An International Journal of Optimization and Control: Theories & Applications
ISSN: 2146-0957 eISSN: 2146-5703
Vol.15, No.4, pp.706-727 (2025)
https://doi.org//10.36922//IJOCTA025080032

RESEARCH ARTICLE

African vultures optimization-based hybrid neural network–proportional-integral-derivative controller for improved robot manipulator tracking

Bashra Kadhim Oleiwi[1*], Mohamed Jasim Mohamed[1], Ahmad Taher Azar[2,3], Saim Ahmed[2,3*], Ahmed Redha Mahlous[2,3], and Walid El-Shafai[2,3,4]

[1]Mechatronics and Robotics Engineering Department, College of Control and Systems Engineering, University of Technology, Baghdad, Iraq
[2]College of Computer and Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia
[3]Automated Systems and Computing Lab (ASCL), Prince Sultan University, Riyadh, Saudi Arabia
[4]Department of Electronics and Electrical Communication Engineering, Faculty of Electronic Engineering, Menoufia University, Menouf, Monufia, Egypt
bushra.k.oleiwi@uotechnology.edu.iq, 60098@uotechnology.edu.iq, aazar@psu.edu.sa, sahmed@psu.edu.sa, arMahlous@psu.edu.sa, eng.waled.elshafai@gmail.com

ARTICLE INFO

Article History:
Received: February 20, 2025
1st revised: February 26, 2025
2nd revised: April 22, 2025
3rd revised: May 6, 2025
4th revised: June 29, 2025
Accepted: July 2, 2025
Published Online: September 4, 2025

Keywords:
3-Link rigid robotic manipulator
African vultures optimization algorithm
Neural network
Proportional-integral-derivative controller
Self-tuning proportional-integral-derivative controller
Trajectory tracking

ABSTRACT

Rigid robotic manipulators encounter several challenges in trajectory tracking control, including low accuracy and poor stability, resulting from uncertainties, external disturbances, and parameter variations. To address these issues, this study proposes two hybrid controllers that integrate the strengths of proportional-integral-derivative (PID) control with neural network (NN) methods for a three-link rigid robotic manipulator. These hybrid structures are the NN–PID controller and the self-tuning NN with PID (STNN–PID) controller. Their performance is compared against that of a conventional PID controller. To optimize control performance metrics, such as the integral time square error (ITSE), the parameters of the proposed controllers were tuned using the African vultures optimization algorithm. MATLAB was used to evaluate the effectiveness. Robustness tests were performed by varying the initial conditions, introducing external disturbances, and modifying system parameters. The NN–PID controller achieved ITSE values of 0.28919×10^{-4}, 0.064321, and 0.001164, respectively, while the STNN–PID controller yielded values of 3.54549×10^{-4}, 3.526199, and 0.883710, respectively. Moreover, when all these conditions were applied simultaneously, the NN–PID controller achieved an ITSE of 0.073968, compared to 2.672754 for the STNN–PID controller. These results demonstrate that the NN–PID controller outperforms the other controllers across all testing conditions. These findings confirm that the NN–PID controller is the most effective controller in terms of tracking accuracy, stability, and robustness across all test scenarios.

1. Introduction

The use of robotic manipulators in industrial applications has grown significantly in recent years.[1] These manipulators primarily serve as handling and positioning tools. Their applications span various fields, including spray painting, automated assembly lines, the handling of hazardous radioactive materials, fabrication, freight loading and unloading, and military operations.[2] A robust and efficient robotic manipulator must be capable of regulating both its motion and the forces exerted on its environment. However, due to their

---

*Corresponding Authors

nonlinear, multi-input–multi-output (MIMO) nature, industrial robotic manipulators are susceptible to nonlinear friction, payload variations, and external disturbances.[3] Consequently, designing an optimal controller requires a comprehensive understanding of the system dynamics.[4] Currently, one of the most critical challenges for researchers and engineers is to develop controllers that provide optimal tracking performance. This challenge arises from the inherent complexity of controlling the position, velocity, and orientation of nonlinear dynamical robotic manipulators.[5,6]

In recent years, numerous studies have proposed various control structures and hybrid schemes for two-link rigid robotic manipulators (2-LRRM) addressing the position tracking problem. Mohamed et al.[7] developed six control structures by integrating neural networks (NNs) with both integer- and fractional-order proportional-integral-derivative (FOPID) controllers, optimized using the gorilla troops optimization algorithm. Likewise, Saha et al.[8] introduced an adaptive framework combining proportional-integral-derivative (PID) and NN controllers for effective trajectory control of robot manipulators. In the study by Mohammed et al.,[9] a novel hybrid control strategy was presented, integrating backstepping control with nonlinear reduced-order active disturbance rejection control. The global stability of the proposed controller was rigorously verified using the Lyapunov method. Similarly, Tlijani et al.[10] proposed a non-singular fast terminal sliding mode control (SMC) strategy, incorporating a wavelet NN observer to ensure precision and stability in joint control. The study by Bankole and Igbonoba[11] presented a hybrid control strategy that combined H-infinity (H) control methodology with proportional-derivative (PD) control, demonstrating superior performance over standalone PD and H∞ controllers. Furthermore, Esmail et al.[12] developed a reliable P-H∞ controller, which integrates proportional and H∞ elements to enhance disturbance rejection, position tracking accuracy, and vibration suppression.

Villa-Tiburcio et al.[13] proposed a novel intelligent control algorithm for force/position control of robotic manipulators, integrating traditional PID/proportional integral (PI) control schemes with back-propagation NNs. In this approach, the back-propagation NNs are responsible for estimating and compensating for dynamic variations in the automated fiber placement process in real time, while the PID/PI controllers handle the core control actions. In the study by Bankole and Igbonoba,[14] a multilayer perceptron NN structure based on a nonlinear autoregressive exogenous model was introduced, integrating PD and H∞ control methods. Likewise, Fani and Shahraki[15] employed genetic algorithms and the estimation of distribution algorithm to select the optimal coefficients for a FOPID controller. Numerical results indicated that the FOPID scheme is more effective when applied to actual robot models, especially when optimized using the particle swarm optimization (PSO) algorithm. Anwaar et al.[16] applied the Nelder–Mead optimization technique to enhance the FOPID controller's ability to mitigate residual tracking errors. In another study, Khalil and Sharkawy[17] introduced an adaptive hybrid PID control strategy for manipulators and compared its performance with SMC, demonstrating the effectiveness and reliability of the proposed approach. More recently, several studies have extended the focus to control structures and hybrid controllers for 3-LRRM, aiming to solve the path tracking problem. In this context, Vineet et al.[18] proposed a self-regulated fuzzy FOPID scheme optimized using a backtracking search algorithm.

Ahmad et al.[19] presented an NN-based PID controller, where the proposed artificial NN, consisting of two layers and utilizing feedforward training, was trained using the back-propagation algorithm. In the study by Kumar et al.,[20] a self-tuned fuzzy FOPID controller optimized via the cuckoo search algorithm was employed to address the behavior of a nonlinear system. Meanwhile, Tohma and Hamoudi[21] focused on designing two types of controllers: an adaptive SMC with a barrier function and a saturation function, and a standard SMC with the same components, aiming to mitigate the chattering phenomenon. Zhu et al.[22] developed a hybrid control scheme combining PID with a fuzzy NN, with parameters tuned to enable simultaneous trajectory and contact force tracking. Similarly, Sathish Kumar et al.[23] proposed a fuzzy direct current linear servo controller for robotic arm control, optimized using the PSO technique. The study by Gümüş et al.[24] introduced a cascade PD controller enhanced by the bees algorithm to regulate a flexible robot arm and suppress tip vibration. Likewise, Nohooji[25] developed a novel adaptive neural-based control scheme utilizing a simplified PID-like structure to manage a robot subjected to external disturbances and incomplete dynamic modeling. Using a direct Lyapunov method, the PID gains were determined, and uncertainties were estimated through a radial basis function (RBF) NN. Additionally, the control strategy is refined to ensure constrained behavior throughout system operation.

In their study, Cheyun et al.[26] proposed a robust adaptive controller that integrates SMC with an RBF NN. In this approach, the unknown dynamics are approximated using the RBF network, while the adaptive update law for the network's parameters is derived based on Lyapunov stability theory. Mohamed et al.[27] introduced six control structures utilizing FOPID and conventional PID (con-PID) controllers, all of which were enhanced through NN integration. These structures included recurrent NN-like FOPID, recurrent NN-like PID, set point-weighted PID, set point-weighted FOPID, hybrid NN–FOPID, and NN–PID controllers. The zebra optimization algorithm was employed to fine-tune the parameters of these controllers with the aim of minimizing the integral time square error (ITSE). Similarly, Chotikunnan and Chotikunnan[28] presented a dual-design PID controller, which utilized two PID controllers—one operating in real-time and the other with a one-time-step delay. The performance of this dual design was evaluated against a con-PID controller, using the Chien–Hrones–Reswick tuning method to achieve minimal overshoot. Mary et al.[29] proposed a fuzzy-like PD controller combined with SMC. This controller was developed based on fuzzy rules, membership functions, and the nominal model of the robotic manipulator.

Coutinho and Hess-Coelho[30] proposed two hybrid control techniques: one combining pure computed torque control with a modified SMC, and the other integrating pure PD control with SMC. In the study by Yahya and Abbas,[31] an enhanced version of the parallel Harris hawks optimization algorithm was introduced along with integral SMC to identify optimal controller parameters. The proposed method was compared with other algorithms, including the standard Harris hawks optimization, demonstrating improved performance. Meanwhile, Çetinkaya et al.[32] conducted positioning analysis using NNs for robot manipulator systems in the textile industry and refined the most accurate NN model for high-precision positioning tasks. In the study by Saeed Al-Khayyt,[33] a self-tuning NN–PID (STNN–PID) controller was developed, combining the adaptive learning capabilities of NNs with a con-PID controller. The NN output, derived from the system's control error, dynamically adjusts the proportional, integral, and derivative gains in real time. To enhance dynamic performance—particularly to minimize overshoot and achieve accurate trajectory tracking with minimal steady-state error—the con-PID controller was replaced with this STNN–PID controller. Adnan and Karam[34] proposed two improved PID controller structures designed to enhance the con-PID's performance in guiding a human leg robot along a predefined path. The grasshopper optimization algorithm was utilized to optimize the improved PID parameters. In the study by Yadav et al.,[35] various soft computing techniques were applied to fine-tune and implement a hybrid controller for robotic manipulator motion control. This hybrid controller combined traditional PID control with an "overwhelming" controller, and the PID parameters were optimized using methods such as Ziegler–Nichols tuning, PSO, ant colony optimization, and genetic algorithms.

Over recent decades, numerous control strategies have been explored in the literature, including classical, adaptive, and SMC theories. However, these approaches often face limitations in adaptability, as they typically account for only a narrow range of disturbances. Moreover, due to the inherent mechanical complexity of robotic systems, they exhibit highly nonlinear dynamics, making the modeling and control processes particularly challenging. This has motivated the development of more resilient and effective controllers capable of handling such complexities.

The primary objective of this work is to design hybrid controllers that integrate NNs with PID operations to leverage the advantages of both techniques in control system design. The proposed controllers require parameter tuning to achieve specific performance objectives. To accomplish this, the African vultures optimization algorithm (AVOA) is employed to identify the optimal set of parameters for the hybrid controllers. This study focuses on designing a controller for a 3-LRRM to enable precise position control by tracking a desired trajectory, while also demonstrating robustness against external disturbances and uncertainties within the system model. The main contributions of this work are as follows:

(i) This work designs two hybrid control structures that combine NNs with PID operations. The first structure, STNN–PID, is a self-tuning PID controller based on NNs. The second structure, NN–PID, is a newly proposed hybrid controller that incorporates PID operations as activation functions within the NN architecture.

(ii) A new technique is proposed to eliminate controllers that exhibit complex control actions with excessive chattering, which are impractical for real-world implementation. This technique is applied and shown to be highly effective.

(iii) The AVOA is employed to optimize the parameters of the proposed hybrid controllers. The algorithm demonstrates strong capability in identifying optimal parameters within complex nonlinear search spaces.
(iv) A series of tests is conducted to determine the most effective controller. All controllers are evaluated under varying initial conditions, the presence of external disturbances, and changes in system parameters.

This work is structured as follows: Section 2 introduces the nonlinear dynamic model of the 3-LRRM; Section 3 presents the design of the proposed controllers; Section 4 details the AVOA technique; Section 5 provides the simulation results; and Section 6 concludes the study.

