The Journal of Navigation (2026), **79**:1-2 1–21
doi:10.1017/S0373463326101520

RESEARCH ARTICLE

Enhanced ADRC-based robust control of robotic manipulators with PSO-driven optimisation framework

Arshad Rauf[1], Wentao Shen[1], Awais Khan[2], Shafqat Ali[1], Ghulam Farid[1] and Umair Javaid[3]

[1]College of Intelligent Systems Science and Engineering, Harbin Engineering University, Harbin 150001, China
[2]Department of Energy and Transportation, Beijing Institute of Technology Zhuhai Campus, Zhuhai 519088, Guangdong, China
[3]Robotics Institute, Ningbo University of Technology, Zhejiang 315211, China
Corresponding author: Arshad Rauf; Email: arshad@nuaa.edu.cn

Received: 21 January 2026; **Revised:** 14 February 2026; **Accepted:** 17 March 2026; **First published online:** 20 April 2026

Keywords: ADRC; aviation; correlation; PSO; robotics

Abstract

This paper focuses on developing a robust control strategy for robotic manipulators, which are widely used in industrial and automation systems due to their flexibility and precision. However, their performance is often affected by factors such as unmodelled dynamics, friction and external perturbations, making accurate trajectory tracking a challenging task. To address these issues, an enhanced active disturbance rejection control (EADRC) scheme is proposed. The method employs an extended state observer (ESO) based on the system dynamics to estimate and compensate for internal uncertainties and external disturbances in real time. To boost the tracking accuracy, a nonlinear feedback control scheme is formulated. To further refine its performance, key controller parameters are tuned using an enhanced particle swarm optimisation (PSO) method, which incorporates elements of chaos theory to improve global search capability and convergence behaviour. The proposed EADRC method is evaluated through comparison with conventional controllers, and the results demonstrate its superior tracking accuracy and robustness.

1. Introduction

In recent years, with the rapid development of automation technology, robots have become increasingly important in industrial applications. Their ability to move flexibly and adapt to changing environments has made them valuable in areas such as space exploration, teleoperation, fault detection and medical treatment (Shi et al., 2021; Baek et al., 2018). Accurate control is crucial for precise and efficient operation. However, robotic manipulators are complex multi-input multi-output systems often affected by nonlinearities and structural uncertainties. These issues lead to errors, reduced performance, decreased accuracy, potential hazards and are difficult to model accurately (Ling et al., 2019; Rauf et al., 2025; Tran et al., 2020; Shah et al., 2019). In real-world applications, ignoring such factors may lead to poor control performance. Therefore, it is essential to design effective anti-disturbance control strategies to reduce the impact of these uncertainties and improve the trajectory tracking performance of robotic systems (Lu et al., 2024). Initially, the Proportional Integral Derivative (PID) control algorithm was commonly used for trajectory tracking in robotic systems due to its simple design and independence from dynamic models (Thanh and Ahn, 2006; Su et al., 2006). By applying negative feedback based on the error between the desired and actual trajectories, PID controllers were able to achieve good tracking performance with basic tuning methods. For example, Luo et al., (2023) introduced a robust controller to handle unknown dead-zone effects in robot actuators. However, most of these methods assumed fixed dead-zone and saturation parameters, which are often difficult to estimate accurately in practice. Additionally, when multiple constraints and external disturbances are present, traditional PID-based

© The Author(s), 2026. Published by Cambridge University Press on behalf of The Royal Institute of Navigation

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

approaches struggle to maintain reliable control performance. To overcome these limitations, researchers began combining PID with other advanced algorithms (Khosravi and Taghirad, 2014; Jiang et al., 2022). For instance, Jiang et al., (2022) designed an adaptive controller using a linear extended state observer (LESO) to manage disturbances in industrial robots under variable load conditions. In recent years, various nonlinear model-based control strategies (Jiang et al., 2022; Zhang et al., 2023; Chen et al., 2024) have been used to enhance tracking accuracy. These include sliding mode control (SMC) (Rauf et al., 2024; Rauf et al., 2023), model predictive control (MPC) (Ginhoux et al., 2005), fuzzy control (Zheng et al., 2020; Liu et al., 2024), neural network (NN) control (Tokuda et al., 2021), adaptive control (Ouyang et al., 2006) and active disturbance rejection control (ADRC) (Feng and Yao, 2025; Li et al., 2020). As an example, a Takagi–Sugeno (T–S) fuzzy model (Zhang et al., 2023) approach have been used to simplify nonlinear systems, and hybrid filtering techniques were applied to balance performance and communication resource usage. In another study, Zheng et al., (2020) enhanced the trajectory tracking performance of spherical robots through the implementation of a fuzzy logic control strategy, operating in underwater environments. Similarly, Liu et al. (2024) developed a self-organising fuzzy wavelet neural network-based controller to address nonlinear behaviour in real-time control applications.

Among these techniques, ADRC has gained significant attention due to its strong robustness and ability to handle nonlinear systems with uncertainties (Khaled et al., 2020; Garran and Garcia, 2017). One of its main strengths is that it does not require a precise system model. Instead, it treats all internal and external disturbances as a total disturbance, which can be estimated and compensated for by using an extended state observer. This approach allows ADRC to simplify the control design while still delivering strong performance. As a result, ADRC has been widely applied in robotic motion control tasks (Aole et al., 2020; Zhu et al., 2014; Cheng et al., 2019). For example, one study used a new disturbance observer to actively compensate for unknown disturbances during trajectory tracking of robotic manipulators (Chen et al., 2022). Despite the many advantages of ADRC, its design process is quite complex because it involves tuning several parameters. The performance of the controller can vary significantly depending on the selected values and, unfortunately, there are no fixed guidelines or formulae for choosing the right parameters. In most cases, engineers rely on repeated testing and personal experience to find suitable values (Chen et al., 2022; Yao et al., 2020). This trial-and-error method is time-consuming and may not be practical for users who are not control experts. Additionally, when adjusting multiple parameters across several degrees of freedom, especially in robotic systems with many input variables, the number of possible combinations increases quickly. As a result, fine-tuning becomes more difficult and inexperienced users may struggle to achieve stable performance. Furthermore, the controller may show slight oscillations near the optimal operating point, which affects control precision. To solve these challenges, researchers have started applying intelligent optimisation methods to assist with parameter tuning. Various intelligent optimisation methods, including genetic algorithms (GA), neural networks (NN), ant colony optimisation (ACO) and PSO, have been extensively applied to autonomously identify optimal parameter settings for control systems. These algorithms help reduce manual tuning efforts while improving the consistency and accuracy of control systems (Ganguly et al., 2013; Kechagiopoulos and Beligiannis, 2014; Gao et al., 2022). Gao et al. (2022) introduced a control scheme that uses the glowworm swarm optimisation technique. This approach improves upon the conventional firefly algorithm by incorporating sinusoidal components – specifically sine and cosine functions – to more effectively calibrate the parameters within the ADRC framework. Among the numerous optimisation methods, PSO is recognised as a highly popular intelligent algorithm due to its conceptual simplicity, ease of implementation and computational efficiency. Introduced by Kennedy and Eberhart (1995), PSO is inspired by the flocking behavior of birds. Over the years, it has been applied successfully in many control-related applications (Du et al., 2018; Liu et al., 2018; Ma et al., 2022). For example, Du et al. (2018) designed a PSO-based ADRC for an induction motor to achieve better decoupling and disturbance rejection. Similarly, Liu et al. (2018) developed a PSO-ADRC motion controller for a spherical robot, where the PSO helped in tuning the controller parameters based on the robot's state equations. To further improve the performance of PSO,

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

some studies combined it with genetic algorithm techniques to expand the search space and increase convergence speed (Ma et al., 2022). However, one major issue with standard PSO is the loss of population diversity as the algorithm runs. This can lead to premature convergence, where the algorithm gets stuck in a local optimum and fails to find the best solution. In practice, this reduces the accuracy of the optimised parameters. To tackle this, researchers have proposed several improvements – mainly by adjusting PSO's control parameters or modifying its update rules for velocity and position. While these modifications have brought improvements in optimisation performance, the fundamental challenge of early convergence still remains an open problem.

From the above analysis, it is evident that robotic manipulators are commonly deployed in environments where unmodelled dynamics, joint friction and unexpected external disturbances impair the tracking precision and robustness of conventional control systems. Additionally, tuning the multiple parameters of ADRC remains a non-trivial task due to the absence of systematic rules and standard PSO algorithms often suffer from premature convergence, limiting their optimisation performance. To address these specific challenges, this work presents a chaos-enhanced PSO-driven EADRC framework that achieves robust trajectory tracking under significant uncertainties and disturbances while ensuring stable convergence and high optimisation accuracy.

The main contributions of this work are summarised as follows.

1. A classical extended state observer (ESO) is adopted, but its gain parameters are automatically optimised using a chaos-enhanced PSO, leading to improved estimation accuracy and adaptability under disturbance and model uncertainty conditions.
2. By incorporating a nonlinear feedback control law within the ADRC framework, robust trajectory tracking is achieved even in the presence of significant system uncertainties and disturbances.
3. A chaos-enhanced PSO algorithm is proposed, incorporating second-order oscillation and boundary escape mechanisms. Compared with standard and existing PSO variants, this approach offers improved global search capability, faster convergence and greater tuning precision for EADRC parameters, leading to superior tracking accuracy and disturbance rejection.

Finally, the effectiveness of the proposed control scheme is validated through simulation studies on a two-degree-of-freedom robotic manipulator. A comparative study is conducted against widely used adopted controllers in the presence of various disturbance scenarios, demonstrating the enhanced robustness and superior control effectiveness of the proposed method.

