JEEEIT TRANSACTIONS, Vol.2,No.1,December 2025

Modelling, control and simulation for a Two-Wheeled Balancing Mobile Robot System

Abdulnasser Shaeban Alfirjani[1], Mohamed Naji Muftah[1]* and Abobaker Zargoun[1]

[1]Control Engineering Department, College of Electronic Technology, Bani Walid, Libya

*Corresponding author: eshlipta@yahoo.com

ABSTRACT Two-Wheeled Balancing Mobile Robots (TWBMRs) are inherently unstable systems that require an efficient control strategy to maintain balance and ensure satisfactory dynamic performance. This paper presents the optimization of a Proportional-Integral-Derivative (PID) controller using the Particle Swarm Optimization (PSO) algorithm for a TWBMR system. The mathematical model of the robot is derived based on the inverted pendulum concept, and the optimized PID controller is implemented and evaluated in the MATLAB/Simulink environment. The PSO algorithm is employed to determine the optimal PID gains by minimizing the Integral of Squared Error (ISE) performance index. The performance of the proposed PSO-PID controller is compared with the conventional Ziegler-Nichols tuning method using transient response characteristics and performance indices. Simulation results demonstrate that the proposed PSO-PID controller significantly improves the system performance, achieving a rise time of 0.0196 s, a settling time of 0.0637 s, a maximum overshoot of 9.83%, and zero steady-state error. Furthermore, the optimized controller achieves lower performance indices, with IAE = 0.0258, ISE = 0.005079, and ITAE = 0.03032, confirming superior tracking accuracy, faster transient response, and enhanced control performance compared with the conventional tuning approach. In addition, robustness analysis under ±25% system parameter variations demonstrate that the proposed controller maintains stable and satisfactory performance, highlighting its effectiveness and reliability for two-wheeled balancing mobile robot applications.

KEYWORDS TWBMR; PID controller; PSO algorithm; System Modelling; MATLAB simulation.

I. INTRODUCTION

Two-wheeled balancing mobile robots as shown in Figure 1, commonly known as self-balancing robots, represent a captivating field in robotics, merging concepts from control systems, dynamics, and mechanical design. These robots operate on two wheels, much like an inverted pendulum, requiring continuous adjustments to maintain balance. Their unique ability to stabilize themselves and navigate dynamic environments makes them an excellent platform for studying advanced control techniques and dynamic modeling. Beyond their academic value, two-wheeled balancing robots have found applications in diverse areas, including educational tools for teaching robotics and control theory, as well as in personal transportation devices such as self-balancing scooters. Their design and operation demonstrate the interplay of mechanical engineering, sensor integration, and sophisticated control algorithms, making them a vital topic in the development of modern robotic systems.

The design of a two-wheeled balancing robot involves integrating multiple components as shown in Figure 2, each playing a crucial role in achieving stability, mobility, and responsiveness.

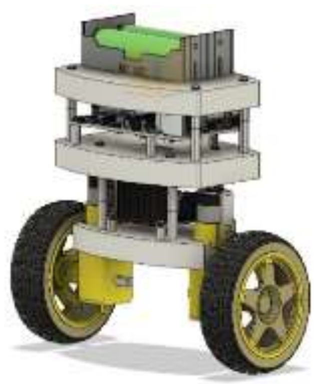

Figure 1. Two-wheeled balancing mobile robot

The mechanical structure forms the backbone of the robot, designed to house all its parts while maintaining a low center of gravity to aid stability. Using modeling software like Solid Edge [1], engineers ensure that the structure is both durable

Received October 2025, Received in received from November 2025, Accepted December 2025, and Published December 2025

and optimized for the robot's balance. The motors and wheels are key to the robot's movement and balance. Two DC motors, connected to the wheels, adjust speed and direction to counteract any tipping or external disturbances. These adjustments are critical for the robot's navigation and maintaining an upright position [2][3]. Sensors, especially Inertial Measurement Units (IMUs) that combine accelerometers and gyroscopes, are central to the robot's ability to balance. They detect angular velocity, acceleration, and tilt angles, providing real-time data about the robot's orientation. This sensor data feeds into the control system, which processes it to make precise motor adjustments [2][4]. Various control algorithms can be employed, including Proportional-Integral-Derivative (PID) controllers for simplicity, Linear Quadratic Regulators (LQR) for optimal performance, and Fuzzy Logic Controllers (FLC) for adaptability to complex environments. Each of these controllers has unique strengths, depending on the robot's design goals and operational conditions [2][5][6]. At the core of the robot's operations is the microcontroller, such as an Arduino or a similar platform. The microcontroller acts as the robot's brain, executing control algorithms, processing the sensor inputs, and sending commands to the motor drivers to adjust their behavior in real time [5]. Together, these components work in harmony, allowing the robot to maintain its balance while navigating dynamic environments, showcasing the integration of mechanical, electronic, and software engineering principles.

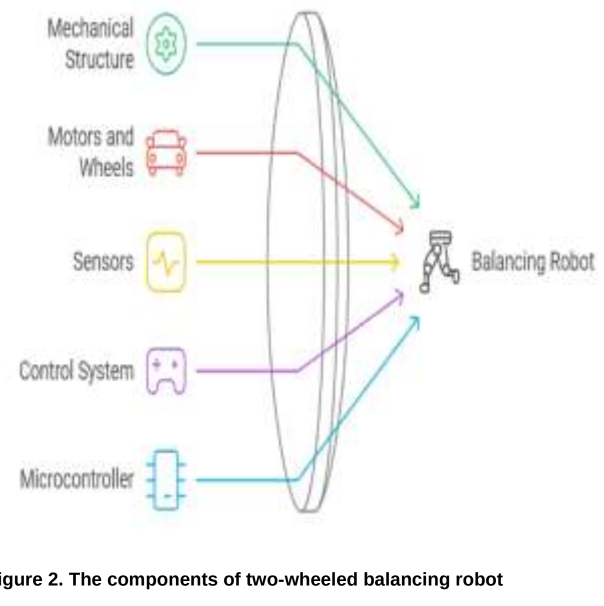

Figure 2. The components of two-wheeled balancing robot

Two-wheeled balancing robots have versatile applications that extend across education, personal use, and commercial domains. As educational tools, they are employed in classrooms and labs to demonstrate core principles of control systems, dynamics, and robotics, offering students hands-on experience with real-world applications of theoretical concepts [1]. In the field of personal transportation, self-balancing robots, such as hoverboards and Segways, provide an innovative solution for short-distance travel. They are particularly valued for their compact design, flexible mobility, and the ability to perform zero-radius turns, making them ideal for navigating crowded or confined spaces [3]. Additionally, these robots have significant potential in home and commercial environments. In homes, they can assist with tasks like security patrols, delivering items, or offering personal assistance. In commercial settings, their applications range from logistics and inventory management to providing customer service in retail or hospitality [1], [6]. This adaptability underscores the potential of two-wheeled robots to integrate seamlessly into a variety of scenarios, enhancing efficiency and convenience in both personal and professional contexts.

The control strategies for two-wheeled balancing robots are critical for managing their inherently unstable dynamics and ensuring precise balance on two wheels. Among the most common methods is PID control, which is simple yet effective. It adjusts motor outputs by calculating the error between the robot's desired and actual positions, using proportional, integral, and derivative components to correct deviations dynamically [2][5]. This method is widely used due to its straightforward implementation and reliable performance in various conditions. For more complex scenarios, Linear Quadratic Regulators (LQR) and Sliding Mode Controllers (SMC) are employed. These advanced strategies provide robust performance by minimizing a cost function (in LQR) or using variable structure control (in SMC) to handle system uncertainties and maintain balance even under disturbances [6]. Additionally, Fuzzy Logic Control (FLC) offers an intelligent alternative by leveraging fuzzy rules to address uncertainties and nonlinearities inherent in the system [6]. This approach is particularly effective in maintaining stability and responding adaptively to external forces or changes in the environment. Together, these strategies demonstrate the diverse techniques available for tackling the challenges of balancing robots, allowing designers to select the most suitable method based on specific application requirements and environmental complexities.

