

PEMODELAN DINAMIKA DAN PERANCANGAN KENDALI LOGIKA FUZZY UNTUK ROBOT SELF-BALANCING RODA DUA

Wira Fadlun^{1*}, Ari Sugiharto²

^{1,2}Universitas Teknologi Yogyakarta; Jl. Siliwangi (Ringroad Utara), Jombor, Sleman, D.I. Yogyakarta 55285

Received: 6 Maret 2025
Accepted: 27 Maret 2025
Published: 14 April 2025

Keywords:
Robot Roda Dua;
Kendali Logika Fuzzy;
Pemodelan Sistem.

Correspondent Email:
wira.fadlun@staff.uty.ac.id

Abstrak. Robot *self-balancing* roda dua memiliki dinamika yang kompleks, sehingga menimbulkan tantangan signifikan dalam perancangan sistem kendalinya. Artikel ini mengusulkan penggunaan kendali logika fuzzy untuk mengendalikan robot tersebut. Model robot diformulasikan berdasarkan dinamika gerakannya dan direpresentasikan dalam bentuk persamaan *state-space*. Struktur kendali logika fuzzy terdiri dua masukan, yaitu error dan perubahan error, serta satu luaran berupa sinyal kendali yang menyesuaikan nilai torsi robot. Kinerja kendali logika fuzzy dievaluasi dan dibandingkan dengan Linear Quadratic Regulator (LQR) melalui simulasi yang dilakukan di MATLAB. Hasil pengujian kinerja kendali menunjukkan *steady-state error* antara sudut kemiringan aktual dan sudut referensi adalah nol untuk kendali logika fuzzy, sedangkan LQR memiliki error sebesar 0,05%. Selain itu, kendali logika fuzzy memiliki *settling time* sebesar 2,3 detik, dibandingkan dengan 2,5 detik untuk LQR. Hal ini menunjukkan bahwa kendali logika fuzzy dapat menyeimbangkan robot lebih cepat dibandingkan dengan LQR. Di bawah kendali logika fuzzy, robot bergerak sejauh 0,3 meter secara horizontal dari posisi awal sebelum kembali ke titik awalnya. Sebaliknya, dengan LQR, robot bergerak sejauh 0,35 meter sebelum kembali ke posisi awalnya. Hasil ini menunjukkan bahwa kendali logika fuzzy lebih unggul dibandingkan LQR dalam hal presisi dan stabilitas dalam menjaga posisi horizontal robot.

Abstract. The two-wheeled self-balancing robot exhibits complex dynamics, resulting in a significant challenge in the design of its control system. This paper proposes the use of fuzzy logic controller to control the robot. The robot model is formulated based on the dynamics of its movement and represented as state-space equations. The fuzzy logic controller structure comprises two inputs, namely error and change in error, and one output, which is the control signal that adjusts the robot's torque value. The performance of the fuzzy logic controller is evaluated and compared to the Linear Quadratic Regulator (LQR) through simulations conducted in MATLAB. The results show the steady-state error between the actual tilt angle and the reference angle is zero for the fuzzy logic controller, while the LQR has SSE of 0.05%. Furthermore, the fuzzy logic controller has a settling time of 2.3 seconds, compared to 2.5 seconds for the LQR. This indicates that the fuzzy logic controller can balance the robot faster than the LQR. With the fuzzy logic controller, the robot moves 0.3 meters horizontally from its initial position before successfully returning to its starting point. In contrast, under the LQR, the robot moves 0.35 meters before returning. These results indicate that the fuzzy logic controller outperforms the LQR in both precision and stability when maintaining the robot's horizontal position.

1. PENDAHULUAN

Robot *self-balancing* roda dua merupakan sebuah robot yang terdiri atas dua buah roda sejajar yang menopang badan robot. Robot tidak bisa berdiri secara seimbang tanpa adanya pengendali yang akan mengatur perputaran roda sehingga dapat mempertahankan keseimbangan. Robot ini memiliki beberapa kelebihan antara lain mobilitas dan kemampuan manuver yang tinggi serta mampu bekerja secara otonom. Tingkat kesulitan dalam merancang sistem kendali umumnya tergantung dari dinamika sistem yang akan dikendalikan. Robot jenis ini termasuk sistem yang tidak mudah untuk dikendalikan karena memiliki dinamika yang kompleks. Hal ini menjadi tantangan tersendiri bagi para peneliti. Beberapa metode kendali telah berhasil dirancang dan diimplementasikan untuk mengendalikan robot *self-balancing* roda dua, seperti PID [1], [2], [3], [4], [5], [6], [7], [8] dan Linear Quadratic Regulator (LQR) [9], [10], [11], [12], [13].

Kendali PID merupakan kendali yang cukup sederhana dan mudah diimplementasikan. Namun, memiliki permasalahan utama yaitu tidak mudah dalam melakukan *tuning* PID yang optimal terutama untuk sistem yang kompleks seperti robot *self-balancing* roda dua. Salah satu metode yang umum digunakan dalam melakukan *tuning* PID adalah *manual tuning* seperti yang dilakukan oleh [1], [2], [3]. Metode *tuning* yang lain adalah metode Ziegler-Nichols seperti yang digunakan pada penelitian [4] dan [5]. Peneliti lain berusaha menawarkan solusi untuk melakukan *tuning* yang optimal seperti yang dilakukan oleh [6] yang menerapkan metode Particle Swarm Optimization (PSO). Beberapa usulan lain yang terkait dengan upaya optimasi performa kendali PID adalah dengan menerapkan pengembangan model PID seperti *cascaded* PID seperti pada penelitian [7] dan 2-DOF PID pada [8].

LQR merupakan metode kendali yang dapat menstabilkan sistem dengan mencari nilai optimal dari *cost function* LQR dengan menentukan nilai parameter Q dan R yang tepat. LQR membutuhkan model sistem dalam bentuk persamaan linier. Jadi, LQR dapat diimplementasikan hanya untuk model sistem yang memiliki model yang akurat. Permasalahan lain dari LQR adalah penentuan nilai Q dan R umumnya masih secara *trial and*

error sehingga kurang efisien dari sisi waktu. Contoh penelitian perancangan LQR pada robot *self-balancing* roda dua yang menerapkan metode *trial and error* dalam menentukan nilai Q dan R adalah [9] dan [10]. Penelitian oleh [11] mengusulkan algoritma *reinforcement learning* untuk mencari nilai parameter LQR secara *on-line*. Penelitian lain mencoba menerapkan metode optimasi Bacteria Foraging Optimization Algorithms (BFOA) untuk meningkatkan performa LQR [12].

