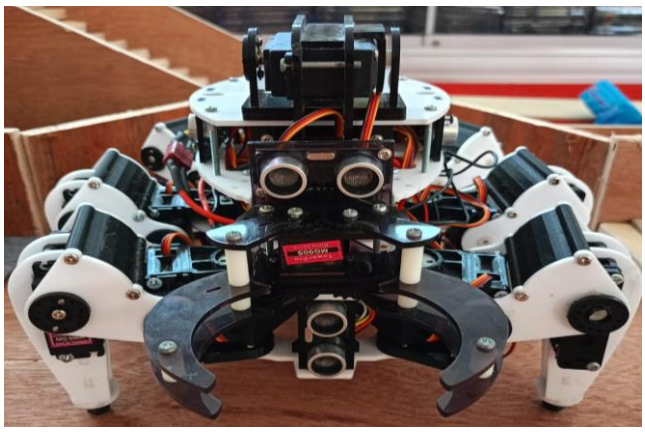
BAB IV

HASIL DAN PEMBAHASAN

4.1 Implementasi Rangkaian Robot

Untuk tahapan ini peneliti melakukan implementasi rangkaian robot akan mengacu pada skema rangkaian robot pada gambar 4.1. Setelah implementasi dilakukan hasil dari robot yang sudah dibangun. Berikut ini adalah hasilnya :

Tampak depan pada robot

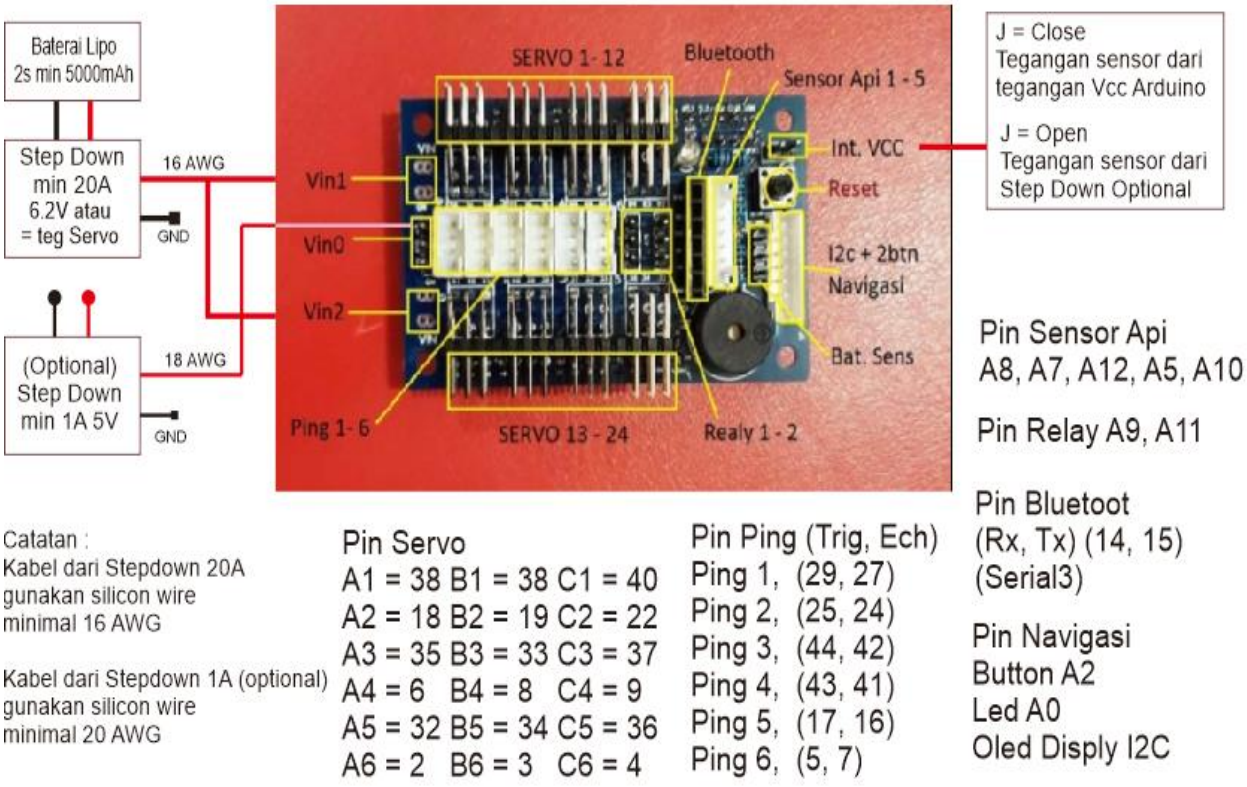


Gambar 4.1 Tampak Depan Pada Robot

Gambar 4.2 Tampak Samping Pada Robot

4.2. Konfigurasi Sistem Keseluruhan

Pada bagian ini, dijelaskan tentang konfigurasi dari setiap komponen yang digunakan untuk membentuk sebuah sistem yang dapat beroperasi sesuai dengan kebutuhan, sehingga siap untuk digunakan. Bagian ini juga mencakup pengujian terhadap perangkat keras setelah sistem diimplementasikan. Setelah pengujian selesai dilakukan, akan teridentifikasi kekurangan dan kelebihan yang dapat digunakan sebagai dasar untuk pengembangan sistem di masa mendatang. Berikut ini adalah langkah-langkah implementasi perangkat keras yang telah dijelaskan sebelumnya.



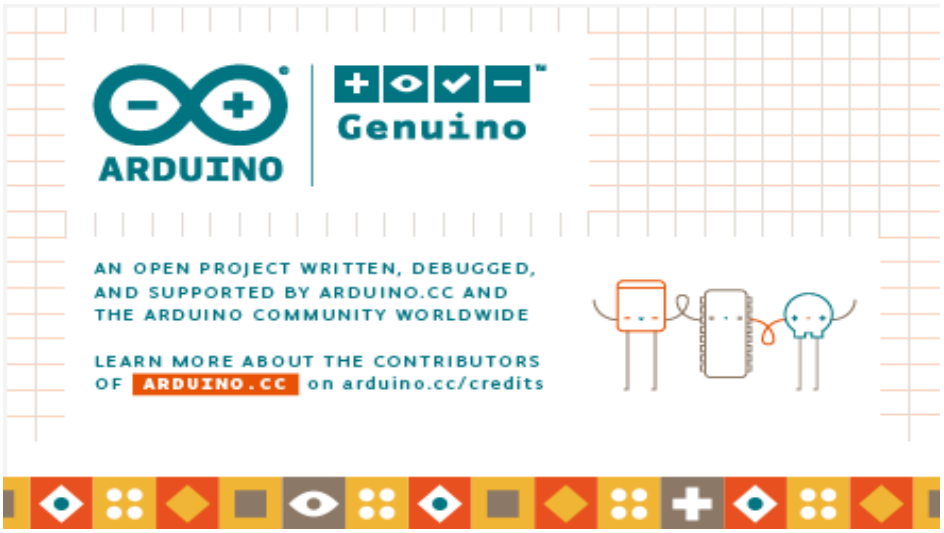
Gambar 4.3 Skema Perangkat Keras

Gambar ini menunjukkan diagram rangkaian dari sistem robot hexapod yang sedang dibuat. Untuk menerapkan diagram rangkaian ini, diperlukan presisi dalam menghubungkan kabel, baik pada pin mikrokontroler maupun pada pin komponen yang digunakan. Komponen yang digunakan, antara lain :

1. Mikrokontroller Arduino Mega Mini 2560
2. Arduino Shield
3. Sensor Ultrasonic HC-SR04
4. 18 Motor Servo
5. Step Up Step Down
6. Push Button
7. Kabel Jumper (sesuai kebutuhan)

Setelah dilakukan implementasi skema perangkat keras, untuk mengimplementasikan sistem yang telah diuraikan peneliti menggunakan software Arduino IDE sebagai lingkungan pengembangan sehingga robot *hexapod* yang dibangun sesuai dengan kebutuhan .

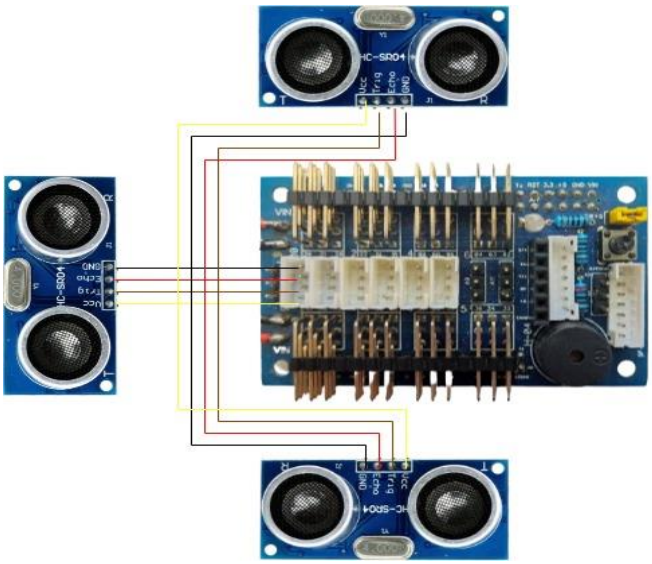
Arduino IDE menggunakan Bahasa pemrrograman C / C++ untuk menuliskan sketsa atau koding untuk mengontrol Mikrokontroller dan komponen lainnya seperti sensor dan actuator supaya berfungsi dengan baik. Sketsa pada Arduino IDE memiliki dua fungsi untuk menjalankan perintah-perintah yaitu *Setup* yang dijalankan satu kali dan *loop* yang akan dijalankan secara terus-menerus (*looping)* setelah perintah *Setup* dijalankan. Pada gambar 4.2 adalah *software* Arduino yang akan dijalankan.



