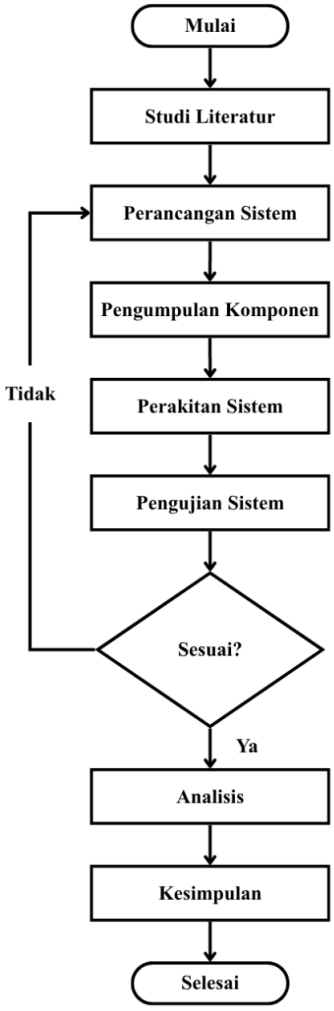
BAB III
METODE PENELITIAN

3.1 *Flowchart* Penelitian

Perancangan sistem ini dilakukan melalui sejumlah tahapan yang digambarkan pada *flowchart* di bawah ini. Setiap tahapan dalam proses tersebut disusun secara terstruktur dan simetris guna memastikan kelancaran serta keberhasilan dalam proses pengembangan sistem. Gambar 3.1 menjelaskan *flowchart* penelitian yang menjadi alur perancangan sistem dalam pengendalian *Mobile robot omni wheel*.



Gambar 3. 1 *Flowchart* Penelitian

3.1.1 Studi Literatur

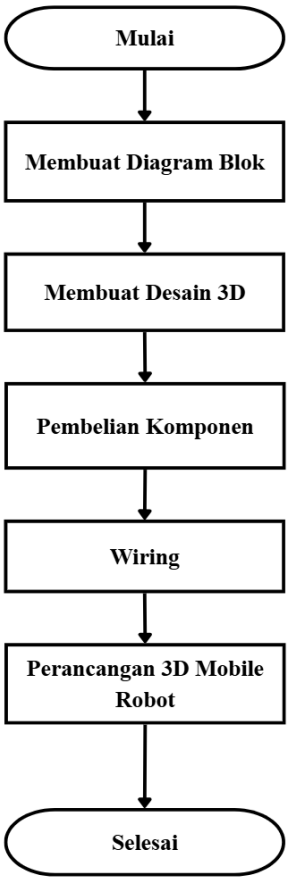
Studi literatur dilakukan untuk mengumpulkan berbagai sumber referensi yang relevan dengan penelitian yang akan dilakukan. Referensi tersebut dapat berupa jurnal, artikel ilmiah, buku, maupun sumber informasi lainnya yang mendukung proses pengembangan penelitian dan penyusunan tugas akhir.

3.1.2 Perancangan Sistem

Perancangan sistem bertujuan untuk menentukan kebutuhan yang diperlukan dalam proses perakitan *mobile robot*, meliputi aspek perangkat keras maupun perangkat lunak. Perancangan perangkat keras mencakup pilihan dan penyusunan komponen seperti mikrokontroler ESP32, driver motor, motor DC dengan encoder, sensor Ultrasonic, serta komponen pendukung lainnya yang berperan dalam pengoperasian robot. Sementara itu, perancangan perangkat lunak meliputi penggunaan berbagai aplikasi dan platform pemrograman untuk merancang serta mengimplementasikan sistem kendali pada mikrokontroler ESP32 yang berfungsi mengatur kerja seluruh robot.

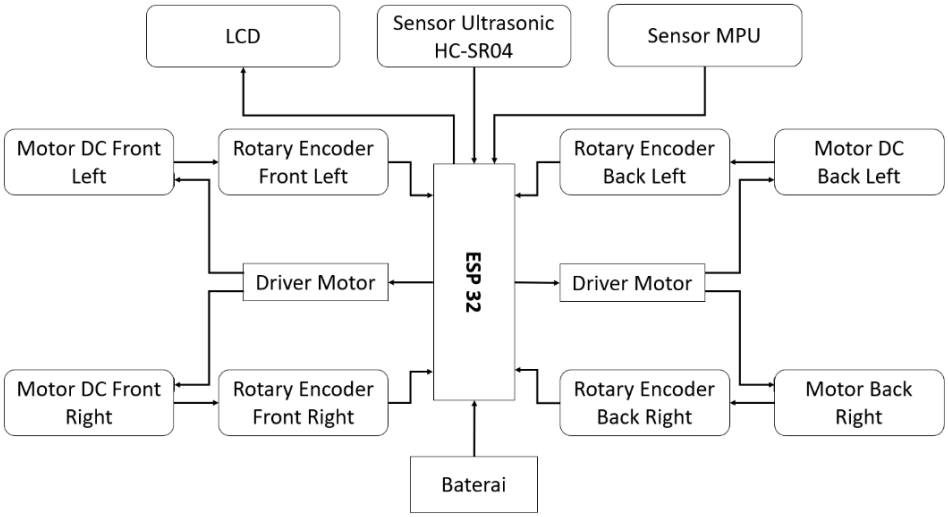
1. Perancangan Perangkat Keras

   Perancangan perangkat keras yang ditunjukkan pada Gambar 3.2 menunjukkan tahapan proses perancangan *hardware* pada *mobile* robot yang akan dibuat, dimulai dari pembuatan diagram sistem, perancangan desain 3D, penyiapan komponen, penyusunan jalur pengkabelan (*wiring*), sampai dengan tahap akhir berupa perakitan *mobile* robot secara keseluruhan.



Gambar 3. 2 Perancangan Perangkat Keras

Dari proses perancangan keras diatas tahapan pertama yaitu pembuatan diagram blok untuk menggambarkan hubungan antara komponen utama dalam sistem *mobile robot* yang ditunjukkan pada Gambar 3.3.



