Vol. 6, No. 2, August 2024, pp. 147~157, p-ISSN 2685-2381, e-ISSN 2715-2626
http://dx.doi.org/10.28989/avitec.v6i2.2304

Implementasi *Fuzzy Logic Controller* untuk Kendali Robot *Line Follower* berbasis *Computer Vision*

Fahril Maula Tanzil Huda[1], Fajri Saputra Rahmat[2], Damianus Cahyo Widi Nugroho[3]*, Merllya Penta Nurlita[4], Zaki Nugraha[5], Ardy Seto Priambodo[6]
Prodi Teknik Elektronika, Universitas Negeri Yogyakarta, Indonesia

Article Info

Article history:
Submitted June 27, 2024
Accepted August 6, 2024
Published August 13, 2024

Keywords:
Robot *line follower*, sistem kendali *fuzzy*, citra digital

Line follower robot, fuzzy control system, digital image

ABSTRACT

Robotika telah menjadi kebutuhan primer di bidang industri. Perkembangannya sangat cepat, bahkan saat ini fokus utama di bidang teknologi merupakan bidang robotika. Dalam pengembangannya, robot membutuhkan sebuah kontrol agar mampu dimanfaatkan dengan baik yang disebut sebagai sistem kendali. Penelitian ini bertujuan untuk membuktikan implementasi sistem kendali *fuzzy* dengan metode citra digital. Aplikasi yang dimanfaatkan pada penelitian ini adalah *software* Webots dengan robot e-puck, sehingga penelitian ini berfokus dengan simulasi. Sistem kendali *fuzzy* dengan metode citra digital terbukti mampu digunakan sebagai kontrol robot secara efisien pada dua skema jalur, A dan B. Hasil penelitian menunjukkan robot berhasil melewati kedua jalur dengan modus *error* 0. Selain itu, robot menghasilkan rata-rata nilai *error* sebesar -11,54 piksel dan -3,11 piksel pada masing-masing jalur. Hal ini menunjukkan *error* yang relatif kecil dan performa robot sangat baik.

Robotics has become a primary need in the industrial field. Its development is very fast, even now the main focus in the field of technology is the field of robotics. In its development, robots need a control to be able to be utilized properly which is called a control system. This research aims to prove the implementation of fuzzy control system with digital image method. The application utilized in this research is Webots software with an e-puck robot, so this research focuses on simulation. The fuzzy control system with digital image method is proven to be able to be used as an efficient robot control on two path schemes, A and B. The results showed that the robot successfully passed both paths. The results showed that the robot successfully passed both paths with an error mode of 0. In addition, the robot generated an average of -11.54 pixels and -3.11 pixels on each path, indicating a relatively small error and excellent robot performance.

Corresponding Author:
Damianus Cahyo Widi Nugroho,
Prodi Teknik Elektro dan Teknik Elektronika, Universitas Negeri Yogyakarta, Indonesia
Jl. Colombo No.1, Karangmalang, Kabupaten Sleman, Daerah Istimewa Yogyakarta, Indonesia
Email: *damianuscahyo.2022@student.uny.ac.id

1. PENDAHULUAN

Sistem kendali atau sistem kontrol adalah salah satu komponen utama dalam struktur robot. Fungsi sistem kendali pada aplikasi elektronika adalah memudahkan pengguna untuk memberikan perintah spesifik kepada robot [1]. Sistem kendali yang umum digunakan pada robot antara lain adalah sistem kendali *fuzzy*, sistem kendali PID (*Proportional-Integral-Derivative*) dan sistem kendali *hybrid*. Salah satu jenis robot yang mampu menggunakan berbagai sistem kendali tersebut adalah robot *line follower*.

Line follower merupakan jenis robot beroda yang bergerak mengikuti lintasan berupa garis (*line*) dengan daya penggerak berupa motor. Robot *line follower* biasanya digunakan dalam ajang perlombaan robot atau diaplikasikan untuk membantu pekerjaan manusia, seperti menangani tugas "*search and rescue*" [2]. Sistem kendali logika *fuzzy* digunakan pada robot *line follower* untuk mengenali jalur dengan variasi lebar antara satu hingga delapan titik sensor [3]. Namun, metode pembacaan delapan titik sensor tersebut mengakibatkan algoritma yang kompleks dan memerlukan waktu pemrosesan yang panjang. Untuk mengatasi masalah tersebut,

terdapat metode alternatif yang diimplementasikan yaitu penggunaan pengolahan citra digital sebagai sistem kendali robot *line follower*, dengan logika *fuzzy*.

Citra digital adalah susunan matriks dari persegi atau bujur sangkar piksel (*picture element*) yang terdiri dari baris dan kolom [4]. Selain itu, pemrosesan citra digital atau gambar dua dimensi menggunakan komputer disebut dengan pengolahan citra digital. Kemampuan komputasi komputer telah meningkat, dan rasio harga terhadap kinerja meningkat sistem komputer mengalami penurunan, sehingga memberikan kontribusi terhadap pengembangan komputer yang lebih luas aplikasi, manusia mempunyai indra penglihatan, yaitu salah satu sensor alami yang ditiru dalam perkembangan komputer yaitu kamera [5]. Gambar yang dihasilkan merupakan bentuk penyajian data-data yang dibutuhkan, dengan tujuan agar mudah dipahami dan dianalisis. Gambar memberikan data unik atau informasi khusus dari obyek yang ada di dalamnya [6]. Dengan informasi yang diterima, maka dilakukan berbagai pengolahan data yang berguna untuk sistem kendali robot *line follower*. Pemanfaatan gambar seperti ini termasuk dalam bidang *computer vision*. Dalam implementasinya, diperlukan sensor yang mampu menangkap hasil gambar dalam dua dimensi sehingga menampilkan garis atau lintasan yang digunakan pada robot *line follower*. Penggunaan sensor kamera ini layaknya mata bagi robot untuk melihat dan mengambil informasi. Lalu, informasi tersebut akan mempengaruhi pembuatan keputusan arah gerak robot, menggunakan sistem kendali *fuzzy* [7]. Dengan algoritma pengolahan citra yang tepat, mampu menghasilkan pola citra yang baik sebagai pengendali robot *line follower*. Oleh karena itu, algoritma *fuzzy* dipilih karena menjadi solusi untuk kemungkinan-kemungkinan yang belum teratasi oleh sistem kendali lainnya.

Algoritma *fuzzy* merupakan cara yang tepat untuk memetakan suatu ruang *input* ke dalam suatu ruang *output*, sehingga menghasilkan nilai yang kontinu. *Fuzzy* dinyatakan dalam derajat suatu keanggotaan serta derajat kebenaran. Oleh karena itu, sesuatu dapat dikatakan sebagian benar dan sebagian salah pada waktu yang sama. Logika *fuzzy* memungkinkan nilai keanggotaan berkisar antara 0 hingga 1 [8][9]. Salah satu keunggulan logika *fuzzy* adalah kemampuannya menangani ketidakpastian dan ambiguitas sistem yang kerap terjadi di dunia nyata. Hal ini disebabkan oleh logika *fuzzy* mampu menangani masukan yang tidak presisi. Oleh karena itu, logika *fuzzy* digunakan sebagai sistem kendali di berbagai bidang, termasuk dalam sistem kontrol robot *line follower* [10]. Beberapa sistem kontrol robot *line follower* memiliki masalah yang umum dijumpai. Salah satunya adalah mengatur parameter sistem kontrol secara manual, untuk memastikan kinerja robot optimal. Masalah ini membutuhkan model matematis yang mampu membuat proses pengerjaan menjadi panjang untuk mendapatkan konfigurasi optimal. Tujuannya adalah agar robot bergerak secara stabil dan akurat saat mengikuti jalur [11].

