdasarkan dari latar belakang dalam pelaksanaan proyek akhir ini,antara lain sebagai berikut:

1. Robot hanya bisa memindahkan benda yang berada di tengah garis.
2. Pengujian dilakukan di arena dengan dimensi ± 250 cm x 250 cm.
3. Koordinat *Line Follower* tidak fleksibel harus diatur sesuai dengan lebar jalur**.**

BAB II
DASAR TEORI

2.1. Robot *Line Follower*

Dengan menggunakan daya penggerak berupa motor, robot pengikut garis dapat bergerak berdasarkan jalur garis (*Line*) yang telah ditentukan sebelumnya dan biasanya di atas arena dengan ukuran yang telah ditentukan sebelumnya [5]. Dasar pembacaan sensor dan gerakan motor DC pengikut garis (*Line Follower*) menentukan bagaimana robot pengikut garis bekerja. Bidang seperti perlombaan, penelitian, dan industri sering menggunakan robot *line follower* [6].

Robot *line follower* membuat manusia lebih mudah dalam memindahkan barang dari suatu tempat ke tempat yang lain serta dapat digunakan sebagai alat transportasi otomatis. Memiliki cara kerja mendeteksi garis sehingga membuat robot *line follower* dapat melakukan perpindahan posisi dari satu titik ke titik yang lain [7].

Adapun dasar dari *Line follower* robot adalah sebagai berikut:

1. Robot di lengkapi dengan sensor optik yang terletak pada bagian ujung depan robot. Sensor merupakan suatu piranti elektronika yang berfungsi untuk mengubah besaran-besaran fisik yang ada di alam menjadi besaran elektrik yang dapat dimengerti oleh rangkaian elektronika. Robot *line follower* menggunakan sensor intensitas cahaya yang berfung si untuk mendeteksi adanya garis hitam pada alas berwarna putih atau adanya garis putih pada lapangan dengan warna hitam.
2. Kelemahan dari *line follower* yaitu pemilihan pada garis tidak bisa diubah di *software* melainkan dibuat pada abstraksi *hardware*.

Menurut jurnal yang berjudul "Robot Pengantar Makanan berbasis *Line Follower* dengan Sensor Warna TCS3200 dan *Internet of Things (IoT)*", robot pengantar makanan menggunakan teknologi *line follower* yang dapat dikendalikan melalui tombol pada sebuah website untuk memilih meja tujuan. Robot ini dilengkapi dengan sensor inframerah untuk mengidentifikasi jalur serta sensor warna TCS3200 untuk mendeteksi kode warna meja tujuan. Hasil pengujian menunjukkan bahwa robot memiliki waktu respons rata-rata sekitar 1,157 detik

untuk menjalankan perintah dari tombol dan mampu mengangkut beban maksimum 3kg. Kecepatan robot dipengaruhi oleh berat beban yang diangkut; semakin berat beban, semakin lambat kecepatan robot. Pembuatan robot ini memiliki potensi untuk meningkatkan kualitas pelayanan dan efisiensi di restoran [8].

Berdasarkan jurnal dengan judul "Robot *Line Follower* Pengantar Surat Menggunakan Metode *Fuzzy Logic* Studi Kasus Fakultas Teknik Universitas Pancasila". Robot *Line Follower* dengan menggunakan metode Logika *Fuzzy*. Logika *Fuzzy* yang diterapkan pada Robot ini sebagai penentu kecepatan Motor Servo serta penentu jarak dari denah lokasi pengantaran surat. Penelitian ini menghasilkan tingkat akurasi sebesar 100% untuk garis lurus, Garis belok huruf U 79%, garis belok patah 53%, deteksi obstacle 94,75%, alarmtone 87,75%, dan pengenalan fungsi *RFID card* 92,75%, sedangkan untuk akurasi *RFID* saat penambahan fitur menu memiliki akurasi sebesar 82% [9].

Berdasarkan penelitian yang dimuat dalam jurnal yang berjudul "Optimasi Sistem Navigasi Pada Robot Pengantar Makanan" Robot *line follower* yang berjalan mengikuti sensor garis terdapat 14 kanal multiplekser dan menggunakan sistem kendali PID dengan Kp = 1,0, Ki = 0,015, dan Kd = 1,0, masing-masing. Untuk mengukur jarak tempuh dan arah pergerakan sistem perjalanan menerapkan metode navigasi berbasis rotary encoder kemudian dipantau melalui aplikasi processing. Mengikuti garis yang dibuat, robot dapat mengantarkan makanan ke meja yang dinginkan.

Berdasarkan penilitian diatas memiliki kesamaan konsep terhadap proyek akhir ini bahwa, robot *line follower* merupakan robot yang dirancang untuk mengikuti garis dan prinsip dasarnya menggunakan sensor intensitas cahaya untuk mendeteksi adanya garis. Penelitian diatas memiliki konsep kesamaan dalam menggunakan sistem kendali atau kontrol PID pada proyek akhir ini dengan menguji Seberapa signifikan pengaruh nilai PID terhadap deteksi sensor garis, menggunakan *setpoint* 0 dan melakukan pengujian *trial error* untuk mendapatkan nilai Kp, Ki, dan Kd yang tepat.

2.2. ESP 32

Mikrokontroler ESP 32 merupakan seri lain dari ESP8266 yang dikenalkan oleh *Espressif System*. Mikrokontroler ini sangat mendukung untuk membuat sistem aplikasi *Internet of Things* karena sudah tersedia modul WiFi 802.11 b/g/n, Bluetooth versi 4.2 dan berbagai peripheral dalam chip. Tidak banyak memiliki perbedaan dengan ESP8266 yang familiar di pasaran, hanya saja ESP32 lebih komplek dibandingkan ESP8266, dan lebih cocok untuk proyek yang besar [10]. Pada ESP32 terdapat processor, penyimpanan dan akses pada GPIO *(General Purpose Input Output)*. ESP 32 ini bisa digunakan untuk rangkaian pengganti pada *Arduino*. Adapun spesifikasi dari ESP32 adalah sebagai berikut:

ESP32 memiliki *board* dua versi, yaitu 30 GPIO dan 36 GPIO. Untuk memudahkan identifikasi, setiap pin diberi label di bagian tas *board*. Keduanya berfungsi hampir sama yang membedakan yaitu pada 30 GPIO dipilih karena memiliki dua pin *GND*. Memiliki *interface* USB to UART yang mudah di program dengan program pengembangan aplikasi seperti *Arduino IDE*. Melalui konektor micro USB bisa memberikan sumber daya ke *board* [11].

Fungsi utama ESP 32 pada robot *line follower* meliputi pemrosesan sinyal sensor, kontrol motor, komunikasi pengolahan data, dan antarmuka pengguna. ESP 32 menerima dan memproses data dari sensor garis photodioda pada proyek akhir ini kemudian mendapatkan data yang digunakan untuk mengendalikan motor agar robot tetap mengikuti jalur dengan presisi. ESP 32 pada proyek akhir ini memanfaatkan komunikasi nirkabel seperti WiFi memungkinkan kontrol jarak jauh dan pengiriman data ke server untuk analisi lebih lanjut. Kemampuan pengolahan data ESP32 termasuk algoritma sistem kendali PID yang membantu dalam menjaga stabilitas dan akurasi pergerakan robot. ESP32 dapat dihubungkan dengan layar atau indikator *LED* memberikan feedback visual kepada pengguna dan dapat menerima input melalui tombol atau sensor lain.

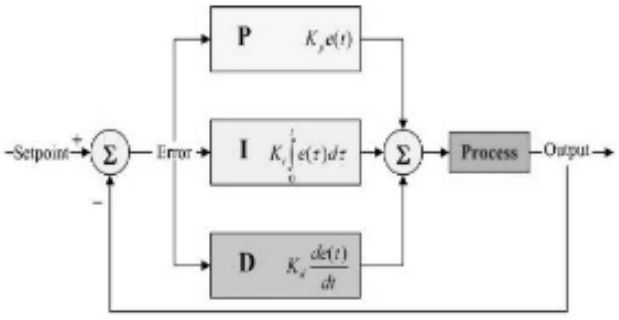
Dari kesimpulan di atas penulis menggunakan ESP 32 karena memiliki beberapa fungsi yang penting dalam pengerjaan proyek akhir ini seperti, modul WiFi, processor, penyimpanan dan pengaksesan pada GPIO yang mendukung

untuk mengembangkan aplikasi sistem Internet of Things pada robot *line follower*.

Gambar 2. 1 ESP 32 [12].

2.3. PID (Proportional Intergral Derivative)

PID merupakan singkatan dari Proportional Integral Derivative yang merupakan gabungan dari tiga sistem kendali yaitu terdiri dari bagian singkatannya sendiri *controller proportional , controller integral, dan controller derivative*. Untuk menghasilkan keluaran dengan waktu naik dan turun yang rendah. Untuk keunggulan sistem kendali integral dalam meredam kesalahan, sedangkan sistem kendali derivative memiliki keunggulan dalam meredam *overshoot* dan *undershoot*. Ada dua jenis PID, yaitu analog dan digital. PID analog digunakan dengan rangkaian elektronika seperti resistor, kapasitor, dan amplifier operasional. PID digital digunakan dengan program [13].



Gambar 2. 2 Blok Diagram

Kontribusi signifikan dari masing-masing parameter P, I, dan D mempengaruhi karakteristik kontroler PID. Penyetelan konstanta Kp, Ti, dan Td akan mengakibatkan penonjolan sifat dari masing-masing elemen. Ada kemungkinan bahwa salah satu dari ketiga konstanta tersebut akan menjadi lebih penting daripada yang lain. Konstanta yang menonjol itulah akan mempengaruhi respon sistem secara keseluruhan. Untuk aplikasi di lapangan, kontroler PID dapat diidentifikasi dengan P : *gain* yang menunjukan peningkatan atau pencapaian, I : *reset* menunjukan pengaturan, dan D : *rate* menunjukan kecepatan/laju [14]. Untuk mendapatkan nilai Kp, Ki, dan Kd yang tepat perlunya melakukan tuning parameter. Sehingga dapat mengendalikan posisi robot agar stabil berjalan di jalur yang berwarna hitam (garis hitam) [15]. Berdasarkan jurnal yang berjudul "Robot Forklift *Line Follower* dengan Kendali PID dan Sensor Warna" (2021) menunjukan bahwa pengujian tanpa menggunakan sistem kendali PID mengakibatkan sistem tidak memiliki kestabilan yang tetap, hal tersebut ditunjukkan dengan tidak adanya nilai konstan pada *setpoint* yang sudah ditentukan. Sebaliknya ketika pengujian menggunakan sistem kendali berupa PID dengan 5 kali percobaan menunjukan adanya kestabilan dengan tingkat kestabilan yang berbeda – beda[16].

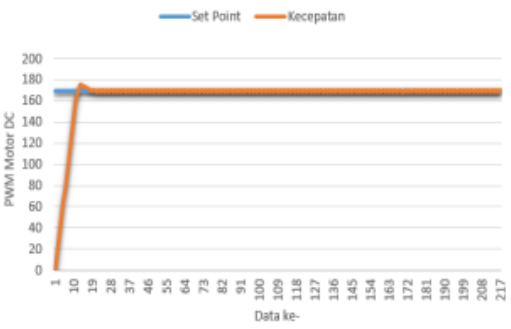
Berdasarkan jurnal penelitian yang berjudul "Robot Mobil *Line Follower* Dengan Kendali PID Sebagai Pengembangan Brt Trans Semarang" terdapat 5 kali pengujian terhadap PID dengan sistem yang mengeluarkan respon yang berbeda – beda.

1. Dengan parameter Kp = 1, Ki = 1, dan Kd = 1, sistem menunjukkan *overshoot* sebesar 11,7% dari jangkauan nilai PWM dalam kondisi stabil adanya osilasi hingga 9% dari nilai keadaan stabil. Mulai dari data ke-50 sistem menujukkan keadaan cukup stabil.
2. Dengan parameter Kp = 3, Ki = 1, dan Kd = 1, sistem menunjukkan *overshoot* sebesar 18,8% dari jangkauan nilai PWM dalam kondisi stabil adanya osilasi hingga 12% dari nilai keadaan stabil. Mulai dari data ke-86 sistem menujukkan keadaan cukup stabil.
3. Sistem dikonfigurasi dengan parameter Kp = 3, Ki = 0,1, dan Kd = 2, dan mencapai kondisi stabil saat data ke-49. Namun respon sistem tergolong

lama untuk mencapai kondisi stbail karena masih memiliki rise time yang tinggi.

4. Sistem dibuat dengan parameter Kp = 10, Ki = 0,1, dan Kd = 2, dan menunjukkan *overshoot* sebesar 5,88% . Mulai dari data ke-25 sistem menujukkan keadaan cukup stabil.

5. Dengan Kp = 8, Ki = 0,1, dan Kd = 2, sistem menunjukkan *overshoot* sebesar 2,35% dari jangkauan nilai PWM untuk kondisi stabil. Mulai dari data ke-23 sistem menujukkan keadaan cukup stabil [17].



Gambar 2. 3 Respon sistem dengan Kp= 8, Ki= 0,1, dan Kd=2

Sumber : "Robot Mobil *Line Follower* Dengan Kendali Pid Sebagai Pengembangan Brt Trans Semarang"

Dari 5 kali percobaan di atas dapat disimpulkan bahwa respon sistem dengan Kp=8, Ki=0,1, dan Kd=2 memiliki nilai *overshoot* yang paling rendah. Untuk kontrol kecepatan motor dc yaitu Kp = 8, Ki = 0,1, dan Kd = 2. Nilai parameter yang didapatkan ini menggunakan metode trial and *error* dan berlaku untuk sistem yang digunakan.

Dari penjelasan di atas dapat disimpulkan bahwa, PID adalah kombinasi dari tiga sistem kontrol yaitu, proporsional, integral, dan derivatif. PID di bagi menjadi versi analog dan digital, dengan analog menggunakan resistor, konsendator, dan amplifier. Sedangkan digital menggunakan program. Proyek akhir ini menggunakan sistem kendali PID karena sangat berpengaruh pada tingkat kestabilan sehingga membuat jalannya robot mulus dan tidak keluar jalur (garis hitam) dengan melakukan percobaan dan memperhatikan nilai *overshoot* sehingga

di dapatnya nilai parameter yang sesuai untuk proyek akhir robot *line follower* ini.

2.4. Sensor *Ultrasonic*

Sensor merupakan tranduser berfungsi unruk mengolah variasi gerak, panas, cahaya atau sinar, magnetis, dan kimia menjadi arus listrik dan tegangan. Sensor adalah bagian penting dari banyak peralatan. Berfungsi untuk mendeteksi dan mengetahui magnitude. Transduser sama dengan mengubah. Kemampuan untuk menghasilkan energi baru. Dengan menggunakan sensor itu sendiri, daripada merubah energi untuk meningkatkan kinerja perangkat. Proses pendeteksian untuk mengukur sensor sering digunakan. Sensor yang umum digunakan dalam berbagai rangkaian elektronik termasuk sensor sinar atau cahaya, sensor suhu, sensor asap, dan sensor tekanan. Sensor mampu menstimulus untuk menghasilkan 17 sinyal elektrik meliputi temperatur, tekanan, gaya, medan magnet cahaya, pergerakan dan sebagainya. Sementara dapat berupa konsentrasi dari bahan kimia baik cairan maupuan asap [18].

Sensor *ultrasonic* merupakan sensor produksi *parallax* 40 KHz banyak digunakan untuk aplikasi atau kontes robot cerdas berfungsi mendeteksi jarak suatu objek dengan cara memancarkan gelombang *ultrasonic* (40 KHz) selama t = 200 us kemudian mendeteksi pantulannya. Sesuai dengan kontrol dari mikrokontroller pengendali (pulsa *trigger* dengan tout min 2 us) sensor ini memancarkan gelombang *ultrasonic*. Spesifikasi dari sensor ini sendiri yaitu :

1. Burst indicator *LED* dapat menampilkan aktifitas sensor.
2. *Echo hold* off 75uS dari *fall of trigger pulse*.
3. Kisaran pengukuran berukuran 3cm – 300cm.
4. *Delay before next measurement* 200uS.
5. Input *trigger*-positive TTL *pulse*, 2uS min, 5uS tipikal.

Dengan diketahui spesifikasi pada sensor, kemudian adanya prinsip kerja dari sensor ini yaitu, memiliki chip pada pembangkit sinyal 40KHz, sebuah speaker dan mikropon *ultrasonic*. Sinyal *ultrasonic* dapat mengubah sinyal 40KHz menjadi suara sementara. Sedangkan mikropon *ultrasonic* berfungsi untuk mendeteksi pantulan suaranya. Sensor *ultrasonic* mendeteksi jarak objek dengan cara

memancarkan gelombang *ultrasonic* atau 40KHz tadi selama tBRUSH (200 us) kemudian mendeteksi pantulannya [19].

Berdasarkan jurnal yang berjudul "Implem entasi Metode *Simple Maze Wall Follower* Dengan Menggunakan *Free Rtos Pada Robot Maze*" Pengujian terhadap sensor *ultrasonic* dilakukan pengukuran sebanyak 7 kali. pengukuran dilakukan mulai dari 4cm – 16cm, kemudian menambah nilai 2 tiap akan melakukan pengujian selanjutnya. Untuk perbandingan pengukuran menggunakan penggaris cm (*centimeter*), sedangkan persentase *error* dengan menggunakan persamaan di bawah ini[18].

$$\%ERROR = \frac{(Pengukuran\ Manual) - (Pengukuran\ Sensor)}{(Pengukuran\ Manual)} \times 100\% \quad (1)$$

Berdasarkan hasil pengujian dan analisis masing-masing sensor memiliki rata-rata *error* 0%, maka kinerja dari sistem di bagian output sistem bisa dikatakan bagus dan akurat. dari penjelasan di atas dapat disimpulkan bahwa sensor *ultrasonic* dapat mendeteksi jarak suatu objek dengan cara memancarkan gelombang *ultrasonic*.

Berdasarkan jurnal yang berjudul "Robot Mobil *Line Follower* Dengan Kendali Pid Sebagai Pengembangan Brt Trans Semarang" Dilakukan pengujian terhadap sensor ultrasonik dan ditemukan nilai *error* yang diperoleh dari selisih antara nilai *setpoint* dan pengukuran langsung, kemudian dibagi dengan nilai *setpoint* dan dikalikan dengan 100%. Dilakukan 6 kali percobaan dengan *setpoint* 5 – 30. Dari nilai pembacaan jarak 5 cm, terdapat nilai *error* 3,4%. Pada pembacaan jarak 10cm terdapat nilai *error* 4,5%, pembacaan jarak 15cm terdapat *error* 2,67%, pembacaan jarak 20cm terdapat *error* 4,55%, pembacaan jarak 25cm terdapat *error* 1,6%, dan pada pembacaan 30cm terdapat *error* 0,93%. Dari hasil pembacaan ini didapatkan nilai *error* berada dibawah angka 5% dan ini dapat dikategorikan masih dalam batas toleransi pada kendaraan [20].

Berdasarkan penelitian-penelitian sebelumnya, proyek akhir ini menggunakan sensor ultrasonik pada robot *line follower*, karena *ultrasonic receiver* dapat menangkap hasil pantulan gelombang *ultrasonic* yang dapat mendeteksi suatu objek di sekitarnya sehingga *gripper* dapat mengambil objek tersebut untuk

dipindahkan. Dengan hasil pengujian dari jurnal yang di kaji, sensor ultranosik memiliki rata-rata *error* 0% , meskipun pada jurnal yang kedua didapatkan nilai *error* tetapi presentase *error* masih dibawah 5% yang dikategorikan masih dalam batas toleransi pada kendaraan. Menandakan bahwa kinerja sensor yang bagus untuk digunakan pada robot *line follower*.