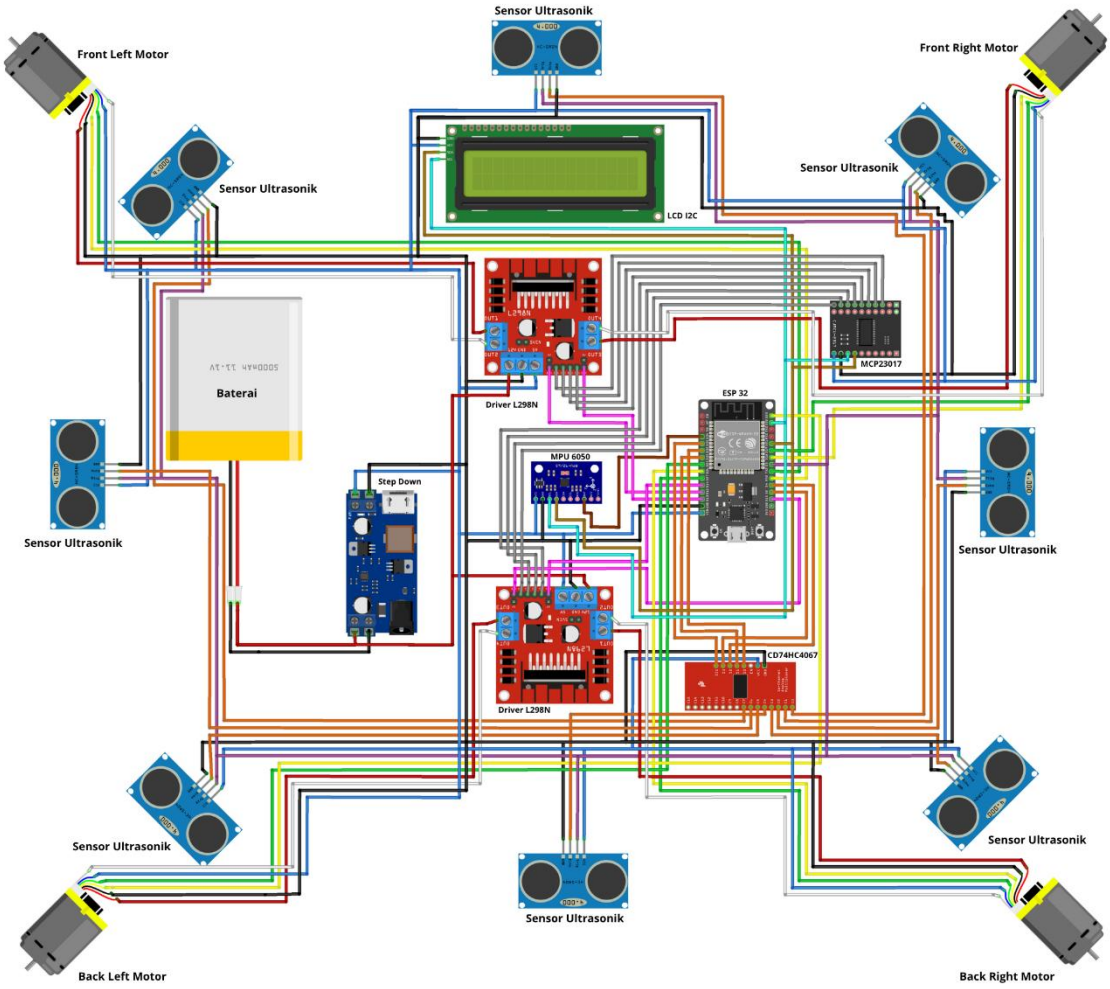
Gambar 3. 3 Diagram blok sistem

Pada Gambar 3.3 menunjukan hubungan kerja sistem ESP32 yang memperoleh suplai daya baterai dan berperan sebagai pusat pengendalian utama. Sensor ultrasonik HC-SR04 yang berperan dalam mendeteksi adanya halangan dan sensor *rotary encoder* yang menghitung kecepatan putaran pada masing-masing roda. Data ini digunakan sebagai umpan balik dalam pengendalian kecepatan motor, serta penerapan metode *Artificial Potential Field* (APF) untuk sistem menghindari rintangan dengan menggunakan kontrol PID. Proses perhitungan kinematika dilakukan untuk menghasilkan kecepatan yang diperlukan pada tiap roda yang selanjutnya dikontrol oleh motor *driver* untuk menggerakkan motor DC.

Setelah memahami struktur dari diagram blok, langkah berikutnya adalah membuat rancangan 3D dengan menggunakan *software Autodesk Fusion* 360 yang berfungsi sebagai visualisasi awal sebelum dilakukan pembuatan desain sebenarnya, ketika seluruh dimensi yang diperlukan telah ditentukan. Pada Gambar 3.4 menunjukkan *design* 3D *mobile* robot.

Gambar 3. 4 Design 3D *mobile* robot

Selanjutnya setelah mengetahui struktur pada diagram blok dan membuat desain, maka akan dibuat perancangan elektronika robot (*wiring*).



Gambar 3. 5 *Wiring* diagram sistem

Gambar 3.5 merupakan *wiring* diagram sistem dengan komponen empat motor DC *encoder*, dua *driver* motor L298N, sensor ultrasonik HC-SR04, LCD I2C, MCP23017, CD74HC4067 dan ESP32 DevKit V1. Dalam pengaplikasiannya sistem *wiring* ini membutuhkan suplai tegangan berupa baterai 12V untuk *driver* motor dan 5V untuk ESP32, sehingga ESP32 dapat memberikan perintah kepada sensor *rotary encoder* motor untuk mengatur kecepatan *mobile robot* nantinya. Untuk mempermudah proses perancangan, perakitan, serta pengujian sistem, setiap jalur koneksi direpresentasikan

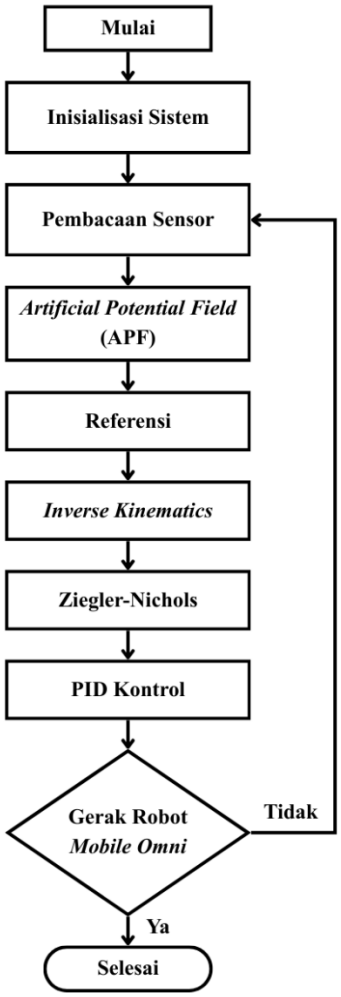
menggunakan kode warna kabel dengan fungsi yang berbeda-beda. Tabel 3.1 menunjukkan rincian mengenai fungsi masing-masing warna kabel dan komponen yang terhubung dari Gambar 3.5.

Tabel 3. 1 Keterangan Warna Kabel pada Sistem Robot *Omni Wheel*

No	Warna Kabel	Fungsi Utama	Komponen yang Terhubung
1	Biru	5 V	Semua Komponen
2	Merah	12 V & Motor (+)	L298N, Motor *Encoder*, *Step Down* & Baterai
3	Hitam	GND	Semua Komponen
4	Oren	ECHO	Ultrasonik, CD74HC4067 & ESP32
5	Ungu	TRIG	Ultrasonik & ESP32
6	Putih	Motor (-)	L298N & Motor Encoder
7	Hijau	Encoder A	Motor Encoder & ESP32
8	Kuning	Encoder B	Motor Encoder & ESP32
9	Cyan	SCL	MPU 6050, LCD I2C & MCP23017
10	Oker	SDA	MPU 6050, LCD I2C & MCP23017
11	Cokelat	INT	MPU 6050 & ESP32
12	Abu-Abu	IN1, IN2, IN3, IN4	L298N & MCP23017
13	Merah Muda	ENA & ENB	L298N & ESP32

2. Perancangan Perangkat Lunak

Setelah perancangan perangkat keras telah selesai dilanjutkan dengan tahap pemrograman Arduino sesuai dengan kebutuhan yang terdapat pada gambar 3.6 yaitu pada *flowchart* perangkat lunak.



Gambar 3. 6 *Flowchart* perancangan perangkat lunak

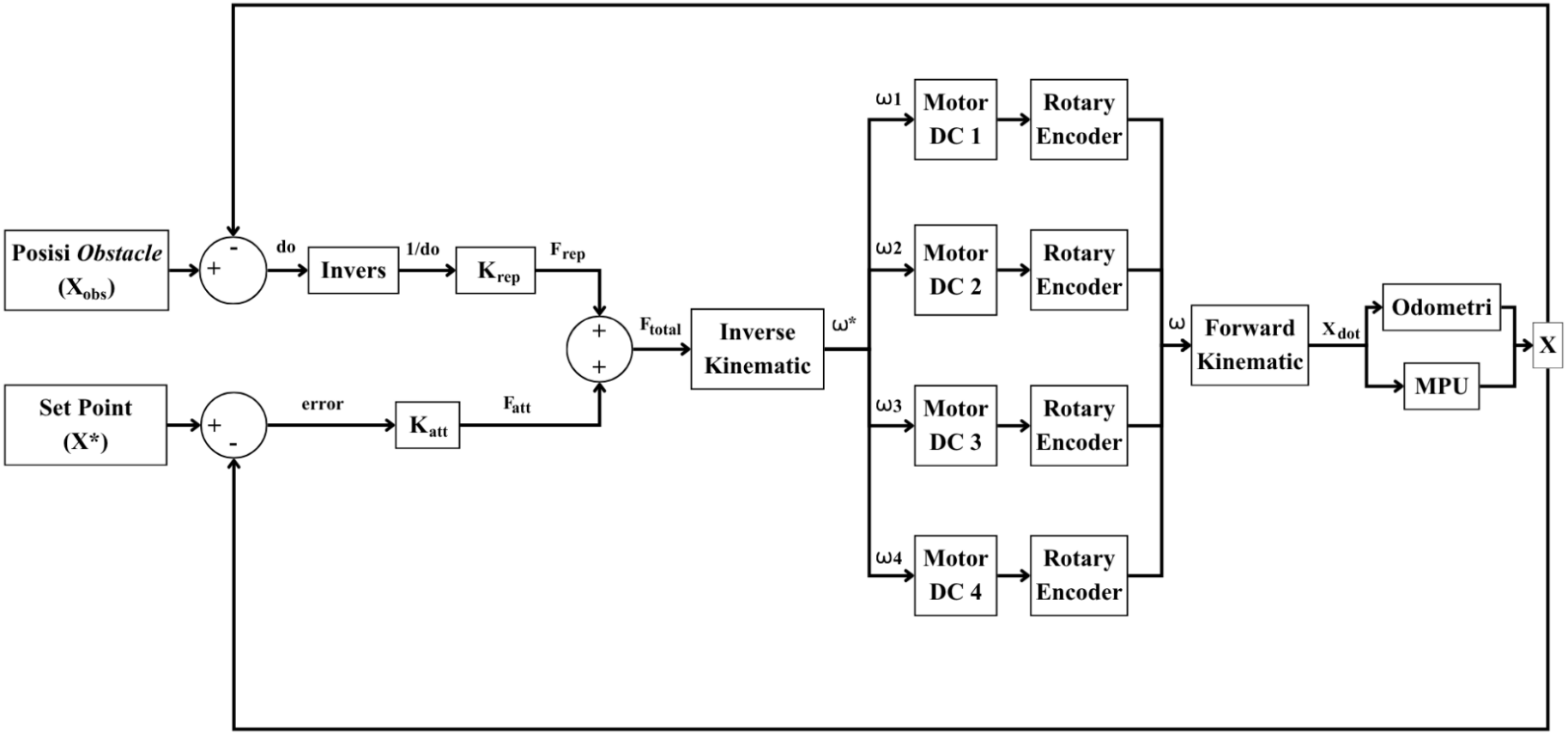
Gambar 3.6 menunjukkan alur perancangan perangkat lunak *robot mobile omni* dengan metode *Artificial Potential Field* (APF) dan kontrol PID menggambarkan alur kerja sistem mulai dari inisialisasi hingga robot mencapai target. Proses diawali dengan inisialisasi sistem, meliputi