Beberapa penelitian membandingkan performa kendali PID dengan LQR seperti yang ditunjukkan pada [14] dan [15]. Pada penelitian oleh [14] menunjukkan bahwa LQR lebih unggul yang mana sudut kemiringan awal maksimum yang lebih besar dan nilai *settling time* yang lebih kecil. Begitu juga dengan penelitian oleh [15] yang menunjukkan keunggulan dari LQR dibandingkan PID. Hal ini terlihat dari hasil pengujian performa sistem kendali diperoleh nilai ISE, ITSE, IAE, dan ITAE dari LQR lebih kecil dibandingkan kendali PID.

Metode logika fuzzy memiliki beberapa kelebihan seperti penggunaan *reasoning* dan pengalaman manusia dalam membangun sistem kendali, tidak membutuhkan model matematis dinamika sistem, dan memiliki ketahanan dalam menangani gangguan dan kondisi tidak menentu seperti ketidakpastian nilai parameter. Oleh karena itu, pada artikel ini dirancang sistem kendali robot *self-balancing* roda dua menggunakan metode logika fuzzy. Tujuan penelitian pada artikel ini adalah mengetahui bagaimana performa sistem kendali logika fuzzy dalam mengendalikan robot *self-balancing* roda dua. Selanjutnya performa sistem kendali logika fuzzy dibandingkan dengan performa LQR untuk melihat keunggulan dari kendali logika fuzzy. Struktur kendali logika fuzzy yang dirancang terdiri dari dua buah *input* yaitu *error* dan perubahan *error* serta sebuah *output* yaitu sinyal kendali yang mengontrol nilai torsi robot.

2. TINJAUAN PUSTAKA

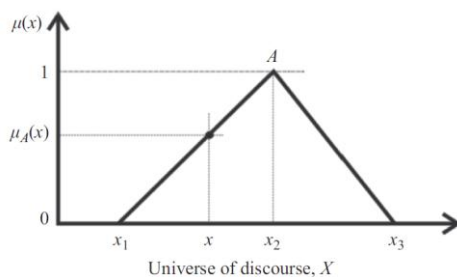
Himpunan fuzzy adalah perluasan dari konsep himpunan klasik dalam matematika yang diperkenalkan oleh Lotfi A. Zadeh pada tahun 1965. Berbeda dengan himpunan klasik yang memiliki batasan tegas (*crisp set*), himpunan fuzzy memungkinkan suatu elemen

memiliki tingkat keanggotaan dalam rentang 0 hingga 1, sehingga cocok digunakan untuk menangani ketidakpastian dan data yang tidak jelas.

2.1. Fungsi Keanggotaan Fuzzy

Fungsi keanggotaan yang mewakili himpunan fuzzy memiliki berbagai bentuk yang ditentukan oleh jenis rumus matematika tertentu. Jenis fungsi keanggotaan yang paling umum digunakan adalah segitiga, trapesium, lonceng, sinusoidal, Gaussian, Cauchy, dan sigmoid. Untuk mempermudah operasi pada himpunan fuzzy, fungsi keanggotaan diformulasikan berdasarkan parameter-parameter tertentu, yang mencakup informasi tentang derajat fuzzy dan rentang lokasi dalam semesta pembicaraan. Fleksibilitas dalam menyesuaikan parameter membuat fungsi keanggotaan dapat disesuaikan pula. Karena sifatnya yang linear, fungsi keanggotaan tipe segitiga dan trapesium lebih disukai dibandingkan jenis lainnya. Namun, dalam beberapa aplikasi, fungsi keanggotaan ditentukan dan disesuaikan oleh algoritma pembelajaran, seperti pada jaringan saraf tiruan, yang memerlukan turunan. Dalam kasus ini, fungsi dengan turunan yang lebih sederhana lebih dipilih.

Fungsi keanggotaan segitiga terdiri dari garis-garis seperti yang ditunjukkan pada Gambar 1 dan direalisasikan melalui kombinasi persamaan garis yang diberikan dalam Persamaan (1) [16].



Gambar 1. Fungsi Keanggotaan Fuzzy Segitiga

$$\mu_A(x) = \begin{cases} \frac{x-x_1}{x_2-x_1}, & \text{untuk } x_1 \leq x \leq x_2 \\ \frac{x_3-x}{x_3-x_2}, & \text{untuk } x_2 \leq x \leq x_3 \\ 0, & \text{untuk nilai lainnya} \end{cases} \quad (1)$$

Di mana parameter x_1 , x_2 , dan x_3 menentukan lokasi fungsi keanggotaan fuzzy A dalam *universe of discourse* (semesta

pembicaraan) X , seperti yang ditunjukkan pada Gambar 1. Sebenarnya, parameter x_1 , x_2 , dan x_3 merepresentasikan alamat dari fungsi keanggotaan A dan menunjukkan di mana fungsi tersebut berada dalam semesta yang bersangkutan.

Untuk mendefinisikan fungsi keanggotaan baru dengan bentuk serupa atau mengubah lokasinya dalam semesta pembicaraan, cukup dengan mengubah nilai parameter-parameter tersebut. Inilah alasan mengapa rumus terparameterisasi penting dalam merepresentasikan fungsi keanggotaan.

Persamaan (1) dapat digunakan sebagai fungsi keanggotaan terparameterisasi yang merepresentasikan himpunan fuzzy tipe segitiga.

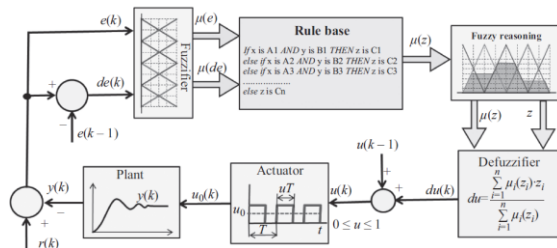
2.2. Fuzzy Logic Controller (FLC)

Penelitian tentang teori logika fuzzy telah berkembang pesat sejak pertama kali dikembangkan oleh Zadeh. Penerapan teori logika fuzzy pada sistem kontrol tekanan oleh Mamdani dan rekan-rekannya telah mendorong penggunaan logika fuzzy dalam aplikasi dunia nyata.