Gambar 2. 4 Sensor *Ultrasonic* [21].

2.5. Photodioda

Berbeda dengan diode biasa, photodioda adalah komponen elektronik berbahan semikonduktor yang berfungsi mengubah cahaya menjadi arus listrik. Photodioda ini dapat mendeteksi berbagai jenis cahaya, termasuk inframerah, cahaya tampak, ultra ungu, dan sinar-X. Jika photodioda terkena cahaya, prisnip kerjanya menghasilkan tegangan dan memiliki nilai resistansi yang rendah. Sebaliknya, jika photodioda tidak terkena cahaya, nilai resistansinya akan meningkat atau dapat bervariasi tergantung pada seberapa kecil radiasi yang dipancarkan oleh sumber cahaya yang dianggap tak hingga. Besarnya arus listrik atau tegangan yang dihasilkan photodioda [22].

Berdasarkan penelitian jurnal yang berjudul ”Robot *Line Follower* Pemisah Benda Menggunakan Sensor Warna Tcs 3200” photodioda digunakan pada robot untuk membedakan warna gelap dan terang pada lintasan dan sebagai penerang jalur atau garis, sehingga robot akan berjalan sesuai jalur atau garis yang dibuat[23].

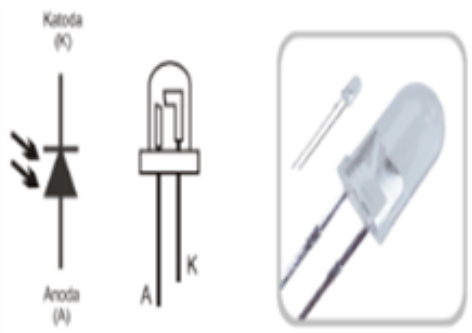
Berdasarkan penelitian jurnal yang berjudul ”Perancangan Robot *Auto Line Follower* yang Menerapkan Metode Osilasi Ziegler-Nichols Untuk Tuning Parameter PID pada Kontes Robot Indonesia” dapat diketahui bahwa hasil dari penggunaan multiplekser memiliki rata-rata kesalahan yang terjadi 0,11%. Dengan hasil perhitungan menggunakan persamaan berikut.

$$Kesalahan = \frac{Selisih\ antara\ Vout\ dan\ Vm\ multiplekser}{Vin\ multiplekser}\ x\ 100\% \quad (2)$$

Hal ini menunjukkan bahwa IC multiplekser 4067 dapat bekerja dengan baik dan bisa diaplikasikan pada robot *line follower*[22]. Berdasarkan penelitian jurnal yang berjudul "Implementasi Metode *Simple Maze Wall Follower* Dengan Menggunakan *Free RTOS* Pada Robot *Maze*" Untuk mengetahui tingkat keakuratan dari modul sensor phodioda dalam mendeteksi warna yang berwarna gelap dan terang terdapat pada sebuah tabel 2 dilakukan pengujian sebanyak 6 kali dengan warna yang berbeda-beda, warna merah dengan nilai ouput pembacaan sensor 150, putih dengan nilai ouput pembacaan sensor 145, hitam dengan nilai ouput pembacaan sensor 1023, coklat dengan nilai ouput pembacaan sensor 152, abu – abu dengan nilai ouput pembacaan sensor 148, dan hijau dengan nilai ouput pembacaan sensor 151. Pembacaan sensor dibagi warna terang dan warna gelap. Dengan *error* 0 %. Dari pengujian ini photodioda digunakan dalam sistem memiliki tingkat keakuratan yang sangat bagus [24].

Dari kesimpulan di atas, adanya kesamaan konsep yaitu penulis ingin robot berjalan sesuai dengan jalur yang telah dibuat, sehingga photodioda dapat membantu dalam mengatasi masalah tersebut pada robot penulis. Pada proyek akhir Penulis membutuhkan 12 sensor dengan multiplekser karena merupakan jumlah yang tidak terlalu banyak dan juga sangat cocok untuk mendeteksi persimpangan pada jalur penulis. Dengan pengujian yang dilakukan pada jurnal dengan judul "Implementasi Metode *Simple Maze Wall Follower* Dengan Menggunakan *Free RTOS* Pada Robot *Maze*" photodioda memiliki tingkat *error* sebesar 0% yang berarti memiliki tingkat keakuratan yang bagus digunakan dalam proyek ini.

Gambar photodioda dapat dilihat pada gambar dibawah ini.



Gambar 2.5 Photodioda [25]

2.6. *Path Planning*

Perencanaan lintasan, atau yang dikenal sebagai *path planning*, adalah algoritma yang paling sering digunakan dalam robot *line follower*. Algoritma ini berfungsi untuk menentukan jalur yang akan diikuti oleh robot saat bergerak menuju tujuannya. Algoritma perencanaan jalur ini dapat digunakan oleh pengguna untuk meminimalkan jalur yang digunakan dengan memilih jalur mana robot akan bergerak. Pada awal robot bergerak dan di persimpangan pertama, robot akan bergerak sesuai dengan program yang telah dibuat oleh pengguna, seperti berbelok ke kanan atau kiri [26]. Jalur terpendek atau optimal antara dua titik mampu ditemukan oleh algoritma *path planning* ini. Lintasan optimal belum tentu yang terpendek, jumlah belok dan berhenti untuk robot melakukan pergerakan merupakan hal yang perlu dipertimbangkan [27].

Berdasarkan jurnal penelitian yang berjudul "Sistem Kontrol PID pada Purwarupa Robot Pembawa Barang berbasis *Line Follower* menggunakan Algoritma *Path Planning*" telah dilakukan 3 kali pengujian yang dipaparkan pada tabel 4,5, dan 6. Pengujian ini dilakukan dengan melihat langsung pada robot dalam menghadapi jalur yang ditempuh sehingga mencapai tujuannya dengan *path planning* yang telah ditetapkan pada robot. Tujuan pengujian ini berguna untuk melihat kesamaaan *path planning* yang dirancang dengan *path planning* yang digunakan pada robot dan menghitung persentase keberhasilan dengan

menggunakan rumus persamaan dibawah ini.

$$Keberhasilan = Keberhasilan\ \ Total\ Percobaan \text{ X } 100\% \quad (3)$$

Pada tabel 4 persimpangan 1 – 5 dengan arah *path planning* dan arah robot yang sama seperti , Belok Kanan, Berhenti, Balik Kanan, Belok Kiri, Berhenti. Menghasilkan Kesesuaian antara arah pada *path planning* tabel 4 yang dirancang dengan *path planning* yang digunakan atau diimplementasikan di robot. Pada tabel 5 persimpangan 1 – 7 dengan arah *path planning* dan arah robot yang sama seperti , Belok Kiri, Berhenti, Balik Kanan, Belok Kanan, Lurus, Berhenti. Menghasilkan kesesuaian antara arah pada *path planning* tabel 5 yang dirancang dengan *path planning* yang digunakan atau diimplementasikan di robot. Pada tabel 6 persimpangan 1 – 9 dengan arah *path planning* dan arah robot yang sama seperti , Lurus, Lurus, Belok Kanan, Berhenti, Balik Kanan, Belok Kiri, Lurus, Lurus, Berhenti. Menghasilkan Kesesuaian antara arah pada *path planning* tabel 6 yang dirancang dengan *path planning* yang digunakan atau diimplementasikan di robot. Hasil dari akurasi untuk 3 pengujian dengan menggunakan *path planning* pertama hingga ketiga adalah 100% [28].

Dari penelitian di atas *path planning* diperlukan pada penelitian ini dengan hasil tingkat keakurasian yang tinggi pada hasil pengujian. Dan memiliki kesamaan konsep berfungsi untuk menentukan jalur mana yang akan digunakan robot saat bergerak untuk mencapai tujuannya. Pada penelitian ini digunakan jalur 250 cm x 250 cm yang dapat memindahkan barang sesuai dengan jalur tercepat yang telah di tentukan program. Algoritma *path planning* yang digunakan pada proyek akhir ini untuk menggerakkan sebuah robot berjalan menuju benda (objek) dan kemudian menuju tujuan dengan menggunakan metode navigasi berdasarkan jarak relatif.

2.7. *Gripper*

Gripper berbentuk seperti lengan robot yang merupakan interkoneksi alat pengendali. *Gripper* juga menyerupai benda kerja sebagai penggenggam (biasanya berbentuk jari *gripper*). *Gripper* memiliki beberapa fungsi antara lain:

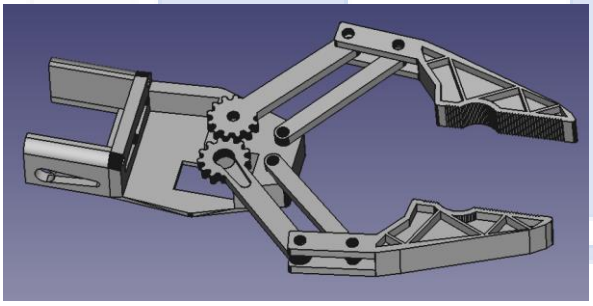
1. Mampu menahan beban statis dan dinamis.
2. Dapat mengatur posisi dan fungsi benda kerja yang jelas.

3. *Gripper* dapat berkerja seperti lengan/jari.

Penggunaan *Gripper* sudah popular saat ini sebagai robot industri dengan unit komponen yang mudah untuk diotomatisasikan. Dalam perakitan khusus *gripper* membuat *gripper* mampu beroperasi sehingga memiliki fungsi seperti tangan manusia (tangan palsu).

Berdasarkan penelitian yang dilakukan oleh Muhammad Basri dan Ira Wahira (2022) *Gripper* yang telah terpasang pada robot akan bergerak untuk mencapit objek dan memindahkan ke tempat yang telah ditentukan berdasarkan warna yang terdeteksi [29].

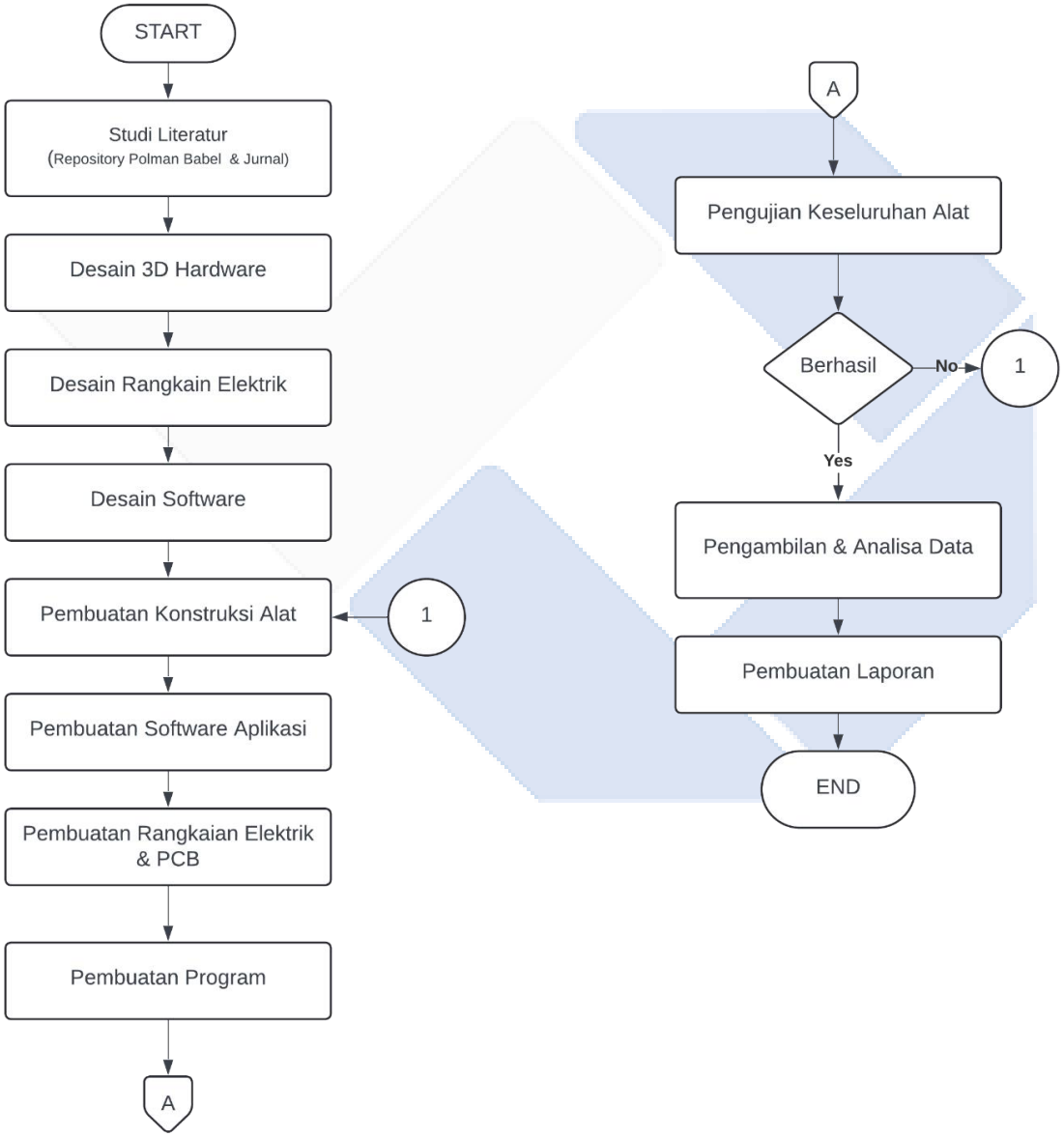
Dari penjelasan di atas penulis menggunakan *Gripper* pada robot *line follower* sesuai dengan judul proyek akhir penulis untuk memindahkan barang. Sehingga *Gripper* merupakan komponen yang tepat dengan bentuknya seperti pencapit yang dapat digunakan sebagai pencapit barang untuk memindahkan barang [30]. Pada proyek akhir ini menggunakan *Gripper* 1 *axis* (1 sumbu) yang hanya bisa mencapit dan mengangkat benda ke atas. *Axis* di sini merujuk pada kemampuan gerakan atau derajat kebebasan yang dimiliki oleh *gripper*.



Gambar 2.6 Desain 3D *Gripper*

BAB III
METODE PELAKSANAAN

Untuk menyelesaikan proyek akhir berjudul Perancangan Robot Pemindah Barang *Line Follower* dengan *Path Planning* Berbasis Aplikasi *Mobile*, dibuat beberapa langkah dan tahapan proses. Untuk mempermudah proses pembuatan proyek akhir, metode pelaksanaan proyek ini digambarkan dalam flowchart berikut.



Gambar 3.1 Diagram Alir Tahapan Pelaksanaan Proyek Akhir

3.1. Studi Literatur

Studi literatur mencakup pencarian atau mengidentifikasi masalah yang penulis coba kaji dalam proyek akhir ini. Beberapa acuan teori yang penulis ambil adalah berupa *repository* Polman Babel dan Jurnal yang relevan dengan judul proyek akhir yang akan dibuat dapat membantu mempercepat penyelesaian tugas akhir.

a. Pembahasan pada robot *line follower* studi literatur mengacu pada jurnal Robot Pengantar Makanan berbasis *Line Follower* dengan Sensor Warna TCS3200 dan *Internet of Things (IoT)* dan Robot *Line Follower* Pengantar Surat Menggunakan Metode *Fuzzy Logic* Studi Kasus Fakultas Teknik Universitas Pancasila.

b. Pembahasan pada sistem kendali PID studi literatur mengacu pada jurnal Robot Forklift *Line Follower* dengan Kendali PID, Sensor Warna (2021) dan Robot Mobil *Line Follower* Dengan Kendali Pid Sebagai Pengembangan Brt Trans Semarang, dan Optimasi Sistem Navigasi Pada Robot Pengantar Makanan.

c. Pembahasan sensor *ultrasonic* studi literatur mengacu pada jurnal Perancangan dan Implementasi Robot *Line Follower* Menggunakan *Avoid Obstacle* dengan Metode Wall Following dan Robot Mobil *Line Follower* Dengan Kendali Pid Sebagai Pengembangan Brt Trans Semarang.

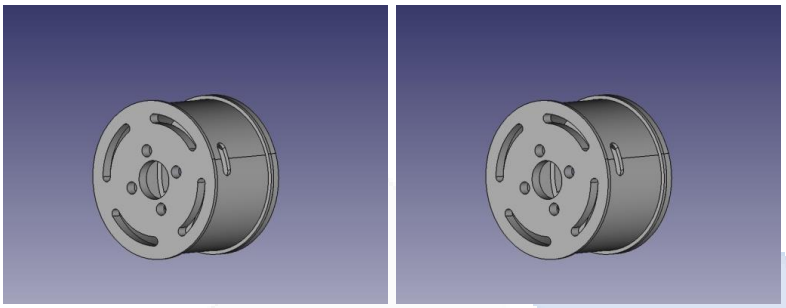
d. Pembahasan pada sensor photodioda studi literatur mengacu pada jurnal Implementasi Metode *Simple Maze Wall Follower* Dengan Menggunakan *Free RTOS* Pada Robot *Maze*.

e. Pembahasan pada *path planning* studi literatur mengacu pada jurnal Sistem Kontrol PID pada Purwarupa Robot Pembawa Barang berbasis *Line Follower* menggunakan Algoritma *Path Planning* dan Implementasi Sistem *Path Planning* dan Routing untuk *Mobile* Robot Berbasis *Vehicle Light Communication.*

f. Pembahasan pada *gripper* studi literatur mengacu pada jurnal Robot *Line Follower* Pemindah Barang Berdasarkan Warna Berbasis Mikrokontroler.

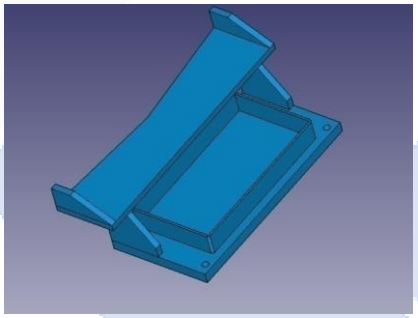
3.2. Desain 3D *Hardware*

Desain 3D pada proyek akhir ini berbentuk badan robot *Line Follower* itu sendiri menggunakan aplikasi *FreeCAD*, badan *Line Follower* yang akan dicetak berbahan dasar *filament PLA(Polylactic Acid)* ukurannya adalah 35,3 cm x 14,5 cm x 7,2 cm (P x L x T). Alat ini akan diuji di arena berbentuk kotak dengan dimensi ± 250 cm x 250 cm. Desain alat dapat dilihat pada gambar dibawah ini.



