machines

MDPI

Article

Time-Optimal Trajectory Planning for Industrial Robots Based on Improved Fire Hawk Optimizer

Shuxia Ye [1,2], Bo Jiang [1], Yongwei Zhang [1,2], Liwen Cai [1], Liang Qi [1,2,*] and Siyu Fei [1]

[1] School of Automation, Jiangsu University of Science and Technology, No. 666 Changhui Road, Zhenjiang 212114, China; yeshuxia@just.edu.cn (S.Y.); 202210331115@stu.just.edu.cn (B.J.); ywzhang@just.edu.cn (Y.Z.); 231210302101@stu.just.edu.cn (L.C.); 241210302101@stu.just.edu.cn (S.F.)
[2] Jiangsu Shipbuilding and Ocean Engineering Design and Research Institute, Zhenjiang 212100, China
* Correspondence: alfred_02030210@just.edu.cn

Abstract

Focusing on joint-space time-optimal trajectory planning for industrial robots, this study integrates 3-5-3 piecewise polynomial parameterization with an improved Fire Hawk Optimization algorithm (TFHO). Subject to joint position, velocity, and acceleration limits, segment durations are optimized as decision variables. TFHO employs Tent-chaotic initialization to improve the uniformity of initial solutions and a two-phase adaptive Lévy–Gaussian–Cauchy hybrid mutation to balance early global exploration with late local exploitation, mitigating premature convergence and enhancing stability. On benchmark functions, TFHO attains the lowest mean area under the convergence curve (AUC; lower is better). Wilcoxon signed-rank tests show statistically significant improvements over FHO, PSO, GWO, and WOA ($p \leq 0.05$). Ablation studies indicate a pronounced reduction in run-to-run variability: the standard deviation decreases from 0.3157 (FHO) to 0.0023 with TFHO, a 99.27% drop. In an ABB IRB-2600 simulation case, the execution time is shortened from 12.00 s to 9.88 s (−17.66%) while preserving smooth and continuous kinematic profiles (position, velocity, and acceleration), demonstrating practical engineering applicability.

Keywords: trajectory planning; time-optimal; adaptive hybrid mutation; 3-5-3 polynomial

Academic Editor: Zheng Chen

Received: 12 July 2025
Revised: 14 August 2025
Accepted: 25 August 2025
Published: 26 August 2025

Citation: Ye, S.; Jiang, B.; Zhang, Y.; Cai, L.; Qi, L.; Fei, S. Time-Optimal Trajectory Planning for Industrial Robots Based on Improved Fire Hawk Optimizer. *Machines* **2025**, *13*, 764. https://doi.org/10.3390/machines13090764

1. Introduction

With continuous advancements in technology, six-axis robots have been widely applied in industrial settings [1]. In the robot motion control chain, trajectory planning is a core component that directly determines operational efficiency, running stability, and processing quality [2]. According to the planning space, joint trajectory planning methods can be categorized into joint space methods and Cartesian space methods [3]. The Cartesian space method directly plans the path for the end-effector, requiring inverse kinematic solutions for the entire working path, which results in high computational load and susceptibility to singularities [4]. In contrast, the joint space method only solves the inverse kinematics for the initial, terminal, and a few key interpolation points, and then reconstructs the continuous trajectory through forward kinematics, thus reducing computational burden and effectively avoiding singularities [5]. Therefore, this paper adopts the joint space trajectory planning strategy.

Time-optimal trajectory planning aims to enable the robot to complete a given task in the shortest possible time while satisfying dynamic, velocity, and acceleration constraints [6]. Among the various trajectory generation methods, polynomial interpolation is favored for its concise mathematical form, fewer parameters, and ability to ensure

continuous and smooth trajectories. Notably, segmented polynomials—including cubic, quintic, and 3-5-3 forms—offer both computational efficiency and flexible adaptability to diverse process requirements by tuning control points and boundary conditions [7]. Consequently, in recent years, a large number of studies have focused on polynomial interpolation, seeking to combine time parameterization and optimization strategies to simultaneously achieve "shape controllability" and "time optimality" [6–8]. Zhang et al. [9] applied cubic polynomial interpolation in the trajectory planning of picking robots, and verified that the generated trajectories are smooth with no significant vibration in the joints. Mohammed et al. [10] applied both cubic and quintic polynomials to a lower limb exoskeleton system and found that cubic polynomials may result in discontinuous acceleration curves and insufficient fitting accuracy for position trajectories, while quintic polynomials provide higher accuracy and smoother acceleration, but may still cause considerable jerk in acceleration and higher derivatives. In his classical work, Koch et al. [11] comprehensively described the segmented strategy of "acceleration–constant speed–deceleration," and presented the standard 3-5-3 interpolation, which uses cubic polynomials for the start and end segments and a quintic polynomial for the middle segment. This approach not only retains the high-order smoothness of the quintic polynomial in the middle segment, but also leverages the inherent zero-jerk feature of the cubic polynomials at the boundaries, effectively suppressing instantaneous impact at start and end points. Furthermore, it is easy to insert an arbitrary number of path points, making it highly scalable for complex multi-segment tasks. Therefore, this paper adopts 3-5-3 polynomial interpolation as the trajectory generation method.

Li et al. [12] applied the 3-5-3 polynomial interpolation method to plan smooth and time-optimal trajectories for industrial robotic arms along specified paths, verifying the performance of these trajectories through simulations and experiments. Experimental results demonstrated that the planned smooth trajectories provide excellent feasible time-optimal motions. Reference [13] addresses the problem that cubic polynomial trajectory planning cannot ensure continuous acceleration, and the acceleration trajectories of each joint have jumps at the starting point, which easily cause jitter during the start and stop of joint motors. It uses 3-5-3 polynomials for optimization. The results show that the introduction of the 3-5-3 polynomial interpolation function can avoid the jitter problem during start and stop.

Although polynomial interpolation methods have achieved notable success in manipulator trajectory planning [14,15], traditional polynomial interpolation typically relies on empirical segmentation for key timings (such as the durations of acceleration, constant speed, and deceleration phases), which fundamentally cannot guarantee global optimality in time allocation and is less responsive to complex dynamic coupling and task cycle changes [16]. With the rise of intelligent optimization algorithms, coupling polynomial interpolation with intelligent algorithms has provided novel solutions for time-optimal trajectory planning and has quickly become a research hotspot in academia [17,18].

Reference [19] proposed a time-optimal 3-5-3 polynomial interpolation trajectory planning algorithm in joint space based on Particle Swarm Optimization (PSO), effectively addressing challenges associated with high-order polynomials and lack of convex hull property that hinder traditional optimization methods. Reference [20] proposed an improved adaptive genetic particle swarm algorithm to overcome the tendency of conventional PSO to fall into local optima and its low efficiency in robotic time-optimal trajectory planning. Reference [21] introduced a chaotic Lévy particle swarm optimization algorithm for solving various test functions and robotic time-optimal trajectory planning problems, and achieved better performance compared to PSO-GA, Whale Optimization Algorithm (WOA), and Cuckoo Search [22].

In 2022, Azizi et al. [23] proposed the Fire Hawk Optimization (FHO) algorithm, inspired by the foraging behaviors of certain raptors such as the shouting kite, black kite, and brown falcon. As a metaheuristic algorithm, FHO is characterized by being parameter-free and fast-converging. Ashraf et al. [24] addressed the issue of insufficient early population diversity in the original algorithm by incorporating low-discrepancy sequences such as Sobol, Halton, and Torus in the initialization stage, resulting in several IFHO variants that outperformed traditional FHO in terms of convergence speed, mean error, and standard deviation on 23 benchmark functions. Said et al. [25] introduced a dual enhancement mechanism involving vector operators and dimension learning-based hunting strategies to maintain population diversity and suppress premature convergence in response to the original FHO's limitations of behavior-based convergence and insufficient accuracy.

In summary, existing 3-5-3 polynomial interpolation methods largely depend on empirical segmentation and are unlikely to achieve global optimality; mainstream intelligent algorithms such as PSO face limitations such as slow convergence and sensitivity to initial values when tackling high-dimensional, multi-constraint problems. While the Falcon Hawk Optimization algorithm has recently shown advantages in function optimization and parameter identification (notably "parameter-free" and "fast convergence"), its systematic validation in 3-5-3 polynomial time-optimal trajectory planning remains lacking. Furthermore, the original FHO's issues of insufficient early population diversity and propensity for premature convergence may be further exacerbated in this context. Therefore, this paper, for the first time, applies FHO to time-optimal manipulator trajectory planning, and improves the algorithm with Tent chaotic initialization and an adaptive Lévy–Gaussian–Cauchy hybrid mutation strategy. Simulation results verify that the proposed method effectively completes the trajectory planning task. The main contributions are as follows:

(1) Integration of 3-5-3 piecewise polynomial interpolation with an improved Fire Hawk Optimization (TFHO) in joint space, using segment durations as decision variables to achieve time-optimal yet smooth trajectories;

(2) Algorithmic enhancements comprising Tent-chaotic population initialization and a two-phase adaptive Lévy–Gaussian–Cauchy hybrid mutation that balances early global exploration and later refined local exploitation, thereby improving diversity, search efficiency, and escape from local optima;

(3) Comprehensive empirical evidence on nine benchmark functions—showing the lowest mean area under the convergence curve (AUC; an integral-of-error measure over iterations that quantifies overall convergence efficiency, where lower is better)—with Wilcoxon signed-rank significance against FHO/PSO/GWO/WOA, together with an ablation demonstrating markedly reduced run-to-run variability (standard deviation decreased from 0.3157 to 0.0023.) and an ABB IRB-2600 case where execution time decreases from 12 s to 9.88 s (−17.66%), indicating engineering applicability.

The remainder of this paper is organized as follows: Section 2 systematically presents the robot's forward and inverse kinematic models, derives the mathematical formulation for 3-5-3 segmented polynomial interpolation, and constructs the integrated objective function subject to constraints. Section 3 introduces the working principles of the Fire Hawk Optimization algorithm (FHO), details the improvements made for time-optimal trajectory planning, and provides the complete TFHO planning procedure. Section 4 compares the optimization performance of TFHO with classical algorithms on standard test functions, verifies the convergence speed and solution quality through convergence curves and statistical indices under equal iteration numbers, and further validates the effectiveness of the proposed method on a manipulator simulation platform. Section 5 concludes the paper by summarizing the main contributions and limitations, and discusses future application prospects of the algorithm.