In this paper, a PID controller is employed to stabilize the two-wheeled balancing robot. However, tuning a PID controller to achieve optimal results is a challenging task, and numerous methods have been developed to identify the best PID parameters. Traditional tuning methods include Ziegler-Nichols, Integral Time Absolute Error (ITAE) equations, gain and phase margins tuning, and single input single output (SISO) tools. These classical approaches have been effective but are often limited by their inability to consistently find global optimal solutions in complex systems. To address these limitations, unconventional techniques have gained prominence, particularly due to their ability to explore broader solution spaces and achieve global optimization. Among these methods, evolutionary algorithms and fuzzy logic have demonstrated significant success in enhancing control systems. A notable example of such evolutionary algorithms is the Particle Swarm Optimization (PSO) algorithm, a heuristic search method designed to optimize continuous nonlinear functions and widely used for PID controller tuning. Inspired by swarm behavior observed in nature, such as bird flocking and fish schooling, PSO simulates these dynamics to iteratively refine solutions. Developed by Kennedy and Eberhart in the 1990s, PSO is rooted in artificial life and evolutionary computation principles. It leverages the collective intelligence and adaptive behaviors seen in animal groups to solve optimization problems efficiently. The application of PSO in control theory has attracted substantial attention, as it has been successfully applied to various fields, including neural networks, structural optimization, topology optimization, and fuzzy systems. For the two-wheeled balancing robot, numerous control strategies have been explored, such as fuzzy controller design enhanced by ant colony optimization, Extended State Observer (ESO)-based Linear Quadratic Regulator (LQR) controllers, and nonlinear dynamic surface control to

address parameter uncertainties. These advanced methods demonstrate the diverse approaches available for managing the challenges inherent in controlling the two-wheeled balancing robot, providing a robust foundation for achieving precise stabilization and dynamic response.

The main contribution of this work is the development and comprehensive evaluation of a PSO-PID controller for a two-wheeled balancing mobile robot using a mathematical model derived from the inverted pendulum dynamics. Unlike conventional PID tuning approaches, the proposed method automatically determines the optimal controller gains by minimizing the Integral of Squared Error (ISE) criterion using the Particle Swarm Optimization algorithm. Furthermore, the proposed controller is extensively evaluated using transient response characteristics, performance indices (IAE, ISE, and ITAE), and robustness analysis under ±25% parameter variations. The obtained results demonstrate that the proposed PSO-PID controller provides faster response, lower overshoot, improved tracking performance, and better robustness than the conventional Ziegler-Nichols tuning method.

II. SYSTEM MODELLING

The transfer function is derived through rigorous mathematical calculations of the self-balancing robot's mechanical system. The mechanical principles underlying a TWBMR are analogous to the concept of an Inverted Pendulum (IP). In such a system, the pendulum will topple in the absence of a balancing force. Consequently, this implies that the system is inherently unstable. However, when the pendulum's position is upright and balanced, the system becomes stable. The inverted pendulum (IP) system as shown in Figure 3, features a simple physical structure but poses significant complexity in its control. The system consists of a pendulum with mass m and length $2L$, hinged to a horizontally moving cart. The cart, which has a mass M, is subjected to a viscous friction force directly proportional to its velocity, expressed as $Fv=bv$, where b represents the damping coefficient. This type of friction can be characterized as a damping force.

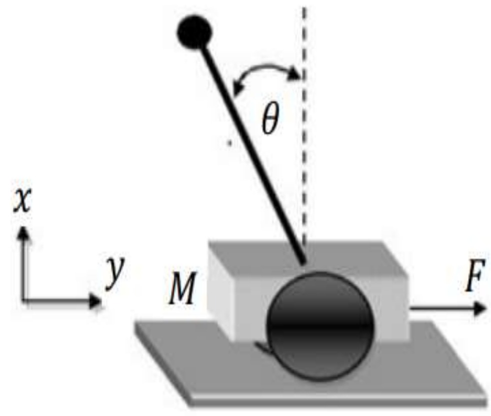

Figure 3. Inverted pendulum model

In addition to the frictional forces, the system is influenced by a control force F, exerted by a controlled motor. This control force plays a crucial role in stabilizing the pendulum in an upright position while simultaneously ensuring the cart reaches the desired position. This dynamic interaction between forces highlights the intricate control requirements of the IP system. The dynamic equation for the Two-Wheeled Balancing Mobile Robot (TWBMR) is derived using Newton's Second Law of Motion. Figure 4 illustrates the Free Body Diagram (FBD) of the mechanism, showing the forces acting on the system. The pendulum's slope, defined by various angles, is decomposed into two components: one along the horizontal direction and the other along the vertical direction. Table I provides a detailed explanation of the parameters used in the TWBMR system

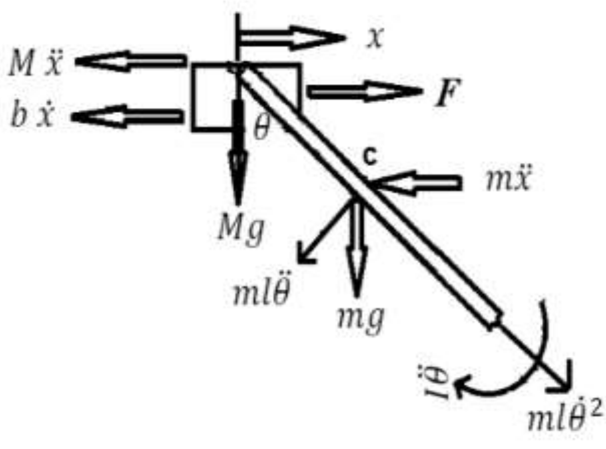

Figure 4. IP FBD and inertia forces diagram

TABLE I. PARAMETERS OF TWBMR

Variable	Description	Value
M	Mass of the cart	*0.3 kg*
M	Mass of pendulum rod	*0.2 kg*
B	Viscous friction coefficient	*0.1 N/m/sec*
L	One half pendulum rod length	*0.15 m*
I	Inertia moment pendulum rod	*0.0015 kg.m^2*
F	Applied force to the cart	*Kg. m.s^2*
G	Gravity	*m/s^2*
Θ	Pendulum angle	*Rad*

Based on the forces and interactions depicted in Figure 4, the mathematical model for the self-balancing robot system can be formulated as follows [14].

$$\Sigma f_x = 0$$

$$F - b\dot{x} - M\dot{x} - m\ddot{x} - ml\ddot{\theta}\cos\theta + ml\dot{\theta}^2 sin\theta = 0 \quad (1)$$

$$\Sigma f_y = 0$$

$$Mg + mg - ml\ddot{\theta}\sin\theta - ml\dot{\theta}^2\cos\theta = 0 \quad (2)$$

$$\sum M_A = 0$$

$$mgl\sin\theta + I\ddot{\theta} + ml^2\ddot{\theta} + ml\ddot{x}cos\theta = 0, \quad (3)$$

Equations (1), (2), and (3) are combined, hence the force equation is obtained along the horizontal and vertical directions,

$$(I + ml^2)\theta + mgl\sin\theta = -ml\ddot{x}\cos\theta \quad (4)$$

$$(M + m)\ddot{x} + b\dot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = F, \quad (5)$$

Equations (4) and (5) are two linear equations of the transfer function, where q = π. With assumption $\theta = \pi + \emptyset$,

$$\cos\theta = -1 \quad (6)$$

$$\sin\theta = -\emptyset \quad (7)$$

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2} = 0 \quad (8)$$

After processing the equations (6), (7), and (8) approach to non-linear, then we obtained two variations of motion equations. The U value represents the input,

$$(I + ml)^2\ddot{\Phi} - mgl\Phi = ml\ddot{x} \quad (9)$$

$$(M + m)\ddot{x} + b\dot{x} + ml\ddot{\Phi} = U, \quad (10)$$

By applying Laplace transform on equations (9) and (10),

$$(I + ml^2)\ddot{\Phi}(s)s^2 - mgl\Phi(s) = mlX(s)s^2 \quad (11)$$

$$(M + m)X(s)s^2 + bX(s)s + ml\Phi(s)s^2 = U(S) \quad (12)$$

we yielded the transfer function for TWBMR from the equations (11) and (12).

$$\frac{\Phi(s)}{U(s)} = \frac{\frac{ml}{q}s}{s^3 + \frac{b(I+ml^2)}{q}s^2 - \frac{(M+m)mgl}{q}s - \frac{bmgl}{q}} \quad (13)$$

Where $q = [(M + m)(I + ml^2) - (ml)^2]$.