pengaturan pin *input*/*output*, sensor ultrasonik, motor *encoder*, serta konstanta APF (k_{att}, k_{rep}) dan PID (Kp, Ki, Kd). Selanjutnya, sistem melakukan pembacaan sensor untuk memperoleh data jarak dari sensor ultrasonik dan kecepatan roda dari *encoder*. Data tersebut digunakan pada tahap perhitungan APF, di mana sistem menghitung gaya tarik menuju target dan gaya tolak dari rintangan untuk menghasilkan vektor resultan arah gerak robot. Vektor ini dikonversi menjadi kecepatan referensi (vx, vy) yang kemudian diolah melalui inverse *kinematics* untuk menentukan kecepatan referensi tiap roda omni.

Tahap berikutnya adalah pengendalian PID, yang membandingkan kecepatan referensi dengan kecepatan aktual dari *encoder*. Sebelum parameter PID digunakan dalam sistem, dilakukan proses penalaan menggunakan metode *Ziegler-Nichols* untuk memperoleh nilai awal konstanta Kp, Ki, dan Kd yang sesuai dengan karakteristik motor. Nilai hasil penalaan tersebut kemudian digunakan sebagai parameter kontrol pada sistem. Hasil perhitungan PID menghasilkan sinyal PWM untuk mengatur kecepatan motor melalui driver. Robot kemudian bergerak mengikuti hasil kendali PID, sementara data sensor terus diperbarui. Sistem melakukan pemeriksaan kondisi tujuan. Jika robot belum mencapai target, proses kembali ke tahap pembacaan sensor; jika sudah, sistem menghentikan pergerakan dan proses berakhir. Secara keseluruhan, *flowchart* ini menunjukkan integrasi metode APF untuk penentuan arah gerak dan PID untuk stabilisasi kecepatan, sehingga robot omni mampu bergerak halus, stabil, dan menghindari rintangan menuju target.

3. Metode Pengendalian

   a. Sistem Pengendalian APF Tanpa PID



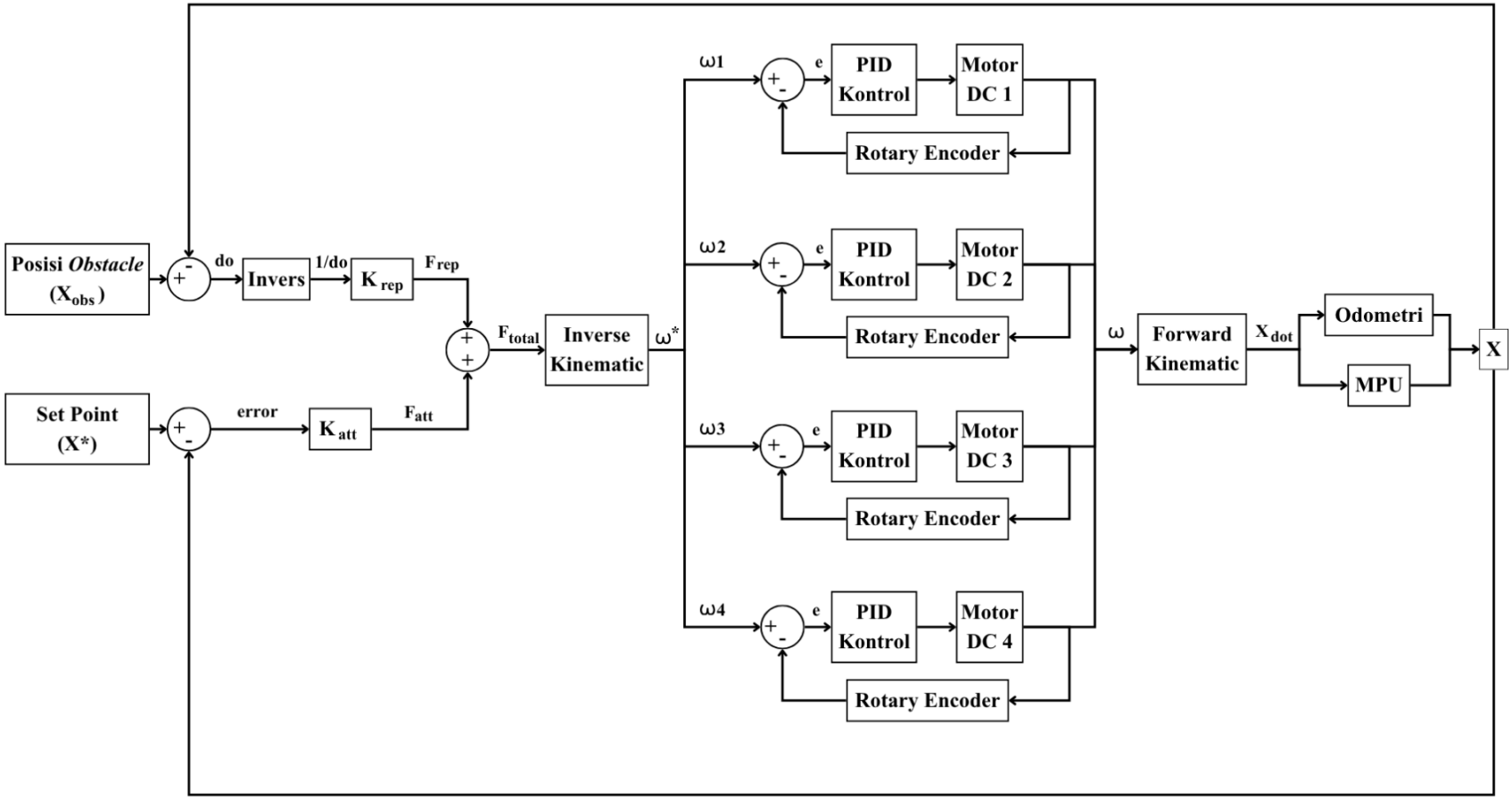
Gambar 3. 7 Blok Diagram Sistem Pengendalian APF Tanpa PID

Pada Gambar 3.7 menjelaskan blok diagram pengendalian dengan sistem APF murni tanpa kontrol PID yang dimulai dengan logika navigasi *outer-loop*, yaitu *Artificial Potential Field* (APF). Blok ini menentukan arah gerak robot dengan menyeimbangkan dua gaya, yaitu gaya tarik (*attractive*) dan gaya tolak (*repulsive*). Secara bersamaan, gaya tolak (F_{rep}) dihitung berdasarkan jarak (do) antara posisi obstacle (X_{obs}) yang diperoleh dari sensor dan posisi aktual robot. Kedua gaya tersebut dijumlahkan untuk menghasilkan F_{total}, yang merupakan vektor perintah gerak berupa arah dan kecepatan robot.