2. Problem Description

2.1. Robot Kinematic Model

Six-degree-of-freedom (Six-DOF) articulated robots, characterized by their flexible structure and wide range of motion, have become the mainstream type of robots in modern industrial automation. This category of robots can achieve arbitrary position and orientation control of the end-effector in three-dimensional space, making them widely used in high-precision operations such as complex surface polishing, fine spraying, welding, and assembly. Compared to other types of manipulator structures, Six-DOF articulated robots possess higher spatial flexibility and adaptability, enabling them to meet the requirements of various manufacturing and processing tasks. In this study, the ABB IRB-2600 robot arm is selected as the research subject, with its physical model and modified Denavit-Hartenberg (DH) coordinate system illustrated in Figure 1 [26].

When modeling the robot using the Denavit–Hartenberg (DH) method, the homogeneous transformation matrix of coordinate frame i with respect to frame $i-1$ is given by:

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1}d_i \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1} $$

where θ_i represents the joint angle, α_i represents the link length, and d_i represents the link offset. By substituting the modified DH parameters of the robot into Equation (1), the six joint pose matrices can be obtained. The overall transformation matrix is then derived by successively multiplying these six pose matrices from right to left, as follows:

$$ {}^{0}_{6}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T \tag{2} $$

where ${}^{0}_{6}T$ represents the position and orientation matrix of the robot end-effector with respect to the base coordinate frame, which is the forward kinematics solution of the robot.

The essence of the inverse kinematics solution is to establish the correspondence between the target pose of the robot end-effector in Cartesian space and the joint variables, that is, to deduce the specific joint angles of the robot based on the desired end-effector position and orientation. The solution formulas for the joint variables are as follows:

$$ \begin{cases} {}^{0}_{1}T^{-1} \times {}^{0}_{6}T = {}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T \\ {}^{1}_{2}T^{-1} \times {}^{0}_{1}T^{-1} \times {}^{0}_{6}T = {}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T \\ {}^{2}_{3}T^{-1} \times {}^{1}_{2}T^{-1} \times {}^{0}_{1}T^{-1} \times {}^{0}_{6}T = {}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T \\ {}^{3}_{4}T^{-1} \times {}^{2}_{3}T^{-1} \times {}^{1}_{2}T^{-1} \times {}^{0}_{1}T^{-1} \times {}^{0}_{6}T = {}^{4}_{5}T\,{}^{5}_{6}T \\ {}^{4}_{5}T^{-1} \times {}^{3}_{4}T^{-1} \times {}^{2}_{3}T^{-1} \times {}^{1}_{2}T^{-1} \times {}^{0}_{1}T^{-1} \times {}^{0}_{6}T = {}^{5}_{6}T \end{cases} \tag{3} $$

2.2. 3-5-3 Polynomial Interpolationl

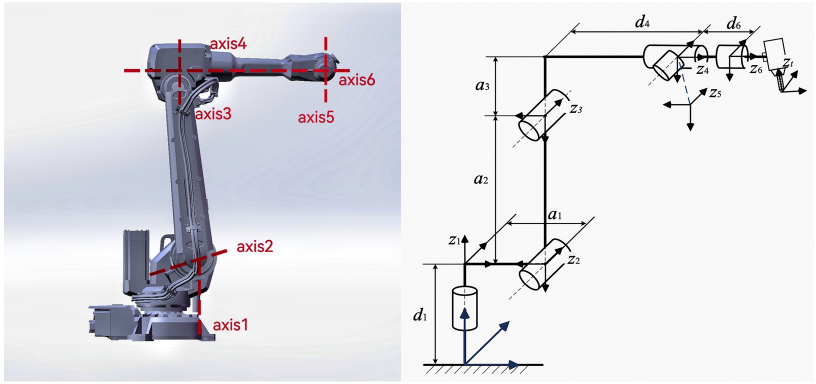
The 3-5-3 polynomial interpolation is a piecewise polynomial interpolation method commonly used in robotic trajectory planning, capable of ensuring continuity, smoothness, and feasibility of the trajectory. This approach divides the overall trajectory into three segments: cubic polynomial interpolation is employed for the initial and terminal segments, while quintic polynomial interpolation is used for the intermediate segment. This effectively enhances the smoothness of the trajectory at critical points and ensures the continuity of derivatives of various orders.

Four interpolation points are defined, and the joint angles of each joint at these interpolation points are obtained by solving the inverse kinematics equations. Let θ_{ij}

denote the joint angle of the i-th joint at the j-th interpolation point, where i represents the joint index ($i = 1, 2, 3, 4, 5, 6$) and j represents the interpolation point index ($j = 0, 1, 2, 3$). represents the interpolation point index. The general expression of the 3-5-3 polynomial for the i-th joint is given by:

$$\begin{cases} l_{i1}(t) = a_{i13}t^3 + a_{i12}t^2 + a_{i11}t + a_{i10} \\ l_{i2}(t) = a_{i25}t^5 + a_{i24}t^4 + a_{i23}t^3 + a_{i22}t^2 + a_{i21}t + a_{i20} \\ l_{i3}(t) = a_{i33}t^3 + a_{i32}t^2 + a_{i31}t + a_{i30} \end{cases} \tag{4}$$

where $l_{i1}(t), l_{i2}(t), l_{i3}(t)$ represent the trajectory segments of the 3-5-3 piecewise polynomial, and a_{ijk} denotes the k-th (k = 0, 1, 2, 3, 4, 5) coefficient of the interpolation function for the j-th segment of the i-th joint.



(**a**) Physical model (**b**) DH coordinate system

Figure 1. The ABB IRB-2600 robot arm Physical model and DH coordinate system.

There are fourteen unknowns a_{ijk} in the above Equation (4). To solve this equation, fourteen boundary conditions are required. Since the manipulator is in a stationary state at both the start and end points, the velocity and acceleration of the manipulator at the start and end points are both zero, that is, the first derivative and the second derivative are zero. In addition, the velocity of the manipulator is stable during operation, so the second derivatives are continuous between segments. After substituting these as constraint conditions, the following coefficient matrix is obtained (the detailed derivation is shown in the Appendix A):

$$A = \begin{bmatrix} t_1^3 & t_1^2 & t_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 3t_1^2 & 2t_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 6t_1 & 2 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & t_2^5 & t_2^4 & t_2^3 & t_2^2 & t_2 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 5t_2^4 & 4t_2^3 & 3t_2^2 & 2t_2 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 20t_2^3 & 12t_2^2 & 6t_2 & 2 & 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_3^3 & t_3^2 & t_3 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3t_3^2 & 2t_3 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6t_3 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5}$$

$$\theta = [0 \ 0 \ 0 \ 0 \ 0 \ x_{i3} \ 0 \ 0 \ x_{i0} \ 0 \ 0 \ x_{i2} \ x_{i1}]^{\mathrm{T}} \tag{6}$$

$$a = A^{-1}\theta \tag{7}$$

In the equation, t_1, t_2, t_3 denote the interpolation times for the three segments of the 3-5-3 polynomial for the i-th joint, x_{ij} represents the position of the i-th joint at the j-th segment, and θ is the joint angle of the robot at the corresponding path point, obtained via inverse kinematics.

In practical applications, the values of t_1, t_2, t_3 are often empirically assigned, and with these three parameters, the joint trajectory can be determined by solving the coefficient matrices in Equations (5)–(7). Considering the physical constraints of the manipulator itself and the set conditional constraints, the values t_1 = 4 s, t_2 = 4 s, t_3 = 4 s were ultimately determined through multiple experiments and the trajectory diagram is shown in the Figure 2.

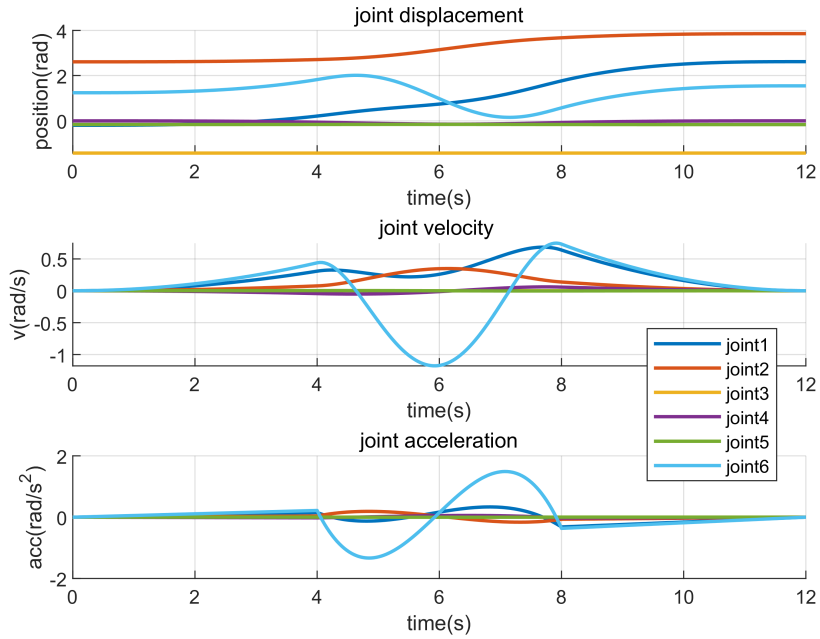


Figure 2. 3-5-3 polynomial trajectory diagram.

However, this value mainly relies on experience and manual parameter adjustment. Multiple trials can only yield approximate values, failing to guarantee optimality. To enhance the scientific rigor and reproducibility of trajectory planning, as well as to obtain a shorter execution time under constraints, it is necessary to introduce optimization algorithms to automatically solve for the optimal time allocation.