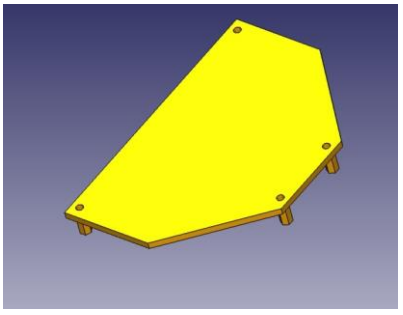
Gambar 3. 2 Roda (a). Roda Bagian Dalam (b). Roda Bagian Luar

Pada gambar 3.2.a merupakan gambar roda pada robot yang di print menggunakan *3D printing.*



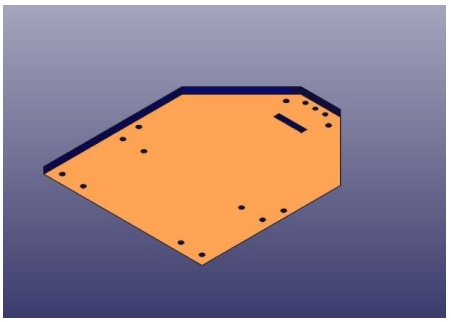
Gambar 3. 3 Tempat Baterai

Gambar diatas merupakan gambar tempat baterai yang di print menggunakan *3D printing.*



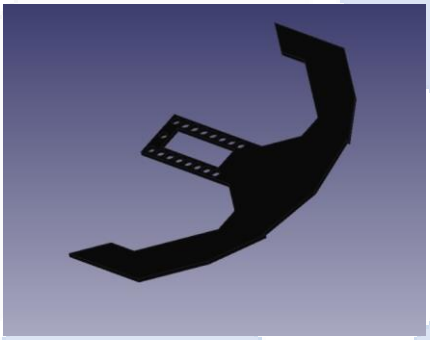
Gambar 3. 4 Tempat PCB

Pada gambar 3.4 berfungsi sebagai tempat meletakkan PCB yang di print menggunakan *3D printing.*



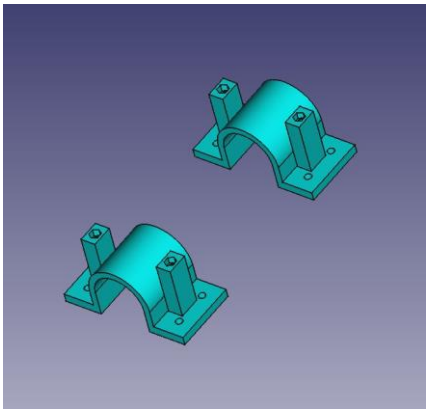
Gambar 3.5 *Body* Utama

Pada gambar 3.5 merupakan gambar *body* utama robot yang di print menggunakan *3D printing.*



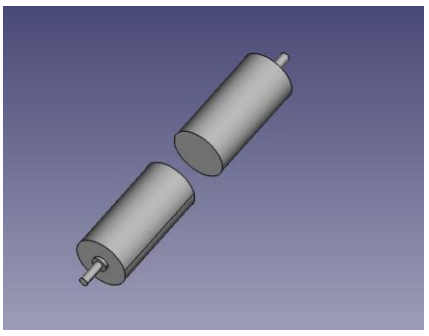
Gambar 3. 6 Sensor *Line Follower*

Merupakan gambar sensor garis pada robot *line follower* yang terdiri dari 12 *LED* dan sensor photodiode yang diolah oleh *multiplexer* yang akan mengirimkan sinyal dari sensor ke mikrokontroller.



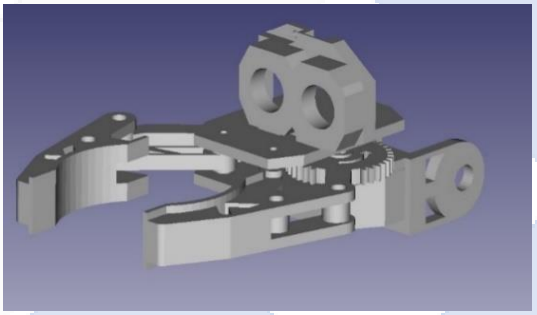
Gambar 3. 7 Mountin (Tempat Motor Dc)

Pada gambar 3.7 diatas merupakan mounting atau tempat motor DC pada robot *line follower* yang di print menggunakan *3D printing.*



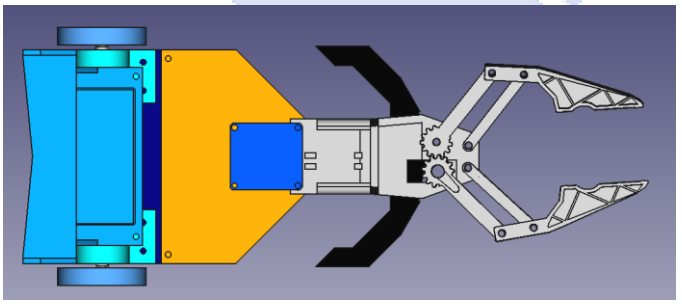
Gambar 3. 8 Motor DC

Gambar tersebut merupakan gambar motor DC yang di print menggunakan *3D printing.*



Gambar 3. 9 *Gripper*

Pada gambar 3.9 merupakan gambar *Gripper* robot yang di print menggunakan *3D printing.*

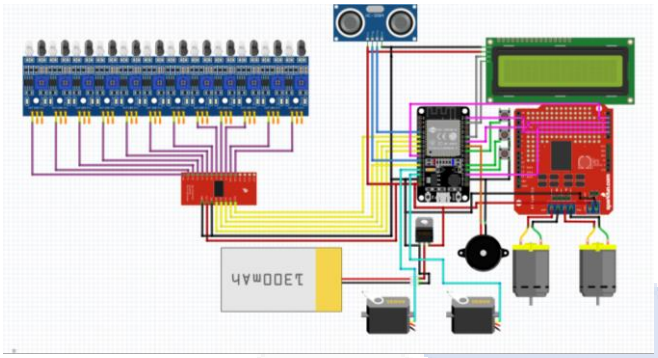


Gambar 3.10 Desain 3D *Line Follower* Tampak Atas.

Pada gambar di atas dapat dilihat desain 3D *Line Follower* tampak atas.

3.3. Desain Rangkaian Elektrik

Berikut desain rangkaian elektrik robot *line follower* pemindah barang berbasis aplikasi *mobile* sebagai berikut:

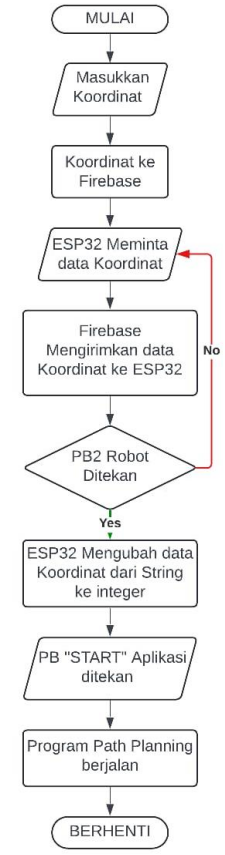


Gambar 3.11 Rangkaian Elektrik

Pada desain rangkaian elektrik diatas terdapat 12 sensor photodioda dengan multiplekser yang digunakan untuk mendeteksi cahaya dan mengukur intensitasnya yang nantinya akan mendeteksi garis pada robot *line follower,* dibagian tengah atas terdapat sensor *ultrasonic* yang berfungsi untuk mengukur jarak dengan mengirimkan gelombang ultrasonik dan mengukur waktu yang dibutuhkan untuk pantulan kembali, sensor ini juga mendeteksi adanya objek, disebelah kanan sensor *ultrasonic* terdapat LCD yang digunakan untuk menampilkan informasi yang dibaca oleh mikrokontroller, kemudian dibagian bawah tengah terdapat baterai yang menyediakan daya atau sebagai sumber untuk seluruh rangkaian elektronik. Motor *driver* terletak dibagian kanan bawah yang digunakan untuk mengontrol dua motor DC yang terhubung berputar maju atau mundur. Motor servo terletak dibagian bawah, digunakan untuk menggerakkan gripper sehingga dapat menggenggam atau melepas objek dengan baik. Buzzer terletak disebelah servo yang digunakan untuk menghasilkan suara atau alarm sebagai output dari sistem, pada proyek akhir ini untuk mendeteksi ketika robot dipersimpangan. Pusat dari seluruh sistem adalah mikrokontroller, seperti ESP32 yang terletak dibagian tengah, berfungsi sebagai otak dari sistem, memproses data yang diterima dari sensor dan memberikan perintah.

3.4. Desain *Software*

Berikut desain *software* robot *line follower* pemindah barang berbasis aplikasi *mobile* sebagai berikut :



Gambar 3.12 *Flowchart* Desain *Software*

Flowchart ini menggambarkan alur kerja dari sistem robot yang menggunakan koordinat yang diinput oleh pengguna dan disimpan di *Firebase* untuk menentukan tujuan dan menggerakkan robot ke tujuan tersebut dengan bantuan modul ESP32. Proses dimulai dengan input koordinat oleh pengguna dan berakhir dengan eksekusi algoritma *path planning* oleh robot.

Pada desain aplikasi terdapat tiga komponen utama seperti *Current Robot Coordinates, Robot Destinantion Coordinates, Object Destinantion Coordinates.* Dan lima komponen tambahan. Dibawah ini merupakan fungsi dari komponen – komponen tersebut:

1. ***Current Robot Coordinates*** : berfungsi untuk menunjukkan posisi awal dari robot dengan koordinat X dan Y.
2. ***Robot Destination Coordinates*** : berfungsi untuk menentukan tujuan robot dengan koordinat X dan Y.
3. ***Object Destination Coordinates*** : berfungsi unuk menentukan tujuan objek yang akan dipindahkan oleh robot dengan koordinat X dan Y.
4. ***Spinner*****: Koordinat Robot Saat Ini(*****Current Robot Coordinates*****): *****Spinner*** **X dan Y**

   Spinner X: Mengirimkan koordinat X saat ini dari robot.

   Spinner Y: Mengirimkan koordinat Y saat ini dari robot.
5. **Koordinat Tujuan Robot(*****Robot Destination Coordinates*****): *****Spinner*** **X dan Y**

   Spinner X: Mengatur dan mengirimkan koordinat X tujuan robot (tempat robot harus pergi untuk mengambil objek).

   Spinner Y: Mengatur dan mengirimkan koordinat Y tujuan robot (tempat robot harus pergi untuk mengambil objek).
6. **Koordinat Tujuan Objek (*****Object Destination Coordinates*****): *****Spinner*** **X dan Y**

   Spinner X: Mengatur dan mengirimkan koordinat X tujuan tempat objek akan diantar.

   Spinner Y: Mengatur dan mengirimkan koordinat Y tujuan tempat objek akan diantar.
7. ***Push Button*****:**

- **Tombol *RESET***

  Mengubah nilai "*start*" menjadi 0.

  Mengakhiri program perencanaan jalur.
- **Tombol *START***

  Memulai program perencanaan jalur.

  Mengubah nilai "*start*" menjadi 1.

  Memulai proses pergerakan robot sesuai dengan koordinat yang ditentukan.

3.5. Pembuatan Konstruksi Alat

Tahapan selanjutnya adalah pembuatan keseluruhan dibuat menggunakan referensi desain kerangka robot yang sudah dirancang pada jurnal referensi. pada jurnal-jurnal referensi. Pada proyek ini juga menambahkan komponen *gripper* yang digunakan untuk memindahkan barang dari suatu tempat/koordinat ketempat tujuan.

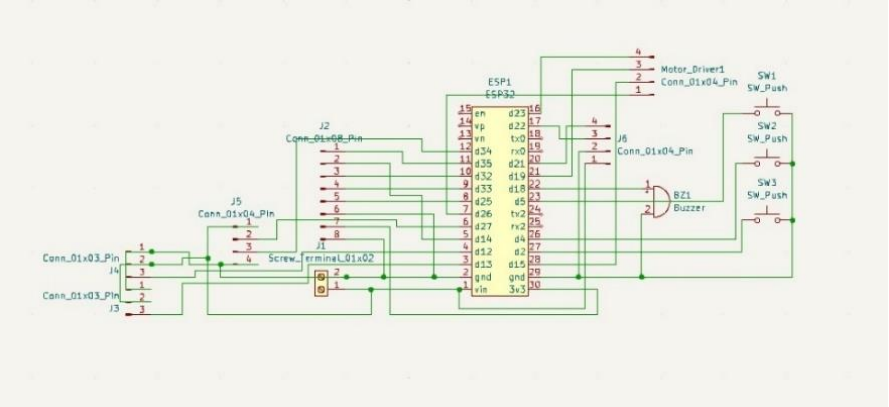
Tabel 3.1 Komponen yang Digunakan

Jenis Komponen	Jumlah
ESP 32 DOIT DevKIT V2	1
Push Button	3
LCD 16 x 2	1
Motor Driver LM298N	1
Baterai Lippo	2
Pin Header Male & Female	6
Sensor *Ultrasonic*	1
Sensor Photodioda	12
LED SuperBright(Putih)	12
Motor DC	2
Motor Servo	2
Multiplexer 12	1
Buzzer	1
DC *Stepdown* 5v	1

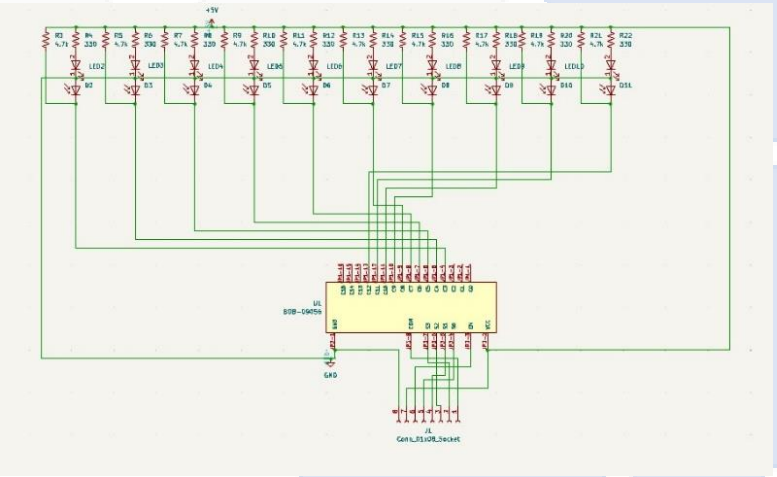
3.6. Pembuatan Rangkaian Elektrik dan PCB

Tahap ini adalah lanjutan dari tahap konstruksi alat. Pada tahap ini rangkaian dan skematik alat dibuat sketsanya menggunakan aplikasi KiCAD dan untuk mencetak PCB penulis menngunakan jasa cetak PCB *online*, PCB yang dicetak ada dua, yaitu PCB sensor dan PCB control yang berisi komponen utama *line follower*. Pada PCB kontrol *line follower* proyek akhir ini menggunakan jasa cetak PCB *online* sedangkan, untuk

PCB dari Sensor *Line Follower* (Photodioda) dicetak dengan cara manual, yaitu dengan cara sablon dan *Etching.*



Gambar 3.13 *Schematic* Kontrol *Line Follower*



Gambar 3.14 *Schematic* Sensor *Line Follower*

Pada gambar diatas, skematik kontrol *line follower* menggunakan mikrokontroler ESP32. Pin EN, VIN, 3V3, dan *GND* terhubung ke sumber daya pada ESP 32. Komponen eksternal seperti sensor *line follower*, tombol, motor *driver*, servo, sensor *ultrasonic*, *LCD* 16x2, dan buzzer dihubungkan oleh Pin GPIO. Tombol SW1, SW2, dan SW3 masing – masing terhubung ke pin GPIO 5,4, dan 2. Motor *driver* terhubung ke pin GPIO 23, 19, 15, dan 26 ESP32 untuk mengontrol motor. *Buzzer* terhubung ke pin GPIO 18 ESP32 untuk menghasilkan bunyi indikator. Servo untuk lift dan grip terhubung ke pin GPIO 13 dan 12 ESP32, sedangkan pin *VCC* dan *GND* servo terhubung ke sumber daya. Sensor

line follower memiliki pin *GND*, *VCC*, dan EN yang terhubung ke sumber daya, serta pin s0, s1, s2, s3, dan sig untuk sinyal yang terhubung ke GPIO 25,33,32,14 ESP32 untuk mengirimkan *OUTPUT* ke selektor multiplexer, sedangkan pin sig terhubung ke GPIO 35 ESP32 untuk menerima inputan sensor garis. Sensor ultrasonik memiliki pin *VCC* dan *GND* terhubung ke sumber daya, sedangkan pin *Trig* dan *Echo* terhubung ke GPIO 27 dan 34 ESP 32. *LCD* I2C16x2 memiliki pin *VCC* dan *GND* terhubung ke sumber daya, sedangkan pin SCL dan SDA terhubung ke GPIO 21 dan 22 ESP 32. Terminal sekrup J1 digunakan untuk menerima input power 5V dan menyalurkannya ke sumber daya ESP32.

3.7. Pembuatan Program

Tahap berikutnya adalah pembuatan program untuk komponen yang digunakan. Program akan digabungkan setelah dibuat agar robot dapat beroperasi sesuai keinginan. Dengan menggunakan algoritma *path planning* dan sistem kontrol PID, maka robot akan mengikuti garis dan mendeteksi serta memindahkan barang sesuai dengan intruksi tujuan atau koordinat yang telah diberikan. Pembuatan program pada proyek akhir ini menggunakan *software Arduino IDE.*

3.8. Pengujian Keseluruhan Alat

Pada tahap ini, dilakukan pengujian alat untuk memastikan bahwa sistem perencanaan jalur otonom dan pelacak jalur robotik yang dibuat sudah berjalan sesuai dengan tujuan. Uji *hardware* dan *software* dilakukan. *Flowchart* sistem keseluruhan alat dapat dilihat pada Gambar 3.15

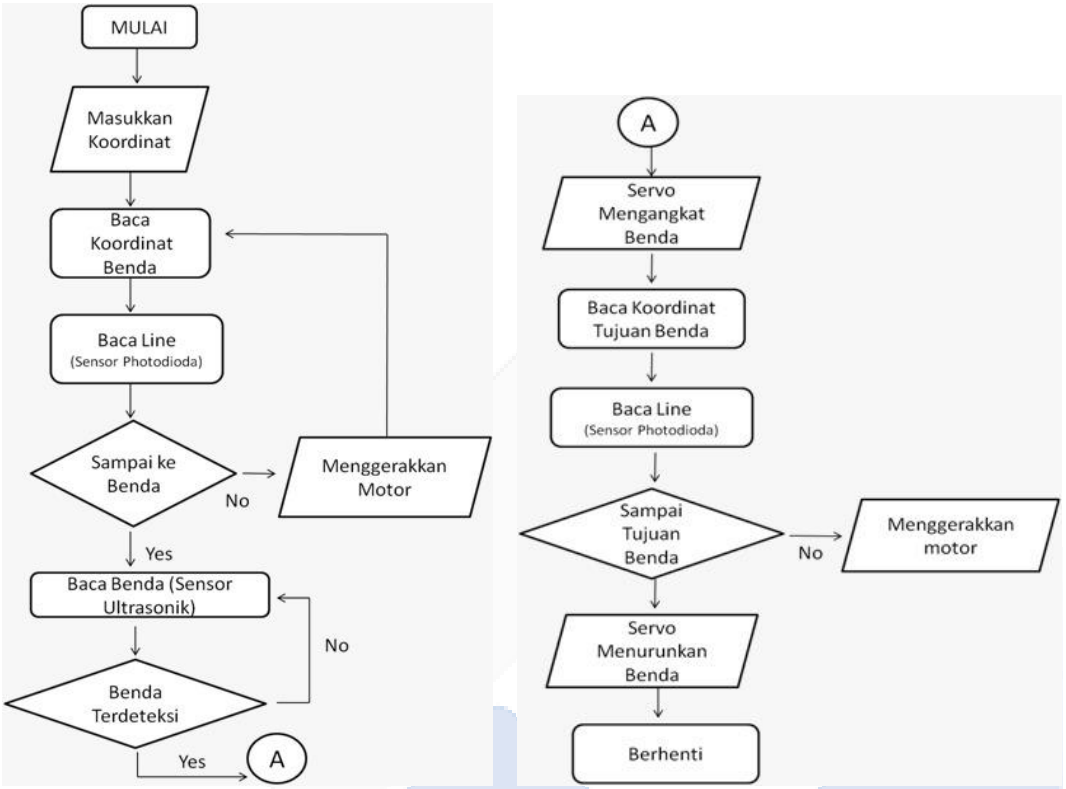
1.1.1. Pengujian *Hardware*

1. Menguji ESP32 dengan sensor photodiode dan sensor *ultrasonic* apakah memberikan *input* yang sesuai.
2. Menguji motor dc apakah berjalan sesuai dengan program dan instruksi.