Vektor total F_{total} hasil dari APF digunakan sebagai masukan ke blok *inverse kinematics* untuk dikonversi menjadi kecepatan masing-masing roda (ω_1 hingga ω_4). Kecepatan tersebut kemudian langsung diberikan ke motor DC tanpa umpan balik, sehingga sistem pengendalian motor bekerja secara *open-loop* tanpa koreksi *error*.

Bagian akhir merupakan proses estimasi posisi robot yang menghasilkan sinyal posisi aktual (X). Proses ini dimulai dengan membaca kecepatan putar roda dari *rotary encoder*, yang kemudian diproses melalui *forward kinematics* untuk memperoleh kecepatan gerak robot ($\dot{X}$). Kecepatan tersebut diintegrasikan untuk mendapatkan posisi (x, y) melalui proses *odometri*. Selain itu, sensor MPU6050 digunakan untuk mengukur orientasi robot (θ). Hasil penggabungan posisi (x, y) dan orientasi (θ) membentuk posisi aktual robot (X), yang selanjutnya digunakan kembali sebagai masukan pada algoritma APF untuk perhitungan arah gerak pada iterasi berikutnya.

b. Sistem Pengendalian APF dan PID



Gambar 3. 8 Blok Diagram Sistem Pengendalian APF dan PID

Pada gambar 3.8 menjelaskan blok diagram pengendalian dengan sistem kontrol *loop* tertutup (*closed-loop*) yang dimulai dengan logika navigasi *outer-loop*, yaitu *Artificial Potential Field* (APF). Blok ini menentukan arah gerak robot dengan menyeimbangkan dua gaya, yaitu gaya tarik (*attractive*) dan gaya tolak (*repulsive*). Pertama, gaya tarik (F_{att}) dihasilkan dengan menghitung selisih (*error*) antara Set Point (X*) (tujuan akhir) dan Posisi Aktual (X) (umpan balik posisi robot saat ini), yang kemudian dikalikan dengan konstanta K_{att}. Secara bersamaan, Gaya Tolak (F_{rep}) dihitung berdasarkan jarak (do) antara posisi *obstacle* (X_{obs}) atau HC-SR04 dan Posisi Aktual (X). Jarak ini diproses (melalui *Invers* dan K_{rep}) untuk menciptakan gaya yang mendorong robot menjauh dari halangan. Kedua gaya ini kemudian dijumlahkan untuk menghasilkan F_{total}, yang merupakan vektor perintah gerak bersih (kecepatan dan arah) yang diinginkan untuk robot.

Vektor F_{total} dari APF kemudian menjadi masukan untuk blok *Invers Kinematic*. Blok ini berfungsi sebagai penerjemah, yang mengurai perintah gerak robot (F_{total}) menjadi empat *setpoint* kecepatan individual (Setpoint ω_1 hingga ω_4), satu untuk setiap motor DC. Di sinilah *inner loop* (loop kontrol internal) bekerja. Sistem ini terdiri dari empat loop PID *Controller* paralel yang identik, di mana setiap loop bertanggung jawab atas satu motor. Dalam setiap *loop*, *setpoint* ω (kecepatan yang diinginkan) dibanding dengan kecepatan putar aktual motor (umpan balik dari *rotary encoder*). Selisih (e) dari perbandingan ini menjadi masukan bagi PID kontrol, yang kemudian mengatur daya ke motor DC untuk memastikan kecepatan putar roda secara sesuai dengan perintah *setpoint* dari *Inverse Kinematics*.

Bagian akhir adalah *loop* umpan balik utama yang menghasilkan sinyal posisi aktual (X). Proses ini dimulai dengan mengambil data kecepatan aktual (ω) dari keempat *Rotary Encoder*. Keempat sinyal ini dimasukkan ke blok *Forward Kinematics*, yang menghitung kecepatan gerak badan robot yang sebenarnya (X_{dot}). Kecepatan ini kemudian diakumulasi seiring waktu oleh blok proses *odometri* untuk menghasilkan estimasi posisi robot di peta, yaitu posisi (x, y). Secara paralel, sensor MPU6050 secara independen mengukur orientasi atau arah hadap robot, yaitu orientasi (θ). Akhirnya, blok posisi aktual (X) berfungsi sebagai titik sensor, yang menggabungkan hasil estimasi (x, y) dari *Odometri* dengan hasil pengukuran (θ) dari MPU. Sinyal x = (x, y, θ) yang lengkap inilah yang menjadi umpan balik (*feedback*) krusial untuk seluruh sistem, yang dikirim kembali ke blok APF untuk perhitungan *error* dan do pada iterasi berikutnya.

3.1.3 Pengumpulan Komponen

Setelah tahapan perancangan *hardware* dan *software* dianggap selesai, dilanjutkan dengan pengumpulan komponen sekaligus dilakukan pengujian setiap komponen untuk mengetahui bahwa dapat berfungsi dengan baik. Beberapa komponen yang diperlukan dalam perakitan sistem pada Tabel 3.2.

Tabel 3. 2 Komponen yang diperlukan dalam perakitan

No.	Hardware dan Software	Fungsi
1.	Laptop/PC	*Hardware* untuk memprogram arduino
2.	ESP32	Prosesor utama
3.	Arduino IDE	Aplikasi program arduino

No.	Hardware dan Software	Fungsi
4.	Kabel USB	Konektor Arduino ke PC
5.	Baterai	Pemasok daya pada Arduino dan motor *driver*
6.	*Driver* Motor L289N	Sebagai pengendali motor
7.	Sensor Ultrasonik	Untuk mendeteksi adanya halangan atau rintangan
8.	LCD	Sebagai tampilan robot
9.	Motor JGA25 – 370	Penghasil putaran untuk roda dengan sensor *encoder*
10.	Sensor MPU6050	Untuk membantu robot bernavigasi
11.	Kabel	Sebagai konektor antar komponen
12.	Saklar	Sebagai pemutus daya baterai
13.	MCP23017	Sebagai penambah pin *input/output* (GPIO)
14.	CD74HC4067	Sebagai saluran *input/output* ke satu pin sinyal analog/digital.
15.	Akrilik	Sebagai bahan bodi
16.	Benner	Sebagai bahan arena pengujian

3.1.4 Perakitan Sistem

Perakitan sistem dilakukan setelah semua tahapan perancangan selesai dan semua komponen terkumpul dan layak digunakan. Perakitan sistem dilaksanakan sesuai dengan sistem yang telah dirancang sebelumnya.

3.1.5 Pengujian Sistem

Sistem yang telah selesai dirancang selanjutnya akan melalui tahap pengujian menyeluruh untuk memastikan apakah kinerjanya sudah sesuai dengan rancangan dan berfungsi sebagaimana yang diharapkan. Proses pengujian dilakukan dengan cara mengamati secara langsung pergerakan *mobile robot* yang telah dirakit, guna menilai tingkat keberhasilan sistem berdasarkan beberapa indikator capaian yang telah ditetapkan. Pengujian dibagi menjadi tiga tahap, yaitu pengujian *outer-loop controller* (APF), pengujian *inner-loop controller* (PID), dan pengujian integrasi sistem.

1. Pengujian Unit
    a. Pengujian sensor Ultrasonik HC-SR04 dan CD74HC4067

        Pengujian ini bertujuan untuk memvalidasi akurasi sistem persepsi robot yang menggunakan delapan sensor ultrasonik HC-SR04 dengan manajemen sinyal nya berbasis *multiplexer* CD74HC4067. Pengujian ini berfokus pada verifikasi akurasi pembacaan jarak pada delapan kanal (C0 – C7) dan efisiensi waktu pemindaian lingkungan 360 derajat untuk memastikan redudansi data rintangan mendukung *algoritma Obstacle Avoidance*.

        Prosedur pengujian dilakukan dengan menempatkan robot pada posisi statis di atas permukaan rata. Objek referensi berupa bidang datar berukuran 15 x 15 cm diletakkan di depan sensor yang sedang diuji dengan jarak referensi tetap sebesar 20 cm. Mikrokontroler ESP32 diprogram untuk melakukan *switching* kanal berurutan (*sequential polling*) melalui pin kontrol S0, S1, S2, dan S3 pada *multiplexer*. Jarak

fisik dihitung berdasarkan durasi waktu pantulan gelombang (*echo*) menggunakan persamaan (3.1).

$$d = \frac{t \times 0{,}034}{2} \tag{3.1}$$

Dimana d adalah jarak (cm), t adalah durasi waktu sinyal (μs), dan 0,034 adalah kontsanta kecepatan suara di udara (cm/μs).