2.3. Establishment of the Time-Optimal Model

Under the objective of time optimality, the overall operating time of the manipulator should be minimized. Given that the manipulator's movement time is synchronized, piecewise interpolation is performed using the 3-5-3 polynomial, which divides the four interpolation points into three time segments. Subsequently, the optimal duration for each time segment of each joint, derived through algorithmic computation, varies. To ensure the proper operation of the manipulator, the maximum value among the optimal durations of the same time segment across all joints is selected, and ultimately, the optimal total execution time of the manipulator is determined. The time-optimal function can be expressed as:

$$f = \min\left[\max(t_{i_1}) + \max(t_{i_2}) + \max(t_{i_3})\right] \tag{8}$$

The constraints are as follows:

$$\begin{cases} |\theta_i(t)| \leq \max S_i \\ |\dot{\theta}_i(t)| \leq \max V_i \\ |\ddot{\theta}_i(t)| \leq \max A_i \end{cases} \tag{9}$$

where $i = (1, \ldots, 6)$,t_{i1}, t_{i2}, t_{i3} are the segment durations for the i-th joint; S_i, V_i, A_i are fixed values, represent the maximum displacement, velocity, and acceleration for each joint, respectively. $\theta_i(t)$, $\dot{\theta}_i(t)$, and $\ddot{\theta}_i(t)$ denote the displacement, velocity, and acceleration of each manipulator joint, respectively.

3. Time-Optimal Trajectory Planning Method for Robots

3.1. Fire Hawk Optimization Algorithm

The Fire Hawk Optimization (FHO) algorithm, proposed by Mahdi Azizi in 2022, is a novel metaheuristic optimization algorithm. It is inspired by the natural foraging behavior of "fire hawks" (including species such as the whistling kite, black kite, and brown falcon), which actively ignite small fires by carrying burning branches to drive out prey. By simulating the interaction between fire hawks and prey, FHO achieves an effective combination of global and local search, featuring fast convergence speed, few parameters, and ease of implementation. The basic search process is as follows:

STEP 1: Randomly generate N candidate solutions within the search space, each represented as a D-dimensional vector. The initial position of each solution is determined as follows: where X_i denotes the i-th candidate solution in the search space; d represents the dimensionality of the problem; N is the total number of candidate solutions in the search space; $x_i^j(0)$ represents the initial position of the j-th decision variable for the i-th candidate solution; $x_{i,\min}^j$ and $x_{i,\max}^j$ are the lower and upper bounds of the j-th decision variable for the j-th candidate solution; and rand is a random number uniformly distributed within the range [0,1].

STEP 2: According to the fitness values, the population is divided into two categories: fire hawks and prey. Individuals with better fitness are selected as fire hawks, while the remaining individuals serve as prey. Each prey is assigned to the "territory" of a specific fire hawk based on criteria such as distance and fitness. The distance formula is as follows:

$$D_k^l = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad \begin{cases} l = 1, 2, \ldots, n \\ k = 1, 2, \ldots, m \end{cases} \tag{10}$$

where D_k^l denotes the total distance between the i-th fire hawk,and the k-th prey; m is is the total number of prey in the search space;n is the total number of fire hawks in the search space; and $(x1, y1)$ and $(x2, y2)$ represent the coordinates of the fire hawk and prey, respectively, in the search space.

STEP 3: Fire hawks update their positions by simulating movement toward the global optimum (main fire source) and cooperating with other fire hawks, aiming to enhance global search capability. The commonly used update equation is:

$$\mathrm{FH}_l^{\mathrm{new}} = \mathrm{FH}_l + (r_1 \times \mathrm{GB} - r_2 \times \mathrm{FH}_{\mathrm{Near}}) \tag{11}$$

where FH_l^{new} represents the new position of the l-th fire hawk, r_1, r_2 are random weighting coefficients, and GB denotes the current global best solution. FH_{Near} represents the position of the nearest neighboring fire hawk.

STEP 4: Prey adopt different escape strategies based on the positions of the fire sources. On one hand, they migrate toward the fire hawk in their own territory and the "safe zone";

on the other hand, they may also move toward other territories or the global safe zone to enhance search diversity. The position update of prey can be described as follows:

$$\mathrm{PR}_q^{\mathrm{new}} = \mathrm{PR}_q + (r_3 \times \mathrm{FH}_l - r_4 \times \mathrm{SP}_l) \tag{12}$$

$$\mathrm{PR}_q^{\mathrm{new}} = \mathrm{PR}_q + (r_5 \times \mathrm{FH}_{\mathrm{Alter}} - r_6 \times \mathrm{SP}) \tag{13}$$

where PR_q^{new} denotes the new position of the q-th prey, SP_l is the center of the safe zone, and SP is the external safe point of the territory. r_3, r_4, r_5, r_6 are random weighting factors. The mathematical expressions for SP_l and SP are as follows:

$$\mathrm{SP}_l = \frac{\sum_{q=1}^{r} \mathrm{PR}_q}{r} \tag{14}$$

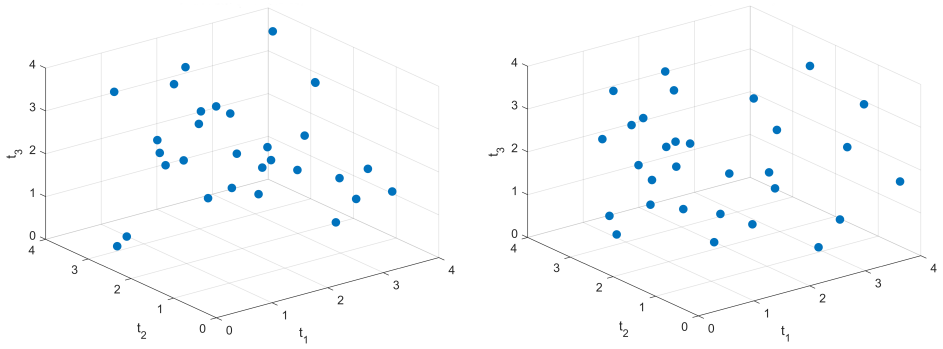
$$\mathrm{SP} = \frac{\sum_{k=1}^{m} \mathrm{PR}_k}{m} \tag{15}$$

where PR_q denotes the q-th prey encircled by the l-th fire hawk, and PR_k represents the k-th prey in the search space.

STEP 5: If a new solution exceeds the boundaries, a boundary correction mechanism is applied to project it back into the feasible region. The algorithm terminates when the maximum number of iterations is reached or the preset accuracy is achieved, and the optimal solution is output.

3.2. Improvements to the Fire Hawk Optimization Algorithm

In the basic Fire Hawk Optimization algorithm, the population is initialized by randomly generating new positions within the specified upper and lower bounds. In this paper, tent chaotic initialization is used as a replacement, and the distribution of particles in the solution space is illustrated reference the Figure 3 properly.



(**a**) (**b**)

Figure 3. Comparison between (**a**) random initialization and (**b**) tent chaotic initialization.

To evaluate the dispersion of particle spacing distribution, this paper introduces the coefficient of variation (CV) of pairwise distances as a measure of uniformity.

$$CV = \frac{\overline{s}_d}{\overline{d}} \tag{16}$$

where s_d is the standard deviation of the pairwise Euclidean distances between particles and $\overline{d}$ is the mean of the pairwise Euclidean distances between particles.

To eliminate data contingency, both initialization methods are configured with a population size of 150 and a dimension of 3. The experiment is repeated 30 times, and the resulting boxplot (Lower is best) is presented in the Figure 4.

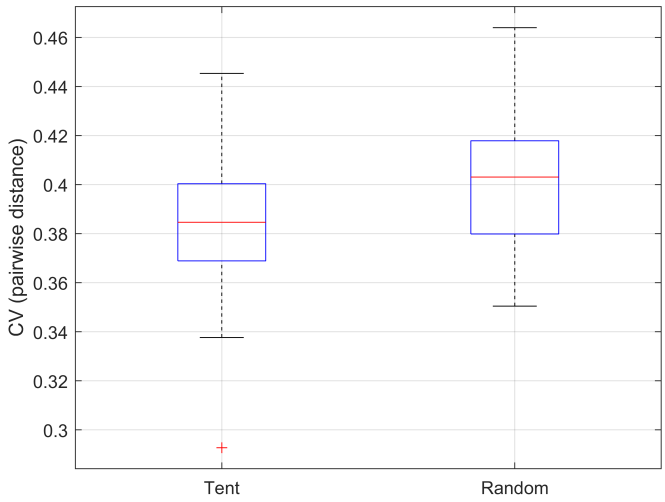


Figure 4. CV boxplot.

As shown in the Figures 3 and 4, particles generated by random initialization are more dispersed and exhibit a typical random scatter pattern, with local clustering observed in certain regions and the presence of voids near the coordinate axes. The corresponding coefficient of variation of pairwise distances is CV = 0.4039 ± 0.0321, indicating a high degree of dispersion among particle spacings. This may lead to incomplete population coverage and insufficient diversity in the early stages of the search, thereby increasing the risk of premature convergence to local optima. In contrast, particles generated by chaotic mapping display an overall "ring-shaped" enclosing structure, with significantly reduced void areas. The corresponding CV = 0.3846 ± 0.0322 is lower than that of the random method, indicating a more uniform distribution of particles. Therefore, the use of chaotic initialization enhances the global search capability of the algorithm and reduces the risk of being trapped in local optima, thus laying a foundation for subsequent convergence and optimization. The mathematical model is given by:

$$x_{n+1} = \begin{cases} \dfrac{x_n}{0.7} & 0 < x_n \leq 0.7 \\ \dfrac{x_n(1-x_n)}{0.3} & 0.7 < x_n \leq 1 \end{cases} \tag{17}$$

The model for tent chaotic mapping initialization is given by:

$$X_i^j(0) = x_{\min} + h \times (x_{\max} - x_{\min}), \quad i = 1, 2, \ldots, n, \quad j = 1, 2, \ldots, d \tag{18}$$

In the equation, $X_i^j(0)$ represents the initial value of the j-th decision variable of the i-th individual in the population, with an initial value of 0.7 in this paper; X_{min} and X_{max} are the minimum and maximum values of the variable range, respectively; n is the population size; d is the dimensionality of the problem; and h is the chaotic factor obtained through the mapping function.

