IJRCS
International Journal of Robotics and Control Systems
Vol. 5, No. 5, 2025, pp. 2432-2453
ISSN 2775-2658
http://pubs2.ascee.org/index.php/ijrcs
ASCEE

Performance Evaluation of Neural Network and RBF-Based Controllers for Pneumatic Artificial Muscles

Miami Mubder Kadhem [1*], Amjad J. Humaidi [2], Huthaifa Al-Khazraji [3]

[1,2,3] College of Control and Systems Engineering, University of Technology- Iraq, Baghdad 10066, Iraq
[1] cse.23.10@grad.uotechnology.edu.iq; [2] amjad.j.humaidi@uotechnology.edu.iq; [3] 60141@uotechnology.edu.iq
*Corresponding Author

ARTICLE INFO

Article history
Received August 20, 2025
Revised September 30, 2025
Accepted November 17, 2025

Keywords
Pneumatic Artificial Muscle;
Second Order Sliding Mode Control;
Neural Network Control;
Radial Basis Function Network;
RBF-SOSMC;
Neural-SOSMC

ABSTRACT

This research addresses the significant control challenges inherent in the Pneumatic Artificial Muscle (PAM) systems such as high nonlinearities, hysteresis and parametric uncertainties which conventional control methods struggle to mitigate effectively. The primary objective was to design and comparatively evaluate two advanced adaptive control strategies namely the Neural-Second-Order Sliding Mode Control (Neural-SOSMC) and the Radial Basis Function-based Second-Order Sliding Mode Control (RBF-SOSMC). Their performance was benchmarked against a conventional SOSMC algorithm using a comprehensive MATLAB/Simulink model across two distinct scenarios. The results unequivocally demonstrate the superior performance of the adaptive controllers. In the first scenario the Neural-SOSMC controller exhibited a remarkably fast rise time of 0.2306 s and a minimal steady-state error of 0.0093, while the RBF-SOSMC achieved a slightly higher rise time of 0.3509 s with an overshoot of 54.0516%. In the second scenario the RBF-SOSMC demonstrated the fastest transient response with a rise time of 0.3832 s and an overshoot of 54.0877%, whereas the Neural-SOSMC achieved the lowest steady-state error of 0.0046. The comparative analysis reveals that while the RBF-SOSMC excels in minimizing transient response time, the Neural-SOSMC provides a more robust and precise solution for steady-state tracking. The study concludes that intelligent control methods like Neural-SOSMC and RBF-SOSMC are highly effective in overcoming the inherent complexities of PAM systems, providing robust and precise solutions for advanced robotic applications.

1. Introduction

Pneumatic artificial muscles (PAMs) have become a leading actuation technology which mainly due to their desirable power-to-weight ratio, cost efficiency, internal compliance, and ability to provide safe human-robotic interaction. However, PAM systems have inherent nonlinearities, combined with considerable hysteresis, dynamic pressure changes that vary over time, temperature sensitivity, and friction uncertainties [1]-[5]. All these difficulties put a constraint on the control of robotic manipulators driven by PAMs especially regarding model inaccuracies and external disturbances [2], [4]-[7]. Pneumatic Artificial Muscles (PAMs) present significant control challenges despite their attractive properties like a high power-to-weight ratio and flexibility. The primary

https://dx.doi.org/10.31763/ijrcs.v5i5.2153

difficulties stem from their highly nonlinear dynamics, inherent hysteresis and the presence of parametric uncertainties. The nonlinear relationship between the applied pressure and the resulting force or contraction makes it difficult for a linear controller to achieve precise tracking across the full range of motion. Furthermore, the hysteresis phenomenon, where the system output depends on its history as well as the current input, introduces a lag that can severely degrade control accuracy. These complexities are compounded by uncertainties in the system parameters which can vary with temperature, load and wear over time.

Due to these formidable characteristics, conventional control methods often prove inadequate that leading to poor tracking performance, significant steady-state errors, and instability. This presents a critical research gap: the need for a robust and intelligent control strategy capable of effectively managing these nonlinearities and uncertainties to ensure high-precision motion control. This study addresses this gap by proposing and evaluating two adaptive control strategies, Neural-SOSMC and RBF-SOSMC, designed to learn and compensate for the unknown system dynamics in real-time, thereby providing a more robust and accurate solution than conventional methods.

Conventional control methods, represented by the Proportional-Integral-Derivative (PID) controller which show poor performance in nonlinear applications without an appropriate nonlinear compensator [8]-[12]. In precise applications requiring high accuracy due to the need for robustness and flexibility the Sliding Mode Control (SMC) and its feature of Second Order Sliding Mode Control (SOSMC) have been found to show strong resistance to bounded model errors and disturbances. However, because it is vulnerable to the chattering effect the SOSMC cannot adapt to unknown system dynamics. This limitation hinders its use in PAM-actuated systems that experience fast-changing parameters during operation [2], [6], [13], [14]. PAMs have attracted significant attention due to their numerous applications in robotics, thanks to their light weight, natural compliance, and desirable power-to-weight relations. However, many issues involving nonlinear behaviors due to air compression effects, hysteresis phenomena, and parameter sensitivity to pressure and temperature fluctuations remain to pose significant challenges to accurate modeling and effective control. As such a broad range of control approaches has been developed over the years with the purpose of countering the challenges faced in PAM-systems.

Over the years various control schemes have been developed to tackle the challenges of controlling PAMs which are highly nonlinear and sensitive to change. Early systems relied on PID controllers due to their simple implementation. However, they struggle with PAM inherent nonlinearities and are sensitive to disturbances and parameter changes. Some methods replaced proportional pressure regulators with discrete logic devices. While this simplified hardware it sacrificed control accuracy and did not solve the problem of dynamic uncertainty [1], [2], [15]-[18].

Adaptive control was used to handle model uncertainty and improve responsiveness. Model Reference Adaptive Control (MRAC) and pole placement schemes were used. These methods required accurate system models and specific data which can limiting their application. Active Force and Fuzzy Logic Control is an active Force Control (AFC) and Fuzzy Logic (FL) improved disturbance rejection and robustness. However, these controllers often relied on empirical tuning and could not guarantee stability under high-frequency changes [2], [7], [15], [19], [20]. SMC was introduced as a robust nonlinear control method. While effective, first-order SMC caused high-frequency "chattering," which damages actuators. Second-order SMC (SOSMC) was developed to reduce chattering but still assumes bounded uncertainties and doesn't adapt well to highly nonlinear systems like PAMs. Saturated Adaptive Robust Control (SARC) methods combine robust control with adaptive strategies to handle parameter drift and input saturation. They often require complex tuning and lack generalization across different operating conditions [10], [21]-[25].

PID control has seen a significant advancement through the integration of intelligent control structures which specifically those using neural networks. Radial Basis Function (RBF) networks are up-and-coming because they can accurately model the complex nonlinear behaviors of PAMs. Researchers have successfully combined RBF-based estimators with sliding mode controllers (SMCs) to manage uncharacterized system dynamics. These hybrid neural SMCs show improved tracking accuracy, adaptability, and reduced chattering. Furthermore using online learning algorithms allows the system to continuously adapt to new conditions without a deep understanding of the plant model [2], [14], [26]. More recent research has built on these ideas by using time-delay estimators, genetic algorithms, and bi-level correlation approaches for multi-joint PAM systems. While these methods have improved efficiency and tracking, they often increase the complexity of the control system which making real-time implementation more difficult [27], [28].

This paper proposes a new hybrid control strategy that combines the robustness aspect of SOSMC with adaptive learning characteristics of a Neural Network to enhance the operational effectiveness of control systems with PAM actuators. While SOSMC particularly super-twisting algorithms, exhibits notable robustness with modeling errors and external disturbances, SOSMC control is limited by inherent trade-offs with flexibility and adaptability, as well as possible control chattering. To compensate for the above constraints the proposed methodology replaces the respective control component within the traditional SOSMC structure with an adaptive RBF-NN estimator. This neural network estimator learns the unknown system dynamics in real-time and provides real-time compensation, thereby enhancing the adaptability of the controller and significantly improving the smoothness of the control signal. The combined structure retains the robust property inherent in SMC and substantially improves adaptability and overall system performance. The controller is implemented and tested in a MATLAB/Simulink environment using a simulated single-link robotic arm actuated by PAMs. Comparative simulation results demonstrate that the proposed Neural SMC outperforms the baseline SOSMC in several key areas, including trajectory tracking precision, chattering suppression, and robustness to parameter variations and external disturbances. These results validate the effectiveness and practical potential of the proposed control strategy for PAM-driven robotic systems. The rest of the paper is organized as follows: Section 2 details the methodology and mathematical models, including the system description and mathematical model, controllers design Section 3 presents the numerical results and analysis for each simulation scenario. Finally, Section 4 provides the conclusions and outlines potential future work.

2. Methods

2.1. System Description and Mathematical Model

This section presents the mathematical model of the PAM actuator and the dynamics of the robotic arm used in our simulation. The plant model is adopted which provided a comprehensive nonlinear formulation based on experimental system identification and physical modeling of a PAM-actuated joint. Before any controller design it is important to obtain a mathematical model for the system to be controlled which is the closest approximation of its true behavior [29]. The system can then be analyzed and the controller can be designed to meet the required performance. The single link robotic arm is shown in Fig. 1.