The parameters in Table 1 are substituted in equation (13) and we obtained the transfer function of TWBMR.

$$\frac{\Phi(s)}{U(s)} = \frac{14.286s}{s^3 + 0.286s^2 - 70s - 0.0294} \quad (14)$$

III. CONTROLLER DESIGN

This research proposes a controller for precise angle control of a two-wheeled balancing robot. The control design employs a feedback loop system to ensure accurate angle tracking and effective response to input signals. The feedback mechanism for controlling the robot's angle is illustrated in Figure 5. The primary concept behind the angle controller is to enable the robot to maintain a specific angle as dictated by the input signal. The system continuously measures the error in real time by comparing the actual output (the robot's current angle) with the desired input angle. When a positive error is detected, the controller directs the robot to tilt in one direction until it stabilizes at the target angle [15]. Conversely, if the error is negative, the robot adjusts in the opposite direction to achieve the desired angle. The controller plays a crucial role in guiding the system's response to any detected deviation, ensuring that the robot accurately maintains its balance and follows the specified angle. This dynamic feedback mechanism allows the system to maintain high precision and reliability in controlling the robot's movements.

Ideally, the motor of the two-wheeled balancing robot (TWBMR) should stop moving when the current angle aligns perfectly with the input signal. However, in real-world scenarios, achieving this ideal state is rarely feasible due to factors such as friction, inertia, and other dynamic influences. Consequently, the motor will adjust its movement until the robot's angle reaches the desired input value, ensuring precise control despite these practical challenges. System errors can lead to oscillations that may destabilize or even damage the two-wheeled balancing robot (TWBMR). To mitigate these risks, a robust control system is essential. This research proposes a PID controller to ensure the stability and reliability of the TWBMR. The controller's effectiveness is thoroughly evaluated through simulation testing, demonstrating its ability to maintain balance and enhance system performance.

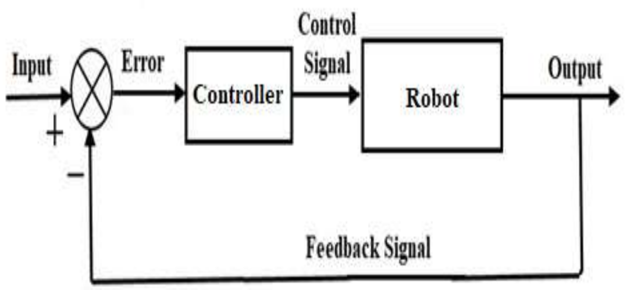

Figure 5. TWBMR controller designed

IV. PROPORTIONAL-INTEGRAL-DERIVATIVE (PID) CONTROLLER

The block diagram of PID controller in a closed loop system is shown in Figure 6. The PID controller consists of three main components: Proportional (P), Integral (I), and Derivative (D). Each component plays a specific role in adjusting the control output based on the error signal between the desired set point and the actual process variable.

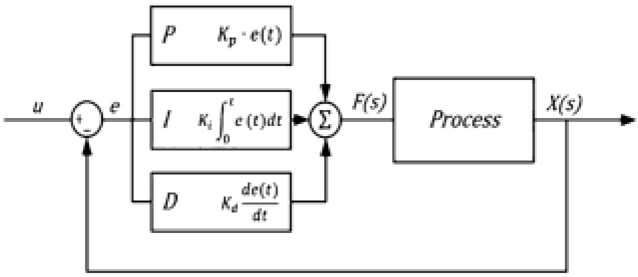

Figure 6. The block diagram of PID controller

The total output from the PID controller is the sum of the contributions from the proportional, integral, and derivative components as shown in Equation (15):

$$U(t) = P_{Output} + I_{Output} + D_{Output}$$

$$U(t) = K_p.e(t) + K_i \int_0^t e(t)dt + K_d \frac{d}{dt}\, e(t) \quad (15)$$

Where:

$U(t)$ is the control output,
$e(t)$ is the error signal,
$K_p, Ki, and\ Kd$, are the proportional, integral, and derivative gains, respectively.

Designing a PID controller for a Two-Wheeled Balancing Mobile Robot (TWBMR) involves addressing the fundamental challenge of maintaining the robot's balance while enabling it to navigate dynamically. The controller must adjust the motor input based on feedback from the robot's sensors to ensure the robot remains in an upright position and responds effectively to disturbances such as external forces or changes in its environment [25]. The first step in designing the PID controller for a TWBMR is to define the robot's dynamics. The robot can be modeled as an inverted pendulum, where the objective is to maintain a vertical orientation by controlling the tilt angle. The system is affected by several factors, including friction, inertia, and sensor noise.

V. PARTICLE SWARM OPTIMIZATION ALGORITHM

Particle swarm optimization invented by Kennedy and Eberhart [31], is a modern heuristic algorithm. It was developed to address problems involving continuous nonlinear optimization and was influenced by the social conduct of schooling fish and birds. The idea behind this algorithm is that a group of birds are looking for food in an area with only one piece [32]. All birds are uninformed of the placement of the food, but they are aware of its distance at any given moment. The most efficient and effective method for locating food would be to follow the bird closest to it. Based on this situation, the PSO algorithm is applied to address the optimization issue.

In particle swarm optimization, a single solution is represented by a "bird" in the search space; this representation is called a "particle". The swarm is modelled as a collection of particles moving through space with multiple dimensions, each with a position and a velocity [33]. These particles are important because they can remember their best position and know where the best position is global. Members of a swarm exchange good positions with one another, and depending on good positions, they modify their position and velocity, as shown in Equation below.

$$\mathrm{v(k+1)_{i,j} = w.v(k)_{i,j} + c_1 r_1\left(gbest - x(k)_{i,j}\right) + c_2 r_2\left(pbest_j - x(k)_{i,j}\right)}$$

$$\mathrm{x(k+1)_{i,j} = \ x(k)_{i,j} + v(k)_{i,j}}$$

where:

$x_{i,j}$ $v_{i,j}$ is the velocity of particle i and dimension j
$x_{i,j}$ is the position of particle i and dimension j
$\mathrm{c_1, c_2}$ are the acceleration constants
w is the inertia weight factor

r_1, r_2 are random numbers between 0 and 1
pbest is the best position of a specific particle
gbest is the best particle of the group

Figure 7 illustrates the state of swarm at the iteration $(k + 1)^{th}$. It shows how the velocity of particle k is updated and that, in turn, updates the position of the kth particle in the iteration $(i + 1)^{th}$.

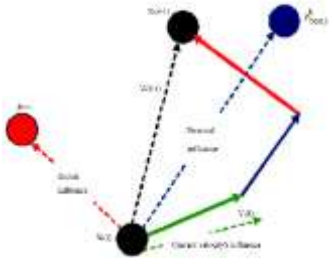

Figure 7. Generation scheme of the particles

Several steps must be taken to implement the PSO method [32], [34], which can be simplified as shown in Figure 8.