1.1.2. Pengujian *Software*

1. Menguji aplikasi *mobile* apakah bisa mengirimkan koordinat dari smartphone dan diterima ESP32.

2. Menguji sistem *path planning* apakah bisa mengirimkan instruksi agar robot *line follower* bisa menuju dan memindahkan barang dengan baik.



Gambar 3.15 *Flowchart* Pengujian Keseluruhan Alat

3.9. Pengambilan dan Analisis Data

Pada tahap ini, akan mengumpulkan dan menganalisis data dan informasi yang diperoleh dari pengujian alat. Tahap ini bertujuan untuk mengidentifikasi kelebihan dan kekurangan alat yang dibuat dari segi desain, sistem pengendalian, dan penggunaan.

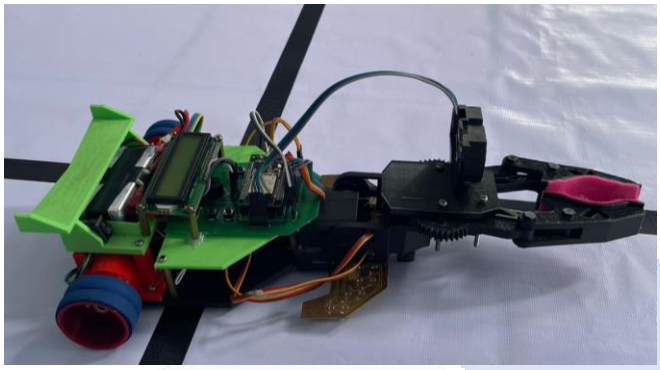
a. Mengumpulkan data PID dan menganalisis parameter yang sesuai untuk robot line folllower.
b. Mengumpulkan data sensor photodioda untuk robot *line follower*.
c. Mengumpulkan data sensor *ultrasonic* untuk robot *line follower*.

3.10. Pembuatan Laporan Proyek Akhir

Tujuan pembuatan laporan pada tahap akhir proyek akhir ini adalah untuk menyimpulkan proses alat yang dibuat secara keseluruhan. Tahap ini akan menyampaikan berbagai informasi dan analisa yang diperoleh penulis dari proses pembuatan alat yang telah dilaksanakan.

BAB IV
PEMBAHASAN

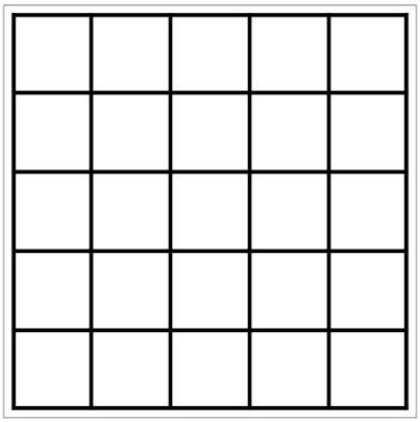
Bab IV akan membahas data dari hasil pengujian proyek akhir berjudul Perancangan Robot Pemindah Barang *Line Follower* Dengan *Path Planning* Berbasis Aplikasi *Mobile*.



Gambar 4. 1 Gambar Keseluruhan Robot *Line Follower*

4.1. Keterangan Alat

Perancangan Robot Pemindah Barang *Line Follower* dengan Path planning berbasis Aplikasi *Mobile* dirancang untuk memindahkan barang yang dapat disetting dengan aplikasi *mobile* kemudian berjalan mengikuti garis *path planning*. Dengan menggunakan sistem kontrol PID membuat robot berjalan dengan stabil. Melalui aplikasi pada *mobile* robot dapat di setting dengan memasukkan koordinat. Kemudian memasukkan koordinat tujuan robot mengambil dan meletakkan barang. *Body* robot dibuat menggunakan *3D Printing* yang diprint dan dipasang sesuai bentuk robot.



Gambar 4. 2 Jalur Robot *Line Follower*

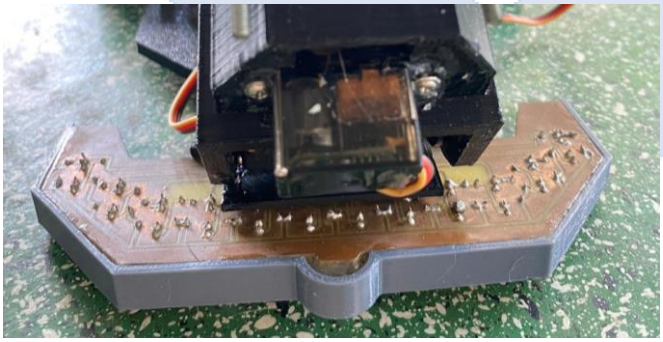
Robot pemindah barang ini bekerja dengan menekan saklar daya *ON*. Kemudian melakukan kalibrasi pada robot yang keterangan menu kalibrasi pada *LCD* akan keluar. Kalibrasi dilakukan secara benar dengan memperhatikan garis pada robot. Ketika berada pada garis hitam maka akan keluar data 1 pada monitor *LCD*, jika berada pada garis putih maka akan kelur angka 0 pada monitor *LCD*. Setelah kalibrasi selesai dilakukan dengan benar. Kemudian kita dapat menjalankan robot. Sebelum itu lakukan setting pada aplikasi melalui koordinat X dan Y pada aplikasi. Setelah diatur maka robot akan berjalan sesuai dengan tujuan dan menggerakkan barang dan meletakkan ke tempat dengan koordinat yang telah di tentukan.

4.2. Perencanaan dan Pembuatan Robot

Pada tahap perencanaan dan pembuatan robot penggerak barang ini, dilakukan proses konstruksi tubuh robot dan instalasi komponen serta perangkat yang digunakan. Proses ini terbagi dalam beberapa tahapan, meliputi :

a. Instalasi Sensor Garis

Sensor garis (*LED*) dipasang di bawah bagian depan robot untuk mendeteksi garis pada arena, yang nantinya akan mengontrol gerakan motor DC robot berdasarkan output dari sensor tersebut.



Gambar 4. 3 Instalasi Sensor Garis

b.Instalasi Roda

Instalasi roda pada *body* robot dengan jumlah 4 buah roda yang terdiri dari 2 roda di pasang pada bagian depan robot, 2 buah di pasang pada bagian belakang dan 2 buah roda pada bagian depan robot. Roda – roda tersebut terbuat

dari 3D printing yang dilapisi karet. Gambar dibawah ini menunjukkan gambar instalasi roda pada robot.

Gambar 4. 4 Pemasangan Roda

c. Pemasangan Sensor *Ultrasonic*

Pemasangan sensor *ultrasonic* di bagian depan dan atas robot bertujuan untuk mengirimkan sinyal ke ESP32. Pemasangan sensor *ultrasonic* di bagian tas mendeteksi barang dan meminta *gripper* mengambilnya dan memindahkan.

Gambar 4. 5 Pemasangan sensor ultrasonik

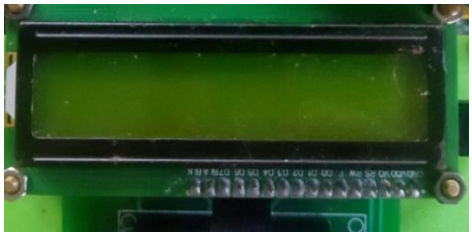
d. Pemasangan ESP 32

Pemasangan ESP 32 pada bagian atas *body* robot di depan *LCD*. ESP 32 berfungsi sebagai pemrosesan sinyal sensor, kontrol motor, komunikasi pengolahan data, dan antarmuka pengguna.

Gambar 4. 6 Pemasangan Modul ESP

e. Pemasangan *LCD*

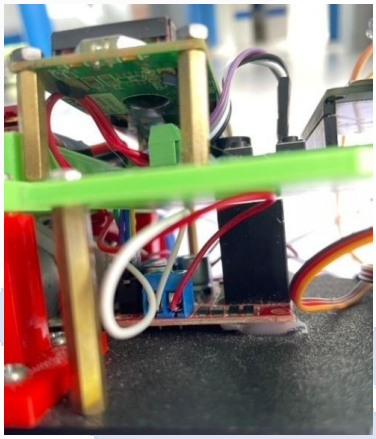
Pemasangan *LCD* di bagian atas robot bertujuan untuk memberikan informasi terkait menu – menu yang ada pada robot ini yang telah di program, seperti menu kalibrasi , mulai jalan, dll.



Gambar 4. 7 Pemasangan *LCD*

f. Pemasangan dan Perakitan Rangakaian Elektrik Komponen

Komponen seperti motor *driver* baterai dan ESP 32, dll. Dipasang di bagian atas dan bawah badan robot.



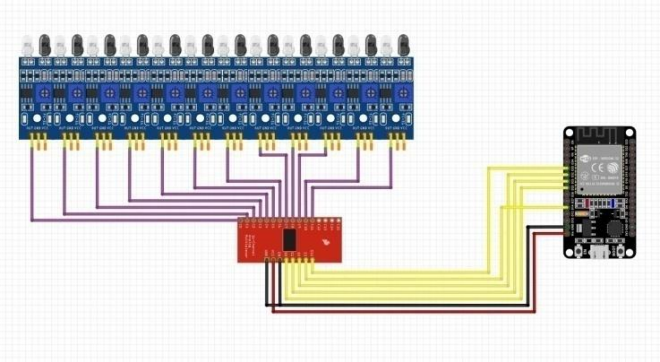
Gambar 4. 8 Rangkaian Elektrik Komponen

4.3. Pengujian Elektrik Robot

4.3.1. Pengujian Sensor Garis

Komponen digunakan untuk menghubungkan sensor ke garis di bawah ini. ESP 32, multiplekser, dan photodiode. Berdasarkan wiring ini, menggunakan mikrokontroler dengan jumlah pin terbatas sistem ini dapat membaca data. Multiplekser memiliki fungsi sebagai pengalih sinyal yang memungkinkan setiap sensor dibaca satu per satu secara berurutan. , Multiplekser sangat berguna untuk aplikasi yang membutuhkan pemantauan dari banyak titik, seperti robot pemindah

barang, sistem keamanan atau proyek *IoT* lainnya. Berikut merupakan kode warna pada gambar untuk mengidentifikasi koneksi, merah untuk sensor, sensor dan multiplekser koneksi berwarna ungu, dan koneksi antara multiplekser dan mikrokontroler berwarna kuning.



Gambar 4. 9 *Wiring* Sensor Garis

Berikut list program pada pengujian sensor garis

```
void Baca_Sensor(){
 Selektor(1,0,1,1);
 Sensor[1] = analogRead(Sig)/4;
 Selektor(0,0,1,1);
 Sensor[2] = analogRead(Sig)/4;
 Selektor(1,1,0,1);
 Sensor[3] = analogRead(Sig)/4;
 Selektor(0,1,0,1);
 Sensor[4] = analogRead(Sig)/4;
 Selektor(1,0,0,1);
 Sensor[5] = analogRead(Sig)/4;
 Selektor(0,0,0,1);
 Sensor[6] = analogRead(Sig)/4;
 Selektor(1,1,1,0);
 Sensor[7] = analogRead(Sig)/4;
 Selektor(0,1,1,0);
 Sensor[8] = analogRead(Sig)/4;
 Selektor(1,0,1,0);
 Sensor[9] = analogRead(Sig)/4;
 Selektor(0,0,1,0);
```

```
Sensor[10] = analogRead(Sig)/4;
Selektor(1,1,0,0);
Sensor[11] = analogRead(Sig)/4;
Selektor(0,1,0,0);
Sensor[12] = analogRead(Sig)/4;}
```

Fungsi Baca_Sensor() secara berurutan memilih masing-masing saluran sensor dengan memanggil fungsi Selektor() dengan pola bit tertentu. Setelah memilih saluran, fungsi ini membaca nilai analog dari pin Sig. Nilai analog tersebut kemudian dibagi dengan 4 agar bit yang dipakai berkurang dan disimpan dalam array Sensor[]. Indeks array Sensor dimulai dari 1 hingga 12, sesuai dengan saluran sensor yang dipilih.

Tabel 4. 1 Hasil Pengujian Sensor Garis

Kondisi	***Output* Sensor**
Bit [1,2]	110000000000
Bit [3,4]	001100000000
Bit [5,6]	000011000000
Bit [7,8]	000000110000
Bit [9 ,10]	000000001100
Bit [11, 12]	000000000011
Garis putih	000000000000
Garis hitam	111111111111

(a). (b).

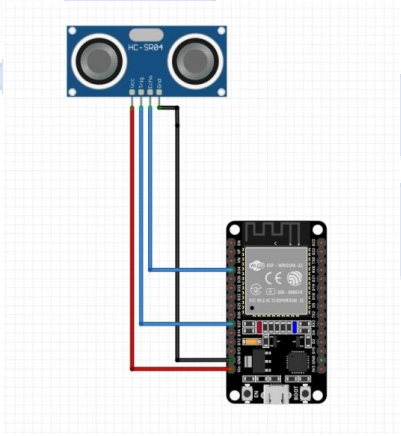
Gambar 4.10 Hasil Pengujian Sensor Garis (a). Bit [1,2] (b). Garis Hitam

Pada pengujian sensor garis di atas menunjukan hasil dari kondisi 12 bit

dan kondisi robot *line follower* pada penuh di garis hitam dan putih. Sensor pada robot *line follower* mampu mendeteksi posisi garis hitam maupun putih dengan baik. Dengan menggunakan nilai biner yang dihasilkan dari sensor dan ditammpilkan pada *LCD* robot dapat menentukan garis hitam relatif terhadap robot. Bit akan bernilai 1 jika robot terkena sensor garis dengan jalur berwarna hitam, sebaliknya bit akan bernilai 0 jika robot terkena sensor garis dengan jalur berwarna putih.

4.3.2. Pengujian Sensor *Ultrasonic*

Wiring Sensor *Ultrasonic* dibawah ini menggunakan komponen ESP 32 dan Sensor *Ultrasonic*. *Wiring* ini bertujuan untuk mengukur jarak menggunakan sensor *ultrasonic* HC-SR04 kemudian memproses data menggunakan mikrokontroler ESP 32. Mikrokontroler dapat mengirimkan sinyal *trigger*, menghitung jarak berdasarkan waktu perjalanan gelombang *ultrasonic*, dan menerima sinyal *echo* untuk mengirimkan data kembali ke mikrokontroler untuk menghitung jarak. Skema *Wiring*: Merah : *VCC*, Biru : *Trig*, Biru : *Echo*, Hitam : *GND*.



Gambar 4. 11 *Wiring* Sensor *Ultrasonic*

Berikut list program pengujian sensor *ultrasonic*

```
void Ultrasonik() {
  while(true){
   digitalWrite(trigPin, LOW);
   delayMicroseconds(2);
```

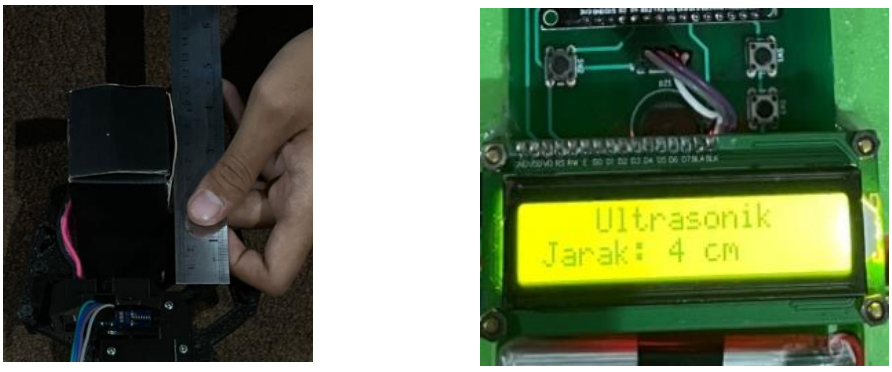
```
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
long duration = pulseIn(echoPin, HIGH);
distance = (duration * 0.0343) / 2; }}
```

Sebelum mengirimkan sinyal ultrasonik, pin trigPin diatur rendah untuk memastikan kondisi awal stabil. "digitalWrite(trigPin, HIGH);" Mengirimkan pulsa tinggi ke pin trigPin, dan delayMicroseconds(10) mempertahankan sinyal tinggi selama 10 mikrodetik sebelum dimatikan kembali dengan digitalWrite(trigPin, *LOW*). Gunakan *pulse*In(echoPin, HIGH) untuk menghitung berapa lama gelombang ultrasonik kembali akan bertahan setelah dipantulkan oleh objek. Perhitungan Jarak, Nilai 0.0343 adalah faktor yang mengubah mikrodetik menjadi sentimeter dan dibagi dua karena menghitung jarak hanya satu arah, yaitu pergi dan kembali dari objek. Waktu yang diukur dikonversi menjadi jarak dengan menggunakan rumus (waktu * 0.0343) / 2.

Tabel 4. 2 Hasil Pengujian Sensor Ultrasonik

Pengujian ke-	**Pengukuran penggaris (cm)**	**Sensor yang diukur**
1	4	4
2	8	8
3	12	12
4	16	16
5	20	20
6	24	24
7	28	28

Hasil pengujian sensor *ultrasonic* dapat dilihat pada gambar dibawah ini.



(a). (b).

Gambar 4.12 Hasil Pengujian Sensor *Ultrasonic* (a). Hasil Pengujian Menggunakan Penggaris (b). Hasil Pengujian Menggunakan Sensor *Ultrasonic*

Pada tabel 4.2 di atas dilakukan pengujian terhadap sensor *ultrasonic* sebanyak 7 kali. Pengujian manual dilakukan dengan menggunakan penggaris (cm) sedangkan pengujian sensor menggunakan program pada *Arduino IDE* yang hasilnya ditampilkan pada *LED*. Ketika nilai pengukuran dengan nilai yang keluar pada *LED* sama maka, menunjukkan hasil yang sesuai antara pengujian manual menggunakan penggaris dan pengujian menggunakan sensor *ultrasonic*. Nilai *error* didapatkan dengan persamaan dibawah ini.

$$\%ERROR = \frac{(Pengukuran\ Manual) - (Pengukuran\ Sensor)}{(Pengukuran\ Manual)}\ x\ 100\% \quad (4)$$

Perhitungan dari hasil data pengujian sensor *ultrasonic* pada proyek akhir ini sebagai berikut :

$$\%ERROR = \frac{(4cm) - (4cm)}{(4cm)}\ x\ 100\% = 0\%$$

Dari hasil pengujian pada tabel 4.3.2 yang dilakukan menghasilkan hasil yang sama dengan presentase *error* 0% untuk setiap pengujian, menunjukkan sensor *ultrasonic* memiliki tingkat akurasi yang tinggi.