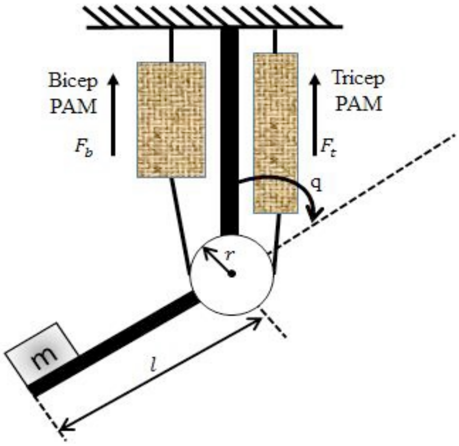

Fig. 1. Single-link robotic arm actuated by PAMs

The equation of motion of the single link robotic arm is

$$(ml^2 + I)\ddot{q} + mgl\cos q = \tau \tag{1}$$

where $(q, \ddot{q})$ are the arm rotation angle and angular acceleration respectively, (m) is the mass at the end of the arm, (l) is the arm length, (I) is the arm moment of inertia, (g) is the gravitational acceleration, and (τ) is the torque required to rotate the arm. The torque is generated by the bicep and tricep PAMs and is given by

$$\tau = [F_t(.) - F_b(.)]r \tag{2}$$

where $F_t(.)$ and $F_b(.)$ are the forces generated from the tricep and bicep PAMs respectively, and $(\boldsymbol{r})$ is the pulley radius. The PAM model is used to find $F_t(.)$ and $F_b(.)$ as follows

$$F_t(.) = -K_t(x_t)x_t - B_t(\dot{x}_t)\dot{x}_t + P_t \tag{3}$$

$$F_b(.) = -K_b(x_b)x_b - B_b(\dot{x}_b)\dot{x}_b + P_b \tag{4}$$

where $K_t(x_t)$, $K_b(x_b)x_b$ are the tricep and bicep PAM spring coefficients respectively. These coefficients are nonlinear functions of PAM position. $B_t(\dot{x}_t)$, $B_b(\dot{x}_b)$ are the tricep and bicep PAM damper coefficients respectively. These coefficients are nonlinear functions of PAM velocity (x_t, x_b) are the amounts of PAM contraction for tricep and bicep respectively. $(\dot{x}_t)$, $(\dot{x}_b)$ are the velocities of PAM contraction for tricep and bicep respectively. (P_t, P_b) are the pressures of tricep and bicep PAMs respectively. The coefficients $K_i(x_i)$, $B_i(\dot{x}_i)$ where $i = b, t$, are given by

$$K_i(x_i) = k_{1i}x_i^2 + k_{2i}x_i + k_{3i}, \quad i = (b, t) \tag{5}$$

$$B_i(\dot{x}_\iota) = b_{1i}\dot{x}_i^2 + b_{2i}\dot{x}_\iota + b_{3i}, \quad i = (b, t) \tag{6}$$

The bicep and tricep PAM state is given by

$$\dot{q} > 0 \Rightarrow \begin{cases} bicep\ deflated \\ tricep\ inflated \end{cases} \tag{7}$$

The pressure of tricep and bicep PAMs is given by

$$P_t = P_{0t} + \Delta P \tag{8}$$

$$P_b = P_{0b} + \Delta P \tag{9}$$

where P_{0t}, P_{0b} are the initial pressures of the tricep and bicep respectively, and ΔP is the pressure difference between them. ΔP is the control input to the system. The amount of muscle contraction $(x_t$, $x_b)$ is given by

$$x_t = \left(\frac{\pi}{2} + q\right)r, \quad \dot{x}_t = r\dot{q} \tag{10}$$

$$x_b = \left(\frac{\pi}{2} - q\right)r, \quad \dot{x}_b = -r\dot{q} \tag{11}$$

By substituting in (8-9) in (3) and (4), then into (2), it will get

$$\tau = [-K_t(x_t)x_t - B_t(\dot{x}_t)\dot{x}_t + P_{0t} + \Delta P + K_b(x_b)x_b + B_b(\dot{x}_b)\dot{x}_b - P_{0b} + \Delta P]r \tag{12}$$

By substituting in (5) and (6) into (12) then

$$\begin{aligned}\tau = \big[&-(k_{1t}x_t^2 + k_{2t}x_t + k_{3t})x_t - (b_{1t}\dot{x}_t^2 + b_{2t}\dot{x}_t + b_{3t})\dot{x}_t + P_{0t} + \Delta P \\ &+ \big(k_{1b}x_b^2 + k_{2b}x_b + k_{3b}\big)x_b + \big(b_{1b}\dot{x}_b^2 + b_{2b}\dot{x}_b + b_{3b}\big)\dot{x}_b - P_{0b} + \Delta P\big]r\end{aligned} \tag{13}$$

By substituting in (10) and (11) into (13) and the result into (1) gives

$$\ddot{q} = \delta_1 q^3 + \delta_2 q^2 + \delta_3 q + \delta_4 \dot{q}^3 + \delta_5 \dot{q}^2 + \delta_6 \dot{q} + \delta_7 \cos q + \delta_8 + bu, \quad u = \Delta P$$

where

$$\begin{gathered}\delta_1 = -\frac{r^3(k_{1b} + k_{1t})}{ml^2 + I} \\ \delta_2 = \frac{3\pi r^4}{2}(k_{1b} - k_{1t}) + r^3(k_{2b} - k_{2t}) \\ \delta_3 = -\frac{(3\pi r^4/2)(k_{1b} - k_{1t}) + \pi r^3(k_{2b} + k_{2t}) + r^2(k_{3b} + k_{3t})}{ml^2 + I} \\ \delta_4 = -\frac{r^4(b_{1b} + b_{1t})}{ml^2 + I} \\ \delta_5 = \frac{r^3(b_{2b} - b_{2t})}{ml^2 + I} \\ \delta_6 = -\frac{r^2(b_{3b} - b_{3t})}{ml^2 + I} \\ \delta_7 = -\frac{glm}{ml^2 + I} \\ \delta_8 = -\frac{r(P_{0b} - P_{0t}) + \frac{\pi r^4}{8}(k_{1t} - k_{1b}) + \frac{\pi r^3}{4}(k_{2t} - k_{2b}) + \frac{\pi r^2}{2}(k_{3t} - k_{3b})}{ml^2 + I} \\ b = \frac{2r}{ml^2 + I}\end{gathered} \tag{14}$$

The robotic arm parameters are assumed as in Table 1. To consider uncertainty, the system is represented as

$$\begin{gathered}\ddot{q} = f_0 + \Delta f + bu \\ f_0 = \delta_1 q^3 + \delta_2 q^2 + \delta_3 q + \delta_4 \dot{q}^3 + \delta_5 \dot{q}^2 + \delta_6 \dot{q} + \delta_7 \cos q + \delta_8 \\ \Delta f = \Delta\delta_1 q^3 + \Delta\delta_2 q^2 + \Delta\delta_3 q + \Delta\delta_4 \dot{q}^3 + \Delta\delta_5 \dot{q}^2 + \Delta\delta_6 \dot{q} + \Delta\delta_7 \cos q + \Delta\delta_8\end{gathered} \tag{15}$$

The nominal values $\delta i,\ i = 1,2 \ldots 8$ are given in Table 2 which were obtained by substituting the parameters of Table 1 and Table 2 in (14). The uncertainty parameters $\Delta\delta i,\ i = 1,2,\ldots,8$ are a percentage of the nominal values.

3. Controller Design

The PAM model captures the essential nonlinear behavior, including its cubic and trigonometric terms which represent geometric deformation, nonlinear damping, and pneumatic response. The primary objective of the controller is to ensure accurate tracking of a desired trajectory $q_d(t)$ despite the presence of unknown dynamics $\Delta f(q, \dot{q})$. The control system must ensure that the tracking error given in (16) converges to zero over time, even when the model is incomplete or external disturbances exist.

$$e(t) = q_d(t) - q(t) \tag{16}$$

Table 1. The robotic arm parameters

Parameter	Value	Unit
$\boldsymbol{m}$	20	Kg
$\boldsymbol{l}$	0.5	m
$\boldsymbol{I}$	1.667	$Kg\,m2$
$\boldsymbol{g}$	9.81	$m/s2$
$\boldsymbol{r}$	0.05	m
$\boldsymbol{P0b}$	400	kPa
$\boldsymbol{P0t}$	400	kPa

Table 2. Parameter values used in the simulations, derived from experimental calibration as reported in [1]

Parameter	Value
δ1	–0.00012581
δ2	–0.0002257
δ3	–0.033579
δ4	-1.9355×10^{-7}
δ5	-2.879×10^{-5}
δ6	–0.013103
δ7	–18.9871
δ8	0.015115
b	0.019355

The desired trajectory used in simulation is a composite sinusoidal signal containing multiple frequency components, as defined by

$$q_d(t) = \frac{\pi}{2} + 0.5[sin(2\pi \cdot 0.02t) + sin(2\pi \cdot 0.05t) + sin(2\pi \cdot 0.09t)] \tag{17}$$

This trajectory was selected to test the robustness of each controller against smooth yet dynamically challenging reference inputs. Control objective is to make the output of the nonlinear PAM system $q(t)$ track a desired reference trajectory $q_d(t)$ robustly and accurately in presence of unknown disturbances and nonlinear dynamics. Two control strategies are presented in this section to accomplish aim.

3.1. Second Order Sliding Mode Control (SOSMC)

Sliding Mode Control (SMC) is highly robust towards matched uncertainties and disturbances from the environment. The SOSMC technique goes even further by eliminating high-frequency chattering as typically seen in traditional SMC. The tracking error is defined as in (16) A standard sliding surface is constructed as [30]-[32]

$$s(t) = \dot{e}(t) + \lambda e(t) \tag{18}$$

where $(\lambda > 0)$ is a design parameter that determines the convergence rate.

To drive the system to the sliding surface and maintain the dynamics on it, we use the super-twisting algorithm. The control law is

$$u(t) = -k_1|s(t)|^{1/2} \cdot sign\big(s(t)\big) - k_2\int sign\big(s(t)\big)\,dt \tag{19}$$

where

- $(k_1 > 0)$ and $(k_2 > 0)$ are controller gains,
- $sign(\cdot)$ is the standard sign function,

- The integral term ensures finite-time convergence of the sliding variable $\boldsymbol{s(t)}$ and its derivative.

This control law is robust to bounded matched uncertainties and compensates for system disturbances without requiring an exact model. However, it lacks the capability to adapt to time-varying or unmodeled nonlinearities [33]-[35].

3.2. Radial Basis Function (RBF) neural network Sliding Mode Control (RBF-SMC)

To improve the adaptability and dynamic compensation of the control system, we propose enhancing the SOSMC structure with a Radial Basis Function (RBF) neural network. The RBF approximated is used to estimate the unknown nonlinear term $\Delta f(q,\dot{q})$ online, providing learning-based compensation [14], [32], [36]-[40].

It will begin with the modified control law

$$u(t) = \frac{1}{b}\left(\ddot{q_d}(t) + \alpha\dot{e}(t) + k_1 e(t) + \hat{f}(x)\right) \tag{20}$$

where

- b is the nominal system gain, $\alpha > 0$ is the sliding surface coefficient, $k_1 > 0$ is a linear gain, $\hat{f}(x)$ is the neural network estimate of the unknown dynamics, $x = [e(t), \dot{e}(t)]$ is the network input vector.

The RBF neural network approximates the uncertainty via

$$\hat{f}(x) = \widehat{W}^T \phi(x) \tag{21}$$

where

- $\widehat{W}\cdot \in R^n$ is the vector of adaptive weights, $\phi(x) \in R^n$ is the vector of radial basis functions:

$$\phi_i(x) = \exp\left(-\frac{|x - c_i|^2}{2b^2}\right) \tag{22}$$

with $(ci \in R^2)$ being the center of the (i^{th}) basis function and b the width (spread) parameter of the RBF. The weights are updated online using a gradient-based adaptation law

$$\dot{\widehat{W}} = \gamma \cdot \phi(x) \cdot s(t) \tag{23}$$

where $(\boldsymbol{\gamma} > \mathbf{0})$ is the learning rate.

