Journal of the Brazilian Society of Mechanical Sciences and Engineering (2023) 45:546
https://doi.org/10.1007/s40430-023-04466-6

TECHNICAL PAPER

Enhancing robustness and noise rejection in flexible joint manipulators: an optimized sliding mode controller with enhanced gray wolf optimization for trajectory tracking

Muhammad I. Azeez[1] · S. Elnaggar[1] · A. M. M. Abdelhaleem[1] · Kamal A. F. Moustafa[2] · Khaled R. Atia[1]

Received: 12 February 2023 / Accepted: 5 September 2023 / Published online: 26 September 2023

Abstract
The objective of this study is to enhance the performance of a nonlinear three-rigid-link manipulator (RLM) with a focus on trajectory tracking, robustness against disturbances and noises, and adaptability to joint flexibility. To achieve this, we have employed an optimized sliding mode controller with a proportional integral derivative (PID) sliding manifold. The tuning process involves selecting the critical gains of the controller that minimizes the integral time absolute error (ITAE), serving as the objective function (OBJF) to optimize the performance of the robot manipulator. To identify the optimal gains of the controller, we have utilized a new optimization algorithm known as memory enhanced linear population size reduction gray wolf optimization (MELGWO). The efficacy of this algorithm is compared to other existing optimization methods in the literature. Moreover, this research has delved into the impact of joint flexibility on the robot system's performance. Encouragingly, the results demonstrate that the optimized SMC–PID with MELGWO adaptation can effectively address joint flexibility while maintaining acceptable performance levels.

Keywords Enhanced gray wolf optimizer · Sliding mode control · Proportional integral derivative · Trajectory tracking · Flexible joint manipulator · Rigid link manipulator

1 Introduction

Robotic manipulators are employed in the current industrial environment to enhance product quality as well as productivity, accuracy, speed, and flexibility in the workplace. Robotic manipulators are rapidly being used in risky, tedious, or repetitive industrial processes, despite the complicated and tough requirements of industrial applications. Material picking and placing is one of the essential uses of mechanical robotic manipulators. These manipulators can also be used for welding, assembling, manufacturing, painting, and other operations in the car industry, as well as for handling radioactive and biohazardous materials during robotically assisted surgery. Manufacturing needs to be more effective and high-quality, and this requirement has motivated the development of more robust and sophisticated robotic manipulators [1–4].

The adoption of flexible link manipulators (FLMs) and flexible joint manipulators (FJMs) have witnessed a notable surge in recent times, attributed to their extensive applicability across domains encompassing industry, medicine, aerospace, instrumentation, satellites, and industrial automation. These manipulators are progressively replacing rigid links characterized by larger dimensions and greater mass within diverse industrial sectors [5]. Furthermore, the design of flexible and lightweight manipulators offers advantages such as enhanced maneuverability and reduced power consumption. Nonetheless, these systems pose challenges in terms of modeling, measurement, stabilization, and control, primarily attributed to the oscillations exhibited by flexible links [6].

The importance of mathematical modeling for multi-DOF manipulators is necessary in any practical control process; the mathematical model of the manipulator is normally developed using the Lagrangian approach [7–10]. In general, mathematical modeling is really challenging and time-consuming, especially for strongly coupled

Technical Editor: Adriano Almeida Gonçalves Siqueira.

✉ Muhammad I. Azeez
mohibrahiem@eng.zu.edu.eg; moazezian97@gmail.com

[1] Mechanical Design and Production Engineering Department, Zagazig University, Zagazig 44519, Egypt

[2] Industrial Engineering Department, Zagazig University, Zagazig 44519, Egypt

multi-input multi-output systems; Recent research offers a thorough and comprehensive critical analysis of mathematical modeling methods [2]. Using a numerical modeling method, the powerful MATLAB Simulink and Simscape toolboxes were utilized to create SimMechanics models of robotic manipulators from CAD assemblies [1, 9, 10]. Simscape multibody models are favored because they are simple to get, and they can be simulated and controlled [1, 5, 11–14]. Using simulation software has several advantages, including understanding of how the multi-DOF robotic manipulator systems work in a simulated environment and avoiding the complexity of mathematical formulations.

Sliding mode control (SMC) is a widely utilized nonlinear control strategy employed to guide state trajectories toward predetermined sliding surfaces through the application of discontinuous control inputs [10, 15–19]. Nevertheless, the SMC law's discontinuity can lead to the manifestation of the chattering effect, characterized by the occurrence of high-frequency oscillations in the controlled variable. This phenomenon has the potential to disrupt the stability of the controlled system or substantially reduce the actuators' operational lifespan [10, 15].

Numerous strategies have been proposed in the literature to address the chattering phenomenon [20]. One of the approaches involves approximating the SMC's discontinuous control with a continuous counterpart. Although this technique effectively reduces tracking error, it comes at the cost of compromising the overall efficiency of SMC by generating a pseudo-sliding mode instead of the desired ideal sliding mode [20]. An alternative solution to mitigate chattering entails employing the higher-order sliding mode control approach. This method confines the discontinuity to a derivative of the control variable, incorporating these derivatives with respect to time up to a specified order in addition to the sliding variable. Notably, this approach exhibits favorable suitability for application in electromagnetic or mechanical systems due to its continuous nature of control action [20].

The existing literature encompasses various studies exploring the sliding mode controller and its diverse applications. For instance, a comparative investigation conducted by prior researchers [21] evaluated the performance of the SMC controller in comparison to the PID controller. The outcomes indicated that the SMC exhibited a faster and more robust response, albeit with a relatively larger control signal than the PID controller.

Geng et al. [22] explored three distinct types of non-singular terminal sliding mode controllers implemented on a robotic arm subject to external disturbances. Through simulation results, they demonstrated the efficacy of the proposed modifications in enhancing the controller's convergence rate while reducing the magnitude of the control input signal.

In a study conducted by Mohammad-Reza Moghanni-Bavil-Olyaei et al. [23], a comparative analysis was performed between the sliding mode control (SMC) with proportional-derivative (PD) surface and the block backstepping control approach. The aim of this comparison was to evaluate the trajectory tracking performance of a single-link flexible-link flexible-joint manipulator (SFLFJM).

An additional research endeavor focused on an integrated second-order SMC controller and the application of an algorithm for designing the control scheme for manipulator robots. This study aimed to evaluate the practicability of the proposed controller and confirm its effectiveness in terms of convergence and robustness. Satisfactory results were obtained through experiments conducted on a real industrial robot, validating the algorithm's practicality and effectiveness [10, 20].

Numerous studies have investigated the concept of adaptive sliding mode control [24–26]. Additionally, there have been several studies that explore the integration of fuzzy logic with sliding mode control (SMC) [27–29]. To enhance the performance of SMC control, Genetic Algorithms (GA) were utilized to determine optimal parameters for a conventional SMC controller [30, 31]. Vijay and Jena devised a Particle Swarm Optimization (PSO)-based backstepping sliding mode controller and observer for robot manipulators [32]. PSO was employed to tune sliding surface parameters of SMC combined with an Artificial Neuro Fuzzy Inference System (ANFIS) and adjust the membership functions of Takagi Sugeno fuzzy SMC, respectively [33, 34]. A novel approach in [35] introduced the combination of a PSO algorithm with super-twisting SMC to eliminate chattering. Medjghou et al. introduced an optimized extended Kalman filter with integral type 2 fuzzy SMC using Biogeography-based optimization (BBO) [36]. Oliveira et al. proposed using gray wolf optimization (GWO) with a chaotic basis to tune robust higher-order SMC for the position control of a robot manipulator [37]. To find the best control parameters for the fuzzy controller in APIDFSMC and control uncertain systems, the GA-based adaptive PID fuzzy sliding mode control strategy (APIDFSMC-GA) is introduced [38]. With a proportional-integral-derivative-type nonsingular fast terminal sliding mode control, an adaptive robust controller for uncertain nonlinear systems is designed to provide the essential attributes of rapid transient response, finite-time convergence, negligible steady-state error, and chattering cancellation [39]. Different optimization techniques, such as the Antlion Optimization Algorithm (ALO), Sine Cosine Algorithm (SCA), gray wolf optimizer (GWO), and Whale Optimizer Algorithm (WOA), are used to control the nonlinear 2-DOF system in a sliding mode controller (SMC) with PID surface [15].

In the context of scientific research and technical applications, the use of evolutionary algorithms (EAs), a type of efficient and effective optimization approach, has recently grown in popularity. Modern practical optimization issues are usually non-convex, discontinuous, or even non-differentiable, which makes it difficult for traditional optimization techniques, like gradient-based approaches, to handle these types of problems [40–42].

Natural processes are used by swarm-based optimization algorithms (SOAs) to drive a search toward the optimal solution. Instead of using just one solution for each iteration, SOAs use a population of solutions, in contrast to simple optimization algorithms like stochastic process and hill climbing. The difference between SOAs and these algorithms in this regard is substantial. A population of solutions is processed at each iteration, and a population of solutions is also created each iteration [41, 42].

The gray wolf optimizer (GWO) is a new developed metaheuristic algorithm inspired by nature, characterized by a small number of parameters for tuning [40]. However, the original GWO and its variants may encounter challenges such as lack of population diversity, premature convergence, and difficulty in achieving a balance between exploration and exploitation. To address these limitations, a new variant called Memory Enhanced Linear Population Size Reduction GWO (MELGWO) is proposed [41]. This variant incorporates memory to retain better solutions, evolutionary operators to enhance exploration, and a stochastic local search technique. Additionally, a linear population size reduction strategy is employed to maintain a diverse population and prevent premature convergence. As iterations progress, the swarm size is gradually reduced to minimize redundant exploration. The local search component significantly improves the exploitation capability of GWO, resulting in the generation of more wolves near the best solutions in the memory swarm [41]. Through the integration of these features, the MELGWO algorithm demonstrates superior performance compared to the original GWO algorithm, effectively addressing the limitations and achieving enhanced optimization results.