- Initialize a collection of particles, giving them random placement, velocities and accelerations.
- Assess each particle's fitness.
- Examine the fitness of every particle against its previous pbest. If the fitness has improved, update the fitness as pbest.
- Examine the fitness of every particle against its previous gbest. If the fitness has improved, update the fitness as gbest.
- Update the velocity and placement of each particle.
- Return to step 2 of the procedure and repeat until a predetermined stopping condition is met.

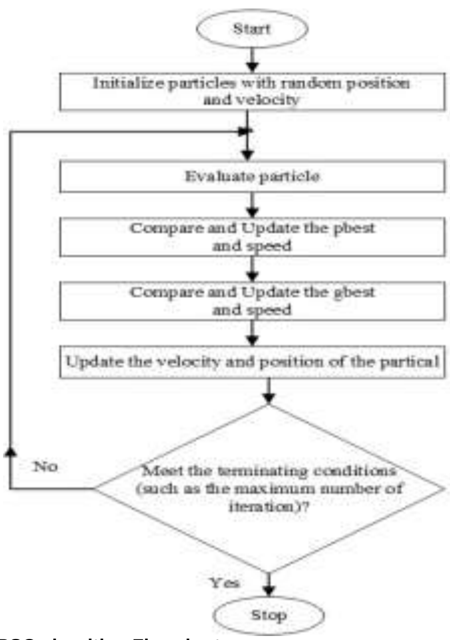

Figure 8. PSO algorithm Flowchart

VI. RESULTS AND DISCUSSION

This part provides a detailed analysis of the system's behavior in two distinct configurations: the open-loop configuration without control and the closed-loop configuration with a PID controller tuned using the Particle Swarm Optimization (PSO) method. First, the results of the open-loop system are presented, showcasing its response characteristics and highlighting its inherent instability due to the lack of a control mechanism. These findings provide a baseline for understanding the system's natural dynamics and limitations. Next, this part delves into the implementation of a PID controller, where its parameters are optimized using the PSO algorithm. The PSO method is employed to achieve optimal tuning by iteratively adjusting the controller's parameters to enhance system stability and performance. The effectiveness of this approach is demonstrated through simulation results, which exhibit improved response metrics such as reduced overshoot, faster settling time, and minimized steady-state error. Finally, the part concludes with a comparative analysis between the system performances achieved using the PSO-tuned PID controller and other previously employed tuning methods. This comparison underscores the advantages and improvements brought about by the PSO-based tuning in terms of stability, efficiency, and overall system behavior.

1. TWBMR SYSTEM WITHOUT CONTROLLER

Referring to the block diagram in Figure 6, we analyzed the response of the self-balancing robot's transfer function through a system simulation performed in Simulink MATLAB without implementing a controller. The simulation results yielded a system response graph, which is illustrated in Figure 9.

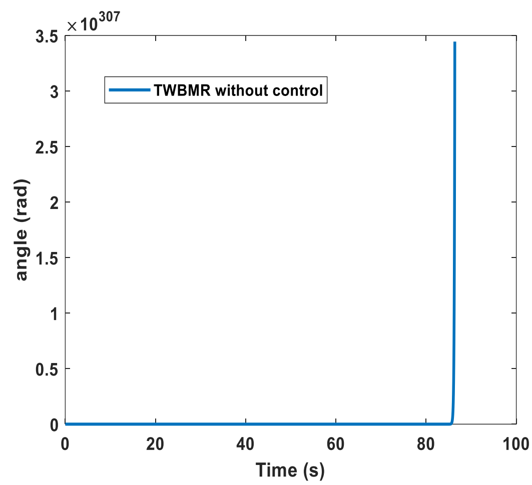

Figure 9. TWBMR without control

As observed in Figure 9, the Two-Wheeled Balancing Mobile Robot (TWBMR) system exhibits instability when operating without a controller. This instability is attributed to the presence of three poles at the origin in the system's transfer function, which inherently leads to an unbounded response. To address this issue and achieve stability with a desirable system response, a PID controller is implemented.

The PID controller adjusts the system's dynamics, ensuring stability and enhancing performance by providing improved response characteristics such as reduced overshoot, faster settling time, and minimized steady-state error.

2. PID CONTROLLER BASED PSO ALGORITHM

The controller simulation was carried out in MATLAB/Simulink, as shown in Figure 10. The implementation is divided into three segments: system input/output, controller design, and performance index (ISE). Initially, the desired input is introduced into the model plant. The feedback loop (closed-loop system) is necessary to achieve the desired performance to achieve desirable outcomes. As Equation (3.14) depicts, the error signal is produced by subtracting the input signal from the output signal. This signal was then sent to the controller. The controller minimizes the error by continuously adjusting the control signal.

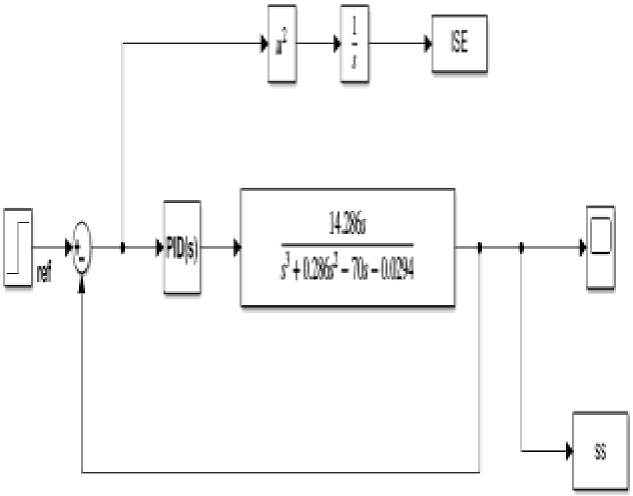

Figure 10. TWBMR with PID control

To evaluate the effectiveness of the controller design, two forms of input (step input and multistep trajectories) are used in this research. To determine the scaling factors in this controller (Kp, Ki, and Kd), the PSO optimization algorithm has been applied in MATLAB/Simulink. Table II shows the parameters utilized in the PSO optimization algorithm. In this research, the conventional and widely used fitness function was employed to determine the optimal value for the scaling factors. Integral Square Error (ISE) is utilized because it can deal with errors over an extended period of time and provides a satisfactory dynamic response [36]. Figure 11 portrays the ISE values obtained from consecutive PSO generations employing a PID controller. The PID controller optimization was accomplished by implementing the algorithm with 10 agents (Np=10) and performing 50 iterations (Ni=50).

TABLE II. PSO PARAMETERS FOR PID CONTROLLER

Parameter	Value
Social coefficient, s	1.42
Cognitive coefficient, c	1.42
Inertia weight, iw	0.9
Dimension of the problem, Nd	3
Upper boundary, Up	[1000, 1000, 1000]
Lower boundary, Lp	[0, 0, 0]
Number of particles (Np)	10
Maximum iterations (Ni)	50

The PSO parameters were selected based on a trade-off between optimization accuracy and computational efficiency. A swarm size of 10 particles was adopted to provide sufficient diversity while maintaining a low computational cost. The maximum number of iterations was set to 50, which was found adequate for convergence of the objective function without unnecessary computational effort. The inertia weight was selected as 0.9 to encourage global exploration during the optimization process, whereas the cognitive and social acceleration coefficients ($c1 = c2 = 1.42$) were chosen to balance individual learning and social cooperation among particles. These parameter values provided stable convergence and consistently produced high-quality PID gains for the proposed two-wheeled balancing mobile robot system.