Penelitian ini bertujuan untuk melakukan simulasi sebuah robot *e-puck* pada *software* Webots sebagai robot *line follower*. Penulis merancang sistem kontrol robot *line follower* menggunakan logika *fuzzy* dan metode citra digital untuk memperoleh data simulasi. Selain itu, penulis hendak membuktikan bahwa robot *line follower* mampu bergerak mengikuti jalur secara stabil tanpa pengaturan parameter yang rumit.

2. METODE PENELITIAN

Penelitian diawali dengan melakukan studi literatur yang bertujuan untuk mencari referensi penelitian. Tahap selanjutnya adalah perancangan robot *line follower*, meliputi perancangan sistem robot *line follower*, perancangan lingkungan, pembuatan kontroler robot, hingga nantinya menghasilkan data yang dapat dianalisis. Fokus utama pada metode penelitian ini adalah performa kontroler dengan metode *fuzzy* yang digunakan oleh robot *line follower*.

2.1 Studi Literatur

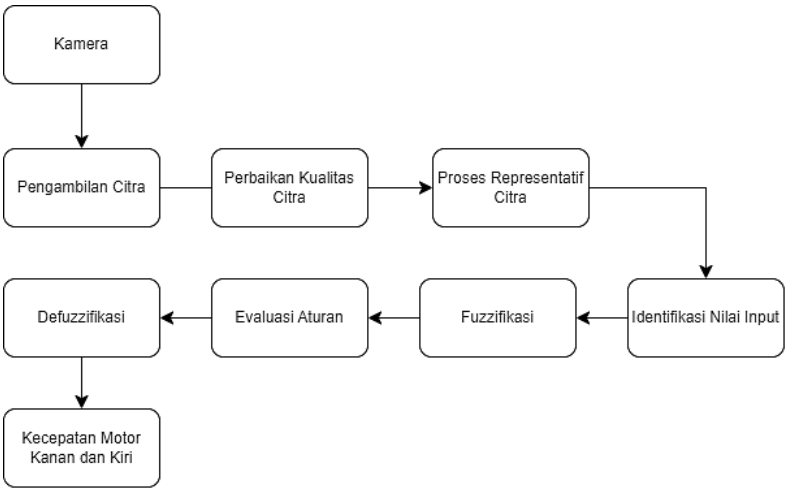
Studi literatur merupakan serangkaian kegiatan yang berkenaan dengan metode pengumpulan data pustaka, membaca dan mencatat, serta mengolah bahan penelitian [12]. Adapun sumber yang menjadi referensi adalah jurnal, buku dan artikel yang didapatkan dari internet. Studi literatur juga diperlukan dalam mencari bukti atau sebagai pendukung argumen atau penelitian yang telah dilakukan.

2.2 Sistem Robot *Line Follower*

Sistem kerja dari robot *line follower* telah digambarkan pada Gambar 1. Sistem robot *line follower* meliputi tiga bagian yaitu masukan dari pembacaan kamera, proses *fuzzy* dari pengolahan gambar hingga defuzzifikasi, dan keluaran berupa kecepatan motor kanan dan kecepatan motor kiri.

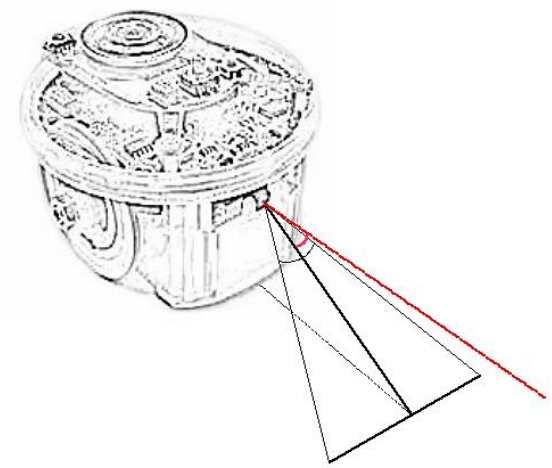
Gambar 1 merupakan diagram blok dari sistem robot *line follower*. Di sistem ini, robot memiliki kamera sebagai alat pengambilan gambar, lalu masuk ke tahap pengolahan gambar di mana citra digital diproses. Proses pengolahan gambar meliputi pengambilan citra, perbaikan kualitas citra, dan proses representatif citra sebelum dijadikan sebagai masukan. Awalnya, kamera akan mengeksekusi pengambilan gambar atau citra secara terus-menerus agar robot mendapatkan informasi tentang jalur yang diikuti. Setiap gambar yang dihasilkan akan diperbaiki kualitasnya, agar data gambar dapat diolah dengan baik. Perbaikan kualitas citra melibatkan proses peningkatan kontras, penyesuaian kecerahan, dan pengurangan *noise*. Setelah itu, gambar dengan kualitas yang baik akan dilakukan proses *threshold* , yaitu mengubah citra menjadi biner (hitam dan putih). Tujuan dari tahap *threshold* adalah agar gambar jalur robot dapat diolah dan dibaca dengan jelas, sehingga menghasilkan data

sebagai nilai masukan. Nilai masukan yang dihasilkan berbentuk nilai *crisp*, maka proses fuzzifikasi diperlukan untuk mengonversi nilai *crisp* menjadi nilai *fuzzy*.



Gambar 1. Diagram blok sistem robot *line follower*

Nilai *fuzzy* kemudian dievaluasi oleh sistem berdasarkan aturan-aturan *fuzzy* yang telah ditentukan. Hasil dari evaluasi tersebut melalui proses defuzzifikasi, mengubah nilai masukan *fuzzy* kembali menjadi nilai *crisp* yang konkret. Pada akhirnya, motor kanan dan kiri robot akan menggunakan nilai *crisp* hasil proses defuzzifikasi. Dengan begitu, robot mampu menyesuaikan kecepatan untuk mengikuti jalur sesuai dengan nilai keluaran defuzzifikasi. Gambar 2 adalah robot yang digunakan untuk penelitian.



Gambar 2. E-*puck* dan posisi kamera e-*puck*

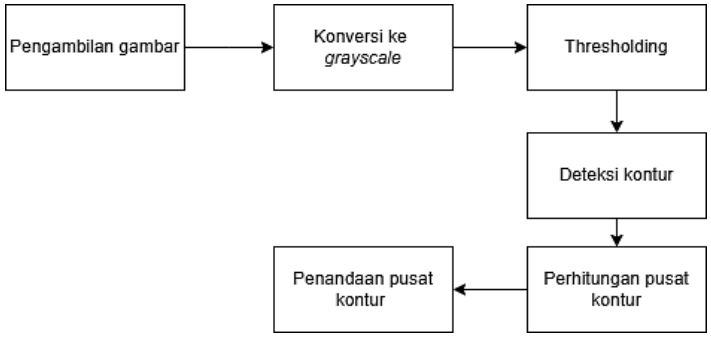
Gambar 2 menunjukkan bagaimana robot dengan kamera depan menggunakan garis hitam dan merah untuk navigasi. Garis hitam menggambarkan batas-batas bidang pandang kamera, sementara garis merah menunjukkan arah pusat pandang atau objek yang diamati. Segitiga yang terbentuk dari garis-garis ini mengilustrasikan area spesifik dalam pandangan kamera, yang dapat digunakan untuk mendeteksi garis. Gambar 2 juga menggambarkan bahwa robot menggunakan teknik pemrosesan citra untuk memahami lingkungannya.