Gambar 4.4 Tampilan *Software* Arduino IDE

Sebelum memulai pengetikan kode pada Arduino IDE, peneliti akan menjelaskan diagram dan pengaturan pin dari komponen yang digunakan dalam pembuatan robot hexapod. Di bawah ini adalah ilustrasi dari rangkaian komponen yang digunakan.

4.3. Sensor Ultrasonik HC-SR04



Gambar 4.5 Sensor Ultrasonik HC-SR04

Sensor Ultrasonik HC-SR04 dapat digunakan untuk mendeteksi rintangan dan menyesuaikan gerakan robot misalnya, jika sensor mendeteksi adanya halangan di depan robot, sensor dapat mengirimkan sinyal ke mikrokontroler robot untuk mengatur arah pergerakannya agar terhindar dari halangan tersebut, berikut konfigurasi pin:

Tabel 4.1 Konfigurasi Pin Sensor Ultrasonic HC-SR04

Nama Sensor	Vcc	Gnd	Trig	Echo
Sensor Depan	5V	Gnd	29	27
Sensor Kanan	5V	Gnd	25	24
Sensor Kiri	5V	Gnd	44	42

Dari pin diatas diinisialisasi pada koding Arduino. Inisialisasi tersebut di tunjukkan pada gambar :

```
#include <NewPing.h>

// Sensor ultrasonik depan
#define echo1Pin 27
#define trig1Pin 29

// Sensor ultrasonik kiri
#define echo2Pin 24
#define trig2Pin 25

// Sensor ultrasonik kanan
#define echo3Pin 42
#define trig3Pin 44

#define jarakMaks 200
#define jumlahSensor 3

NewPing sensors[jumlahSensor] = {
  NewPing(trig1Pin, echo1Pin, jarakMaks),
  NewPing(trig2Pin, echo2Pin, jarakMaks),
  NewPing(trig3Pin, echo3Pin, jarakMaks)
};
```

Gambar 4. 6 Konfigurasi Pin Ultrasonik

Konfigurasi pin untuk Sensor Ultrasonik HC-SR04 dengan sensor depan kiri dan kanan pada umumnya melibatkan dua pin penting: *Trig* dan *Echo*. Sensor ini digunakan untuk mengukur jarak antara sensor dan objek terdekat dengan mengirimkan gelombang ultrasonik dan mengukur waktu pantulannya kembali. Dalam konfigurasi Anda, masing-masing pin diberi label dan pasangan yang cocok adalah sebagai berikut:

1. Sensor Depan Pin 29 (*Trig*) dan Pin 27 (*Echo*) :

   Pin 29 (*Trig*): pin yang digunakan untuk mengirimkan sinyal ultrasonik.

   Pin 27 (*Echo*): Ini adalah pin yang digunakan untuk menerima sinyal pantulan dan mengukur waktu yang diperlukan untuk gelombang ultrasonik kembali ke sensor setelah memantul pada objek.

2. Sensor Kiri Pin 25 (*Trig*) dan Pin 24 (*Echo*) :

   Pin 25 (*Trig*): pin yang digunakan untuk mengirimkan sinyal ultrasonik.

   Pin 24 (*Echo*): Ini adalah pin yang digunakan untuk menerima sinyal pantulan dan mengukur waktu yang diperlukan untuk gelombang ultrasonik kembali ke sensor setelah memantul pada objek.

3. Sensor Kanan Pin 44 (*Trig*) dan Pin 42 (*Echo*) :

   Pin 44 (*Trig*): pin yang digunakan untuk mengirimkan sinyal ultrasonik.

   Pin 42 (*Echo*): Ini adalah pin yang digunakan untuk menerima sinyal pantulan dan mengukur waktu yang diperlukan untuk gelombang ultrasonik kembali ke sensor setelah memantul pada objek.

4.4. Pengujian Sensor Ultrasonik HC-SR04

Untuk mengetahui akurasi hasil pengukuran pada sensor ultrasonic dengan cara melakukan beberapa kali pengujian pada masing-masing sensor ultrasonik yang digunakan untuk bernavigasi. Untuk mengetahui bagaimana pengaruh kombinasi nilai jarak sensor ultrasonik Lalu, mengetahui kesalahan pengukuran dari pembacaan sensor ultrasonik. Pengujian ini dilakukan dengan 3 sensor ultrasonik yang terdapat di depan kiri dan kanan dari robot *hexapod*. Kesalahan pembacaan muncul berupa hasil pengukuran pada bilangan desimal dua di belakang koma. Kesalahan ini tidak mempengaruhi kinerja sistem yang dirancang. Dengan percobaan ini dapat disimpulkan sensor yang digunakan untuk ketiga sensor bekerja dengan baik. Hasil pengujian ditunjukkan oleh table berikut ini :

Tabel 4.2 Pengujian Sensor Ultrasonic

Sensor	**JU 1**	**HU 1**	**A1**	**JU 2**	**HU 2**	**A2**	**JU 3**	**HU 3**	**A3**
Sensor Depan	5cm	5 cm	100%	10 cm	10 cm	100%	15 cm	15 cm	100%
Sensor Kanan	5 cm	5 cm	100%	10 cm	10 cm	100%	15 cm	15 cm	100%
Sensor Kiri	5 cm	5 cm	100%	10 cm	10 cm	100%	15 cm	15 cm	100%
Rata Rata Akurasi			100%			100%			100%

Keterangan :

JU = Jarak Ukur (Nilai sebenarnya) HU = Hasil Ukur (Nilai yang diperoleh)

A = Akurasi

Tabel diatas adalah data pengukuran (JU dan HU) beserta akurasinya (A).

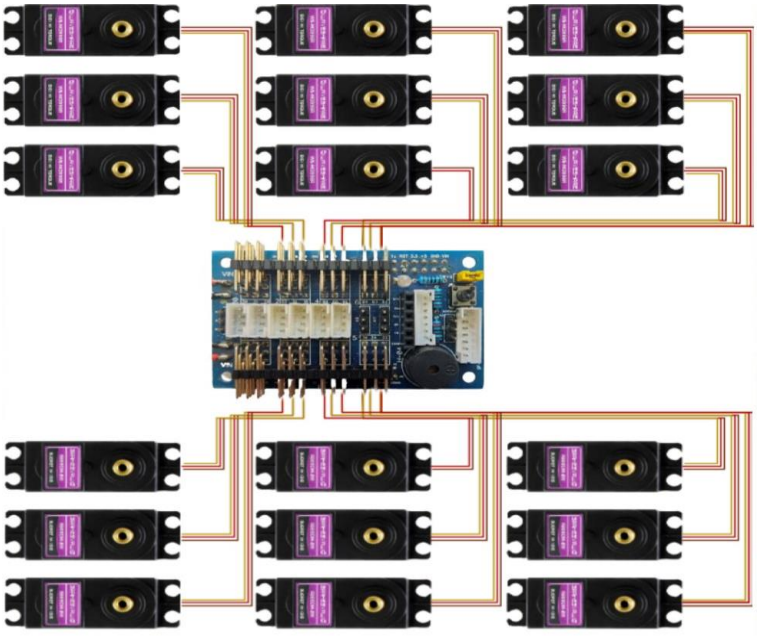
1. JU 1, HU 1, A1 : Ini adalah hasil dari pengukuran pertama. JU 1 adalah nilai sebenarnya dari jarak ukur, HU 1 adalah nilai yang diperoleh dari hasil pengukuran awal, dan A1 adalah akurasi pengukuran pertama dalam persentase.
2. JU 2, HU 2, A2 : Ini adalah hasil dari pengukuran kedua. JU 2 adalah nilai sebenarnya dari jarak ukur, HU 2 adalah nilai yang diperoleh dari hasil pengukuran kedua, dan A2 adalah akurasi pengukuran kedua dalam persentase.
3. JU 3, HU 3, A3 : Ini adalah hasil dari pengukuran ketiga. JU 3 adalah nilai sebenarnya dari jarak ukur, HU 3 adalah nilai yang diperoleh dari hasil pengukuran ketiga, dan A3 adalah akurasi pengukuran ketiga dalam persentase.

Dalam semua kasus, nilai JU (Jarak Ukur) adalah nilai yang sebenarnya atau diharapkan dari pengukuran. Nilai HU (Hasil Ukur) adalah nilai yang diperoleh dari pengukuran yang sebenarnya dilakukan. Akurasi (A) adalah perbandingan persentase antara nilai HU dan JU. Jika akurasi bernilai 100%, itu berarti nilai HU sama persis dengan nilai JU.

Kesimpulan akhir akan sangat bergantung pada metode pengujian yang tepat, pengaturan eksperimen, lingkungan pengujian, dan seberapa baik sensor HC-SR04 mengatasi faktor-faktor yang dapat memengaruhi akurasi pengukuran. Jika sensor mampu memberikan hasil yang konsisten dan akurat dalam berbagai kondisi, maka kesimpulan yang positif mengenai akurasi sensor dapat diambil. Dengan menggabungkan hasil pengujian, Anda dapat menilai apakah sensor ini cocok untuk aplikasi Anda dan bagaimana cara mengoptimalkan penggunaannya.