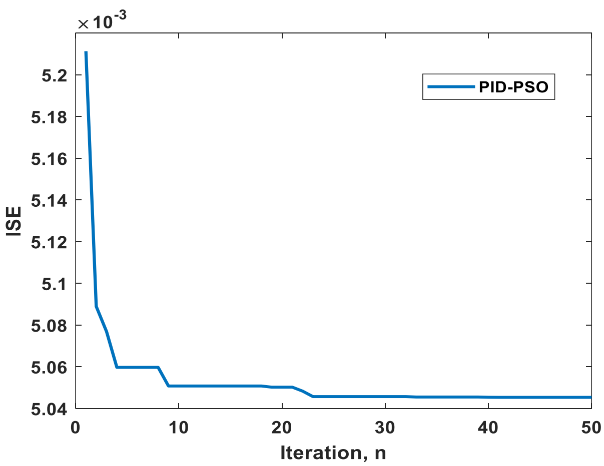

Figure 11. The value of ISE in successive generations of the PSO PID

Table III and Figure 12 display the optimum parameters for the PID controller derived by the PSO optimization algorithm. The simulation test runs for angle responses within 15s and multistep responses within 25s. The results indicate that the PID controller can control the angle of the TWBMR. Figure 13 illustrates the step response of the proposed PSO-PID controller for step input at 90 rad, while Figure 14 illustrates the multistep tracking response of the proposed controller for the PID controller.

TABLE III. THE PSO-PID CONTROLLER PARAMETERS

Scaling factors	Parameters value
Kp	293.5258
Ki	988.6141
Kd	333.9884

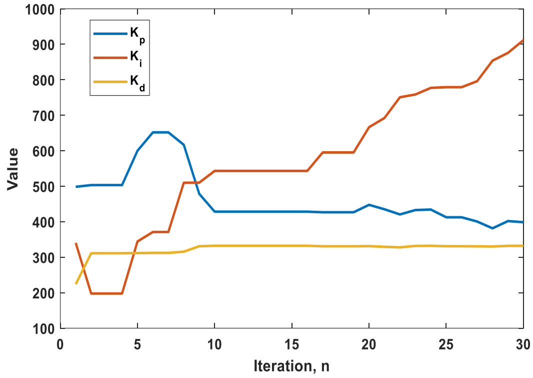

Figure 12. The optimum parameters for the PID controller

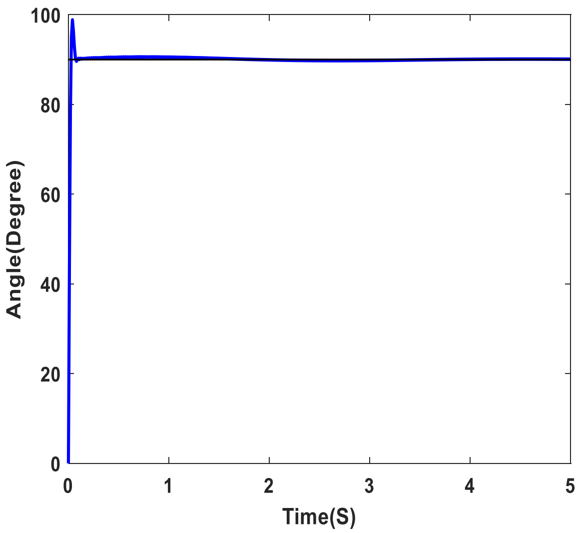

Figure 13. Step response for PID simulation

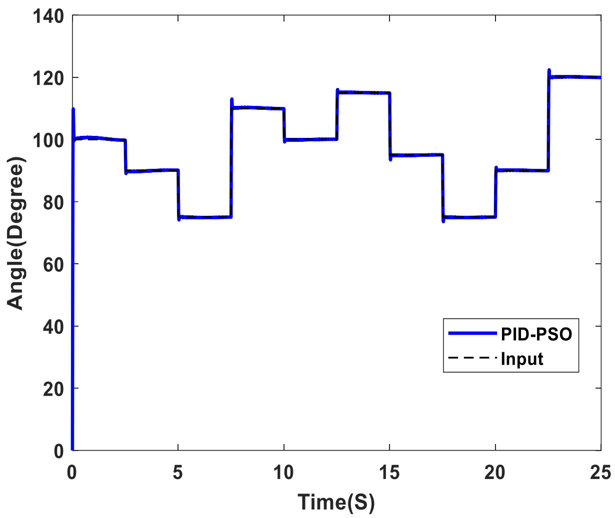

Figure 14. Multistep response for PID simulation

Figures 13 and 14 demonstrate the effectiveness of the proposed PSO-PID controller under both step and multistep reference inputs. As shown in Figure 13, the controller provides a rapid transient response with a short rise time and settling time while maintaining a relatively small overshoot. This behavior indicates that the optimized PID gains obtained by the PSO algorithm successfully improve the dynamic characteristics of the TWBMR. Figure 14 further confirms the controller's capability to accurately track varying reference signals with smooth transitions and negligible steady-state error. The fast response, high tracking accuracy, and stable behavior demonstrate the suitability of the proposed controller for balancing applications requiring rapid adaptation to changing operating conditions. This demonstrates how the PSO-PID controller is able to respond faster while having very little overshoot as shown in Table IV.

TABLE IV. PSO-PID CONTROLLER PERFORMANCE

Rise Time (s)	Settling Time (s)	Maximum Overshoot (%)	Error Steady State (%)
0.0196	0.0637	9.8276	0

3. COMPARISON OF THE PID CONTROLLERS' PERFORMANCE

Mudeng et al. [35] employed the Ziegler-Nichols method to design the PID parameters for controlling a Two-Wheeled Balancing Mobile Robot, demonstrating the application of this classical tuning approach. The resulting parameters are shown in Table V.

TABLE V THE PID-ZIEGLER-NICHOLS PARAMETERS

Scaling factors	Parameters value
Kp	4.0602
Ki	4.4100
Kd	1.1025

TABLE VI. THE COMPARISON OF THE PID CONTROLLERS' PERFORMANCE

PID Controller	Rise Time (s)	Settling Time (s)	Maximum Overshoot (%)	Error Steady State (%)
PSO	0.0196	0.0637	9.8276	0
Ziegler-Nichols (2 times Kp, Ki, and Kd)	0.316	8.879	92.6	0
Ziegler-Nichols (25 times Kp, Ki, and Kd)	0.374	3.64	18.6	0

Table VI evaluates the performance of a PID controller tuned using three different methods: Particle Swarm Optimization (PSO) and Ziegler-Nichols (ZN) with gain multipliers of 2x and 25x. Each method is assessed based on four key metrics: rise time, settling time, maximum overshoot, and steady-state error. PSO delivers the best performance across the board, achieving the fastest rise time (0.0196 seconds), the quickest settling time (0.0637 seconds) as shown in Figure 15, and a relatively low overshoot (9.83%) as shown in Figure 16, all while ensuring zero steady-state error. On the other hand, ZN (2x) performs poorly, with a slow rise time (0.316 seconds), excessively high settling time (8.879 seconds), and extremely high overshoot (92.6%), indicating instability and aggressive tuning. ZN (25x) shows improvements in overshoot (18.6%) and settling time (3.64 seconds) compared to ZN (2x), but it still does not match the efficiency of PSO. Both ZN methods manage zero steady-state error, but their performance is less optimal overall. In summary, PSO tuning stands out for its ability to produce a fast, stable, and well-controlled response, outperforming the traditional Ziegler-Nichols tuning methods.

The superior performance of the PSO-PID controller can be attributed to the optimization capability of the PSO algorithm, which efficiently searches the solution space to determine the optimal combination of PID gains. Unlike the Ziegler-Nichols method, which relies on empirical tuning rules, PSO minimizes the selected objective function by iteratively refining the controller parameters. Consequently, the proposed controller achieves a significantly faster transient response, lower overshoot, and excellent steady-state accuracy. These improvements indicate that the optimized controller provides better damping characteristics and more effective control of the inherently unstable dynamics of the two-wheeled balancing mobile robot.