2.3 Pengolahan Citra Digital

Pengolahan citra digital telah mengalami banyak perkembangan yang signifikan, seperti dalam meningkatkan resolusi rendah pada gambar CT scan, MRI, gambar geografis, gambar yang diterima di ponsel dan dari satelit. Teknologi ini digunakan untuk mengambil sampel ulang gambar baik untuk mengurangi atau meningkatkan resolusi [13]. Kualitas gambar yang diproses bergantung pada teknik interpolasi yang diadopsi [14]. Dalam penelitian ini, kamera digunakan sebagai sensor untuk mendapatkan gambar yang kemudian diolah oleh sistem.

Gambar 3 menunjukkan alur pengolahan citra digital. Pengolahan citra digital dimulai dengan pengambilan gambar menggunakan kamera pada e-*puck*. Gambar yang diambil kemudian diubah menjadi abu-abu (*grayscale*) untuk menyederhanakan informasi visual. Selanjutnya, gambar grayscale diubah menjadi gambar biner menggunakan metode *thresholding*. *Thresholding* adalah proses perbandingan piksel dengan nilai *threshold*. Di mana piksel dengan intensitas di atas *threshold* diubah menjadi putih (1), sedangkan piksel dengan

intensitas di bawah *threshold* diubah menjadi hitam (0). Pada gambar biner tersebut, dilakukan identifikasi untuk mendeteksi kontur. Pusat dari kontur terbesar kemudian dihitung dengan momen geometris yang memberikan informasi tentang bentuk dan posisi kontur. Pusat kontur dihitung dari sumbu x (Cx) dan sumbu y (Cy) dengan Persamaan (1).



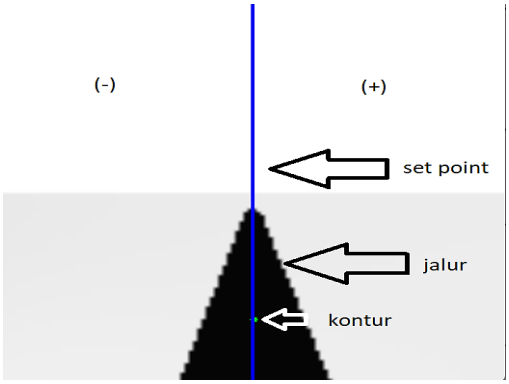
Gambar 3. Diagram blok proses pengolahan citra digital

$$Cx = \frac{M_{10}}{M_{00}}, cY = \frac{M_{01}}{M_{00}} \quad (1)$$

Momen orde-nol (M_{00}) adalah jumlah total piksel dalam kontur, sementara momen orde satu dalam sumbu x (M_{10}) dan sumbu y (M_{01}) menghitung distribusi piksel dalam arah horizontal dan vertikal. Dengan cara ini, koordinat rata-rata dari semua piksel dalam kontur bisa didapatkan, yang merupakan pusat dari kontur tersebut. Setelah itu, pusat kontur ditandai pada gambar asli dengan sebuah lingkaran hijau kecil untuk memudahkan visualisasi dan pengambilan data. Informasi posisi jalur relatif terhadap robot digunakan untuk menghasilkan kecepatan motor kiri dan motor kanan.

2.4 *Error* dan Delta *Error*

Gambar 4 menampilkan posisi garis vertikal berwarna biru sebagai *set point* yang berada di tengah gambar dan titik hijau sebagai pusat kontur. Hasil tangkapan gambar dari kamera memiliki lebar 420 piksel dan tinggi 640 piksel, sehingga titik tengah vertikalnya berada pada piksel 210.



Gambar 4. Tangkapan gambar dari kamera

Error (*E*) diperoleh dari selisih antara posisi aktual pusat kontur (titik hijau) dengan set point (garis biru). *Error* dapat bernilai positif atau negatif. Bernilai positif jika nilai *error* pada posisi kanan *set point*. Sementara itu *error* bernilai negatif pada posisi kiri *set point*. Persamaan (2) digunakan untuk mendapatkan nilai *error*.

$$E = Cx - 210 \quad (2)$$

Di mana 210 adalah titik tengah yang diinginkan (*set point*). Sedangkan *delta error* (*ΔE*) merupakan selisih *error* sekarang dengan *error* sebelumnya [9]. *Delta error* dapat dihitung dengan Persamaan (3).

$$\Delta E = E_n - E_{n-1} \quad (3)$$

Delta error diperlukan untuk mengetahui perubahan nilai *error* dari satu iterasi ke iterasi berikutnya sehingga dapat diketahui seberapa cepat posisi robot berubah.

Error dan *delta error* ini kemudian digunakan sebagai masukan untuk sistem kendali *fuzzy*. Sistem kendali *fuzzy* menggunakan *rule base* yang telah didefinisikan sebelumnya untuk menentukan kecepatan motor kiri dan motor kanan (LMS dan RMS).