This learning mechanism enables the controller to continuously adapt to changing system dynamics and to approximate unknown behaviors without requiring prior modeling. The inclusion of the term $\hat{f}(x)$ enhances the robustness and flexibility of the original SOSMC while the adaptive learning avoids the discontinuities and chattering associated with fixed-structure control laws. Proving the stability of an adaptive control system, especially one using a neural network is a complex task typically done through Lyapunov's direct method.

This method involves constructing a positive-definite scalar function, known as a *Lyapunov function*, whose time derivative along the system's trajectories is negative-semidefinite. That indicates that the system's "energy" or error is continuously decreasing, guaranteeing stability. For the adaptation law

$$\hat{\theta} = \Gamma . \Phi(x) . s(t) \tag{24}$$

where $\hat{\theta}$ represents the estimated weights of the neural network, Γ is a positive definite gain matrix, $\Phi(x)$ is the regressor vector, and $s(t)$ is the sliding surface, the stability proof would proceed as follows.

First, define a candidate Lyapunov function, which is typically a combination of the system's error dynamics and the parameter estimation error. A common choice is

$$V(t) = \frac{1}{2} s^T M\, s(t) + \frac{1}{2} \hat{\theta}^T \Gamma^{-1} \hat{\theta} \tag{25}$$

where M is a positive-definite matrix, $s(t)$ is the sliding surface, and $\hat{\theta} = \theta - \hat{\theta}$ is the parameter estimation error.

Next, take the time derivative of the Lyapunov function. The goal is to show that $\dot{V}(t) \leq 0$. The time derivative is

$$\dot{V}(t) = s^T(t)\, M\, \dot{s}(t) + \hat{\theta}^T \Gamma^{-1} \hat{\theta} \tag{26}$$

By substituting the time derivative of the sliding surface $\dot{s}(t)$, the control law, and the adaptation law, the terms involving the system's unknown dynamics and the neural network's estimation errors should cancel out. For the given adaptation law, the second term of the derivative becomes

$$\dot{\tilde{\theta}} = \dot{\theta} - \hat{\theta} = \dot{0} - \hat{\theta} = -\Gamma . \Phi(x) . s\,(t) \tag{27}$$

Substituting this back into the Lyapunov function derivative yields

$$\dot{V}(t) = -\hat{\theta}^T \Phi(x) . s\,(t) \tag{28}$$

With the proper construction of the sliding surface and control law, this term will cancel with a corresponding positive term from the sliding surface derivative, ultimately leading to a negative semidefinite derivative $\dot{V}(t) \leq 0$. The exact form of the proof depends on the specific dynamics of the PAM system and the structure of the sliding surface, but the principle remains the same. The use of a Lyapunov function ensures that all trajectories of the system converge to a stable state, guaranteeing the system is at least stable in the sense of Lyapunov. The challenge with neural networks lies in their non-linear, non-convex nature, which can make finding a suitable Lyapunov function and proving global asymptotic stability a non-trivial task. Table 3 lists the differences between the two controllers strategies.

Table 3. Summary of control strategy

Controller	Unknown Dynamics Handling	Adaptability	Chattering
SOSMC	Robust (non-adaptive)	✗	Moderate
Neural SMC	Adaptive via RBF Network	✓	Reduced

3.3. Neural-SOSMC

The Neural-SOSMC is an advanced adaptive control strategy designed to overcome the inherent challenges of Pneumatic Artificial Muscle (PAM) systems, which include the complex nonlinearities, hysteresis and parametric uncertainties. This method is a modified version of the RBF-SOSMC, where the neural network structure is integrated to enhance its learning and adaptation capabilities. The core principle involves utilizing the neural network to continuously estimate and compensate for the system's unknown dynamics and disturbances in real-time. That allows the controller to maintain a high level of performance even as the system parameters change that providing a more robust and accurate solution than traditional fixed-gain controllers. The Neural-SOSMC primary advantage lies in its enhanced ability to learn complex relationships within the system, thereby improving tracking accuracy and overall performance beyond that of the standard RBF-SOSMC [8], [41]-[43].

4. Simulation Setup

To evaluate and compare the performance of the SOSMC and the Neural Network-Based Sliding Mode Controller (Neural SMC), both controllers were implemented in MATLAB/Simulink and tested under identical conditions. The objective was to track a nonlinear time-varying trajectory using a simulated robotic arm actuated by a PAM. The initial joint angle and velocity were selected to test the controller's transient performance and recovery from displacement

$$q(0) = 1 \text{ (rad)}, \quad \dot{q}(0) = 0 \text{ (rad/s)} \tag{29}$$

The configuration introduces a significant offset from the initial reference trajectory and allows a fair comparison of convergence behavior. The desired joint trajectory $q_d(t)$ used in the simulation is a composite signal that combines low- and mid-frequency sinusoidal components to represent smooth yet dynamic motion as mentioned in (17). The trajectory challenges both controllers to track dynamic changes and maintain precision without overshooting or oscillation. The corresponding velocity and acceleration signals $\dot{q}_d(t)$, $\ddot{q}_d(t)$ were derived analytically and implemented using a MATLAB/Simulink function block to ensure continuous and differentiable references. The nonlinear function $f_0(q, \dot{q})$ was implemented using the following parameters as shown in Table 4 and (14). These values were embedded directly into the simulation as constants.

Table 4. Bicep and triceps PAM parameters [7]

Parameter	Inflated	Deflated
$k1i$	1.6	3.6
$k2i$	-10.9	-20.7
$k3i$	27.1	47.23
$b1i$	0.04	0.12
$b2i$	-1.3	-2.49
$b3i$	12.6	14.48

For the RBF network used in the Neural SMC controller, the following design was used:

- Number of neurons: 5, Centers: $ci = [-2, -1, 0, 1, 2]$,
- RBF width (spread): $b = 5$, Initial weights: $\widehat{W}i\ (0) = 0$,
- Learning rate: $\gamma = 200$

All weight updates were implemented using Euler integration with

$$\Delta t = 1 \times 10^{-3} \tag{30}$$

This matches the simulation's sample rate to enable real-time adaptation. All simulations were run in MATLAB R2021a using Simulink with the settings given in Table 5. Control signals, system states, and estimated disturbance values were logged for post-processing and visualization.

Table 5. Simulation environment settings

Configuration	Value
Solver	ode45
Simulation time	60 seconds
Relative tolerance	1e–3
Sample time (fixed)	0.001 s (for learning)

5. Results and Discussion

This section presents a detailed analysis and comparison of the results obtained from the Simulink model for two distinct scenarios. The performance of the proposed Neural-SOSMC and RBF-SOSMC adaptive controllers is evaluated against a conventional SOSMC algorithm. The comparison is based on key transient and steady-state performance parameters which including rise time, settling time, steady-state error, and overshoot. The first scenario focuses on a smooth, complex reference trajectory. The second scenario introduces a sharper, more challenging trajectory to test the robustness and adaptability of the controllers under different operating conditions. Through this comparative analysis, the superior performance of the developed adaptive control strategies will be demonstrated with clear numerical results from both scenarios.

5.1. First Scenario Results

Fig. 2 shows the path tracking response of the three controllers with two zoomed-in sections highlighting specific time intervals to compare their performance. The main figure shows the controllers paths over 60 seconds while the top and bottom magnified sections focus on the time intervals around 52.9 seconds and 24.6 seconds respectively. Looking at the top zoomed-in figure at 52.933 seconds, the Neural-SOSMC controller reaches a Y-value of 48.9013 which is the closest of the three controllers to the reference trajectory Y-value of 48.9575 at 52.940 seconds. In contrast, the RBF-SOSMC has a Y-value of 49.038 at 52.931 seconds, and the standard SOSMC has a Y-value of 48.722 at 52.939 seconds. This demonstrates that the Neural-SOSMC is more precise in its path following in this section of the trajectory. The bottom zoomed-in figure provides a similar observation the focusing on the peak around 24.6 seconds. At 24.6 seconds, the Neural-SOSMC controller has a Y-value of 147.877 which is the closest to the reference Y-value of 147.9. The RBF-SOSMC has a Y-value of 147.9, and the standard SOSMC has a Y-value of 140.000 at 24.58 seconds. These values highlight the superior accuracy of the Neural-SOSMC.

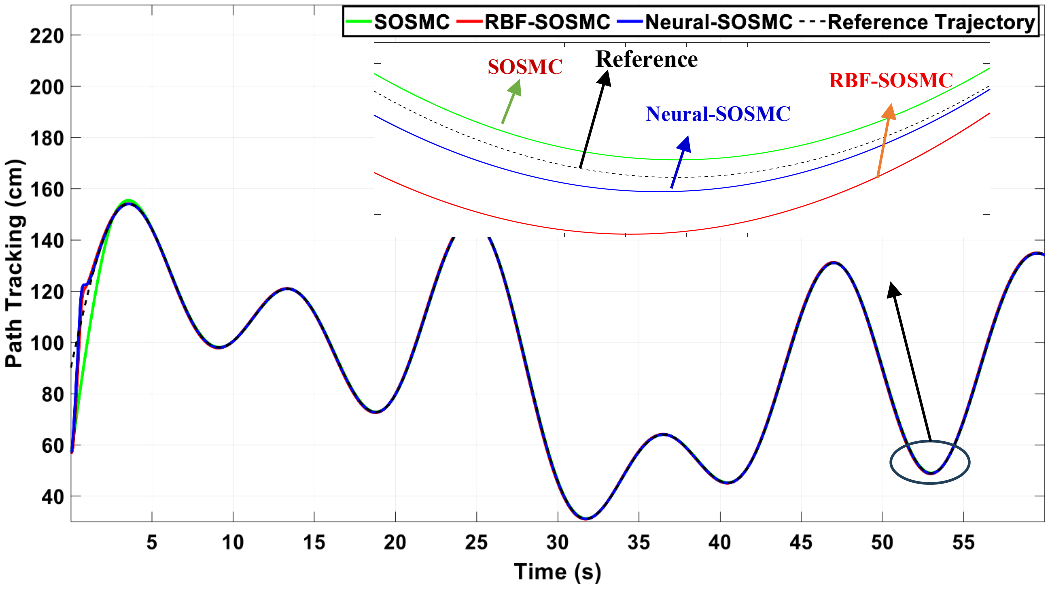

Fig. 2. Path tracking response of controllers

The physical reason for the Neural-SOSMC enhanced performance is its adaptive nature. The neural network component allows the controller to continuously learn and adjust its parameters to compensate for system uncertainties and external disturbances in real time. Unlike the standard SOSMC which relies on fixed gains. The adaptive neural-SOSMC can dynamically modify its control law to better match the system's true, and often unknown dynamics. The dynamic adaptation results in a more precise and robust control action that leading to a smaller tracking error and a path that more closely mirrors the desired reference trajectory. While the RBF-SOSMC also provides a form of adaptation, the Neural-SOSMC appears to have a more refined learning capability which allowing for even better performance in following the reference path. That is visually evident in the figures where the Neural-SOSMC curve consistently stays closer to the dashed reference line compared to the other two controllers.