Prinsip kerja FLC serupa dengan proses pengambilan keputusan oleh operator manusia yang melakukan tugas yang sama. FLC menjalankan tindakan yang sama seperti operator manusia dengan menyesuaikan sinyal masukan berdasarkan hanya pada keluaran sistem. Arsitektur FLC sangat mirip dengan algoritma penalaran fuzzy. Algoritma yang paling umum digunakan untuk FLC adalah algoritma penalaran Mamdani.

Seperti halnya dalam arsitektur penalaran fuzzy Mamdani, FLC juga terdiri dari empat tahap: fuzzifier, basis aturan, konsekuensi, dan defuzzifier, sebagaimana ditunjukkan pada Gambar 2. Sinyal masukan dan keluaran FLC berbentuk nilai *crisp*, sehingga memerlukan proses fuzzifikasi dan defuzzifikasi. Oleh karena itu, sinyal utama dan perubahannya pada setiap sampel adalah masukan *crisp* yang dikonversi menjadi nilai keanggotaan fuzzy melalui fuzzifier. Nilai fuzzy ini kemudian digunakan dalam sistem basis aturan untuk menghasilkan konsekuensi fuzzy, yang selanjutnya dikombinasikan dalam proses penalaran fuzzy dan dikonversi menjadi keputusan *crisp* melalui proses yang disebut defuzzifikasi.

Pengendali berbasis logika fuzzy yang ditunjukkan pada Gambar 2 dirancang untuk bertindak sebagai pengendali integral, di mana konsekuensi *crisp* akhir dari FLC adalah perubahan $du(k)$, yang ditambahkan ke nilai sebelumnya $u(k-1)$ dari sinyal kendali $u(k)$ untuk menghasilkan keluaran saat ini, yaitu sinyal kendali yang diperbarui $u(k)$ [16].



Gambar 2. Struktur Fuzzy Logic Controller

Jika mengacu pada solusi digital dari sebuah integrator menggunakan metode integrasi Euler dapat direpresentasikan dalam Persamaan (2)

$$u(k) = u(k-1) + du(k) \quad (2)$$

Dalam integrasi digital, istilah $du(k)$ diekspresikan seperti pada Persamaan (3)

$$du(k) = K_I T e(k). \quad (3)$$

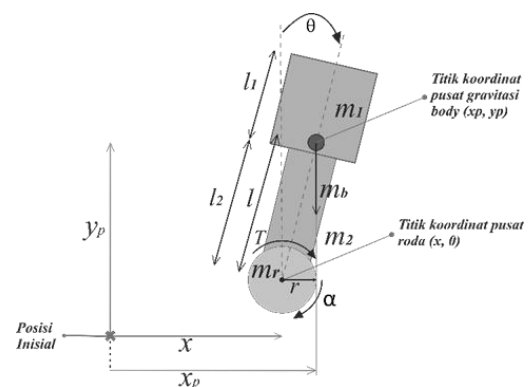
3. METODE PENELITIAN

Tahapan-tahapan yang dilakukan pada penelitian ini meliputi pemodelan sistem, perancangan sistem kendali, pengujian hasil perancangan sistem kendali melalui simulasi menggunakan perangkat lunak MATLAB serta analisis hasil simulasi. Pemodelan sistem bertujuan untuk merepresentasikan dinamika robot dalam bentuk persamaan matematis sehingga memudahkan dalam melakukan perancangan sistem kendali dan simulasi menggunakan MATLAB. Proses pemodelan sistem diawali dengan penurunan persamaan dinamika gerakan robot berdasarkan koordinat sistem yang telah didefinisikan. Selanjutnya, diturunkan persamaan Lagrangian sebagai pendekatan pemodelan sistem. Hasil penurunan persamaan sebelumnya masih *nonlinear* sehingga perlu diubah menjadi bentuk linier untuk memudahkan dalam menyelesaikan persamaan gerakan robot dengan mereduksi bagian *nonlinear* dari sistem. Persamaan hasil linierisasi diubah menjadi bentuk persamaan *state-space*. Perancangan kendali meliputi analisis tanggapan sistem orde dua terhadap

input sinyal *step*, perancangan kendali logika fuzzy yang mana penetapan aturan fuzzy diturunkan berdasarkan hasil analisis tanggapan sistem orde dua, dan perancangan blok simulasi pada MATLAB. Performa sistem kendali logika fuzzy diuji melalui simulasi menggunakan MATLAB. Performa kendali logika fuzzy dibandingkan dengan LQR. Pengujian terdiri atas pengujian kemampuan sistem kendali untuk mengendalikan robot dalam mempertahankan keseimbangannya (posisi tegak terhadap lantai) dan pengujian kemampuan sistem kendali untuk mengendalikan robot dalam mempertahankan posisinya secara horizontal pada posisi inisial.

3.1. Pemodelan Sistem

Model sistem umumnya dapat direpresentasikan dalam bentuk persamaan *differential*, *transfer function*, atau *state-space*. Pada penelitian ini, model robot direpresentasikan dalam bentuk persamaan *state-space* dengan menerapkan konsep Lagrangian. Robot *self-balancing* roda dua memiliki prinsip kerja seperti pendulum terbalik yang mana robot akan mempertahankan posisi vertikal terhadap lantai dengan mengatur nilai torsi yang memutar roda robot. *Free body diagram* robot *self-balancing* roda dua dapat dilihat pada Gambar 3.



Gambar 3. Free Body Diagram Robot Self-Balancing Roda Dua

Koordinat-koordinat pada robot yang ditunjukkan pada Gambar 1 dapat didefinisikan sebagai berikut:

- x : posisi horizontal dari pusat roda robot terhadap posisi inisial;
- x_p : posisi horizontal dari pusat massa badan robot terhadap posisi inisial;

y_p : posisi vertikal dari pusat massa badan robot terhadap lantai;

α : sudut angular roda robot terhadap sumbu horizontal saat $t = 0$;

θ : sudut angular badan robot terhadap sumbu vertikal (sudut kemiringan robot).

Nilai parameter robot ditunjukkan pada Tabel 1.