4.3.3. Pengujian *Line*aritas antara PWM dan RPM

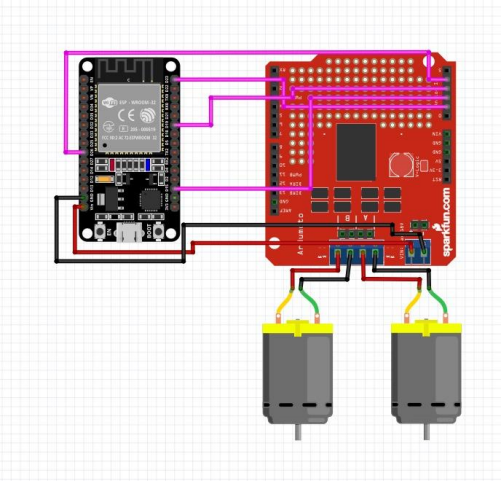
Berikut adalah diagram pengkabelan motor *driver* yang digunakan untuk menguji hubungan linear antara PWM *(Pulse Width Modulation)* dan *RPM (Rotations Per Minute)*. Diagram ini melibatkan beberapa komponen sebagai berikut:

1. Esp 32
2. Motor *driver*
3. Motor DC

Koneksi mikrokontroler ke moto *driver*

- Power dan Ground
  - Pin *VCC* terhubung ke pin *VCC* pada motor *driver*
  - Pin *GND* mikrokontroler terhubung ke pin *GND* pada motor *driver*
- Kontrol Pin
  - Pin GPIO dari mikrokontroler terhubung ke pin kontrol pada motor *driver* yang berfungsi sebagai pengendali arah dan kecepatan motor.
  - Pin GPIO ke berbagai input pada motor *driver* yang ditunjukkan kabel berwarna ungu.

Mikrokontroler ESP 32 dapat mengendalikan dua motor DC menggunakan motor *driver* dapat mengendalikan motor sesuai dengan arah dan kecepatan yang diinginkan.



Gambar 4. 13 *Wiring* Motor Driver

Berikut merupakan list program pengujian persamaan linear antara PWM dan RPM.

```
//Variable Motor Driver
#define in1 23
#define in2 15
#define in3 19
#define in4 26
//Kecepatan Motor
int rightspeed = 200;
int leftspeed = 200;
void setup() {
//Deklarasi Motor Driver
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);}
void loop() {
  analogWrite(in1, 0);
  analogWrite(in2, rightspeed);
  analogWrite(in3, 0);
  analogWrite(in4, leftspeed);}
```

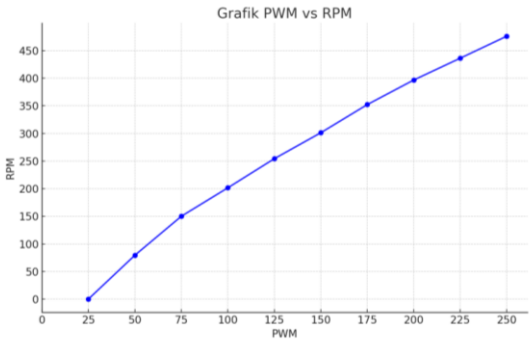
Pada program ini berfungsi untuk mengendalikan dua motor menggunakan motor *driver* L298N. Program ini mengatur pin yang terhubung ke motor *driver* menggunakan #define untuk kemudahan perubahan kode. Motor *driver* digunakan untuk mengatur arah putaran dan kecepatan motor secara mandiri. Kecepatan masing-masing motor ditentukan oleh variabel kecepatan kiri dan kanan.

Pada fungsi utama “loop()’ berisi perintah analogWrite() yang digunakan untuk mengirimkan sinyal PWM ke pin-pin yang mengontrol motor *driver*. Pada motor kanan, pin in1 diatur menjadi *LOW* dan in2 diatur menggunakan rightspeed untuk mengatur kecepatan motor. Pada motor kiri, pin in3 diatur menjadi *LOW* dan in4 diatur menggunakan *left speed* untuk mengatur kecepatan motor.

Tabel 4.3 Pengujian *Line*aritas antara PWM dan RPM

Pengujian	PWM	RPM
1	25	0
2	50	79.4
3	75	150
4	100	201.5
5	125	254.2
6	150	301.2
7	175	352.2
8	200	396.7
9	225	436.2
10	250	476.0

Gambar 4. 14 Hasil Pengujian Persamaan *Line*ar antara PWM dan RPM

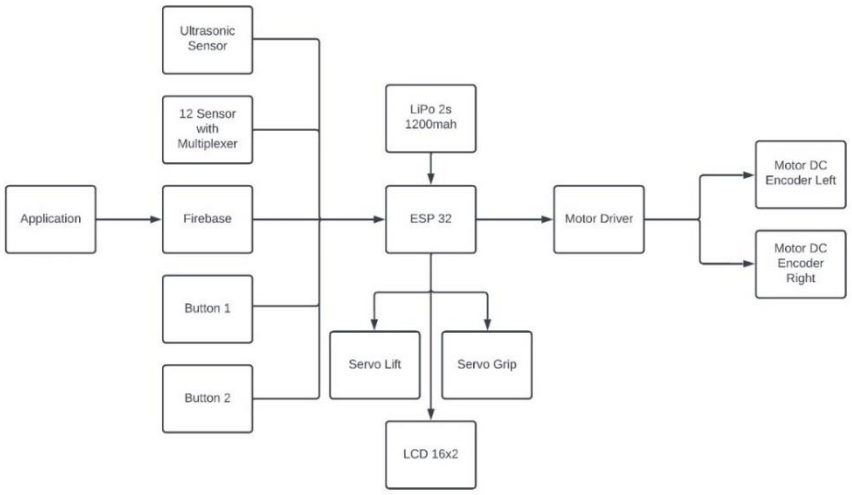


Gambar 4. 15 Grafik Pengujian Persamaan *Line*ar antara PWM dan RPM

Pada hasil pengujian linearitas antara PWM dan RPM di atas dapat diketahui bahwa ada RPM meningkat secara proporsional ketika nilai PWM meningkat. Pada nilai PWm 25, RPM adalah 0, menunjukkan bahwa pada nilai RPM rendah motor tidak bergerak sehingga tidak terukur. RPM meningkat secara signifikan dengan peningkatan PWM. Dimulai dari PWM 50 hingga 250, RPM meningkat dari 79.4 hingga 476.0. menunjukkan bahwa motor berfungsi secara efektif dan merespon perubahan PWM dengan baik. Setiap peningkatan PWM sebesar 25 menghasilkan kenaikan RPM yang cukup konsisten, misalnya: dari PWM 25 ke 50, RPM meningkat dari 0 ke 79.4, dari PWM 50 ke 75, RPM meningkat dari 79.4 ke 150, dan seterusnya hingga PWM 225 ke 250, RPM meningkat dari 436.2 ke 476.0.

4.3.4. Blok Diagram Sistem

Dibawah ini merupakan gambar dari Blok Diagram Sistem.



Gambar 4.16 Blok Diagram Sistem

Pada blok diagram di atas menggambarkan sistem robot yang dikendalikan ESP 32 sebagai otak sistem yang mendapatkan daya dari baterai LiPo. Dilengkapi dengan sensor ultrasonik untuk mendeteksi jarak atau keberadaan objek disekitarnya. Terdapat juga multiplexer yang memungkinkan ESP32 membaca data dari 12 sensor yang berbeda menggunakan input yang lebih sedikit. Sensor – sensor ini digunakan untuk berbagai keperluan seperti

pendeteksian garis yang membantu jalannya robot. Sistem ini terintegrasi dengan *Firebase*, sebuah layanan cloud yang digunakan untuk menyimpan dan mengambil data secara real-time. Push button 1 dan 2 terhubung ke ESP32 dan digunakan untuk kontrol manual langsung, misalnya untuk mengaktifkan atau menonaktifkan robot. Motor *driver* untuk mengendalikan motor DC, mengatur arah dan kecepatan motor yang menggerakkan roda kiri dan kanan robot. Selain itu, robot ini dilengkapi dengan servo lift dan servo grip untuk melakukan tindakan mekanis seperti mengangkat dan menggenggam objek. Informasi mengenai status operasi robot, data sensor, atau pesan lainnya ditampilkan pada layar *LCD* 16x2 yang terhubung ke ESP32.

4.3.5. Pengujian Aplikasi

Pengujian aplikasi dilakukan dengan membuka aplikasi MIT pada *mobile* kemudian memasukkan koordinat X dan Y pada *Current Robot Coordinates*, Robot *Destination Coordinates*, dan *Object Destination Coordinates*, setelah itu klik *start*. Jika ingin mereset klik tombol reset, tombol kalibrasi digunakan untuk kalibrasi robot, lihat sensor untuk sensor ultrasonik dan sensor garis, kemudian ambil data digunakan untuk melihat data yang sudah di input pada *current*, *destination*, dan *object* pada aplikasi apakah sesuai dengan yang sudah diinput atau tidak.

Gambar 4. 17 Pengujian Aplikasi

Berikut list program pengujian aplikasi

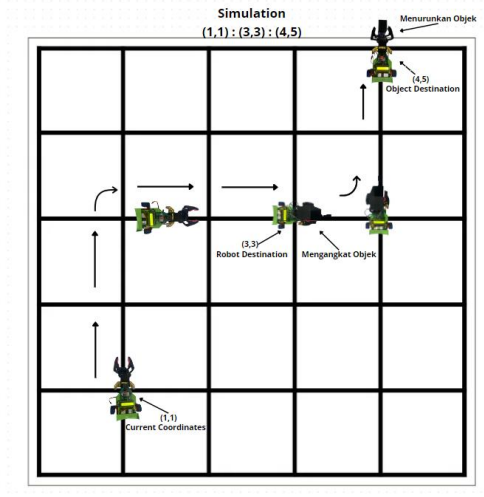
```
void ambilData(){
  lcd.clear();
  while(true){
    lcdPrint(0,0, "Ambil Data" );
    if (Firebase.ready() && signupOK ) {
      if (Firebase.RTDB.getString(&fbdo, "TA/cx")) {
        if (fbdo.dataType() == "string") {
cx = fbdo.stringData();
          CX = cx.toInt();
sprintf(buff, "CX Berhasil: %i",CX);
          lcdPrint(0, 1, buff);
          delay(100);
        }
      }
      if (Firebase.RTDB.getString(&fbdo, "TA/cy")) {
        if (fbdo.dataType() == "string") {
          cy = fbdo.stringData();
          CY = cy.toInt();
          sprintf(buff, "CY Berhasil: %i",CY);
          lcdPrint(0, 1, buff);
          delay(100);
        }
      }
      if (Firebase.RTDB.getString(&fbdo, "TA/ox")) {
        if (fbdo.dataType() == "string") {
ox = fbdo.stringData();
OX = ox.toInt();
lcdPrint(0,1, "OX Berhasil" );
          sprintf(buff, "OX Berhasil: %i",OX);
          lcdPrint(0, 1, buff);
          delay(100);
        }
      }
```

```
      if (Firebase.RTDB.getString(&fbdo, "TA/oy")) {
        if (fbdo.dataType() == "string") {
          oy = fbdo.stringData();
          OY = oy.toInt();
          sprintf(buff, "OY Berhasil: %i",OY);
          lcdPrint(0, 1, buff);
          delay(100);
        }
      }
      if (Firebase.RTDB.getString(&fbdo, "TA/dx")) {
        if (fbdo.dataType() == "string") {
dx = fbdo.stringData();
          DX = dx.toInt();
sprintf(buff, "DX Berhasil: %i",DX);
          lcdPrint(0, 1, buff);
          delay(100);
        }
      }
      if (Firebase.RTDB.getString(&fbdo, "TA/dy")) {
        if (fbdo.dataType() == "string") {
          dy = fbdo.stringData();
          DY = dy.toInt();
          sprintf(buff, "DY Berhasil: %i",DY);
          lcdPrint(0, 1, buff);
          delay(100);
        }
      }
      if(Button(2) == LOW){
        while(Button(2) == LOW){}
        delay(BOUNCE);
        break;
      }
      Serial.println(cx + ":" + cy + ":" + ox + ":" +
oy + ":" + dx + ":" + dy); }}}
```

Pada program fungsi “ambilData()” yang dirancang untuk berjalan secara terus-menerus, membaca data koordinat dari *Firebase* RTDB, setelah itu menampilkan data tersebut di layar *LCD*, dan menunggu input dari pengguna untuk memberhentikan program tersebut melalui pushbutton. Sebelum menjalankan program, pastikan bahwa koneksi internet berjalan lancer sehingga tidak terjadi gangguan penerimaan data dari *Firebase* RTDB.

Tabel 4.4 Pengujian Aplikasi

No.	Current Coordinates (x,y)	Robot Destination (x,y)	Object Destination (x,y)	Hasil
1	(1,2)	(3,4)	(4,4)	Berhasil
2	(1,1)	(3,3)	(4,5)	Berhasil
3	(2,2)	(4,4)	(5,5)	Berhasil
4	(0,1)	(2,4)	(4,5)	Berhasil
5	(1,2)	(2,2)	(3,4)	Berhasil



Gambar 4. 18 Simulasi Pengujian Aplikasi

Pada tabel diatas dilakukan 5 kali pengujian pada aplikasi dengan 5 data koordinat yang berbeda robot berhasil memindahkan dan meletakkan barang sesuai koordinat yang telah diinput pada aplikasi. Robot mampu mencapai

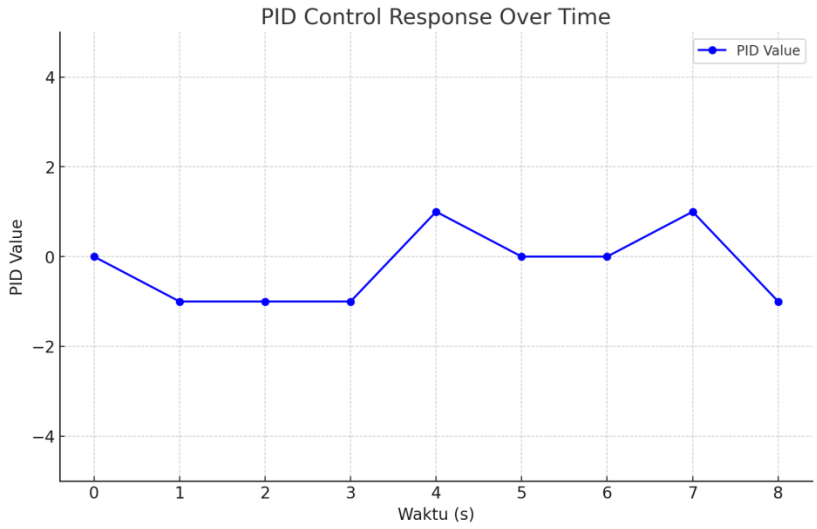
tujuan dengan akuarsi tinggi, baik untuk koordinat tujuan robot maupun tujuan objek, tanpa mengalami kesalahan. Dapat disimpulkan bahwa apliaksi dan algoritma yang digunakan dalam robot *line follower* ini efektif untuk tugas pemindahan barang sesuai dengan koordinat yang diinput.

4.3.6. Pengujian Kontrol PID Terhadap Sensor Garis

Pengujian menggunakan *setpoint* dilakukan untuk mengevaluasi dampak nilai PID terhadap pembacaan sensor garis. Berikut adalah daftar program uji untuk kontrol PID terhadap sensor garis.

```
void Piddd(){
   check_sensor();
   error = setpoint  - sensor;
   integral = error + integral;
   // //batas nilai integral
   if(integral >=15 ){
     integral = 15;
   }
   if(integral <=-15 ){
     integral = -15;
   }
   derivative = (error-errorTerakhir);
   //penghitung PID
   PID = (Kp * error + Kd * derivative + Ki * integral);
   errorTerakhir = error;
   if(PID >= 100){
     PID = 100;
   }
   if(PID <= -100){
     PID = -100;
   }
   totalkiri = leftspeed - (PID);
   totalkanan = rightspeed + PID;
   forward(totalkanan, totalkiri); }
```

Berikut adalah hasil eksperimen dari pengujian kontrol PID terhadap sensor garis.



Gambar 4. 19 Grafik Pengujian PID Kontrol terhadap Sensor Garis

Dengan nilai KP = 8, Ki = 0, Kd = 2, 5

Pada tahap ini, pengujian PID dilaksanakan agar dapat membuat jalan robot *line follower* stabil. Motor kiri dan kanan menggunakan base PWM yang sama, yaitu 150. Kp =8, kesalahan kecil dari garis akan menghasilkan koreksi yang cukup besar membuat ropot cepat kembali ke jalur. Ki = 0 , untuk kesalahan *steady-state* tidak ada koreksi pada robot *line follower* kesalahan *steady-state* sangat kecil. Kd = 2,5 akan meredam komponen *proportional* dan respon yang terlalu agresif. Pada grafik di atas menunjukkan bahwa sistem kontrol PID relatif stabil dan tidak mengalami fluktasi yang ekstrem, nilai PID sebagian besar berkisara antara -1 dan 1 selama periode waktu yang dianalisis.

Koreksi kesalahan pada grafik PID diatas antara lain sebagai berikut :

1. Nilai PID adalah -1, pada detik 1 hingga 3, menunjukkan bahwa sistem mencoba mengoreksi kesalahan yang negatif.
2. Nilai PID berubah menjadi 1, menunjukkan adanya koreksi yang positif untuk menyeimbangkan kembali sistem.
3. Nilai PID kembali ke 0, pada detik 5 dan 6 berarti telah mencapai titik

keseimbangan.

4. Nilai PID berubah menjadi 1 lagi, adanya gangguan atau perubahan kondisi yang memerlukan penyesuaian.
5. Nilai PID kembali menjadi -1, adanya koreksi kesalahan negatif yang diperlukan.

4.3.7. Pengujian Keseluruhan

Pada tahap ini, dilakukan pengujian menyeluruh untuk memverifikasi apakah alat yang telah dibuat dapat berfungsi dan beroperasi sesuai dengan algoritma yang menggunakan *path planning* berbasis aplikasi *mobile*.

Tabel 4. 5 Hasil Pengujian Robot Tanpa Beban

Percobaan (kali)	Current (x,y)	Object (x,y)	Destination (x,y)	Keberhasilan	Persentase (%)
5	(1,1)	(2,3)	(3,4)	4 kali	80%
5	(2,2)	(3,4)	(4,5)	4 kali	80%
5	(1,3)	(2,4)	(4,5)	4 kali	80%

Dari tabel percobaan 4.5 diatas, bisa dilihat bahwa di percobaan tanpa beban rata-rata keberhasilan 80%, ada dua hal yang mempengaruhi beberapa kegagalan tersebut, seperti kondisi baterai dan jarak koordinat. Pada pengujian dengan koordinat (1,1),(2,3),(3,4) robot tidak berhasil mencapai tujuan karena kesalahan pada putaran belok robot dipersimpangan yang tidak konsisten kadang kurang, kadang lebih dan gagal di titik koordinat (1,3). Pada pengujian dengan koordinat (2,2),(3,4),(4,5) pada kondisi ini robot tidak berhasil mencapai tujuan karena putaran belok kekiri robot dipersimpangan kurang dan gagal di titik koordinat (4,4). Pada pengujian dengan koordinat (1,3),(2,4),(4,5) robot tidak berhasil mencapai tujuan karena putaran belok robot dipersimpangan kurang dan gagal di titik koordinat (1,4).