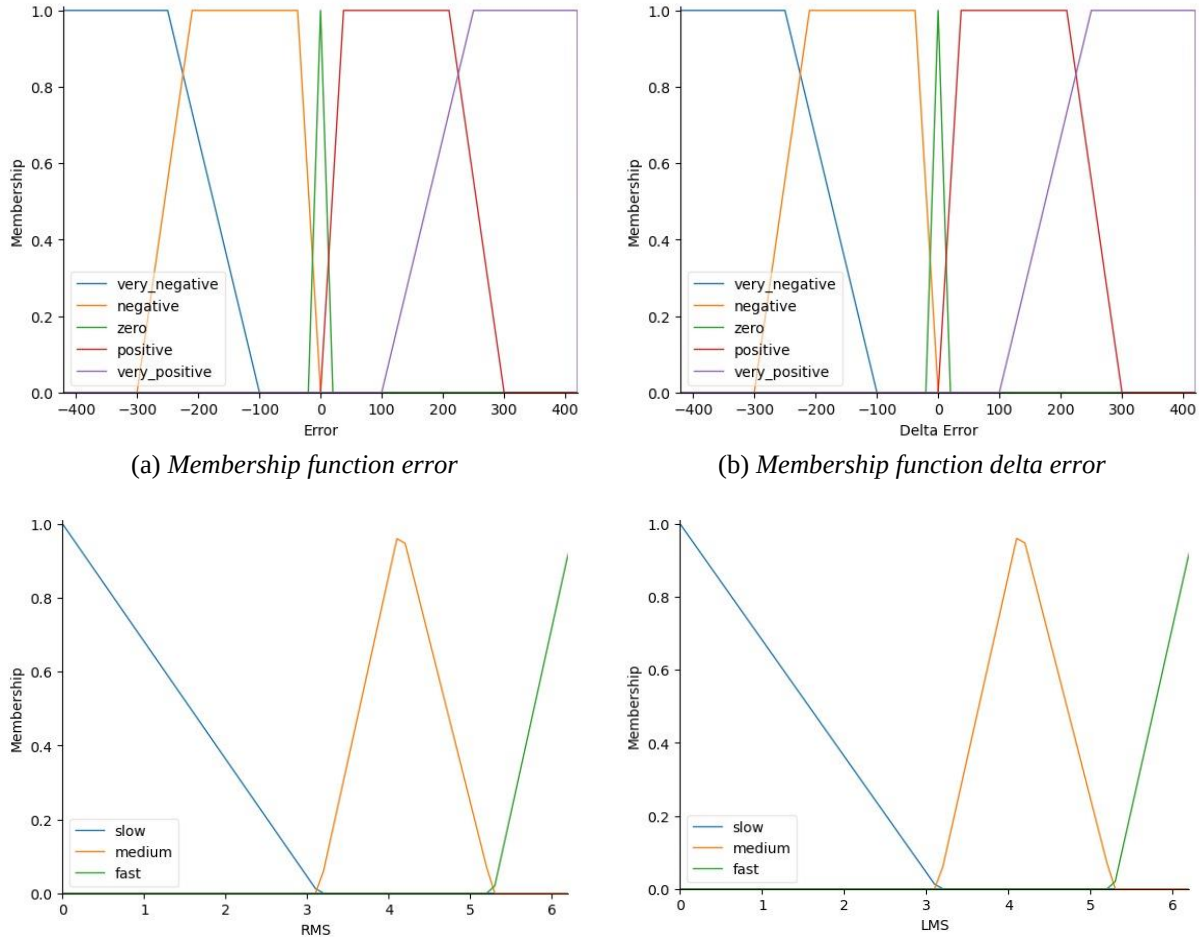
2.5 Kendali Fuzzy

Logika fuzzy digunakan dalam penelitian ini untuk merancang sistem robot *line follower* dengan metode Mamdani. Metode Mamdani merupakan salah satu metode logika *fuzzy* yang paling banyak digunakan karena pendekatannya yang intuitif dan mirip dengan cara berpikir manusia. Metode logika *fuzzy* Mamdani dikembangkan oleh Ebrahim Mamdani pada tahun 1975 [12]. Penggunaan metode logika *fuzzy* Mamdani bertujuan untuk mempermudah perancangan sistem kendali logika *fuzzy* pada robot *line follower*.

Perancangan sistem kendali logika *fuzzy* diperlukan untuk memberikan tingkatan-tingkatan dalam pembuatan program yang akan menjadi otak dari robot dalam menjalankan misinya. Sistem terlebih dahulu menerima masukan dari sensor kamera yang selanjutnya dijadikan masukan dalam kontroler logika *fuzzy*. Masukan dari sensor tersebut kemudian diproses melalui serangkaian proses logika *fuzzy*.

Proses dalam logika *fuzzy* sebagai kendali diperlukan beberapa tahapan yaitu fuzzifikasi, evaluasi aturan, dan defuzzifikasi. Tahap fuzzifikasi diperlukan untuk memproses masukan *crisp* atau variabel linguistik dari pembacaan sensor menjadi masukan dalam bentuk nilai linguistik pada *fuzzy set.* Data masukan numerik diproses ke dalam representasi linguistik melalui *fuzzy set* dengan nilai yang berkisar antara 0 hingga 1. Nilai linguistik merupakan *membership function* dalam logika *fuzzy* yang memiliki beberapa bentuk fungsi seperti triangular, trapezoidal dan gaussian.

Fuzzy set yang digunakan terdiri dari dua variabel masukan dan dua variabel keluaran yang diwakili oleh *membership function* berbentuk kurva *triangular* dan *trapezoidal*. Gambar 5 menunjukkan *membership function* menggunakan kombinasi kurva *triangular* dan *trapezoidal*. Terdapat 4 *membership function*, yaitu *error, delta error,* RMS (*Right Motor Speed*), dan LMS (*Left Motor Speed*).



(a) *Membership function error* (b) *Membership function delta error*

(c) *Membership function RMS* (*Right Motor Speed*) (d) *Membership function LMS* (*Left Motor Speed*)

Gambar 5. *Membership function*

Pada variabel masukan, "*error*" dan "*delta error*" memiliki lima himpunan *fuzzy* yang telah didefinisikan. Himpunan *fuzzy* "*very_negative*" memiliki rentang nilai di bawah -400 sampai -100, dengan menggunakan *membership function* berbentuk kurva *trapezoidal*. Himpunan fuzzy "*negative*" memiliki rentang nilai -300 sampai 0, menggunakan *membership function* serupa. Untuk nilai tengah, digunakan himpunan *fuzzy* "*zero*" dengan nilai puncak 0 dengan *membership function* berbentuk kurva *triangular*. Sementara itu, himpunan

fuzzy "*positive*" memiliki rentang nilai 0 sampai 300, dan terakhir, himpunan *fuzzy* "very_positive" memiliki rentang nilai 100 sampai 400, dengan bentuk kurva serupa dengan "*very negative*" dan *"negative*".

Variabel keluaran yang digunakan adalah kecepatan motor, yaitu RMS dan LMS dengan 3 himpunan *fuzzy*. Himpunan *fuzzy* "slow" memiliki rentang nilai 0 sampai 3, himpunan *fuzzy* "medium" memiliki rentang nilai 3 sampai 5, dan himpunan *fuzzy* "fast" memiliki rentang nilai ≥ 5. Ketiga himpunan *fuzzy* yang dipakai pada variabel keluaran menggunakan *membership function* berbentuk kurva *triangular*.

Penggunaan *membership function* berbentuk kurva *triangular* dan t*rapezoidal* bertujuan untuk memberikan kemudahan dalam implementasi kendali logika *fuzzy* pada robot *line follower*. *Membership function triangular* dan *trapezoidal* memiliki bentuk yang sederhana dan mudah untuk direpresentasikan secara matematis.

Tahapan setelah dilakukan fuzzifikasi adalah pembuatan basis aturan *(rule base*) yang akan digunakan dalam mekanisme inferensi logika *fuzzy*. Basis aturan ini terdiri dari serangkaian aturan yang menghubungkan antara nilai-nilai masukan (*error* dan *delta error*) dengan nilai-nilai keluaran (RMS dan LMS) yang diinginkan. Tabel 1 menunjukkan basis aturan (*rule base*) sebanyak 25 basis aturan yang digunakan dalam sistem kontrol yang dipakai.