To further improve the algorithm's global exploration and local exploitation capabilities at different search phases, this paper incorporates a hybrid mutation strategy based on chaos initialization. Specifically, Lévy flight, Gaussian mutation, and Cauchy muta-

tion mechanisms are introduced, and their weights are adaptively adjusted based on the evaluation function value (FEs). The specific procedure is as follows.

1. Lévy Flight Mechanism: Rapid Cross-Domain Jumps

Lévy flight exhibits a "frequent small steps, occasional large jumps" random walk characteristic, which can significantly enhance the scope of global exploration. The step size is defined by

$$\Delta x = \alpha \frac{u}{|v|^{1/\beta}}, \qquad u \sim \mathcal{N}(0, \sigma^2),\ v \sim \mathcal{N}(0,1) \tag{19}$$

where

$$\sigma = \left[\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma\left(\frac{1+\beta}{2}\right)\beta 2^{(\beta-1)/2}} \right]^{1/\beta}, \qquad \beta = 1.5 \tag{20}$$

The number of individuals employing the Lévy mutation, n_{Leavy} is adaptively adjusted according to the search phase as follows:

$$n_{\text{Levy}} = \begin{cases} \text{round}\left(0.3 n_{\text{Pop}}(1-p^2)\right) & \text{FEs} < \theta \\ \text{round}\left(0.05 n_{\text{Pop}}\right) & \text{FEs} \geq \theta \end{cases} \tag{21}$$

where $p = FE_s/\theta$,$\theta = 0.3MaxFes$ and the corresponding step size scaling factor is:

$$\text{levy}_{\text{scale}} = \begin{cases} 0.15e^{-3p} & \text{FEs} < \theta \\ 0.01 & \text{FEs} \geq \theta \end{cases} \tag{22}$$

2. Gaussian Mutation: Fine-Grained Local Search

When the algorithm approaches convergence, a more refined neighborhood search is required to further exploit the vicinity of the optimal solution. Therefore, Gaussian noise is added to all individuals:

$$x_{\text{new}} = x + s_{\text{G}}\mathcal{N}(0,1) \tag{23}$$

where the scaling factor for Gaussian noise is gradually increased in the later stages:

$$s_{\text{G}} = \begin{cases} 0.05 & \text{FEs} < \theta \\ 0.10(1-0.5p) & \text{FEs} \geq \theta \end{cases} \tag{24}$$

3. Cauchy Mutation: Enhanced Escape Mechanism

The Cauchy distribution has a heavier tail than the normal distribution, enabling individuals to perform extreme jumps and thereby escape from local minima. The mutation form is given by

$$x_{\text{new}} = x + s_{\text{C}}\mathcal{C}(0,1) \tag{25}$$

where

$$s_{\text{C}} = \begin{cases} 0.05 & \text{FEs} < \theta \\ 0.10(1-0.5p) & \text{FEs} \geq \theta \end{cases} \tag{26}$$

Specifically, the search process is divided into two phases, with the first 30% of function evaluations (FEs) dedicated to the global exploration phase, where Lévy mutation predominates and the scaling factors for Gaussian and Cauchy mutations are kept small. The remaining 70% corresponds to the local exploitation phase, in which Lévy mutation is largely suppressed, Gaussian mutation's scaling factor is gradually increased to enhance refined search, and Cauchy mutation maintains a moderate value (decreasing from 0.10 to

0.05) to ensure sporadic long jumps. This dual-phase strategy realizes a balanced transition from global exploration to refined local exploitation. The algorithm pseudocode (as shown in Table 1) and flowchart (as shown in Figure 5) are as follows:

Table 1. Pseudocode of TFHO algorithm.

Pseudocode of TFHO

Generate initial positions (X_i) via Tent chaotic mapping
Evaluate fitness values for X_i
Determine the Global Best (GB) solution
4. **while** FEs $<$ MaxFEs
Generate n to set the number of Fire Hawks
Determine Fire Hawks (FH) and Preys (PR)
Calculate total distance between FH and PR
Determine the territory of each FH by dispersing the PR
for $l = 1 : n$
Update the position of FH by Equation (13)
for $q = 1 : r$
Calculate the safe place under FH territory by Equation (16)
Update prey inside the territory by Equation (14)
Calculate the safe place outside by Equation (17)
Update prey outside the territory by Equation (15)
end for
end for
Add N Levy-flight particles (step = $levy_{scale}$)
Add N Gaussian-perturbed particles (step = s_G)
Add N Cauchy-perturbed particles (step = s_C)
Evaluate fitness values for all particles
Update the Global Best (GB) solution
end while
return GB
end procedure

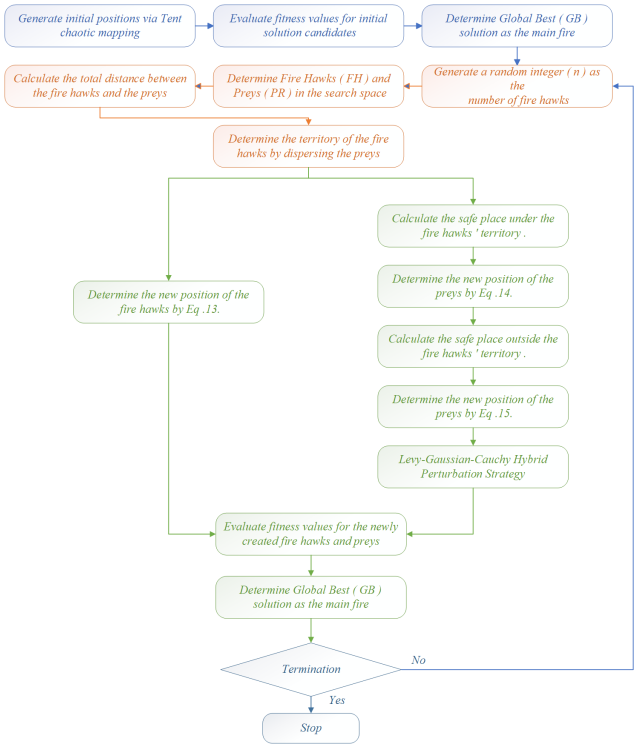


Figure 5. Flowchart of the TFHO algorithm.

4. Simulation and Analysis

4.1. Algorithm Testing and Analysis

The simulation environment consists of a 64-bit Windows 11 operating system, an Intel Core i7-13700H CPU, and MATLAB 2024b. To validate the optimization capability of the algorithm, PSO, WOA [27], GWO [28], FHO, and the proposed TFHO are compared using the nine benchmark test functions listed in Table 2.

Table 2. Benchmark test functions.

Function	Range
$\mathrm{F1}(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]$
$\mathbf{F2}(\mathbf{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]$
$\mathbf{F3}(\mathbf{x}) = \max_{1 \le i \le n} \lvert x_i \rvert$	$[-100, 100]$
$\mathbf{F4}(\mathbf{x}) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-30, 30]$
$\mathbf{F5}(\mathbf{x}) = \frac{\pi}{n}[A + B] + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$	$[-50, 50]$
$\mathrm{F6}(\mathbf{x}) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	$[-600, 600]$
$\mathbf{F7}(\mathbf{x}) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	$[-5.12, 5.12]$
$\mathrm{F8}(\mathbf{x}) = 0.1[\sin^2(3\pi x_1) + C + D] + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	$[-50, 50]$
$\mathrm{F9}(\mathbf{x}) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right]^{-1}$	$[-65.536, 65.536]$

To ensure the fairness of the experiments, the population size of all algorithms was set to 30, and the maximum number of iterations was set to 500. After thirty independent runs, the optimal value, mean, and standard deviation were calculated as performance metrics. The experimental results are presented in Table 3, and the convergence curves of the test functions are reference the Figure 6 properly.

Based on Table 3 and the convergence curves in Figure 6, it can be observed that the TFHO algorithm demonstrates significant overall performance advantages on all nine selected benchmark test functions. Specifically, the TFHO algorithm exhibits a faster and more stable convergence trend across most test functions. Its fitness value decreases significantly during the initial iterations and continues to improve in the later stages, ultimately achieving the lowest objective function values.

Taking typical unimodal functions such as F1 and F2 as examples, TFHO converges considerably faster than competing algorithms, including FHO and PSO, while achieving markedly higher final accuracy. The corresponding optimal and mean values reach magnitudes of 9.33×10^{-90} and 4.06×10^{-77}, respectively, with significantly lower standard deviations, highlighting the algorithm's excellent stability and robustness.

For complex multimodal functions, TFHO is able to effectively avoid premature convergence and maintain strong global search capability. As shown in the convergence curves, TFHO consistently demonstrates a leading advantage throughout the optimization process on these functions, and its final fitness values are significantly lower than those of PSO, WOA, and even FHO and GWO. The tabulated results further support this conclusion.

For example, on the F5 function, TFHO achieves the best optimal value of 3.16×10^{-4} and a mean value of 1.26×10^{-3}, both superior to those of the compared algorithms. For certain functions (such as F6 and F7), TFHO and other advanced algorithms can all converge to the theoretical optimal solution; however, the results obtained by TFHO are more concentrated, with standard deviations approaching zero, which further demonstrates its stability and robustness.

Based on the experimental results on standard benchmark functions, it can be observed that the proposed improved algorithm (TFHO) demonstrates superior convergence speed and solution accuracy compared to the original FHO and other algorithms in most test cases. Moreover, the results are more concentrated, indicating enhanced stability. Overall, the improved algorithm achieves better performance across different types of optimization problems, providing methodological support and a theoretical foundation for its subsequent application in trajectory planning.

Table 3. Results of benchmark function tests.