4.5. Rangkaian Motor Servo

Motor Servo dapat digunakan untuk mengontrol pergerakan setiap kaki robot hexapod. Hal ini memungkinkan kontrol gerakan robot yang tepat dan memungkinkannya bergerak dalam berbagai arah dan kecepatan. Secara keseluruhan, kombinasi servo motor dan desain *hexapod* dapat menghasilkan robot yang sangat bermanuver.



Gambar 4.7 Rangkaian Motor Servo

Rangkaian berikut ini adalah rangkaian dari motor servo yang digunakan sebagai aktuator atau penggerak servo dari robot *hexapod*. Motor servo memiliki masing –masing memiliki tiga pin yaitu vcc, gnd, dan signal yang harus terhubung pada mikrokontroler Arduino mega 2560. Rangkaian motor servo ditunjukkan pada table berikut ini :

Tabel 4.3 Konfigurasi Pin Motor Servo

Nama Motor Servo	Vcc	Gnd	Pin
Servo 1	5V	Gnd	22, 19, 18
Servo 2	5V	Gnd	9, 8, 6
Servo 3	5V	Gnd	4, 3, 2
Servo 4	5V	Gnd	36, 34, 32
Servo 5	5V	Gnd	37, 33, 35
Servo 6	5V	Gnd	40, 39, 38

Dari penempatan motor servo tersebut dapat di inisialisasikan pada Arduino IDE seperti pada gambar 4.8 berikut ini

```
//======== Servo variable
byte tunda = 80;

//float pos_1A, pos_1B, pos_1C, pos_2A, pos_2B, pos_2C, pos_3A, pos_3B, pos_3C;
//float pos_4A, pos_4B, pos_4C, pos_5A, pos_5B, pos_5C, pos_6A, pos_6B, pos_6C;

float posA[7];
float posB[7];
float posC[7];

int coxaPin[] = {0, 22, 9, 4, 36, 37, 40};

int femurPin[] = {0, 19, 8, 3, 34, 33, 39};

int tibiaPin[] = {0, 18, 6, 2, 32, 35, 38};

Servo myservoA; Servo myservoB; Servo myservoC;
Servo srv_GR; Servo srv_UP; Servo srv_k;
Servo sCoxa[7];
Servo sFemur[7];
Servo sTibia[7];
```

Gambar 4.8 Konfigurasi Pin Servo

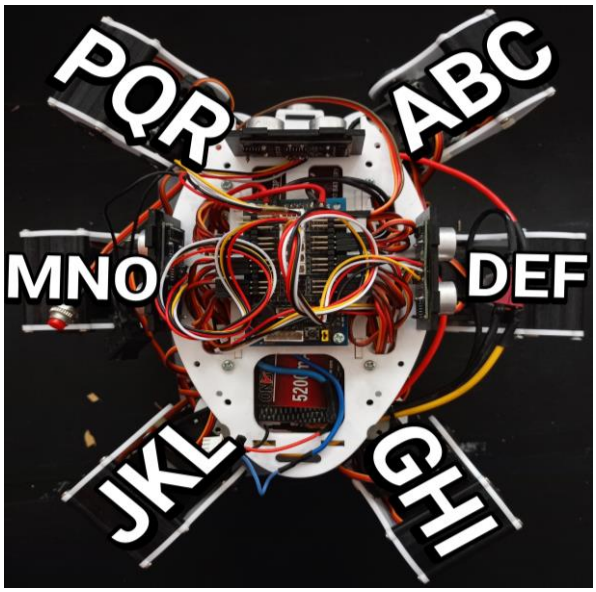
Tabel diatas adalah daftar motor servo beserta koneksi daya dan pin yang diperlukan untuk mengendalikannya. Di bawah ini adalah penjelasan dari setiap kolom dalam tabel tersebut:

1. Nama Motor Servo : Ini adalah identifikasi atau label untuk setiap motor servo yang terhubung. Nama ini dapat digunakan untuk mengidentifikasi servo tertentu dalam program atau pengaturan perangkat keras.

2. Vcc : koneksi daya positif (tegangan) untuk motor servo. Tegangan ini biasanya adalah 5V, untuk menyediakan daya ke motor servo agar dapat berfungsi.
3. Gnd : Ini adalah koneksi *ground* atau referensi nol untuk motor servo. Koneksi ini untuk melengkapi sirkuit dan menyediakan jalur kembali untuk arus listrik.
4. Pin : Ini adalah pin yang digunakan untuk mengendalikan motor servo. Motor servo diatur melalui sinyal PWM (*Pulse Width Modulation*) yang diberikan melalui pin ini. Sinyal PWM mengontrol posisi sudut motor servo. Setiap motor servo memiliki beberapa pin yang terhubung, yang mewakili sinyal kontrol, tegangan positif (Vcc), dan ground (Gnd).

Jadi, dalam tabel tersebut, terdapat daftar motor servo beserta koneksi daya (Vcc dan Gnd) serta pin kontrol (Pin) yang diperlukan untuk mengoperasikan masing-masing motor servo.

4.6. Pengujian Motor Servo



Gambar 4.9 Ilustrasi Penamaan Servo

Tabel 4.4 Pengujian Motor Servo

Servo	Coxa	Femur	Tibia
Servo ABC	90°	90°	90°
Servo DEF	90°	90°	90°
Servo GHI	90°	90°	90°
Servo JKL	90°	90°	90°
Servo MNO	90°	90°	90°
Servo PGR	90°	90°	90°

Tabel diatas adalah pengaturan sudut pada beberapa servo yang terhubung dengan sendi-sendi dalam suatu mekanisme atau robot.

1. Servo: Ini adalah nama atau label dari setiap servo yang ada dalam mekanisme atau robot. Setiap servo mungkin terhubung dengan satu atau beberapa sendi untuk mengontrol pergerakan dan posisi.
2. *Coxa* : Ini adalah jenis sendi atau bagian yang mirip dengan panggul pada makhluk hidup. Sudut 90° menunjukkan bahwa setiap servo pada sendi *coxa* diatur pada posisi awal atau netral 90 derajat.
3. *Femur* : Ini adalah jenis sendi atau bagian yang menyerupai fungsi tulang *femur* pada kaki vertebrata darat. Sekali lagi, sudut 90° menunjukkan bahwa setiap servo pada sendi *femur* diatur pada posisi awal atau netral 90 derajat.
4. *Tibia* : Ini adalah jenis sendi atau bagian yang menyerupai fungsi tulang tibia pada kaki vertebrata darat. Seperti sebelumnya, sudut 90° menunjukkan bahwa setiap servo pada sendi *tibia* diatur pada posisi awal atau netral 90 derajat.

Tabel ini adalah konfigurasi awal dari suatu mekanisme kaki robotik atau serangkaian servo untuk mensimulasikan gerakan kaki. Sudut 90 derajat sering digunakan sebagai posisi awal karena merupakan posisi yang stabil dan netral.

Kesimpulan dari pengujian Setiap motor servo memiliki sudut derajat yang sama yaitu 90° untuk setiap servo, karena, Ketika sudut servo diatur sama, gerakan yang dihasilkan pada kedua sisi robot akan seimbang, membantu mencegah robot terjatuh atau kehilangan keseimbangan saat bergerak.

4.7. Perhitungan Metode *Proportional Integral Derivative*

Kombinasi dari aksi pengendalian *proporsional, integral, dan diferensial* memiliki keunggulan yang lebih baik daripada penggunaan masing-masing dari ketiga tindakan pengendalian tersebut secara terpisah. Setiap kontroler P, I, dan D memiliki peran masing-masing dalam mempercepat respons sistem, menghilangkan pergeseran (*offset*), dan mendapatkan energi tambahan saat terjadi perubahan dalam beban sistem. Perhitungan metode PID didasarkan pada perbedaan antara setpoint (nilai yang diinginkan) dan output aktual dari sistem. Perbedaan ini disebut *error*. Komponen P, I, dan D akan menghitung sinyal kontrol berdasarkan *error* untuk mengubah *output* sistem agar mendekati *setpoint*.

Berikut adalah rumus umum untuk perhitungan metode PID:

Output = Kp * (P + Ki * I + Kd * D) Di mana:

Output adalah sinyal kontrol yang dihasilkan oleh metode PID.

Kp, Ki, dan Kd adalah parameter yang disebut sebagai gain yang digunakan untuk mengatur seberapa besar pengaruh dari masing-masing komponen PID.

P adalah komponen *proporsional* yang *proporsional* dengan *error*.

I adalah komponen *integral* yang mengintegrasikan *error* seiring waktu untuk mengatasi *error statis* dalam sistem.

D adalah komponen *derivative* yang mengukur laju perubahan *error* untuk mengatasi *error* yang terjadi secara *dinamis*.

Perhitungan masing-masing komponen PID adalah sebagai berikut:

P = error

I = I + (error * waktu_sampel)

D = (error - error_sebelumnya) / waktu_sampel

Dalam perhitungan komponen I, waktu_sampel adalah *interval* waktu antara dua pengambilan sampel. Error_sebelumnya adalah *error* pada pengambilan sampel sebelumnya. Namun, perhitungan metode PID yang lebih kompleks bisa melibatkan penggunaan faktor penghalus (*filtering*) atau metode tuning khusus untuk mendapatkan hasil yang optimal. Juga, dalam implementasi praktis, biasanya digunakan mekanisme anti-windup dan batasan output agar kontrol PID lebih stabil dan aman.

Dalam perhitungan kontrol PID (*Proportional-Integral-Derivative*), kita akan menggunakan parameter kp, ki, dan kd untuk mengatur respons sistem terhadap set point. Dalam kasus ini, kita akan mengambil contoh penggunaan kontrol PID pada robot *hexapod* untuk menjaga jarak antara dinding pada nilai set point 15.

