LOAD VARIATION ESTIMATION AND HIERARCHICAL SLIDING MODE CONTROL FOR PENDUBOT USING SWARM ALGORITHM

Thanh-Khang Tran [1], Quang–Thien Nguyen [1], Duc-Anh-Quan Nguyen [2], Xuan-Dung Huynh [3], Dinh-Phu Nguyen [1], Thi-Bich-Nga Truong [1], Nguyen-Phuong-Thao Do [1], Anh-Huy Nguyen [2*]

[1]Ho Chi Minh City University of Technology and Education (HCMUTE), Ho Chi Minh City (HCMC), Vietnam
Vo Van Ngan Str., No. 01, HCMC, Vietnam

[2] Ho Chi Minh City University of Technology (HCMUT), HCMC, Viet Nam
No. 268, Ly Thuong Kiet Str. HCMC, Vietnam

[3] Cao Thang Technical College
Huynh Thuc Khang Str., No. 65, HCMC, Vietnam

***** Corresponding author: huy.nguyen141@hcmut.edu.vn

Abstract: This study integrates varying-load estimation with hierarchical sliding-mode control (HSMC) for a Pendubot using a swarm-intelligence algorithm. A nonlinear Pendubot model with static load parameters is optimized to minimize estimation error, and the resulting estimates inform an HSMC scheme that compensates disturbances and uncertainties—simulations show marked improvements in stability, accuracy, and control efficiency.
Keywords: Pendubot, Sliding Mode Control (SMC), Trajectory Tracking, Particle Swarm Optimization (PSO).

1. Introduction

Pendubot—a two-link, underactuated, highly nonlinear benchmark system [1,2]—experiences significant performance degradation when subjected to uncertain, time-varying loads [3]. Traditional parameter estimation techniques often fail to cope with their complex dynamics and environmental disturbances, leading to suboptimal control actions [4]. Meanwhile, Hierarchical Sliding-Mode Control (HSMC) is celebrated for its robustness against external perturbations and its capacity to guarantee global stability, but its efficacy hinges on the accuracy of the system's parameter values [5,6]. Swarm-intelligence algorithms such as Particle Swarm Optimization (PSO) offer powerful global search capabilities and adaptability, making them well-suited for nonlinear parameter estimation challenges [7–9]. In this work, we present a novel framework that tightly couples PSO-based estimation of varying load parameters with an HSMC scheme. By iteratively refining load estimates and feeding them into the sliding-mode controller, our approach achieves superior disturbance rejection, faster convergence, and markedly improved trajectory tracking—an outcome validated through comprehensive MATLAB simulations.

2. Mathematical

2.1 Modeling the Pendubot System

Pendubot—a two-link, underactuated robot with one motorized and one freely rotating joint—exhibits strong nonlinear dynamics that challenge controller design [3,9]. As a benchmark in automatic control research, it underpins the development and evaluation of advanced strategies such as HSMC, LQR, and adaptive parameter estimation, offering key insights into stability and disturbance mitigation in nonlinear systems.

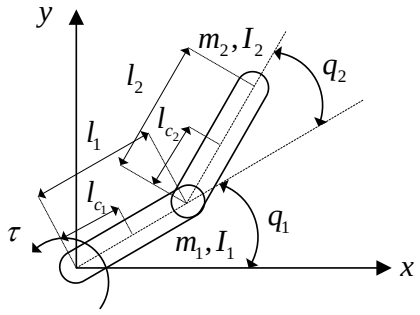

Fig. 1. Pendubot structure.

Parameters of Pendubot is shown in Tab. 1 below

Tab. 1. Physical parameters of the Pendubot.

