PARTICLE SWARM OPTIMIZATION OF BACKSTEPPING CONTROL PARAMETERS FOR BALL AND BEAM

Vo-Hoang-Lap Tran [1], Duc-Anh-Quan Nguyen [1], Hoang-Dung Nguyen [1], Anh-Khoa Dang [1], The-Hoang Pham [1], Thai-Hieu-Phong Tran [1], Gia-Bao Nguyen [1], Minh-Phuoc Cu [2]*

[1] Ho Chi Minh City University of Technology and Education (HCMUTE)
01- Vo Van Ngan Street, Ho Chi Minh City, Vietnam
[2] Cao Thang Technical College
65 Huynh Thuc Khang Street, Ben Nghe Ward, District 1, Ho Chi Minh City
* Corresponding author. Email: cuminhphuoc@caothang.edu.vn

Abstract: Throughout the evolution of automatic control, numerous controllers have been developed with the primary goal of stabilizing systems through proven algorithms. One of the most crucial tasks in controller design is optimizing the controller itself. In this paper, our team presents an evolutionary algorithm known as Particle Swarm Optimization (PSO), inspired by the social behavior and cognitive processes of organisms such as fish schools and bird flocks. With the Backstepping controller successfully designed for our ball and beam system, we will use the PSO algorithm to fine-tune the controller's parameters for optimal performance. The ultimate objective is to stabilize the ball and beam system, meaning the ball must remain steady at the desired position. Our team will conduct multiple trials, gathering data from the PSO algorithm, which will allow us to compare the control quality of the solutions we find. The system will be tested both in simulations and in practical experiments to verify its accuracy.
Keywords: Ball and Beam system, Particle Swarm Optimization, PSO, Backstepping controller.

1. Introduction

The Ball and Beam model is a nonlinear system with a complex dynamic structure, often utilized for studying control algorithms. The nonlinear nature of this system demands in-depth understanding of controller design, filtering, and the ability to establish appropriate operational trajectories. Research and development of control algorithms applied to the Ball and Beam system not only carry theoretical significance but also have broad applications in various control engineering fields, such as stability control of unmanned aerial vehicles, missile guidance, and complex robotic systems.

Given the complexity of this model, researchers have developed various control methods to maintain balance or track desired trajectories. One of the prominent control methods is backstepping, renowned for its capability to handle nonlinear systems effectively. However, the implementation of backstepping poses design challenges, requiring the system to be transformed into different forms to ensure stability. This has led to extensive research aimed at refining backstepping, making it more efficient and applicable in practice.

Additionally, combining different control strategies, such as sliding mode control and backstepping, has proven to overcome the limitations of each individual method. This combination mitigates the "chattering" phenomenon often seen in sliding mode control while enhancing the stability of backstepping when dealing with nonlinear systems.

Recently, the development of modern optimizing techniques has opened up new avenues for enhancing the performance of control systems. Among these, PSO has emerged as a powerful tool for fine-tuning control parameters[1]. Introduced by Eberhart and Kennedy in 1995, PSO draws inspiration from the natural behavior of bird flocks and fish schools in their quest for food. Unlike traditional optimization methods such as genetic algorithms (GA), PSO emphasizes the learning and cooperation between individuals within a group. This method strikes a balance between finding the optimal solution globally by leveraging both individual experience and collective information from the entire swarm.

Numerous studies have shown that PSO not only exhibits rapid convergence but is also easily applicable to complex optimization problems. However, PSO has certain limitations, such as being prone to getting trapped in local minima, though these can be overcome through careful tuning of the algorithm's parameters. Subsequent research has proposed several enhancements to PSO[2], such as adjusting the inertia factor and employing variations of the algorithm to improve its ability to escape local minima and broaden its capacity for finding optimal solutions over a larger search space.

2. Mathematical Model

The Ball and Beam system is a SIMO (Single Input – Multiple Output) configuration, in which a ball rolls freely along the length of the beam[3]. The system

consists of two critical components: a motor that controls the rotational angle of a disk, thereby altering the tilt of the beam, and the ball-beam assembly itself. This system operates with two degrees of freedom, where the ball moves translationally along the horizontal axis of the beam, while the beam oscillates vertically with one end fixed. The beam's angle shifts in response to the torque exerted by the motor..

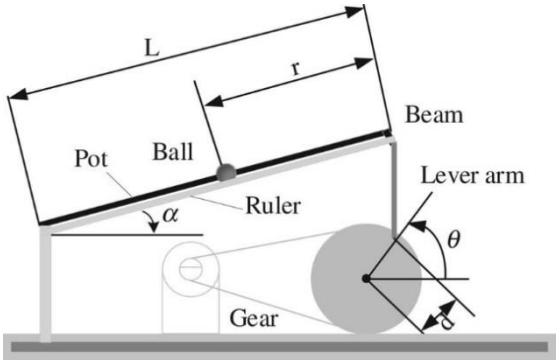

Fig. 1. Modeling the system [4]

To derive the equations describing the motion of the Ball and Beam system, according to reference[5], the Lagrange method based on energy balance is employed[6].