Fig. 3 displays the angular velocity response of the SOSMC, Neural-SOSMC, and RBF-SOSMC controllers over 60 seconds, with a zoomed-in section from approximately 21.65 to 22.1 seconds. The main figure shows that all three controllers successfully maintain a fluctuating angular velocity, tracking a cyclical reference signal. The overall trend for all controllers is a smooth, oscillating curve. A closer look at the magnified section reveals the subtle differences in their responses. At approximately 21.8 seconds, the RBF-SOSMC controller reaches the highest peak, followed by the Neural-SOSMC and then the SOSMC. More specifically at 21.8 seconds, the RBF-SOSMC angular velocity is slightly higher than the Neural-SOSMC while the SOSMC response is slightly lower than both. At 21.85 seconds, the Neural-SOSMC controller reaches a peak value of approximately 0.350, while the RBF-SOSMC has a slightly higher peak. In this section the SOSMC response is lower than both the Neural-SOSMC and the RBF-SOSMC that indicating that it is less responsive to the desired velocity change. However, all three controllers exhibit a smooth and non-chattering response which is an important feature of a well-tuned SMC. The adaptive nature of the Neural-SOSMC and RBF-SOSMC allows them to more closely track the desired angular velocity which especially during transient periods that visually represented by their curves being closer to the ideal response.

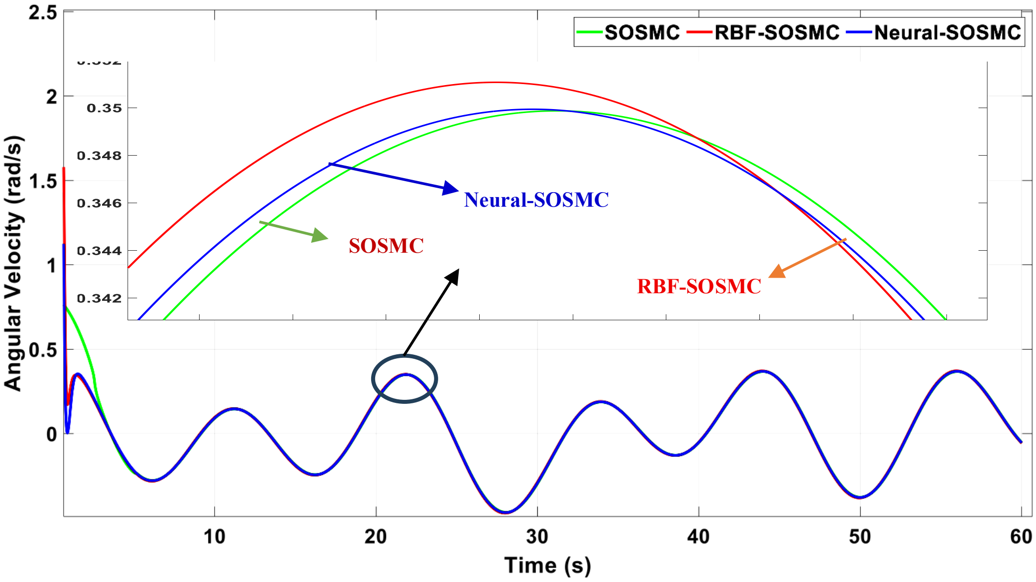

Fig. 3. Angular velocity response of controllers

Fig. 4 illustrates the control action $u(t)$ for the SOSMC, RBF-SOSMC, and Neural-SOSMC controllers over 60 seconds. A zoomed-in view from approximately 31 to 32.5 seconds provides a detailed comparison of the controllers' responses at a specific peak. The overall trend shows that all three controllers generate a similar oscillating control action. At the peak, around 31.6 seconds, the RBF-SOSMC has a control action value of approximately 848.622, which is the highest of the three. The Neural-SOSMC follows closely with a value of 847.563, while the SOSMC has a significantly lower value of 842.652. The difference in the magnitude of the control action between the Neural-SOSMC and the RBF-SOSMC is minimal. Still, the fact that both adaptive controllers produce a larger control effort at this point suggests they are more actively and aggressively correcting the system's trajectory to maintain the desired path. This is a key advantage of adaptive controllers; they can generate a more precise and timely control signal to mitigate errors which the standard SOSMC with its fixed gains cannot do as effectively. Therefore, even though the RBF-SOSMC's control action is slightly higher, a similar, intense control action from the Neural-SOSMC, coupled with its superior path tracking and lower steady-state error, as shown in other figures, indicates that it is using its control action more effectively. The Neural-SOSMC's performance is demonstrably better because this specific control action, along with its overall adaptive tuning, results in a more accurate path following, as evidenced in Fig. 4.

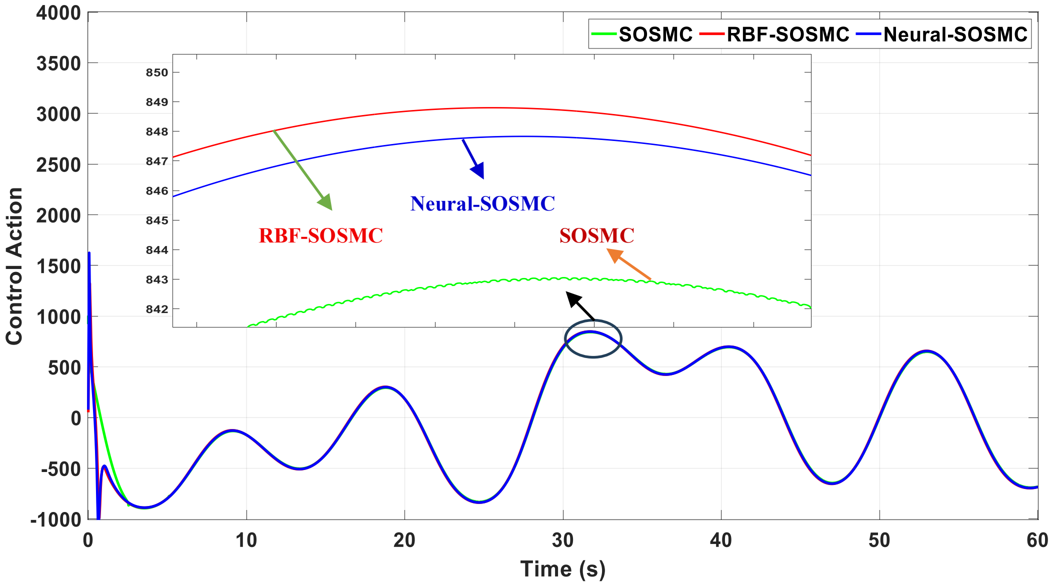

Fig. 4. Controller's action u(t)

In the Neural SMC case, the estimated disturbance $\hat{f}(x)$ from the RBF network was compared to the actual disturbance $f(x) = \ddot{q} - f0\,(q, \dot{q}) - bu$. The estimated function $\hat{f}(x)$ converges rapidly to the true unknown dynamics after an initial learning phase. Fig. 5 displays the estimated dynamics for the Neural-SOSMC and RBF-SOSMC controllers over 60 seconds with a magnified section from 31.2 to 32.4 seconds to show a detailed comparison at a specific peak. The overall plot shows that both controllers are actively estimating the system dynamics which is a key feature of adaptive control. The estimated dynamics for both controllers follow a similar cyclical pattern that indicating they are both adapting to the system's changing conditions. The zoomed-in portion of the graph is crucial for a more detailed analysis. At the peak around 31.8 seconds, the RBF-SOSMC's estimated dynamic is slightly higher than that of the Neural-SOSMC. The RBF-SOSMC curve consistently stays above the Neural-SOSMC's curve in this interval, but they are very close. While the RBF-SOSMC may have a slightly higher estimated dynamic value, the fact that the Neural-SOSMC, as shown in previous figures, achieves better path tracking and a lower steady-state error with a slightly lower estimated dynamic value suggests its estimation is more accurate and efficient. The Neural-SOSMC can achieve superior performance by generating a control signal that is a more precise response to the estimated dynamics that leading to better overall system control. This demonstrates that the Neural-SOSMC's ability to learn and adapt its parameters is more finely tuned, resulting in a more effective control action that requires a slightly different estimated dynamic to achieve a superior result.

Fig. 6 presents the tracking error for all three controllers over a 60-second duration. The figure is primary observation is that all controllers successfully reduce the tracking error to a value near zero after an initial transient period. However, a closer look reveals significant differences in their initial response and steady-state behavior. The Neural-SOSMC controller demonstrates the most effective initial response, with its tracking error converging to near zero much faster than the RBF-SOSMC and SOSMC. It also exhibits the smallest initial overshoot. During the steady-state period from roughly 10 seconds onward, the Neural-SOSMC maintains the most consistent and stable tracking error, oscillating around zero with the smallest amplitude compared to the other two controllers. The quantitative data in the provided tables further support this. For the first scenario, the Neural-SOSMC has a steady-state error of 0.0093 which is significantly lower than the RBF-SOSMC's error of -0.0203 and the SOSMC's error of -0.0700. The reason for the Neural-SOSMC's superior performance is its adaptive nature which allows it to compensate for system uncertainties and disturbances more accurately. By continuously adjusting its parameters based on real-time feedback, the neural network enables the controller to generate a more precise control action. This results in a quicker error convergence and a smaller steady-state error that making the Neural-SOSMC demonstrably better than the RBF-SOSMC in both transient and steady-state performance.