Selain akurasi, analisis *sampling rate* juga dilakukan untuk memastikan sistem mampu mendeteksi rintangan secara *real-time*. Berdasarkan batasan fisik sensor, untuk mendeteksi rintangan maksimal pada jarak 25 dan 30 cm dengan *safety margin*, estimasi waktu siklus pemindaian total (T_{cycle}) untuk 8 sensor dihitung menggunakan persamaan (3.2).

$$T_{cycle} = Jumlah\ Sensor \times t \tag{3.2}$$

Hasil pengukuran jarak dan persentase *error* pada setiap kanal sensor dicatat menggunakan format pada Tabel 3.3, sedangkan analisis latensi sistem akan dicatat pada Tabel 3.4.

Tabel 3. 3 Rancangan Data Uji Akurasi dan Validasi Kanal

No	**Kana (Mux)**	**Sudut Posisi**	**Posisi Relatif**	**Jarak Ukur (cm)**	**Error (%)**
1	C0	0°	Depan	...	...
2	C1	45°	Depan-Kanan	...	...
3	C2	90°	Kanan	...	...
4	C3	135°	Belakang-Kanan	...	...
5	C4	180°	Belakang	...	...
6	C5	225°	Belakang-Kiri	...	...

No	Kana (Mux)	Sudut Posisi	Posisi Relatif	Jarak Ukur (cm)	Error (%)
7	C6	270°	Kiri	...	...
8	C7	315°	Depan-Kiri	...	...

Selain pengujian akurasi pembacaan jarak pada setiap kanal, dilakukan pula pengujian untuk menganalisis *latensi* sistem dalam mendeteksi objek pada berbagai jarak. Rancangan data pengujian *latensi* tersebut disajikan pada Tabel 3.4.

Tabel 3. 4 Rancangan Data Analisis *Latensi*

No	Jarak Target (cm)	Waktu Teoritis (ms)	Waktu Terukur (ms)	Selisih/Latensi (ms)
1	10	...	...	...
2	50	...	...	...
3	100	...	...	...
4	150	...	...	...
5	200	...	...	...

b. Pengujian Sistem Penggerak

Pengujian sistem penggerak bertujuan untuk memvalidasi kinerja komponen aktuator robot yang terdiri dari Driver Motor L298N, IC expander MCP23017, serta motor DC JGA25-370 yang dilengkapi sensor encoder. Fokus utama pengujian dibagi menjadi tiga tahapan diantaranya, yaitu validasi logika arah untuk memastikan kontrol driver, validasi akurasi sensor encoder sebagai dasar *odometri*, dan analisis karakteristik hubungan PWM terhadap RPM untuk mengetahui respon kecepatan motor.

Tahap pertama adalah validasi logika arah (MCP23017). Pengujian ini dilakukan sebelum motor dijalankan dengan kecepatan tinggi untuk memastikan IC MCP23017 mampu mengirimkan sinyal logika High/Low yang tepat ke pin *input* driver motor. Prosedur pengujian dilakukan dengan mengirimkan perintah program arah gerak (Maju, Mundur, Stop) ke mikrokontroler ESP32, kemudian mengamati respons roda. Jika logika benar namun roda berputar terbalik, kesalahan pengkabelan dapat dideteksi dini. Rancangan data pengujian arah dicatat pada Tabel 3.5.

Tabel 3. 5 Pengujian Validasi Logika Arah (MCP23017)

Perintah Program	**Target Logika Pin MCP (IN1)**	***Output* Logika Pin MCP (IN2)**	**Keterangan**
Maju	...	...	CW
Mundur	...	...	CCW
Stop	...	...	Diam

Tahap kedua adalah pengujian sensor encoder. Pengujian ini bertujuan untuk memvalidasi akurasi pembacaan pulsa yang akan digunakan sebagai acuan *odometri*. Metode yang digunakan adalah pengujian manual dengan memutar roda tepat sebanyak 1 putaran penuh (360°) dan 0,5 putaran (180°). Nilai pulsa yang terbaca oleh mikrokontroler kemudian dibandingkan dengan nilai referensi teoritis. Berdasarkan spesifikasi motor dengan rasio roda gigi 1:46, nilai referensi *encoder*. Tingkat kesalahan (*error*) dihitung menggunakan persamaan (3.3).

$$Error(\%) = \frac{Pulsa\ Terukur - Pulsa\ Reverensi}{Pulsa\ Reverensi} \times 100\% \quad (3.3)$$

Data hasil pengujian akurasi *encoder* target 1 putaran dicatat pada Tabel 3.6.

Tabel 3. 6 Pengujian Sensor *Encoder* 1 Putaran Roda

No.	**Putaran Roda**	**Jumlah Pulsa Acuan**	**Jumlah Pulsa Terbaca**	***Error* (%)**
1	1	...	...	...
2	1	...	...	...
3	1	...	...	...
4	1	...	...	...
5	1	...	...	...
6	1	...	...	...
7	1	...	...	...
8	1	...	...	...
9	1	...	...	...
10	1	...	...	...
Rata-Rata *Error*				...

Setelah pengujian *encoder* pada kondisi satu putaran penuh, pengujian dilanjutkan pada kondisi setengah putaran (0,5 putaran). Pengujian ini bertujuan untuk mengevaluasi konsistensi pembacaan pulsa *encoder* pada perpindahan sudut yang lebih kecil. Rancangan data hasil pengujian tersebut disajikan pada Tabel 3.7.

Tabel 3. 7 Pengujian Sensor *Encoder* Setengah Putaran Roda

No.	**Putaran Roda**	**Jumlah Pulsa Acuan**	**Jumlah Pulsa Terbaca**	***Error* (%)**
1	0,5	...	...	...
2	0,5	...	...	...
3	0,5	...	...	...
4	0,5	...	...	...
5	0,5	...	...	...
6	0,5	...	...	...
7	0,5	...	...	...
8	0,5	...	...	...
9	0,5	...	...	...
10	0,5	...	...	...
Rata-Rata *Error*				...

Tahap ketiga adalah pengujian karakteristik Motor DC (PWM dan RPM). Pengujian ini dilakukan untuk memetakan respon kecepatan sudut roda terhadap perubahan nilai PWM. Prosedur pengujian dilakukan dengan membangkitkan sinyal PWM bertingkat (0 – 255) melalui mikrokontroler dan pengujian dilakukan dari motor satu hingga motor ke empat. Pada setiap tingkat PWM, kecepatan roda diukur menggunakan dua metode secara bersamaan, yaitu perhitungan internal berdasarkan data *encoder* dan pengukuran eksternal menggunakan alat ukur *Tachometer Digital*. Data dari kedua metode pengukuran tersebut dicatat pada Tabel 3.8 untuk dianalisis selisih akurasinya.

Tabel 3. 8 Pengujian Karakteristik Motor DC

Percobaan	**PWM**	**Pembacaan Taco Meter (RPM)**	**Pembacaan Mikrokontroler (RPM)**	***Error* (%)**
1	150	...	...	...
2	165	...	...	...
3	175	...	...	...
4	185	...	...	...
5	195	...	...	...
6	200	...	...	...
7	215	...	...	...
8	230	...	...	...
9	245	...	...	...
10	255	...	...	...
Rata-Rata *Error*				...

c. Pengujian Sensor MPU

Pengujian sensor MPU6050 bertujuan untuk memverifikasi kestabilan pembacaan sudut yaw terhadap waktu (*uji drift*), serta mengukur tingkat akurasi sensor saat robot diputar ke sudut-sudut istimewa, yaitu 90°, 180°, dan 270°.

Prosedur pengujian dilakukan dengan meletakkan robot pada posisi referensi 0° di atas busur derajat dan menjalankan proses kalibrasi awal. Selanjutnya, robot diletakan dalam kondisi diam untuk mencatat pergeseran nilai sudut akhir akibat drift. Program kemudian di-reset kembali ke posisi 0°, dan robot diputar secara manual menuju sudut yang telah ditentukan. Nilai sudut yang terbaca pada sensor

dibandingkan dengan sudut fisik aktual pada busur derajat. Data hasil pengujian selanjutnya dicatat dan disajikan pada tabel 3.9.

Tabel 3. 9 Rancangan hasil pengujian performa *Gyroscope* (Stabilitas & Akurasi)

No	**Kondisi Pengujian**	**Reverensi Fisik (Target)**	**Terukur Sensor (MPU)**	**Selisih (*Error*)**
1	Uji Diam	0°	...	...
2	Rotasi Kanan	90°	...	...
3	Rotasi Belakang	180°	...	...
4	Rotasi Kiri	-90°	...	...