Robot *hexapod* memiliki sensor yang dapat mengukur jarak antara dinding dan dapat menghasilkan output sebagai *error*. Dan akan menggunakan rumus

dasar kontrol PID untuk menghitung sinyal kontrol yang diberikan ke robot *hexapod*:

1. *Proporsional (P)* : Respon terhadap selisih antara *setpoint* dan posisi saat ini. Nilai ini mengontrol seberapa cepat robot bereaksi terhadap perubahan error.
2. *Integral (I)* : Mengontrol akumulasi dari *error* selama waktu. Ini membantu menangani *steady-state error* dan mencegahnya terus meningkat seiring waktu.
3. *Derivatif (D)* : Menanggapi laju perubahan *error*. Ini membantu meredakan *overshoot* dan mengurangi osilasi.

Jika sudah mengukur *error* (selisih antara *setpoint* dan posisi saat ini) dan mendapatkan hasil berikut:

Error saat ini: e(t) = 10 (dalam satuan yang sesuai dengan posisi robot).

Derivative dari *error*: de(t)/dt = 3 (dalam satuan yang sesuai dengan perubahan posisi per waktu).

Integral dari *error*: ∫e(t) dt = 25 (dalam satuan yang sesuai dengan akumulasi *error* selama waktu).

Sekarang, kita dapat menggunakan rumus kontrol PID untuk menghitung aksi kontrol total:

Aksi Kontrol Total

$$(u(t)) = Kp * e(t) + Ki * \int e(t)\,dt + Kd * de(t)/dt$$

Dengan nilai yang Anda berikan

$$(Kp = 5.0, Ki = 1.0, Kd = 0.1):$$

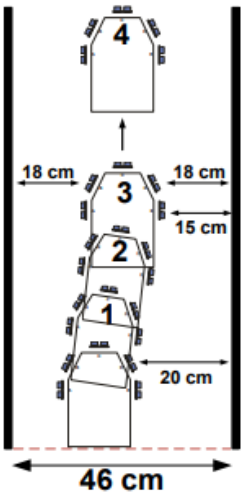
$$u(t) = 5.0 * 10 + 1.0 * 25 + 0.1 * 3 = 50 + 25 + 0.3 = 75.3$$

Hasil aksi kontrol total adalah 75.3. Nilai ini dapat digunakan sebagai input ke sistem kontrol robot hexapod Anda, misalnya, untuk menggerakkan kaki-kaki robot dalam mengatasi *error* yang terdeteksi.

Selanjutnya, sinyal kontrol ini akan digunakan untuk menggerakkan robot hexapod agar mendekati *setpoint*. Jika dinding semakin mendekat, *error* akan berkurang dan robot akan mengurangi kecepatan pergerakan.

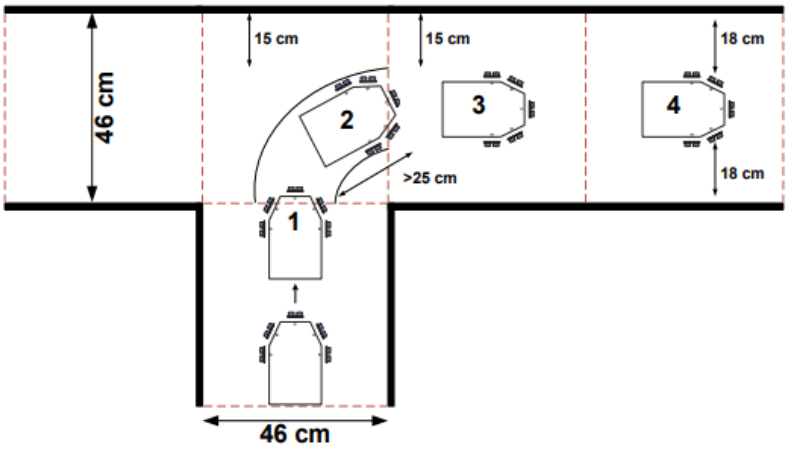
4.8. Pengujian PID Wall Following

Saat pembacaan sensor ultrasonik membandingkan jarak setpoint dengan jarak robot terlebih dahulu. Perbandingan ini memberikan nilai kesalahan. Variabel input mengikuti dinding adalah jarak baca sensor ultrasonik keluarannya adalah gerakan maju dan berbelok yang dipilih dan diulang untuk memenuhi nilai kesalahan yang meningkat. Penentuan limit fungsi input mempertimbangkan objek yang ditangkap oleh sensor ultrasonik, robot ada di dalam lintasan. Fungsi input sensor ultrasonik di sebelah kiri adalah set point untuk proses tuning dikendalikan PID. Sensor ultrasonik menentukan limit fungsi pada masukan pada bagian kiri dilakukan pembacaan sensor saat robot berada di lintasan. Saat posisi robot tidak ada pada jarak set point, jarak terbaca pada sensor ultrasonik samping kiri adalah 10. Sedangkan jarak dari set point yaitu 15 jadi robot akan mengalami beberapa kali mengulang gerakan berbelok ke kanan supaya dapat mencapai jarak set point yang dicapai. Ilustrasi pergerakan robot pada sisi samping berikut adalah Gambar 5.



Gambar 4.10 Ilustrasi Robot dalam Lintasan

Fungsi input dari sisi sensor ultrasonik bagian depan juga digunakan untuk pendeteksian adanya belokan. Fungsi input sensor depan digunakan untuk menggerakkan robot berbelok ke kanan. Pada gambar 4.10 ilustrasi posisi robot untuk menentukan batas-batas fungsi masukan dalam mengikuti sisi dinding kiri. Saat ada di posisi 1, ultrasonik depan menangkap objek sebesar 15 cm, sehingga robot berbelok ke kanan ke arah sisi dinding dan berlanjut hingga robot berada di posisi 2.



Gambar 4.11 Robot Mengikuti Dinding

Input dari sensor ultrasonik di sisi kanan tengah dan sensor ultrasonik di sisi kiri tengah juga dimanfaatkan untuk mendeteksi kemungkinan adanya tikungan atau persimpangan.

Tabel 4.5 Pengujian *Wall Following*

No	Benturan	Waktu	Hasil Mengikuti Dinding
1	4	4.53	Berhasil
2	1	3.94	Berhasil
3	3	4.26	Berhasil
4	2	4.10	Berhasil
5	1	3.96	Berhasil
6	0	3.88	Berhasil
7	2	4.07	Berhasil
8	0	3.91	Berhasil
9	3	4.28	Berhasil
10	0	3.86	Berhasil

Tabel di atas merupakan hasil pengujian atau eksperimen yang melibatkan suatu objek atau sistem yang berinteraksi dengan dinding. Tabel ini mencatat jumlah benturan, lama waktu, dan hasil pengujian dari beberapa percobaan yang dilakukan. Berikut adalah penjelasan dari kolom-kolom dalam tabel tersebut:

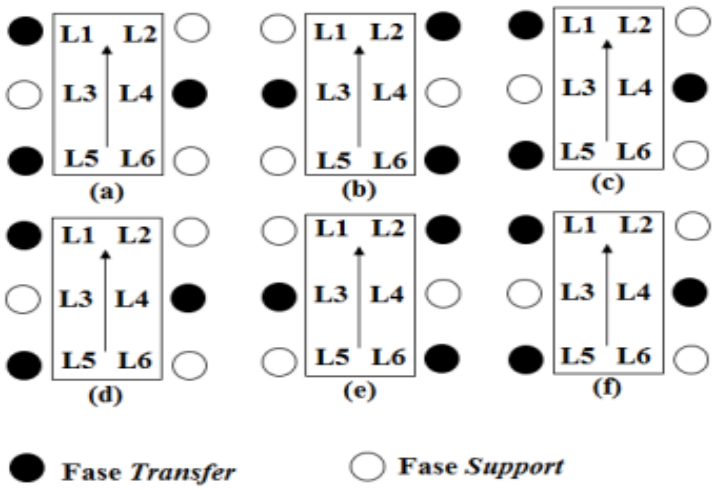
Benturan: Ini mewakili jumlah kali objek atau sistem tersebut bertabrakan atau mengenai dinding selama percobaan.

Waktu: Ini adalah durasi atau waktu yang diperlukan dalam detik untuk setiap percobaan. Ini bisa saja mencerminkan berapa lama objek atau sistem tersebut bergerak sebelum akhirnya mencapai dinding atau sebelum percobaan dihentikan.

Kesimpulan dari pengujian wall following membantu mengukur sejauh mana robot atau sistem mampu mengikuti bentuk dinding atau penghalang yang diberikan. Kesimpulan yang dapat diambil adalah seberapa baik robot mengikuti kontur dinding dengan presisi dan kecepatan yang sesuai.