Description	Symbol	Param.
Mass of the first link	m_1	$0.11129kg$
Mass of the second link	m_2	$0.02929kg$
Distance from the pivot to the arm link's center of mass	l_{c_1}	$0.13063m$
Distance from the pivot to the pendulum's center of mass	l_{c_2}	$0.1\,\mathrm{m}$
Gravitational acceleration	g	$9.8\frac{m}{s^2}$
Friction coefficient associated with the arm link	b_1	0.0067476
Friction coefficient associated with the pendulum rod	b_2	9.8684e-10

Description	Symbol	Param.
Moment of inertia of link 1	I_{c_1}	6.33e-4kg.m^2
Moment of inertia of link 2	I_{c_2}	1e-4kgm^2
Motor Resistance	R_m	Ω
Back-EMF Constant	K_b	Vs/rad
Torque Constant	K_t	Nm/A
Rotor Moment of Inertia	J_m	$kg.m^2$
Viscous Friction Coefficient	C_m	Nm.s/rad
Resisting Torque	τ_m	$N.m$
Orientation of link 1 with respect to the horizontal line	q_1	rad
Angular position of link 2 relative to link 1	q_2	rad
Rotational speed of the arm segment	$\dot{q}_1$	$\frac{rad}{s}$
Rotational speed of the arm segment	$\dot{q}_2$	$\frac{rad}{s}$
Rate of change of angular velocity for the arm segment	$\ddot{q}_1$	$\frac{rad}{s^2}$
Rate of change of angular velocity for the pendulum segment	$\ddot{q}_2$	$\frac{rad}{s^2}$

The system dynamics are derived using the Euler–Lagrange method—one of the two principal approaches (alongside Newton–Euler)—chosen here for its flexibility and suitability.

$$\begin{cases} \dfrac{d}{dt}\left(\dfrac{\partial L}{\partial \dot{q}_1}\right)-\dfrac{\partial L}{\partial q_1}=\tau \\ \dfrac{d}{dt}\left(\dfrac{\partial L}{\partial \dot{q}_2}\right)-\dfrac{\partial L}{\partial q_2}=0 \end{cases} \quad (1)$$

In Lagrangian mechanics, the Lagrangian (L) of a system is defined as the difference between the kinetic energy (K) and the potential energy (U) of the system, expressed by the formula:

$$L=\sum K-\sum U \quad (2)$$

The authors define the notations that will be used in this manuscript: $\cos q_1 = c_{q_1}$, $\sin q_1 = s_{q_1}$.

Based on Eq. (2), the kinetic energy and potential energy components are defined as follows:

$$K=K_1+K_2 \quad (3)$$

$$U=U_1+U_2 \quad (4)$$

where:

$$K_1=\frac{1}{2}\left(I_1+m_1 l_{c_1}{}^2\right)\dot{q}_1{}^2 \quad (5)$$

$$\begin{aligned} K_2=&\frac{1}{2}\left(I_2+m_2 l_1 l_{c_2} c_{q_2}+m_2 l_{c_2}{}^2+m_2 l_1{}^2\right)\dot{q}_1{}^2 \\ &+\left(I_2+m_2 l_1 l_{c_2} c_{q_2}+m_2 l_{c_2}{}^2\right)\dot{q}_1\dot{q}_2 \\ &+\frac{1}{2}\left(I_2+m_2 l_{c_2}{}^2\right)\dot{q}_2{}^2 \end{aligned} \quad (6)$$

$$U_1=m_1 l_{c1}\, s_{q_1}\, g \quad (7)$$

$$U_2=m_2\left[l_1 s_{q_1}+l_{c_2}\, s_{q_1 q_2}\right]g \quad (8)$$

From the components defined in Eq. (3) and (4), substituting them into the Lagrange equation Eq. (2), we obtain the Lagrangian function.

The equations of motion obtained from the Lagrangian function are as follows:

$$\begin{aligned} &\begin{cases} \beta_1\ddot{q}_1+\beta_2\ddot{q}_1+\beta_2\ddot{q}_2+2\beta_3\ddot{q}_1\, c_{q_2} \\ -2\beta_3\dot{q}_2\ddot{q}_1\, s_{q_2}+\beta_3\ddot{q}_2\, c_{q_2}-\dot{\beta}_3\dot{q}_2 s_{q_2} \\ +\beta_4 g c_{q_1}+\beta_5 g c_{q_1 q_2} \\ \beta_2\ddot{q}_1+\beta_2\ddot{q}_2+\beta_3\ddot{q}_1\, c_{q_2}-\beta_3\dot{q}_2\dot{q}_1\, s_{q_2} \\ +\beta_3\ddot{q}_2\, c_{q_2}+\beta_3 s_{q_2}\dot{q}_1{}^2+\beta_3 s_{q_2}\dot{q}_1\dot{q}_2 \\ +\beta_5 g c_{q_1 q_2} \end{cases} \\ &=\begin{cases} \tau-b_1\dot{q}_1 \\ -b_2\dot{q}_2 \end{cases} \end{aligned} \quad (9)$$