Tabel 1. *Rule base*

No	Masukan		Keluaran	
	Error	*Delta Error*	LMS	RMS
1.	*Very_negative*	*Very_negative*	*Fast*	*Slow*
2.	*Very_negative*	*Negative*	*Fast*	*Slow*
3.	*Very_negative*	*Zero*	*Fast*	*Medium*
4.	*Very_negative*	*Positive*	*Medium*	*Fast*
5.	*Very_negative*	*Very_positive*	*Slow*	*Fast*
6.	*Negative*	*Very_negative*	*Fast*	*Slow*
7.	*Negative*	*Negative*	*Fast*	*Slow*
8.	*Negative*	*Zero*	*Medium*	*Slow*
9.	*Negative*	*Positive*	*Medium*	*Medium*
10.	*Negative*	*Very_positive*	*Slow*	*Fast*
11.	*Zero*	*Very_negative*	*Fast*	*Medium*
12.	*Zero*	*Negative*	*Fast*	*Medium*
13.	*Zero*	*Zero*	*Medium*	*Medium*
14.	*Zero*	*Positive*	*Medium*	*Fast*
15.	*Zero*	*Very_positive*	*Slow*	*Medium*
16.	*Positive*	*Very_negative*	*Medium*	*Fast*
17.	*Positive*	*Negative*	*Medium*	*Fast*
18.	*Positive*	*Zero*	*Medium*	*Medium*
19.	*Positive*	*Positive*	*Slow*	*Medium*
20.	*Positive*	*Very_positive*	*Slow*	*Fast*
21.	*Very_positive*	*Very_negative*	*Slow*	*Fast*
22.	*Very_positive*	*Negative*	*Slow*	*Fast*
23.	*Very_positive*	*Zero*	*Medium*	*Fast*
24.	*Very_positive*	*Positive*	*Medium*	*Medium*
25.	*Very_positive*	*Very_positive*	*Fast*	*Slow*

Tabel 1 menunjukkan *fuzzy rule* atau aturan dasar yang dimiliki oleh sistem kendali. Aturan dibuat dengan mempertimbangkan kemungkinan apa saja yang bisa terjadi. Aturan yang berlaku tergantung pada nilai masukan. Tidak ada batasan juga mengenai jumlah maksimal aturan yang berlaku. Dengan kondisi tersebut, robot akan bergerak mengikuti segala kondisi yang ada pada jalurnya.

Tahap selanjutnya adalah *inference* (inferensi), *aggregation* (agregasi), dan *defuzzification* (defuzzifikasi). Dalam tahap *inference*, kombinasi aturan *fuzzy* yang relevan diterapkan pada nilai masukan, menghasilkan derajat kebenaran berdasarkan *membership functions* yang sesuai. Kemudian, hasil dari semua aturan yang relevan digabungkan dalam proses agregasi untuk membentuk satu set keluaran *fuzzy*. Terakhir, adalah proses defuzzifikasi. Metode yang dipakai adalah Mamdani, di mana keluaran *fuzzy* yang telah digabungkan diubah menjadi nilai *crisp* (nilai yang dapat digunakan oleh sistem kontrol, seperti kecepatan motor) menggunakan metode *centroid*. Pusat area dari keluaran *fuzzy* diambil sebagai nilai *crisp* dalam proses defuzzifikasi ini. Nilai *crisp* dari proses defuzzifikasi ini digunakan untuk mengatur kecepatan motor, memastikan bahwa robot dapat menyesuaikan jalurnya dengan baik sesuai kondisi *error* yang diidentifikasi. Defuzzifikasi dapat dihitung menggunakan Persamaan (4).

$$z* = \int \mu(z)\, z\, dz \,/\, \int \mu(z)\, dz \quad (4)$$

Di mana:

$z*$ = Nilai keluaran *crisp*
$\mu(z)$ = Fungsi keanggotaan agregat
z = Variabel keluaran
dz = Diferensial z

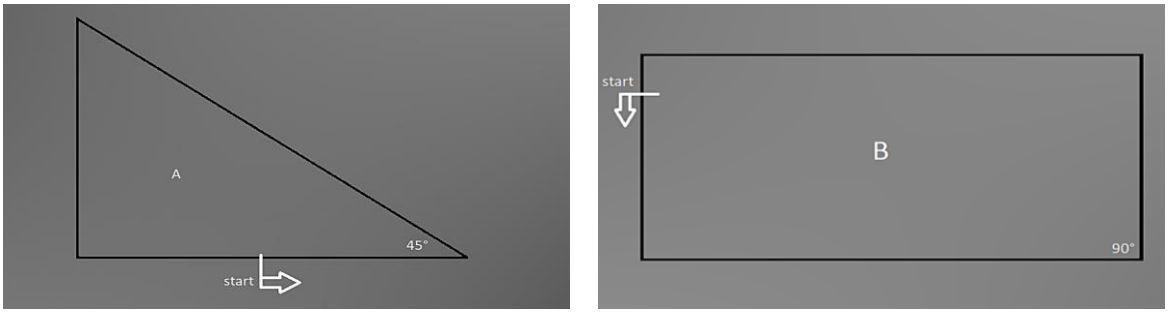
Persamaan (4) pada dasarnya membagi momen area (pembilang) dengan total area atau luas (penyebut). Integrasi pada $\int \mu(z)\, z\, dz$ menghitung momen dari area *fuzzy* dan $\int \mu(z)\, dz$ menghitung luas total area *fuzzy*. Hasil dari pembagian memberikan titik pusat atau *centroid* pada area *fuzzy*, yang menjadi nilai *crisp output*.

2.6 *Output* Sistem Kendali

Fuzzy rule digunakan untuk mengatur kecepatan motor berdasarkan kombinasi nilai *error* dan *delta error*. *Output* atau keluaran dari sistem kendali ini adalah kecepatan motor kiri (LMS) dan kecepatan motor kanan (RMS). Kecepatan motor maksimal adalah 6,28. Sebagai contoh, jika *error* sangat negatif dan *delta error* sangat negatif, maka RMS akan cepat dan LMS akan lambat. Selanjutnya, kecepatan motor dihitung menggunakan simulasi kontrol sistem *fuzzy* dengan memberikan nilai *error* dan *delta error* sebagai masukan, kemudian hasilnya dihitung untuk mendapatkan kecepatan motor kanan dan kiri.

2.7 Pengujian

Pengujian diawali dengan pembuatan lingkungan diantaranya, pembuatan jalur *line follower* dan penempatan robot e-puck pada *software* Webots. Jalur pengujian robot *line follower* dapat dilihat pada Gambar 6**.** Jalur A dengan sudut tajam yaitu 45°, dan jalur B sebagai jalur lurus dengan sudut 90°. Pengujian dilakukan pada simulasi dengan *software* Webots, dengan fokus pada performa robot saat bergerak di berbagai kondisi jalur. Semakin kecil nilai *error*, maka semakin baik performa robot *line follower*.



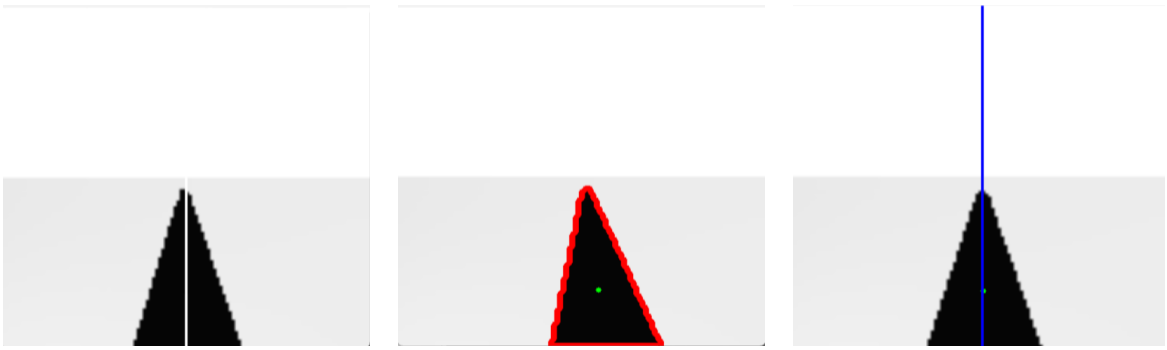
(a) Jalur pengujian A (b) Jalur pengujian B

Gambar 6. Jalur pengujian robot

Robot bergerak maju sesuai arah panah yang ada pada Gambar 6. Robot memiliki kecepatan awal 0, lalu akan berubah secara *real time.* Pengujian ini bertujuan untuk memastikan robot bekerja dengan performa terbaik di berbagai kondisi.