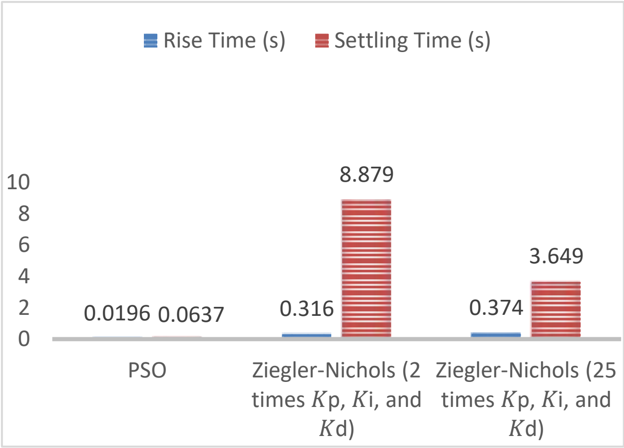

Figure 15. Rise time and Settling time for PID controller

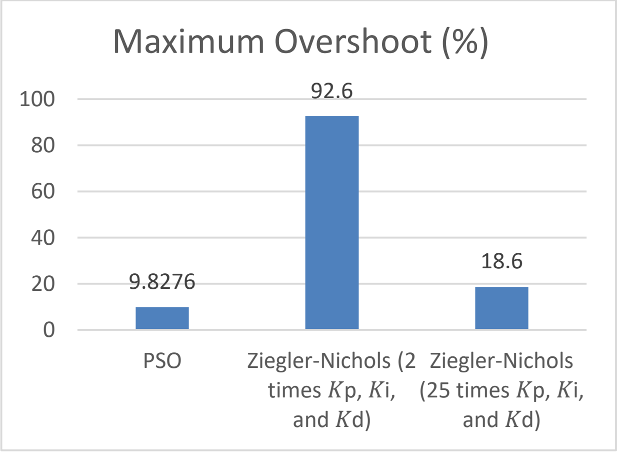

Figure 16. Maximum overshoot for PID controller

4. EVALUATION OF PERFORMANCE INDICES FOR CONTROLLER EFFECTIVENESS

To comprehensively assess the effectiveness of the proposed controller designs, this research evaluates their performance using three widely recognized performance indices: The Integral of Absolute Error (IAE), the Integral of Squared Error (ISE), and the Integral Time Absolute Error (ITAE), as defined in Equations (17), (18), and (19), respectively. The controller simulation with ISE, IAE and ITAE is carried out in MATLAB/Simulink, as seen in Figure 17. The IAE measures the total absolute error over time, with lower values indicating better performance by minimizing deviations from the desired output. The ISE calculates the sum of squared errors, penalizing larger deviations more significantly, making it useful for assessing how well the controller reduces substantial errors. Meanwhile, the ITAE incorporates both error magnitude and duration by weighting the error with time, prioritizing rapid error correction to minimize long-term deviations. These indices provide valuable quantitative criteria for comparing and optimizing different controller designs, ensuring the selection of the most effective approach based on performance trade-offs.

$$IAE = \int_0^{\infty} |e(t)| dt \tag{17}$$

$$ISE = \int_0^{\infty} e^2(t)\, dt \tag{18}$$

$$IATE = \int_0^{\infty} t|e(t)| dt \tag{19}$$

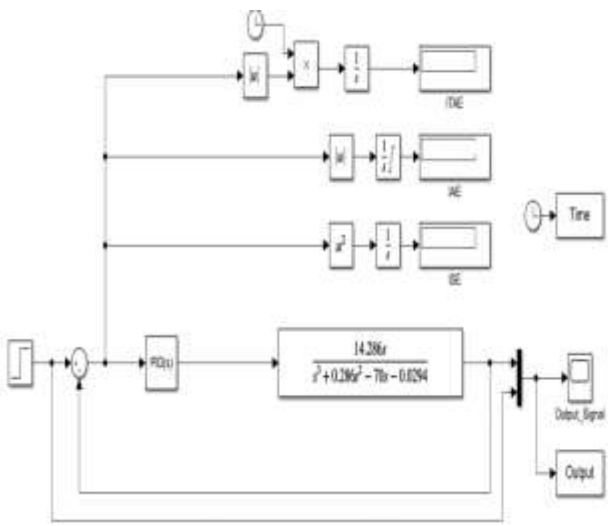

Figure 17. The controller simulation with ISE, IAE and ITAE

Table 7 presents a comparison of the performance of two PID controller tuning methods, PSO (Particle Swarm Optimization) and Ziegler-Nichols (with gains multiplied by 25), using three performance indices: Integral of Absolute Error (IAE), Integral of Squared Error (ISE), and Integral Time Absolute Error (ITAE). These indices measure how well the controllers minimize error over time.

TABLE VII. THE COMPARISON OF THE PID CONTROLLERS' PERFORMANCE

PID Controller	IAE	ISE	IATE
PSO	0.0258	0.005079	0.03032
Ziegler-Nichols (25 times Kp, Ki, and Kd)	0.06096	0.007691	0.04617

The PSO-tuned controller outperforms the Ziegler-Nichols method across all indices, with lower values for IAE (0.0258 vs. 0.06096), ISE (0.005079 vs. 0.007691), and ITAE (0.03032 vs. 0.04617). This indicates that the PSO method results in smaller overall errors, better stability by reducing large deviations, and quicker error correction, making it a more effective tuning approach for minimizing control errors and improving system performance.

5. ROBUSTNESS ANALYSIS OF PSO-PID

To evaluate the robustness of the proposed PID controller, the parameters of the testbed system were modified simultaneously by ± 25% from their nominal values. These cases represent the most common uncertainty in parameters that the testbed system may encounter during real-time use. The optimal scaling factors of PID acquired under normal conditions will not be returned when the model is exposed to system parameter fluctuation.

A±25% parameter variation was selected because it represents a sufficiently large yet realistic level of modeling uncertainty commonly adopted in robustness evaluation studies. Such variations may arise from uncertainties in system parameters due to manufacturing tolerances, payload changes, component aging, friction variations, or unmodeled dynamics encountered during practical operation. Evaluating the proposed controller under these conditions provides a rigorous assessment of its ability to maintain stability and acceptable dynamic performance despite significant parameter deviations from the nominal model.

In case 1, the system's poles are altered by +25%, and the original system equation is represented by Equation (20) below:

$$\frac{14.286\,S}{1.25\,S^3 + 0.536\,S^2 - 70.25S - 0.2794} \tag{20}$$

where the Poles values of the system are: P1=-7.7122, P2=7.2874, and P3 =-0.0040, while Zero value of the system is: Z1= 0.

In case 2, the system's Zeros are altered by +25%, and the original system equation is represented by Equation (21) below:

$$\frac{14.536\,S}{S^3 + 0.286\,S^2 - 70S - 0.0294} \tag{21}$$

where the Poles values of the system are: P1=-8.5106, P2=8.2250, P3=-0.0004, while Zero value of the system is: Z1=0.
In case 3, the system's poles are altered by -25%, and the original system equation is represented by Equation (22) below:

$$\frac{14.286\,S}{0.75\,S^3 + 0.036\,S^2 - 69.75S + 0.2206} \qquad (22)$$

where the Poles values of the system are: P1=-9.6693, P2=9.6181, P3=0.0032, while Zero value of the system is: Z1=0.
In case 4, the system's Zeros are altered by -25%, and the original system equation is represented by Equation (23) below:

$$\frac{14.036\,S}{S^3 + 0.286\,S^2 - 70S - 0.0294} \qquad (23)$$

where the Poles values are: P1=-8.5106, P2=8.2250, P3=-0.0004, while Zero value of the system is: Z1=0.
The dynamic responses of the system with PID controller at 900 step input for all cases are depicted in Figures 18–21, respectively, while the characteristics of the system (settling time Ts and rise time Tr) are summarized in Table VIII.