Where:

$$\beta_1=m_1 l_{c_1}{}^2+m_2 l_1{}^2+I_{c_1}, \beta_2=m_2 l_{c_2}{}^2+I_{c_2}$$

$$, \beta_3=m_2 l_1 l_{c_2}, \beta_4=m_1 l_{c_1}+m_2 l_{c_1}, \beta_5=m_2 l_{c_2}$$

The equation of motion in Eq. (9) can be rewritten in matrix form as follows:

$$\boldsymbol{M}\ \boldsymbol{q}\ \ddot{\boldsymbol{q}}+\boldsymbol{C}\ \boldsymbol{q},\dot{\boldsymbol{q}}\ \dot{\boldsymbol{q}}+\boldsymbol{G}(\boldsymbol{q})=\boldsymbol{\tau} \quad (10)$$

Where $\boldsymbol{q}=\begin{bmatrix} q_1 & q_2 \end{bmatrix}^T \in \mathbb{R}^{2x1}$ are the output control variables of the system, which are the angles of deviation that need to be controlled as desired. The velocity of the changes in the joint variables is considered as the vector $\dot{\boldsymbol{q}}=\begin{bmatrix} \dot{q}_1 & \dot{q}_2 \end{bmatrix}^T \in \mathbb{R}^{2x1}$, similarly with the acceleration vector $\ddot{\boldsymbol{q}}=\begin{bmatrix} \ddot{q}_1 & \ddot{q}_2 \end{bmatrix}^T \in \mathbb{R}^{2x1}$. The input torque that controls the system is represented by the vector $\boldsymbol{\tau}=\begin{bmatrix} \tau & 0 \end{bmatrix}^T$. $\mathbf{M} \in \mathbb{R}^{2\times 2}$ is the inertia matrix, $\boldsymbol{C}\ \boldsymbol{q},\dot{\boldsymbol{q}}$ is the Coriolis matrix, and $\boldsymbol{G}(\boldsymbol{q})$ is the gravitational vector with dimensions $\mathbf{G} \in \square^{2\times 1}$.

The matrices in Eq. (10) are presented as follows:

$$\boldsymbol{M}(\boldsymbol{q})=\begin{bmatrix} \beta_1+\beta_2+2\beta_3\, c_{q_2} & \beta_2+\beta_3\, c_{q_2} \\ \beta_2+\beta_3\, c_{q_2} & \beta_2 \end{bmatrix} \quad (11)$$

$$\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})=\begin{bmatrix} -\beta_3 s_{q_2}\dot{q}_2 & -\beta_3 s_{q_2}\dot{q}_2-\beta_3 s_{q_2}\dot{q}_1 \\ \beta_3 s_{q_2}\dot{q}_1 & 0 \end{bmatrix} \quad (12)$$

$$\boldsymbol{G}(\boldsymbol{q})=\begin{bmatrix} \beta_4 g\, c_{q_1}+\beta_5 g\, c_{q_1 q_2} \\ \beta_5 g\, c_{q_1 q_2} \end{bmatrix} \quad (13)$$

The motor input in Eq. (10) is in the form of torque. In practice, when experimenting with the Pendubot model, it is necessary to use motors to generate the force supplied to the system. Therefore, it is essential to design an equation that relates the input torque to the input voltage. In [10], the authors discuss the conversion between τ and u (voltage), which facilitates the correspondence between simulation and experimentation.

The input voltage to the torque:

$$\tau_m = -J_m\ddot{q}_1 - \beta_6\dot{q}_1 + \beta_7 e \tag{14}$$

Where: $\beta_6 = \mathrm{C}_m + \frac{K_t}{R_m}K_b; \beta_7 = \frac{K_t}{R_m}$.