First, we need to determine the kinetic energy of the system, which consists of two components: the kinetic energy of the beam and the kinetic energy of the ball.

Finally, after simplification, the mathematical model of the ball and beam system is expressed as follows:

$$\ddot{r} = \frac{m_b r\dot{\alpha}^2 - m_b g\sin(\alpha)}{m_b + \frac{J_B}{R_B{}^2}} \tag{1}$$

$$\ddot{\alpha} = \frac{\tau_b - [2m_b r\dot{r}\dot{\alpha} + gm_b r\cos(\alpha) + \frac{L}{2}gm_B\cos(\alpha)]}{J_b + r^2 m_b} \tag{2}$$

The parameters obtained from the physical model, depicted in **Fig. 7**, are listed in **Tab.2**.

The angles α and θ are related to each other as described in document [4]. From angle θ, we can infer α using the formula below:

$$\sin(\theta) = \frac{L}{d}\sin(\alpha) \tag{3}$$

To control the torque of the motor, we need to manage its voltage. According to document [7], there is a relationship between torque and voltage. Based on the mathematical equations describing torque and the input voltage of the system from document [7], we proceed to identify the motor parameters to apply the equations to the physical model.

With the relationship between torque and voltage, we substitute into equations (1), (2), and (3) to obtain the following equation:

$$\ddot{r} = \frac{R_B{}^2 m_B r\dot{\alpha}^2 - R_B{}^2 g m_B\sin(\alpha)}{(m_B R_B{}^2 + J_B)} \tag{4}$$

$$\ddot{\alpha} = \frac{\begin{bmatrix} -T_f k_3 + uk_1k_3 - \dot{\alpha}k_2k_3{}^2 + 2\dot{\alpha}m_B r\dot{r} + \ldots \\ \ldots - \frac{1}{2}Lgm_b\cos(\alpha) - gm_B r\cos(\alpha) - C_m\dot{\alpha}k_3{}^2 \end{bmatrix}}{(m_B r^2 + J_b + J_m k_3{}^2)} \tag{5}$$

To linearize the system, it is necessary to construct a state-space model, in which $\sin(\alpha) = \alpha$, $\sin(\theta) = \theta$.

Define the state variables as follows:

$$x_1 = r; x_2 = \dot{r}; x_3 = \alpha; x_4 = \dot{\alpha} \tag{6}$$

$$x = [x_1 \quad x_2 \quad x_3 \quad x_4]^T; y = [x_1 \quad x_3]^T$$

The system in the state-space equations is represented as follows:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \tag{7}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & a_{24} \\ 0 & 0 & 0 & 1 \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}; B = \begin{bmatrix} 0 \\ b_2 \\ 0 \\ b_4 \end{bmatrix} \tag{8}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

We set the values as follows:

$$a_{21} = \frac{\partial\ddot{r}}{\partial x_1}; a_{22} = \frac{\partial\ddot{r}}{\partial x_2}; a_{23} = \frac{\partial\ddot{r}}{\partial x_3}; a_{24} = \frac{\partial\ddot{r}}{\partial x_4};$$

$$a_{41} = \frac{\partial\ddot{\alpha}}{x_1}; a_{42} = \frac{\partial\ddot{\alpha}}{x_2}; a_{43} = \frac{\partial\ddot{\alpha}}{x_3}; a_{44} = \frac{\partial\ddot{\alpha}}{x_4};$$

We can compute the matrices A and B using the model parameters provided in **Tab.2**. The nonlinear system is considered to operate only around an equilibrium point:

$$x_1 = 0.2; x_2 = x_3 = x_4 = 0 \tag{9}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -7.0071 & 0 \\ 0 & 0 & 0 & 1 \\ -1.9239 & 0 & 0 & -6.1373 \end{bmatrix}; B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1.5112 \end{bmatrix} \tag{10}$$

3. Linear Backstepping Controller

The backstepping algorithm referenced from document [8] is applied to system , where we formulate a control law to stabilize the system. According to document [8], the virtual control functions are designed such that the subsystems become stable. Based on

Lyapunov's theory[9], the system's control law is developed through four steps, which are outlined below.

For the ball and beam system, we convert it into a state-space model using state variables.

Step 1: The first step in designing the linear backstepping controller is to define an error variable, which is specified as follows:

$$z_1 = x_1 - \lambda_1 x_3 \tag{11}$$

Wherein, λ_1 represents a constant specific to the system. We compute the derivative of z_1 as follows:

$$\dot{z}_1 = \dot{x}_1 - \lambda_1 \dot{x}_3 = x_2 - \lambda_1 x_4 \tag{12}$$

Step 2: To guide the control law z_1 toward zero, we consider x_2 as a virtual control variable. A positive definite Lyapunov function is selected as follows:

$$V_1 = \frac{{z_1}^2}{2} \geq 0 \tag{13}$$

According to Lyapunov's criterion, an appropriate function must be chosen to satisfy the condition $V_1 \geq 0$. Thus, the derivative of V_1 is given as follows:

$$\dot{V}_1 = -c_1 {z_1}^2 \leq 0 \tag{14}$$

From Appendix (A1), we define the function β_1 in such a way as to ensure system stability as follows:

$$\beta_1 = \lambda_1 x_4 - c_1 z_1 \tag{15}$$

Where c_1 is a positive constant.