2. Three-link rigid robotic manipulator nonlinear dynamical system

The 3-LRRM robot consists of three links, each connected to its adjacent link through a joint. These linkages form a nonlinear robotic manipulator. A planar robotic manipulator only operates within a single plane.[36] Prototypes of such nonlinear planar rigid manipulators are commonly utilized in automated systems and medical applications. In this study, a planar nonlinear rigid robotic arm with three revolute joints was considered, with each joint assumed to be equipped with an actuator.[18] The structure of the 3-LRRM is illustrated in Figure 1. The first link is attached to a fixed base via a frictionless pivot joint. The opposite end of the first link connects to the second link through a frictionless ball bearing, and similarly, the second link connects to the third link in the same manner.[37] The dynamic equations governing the manipulator's motion are essential for analyzing the system and designing effective control strategies. In robotic systems, the dynamic motion of the manipulator arms is generated by the control torques applied by the actuators.[38] The dynamic model of the 3-LRRM is presented in the following sections.

The Lagrange dynamic of the 3-LRRM is illustrated as follows[39]: The (x_1, y_1) position for link 1 is calculated[39] and is shown in Equations (1) and (2):

$$x_1 = L_a\ cos\ (\psi_1) \tag{1}$$

$$y_1 = L_a\ sin\ (\psi_1) \tag{2}$$

Correspondingly, the (x_2, y_2) position for link 2 is represented in Equations (3) and (4):

$$x_2 = L_a\ cos\ (\psi_1) + L_b\ cos\ (\psi_1 + \psi_2) \tag{3}$$

$$y_2 = L_a\ sin\ (\psi_1) + L_b\ sin\ (\psi_1 + \psi_2) \tag{4}$$

while the (x_3, y_3) position for link 3 is computed as in Equations (5) and (6):

$$\begin{aligned} x_3 &= L_a\ cos\ (\psi_1) + L_b\ cos\ (\psi_1 + \psi_2) \\ &+ L_c \cos(\psi_1 + \psi_2 + \psi_3) \end{aligned} \tag{5}$$

$$\begin{aligned} y_3 &= L_a\ sin\ (\psi_1) + L_b\ sin\ (\psi_1 + \psi_2) \\ &+ L_c \sin(\psi_1 + \psi_2 + \psi_3) \end{aligned} \tag{6}$$

where L_i, ψ_i, x_i, and y_i are the length of the link, angle of the link, x-position, and y-position of the link i, respectively.

The equation of kinetic energy (KEn) is defined as in Equation (7):

$$KEn = \frac{1}{2}\ M_a\ V_a^2 + \frac{1}{2}\ M_b\ V_b^2 + \frac{1}{2}\ M_c\ V_c^2 \tag{7}$$

where V_a, V_b, and V_c are the velocities of links, M_a, M_b, and M_c are the masses of links. The velocities are computed as in Equation (8).

$$\begin{aligned} V_a &= \sqrt{\dot{x}_1^2 + \dot{y}_1^2}, \\ V_b &= \sqrt{\dot{x}_2^2 + \dot{y}_2^2}, \\ V_c &= \sqrt{\dot{x}_3^2 + \dot{y}_3^2} \end{aligned} \tag{8}$$

Therefore, the kinetic energy can be expressed as in Equation (9).

$$\begin{aligned} KEn &= \frac{1}{2} M_a\ (\dot{x}_1^2 +\ \dot{y}_1^2) + \frac{1}{2}\ M_b\ \ (\dot{x}_2^2 + \dot{y}_2^2) \\ &+ \frac{1}{2}\ M_c\ \ (\dot{x}_3^2 + \dot{y}_3^2) \end{aligned} \tag{9}$$

Also, the equations of potential energy (PEn) can be written as in Equations (10) and (11).

$$PEn = \sum_{i=1}^{3} M_i\ g\ h_i \psi(10) \tag{10}$$

where g is the gravity and h_i is the height of the link i, where $i = a, b, c$.

$$\begin{aligned} PEn &= M_a\ g\ L_a \sin(\psi_1) + M_b\ g\,(L_a\ \ \sin(\psi_1) \\ &+ L_b \sin(\psi_1 + \psi_2) \qquad + \qquad M_c g(L_a \sin(\psi_1) \\ &+ L_b \sin(\psi_1 + \psi_2) + L_c \sin(\psi_1 + \psi_2 + \psi_3)) \end{aligned} \tag{11}$$

Subsequently, the Lagrange dynamic (LD) is defined as in Equation (12);

$$LD = KEn - PEn \tag{12}$$

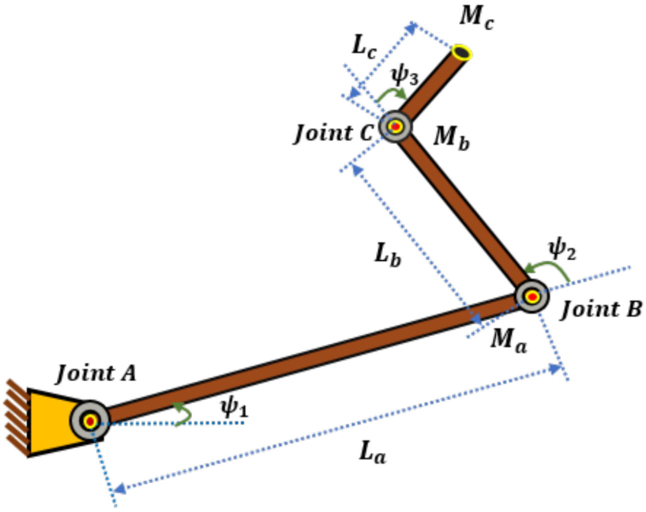

Figure 1. Three-link rigid robotic manipulator model

And the Euler–Lagrange expression is written as in Equation (13):

$$\frac{d}{dt}\left[\frac{\partial LD}{\partial \dot{\psi}_i}\right]-\frac{\partial LD}{\partial \psi_i}=F\psi_i \tag{13}$$

where $F\psi_i$ or τ_i is the control torque applied to the *i-th* link.

The robotic manipulator dynamics are given in Equations (14)–(24)[39]:

$$Q(\psi)\ddot{\psi}+F(\psi,\dot{\psi}^2)+R(\psi,\dot{\psi}_i\dot{\psi}_j)+P(\psi)=\tau \tag{14}$$

where $Q(\theta)$ is the inertia matrix.

$$Q=\begin{bmatrix} Q_{11} & Q_{12} & Q_{13} \\ Q_{21} & Q_{22} & Q_{23} \\ Q_{31} & Q_{32} & Q_{33} \end{bmatrix} \tag{15}$$

$$Q_{11}=(M_a+M_b+M_c)\,L_a^2+(M_b+M_c)\,L_b^2 + M_cL_c^2+2M_cL_aL_c\ \cos(\psi_2+\psi_3)+2(M_b + M_c)L_aL_b\cos(\psi_2)+2M_cL_bL_ccos(\psi_3) \tag{16}$$

$$Q_{12}=(M_b+M_c)\,L_b^2+M_cL_c^2+M_cL_aL_c\cos(\psi_2+\psi_3) +(M_b+M_c)\,L_aL_b\cos(\psi_2)+2M_cL_bL_ccos(\psi_3) \tag{17}$$

$$Q_{13}=M_cL_c^2+M_cL_aL_c\cos(\psi_2+\psi_3) +M_cL_bL_ccos(\psi_3) \tag{18}$$

$$Q_{21}=M_bL_b^2+M_cL_b^2+M_cL_c^2+M_cL_aL_c\cos(\psi_2 +\psi_3)+M_bL_aL_b\cos(\psi_2)+M_cL_aL_b\cos(\psi_2) +2M_cL_bL_ccos(\psi_3) \tag{19}$$

$$Q_{22}=M_bL_b^2+M_cL_b^2+M_cL_c^2+2M_cL_bL_ccos(\psi_3) \tag{20}$$

$$Q_{23}=M_cL_c^2+M_cL_bL_ccos(\psi_3) \tag{21}$$

$$Q_{31}=M_cL_c^2+M_cL_aL_c\cos(\psi_2 +\psi_3)+M_cL_bL_ccos(\psi_3) \tag{22}$$

$$Q_{32}=M_cL_c^2+M_cL_bL_ccos(\psi_3) \tag{23}$$

$$Q_{33}=M_c\ L_c^2 \tag{24}$$

The term of centrifugal forces $F\left(\psi,\dot{\psi}^2\right)$ is shown in Equation (25):

$$F=[F_1\ F_2\ F_3]^T \tag{25}$$

The centrifugal forces of all links are given in Equations (26)–(28):

$$F_1=-L_a(M_cL_c\sin(\psi_2+\psi_3)+M_bL_b\sin(\psi_2) +M_cL_b\sin(\psi_2)\,\dot{\psi}_2^2-M_cL_c(L_a\sin(\psi_2+\psi_3) +L_b\sin(\psi_3))\dot{\psi}_3^2 \tag{26}$$

$$F_2=L_a\,(M_cL_c\sin(\psi_2+\psi_3)+M_bL_b\sin(\psi_2) +M_cL_b\sin(\psi_2)\dot{\psi}_1^2-M_cL_bL_csin(\psi_3)\dot{\psi}_3^2 \tag{27}$$

$$F_3=M_cL_c\,(L_a\sin(\psi_2+\psi_3)+L_b\sin(\psi_3))\,\dot{\psi}_1^2 +M_cL_bL_csin(\psi_3)\dot{\psi}_2^2 \tag{28}$$

The term of Coriolis $R(\psi,\dot{\psi}_i\dot{\psi}_j)$ is shown in Equation (29):

$$R=[R_1\ R_2\ R_3]^T \tag{29}$$

and the Coriolis terms of all links are given in Equations (30)–(32):

$$\begin{aligned} R_1 = &-2L_a\left(M_cL_c\sin\left(\psi_2+\psi_3\right)+\left(\psi_2+\psi_3\right)\right.\\ &\times L_b\sin\left(\psi_2\right)\dot{\psi}_1\dot{\psi}_2-2M_cL_c\left(L_a\sin\left(\psi_2+\psi_3\right)\right.\\ &+L_b\sin\left(\psi_3\right)\dot{\psi}_2\dot{\psi}_3-2\ M_cL_c\left(L_a\sin(\psi_2+\psi_3)\right.\\ &\left.+L_b\sin\left(\psi_3\right)\right)\dot{\psi}_1\dot{\psi}_3 \end{aligned} \tag{30}$$

$$\begin{aligned} R_2 = &-2M_cL_bL_c\sin\left(\psi_3\right)\dot{\psi}_1\dot{\psi}_3\\ &-2M_cL_bL_csin(\psi_3)\dot{\psi}_2\dot{\psi}_3 \end{aligned} \tag{31}$$

$$R_3 = 2M_cL_bL_csin(\psi_3)\dot{\psi}_1\dot{\psi}_2 \tag{32}$$

The term of potential energy $P(\psi)$ is shown in Equation (33):

$$P = \left[P_1\ P_2\ P_3\right]^T \tag{33}$$

and the potential energy terms of all links are shown in Equations (34)–(36):

$$\begin{aligned} P_1 = &\left(M_a+M_b+M_c\right)gL_a\cos\left(\psi_1\right)\\ &+\left(M_b+M_c\right)gL_b\cos\left(\psi_1+\psi_2\right)\\ &+M_cgL_ccos(\psi_1+\psi_2+\psi_3) \end{aligned} \tag{34}$$

$$\begin{aligned} P_2 = &\left(M_b+M_c\right)gL_b\cos(\psi_1\\ &+\psi_2)+M_cgL_ccos(\psi_1+\psi_2+\psi_3) \end{aligned} \tag{35}$$

$$P_3 = M_cgL_ccos(\psi_1+\psi_2+\psi_3) \tag{36}$$

By applying forward kinematics[39] and using the desired joint angles ψ_{r1}, ψ_{r2}, and ψ_{r3}, the coordinates of the required 3-LRRM's end-effector can be calculated, as shown in Equations (37) and (38) for the reference trajectory:

$$\begin{aligned} x_r = &L_a\cos\left(\psi_{r1}\right)+L_b\cos\left(\psi_{r1}+\psi_{r2}\right)\\ &+L_c\cos\left(\psi_{r1}+\psi_{r2}+\psi_{r3}\right) \end{aligned} \tag{37}$$

$$\begin{aligned} y_r = &L_a\sin\left(\psi_{r1}\right)+L_b\sin\left(\psi_{r1}+\psi_{r2}\right)\\ &+L_c\sin\left(\psi_{r1}+\psi_{r2}+\psi_{r3}\right) \end{aligned} \tag{38}$$

where ψ_{r1}, ψ_{r2}, and ψ_{r3} are the angles of the desired trajectories, and x_r, y_r are the desired coordinates of the end effector. Table 1 illustrates the nominal values of the 3-LRRM parameters used in this study.[39]