This study encompasses the utilization of the Lagrangian method and the Simscape Multibody toolbox within MATLAB to construct a comprehensive model of a 3-DOF RLM. To validate the fidelity of this model, the investigation commences with an analysis of open-loop system responses for both the mathematical and Simscape representations, elucidated in Sect. 2. In Sect. 3, the formulation of a Sliding Mode Controller (SMC) with a PID sliding surface is elaborated. Section 4 expounds on the employment of the memory enhanced linear population size reduction gray wolf optimizer (MELGWO) metaheuristic optimization technique, designed to confer overall robustness, stability, and exceptional tracking performance to the closed-loop three-link maneuver system. A meticulous examination of the proposed controller's efficacy in mitigating disturbances and noise, along with its resilience against variations in payload mass, is undertaken in Sect. 5. Furthermore, the adaptability of the controller to the maneuver's joint flexibility is discussed. Finally, Sect. 6 encapsulates the study's key findings and concluding remarks.

In the entire study, all vectors are written with bold lowercase letters and all matrices are written with uppercase letters.

2 Modeling the three DOF rigid-link robotic manipulator

The model of the 3-DOF rigid-link manipulator derived from the Lagrangian method depends on the kinetic and potential energies of the links. The more complex the problem, the more variables, assumptions, and iterations we might need to make, and this takes long derivation time. On the other side, the powerful tool Simscape Multibody that is fully integrated with MATLAB and Simulink, enables rapid modeling, simulation, and analysis of systems.

The Simscape modeling and validation of the Lagrange's model with Simscape model are included in the following subsections.

2.1 Lagrange mathematical modeling

By considering the kinetic and potential properties of the mechanical system, the Lagrange's equations of motion can be obtained. Figure 1 illustrates a system comprising three linkages and three joints, denoted as J_1, J_2, and J_3. The lengths of the links are represented as l_1, l_2, and l_3, while their masses are indicated by m_1, m_2, and m_3, respectively. A comprehensive listing of these properties is provided in Table 1.

In the context of the manipulator, the dynamic model can be expressed as follows:

$$B(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{f}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right) + \mathbf{g}(\boldsymbol{\theta}) + \boldsymbol{D} = \boldsymbol{\tau} \tag{1}$$

where $\boldsymbol{\theta}(t), \dot{\boldsymbol{\theta}}(t), \ddot{\boldsymbol{\theta}}(t)$ denote the vectors of joints angles, joints velocities and joints accelerations, $B(\boldsymbol{\theta})$ is the inertia matrix, $\mathbf{f}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right)$ is the vector of centrifugal and Coriolis forces, $\mathbf{g}\,(\boldsymbol{\theta})$ is the gravitational force vector, $\boldsymbol{\tau}$ is the vector of input torques and $\boldsymbol{D}$ is the vector of external disturbances which is assumed to be bounded with $\|\boldsymbol{D}\| \le d$, where d is a positive constant.

Using the information given in Table 1, the entries of the inertia matrix are as follows:

$$\begin{aligned} B_{11} = {} & 0.1306 + 0.12m_L + 0.04\left(1 + 2m_L\right)\cos\left(\theta_2 + \theta_3\right) \\ & + 0.08\left(1.125 + m_L\right)\cos\left(\theta_2\right) + 0.04\left(1 + 2m_L\right)\cos\left(\theta_3\right) \end{aligned} \tag{2}$$

$$\begin{aligned} B_{12} = {} & 0.0571 + 0.08m_L + 0.02\left(1 + 2m_L\right)\cos\left(\theta_2 + \theta_3\right) \\ & + 0.04\left(1.125 + m_L\right)\cos\left(\theta_2\right) + 0.04\left(1 + 2m_L\right)\cos\left(\theta_3\right) \end{aligned} \tag{3}$$

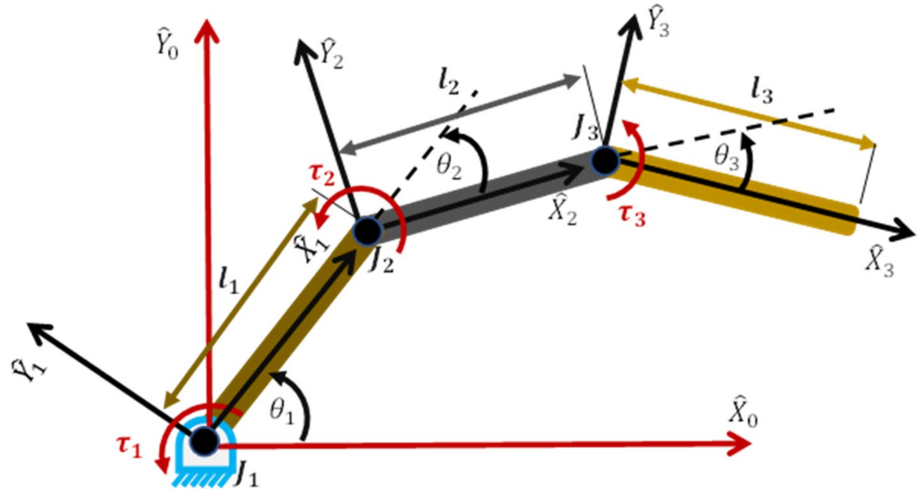


Fig. 1 Representation of the three-link robotic model

Table 1 The physical parameters of the three-link robotic manipulator

Parameters	Symbol	Description	Value
All links	m_i	Mass (kg)	1.0
	l_i	Length (m)	0.2
	w_i	Width (m)	0.02
	t_i	Thickness (m)	0.02
	I_i	Moment of Inertia (kgm^2)	0.0033

$$B_{13} = D_{31} = 0.0135 + 0.04m_L + 0.02(1 + 2m_L)\cos(\theta_2 + \theta_3) + 0.02(1 + 2m_L)\cos(\theta_3) \tag{4}$$

$$B_{22} = 0.0135 + 0.04m_L + 0.02(1 + 2m_L)\cos(\theta_3) \tag{5}$$

$$B_{23} = B_{32} = 0.0571 + 0.08m_L + 0.04(1 + 2m_L)\cos(\theta_3) \tag{6}$$

$$B_{33} = 0.0135 + 0.04m_L \tag{7}$$

where m_L is the payload mass.

The elements of the centrifugal and Coriolis force vector are defined by:

$$\begin{aligned} f_1 = & -[0.04(1.125 + m_L)\sin(\theta_2) + 0.02(1 + 2m_L)\sin(\theta_2 + \theta_3)]\dot{\theta}_2^2 \\ & - [0.02(1 + 2m_L)\sin(\theta_3) + 0.02(1 + 2m_L)\sin(\theta_2 + \theta_3)]\dot{\theta}_3^2 \\ & - [0.08(0.125 + m_L)\sin(\theta_2) + 0.04(1 + 2m_L)\sin(\theta_2 + \theta_3)]\dot{\theta}_1\dot{\theta}_2 \\ & - [0.04(1 + 2m_L)\sin(\theta_3) + 0.04(1 + 2m_L)\sin(\theta_2 + \theta_3)]\dot{\theta}_2\dot{\theta}_3 \\ & - [0.04(1 + 2m_L)\sin(\theta_3) + 0.04(1 + 2m_L)\sin(\theta_2 + \theta_3)]\dot{\theta}_1\dot{\theta}_3 \end{aligned} \tag{8}$$

$$\begin{aligned} f_2 = & [0.04(0.125 + m_L)\sin(\theta_2) + 0.02(1 + 2m_L)\sin(\theta_2 + \theta_3)]\dot{\theta}_1^2 \\ & - [0.02(1 + 2m_L)\sin(\theta_3)]\dot{\theta}_3^2 - [0.04(1 + 2m_L)\sin(\theta_3)]\dot{\theta}_1\dot{\theta}_3 \\ & - [0.04(1 + 2m_L)\sin(\theta_3)]\dot{\theta}_2\dot{\theta}_3 \end{aligned} \tag{9}$$

$$\begin{aligned} f_3 = & 0.2(1 + 2m_L)[\sin(\theta_3) + \sin(\theta_2 + \theta_3)]\dot{\theta}_1^2 \\ & + 0.2(1 + 2m_L)\sin(\theta_3)\dot{\theta}_2^2 \\ & + 0.4(1 + 2m_L)\ \sin(\theta_3)\dot{\theta}_1\dot{\theta}_2 \end{aligned} \tag{10}$$

The components of the gravity force vector are as follows:

$$\begin{aligned} g_1 = & 0.1g\ [(5 + 2m_L)\cos(\theta_1) \\ & + (1 + 2m_L)\cos(\theta_1 + \theta_2 + \theta_3) \\ & + (3 + 2m_L)\cos(\theta_1 + \theta_2)] \end{aligned} \tag{11}$$

$$g_2 = 0.1g\ [(1 + 2m_L)\cos(\theta_1 + \theta_2 + \theta_3) + (3 + 2m_L)\cos(\theta_1 + \theta_2)] \tag{12}$$

$$g_3 = 0.1g\ [(1 + 2m_L)\cos(\theta_1 + \theta_2 + \theta_3)] \tag{13}$$

where g is acceleration of gravity.

2.2 MATLAB Simscape multibody model

The use of Simscape add-on products grants access to more sophisticated parts and analysis tools, which aid in the design and analysis of control systems. The created model can be parameterized using MATLAB variables and expressions, while Simulink is employed to develop control schemes for our physical system. The process of developing the Simscape model encompasses several sequential stages, as outlined below:

I. *System Solver* In this phase, the comprehensive configuration of mechanical and simulation parameters for the entire system is undertaken. Factors such as acceleration, the gravity vector, linearization delta, and the global frame are defined. Furthermore, the specification of the solver configuration, utilizing the Runge–Kutta method, is executed.

II. *Link Creation* Within the framework of the Simscape toolbox, the "Brick Solid" block is employed. This block functions as a representation of a solid structure, integrating geometry, inertia, mass, graphics elements, and associated frames. Notably, it serves as the fundamental unit for constructing rigid bodies. During this stage, the intrinsic attributes of the links, encompassing geometric dimensions and mass properties, are meticulously defined.