Thus, based on Eq. (14) substituting it into Eq. (10), we obtain the complete dynamic equation with the input being the voltage supplied to the DC motor, which can be presented as follows:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{e} \tag{15}$$

The components of the matrix have the following form:

$$\boldsymbol{M(q)} = \begin{bmatrix} \beta_1 + \beta_2 + 2\beta_3 \mathrm{c}_{q_2} + J_m & \beta_2 + \beta_3 \mathrm{c}_{q_2} \\ \beta_2 + \beta_3 \mathrm{c}_{q_2} & \beta_2 \end{bmatrix}$$

$$\boldsymbol{C(q,\dot{q})} = \begin{bmatrix} -\beta_3 \mathrm{s}_{q_2}\dot{q}_2 + \beta_6 & -\beta_3 \mathrm{s}_{q_2}\dot{q}_2 - \beta_3 \mathrm{s}_{q_2}\dot{q}_2 \\ \beta_3 \mathrm{s}_{q_2}\dot{q}_2 & 0 \end{bmatrix}$$

$$\boldsymbol{G(q)} = \begin{bmatrix} \beta_4 g c_{q_2} + \beta_5 g c_{q_1 q_2} \\ \beta_5 g c_{q_1 q_2} \end{bmatrix}$$

$$\boldsymbol{e} = \begin{bmatrix} \beta_7 e \\ 0 \end{bmatrix}$$

2.2 The State Equations of the System

We transform the system into the form of differential equations based on Eq. (15):

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}\ \boldsymbol{q}^{-1}\left[\boldsymbol{e} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{G}\ \boldsymbol{q}\right] \tag{16}$$

We define the state variables as follows:

$$\boldsymbol{x} = \begin{matrix} x_1 & x_2 & x_3 & x_4 \end{matrix} \tag{17}$$

In which:

$$x_1 = q_1, x_2 = \dot{q}_1, x_3 = q_2, x_4 = \dot{q}_2 \tag{18}$$

Taking the derivative of the state variables, we obtain the state equation in the form:

$$\begin{cases} \dot{x}_1 = \dot{q}_1 \\ \dot{x}_2 = \ddot{q}_1 = f_1\ \boldsymbol{q},\dot{\boldsymbol{q}}\ + g_1\ \boldsymbol{q},\dot{\boldsymbol{q}}\ e \\ \dot{x}_3 = \dot{q}_2 \\ \dot{x}_4 = \ddot{q}_2 = f_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ + g_1\ \boldsymbol{q},\dot{\boldsymbol{q}}\ e \end{cases} \tag{19}$$

3. Design of the Hierarchical Sliding Mode Controller

3.1. Hierarchical Sliding Mode Control Law

SMC forces errors onto a sliding surface for robust stability (using smoothing functions like Sat() or Tanh() to reduce chattering), and HSMC adds hierarchical layers of these surfaces to handle more complex systems.

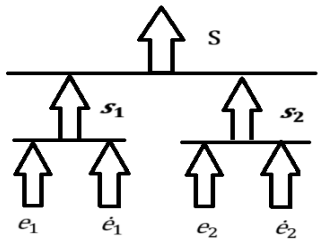

Fig. 2. Sliding surface structure.

Based on Fig. 2, we construct the sliding surface for the first layer:

$$s_1 = \alpha_1 e_1 + \dot{e}_1 \tag{20}$$

$$s_1 = \alpha_1 e_1 + \dot{e}_1 \tag{21}$$

Where α_1, α_2 are the design constants for the sliding surface in the first layer, and the trajectory tracking errors are e_1, e_2, which represent the difference between the signal q and q_d. Here, q_d is the desired reference trajectory.

Taking derivative of sliding surface in the first layer in Eq. (20)and Eq. (21), we obtain the following:

$$\dot{s}_1 = \alpha_1\dot{e}_1 + \ f_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ + g_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ e - \ddot{q}_{1_d} \tag{22}$$

$$\dot{s}_2 = \alpha_2\dot{e}_2 + \ f_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ + g_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ e - \ddot{q}_{2d} \tag{23}$$

The control components of the Hierarchical Sliding Mode Controller (HSMC) are presented as follows:

$$u_{HSMC} = \sum u_{eq} + u_{sw} \tag{24}$$

Here, u_{eq} denotes the equivalent control component keeping the system on the sliding surface, while u_{sw} is the switching control term—typically involving a sign() function and an exponential law—to drive the states rapidly toward that surface.