Tabel 4. 6 Hasil Pengujian Robot Dengan Beban

Percobaan (kali)	Current	Object	Destination	Beban	Keberhasilan	Persentase (100%)
	(x,y)	(x,y)	(x,y)			
3	(1,1)	(3,5)	(5,4)	31	3 kali	100%
3	(1,1)	(3,3)	(4,5)	52	3 kali	100%
3	(0,2)	(3,3)	(4,4)	100	2 kali	66.67%
3	(1,3)	(3,3)	(5,4)	152	2 kali	66.67%

Pada pengujian robot dengan beban, selain dari kedua hal tersebut berat beban juga berpengaruh besar dengan pengujian robot sehingga dari 4 percobaan, 2 percobaan dengan beban yang lebih berat tidak 100% berhasil mencapai tujuan. Pada pengujian di koordinat (0,2),(3,3),(4,4) sensor gagal membaca simpang karena tidak ada simpang di kiri sensor yang kadang mengakibatkan sensor tidak memenuhi syarat untuk berbelok dan gagal di titik (0,3). Pada pengujian di koordinat (1,3),(3,3),(5,4) sensor gagal membaca simpang karena tidak ada simpang di kanan sensor yang kadang mengakibatkan sensor tidak memenuhi syarat untuk menurunkan benda dan gagal di titik (4,4).

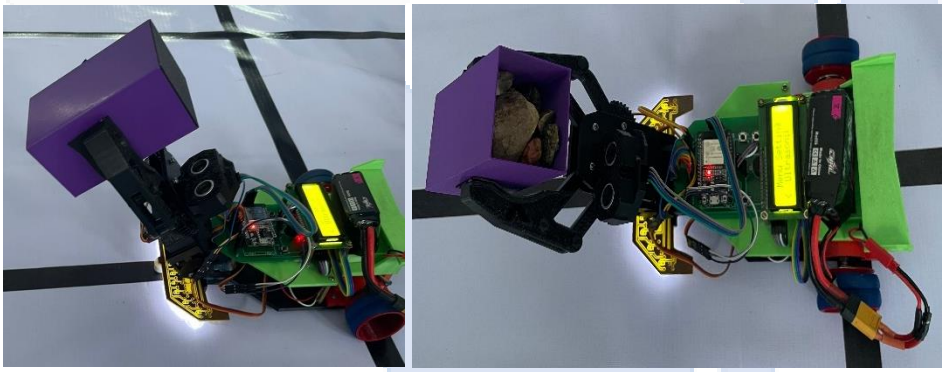
Tabel 4. 7 Hasil Waktu Tempuh Robot Pengantar Barang

Koordinat	Beban (gram)	Waktu (s)	Keterangan
(2,3), (3,5), (4,4)	31	21 detik	Berhasil
(2,3), (3,5), (4,4)	52	25 detik	Berhasil
(2,3), (3,5), (4,4)	75	27 detik	Berhasil
(2,3), (3,5), (4,4)	93	29,5 detik	Berhasil
(2,3), (3,5), (4,4)	100	30 detik	Berhasil
(2,3), (3,5), (4,4)	152	32 detik	Berhasil
(2,3), (3,5), (4,4)	200	35 detik	Berhasil
(2,3), (3,5), (4,4)	245	37 detik	Berhasil
(2,3), (3,5), (4,4)	317	39,5 detik	Berhasil

Pengujian hasil pengukuran beban dapat dilihat pada gambar dibawah ini.

(a). (b)

Gambar 4.18 Hasil Pengukuran Beban (a) Hasil Pengukuran Beban Teringan (b) Hasil Pengukuran Beban Terberat.



(a). (b).

Gambar 4.20 Robot Berhasil Angkat Beban (a). Robot Berhasil Angkat Beban Teringan (b). Robot Berhasil Angkat Beban Terberat

Berdasarkan pengujian diatas, semua variable mempengaruhi durasi perjalanan robot, seperti berat beban yang diangkat, jarak koordinat dari satu titik ke titik yang lain, dan kondisi baterai. Ketika pertama robot berjalan sebelum mengangkat beban, daya dibutuhkan sama. Semakin berat beban maka akan semakin besar pula daya motor servo untuk mengangkat beban tersebut, sehingga mempengaruhi kecepatan motor DC.

Pada pengujian diatas current koordinat atau kondisi robot saat ini adalah (2,3), dengan koordinat *object* berada di (3,5), dan koordinat destination atau tujuan akhir berada di (4,4). Semua pengujian beban yang berbeda memiliki koordinat current, *object* , dan destination yang sama. Pada tabel diatas dapat dilihat bahwa, pada beban 31 gram, waktu perjalanan 21 detik. Begitu pula pada beban 52 gram, adanya peningkatan waktu menjadi 25 detik yang menunjukkan penggunaan daya yang lebih besar akibat peningkatan beban sehingga kecepatan motor melambat. Waktu perjalanan meningkat seiring penambahan berat beban. Kenaikan ini lebih signifikan, mempengaruhi kinerja robot secara nyata yang diakibatkan oleh beban. Pada beban 245 dan 317 gram, waktu perjalanan meningkat masing – masing menjadi 37 detik dan 39,5 detik. Seiring peningkatan beban ini menunjukkan bahwa robot mengalami penurunan kinerja yang dikarenakan penggunaan daya yang semakin besar sehingga motor melambat, yang mengakibatkan waktu tempuh robot menjadi semakin lama.

BAB V
KESIMPULAN & SARAN

5.1. KESIMPULAN

Berdasarkan data dan hasil pengujian yang telah dilakukan, dapat disimpulkan bahwa :

1. Pada keseluruhan, robot dapat berjalan sesuai koordinat yang telah dibuat. Dengan berat minamal beban 31 gram dalam waktu 21 detik dan maksimal berat beban 317 gram dalam waktu 39,5 detik.
2. Robot dapat memudahkan operator dalam pemindahan barang melalui aplikasi *mobile* secara otomatis. Dengan menginput current coordinates, *object* coordinates, dan destination coordinates, kemudian klik *start* melalui aplikasi robot dapat berjalan dan memindahkan barang.
3. *Gripper* pada robot berfungsi dengan baik dan mampu menggerakkan barang dari satu titik koordinat ke titik lainnya, dengan beban yang bervariasi dari 31 - 317 gram dalam waktu yang meningkat seiring betambahnya beban. Menggunakan multiplekser dengan 12 kanal, jalur input ADC 10bit robot dapat mengikuti garis. Nilai ADC (0-1023) dihasilkan oleh sensor photodiode. Nilai putih harus kurang dari 800 dan nilai hitam harus lebih dari 1023.
4. Hasil pengujian sistem kontrol PID pada proyek akhir ini menggunakan metode trial dan *error* dan menghasilkan nilai Kp, Ki , dan Kd terbaik yaitu, nilai Kp = 8, Ki = 0 Kd = 2,5.

5.2. SARAN

1. Sebaiknya menggunakan encoder pada motor DC agar pergerakan untuk berbelok jadi lebih maksimal dan akurat.
2. Menambahkan sensor arah/kompas agar mempermudah membuat algoritma *path planning*.

DAFTAR PUSTAKA

[1] M. Basri and I. Wahira, "ROBOT LINE FOL*LOW*ER PEMINDAH BARANG BERDASARKAN WARNA BERBASIS MIKROK*ON*TROLER Informasi Artikel," 2022. [Online]. Available: http://jurnal.umpar.ac.id/indeks/jmosfet▪11

[2] M. Basri and I. Wahira, "Robot *Line Follower* Pemindah Barang Berdasarkan Warna Berbasis Mikrokontroler," *Jurnal Mosfet*, vol. 2, no. 2, pp. 11–15, 2022, doi: 10.31850/jmosfet.v2i2.1973.

[3] A. Mukti, D. Nurhayati, D. Widianto, and J. P. Sudharto, "RANCANG BANGUN SISTEM K*ON*TROL ROBOT LINE FOL*LOW*ER MENGGUNAKAN LOGIKA FUZZY."

[4] N. Wenda, "Rancang Bangun Robot *Line Follower* Untuk Menentukan Jarak Terpendek pada *Maze* Mapping," *Universitas Sam Ratulangi*, pp. 1–9, 2022.

[5] M. Sony and T. Via, "Optimasi Sistem Navigasi Pada Robot Pengantar Makanan," 2023.

[6] D. Y. Krisna and S. Satrio, "Perancangan Robot *Line Follower* Pemadam Api," *Jurnal Algoritma, Logika dan Komputasi*, vol. 2, no. 2, pp. 176–189, 2019, doi: 10.30813/j-alu.v2i2.1847.

[7] A. Putri and F. Maspiyanti, "Robot *Line Follower* Pengantar Surat Menggunakan Metode *Fuzzy Logic* Studi Kasus Fakultas Teknik Universitas Pancasila," *Jurnal Teknologi Terpadu*, vol. 3, no. 1, pp. 1–8, 2017, doi: 10.54914/jtt.v3i1.76.

[8] B. Kriesna Nugraha, A. S. Santia, Z. Aurellia, P. Pangaribuan, and I. M. Rodiana, "Robot Pengantar Makanan berbasis *Line Follower* dengan Sensor Warna TCS3200 dan *Internet of Things (IoT)*," 2023. [Online]. Available: http://Jiip.stkipyapisdompu.ac.id

[9] A. Putri and F. Maspiyanti, "Robot *Line Follower* Pengantar Surat Menggunakan Metode *Fuzzy Logic* Studi Kasus Fakultas Teknik Universitas Pancasila," *Jurnal Teknologi Terpadu*, vol. 3, no. 1, pp. 1–8, 2017, doi: 10.54914/jtt.v3i1.76.

[10] A. Karakteristik, M. Berdasarkan, P. Hdpe, S. Pengganti, S. Aspal, and P. Lapis, “Jurnal simetrik vol 13, no. 2, desember 2023,” vol. 13, no. 2, pp. 716–722, 2023.

[11] M. N. Nizam, Haris Yuana, and Zunita Wulansari, “Mikrokontroler Esp 32 Sebagai Alat Monitoring Pintu Berbasis Web,” *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 6, no. 2, pp. 767–772, 2022, doi: 10.36040/jati.v6i2.5713.

[12] M. I. Marzuki, I. N. Farida, and J. Sahertian, “Implementasi Controller PID (Proportional, Integral, Derivative) pada Robot Sepak Bola Beroda,” *Prosiding SEMNAS Inovasi Teknologi*, pp. 297–302, 2021.

[13] M. Restu and B. Syaiful, “Perbandingan Sistem Pengontrolan Pid Konvensional Dengan Pengontrolan Cmac , *Fuzzy Logic* Dan Ann Pada Water Level,” *Sigma Epsilon*, vol. 17, no. 3, pp. 129–141, 2013.

[14] I. Riyanto, L. Margatama, R. Rizkia, and E. Marantika, “Robot Forklift *Line Follower* dengan Kendali PID dan Sensor Warna.” [Online]. Available: http://journal.univpancasila.ac.id/index.php/joule/

[15] I. D. Gusti Made Andi and B. Setiyono dan Enda Wista Sinuraya, “ROBOT MOBIL LINE FOL*LOW*ER DENGAN KENDALI PID SEBAGAI PENGEMBANGAN BRT TRANS SEMARANG.” [Online].

[16] D. Persada, D. Andayati, and E. Fakhiyah, “PENDETEKSI DINI KEBOCORAN PADA TABUNG GAS MENGGUNAKAN SENSOR MQ-6 BERBASIS ARDUINO,” 2019. [Online]. Available: https://fahmizaleeits.com/

[17] A. Widya Gumelar, D. Syauqy, and S. R. Akbar, “Implementasi Metode *Simple Maze Wall Follower* Dengan Menggunakan *Free RTOS* Pada Robot *Maze*,” 2018. [Online]. Available: http://j-ptiik.ub.ac.id

[18] P. Stevano *et al.*, “JURNAL EINSTEIN Jurnal Hasil Penelitian Bindang Fisika IMPLEMENTASI SENSOR ULTRAS*ON*IK HC-SR04 SEBAGAI SENSOR PARKIR MOBIL BERBASIS ARDUINO,” Dipublikasikan, 2017.[Online].Available: http://jurnal.unimed.ac.id/2012/index.php/inpafie-issn:2407-747x,p-issn2338-1981

[19] A. Syariffudin, A. Ashari, and U. Pratiwi, “Perancangan Alat Peraga Gerak Harmonik Berupa Bandul Matematis Menggunakan Sensor Photodioda Berbasis *Arduino*,” *Jurnal Pendidikan Sains dan Komputer*, vol. 2, no. 01, pp. 196–207, 2022, doi: 10.47709/jpsk.v2i01.1462.

[20] A. S. Romadhon and F. Adiputra, “ROBOT LINE FOL*LOW*ER PEMISAH BENDA MENGGUNKAN SENSOR WARNA TCS 3200,” vol. 6, no. 3, 2018.

[21] “Perancangan Robot *Auto Line Follower* yang Menerapkan Metode Osilasi Ziegler-Nichols Untuk Tuning Parameter PID pada Kontes Robot Indonesia.”

[22] Z. Ayu Kurnia Sari, H. Permana, and W. Indrasari, “KARAKTERISASI SENSOR PHOTODIODA, DS18B20, DAN KO*N*DUKTIVITAS PADA RANCANG BANGUN SISTEM DETEKSI KEKERUHAN DAN JUMLAH ZAT PADAT TERLARUT DALAM AIR,” 2017, doi: 10.21009/SPEKTRA.

[23] P. Prasetyo, R. Maulana, and E. Setiawan, “Sistem Kontrol PID pada Purwarupa Robot Pembawa Barang berbasis *Line Follower* menggunakan Algoritma *Path Planning*,” 2022. [Online]. Available: http://j-ptiik.ub.ac.id

[24] R. Amirullah, A. Rusdina, and D. Darlis, “Implementasi Sistem *Path Planning* dan Routing untuk *Mobile* Robot Berbasis Visible Light Communication Implementation of *Path Planning* and Routing System Based-on Visible Light Communication for *Mobile* Robot,” vol. 8, no. 5, p. 4283, 2021.

[25] M. Basri and I. Wahira, “Robot *Line Follower* Pemindah Barang Berdasarkan Warna Berbasis Mikrokontroler,” *Jurnal Mosfet*, vol. 2, no. 2, pp. 11–15, 2022, doi: 10.31850/jmosfet.v2i2.1973.

[26] S. Sirmayanti, S. Amelia, N. Afifah, and I. Abduh, “Rekayasa Sistem Kendali *Gripper* melalui Robot Transporter menggunakan WiFi Module ESP8266,” *Jurnal Telekomunikasi dan Komputer*, vol. 11, no. 1, p. 51, 2021,

LAMPIRAN 1

DAFTAR RIWAYAT HIDUP

1. Data Pribadi

Nama Lengkap : Dilah Andini

Tempat Tanggal Lahir : Bekasi, 02 Agustus 2003

Alamat Rumah : Jl. Jendral Sudirman, Gang Kelud, Air Merapin, RT 001 RW 000, Kota Sungailiat Kab.Bangka Prov. Kep. Bangka Belitung

No.Telp/HP : 082175433831

Email : dilahandini36@gmail.com

Jenis Kelamin : Perempuan

Agama : Islam

2. Riwayat Pendidikan

SD Negeri 26 Sungailiat (2008-2014)

SMP Negeri 5 Sungailiat (2014-2017)

SMK Negeri 1 Sungailiat (2017-2020)

Sungailiat, 16 Juli 2024

Dilah Andini

1. Data Pribadi

Nama Lengkap	: Sahrul Ramadhan
Tempat Tanggal Lahir	: Pangkalpinang, 25 Oktober 2003
Alamat Rumah	: Jl. Marica no.292 Pintu Air RT08/RW03 Kec.Rangkui, Kota Pangkalpinang Prov. Kep. Bangka Belitung
No.Telp/HP	: 082192039236
Email	: sahrulramadhan.xitkj@gmail.com
Jenis Kelamin	: Laki - laki
Agama	: Islam

2. Riwayat Pendidikan

SD Negeri 16 Pangkalpinang (2008-2014)

SMP Negeri 5 Pangkalpinang (2014-2017)

SMK Negeri 2 Pangkalpinang (2017-2020)

Sungailiat,16 Juli 2024

Sahrul Ramadhan

LAMPIRAN 2

PEMROGRAMAN SISTEM KESELUHURAN

```
//Menginclude Semua Library Yang Dibutuhkan
#include<Wire.h>
#include<ESP32Servo.h>
#include<Firebase_ESP_Client.h>
#include<WiFi.h>
#include"addons/TokenHelper.h"
#include"addons/RTDBHelper.h"
//Mendefine SSID dan Password WIFI
#defineWIFI_SSID"E'Ok"
#defineWIFI_PASSWORD"Edo'2024"
//Mendefine Data API & URL Firebase
#defineAPI_KEY"AIzaSyCtsdmvLEyR9ST8rfgv0BvBUheEoqjtrxE"
#defineDATABASE_URL"https://line-follower-mapping-default-
rtdb.firebaseio.com/"
//Define Firebase Data object
FirebaseDatafbdo;
FirebaseAuth auth;
FirebaseConfig config;
String sValue, sValue2;
boolsignupOK = false;
String cx, cy, ox, oy, dx, dy, start;
int CX, CY, OX, OY, DX, DY, Start;
int  jarak Ke Benda X, jarak Ke Benda Y, jarak Ke Tujuan X,
jarak Ke Tujuan Y;
//Variable Selector Sensor
#defineS025
#defineS133
#defineS232
#defineS314
#defineSig35
//Variable Push Button
#definePb15
#definePb24
```

```
#definePb32
//Variable Motor Driver
#definein123
#definein215
#definein319
#definein426
//Variable Ultra Sonik Sensor
constinttrigPin = 27; // Pin untuk Trig pada sensor
constintechoPin = 34; // Pin untuk Echo pada sensor
int distance;
//Variable Servo
Servo servoLift;
Servo servoGrip;
#defineservoLiftPin12
#defineservoGripPin13
intposLift = 100;
intposGrip = 0;
inttargetLift = 35;
inttargetDrop = 100;
inttargetGrip = 180;
inttargetRelease = 0;
int increment = 15;
intincrementl = 1;
unsignedlong interval = 20;
//Variable Sensor Garis
intSensor[13], Nilai_Max[13], Nilai_Min[13], Nilai_Ref[13];
int BOUNCE = 300;
int menu = 1;
booleanBit[13];
charbuff[16];
//Deklarasi LCD
#include<LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2); // I2C address 0x27, 16
column and 2 rows
```

```
//Deklarasi Data PID
#definesetpoint 0
floatKp = 8;
float Ki = 0;
floatKd = 2.5;
int error, sensor, derivative, integral, PID, error  Terakhir,
total kiri, total kanan;
//Deklarasi Kecepatan Motor
intrightspeed = 200;
intleftspeed = 200;
voidsetup() {
//Mengaktifkan Serial Monitor
  Serial.begin(115200);
//Mengkoneksikanke WIFI
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(300); }
//Menampilkan IP dan Mengkoneksikan ESP32 ke Firebase
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  config.api_key = API_KEY;
  config.database_url = DATABASE_URL;
  if (Firebase.signUp(&config, &auth, "", "")) {
    Serial.println("ok");
    signupOK = true;
  }
  else {
    Serial.printf("%s\n",
config.signer.signupError.message.c_str());
  }
```

```
  config.token_status_callback = tokenStatusCallback;
  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
//Set Push Button START Aplikasike 0
  if(Firebase.RTDB.setString(&fbdo, "TA/start", 0)) {
    Serial.println("Button state updated successfully");
  } else {
    Serial.println("Failed to update button state");
    Serial.println(fbdo.errorReason()); }
  //Deklarasi LCD
  lcd.init(); // initialize the lcd
  lcd.backlight();
  //DeklarasiSelektorSebagai Output
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);
  pinMode(Sig, INPUT);
  //Deklarasi Buzzer
  pinMode(18, OUTPUT);
  //Deklarasi Sensor Ultra Sonik
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  //Deklarasi Pin Servo
  servoLift.attach(servoLiftPin);
  servoGrip.attach(servoGripPin);
  servoLift.write(posLift);
  servoGrip.write(posGrip);
  //Deklarasi Push Button Sebagai Output
  pinMode(Pb1, INPUT_PULLUP);
  pinMode(Pb2, INPUT_PULLUP);
  pinMode(Pb3, INPUT_PULLUP);
  //Deklarasi Motor Driver
  pinMode(in1, OUTPUT);
```

```
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT); }
voidloop() {
  START();
  lcdPrint(0,0, "  Menu Setting  ");
//Program Menu Setting di LCD
  if(Button(1) == LOW){
    while(Button(1) == LOW){}
    delay(BOUNCE);
    menu++;
    if(menu>5)menu=1; }
  if(Button(2) == LOW){
    while(Button(2) == LOW){}
    delay(BOUNCE);
    switch(menu){
      case1:KALIBRASI();break;
      case2:Tampil_Biner();break;
      case3:mapping();break;
      case4:ambilData();break;
      case5:Ultrasonik();break; }}
  switch(menu){
    case1:lcdPrint(0,1, "<   Kalibrasi  >");break;
    case2:lcdPrint(0,1, "<  Cek Sensor  >");break;
    case3:lcdPrint(0,1, "<     Mulai    >");break;
    case4:lcdPrint(0,1, "<  Ambil Data  >");break;
    case5:lcdPrint(0,1, "<  Ultrasonik  >");break;
  }}
```