Table 1. Three-link rigid robotic manipulator parameters

Parameter	Nominal value
L_a	0.8 m
L_b	0.4 m
L_c	0.2 m
M_a	0.1 kg
M_b	0.1 kg
M_c	0.1 kg
g	9.81 m/s^2

3. The proposed controller structures

This section provides a detailed description of the proposed hybrid controllers. For simplicity, the initial analysis focuses on single-input single-output systems to facilitate primary demonstration. However, the proposed control structures are readily extendable to MIMO systems. The strategy of constructing the controllers in this study is different from that used in the previous study.[39] Moreover, the current study incorporated different control structures than those presented previously.[39] The first strategy is represented by the STNN–PID controllers, where the construction of this controller represents the collaboration between STNNs and PID processes. In other words, the controller nature is still a PID controller, equipped with self-tuning technology enabled by NNs. The second strategy is represented by the NN–PID controller, which is a hybrid controller that merges the NN with PID operations. The previous study introduced different controller structures that all belong to the second strategy.[39]

3.1. Conventional proportional-integral-derivative controller with filter

Among various control strategies, the PID method remains a widely adopted approach due to its inherent simplicity and effectiveness. A PID controller combines three fundamental control actions: proportional, integral, and derivative. The proportional component accelerates the system's response, while the integral component works to eliminate steady-state errors. However, the derivative component is highly sensitive to measurement noise, which can lead to excessive control activity in response to small error fluctuations. To address this issue, a filter is commonly applied to the derivative term to suppress noise, thereby enhancing the robustness and reliability of the controller's performance.

The transfer function of a PID controller incorporating a filter is expressed in Equation (39):

$$G(s)_{PID} = K_p + K_i \frac{1}{S} + K_d s\ \frac{N}{S+N} \tag{39}$$

The performance of a PID controller is determined by three primary gain parameters: K_p, K_i, and K_d, which control the proportional, integral, and derivative responses, respectively. Additionally, the parameter N represents the filter's corner frequency and plays a critical role in attenuating high-frequency noise in the derivative term. Figure 2 illustrates the schematic structure of the con-PID controller with a filter. For further clarification, Figure 3 presents the block diagram of a feedback control system that includes the PID controller.

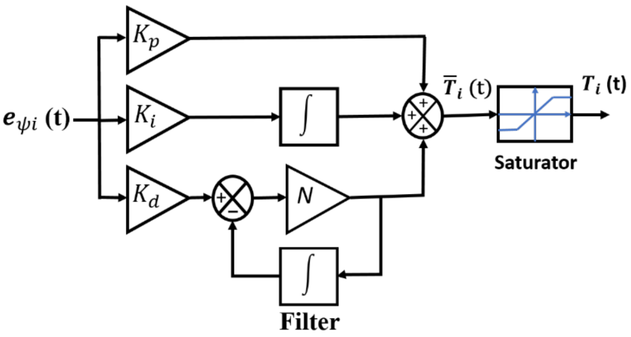

Figure 2. Proportional-integral-derivative controller with filter

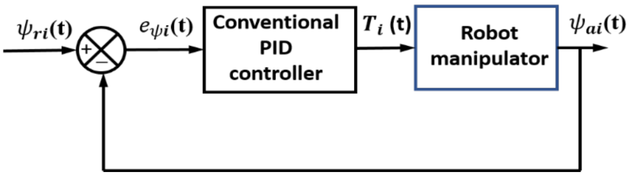

Figure 3. Block diagram of a conventional proportional-integral-derivative (PID) control system

3.2. Proportional-integral-derivative controller with integrated self-tuning neural network and filter

A STNN–PID controller consists of two primary components. The first is a pre-trained NN that dynamically adjusts the values of K_p, K_i, and K_d in the PID controller to improve system performance. The second component is the con-PID controller, which generates the control signal by applying proportional, integral, and derivative operations to the error between the desired and actual outputs. The NN is trained offline to meet a specific performance objective and then used to supply the PID controller with updated gain values during operation. It is important to note that the filter parameter N, representing the corner frequency (or equivalently, the time constant as $1/N$, is not adjusted dynamically. Instead, it must be tuned offline alongside the NN parameters prior to deployment.

Figure 4 illustrates the architecture of the NN component within the STNN–PID controller. The network comprises several layers, including an input layer, a first hidden layer, and an activation function. The input layer consists of four neurons, each representing a key variable: the applied torque for the *i-th* link T_i, the desired angular position (ψ_{ri}), the actual angular position (ψ_{ai}), and the position error between the desired and actual angles ($e_{\psi i}$). The first hidden layer also contains four neurons, each connected to all neurons in the input layer as well as to a bias neuron through weighted connections. For each neuron in the first hidden layer, the activation function $H(\Sigma)$ is applied to the weighted sum of its inputs to produce the neuron's output. The input vector is defined in Equation (40), while the outputs of the first hidden layer neurons are given in Equations (41) and (42). The specific activation function employed is described in Equation (43):

$$\begin{bmatrix} \psi_{ai} \\ \psi_{ri} \\ e_{\psi i} \\ T_i \\ 1 \end{bmatrix} \text{Input layer neurons of the neural network} \tag{40}$$

$$\begin{bmatrix} \sum(k)_1^1 \\ \sum(k)_2^1 \\ \sum(k)_3^1 \\ \sum(k)_4^1 \end{bmatrix} = \begin{bmatrix} w11 & w12 & w13 & w14 & w15 \\ w21 & w22 & w23 & w24 & w25 \\ w31 & w32 & w33 & w34 & w35 \\ w41 & w42 & w43 & w44 & w45 \end{bmatrix} \begin{bmatrix} \psi_{ai} \\ \psi_{ri} \\ e_{\psi i} \\ T_i \\ 1 \end{bmatrix} \tag{41}$$

$$\begin{bmatrix} S_1^1(k) \\ S_2^1(k) \\ S_3^1\ (k) \\ S_4^1\ (k) \end{bmatrix} = \begin{bmatrix} H(\sum(k)_1^1) \\ H(\sum(k)_2^1) \\ H\left(\sum(k)_3^1\right) \\ H\left(\sum(k)_4^1\right) \end{bmatrix} \tag{42}$$

where $\sum(k)_i^1$ is defined as the sum of input connections for each neuron in the first hidden layer, while $S_i^1(k)$ represents the *i-th* output of the same layer's neuron, and *wij* are weights of connections.

$$H(\Sigma) = \left(2 - \Sigma^2\right) \times\ e^{-\Sigma^2} \tag{43}$$

The second hidden layer, similar in structure to the first, consists of four neurons. Each neuron in this layer is fully connected to all outputs of the neurons in the first hidden layer, in addition to a bias unit, through corresponding connection weights. The output of each neuron is computed by summing its total weighted inputs, as described in Equations (44) and (45):

$$\begin{bmatrix} \sum(k)_1^2 \\ \sum(k)_2^2 \\ \sum(k)_3^2 \\ \sum(k)_4^2 \end{bmatrix} = \begin{bmatrix} v11 & v12 & v13 & v14 & v15 \\ v21 & v22 & v23 & v24 & v25 \\ v31 & v32 & v33 & v34 & v35 \\ v41 & v42 & v43 & v34 & v45 \\ v51 & v52 & v53 & v54 & v55 \end{bmatrix} \begin{bmatrix} S_1^1(k) \\ S_2^1(k) \\ S_3^1(k) \\ S_4^1(k) \\ 1 \end{bmatrix} \tag{44}$$

$$\begin{bmatrix} S_1^2(k) \\ S_2^2(k) \\ S_3^2(k) \\ S_4^2(k) \end{bmatrix} = \begin{bmatrix} \sum(k)_1^2 \\ \sum(k)_2^2 \\ \sum(k)_3^2 \\ \sum(k)_4^2 \end{bmatrix} \tag{45}$$

where $\sum(k)_i^2$ is defined as the sum of input connections for each neuron in the second hidden layer, while $S_i^2(k)$ represents the $i\text{-}th$ output of the same layer's neuron, and vij are weights of connections.

The output layer, also known as the final layer, consists of three neurons. Each neuron in this layer is connected to all outputs from the second hidden layer, as well as to a bias unit, through weighted connections. The output of each neuron is calculated as the weighted sum of its inputs. These three outputs correspond directly to the PID controller parameters K_p, K_i, and K_d as represented in Equations (46) and (47). These values are then applied in the same manner as in a con-PID controller. The architecture of the self-tuning PID controller and the complete feedback control system is illustrated in Figure 5.

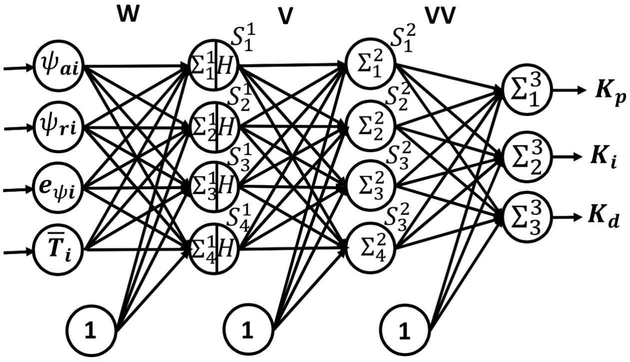

Figure 4. The neural network of the self-tuning proportional-integral-derivative controller

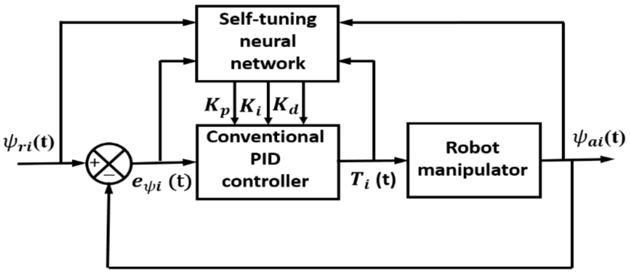

Figure 5. Block diagram of the self-tuning neural network–proportional-integral-derivative (PID) control system

$$\begin{bmatrix} \sum(k)_1^3 \\ \sum(k)_2^3 \\ \sum(k)_3^3 \end{bmatrix} = \begin{bmatrix} vv11 & vv12 & vv13 & vv14 & vv15 \\ vv21 & vv22 & vv23 & vv24 & vv25 \\ vv31 & vv32 & vv33 & vv34 & vv35 \end{bmatrix} \begin{bmatrix} S_1^2(k) \\ S_2^2(k) \\ S_3^2(k) \\ S_4^2(k) \\ 1 \end{bmatrix} \tag{46}$$

$$\begin{bmatrix} K_p \\ K_i \\ K_d \end{bmatrix} = \begin{bmatrix} \sum(k)_1^3 \\ \sum(k)_2^3 \\ \sum(k)_3^3 \end{bmatrix} \tag{47}$$

where $\sum(k)_i^3$ is defined as the sum of input connections for each neuron in the output layer, while (K_p, K_i, and K_d) are neurons' outputs of the output layer, and $vvij$ are weights of connections.

3.3. Neural network–proportional-integral-derivative controller

The proposed hybrid NN–PID controller is depicted in Figure 6. A single neuron representing the error $e_{\psi i}(k)$ between the desired and actual control variables constitutes the input layer. Three neurons constitute the first hidden layer, imitating the PID functions of a traditional PID controller. The $e_{\psi i}(k)$, multiplied by a weight assigned to each neuron, serves as the input for these neurons. These weights function similarly to the gains (K_p ,K_i , and K_d) in con-PID control, as shown in Equations (48)–(51).