Tabel 1. Nilai Parameter Robot

Parameter	Keterangan	Nilai	Satuan
m_b	massa badan robot	0,513	kg
m_1	massa kepala robot	0,413	kg
m_2	massa poros tiang robot	0,1	kg
m_r	massa roda robot	0,0072	kg
r	radius roda robot	0,0315	m
l	jarak antara pusat massa terhadap pusat roda robot	0,0703	m
l_1	tinggi kepala robot	0,04	m
l_2	tinggi poros tiang robot	0,06	m
I_b	inersia badan robot	0,0027	kg.m ²
I_r	inersia roda robot	$7,792 \times 10^{-7}$	kg.m ²

3.1.1. Persamaan Dinamika Gerakan Robot

Posisi horizontal dari pusat roda robot terhadap posisi inisial (x), posisi horizontal dari pusat massa badan robot terhadap posisi inisial (x_p), dan posisi vertikal dari pusat massa badan robot terhadap lantai (y_p) secara berturut-turut dapat direpresentasikan ke dalam Persamaan (4), (5), dan (6).

$$x = r\alpha \quad (4)$$

$$x_p = r\alpha + l \sin \theta \quad (5)$$

$$y_p = r + l \cos \theta \quad (6)$$

Selanjutnya x , x_p , dan y_p diturunkan sehingga diperoleh Persamaan (7), (8), dan (9)

$$\dot{x} = r\dot{\alpha} \quad (7)$$

$$\dot{x}_p = r\dot{\alpha} + l\dot{\theta} \sin \theta \quad (8)$$

$$\dot{y}_p = -l\dot{\theta} \cos \theta \quad (9)$$

Persamaan energi potensial dan energi kinetik robot ditunjukkan pada Persamaan (10) dan (11)

$$E_p = m_b g(r + l \cos \theta) - m_b g(r + l) = m_b gL(\cos \theta - 1) \quad (10)$$

$$E_k = \frac{1}{2} m_r \dot{x}^2 + \frac{1}{2} I_r \dot{\alpha}^2 + \frac{1}{2} m_b \dot{x}_p^2 + \frac{1}{2} m_b \dot{y}_p^2 + \frac{1}{2} I \dot{\theta}^2 = \frac{1}{2} (I_r + m_r r^2 + m_b r^2) \dot{\alpha}^2 + m_b r l \cos \theta \dot{\alpha} \dot{\theta} + \frac{1}{2} (I_b + m_b l^2) \dot{\theta}^2 \quad (11)$$

3.1.2. Persamaan Lagrangian

Persamaan Lagrangian merupakan selisih antara energi kinetik dan energi potensial sehingga dapat dituliskan seperti pada Persamaan (12)

$$L = E_k - E_p \quad (12)$$

Persamaan (10) dan (11) disubstitusikan ke Persamaan (12) sehingga diperoleh Persamaan (13)

$$L = \frac{1}{2} (I_r + m_r r^2 + m_b r^2) \dot{\alpha}^2 + m_b r l \cos \theta \dot{\alpha} \dot{\theta} + \frac{1}{2} (I_b + m_b l^2) \dot{\theta}^2 - m_b g l (\cos \theta - 1) \quad (13)$$

Dari Persamaan (13) selanjutnya dapat dicari persamaan dinamika untuk koordinat α dan koordinat θ . Persamaan dinamika untuk koordinat α dapat diperoleh dengan melakukan diferensiasi Persamaan (13) terhadap $\dot{\alpha}$ dan α seperti terlihat pada Persamaan (14) dan (15)

$$\frac{\partial L}{\partial \dot{\alpha}} = (I_r + m_r r^2 + m_b r^2) \dot{\alpha} + m_b r l \cos \theta \dot{\theta} \quad (14)$$

$$\frac{\partial L}{\partial \alpha} = 0 \quad (15)$$

sehingga diperoleh Persamaan (16)

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\alpha}} \right) - \left(\frac{\partial L}{\partial \alpha} \right) = (I_r + m_r r^2 + m_b r^2) \ddot{\alpha} + m_b r l \cos \theta \ddot{\theta} - m_b r l \sin \theta \dot{\theta}^2 = \gamma_1 \quad (16)$$

γ_1 didefinisikan sebagai torsi pada koordinat α . Persamaan dinamika untuk koordinat α dapat diperoleh dengan melakukan diferensiasi Persamaan (13) terhadap $\dot{\theta}$ dan θ seperti terlihat pada Persamaan (17) dan (18)

$$\frac{\partial L}{\partial \dot{\theta}} = m_b r l \cos \theta \dot{\alpha} + (I_b + m_b l^2) \dot{\theta} \quad (17)$$

$$\frac{\partial L}{\partial \theta} = -m_b r l \sin \theta \dot{\alpha} \dot{\theta} + m_b g l \sin \theta \quad (18)$$

sehingga diperoleh Persamaan (19)

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\theta}} \right) - \left(\frac{\partial L}{\partial \theta} \right) = (I_b + m_b l^2) \ddot{\theta} + m_b r l \cos \theta \ddot{\alpha} - m_b g l \sin \theta = \gamma_2 \quad (19)$$

γ_2 didefinisikan sebagai torsi pada koordinat θ . Persamaan (16) dan (19) dapat disusun ulang menjadi Persamaan (20) dan (21).

$$(I_r + (m_r + m_b) r^2) \ddot{\alpha} + m_b r l \cos \theta \ddot{\theta} - m_b r l \sin \theta \dot{\theta}^2 = \gamma_1 \quad (20)$$

$$m_b r l \cos \theta \ddot{\alpha} + (I_b + m_b l^2) \ddot{\theta} - m_b g l \sin \theta = \gamma_2 \quad (21)$$

Torsi pada sistem juga dipengaruhi oleh nilai *rolling damping ratio* μ_r dan *friction damping ratio* μ_f sehingga persamaan torsi robot (T) digeneralisasi menjadi Persamaan (22) dan (23).

$$\gamma_1 = T - (\mu_r + \mu_f) \dot{\alpha} + \mu_f \dot{\theta} \quad (22)$$

$$\gamma_2 = -T + \mu_f \dot{\alpha} - \mu_f \dot{\theta} \quad (23)$$

Persamaan sistem diperoleh dengan mensubstitusikan Persamaan (22) ke (20) dan Persamaan (23) ke (21). Persamaan sistem ditunjukkan oleh Persamaan (24) dan (25)

$$(I_r + (m_r + m_b) r^2) \ddot{\alpha} + (\mu_r + \mu_f) \dot{\alpha} + m_b r l \cos \theta \ddot{\theta} - m_b r l \sin \theta \dot{\theta}^2 - \mu_f \dot{\theta} = T \quad (24)$$

$$m_b r l \cos \theta \ddot{\alpha} + (I_b + m_b l^2) \ddot{\theta} - \mu_f \dot{\alpha} + \mu_f \dot{\theta} - m_b g l \sin \theta = -T \quad (25)$$

Persamaan sistem tersebut masih dalam bentuk persamaan *nonlinear*.