Step 3: The next error variable z_2, is defined as follows:

$$z_2 = \lambda_2 x_2 - \beta_1 \tag{16}$$

The derivative of z_2 is calculated as follows:

$$\dot{z}_2 = \lambda_2 \dot{x}_2 + c_1 \dot{z}_1 - \lambda_1 \dot{x}_4 \tag{17}$$

Substituting equation (10) into (17), we obtain:

$$\begin{aligned} \dot{z}_2 &= (a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 + b_2 u)\lambda_2 + c_1 \dot{z}_1 \\ &-(a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 + b_4 u)\lambda_1 \end{aligned} \tag{18}$$

Substituting equation (12) into (18), we obtain:

$$\begin{aligned} \dot{z}_2 &= (a_{21}\lambda_2 - a_{41}\lambda_1)x_1 + (a_{22}\lambda_2 - a_{42}\lambda_1 + c_1)x_2 \\ &+(a_{23}\lambda_2 - a_{43}\lambda_1)x_3 + (a_{24}\lambda_2 - a_{44}\lambda_1 + c_1\lambda_1)x_4 \\ &+(b_2\lambda_2 - b_4\lambda_1)u \end{aligned} \tag{19}$$

We set:

$$\begin{aligned} &h_1 = (a_{21}\lambda_2 - a_{41}\lambda_1);\ h_2 = (a_{22}\lambda_2 - a_{42}\lambda_1 + c_1); \\ &h_3 = (a_{23}\lambda_2 - a_{43}\lambda_1)\ h_4 = (a_{24}\lambda_2 - a_{44}\lambda_1 + c_1\lambda_1); \\ &h_5 = (b_2\lambda_2 - b_4\lambda_1) \end{aligned} \tag{20}$$

Rewrite equation (19) as follows:

$$\dot{z}_2 = h_1 x_1 + h_2 x_2 + h_3 x_3 + h_4 x_4 + h_5 u \tag{21}$$

Step 4: To ensure that the function z_2 is truly stable, we consider V_2. The goal of this step is to select a function V_2 such that $V_2 > 0, \dot{V}_2 < 0$, and the function V_2 must include z_1, z_2, defined as follows:

$$V_2 = \frac{{z_1}^2}{2} + \frac{{z_2}^2}{2} \tag{22}$$

From equation (16) to (12), we have:

$$\dot{z}_1 = z_2 - c_1 z_1 \tag{23}$$

The derivative of V_2 yields the following equation:

$$\dot{V}_2 = z_1 \dot{z}_1 + z_2 \dot{z}_2 \tag{24}$$

Substituting (23) into (24), the equation becomes:

$$\begin{aligned} \dot{V}_2 &= z_1(z_2 - c_1 z_1) + z_2 \dot{z}_2 \\ &= -c_1 {z_1}^2 + z_2(z_1 + \dot{z}_2) \end{aligned} \tag{25}$$

To $\dot{V}_2$ ensure is negative in accordance with Lyapunov stability theory, we define $\dot{z}_2$ as follows:

$$\dot{z}_2 = -z_1 - c_2 z_2 \tag{26}$$

Where c_2 is a positive constant designed for the system. Substituting (26) into (25) $\dot{V}_2$ yields the following form:

$$\dot{V}_2 = -c_1 {z_1}^2 - c_2 {z_2}^2 \tag{27}$$

We substitute equations (11), (16), and (21) into (26). From this equation, we derive the control law as follows

$$u = \frac{\begin{bmatrix} -x_1 + \lambda_1 x_2 + c_2 x_2 + c_2\lambda_1 x_4 - c_1 c_2 x_1 + \\ +c_1 c_2 \lambda_1 x_3 - h_1 x_1 - h_2 x_2 - h_3 x_3 - h_4 x_4 \end{bmatrix}}{h_5} \tag{28}$$

Where c_1, c_2, k_1, k_2 is a design constant of the system

4. Particle Swarm Optimization Algorithm

PSO technique was first introduced to the scientific community by Eberhart and Kennedy in 1995[10], inspired by the social behaviors of animals, particularly birds and fish. These swarms develop cooperative strategies for locating food, leveraging both the individual learning experiences of each member and the collective insights of the entire swarm. The objective function is an essential aspect of the search and optimization process in PSO. It serves to evaluate the fitness of the optimal parameters found in relation to the problem being addressed. In each generation, particle information is aggregated to adjust the velocity across each dimension, allowing the calculation of the particle's new position. PSO is a swarm-based search process, where each individual particle represents a potential solution to the optimization problem within a D-dimensional search space. Each particle is capable of recalling the optimal positions it has found personally, as well as the swarm's global best, along with its velocity. The particles continuously adapt their search trajectories within the multi-dimensional space until they reach either an equilibrium or an optimal state, or exceed computational limits.