The equivalent control component u_{eq} is obtained by solving Equations (22) and (23) under the condition that $\dot{s}_1 = 0$ and $\dot{s}_2 = 0$. After solving these equations, the control components are determined as follows:

$$u_{eq_1} = \frac{-\alpha_1\dot{e}_1 - f_1\ \boldsymbol{q},\dot{\boldsymbol{q}}\ + \ddot{q}_{1d}}{g_1\ \boldsymbol{q},\dot{\boldsymbol{q}}} \tag{25}$$

$$u_{eq_2} = \frac{-\alpha_2\dot{e}_2 - f_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ + \ddot{q}_{2d}}{g_2\ \boldsymbol{q},\dot{\boldsymbol{q}}} \tag{26}$$

The overall sliding surface is defined as follows:

$$s = \gamma_1 s_1 + \gamma_2 s_2 \tag{27}$$

Taking the derivative of Eq. (27), we obtain:

$$\begin{aligned} \dot{s} = \gamma_1 &\left(\alpha_1\dot{e}_1 + \left(\begin{matrix} f_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ + g_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ \\ u_{eq_1} + u_{sw}\ - \ddot{q}_{1_d}\ \end{matrix} \right) \right) \\ + \gamma_2 &\left(\alpha_2\dot{e}_2 + \left(\begin{matrix} f_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ + g_2\ \boldsymbol{q},\dot{\boldsymbol{q}}\ \\ u_{eq_1} + u_{sw}\ - \ddot{q}_{1_d}\ \end{matrix} \right) \right) \end{aligned} \tag{28}$$

On the overall sliding surface, the control u of each sliding surface is analyzed, including the equivalent and switching components

We define:

$$\dot{s} = -ks + \mu s ign\ s \tag{29}$$

By combining equations (28) and (29), we obtain the switching control component:

$$u_{sw} = \frac{-ks + \eta sign\ s\ - \gamma_1 g_1\ \boldsymbol{q}, \dot{\boldsymbol{q}}\ u_{eq_1}}{\gamma_1 g_1\ \boldsymbol{q}, \dot{\boldsymbol{q}}\ + \gamma_2 g_2\ \boldsymbol{q}, \dot{\boldsymbol{q}}} \tag{30}$$

By combining the three control components in Eq. (25), (26) and (30) and substituting them into Eq. (24) a main control signal for the entire system is presented as follows:

$$u_{HSMC} = \frac{-ks + \eta sign\ s\ + \gamma_2 g_2\ \boldsymbol{q}, \dot{\boldsymbol{q}}\ u_{eq_1}}{\gamma_1 g_1\ \boldsymbol{q}, \dot{\boldsymbol{q}}\ + \gamma_2 g_2\ \boldsymbol{q}, \dot{\boldsymbol{q}}} \tag{31}$$

3.2. Stability Analysis

To analyze the stability of a nonlinear system, we define a Lyapunov energy function in the quadratic form $V \geq 0 \forall s \neq 0$. This stability function is related to the overall sliding surface:

$$V = \frac{1}{2} s^2 \tag{32}$$

Taking the derivative of Eq. (32), we obtain:

$$\begin{aligned} \dot{V} &= s\dot{s} \\ &= s\ -ks - \eta sign\ s \\ &= -ks^2 - \eta |s| \end{aligned} \tag{33}$$

In Eq. (33), we can conclude that $\dot{V} \leq 0 \forall s \neq 0$. According to Lyapunov's theorem, we can conclude that the system is a energy function with a time-decreasing derivative, where the sliding surfaces, which include the system errors, will also tend towards zero as time approaches infinity.

We proceed to integrate Eq. (33) and obtain:

$$\int_0^t \dot{V} = \int_0^t -ks^2 - \eta |s|\ dt = V\ t\ - V\ 0 \tag{34}$$

Eq. (34) can be deduced as follows:

$$V\ 0\ = V\ t\ + \int_0^t ks^2 + \eta |s|\ dt \geq \int_0^t ks^2 + \eta |s|\ dt \tag{35}$$