2. Pengujian dan penalaan (*Tuning*) kontrol PID

Pengujian ini bertujuan untuk menentukan nilai parameter kontrol PID (Kp, Ki, Kd) yang optimal bagi setiap motor penggerak roda *omni-wheel*. Metode penalaan yang digunakan adalah metode *Ziegler-Nichols* tipe pertama (Kurva Reaksi), yang didasarkan pada respon transien sistem terhadap masukan langkah (*step input*) secara *open-loop.*

Prosedur pengujian dilakukan melalui beberapa tahapan diantaranya sebagai berikut:

a. Pengambilan data respon *open-loop*

Robot diposisikan menggantung (*jack stand*) agar roda dapat berputar bebas. Mikrokontroler diprogram untuk memberikan sinyal PWM konstan (*step input*) secara tiba-tiba dari kondisi diam ke motor DC tanpa umpan balik kendali.

b. Analisis Grafik Kurva S

Data kecepatan putar motor yang dibaca oleh *rotary encoder* direkam terhadap waktu. Grafik respon yang dihasilkan akan membentuk kurva karakteristik berbentuk S.

c. Penentuan Parameter L dan T

Berdasarkan grafik tersebut, ditarik garis singgung pada titik belok kurva. Perpotongan garis singgung dengan sumbu waktu dan garis kecepatan maksimum digunakan untuk mendapatkan nilai waktu tunda (L) dan konstanta waktu (T).

d. Perhitungan Parameter PID

Nilai L dan T yang diperoleh kemudian disubstitusikan ke dalam persamaan aturan penalaan *Ziegler-Nichols* (sesuai Tabel 2.2) untuk menghitung nilai Kp, Ti, dan Td. Nilai tersebut kemudian dikonversi menjadi konstanta Ki dan Kd agar dapat diterapkan pada program mikrokontroler.

Setelah nilai parameter didapatkan, dilakukan pengujian verifikasi (*closed-loop*) dengan memberikan *setpoint* kecepatan tertentu untuk memastikan sistem memiliki respon yang cepat dan kesalahan (*error*) yang minim. Hasil perbandingan antara *setpoint* dan kecepatan aktual setelah penerapan PID ditampilkan pada Tabel 3.10.

Tabel 3. 10 Perbandingan antara *Setpoint* dan Kecepatan Aktual

Motor	***Setpoint* (RPM)**	**Kecepatan Aktual (RPM)**	**Settling Time (s)**	***Error* (%)**
Motor 1	100	...	...	...

Motor	***Setpoint* (RPM)**	**Kecepatan Aktual (RPM)**	**Settling Time (s)**	***Error* (%)**
Motor 2	100	...	...	...
Motor 3	100	...	...	...
Motor 4	100	...	...	...
Rata-Rata *Error*				...

Nilai parameter PID yang diperoleh dari metode *Ziegler-Nichols* pada kondisi tanpa beban (*no-load*) digunakan sebagai nilai referensi awal. Mengingat adanya pengaruh gaya gesek lantai dan inersia beban total robot saat beroperasi di lintasan nyata, dilakukan tahap penalaan halus (*fine-tuning*).

3. Pengujian Algoritma *Artificial Potential Field* (APF)

Pengujian algoritma APF bertujuan untuk memvalidasi fungsi kalkulasi dari sistem navigasi *mobile robot* yang ditanamkan pada mikrokontroler ESP32, sebelum sinyal kendali dikirimkan ke aktuator (motor DC). Pengujian ini dilakukan secara statis (*dry-run*) dimana robot tidak dijalankan, melainkan nilai keluarannya dipantau secara real-time melalui Serial Monitor pada Arduino IDE. Pengujian algoritma ini dibagi menjadi tiga tahap, yaitu validasi gaya tarik (F_{att}), validasi gaya tolak (F_{rep}), dan pengujian resultan gaya.

a. Validasi Gaya Tarik (F_{att})

Pengujian ini bertujuan untuk melihat respon matematis algoritma terhadap titik target. Berdasarkan persamaan gaya tarik, nilai F_{att} seharusnya sebanding lurus dengan jarak, dimana nilainya akan semakin

mengecil saat robot didekatkan ke target, dan bernilai nol saat robot tepat berada di koordinat target.

Prosedur pengujian dilakukan dengan menetapkan satu koordinat target statis pada program. Kemudian, koordinat posisi awal diubah-ubah secara manual di dalam program untuk merepresentasikan jarak yang berbeda-beda. Nilai keluaran $F_{att,x}$ dan $F_{att,y}$ diamati dan dicatat pada Tabel 3.11.

Tabel 3. 11 Rancangan Pengujian Validasi Gaya Tarik (F_{att})

Posisi Robot (X, Y)	**Posisi Target (X*, Y*)**	**Jarak ke Target (cm)**	**$F_{att,x}$**	**$F_{att,y}$**
(0, 0)	(90, 0)	90	...	...
(20, 0)	(90, 0)	70	...	...
(30, 0)	(90, 0)	60	...	...
(60, 0)	(90, 0)	30	...	...
(90, 0)	(90, 0)	0	...	...

b. Validasi Gaya Tolak (F_{rep})

Pengujian gaya tolak bertujuan memvalidasi respon algoritma saat mendeteksi rintangan dari sensor ultrasonic. Sesuai dengan Batasan algoritma APF, gaya tolak (F_{rep}) hanya akan aktif dan membesar secara eksponensial jika jarak rintangan lebih kecil dari radius aman (ρ_0) yang telah ditetapkan. Prosedur pengujian dilakukan dengan menempatkan robot dalam kondisi diam. Sebuah objek penghalang datar (seperti papan atau kardus) diletakkan di depan sensor ultrasonik utama. Objek tersebut kemudian digeser perlahan mendekati sensor mulai dari jarak di luar radius aman hingga jarak yang sangat dekat. Nilai keluaran gaya

tolak yang dihitung oleh ESP32 dipantau melalui Serial Monitor dan dicatat pada Tabel 3.12.

Tabel 3. 12 Rancangan Pengujian Validasi Gaya Tolak (F_{rep})

Jarak Rintangan (d0)	**Radius Aman (P0)**	**$F_{rep,x}$**	**$F_{rep,y}$**
>50	50	…	…
50	50	…	…
40	50	…	…
30	50	…	…
20	50	…	…

c. Pengujian Resultan Gaya (Vektor Arah)

Setelah gaya tarik dan gaya tolak tervalidasi secara individual, tahap selanjutnya adalah menguji resultan dari kedua gaya tersebut ($F_{total} = F_{att} + F_{rep}$). Resultan gaya ini kemudian diproyeksikan ke dalam komponen sumbu X dan Y sehingga menghasilkan vektor kecepatan referensi (V_x dan V_y) yang menentukan arah gerak robot.

Pengujian dilakukan dengan memberikan variasi koordinat target dan posisi rintangan ke dalam algoritma berdasarkan skenario manuver yang telah dirancang. *Output* vektor gerak divalidasi dengan menganalisis nilai Vx dan Vy (positif, negatif, atau nol) terhadap arah gerak yang diharapkan sesuai dengan prinsip kerja metode *Artificial Potential Field* (APF). Data rancangan pengujian disajikan pada Tabel 3.13.

Tabel 3. 13 Rancangan Pengujian Resultan Gaya APF (vektor kecepatan)

Skenario	***Input* Posisi**			***Output* Vektor Kecepatan (Vx, Vy)**	
	Robot (x, y)	**Target (x, y)**	**Rintangan (x, y)**	**Vx (m/s)**	**Vy (m/s)**
Menuju Target	(0, 0)	(1.5, 0)	-	...	...
Menghindari Rintangan	(0, 0)	(1.5, 0.6)	(0.9, 0)	...	...
Gerak Diagonal	(0, 0)	(1.5, 1.5)	-	...	...
Rintangan Samping	(0, 0)	(0, 1.5)	(0.3, 0.9)	...	...