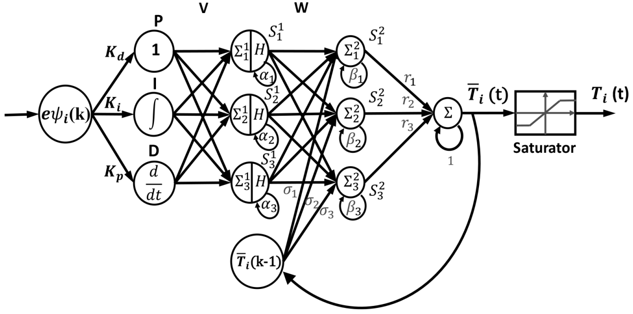

Figure 6. The neural network of a neural network–proportional-integral-derivative controller

$$Sum(k) = Sum(k-1) + h \times e_{\psi}(k) \tag{48}$$

where $Sum(0) = 0$.

$$P(k) = K_p \times e_{\psi}(k) \tag{49}$$

$$I(k) = K_i \times Sum(k) \tag{50}$$

$$D(k) = K_d \times (e_{\psi}(k) - e_{\psi}(k-1))/h \tag{51}$$

Within the first hidden layer, the variable Sum represents the accumulated value used for the integral operation. The outputs of the three neurons in this layer, denoted as $P(k)$, $I(k)$, and $D(k)$, correspond to the proportional, integral, and derivative components of the error signal, respectively. The parameter h represents the step size used in the simulation.

The second hidden layer also consists of three neurons. Each neuron receives weighted inputs from all outputs of the first hidden layer. The sum of these weighted inputs is processed through a nonlinear activation function $H(\sum)$. The result of the activation function is then combined with the previous output of the same neuron to produce the current output, as described in Equations (52) and (53). The activation function is defined in Equation (54):

$$\begin{bmatrix} \sum(k)_1^1 \\ \sum(k)_2^1 \\ \sum(k)_3^1 \end{bmatrix} = \begin{bmatrix} v11 & v12 & v13 \\ v21 & v22 & v23 \\ v31 & v32 & v33 \end{bmatrix} \begin{bmatrix} P(k) \\ I(k) \\ D(k) \end{bmatrix} \tag{52}$$

$$\begin{bmatrix} S_1^1(k) \\ S_2^1(k) \\ S_3^1\ (k) \end{bmatrix} = \begin{bmatrix} H(\sum(k)_1^1) \\ H(\sum(k)_2^1) \\ H(\sum(k)_3^1) \end{bmatrix} + \begin{bmatrix} \alpha_1 \times S_1^1(k-1) \\ \alpha_2 \times S_2^1(k-1) \\ \alpha_3 \times S_3^1(k-1) \end{bmatrix} \tag{53}$$

$$H(\Sigma) = \frac{4}{(1+e^{-\Sigma})} - 2 \tag{54}$$

where $\sum(k)_i^1$ and $S_i^1(k)$ are the sum of input connections for each second-layer neuron and the result of the second layer's $i-th$ neuron, and vij and α_i represent the weights.

The third hidden layer consists of three neurons. Each neuron in this layer receives weighted inputs from all outputs of the second hidden layer, in addition to its previous output and a weighted contribution from the previous output of the output layer neuron. The process for computing the output of each third-layer neuron is described in Equations (55) and (56):

$$\begin{bmatrix} \sum(k)_1^2 \\ \sum(k)_2^2 \\ \sum(k)_3^2 \end{bmatrix} = \begin{bmatrix} w11 & w12 & w13 \\ w21 & w22 & w23 \\ w31 & w32 & w33 \end{bmatrix} \begin{bmatrix} S_1^1(k) \\ S_2^1(k) \\ S_3^1\ (k) \end{bmatrix} \tag{55}$$

$$\begin{bmatrix} S_1^2(k) \\ S_2^2(k) \\ S_3^2\ (k) \end{bmatrix} = \begin{bmatrix} \sum(k)_1^2 \\ \sum(k)_2^2 \\ \sum(k)_3^2 \end{bmatrix} + \begin{bmatrix} \beta_1 \times S_1^2(k-1) \\ \beta_2 \times S_2^2(k-1) \\ \beta_3 \times S_3^2\ (k-1) \end{bmatrix} + \begin{bmatrix} \sigma_1 \times \overline{T}(k-1) \\ \sigma_2 \times\ \overline{T}(k-1) \\ \sigma_3 \times \overline{T}(k-1) \end{bmatrix} \tag{56}$$

where $\sum(k)_i^2$ is defined as the sum of input connections for each neuron in the third hidden layer, while $S_i^2(k)$ represents the $i-th$ output of the same layer's neuron. Additionally, the weight parameters wij, β_i, and σ_i play a crucial role in these calculations. Moving forward, the output layer comprises a neuron. This neuron receives inputs from all the outputs of the third hidden layer, each weighted accordingly, as well as its previous output, as demonstrated in Equation (57):

$$\overline{T}(k) = \overline{T}(k-1) + r1 \times S_1^2\ (k) + r2 \times S_2^2\ (k) + r3 \times S_3^2\ (k) \tag{57}$$

where r_i are the parameters of weights. The control signal given to the system is directly translated from the NN's final output. Figure 7 presents the block diagram of the feedback control system with the NN–PID controller.

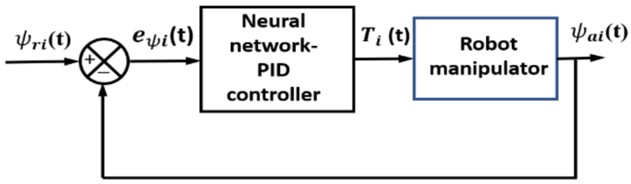

Figure 7. Block diagram of the feedback control system with a neural network–proportional-integral-derivative (PID) controller

4. African vultures optimization algorithm

The AVOA is a modern metaheuristic optimization technique inspired by the scavenging behavior of African vultures.[40] These birds, characterized by their featherless heads and ground-dwelling habits,[41] serve as the basis for the algorithm's conceptual framework. In AVOA, potential solutions are evaluated, and the two optimal candidates are designated as the "first" and "second" vultures. These elite solutions guide the iterative improvement of the population by influencing the search process.[40]

4.1. Step 1: selection of the most suitable vulture

After each iteration, the population is re-evaluated, and the optimal solution is selected using Equation (58):

$$C(i) = \begin{cases} Best\ vulture1, & if\ pi = L1 \\ Best\ vulture2, & if\ pi = L2 \end{cases} \tag{58}$$

where L1 and L2 are parameters within the interval (0,1), specified prior to the search process, and their total sum equals 1.

4.2. Step 2: vulture hunger rate

The hunger rate of the vultures is quantitatively modeled using Equation (59). This parameter reflects the vultures' behavioral response under conditions of starvation or insufficient energy, leading them to exhibit aggressive tendencies or reduced mobility over long distances.[40]

$$Fo = (2 \times rand1 + 1) \times z \times \left(1 - \frac{iterationi}{max\ iterations}\right) + t \tag{59}$$

where $iterationi$ denotes the $i - th$ cycle $max\ iterations$ is the total number of cycles, and F represents vulture satiety (the stopping threshold). Random variables $zrange$ from -1 to 1 whereas $rand1$ ranges from 0 to 1.

4.3. Step 3: exploration

Equations (60)–(63) carry out this task during the exploratory phase.

$$Po(i+1) = \begin{cases} \textbf{Equation 54}, & if\ Po1 \geq randPo1 \\ \textbf{Equation 56}, & if\ Po1 < randPo1 \end{cases} \tag{60}$$

$$Po(i+1) = C(i) - D(i) \times Fo \tag{61}$$

$$D(i) = |X \times C(i) - Po(i)| \tag{62}$$

$$Po(i+1) = C(i) - Fo + rand2 \times ((ub - lb) \times rand3 + lb \tag{63}$$

where Fo signifies the vulture's satiety level, a measure of how full it is, determined using Equation (59); $Po(i+1)$ denotes the vulture's position in the subsequent iteration; all random variables ($randPo1$, $rand2$, and $rand3$) range between 0 and 1; $C(i)$ represents one of the top vultures (e.g., "first" or "second"), which generate a random location X to guard food by multiplying $rand$ by $2(X = 2 \times rand)$, leveraging their current position $Po(i)$; and lb and ub represent the variables' upper and lower bounds.[42, 43]

4.4. Step 4: the initial stage of exploitation

When $|Fo|$ falls between 1 and 0.5, the AVOA is ready to enter the first step of the utilization stage, as presented in Equations (64)-(68):

$$Po(i+1) = \begin{cases} \textbf{Equation } (58) & if\ Po2 \geq randPo2 \\ \textbf{Equation } (60) & if\ Po2\ < randPo2 \end{cases} \tag{64}$$

$$Po(i+1) = D(i) \times (F + rand4) - d(t) \tag{65}$$

$$d(t) = C(i) - Po(i) \tag{66}$$

$$Po(i+1) = C(i) - (S1 + S2) \tag{67}$$

$$\left.\begin{aligned} S1 = C(i) \times ((rand5 + Po(i))/2\pi) \times cos(Po(i)) \\ S2 = C(i) \times ((rand6 + Po(i))/2\pi) \times sin(Po(i)) \end{aligned}\right\} \tag{68}$$

where $rand4$ is a random number between 0 and 1.

When the value $|Fo|$ is less than 0.5, the AVOA moves on to the second stage of the utilization stage,[44] as displayed in Equations (69)–(72):

$$Po(i+1) = \begin{cases} \textbf{Equation } (\textbf{64}) & if\ Po3 \geq randPo3 \\ \textbf{Equation } (\textbf{66}) & if\ Po3\ < randPo3 \end{cases} \tag{69}$$

$$Po(i+1) = \frac{AS1 + AS2}{2} \tag{70}$$

$$\left.\begin{array}{l} AS1 = Best\ vulture1(i) - \frac{BV1(i) \times Po(i)}{BV1(i) - Po(i)} \times Fo \\ AS2 = Best\ vulture2(i) - \frac{BV2(i) \times Po(i)}{BV2(i) - Po(i)} \times Fo \end{array}\right\} \tag{71}$$

$$Po(i+1) = C(i) - |d(t)| \times Fo \times Levy(d) \tag{72}$$

The top vultures in both the first and second groups in this iteration are $BV1(i)$and $BV2(i)$. $AS1$ and $AS2$ represent the position updating formula of vultures. The distance that separates the vulture and one of the most successful vultures in the two groups is represented by $d(t)$. The average Lévy flight is determined by applying Equation (73) to Lévy(d).[44]

$$\begin{aligned} L\acute{e}vy(d) &= 0.01 \times \frac{r2 \times \sigma}{|r1|^{\frac{1}{\beta}}}, \\ \sigma &= \left(\frac{\lceil (1+\beta) \times sin\left(\frac{\pi\beta}{2}\right)}{\lceil (1+\beta 2)\ \ \times \beta \times 2\left(\frac{\beta-1}{2}\right)} \right)^{\frac{1}{\beta}} \end{aligned} \tag{73}$$

where β is a fixed value of 1.5 and $r1$ and $r2$ are random values between 0 and 1.

Figure 8 presents the flowchart for the AVOA. A strong mathematical technique for resolving resource allocation issues in various contexts is stochastic optimization. In order to minimize or maximize the objective function when there is randomness in the optimization process, this technique produces and utilizes random variables.[45]

5. Simulation results

This section presents the tracking performance of the proposed controllers when applied to the nominal model of the 3-LRRM. The control algorithms were implemented in MATLAB (2018b, MathWorks, USA) to address the trajectory tracking problem. The simulation was conducted over a 10-s duration with a time step (h) of 1 ms. The control signals for each link were constrained within the range of -200 to 200 N.m. To ensure accurate tracking of the desired trajectory, the control objective was defined as the minimization of the total ITSE across all three links. Accordingly, a separate trajectory analysis was performed for each link in every test scenario. This objective ensures effective error reduction and facilitates accurate and rapid convergence to the desired trajectories. The classical ITSE performance metric was calculated as defined in Equation (74):

$$\underset{\min}{J} = \int \left(t \times e_{\psi 1}(t)^2 + t \times e_{\psi 2}(t)^2 + t \times e_{\psi 3}(t)^2\right) dt \tag{74}$$

The tracking performance of the 3-LRRM was evaluated by summing the ITSE between the actual and desired trajectories for each link. The AVOA was employed to optimize the parameters of the proposed controllers. The goal of this optimization process was to identify the parameter set that minimizes the total ITSE across all three links. To enhance the robustness of the training process for each proposed controller, two distinct initial position configurations, $(-0.15, -0.85, -1.15)$ and $(0.15, -0.55, -0.85)$ rad for $\psi 1, \psi 2$, and $\psi 3$, respectively, were used in the simulations. The total error from both initial conditions was aggregated to evaluate the fitness of each candidate solution.

The parameters for AVOA were set as follows: a maximum of 1000 iterations and a population size of 100. The performance of each proposed controller was assessed based on its corresponding ITSE value, with the most effective controller being the one that achieves the lowest ITSE.