Thus, we can conclude that:

$$\int_0^t ks^2 + \eta |s|\ dt \leq V\ 0 \tag{36}$$

According to Barbalat's lemma, it can be concluded that:

$$\lim_{t \to 0} \int_0^t ks^2 + \eta |s|\ dt = 0 \tag{37}$$

This demonstrates that, with Eq. (37) having been proven, the overall sliding surface will converge to zero, which proves that the system will converge and achieve global asymptotic stability:

$$\lim_{t \to 0} s = 0 \tag{38}$$

4. Particle Swarm Optimization

4.1. Particle Swarm Optimization Algorithm

Particle Swarm Optimization (PSO), introduced by Eberhart and Kennedy [12], mimics the decentralized, collective behavior of birds or bees to solve complex, high-dimensional optimization problems. Each "particle" in a D-dimensional search space updates its velocity and position based on its own best solution and the swarm's global best, iterating until convergence or computational limits are reached. PSO's robust, self-organizing mechanism—and its variants like ACO and Bee Colony Optimization—has been validated through extensive empirical studies [13].

The optimal position of an individual (i.e., the optimal position that the element has passed through):

$$p_{t+1}^i = \begin{cases} x_{k+1}^i, f\ \ X_{t+1}^i < p_k^i \\ p_k^i, otherwise \end{cases} \tag{39}$$

where:

x_{k+1}^i : Individual i at iteration k+1.

p_k^i (pBest): Is the best individual at iteration k.

At each iteration, particles update their velocity (motion vector) and position in the swarm, as originally formulated by Eberhart and Shi [13]:

$$\begin{cases} v_{k+1}^i = w v_k^i + c_1 rand_1()\ \ p_k^i - x_k^i \\ \qquad\qquad + c_2 rand_2()\ \ g_k^i - x_k^i \\ x_{k+1}^i = x_k^i + v_{k+1}^i \end{cases} \tag{40}$$

where:

w : Inertia weight.

v_{k+1}^i : Velocity of particle i at iteration k+1.

v_k^i : Velocity of particle i at iteration k.

x_k^i : Position of particle i at iteration k.

x_{k+1}^i : Position of particle i at iteration k+1.

g_k^i : Best position of the swarm.

p_k^i : Represents the best position that particle i has achieved up to iteration k.

$c_{1,2}$: Weight factor.

Despite persistent oscillations and no guaranteed convergence, Eberhart and Shi introduced a constriction factor to ensure the swarm converges:

$$\begin{cases} v_{k+1}^i = \chi \begin{pmatrix} v_k^i + rand_1()\ \ p_k^i - x_k^i \\ + rand_2()\ \ g_k^i - x_k^i \end{pmatrix} \\ x_{k+1}^i = x_k^i + v_{k+1}^i \end{cases} \tag{41}$$

Where the Constriction Factor is presented as follows:

$$\chi = \frac{2}{\left|2-\phi-\sqrt{\phi^2-4\phi}\right|}, \phi = c_1 + c_2 \qquad (42)$$

Pseudo code of algorithm is presented as follows:

```
Initialize swarm
For number of iterations:
    For each particle:
        Evaluate fitness
        If fitness > Pbest:
            Update Pbest to current position
    Update GBest to the best Pbest in swarm
```

For the swarm to update velocity, position, pBest, GBest, or terminate, an evaluation criterion is required. This is commonly referred to as the objective function [14, 15]. The following table presents the criteria that can be used for evaluation:

Tab. 2. Types of objective functions.

Objective function	Fomula
Integral square error (ISE)	$\int_0^\infty \sum_i \left[e_i\ t\right]^2 dt$
Integral absolute error (IAE)	$\int_0^\infty \sum_i \left\|e_i\ t\right\| dt$
Integrated time absolute error (ITAE)	$\int_0^\infty t\sum_i \left\|e_i\ t\right\| dt$
Sum of Square error (SSE)	$\sum_i \mathbf{e}_i^T \mathbf{e}_i$

where:

$e = y - y_d$: Error between the output and the desired reference signal.

4.2 Applying the PSO algorithm to estimate load variation

Overall process of estimating parameters of load variation can be illustrated as shown in Fig. 4 below:

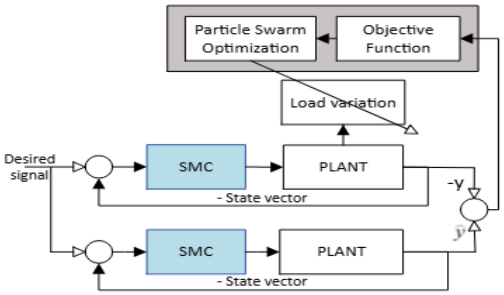

Fig. 3. Structure for load variation estimation.