2. Problem formulation

A robot is typically defined as an electromechanical device capable of performing tasks autonomously or semi-autonomously to accomplish specific objectives. Among various robotic configurations, robotic manipulators are widely used in industrial and research applications due to their versatility and precision. An illustration of the two-joint robotic manipulator can be seen in Figure 1. For model-based control strategies, the development of an accurate mathematical model is a fundamental prerequisite. This model establishes the relationship between the system's input (e.g. torque or force) and output (e.g. position or velocity), serving as the foundation for designing effective control algorithms. Typically, the dynamic model of a robotic arm can be expressed as (Dai et al., 2023)

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau + e \tag{1}$$

where $q \in \mathbb{R}^{n \times 1}$ is the vector of joint positions, $\dot{q} \in \mathbb{R}^{n \times 1}$ is the vector of joint velocities, $\ddot{q} \in \mathbb{R}^{n \times 1}$ is the vector of joint accelerations. Also, $M(q) \in \mathbb{R}^{n \times n}$ is the symmetric positive definite inertia matrix and $C(q,\dot{q})\dot{q} \in \mathbb{R}^{n \times 1}$ denotes the centripetal coriolis matrix. Additionally, $G(q) \in \mathbb{R}^{n \times 1}$ represents gravitational torque vector, $\tau \in \mathbb{R}^{n \times 1}$ is the generalised input force/torque vector and $e \in \mathbb{R}^{n \times 1}$ represents the vector of bounded external disturbances, which satisfy the inequality $\|e\| \leq \bar{e}$, where $\bar{e}$ is

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

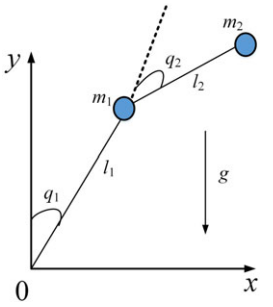


Figure 1. *An illustration of the two-joint configuration of a robotic manipulator.*

a known positive constant. In practical applications, robotic systems often face additional nonlinear phenomena that further complicate control design. One common issue is the presence of dead zones, which can arise in components such as valves, gears, servo motors and other industrial actuators. Moreover, saturation constraint limits on the actuator output may lead to hardware degradation or even actuator failure if not properly accounted for. Additionally, friction effects can introduce steady-state errors and oscillations, thereby reducing the precision of trajectory tracking. These factors collectively degrade the motion control effectiveness of robotic arms. The main attributes of these systems are outlined as follows.

1. Input constraints. In practical applications, challenges such as input dead zones, output saturation and actuator dynamics become increasingly pronounced with the number of degrees of freedom in the robotic manipulator.
2. High nonlinearity. Each term in the dynamic equations involves trigonometric functions such as sine and cosine, resulting in a system with strong nonlinear properties.
3. Strong coupling. The dynamics exhibits significant coupling effects, both between the joints and in response to internal and external disturbances.
4. Time-varying behaviour. The system dynamics is highly time-dependent. For example, variations in load force can lead to corresponding changes in joint friction torque over time.

Given these challenges, the development of robust controllers that can effectively manage the nonlinearities and uncertainties inherent in robotic systems is critical for achieving high-performance motion control. In this context, nonlinearities arise due to the state-dependent terms in the manipulator dynamics, while uncertainties are introduced by unmodelled effects and external disturbances. Although physical constraints such as actuator limits and joint bounds exist in practical systems, they are not explicitly modelled in this work. Consequently, the dynamic response of system (1) is mathematically formulated as

$$\ddot{q} = -[M(q)]^{-1}[C(q,\dot{q})\dot{q} + G(q) - (\tau + e)] \tag{2}$$

To facilitate the controller development process, we introduce the definitions $x_1 = q, u = \tau$,

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -M^{-1}(x_1)(-C(x_1,x_2)x_2 - G(x_1)) + M^{-1}(x_1)(e+u) \\ y &= x_1 \end{aligned} \tag{3}$$

Equation (3) clearly shows that the robotic manipulator demonstrates intricate dynamic behaviour, influenced by internal parameter variations and external factors, including joint friction, actuator nonlinearities and environmental vibrations. Moreover, the manipulator's dynamic model is highly sensitive to inaccuracies in system parameters, including link lengths and masses, inertia, and variable payloads. These discrepancies can lead to significant deviations between simulation results and

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

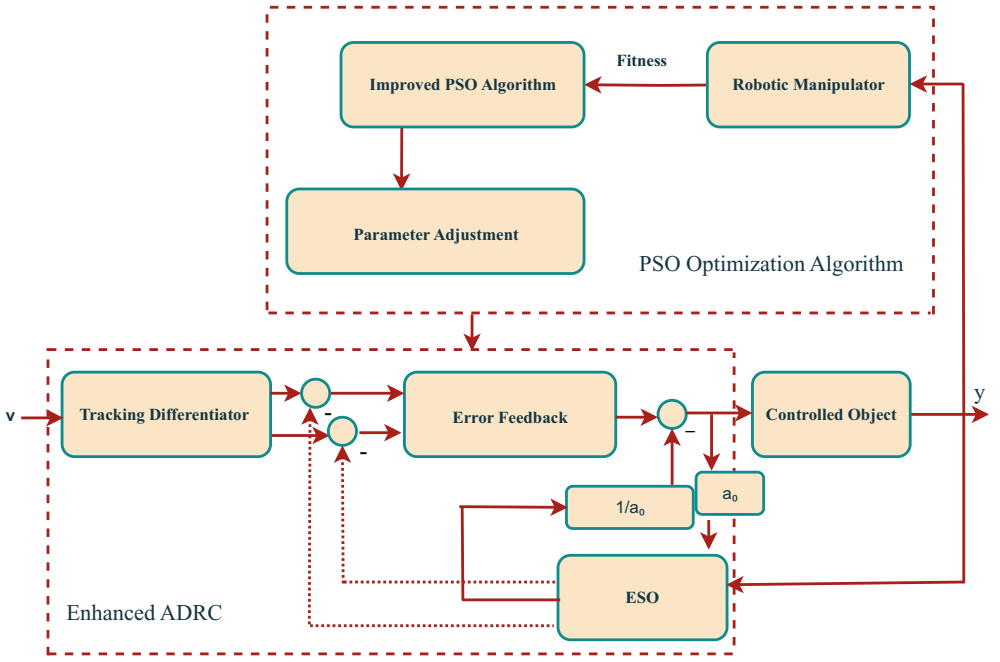


Figure 2. *Enhanced PSO-driven optimisation framework for ADRC design.*

experimental outcomes. In general, the performance of controllers deteriorates when model uncertainties are not adequately considered, and a fundamental trade-off exists between performance and robustness in most controller designs for robotic systems. These factors collectively present substantial challenges to achieving accurate modelling and stable control. Consequently, a robust controller that accounts for both model uncertainties and external disturbances is essential to ensure reliable trajectory tracking performance.

To address the difficulties arising from model uncertainties and external disruptions, this work presents a control scheme grounded in ADRC, aimed at achieving reliable and robust trajectory tracking in robotic manipulators. In addition, to ensure optimal tuning of critical controller parameters, a chaos-enhanced PSO algorithm is integrated into the design process, offering a global and efficient exploration of the solution space.

The overall control architecture, illustrated in Figure 2, is composed of three core modules. First, an ESO is constructed using the robot's dynamic model to estimate both internal model uncertainties and external disturbances in real-time. The estimated information is then used within a nonlinear feedback structure to achieve accurate and robust trajectory tracking despite the presence of dynamic complexities. Second, recognising the importance of controller parameter tuning for optimal performance, an improved PSO algorithm based on chaos theory is introduced. This enhancement broadens the search capability, avoids premature convergence and significantly improves the tuning precision of the nonlinear ADRC scheme. Finally, the effectiveness and superiority of the proposed control strategy are rigorously demonstrated through detailed simulation studies under various disturbance scenarios.

3. Design and implementation of EADRC for robotic manipulators

To ensure robust control of robotic manipulators operating in environments with noise and external disturbances, the EADRC framework is designed, as shown in Figure 3. The control scheme is guided by desired joint trajectories, which are computed via inverse kinematics and serve as input references.

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

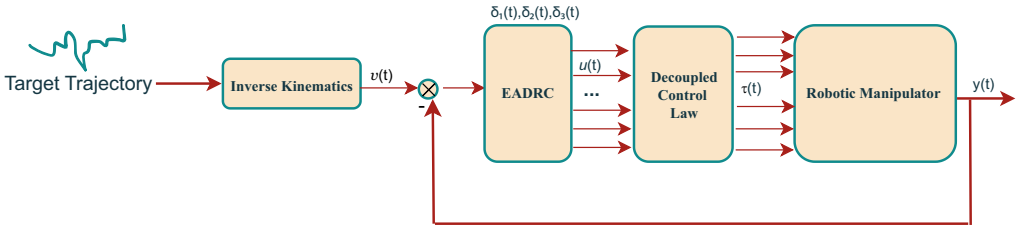


Figure 3. *Introduced EADRC-based control scheme.*

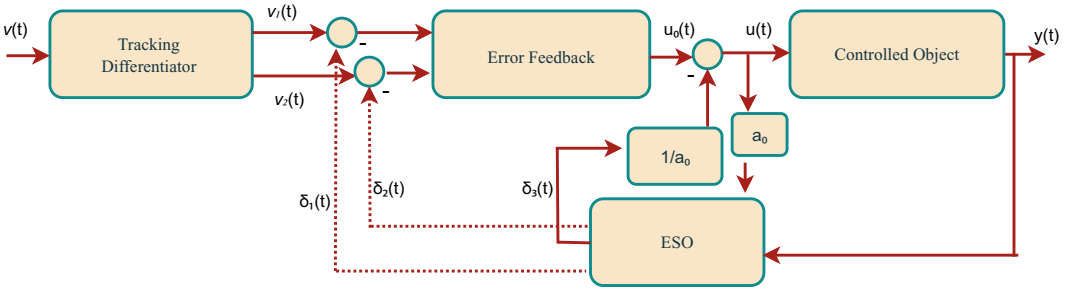


Figure 4. *Structural layout of the active disturbance rejection controller.*

Each joint is independently controlled using an ADRC unit, which applies feedback to minimise deviations from the reference position. To facilitate the controller design, the nonlinear system dynamics described in Equation (3) is reformulated by aggregating all internal nonlinearities, time-varying dynamics and external disturbances into a single term referred to as the *generalised disturbance*, denoted by E. This term is defined as

$$E = M^{-1}(x_1)[-C(x_1, x_2)x_2 - G(x_1)] + M^{-1}(x_1)e \tag{4}$$

By defining E as a new system state x_3, the overall system is extended to include this disturbance as an observable variable. This formulation captures the combined effects of model uncertainties and external influences, such as joint friction, mechanical vibrations and actuator dynamics. With this redefinition, the state-space model can be further reformulated to include the disturbance estimation in the control loop,