3. HASIL DAN PEMBAHASAN

Hasil dari pengujian berupa performa robot. Selama robot berjalan, didapatkan hasil tangkapan gambar dari kamera e-*puck*. Gambar yang diambil tersebut akan diproses seperti pada Gambar 7. Proses tersebut menghasilkan pusat kontur dan *set point*.



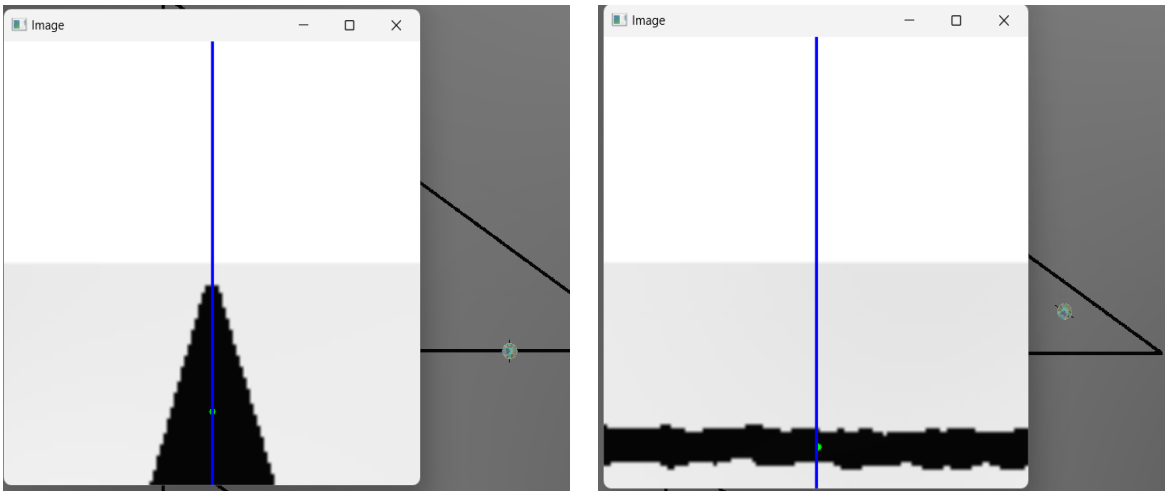
(a) Citra hitam dan putih (b) *Thresholding* jalur (c) Pusat kontur dan garis tengah

Gambar 7. Tahap pengambilan data citra

Berdasarkan percobaan, telah diperoleh grafik modus nilai *error* selama robot bergerak. Pengambilan gambar bermula dari gambar yang diubah menjadi hitam putih, lalu proses *thresholding* untuk menemukan pusat kontur dan dilakukan perhitungan *error* terhadap *set point*. Modus dan rata-rata *error* dapat merepresentasikan performa robot selama bergerak.

3.1 Hasil Pengujian Jalur A

Pengujian jalur dilakukan dengan menjalankan robot dari titik start hingga kembali ke titik start. Jalur A memiliki tiga sudut yaitu dua sudut 45° dan satu sudut 90°. Robot harus bisa melalui sudut tersebut dengan kecepatan yang sesuai.



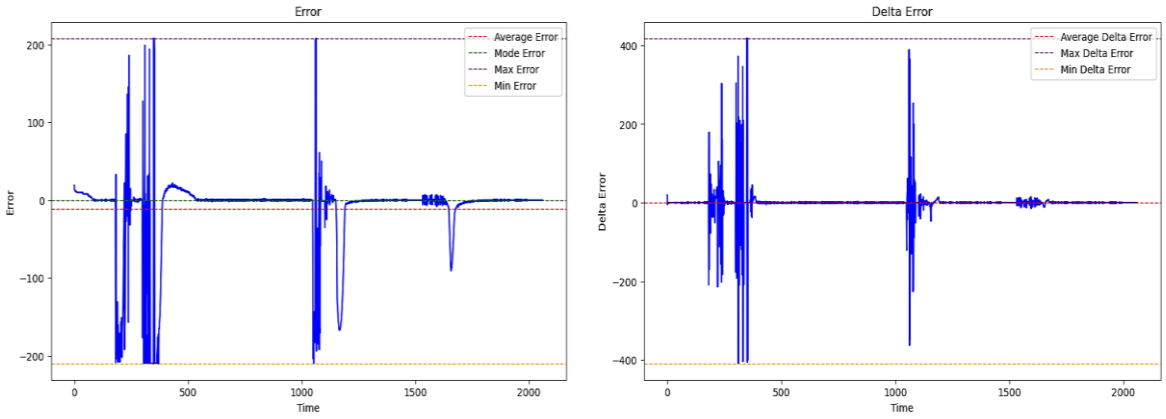
(a) Posisi robot dan tangkapan kamera ketika lurus (b) Posisi robot dan tangkapan kamera pada sudut 45°

Gambar 8. Posisi gerak robot ketika lurus dan berbelok

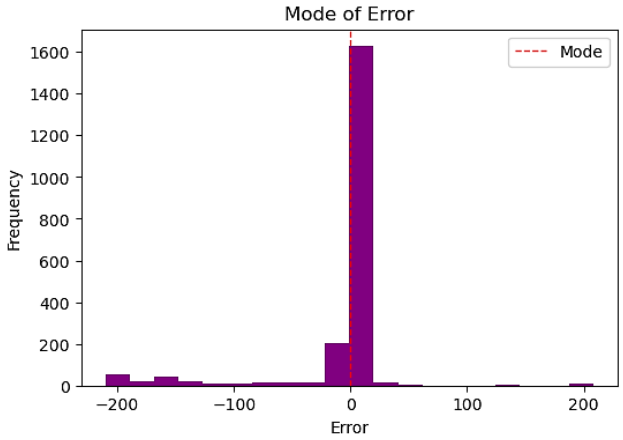
Gambar 8 menunjukkan pergerakan robot pada jalurnya. Awalnya robot bergerak dengan kecepatan maksimal pada jalur yang lurus, namun ketika bertemu dengan sudut 45°, robot menjadi tidak stabil. Ketidakstabilan tersebut memunculkan nilai *error* yang besar seperti pada Gambar 9. Ketika robot bertemu dengan sudut tersebut robot akan melakukan koreksi dengan menurunkan kecepatan pada salah satu motornya, pada kasus ini adalah motor kiri lebih lambat daripada motor kanan. Karena *error* bernilai negatif. Sebaliknya, jika *error* bernilai positif, maka motor kanan akan lebih lambat daripada motor kiri.