4. Pengujian Integrasi *Artificial Potential Field* (APF) dan Kontrol PID

   a. Pengujian Kontrol PID Kecepatan Motor DC Langsung di Lintasan

   Pengujian ini bertujuan mengevaluasi respon sistem kontrol PID secara dinamis saat robot diletakkan langsung di atas lintasan nyata yang memiliki gaya gesek dan inersia beban bodi robot. Sinyal langkah (step *input*) diberikan kepada keempat motor secara bersamaan. Data kecepatan (RPM) aktual keempat *rotary encoder* direkam secara *real-time* dan akan diplot ke dalam satu grafik yang sama. Penggabungan grafik keempat motor bertujuan untuk membuktikan tingkat sinkronisasi kecepatan antar roda yang merupakan syarat mutlak kestabilan pergerakan robot *omni-directional*.

b. Pengujian Validasi Posisi Robot dalam Lintasan

Pengujian posisi mobile robot dilakukan dengan cara menginput *setpoint* koordinat berupa nilai X dan Y pada sistem, lalu mencatat hasil *error* (selisih) antara *setpoint* koordinat tersebut dengan koordinat aktual yang ditempuh oleh *mobile robot* di lintasan fisik. Pengujian ini bertujuan untuk memvalidasi tingkat akurasi sistem *odometry* (berbasis sensor *encoder*) dalam melacak dan mengestimasi pergerakan robot di dunia nyata. Untuk mendapatkan data yang komprehensif, pengujian dilakukan pada tiga variasi arah pergerakan, yaitu pergerakan pada sumbu X, sumbu Y dan pergerakan diagonal (X, Y).

Tabel 3. 14 Hasil pengujian pada sumbu X dan Y

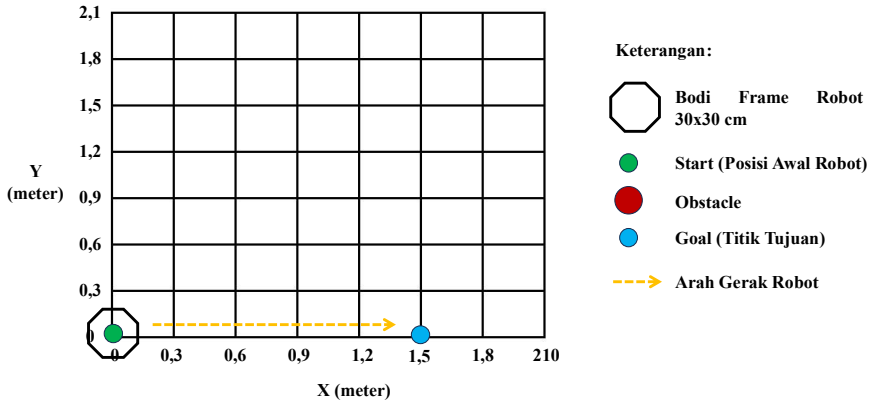
Percobaan	**Target X (cm)**	**Target Y (cm)**	**Aktual X (cm)**	**Aktual Y (cm)**	***Error***
1	30	0	...	...	...
2	30	0	...	...	...
3	30	0	...	...	...
Dst.	...	...	...	...	...
Rata-rata *error* (%)					...

c. Perbandingan Hasil Lintasan (*Trajectory*) *Mobile Robot* (PID dan APF serta APF saja

Pengujian ini bertujuan membandingkan kualitas lintasan (*trajectory*) secara visual antara sistem navigasi yang hanya menggunakan metode APF (*open-loop*) dengan sistem yang mengintegrasikan metode APF dan kontrol PID (*closed-loop*). Koordinat pergerakan aktual robot (X, Y) yang didapatkan dari

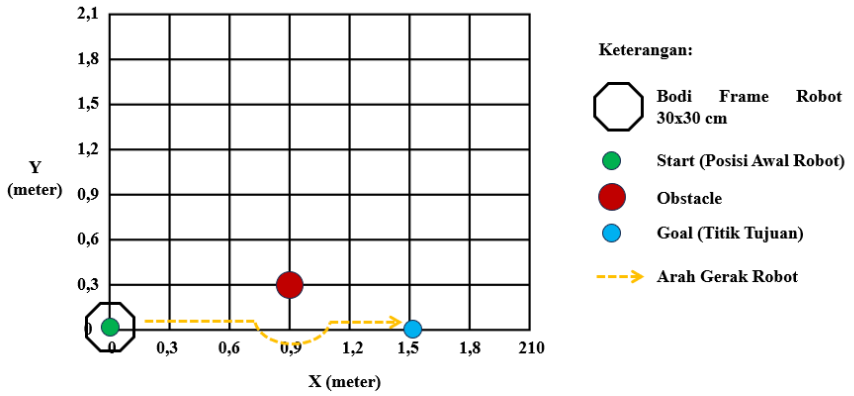
perhitungan sistem *odometri* akan direkam dan di plot menjadi grafik 2D. untuk memvalidasi pergerakan *holonomic* robot pada berbagai arah, rancangan pengujian dilakukan berdasarkan 6 skenario manuver dengan posisi awal (*start*) selalu pada koordinat (0, 0).

Skenario pertama dan kedua difokuskan pada pergerakan translasi lurus menuju target di sumbu X pada koordinat (1.5, 0) meter. Pada skenario ini, pengujian dilakukan tanpa adanya rintangan untuk mengamati apakah sistem PID mampu mencegah penyimpangan lintasan akibat perbedaan putaran keempat motor, sehingga diharapkan menghasilkan garis lurus horizontal yang sempurna.



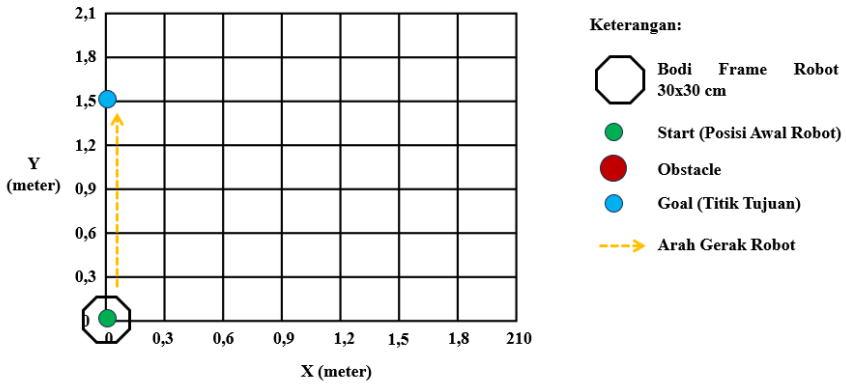
Gambar 3. 9 Skenario lintasan gerak lurus sumbu X menuju target (1.5, 0) tanpa rintangan

Selanjutnya skenario kedua, sebuah rintangan ditambahkan pada koordinat (0.9, 0.3) meter. Rintangan ini untuk menghasilkan gaya tolak asimetris, yang memaksa robot bergeser secara lateral menjauhi rintangan sebelum kembali ke lintasan utama menuju target.



Gambar 3. 10 Skenario lintasan gerak lurus sumbu X menuju target (1.5, 0) dengan rintangan (0.9, 0.3)

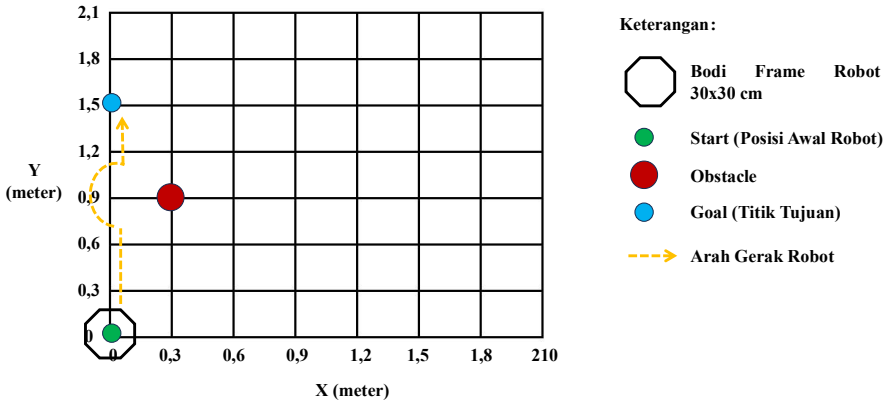
Pengujian selanjutnya, yaitu skenario ketiga dan keempat, dirancang untuk mengevaluasi kemampuan gerak menyamping atau strafing menuju target lateral pada koordinat (0, 1.5) meter. Pada skenario ini tanpa rintangan, robot diperintahkan bergerak sejajar sumbu Y. Peran kontrol PID sangat krusial diuji pada tahap ini untuk mencegah bodi robot berputar (*yawing*) tanpa disengaja akibat kesalahan sinkronisasi motor yang sering terjadi pada mode open-loop.