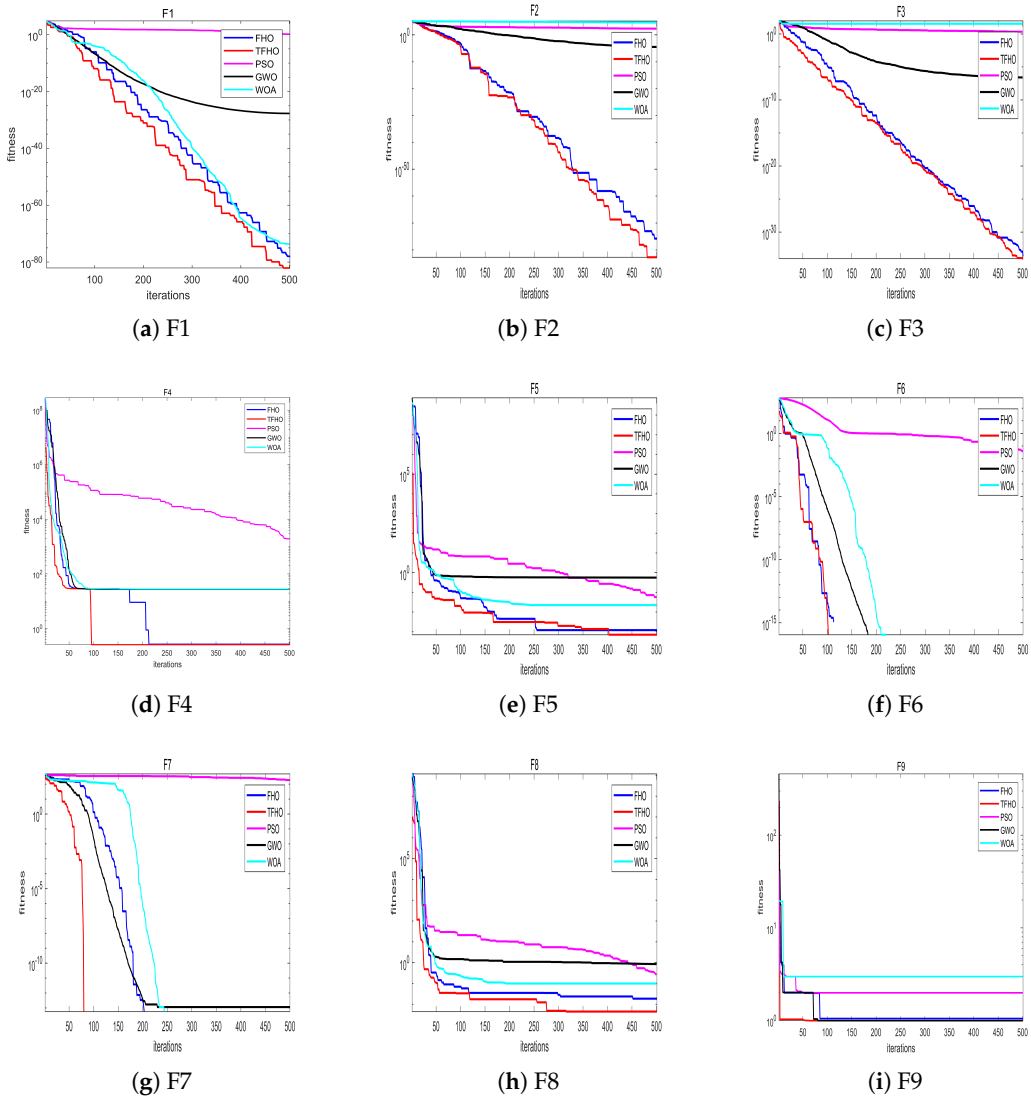
Function	Indicator	TFHO	FHO	PSO	GWO	WOA
F1	BEST	9.3332×10^{-90}	8.3170×10^{-86}	7.5761×10^{-01}	4.4857×10^{-29}	5.3442×10^{-88}
	MEAN	4.0586×10^{-77}	7.6274×10^{-71}	$2.6402 \times 10^{+04}$	$9.0575 \times 10^{+81}$	$1.1232 \times 10^{+73}$
	STD	1.8249×10^{-76}	4.1777×10^{-70}	$1.2790 \times 10^{+04}$	$2.3558 \times 10^{+81}$	$5.6087 \times 10^{+73}$
F2	BEST	5.5837×10^{-87}	1.6638×10^{-85}	9.5425×10^{-01}	3.9896×10^{-31}	$4.4092 \times 10^{+03}$
	MEAN	9.0182×10^{-72}	6.0345×10^{-74}	$1.8226 \times 10^{+01}$	1.1951×10^{-05}	$4.0457 \times 10^{+04}$
	STD	4.9101×10^{-71}	3.2322×10^{-73}	$5.4329 \times 10^{+01}$	4.2108×10^{-05}	$1.4234 \times 10^{+04}$
F3	BEST	2.0503×10^{-39}	1.5562×10^{-36}	$1.5749 \times 10^{+00}$	6.8942×10^{-08}	$2.2621 \times 10^{+03}$
	MEAN	1.7399×10^{-32}	5.8647×10^{-32}	$2.0354 \times 10^{+04}$	$8.3788 \times 10^{+07}$	$5.0796 \times 10^{+01}$
	STD	9.1200×10^{-32}	2.3642×10^{-31}	$2.1985 \times 10^{+04}$	$1.2778 \times 10^{+06}$	$2.9675 \times 10^{+01}$
F4	BEST	3.7080×10^{-40}	6.0413×10^{-37}	$1.3589 \times 10^{+04}$	9.6867×10^{-08}	$3.4166 \times 10^{+01}$
	MEAN	2.8324×10^{-33}	1.1187×10^{-31}	$1.9498 \times 10^{+04}$	$8.2158 \times 10^{+07}$	$4.4158 \times 10^{+01}$
	STD	9.9168×10^{-33}	3.4271×10^{-31}	$2.9817 \times 10^{+04}$	6.7533×10^{-07}	$3.0512 \times 10^{+01}$
F5	BEST	3.1683×10^{-04}	4.2390×10^{-04}	5.3343×10^{-03}	1.9664×10^{-02}	3.3214×10^{-03}
	MEAN	1.2649×10^{-03}	1.5212×10^{-03}	3.6996×10^{-02}	4.8623×10^{-02}	3.2024×10^{-02}
	STD	7.0899×10^{-04}	7.4028×10^{-04}	3.2472×10^{-02}	2.6650×10^{-02}	3.6776×10^{-02}
F6	BEST	$0.0000 \times 10^{+00}$	$0.0000 \times 10^{+00}$	6.4671×10^{-02}	$0.0000 \times 10^{+00}$	$0.0000 \times 10^{+00}$
	MEAN	$0.0000 \times 10^{+00}$	$0.0000 \times 10^{+00}$	1.3247×10^{-01}	5.4419×10^{-03}	4.1169×10^{-03}
	STD	$0.0000 \times 10^{+00}$	$0.0000 \times 10^{+00}$	4.5195×10^{-02}	9.1032×10^{-03}	2.2549×10^{-02}
F7	BEST	$0.0000 \times 10^{+00}$	$0.0000 \times 10^{+00}$	$1.0548 \times 10^{+02}$	5.6843×10^{-14}	$0.0000 \times 10^{+00}$
	MEAN	$0.0000 \times 10^{+00}$	$0.0000 \times 10^{+00}$	$1.6379 \times 10^{+02}$	$2.8309 \times 10^{+00}$	1.8948×10^{-15}
	STD	$0.0000 \times 10^{+00}$	$0.0000 \times 10^{+00}$	$3.1140 \times 10^{+01}$	$4.1360 \times 10^{+00}$	1.0378×10^{-14}
F8	BEST	1.7785×10^{-03}	2.3278×10^{-03}	1.3893×10^{-01}	9.9800×10^{-01}	$1.0000 \times 10^{+00}$
	MEAN	9.4593×10^{-03}	9.6664×10^{-03}	5.4589×10^{-01}	7.0111×10^{-01}	4.9766×10^{-01}
	STD	4.4443×10^{-03}	5.4414×10^{-03}	2.4837×10^{-02}	2.6356×10^{-01}	2.3741×10^{-01}
F9	BEST	9.9800×10^{-01}	9.9800×10^{-01}	9.9800×10^{-01}	9.9800×10^{-01}	9.9800×10^{-01}
	MEAN	$1.3951 \times 10^{+00}$	$1.4612 \times 10^{+00}$	$3.5278 \times 10^{+00}$	$4.4567 \times 10^{+00}$	$3.5463 \times 10^{+00}$
	STD	7.5542×10^{-01}	7.5262×10^{-01}	$2.6378 \times 10^{+00}$	$4.2713 \times 10^{+00}$	$3.7762 \times 10^{+00}$

To further highlight the superiority of the improved algorithm over other algorithms, the area under the convergence curve (AUC) of the aforementioned test functions is selected as a statistical indicator. A smaller AUC indicates faster and more efficient convergence because it reflects a lower cumulative optimization error across iterations. As summarized in Table 4, TFHO attains the lowest mean AUC (7.819×10^2), outperforming FHO

(1.1406×10^3), PSO (1.8404×10^3), GWO (2.0234×10^3), and WOA (2.5798×10^3). In relative terms, TFHO reduces the mean AUC by 31.4% versus FHO, 57.5% versus PSO, 61.4% versus GWO, and 69.7% versus WOA, evidencing both higher convergence speed and better optimization efficiency.

Table 4. Comparison of algorithms based on AUC (mean values).

	FHO	TFHO	PSO	GWO	WOA
Mean	1.1406×10^3	7.8190×10^2	1.8404×10^3	2.0234×10^3	2.5798×10^3



(**a**) F1 (**b**) F2 (**c**) F3

(**d**) F4 (**e**) F5 (**f**) F6

(**g**) F7 (**h**) F8 (**i**) F9

Figure 6. Convergence curves of different algorithms on F1–F9 benchmark functions.

The box-plot in Figure 7 further corroborates these results. TFHO exhibits the lowest median and the tightest interquartile range, indicating markedly smaller run-to-run variability. In contrast, PSO and GWO display broader IQRs with long upper tails, while WOA shows the largest dispersion and multiple extreme outliers, pointing to unstable convergence behavior. FHO performs moderately but still trails TFHO in both central tendency and robustness. Collectively, the distributional evidence (box-plot) and the central tendency statistics (Table 4) demonstrate that TFHO converges faster and more stably than the competing methods across repeated runs.