III. *Creation of Joints* Within the Simscape toolbox, the "Revolute Joint" block is utilized to establish a revolute joint connecting two frames. This specific joint configuration enables rotational motion along a singular degree of freedom.

IV. *Frame Transformations* The integration of rigid transformations assumes a pivotal role in defining fixed 3D transformations between frames. These transformations encompass both translational and rotational components, thereby accommodating diverse combinations of translational shifts and rotational adjustments.

The MATLAB Simscape model, which represents the entire rigid link manipulator (RLM) system is shown in Fig. 2.

To validate the accuracy of the mathematical model, a comparative analysis is conducted between the open-loop behaviors of the mathematical model and the MATLAB Simscape model. This evaluation is performed under varying input torque conditions of 0.5Nm, 1Nm, and 1.5Nm for each joint, as depicted in Fig. 3. The responses of the Lagrange model are compared with those of the Simscape model in Fig. 4. Evidently, a significant congruence between the two responses is observed, substantiating the suitability of the Simscape model for subsequent investigations.

3 Sliding mode controller with linear PID sliding surface

Selecting the sliding surface function is a necessary step before designing a sliding mode controller. Then, an equivalent controller and switching mode controller should be constructed depending on the states of the dynamical system. The basic principle behind this kind of control is to place, regardless of the initial conditions, the representative point of the system's evolution on a hypersurface of the phase space, representing a set of static relationships between the state variables. There are two components in the sliding mode control as described in Eq. (14).

$$\boldsymbol{\tau}(\boldsymbol{t}) = \boldsymbol{\tau}_{\mathbf{cont}}(\boldsymbol{t}) + \boldsymbol{\tau}_{\mathbf{discont}}(\boldsymbol{t}) \tag{14}$$

where $\boldsymbol{\tau}_{\mathbf{cont}}(\boldsymbol{t})$ is the continuous part of the controller (called the equivalent controller) and $\boldsymbol{\tau}_{\mathbf{discont}}(\boldsymbol{t})$ is the discontinuous part (called the switching mode controller).

The PID sliding surface for the existing SMC can be specified with Eq. (15).

$$\boldsymbol{\sigma}(\boldsymbol{t}) = K_p\boldsymbol{e}(\boldsymbol{t}) + K_i\int_0^t \boldsymbol{e}(\boldsymbol{t})dt + K_d\dot{\boldsymbol{e}}(\boldsymbol{t}) \tag{15}$$

where $K_p, K_i andK_d$ are positive definite diagonal matrices. Furthermore, $\boldsymbol{e}(\boldsymbol{t}) = \boldsymbol{\theta}_d(\boldsymbol{t}) - \boldsymbol{\theta}_m(\boldsymbol{t})$ is the tracking error vector with $\boldsymbol{\theta}_d(\boldsymbol{t}), \boldsymbol{\theta}_m(\boldsymbol{t})$ denote the desired and the measured angular position vectors.

To find the continuous equivalent controller $\boldsymbol{\tau}_{\mathbf{cont}}(\boldsymbol{t})$, it is necessary that $\dot{\boldsymbol{\sigma}}(\boldsymbol{t}) = 0$. Applying this, we arrive at the following equation:

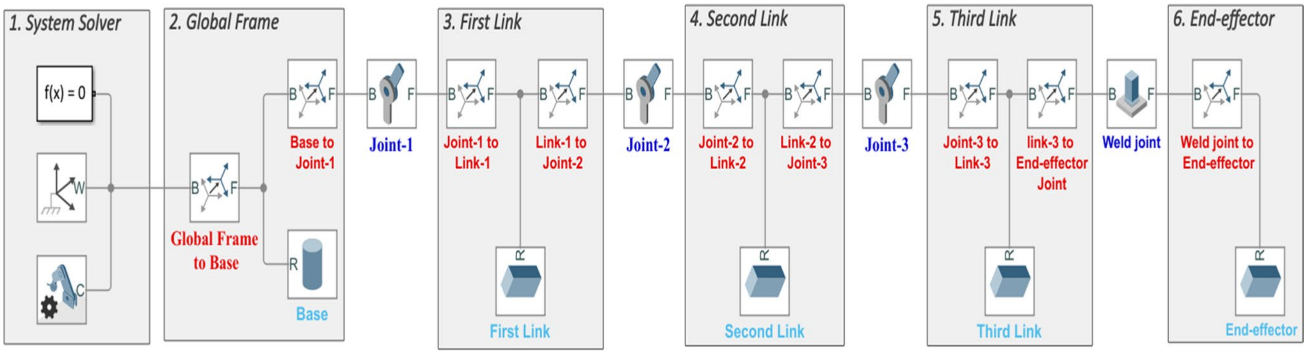


Fig. 2 MATLAB Simscape model of the robotic manipulator

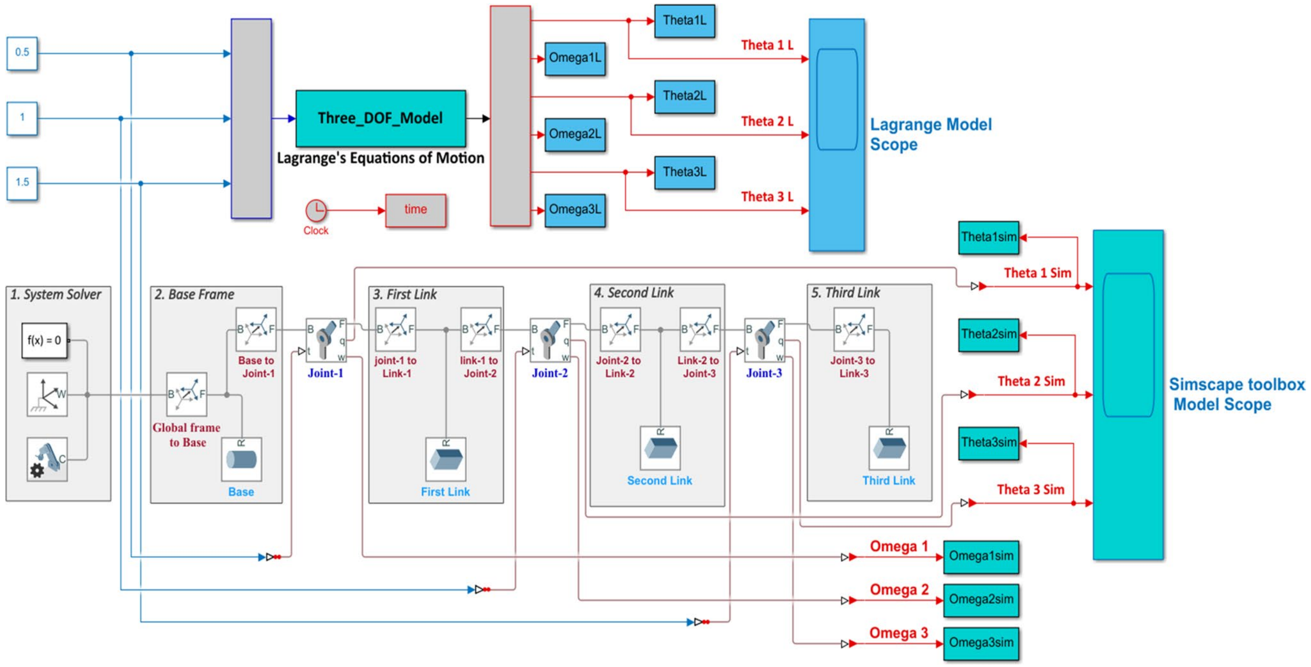


Fig. 3 System configuration during validation of the Lagrange's model and the Simscape model of the 3-link robot manipulator

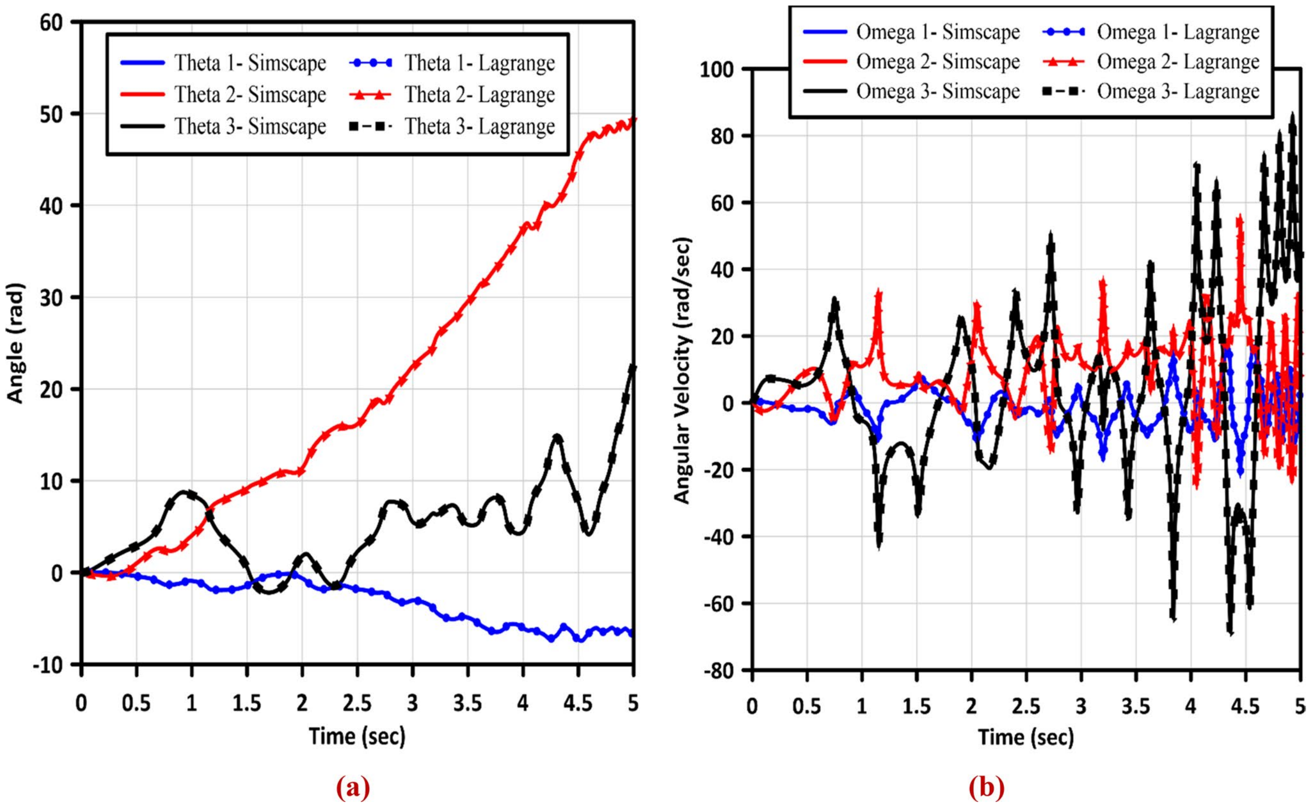


Fig. 4 Validation results **a** The measured angles of both modeling techniques and **b** The measured angular velocities of both modeling techniques