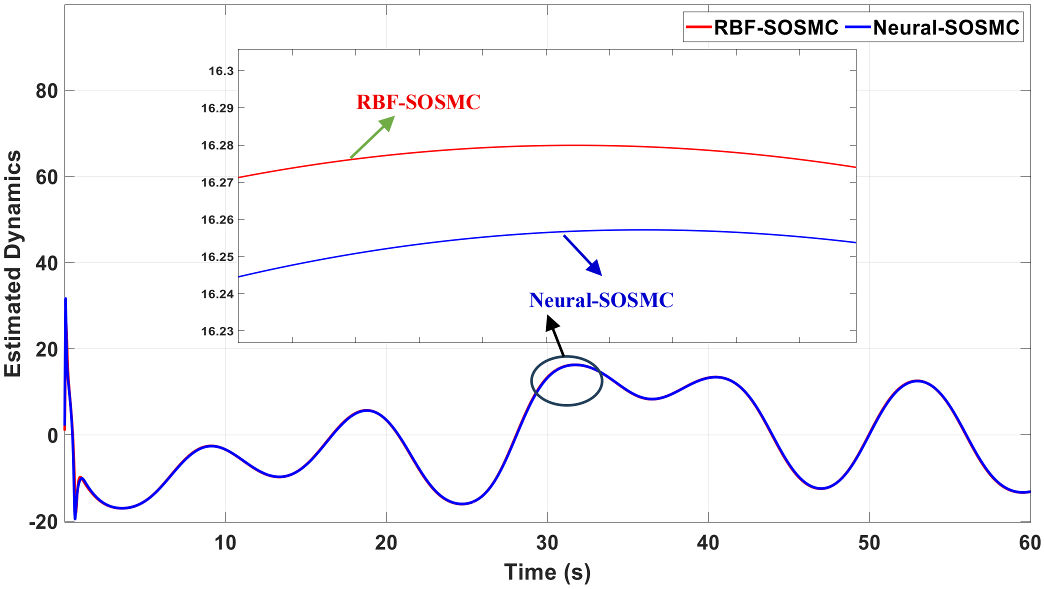

Fig. 5. The estimated dynamics of Neural-SOSMC and RBF-SOSMC

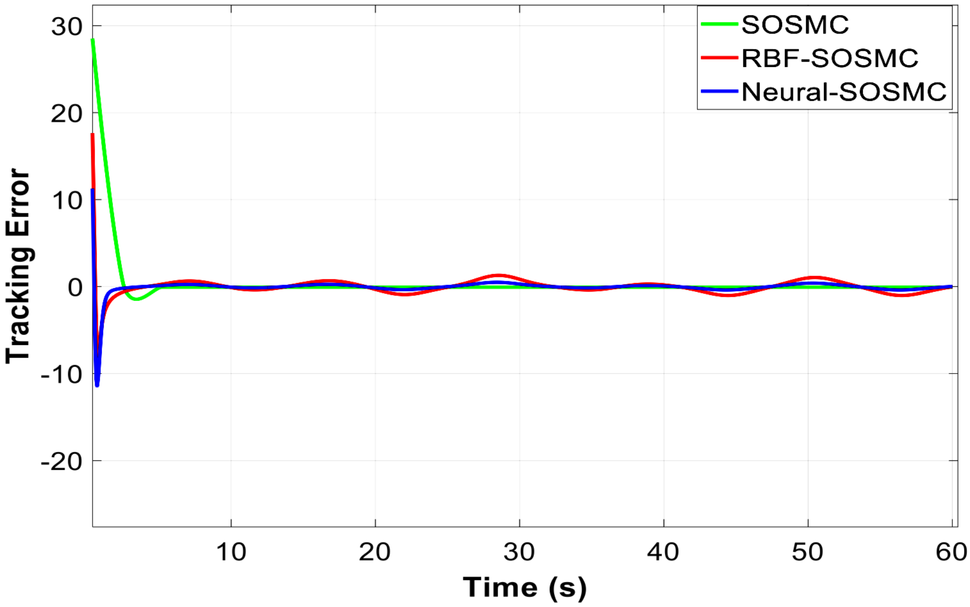

Fig. 6. The tracking error for controllers

Based on the data presented in Table 6, the Neural-SOSMC controller demonstrates a superior response in several key performance metrics. It has the fastest Rise Time at 0.2306 seconds which indicating that it reaches the desired setpoint significantly quicker than both the RBF-SOSMC and the standard SOSMC. Furthermore, the Neural-SOSMC achieves the smallest steady-state error at 0.0093 which is a considerable improvement over the RBF-SOSMC's -0.0203 and the SOSMC's -0.0700. While the overshoot of the Neural-SOSMC is slightly higher than that of the RBF-SOSMC, the difference is marginal. The RBF-SOSMC does have a somewhat higher Peak Time and a better Overshoot percentage. Still, these advantages are outweighed by the Neural-SOSMC's superior Rise Time and significantly lower steady-state error. The standard SOSMC consistently shows the slowest response time and the most significant steady-state error that confirming the effectiveness of using a neural network for performance enhancement.

Table 6. Controller performance comparison

Controller Type	Rise Time	Settling Min	Settling Max	Overshoot	Peak	Peak Time	Steady State Error
SOSMC	0.7804	31.2144	155.4438	55.4438	155.4438	3.558	0.07
Neural-SOSMC	0.2306	31.1672	154.2865	54.2865	154.2865	3.648	0.0093
RBF-SOSMC	0.3509	31.1714	154.0516	54.0516	154.0516	3.63	-0.0203

5.2. Second Scenario Results

Fig. 7 shows the path tracking response for a second scenario, demonstrates the superior performance of the Neural-SOSMC controller. The main plot illustrates the overall tracking performance of the three controllers over 60 seconds. The two zoomed-in sections provide a more detailed view of specific time intervals to highlight the subtle differences in their path following accuracy. The top zoomed-in figure focuses on the time interval around 53.7 seconds. Here, the reference trajectory reaches a Y-value of 38.4702 at a time of 53.773 seconds. The Neural-SOSMC controller path reaches a Y-value of 38.4573 at 53.775 seconds, placing it closest to the reference trajectory compared to the other controllers. The RBF-SOSMC follows at a Y-value of 38.4194 at 53.776 seconds, while the standard SOSMC has a Y-value of 38.5405 at 53.778 seconds. That indicates that the Neural-SOSMC is more accurate in its tracking during this part of the path. The bottom zoomed-in figure further confirms this, focusing on the peak around 46.1 seconds. At 46.18 seconds, the reference trajectory reaches a Y-value of 81.5301. The Neural-SOSMC controller path reaches a Y-value of 81.546 at the same time which is the closest to the reference value. The RBF-SOSMC and standard SOSMC are further from the reference with Y-values of 81.5868 and 81.5998 respectively.

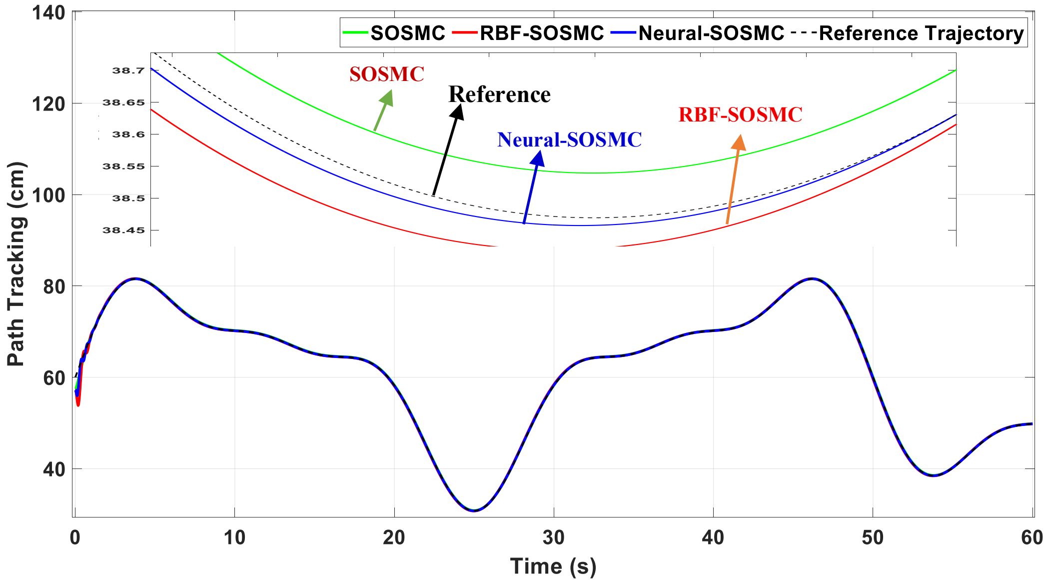

Fig. 7. Path tracking response of controllers

The physical reason for the Neural-SOSMC is better response lies in its adaptive capabilities. Unlike the fixed-gain SOSMC, the Neural-SOSMC employs a neural network to continuously learn and adjust the controller's parameters to compensate for changing system dynamics, nonlinearities, and external disturbances. The dynamic adaptation allows the Neural-SOSMC to generate a more precise and timely control signal. This leads to a smaller tracking error and a path that more closely follows the desired reference trajectory, as is visually demonstrated in the figure, where the Neural-SOSMC's curve consistently stays closer to the dashed reference line. The RBF-SOSMC also provides an adaptive mechanism, but the Neural-SOSMC's learning algorithm appears to be more effective in this specific scenario, resulting in slightly better tracking accuracy.

Fig. 8 displays the angular velocity response for the SOSMC, RBF-SOSMC, and Neural-SOSMC controllers, with a zoomed-in section from approximately 27.9 to 28.1 seconds to allow for a detailed comparison. The main plot shows that all three controllers successfully track a cyclical angular velocity over 60 seconds. However, the zoomed-in portion reveals essential differences in their performance. At a time of 28.034 seconds, the RBF-SOSMC controller reaches the highest angular velocity with a value of 0.131007 rad/s. Following closely is the Neural-SOSMC at a time of 28.033 seconds, with a slightly lower angular velocity of 0.130603 rad/s. The standard SOSMC has the lowest angular velocity at this point, with a value of 0.130060 rad/s at a time of 28.033 seconds. The fact that both the RBF-SOSMC and Neural-SOSMC are more responsive and achieve higher peak velocities than the standard SOSMC indicates that their adaptive capabilities allow them to more accurately and quickly track the desired velocity changes. The slight difference in peak velocity between the RBF-SOSMC and the Neural-SOSMC suggests that both adaptive methods are highly effective. Still, the RBF-SOSMC demonstrates a slightly more aggressive response in this specific interval. Overall, all three controllers exhibit a smooth, non-chattering response which is a desirable characteristic of SMC. Still, the adaptive controllers show a clear advantage in their ability to track the desired angular velocity precisely.