However, during extensive testing, it was observed that in many cases, the resulting control signals exhibited high-frequency chattering, rendering them impractical for real-world actuators and compromising the reliability of the design. This chattering phenomenon arose from the NN's ability to overfit or adapt excessively to intricate data patterns. To prevent the optimization algorithm from favoring such solutions, a modified performance index was proposed. This new objective function, designed to penalize chattering behavior, is presented in Equation (75):

$$\begin{aligned} \underset{\min}{J} = \int &\left(t \times e_{\psi 1}(t)^2 + t \times e_{\psi 2}(t)^2\right. \\ &\left. + t \times e_{\psi 3}(t)^2\right) dt + Co \times\ \sigma \end{aligned} \tag{75}$$

where σ is a small number chosen as 10^{-8} and Co is the number of times the control signal slope changes its sign. This modified objective function excludes the solution that gives a high chattering control signal from competition between the candidate solutions. The desired trajectories ψ_{r1},ψ_{r2}, and ψ_{r3} for link1, link2, and link3 are provided in Equations (76)–(78), respectively:

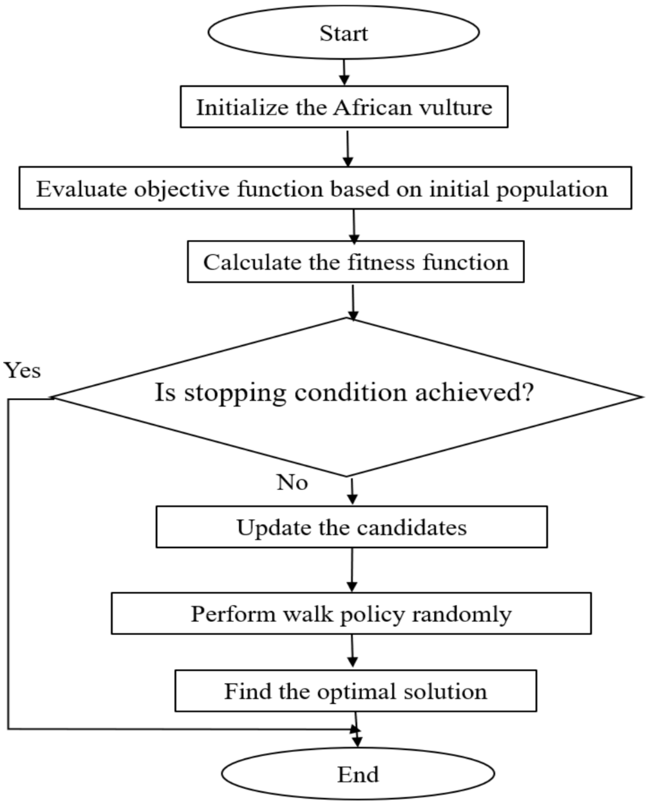

Figure 8. Flowchart of the African vultures optimization algorithm

$$\psi_{r1} = sin(0.2\pi t) \tag{76}$$

$$\psi_{r2} = sin(0.2\pi t - \pi/4\ ) \tag{77}$$

$$\psi_{r3} = sin(0.2\pi t - \pi/2) \tag{78}$$

where the initial conditions are: $\psi_{r1} = 0$, $\psi_{r2} = -0.7$, and $\psi_{r3} = -1$ rad, and $x(0) = 1.0769\,\mathrm{m}, y(0) = -0.4580\,\mathrm{m}$.

All necessary information regarding the nominal model and simulation setup had been established. The next step used the AVOA in conjunction with the nominal model to optimize the parameters of all proposed controllers, aiming to minimize the ITSE. Due to the stochastic nature of AVOA, each controller configuration was simulated 10 times to obtain the optimal results for each proposed controller.

Table 2 summarizes the number of tunable parameters and their corresponding search spaces for each controller. The ITSE values for all recommended controllers, obtained under two initial conditions, are presented in Table 3. Additionally, for the initial condition $(0.15, -0.55, -0.85)$, performance metrics including rise time, settling time, maximum overshoot, and ITSE are provided in Table 4. Figure 9 illustrates the position tracking accuracy, control signal outputs, and the x–y end-effector trajectories for each proposed controller.

The results indicate that the NN–PID controller demonstrates smooth trajectory tracking, rapid convergence to the desired path, and the lowest ITSE compared to the other controllers. Among the proposed controllers, the NN–PID controller exhibits superior performance, achieving the minimum ITSE, as well as the shortest rise and settling times. In contrast, the con-PID controller exhibits the weakest performance across all evaluated metrics.

6. Robust performance

To assess the effectiveness and resilience of each proposed controller, this section evaluates their robustness under varying test conditions without modifying the controllers' parameters.

6.1. Change initial position

To evaluate the controllers' robustness to changes in initial conditions, the joint angles were initialized as follows: Psi-1 = 0.2, Psi-2 = −0.5, and Psi-3 = −0.8 rad. This test examined the performance of the proposed controllers under different initial configurations. The corresponding ITSE values for the trajectory tracking task are presented in Table 5. Figure 10 illustrates the position tracking results for Psi-1, Psi-2, and

Table 2. The number of parameters and their range values across controllers

Controller	Total parameters in controller	Bounds of PID parameters $K_p, K_i,$ and K_d	Corner frequency of derivative filter N	All other parameters range
Con-PID	12	-150 to 150	10 to 100	-
STNN–PID	171	-150 to 150	10 to 100	-1 to 1
NN–PID	102	-150 to 150	-	-1 to 1

Abbreviations: Con-PID: Conventional proportional-integral-derivative control; NN: Neural network; STNN: Self-tuning neural network.

Table 3. Integral time square error (ITSE) and the number of control signals' sign changes across controllers

Controller	ITSE	Total slope sign changes across all control signals
Con-PID	4.67050×10^{-4}	52
STNN–PID	3.35957×10^{-4}	61
NN–PID	0.31764×10^{-4}	169

Note: Under the condition of the nominal plant with two initial positions $(-0.15, -0.85, -1.15)$ and $(0.15, -0.55, -0.85)$ rad.
Abbreviations: Con-PID: Conventional proportional-integral-derivative control; NN: Neural network; STNN: Self-tuning neural network.

Table 4. Detailed features across controllers under a nominal system with initial position set to $(0.15, -0.55, -0.85)$ rad

Controller	Link	Rise time (s)	Settling time (s)	Overshoot (%)	ITSE ($\times 10^{-5}$)
Con-PID	L1	0.153	0.425	5.897	3.84196
	L2	0.163	0.438	6.140	5.21498
	L3	0.184	0.458	5.756	7.34322
STNN–PID	L1	0.049	1.467	40.940	7.26176
	L2	0.050	1.700	25.837	7.86294
	L3	0.188	1.852	4.0386	12.0803
NN–PID	L1	0.032	0.174	23.514	0.42885
	L2	0.034	0.200	28.408	0.59636
	L3	0.039	0.140	17.844	0.50671

Abbreviations: Con-PID: Conventional proportional-integral-derivative control; ITSE: Integral time square error; NN: Neural network; STNN: Self-tuning neural network.

Psi-3, as well as the end-effector trajectory of the 3-LRRM under these altered initial conditions.

Table 5. Integral time square error (ITSE) across controllers under the condition of initial values set to $(0.2, -0.5, -0.8)$ rad

Controller	ITSE ($\times 10^{-4}$)
Con-PID	3.10014
STNN–PID	3.54549
NN–PID	0.28919

Abbreviations: Con-PID: Conventional proportional-integral-derivative control; NN: Neural network; STNN: Self-tuning neural network.

It is noteworthy that the NN–PID controller achieves the smallest ITSE value, along with the shortest rise time, lowest settling time, and fastest overall response. As a result, the end effector closely follows the intended trajectory. Therefore, under variations in the initial positions of Psi-1, Psi-2, and Psi-3, the NN–PID controller demonstrates superior performance compared to other controllers.

6.2. Disturbance addition

To evaluate the disturbance rejection capabilities of the proposed controllers, a sinusoidal disturbance of sin (100t) N.m was applied to the control output of each link between 2 and 6 s. The initial joint positions were set to $(0, -0.7, -1)$ rad. The controller parameters remained unchanged during this test. The corresponding results are summarized in Table 6. Additionally, Figure 11 illustrates the position tracking performance of each joint, along with the trajectory followed by

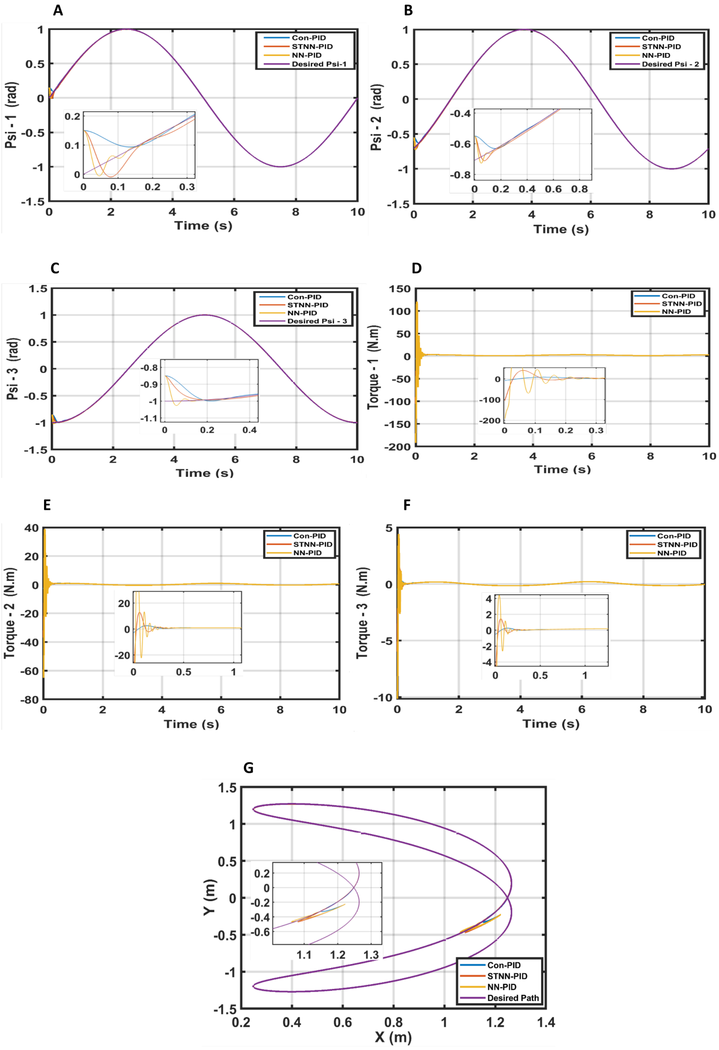

Figure 9. The trajectories tracking of all angles of 3-links robot and the applied control signals when nominal model is used. "A" , "B", and "C" are the positions tracking for Psi-1, Psi-2, and Psi-3, respectively ."D" , "E" , and "F" are the applied control torques, respectively. "G" the end effector x–y plot. Abbreviations: Con-PID, conventional proportional-integral-derivative control; NN, neural network; STNN, self-tuning neural network.

the 3-LRRM end-effector under the influence of the disturbance.

Table 6. Integral time square error (ITSE) across controllers under the condition of sin (100t) as disturbance to every control signal during the period 2–6 s, and an initial value set to (0, −0.7, −1) rad

Controller	ITSE
Con-PID	0.088811
STNN–PID	3.526199
NN–PID	0.064321

Abbreviations: Con-PID, conventional proportional-integral-derivative control; NN, neural network; STNN, self-tuning neural networ.

The results clearly indicate that the NN–PID and con-PID controllers deliver comparable performance. However, the NN–PID controller achieves the lowest ITSE value and provides better overall tracking, as its end-effector trajectory is closest to the desired path. In contrast, the STNN–PID controller demonstrates the weakest performance. Therefore, the NN–PID controller outperforms the other controllers in terms of disturbance rejection.