SMC stabilizes the plant under nominal load, the resulting error under varying load defines the PSO objective, and PSO then estimates the unknown load to minimize that error.

5. Results and Discussion

A stable controller was first designed for sine-wave tracking. To compensate for deviations caused by load changes, the model was augmented with a load parameter and PSO was used to estimate the added or removed load. All simulations were carried out in MATLAB 2020b..

The simulation results with the stabilized controller are presented in Section 3.1:

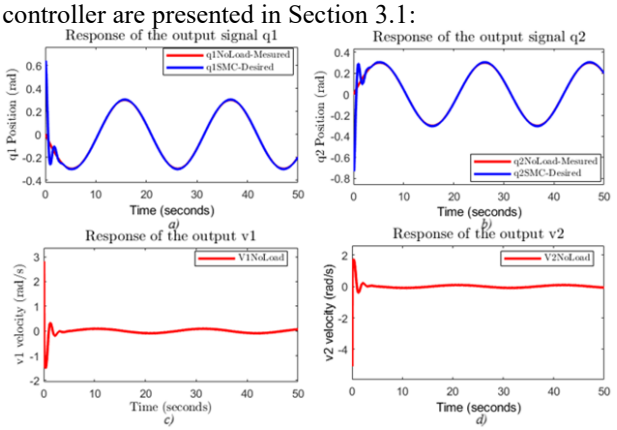

Fig. 4. a) Response of variable q1 without load variation, b) Response of variable q2 without load variation, c) Velocity variation of q1 without load variation, d) Velocity variation of q2 without load variation

After selecting the design constants as follows: K1=11.8090, K2=16.0725, K3=100, K4=4.5443, K5=15.2828, K6=15.8056, Fig. 4 demonstrates that trajectory tracking ability is quite good, with an SSE criterion value of 14.2002 as shown in Tab. 2. Despite a relatively high initial deviation $q_{1_{init}} = \frac{\pi}{6}, q_{2_{init}} = -\frac{\pi}{6}$, controller still has the capability to drive the system to equilibrium and follow the given trajectory as shown in Fig. 4 a) and Fig. 4 b).

Output response when an additional 0.5 kg load is added:

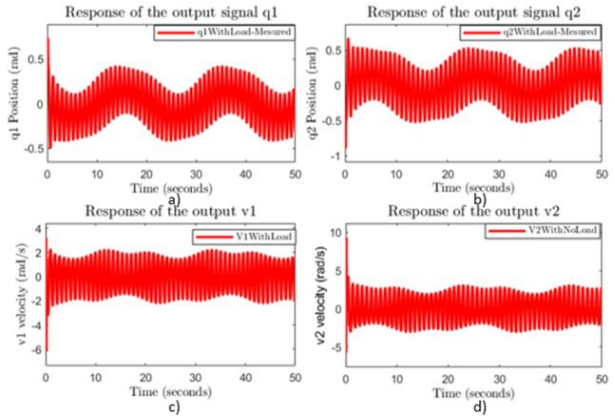

Fig. 5. a) Response of variable q1 when load variation occurs, b) Response of variable q2 when load variation occurs, c) Velocity variation of variable q1 when load variation occurs, d) Velocity variation of variable q when load variation occurs.

Load variations induce sensitivity and reduced performance; estimating them lets the controller adapt and maintain optimal stability.

Parameter estimation results when the load fluctuates:

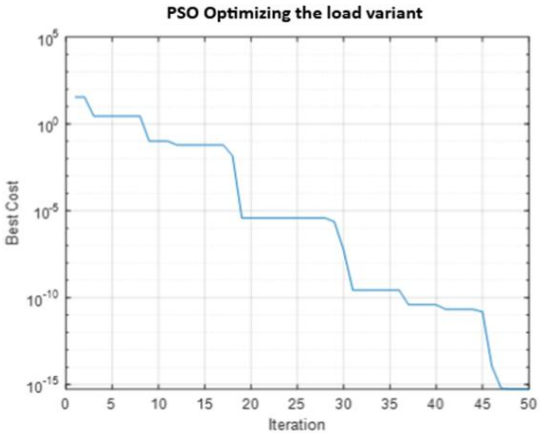

Fig. 6. Optimization graph through each generation.