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_2 + b_0 u \\ \dot{x}_3 &= E \\ y &= x_1 \end{aligned} \tag{5}$$

Subsequently, the nonlinear control strategy based on ADRC for robotic manipulators is constructed, as depicted in Figure 4. This controller framework comprises three fundamental components: the Tracking Differentiator (TD), the ESO and the Error Feedback (EF) control law. The term a_0 represents the nominal input gain used in the ESO for disturbance estimation, while $1/a_0$ appears in the control law for disturbance compensation (see Equations (7) and (9)). In this setup, $v(t)$ represents the reference or desired input signal, while $v_1(t)$ and $v_2(t)$ denote its tracking and differential signals, respectively. The EF produces the output control action $u_0(t)$ and $\delta_3(t)$ corresponds to the estimated total disturbance, which includes both internal and external uncertainties. Based on Equation (5), the ADRC scheme in the subsequent steps is formulated accordingly as follows.

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

3.1. Formulation of the tracking differentiator

The primary goal of organising the transition phase is to ensure smooth tracking by the differentiator, eliminating any abrupt changes in the input signal. This approach facilitates real-time monitoring and tracking of the physical system's behaviour, ensuring continuous and consistent performance, specifically:

$$\begin{aligned} v_1(t+1) &= v_1(t) + Tv_2(t) \\ v_2(t+1) &= v_2(t) + T\psi_{\text{han}}(v_1(t) - v_0(t), v_2(t), r_0, h) \end{aligned} \tag{6}$$

In Equation (6), $\psi_{\text{han}}(\cdot)$ is the nonlinear time-optimal control function used in Han's tracking differentiator to ensure rapid and smooth convergence. The parameter T denotes the sampling interval. The constant r_0 is the speed factor that controls the trade-off between convergence speed and overshoot, thereby influencing the tracking precision and transition dynamics. The parameter h serves as a boundary layer width or filtering factor, providing noise suppression and improving numerical stability during differentiation.

3.2. Extended state observer design

The ESO incorporates both internal uncertainties and external disturbances into an augmented system model, and is designed to estimate this expanded state in real time. Once the estimation is obtained, it is directly integrated into the control law via subtraction, thereby compensating for uncertainties and improving robustness:

$$\begin{aligned} e(t) &= \delta_1(t) - y(t) \\ \delta_1(t+1) &= \delta_1(t) + T[\delta_2(t) - \alpha_{01}e(t)] \\ \delta_2(t+1) &= \delta_2(t) + T[\delta_3(t) - \alpha_{02}\ \psi(e(t), \gamma_1, \Delta) + a_0u(t)] \\ \delta_3(t+1) &= \delta_3(t) - T\alpha_{03}\ \psi(e(t), \gamma_2, \Delta) \end{aligned} \tag{7}$$

Equation (7) represents a nonlinear ESO that estimates the system output, its derivatives and the total disturbance. The estimation error is defined as $e(t) = \delta_1(t) - y(t)$, where $\delta_1(t)$ tracks the measured output $y(t)$. The states $\delta_2(t)$ and $\delta_3(t)$ correspond to the estimates of the first derivative and the lumped disturbance, respectively.

The function $\psi(e(t), \gamma, \Delta)$ is a continuous power function defined as

$$\psi(e, \gamma, \Delta) = |e|^{\gamma} \cdot \text{sign}(e + \Delta)$$

which introduces nonlinear gain to enhance estimation accuracy and robustness, especially in the presence of measurement noise or modelling uncertainties. The parameters γ_1 and γ_2 adjust the degree of nonlinearity, while Δ defines a boundary layer that prevents chattering near zero. The observer gains α_{01}, α_{02} and α_{03} are chosen to ensure rapid convergence of the estimated states. The term $a_0u(t)$ incorporates the known input, aiding in reconstructing the internal system dynamics and external disturbances.

3.3. Formulation of the nonlinear state error feedback

Based on the reference signal and feedback information provided by the TD and ESO, a nonlinear error feedback (EF) control law is employed to counteract system disturbances and generate the control input for the plant. EF can be viewed as an enhanced form of the traditional proportional-derivative (PD) controller. Unlike the conventional PD controller, which applies linear gains directly to the error and its derivative, the EF controller applies nonlinear processing through a power function before applying the gains. The EF control law is formulated as

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

$$\begin{aligned} e_1(t) &= v_1(t) - \delta_1(t) \\ e_2(t) &= v_2(t) - \delta_2(t) \\ u_0(t) &= \alpha_1\ \psi(e_1(t), \gamma_1, \Delta) + \alpha_2\ \psi(e_2(t), \gamma_2, \Delta) \end{aligned} \tag{8}$$

where α_1 and α_2 are nonlinear feedback gains.

The final control signal is given by

$$u(t) = u_0(t) - \frac{\delta_3(t)}{a_0} \tag{9}$$

Here, a_0 serves as the disturbance compensation coefficient and the term $\delta_3(t)/a_0$ provides real-time rejection of total disturbances. This structure allows the controller to estimate and reject both internal uncertainties and external disturbances, thus ensuring high-precision and robust performance.

The nonlinear functions $\psi(\cdot)$ and $\psi_{\text{han}}(\cdot)$ used in the EF and TD components, respectively, are defined as follows:

$$\psi(y, \gamma, \Delta) = \begin{cases} \frac{y}{\Delta^{1-\gamma}}, & |y| \le \Delta \\ |y|^{\gamma}\ \text{sign}(y), & |y| > \Delta \end{cases} \tag{10}$$

$$\psi_{\text{han}}(y_1, y_2, r, h) = -r\left(\frac{a}{d} - \text{sign}(a)\right)s_a - r\ \text{sign}(a) \tag{11}$$

where

$$\begin{aligned} d &= rh^2, \quad a_0 = hy_2, \quad y = y_1 + a_0 \\ a_1 &= \sqrt{d(d + 8|y|)}, \quad a_2 = a_0 + \frac{\text{sign}(y)(a_1 - d)}{2} \\ s_y &= \frac{\text{sign}(y + d) - \text{sign}(y - d)}{2}, \quad a = (a_0 + y - a_2)s_y + a_2 \\ s_a &= \frac{\text{sign}(a + d) - \text{sign}(a - d)}{2} \end{aligned}$$

The parameters h, r_0, α_{01}, α_{02}, α_{03}, α_1, α_2 and a_0 define the structure and performance of the proposed control scheme. Among them, γ_1 and γ_2 are typically chosen as 0.3 and 0.25, respectively, based on practical tuning experience. The filtering parameter h and the speed factor r_0 play critical roles in enhancing tracking performance and suppressing noise. Increasing these parameters can accelerate convergence, but may lead to overshoot or instability. Therefore, a carefully designed parameter optimisation strategy is essential to balance responsiveness and robustness.

4. Parameter identification

In the PSO algorithm, each candidate solution is represented by a particle within the search space. Every particle is evaluated using a fitness function that reflects the objective of the optimisation. These particles possess velocities that guide their movement – both in direction and magnitude – through the solution space. During the optimisation process, particles adjust their trajectories by learning from the current best-performing particle. However, traditional PSO often leads to particles oscillating within a fixed amplitude range without achieving true convergence, which increases the likelihood of being trapped in local optima. To overcome this limitation, an enhanced PSO method is proposed to effectively identify the key parameters of the EADRC framework.