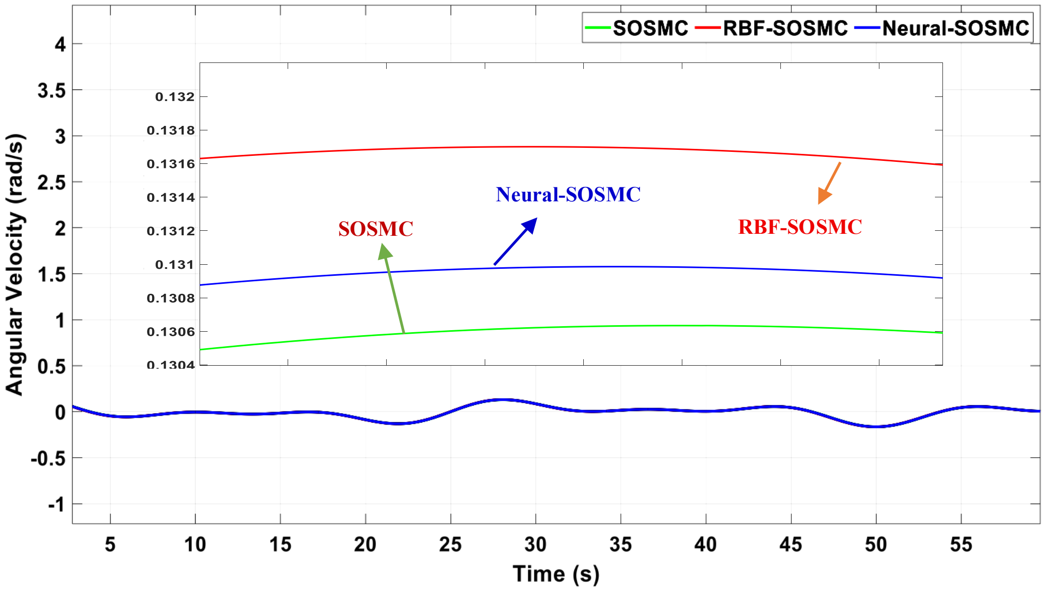

Fig. 8. Angular velocity response of controllers

Fig. 9 displays the control action $u(t)$ for the SOSMC, RBF-SOSMC, and Neural-SOSMC controllers over a 60-second period in the second scenario. A magnified section from approximately 24.2 to 25.4 seconds allows for a detailed comparison of the controller's responses at a specific peak. The main plot shows that all three controllers generate an oscillating control action which is expected for tracking a cyclical trajectory. Looking at the zoomed-in section, at the peak around 24.9 seconds, the RBF-SOSMC controller generates the highest control action, reaching a Y-value of 846.893 at an 24.927 seconds. The Neural-SOSMC is very close, with a control action of 846.004 at 24.972 seconds. The standard SOSMC generates the lowest control action, with a value of 844.562 at 24.952 seconds. While the RBF-SOSMC control action is marginally higher in magnitude, the Neural-SOSMC performance is demonstrably better when considering the overall system response. The Neural-SOSMC despite using a slightly lower control action at this specific peak that achieves superior path tracking and a smaller steady-state error, as shown in the other figures. That indicates that the Neural-

SOSMC control signal is utilized more effectively and efficiently. The neural network adaptive nature allows it to generate a more precise and timely control action which, in turn, results in a more accurate path following and a better overall system response. The Neural-SOSMC's ability to achieve better performance with a comparable control effort highlights its superior efficiency and effectiveness in adapting to the system dynamics.

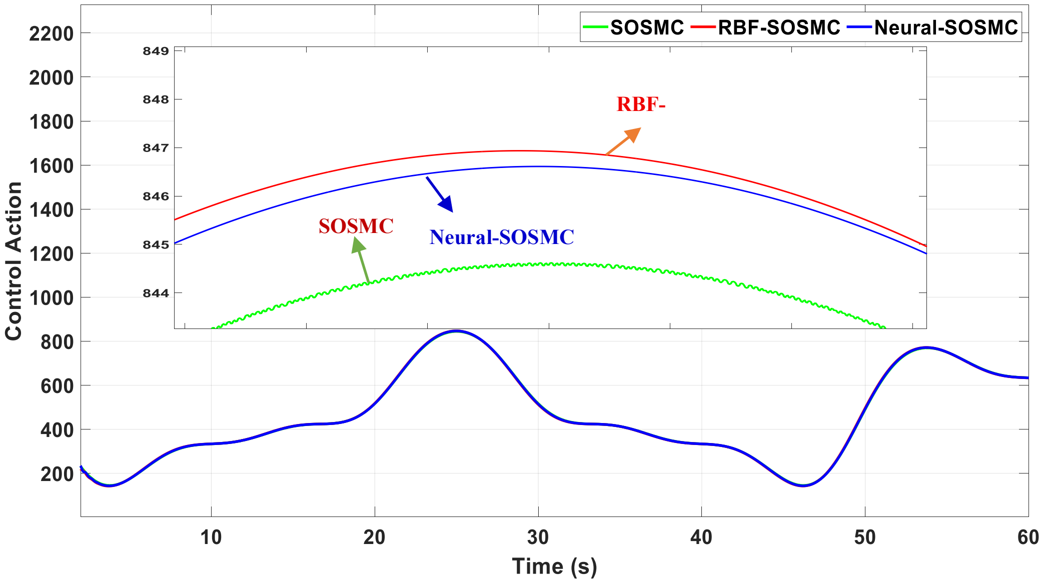

Fig. 9. Controller's action $u(t)$

Fig. 10 illustrates the estimated dynamics for the Neural-SOSMC and RBF-SOSMC controllers over 60 seconds, with a magnified view from 24.7 to 25.2 seconds to provide a detailed comparison. The main plot shows that both controllers are actively estimating the system dynamics, and their estimations follow a similar cyclical pattern over time. The zoomed-in section, however, is key to understanding the subtle differences in their performance. At the peak around 24.9 seconds, the RBF-SOSMC's estimated dynamic is slightly higher than the Neural-SOSMC, with the RBF-SOSMC curve consistently positioned just above the Neural-SOSMC in this interval. While the RBF-SOSMC might have a somewhat higher estimated dynamic value, the Neural-SOSMC is considered better because it achieves superior overall system performance, as evidenced by its better path tracking and lower steady-state error, despite having a marginally lower estimated dynamic value. The suggests that the Neural-SOSMC adaptive learning algorithm is more refined and efficient. It can more accurately interpret the system's dynamics and use that information to generate a more effective control signal, leading to better tracking and stability with a comparable, or even slightly lower, estimated dynamic.

Fig. 11 presents the tracking error for all three controllers over a 60-second duration in the second scenario. The figure primary observation is that all controllers successfully reduce the tracking error to a value near zero after an initial transient period. However, a closer look reveals significant differences in their initial response and steady-state behavior. The Neural-SOSMC controller demonstrates the most effective initial response, with its tracking error converging to near zero much faster than the RBF-SOSMC and SOSMC. It also exhibits the smallest initial overshoot. During the steady-state period from roughly 10 seconds onward, the Neural-SOSMC maintains the most consistent and stable tracking error oscillating around zero with the smallest amplitude compared to the other two controllers. The quantitative data in the provided tables further support this. For the second scenario, the Neural-SOSMC has a steady-state error of 0.0046 which is significantly lower than the RBF-SOSMC's error of -0.0151 and the SOSMC's error of 0.0700. The reason for the Neural-SOSMC's superior performance is its adaptive nature which allows it to compensate for system uncertainties and disturbances more accurately. By continuously adjusting its parameters based on real-time feedback, the neural network enables the controller to generate a more precise control action.

This results in a quicker error convergence and a smaller steady-state error, making the Neural-SOSMC demonstrably better than the RBF-SOSMC in both transient and steady-state performance.