The original formula for adjusting velocity and speed, introduced by Eberhart and Kennedy in 2015 (R. Eberhart & Kennedy, 1995), was presented as follows:

$$v_{k+1}^i = v_k^i + c_1 rand_1(\ )(p_k^i - x_k^i) + c_2 rand_2(\ )(g_k^i - x_k^i) \quad (29)$$

$$x_{k+1}^i = x_k^i + v_{k+1}^i \quad (30)$$

Where v_{k+1}^i is the new velocity of the particle at iteration $k+1$; v_k^i is the velocity of the particle from the previous iteration; c_1 is the cognitive component weight; c_2 is the social component weight; p_k^i represents the current personal best fitness value of each particle; and g_k^i is the current global best fitness value of the entire swarm.

In the formula presented (30), the coefficients c_1 and c_2 correspond to the cognitive and social components. If the coefficient c_1 is emphasized, the swarm will concentrate more on the individual's best position within the swarm. Conversely, the swarm tends to explore around the global best position in the swarm's history. The velocity coefficient from the previous cycle affects the swarm's search process; if a particular individual i attains the global best position, the particles will "fly" with zero velocity, meaning the individual will remain stationary until another particle reaches a new global best position.

In the version by Bratton & Kennedy (2007)[11], particles had limited ability to explore new or promising areas, as they relied solely on their previous velocity v_k^i. Therefore, in 1998, authors Shi & Rc (1998)[12] researched and introduced an inertia weight coefficient w before the previous velocity to balance local and global search. This modification was presented as follows:

$$v_{k+1}^i = wv_k^i + c_1 rand_1(\ )(p_k^i - x_k^i) + c_2 rand_2(\ )(g_k^i - x_k^i) \quad (31)$$

$$x_{k+1}^i = x_k^i + v_{k+1}^i \quad (32)$$

A higher inertia weight w allows particles to maintain momentum while exploring new regions of the search space. Conversely, a lower w encourages local exploitation of the vicinity around the current solution. Additionally, in 2000, as presented in the work by R. C. Eberhart & Shi (2000)[13], a method was proposed to enhance the swarm's convergence through the introduction of a constriction factor:

$$v_{k+1}^i = \chi \begin{pmatrix} v_k^i + c_1.rand_1(\ )\left(p_k^i - x_k^i\right) \\ + c_2.rand_2(\ )\left(g_k^i - x_k^i\right) \end{pmatrix} \quad (33)$$

The constriction factor was incorporated into equation (33) as follows:

$$\chi = \frac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|}; \phi = c_1 + c_2 \quad (34)$$

In their 2007 study, Bratton & Kennedy (2007) discovered that after extensive trials, as the swarm searches for a solution, the best solution found in the search space tends to gradually converge. However, there is no guarantee of complete convergence. Conversely, when methods to ensure convergence were implemented, the swarm often converged more rapidly. Typically, constants are used to guarantee convergence, with their values being adjusted accordingly.

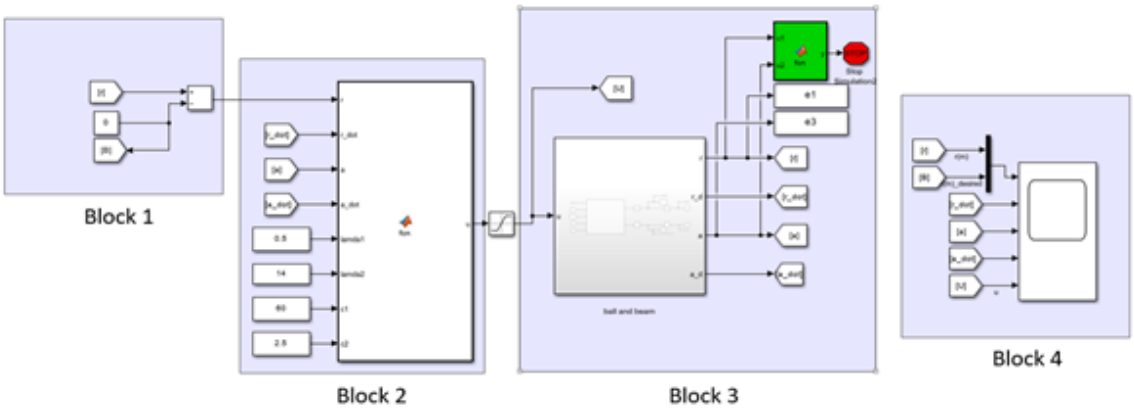

Fig. 2. Ball and beam in Matlab Simulink

Optimization functions have various definitions. In most PSO algorithms, research papers commonly propose methods like the Integral of Absolute Error (IAE) or the Integral of Time Multiply Absolute Error (ITAE), among other criteria, as presented in Chaudhari (2013)[14]. In this paper, the authors applied the Sum of Square Error (SSE) criterion, which is referenced in the work of Lu, Xie, & Zhou (2016)[15]:

$$SSE = \sum_{k=1}^{M} \left(y_d - y(k)\right)^2 \quad (35)$$

Where:

- y_d : Desired signal to be achieved.
- $y(k)$: Actual output signal of the model.
- M: Number of available data points.