4.9. Pengujian Tripod Gait

Pengujian tripod gait merujuk pada pengujian atau evaluasi dari suatu sistem atau mekanisme gerakan tripod dalam robotika atau ilmu komputer. Gait tripod adalah salah satu pola gerakan yang digunakan oleh beberapa robot untuk berjalan atau bergerak. Pengujian ini dilakukan dengan mengimplementasikan algoritma program langsung pada robot, dengan memeriksa pola langkah yang dihasilkan oleh setiap kaki robot. Algoritma yang digunakan dalam pengujian ini adalah tripod gait. Gambar (4.11) menunjukkan pola langkah tripod gait



Gambar 4.12 Urutan Langkah Pola *Tripod Gait*

```
if (g == 5) {//tripod
  steepNumber = 14;
  perStep = steepNumber;

  float tp1[15] = {0, -15, -10, -5, 0, 5, 10, 15, 18, 12, 6, 0, -6, -12, -18};
  float tp2[15] = {0, 18, 12, 6, 0, -6, -12, -18, -15, -10, -5, 0, 5, 10, 15};

  sl[1] = tp1[steep];
  sl[2] = tp2[steep];
  sl[3] = tp1[steep];
  sl[4] = tp2[steep];
  sl[5] = tp1[steep];
  sl[6] = tp2[steep];
}
```

Gambar 4.13 Koding *Tripod Gait*

Tabel 4. 6 Pengujian *Tripod Gait*

Pengujian	Kp	Ki	Kd	Jumlah Benturan	Lama Waktu
1	4.3	0.3	1.0	5	3.43
2	4.4	0.4	0.9	4	3.37
3	4.5	0.5	0.8	4	3.39
4	4.6	0.6	0.7	2	3.28
5	4.7	0.7	0.6	3	3.31
6	4.8	0.8	0.1	2	3.22
7	4.9	0.9	0.4	1	3.19
8	5.0	1.0	0.1	0	3.10
9	5.1	1.1	0.2	2	3.27
10	5.2	1.2	0.1	2	3.26

Tabel diatas adalah suatu eksperimen atau data yang dikumpulkan dalam sebuah penelitian. Dari kolom-kolom yang tercantum, kita dapat menjelaskan informasi sebagai berikut:

1. Kp, Ki, Kd : Tiga parameter ini adalah bagian dari PID yang digunakan dalam sistem kontrol untuk mengatur respons dari suatu sistem terhadap perubahan dalam input atau lingkungannya.

   Kp (*Proporsional*) : Parameter ini mengontrol seberapa besar respons output terhadap perbedaan nilai.

   Ki (*Integral*) : Parameter ini mengontrol penyesuaian terhadap kesalahan yang terakumulasi seiring waktu, untuk menghilangkan kesalahan bias dalam sistem.

   Kd (*Derivatif*) : Parameter ini mengontrol respons terhadap tingkat perubahan kesalahan. Ini membantu dalam mengurangi osilasi yang berlebihan.
2. Jumlah Benturan : mengacu pada jumlah kali sistem mengalami "benturan" atau "kegagalan." Benturan di sini merujuk pada situasi di mana sistem tidak beroperasi seperti yang diharapkan atau mencapai tujuan yang diinginkan.
3. Lama Waktu : adalah durasi atau waktu yang diperlukan untuk satu siklus eksperimen hingga melewati rintangan.

   Sesuai dengan data dalam tabel, peneliti sedang mengamati atau menguji sistem di berbagai Kp, Ki, dan Kd. Anda juga mungkin mencatat jumlah benturan atau kegagalan sistem serta lama waktu yang diperlukan dalam setiap pengujian.

Untuk memberikan penjelasan lebih lanjut atau menganalisis hasil dari tabel ini, diperlukan konteks lebih lanjut tentang eksperimen atau penelitian yang dilakukan serta tujuan dari pengujian yang dilakukan.

Kesimpulan dari pengujian tripod gait dengan nilai PID (*Proportional-Integral-Derivative*) Kp= 5.0, Ki= 1.0, Kd= 0.1 adalah sebagai berikut:

Kinerja Gerakan: Pengaturan PID dengan nilai-nilai yang diberikan (KP 5.0, KI 1.0, KD 0.1) menghasilkan kinerja gerakan yang baik pada tripod gait. Hal ini mungkin ditunjukkan oleh stabilitas gerakan, minimnya getaran, dan respons yang cepat terhadap perubahan kondisi.

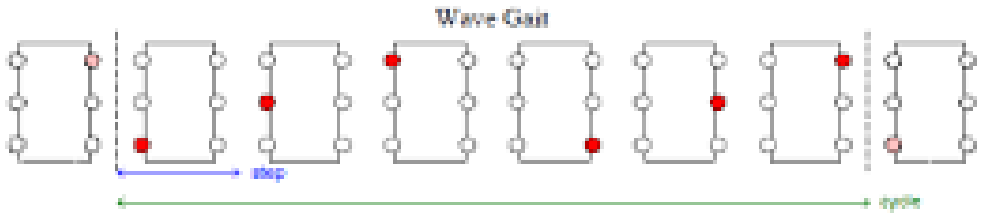
Respons terhadap Perubahan: Nilai-nilai PID tersebut menghasilkan respons yang cukup cepat terhadap perubahan kondisi atau lingkungan. Robot dengan cepat menyesuaikan gerakannya untuk mengatasi gangguan atau perubahan dalam lingkungan sekitarnya.

4.10. Pengujian *Wave Gait*

Pola gerakan ini didasarkan pada koordinasi yang diatur dengan baik antara kaki-kaki di sisi yang berlawanan. Pada *wave gait*, kaki-kaki di satu sisi (misalnya, kaki kanan) bergerak bersama-sama, sementara kaki-kaki di sisi lainnya (misalnya, kaki kiri) bergerak bersama-sama dengan fase yang berbeda. *Wave gait* pada hexapod memiliki beberapa keuntungan. Pertama, *wave gait* memberikan stabilitas yang baik karena pada setiap saat ada tiga kaki yang bertindak sebagai penopang. Hal ini membantu mengurangi kemungkinan terjatuh atau kehilangan keseimbangan saat bergerak. Kedua, *wave gait* memungkinkan

hexapod untuk bergerak dengan kecepatan yang relatif tinggi karena kaki-kaki yang bergerak secara bergantian mengurangi hambatan saat melangkah.

Pengujian ini dilaksanakan dengan metode langsung mengaplikasikan algoritma program ke robot, sambil memantau pola gerakan masing-masing kaki pada robot. Jenis algoritma yang digunakan adalah "*wave gait*" Gambar (4.14) menunjukkan pola gerakan *wave gait*.



Gambar 4.14 Urutan Langkah Pola *Wave Gait*

Pada gambar (4.14) berjalan gelombang, gerakan kaki dimulai dari salah satu sisi robot dan berlanjut ke sisi lainnya secara berurutan seperti gelombang yang bergerak. Misalnya, gerakan dimulai dengan menggerakkan kaki depan di satu sisi, lalu diikuti oleh kaki tengah dan kaki belakang pada sisi yang sama. Setelah itu, gerakan dilanjutkan ke sisi sebaliknya dengan menggerakkan kaki depan di sisi yang berlawanan, diikuti oleh kaki tengah dan kaki belakang. Pola gerakan ini memberikan kestabilan dan keseimbangan yang baik pada robot hexapod saat berjalan. Dengan menggunakan gaya berjalan gelombang, robot hexapod dapat bergerak maju secara efisien dan dapat menyesuaikan diri dengan berbagai jenis medan yang tidak rata.

```
if (g <= 2) { //wave
  steepNumber = 36;
  perStep = steepNumber / 2.4;

  float  gp1[] = {0, -15, -10, -5, 0, 5, 10, 15, 18, 16.7, 15.4, 14.1, 12.9, 11.6, 10.3, 9, 7.7, 6.4, 5.1, 3.9, 2.6, 1.3, 0, -1.3, -2.6, -3.9, -5.1, -6.4, -7.7, -9, -10.3, -11.6, -10.3, -11.6,
  float  gp2[] = {0, -11.6, -10.3, -11.6, -12.9, -14.1, -18, -15, -10, -5, 0, 5, 10, 15, 18, 16.7, 15.4, 14.1, 12.9, 11.6, 10.3, 9, 7.7, 6.4, 5.1, 3.9, 2.6, 1.3, 0, -1.3, -2.6, -3.9, -5.1, -6.
  float  gp3[] = {0, -3.9, -5.1, -6.4, -7.7, -9, -10.3, -11.6, -10.3, -11.6, -12.9, -14.1, -18, -15, -10, -5, 0, 5, 10, 15, 18, 16.7, 15.4, 14.1, 12.9, 11.6, 10.3, 9, 7.7, 6.4, 5.1, 3.9, 2.6,
  float  gp4[] = {0, 3.9, 2.6, 1.3, 0, -1.3, -2.6, -3.9, -5.1, -6.4, -7.7, -9, -10.3, -11.6, -10.3, -11.6, -12.9, -14.1, -18, -15, -10, -5, 0, 5, 10, 15, 18, 16.7, 15.4, 14.1, 12.9, 11.6, 10.3
  float  gp5[] = {0, 11.6, 10.3, 9, 7.7, 6.4, 5.1, 3.9, 2.6, 1.3, 0, -1.3, -2.6, -3.9, -5.1, -6.4, -7.7, -9, -10.3, -11.6, -10.3, -11.6, -12.9, -14.1, -18, -15, -10, -5, 0, 5, 10, 15, 18, 16.7
  float  gp6[] = {0, 15, 18, 16.7, 15.4, 14.1, 12.9, 11.6, 10.3, 9, 7.7, 6.4, 5.1, 3.9, 2.6, 1.3, 0, -1.3, -2.6, -3.9, -5.1, -6.4, -7.7, -9, -10.3, -11.6, -10.3, -11.6, -12.9, -14.1, -18, -15,

  if (g == 1) {
    sl[1] = gp1[steep];
    sl[2] = gp2[steep];
    sl[3] = gp3[steep];
    sl[4] = gp4[steep];
    sl[5] = gp5[steep];
    sl[6] = gp6[steep];
  }
  if (g == 2) {
    sl[1] = gp1[steep];
    sl[2] = gp2[steep];
    sl[3] = gp3[steep];
    sl[4] = gp6[steep];
    sl[5] = gp5[steep];
    sl[6] = gp4[steep];
  }
}
```