$$\boldsymbol{\tau}_{\mathbf{cont}}(t) = B(\theta)\left[\ddot{\theta}_d(t) + K_d^{-1}\left(K_p\dot{e}(t) + K_i e(t)\right)\right] + f\left(\theta,\dot{\theta}\right) + \mathbf{g}(\theta) \quad (16)$$

where we have used Eqs. (1) and (15).

The switching mode control law $\boldsymbol{\tau}_{\mathbf{discont}}(t)$ is chosen to be of the following form [43–45]

$$\boldsymbol{\tau}_{\mathbf{discont}}(t) = B(\theta)\lambda\,\mathbf{sign}\,(\boldsymbol{\sigma}(t)) \quad (17)$$

where λ is a positive definite diagonal matrix and **sign**(•) is the signum vector function.

To find the minimum values of the controller gains that ensures the system stability we consider the following positive definite Lyapunov function candidate:

$$V = 0.5\boldsymbol{\sigma}^{T}B(\theta)\boldsymbol{\sigma} \quad (18)$$

Using Eqs. (1), (15), (16) and (17), the time derivative of V is upper bounded as follows

$$\dot{V} \le -\left(\rho_{\min}\left\{K_d B(\theta)\lambda\right\} - d\right)\|\boldsymbol{\sigma}\| \quad (19)$$

where $\rho_{\min}\{\bullet\}$ is the minimum eigen value of the matrix $\{\bullet\}$.

The structure of Eq. (19) proves that the system is globally exponentially stable, and the system reaches the sliding surface in a finite time provided that $\rho_{\min}\left\{K_d B(\theta)\lambda\right\} - d > 0$.

Since the signum function causes chattering in practice, it will be replaced here with the saturated vector function $\mathbf{tanh}(\boldsymbol{\sigma}(t)/\delta)$, where δ is a very small positive constant. This replacement changes the state of the system stability, and the system becomes globally ultimately bounded stable. The later causes the system states converge to the sphere defined by$\|\sigma\| \le \delta$. Since the sliding surface is a stable system, the ultimate bound of the tracking error will be $\|e\| \le \frac{\delta}{\rho_{\min}\{K_p\}}$. Due to the later and to improve the system performance, the elements of the diagonal matrices$k_{p_{ii}}$,$k_{I_{ii}}$,k_{dii}, λ_{ii} and the value of δ are to be optimized with the metaheuristic optimization algorithms.

4 The MELGWO algorithm

The gray wolf optimizer (GWO) algorithm [40] suffers from slow convergence and being trapped in local optima when applied to multi-modal optimization problems. To address these limitations, a modified version of GWO called MELGWO was proposed [41]. The modifications include a memory swarm to store the best solutions found during iterations, evolutionary operators to enhance search efficiency, stochastic local search, and Linear population size reduction technique (LPSR) into the basic GWO algorithm to improve its capabilities for both exploitation and exploration, and to maintain a better balance between the two. The following subsections describe the procedures of exploration and exploitation of the proposed algorithm, also the pseudocode is described in Algorithm (1).

4.1 Initialization and fitness evaluation

Initially, the algorithm identifies the three wolves with the lowest cost function values in the population, denoted as X_α, X_β and X_δ, corresponding to the alpha, beta, and delta wolves, respectively. During each iteration, these three wolves are regarded as the leaders and are assumed to contain the optimal solution. All wolves randomly move toward these leaders to explore the region for the optimum. The position of each wolf (i.e., $i = 1, 2, ..., N_P$) is updated in every iteration using Eqs. (20)–(23) [40, 41], where N_{P} represents the population size.

$$X1_j = X_{\alpha,j} - [2\times\theta\times\mathrm{rand} - \theta]\left|2\times rand\times X_{\alpha,J} - X_{i,j}\right| \quad (20)$$

$$X2_j = X_{\beta,j} - [2\times\theta\times\mathrm{rand} - \theta]\left|2\times rand\times X_{\beta,J} - X_{i,j}\right| \quad (21)$$

$$X3_j = X_{\delta,j} - [2\times\theta\times\mathrm{rand} - \theta]\left|2\times rand\times X_{\delta,J} - X_{i,j}\right| \quad (22)$$

$$\theta = 2\left(1 - \frac{It}{It_{\max}}\right) \quad (23)$$

where j represents the index for the dimensions of the problem and ranges from 1 to D. θ denotes a coefficient that gradually decreases from 2 to 0 as the iterations progress. The term *rand* represents a random number generated from a uniform distribution between 0 and 1. The calculated positions of the ith wolf, X_1, X_2, and X_3, correspond to the directions toward the alpha, beta, and delta leaders, respectively, during the random movement. Since the *ith* wolf can only occupy one position, its position after the iteration is determined by taking the average of X_1, X_2, and X_3 [40, 41] as follows.

$$X_{i,j} = \frac{X1_j + X2_j + X3_j}{3} \quad (24)$$

The value of $X_{i,j}$, representing the position of the ith wolf in the *jth* dimension, is compared with the lower and upper bounds of the *jth* variable. If $X_{i,j}$ is outside the range defined by these bounds, it is set to the corresponding limit.

4.2 Memory swarm modification

The memory swarm of wolves is designed to store the present position and corresponding cost function value of the explorer swarm wolves, including the initial best solutions (alpha, beta, and delta). The memory swarm is updated with better wolves from the explorer swarm after performing evolutionary operations.

4.3 Evolutionary operators

The incorporation of evolutionary operators into the gray wolf optimizer (GWO) algorithm's explorer swarm can improve its search efficiency. Among the evolution-based algorithms, Differential Evolution (DE) has simple principles, few parameters, and is widely used. DE consists of three primary processes: mutation, crossover, and selection. After selecting an individual (target vector, X_j), two weight variances are applied to the population to provide variation. The mutation operator is defined as listed in Eq. (25) [41]

$$V_j^{(t+1)} = X_j + F \times \left(X_{\text{alpha}} - X_j\right) \tag{25}$$

where X_{alpha} and X_j represent the position of the best wolf and a random wolf. To ensure that wolves can evolve in a way that benefits their development, an optimal variation factor must be created. A dynamic scaling factor listed in Eq. (26) [41] is developed to increase the algorithm's exploration ability during the initial search process, allowing it to escape local optima and strengthen the local search in later iterations.

$$F = f_{\min} + \left(f_{\max} - f_{\min}\right) \times \frac{It_{\max} - (It - 1)}{It_{\max}} \tag{26}$$

where $f_{\max}$, $f_{\min}$, $It_{\max}$ and It represents the maximum and minimum values of F, maximum number of iterations and current iteration number, respectively.

The target vector, X_j, is then crossed over with the corresponding variation vector $V_j^{(t+1)}$ to generate a test individual, $U_j^{(t+1)}$ as listed in Eq. (27) [41] where a stochastic technique is employed to ensure that at least one bit of $U_j^{(t+1)}$ is different from the variation vector, $V_j^{(t+1)}$.

$$U_j^{(t+1)} = \left\{ \begin{array}{cc} V_j^{(t+1)} & \text{if } rand\,(j) \le P_c \\ X_j & \text{otherwise} \end{array} \right\} \text{for} j = 1, 2, \ldots \ldots D \tag{27}$$

where V_j, U_j and X_j represent the variation, trial, and target vectors and P_{c} is the probability of crossover. The process of selecting between trial and target vectors involves the implementation of a greedy strategy. Once mutation and crossover have been performed, the resulting trial vector, $U_j^{(t+1)}$, is compared with the target vector, X_j^t. The superior vector is then chosen as the new solution, as illustrated in Eq. (28) [41].

$$X_j^{(t+1)} = \left\{ \begin{array}{cc} U_j^{(t+1)} & \text{if } U_j^{(t+1)} \quad lt;f\left(X_j^t\right) \\ X_j^{(t+1)} = X_j^t & \text{otherwise} \end{array} \right\} \text{for } j = 1, 2, \ldots \ldots D \tag{28}$$

4.4 Stochastic local search

The implementation of stochastic local search is executed in proximity to half of the wolves that have been randomly selected. For the ith wolf within the memory swarm, the local search process entails several steps. Initially, the nearest neighboring wolf is identified among all the wolves within the memory swarm, based on the Euclidean distance between their respective positions within the search space. The corresponding position of the nearest neighbor is denoted by$X_{n,j}$, where$j = 1, 2, ..., D$.