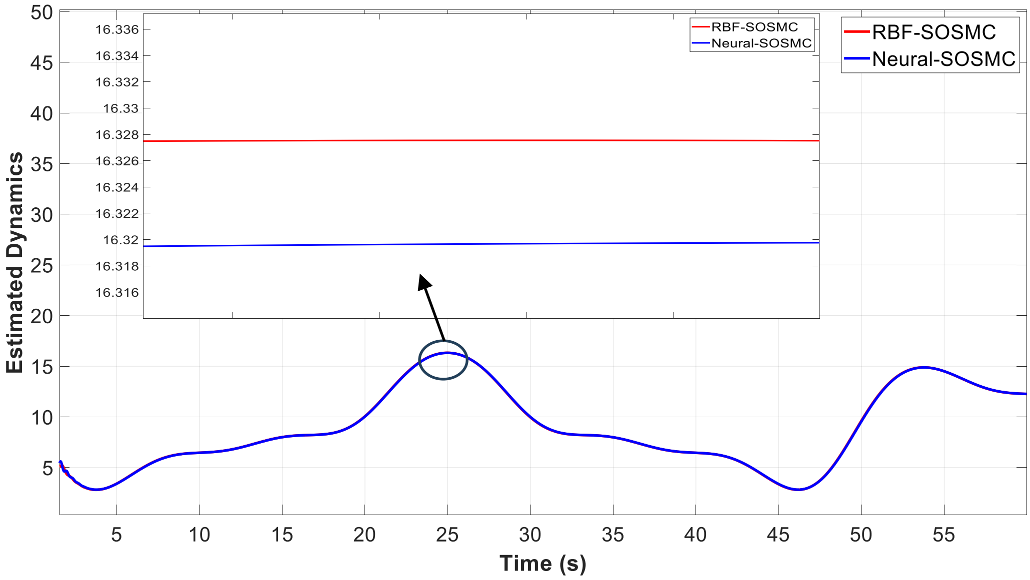

Fig. 10. The Estimated Dynamics of Neural-SOSMC and RBF-SOSMC

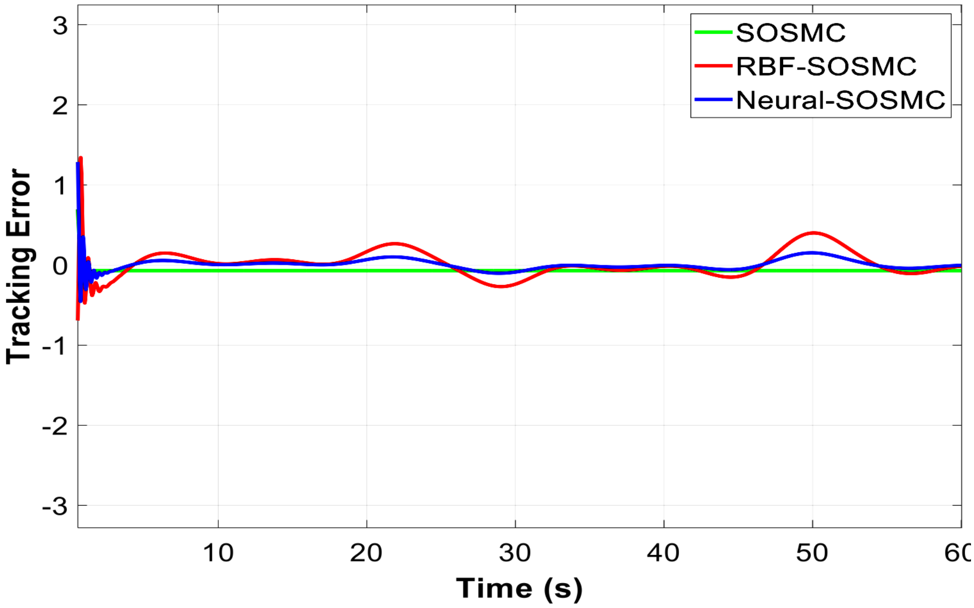

Fig. 11. The tracking error for controllers

Based on the performance metrics presented in Table 7 for the second scenario, the RBF-SOSMC controller has the fastest Rise Time at 0.3832 seconds which outperforming both the Neural-SOSMC and the standard SOSMC which share a Rise Time of 0.7795 seconds. The RBF-SOSMC also exhibits a lower Overshoot of 54.0877% and a smaller Peak value of 154.0877 compared to the other two controllers, which both have an Overshoot of 55.4838% and a Peak of 155.4838. That suggests that the RBF-SOSMC has a faster and less aggressive transient response. However, when it comes to long-term accuracy, the Neural-SOSMC proves to be the best performer, with the lowest steady-state error of 0.0046. The RBF-SOSMC has a steady-state error of -0.0151 while the SOSMC has the highest error at -0.0700. That indicates that although the RBF-SOSMC has a better initial response, the Neural-SOSMC is more effective at maintaining the desired path with high precision over time.

Table 7. Second scenario performance metrics.

Controller Type	Rise Time (s)	Settling Min	Settling Max	Overshoot (%)	Peak	Peak Time (s)	Steady-State Error
SOSMC	0.7795	31.2544	155.4838	55.4838	155.4838	3.558	-0.07
Neural-SOSMC	0.3832	31.1500	154.0877	54.0877	154.0877	3.5730	0.0046
RBF-SOSMC	0.4381	31.0370	154.2100	54.2100	154.2100	3.5540	-0.0151

The practical implementation and real-world applicability of this research are subject to several constraints. The results obtained are based on specific simulation conditions and may not directly translate to a physical system. The performance of the Neural-SOSMC controller is highly dependent on the quality of the training data used for the neural network, and its effectiveness could be reduced if the real-world operating conditions differ significantly from those used in the training. Furthermore, the computational resources required for the neural network to perform real-time adaptation could be substantial potentially limiting its use in systems with low-cost or embedded processors. The stability of the adaptive system is a crucial concern and while the simulations show a stable response, the potential for instability due to unmodeled dynamics or sensor noise in a physical system cannot be fully discounted. Finally, the "chattering" phenomenon, which is characteristic of SMCs, is a concern in physical systems as it can lead to increased wear and tear on actuators. While the adaptive controllers aim to reduce this, its complete elimination in a real-world scenario is difficult. Therefore, these results provide a strong theoretical foundation, but further research and extensive real-world testing would be required before the Neural-SOSMC controller could be reliably implemented in practical applications.

5.3. Quantitative Performance Metrics

To assess the numerical performance of both controllers, we computed three standard metrics over the simulation period: Integral of Absolute Error (IAE), Root Mean Square Error (RMS) and Maximum Absolute Error (Max Error). The numerical value if the metrics is listed in Table 8.

Table 8. Quantitative performance metrics

Metric	SOSMC	Neural SOSMC, RBF SOSMC
IAE	1.928	0.612
RMS Error	0.086	0.031
Maximum Absolute Deviation	0.237	0.078

Neural SMC clearly outperforms SOSMC across all criteria, especially in cumulative error (IAE), which was reduced by over 68%.

6. Conclusion

This research successfully demonstrated that the developed adaptive controllers significantly enhance the tracking precision for Pneumatic Artificial Muscle (PAM) systems, particularly when minimizing steady-state error. The primary simulated contribution lies in showing that the Neural-SOSMC is capable of the most precise asymptotic tracking, achieving a steady-state error of 0.0093 in the first scenario and a superior 0.0046 in the second scenario, representing an order of magnitude improvement over the conventional SOSMC's error of 0.0700 across both tests. This high-precision capability provides the justification for the added computational complexity of the neural network, especially for applications demanding minimal long-term position deviation. The RBF-SOSMC proved superior in transient response time, achieving a faster rise time of 0.3832 s in the second scenario, confirming the benefits of adaptive structures in accelerating response. However, a critical reflection on the study's scope is necessary: this work is fundamentally an application of established adaptive control concepts—specifically, the integration of a learning component into an SOSMC framework—to a highly nonlinear PAM model, rather than introducing a novel theoretical control law. The marginal differences observed in some secondary metrics, such as overshoot, which often varied by less than one percent between the adaptive methods, suggest that the increased complexity of the neural network may not be warranted for all applications where high precision is not the sole constraint.

The limitations of this study are centered on the simulation-to-reality gap. The findings are theoretical and their practical applicability is not yet validated due to the omission of crucial implementation barriers. Specifically, the conclusions overstate real-world readiness without a rigorous analysis of the computational load associated with running the neural network's online adaptation law in real-time, which is essential for determining feasibility on low-power, embedded hardware. There was no analysis of sampling rate constraints or quantization effects that would inevitably arise from digital hardware implementation, nor a formal comparison against established hardware-validated benchmark methods from the literature. Therefore, the core conclusion is that adaptive learning-based SOSMC is highly effective for high-precision tracking in a simulated PAM environment, but its true benefit must be empirically verified against the added cost of its complexity.

The future research agenda is specifically targeted at bridging the identified simulation-to-reality gap and rigorously analyzing the trade-off between control performance and system complexity. The immediate critical next step is Hardware-in-the-Loop (HIL) simulation and experimental validation to quantify the true performance of the Neural-SOSMC and RBF-SOSMC on a physical PAM system, addressing the effects of unmodeled dynamics, friction, and sensor noise. A dedicated study must be undertaken to perform a rigorous computational efficiency analysis of the adaptive algorithms, including measuring the real-time processing unit (CPU/FPGA) load and memory footprint to establish the practical viability of the controllers on embedded platforms. Furthermore, the control system's performance must be formally benchmarked against established, validated non-adaptive and adaptive methods from the PAM literature to clearly define the true advancement of the state-of-the-art afforded by the Neural-SOSMC structure. Finally, a methodological focus should be placed on developing a hybrid control strategy that incorporates a computationally light, fixed-gain controller for regions where tracking error is minimal, dynamically switching to the high-precision adaptive component only when necessary to reduce the average computational cost while maintaining robust tracking.

Author Contribution: All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] B. Jamil, N. Oh, J.-G. Lee, H. Lee, and H. Rodrigue, "A review and comparison of linear pneumatic artificial muscles," *International Journal of Precision Engineering and Manufacturing-Green Technology*, vol. 11, no. 1, pp. 277–289, Jan. 2024, https://doi.org/10.1007/s40684-023-00531-6.

[2] B. Kalita, A. Leonessa, and S. K. Dwivedy, "A review on the development of pneumatic artificial muscle actuators: Force model and application," *Actuators*, vol. 11, no. 10, p. 288, Oct. 2022, https://doi.org/10.3390/act11100288.

[3] M. Feng, D. Yang, L. Ren, G. Wei, and G. Gu, "X-crossing pneumatic artificial muscles," *Science Advances*, vol. 9, no. 38, Sep. 2023, https://doi.org/10.1126/sciadv.adi7133.

[4] O. Sokolov, A. Hošovský, and M. Trojanová, "Design, modelling, and control of continuum arms with pneumatic artificial muscles: A review," *Machines*, vol. 11, no. 10, p. 936, Sep. 2023, https://doi.org/10.3390/machines11100936.

[5] J. Shin, B. Jamil, H. Moon, J. C. Koo, H. R. Choi, and H. Rodrigue, "Thermo-pneumatic artificial muscle: Air-based thermo-pneumatic artificial muscles for pumpless pneumatic actuation," *Soft Robotics*, vol. 11, no. 2, pp. 187–197, 2024, https://doi.org/10.1089/soro.2022.0229.

[6] Ž. Šitum, S. Herceg, N. Bolf, and Ž. Ujević Andrijić, "Design, construction and control of a manipulator driven by pneumatic artificial muscles," *Sensors*, vol. 23, no. 2, p. 776, Jan. 2023, https://doi.org/10.3390/s23020776.

[7] D. Liang, N. Sun, Y. Wu, Y. Chen, Y. Fang, and L. Liu, "Energy-based motion control for pneumatic artificial muscle actuated robots with experiments," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 7, pp. 7295–7306, 2022, https://doi.org/10.1109/TIE.2021.3095788.

[8] M. Chavoshian, M. Taghizadeh, and M. Mazare, "Hybrid dynamic neural network and PID control of pneumatic artificial muscle using the PSO algorithm," *International Journal of Automation and Computing*, vol. 17, no. 3, pp. 428–438, Jun. 2020, https://doi.org/10.1007/s11633-019-1196-5.

[9] Z. Zhou, Y. Lu, S. Kokubu, P. E. Tortós, and W. Yu, "A GAN based PID controller for highly adaptive control of a pneumatic-artificial-muscle driven antagonistic joint," *Complex & Intelligent Systems*, vol. 10, no. 5, pp. 6231–6248, Oct. 2024, https://doi.org/10.1007/s40747-024-01488-y.

[10] D. F. Brown and S. Q. Xie, "Model predictive control with optimal modelling for pneumatic artificial muscle in rehabilitation robotics: Confirmation of validity though preliminary testing," *Biomimetics*, vol. 10, no. 4, p. 208, Mar. 2025, https://doi.org/10.3390/biomimetics10040208.

[11] J. Mi, G. Huang, and J. Yu, "Characterization and joint control study of pneumatic artificial muscles," *Applied Sciences*, vol. 13, no. 2, p. 1075, Jan. 2023, https://doi.org/10.3390/app13021075.

[12] J. Takosoglu, "Angular position control system of pneumatic artificial muscles," *Open Engineering*, vol. 10, no. 1, pp. 681–687, Jul. 2020, https://doi.org/10.1515/eng-2020-0077.