Gambar 4.15 Koding *Wave Gait*

Tabel 4.7 Pengujian *Wave Gait*

Pengujian	Kp	Ki	Kd	Jumlah Benturan	Lama Waktu
1	4.3	0.3	1.0	4	4.54
2	4.4	0.4	0.9	5	4.71
3	4.5	0.5	0.8	3	4.38
4	4.6	0.6	0.7	5	4.67
5	4.7	0.7	0.6	3	4.31
6	4.8	0.8	0.1	2	4.23
7	4.9	0.9	0.4	1	4.18
8	5.0	1.0	0.1	0	4.16
9	5.1	1.1	0.2	2	4.29
10	5.2	1.2	0.1	2	4.25

Kesimpulan dari pengujian menggunakan metode gerakan gelombang (*wave gait*) dengan parameter PID (*Proporsional-Integral-Derivative*) yang memiliki nilai, yaitu Kp 5.0, Ki 1.0, dan Kd 0.1, adalah sebagai berikut:

Stabilitas Gerakan: Penggunaan parameter PID dengan nilai yang telah diuji memberikan hasil yang stabil dalam gerakan gelombang. Nilai-nilai KP, KI, dan KD yang dipilih tampaknya bekerja bersama-sama untuk mencegah fluktuasi berlebihan dan perubahan yang tiba-tiba dalam gerakan.

Akurasi Gerakan: Dengan parameter PID yang diatur pada nilai-nilai tertentu, gerakan gelombang yang dihasilkan mungkin lebih akurat dan terkendali. Parameter PID membantu mengoreksi kesalahan gerakan dan mencegah pergeseran yang tidak diinginkan.

Performa Keseluruhan: Dari pengujian yang dilakukan, nilai-nilai KP 5.0, KI 1.0, dan KD 0.1 tampaknya memberikan keseimbangan yang baik antara respons gerakan yang cepat, penyeimbangan yang baik, dan minimnya osilasi berlebihan. Ini mengindikasikan bahwa parameter PID yang telah dipilih dapat menghasilkan performa keseluruhan yang baik.

4.11. Pengujian Rintangan

Pengujian rintangan (*obstacle testing*) adalah sebuah proses untuk menguji kemampuan suatu sistem atau perangkat untuk melewati atau mengatasi berbagai rintangan atau hambatan tertentu. Tujuan pengujian ini adalah untuk memastikan bahwa robot mampu mengatasi berbagai rintangan yang mungkin dihadapinya selama operasional di lingkungan nyata. Pengujian rintangan ini juga membantu dalam mengidentifikasi potensi masalah dan meningkatkan performa keseluruhan

robot. Pengujian ini mengevaluasi sejauh mana sistem tersebut dapat berfungsi baik dalam situasi yang penuh tantangan, serta untuk mengidentifikasi masalah atau kelemahan yang mungkin muncul selama penggunaan normal.

Tabel 4.8 Pengujian Rintangan

No	Melewati Rintangan	Hasil Pengujian
1	Puing-Puing	Berhasil
2	Puing-Puing	Berhasil
3	Kelereng	Berhasil
4	Kelereng	Berhasil
5	Batu	Berhasil
6	Batu	Berhasil
7	Batu + Kelereng	Berhasil
8	Batu + Kelereng	Berhasil

Tabel diatas berisi informasi tentang hasil pengujian melewati berbagai jenis rintangan, seperti puing-puing, kelereng, dan batu. setiap kali pengujian dilakukan dengan melewati rintangan tertentu, hasilnya adalah "Berhasil". Terdapat beberapa jenis rintangan yang diuji, yaitu:

1. Puing-Puing: Pengujian melewati rintangan ini dilakukan dua kali, dan dalam kedua kasus hasilnya adalah "Berhasil".
2. Kelereng: Pengujian melewati rintangan ini juga dilakukan dua kali, dan dalam kedua kasus hasilnya adalah "Berhasil".
3. Batu: Pengujian melewati rintangan ini dilakukan tiga kali, dan dalam semua kasus hasilnya adalah "Berhasil".

4. Batu + Kelereng: Pengujian melewati kombinasi rintangan batu dan kelereng dilakukan dua kali, dan dalam kedua kasus hasilnya adalah "Berhasil".

Kesimpulan "Robot Bisa Melewati Beberapa Rintangan" adalah bahwa robot memiliki kemampuan untuk melewati atau mengatasi beberapa hambatan fisik atau rintangan yang ada di sekitarnya. Ini mencakup situasi di mana robot dapat bergerak, berinteraksi, atau menavigasi melalui berbagai jenis hambatan seperti penghalang fisik, tanah yang tidak rata, atau bahkan medan yang sulit dijangkau.

4.12. Pengujian PID *Tripod Gait* dan *Wave Gait*

Pengujian PID *(Proportional-Integral-Derivative)* pada T*ripod Gait* dan *Wave Gait* berkaitan dengan penggunaan kontroler PID untuk mengoptimalkan dan mengatur pergerakan sebuah robot atau sistem dengan dua jenis gait yang berbeda: *Tripod* dan *Wave.*

Pengujian PID pada *Tripod* bertujuan untuk menemukan parameter PID yang optimal untuk mengontrol gerakan kaki robot agar dapat bergerak dengan stabil, efisien, dan tepat di bawah kondisi yang berbeda, seperti berbagai kecepatan, beban, atau medan yang berbeda. Selain itu, pengujian juga akan melihat bagaimana robot beradaptasi dengan perubahan beban atau kondisi lingkungan.

Pengujian PID pada *Wave Gait* juga bertujuan untuk mencari parameter PID yang optimal agar robot dapat bergerak dengan lancar dan stabil. Pada *Wave Gait,* koordinasi antara kelompok kaki yang bergerak harus diatur dengan baik, dan kontroler PID dapat membantu mengoreksi dan memperbaiki setiap

perbedaan dalam gerakan kaki untuk menghasilkan pergerakan yang lebih efisien dan akurat.

Tabel 4.9 Pengujian PID *Tripod Gait* dan *Wave Gait*

Pengujian	Kp	Ki	Kd	Jumlah Benturan	Lama Waktu
1	4.3	0.3	1.0	3	4.30
2	4.4	0.4	0.9	2	4.26
3	4.5	0.5	0.8	2	4.23
4	4.6	0.6	0.7	1	4.21
5	4.7	0.7	0.6	1	4.20
6	4.8	0.8	0.1	2	4.24
7	4.9	0.9	0.4	1	4.19
8	5.0	1.0	0.1	0	4.16
9	5.1	1.1	0.2	0	4.18
10	5.2	1.2	0.	2	4.25

Tabel berisi hasil pengujian dua jenis pola gerakan ("*gait*") dari suatu sistem, yaitu "*tripod gait*" dan "*wave gait*". Dalam tabel tersebut, terdapat beberapa parameter pengaturan (Kp, Ki, Kd) serta hasil pengukuran yang dilakukan selama pengujian.

Berikut penjelasan dari tabel tersebut:

1. Pola Gerakan (*Gait*) **:**

   Tripod Gait : Pola gerakan *tripod* adalah salah satu pola gerakan yang umum digunakan dalam robotik berbasis kaki enam. Pada pola ini, tiga kaki di satu sisi tubuh (satu setengah tubuh) digunakan bersamaan, sedangkan sisi lainnya digunakan untuk menggerakkan kaki lainnya. Ini memberikan stabilitas yang baik.

Wave Gait : Pola gerakan gelombang adalah pola gerakan yang juga digunakan dalam robotik berbasis kaki enam. Pada pola ini, robot menggerakkan kaki-kaki di sisi yang sama secara berurutan, memberikan kesan gerakan seperti gelombang melalui tubuh robot. Ini juga menciptakan gerakan maju secara bertahap.

2. Parameter Pengaturan **:**

Kp, Ki, Kd **:** Ini adalah parameter-parameter pengendalian pada kontrol PID (*Proporsional-Integral-Derivative*) yang digunakan untuk mengontrol gerakan robot. Setiap parameter memiliki peranannya masing-masing dalam mengatur bagaimana sistem merespons perubahan input.

Kp (*Proportional Gain*): Mengontrol seberapa besar respons sistem terhadap perbedaan antara setpoint dan posisi aktual.

Ki (*Integral Gain*): Mengontrol pengaruh dari akumulasi kesalahan sebelumnya terhadap respons sistem.

Kd (*Derivative Gain*): Mengontrol seberapa cepat sistem merespons perubahan cepat dalam kesalahan.

3. Hasil Pengujian :

Jumlah Benturan : Ini mungkin mengacu pada jumlah kali robot mengalami tabrakan atau benturan selama pengujian.

Lama Waktu : Ini adalah waktu yang dibutuhkan selama pengujian untuk setiap konfigurasi parameter dan pola gerakan tertentu.