During step 2 of the process, a local or temporary wolf (referred to as$X_{t,j}$, where$j = 1, 2, ..., D$) is stochastically generated based on the cost function of the nearest neighbor. If the cost function of the nearest neighbor is less than or equal to that of the current *ith* wolf, the temporary wolf is generated using Eq. (29) [41], which generates a new neighbor that lies between the i^{th} wolf and its neighbor. Conversely, if the cost function of the nearest neighbor is higher than that of the *ith* wolf, the temporary wolf is generated using Eq. (30) [41], which generates a new neighbor that is situated away from the nearest neighbor.

$$X_{t,j} = X_{i,j} + c_1 \times r \text{ and} \times \left(X_{n,j} - X_{i,j}\right) \tag{29}$$

$$X_{t,j} = X_{i,j} + c_1 \times r \text{ and} \times \left(X_{i,j} - X_{n,j}\right) \tag{30}$$

In this study, the acceleration coefficient, denoted as c_1, has a fixed value of 2. The position of the particle, $X_{t,j}$, is assessed for satisfaction of the specified bounds. If any of the bounds are violated, the corresponding value of $X_{t,j}$ is adjusted to the violated bound.

4.5 Linear population size reduction technique

LPSR involves removing members of the wolf population by utilizing a linear equation. At iteration 1, the wolf population size is P_{int}, while the final wolf population size is represented as $P_{\min}$. The wolf population size for the succeeding generation, N_{G+1}, is calculated using Eq. (31) [41].

$$N_{G+1} = \text{round}\left[\frac{P_{\min} - P_{\text{int}}}{\text{MaxNFEs}}\right] \times \text{NFE} + P_{\text{int}} \tag{31}$$

In this study, $P_{\min}$ is defined as the minimum number of wolves, which is determined by the algorithm's parameters or can be specified by the user. For this research, $P_{\min}$ is set to 10. The current number of function evaluations is represented by NFE, while the maximum number of function evaluations is denoted as MaxNFEs.

process is shown in Fig. 5. Table 2 lists the various parameters of the MELGWO and the other algorithms in literature have been utilized to optimize controller gains.

Algorithm (1): MELGWO Pseudo-code

```
Input: Number of wolves (Np), Lower bound (LB), Upper bound (UB), Crossover
percentage(CR), Local search percentage (LS) and Max. no. of iterations ($\mathrm{It}_{\max}$).
Initialize population then select ($\alpha, \beta, \delta$) wolves and store the population and fitness values in the
memory swarm.
While ($It < It_{max}$)
    for $i = 1{:}N_p$
        Check for bounds and select the three best wolves ($\alpha, \beta, \delta$) and store values in the memory
        swarm.
    end for
    for $i = 1{:}N_p$
        Update the positions of wolves using Eqs. ($20-23$), check for bounds and evaluate fitness
        values.
    end for
    /* Evolutionary Operators Phase */
    for $i = 1{:}N_p$
        Apply evolutionary operators using Eqs. ($25-28$), Compare and update memory swarm.
    end for
    /* Stochastic Local Search Phase*/
    for $i = 1{:}N_p/2$
        Apply local search using Eqs. ($29-30$) near half of memory swarm, update the best three
        wolves and update  memory swarm.
    end for
    /* Linear Population Size Reduction Phase */
    Apply LPSR to delete the worst wolves using Eq. ($31$)
end while
Output: Print position and fitness value of the alpha wolve as the best solution.
Stop
```

5 Enhancing the controller performance through optimization

All of the simulations provided in this study have been run in MATLAB/SIMULINK with the ODE45 solver,listed in Eq and the simulation time is taken to be 5 s. Equations (32), (33) and (34), list the desired trajectories of the links. The integral time absolute error (ITAE) is listed in Eq. (35) served as the objective function of the tuning process. A representation of the SMC–PID optimization

The curves of objective function values versus iteration for MELGWO and other optimization techniques from literature study such as Particle Swarm Optimization (PSO), Golden Jackal Optimizer (GJO), Enhanced Artificial Bee Colony with Multi-elite Guidance (MELGWO), Genetic Algorithm (GA), Artificial Bee Colony (ABC), Sine Cosine Algorithm (SCA), Jellyfish Search Optimizer (JSO), Arithmetic Optimization Algorithm (AOA), Whale Optimizer Algorithm (WOA), Differential Evolution (DE), and gray wolf optimizer (GWO) are depicted in Fig. 6.

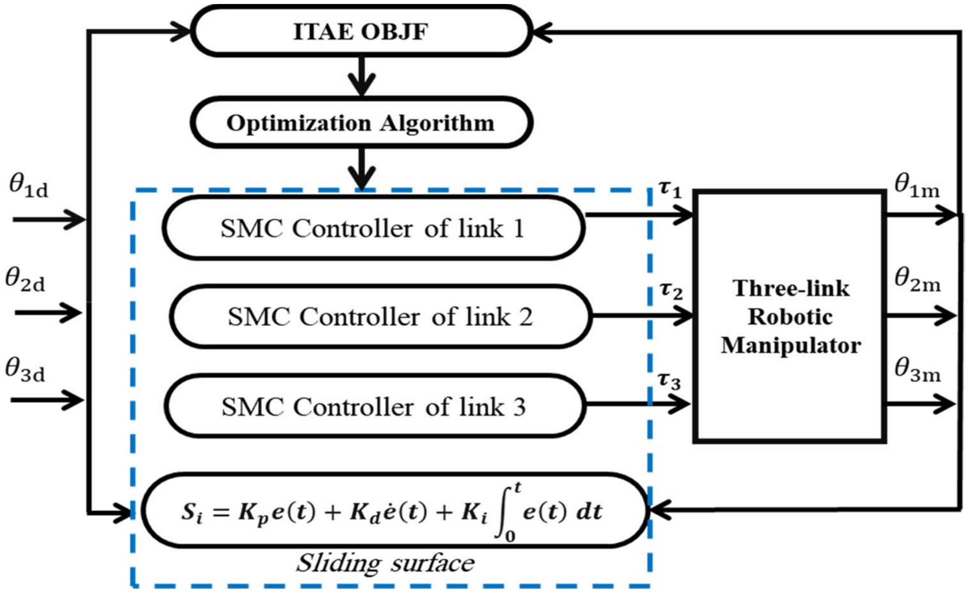


Fig. 5 Schematic representation of the optimization process

Table 2 The design parameters of MELGWO and the algorithms in literature

	Design parameter	Value
All Algorithm	Population size (N)	14
	Max iteration (Itr_{max})	50
	Independent run number	10
	Lower bound (LB)	[5 0.1 0.1 50 0.3]
	Upper bound (UB)	[50 10 1 300 5]
PSO	Social & cognitive coefficients	1.49, 1.49
	Inertia weight	0.5
GJO	Constant (β)	1.5
	Constant (s_1)	1.5
GA	Crossover times	4
	Mutation times	5
WOA	Fluctuation range	Decreases from 2 to 0
	Coeff. of the logarithmic spiral shape	1
AOA	Control parameter (μ)	0.5
	Math optimizer prob. (MOP)	[0.2 1]
	Sensitivity parameter (α)	5
MGABC	Modification rate (MR)	0.5
	Elite group (Ge)	3
DE	Crossover probability	0.8
	Scaling factor	0.85
MELGWO	P_{min}	10
	θ	Decreases from 2 to 0
	c_1	2

$$\theta_{d1} = \sin(2t) - 0.1 \tag{32}$$

$$\theta_{d2} = \cos(2t) - 0.1 \tag{33}$$

$$\theta_{d3} = \cos(2t) - 0.1 \tag{34}$$

$$\text{ITAE} = \int_0^t t|\boldsymbol{e}(\boldsymbol{t})|\text{d}t \tag{35}$$

Table 3 displays the parameters of the robotic system model's upper and lower search spaces, together with the MELGWO optimized values. The lower bound values for

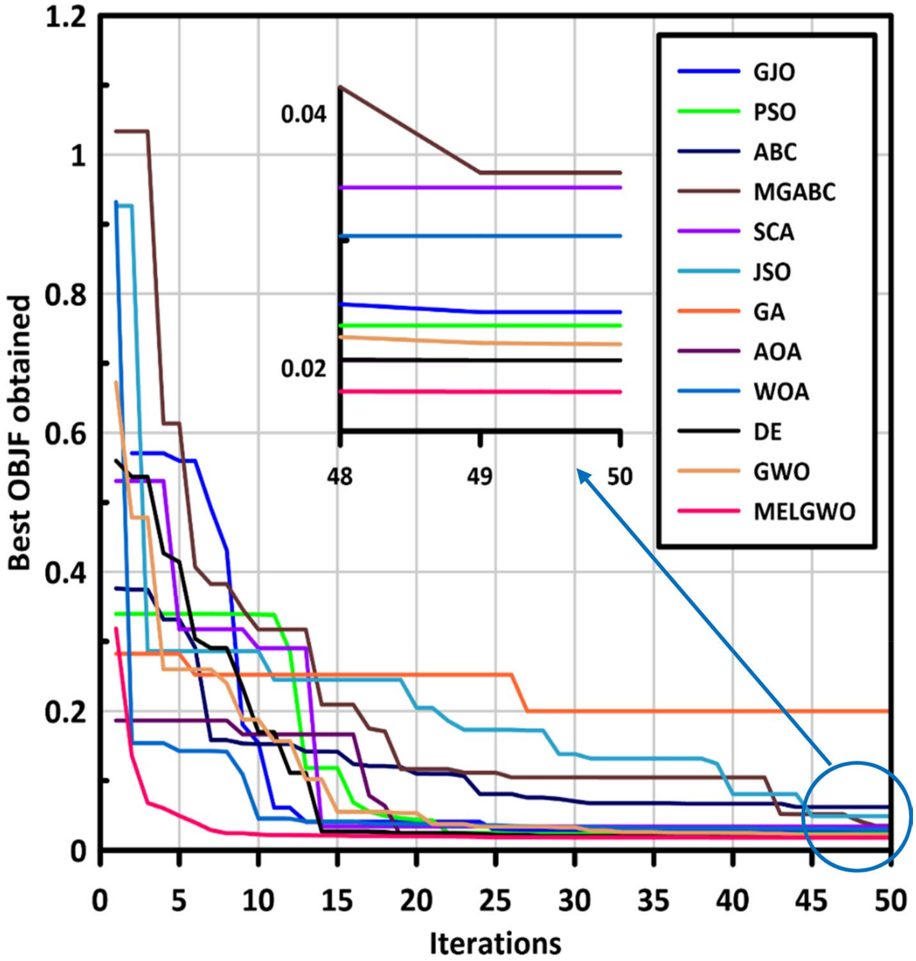


Fig. 6 Convergence history of the optimization process versus iteration

Table 3 The search space and best parameters of the controller using MELGWO algorithm

Controller Parameter	Search space						Best parameters		
	Lower bound			Upper bound					
	Link (1)	Link (2)	Link (3)	Link (1)	Link (2)	Link (3)	Link (1)	Link (2)	Link (3)
k_p	5.00	5.00	5.00	50.00	50.00	50.00	50.00	50.00	50.00
k_i	0.10	0.10	0.10	10.00	10.00	10.00	9.9985	9.8419	0.1826
k_d	0.10	0.10	0.10	1.00	1.00	1.00	0.2553	0.1309	0.5168
λ	50.00	50.00	50.00	300.00	300.00	300.00	300.00	300.00	300.00
δ	0.30	0.30	0.30	5.00	5.00	5.00	0.3002	0.3197	0.3607

the sliding surface and switching mode gains in the controller have been specifically chosen to ensure the system stability. The process of selecting these values incorporated several factors, including the physical properties of the system such as inertia and weight of each link, as well as the system dynamics, desired performance, and stability criteria. These considerations were crucial in determining the lower bounds that would guarantee the stability of the system and ensure its proper functioning.