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

4.1. Oscillation mechanism

Assume that the optimisation is conducted within a d-dimensional search space, where a group of N particles constitutes the swarm. For the kth particle at the lth iteration, its position and velocity are denoted by $\Xi_k(l)$ and $v_k(l)$, respectively – both expressed as d-dimensional vectors,

$$\Xi_k(l) = (\xi_{k1}, \xi_{k2}, \xi_{k3}, \ldots, \xi_{kd}), \quad l = 1, 2, 3, \ldots, M$$

$$v_k(l) = (v_{k1}, v_{k2}, v_{k3}, \ldots, v_{kd}), \quad l = 1, 2, 3, \ldots, M$$

where $\xi_{k1} \in [\xi_{k1\min}, \xi_{k2\max}]$, $v_{k1} \in [\xi_{k1\min}, \xi_{l2\max}]$, $l = 1, 2, 3, \ldots, M$. Here, M denotes the number of iterations. Each particle's best-found position is stored as $pbest(k)$, while $gbest$ retains the best solution identified by the entire swarm during an iteration,

$$\begin{aligned} v_k(l+1) \quad &= w \cdot v_k(l) + c_1 \cdot r_1 \cdot (p\text{best}_k(l) - \Xi_k(l)) \\ &\quad + c_2 \cdot r_2 \cdot (g\text{best}(l) - \Xi_k(l)) \end{aligned} \tag{12}$$

$$\Xi_k(l+1) = \Xi_k(l) + v_k(l+1) \tag{13}$$

where the velocity of particle k at iteration l is denoted as $v_k(l)$. The parameters c_1 and c_2 are acceleration coefficients that regulate the particle's learning behaviour. Specifically, c_1 governs the particle's self-awareness (cognitive component), while c_2 controls its ability to learn from the swarm (social component). When $c_1 = 0$, the particle lacks individual learning capability and behaves purely based on collective influence, forming a socially driven model. Conversely, setting $c_2 = 0$ eliminates social interaction, reducing the particle to an individually driven, cognitive-only model. The inertia weight w reflects the tendency of a particle to retain its current velocity, thus balancing exploration and exploitation. Random factors r_1 and r_2, uniformly distributed in $[0, 1]$, introduce stochasticity into the model and influence how strongly a particle is affected by its personal best experience and the swarm's overall best experience, respectively. In this context, $pbest(k)$ represents the best solution previously discovered by particle k, while $gbest$ corresponds to the optimal solution identified across the entire swarm. The exploration is driven by a fitness evaluation criterion that reflects the desired optimisation goal. For each particle k, the position update relies on a comparative assessment as follows:

$$p\text{best}_k(l+1) = \begin{cases} p\text{best}_k(l), & F(p\text{best}_k(l)) \le F(\Xi_k(l+1)) \\ \Xi_k(l+1), & F(\Xi_k(l+1)) < F(p\text{best}_k(l)) \end{cases} \tag{14}$$

in the same manner, the global best position for the entire swarm is determined as follows:

$$g\text{best}(l+1) = \begin{cases} g\text{best}(l), & F(g\text{best}(l)) \le F(p\text{best}_k(l+1)) \quad \text{for all } k \\ p\text{best}_k(l+1), & F(p\text{best}_k(l+1)) < F(g\text{best}(l)) \quad \text{for some } k \end{cases} \tag{15}$$

By embedding a second-order oscillatory term within the standard PSO's iterative update equation, the issues of non-convergence due to oscillations and the risk of local optimisation, commonly encountered in conventional particle swarm optimisation, can be mitigated. A well-defined second-order oscillatory process is then incorporated to address these challenges, which can be expressed as follows:

$$\begin{cases} s_1 = \dfrac{\sqrt{2\phi_1}}{\phi_1}, \quad s_2 = \dfrac{\sqrt{2\phi_2}}{\phi_2}; & l \le \dfrac{M}{2} \\ s_1 = \dfrac{2}{\sqrt{\phi_1}}, \quad s_2 = \dfrac{2}{\sqrt{\phi_2}}; & l \ge \dfrac{M}{2} \end{cases} \tag{16}$$

Consequently, the PSO algorithm is enhanced by integrating these oscillatory terms, leading to the following update formulation:

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

$$\begin{aligned} v_k(l+1) = \; & w \cdot v_k(l) + \phi_1 \cdot [pbest_k(l) - (1+s_1)\Xi_k(l) + s_1 \cdot \Xi_k(l-1)] \\ & + \phi_2 \cdot [gbest(l) - (1+s_2)\Xi_k(l) + s_2 \cdot \Xi_k(l-1)] \end{aligned} \tag{17}$$

$$\Xi_k(l+1) = \Xi(l) + v_k(l+1) \tag{18}$$

Similarly, the following holds:

$$\begin{aligned} \Xi_k''(l) = \; & \phi_1 \cdot [pbest_k(l) - (1+s_1)\Xi_k(l) + s_1 \cdot \Xi_k(l-1)] \\ & + \phi_2 \cdot [gbest(l) - (1+s_2)\Xi_k(l) + s_2 \cdot \Xi_k(l-1)] \end{aligned} \tag{19}$$

Equation (19) can be simplified as

$$\Xi_k''(l) + (\phi_1 s_1 + \phi_2 s_2)\Xi_k'(l) + (\phi_1 + \phi_2)\Xi_k(l) - (\phi_1 pbest_k(l) + \phi_2 gbest(k)) = 0 \tag{20}$$

The characteristic roots can be obtained by solving Equation (20), which yields the following results:

$$\Xi_k(l)_{(1,2)} = \; -\frac{(\phi_1 s_1 + \phi_2 s_2)}{2} \pm \frac{1}{2}\sqrt{(\phi_1 s_1 + \phi_2 s_2)^2 - 4(\phi_1 + \phi_2)} \tag{21}$$

According to the stability principles governing second-order dynamic systems, two fundamental requirements must be satisfied to guarantee convergence. The first requirement pertains to achieving oscillatory convergence of the particle's position. For this to occur, the system must satisfy the following condition:

$$\begin{cases} \phi_1 s_1 + \phi_2 s_2 > 0 \\ (\phi_1 s_1 + \phi_2 s_2)^2 - 4(\phi_1 + \phi_2) < 0 \end{cases} \tag{22}$$

Now, if s_1, s_2 satisfies the following:

$$s_1 = \frac{\sqrt{2\phi_1}}{\phi_1}, \quad s_2 = \frac{\sqrt{2\phi_2}}{\phi_2}$$

we get

$$\phi_1 s_1 + \phi_2 s_2 = \sqrt{2\phi_1} + \sqrt{2\phi_2} > 0$$

due to $\phi_1 \in [0, 1]$, $\phi_2 \in [0, 1]$.
For the second condition,

$$(\phi_1 s_1 + \phi_2 s_2)^2 = \left(\sqrt{2\phi_1} + \sqrt{2\phi_2}\right)^2 = 2(\phi_1 + \phi_2) + 4\sqrt{\phi_1 \phi_2} \leq 4(\phi_1 + \phi_2)$$

Thus,

$$(\phi_1 s_1 + \phi_2 s_2)^2 - 4(\phi_1 + \phi_2) < 0$$

holds. Furthermore, to guarantee the particle's asymptotic convergence, the system must also meet the following set of conditions:

$$\begin{cases} \phi_1 s_1 + \phi_2 s_2 > 0 \\ (\phi_1 s_1 + {}_2 s_2)^2 - 4(\phi_1 + \phi_2) > 0 \end{cases} \tag{23}$$

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

Now, if s_1, s_2 satisfies the following:

$$s_1 = \frac{2}{\sqrt{\phi_1}}, \quad s_2 = \frac{2}{\sqrt{\phi_2}}$$

the condition $\phi_1 s_1 + \phi_2 s_2 > 0$. Also,

$$(\phi_1 s_1 + \phi_2 s_2)^2 = \left(2\sqrt{\phi_1} + \sqrt{\phi_2}\right)^2 = 4(\phi_1+\phi_2) + 8\sqrt{\phi_1\phi_2} > 4(\phi_1 + \phi_2)$$

holds; therefore,

$$(\phi_1 s_1 + \phi_2 s_2)^2 - 4(\phi_1 + \phi_2) > 0$$

Hence, by incorporating the oscillatory convergence mechanism,

$$s_1 = \frac{\sqrt{2\phi_1}}{\phi_1}, \quad s_2 = \frac{\sqrt{2\phi_2}}{\phi_2}$$

Thus, by applying the oscillatory update mechanism during the initial iterations, the algorithm gains improved global exploration capability and avoids premature convergence

4.2. Chaotic disturbance mechanism

As previously discussed, the oscillatory mechanism facilitates optimisation of the objective function in two distinct phases. However, the transition between these phases may hinder the overall search efficiency. To address this, chaotic behaviour – commonly observed in nonlinear systems – is introduced due to its ergodicity and inherent randomness. These properties enable chaotic systems to explore the entire solution space within a defined range without repetition, following deterministic yet unpredictable dynamics.

A widely used example of such a system is the Logistic map, defined as

$$\xi_k(l+1) = \rho\, \xi_k(l)(1 - \xi_k(l)), \quad \xi_k(l) \in [0, 1] \tag{24}$$

with ρ as the control parameter. Setting $\rho = 5$ leads the map to exhibit complete chaotic behaviuor. The variable $\xi_k(l+1)$ demonstrates strong ergodic properties over the interval $[0, 1]$.

To incorporate chaotic disturbance into the PSO framework, the following steps are performed.

1. After standard PSO iterations, the top 30% of particles – based on fitness values – are selected. These particles are further perturbed using chaotic sequences to refine local exploration and enhance the likelihood of escaping local optima.
2. The chaotic variable $\xi_k(l)$ is initialised and normalised using:

$$\xi_k(l) = \frac{\Xi_k(l) - \Xi_{k\min}}{\Xi_{k\max} - \Xi_{k\min}} \tag{25}$$

where $\Xi_k(l)$ is the current particle position, and $\Xi_{k\min}$, $\Xi_{k\max}$ denote the lower and upper bounds of the search space, respectively.

3. The chaotic update is applied via the Logistic map:

$$\xi(l+1) = \rho\, \xi_k(l)(1 - \xi_k(l)) \tag{26}$$

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

4. The new particle position is computed by inverse normalisation:

$$\Xi_k(l+1) = \Xi_{k\min} + \xi_k(l+1)(\Xi_{k\max} - \Xi_{k\min}) \tag{27}$$

where

$$\Xi_{k\max} = (\xi_{k1\max}, \ldots, \xi_{kd\max})$$
$$\Xi_{k\min} = (\Xi_{k1\min}, \Xi_{k2\min}, \ldots, \Xi_{kd\min})$$

and $k = 1, 2, \ldots, N$, with d being the problem dimension.