6.3. Parameter variations

In industrial applications, manipulators are often tasked with object placement and retrieval using end effectors of varying masses. To simulate this scenario, the mass of Link 3 was increased by 10%, while the initial joint positions were set to

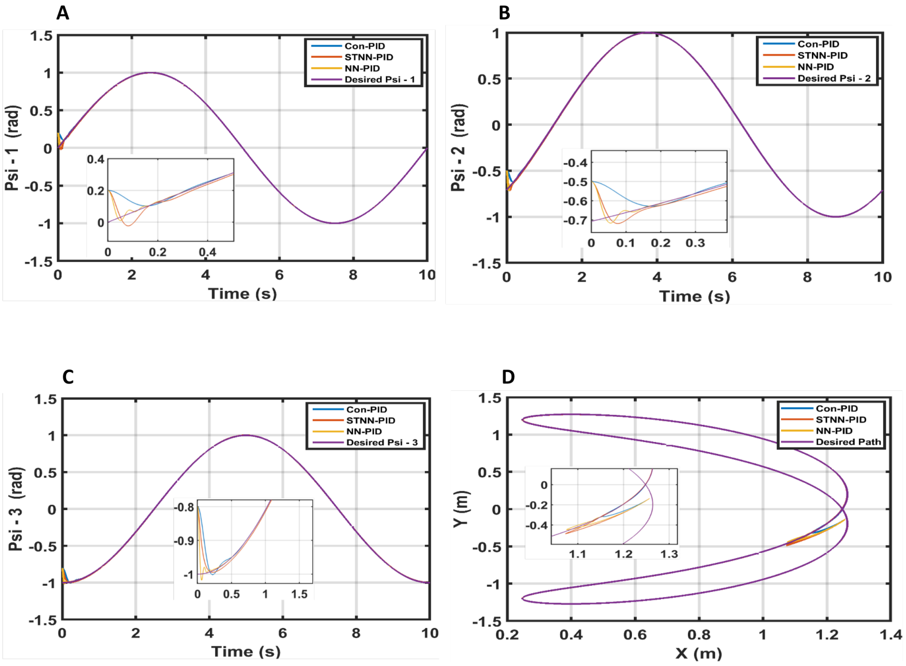

Figure 10. The trajectories tracking of all angles of 3-links robot when initial value set to (0.2, −0.5, −0.8) rad. “A” , “B”, and “C” are the positions tracking for Psi-1, Psi-2, and Psi-3, respectively. “D” is the end effector x–y plot.
Abbreviations: Con-PID: Conventional proportional-integral-derivative control; NN: Neural network; STNN: Self-tuning neural network.

(0.0, −0.7, −1) rad. All other controller and system model parameters remained unchanged. This test evaluates the performance of the proposed controllers under system parameter variations. The resulting ITSE values are reported in Table 7, and the tracking performance of Psi-1, Psi-2, and Psi-3 for each control scheme is illustrated in Figure 12.

Table 7. Integral time square error (ITSE) across controllers under the condition of a 10% increase in M_c and an initial values set to $(0.0, -0.7, -1)$ rad

Controller	ITSE
Con-PID	0.022231
STNN–PID	0.883710
NN–PID	0.001164

Abbreviations: Con-PID, conventional proportional-integral-derivative control; NN, neural network; STNN, self-tuning neural network.

The results show that the NN–PID controller achieves the lowest ITSE among all controllers under parameter variation. Furthermore, it records the fastest response and most accurately tracks the desired trajectories of Psi-1, Psi-2, and Psi-3. In contrast, the STNN–PID controller exhibits the poorest performance, yielding the highest ITSE value.

6.4. Overall efficiency

The overall efficiency and robustness of the proposed controllers were evaluated by simultaneously combining the conditions of altered initial values, external disturbances, and parameter variations. This comprehensive test represents the most critical evaluation scenario, as it encompasses all potential challenges that may affect the control system. The resulting ITSE values are reported in Table 8. Figure 13 illustrates the position tracking of Psi-1, Psi-2, and Psi-3, along with the trajectory followed by the 3-LRRM under the combined test conditions for each controller.

Table 8. Integral time square error (ITSE) across controllers under the condition of a 10% increase in M_c, an initial values set to $(0.2, -0.5, -0.8)$ rad, and disturbance

Controller	ITSE
Con-PID	0.0915603
STNN–PID	2.672754
NN–PID	0.073968

Abbreviations: Con-PID, conventional proportional-integral-derivative control; NN, neural network; STNN, self-tuning neural network.

The results confirm that the NN–PID controller achieves the lowest ITSE value, indicating superior performance, while the STNN–PID

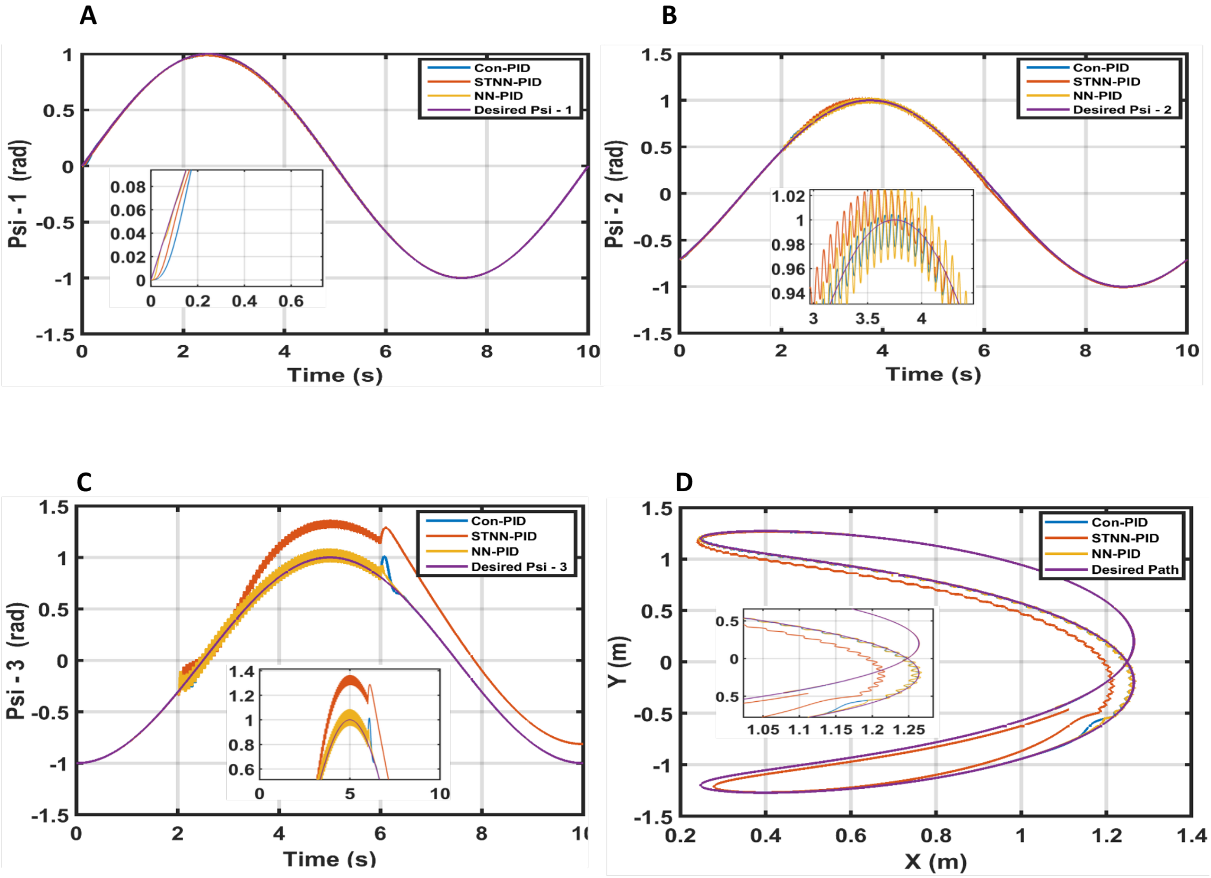

Figure 11. The trajectories tracking of all angles of the 3-links robot when the robot model is under disturbances and start with initial value (0.0, -0.7, -1.0) rad. “A” , “B”, and “C” are the positions tracking for Psi-1, Psi-2, and Psi-3, respectively. “D” is the end effector x–y plot.
Abbreviations: Con-PID: Conventional proportional-integral-derivative control; NN: Neural network; STNN: Self-tuning neural network.

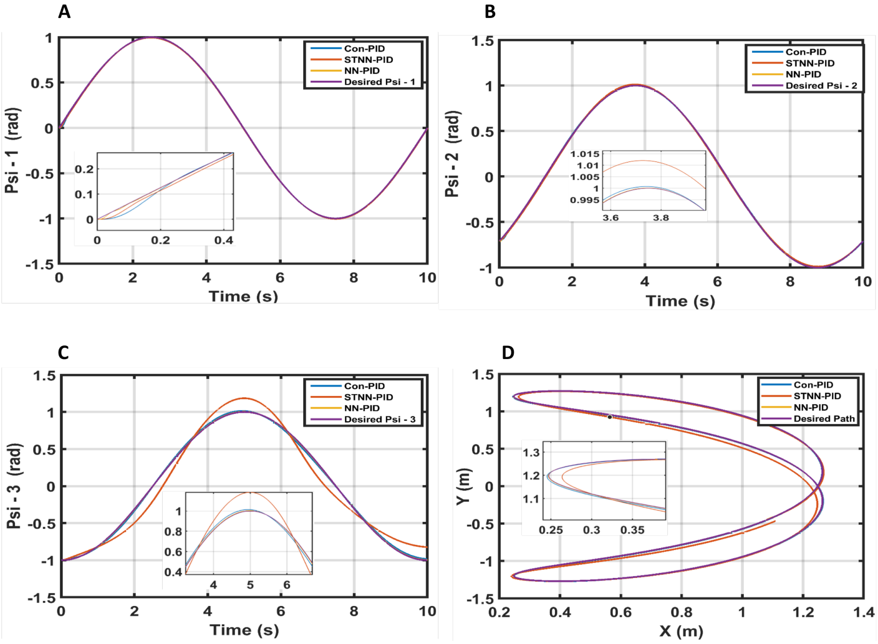

Figure 12. The trajectories tracking of all angles of the 3-links robot when the weight of third link Mc increased by 10% and the motion start with initial value (0.0, -0.7, -1.0) rad. “A” , “B”, and “C” are the positions tracking for Psi-1, Psi-2, and Psi-3, respectively. “D” is the end effector x–y plot.
Abbreviations: Con-PID: Conventional proportional-integral-derivative control; NN: Neural network; STNN: Self-tuning neural network.