On the other hand, the upper bound values were selected in a more approximate manner. While these values have not been derived through a rigorous optimization process, they were carefully chosen to establish a reasonable range for the parameters without violating any operational constraints. The intent behind this selection was to encompass a broad range of feasible values that could be explored for experimentation and analysis purposes.

Table 4 Objective function of MELGWO Algorithm compared with other algorithms

Algorithm	Best	Worst	Avg	Std. dev
GJO	0.024354005	0.078534552	0.039024230	0.018407953
PSO	0.023293143	0.277158312	0.146981389	0.100421388
ABC	0.062191106	0.181809045	0.128324291	0.033917559
MGABC	0.035312476	0.193653595	0.116437332	0.057308906
SCA	0.034151176	0.111192778	0.066880495	0.024870223
JSO	0.049140147	0.234923876	0.129393002	0.046604170
GA	0.199794175	1.519026201	0.672425637	0.367326386
AOA	0.020566611	0.034220247	0.024705097	0.006562954
WOA	0.030343802	0.339648676	0.241490241	0.116835134
DE	0.020581955	0.191182825	0.039040350	0.053479170
GWO	0.021832690	0.225627685	0.071245150	0.060885039
MELGWO	**0.018083906**	**0.026065052**	**0.019314763**	**0.002444666**

Bold indicates the specific values that denote the results achieved through the application of the present algorithm

Table 4 presents the best, average, and worst objective function values obtained by the MELGWO optimizer in comparison to other meta-heuristic algorithms in the literature where ten separate runs of each algorithm have been performed. It is notable that the MELGWO algorithm has produced the minimum results overall runs with the lowest standard deviation with the lowest objective function values. The later finding illustrates how the MELGWO method outperforms other algorithms in the literature in terms of result superiority and robustness. Figure 7a, b, c demonstrates the trajectory tracking of each link and Fig. 7d shows the X–Y plot of end-effector. As it can be seen, the desired and actual trajectories are nearly identical. The controller exhibits superior tracking performance, as evidenced by the objective function values of links 1, 2, and 3. Which are 0.002709, 0.005522 and 0.000042, respectively.

6 Comprehensive performance assessment

A robust controller must eliminate measured and unmeasured noise and disturbance and be adaptable to any uncertainties in the system [3]. Therefore, disturbance and noise rejection and adaptability to mass uncertainty in the payload have been investigated and provided in this section. Furthermore, the effect of the joint's flexibility is also investigated.

6.1 Disturbance and noise rejection

The dynamic disturbance signal and noise, listed in Eqs. (36) and (37), are injected at the output of each controller and sensor of joint at the same time as shown in Fig. 8.

$$D_i = \text{per} \times \tau_i \tag{36}$$

$$N_i = \text{per} \times (\theta_{mi}) \tag{37}$$

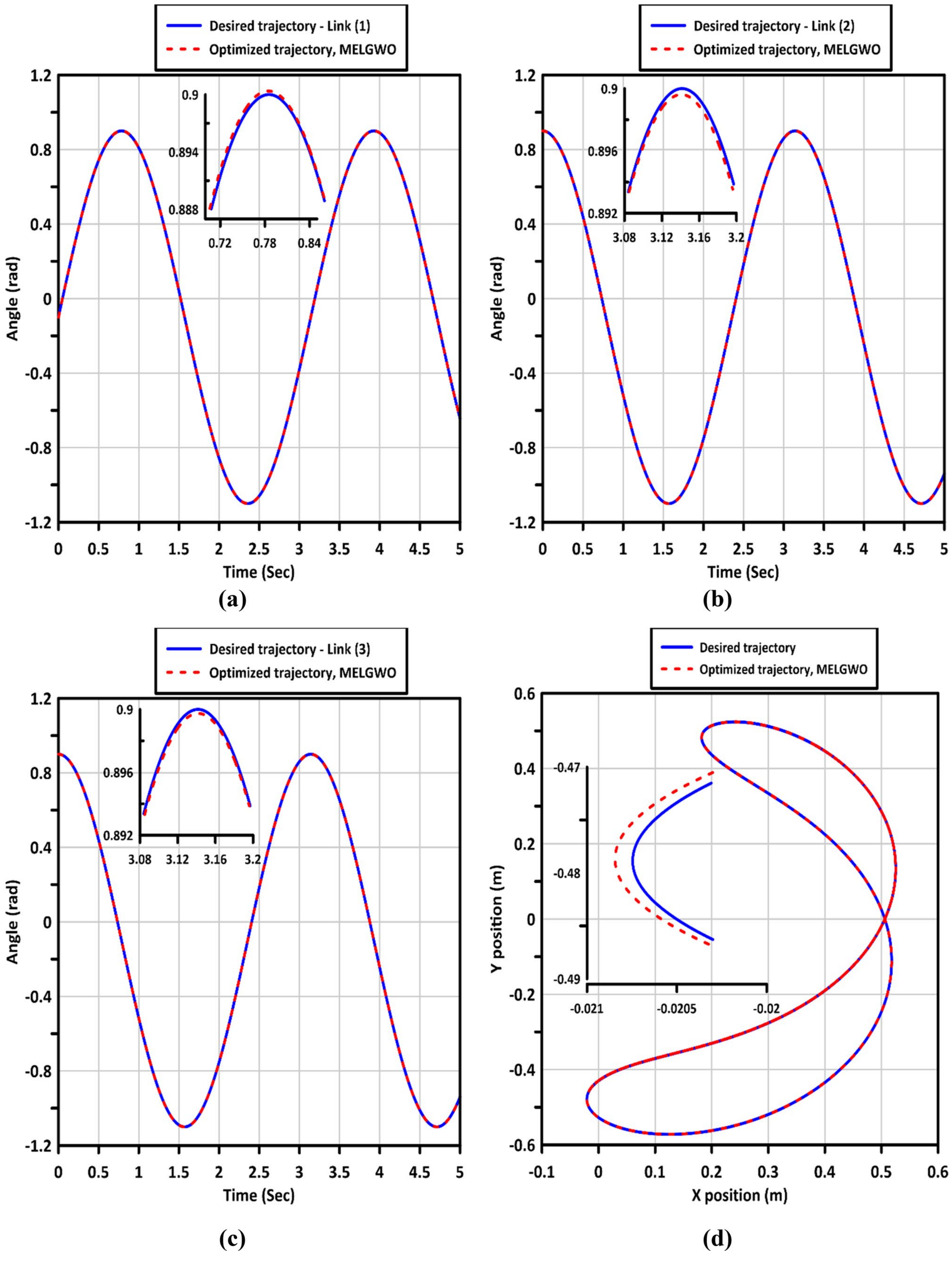
where i=1:3 and τ_i, θ_i correspond to the applied torques and measured angles, respectively.

The systems performance is checked for varying percentage (per) values from 1% to severe conditions of 5% percentage of disturbance and noise. The corresponding objective function values of the optimized controller with MELGWO to the variation in percentage of disturbance and noise are listed in Table 5.

The controller has been designed to minimize the error-based OBJF, and its effectiveness is evident from the observed percentage of increase in OBJF values. This increase ranges from 0.829% under conditions of low disturbance and noise to 43.108% under conditions of severe disturbance and noise.