3.1.3. Linearisasi

Proses linierisasi bertujuan untuk memudahkan dalam menyelesaikan persamaan gerakan robot dengan mereduksi bagian *nonlinear* dari sistem. Linierisasi sistem dilakukan dengan pendekatan kondisi robot diasumsikan berada pada posisi tegak terhadap sumbu vertikal (kondisi stabil) sehingga nilai kecepatan angular roda robot terhadap sumbu horizontal ($\dot{\alpha}$), sudut angular (θ) dan kecepatan angular badan robot terhadap sumbu vertikal ($\dot{\theta}$) masing-masing bernilai nol. Dengan demikian, nilai $\cos \theta \cong 1$ dan $\sin \theta \cong \theta$. Berdasarkan penyesuaian tersebut, Persamaan (24) dan (25) dapat ditulis menjadi bentuk linier seperti yang ditunjukkan pada Persamaan (26) dan (27)

$$(I_r + (m_r + m_b) r^2) \ddot{\alpha} + m_b r l \ddot{\theta} + (\mu_r + \mu_f) \dot{\alpha} - \mu_f \dot{\theta} = T \quad (26)$$

$$m_b r l \ddot{\alpha} + (I_b + m_b l^2) \ddot{\theta} - \mu_f \dot{\alpha} + \mu_f \dot{\theta} - m_b g l \theta = -T \quad (27)$$

Selanjutnya untuk membangun model *state-space* diperlukan persamaan untuk $\ddot{\alpha}$ dan $\ddot{\theta}$ dengan mengikuti beberapa penyesuaian persamaan berikut:

Persamaan (26) disusun ulang menjadi Persamaan (28)

$$\ddot{\alpha} = \frac{1}{I_r + (m_r + m_b) r^2} T - \frac{m_b r l}{I_r + (m_r + m_b) r^2} \ddot{\theta} - \frac{\mu_r + \mu_f}{I_r + (m_r + m_b) r^2} \dot{\alpha} + \frac{\mu_f}{I_r + (m_r + m_b) r^2} \dot{\theta} \quad (28)$$

Persamaan (27) disusun ulang menjadi Persamaan (29)

$$\ddot{\theta} = -\frac{1}{I_b + m_b l^2} T - \frac{m_b r l}{I_b + m_b l^2} \ddot{\alpha} + \frac{\mu_f}{I_b + m_b l^2} \dot{\alpha} - \frac{\mu_f}{I_b + m_b l^2} \dot{\theta} + \frac{m_b g l}{I_b + m_b l^2} \theta \quad (29)$$

Persamaan (29) disubstitusikan ke (28) sehingga diperoleh Persamaan (30)

$$\ddot{\alpha} = \frac{\left(\frac{\mu_f}{I_r + (m_r + m_b) r^2} - \frac{m_b^2 g r l^2}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2)} \right)}{Q} \dot{\theta} - \frac{\left(\frac{m_b r l \mu_f}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2)} + \frac{\mu_r + \mu_f}{I_r + (m_r + m_b) r^2} \right)}{Q} \dot{\alpha} \quad (30)$$

Persamaan (28) disubstitusikan ke (29) sehingga diperoleh Persamaan (31)

$$\ddot{\theta} = \frac{\left(\frac{m_b g l}{I_b + m_b l^2} \right)}{Q} \theta - \frac{\left(\frac{m_b r l \mu_f}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2)} + \frac{\mu_r + \mu_f}{I_b + m_b l^2} \right)}{Q} \dot{\theta} + \frac{\left(\frac{m_b r l (\mu_r + \mu_f)}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2)} + \frac{\mu_r + \mu_f}{I_b + m_b l^2} \right)}{Q} \dot{\alpha} \quad (31)$$

yang mana

$$Q = 1 - \frac{m_b^2 r^2 l^2}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2)} \quad (32)$$

3.1.4. Persamaan State-Space

Variabel matriks *state-space* didefinisikan seperti pada Persamaan (33)-(36)

$$x_1 = \theta \quad (33)$$

$$x_2 = \dot{\theta} \quad (34)$$

$$x_3 = \alpha \quad (35)$$

$$x_4 = \dot{\alpha} \quad (36)$$

Berdasarkan variabel yang telah didefinisikan sebelumnya, diperoleh turunan dari masing-masing variabel seperti ditunjukkan pada Persamaan (37)-(40)

$$\dot{x}_1 = x_2 \quad (37)$$

$$\dot{x}_2 = k_1 x_1 - k_2 x_2 + k_3 x_4 - k_6 u \quad (38)$$

$$\dot{x}_3 = x_4 \quad (39)$$

$$\dot{x}_4 = k_4 x_2 - k_5 x_4 + k_7 u \quad (40)$$

yang mana

$$k_1 = \frac{(m_b g l)}{Q} \quad (41)$$

$$k_2 = \frac{(m_b r l \mu_f)}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2) + I_b + m_b l^2} + \frac{\mu_r + \mu_f}{Q} \quad (42)$$

$$k_3 = \frac{(m_b r l (\mu_r + \mu_f))}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2) + I_b + m_b l^2} + \frac{\mu_r + \mu_f}{Q} \quad (43)$$

$$k_4 = \frac{(\frac{\mu_f}{I_r + (m_r + m_b) r^2} - \frac{m_b^2 g r l^2}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2)})}{Q} \quad (44)$$

$$k_5 = \frac{(\frac{m_b r l \mu_f}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2) + I_b + m_b l^2} + \frac{\mu_r + \mu_f}{I_r + (m_r + m_b) r^2})}{Q} \quad (45)$$

$$k_6 = \frac{(\frac{1}{I_b + m_b l^2} + \frac{m_b r l}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2)})}{Q} \quad (46)$$

$$k_7 = \frac{(\frac{1}{I_r + (m_r + m_b) r^2} + \frac{m_b r l}{(I_r + (m_r + m_b) r^2)(I_b + m_b l^2)})}{Q} \quad (47)$$