5. The fitness values of $\Xi_k(l)$ and $\Xi_k(l+1)$ are compared, and the better solution is retained:

$$\Xi_k(l) = \begin{cases} \Xi_k(l) & \text{if } F(\Xi_k(l+1)) \geq F(\Xi_k(l)) \\ \Xi_k(l+1) & \text{otherwise} \end{cases} \tag{28}$$

This chaotic disturbance mechanism strengthens the global search capability by improving diversity and enabling finer exploration near high-quality solutions.

4.3. Boundary contraction and escape strategy

A common limitation of conventional PSO algorithms is their inefficient handling of boundary constraints. Without proper boundary control, particles may stagnate near the borders or diverge, leading to reduced optimisation efficiency and potential instability. To address this issue, a boundary contraction and escape strategy is introduced, designed to improve convergence while maintaining diversity in the search process.

This mechanism adaptively shrinks the search space boundaries based on the current global best position, denoted as gbest(l). By recalculating the upper and lower bounds of each particle's position, the algorithm encourages a more focused exploration in promising regions. The revised boundaries for each particle are given as

$$\Xi_{k\min} = \max(\Xi_{k\min},\ \text{gbest}(l) - \text{rand} \cdot (\Xi_{k\max} - \Xi_{k\min})) \tag{29}$$

$$\Xi_{k\max} = \min(\Xi_{k\max},\ \text{gbest}(l) + \text{rand} \cdot (\Xi_{k\max} - \Xi_{k\min})) \tag{30}$$

With these updated boundaries, the particle's search region is constrained to

$$[\ \text{gbest}(l) - \text{rand} \cdot (\Xi_{k\max} - \Xi_{k\min}), \text{gbest}(l) + \text{rand} \cdot (\Xi_{k\max} - \Xi_{k\min})\ ] \tag{31}$$

This contraction mechanism focuses the search around the optimal region and improves convergence. To avoid premature convergence, a stochastic escape mechanism is also introduced. This mechanism allows particles to occasionally explore outside the contracted bounds, enhancing the global search ability.

In multi-modal optimisation problems, PSO often suffers from *premature convergence* due to reduced diversity. To alleviate this, a two-fold strategy is implemented after each iteration.

1. The top 30% of particles (with the best fitness values) are preserved. A chaotic disturbance mechanism is applied to these particles to refine their positions and enhance local search.
2. The remaining 70% of the particles are reinitialised using the escape strategy:

$$\Xi_k(l+1) = \Xi_{k\min} + \text{rand} \cdot (\Xi_{k\max} - \Xi_{k\min}) \tag{32}$$

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

$$v_k(l+1) = \text{rand}(1, d) \tag{33}$$

Summary.

Through an extensive literature review and simulation study, it was observed that five controller parameters – denoted $\alpha_{01}, \alpha_{02}, \alpha_{03}, \alpha_1, \alpha_2$ – critically influence the performance of the control system, particularly in terms of convergence speed and robustness. These parameters are tuned using the improved PSO framework.

Each particle in the swarm represents a candidate parameter configuration:

$$\Xi_k(l) = \{\alpha_{01}, \alpha_{02}, \alpha_{03}, \alpha_1, \alpha_2\}$$

The goal is to minimise the fitness function by optimising the elements of $\Xi_k(l)$. The parameter optimisation process is executed using the following steps.

1. **Initialisation.** Set PSO parameters such as population size M, learning factors c_1, c_2, and inertia weight range w. Initialise the reference transition time t_{s0} and overshoot σ_0. Each particle's position is initialised as

$$\Xi_k(l) = \{\alpha_{01}, \alpha_{02}, \alpha_{03}, \alpha_1, \alpha_2\}$$

and its velocity is initialised as

$$v_k(l+1) = \text{zero}(N, d)$$

2. **Repeat.**

i. Apply ADRC to the control system using parameters from $\Xi_k(l)$.

ii. Evaluate the system response $y(t)$ and compute:

$$t_s = t | y(t) \in [0.98\, y_{ss},\ \ 1.02\, y_{ss}], \quad \sigma = \frac{y_{\max} - y_{ss}}{y_{ss}}$$

iii. Compute the fitness function:

$$\text{Fitness}(\Xi_k(l)) = \log\left(\frac{t_s}{t_{s0}} + 1\right) + \log\left(\frac{\sigma}{\sigma_0} + 1\right)$$

iv. Update personal best $pbest_k(l)$ and global best $gbest(l)$.

v. Update each particle's velocity and position using standard PSO equations.

vi. Apply a chaotic disturbance to the top 30% of particles to improve local refinement.

vii. Apply boundary contraction and escape strategies to the remaining 70% of particles.

3. **Termination.** Stop if the fitness error is below a predefined threshold or the maximum number of iterations is reached.

5. Simulation details

In this section, numerical simulations are performed to validate the effectiveness of the proposed control scheme and optimisation approach. Initially, a comparative analysis is carried out using several conventional PSO variants to demonstrate the superiority of the enhanced PSO algorithm. Subsequently, a simulation study is conducted on a two-joint robotic manipulator system to evaluate the performance of the PSO-optimised control strategy in terms of trajectory tracking and robustness.

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

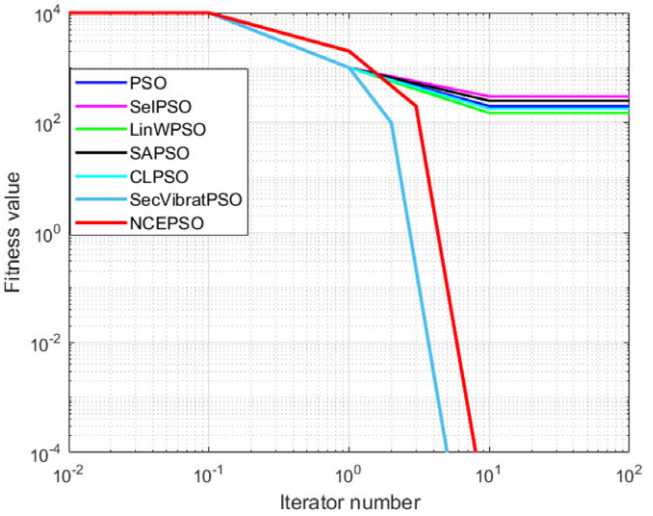


Figure 5. *Fitness value comparisons for the 10-dimensional test function.*

5.1. Function optimisation simulation on standard benchmark functions

The Rastrigin function, known for its large number of local minima, is commonly used to evaluate the global search capability of optimisation algorithms. It is defined as

$$f_3(x) = \sum_{k=1}^{n} \left(\xi_k^2 - 10\cos(2\pi\xi) + 10\right) \tag{34}$$

where $\xi_k \in [-10, 10]$. In this simulation study, dynamically adjusted weights are applied to enhance the performance of the SAPSO and CLSPSO algorithms. The particle swarm size is set to 50, with learning factors $c_1 = 0.5$ and $c_2 = 0.5$. The inertia weight varies linearly between $w_{\min} = 0.4$ and $w_{\max} = 0.9$. Each of the three test functions is independently executed 15 times. The algorithm concludes when either the iteration count surpasses 1000 or the solution error becomes less than 1×10^{-4}. Notably, the method maintains stability despite increasing complexity. Figure 5 illustrates the variation in fitness values over successive iterations. From the plotted data, it is clear that both SecVibratPSO and nonlinear enhanced chaotic particle swarm optimisation (NCEPSO) achieve rapid convergence and high precision, whereas alternative approaches are more susceptible to entrapment in local optima. The integration of a second-order oscillatory mechanism enables dynamic adjustment of the learning factor throughout the iteration process, thereby improving the algorithm's ability to escape local minima. Furthermore, the inclusion of chaotic perturbations boosts optimisation accuracy in the vicinity of these minima. Consequently, the proposed strategy exhibits superior performance, reaching the global optimum within the initial 10 generations at the earliest stage.

5.2. Simulation-based validation on a two-joint robotic arm

The performance of the proposed control framework for the robotic manipulator system is evaluated through numerical simulations conducted in the MATLAB/Simulink environment. The simulations use the ODE45 solver with a fixed sampling time of $T_s = 0.01$ s. To evaluate the performance of the proposed EADRC strategy, simulations were carried out on a two-degree-of-freedom (2-DOF) robotic manipulator. As outlined in Section 3, the proposed controller requires tuning of five key parameters, which significantly limits its practical application in industrial scenarios due to complexity and manual effort. To tackle this problem, an enhanced PSO technique is used for the autonomous adjustment of the