6.2 Adaptability to the variation in mass of payload

The primary objective of a maneuvering system is to perform the task of gripping and manipulating objects with diverse masses using its end-effector. When there is a change in the mass of the end-effector, it introduces a new configuration to the system, necessitating the need for a robust controller. The robust controller is essential to mitigate the impact of variations in the end-effector mass and ensure stable and reliable performance [3]. The mass of the payload has been incrementally increased by 0.1 kg, ranging from 0.1 to 0.5 kg, in order to evaluate the robustness of the controllers. Table provides a comprehensive list of the obtained



(a)

(b)

(c)

(d)

Fig. 7 Tracking trajectory curves for **a** link 1, **b** link 2, **c** link 3 and **d** $X - Y$ plot of the end-effector

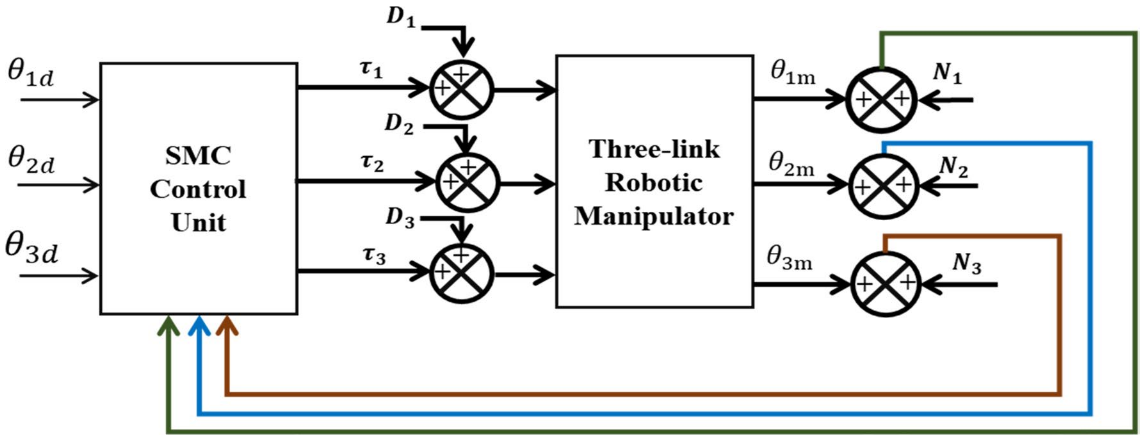


Fig. 8 The optimized manipulator subjected to disturbance at controller output and noise at sensors

Table 5 OBJF variation against the percentage of disturbance and noise

Percentage of disturbance and noise	OBJF value
1.00%	0.017933867
2.00%	0.019923741
3.00%	0.022728715
4.00%	0.025879578
5.00%	0.029699529

Table 6 OBJF variation against increasing the mass of payload

Mass of payload (kg)	OBJF value
0.1	0.018428072
0.2	0.019246539
0.3	0.020422679
0.4	0.021730710
0.5	0.023093656

Table 7 The internal mechanical properties of the FJM configuration

Parameter	Joint (1)	Joint (2)	Joint (3)
Spring stiffness (Nm/rad)	2.5	2.5	2.5
Viscous damping (Nms/rad)	0.01	0.01	0.01

OBJF values corresponding to the increasing payload mass (Table 6).

The optimized controller utilizing the MELGWO algorithm demonstrates resilience in the face of payload mass variations. Through analysis, it is observed that the OBJF values exhibit an inflation ranging from 1.903% under low uncertainty conditions to 27.7028% under severe uncertainty conditions. This indicates that the controller maintains its performance and robustness even in challenging situations with significant uncertainties.

6.3 Effect of joint flexibility

In this section, we will examine the effectiveness of the optimized controller approach in dealing with the impact of joint flexibility. To simulate the system, we utilize the Simscape Multibody toolbox, which allows us to model the three joints as flexible elements. These joints exhibit internal mechanics characterized as listed in Table 7. This configuration enables us to analyze how the controller responds when joint flexibility is present and assess its ability to adapt to such conditions. Figure 9a, b, c shows the error in the trajectories of link 1,2 and 3, respectively, and Fig. 9d shows the *X*–*Y* plot of end-effector. It can be observed that the proposed controller demonstrates exceptional tracking performance with 0.0068, 0.0192 and 0.2162 objective function value for each link, respectively. This indicates the controller's capability to effectively handle flexible joint manipulator (FJM), ensuring precise and smooth trajectory tracking.

7 Conclusion

The optimization of a sliding mode controller with a proportional integral derivative (PID) surface has been achieved by employing an enhanced gray wolf optimizer algorithm (MELGWO). The selection of the optimal parameters of the sliding surface poses a challenge due

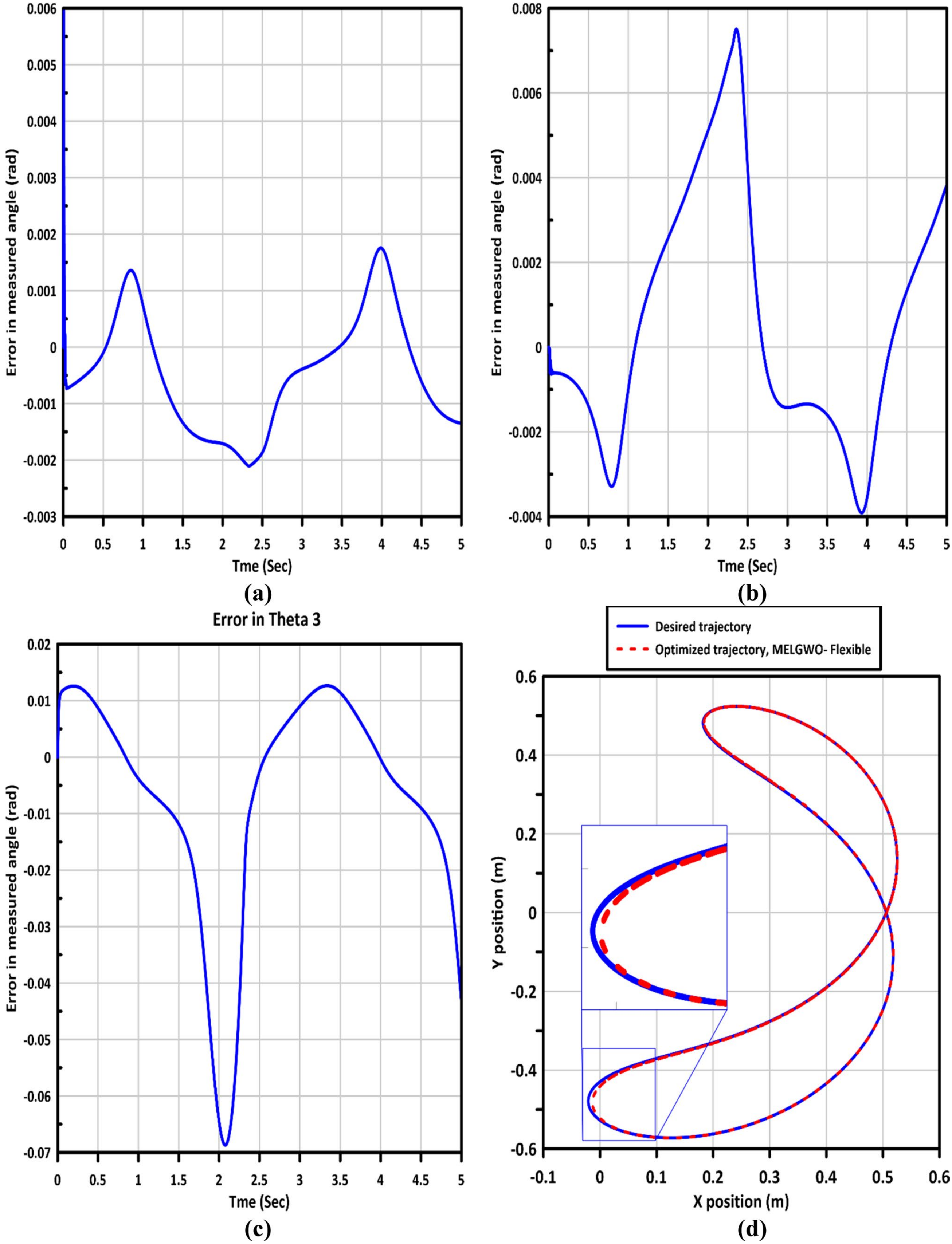


Fig. 9 The trajectory tracking curves of the FJM system: **a** Link 1, **b** Link 2, **c** Link 3, and **d** the $X - Y$ plot of the end-effector

to the vast search space, leading many existing optimization algorithms to become trapped in local optima and fail to identify the optimal parameters for the sliding surface and switching mode controller gain. The MELGWO algorithm is chosen for its exceptional exploratory and exploitative techniques, enabling it to find the best solution. The convergence analysis conducted in this study provides evidence of the MELGWO algorithm's exceptional capability to overcome local minima through its proficient exploration and exploitation techniques. This algorithm demonstrates superiority over other algorithms mentioned in the existing literature in terms of its ability to navigate and escape from local minima solutions. The investigation focuses on disturbance and noise rejection, as well as robustness against uncertainty in payload mass. The proposed controller demonstrates the ability to mitigate the effects of disturbances and noises, maintaining the objective function (OBJF) at a minimum even in highly noisy and disturbed systems. The controller effectively minimizes the OBJF, with the percentage increase in OBJF values ranging from 0.829% under conditions of low disturbance and noise to 43.108% under conditions of severe disturbance and noise. Additionally, the optimized controller exhibits resilience to variations in payload mass analysis, with the percentage increase in OBJF values ranging from 1.903% under low uncertainty conditions to 27.7028% under severe uncertainty conditions. Furthermore, the adaptability of the controller to joint flexibility is evaluated, demonstrating superior tracking performance with minimal vibration in the movement of the end effector.

Author contribution MIA done Simscape modeling, writing-conceptualization and methodology, original draft preparation, and optimization developing. AMMA done writing-original draft preparation and preparation of figures. SE helped in preparation of figures and supervision, writing—reviewing and editing. KAFM contributed to original draft preparation, investigation, writing—reviewing and editing. KRA helped in idea, mathematical modeling, preparation of figures, writing-conceptualization and methodology, writing-reviewing and editing and supervision. All authors have reviewed the manuscript.

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB). Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB).