The flowchart of the swarm algorithm is presented below:

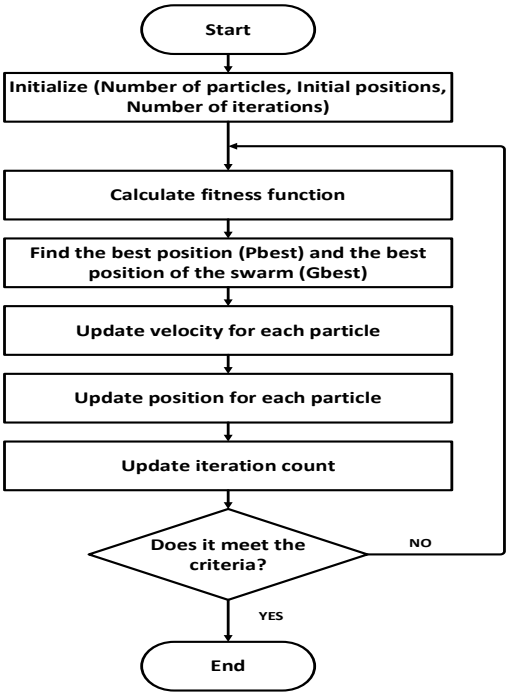

Fig. 3. Block diagram

5. Simulation and Experimentation

5.1 Simulation

In this section, the research team will simulate the system using a backstepping controller that requires parameter optimization.

Utilizing MATLAB, we proceed with the optimization of the backstepping controller's parameters Using the PSO algorithm, we aim to determine the four parameters of the controller. The initial system values are presented below:

$$[r_0 \quad \dot{r}_0 \quad \alpha_0 \quad \dot{\alpha}_0]^T = [0.542 \quad 0 \quad -0.1416 \quad 0]^T$$

Additionally, the motor input voltage is constrained within a certain range $[-12 \quad 12]V$. The system's sampling time is 0.01s, and the simulation is conducted over 25 seconds.

Below are the simulation results, showing the system's behavior with the parameters obtained r, α, u via the PSO algorithm

The above are the functional blocks that the authors have developed based on MATLAB simulation. These include four blocks with the following respective functions:

Block 1: This block generates reference signals for the system, such as sine waves, square pulses, and more.

Block 2: This is the backstepping controller, designed to process input data and produce control signals for the actuator, ensuring the system reaches the desired position.

Block 3: Representing the system's actuator, this block models the system's movement in response to the control inputs.

Block 4: The final block is a scope, which captures data from Matlab Simulink and provides a way to visualize it.

Tab. 1.Controller parameters and adaptive parameters

No.	λ_1	λ_2	c_1	c_2	*Js*
1	0.5	14	60	2.5	8.8100
2	0.8	20	70	5	9.1469
3	1	14	50	10	9.2688
4	1	30	60	5	7.0129
5	1	50	66	8	9.8524

In **Tab. 1**, the adaptability of the parameters identified using the PSO algorithm is presented. After multiple parameter searches for the controller, we identified the top five optimal parameter sets, labeled as Trial_1, Trial_2, Trial_3, Trial_4, and Trial_5, which are depicted in **Fig. 4** to **Fig. 6**.

However, a certain degree of error remains due to the fact that our controller is linear, resulting in an error of 0.03 meters. Despite this, the controller parameters are still highly effective, as they stabilize the system with a rise time of 4 seconds and minimal overshoot. Due to the existence of this error, the fitness function value remains relatively large. The system's stability was analyzed according to equation.

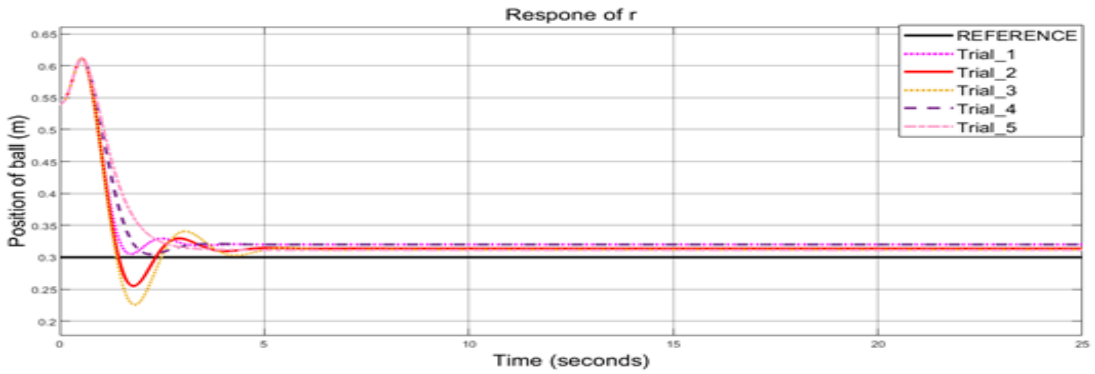

Fig. 4. Position of Ball and beam(m)