Selanjutnya dibangun persamaan sistem dalam bentuk *state-space* seperti yang ditunjukkan pada Persamaan (48) dan (49)

$$\dot{x} = Ax + Bu \quad (48)$$

$$y = Cx \quad (49)$$

Variabel input u merupakan nilai torsi T . Persamaan *state-space* secara detail dapat dilihat pada Persamaan (50) dan (51)

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ k_1 & -k_2 & 0 & k_3 \\ 0 & 0 & 0 & 1 \\ 0 & k_4 & 0 & -k_5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ -k_6 \\ 0 \\ k_7 \end{bmatrix} T \quad (50)$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & r & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad (51)$$

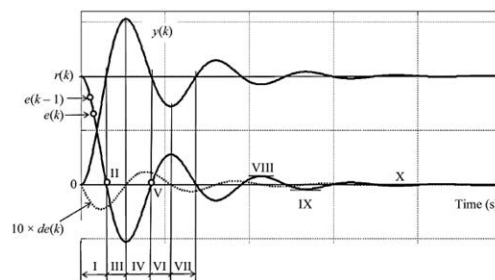
3.2. Perancangan Sistem Kendali

Logika fuzzy bekerja melalui tiga tahap yaitu fuzzifikasi, inferensi fuzzy, dan defuzzifikasi. Pada tahap fuzzifikasi, nilai input *crisp* (tegas) diubah menjadi nilai fuzzy dalam bentuk himpunan fuzzy. Selanjutnya pada tahap inferensi fuzzy, nilai fuzzy diolah dengan mengombinasikan basis aturan fuzzy untuk memperoleh *output* berupa nilai fuzzy baru. Tahap terakhir adalah tahap defuzzifikasi yaitu proses pengubahan nilai dari output fuzzy yang dihasilkan menjadi nilai *crisp* [17]. Kendali

logika fuzzy yang dirancang pada penelitian ini memiliki spesifikasi sebagai berikut:

- Fungsi keanggotaan fuzzy untuk *input* maupun *output* menggunakan fungsi keanggotaan segitiga.
- Metode yang diterapkan pada sistem inferensi fuzzy adalah metode Mamdani.
- Metode yang digunakan dalam proses defuzzifikasi adalah *centre of gravity* (*centroid*).

Pendekatan yang digunakan dalam merancang kendali logika fuzzy pada penelitian ini adalah dengan menganalisis tanggapan sistem orde dua terhadap input sinyal *step* seperti yang ditunjukkan pada Gambar 4 [16]. Hasil analisis tanggapan sistem tersebut dijadikan dasar dalam pembentukan tabel aturan fuzzy.



Gambar 4. Tanggapan Sistem Orde Dua

Pada Gambar 4 terdapat 10 daerah operasi yang menunjukkan pengaruh *error* $e(k)$ dan *change of error* $de(k)$ terhadap sinyal kendali $du(k)$. Hal tersebut dapat direpresentasikan ke dalam bentuk tabel seperti yang terlihat pada Tabel 2.

Tabel 2. Representasi Daerah Operasi Tanggapan Sistem Orde Dua

	Daerah operasi									
	I	II	III	IV	V	VI	VII	VIII	IX	X
$e(k)$	+	0	-	-	0	+	+	-	+	0
$de(k)$	-	-	-	+	+	+	-	0	0	0
$du(k)$	+	-	-	-	+	+	+	-	+	0

Berdasarkan Tabel 2 dapat diperoleh kesimpulan bahwa jika $e(k)$ bernilai nol, maka $du(k)$ mengikuti tanda $de(k)$, jika tidak, $du(k)$ mengikuti tanda $e(k)$. Tanda dari $e(k)$, $de(k)$ dan $du(k)$ tersebut dijadikan dasar dalam menentukan struktur aturan fuzzy (9-aturan fuzzy) seperti yang ditunjukkan pada Tabel 3.

Tabel 3. Kombinasi 9-Aturan Fuzzy

$e(k)$	$de(k)$		
	N	Z	P

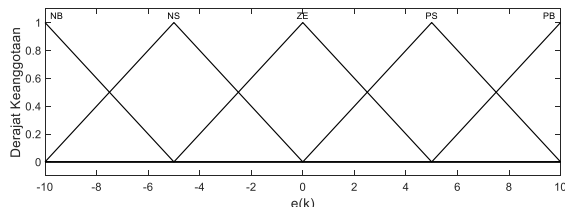
<i>P</i>	<i>P</i>	<i>P</i>	<i>P</i>
<i>Z</i>	<i>N</i>	<i>Z</i>	<i>P</i>
<i>N</i>	<i>N</i>	<i>N</i>	<i>N</i>

Pada Tabel 3 terdapat tiga buah himpunan fuzzy untuk masing-masing variabel input $e(k)$ dan $de(k)$ serta output $du(k)$ yaitu N (*Negative*), Z (*Zero*), dan P (*Positive*). Aturan fuzzy pada Tabel 3 dapat dikembangkan menjadi 25-aturan fuzzy dengan menguraikan sub-kategori N (*Negative*) menjadi NS (*Negative Small*) dan NB (*Negative Big*), begitu juga sub-kategori P (*Positive*) diuraikan menjadi PS (*Positive Small*) dan PB (*Positive Big*). Aturan fuzzy dengan total 25 buah aturan ditunjukkan pada Tabel 4.

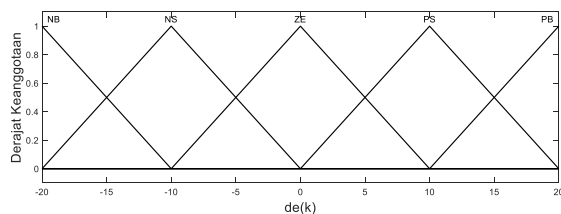
Tabel 4. Kombinasi 25-Aturan Fuzzy

$e(k)$	$de(k)$				
	NB	NS	ZE	PS	PB
PB	PB	PB	PB	PB	PB
PS	PS	PS	PS	PS	PS
ZE	NB	NS	ZE	PS	PB
NS	NS	NS	NS	NS	NS
NB	NB	NB	NB	NB	NB

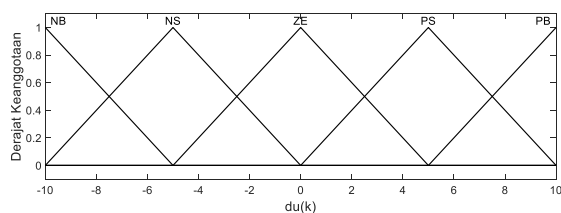
Fungsi keanggotaan input $e(k)$ dan $de(k)$ masing-masing ditunjukkan oleh Gambar 5 dan Gambar 6. Sedangkan fungsi keanggotaan output $du(k)$ ditunjukkan pada Gambar 7.