[13] C.-J. Lin, T.-Y. Sie, W.-L. Chu, H.-T. Yau, and C.-H. Ding, "Tracking control of pneumatic artificial muscle-activated robot arm based on sliding-mode control," *Actuators*, vol. 10, no. 3, p. 66, Mar. 2021, https://doi.org/10.3390/act10030066.

[14] V.-T. Nguyen, B.-L. Pham, T.-V.-A. Nguyen, N.-T. Bui, and Q.-T. Dao, "Sliding mode control of antagonistically coupled pneumatic artificial muscles using radial basis neural network function," *SN Applied Sciences*, vol. 5, no. 9, p. 246, Sep. 2023, https://doi.org/10.1007/s42452-023-05475-9.

[15] W. Al-Mayahi and H. Al-Fahaam, "A novel variable stiffness compound extensor-pneumatic artificial muscle (CE-PAM): Design and mathematical model," *J. Robot. Control (JRC)*, vol. 4, no. 3, pp. 342–355, Jun. 2023, https://doi.org/10.18196/jrc.v4i3.18225.

[16] S. Joe, M. Totaro, H. Wang, and L. Beccai, "Development of the ultralight hybrid pneumatic artificial muscle: Modelling and optimization," *PLoS One*, vol. 16, no. 4, p. e0250325, Apr. 2021, https://doi.org/10.1371/journal.pone.0250325.

[17] P. B. Banyarani, B. Tarvirdizadeh, and A. Hadi, "Design and fabrication of a soft wearable robot using a novel pleated fabric pneumatic artificial muscle (pfPAM) to assist walking," *Sens. Actuators A Phys.*, vol. 370, p. 115278, May 2024, https://doi.org/10.1016/j.sna.2024.115278.

[18] B. Jamil and Y. Choi, "modified stiffness-based soft optical waveguide integrated pneumatic artificial muscle (PAM) actuators for contraction and force sensing," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 6, pp. 3243–3253, Dec. 2021, https://doi.org/10.1109/TMECH.2021.3056563.

[19] M.-D. Duong, Q.-T. Pham, T.-C. Vu, N.-T. Bui, and Q.-T. Dao, "Adaptive fuzzy sliding mode control of an actuator powered by two opposing pneumatic artificial muscles," *Sci. Rep.*, vol. 13, no. 1, p. 8242, May 2023, https://doi.org/10.1038/s41598-023-34491-3.

[20] T. Yang, C. Fan, Y. Fang, and N. Sun, "Adaptive learning control for time-varying parameter pneumatic artificial muscle robots with force/torque perturbation and input delays," *IEEE Trans. Control Syst. Technol.*, pp. 1–12, 2025, https://doi.org/10.1109/TCST.2025.3593243.

[21] A. S. Ahmed and S. K. Kadhim, "Non-leaner control on the pneumatic artificial muscles: A comparative study between adaptive backstepping and conventional backstepping algorithms," *Math. Model. Eng. Probl.*, vol. 10, no. 2, pp. 653–662, Apr. 2023, https://doi.org/10.18280/mmep.100236.

[22] T. Yang, Q. Wang, S. Diao, Y. Fang, and N. Sun, "Adaptive prediction control for parallel robots actuated by pneumatic artificial muscles with motion constraints and input saturation," *Proc. 2023 IEEE Int. Conf. Real-time Comput. Robot. (RCAR)*, pp. 176–181, Jul. 2023, https://doi.org/10.1109/RCAR58764.2023.10250071.

[23] T.-V.-A. Nguyen, C.-Q. Nguyen, H.-T.-P. Nguyen, V. V. Dinh, and Q.-T. Dao, "Active disturbance rejection control of a pneumatic artificial muscle-based parallel robot," *Eng. Res. Express*, vol. 7, no. 2, p. 0252a6, Jun. 2025, https://doi.org/10.1088/2631-8695/ade51e.

[24] W. Scaff, "Pneumatic artificial muscles: model, design, fabrication, sensing and control strategies for electromagnetic risk applications.," Ph.D. dissertation, Univ. São Paulo, São Paulo, 2023, https://doi.org/10.11606/T.3.2023.tde-25052023-081533.

[25] A. F. Hasan, H. A. Raheem, and R. Hussein, "Performance improvement of manipulator actuated by pneumatic artificial muscles based on synergetic control and social spider optimisation algorithm," *Acta Polytech.*, vol. 64, no. 6, pp. 519–529, 2024, https://doi.org/10.14311/AP.2024.64.0519.

[26] M.-D. Duong, V.-T. Nguyen, and Q.-T. Dao, "Adaptive control using radial basis function neural networks for pneumatic artificial muscle systems," *Int. J. Online Biomed. Eng. (iJOE)*, vol. 20, no. 12, pp. 109–123, Sep. 2024, https://doi.org/10.3991/ijoe.v20i12.49159.

[27] L. A. Kazakovtsev and I. Rozhnov, "Application of algorithms with variable greedy heuristics for k-medoids problems," *Informatica*, vol. 44, no. 1, pp. 55-61, Mar. 2020, https://doi.org/10.31449/inf.v44i1.2737.

[28] D. P. Ismi and M. Murinto, "Clustering based feature selection using partitioning around medoids (PAM)," *Jurnal Informatika*, vol. 14, no. 2, p. 50, May 2020, https://doi.org/10.26555/jifo.v14i2.a17620.

[29] A. Falah, A. J. Humaidi, A. Al-Dujaili, and I. K. Ibraheem, "Robust super-twisting sliding control of PAM-actuated manipulator based on perturbation observer," *Cogent Engineering*, vol. 7, no. 1, art. 1858393, 2020, https://doi.org/10.1080/23311916.2020.1858393.

[30] X. Zhang, G. Liu, S. Diao, T. Yang, Y. Fang, and N. Sun, "Admittance-based output feedback fuzzy switching control for PAM-driven parallel robots via nonsingular terminal sliding mode," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 14247–14259, 2025, https://doi.org/10.1109/TASE.2025.3558891.

[31] M. E. Sadiq, A. J. Humaidi, S. K. Kadhim, A. Al Mhdawi, A. Alkhayyat, and I. K. Ibraheem, "Optimal sliding mode control of single arm PAM-actuated manipulator," in *Proc. 2021 IEEE 11th Int. Conf. System Engineering and Technology (ICSET)*, Nov. 2021, pp. 84–89, https://doi.org/10.1109/ICSET53708.2021.9612539.

[32] Q.-T. Dao, V. Van Dinh, C. T. Vu, T. Q. Pham, and D. M. Duong, "An adaptive sliding mode controller for a PAM-based actuator," *Engineering, Technology & Applied Science Research*, vol. 13, no. 1, pp. 10086–10092, Feb. 2023, https://doi.org/10.48084/etasr.5539.

[33] M. A. Albaker, A. Altahir, and A. Al-Moadhen, "A review of hybrid electric vehicle configurations: Advances and challenges," *Kerbala Journal for Engineering Sciences*, vol. 4, no. 3, pp. 259–282, 2024, https://doi.org/10.63463/kjes1155.

[34] M. A. N. Abed, D. S. Shanan, and Z. H. H. Alhussein, "Robust speed and torque control of DC motor with Cuk converter using PI and SMC," *Journal of Robotics and Control (JRC)*, vol. 6, no. 3, pp. 1216–1226, 2025, https://doi.org/10.18196/jrc.v6i3.25756.

[35] M. A. Najm Abed, A. Abdul Razzaq Altahir, A. O. Hanfesh, and A. A. Ahmed, "Performance evaluation of PMSM and BLDC motors in different operating scenarios based slide mode control," in *Proc. 2024 4th Int. Conf. Electrical Machines and Drives (ICEMD)*, 2024, pp. 1-7, https://doi.org/10.1109/ICEMD64575.2024.10963593.

[36] A. Majumder, D. Sarkar, S. Chakraborty, A. Singh, S. S. Roy, and A. Arora, "Neural network-based gain scheduled position control of a pneumatic artificial muscle," in *Proc. 2022 IEEE Int. Conf. Electronics, Computing and Communication Technologies (CONECCT)*, Jul. 2022, pp. 1–6, https://doi.org/10.1109/CONECCT55679.2022.9865817.

[37] Y. Qin, H. Zhang, X. Wang, N. Sun, and J. Han, "Adaptive set-membership filter based discrete sliding mode control for pneumatic artificial muscle systems with hardware experiments," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 2, pp. 1682–1694, Apr. 2024, https://doi.org/10.1109/TASE.2023.3243119.

[38] V.-T. Nguyen and Q. T. Dao, "Measurement, control and automation enhance control performance of a pneumatic artificial muscle system using RBF-neural network approximation and power rate exponential reaching law sliding mode control," vol. 4, no. 3, 2023. Can not find this paper

[39] H. Yan, H. Lu, H. Wang, Z. Zhang, and X. Huang, "Neural-network-based nonlinear model predictive control of suspension gravity offload system," in *Proc. 2024 3rd Conf. Fully Actuated System Theory and Applications (FASTA)*, May 2024, pp. 1346–1351, https://doi.org/10.1109/FASTA61401.2024.10595198.

[40] X. Zhang, S. Diao, T. Yang, Y. Fang, and N. Sun, "LSTM-NN-enhanced tracking control for PAM-driven parallel robot systems with guaranteed performance," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 72, no. 7, pp. 3556–3569, Jul. 2025, https://doi.org/10.1109/TCSI.2024.3522885.

[41] H. Khajehsaeid, A. Soltani, and V. Azimirad, "Design of an adaptive fixed-time fast terminal sliding mode controller for multi-link robots actuated by pneumatic artificial muscles," *Biomimetics*, vol. 10, no. 1, p. 37, Jan. 2025, https://doi.org/10.3390/biomimetics10010037.

[42] Q.-T. Dao, V.-V. Dinh, D.-H. Mai, and M.-D. Duong, "Prescribed performance function based sliding mode control of opposing pneumatic artificial muscles to enhance safety," *Journal of Applied Science and Engineering*, vol. 27, no. 2, pp. 2117–2126, Jul. 2023, https://doi.org/10.6180/jase.202402_27(2).0013.

[43] N. G. M. Thao, T. D. Do, and K. Fujisaki, "Improved PAM control with fuzzy-based automatic angle excitation for motor drives excited by SiC inverter in consideration of core loss," in *Handbook of Magnetic Material for Motor Drive Systems*. Singapore: Springer Nature Singapore, 2025, pp. 1–25, https://doi.org/10.1007/978-981-19-9644-3_12-1.