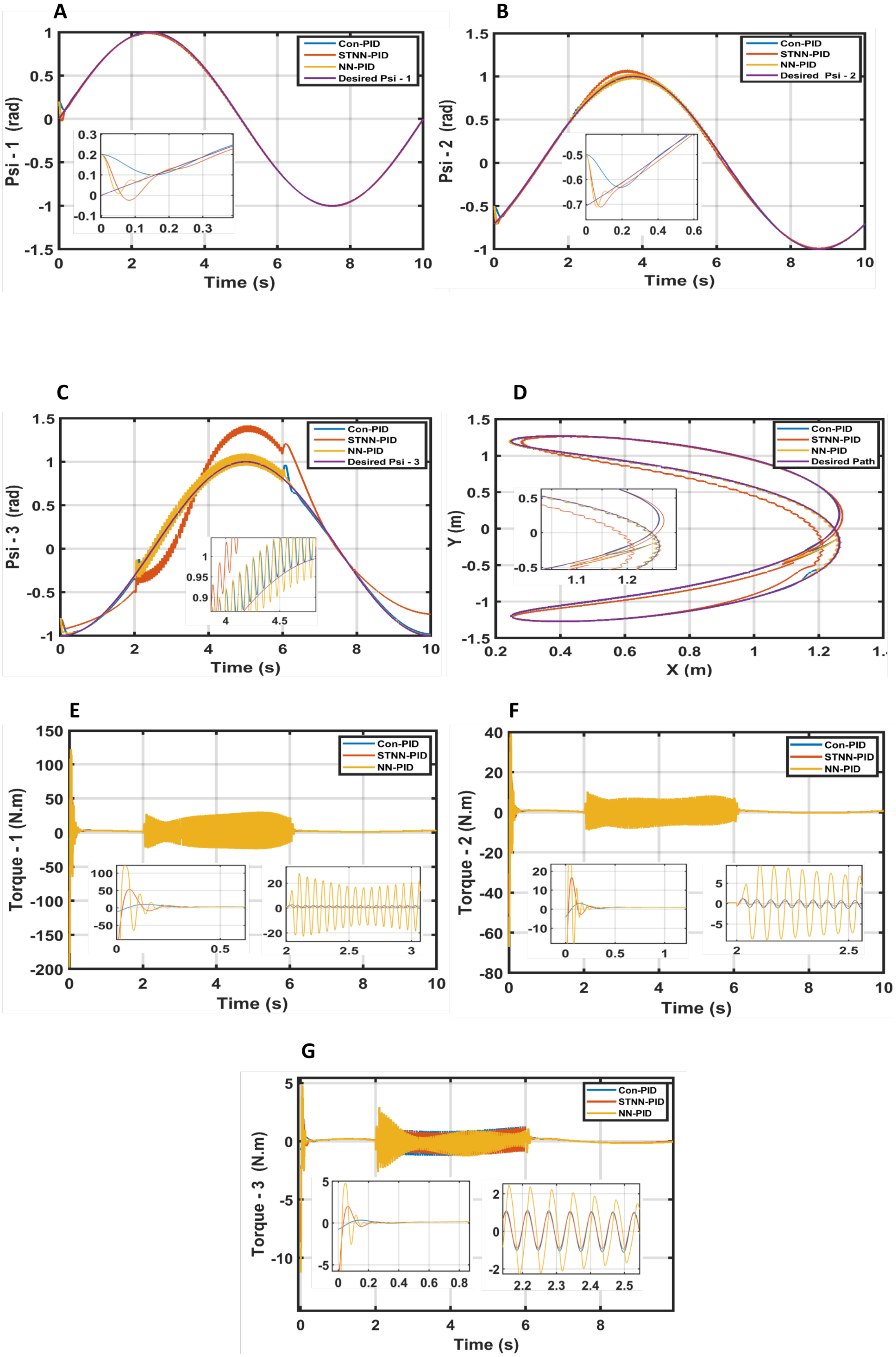

Figure 13. The trajectories tracking of all angles of the 3-links robot and the applied control signals when disturbance disturbed all control signals and the link-3 weight Mc increased by 10% while th motion start from initial values (0.2, 0.5, 0.8).. “A” , “B”, and “C” are the positions tracking for Psi-1, Psi-2, and Psi-3, respectively .”E” , “F, and “G” are the applied control torques respectively. “D” the end effector x–y plot. Abbreviations: Con-PID: Conventional proportional-integral-derivative control; NN: Neural network; STNN: Self-tuning neural network.

controller yields the highest ITSE, reflecting the poorest performance. The figure clearly shows that the actual trajectories of Psi-1, Psi-2, and Psi-3 closely align with the desired trajectories when the NN–PID controller is used.

Table 9 summarizes the performance characteristics of all proposed controllers. Here, overshoot is defined as the maximum deviation from the desired trajectory at any point during the simulation, regardless of disturbances. The data reveal strong competition between the con-PID and NN–PID controllers. However, the NN–PID controller consistently delivers slightly better performance. It exhibits the shortest rise time and settling time, with minimal overshoot, resulting in smooth and rapid convergence to the desired trajectories. On the other hand, the results suggest that the NN–PID controller consumes more energy than the con-PID.

Overall, across all tests and performance metrics, the NN–PID controller outperforms the others, establishing itself as the most effective and reliable control strategy among the proposed controllers.

7. Conclusions and future work

Robotic manipulators are inherently complex, nonlinear, and highly coupled MIMO systems. Their performance is significantly affected by external disturbances and uncertainties in system parameters. As such, control strategies for these systems must be capable of managing this complexity while ensuring robustness and high tracking accuracy. This research proposed two hybrid controllers—STNN–PID and NN–PID—by integrating the strengths of NNs and PID controllers to address the position tracking issue in a 3-LRRM. These hybrid controllers were evaluated against a con-PID controller. The parameters of all controllers were optimized using the AVOA, aiming to minimize the ITSE as the performance index. The ITSE was computed as the sum of tracking errors for all three links. Under nominal conditions, the NN–PID controller outperformed the others, yielding the lowest ITSE value of 0.31764×10^{-4}, compared to 3.35957×10^{-4} for the STNN–PID controller and 4.67050×10^{-4} for the con-PID controller. The NN–PID controller also demonstrated superior dynamic behavior, including minimal rise time, faster settling time, and rapid convergence to the desired trajectories. Robustness evaluations were conducted through individual tests involving variations in initial conditions, the introduction of external disturbances, and parameter uncertainties. In all these cases, the NN–PID controller consistently outperformed the other controllers. Specifically, it achieved ITSE values of 0.28919×10^{-4}, 0.064321, 0.001164 across the respective tests, whereas the STNN–PID controller yielded higher ITSE values of 3.54549×10^{-4}, 3.526199, 0.883710, respectively. The most comprehensive and decisive evaluation involved a combined robustness test, incorporating all perturbations simultaneously. The NN–PID controller demonstrated the highest resilience, achieving the lowest ITSE value of 0.073968, while the STNN–PID controller recorded the weakest performance with an ITSE value of 2.672754. These findings confirm that the NN–PID controller is the most effective controller in terms of tracking accuracy, stability, and robustness across all test scenarios. For future work, the proposed approach could be extended by employing alternative metaheuristic optimization techniques—such as the dragonfly algorithm, artificial bee colony, cuckoo search, or salp swarm algorithm—to optimize controller gains. Additionally, practical implementation and validation using a physical robotic manipulator and the required hardware setup would strengthen the results. Comparative studies with other hybrid schemes, such as the NN–PIPD controller, using the same manipulator model, are also recommended to further evaluate the effectiveness of the proposed strategy.

Acknowledgments

This paper is derived from a research grant funded by the Research, Development, and Innovation Authority (RDIA), Kingdom of Saudi Arabia, with grant number 13382-psu-2023-PSNU-R-3-1-EI-. The authors would like to acknowledge Prince Sultan University, Riyadh, Saudi Arabia, for their support of this publication. This research is supported by the Automated Systems and Computing Lab (ASCL), Prince Sultan University, Riyadh, Saudi Arabia.

Funding

This work is funded by Prince Sultan University, Riyadh, Saudi Arabia.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Table 9. Detailed features across controllers under the condition of a 10% increase in M_c, an initial values set to $(0.2, -0.5, -0.8)$ rad, and disturbance

Controller	Link	Rise time (s)	Settling time (s)	Overshoot (%)	ITSE
Con-PID	L1	0.160	5.360	6.253	0.000762
	L2	0.169	8.053	23..950	0.006914
	L3	0.208	https://doi.org/10.000	60.150	0.083883
STNN–PID	L1	0.051	https://doi.org/10.000	39.320	0.004185
	L2	0.055	9.661	27.969	0.041030
	L3	1.097	https://doi.org/10.000	198.040	2.627539
NN–PID	L1	0.040	5.939	11.895	0.000416
	L2	0.460	6.156	17.696	0.008044
	L3	0.706	https://doi.org/10.000	61.250	0.065506

Abbreviations: Con-PID: Conventional proportional-integral-derivative control;
ITSE: Integral time square error; NN: Neural network; STNN: Self-tuning neural network.

Author contributions

Conceptualization: Bashra Kadhim Oleiwi, Mohamed Jasim Mohamed, Ahmad Taher Azar, Ahmed Redha Mahlous
Formal analysis: Bashra Kadhim Oleiwi, Mohamed Jasim Mohamed, Ahmad Taher Azar, Ahmed Redha Mahlous, and Walid El-Shafai
Investigation: Bashra Kadhim Oleiwi, Mohamed Jasim Mohamed, Ahmad Taher Azar, Ahmed Redha Mahlous, and Walid El-Shafai
Methodology: Bashra Kadhim Oleiwi, Mohamed Jasim Mohamed, Ahmad Taher Azar, Ahmed Redha Mahlous, and Walid El-Shafai
Writing–original draft: Bashra Kadhim Oleiwi
Writing–review & editing: All authors

Availability of data

The datasets generated and/or analyzed during the current study are available from the corresponding author upon reasonable request.

AI tools statement

All authors confirm that no AI tools were used in the preparation of this manuscript.

References

1. Elsisi M, Zaini HG, Mahmoud K, Bergies S, Ghoneim SSM. Improvement of trajectory tracking by robot manipulator based on a new cooperative optimization algorithm. *Mathematics.* 2021;9(24):3231. https://doi.org/10.3390/math9243231
2. Baccouch M, Dodds S. A two-link robot manipulator: simulation and control design. *Int J Robot Eng.* 2020;5(2):1-17. https://doi.org/10.35840/2631-5106/4128
3. Ahmed S, Ghous I, Mumtaz F. TDE based model-free control for rigid robotic manipulators under nonlinear friction. *Sci Iran.* 2024;31(2):137-148. https://doi.org/10.24200/sci.2022.57252.5141
4. Wanigasekara C, Swain A, Almakhles D, Zhou L. Design of delta–sigma-based PID controller for networked wind energy conversion systems. *IEEE Trans Ind Appl.* 2022;58(1):879-889. https://doi.org/10.1109/TIA.2021.3126574
5. Ahmed S, Azar AT, Tounsi M, Anjum Z. Trajectory tracking control of Euler–Lagrange systems using a fractional fixed-time method. *Fractal Fract.* 2023;7(5):355. https://doi.org/10.3390/fractalfract7050355
6. Devaraj SV, Gunasekaran M, Sundaram E, et al. Robust queen bee assisted genetic algorithm (QBGA) optimized fractional order PID (FOPID) controller for not necessarily minimum phase power converters. *IEEE Access.* 2021;9:93331-93337. https://doi.org/10.1109/ACCESS.2021.3091760
7. Mohamed MJ, Oleiwi BK, Abood LH, Azar AT, Hameed IA. Neural fractional order PID controllers design for 2-link rigid robot manipulator. *Fractal Fract.* 2023;7(9):693. https://doi.org/10.3390/fractalfract7090693
8. Saha A, Bhaskar MS, Almakhles D, Elmorshedy MF. Optimization of dual-stage controllers in renewable energy sources-based interconnected power systems through refinement of the African Vultures Optimization Algorithm. *Ain Shams Eng J.* 2024;15(11):103039. https://doi.org/10.1016/j.asej.2024.103039
9. Mohammed A, Hossein M. Robust tracking control of flexible manipulators using hybrid backstepping/nonlinear reduced-order active disturbance rejection control. *ISA Trans.* 2024;149:229-236. https://doi.org/10.1016/j.isatra.2024.04.026
10. Tlijani H, Jouila A, Nouri K. Wavelet neural network sliding mode control of two rigid joint robot manipulator. *Adv Mech Eng.* 2022;14(8):1-13. https://doi.org/10.1177/16878132221119886
11. Bankole AT, Igbonoba EEC. Simulation-based novel hybrid proportional derivative/H-infinity controller design for improved trajectory tracking of a two-link robot arm. *J Shanghai Jiaotong Univ (Sci).* 2023;28:673-687. https://doi.org/10.1007/s12204-023-2660-5