After each search cycle, the 'best cost' represents the system's adaptability evaluation. The smaller the best cost, the closer the parameters of the predicted model are to the reference model (in this case, the model with load variation). The gradual decrease after each iteration proves that with the PSO algorithm, we can estimate the changing parameters with the highest possible accuracy.

6. Conclusion

In this paper, we propose an algorithm capable of estimating the changing parameters of the system, which is a significant challenge in improving control performance. At the same time, the research team also suggests a hierarchical sliding mode control (HSMC) approach to stabilize and track the trajectory. The simulation results show that the sliding mode controller successfully achieves trajectory tracking for the system, and with the PSO algorithm, the team successfully estimated the fluctuating load of the real system based on the evaluation using the SSE adaptation function we proposed. Future developments, such as directly changing parameters, can be considered for further investigation.

7. Acknowledgements

This paper belongs to project SV2025-80 which is funded by Ho Chi Minh City University of Technology and Education (HCMUTE). We sincerely thank for that support. We also want to give thanks to PhD. Van-Dong-Hai (HCMUTE) for his supervison.

8. Reference:

[1] Luca A.D. et al: "Control of underactuated mechanical systems: application to the planar 2R robot", in Proceedings of 35th IEEE Conference on Decision and Control, 1996, vol. 2, pp. 1455-1460 vol. 2.

[2] Reyhanoglu M. et al: "Dynamics and control of a class of underactuated mechanical systems", IEEE Transactions on Automatic Control, vol. 44, no. 9, pp. 1663-1671, 1999.

[3] Nguyen D.-A.-Q. et al.: "Application of LQG Control for Pendubot System", Journal of Fuzzy Systems and Control, vol. 2, no. 1, pp. 40-44, 04/02 2024.

[4] Jeon J.-H. et al: "Compensation of Sinusoidal Disturbance in Pendubot System using Disturbance Observer", The Transactions of The Korean Institute of Electrical Engineers, vol. 59, 12/01 2010.

[5] Jong-Min Y., Jong-Hwan K.: "Sliding mode control for trajectory tracking of nonholonomic wheeled mobile robots", IEEE Transactions on Robotics and Automation, vol. 15, no. 3, pp. 578-587, 1999.

[6] Zhang Y. et al: "Sensorless sliding-mode control of induction motors", *I*EEE Transactions on Industrial Electronics, vol. 47, no. 6, pp. 1286-1297, 2000.

[7] Jain N.K. et al,: "A Review of Particle Swarm Optimization", Journal of The Institution of Engineers (India): Series B, vol. 99, no. 4, pp. 407-411, 2018.

[8] Shami T.M. et al: "Particle Swarm Optimization: A Comprehensive Survey", IEEE Access, vol. 10, pp. 10031-10061, 2022.

[9] Nguyen D.-A.-Q. et al.: "Adaptive Evaluation of LQR Control using Particle Swarm Optimization for Pendubot", Journal of Fuzzy Systems and Control, vol. 2, no. 2, pp. 58-66, 05/14 2024.

[10] Hoang D.-P. et al.: "A Survey of Experimental LQR for Cart and Pole", Journal of Fuzzy Systems and Control, vol. 2, no. 2, pp. 97-103, 05/22 2024.

[11] Qian D. et al: "Hierarchical sliding mode control for a class of SIMO under-actuated systems", Control and Cybernetics, vol. 37, 01/01 2008.

[12] Eberhart R. and Kennedy J.: "A new optimizer using particle swarm theory", in MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995, pp. 39-43.

[13] Wang D. et al: "Particle swarm optimization algorithm: an overview", Soft Computing, vol. 22, no. 2, pp. 387-408, 2018/01/01 2018.

[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, pp. 325-335, 01/01 2013.

[15] Bingül Z., Karahan O.: "A Fuzzy Logic Controller tuned with PSO for 2 DOF robot trajectory control", Expert Systems with Applications, vol. 38, no. 1, pp. 1017-1031, 2011.