Gambar 5. Fungsi Keanggotaan Error $e(k)$

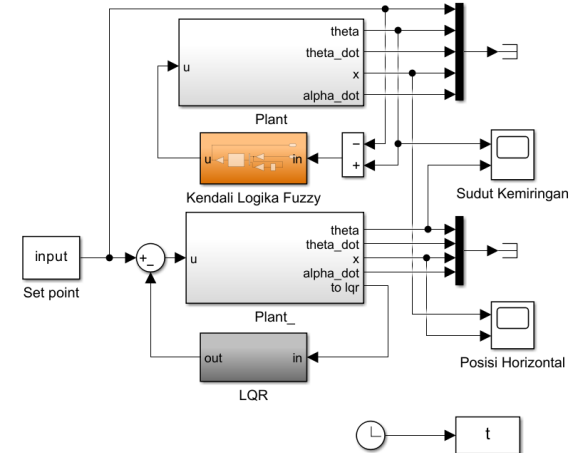


Gambar 6. Fungsi Keanggotaan Perubahan Error $de(k)$



Gambar 7. Fungsi Keanggotaan Output $du(k)$

Struktur sistem kendali untuk mengontrol robot yang dibangun menggunakan perangkat lunak MATLAB ditunjukkan pada Gambar 8.



Gambar 8. Struktur Sistem Kendali Robot Self-Balancing Roda Dua

Blok simulasi yang dirancang terdiri dari dua buah metode yaitu kendali logika fuzzy dan LQR yang bertujuan untuk membandingkan sistem yang telah dirancang pada penelitian ini yaitu kendali logika fuzzy dengan LQR. Parameter output yang dijadikan pembanding untuk performa kedua kendali adalah parameter sudut kemiringan (θ) dan posisi horizontal (x) dari robot. Input sistem adalah posisi *set point* yaitu posisi robot tegak lurus terhadap lantai ($\theta = 0$).

LQR dirancang untuk mencari nilai *cost function* yang optimal menggunakan Persamaan (52) [9].

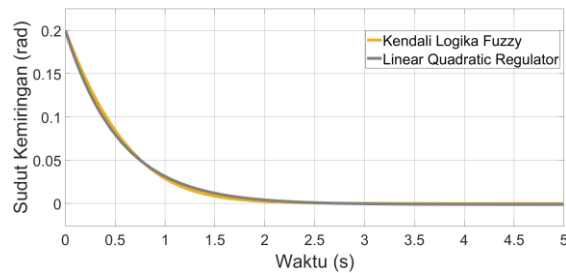
$$J = x^T(t_f)Sx(t_f) + \int_0^{t_f} (x^T(t)Qx(t) + u^T R u(t))dt \quad (52)$$

Nilai optimal dari *cost function* tergantung pada nilai Q dan R . LQR yang diimplementasikan pada penelitian ini memiliki nilai $Q = 10$ dan $R = 0,001$. Nilai Q dan R tersebut diperoleh menggunakan metode *trial and error* melalui proses simulasi secara berulang dengan mengganti nilai Q dan R sehingga mendapatkan nilai yang paling optimal.

4. HASIL DAN PEMBAHASAN

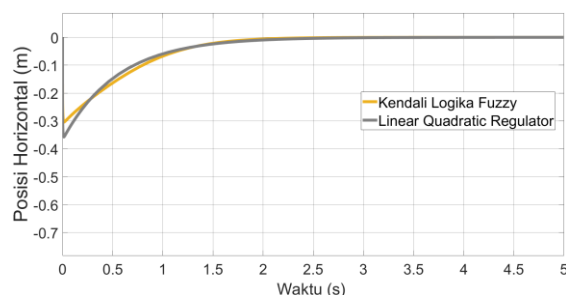
Kinerja kendali logika fuzzy dievaluasi dan dibandingkan dengan kinerja LQR melalui simulasi di MATLAB. Evaluasi ini menguji kemampuan masing-masing sistem kendali dalam mempertahankan keseimbangan robot

(posisi tegak relatif terhadap lantai) serta menjaga posisi horizontal robot pada *setpoint* awal. Pengujian ini bertujuan untuk membandingkan kinerja kendali logika fuzzy dan LQR dalam menjaga posisi tegak robot serta menstabilkan posisinya secara horizontal. Gambar 9 dan 10 menggambarkan perilaku robot terhadap waktu.



Gambar 9. Sudut Kemiringan Robot Terhadap Waktu

Pada Gambar 9, sudut kemiringan awal robot ditetapkan sebesar 0,2 radian (sekitar 11,46°) relatif terhadap sudut referensi. Grafik menunjukkan bahwa *steady-state error* (galat keadaan tunak) antara sudut kemiringan aktual dan sudut referensi adalah nol untuk kendali logika fuzzy, sedangkan LQR memiliki nilai *steady-state error* sebesar 0,05%. Hal ini menunjukkan bahwa kendali logika fuzzy lebih unggul dalam mempertahankan keseimbangan robot pada sudut target ($\theta = 0$) dibandingkan LQR. Selain itu, kendali logika fuzzy memiliki nilai *settling time* sebesar 2,3 detik, sedangkan pada LQR 2,5 detik. Ini mengindikasikan bahwa kendali logika fuzzy menstabilkan sistem lebih cepat, sehingga menunjukkan efisiensinya dalam menyeimbangkan robot.



Gambar 10. Posisi Horizontal Robot terhadap waktu

Gambar 10 menunjukkan bahwa dengan kendali logika fuzzy, robot bergerak sejauh 0,3 meter secara horizontal dari posisi awalnya sebelum berhasil kembali ke titik awal. Sebaliknya, dengan LQR, robot bergerak sejauh

0,35 meter sebelum kembali ke posisi awal. Hasil ini menunjukkan bahwa kendali logika fuzzy lebih unggul dibandingkan LQR dalam hal presisi dan stabilitas dalam mempertahankan posisi horizontal robot. Deviasi yang lebih kecil pada kendali logika fuzzy menunjukkan kinerja kendali yang lebih efektif dalam menjaga posisi robot.