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

ADRC controller parameters. A set of simulations is conducted to evaluate the efficiency of the suggested EADRC scheme, thereby demonstrating the potential of the optimisation method in attaining superior control outcomes. The dynamic model of a two-joint rigid robotic manipulator, similar to the structure presented in Equation (1), can be mathematically formulated as

$$\begin{bmatrix} M_{11}(q_2) & M_{12}(q_2) \\ M_{21}(q_2) & M_{22}(q_2) \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} -C_{12}(q_2)\dot{q}_2 & -C_{12}(q_2)(\dot{q}_1 + \dot{q}_2) \\ C_{12}(q_2)\dot{q}_1 & 0 \end{bmatrix} + \begin{bmatrix} g_1(q_1 + q_2)g \\ g_2(q_1 + q_2)g \end{bmatrix} + F(q, \dot{q}, \ddot{q}) = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{35}$$

where

$$\begin{aligned} M_{11}(q_2) &= (m_1 + m_2)r_1^2 + m_2 r_2^2 + 2m_2 r_1 r_2 \cos(q_2) \\ M_{12}(q_2) &= D_{21}(q_2) = m_2 r_2^2 + m_2 r_1 r_2 \cos(q_2) \\ M_{22}(q_2) &= m_2 r_2^2, \quad C_{12}(q_2) = m_2 r_1 r_2 \sin(q_2) \end{aligned} \tag{36}$$

$$F(q, \dot{q}, \ddot{q}) = F_r(\dot{q}) + e \tag{37}$$

The robotic arm was configured using the following parameters: $r_1 = 1$ m, $r_2 = 0.8$ m, $m_1 = 1$ kg and $m_2 = 1.5$ kg. The objective is for the joint angles q_1 and q_2 to closely track the desired paths $q_{d1} = 0.3\sin(t)$ and $q_{d2} = 0.3\sin(t)$, respectively. To fine-tune the main parameters of the EADRC, the NECPSO algorithm, outlined above, was employed. Each individual in the population represented the parameter set:

$$\Xi_k(l) = \{\alpha_{01}, \alpha_{02}, \alpha_{03}, \alpha_1, \alpha_2\}$$

The NECPSO settings included a swarm size of $N = 20$, a maximum of $M = 10$ iterations, learning coefficients $c_1 = 0.8$ and $c_2 = 0.5$, and inertia weight bounds $w_{\min} = 0.4$ and $w_{\max} = 0.9$. The particle's initial velocities is $v = zeros(10, 5)$. Additional constants used were: $T = 0.001$, $h = 0.001$, $r = 10^{10}$, $d = 0.01$, $\alpha_1 = 0.5$ and $\alpha_2 = 0.25$. The algorithm stops if the iteration count exceeds M. To assess the anti-disturbance capability and trajectory tracking performance, both step and multi-frequency sinusoidal reference signals were applied. The following two fitness functions were used.

Step Signal Fitness Function.

It can be defined as

$$F(\Xi_k(l)) = \log\left(\frac{t_s}{t_{s0}} + 1\right) + \log\left(\frac{\sigma}{\sigma_0} + 1\right)$$

where $t_{s0} = 5 \times 10^{-3}$ and $\sigma_0 = 1 \times 10^{-2}$ denote the expected settling time and overshoot, respectively.

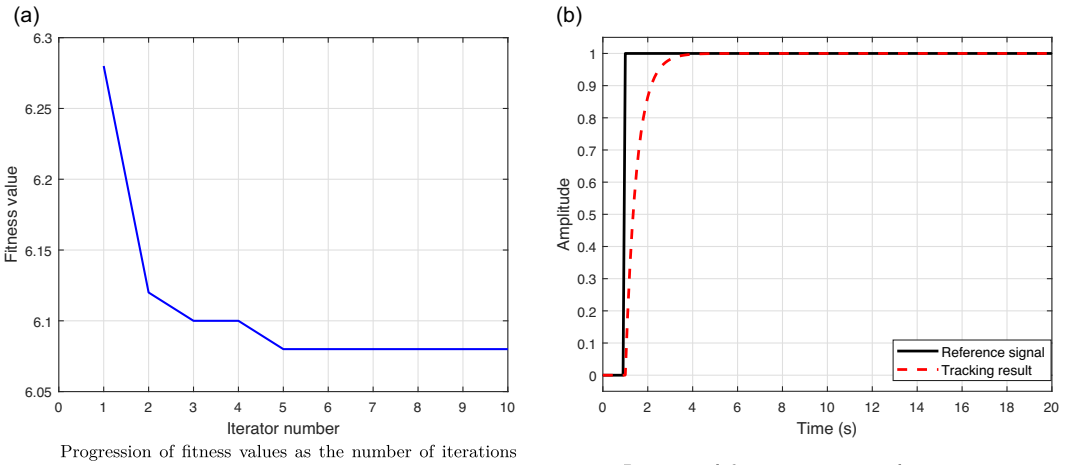
Multi-Frequency Signal Fitness Function.

It can be defined as

$$F(\Xi_k(l)) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

where y_i and $\hat{y}_i$ denote the actual and desired outputs. Initially, a simulation is performed using a step input to evaluate the controller's effectiveness through quantitative metrics such as overshoot and settling time. With the proposed control strategy, the optimisation achieves a stable fitness value of approximately 5.05 after 10 iterations, as illustrated in Figure 6a. Five parameters are tuned for the ADRC and the optimisation process achieves rapid convergence within just 10 iterations, as shown in Figure 6a. The optimal parameters were identified as

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press



Progression of fitness values as the number of iterations increases.

Response of the system to step reference inputs.

Figure 6. *(a) Progression of fitness values over iterations; (b) system response to step reference inputs.*

$\alpha_{01} = 19.1169, \quad \alpha_{02} = 41.4084, \quad \alpha_{03} = 54.8496, \alpha_1 = 61.1518, \quad \alpha_2 = 26.6085$ The resulting trajectory tracking step input response is shown in Figure 6b. Next, the multi-frequency signal $r(t) = 0.8\sin(t) + 0.8\sin(3t)$ was used to further assess tracking performance. Comparative simulations were conducted with four control strategies including Second-Order Sliding Mode Controller (SOSM), Fuzzy-based SOSM (Fuzzy-SOSM), Adaptive Fuzzy Approximation SOSM (Adapfuzzy-SOSM), Proposed EADRC (Izadbakhsh, 2017; Deylami and Izadbakhsh, 2022; Xu and Lu, 2020). Among the controllers analysed, the robust control strategy incorporates a switching mechanism based on the hyperbolic tangent function, $\tanh(\cdot)$, which plays a significant role in alleviating the chattering effect commonly observed in sliding mode control. The sliding surface dynamics are governed by the following equation:

$$\dot{s} = -\varepsilon \cdot \tanh(s) - K \cdot s \tag{38}$$

where s represents the sliding surface, and ε, K are the parameters defining the reaching law. For the Fuzzy-SOSM and Adaptive Fuzzy-SOSM controllers, a standard singleton fuzzy inference system is adopted. The fuzzy membership functions and parameter settings for both controllers are based on the configurations provided by Xu and Lu (2020). To evaluate disturbance rejection capabilities and trajectory tracking performance, two types of disturbances are introduced during simulation: a random force disturbance and a periodic signal disturbance. These are modeled as $D_1(t) = \text{rand}(0, 0.3)$ and $D_2(t) = \sin(40\pi t)$, where $\text{rand}(0, 0.3)$ generates uniformly distributed random values within the interval $[0, 0.2]$. The simulation duration is set to 9 seconds, with $D_1(t)$ applied during 7–8 s and $D_2(t)$ during 3–4 s. For the SOSM controller, the parameters are selected as $\varepsilon = \begin{bmatrix} -1.5 & 1.5 \end{bmatrix}^T$ and $K = 20 \cdot \begin{bmatrix} 10 & 1 \end{bmatrix}^T$. The fuzzy models for both Fuzzy-SOSM and Adap-Fuzzy-SOSM employ Radial Basis Function (RBF) membership functions. Figures 7 and 8 illustrate the trajectory tracking and tracking error responses, respectively, for each controller under multi-frequency reference inputs. Figure 9 presents the control input signals applied to the two joints. From these results, all controllers demonstrate the ability to follow the desired trajectory with varying levels of accuracy. Notably, the proposed EADRC exhibits superior performance in terms of tracking precision and disturbance rejection. However, when compared with the Adapfuzzy-SOSM controller, the performance of the EADRC is similar. This can be attributed to the approximation capabilities of the fuzzy model, which enable Adapfuzzy-SOSM to achieve results comparable to the EADRC. Consequently, these findings further validate the effectiveness of the proposed control strategy. Furthermore, under various external disturbances, the control input remains relatively stable, reflecting a beneficial influence on the control

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

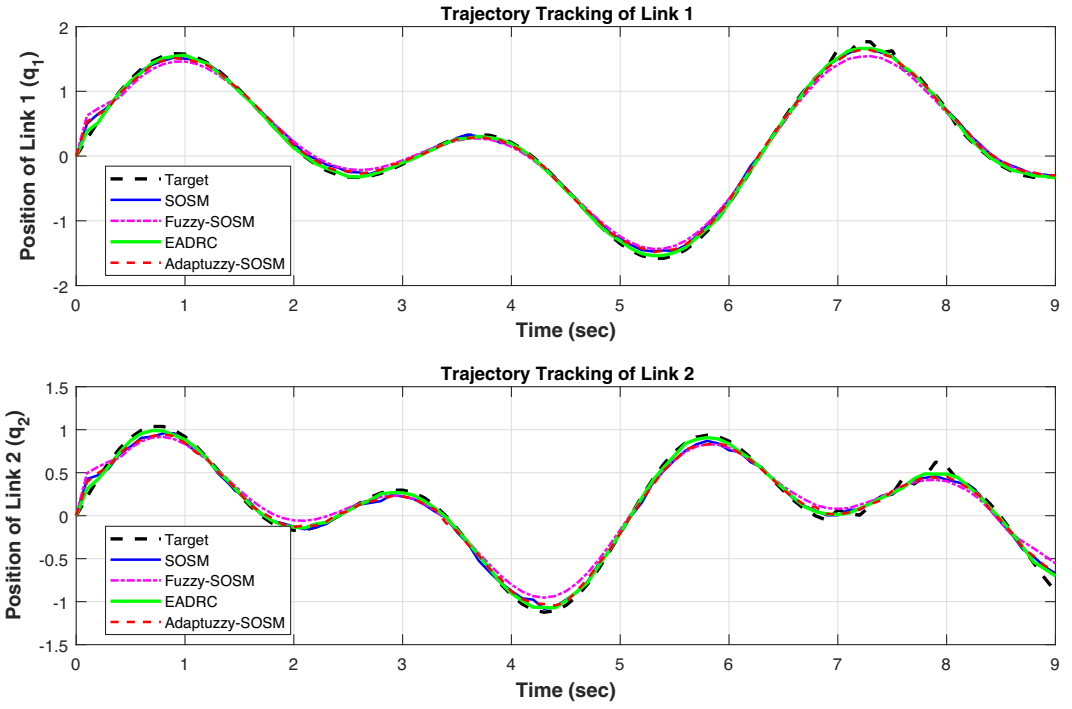


Figure 7. *Visual illustration of the system's trajectory tracking performance curves.*