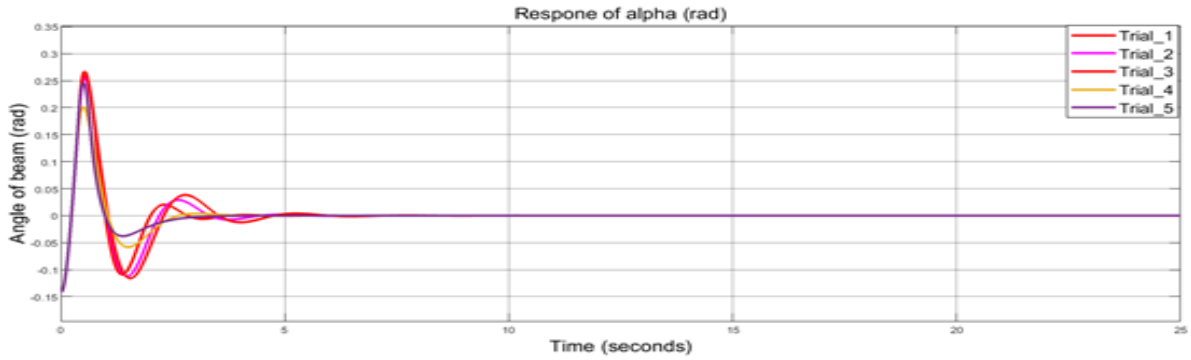

Fig. 5. Angle of beam(rad)

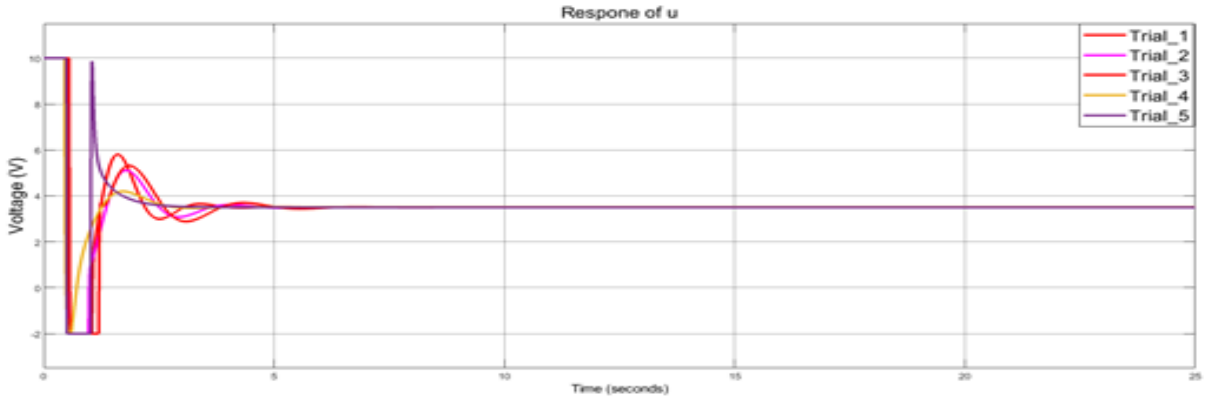

Fig. 6. Control signal

5.2. Experimentation

Below is the fully developed hardware model of the ball and beam system, which has been subjected to experimentation.

Tab. 2. System Specifications

	Parameter	**Value**	**Unit**
$r(t)$	Ball position		
$\alpha(t)$	Beam Angle		
d_t	Gear ratio	5.6	
τ_b	Torque apply to beam		$N \cdot m$
L	Beam length	0.54	m
m_B	Ball mass	0.065	Kg
m_b	Beam mass	0.34	Kg
g	Gravitation acceleration	9.81	m/s^2
d	Lever radius	0.075	m
J_B	Moment of inertia of the ball	$\frac{2}{5} m_B R_B^{\,2}$	$Kg \cdot m^2$
J_b	Moment of inertia of the beam	$\frac{1}{3} m_b L^2$	$Kg \cdot m^2$
R_B	Ball radius	0.0125	m
R_m	Motor resistance	6.83527	Ω
K_t	Torque constant	0.064943	$N \cdot m/s^2$
K_b	Back EMF constant	0.064943	$V \cdot s/rad$
C_m	Coeffcient of viscous friction	0.00034	$N \cdot m/(rad/s)$
J_m	Moment of inertia of Rotor	0.000134	$Kg \cdot m^2$
T_f	Moment friction	0.010764	$N \cdot m$
u	Voltage Input		V
K_1	Constant	$\frac{K_t}{R_m}$	
K_2	Constant	$\frac{K_t K_b}{R_m}$	
K_3	Constant	$\frac{d_t L}{d}$	