Gambar 9 menampilkan grafik dari nilai *error* dan *delta error* selama robot bergerak. Kedua grafik menunjukkan *overshoot* dan *underdamped* yang terjadi karena robot menjadi tidak stabil pada sudut 45°. Rata-rata nilai *error* yang tercatat adalah sebesar -11,54 piksel. Dengan *error* tertingginya adalah 208 dan *error* terkecilnya adalah -210. Dapat dilihat bahwa robot mengalami *error* yang tinggi pada beberapa kondisi. Hal ini disebabkan oleh kamera robot yang langsung mencari titik tengah jalur yang ada di depannya. Ketika jalur memiliki sudut, titik tengah jalur yang dideteksi oleh kamera akan semakin jauh dari sudut jalur, sehingga nilai *error* pun meningkat secara tajam. Tetapi robot dengan segera melakukan koreksi sehingga robot tetap pada jalurnya.

Gambar 10 menunjukkan modus nilai *error* yang muncul ketika robot bergerak pada jalur A. Pada gambar, nilai modus *error* adalah 0. *Error* yang minimum menunjukkan bahwa robot sangat jarang melenceng dari jalur ideal, sehingga kontrol robot secara umum sudah cukup baik dalam menjaga robot dekat dengan jalur yang benar. Namun, adanya rata-rata *error* yang cukup tinggi, yaitu sebesar -11,54 piksel menunjukkan bahwa ada ketidakstabilan atau kesalahan yang cukup besar yang terjadi pada beberapa titik. Ketidakstabilan tersebut muncul karena bentuk jalur yang memiliki sudut yang tajam.



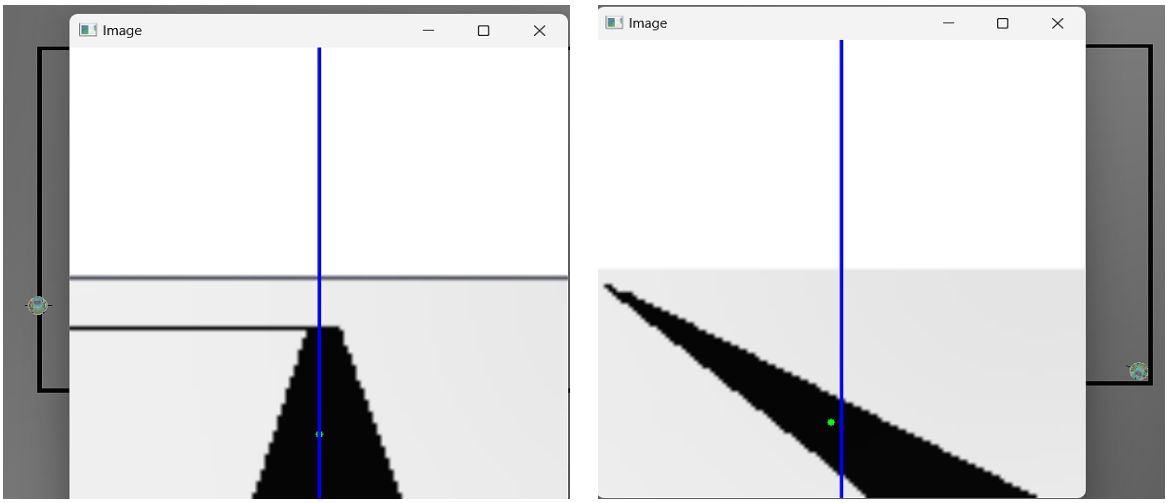
(a) Grafik nilai *error* (b) Grafik nilai *delta error*



Gambar 10. Modus error pada jalur A

3.2 Hasil Pengujian Jalur B

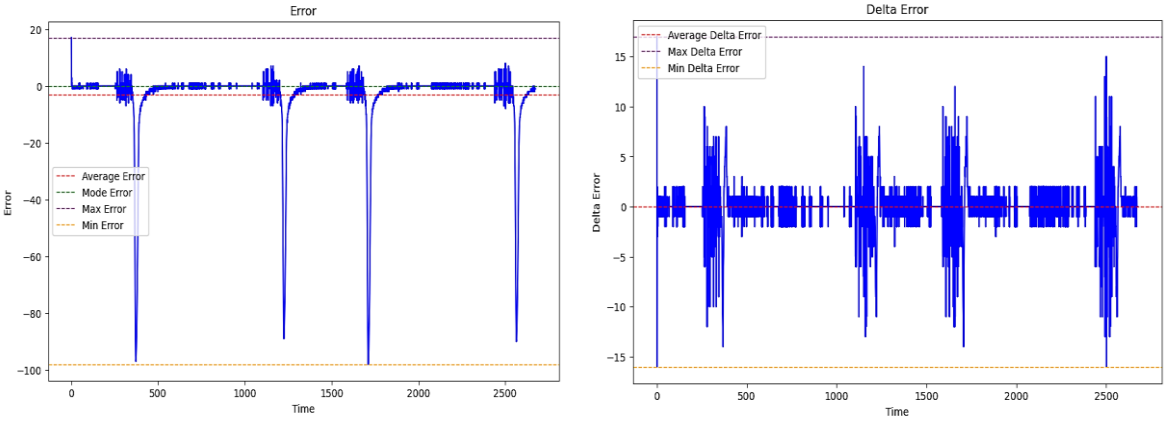
Percobaan yang serupa dilakukan menggunakan jalur B. Jalur B memiliki bentuk yang lebih sederhana dibandingkan dengan jalur A. Jalur B ini berbentuk lurus dengan belokan sebesar sudut 90°, baik ke kanan maupun ke kiri. Gambar 11 menunjukkan 2 sudut pandang kamera robot yang berbeda, yaitu posisi robot pada jalur lurus dan posisi robot pada jalur berbelok.



(a) Posisi robot dan tangkapan kamera ketika lurus (b) Posisi robot dan tangkapan kamera pada sudut 90°

Gambar 11. Posisi gerak robot ketika lurus dan berbelok

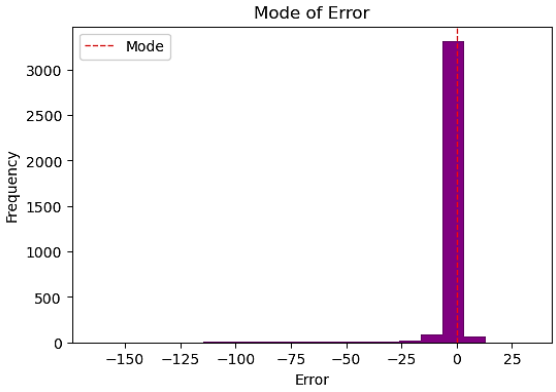
Nilai *error* yang muncul selama robot berada pada jalur lurus adalah 0 atau ±1. Hal tersebut menunjukkan bahwa robot telah bergerak dengan sangat baik pada jalur yang lurus. Akan tetapi, ketika robot berhadapan dengan sudut, akan muncul *error* dan *delta error* seperti pada grafik Gambar 12.