PROGRAM FUNGSI

```
//Program Kalibrasi Sensor Garis
voidKALIBRASI(){
  Serial.println ("proses kalibrasi on!");
  lcd.clear();
  lcdPrint(0,0,"Mulai Kalibrasi");
```

```
  for(byte i=0; i<13; i++){
    Nilai_Max[i] = 0;
    Nilai_Min[i] = 1023;
  }
  delay(BOUNCE);
  while(true){
    delay(100);
    Baca_Sensor();
    for(byte i=0; i<13; i++){
      if(Sensor[i] >Nilai_Max[i]) Nilai_Max[i] = Sensor[i];
      if(Sensor[i] <Nilai_Min[i]) Nilai_Min[i] = Sensor[i];
    }
    delay(BOUNCE);
    if(Button(2) == LOW){
      while(Button(2) == LOW){}
      delay(BOUNCE);
      break;
    } }
  // for(byte i=0; i<13; i++){
  //    Nilai_Ref[i] = (Nilai_Max[i] + Nilai_Min[i])/2.1;
  //    EEPROM.write(i, Nilai_Ref[i]);
  // }
  delay(BOUNCE);
}
//Program Selektor Sensor di multiplexer
voidSelektor(bool a, bool b, bool c, bool d){
  digitalWrite(S0, a);
  digitalWrite(S1, b);
  digitalWrite(S2, c);
  digitalWrite(S3, d);
}
//Program Membaca Nilai Selektor Sensor di multiplexer
voidBaca_Sensor(){
  Selektor(1,0,1,1);
```

```
  Sensor[1] = analogRead(Sig)/4;
  Selektor(0,0,1,1);
  Sensor[2] = analogRead(Sig)/4;
  Selektor(1,1,0,1);
  Sensor[3] = analogRead(Sig)/4;
  Selektor(0,1,0,1);
  Sensor[4] = analogRead(Sig)/4;
  Selektor(1,0,0,1);
  Sensor[5] = analogRead(Sig)/4;
  Selektor(0,0,0,1);
  Sensor[6] = analogRead(Sig)/4;
  Selektor(1,1,1,0);
  Sensor[7] = analogRead(Sig)/4;
  Selektor(0,1,1,0);
  Sensor[8] = analogRead(Sig)/4;
  Selektor(1,0,1,0);
  Sensor[9] = analogRead(Sig)/4;
  Selektor(0,0,1,0);
  Sensor[10] = analogRead(Sig)/4;
  Selektor(1,1,0,0);
  Sensor[11] = analogRead(Sig)/4;
  Selektor(0,1,0,0);
  Sensor[12] = analogRead(Sig)/4;
}
//Program Membaca Nilai Push Button
boolButton(intNomor_Button){
  boolButton_Result;
  if(Nomor_Button == 1) Button_Result = digitalRead(Pb1);
  if(Nomor_Button == 2) Button_Result = digitalRead(Pb2);
  if(Nomor_Button == 3) Button_Result = digitalRead(Pb3);
  returnButton_Result;
}
//Program Menampilkan Nilai Biner dari Sensor Garis
voidTampil_Biner(){
```

```
  lcd.clear();
  while(true){
    Baca_Bit();
    Serial.print(Bit[1]);
    Serial.print(Bit[2]);
    Serial.print(Bit[3]);
    Serial.print(Bit[4]);
    Serial.print(Bit[5]);
    Serial.print(Bit[6]);
    Serial.print(Bit[7]);
    Serial.print(Bit[8]);
    Serial.print(Bit[9]);
    Serial.print(Bit[10]);
    Serial.print(Bit[11]);
    Serial.println(Bit[12]);
    lcdPrint(0,0,"   Cek Sensor   ");
    sprintf(buff,"                    %i%i%i%i%i%i%i%i%i%i%i
",Bit[1],Bit[2],Bit[3],Bit[4],Bit[5],Bit[6],Bit[7],Bit[8],Bi
t[9],Bit[10],Bit[11],Bit[12]);
    lcdPrint(0,1,buff);
     if(Button(2) == LOW){
      while(Button(2) == LOW){}
      delay(BOUNCE);
      break;
    } }}
//Program Menyimpan Data Kalibrasi Sensor Garis
voidBaca_Bit(){
  Baca_Sensor();
  if(Sensor[1] >Nilai_Ref[1]){
    Bit[1] = 1;
  }else{
    Bit[1] = 0;
  }
  if(Sensor[2] >Nilai_Ref[2]){
```

```
  Bit[2] = 1;
}else{
  Bit[2] = 0;
}
if(Sensor[3] >Nilai_Ref[3]){
  Bit[3] = 1;
}else{
  Bit[3] = 0;
}
if(Sensor[4] >Nilai_Ref[4]){
  Bit[4] = 1;
}else{
  Bit[4] = 0;
}
if(Sensor[5] >Nilai_Ref[5]){
  Bit[5] = 1;
}else{
  Bit[5] = 0;
}
if(Sensor[6] >Nilai_Ref[6]){
  Bit[6] = 1;
}else{
  Bit[6] = 0;
}
if(Sensor[7] >Nilai_Ref[7]){
  Bit[7] = 1;
}else{
  Bit[7] = 0;
}
if(Sensor[8] >Nilai_Ref[8]){
  Bit[8] = 1;
}else{
  Bit[8] = 0;
}
```

```
  if(Sensor[9] >Nilai_Ref[9]){
    Bit[9] = 1;
  }else{
    Bit[9] = 0;
  }
  if(Sensor[10] >Nilai_Ref[10]){
    Bit[10] = 1;
  }else{
    Bit[10] = 0;
  }
  if(Sensor[11] >Nilai_Ref[11]){
    Bit[11] = 1;
  }else{
    Bit[11] = 0;
  }
  if(Sensor[12] >Nilai_Ref[12]){
    Bit[12] = 1;
  }else{
    Bit[12] = 0;
  }}
//Program Fungsi “lcdPrint”
voidlcdPrint(uint8_t x, uint8_t y, char*msg){
  lcd.setCursor(x, y);
  lcd.print(msg);
}
//Program PID Kontrol
voidPiddd(){
    lcdPrint(0, 0, "     JALAN!     ");
    sprintf(buff,
"<<%i%i%i%i%i%i%i%i%i%i%i%i>>",Bit[1],Bit[2],Bit[3],

Bit[4],Bit[5],Bit[6],Bit[7],Bit[8],Bit[9],Bit[10],Bit[11],Bi
t[12]);
    lcdPrint(0, 1, buff);
```

```
    check_sensor();
    error = setpoint  - sensor;
    integral = error + integral;
    // //batas nilai integral
    if(integral >=15 ){
      integral = 15;
    }
    if(integral <=-15 ){
      integral = -15;
    }
    derivative = (error-errorTerakhir);
    //penghitung PID
    PID = (Kp * error + Kd * derivative + Ki * integral);
    errorTerakhir = error;
    if(PID >= 100){
      PID = 100;
    }
    if(PID <= -100){
      PID = -100;
    }
    totalkiri = leftspeed - (PID);
    totalkanan = rightspeed + PID;
    forward(totalkanan, totalkiri);
}
//Program Cek Sensor Untuk PID Kontrol
voidcheck_sensor(){
  Baca_Bit();
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
0&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 1) sensor
= 13;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
0&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 1&&Bit[10] == 1) sensor
= 11;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
```

```
0&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 1&&Bit[10] == 0) sensor
= 9;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
0&&Bit[7] == 0&&Bit[8] == 1&&Bit[9] == 1&&Bit[10] == 0) sensor
= 7;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
0&&Bit[7] == 0&&Bit[8] == 1&&Bit[9] == 0&&Bit[10] == 0) sensor
= 5;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
0&&Bit[7] == 1&&Bit[8] == 1&&Bit[9] == 0&&Bit[10] == 0) sensor
= 3;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
0&&Bit[7] == 1&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 0) sensor
= 1;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
1&&Bit[7] == 1&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 0) sensor
= 0;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
1&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 0) sensor
= -1;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  1&&Bit[6]  ==
1&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 0) sensor
= -3;
  if(Bit[3]  ==  0&&Bit[4]  ==  0&&Bit[5]  ==  1&&Bit[6]  ==
0&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 0) sensor
= -5;
  if(Bit[3]  ==  0&&Bit[4]  ==  1&&Bit[5]  ==  1&&Bit[6]  ==
0&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 0) sensor
= -7;
  if(Bit[3]  ==  0&&Bit[4]  ==  1&&Bit[5]  ==  0&&Bit[6]  ==
0&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 0) sensor
= -9;
  if(Bit[3]  ==  1&&Bit[4]  ==  1&&Bit[5]  ==  0&&Bit[6]  ==
0&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 0) sensor
```

```
= -11;
  if(Bit[3]  ==  1&&Bit[4]  ==  0&&Bit[5]  ==  0&&Bit[6]  ==
0&&Bit[7] == 0&&Bit[8] == 0&&Bit[9] == 0&&Bit[10] == 0) sensor
= -13;
  Serial.println(sensor);
}
//Program Menggerakkan Motor Sesuai Output PID
voidforward(int speed1, int speed2){
  if(speed1 <0){
    speed1 = 0;
  }
  if(speed1 >200){
    speed1 = 200;
  }
  if(speed2 <0){
    speed2 = 0;
  }
  if(speed2 >200){
    speed2 = 200;
  }
  //MOTOR KANAN
  digitalWrite(in1, LOW);
  analogWrite(in2, speed1);
  //MOTOR KIRI
  digitalWrite(in3, LOW);
  analogWrite(in4, speed2);
}
//Program Mengambil Data Koorinatdari Database Firebase
voidambilData(){
  lcd.clear();
  while(true){
    lcdPrint(0,0, "Ambil Data" );
    if (Firebase.ready() &&signupOK ) {
      if (Firebase.RTDB.getString(&fbdo, "TA/cx")) {
```

```
  if (fbdo.dataType() == "string") {
    cx = fbdo.stringData();
    CX = cx.toInt();
    sprintf(buff, "CX Berhasil: %i",CX);
    lcdPrint(0, 1, buff);
    delay(100);
  }
}
if (Firebase.RTDB.getString(&fbdo, "TA/cy")) {
  if (fbdo.dataType() == "string") {
    cy = fbdo.stringData();
    CY = cy.toInt();
    sprintf(buff, "CY Berhasil: %i",CY);
    lcdPrint(0, 1, buff);
    delay(100);
  }
}
if (Firebase.RTDB.getString(&fbdo, "TA/ox")) {
  if (fbdo.dataType() == "string") {
    ox = fbdo.stringData();
    OX = ox.toInt();
    lcdPrint(0,1, "OX Berhasil" );
    sprintf(buff, "OX Berhasil: %i",OX);
    lcdPrint(0, 1, buff);
    delay(100);
  }
}
if (Firebase.RTDB.getString(&fbdo, "TA/oy")) {
  if (fbdo.dataType() == "string") {
    oy = fbdo.stringData();
    OY = oy.toInt();
    sprintf(buff, "OY Berhasil: %i",OY);
    lcdPrint(0, 1, buff);
    delay(100);
```

```
        }
      }
      if (Firebase.RTDB.getString(&fbdo, "TA/dx")) {
        if (fbdo.dataType() == "string") {
          dx = fbdo.stringData();
          DX = dx.toInt();
          sprintf(buff, "DX Berhasil: %i",DX);
          lcdPrint(0, 1, buff);
          delay(100);
        }
      }
      if (Firebase.RTDB.getString(&fbdo, "TA/dy")) {
        if (fbdo.dataType() == "string") {
          dy = fbdo.stringData();
          DY = dy.toInt();
          sprintf(buff, "DY Berhasil: %i",DY);
          lcdPrint(0, 1, buff);
          delay(100);
        }
      }
      if(Button(2) == LOW){
        while(Button(2) == LOW){}
        delay(BOUNCE);
        break;
      }
      Serial.println(cx + ":" + cy + ":" + ox + ":" + oy +
":" + dx + ":" + dy);
    }}}
//Program Buzzer
voidbuzz(boolbunyi){
  digitalWrite(18, bunyi);
}
//Program Membaca Nilai Start dariAplikasi
voidSTART(){
```

```
while(true){
  if (Firebase.RTDB.getString(&fbdo, "TA/start")) {
    if (fbdo.dataType() == "string") {
      start = fbdo.stringData();
      Start = start.toInt();
    } }
  if(Start==1){
    planning();
  }else{
    break;
  }}
```

PROGRAM SENSOR ULTRASONIK

```
voidUltrasonik() {
  while(true){
//Membaca Nilai Sensor Ultrasonik
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    long duration = pulseIn(echoPin, HIGH);
    distance = (duration * 0.0343) / 2;
    //Menampilkanjarak pada Serial Monitor
    Serial.print("Jarak: ");
    Serial.print(distance);
    Serial.println(" cm");
    //Menampilkanhasilpengukuranjarakke lcd 16x2
    lcd.clear();
    lcdPrint(3,0, "Ultrasonik");
    sprintf(buff, "Jarak: %d cm", distance);
    lcdPrint(0,1,buff);
    delay(1000);
//Kondisi Untuk Keluar dari Menu Ultrasonik
    if(Button(2) == LOW){
```

```
    while(Button(2) == LOW){}
    delay(BOUNCE);
    break;
    }
//Kondisi Sensor Ultrasonik untuk Program Path Planning
    if(distance <= 5&&jarakKeBendaX == 0){
      berhenti();
      cengkram();
      angkat();
      break;
    }else{
      maju();
      berhenti();   }}}
```

PRORAM GRIPPER

```
//Fungsi Untuk Mencengkram Benda
voidcengkram() {
  while (posGrip != targetGrip) {
    if (posGrip<targetGrip) {
      posGrip += increment;
    } elseif (posGrip>targetGrip) {
      posGrip -= increment;
    }
    servoGrip.write(posGrip);
    delay(interval);
  }
}
//Fungsi Untuk Melepas Cengkraman Benda
voidlepas() {
  while (posGrip != targetRelease) {
    if (posGrip<targetRelease) {
      posGrip += increment;
    } elseif (posGrip>targetRelease) {
      posGrip - = increment;
    }
```

```
    servoGrip.write(posGrip);
    delay(interval);
  }
}
//Fungsi Untuk Mengangkat Benda
voidangkat() {
  while (posLift != targetLift) {
    if (posLift<targetLift) {
      posLift += incrementl;
    } elseif (posLift>targetLift) {
      posLift -= incrementl;
    }
    servoLift.write(posLift);
    delay(interval);
  }
}
//Fungsi Untuk Menurunkan Benda
voidturun() {
  while (posLift != targetDrop) {
    if (posLift<targetDrop) {
      posLift += incrementl;
    } elseif (posLift>targetDrop) {
      posLift -= incrementl;
    }
    servoLift.write(posLift);
     delay(interval);
  } }
```

PROGRAM PATH PLANNING

```
//ALGORITMA Path Planning
voidplanning() {
  // Menghitung jarak  dari  posisi awal keposisi benda dan
tujuan
  jarakKeBendaX = OX - CX;
  jarakKeBendaY = OY - CY;
```

```
  jarakKeTujuanX = DX - OX;
  jarakKeTujuanY = DY - OY;
  byte proses = 0;
// Menyimpan status putar balik untuk navigasi ke  benda
  boolsudahPutarBalikBenda = false;
// Menyimpan status putar balik untuk navigasi ketujuan
  boolsudahPutarBalikTujuan = false;
  while (true) {
    switch(proses) {
      case0:
        proses = navigasiMenujuBenda(jarakKeBendaY,
jarakKeBendaX, sudahPutarBalikBenda);
        break;
      case1:
        proses = angkatBenda(jarakKeBendaX);
        break;
      case2:
        proses     =     navigasiMenujuTujuan(jarakKeTujuanX,
jarakKeTujuanY, sudahPutarBalikTujuan);
        break;
      case3:
        proses = antarBenda(jarakKeTujuanY);
        break;
      case4:
        // Mengubah start kembalimenjadi 0
        if (Firebase.RTDB.setString(&fbdo, "TA/start", 0)) {
        Serial.println("Button state updated successfully");
        }else{
        Serial.println("Failed to update button state");
        Serial.println(fbdo.errorReason());
        }
        return;
        break; }}}
byte    navigasiMenujuBenda(int&jarakY,    intjarakKeBendaX,
```

```
bool&sudahPutarBalik) {
  if (jarakY<0&& !sudahPutarBalik) {
    putarBalik();
    sudahPutarBalik = true;
    berhenti();
    return0; // Tetap di proses inisetelahputarbalik
  }
  Piddd();
  if ((Bit[1] == 1&&Bit[2] == 1) || (Bit[11] == 1&&Bit[12] ==
1)) {
    if (jarakY>0) {
      jarakY--;
    } else {
      jarakY++;
    }
    buzz(1);
    delay(50);
  }
  buzz(0);
  if (jarakY == 0) {
    if (sudahPutarBalik) {
      // Jika sudahputarbalik, sesuaikanarahbelok
      if (jarakKeBendaX>0) {
        belokKiri();
      } elseif (jarakKeBendaX<0) {
        belokKanan();
      }
    } else {
      // Jika belum putar balik, tentukan  arah belok seperti
sebelumnya
      if (jarakKeBendaX>0) {
        belokKanan();
      } elseif (jarakKeBendaX<0) {
        belokKiri();
```

```
      }
    }
    berhenti();
    return1;  // Pindahke proses berikutnya
  }
  return0;  // Tetap di proses ini
}
byte angkatBenda(int&jarakX) {
  Piddd();
  if ((Bit[1] == 1&&Bit[2] == 1) || (Bit[11] == 1&&Bit[12] ==
1)) {
    if (jarakX>0) {
      jarakX--;
    } else {
      jarakX++;
    }
    buzz(1);
    delay(50);
  }
  buzz(0);

  if (jarakX == 0) {
    berhenti();
    Ultrasonik();
    return2;  // Pindahke proses berikutnya
  }
  return1;  // Tetap di proses ini
}
byte navigasi Menuju Tujuan (int & jarak X, int jarak Ke
Tujuan Y, bool & sudah Putar Balik) {
  if (jarakX<0&& !sudahPutarBalik) {
    putarBalik();
    sudahPutarBalik = true;
    berhenti();
```

```
    return2; // Tetap di proses ini setelah putar balik
  }
  Piddd();
  if ((Bit[1] == 1&&Bit[2] == 1) || (444444444444445tBit[11]
== 1&&Bit[12] == 1)) {
    if (jarakX>0) {
      jarakX--;
    } else {
      jarakX++;
    }
    buzz(1);
    delay(50);
  }
  buzz(0);
  if (jarakX == 0) {
    if (sudahPutarBalik) {
      // Jika sudah putar balik, sesuaikan arah belok
      if (jarakKeTujuanY>0) {
        belokKiri();
      } elseif (jarakKeTujuanY<0) {
        belokKanan();
      }
    } else {
      //Jika belum putar  balik, tentukan arah belok seperti
sebelumnya
      if (jarakKeTujuanY>0) {
        belokKiri();
      } elseif (jarakKeTujuanY<0) {
        belokKanan();
      }
    }
    berhenti();
    return3;  // Pindahke proses berikutnya
  }
```

```
  return2;  // Tetap di proses ini
}
byte antarBenda(int&jarakY) {
  Piddd();
  if ((Bit[1] == 1&&Bit[2] == 1) || (Bit[11] == 1&&Bit[12] ==
1)) {
    if (jarakY>0) {
      jarakY--;
    } else {
      jarakY++;
    }
    buzz(1);
    delay(50);
  }
  buzz(0);
  if (jarakY == 0) {
    berhenti();
    turun();
    lepas();
    mundur();
    berhenti();
    return4;  // Selesai
  }
  return3;  // Tetap di proses ini
}
```