Furthermore, to rigorously validate whether the differences in AUC between TFHO and the other algorithms are statistically significant, the Wilcoxon signed-rank test Appendix B was performed, and the results are presented in Table 5. All comparisons yield p-values below the 0.05 significance level, confirming that the performance advantage of TFHO over FHO, PSO, GWO, and WOA is statistically significant. In particular, the most pronounced differences are observed against PSO (Z = −4.103) and WOA (Z = −3.918), indicating that TFHO achieves a markedly superior convergence profile relative to these methods. These statistical findings are consistent with the earlier AUC analysis, thereby reinforcing the conclusion that TFHO delivers both faster convergence and more stable optimization performance across repeated trials.

Table 5. Statistical comparison of TFHO against other algorithms.

Comparison	p-Value	Z-Score	Significance
TFHO vs. FHO	4.95×10^{-2}	−1.964	Significant
TFHO vs. PSO	4.072×10^{-5}	−4.103	Significant
TFHO vs. GWO	6.836×10^{-3}	−2.705	Significant
TFHO vs. WOA	8.919×10^{-5}	−3.918	Significant

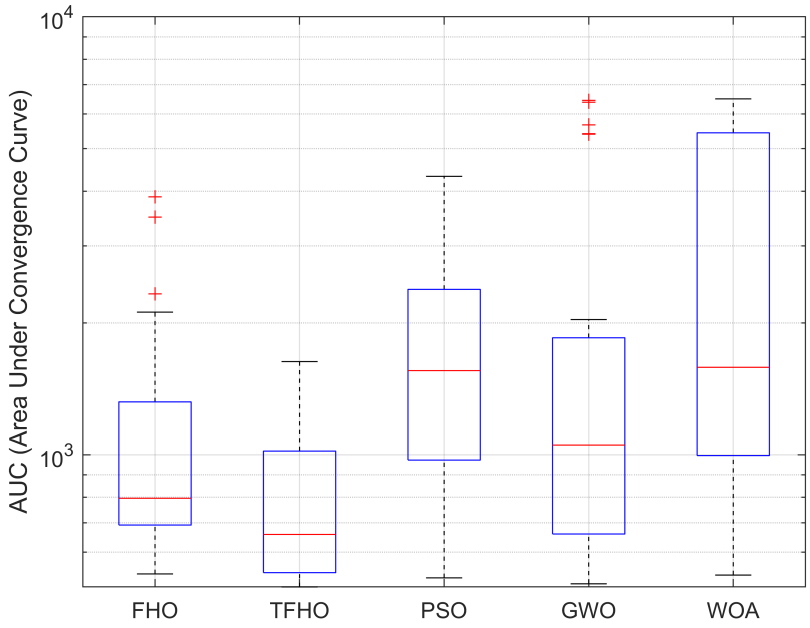


Figure 7. AUC boxplot.

4.2. Trajectory Planning

The link parameters of the ABB IRB-2600 robot are listed in Table 6. According to the aforementioned kinematic principles, the joint angles at each interpolation point corresponding to the specified path points (Table 7) are determined via inverse kinematics, and summarized in Table 8. Utilizing the 3-5-3 polynomial interpolation method described in Section 2, and taking time as the optimization variable, the proposed algorithm is employed to generate the trajectory under a maximum joint velocity constraint of 80°/s. The algorithm proceeds as follows:

1. Initialize the three-dimensional variable particles t_1, t_2, t_3.
2. Substitute the time variables into Equations (5)–(7) to compute the coefficient matrix.
3. Check the velocity constraints by substituting the coefficients into the 3-5-3 polynomial.
4. Apply a large penalty if constraints are violated and proceed to the next iteration.
5. Repeat the process until the algorithm terminates.

To validate the effectiveness of the proposed improvements, ablation experiments are performed using joint 1 as an example. Here, the maximum number of function evaluations is set to 15,000, the population size is 30, and the initial particles are randomly distributed within $[0.1, 4]$. The convergence curves of the basic FHO, FHO with Tent chaotic initialization, and the complete TFHO are shown in Figure 8, while the best, mean, and standard deviation from thirty independent runs are summarized in Table 9.

The results indicate that, with the introduction of the Tent mechanism, the convergence speed of the basic FHO is enhanced, the mean solution value is significantly reduced, and the standard deviation drops to 0.1335. The optimal value also improves, demonstrating that chaotic mapping increases both the global search capability and stability of the algorithm. With the further integration of the three-layer adaptive perturbation mechanism, TFHO not only achieves better solutions at a faster rate, but also exhibits minimal fluctuation, with the standard deviation reduced to only 0.0023. This confirms the improved algorithm's superior global search ability and stability, thereby validating the effectiveness of the enhancements.

Comprehensive analysis shows that TFHO not only converges more rapidly, but also yields optimal fitness for the final trajectory solution, surpassing other swarm intelligence algorithms on multiple benchmark functions. Moreover, its convergence process is smoother, further underscoring its robustness and engineering applicability to complex optimization problems. The above results confirm that the improvement strategies adopted in TFHO significantly enhance both convergence performance and optimization accuracy.

Table 6. DH parameters of the ABB IRB-2600 robot.

Joint No.	θ_i	d_i/mm	a_i/mm	$a_i/(^\circ)$
1	θ_1	445	0	0
2	θ_2	0	150	−90
3	θ_3	0	−700	0
4	θ_4	795	−115	90
5	θ_5	0	0	−90
6	θ_6	85	0	90

Table 7. Cartesian space path points of the robot (mm).

Point 1	Point 2	Point 3	Point 4
(1489.0, −297.8, 1251.1)	(1557.3, 334.9, 1108.8)	(−299.0, 1489.4, −360.1)	(−1169.9, 675.4, −594.4)

Table 8. Joint space angles of the robot (rad).

	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
Point 1	−0.1974	2.6103	−1.4271	−0.0014	−0.1634	1.2424
Point 2	0.2113	2.7113	−1.4272	−0.0608	−0.1603	1.8252
Point 3	1.7682	3.6742	−1.4272	−0.0764	−0.1623	0.5747
Point 4	2.6180	3.8563	−1.4272	−0.0005	−0.1668	1.5479

Table 9. Ablation study results (s).

	MEAN	STD	BEST
Only tent	4.305694	0.1335	4.138821
FHO	4.407033	0.3157	4.140935
TFHO	4.133064	0.0023	4.129682

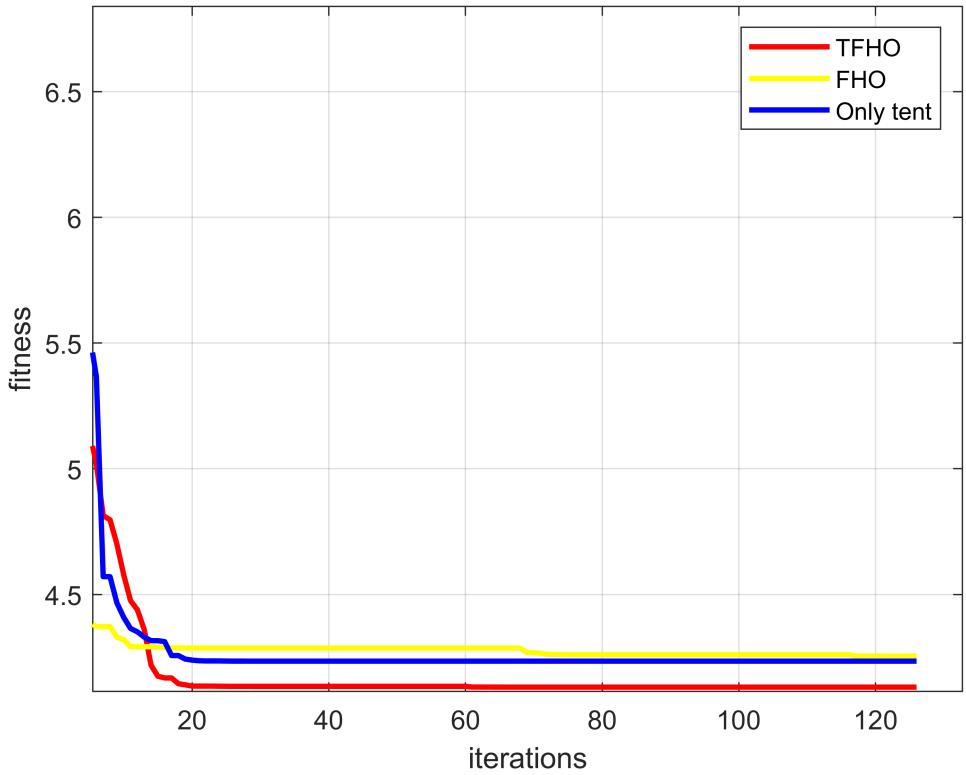


Figure 8. Comparison of convergence curves.

By applying the above method to the optimization of all robot joints, the optimal interpolation time for each joint is obtained, as summarized in Table 10. Since all joints operate synchronously, the maximum interpolation time among all joints is selected to ensure that each joint reaches the target position simultaneously. Consequently, the total operating time of the robot using the TFHO algorithm is $T = 9.88$ s. In comparison, manual interpolation without the algorithm requires 12 s. Therefore, the trajectory planning time is reduced by 17.66% with the improved TFHO algorithm.

Table 10. Minimum time for each joint (s).

	t_1	t_2	t_3
joint1	1.0149	1.1845	1.9336
joint2	0.2990	0.7421	0.4701
joint3	0.1000	0.1000	0.1000
joint4	0.1816	0.2231	0.2449
joint5	0.1000	0.1000	0.1000
joint6	2.6945	3.7020	3.4843

As shown in Figure 9, the motion trajectory curves of all six robot joints, generated by the proposed algorithm in conjunction with the 3-5-3 polynomial, exhibit excellent continuity and smoothness in displacement, velocity, and acceleration. The top panel depicts the temporal evolution of each joint's position, where all joint angles transition smoothly from their initial to target values, satisfying the prescribed parameter requirements and exhibiting no abrupt changes throughout the motion.