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

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Alandoli EA, Lee TS, Lin YJ, Vijayakumar V (2021) Dynamic model and intelligent optimal controller of flexible link manipulator system with payload uncertainty. Arab J Sci Eng 46(8):7423–7433. https://doi.org/10.1007/s13369-021-05436-7
2. Lee TS, Alandoli EA (2020) A critical review of modelling methods for flexible and rigid link manipulators. J Braz Soc Mech Sci Eng. https://doi.org/10.1007/s40430-020-02602-0
3. Kumar J, Kumar V, Rana KPS (2020) Fractional-order self-tuned fuzzy PID controller for three-link robotic manipulator system. Neural Comput Appl 32(11):7235–7257. https://doi.org/10.1007/s00521-019-04215-8
4. Real-Time Implementation of Sliding Mode Control Technique for Two-DOF Industrial Robotic Arm İki Serbestlik Derecesine Sahip Endüstriyel Bir Robotun Kayan Kipli Kontrol Yöntemi ile Kontrolünün Gerçek Zamanlı Uygulaması. vol 8(4), pp 77–85 (2018)
5. Korayem MH, Dehkordi SF, Mojarradi M, Monfared P (2019) Analytical and experimental investigation of the dynamic behavior of a revolute-prismatic manipulator with N flexible links and hubs. Int J Adv Manuf Technol 103(5–8):2235–2256. https://doi.org/10.1007/s00170-019-03421-x
6. Korayem MH, Dehkordi SF (2017) Derivation of dynamic equation of viscoelastic manipulator with revolute–prismatic joint using recursive Gibbs-Appell formulation. Nonlinear Dyn 89(3):2041–2064. https://doi.org/10.1007/s11071-017-3569-z
7. Sadegh Lafmejani H, Zarabadipour H (2014) Modeling, simulation and position control of 3DOF articulated manipulator. Indones J Electr Eng Inform. https://doi.org/10.11591/ijeei.v2i3.119
8. Renfrew A (2004) Book review: introduction to robotics: mechanics and control. Int J Electr Eng Educ 41(4):388–388. https://doi.org/10.7227/ijeee.41.4.11
9. S G, A S, M S, F G (2018) Dynamic modelling with a modified PID controller of a three link rigid manipulator. Int J Comput Appl 179(34):37–42. https://doi.org/10.5120/ijca2018916772
10. Bendimrad A, El Amrani A, El Amrani B (2022) Optimization of the performances of a two-joint robotic arm using sliding mode control. Int J Automot Mech Eng 19(2):9634–9646. https://doi.org/10.15282/ijame.19.2.2022.01.0743
11. West C, Montazeri A, Monk SD, Taylor CJ (2016) A genetic algorithm approach for parameter optimization of a 7DOF robotic manipulator. IFAC-PapersOnLine 49(12):1261–1266. https://doi.org/10.1016/j.ifacol.2016.07.688
12. Boujnah F, Knani J (2016) Motion simulation of a manipulator robot modeled by a CAD software. In: Proc. 2015 7th International Conference on Modelling, Identification and Control ICMIC 2015, no. Icmic, pp 3–8. https://doi.org/10.1109/ICMIC.2015.7409442

13. Schlotter M (2003) Multibody system simulation with SimMechanics. Analysis 1–23
14. Manjaree S, Thomas M (2017) Modeling of multi-DOF robotic manipulators using sim-mechanics software. Indian J Sci Technol. https://doi.org/10.17485/ijst/2016/v9i48/105833
15. Loucif F, Kechida S (2020) Optimization of sliding mode control with PID surface for robot manipulator by evolutionary algorithms. Open Comput Sci 10(1):369–407. https://doi.org/10.1515/comp-2020-0144
16. Zhang J, Meng W, Yin Y, Li Z, Ma L, Liang W (2022) High-order sliding mode control for three-joint rigid manipulators based on an improved particle swarm optimization neural network. Mathematics. https://doi.org/10.3390/math10193418
17. Pathak RR, Mohanty S, Sengupta A (2022) An optimization-based adaptive sliding mode control for an inverted pendulum. IETE J Res. https://doi.org/10.1080/03772063.2022.2127941
18. Alandoli EA, Lee TS (2020) A critical review of control techniques for flexible and rigid link manipulators. Robotica 38(12):2239–2265. https://doi.org/10.1017/S0263574720000223
19. Moghanni-Bavil-Olyaei MR, Keighobadi J, Ghanbari A, Olegovna Zekiy A (2022) Passivity-based hierarchical sliding mode control/observer of underactuated mechanical systems. JVC/J Vib Control. https://doi.org/10.1177/10775463221091035
20. Ferrara A, Incremona GP (2015) Design of an integral suboptimal second-order sliding mode controller for the robust motion control of robot manipulators. IEEE Trans Control Syst Technol 23(6):2316–2325. https://doi.org/10.1109/TCST.2015.2420624
21. Mohammed AA, Eltayeb A (2018) Dynamics and control of a two-link manipulator using PID and sliding mode control. In: 2018 International Conference on Computer, Control, Electrical, and Electronics Engineering ICCCEEE 2018, pp 1–5. https://doi.org/10.1109/ICCCEEE.2018.8515795
22. Geng J, Sheng Y, Liu X (2014) Time-varying nonsingular terminal sliding mode control for robot manipulators. Trans Inst Meas Control 36(5):604–617. https://doi.org/10.1177/0142331213512367
23. Moghanni-Bavil-Olyaei MR, Ghanbari A, Keighobadi J (2019) Trajectory tracking control of a class of underactuated mechanical systems with nontriangular normal form based on block backstepping approach. J Intell Robot Syst Theory Appl 96(2):209–221. https://doi.org/10.1007/s10846-019-00984-5
24. Zhang J, Shi P, Xia Y (2010) Robust adaptive sliding-mode control for fuzzy systems with mismatched uncertainties. IEEE Trans Fuzzy Syst 18(4):700–711. https://doi.org/10.1109/TFUZZ.2010.2047506
25. Yi S, Zhai J (2019) Adaptive second-order fast nonsingular terminal sliding mode control for robotic manipulators. ISA Trans 90:41–51. https://doi.org/10.1016/j.isatra.2018.12.046
26. Jing C, Xu H, Niu X (2019) Adaptive sliding mode disturbance rejection control with prescribed performance for robotic manipulators. ISA Trans 91:41–51. https://doi.org/10.1016/j.isatra.2019.01.017
27. Esfahani HN, Azimirad V (2013) A new fuzzy sliding mode controller with PID sliding surface for underwater manipulators. Int J Mechatron Electr Comput Technol 3(9):224–249
28. Mohammad A, Ehsan SS (2008) Sliding mode PID-controller design for robot manipulators by using fuzzy tuning approach. In: Proc. 27th Chinese Control Conf. CCC, pp 170–174. https://doi.org/10.1109/CHICC.2008.4605499
29. Deng Y (2019) Fractional-order fuzzy adaptive controller design for uncertain robotic manipulators. Int J Adv Robot Syst 16(2):1–10. https://doi.org/10.1177/1729881419840223
30. Zhou T, Liang XF (2014) Position sliding mode control of manipulator joint based on genetic algorithm. Adv Mater Res 912–914:727–731. https://doi.org/10.4028/www.scientific.net/AMR.912-914.727
31. Long Y, Du ZJ, Wang WD, Dong W (2016) Robust sliding mode control based on GA optimization and CMAC compensation for lower limb exoskeleton. Appl Bionics Biomech. https://doi.org/10.1155/2016/5017381
32. Vijay M, Jena D (2018) Backstepping terminal sliding mode control of robot manipulator using radial basis functional neural networks. Comput Electr Eng 67:690–707. https://doi.org/10.1016/j.compeleceng.2017.11.007
33. Soltanpour MR, Khooban MH (2013) A particle swarm optimization approach for fuzzy sliding mode control for tracking the robot manipulator. Nonlinear Dyn 74(1–2):467–478. https://doi.org/10.1007/s11071-013-0983-8
34. Vijay M, Jena D (2017) PSO based neuro fuzzy sliding mode control for a robot manipulator. J Electr Syst Inf Technol 4(1):243–256. https://doi.org/10.1016/j.jesit.2016.08.006
35. Al-Dujaili AQ, Falah A, Humaidi AJ, Pereira DA, Ibraheem IK (2020) Optimal super-twisting sliding mode control design of robot manipulator: design and comparison study. Int J Adv Robot Syst 17(6):1–17. https://doi.org/10.1177/1729881420981524
36. Medjghou A, Ghanai M, Chafaa K (2018) BBO optimization of an EKF for interval type-2 fuzzy sliding mode control. Int J Comput Intell Syst 11(1):770–789. https://doi.org/10.2991/ijcis.11.1.59
37. Oliveira J, Oliveira PM, Boaventura-Cunha J, Pinho T (2017) Chaos-based grey wolf optimizer for higher order sliding mode position control of a robotic manipulator. Nonlinear Dyn 90(2):1353–1362. https://doi.org/10.1007/s11071-017-3731-7
38. Elawady WM, Lebda SM, Sarhan AM (2020) An optimized fuzzy continuous sliding mode controller combined with an adaptive proportional-integral-derivative control for uncertain systems. Optim Control Appl Methods 41(3):980–1000. https://doi.org/10.1002/oca.2580
39. Goel A, Mobayen S (2021) Adaptive nonsingular proportional–integral–derivative-type terminal sliding mode tracker based on rapid reaching law for nonlinear systems. JVC/J Vib Control 27(23–24):2669–2685. https://doi.org/10.1177/1077546320964287
40. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey Wolf Optimizer. Adv Eng Softw 69:46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
41. Ahmed R, Rangaiah GP, Mahadzir S, Mirjalili S, Hassan MH, Kamel S (2023) Memory, evolutionary operator, and local search based improved Grey Wolf Optimizer with linear population size reduction technique. Knowl-Based Syst. https://doi.org/10.1016/j.knosys.2023.110297
42. Azeez MI, Abdelhaleem AMM, Elnaggar S, Moustafa KAF, Atia KR (2023) Optimization of PID trajectory tracking controller for a 3-DOF robotic manipulator using enhanced Artificial Bee Colony algorithm. Sci Rep 13(1):1–19. https://doi.org/10.1038/s41598-023-37895-3
43. Slotine J-J, Li W (1991) Applied nolinear optimal control
44. Mahmoodabadi MJ, Shahangian MM (2022) A new multi-objective artificial bee colony algorithm for optimal adaptive robust controller design. IETE J Res 68(2):1251–1264. https://doi.org/10.1080/03772063.2019.1644211
45. Azeez MI, Abdelhaleem AMM, Elnaggar S, Moustafa KAF, Atia KR (2023) Optimized sliding mode controller for trajectory tracking of flexible joints three—link manipulator with noise in input and output. Sci Rep 0123456789:1–18. https://doi.org/10.1038/s41598-023-38855-7

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.