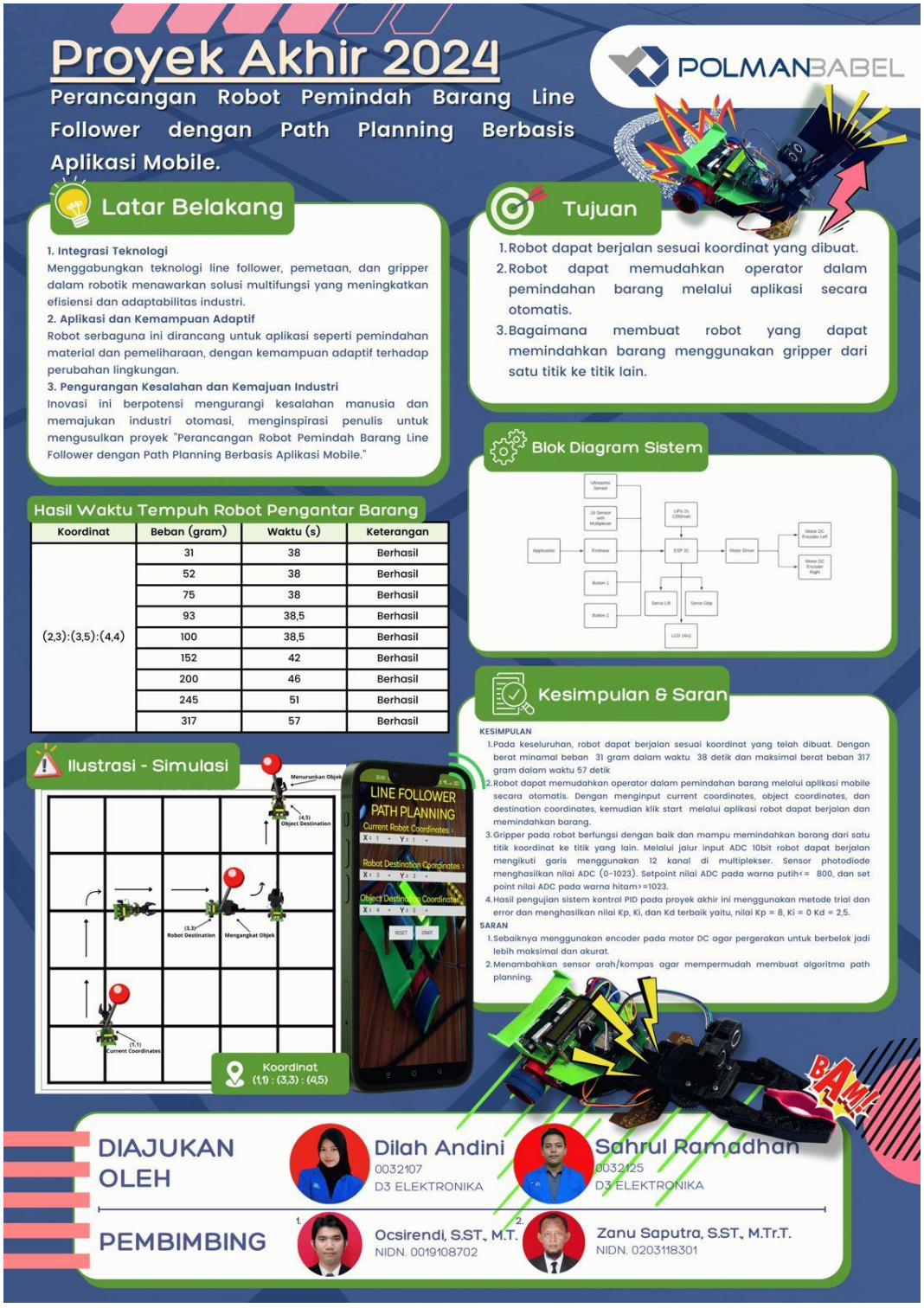
PROGRAM MOTOR

```
//Semua Fungsi Pergerakan Motor
voidbelokKanan(){
  while(true){
    analogWrite(in1, 50);
    analogWrite(in2, 0);
    analogWrite(in3, 0);
    analogWrite(in4, 200);
    delay(650);
```

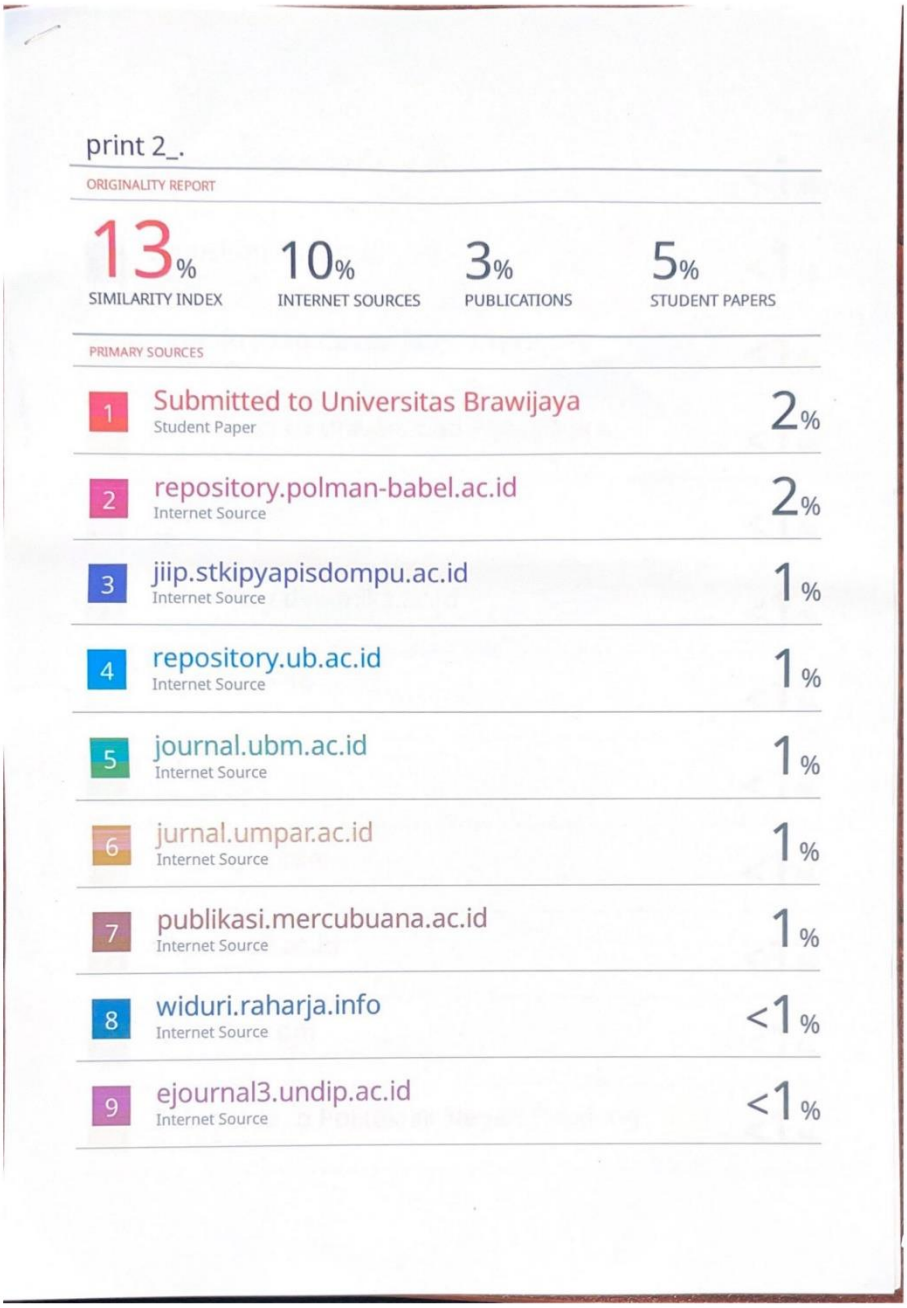
```
    if (Bit[6] == 1&&Bit[7] == 1){
    berhenti();
    break;
    }
  }
}
voidbelokKiri(){
  while(true){
    analogWrite(in1, 0);
    analogWrite(in2, 200);
    analogWrite(in3, 50);
    analogWrite(in4, 0);
    delay(650);
    if (Bit[6] == 1&&Bit[7] == 1){
    berhenti();
    break; }}}
voidputarBalik(){
  while(true){
    analogWrite(in1, 240);
    analogWrite(in2, 0);
    analogWrite(in3, 0);
    analogWrite(in4, 240);
    delay(1000);
    if (Bit[6] == 1&&Bit[7] == 1){
    berhenti();
    break;
    }}}
voidmaju(){
  analogWrite(in1, 0);
  analogWrite(in2, 150);
  analogWrite(in3, 0);
  analogWrite(in4, 150);
  delay(200);
}
```

```
voidmundur(){
  analogWrite(in1, 200);
  analogWrite(in2, 0);
  analogWrite(in3, 200);
  analogWrite(in4, 0);
  delay(500);
}
voidberhenti(){
  analogWrite(in1, 0);
  analogWrite(in2, 0);
  analogWrite(in3, 0);
  analogWrite(in4, 0);
  delay(500);
}
```

POSTER


Proyek Akhir 2024
Perancangan Robot Pemindah Barang Line Follower dengan Path Planning Berbasis Aplikasi Mobile.
POLMANBABEL
Latar Belakang
1. Integrasi Teknologi
Menggabungkan teknologi line follower, pemetaan, dan gripper dalam robotik menawarkan solusi multifungsi yang meningkatkan efisiensi dan adaptabilitas industri.
2. Aplikasi dan Kemampuan Adaptif
Robot serbaguna ini dirancang untuk aplikasi seperti pemindahan material dan pemeliharaan, dengan kemampuan adaptif terhadap perubahan lingkungan.
3. Pengurangan Kesalahan dan Kemajuan Industri
Inovasi ini berpotensi mengurangi kesalahan manusia dan memajukan industri otomasi, menginspirasi penulis untuk mengusulkan proyek "Perancangan Robot Pemindah Barang Line Follower dengan Path Planning Berbasis Aplikasi Mobile."
Tujuan
1. Robot dapat berjalan sesuai koordinat yang dibuat.
2. Robot dapat memudahkan operator dalam pemindahan barang melalui aplikasi secara otomatis.
3. Bagaimana membuat robot yang dapat memindahkan barang menggunakan gripper dari satu titik ke titik lain.
Blok Diagram Sistem
Hasil Waktu Tempuh Robot Pengantar Barang
Koordinat | Beban (gram) | Waktu (s) | Keterangan
(2,3):(3,5):(4,4) | 31 | 38 | Berhasil
52 | 38 | Berhasil
75 | 38 | Berhasil
93 | 38,5 | Berhasil
100 | 38,5 | Berhasil
152 | 42 | Berhasil
200 | 46 | Berhasil
245 | 51 | Berhasil
317 | 57 | Berhasil
Kesimpulan & Saran
KESIMPULAN
1. Pada keseluruhan, robot dapat berjalan sesuai koordinat yang telah dibuat. Dengan berat minamal beban 31 gram dalam waktu 38 detik dan maksimal berat beban 317 gram dalam waktu 57 detik
2. Robot dapat memudahkan operator dalam pemindahan barang melalui aplikasi mobile secara otomatis. Dengan menginput current coordinates, object coordinates, dan destination coordinates, kemudian klik start melalui aplikasi robot dapat berjalan dan memindahkan barang.
3. Gripper pada robot berfungsi dengan baik dan mampu memindahkan barang dari satu titik koordinat ke titik yang lain. Melalui jalur input ADC 10bit robot dapat berjalan mengikuti garis menggunakan 12 kanal di multiplekser. Sensor photodiode menghasilkan nilai ADC (0-1023). Setpoint nilai ADC pada warna putih<= 800, dan set point nilai ADC pada warna hitam>=1023.
4. Hasil pengujian sistem kontrol PID pada proyek akhir ini menggunakan metode trial dan error dan menghasilkan nilai Kp, Ki, dan Kd terbaik yaitu, nilai Kp = 8, Ki = 0 Kd = 2,5.
SARAN
1. Sebaiknya menggunakan encoder pada motor DC agar pergerakan untuk berbelok jadi lebih maksimal dan akurat.
2. Menambahkan sensor arah/kompas agar mempermudah membuat algoritma path planning.
Ilustrasi - Simulasi
LINE FOLLOWER PATH PLANNING
Koordinat (1,1) : (3,3) : (4,5)
BAM!
DIAJUKAN OLEH
Dilah Andini
0032107
D3 ELEKTRONIKA
Sahrul Ramadhan
0032125
D3 ELEKTRONIKA
PEMBIMBING
1. Ocsirendi, S.ST., M.T.
NIDN. 0019108702
2. Zanu Saputra, S.ST., M.Tr.T.
NIDN. 0203118301

PLAGIARISME



print 2_.

ORIGINALITY REPORT

13%	10%	3%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universitas Brawijaya Student Paper	2%
2	repository.polman-babel.ac.id Internet Source	2%
3	jiip.stkipyapisdompu.ac.id Internet Source	1%
4	repository.ub.ac.id Internet Source	1%
5	journal.ubm.ac.id Internet Source	1%
6	jurnal.umpar.ac.id Internet Source	1%
7	publikasi.mercubuana.ac.id Internet Source	1%
8	widuri.raharja.info Internet Source	<1%
9	ejournal3.undip.ac.id Internet Source	<1%

SURAT PERNYATAAN

Lampiran Nomor : 034/PROYEKAKHIR/DIII/2024

SURAT PERNYATAAN

Saya/Kami yang bertandatangan dibawah ini telah menyelesaikan Proyek Akhir yang berjudul:

Perancangan Robot Pemindah Barang Line Follower Dengan PATH PLANNING BERBASIS APLIKASI MOBILE

Oleh :

1. Dilah Andini /NPM 0032107
2. Sahrul Ramadhan /NPM 0032125

Dengan ini menyatakan bahwa isi laporan akhir proyek akhir sama dengan *hardcopy*.

Demikian surat pernyataan ini dibuat dengan sebenar-benarnya.

Sungailiat, 8 Agustus 2024

1. Dilah Andini (.....)
2. Sahrul Ramadhan (.....)

Mengetahui,

Pembimbing 1,	Pembimbing 2,
(.....)	(.....)
Ocsirendi, M.T	Zanu Saputra, S.ST., M.Tr.T.

FORM REVISI

FORM-PPR-3- 8: Form Revisi Laporan Akhir

FORM REVISI LAPORAN AKHIR
TAHUN AKADEMIK
...../.....

JUDUL : Perancangan Robot Pemindah Barang Line Follower Dengan Path Planning Berbasis Aplikasi Mobile

Nama Mahasiswa :
1. Dilah Andini NIM: 0032107
2. Sahrul Ramadhan NIM: 0032105
3. _____ NIM: _____
4. _____ NIM: _____
5. _____ NIM: _____

Bagian yang direvisi	Halaman
- Penulisan font utk kata asing	
- spasi antar paragraf sebaiknya tetap 1,5	

Sungailiat, 16 juli 2024
Penguji
(Zamri. S)

Menyatakan telah menyetujui revisi laporan akhir yang telah dilakukan oleh mahasiswa

Mengetahui,
Pembimbing
(Okta Fauzi)

Sungailiat, 22-07-2024
Penguji
(Zamri. S)

FORM-PPR-3- 8: Form Revisi Laporan Akhir

FORM REVISI LAPORAN AKHIR
TAHUN AKADEMIK
2023 / 2024

JUDUL : Perancangan Robot pemindah barang line follower dengan path planning berbasis aplikasi mobile.

Nama Mahasiswa :
1. Ricoh Andri NIM: _____
2. Sahrul Ramadhan NIM: _____
3. _____ NIM: _____
4. _____ NIM: _____
5. _____ NIM: _____

Bagian yang direvisi	Halaman
⇒ Alat → Sensor dipasang penutup agar di uji coba lagi untuk data ulang.	
⇒ makalah.	

Sungailiat, 16 Juli 2024
Penguji
(Indra Darmaputra)

Menyatakan telah menyetujui revisi laporan akhir yang telah dilakukan oleh mahasiswa

Mengetahui, Pembimbing	Sungailiat, 23 Juli 2024 Penguji
( )	(Indra Darmaputra)

FORM-PPR-3- 8: Form Revisi Laporan Akhir

FORM REVISI LAPORAN AKHIR
TAHUN AKADEMIK
23 / 24

JUDUL : Perancangan Robot Pemindah Barang Line Follower dengan path Planning Berbasis Mobile

Nama Mahasiswa :
1. Dilah Andini NIM:
2. Fahrul Ramadhan NIM:
3. NIM:
4. NIM:
5. NIM:

Bagian yang direvisi	Halaman
- Perbaiki alat agar tidak error	
- Kata serapan huruf miring	
- Laporan P. akhir diraplikan	

Sungailiat, 16 Juli 24

Penguji

(Yang Agha R.)

Menyatakan telah menyetujui revisi laporan akhir yang telah dilakukan oleh mahasiswa

Mengetahui, Pembimbing	Sungailiat, ... Penguji
(.....)	(.....)