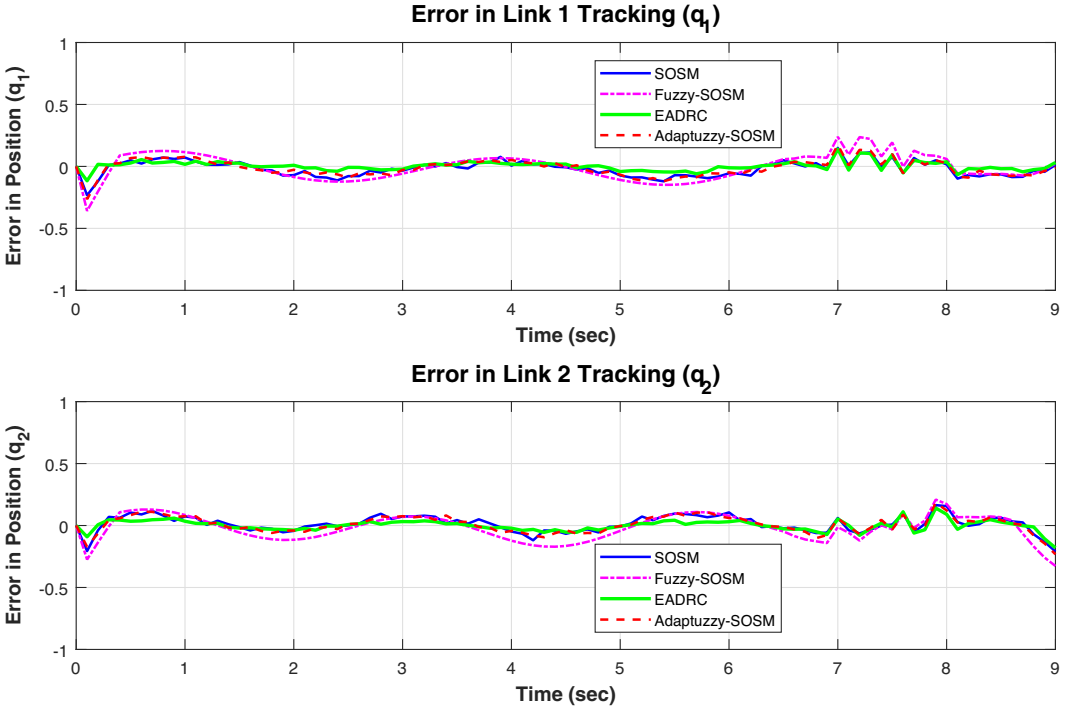


Figure 8. *Trajectory tracking error curves.*

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

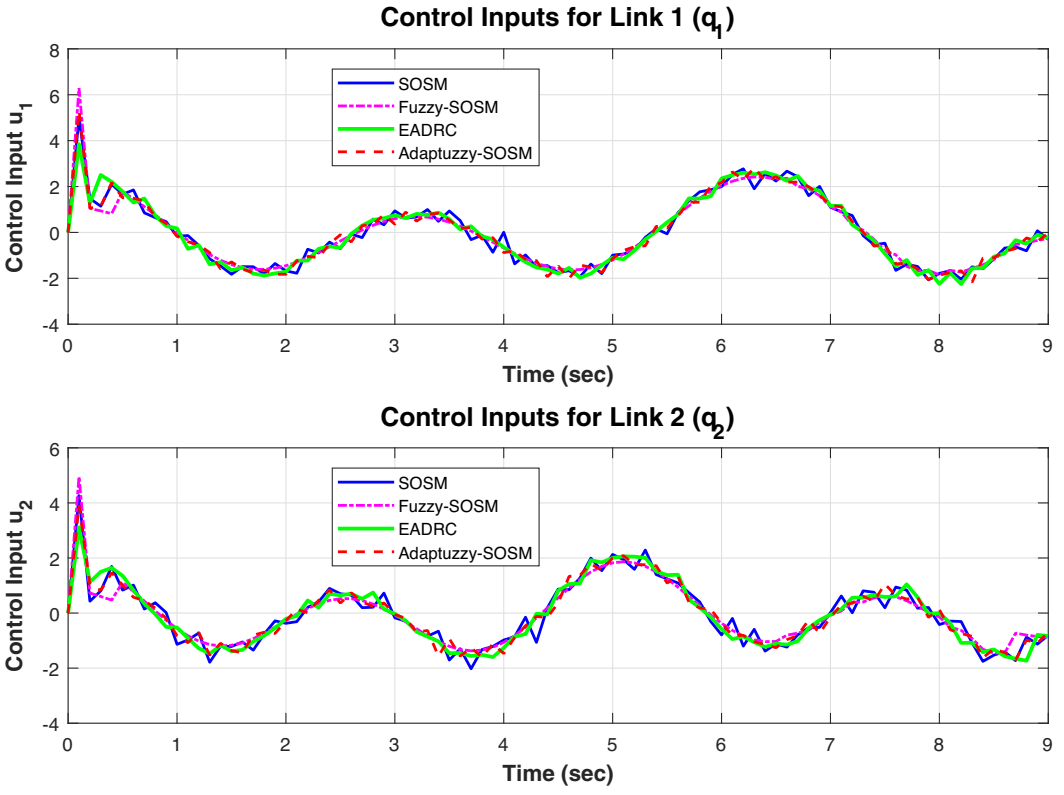


Figure 9. *Control input signals.*

components, as illustrated in Figures 7 and 8. As observed in Figure 9, except during the initial 0.1 seconds, the control signals remain confined within the range of $[-2, 2.8]$ throughout the control duration and the responses are smooth without noticeable chattering. Notably, both control inputs exhibit a high initial amplitude, which gradually stabilises. In practical applications, it is advisable to impose amplitude constraints on the control input to ensure the safety of the servo mechanisms. While such constraints may slightly compromise the tracking speed, they do not negatively affect the system's overall stability.

To further validate the results, a quantitative comparison is conducted between the proposed controller and conventional techniques. The comparison is based on the mean absolute percentage error (MAPE), defined as

$$\text{MAPE} = \frac{1}{L}\sum_{k=1}^{L}\left|\frac{r(t_k) - y(t_k)}{r(t_k)}\right| \tag{39}$$

As presented in Table 1, the proposed EADRC achieves superior performance with MAPE values of $[0.0284, 0.0117]$ for Link 1 and Link 2, respectively. This performance is approximately 2% better – or twice as accurate – compared with the Fuzzy-SOSM controller. Moreover, although the Adaptive Fuzzy-SOSM controller slightly outperforms EADRC due to the use of function approximation techniques, the EADRC still shows marked improvement over SOSM and Fuzzy-SOSM controllers. Among these, Fuzzy-SOSM outperforms SOSM significantly because the integration of fuzzy logic enhances adaptability to model uncertainties and external disturbances during trajectory tracking.

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

Table 1. *Comparison of MAPE (%) for different controllers.*

Controller	MAPE for q_1 (%)	MAPE for q_2 (%)
SOSM	2.3451	3.1297
Fuzzy-SOSM	1.9843	2.7385
EADRC	1.2147	1.8692
Adapfuzzy-SOSM	0.9561	1.5278

6. Conclusion

To address the impact of various structural uncertainties – such as unmodelled dynamics, frictional effects and external disturbances – on the trajectory tracking performance of robotic manipulators, an anti-disturbance control strategy termed EADRC was proposed. This strategy is based on an enhanced PSO algorithm and aims to improve the trajectory tracking accuracy of robotic joint actuators. The proposed control framework comprises two key components. First, it employs an ESO to effectively compensate for both internal and external disturbances. Second, a chaos-enhanced improved PSO algorithm is introduced to optimise critical controller parameters in a more robust and efficient manner. To validate the efficacy of the EADRC method, it was benchmarked against several widely used control techniques. Simulation results demonstrate that the proposed approach exhibits superior anti-disturbance capabilities, particularly under abrupt disturbances and periodic external interferences. Moreover, the tracking errors were maintained within 2%, confirming the controller's precision and robustness.

Acknowledgements. The authors would like to express their sincere gratitude to Harbin Engineering University for providing research support and resources during this work.

Data availability statement. Data will be made available on request.

Funding statement. This research received no specific grant from any funding agency in the public, commercial or not-for-profit sectors.

Competing interests. The authors declare that there are no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Aole, S., Elamvazuthi, I., Waghmare, L., Patre, B. and Meriaudeau, F. (2020). Improved Active Disturbance Rejection Control for Trajectory Tracking Control of Lower Limb Robotic Rehabilitation Exoskeleton. *Sensors*, **20**(13), 3681.

Baek, J., Kwon, W., Kim, B. and Han, S. (2018). A Widely Adaptive Time-Delayed Control and Its Application to Robot Manipulators. *IEEE Transactions on Industrial Electronics*, **66**(7), 5332–5342.

Chen, C., Du, F., Chen, B., Chen, D., He, W., Chen, Q., Zhang, C., Wu, J. and Wang, J. (2024). Practical Prescribed Tracking Control of n-DOF Robotic Manipulator with Uncertainties Via Friction Compensation Approach. *Symmetry*, **16**(4), 423.