Gambar 3. 11 Skenario lintasan gerak lateral sumbu Y menuju target (0, 1.5) tanpa rintangan

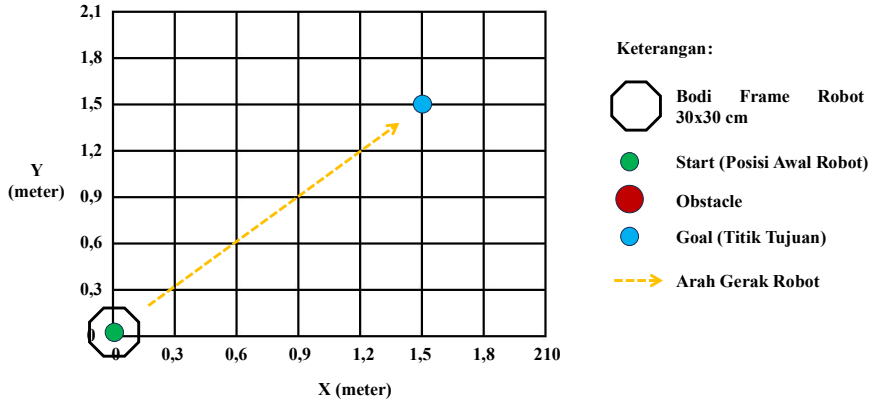
Pada skenario keempat, mengintegrasikan gerak menyamping dengan rintangan yang diletakkan pada koordinat (0.3, 0.9) meter. Keberadaan rintangan ini akan memicu gaya tolak ke arah sumbu X

negatif, sehingga lintasan lateral robot diharapkan akan sedikit membelok melengkung menjauhi titik (0.3, 0.9) sebelum akhirnya kembali merayap lurus menuju target akhir.



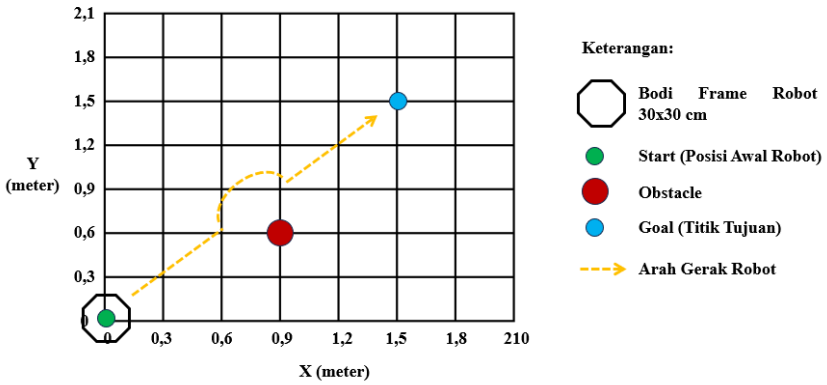
Gambar 3. 12 Skenario lintasan gerak lateral sumbu Y menuju target (0, 1.5) dan rintangan (0.3, 0.9)

Skenario kelima dan keenam, dirancang untuk menguji pergerakan diagonal menuju target pada koordinat (1.5, 1.5) meter. Pada skenario ketiga yang dilakukan tanpa rintangan, robot diharapkan mampu meluncur miring dengan sudut 45° tanpa mengubah orientasi hadap bodi (heading), suatu pergerakan yang menuntut sinkronisasi kecepatan presisi pada resultan sumbu X dan Y.



Gambar 3. 13 Skenario lintasan gerak lurus sumbu X dan Y menuju target (1.5, 1.5) tanpa rintangan

Terakhir, skenario keenam rintangan diletakan di area jalur diagonal pada koordinat (0.9, 0.6) meter. Skenario ini bertujuan menguji kemampuan sistem dalam mengoreksi arah diagonalnya menjadi melengkung saat menghindari rintangan, sekaligus membuktikan seberapa responsive PID dalam menstabilkan keempat roda setelah terganggu oleh gaya rolak APF.



Gambar 3. 14 Skenario lintasan gerak lurus sumbu X dan Y menuju target (1.5, 1.5) dengan rintangan (0.9, 0.6)

d. Perbandingan Akurasi (*Goal Point Error*) *Mobile Robot* Menuju Target

Berdasarkan keenam skenario lintasan di atas, pengujian dilanjutkan dengan mengukur tingkat akurasi posisi akhir robot. Parameter yang diukur adalah *Goal Point Error*, yaitu jarak selisih (dalam cm) antara posisi akhir robot berhenti dengan koordinat target yang sesungguhnya. Hasil pengujian untuk ke-6 skenario dirangkum dalam tabel perbandingan akurasi.

Tabel 3. 15 Perbandingan akurasi *mobile* robot koordinat X, Y

Percobaan	***Mobile Robot* APF dan PID**				**Err (%)**	***Mobile Robot* APF**				**Err (%)**
	***Setpoint* (cm)**		**Aktual (cm)**			***Setpoint* (cm)**		**Aktual (cm)**		
	X	**Y**	**X**	**Y**		**X**	**Y**	**X**	**Y**	
1	150	...	...	...	...	150	...	...	...	...
2	150	...	...	...	...	150	...	...	...	...
3	150	...	...	...	...	150	...	...	...	...
Dst.	...	...	...	...	...	...	...	...	...	...
Rata-rata *Error*					...	Rata-rata *Error*				...

e. Perbandingan Kinerja Waktu Tempuh *Mobile Robot* Menuju Target

Keakuratan lintasan dan posisi yang telah diuji, efisiensi waktu juga menjadi parameter keberhasilan navigasi. Pengujian ini mengukur durasi waktu (*Travel Time*) yang dibutuhkan robot sejak mulai bergerak dari titik start (0, 0) hingga dinyatakan sampai dan berhenti di titik target. Waktu tempuh dicatat menggunakan *time* internal mikrokontroler untuk membandingkan kecepatan penyelesaian misi antara APF murni dengan sistem APF yang distabilkan oleh PID.

Tabel 3. 16 Pengujian Kinerja Waktu Tempuh *Mobile* robot Menuju Target

Percobaan	**Target**	**Rintangan**	**Waktu Tempuh APF (s)**	**Waktu Tempuh APF dan PID (s)**
1	(1.5, 0)	-	…	…
2	(1.5, 0)	-	…	…
3	(1.5, 0)	-	…	…
4	(1.5, 0)	-	…	…

Percobaan	Target	Rintangan	Waktu Tempuh APF (s)	Waktu Tempuh APF dan PID (s)
Dst.	…	…	…	…
Rata-Rata *Error* (%)			…	…

3.1.6 Analisis Kinerja

Pada tahap ini dilakukan proses pengumpulan data dari hasil pengujian sistem yang kemudian dianalisis untuk mengevaluasi tingkat kesesuaian antara teori dan hasil implementasi. Tujuan utama dari tahap ini adalah untuk memastikan apakah rancangan yang dibuat telah mencapai tujuan yang diharapkan. Selain itu, melalui analisis tersebut dapat diidentifikasi berbagai keunggulan dan keterbatasan dari hasil perancangan robot *mobile* yang telah direalisasikan.

3.1.7 Penarikan Kesimpulan

Setelah proses analisis selesai dilaksanakan, dapat disimpulkan mengenai tujuan yang telah ditentukan sesuai dengan yang diharapkan.

3.2 Lokasi Penelitian

Penelitian ini dilaksanakan di lingkungan Universitas Siliwangi kampus 2 mugarsari yaitu di Laboratorium Teknik Elektro Fakultas Teknik Universitas Siliwangi, JL. Mugarsari, Kec. Tamansari, Kota Tasikmalaya, Jawa Barat 46196.

3.3 *Timeline* Pelaksanaan Penelitian

Pelaksanaan penelitian ini mengikuti jadwal kegiatan yang tercantum pada Tabel 3.17 di bawah ini, dengan perkiraan waktu pelaksanaan sekitar empat bulan secara keseluruhan.

Tabel 3. 17 *Timeline* Pelaksanaan Penelitian

Tahap Penelitian	**Bulan ke-1**				**Bulan ke-2**				**Bulan ke-3**				**Bulan ke-4**			
	1	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
Studi Literatur																
Penyusunan Proposal																
Pembuatan Desain Sistem																
Perakitan Sistem																
Pengujian Sistem																
Analisis Kinerja dan Pengumpulan Data																
Kesimpulan																
Penyusunan Laporan Akhir																