5. KESIMPULAN

Dalam penelitian ini, model robot *self-balancing* roda dua diturunkan dari dinamika gerak robot dan direpresentasikan dalam bentuk persamaan *state-space*. Kendali logika fuzzy yang dikembangkan memiliki dua masukan yaitu *error* dan perubahan *error*, serta satu keluaran yaitu sinyal kendali yang mengatur torsi robot. Basis aturan fuzzy dibangun berdasarkan karakteristik tanggapan sistem orde dua ketika diberikan masukan berupa sinyal *step*. Kinerja sistem kendali logika fuzzy dievaluasi dan dibandingkan dengan LQR melalui simulasi MATLAB. Hasil pengujian menunjukkan bahwa kendali logika fuzzy memiliki kinerja lebih baik dibandingkan LQR. Secara spesifik, kendali logika fuzzy memiliki nilai *steady-state error* nol, sedangkan LQR memiliki nilai *steady-state error* sebesar 0,05%. Selain itu, kendali logika fuzzy memiliki nilai *settling time* yang lebih cepat, yaitu 2,3 detik dibandingkan 2,5 detik pada LQR. Dalam hal pergerakan horizontal, di bawah kendali logika fuzzy, robot bergerak sejauh 0,3 meter dari posisi awal sebelum kembali dengan sukses, sedangkan dengan LQR, robot bergerak sejauh 0,35 meter sebelum kembali ke posisi awal.

UCAPAN TERIMA KASIH

Tim penelitian menyampaikan terima kasih kepada Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi Republik Indonesia. Penelitian ini didanai oleh Direktorat Riset dan Pengabdian kepada Masyarakat sesuai dengan Kontrak Penelitian Tahun Tunggal 2020 dengan Lembaga Layanan Pendidikan Tinggi (LLDIKTI) Wilayah V berdasarkan Nomor Kontrak 081/SP2H/LT/DRPM/2020 dan 03.36/UTY-R/KLP-BP/VI/2020.

DAFTAR PUSTAKA

- [1] A. T. Ali, A. M. O. Mohamedy, A. S. A. Salim, E. A. O. M. El-Aminx, and O. M. K. Ahmed, "Design and Implementation of

- Two-Wheeled Self-Balancing Robot Using PID Controller,” in *Proceedings of: 2020 International Conference on Computer, Control, Electrical, and Electronics Engineering, ICCCEEE 2020*, Institute of Electrical and Electronics Engineers Inc., Feb. 2021.
- [2] I. Siradjuddin *et al.*, “PID controller for a differential drive robot balancing system,” in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Dec. 2019.
- [3] M. R. Islam, M. R. T. Hossain, and S. C. Banik, “Synchronizing of stabilizing platform mounted on a two-wheeled robot,” *Journal of Robotics and Control (JRC)*, vol. 2, no. 6, pp. 552–558, Nov. 2021.
- [4] A. F. M. Sulaiman, I. Siradjuddin, and N. R. Dzakiyullah, “The Design of a Microcontroller-based Self Balancing Robot Employing PID Control and Kalman Filter,” *Webology*, vol. 19, no. 1, pp. 1194–1206, Jan. 2022.
- [5] V. Mudeng, B. Hassanah, Y. Tonce, K. Priyanto, and O. Saputra, “Design and Simulation of Two-Wheeled Balancing Mobile Robot with PID Controller,” 2020.
- [6] Efrain Mendez-Flores *et al.*, “Self-Balancing Robot Control Optimization Using PSO,” in *The 5th International Conference on Control and Robotics Engineering*, 2020.
- [7] E. Philip and S. Golluri, “Implementation of an Autonomous Self-Balancing Robot Using Cascaded PID Strategy,” in *2020 6th International Conference on Control, Automation and Robotics, ICCAR 2020*, Institute of Electrical and Electronics Engineers Inc., Apr. 2020, pp. 74–79.
- [8] A. T. Azar, H. H. Ammar, M. H. Barakat, M. A. Saleh, and M. A. Abdelwahed, “Self-balancing Robot Modeling and Control Using Two Degree of Freedom PID Controller,” in *Advances in Intelligent Systems and Computing*, Springer Verlag, 2019, pp. 64–76.
- [9] M. Stanese, M. Susca, V. Mihaly, and I. Nascu, “Design and Control of a Self-Balancing Robot,” in *2020 22nd IEEE International Conference on Automation, Quality and Testing, Robotics - THETA, AQTR 2020 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., May 2020.
- [10] L. Mg, M. Thwin, and Y. Chan, “Application of LQR Control for Two-Wheel Self-Balancing Robot,” 2020.
- [11] L. Guo, S. A. A. Rizvi, and Z. Lin, “Optimal control of a two-wheeled self-balancing robot by reinforcement learning,” *International Journal of Robust and Nonlinear Control*, vol. 31, no. 6, pp. 1885–1904, Apr. 2021.
- [12] I. K. Mohammed and A. I. Abdulla, “Balancing a Segway robot using LQR controller based on genetic and bacteria foraging optimization algorithms,” *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 5, pp. 2642–2653, Oct. 2020.
- [13] A. Pajaziti and L. Gara, “Navigation of self-balancing mobile robot through sensors,” in *IFAC-PapersOnLine*, Elsevier B.V., Nov. 2019, pp. 429–434.
- [14] L. Bin Lau, N. S. Ahmad, and P. Goh, “Self-balancing robot: modeling and comparative analysis between PID and linear quadratic regulator,” *International Journal of Reconfigurable and Embedded Systems*, vol. 12, no. 3, pp. 351–359, Nov. 2023.
- [15] J. Díaz-Téllez, R. S. García-Ramírez, J. Pérez-Pérez, J. Estevez-Carreón, and M. A. Carreón-Rosales, “ROS-based Controller for a Two-Wheeled Self-Balancing Robot,” *Journal of Robotics and Control (JRC)*, vol. 4, no. 4, pp. 491–499, 2023.
- [16] I. H. Altas, *Fuzzy logic control in energy systems: with design applications in MATLAB /Simulink*. The Institution of Engineering and Technology, 2017.
- [17] F. T. Hartadi, B. A. Wicaksana, H. Saputro, and A. S. Priambodo, “Sistem Kendali Fuzzy untuk Robot Mobile: Studi Kasus Pelacakan Objek Bergerak Menggunakan Simulasi Webots,” *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 3, Aug. 2024.