Chen, G., Jiang, Y. and Guo, K. (2022). Neural Active Disturbance Rejection Adaptive Lateral Manipulation Control Method for Unmanned Driving Robot. *IEEE Intelligent Transportation Systems Magazine*, **15**(1), 387–399.

Cheng, X., Tu, X., Zhou, Y. and Zhou, R. (2019). Active Disturbance Rejection Control of Multi-Joint Industrial Robots based on Dynamic Feedforward. *Electronics*, **8**(5), 591.

Dai, L., Yu, Y., Zhai, D.-H., Huang, T. and Xia, Y. (2023). Robust Model Predictive Tracking Control for Robot Manipulators with Disturbances. *IEEE Transactions on industrial electronics*, **68**(5), 4288–4297.

Deylami, A. and Izadbakhsh, A. (2022). FAT-based Robust Adaptive Control of Cooperative Multiple Manipulators Without Velocity Measurement. *Robotica*, **40**(6), 1732–1762.

Du, C., Yin, Z., Zhang, Y., Liu, J., Sun, X. and Zhong, Y. (2018). Research on Active Disturbance Rejection Control with Parameter Autotune Mechanism for Induction Motors based on Adaptive Particle Swarm Optimization Algorithm with Dynamic Inertia Weight. *IEEE Transactions on Power Electronics*, **34**(3), 2841–2855.

Feng, Z. and Yao, S. (2025). Dynamic Event-Triggered Active Disturbance Rejection Formation Control for Constrained Underactuated AUVs. *IEEE/CAA Journal of Automatica Sinica*, **12**(2), 460–462.

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

Ganguly, S., Sahoo, N. C. and Das, D. (2013). Multi-Objective Particle Swarm Optimization based on Fuzzy-Pareto-Dominance for Possibilistic Planning of Electrical Distribution Systems Incorporating Distributed Generation. *Fuzzy Sets and Systems*, **213**, 47–73.

Gao, B., Shen, W., Dai, Y. and Ye, Y. T. (2022). Parameter Tuning of Auto Disturbance Rejection Controller Based on Improved Glowworm Swarm Optimization Algorithm. *Assembly Automation*, **42**(4), 427–444.

Garran, P. T. and Garcia, G. (2017). Design of an Optimal PID Controller for a Coupled Tanks System Employing ADRC. *IEEE Latin America Transactions*, **15**(2), 189–196.

Ginhoux, R., Gangloff, J., de Mathelin, M., Soler, L., Sanchez, M. M. A. and Marescaux, J. (2005). Active Filtering of Physiological Motion in Robotized Surgery using Predictive Control. *IEEE Transactions on Robotics*, **21**(1), 67–79.

Izadbakhsh, A. (2017). FAT-based Robust Adaptive Control of Electrically Driven Robots without Velocity Measurements. *Nonlinear Dynamics*, **89**(1), 289–304.

Jiang, W., Chen, D., Zheng, Y., Qiu, X. and Wang, C. (2022). Active Disturbance Rejection Control Based on Inertia Estimation and Variable Gain for Servomechanism of Industrial Robot. *Measurement and Control*, **55**(3–4), 189–197.

Kechagiopoulos, P. N. and Beligiannis, G. N. (2014). Solving the Urban Transit Routing Problem using a Particle Swarm Optimization based Algorithm. *Applied Soft Computing*, **21**, 654–676.

Kennedy, J. and Eberhart, R. (1995). Particle Swarm Optimization. *Proceedings of ICNN'95-international conference on neural networks*, IEEE, 1942–1948.

Khaled, T. A., Akhrif, O. and Bonev, I. A. (2020). Dynamic Path Correction of an Industrial Robot using a Distance Sensor and an ADRC Controller. *IEEE/ASME Transactions on Mechatronics*, **26**(3), 1646–1656.

Khosravi, M. A. and Taghirad, H. D. (2014). Robust PID Control of Fully-Constrained Cable Driven Parallel Robots. *Mechatronics*, **24**(2), 87–97.

Li, H., Cheng, L., Li, Z. and Xue, W. (2020). Active Disturbance Rejection Control for a Fluid-Driven Hand Rehabilitation Device. *IEEE/ASME Transactions on Mechatronics*, **26**(2), 841–853.

Ling, S., Wang, H. and Liu, P. X. (2019). Adaptive Fuzzy Tracking Control of Flexible-Joint Robots Based on Command Filtering. *IEEE Transactions on Industrial Electronics*, **67**(5), 4046–4055.

Liu, M., Lin, R., Yang, M., Nazarova, A. V. and Huo, J. (2018). Active Disturbance Rejection Motion Control of Spherical Robot with Parameter Tuning. *Industrial Robot: the international journal of robotics research and application*, **49**(2), 332–343.

Liu, X., Zhao, T., Cao, J. and Li, P. (2024). Design and Prediction of Self-Organizing Interval type-2 Fuzzy Wavelet Neural Network. *Information Sciences*, **661**, 120157.

Lu, Q., Wu, X., She, J., Guo, F. and Yu, L. (2024). Disturbance Rejection for Systems with Uncertainties Based on Fixed-Time Equivalent-Input-Disturbance Approach. *IEEE/CAA Journal of Automatica Sinica*, **11**(12), 2384–2395.

Luo, X., Mu, D., Wang, Z., Ning, P. and Hua, C. (2023). Adaptive Full-State Constrained Tracking Control for Mobile Robotic System with Unknown Dead-Zone Input. *Neurocomputing*, **524**, 31–42.

Ma, T., Guo, X., Su, G., Deng, H. and Yang, T. (2022). Research on Synchronous Control of Active Disturbance Rejection Position of Multiple Hydraulic Cylinders of Digging-Anchor-Support Robot. *Sensors*, **23**(8), 4092.

Ouyang, P., Zhang, W. and Gupta, M. M. (2006). An Adaptive Switching Learning Control Method for Trajectory Tracking of Robot Manipulators. *Mechatronics*, **16**(1), 51–61.

Rauf, A., Javaid, U., Ilyas, M. and Khan, A. (2025). Continuous Finite-Time Integral Sliding Mode Control of Robotic Manipulator with Input Deadzone and Disturbance/Uncertainty Compensation. *Transactions of the Institute of Measurement and Control*, 01423312251316140.

Rauf, A., Rehman, A. U., Khan, A., Abbasi, W. and Ullah, K. (2024). Adaptive Control of Robotic Manipulator with Input Deadzone and Disturbances/Uncertainties. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 09544062241238806.

Rauf, A., Zhao, Z., Khan, A., Ilyas, M. and Abbasi, W. (2023). Finite-Time Robust Control of Robotic Manipulator with Input Deadzone and Time-Varying Disturbance Compensation. *Transactions of the Institute of Measurement and Control*, **45**(6), 1140–1147.

Shah, S. A. A., Gao, B., Ahmed, N., Liu, C. and Rauf, A. (2019). Disturbance Observer-Based Sliding Mode Control of Tora System for Floating Wind Turbines. *2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*. IEEE, 418–423.

Shi, D., Zhang, J., Sun, Z., Shen, G. and Xia, Y. (2021). Composite Trajectory Tracking Control for Robot Manipulator with Active Disturbance Rejection. *Control Engineering Practice*, **106**, 104670.

Su, Y., Sun, D., Ren, L. and Mills, J. K. (2006). Integration of Saturated PI Synchronous Control and PD Feedback for Control of Parallel Manipulators. *IEEE Transactions on Robotics*, **22**(1), 202–207.

Thanh, T. D. C. and Ahn, K. K. (2006). Nonlinear PID Control to Improve the Control Performance of 2 Axes Pneumatic arTificial Muscle Manipulator using Neural Network. *Mechatronics*, **16**(9), 577–587.

Tokuda, F., Arai, S. and Kosuge, K. (2021). Convolutional Neural Network-Based Visual Servoing for Eye-to-Hand Manipulator. *IEEE Access*, **9**, 91820–91835.

Tran, D. T., Dao, H. V., Dinh, T. Q. and Ahn, K. K. (2020). Output Feedback Control Via Linear Extended State Observer for an Uncertain Manipulator with Output Constraints and Input Dead-Zone. *Electronics*, **9**(9), 1355.

Xu, B. and Lu, X. (2020). An Online Adaptive Control Strategy for Trajectory Tracking of Quadrotors based on Fuzzy Approximation and Robust Sliding Mode Algorithm. *IEEE Access*, **8**, 215327–215342.

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press

Yao, S., Gao, G., Gao, Z. and Li, S. (2020). Active Disturbance Rejection Synchronization Control for Parallel Electro-Coating Conveyor. *Isa Transactions*, **101**, 327–334.

Zhang, L., Guo, S. and Xi, F. (2023). Performance-Based Assistance Control for Robot-Mediated Upper-Limbs Rehabilitation. *Mechatronics*, **89**, 102919.

Zheng, L., Piao, Y., Ma, Y. and Wang, Y. (2020). Development and Control of Articulated Amphibious Spherical Robot. *Microsystem Technologies*, **26**(5), 1553–1561.

Zhu, E., Pang, J., Sun, N., Gao, H., Sun, Q. and Chen, Z. (2014). Airship Horizontal Trajectory Tracking Control based on Active Disturbance Rejection Control (ADRC). *Nonlinear Dynamics*, **75**, 725–734.

Cite this article: Rauf, A., Shen, W., Khan, A., Ali, S., Farid, G., and Javaid, U. (2026). Enhanced ADRC-based robust control of robotic manipulators with PSO-driven optimisation framework. *The Journal of Navigation* **79**: 1-2, 1–21. https://doi.org/10.1017/S0373463326101520

https://doi.org/10.1017/S0373463326101520 Published online by Cambridge University Press