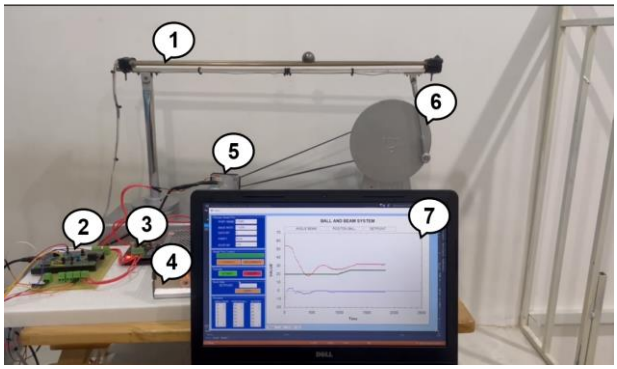

Fig. 7. Physical model

The components of the actual system are divided into six parts, as shown in **Fig. 7**, which include:

1. Beam
2. DC Motor
3. STM 32F407VG Microcontroller
4. HI216 H-Bridge
5. 220 VAC Power Supply
6. Lever Arm
7. Graphical user interface

The results collected from the practical experiments are presented in the table below.

Tab. 3. Controller parameters and adaptive parameters

No.	λ_1	λ_2	c_1	c_2	*Js*
1	0.5	14	60	2.5	11.2749
2	0.8	20	70	5	11.7081
3	1	14	50	10	11.4059
4	1	30	60	5	10.4897
5	1	50	66	8	11.6445