The middle panel presents the joint velocity profiles over time. All angular velocity curves are continuous, without any discontinuities or sudden jumps, and their extrema remain within reasonable bounds. The bottom panel illustrates the joint accelerations, which are likewise continuous and free from any excessive fluctuations. Throughout the

entire trajectory, the angular position, velocity, and acceleration of each joint maintain smooth transitions, confirming the physical feasibility and practicality of the planned path.

Figure 10 shows the end-effector trajectory after optimizing the 3-5-3 polynomial parameters using the TFHO algorithm. The results indicate that the end-effector passes smoothly and precisely through all specified interpolation points, and the overall trajectory is continuous and smooth. These findings further validate the effectiveness and practical applicability of the proposed trajectory optimization method.

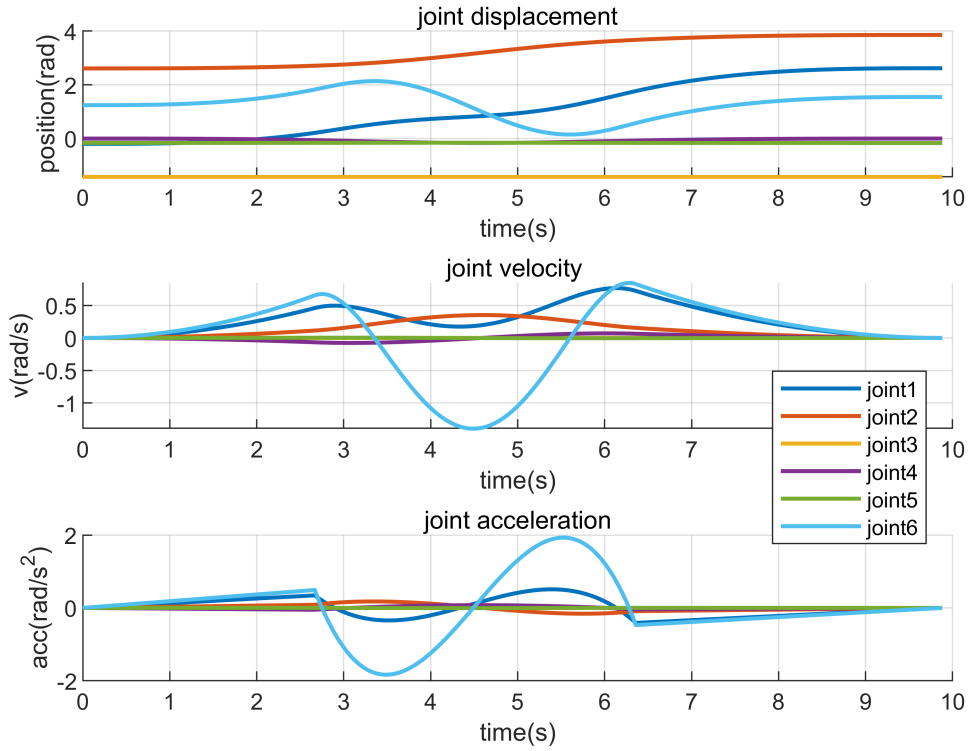


Figure 9. Joint trajectory curves of the robot.

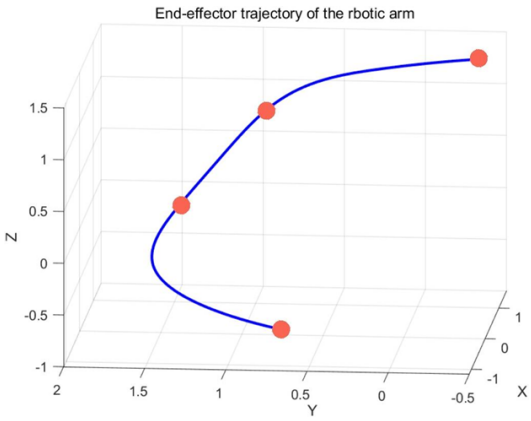


Figure 10. End-effector trajectory of the robot.

5. Conclusions

A joint-space time-optimal trajectory planning paradigm has been established by coupling 3–5–3 piecewise polynomial parameterization with an improved Fire Hawk Optimization (TFHO). The optimizer incorporates Tent-chaotic initialization and a two-phase adaptive Lévy–Gaussian–Cauchy hybrid mutation, enabling diversity preservation in early iterations and refined local search in later iterations, thereby suppressing premature convergence and improving stability. On nine benchmark functions, the approach attains

the lowest mean area under the convergence curve (AUC) and statistically significant Wilcoxon signed-rank results against representative metaheuristics (FHO, PSO, GWO, WOA), reflecting consistently faster convergence and higher solution quality; in relative terms, the mean AUC is reduced by 31.4% (vs. FHO), 57.5% (vs. PSO), 61.4% (vs. GWO), and 69.7% (vs. WOA).

In the ABB IRB-2600 case, time-optimal segment durations computed under a joint-velocity bound of 80°/s yield smooth, continuous position/velocity/acceleration profiles and reduce the total execution time from 12 s to 9.88 s (a reduction of 17.66%), indicating feasibility for engineering deployment under standard kinematic constraints. An ablation analysis indicates that, relative to the basic FHO and the Tent-only variant, TFHO reduces the mean segment time (illustrative joint) from 4.407033 s and 4.305694 s to 4.133064 s (by 0.273969 s, 6.22% and by 0.172630 s, 4.01%, respectively), improves the best value from 4.140935 s/4.138821 s to 4.129682 s (0.27%/0.22%), and suppresses run-to-run variability as the standard deviation drops from 0.3157/0.1335 to 0.0023 (99.27%/98.28%), confirming that the hybrid mutation mechanism contributes to both accuracy and robustness.

The proposed pipeline provides a unified route from kinematic modeling and 3–5–3 trajectory construction to constrained time allocation via TFHO, offering a practical template for industrial scenarios where smoothness and time optimality must be balanced. Notwithstanding these advantages, current evidence is limited to simulation and remains sensitive to modeling accuracy, constraint tuning, and parameter settings. Future work will emphasize (i) validation on physical manipulators with closed-loop execution; (ii) incorporation of additional practical constraints such as joint torque/jerk limits, actuator-rate bounds, and collision margins; (iii) extension from single-task runs to multi-task or multi-segment missions with synchronized multi-joint timing; (iv) multi-objective formulations trading off time, energy, and smoothness with Pareto-front analysis.

Author Contributions: conceptualization, S.Y. and B.J.; investigation, L.C.; methodology, B.J.; software, Y.Z.; validation, L.Q. and S.F.; writing—original draft, B.J. All authors have read and agreed to the published version of the manuscript.

Funding: Zhenjiang Science and Technology Program (grant number: JC2024021).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within this article.

Conflicts of Interest: Author Yongwei Zhang and Liang Qi were employed by the company Jiangsu Shipbuilding and Ocean Engineering Design and Research Institute. The remaining 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.

Nomenclature

The following abbreviations are used in this manuscript:

A	$10\sin^2\left(\pi\left(1+\frac{x_1+1}{4}\right)\right)$
B	$\sum_{i=1}^{n-1}\left(\frac{x_i+1}{4}\right)^2\left(1+10\sin^2\left(\pi\left(1+\frac{x_{i+1}+1}{4}\right)\right)\right)+\left(\frac{x_n+1}{4}\right)^2$
C	$\sum_{i=1}^{n-1}(x_i-1)^2(1+\sin^2(3\pi x_{i+1}))$
D	$(x_n-1)^2(1+\sin^2(2\pi x_n))$
x	Current individual .
x_{new}	Individual after mutation.
Δx	Lévy-flight step added to x.

$\alpha > 0$	Scale factor for the Lévy step.
$u \sim \mathcal{N}(0, \sigma^2), v \sim \mathcal{N}(0,1)$	Gaussian random variables
$\lvert v \rvert$	Absolute value (component-wise when v is a vector).
β	Stability parameter of the Lévy distribution (here $\beta = 1.5$).
σ	Scale parameter computed from β used to sample u.
$\Gamma(\cdot)$	Gamma function; $\sin(\cdot)$ is sine; π is the circle constant.
n_{Levy}	Number of individuals using Lévy mutation in the current iteration.
n_{Pop}	Population size.
FEs	Number of objective function evaluations used so far.
MaxFes	Maximum allowed function evaluations.
θ	Phase-switch threshold.
p	Normalized progress in the early phase .
$\text{levy}_{\text{scale}}$	Additional scale for the Lévy step .
e	Base of the natural logarithm.
$\mathcal{N}(0,1)$	Standard normal distribution.
s_{G}	Scale factor for Gaussian mutation.
s_{C}	Scale factor for Cauchy mutation.
$\mathcal{C}(0,1)$	Standard Cauchy distribution (location 0, scale 1).

Appendix A. Derivation of the 3-5-3 Polynomial Trajectory

Appendix A.1. Trajectory Structure and Segmentation

The 3-5-3 polynomial is a commonly used segmented interpolation method, consisting of three polynomial segments:

1. Acceleration phase (cubic polynomial, $0 \le t \le t_1$);
2. Constant velocity phase (quintic polynomial, $t_1 \le t \le t_1 + t_2$);
3. Deceleration phase (cubic polynomial, $t_1 + t_2 \le t \le T$).

Each segment satisfies continuity in position, velocity, and acceleration.