(a) Grafik nilai *error* (b) Grafik nilai delta *error*

Gambar 12. Nilai *error* dan delta *error* selama robot bergerak

Grafik pada Gambar 12 menunjukkan performa yang stabil dengan beberapa *overshoot* dan *underdamped* di berbagai titik. Nilai *error* terendah ada pada -98 dan tertingginya pada 17, ditunjukkan pada gambar sebelah kiri. Selain itu, nilai *delta error* menunjukkan perubahan dalam nilai *error* dari satu waktu ke waktu berikutnya, ditunjukkan pada gambar sebelah kanan. Perubahan nilai *error* yang drastis tersebut muncul karena robot berada pada kondisi jalur dengan sudut 90°. Ketika robot bertemu dengan sudut 90°, robot akan belok menyerong.



Gambar 13. Modus *error* robot pada jalur B

Gambar 13 memperlihatkan modus nilai *error* yang muncul ketika robot bergerak pada jalur B, menunjukkan performa robot pada jalur B yang sangat baik. Modus *error* yang bernilai 0 menyatakan bahwa robot biasanya berada tepat di jalur yang diinginkan, dengan rata-rata *error* yang kecil yaitu -3,11 piksel. Hal Ini berarti sistem kendali yang digunakan untuk jalur B sangat efektif dalam menjaga robot tetap berada di jalur, dengan penyimpangan rata-rata robot dari jalur sangat minimal. Tidak ada ketidakstabilan atau penyimpangan besar yang tercatat, sehingga robot mampu mengikuti jalur dengan sangat konsisten dan akurat. Performa robot pada jalur B dianggap optimal.

4. KESIMPULAN

Sistem kendali robot *line follower* dengan pengolahan citra digital dan algoritma *fuzzy* menunjukkan bahwa robot dapat beroperasi dengan performa baik di berbagai kondisi jalur. Sensor kamera menangkap gambar jalur hitam dengan latar belakang putih, dan data citra ini diolah menggunakan teknik pengolahan citra digital. Robot mampu melewati dua jenis jalur pengujian. Ketika nilai *error* kecil atau bahkan 0, robot akan melaju dengan kecepatan yang tinggi, sedangkan saat *error* besar, robot akan melakukan koreksi dengan menurunkan kecepatan di salah satu motornya. Sistem kontrol menunjukkan bahwa robot dapat bergerak pada jalur A dengan modus *error* 0. Meskipun terdapat rata-rata *error* sebesar -11,54 piksel yang menunjukkan adanya ketidakstabilan pada beberapa titik, namun secara umum robot hanya sedikit melenceng dari jalur ideal. Performa robot pun sangat baik pada jalur B dengan modus *error* 0 dan rata-rata *error* yaitu -3,11 piksel, menunjukkan penyimpangan yang minimal. Data ini mengindikasikan sistem kendali pada jalur B sangat efektif, hasilnya robot tetap berada di jalur dengan konsisten tanpa adanya penyimpangan besar. Setelah melakukan pengujian dan analisis di penelitian ini, terdapat beberapa saran yang harus dilakukan untuk penelitian selanjutnya agar lebih baik yaitu dengan mengoptimalkan algoritma pengolahan citra, seperti menggunakan teknik *adaptive*

thresholding atau filter tambahan agar dapat membantu dalam mendeteksi garis lebih akurat terutama pada kondisi pencahayaan yang bervariasi.

REFERENSI

[1] Adrie Sentosa, Djoko Purwanto, Rudy Dikairono, "Rancang Bangun Kendali Jarak Jauh Robot Servis Pembersih Debu Berbasis Internet of Things," *Jurnal Teknik ITS*, vol. 5, no. 2, 2016. http://dx.doi.org/10.12962/j23373539.v5i2.16281

[2] Purwanto, Imam Fauzi, Erni Yudaningtyas, "Pengaturan Kecepatan Motor Dalam Mempertahankan Batas Tepi Badan Robot Line Follower Terhadap Line Menggunakan Fuzzy Logic Controler," *Jurnal Mahasiswa Teknik Elektro Universitas Brawijaya*, vol. 2, no. 4, 2014.

[3] Putu Pratama, G., Dharmawan, A., Atmaji, C, "Implementasi Kendali Logika Fuzzy pada Robot Line Follower," *IJEIS* (*Indonesian Journal of Electronics and Instrumentation Systems*), vol. 4, no. 1, hal. 45-56, 2014. https://doi.org/10.22146/ijeis.4221

[4] A. Suryowinoto, A. Hamid, "Penggunaan Pengolahan Citra Digital dengan Algoritma Edge Detection dalam Mengidentifikasi Kerusakan Kontur Jalan," *dalam Seminar Nasional Sains dan Teknologi Terapan V*, Surabaya, 2017.

[5] Priambodo, A. S., F. Arifin, A. Nasuha, A. Winursito, "Face Tracking for Flying Robot Quadcopter Based on Haar Cascade Classifier and PID Controller," *Journal of Physics: Conference Series 2111*, no. 1, 2021. https://doi.org/10.1088/1742-6596/2111/1/012046

[6] I. A. Wicaksono, T. Winarno, A. Komarudin, "Implementasi Kontrol PID pada Gerakan robot line follower Berkaki Menggunakan Sensor Kamera," *Jurnal Elektronika dan Otomasi Industri*, vol. 7, no. 3, hal. 66, 2021.

[7] Hariyanto, Didik, "Pengolahan Citra Digital pada Sensor Kamera Sebagai Pengendali Arah Gerak Robot Line Follower," *Proceeding Seminar Nasional Pendidikan Teknik Elektro* (*SNPTE*), Universitas Negeri Yogyakarta, 2013.

[8] Rahakbauw, Dorteus L., "Implementasi Fuzzy C-means Clustering Dalam Penentuan Beasiswa," *BAREKENG: Jurnal Ilmu Matematika dan Terapan*, vol. 11, no. 1, 2017, hal. 1-12, 2017. https://doi.org/10.30598/barekengvol11iss1pp1-12

[9] Anggoro Mukti, Oky Dwi Nurhayati, Eko Didik Widianto, "Rancang Bangun Sistem Kontrol Robot Line Follower Menggunakan Logika Fuzzy," *Jurnal Teknologi dan Sistem Komputer*, vol. 3, no. 4, hal. 536, 2015. https://doi.org/10.14710/jtsiskom.3.4.2015.536-543 .

[10] A. Setiawan, N. A. Zakaria, A. Musafa, S. Sujono, "Perancangan Pembangkit Listrik Termoelektrik pada Proses Refrigerasi Air Conditioner dengan Metode Fuzzy Logic," *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, vol. 9, no. 1, hal. 1, 2021. https://doi.org/10.26760/elkomika.v9i1.1

[11] A. Wajiansyah, S. Supriadi, S. Nur, A. B. Wicaksono P., "Implementasi Fuzzy Logic pada Robot Line Follower," *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 5, no. 4, hal. 395–402, 2018. https://doi.org/10.25126/jtiik.201854747

[12] G. N. P. Pratama, A. Dharmawan, C. Atmaji, "Implementasi Kendali Logika Fuzzy pada Robot Line Follower," *IJEIS - Indonesian Journal of Electronics and Instrumentation Systems*, vol. 4, no. 1, 2014. https://doi.org/10.22146/ijeis.4221

[13] R. C. Gonzalez, R. E Woods, "Digital Image Processing, 2nd edition," Prentice Hall, hal. 272-274, 2002.

[14] C. Khare, K. K. Nagwanshi, "Image Restoration Technique with Non Linear Filter," *International Journal of Advanced Science and Technology*, vol. 39, 2012.