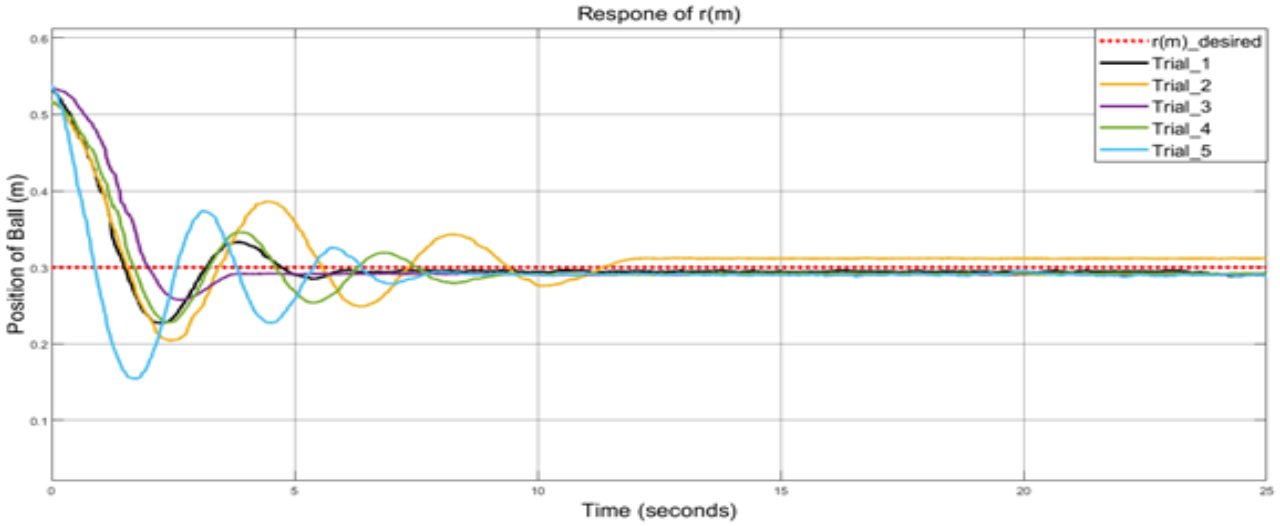

Fig. 8. The position of the ball during experimentation

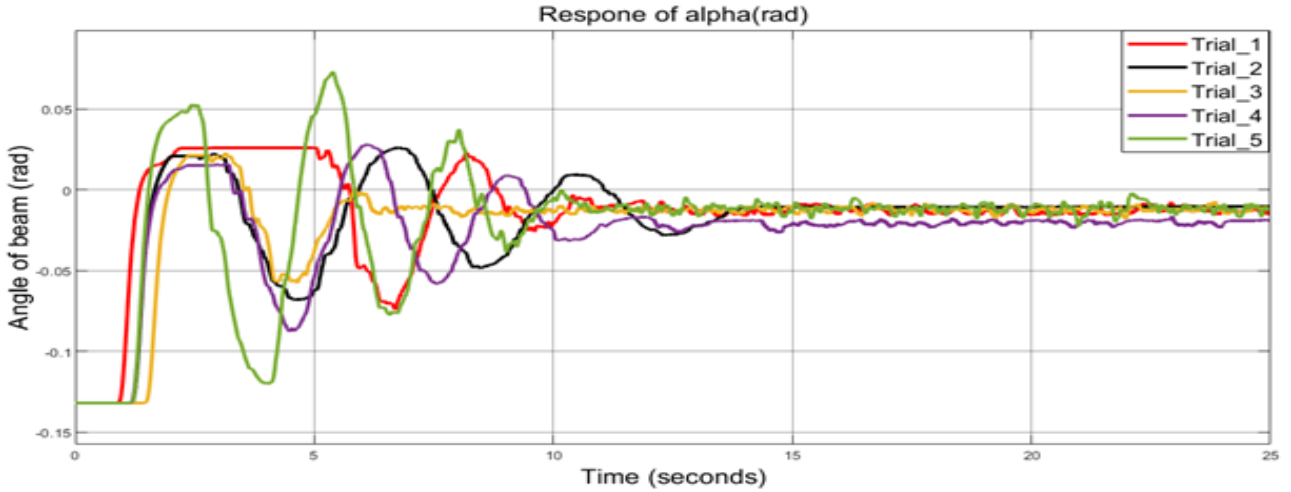

Fig. 9. The angle of the beam during experimentation

The output response of the parameters is evaluated based on the standard (35), and the results obtained are presented in **Tab. 3**, as well as **Fig. 8** and **Fig. 9**. These figures demonstrate that the use of the optimization swarm has improved the model's response according to the variation of each calculated fitness function. All four sets of parameters yielded successful experimental outcomes. However, for smaller fitness functions, the model's stability increases, as shown in **Fig. 8** and **Fig. 9**.

6. Conclusions

The paper provides an in-depth evaluation of the adaptability of control parameters in the nonlinear Backstepping controller through the application of the PSO algorithm. The assessment is conducted by tracking the evolution of the fitness function, revealing the potential to enhance response performance and improve control quality as the fitness function decreases. Simulation and experimental results on a real system demonstrate the algorithm's effectiveness and confirm PSO's capability to identify the optimal solution for the control problem. This research opens new avenues for exploration in control engineering, while

also inspiring further advancements and refinements in optimization algorithms for future applications.

Acknowledgement

This paper belongs to project of student of Ho Chi Minh City University of Technology and Education (HCMUTE) and is funded by HCMUTE. We also want to give thanks to Assoc. PhD. Minh-Tam Nguyen and PhD. Van-Dong-Hai Nguyen due to their supervision for us to achieve this project.

7. References

[1] Rana M.A. et al: "Automatic control of ball and beam system using particle swarm optimization", IEEE 12th International Symposium on Computational Intelligence and Informatics, pp. 529-534, 2011.
[2] Ali H. et al: "Optimization of PID parameters based on Particle Swarm optimization for ball and beam system", International Journal of Engineering Technologies and Management Research, vol. 5, no. 9, pp. 59-69, 2018.
[3] Bolívar-Vincenty C.G., Beauchamp-Báez G.: "Modelling the ball-and-beam system from newtonian mechanics and from lagrange methods", Twelfth LACCEI Latin American and Caribbean Conference for Engineering and Technology, vol. 22, p. 24, 2014.
[4] Shekhawat R.S., Singh N.: "Application of Reinforcement Learning in Control Systems for Designing Controllers", in Machine Intelligence and Smart Systems, chapter 9, pp. 117-128, 2022.
[5] Maalini P.M. et al: "Modelling and control of ball and beam system using PID controller", in 2016 International Conference on Advanced Communication Control and Computing Technologies, IEEE, pp. 322-326, 2016.
[6] Keshmiri M. et al: "Modeling and control of ball and beam system using model based and non-model based control approaches", International Journal on smart sensing and intelligent systems, vol. 5, no. 1, pp. 14-35, 2012.
[7] Hamza B. et al: "A New method for the parametric identification of DC machines using MATLAB identification toolbox and experimental measurements", E3S Web of Conferences, vol. 336: EDP Sciences, p. 00007, 2022.
[8] Vo M.T. et al: "Back-stepping control for rotary inverted pendulum", Journal of Technical Education Science, vol. 15, no. 4, pp. 93-101, 2020.
[9] Aguilar-Ibañez C.: "On the stabilization of the ball and beam system using a direct Lyapunov method", in 2009 6th International Conference on Electrical Engineering, Computing Science and Automatic Control, IEEE, pp. 1-6, 2009.
[10] Eberhart R., Kennedy J.: "A new optimizer using particle swarm theory", in MHS'95. Proceedings of the sixth international symposium on micro machine and human science, IEEE, pp. 39-43, 1995.
[11] Poli R. et al: "Theoretical derivation, analysis and empirical evaluation of a simpler particle swarm optimiser", Congress on Evolutionary Computation, IEEE, pp. 1955-1962, 2007.
[12] Eberhart R.C., Shi Y.: "Comparison between genetic algorithms and particle swarm optimization", International conference on evolutionary programming, Springer, pp. 611-616, 1998.
[13] Eberhart R.C., Shi Y.: "Comparing inertia weights and constriction factors in particle swarm optimization", Proceedings of the 2000 congress on evolutionary computation. CEC00 (Cat. No. 00TH8512), IEEE, vol. 1, pp. 84-88, 2000.
[14] Chaudhari Y.: "Design and implementation of intelligent controller for a continuous stirred tank reactor system using genetic algorithm", International Journal of Advances in Engineering & Technology, vol. 6, no. 1, p. 325, 2013.
[15] Lu J. et al: "Combined fitness function based particle swarm optimization algorithm for system identification", Computers & Industrial Engineering, vol. 95, pp. 122-134, 2016.