Appendix A.2. Polynomial Expressions

- Acceleration phase:

$$q_1(t) = a_{10} + a_{11}t + a_{12}t^2 + a_{13}t^3 \tag{A1}$$

- Constant velocity phase:

$$q_2(t) = a_{20} + a_{21}t + a_{22}t^2 + a_{23}t^3 + a_{24}t^4 + a_{25}t^5 \tag{A2}$$

- Deceleration phase:

$$q_3(t) = a_{30} + a_{31}t + a_{32}t^2 + a_{33}t^3 \tag{A3}$$

Appendix A.3. Boundary Conditions

1. At $t = 0$ (start point):

$$q_1(0) = q_s, \quad \dot{q}_1(0) = 0, \quad \ddot{q}_1(0) = 0 \tag{A4}$$

2. At $t = t_1$ (acceleration $\rightarrow$ constant velocity):

$$q_1(t_1) = q_2(t_1), \quad \dot{q}_1(t_1) = \dot{q}_2(t_1), \quad \ddot{q}_1(t_1) = \ddot{q}_2(t_1) \tag{A5}$$

3. At $t = t_1 + t_2$ (constant velocity $\rightarrow$ deceleration):

$$q_2(t_1 + t_2) = q_3(t_1 + t_2), \quad \dot{q}_2(t_1 + t_2) = \dot{q}_3(t_1 + t_2), \quad \ddot{q}_2(t_1 + t_2) = \ddot{q}_3(t_1 + t_2) \tag{A6}$$

4. At $t = T$ (end point):

$$q_3(T) = q_e, \quad \dot{q}_3(T) = 0, \quad \ddot{q}_3(T) = 0 \tag{A7}$$

Appendix A.4. System of Equations and Coefficient Matrix A

Substituting the above conditions into the polynomial and derivative equations yields a system of linear equations with 14 unknown coefficients:

$$Aa = \theta \tag{A8}$$

where

- A is the 14×14 coefficient matrix, with each row corresponding to a boundary or continuity condition and each column corresponding to a polynomial coefficient a_{ij};
- **a** is the vector of unknown coefficients;
- θ is the vector of boundary condition values determined by q_s, q_e, t_1, t_2, T.

Appendix A.5. Solution of the Coefficients

The polynomial coefficients can be obtained by solving:

$$a = A^{-1}\theta \tag{A9}$$

This yields the complete 3-5-3 trajectory.

Appendix B. Wilcoxon Signed-Rank Test

The Wilcoxon signed-rank test is a non-parametric statistical method used to compare two related or paired samples in order to determine whether their population mean ranks differ. Unlike the paired t-test, it does not require the data to be normally distributed, making it suitable for data with skewed distributions or outliers.

Appendix B.1. Procedure

Given two sets of paired observations, (x_i, y_i) for $i = 1, \ldots, n$, the Wilcoxon signed-rank test proceeds as follows:

1. Compute the difference for each pair:

$$d_i = x_i - y_i \tag{A10}$$

2. Remove pairs where $d_i = 0$.
3. Rank the remaining $|d_i|$ values from smallest to largest, assigning average ranks in case of ties.
4. Assign the sign of d_i to its corresponding rank.
5. Calculate:

$$W^+ = \text{sum of positive ranks}, \quad W^- = \text{sum of negative ranks} \tag{A11}$$

6. The test statistic W is the smaller of W^+ and W^-.
7. For large n, approximate with a normal distribution:

$$Z = \frac{W - \frac{n(n+1)}{4}}{\sqrt{\frac{n(n+1)(2n+1)}{24}}} \tag{A12}$$

Appendix B.2. Interpretation

If the p-value is less than the chosen significance level (e.g., $\alpha = 0.05$), we reject the null hypothesis that there is no difference between the two paired samples. A negative Z-score indicates that the second sample tends to have larger values than the first, while a positive Z-score indicates the opposite.

Appendix B.3. Application in This Study

In this work, the Wilcoxon signed-rank test was applied to compare the convergence performance (measured by AUC) of the proposed TFHO algorithm with each of the benchmark algorithms. All comparisons yielded p-values below 0.05, indicating statistically significant performance differences.

References

1. Lee, J.; Cho, B.K. Development of a DDP-based trajectory generation method considering the manipulability measure for 6-DOF collaborative robots. *J. Comput. Des. Eng.* **2025**, *12*, 98–114. [CrossRef]
2. Hu, H.; Wen, X.; Hu, J.; Chen, H.; Xia, C.; Zhang, H. Safe Trajectory Planning for Incremental Robots Based on a Spatiotemporal Variable-Step-Size A* Algorithm. *Sensors* **2024**, *24*, 3639. [CrossRef]
3. He, W.; Zhang, P.; Guo, K.; Sun, J.; Sivalingam, V.; Huang, X. Kinematic calibration and compensation of industrial robots based on extended joint space. *IEEE Access* **2023**, *11*, 109331–109340. [CrossRef]
4. Zhou, Y.; Han, G.; Wei, Z.; Huang, Z.; Chen, X.; Wu, J. Optimal trajectory planning of robot energy consumption based on improved sparrow search algorithm. *Meas. Control* **2024**, *57*, 1014–1021. [CrossRef]
5. Oelerich, T.; Beck, F.; Hartl-Nesic, C.; Kugi, A. Boundmpc: Cartesian trajectory planning with error bounds based on model predictive control in the joint space. *arXiv* **2024**, arXiv:2401.05057. [CrossRef]
6. Luo, L.; Guo, T.; Cui, K.; Zhang, Q. Trajectory planning in robot joint space based on improved quantum particle swarm optimization algorithm. *Appl. Sci.* **2023**, *13*, 7031. [CrossRef]
7. Li, J.; Shi, X.; Liang, P.; Li, Y.; Lv, Y.; Zhong, M.; Han, Z. Research on Gait Trajectory Planning of Wall-Climbing Robot Based on Improved PSO Algorithm. *J. Bionic Eng.* **2024**, *21*, 1747–1760. [CrossRef]
8. Zhang, T.; Zhang, M.; Zou, Y. Time-optimal and smooth trajectory planning for robot manipulators. *Int. J. Control. Autom. Syst.* **2021**, *19*, 521–531. [CrossRef]
9. Zhang, Z.; He, D.; Tang, L.; Meng, L. Picking robot arm trajectory planning method. *Sens. Transducers* **2014**, *162*, 11.
10. Mohammed, M.; Muhammad, F.; Bazli, B.M.; Ali, S. Comparative study between quintic and cubic polynomial equations based walking trajectory of exoskeleton system. *Int. J. Mech. Mechatronics Eng.* **2017**, *17*, 43–51.
11. Koch, P.E.; Wang, K. Introduction of B-splines to trajectory planning for robot manipulators. *Model. Identif. Control* **1988**, *9*, 69–80 [CrossRef]
12. Li, X.; Zhao, H.; He, X.; Ding, H. A novel cartesian trajectory planning method by using triple NURBS curves for industrial robots. *Robot. Comput. Integr. Manuf.* **2023**, *83*, 102576. [CrossRef]
13. Feng, J.C.; Lu, W.Q.; Lin, J.H.; Lu, Y.J.; Cen, G.J. Six-axis Robot Trajectory Optimization Based on 3-5-3 Polynomial Interpolation. *Mach. Electron.* **2023**, *41*, 75–80. (In Chinese)
14. Zhang, X.; Xiao, F.; Tong, X.; Yun, J.; Liu, Y.; Sun, Y.; Tao, B.; Kong, J.; Xu, M.; Chen, B. Time optimal trajectory planing based on improved sparrow search algorithm. *Front. Bioeng. Biotechnol.* **2022**, *10*, 852408. [CrossRef] [PubMed]
15. Zhang, H.D.; Liu, S.B.; Lei, Q.J.; He, Y.; Yang, Y.; Bai, Y. Robot programming by demonstration: A novel system for robot trajectory programming based on robot operating system. *Adv. Manuf.* **2020**, *8*, 216–229. [CrossRef]
16. Chen, G.; Wei, N.; Yan, L.; Lu, H.; Li, J. Time-optimal trajectory planning based on event-trigger and conditional proportional control. *PLoS ONE* **2023**, *18*, e0273640. [CrossRef]
17. Nadir, B.; Mohammed, O.; Minh-Tuan, N.; Abderrezak, S. Optimal trajectory generation method to find a smooth robot joint trajectory based on multiquadric radial basis functions. *Int. J. Adv. Manuf. Technol.* **2022**, *120*, 297–312. [CrossRef]
18. Yu, X.; Dong, M.; Yin, W. Time-optimal trajectory planning of manipulator with simultaneously searching the optimal path. *Comput. Commun.* **2022**, *181*, 446–453. [CrossRef]
19. Fu, R.; Ju, H. Time-Optimal Trajectory Planning Algorithm for Robotic Manipulator Based on Particle Swarm Optimization. *Inf. Control* **2011**, *40*, 802–808. (In Chinese)
20. Liu, H.Q.; Chang, Z.Y.; Wang, J.W.; Liu, Y.; Chu, H.Y.; Zhang, X.Q. Research on Robotic Arm Trajectory Planning Algorithm Based on Improved Genetic Particle Swarm Optimization. *Manuf. Technol. Mach. Tools* **2024**, *12*, 13–20. (In Chinese) [CrossRef]

21. Gai, R.; Wang, K.; Wang, X. Robotic Manipulator Trajectory Planning Based on Chaotic Lévy Particle Swarm Optimization. *J. Mod. Mach. Tools Autom. Manuf. Technol.* **2025**, *5*, 101–105. (In Chinese)
22. Wang, W.; Tao, Q.; Cao, Y.; Wang, X.; Zhang, X. Robot time-optimal trajectory planning based on improved cuckoo search algorithm. *IEEE Access* **2020**, *8*, 86923–86933. [CrossRef]
23. Azizi, M.; Talatahari, S.; Gandomi, A.H. Fire Hawk Optimizer: A novel metaheuristic algorithm. *Artif. Intell. Rev.* **2023**, *56*, 287–363. [CrossRef]
24. Ashraf, A.; Anwaar, A.; Haider Bangyal, W.; Shakir, R.; Ur Rehman, N.; Qingjie, Z. An improved fire hawks optimizer for function optimization. In Proceedings of the International Conference on Swarm Intelligence, Shenzhen, China, 14–18 July 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 68–79.
25. Said, M.; Ismaeel, A.A.; El-Rifaie, A.M.; Hashim, F.A.; Bouaouda, A.; Hassan, A.Y.; Abdelaziz, A.Y.; Houssein, E.H. Evaluation of modified fire hawk optimizer for new modification in double diode solar cell model. *Sci. Rep.* **2024**, *14*, 30079. [CrossRef] [PubMed]
26. Pan, J.; Fu, Z.; Xiong, J.; Lei, X.; Zhang, K.; Chen, X. RobMach: G-Code-based off-line programming for robotic machining trajectory generation. *Int. J. Adv. Manuf. Technol.* **2022**, *118*, 2497–2511. [CrossRef]
27. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
28. Mirjalili, S.; Mirjalili, S.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]