12. Esmail AA, Lee TS, My CA, Mohammed MQ. Robust P-H integrated controller for flexible link manipulator system in the presence of disturbance. *J Appl Comput Mech.* 2021;7(2):646-654. https://doi.org/10.22055/JACM.2020.35475.2668
13. Villa-Tiburcio JF, Estrada-Torres JA, Hernández-Alvarado R, Montes-Martínez JR, Bringas-Posadas D, Franco-Urquiza EA. ANN enhanced hybrid force/position controller of robot manipulators for fiber placement. *Robotics.* 2024;13(7):105. https://doi.org/10.3390/robotics13070105
14. Bankole AT, Igbonoba EEC. A novel hybrid proportional derivative/H-infinity controller design for improved trajectory tracking of a two-link robot arm. *Res Sq.* 2022. https://doi.org/10.21203/rs.3.rs-2193905/v1
15. Fani D, Shahraki E. Two-link robot manipulator using fractional order PID controllers optimized by evolutionary algorithms. *Biosci Biotech Res Asia.* 2016;13(1):389-396. https://doi.org/10.13005/bbra/2075
16. Anwaar H, Yixin Y, Ijaz S, Ashraf MA, Anwaar W. Fractional order based computed torque control of 2-link robotic arm. *Adv Sci Technol Res J.* 2018;12(1):273-284. https://doi.org/10.12913/22998624/85658
17. Khalil I, Sharkawy AB. A hybrid PID control scheme for flexible joint manipulators and a comparison with sliding mode control. *Ain Shams Eng J.* 2018;9(4):3451-3457. https://doi.org/10.1016/j.asej.2018.01.004
18. Vineet K, Rana KPS, Kler D. Efficient control of a 3-link planar rigid manipulator using self-regulated fractional-order fuzzy PID controller. *Appl Soft Comput.* 2019;82:105531. https://doi.org/10.1016/j.asoc.2019.105531
19. Ahmad SG, El-Gohary MA, Elksas MS, Areed FG. Three link rigid manipulator control using improved neural network based PID controller . *Int J Neural Netw Adv Appl.* 2019;6:60-67.
20. Kumar J, Kumar V, Rana KPS. Fractional-order self-tuned fuzzy PID controller for three-link robotic manipulator system. *Neural Comput Appl.* 2020;32(11):7235-7257. https://doi.org/10.1007/s00521-019-04215-8
21. Tohma DH, Hamoudi AK. Design of adaptive sliding mode controller for uncertain pendulum system. *Eng Technol J.* 2021;39(3):355-369. https://doi.org/10.30684/etj.v39i3A.1546
22. Zhu D, Du B, Zhu P, Chen S. Constant force PID control for robotic manipulator based on fuzzy neural network algorithm. *Complexity.* 2020;2020:3491845. https://doi.org/10.1155/2020/3491845
23. Sathish Kumar A, Naveen S, Vijayakumar R, et al. An intelligent fuzzy-particle swarm optimization supervisory-based control of robot manipulator for industrial welding applications. *Sci Rep.* 2023;13:8253. https://doi.org/10.1038/s41598-023-35189-2
24. Gümüş MS, Çakan A, Kalyoncu M. Cascade proportional derivative controller for a flexible link robot manipulator using the Bees Algorithm. *APJESS.* 2023;11(1):27-34. https://doi.org/10.21541/apjess.1084875
25. Nohooji HR. Constrained neural adaptive PID control for robot manipulators. *J Frank Inst.* 2020;357(7):3907-3923. https://doi.org/10.1016/j.jfranklin.2019.12.042
26. Chuyen TD, Doan HV, Minh PV, Thong VV. Design of robust adaptive controller for industrial robot based on sliding mode control and neural network. *Int J Mech Eng Robot Res.* 2023;12(3):183-189. https://doi.org/10.18178/ijmerr.12.3.145-150
27. Mohamed MJ, Oleiwi BK, Azar AT, Mahlous AR. Hybrid controller with neural network PID/FOPID operations for two-link rigid robot manipulator based on the zebra optimization algorithm. *Front Robot AI.* 2024;11:1386968. https://doi.org/10.3389/frobt.2024.1386968
28. Chotikunnan P, Chotikunnan R. Dual design PID controller for robotic manipulator application. *J Robot Control.* 2023;4(1):23-34. https://doi.org/10.18196/jrc.v4i1.16990
29. Mary AH, Al-Talabi A, Kara T, Muneam DS, Almuhanna MY, Mayyahi LAK. Adaptive robust tracking control of robotic manipulator based on SMC and fuzzy control strategy. *Al-Khwarizmi Eng J.* 2024;20(1):63-75. https://doi.org/10.22153/kej.2024.11.002
30. Coutinho AG, Hess-Coelho TA. Improving the performance of parallel robots by applying distinct hybrid control techniques. *Robotica.* 2022;40(4):951-975. https://doi.org/10.1017/S0263574721000874
31. Yahya RJ, Abbas NH. Adaptive optimal trajectory tracking control for four flexible joint robot manipulator. *Int J Intell Eng Syst.* 2022;15(2):1-12. https://doi.org/10.22266/ijies2022.0430.01
32. Çetinkaya MB, Yildirim K, Yildirim Ş. Trajectory analysis of 6-DOF industrial robot manipulators by using artificial neural networks. *Sensors (Basel).* 2024;24(13):4416. https://doi.org/10.3390/s24134416
33. Al-Khayyt S. Tuning PID controller by neural network for robot manipulator trajectory tracking. *Al-Khwarizmi Eng J.* 2017;9(1):19-28.
34. Adnan A, Karam HE. Optimal improved PID controller with GOA algorithm for single link human leg robot. *J Eng Sustain Dev.* 2022;26(2):103-110. https://doi.org/10.31272/jeasd.26.2.10
35. Yadav S, Kumar S, Goyal M. PID tuning and stability analysis of hybrid controller for robotic arm using ZN, PSO, ACO, and GA. *Int Rev Mech Eng.* 2022;16(5):257-264. https://doi.org/10.15866/ireme.v16i5.21982
36. Abdulameer HI, Mohamed MJ. Fractional order fuzzy PID controller design for 2-link

rigid robot manipulator. *Int J Intell Eng Syst.* 2022;15(3):103-117.
https://doi.org/10.22266/ijies2022.0630.10

37. Lutfey OF. A wavelet functional link neural network controller trained by a modified sine cosine algorithm using the feedback error learning strategy. *J Eng Sci Technol.* 2020;15(1):709-727.
38. Raafat SM, Raheem FA. Introduction to robotics-mathematical issues. In: *Mathematical Advances Towards Sustainable Environmental Systems.* Cham: Springer; 2017:261-289. https://doi.org/10.1007/978-3-319-43901-$3_1$2
39. Mohamed MJ, Oleiwi BK, Azar AT, Hameed IA. Coot optimization algorithm-tuned neural network-enhanced PID controllers for robust trajectory tracking of three-link rigid robot manipulator. *Heliyon.* 2024;10(13):e32661. https://doi.org/10.1016/j.heliyon.2024.e32661
40. Abdollahzadeh B, Gharehchopogh FS, Mirjalili S. African vultures optimization algorithm: a new nature-inspired metaheuristic algorithm for global optimization problems. *Comput Ind Eng.* 2021;158:107408. https://doi.org/10.1016/j.cie.2021.107408
41. Abed MS, Lutfy OF, Al-Doori QF. Online path planning of mobile robots based on African vultures optimization algorithm in unknown environments. *J Eur Syst Autom.* 2022;55(3):405-412. https://doi.org/10.18280/jesa.550313
42. Rajinikanth V, Razmjooy N. A comprehensive survey of meta-heuristic algorithms. In: *Metaheuristics and Optimization in Computer and Electrical Engineering.* Lecture Notes in Electrical Engineering. Cham: Springer; 2023:1-29. https://doi.org/10.1007/978-3-031-42685-8_1
43. Liu Q, Kosarirad H, Meisami S, Alnowibet KA, Hoshyar AN. An optimal scheduling method in IoT-fog-cloud network using combination of Aquila optimizer and African vultures optimization. *Processes.* 2023;11(4):1162. https://doi.org/10.3390/pr11041162
44. Maashi M, Al-Hagery MA, Rizwanullah M, et al. Automated gesture recognition using African vulture optimization with deep learning for visually impaired people on sensory modality data. *JDR.* 2023;2(2):37-48. https://doi.org/10.57197/JDR-2023-001
45. Maroufpoor S, Azadnia R, Bozorg-Haddad O. Stochastic optimization: stochastic diffusion search algorithm. In: *Handbook of Probabilistic Models.* Butterworth -Heinemann; 2020:437-448. https://doi.org/10.1016/B978-0-12-816514-0.00017-5

Bashra Kadhim Oleiwi *completed a Master's degree in Department of Mechatronics Engineering/College of Control and Systems Engineering at the University of Technology (UOT), Baghdad, Iraq. She finished her Ph.D. at Department of Control Engineering (RST), Siegen University, Germany. Her research interests are robotics, path planning, control, multi-objective optimization, artificial intelligence systems, deep learning, machine learning, optimization techniques, IoT, image processing, control systems, embedded systems, FPGA, microcontrollers, and microprocessors. She is a faculty member in Department of UOT's Control and Systems Engineering. She is currently the head of Mechatronics and Robotics Engineering Department. She was the head of Automation and Robotics unit. She was a manager of IT and Internet unit. She published nearly 53 papers in scientific journals and conference proceedings. She is the reviewer for 1894 papers. She is associate editor in local and international journals. She can be contacted at email: bushra.k.oleiwi@uotechnology.edu.iq.*
`https://orcid.org/0000-0001-5972-0625`

Mohamed Jasim Mohamed *received a B.Sc. degree in Control and Systems Engineering in 1984 from the University of Technology, College of Control and Systems Engineering, Baghdad, Iraq. From 1984 to 2000, he worked at the University of Technology in computer laboratories as a programmer of different programming languages and a maintenance engineer for computers and equipment. In 1992, he became the manager of various computer labs. In 2001, he received an M.Sc. degree in Control and Measurements from the University of Technology, Iraq, and he joined the faculty of control and systems engineering at this University. In 2008, he received his Ph.D. in Control System Engineering from the same University. He became an assistant professor in 2013 and an editor of many journals. He published nearly 40 papers in scientific journals and conference proceedings. He can be contacted at email: Mohamed.J.Mohamed@uotechnology.edu.iq.*
`https://orcid.org/0000-0002-7078-1290`

Ahmad Taher Azar *(Senior Member, IEEE) is a Full Professor with the College of Computer and Information Sciences (CCIS), Prince Sultan University, Riyadh, Saudi Arabia. He is a Leader of the Automated Systems and Soft Computing Laboratory (ASSCL), Prince Sultan University. He has expertise in artificial intelligence, control theory and applications, robotics, machine learning, computational intelligence, and dynamical system modeling. He has authored/co-authored over 500 research articles in prestigious peer-reviewed journals, book chapters, and conference proceedings. He is currently an Editor of IEEE Systems Journal, IEEE Transactions on Neural Networks*

and Learning Systems, Human-Centric Computing and Information Sciences (Springer), and Engineering Applications of Artificial Intelligence (Elsevier).
https://orcid.org/0000-0002-7869-6373

Saim Ahmed *received the B.Sc. degree in electronics from the Sir Syed University of Science and Technology, Pakistan, in 2009, the M.E. degree in industrial control and automation from Hamdard University, Pakistan, in 2013, and the Ph.D. degree in control science and engineering from Nanjing University of Science and Technology, China, in 2019. He is currently a Postdoctoral Researcher with the Department of Computer Science, Prince Sultan University, Riyadh, Saudi Arabia. His research interests include the theory and applications of adaptive control, sliding mode control, time delay control, robotic manipulators, and nonlinearities and their compensation.*
https://orcid.org/0000-0002-2302-705X

Ahmed Redha Mahlous *is an Associate Professor in the Department of Computer Science at Prince Sultan University (KSA). He earned a master's degree (MSc) from South Bank University (London, UK) and a PhD degree from the University of Bradford (UK). He worked as an IT consultant for a number of UK-based companies, and worked for several years as an academic instructor and as a Pearson VUE Exam Center Director at the Regional Cisco Academy (KSA). He holds multiple commercial certifications from Microsoft, Cisco, and Palo Alto, among others. His research interests include Cybersecurity, Networking, QoS, and Data mining. He is a reviewer for some prominent journals and has multiple scientific publications.*
https://orcid.org/0000-0001-7745-3428

Walid El-Shafai *(Senior Member, IEEE) was born in Alexandria, Egypt. He received the B.Sc. degree (Hons.) in electronics and electrical communication engineering from the Faculty of Electronic Engineering (FEE), Menoufia University, Menouf, Egypt, in 2008, the M.Sc. degree from Egypt-Japan University of Science and Technology (E-JUST), in 2012, and the Ph.D. degree from FEE, Menoufia University, in 2019. From January 2021 to December 2024, he was a Researcher with the Security Engineering Laboratory (SEL), Prince Sultan University (PSU), Riyadh, Saudi Arabia. He is currently a Senior Researcher with the Automated Systems and Computing Laboratory (ASCL) and an Assistant Professor with the College of Computer Science and Information Systems, PSU. In addition, he is an Associate Professor with the Department of Electronics and Communication Engineering (ECE), Faculty of Electronic Engineering, Menoufia University. His research interests include wireless mobile and multimedia communication systems, image and video signal processing, efficient 2D/3D video coding and transmission, quality of service and experience, digital communication techniques, cognitive radio networks, adaptive filter design, 3D video watermarking, steganography, encryption, error resilience and concealment algorithms for video codecs (H.264/AVC, H.264/MVC, and H.265/HEVC), cognitive cryptography, medical image processing, speech processing, security algorithms, softwaredefined networks, the Internet of Things, FPGA implementations of signal processing algorithms and communication systems, cancellable biometrics, pattern recognition, image and video magnification, artificial intelligence applications in signal processing and communication systems, modulation identification and classification, image and video super-resolution and denoising, automated systems, cybersecurity applications, malware and ransomware detection, and DL for signal processing and communication systems. He is a dedicated reviewer for numerous international journals and conferences, contributing to the advancement of research in his areas of expertise.*
https://orcid.org/0000-0001-7509-2120