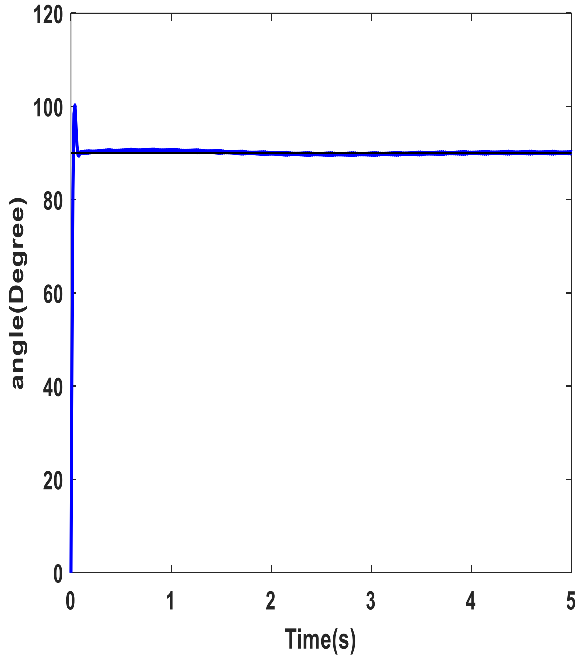

Figure 19. Dynamic response of the system with PID controller under parametric uncertainties, case 2

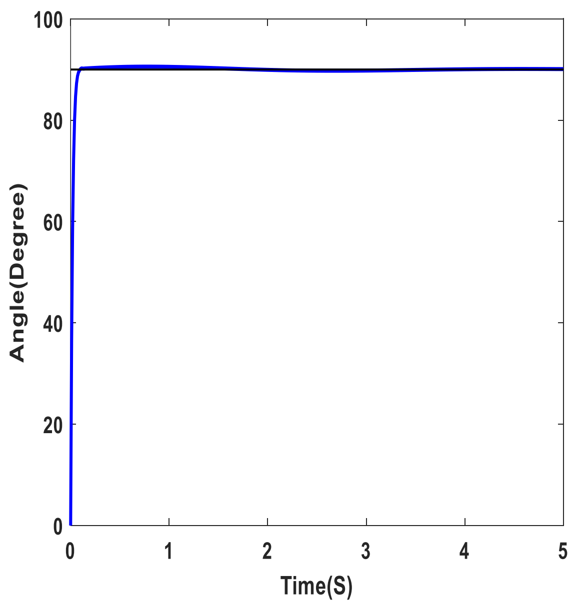

Figure 18. Dynamic response of the system with PID controller under parametric uncertainties, case 1

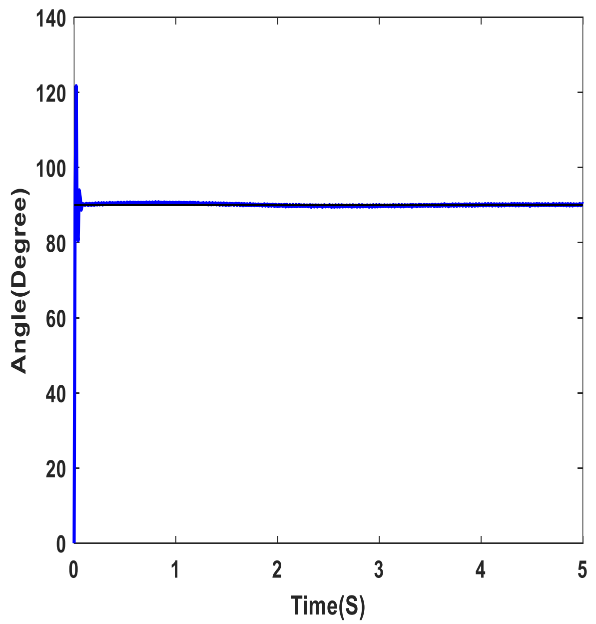

Figure 20. Dynamic response of the system with PID controller under parametric uncertainties, case 3

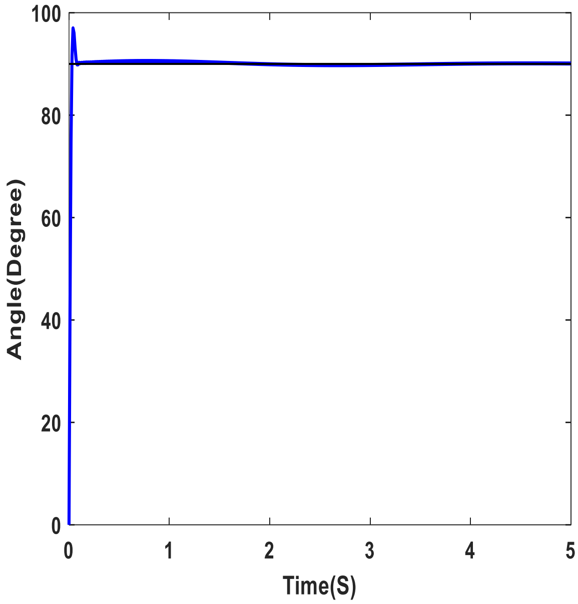

Figure 21. Dynamic response of the system with PID controller under parametric uncertainties, case 4

TABLE VIII. THE PERFORMANCE OF THE SYSTEM WITH PID CONTROLLER UNDER PARAMETRIC UNCERTAINTIES

Criteria	Robustness Analysis of PSO-PID			
	Case 1	Case 2	Case 3	Case 4
Tr (s)	0.039	0.018	0.007	0.022
Ts (s)	0.066	0.0595	0.060	0.066
OS%	0.631	11.19	35.04	7.726

The statement indicates that the implemented PID controller has demonstrated robustness against a broad range of uncertainties in the investigated TWBMR system, as shown in Figures 18-21 and Table 8. The rise time (Tr) and settling time (Ts) for all cases were less than 0.1 second, and small overshoot.

VII. CONCLUSION

The optimization of Proportional-Integral-Derivative (PID) parameters using the Particle Swarm Optimization (PSO) algorithm for the two-wheeled balancing mobile robot system has been a significant step forward in enhancing robotic control systems. This study has demonstrated that PSO, as an intelligent optimization technique, can effectively address the challenges of tuning PID controllers, resulting in improved system performance, stability, and reliability. By leveraging PSO, the tuned PID controller achieved a notable reduction in overshoot, faster settling time, and minimized steady-state error, all of which are critical metrics for evaluating control system performance. These improvements underline the ability of PSO to optimize complex and nonlinear systems efficiently. Moreover, the PSO-based PID controller showcased robust stability, maintaining consistent performance across a variety of operating conditions and external disturbances. This robustness highlights the reliability of PSO as a tuning method, making it suitable for real-world applications where unpredictable factors may affect system dynamics. The algorithm’s high convergence rate further demonstrated its computational efficiency, making it viable for time-sensitive and dynamic environments such as two-wheeled balancing robots. Additionally, the methodology’s scalability suggests that it can be adapted to other robotic platforms and control systems, broadening its scope and potential applications. In conclusion, the integration of PSO with PID controllers has proven to be a powerful approach for improving the performance of balancing robots. Future work will focus on implementing the proposed controller on a real TWBMR platform and comparing its performance with other intelligent optimization techniques. for further exploration and innovation in the field of intelligent control systems, paving the way for more efficient, reliable, and versatile robotic solutions.