Dari tabel ini, melihat bahwa variasi parameter Kp, Ki, dan Kd serta pola gerakan berbeda-beda menghasilkan pengaruh yang berbeda terhadap jumlah benturan dan lama waktu selama pengujian

Kesimpulan pengujian *tripod gait* dan *wave gait* dimana saat tidak ada rintangan robot menggunakan pola langkah *tripod gait* dan jika terdapat rintangan maka robot menggunakan pola langkah *wave gait,* Pilihan antara tripod gait dan wave gait tergantung pada tujuan dan lingkungan aplikasi robot. Jika stabilitas dan efisiensi adalah prioritas utama, *tripod gait* mungkin lebih cocok. Namun, jika kecepatan dan kemampuan mengatasi medan yang sulit lebih penting, maka *wave gait* bisa menjadi pilihan yang lebih baik. Pemilihan gait juga dapat dipengaruhi oleh faktor-faktor teknis seperti konstruksi mekanik, kendali, Keduanya memiliki kelebihan dan kelemahan masing-masing, dan desain robot berjalan yang optimal mungkin memadukan aspek-aspek dari keduanya untuk mencapai kinerja terbaik dalam konteks tertentu.

4.13 Pengaruh PID Terhadap Nilai Dari *Tripod Gait* Dan *Wave Gait*

"PID" adalah singkatan dari "*Proportional-Integral-Derivative*," yang merupakan suatu metode umum dalam pengendalian otomatis untuk menjaga variabel tertentu pada nilai yang diinginkan melalui penyesuaian *proporsional, integral,* dan *derivatif* dari suatu tindakan kontrol. "*Tripod gait*" dan "*wave gait*" mengacu pada dua jenis gerakan yang sering digunakan dalam robotika untuk mengendalikan pergerakan robot berjalan.

1. *Tripod Gait* adalah pola gerakan di mana pada suatu waktu, tiga kaki dari robot (yang membentuk suatu *tripod*) berada di tanah untuk memberikan stabilitas,

sementara tiga kaki lainnya diangkat untuk bergerak maju atau mundur. Dalam hal ini, penggunaan PID dalam pengendalian pergerakan dapat membantu dalam mengatur kecepatan, arah, dan stabilitas gerakan robot saat berjalan dengan pola ini. Penggunaan PID dapat membantu mengoreksi perubahan dalam posisi dan orientasi robot secara real-time untuk menjaga gerakan yang lebih terkontrol dan stabil.

2. *Wave Gait* adalah pola gerakan di mana kaki-kaki robot bergerak seperti "gelombang" dari satu sisi ke sisi lainnya. Dalam kasus ini juga, penggunaan PID dapat membantu dalam mengontrol fase pergerakan kaki, frekuensi gelombang, dan stabilisasi gerakan secara keseluruhan. Pengendalian PID dapat membantu dalam menjaga keseimbangan dan koordinasi gerakan yang tepat.

Pengaruh PID terhadap nilai dari kedua pola gerakan ini adalah bahwa dengan menerapkan kontrol PID yang baik, robot dapat mencapai gerakan yang lebih stabil, presisi, dan terkontrol. PID dapat membantu robot dalam mengatasi gangguan eksternal, seperti perubahan beban atau permukaan yang tidak rata, dengan cepat mengoreksi posisi dan orientasi robot.

Namun, perlu diingat bahwa implementasi PID yang efektif memerlukan tuning yang cermat dan pemahaman mendalam tentang karakteristik mekanis robot dan lingkungan kerjanya. PID hanyalah salah satu aspek dalam mengendalikan gerakan robot. Terkadang, algoritma kontrol yang lebih kompleks juga digunakan untuk mengatasi tantangan yang lebih kompleks dalam pergerakan robot.

4.14 Perbedaan Menggunakan PID Dengan Tidak Menggunakan PID

PID adalah metode umum dalam kendali otomatis mengatur sistem agar mencapai tujuan tertentu. Berikut adalah perbedaan antara menggunakan dan tidak menggunakan kontrol PID pada robot *hexapod*:

1. Menggunakan PID pada Robot *Hexapod* :

   Stabilisasi Gerakan: Dengan menggunakan kontrol PID pada robot *hexapod*, dapat mengatur gerakan kaki dengan lebih akurat dan stabil. PID membantu mengurangi overshoot (kelebihan respons) dan settling time (waktu pemulihan) dari gerakan kaki robot, sehingga gerakan menjadi lebih halus dan lebih terkontrol.

   Perubahan Kondisi: Robot *hexapod* yang menggunakan kontrol PID dapat merespons perubahan kondisi lingkungan atau beban dengan lebih baik. PID akan memantau dan mengatur keseimbangan robot saat berjalan di permukaan yang tidak rata atau ketika membawa beban tambahan.

   Optimalisasi Kecepatan: Kontrol PID memungkinkan Anda untuk mengatur kecepatan gerakan robot secara tepat. Ini penting dalam situasi di mana robot perlu bergerak dengan cepat atau lambat tanpa mengorbankan kestabilan.

2. Tidak Menggunakan PID pada Robot *Hexapod*:

   Sederhana**:** Menghindari penggunaan kontrol PID dapat membuat sistem lebih sederhana. Jika robot hexapod hanya melakukan tugas-tugas dasar atau pergerakan terbatas, menggunakan kontrol sederhana mungkin sudah cukup.

Kurangnya Penyesuaian Otomatis: Tanpa kontrol PID, robot mungkin akan kesulitan dalam menyesuaikan gerakan dan keseimbangan saat menghadapi perubahan kondisi atau tuntutan.

Kurang Stabil dan Tidak Presisi**:** Tanpa kontrol PID, robot mungkin memiliki gerakan yang kurang stabil dan presisi. Ini dapat membatasi kemampuan robot dalam melakukan tugas yang membutuhkan keakuratan tinggi.

Kesimpulan:

Pemilihan apakah akan menggunakan kontrol PID pada robot hexapod tergantung pada kompleksitas tugas yang harus dilakukan oleh robot, tingkat akurasi yang diperlukan, dan lingkungan di mana robot akan beroperasi. Kontrol PID dapat memberikan keuntungan dalam hal stabilitas, presisi, dan respons terhadap perubahan, tetapi juga menambah tingkat kompleksitas dalam pengembangan dan pemrograman robot.

4.15 Pengaruh Sensor Ultrasonik kiri pada PID

Sensor ultrasonik kiri pada sebuah robot hexapod yang menggunakan kontrol *Proportional-Integral-Derivative* (PID) memiliki pengaruh yang signifikan terhadap kemampuan robot untuk mengikuti dinding dengan akurat. Pengaruh ini berkaitan dengan bagaimana sensor ultrasonik kiri membantu robot dalam memahami jarak antara dirinya dan dinding, serta bagaimana informasi tersebut digunakan dalam kendali PID.

Berikut adalah beberapa pengaruh utama dari sensor ultrasonik kiri pada kontrol PID saat robot *hexapod* mengikuti dinding:

1. *Feedback* Jarak : Sensor ultrasonik kiri memberikan umpan balik visual tentang jarak antara robot dan dinding yang ada di sebelah kirinya. Informasi ini penting bagi kontrol PID karena membantu dalam mengukur kesalahan (*error*) antara posisi yang diinginkan (sejajar dengan dinding) dan posisi aktual robot. Pengukuran jarak yang akurat memungkinkan PID untuk menghitung seberapa jauh robot terletak dari dinding.
2. Pengukuran *Error* : Sensor ultrasonik kiri membantu dalam mengukur error atau selisih antara posisi seharusnya dan posisi aktual robot terhadap dinding. Error ini diteruskan ke kontrol PID sebagai masukan, yang kemudian menghasilkan sinyal kendali untuk mengoreksi gerakan robot. Semakin akurat pengukuran error, semakin tepat pula respon kendali PID.
3. Menghindari Benturan : Sensor ultrasonik kiri juga dapat digunakan untuk mendeteksi rintangan yang mendekat atau potensial benturan dengan dinding. Ketika jarak antara robot dan dinding terlalu dekat, kontrol PID dapat merespons dengan mengubah arah atau menghentikan gerakan robot, mencegah benturan yang tidak diinginkan.

Kesimpulan diatas bahwa efektivitas penggunaan sensor ultrasonik kiri dalam kontrol PID kecepatan bacaan, algoritma kendali yang digunakan. Kombinasi yang baik antara sensor yang akurat dan algoritma PID yang efisien akan memberikan hasil yang optimal dalam mengikuti dinding dengan robot hexapod.

4.16 Pembahasan Pengujian Konstanta PID

Pengujian konstanta PID pada robot *hexapod* adalah proses penting dalam mengoptimalkan kinerja pergerakan robot. yang merupakan metode kontrol umum yang digunakan untuk mengatur posisi, kecepatan, atau pergerakan suatu sistem. Pada robot hexapod, PID digunakan untuk mengatur gerakan kaki-kaki robot akan berjalan stabil.