REFERENCES

[1] Filipek, p. remote-controlled two-wheeled self-balancing robot. Journal of Technology and Exploitation in Mechanical Engineering. 2022 https://doi.org/10.35784/jteme.3147
[2] Mudeng, V., Hassanah, B., Priyanto, Y., & Saputra, O. Design and Simulation of Two-Wheeled Balancing Mobile Robot with PID Controller. International Journal of Sustainable Transportation Technology. 2020 https://doi.org/10.31427/ijstt.2020.3.1.3
[3] Nawawi, S., Noh, A., & Osman, J. Two-wheeled balancing robot, 2008.
[4] Iwendi, C., Alqarni, M., Anajemba, J., Alfakeeh, A., Zhang, Z., & Bashir, A. Robust Navigational Control of a Two-Wheeled Self-Balancing Robot in a Sensed Environment. IEEE Access. 2019;7. https://doi.org/10.1109/ACCESS.2019.2923916
[5] Lim, B. Two Wheeled Balancing Robot. . 2007
[6] Hong-Ge, R. Two-wheeled self-balancing mobile robot dynamic model and balancing control. Application Research of Computers. 2009
[7] Pawananont, K., & Charoensuk, K. Design of a Two-Wheeled Auto-Balancing Robot under Impulse Interruption. ECTI Transactions on Electrical Engineering, Electronics, and Communications. 2022 https://doi.org/10.37936/ecti-eec.2022203.247520
[8] Nemec, D., Adamkovic, D., Hruboš, M., Pirník, R., & Mihálik, M. Fast Two-Wheeled Balancing Robot. 2021 22nd International Carpathian Control Conference

(ICCC). 2021 https://doi.org/10.1109/ICCC51557.2021.9454659
[9] Junfeng, W., & Wanying, Z. (2011). Research on Control Method of Two-wheeled Self-balancing Robot. 2011 Fourth International Conference on Intelligent Computation Technology and Automation. doi:10.1109/icicta.2011.132.
[10] Zhao, J. W., & Ruan, X. G. (2011). Modelling and control of a flexible two-wheeled self-balancing mobile robot. International Journal of Systems, Control and Communications, 3(3), 330. doi:10.1504/ijscc.2011.042438
[11] Sikander, A., & Prasad, R. Reduced order modelling based control of two wheeled mobile robot. Journal of Intelligent Manufacturing. 2017; 30. https://doi.org/10.1007/s10845-017-1309-3
[12] Kim, S., & Ahn, C. Self-Tuning Position-Tracking Controller for Two-Wheeled Mobile Balancing Robots. IEEE Transactions on Circuits and Systems II: Express Briefs. 2019; 66. https://doi.org/10.1109/TCSII.2018.2869915
[13] Lima-Perez, A., Díaz-Téllez, J., Gutierrez-Vicente, V., Estevez-Carreon, J., Pérez-Pérez, J., García-Ramirez, R., & Chavez-Galan, J. Robust Control of a Two-Wheeled Self-Balancing Mobile Robot. 2021 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE). 2021 https://doi.org/10.1109/ICMEAE55138.2021.00038
[14] Mudeng, V., Hassanah, B., Priyanto, Y., & Saputra, O. Design and Simulation of Two-Wheeled Balancing Mobile Robot with PID Controller. International Journal of Sustainable Transportation Technology. 2020 https://doi.org/10.31427/ijstt.2020.3.1.3
[15] Lee, Y., Ra, W., & Kim, T. Robust Bearing Angle Error Estimation for Mobile Robots With a Gimballed Ultrasonic Seeker. IEEE Transactions on Industrial Electronics. 2018; 65. https://doi.org/10.1109/TIE.2017.2777411
[16] Lee, A., & Joo, Y. Iterative PID Control Design Approach for Maximizing Proportional Gain. IEEE Access. 2024; 12. https://doi.org/10.1109/ACCESS.2024.3444925
[17] Ang, K., Chong, G., & Li, Y. PID control system analysis, design, and technology. IEEE Transactions on Control Systems Technology. 2005; 13. https://doi.org/10.1109/TCST.2005.847331
[18] Somefun, O., Akingbade, K., & Dahunsi, F. The dilemma of PID tuning. Annu. Rev. Control. 2021; 52. https://doi.org/10.1016/j.arcontrol.2021.05.002
[19] Zhang, J., & Guo, L. Theory and Design of PID Controller for Nonlinear Uncertain Systems. IEEE Control Systems Letters. 2019; 3. https://doi.org/10.1109/LCSYS.2019.2915306
[20] Yamamoto, T., Takao, K., & Yamada, T. Design of a Data-Driven PID Controller. IEEE Transactions on Control Systems Technology. 2009; 17. https://doi.org/10.1109/TCST.2008.921808
[21] Verma, B., & Padhy, P. Robust Fine Tuning of Optimal PID Controller With Guaranteed Robustness. IEEE Transactions on Industrial Electronics. 2020; 67. https://doi.org/10.1109/TIE.2019.2924603
[22] Ren, Y., & , W. Proportional-derivative control and motion stability analysis of a 16-pole legs rotor-active magnetic bearings system. Journal of Vibration and Control. 2024 https://doi.org/10.1177/10775463241276712
[23] Zhang, Z., Ling, Z., & Sarlette, A. Modified Integral Control Globally Counters Symmetry-Breaking Biases. Symmetry. 2019; 11. https://doi.org/10.3390/SYM11050639
[24] Liu, D., Gibaru, O., & Perruquetti, W. Error analysis of Jacobi derivative estimators for noisy signals. Numerical Algorithms. 2011; 58. https://doi.org/10.1007/s11075-011-9447-8
[25] Wang, C., Wang, T., Chen, G., & Pan, J. Automatic Detection and Self-Adjustment Compensation for Mechanical Oscillations and Disturbances in Servo Systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2024; 54. https://doi.org/10.1109/TSMC.2023.3325804
[26] Hägglund, T., & Åström, K. Revisiting the Ziegler-Nichols step response method for PID control. Journal of Process Control. 2004; 14. https://doi.org/10.1016/J.JPROCONT.2004.01.002
[27] He, S., Tan, S., Xu, F., & Wang, P. Fuzzy self-tuning of PID controllers. Fuzzy Sets and Systems. 1993; 56. https://doi.org/10.1016/0165-0114(93)90183-I
[28] Stradner, J., Thenius, R., Zahadat, P., Hamann, H., Crailsheim, K., & Schmickl, T. Algorithmic requirements for swarm intelligence in differently coupled collective systems. Chaos, Solitons, and Fractals. 2013; 50. https://doi.org/10.1016/J.CHAOS.2013.01.011
[29] Cordón, O., Triguero, F., & Stützle, T. A Review on the Ant Colony Optimization Metaheuristic: Basis, Models and New Trends. . 2002; 9.
[30] Dorigo, M. Ant Colony Optimization. IEEE Transactions on Evolutionary Computation. 2001; 8. https://doi.org/10.1109/TEVC.2004.827005
[31] J. K. and R. Eberhart, “Particle Swarm Optimisation,” IEEE Int. Conf. Neural Networks, pp. 1942–1948, 1995, doi: 10.1007/978-3-030-61111-8_2.
[32] R. Rajesh, “Optimal tuning of FOPID controller based on PSO algorithm with reference model for a single conical tank system,” SN Appl. Sci., vol. 1, no. 7, pp. 1–14, 2019, doi: 10.1007/s42452-019-0754-3
[33] Muftah, M. N., Athif, A., Faudzi, M., Sahlan, S. and Shouran, M. (2022) ‘Modeling and Fuzzy FOPID Controller Tuned by PSO for Pneumatic Positioning System’, Energies, 15(3757), pp. 1–19.
[34] M. Shouran and A. Alsseid, “Particle Swarm Optimization Algorithm-Tuned Fuzzy Cascade Fractional Order PI-Fractional Order PD for Frequency

Regulation of Dual-Area Power System," Processes, vol. 10, no. 3, 2022, doi: 10.3390/pr10030477

[35] Vicky Mudeng, Barokatun Hassanah, Yun Tonce Kusuma Priyanto, Okcy Saputra, "Design and Simulation of Two-Wheeled Balancing Mobile Robot with PID Controller," International Journal of Sustainable Transportation Technology, no. 1, April 2020, 12-1912 http://dx.doi.org/10.31427/IJSTT.2020.3.1.3

[36] Y. Zhang, Q. Fei, J. Lu, W. Lei, and Q. Wu, "Performance criteria research on PSO-PID control systems," Proc. - 2010 Int. Conf. Intell. Comput. Cogn. Informatics, ICICCI 2010, no. 2, pp. 316–320, 2010, doi: 10.1109/ICICCI.2010.51.