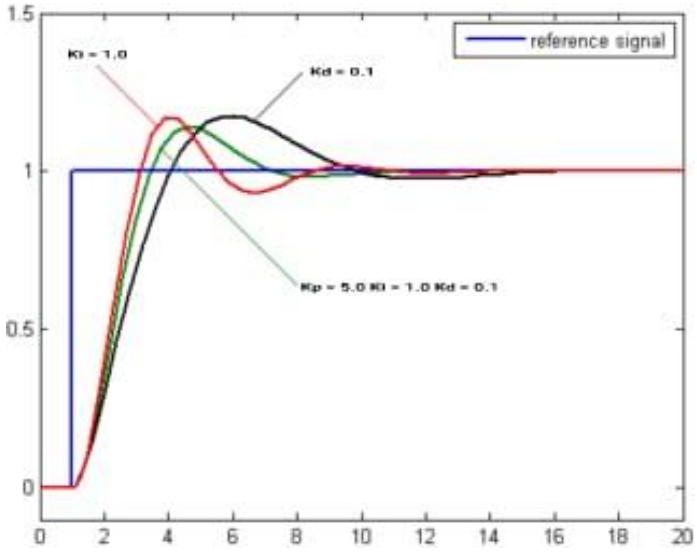
Pada pengujian konstanta PID, beberapa langkah umum yang dapat dilakukan meliputi:

1. Pengaturan Awal: Mulailah dengan mengatur nilai-nilai awal konstanta P (*proporsional*), I (*integral*), dan D (*diferensial*) ke nilai-nilai yang masuk akal atau yang umumnya digunakan sebagai titik awal. Nilai-nilai ini kemudian akan disesuaikan selama proses pengujian.
2. Pengujian *Proporsional* (P): Pertama, uji pengaruh konstanta P terhadap respons pergerakan. Naikkan nilai P hingga robot mulai bergerak. Jika nilai P terlalu rendah, robot mungkin tidak akan bergerak sama sekali atau gerakannya lamban. Namun, jika nilai P terlalu tinggi, robot dapat mengalami osilasi atau getaran berlebihan.
3. Pengujian *Integral* (I): Kemudian, uji pengaruh konstanta I terhadap kinerja robot. Nilai I digunakan untuk mengatasi error yang terjadi dalam jangka waktu lebih lama. Naikkan nilai I dan amati bagaimana robot menyesuaikan gerakannya terhadap perubahan lingkungan atau beban. Nilai I yang terlalu tinggi dapat menyebabkan overshoot atau peningkatan osilasi.

4. Pengujian *Diferensial* (D): Terakhir, uji pengaruh konstanta D. Konstanta D membantu meredam perubahan cepat dalam *error*. Naikkan nilai D dan perhatikan apakah robot mampu merespons perubahan secara lebih halus dan mengurangi osilasi. Nilai D yang terlalu tinggi bisa menghasilkan gerakan yang tidak stabil atau bereaksi terlalu kuat terhadap perubahan kecil.
5. Fine-Tuning dan Percobaan: Selama pengujian, terus sesuaikan konstanta P, I, dan D untuk mencapai kinerja terbaik. Biasanya, pengaturan optimal adalah hasil dari berbagai percobaan.

Pengujian konstanta PID pada robot *hexapod* merupakan proses *iteratif* dan mungkin memerlukan waktu untuk mencapai hasil yang memuaskan. Dengan melakukan pengujian yang cermat dan memahami bagaimana setiap konstanta mempengaruhi kinerja robot, sehingga dapat meningkatkan kemampuan robot *hexapod* untuk bergerak dengan stabil dan akurat.

4.17 *Trial Error* PID Berupa Grafik



Gambar 4.16 Grafik PID

Grafik *Proportional Integral Derivative* adalah cara untuk mengilustrasikan bagaimana pengontrol PID bekerja pada sistem tertentu, dalam hal ini robot hexapod. Grafik PID akan menunjukkan respons sistem terhadap kontrol PID dengan tiga komponen utama: P (*Proportional*), I (*Integral*), dan D (*Derivative*). Dengan nilai-nilai berikut (Kp = 5.0, Ki = 1.0, Kd = 0.1), bagaimana grafik PID pada robot *hexapod* dapat terlihat dengan nilai-nilai ini

1. Respon *Proportional* (P):

   Komponen P bertanggung jawab untuk menanggapi kesalahan saat ini antara posisi yang diinginkan dan posisi aktual robot. Semakin besar Kp, semakin besar respons P terhadap kesalahan.

   Dengan Kp = 5.0, respons P akan kuat, sehingga robot akan merespons cepat terhadap kesalahan saat ini. Namun, jika Kp terlalu besar, bisa menyebabkan overshoot (melewati nilai yang diinginkan) dan osilasi.

2. Respon *Integral* (I):

   Komponen I bertanggung jawab untuk menangani kesalahan akumulatif dari waktu ke waktu. Ini membantu menghilangkan kesalahan yang mungkin terakumulasi seiring waktu.

   Dengan Ki = 1.0, respons I akan cukup kuat untuk menghilangkan kesalahan yang terakumulasi dengan cepat.

3. Respon *Derivative* (D**):**

   Komponen D bertanggung jawab untuk meredakan laju perubahan kesalahan. Ini membantu menghindari osilasi dan overshoot.

Dengan Kd = 0.1, respons D akan memberikan peredaman yang cukup, tetapi tidak terlalu kuat. Ini membantu mencegah robot menjadi terlalu sensitif terhadap perubahan cepat dalam kesalahan.

Grafik respons PID akan tergantung pada situasi kontrol dan bagaimana robot hexapod diatur. Dalam situasi ideal, respons PID akan menghasilkan osilasi yang minim, waktu penyeimbangan yang cepat, dan akurasi yang baik.

4.18 *Tripod Gait* Dan *Wave Gait* Saat Berbelok

Tripod Gait dan *Wave Gait* adalah dua metode gerakan yang sering digunakan oleh *robot hexapod* (robot dengan enam kaki) saat berbelok. Kedua metode ini memungkinkan robot untuk mengubah arahnya dengan tetap menjaga keseimbangan dan efisiensi gerakan. Berikut masing-masing dari metode ini:

1. *Tripod Gait:*

Gaya *Tripod* (tiga kaki) adalah salah satu cara paling umum untuk robot hexapod berbelok. Pada dasarnya, robot ini memiliki tiga kaki di tanah (*tripod* statis) yang berfungsi sebagai penopang utama saat robot diam, sementara tiga kaki lainnya mengangkat dan melangkah maju bersama-sama. Ketika robot harus berbelok, *tripod* penopang berubah, yaitu tiga kaki yang tadinya bergerak berubah menjadi tiga kaki yang menopang, dan sebaliknya.

Misalnya, untuk berbelok ke kiri, kaki-kaki di sisi kanan akan diangkat dan ditaruh ke depan, sementara kaki-kaki di sisi kiri tetap menopang. Hal ini memberikan stabilitas saat berbelok, karena ada tiga kaki yang tetap menjaga keseimbangan. Kemudian, untuk bergerak maju lagi, kaki-kaki di sisi kiri yang tadinya menopang akan diangkat dan ditempatkan ke depan.

2. *Wave Gait:*

Gaya *Wave* (gelombang) melibatkan gerakan yang lebih halus dan lebih kompleks. Saat robot bergerak maju, kaki-kaki bergerak seolah-olah mengikuti pola gelombang yang bergerak dari depan ke belakang. Saat berbelok, gelombang ini diubah agar gerakan menjadi memutar. Ini dilakukan dengan mengubah fase gerakan gelombang pada kaki-kaki yang bergerak maju dan yang menopang.

Misalnya, untuk berbelok ke kiri, kaki-kaki yang berada di sisi kiri akan memiliki fase gerakan yang lebih cepat, sementara kaki-kaki di sisi kanan memiliki fase gerakan yang lebih lambat. Ini menghasilkan gerakan melingkar saat robot berbelok. *Wave Gait* cenderung lebih halus dan memberikan gerakan yang lebih efisien, tetapi mungkin membutuhkan algoritma yang lebih rumit untuk mengontrolnya.

Kedua metode tersebut ada kelebihan kekurangan dan pemilihan metode sesuai pada tujuan, kebutuhan lingkungan, dan desain robot. Pada dasarnya, baik *Tripod Gait* maupun *Wave Gait* bertujuan untuk menjaga keseimbangan dan efisiensi gerakan saat robot *hexapod* berbelok.

4.19 Pembahasan

Sistem Kontrol *Proporsional Integral Derivatif (PID)* dapat diterapkan pada navigasi *Wall Following Robot Hexapod* dalam *Tripod Gait* dan *Wave Gait* untuk memastikan robot mengikuti dinding dengan presisi yang baik. Dalam *Tripod Gait*, ketika berjalan 3 kaki yang bersentuhan sama tanah, PID dapat

digunakan untuk mengontrol sudut kemiringan dan posisi robot. Berikut adalah bagaimana setiap komponen PID dapat diterapkan:

1. *Proporsional* (P): Komponen ini mengukur perbedaan antara posisi robot sekarang dan posisi yang diinginkan. Misalnya, jika robot terlalu jauh dari dinding, proporsional akan menghasilkan sinyal yang mengarahkan robot mendekati dinding dengan cepat.
2. *Integral* (I): Komponen ini untuk menyesuaikan kesalahan sistem yang terakumulasi dalam waktu. kesalahan yang terakumulasi adalah perbedaan antara posisi robot sebenarnya dengan posisi yang diinginkan sepanjang waktu. Integral dapat membantu mengoreksi ketidakseimbangan sistem dan menghilangkan kesalahan statis yang mungkin terjadi.
3. *Derivatif* (D): Komponen ini mengukur perubahan cepat dalam kesalahan sistem. Jika robot bergerak terlalu cepat mendekati dinding, derivatif akan menghasilkan sinyal yang mengurangi kecepatan robot agar tidak terlalu mendekati dinding dengan cepat.

Dalam *Wave Gait*, ketika robot berjalan dengan pola perpindahan berjalan kaki yang kompleks, PID dapat digunakan untuk mengontrol posisi dan gerakan setiap kaki. Setiap komponen PID akan diatur sesuai dengan persyaratan gerakan dan koordinasi yang diinginkan.

Pada dasarnya, PID akan mengambil masukan dari sensor yang melacak posisi dan jarak robot terhadap dinding. Berdasarkan masukan ini, output PID akan mengendalikan motor atau servomotor yang menggerakkan kaki robot dengan cara yang sesuai untuk mencapai posisi dan pergerakan yang diinginkan.