




## Article

# Industrial Robot Trajectory Optimization Based on Improved Sparrow Search Algorithm

Fei Ma <sup>1</sup>, Weiwei Sun <sup>1</sup>, Zhouxiang Jiang <sup>1</sup>, Shuangfu Suo <sup>2</sup>, Xiao Wang <sup>1</sup> and Yue Liu <sup>1,\*</sup>

<sup>1</sup> Mechanical Electrical Engineering School, Beijing Information Science & Technology University, Beijing 100192, China; mf@bistu.edu.cn (F.M.); sww@bistu.edu.cn (W.S.)

<sup>2</sup> State Key Laboratory of Tribology, Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China

\* Correspondence: yliu@bistu.edu.cn

**Abstract:** This paper proposes an enhanced multi-strategy sparrow search algorithm to optimize the trajectory of a six-axis industrial robot, addressing issues of low efficiency and high vibration impact on joints during operation. Initially, the improved D-H parametric method is employed to establish both forward and inverse kinematic models of the robot. Subsequently, a 3-5-3 mixed polynomial interpolation trajectory planning approach is applied to the robot. Building upon the conventional sparrow algorithm, a two-dimensional Logistic chaotic system initializes the population. Additionally, a Levy flight strategy and nonlinear adaptive weighting are introduced to refine the discoverer position update operator, while an inverse learning strategy enhances the vigilante position update operator. These modifications boost both the local and global search capabilities of the algorithm. The improved sparrow algorithm, based on 3-5-3 hybrid polynomial trajectory planning, is then used for the time-optimal trajectory planning of the robot. This is compared with traditional sparrow search algorithm and particle swarm algorithm optimization results. The findings indicate that the proposed enhanced sparrow search algorithm outperforms both the standard sparrow algorithm and the particle swarm algorithm in terms of convergence speed and accuracy for robot trajectory optimization. This can lead to the increased work efficiency and performance of the robot.

**Keywords:** industry 4.0; industrial robot; trajectory optimization; improved sparrow search algorithm



**Citation:** Ma, F.; Sun, W.; Jiang, Z.; Suo, S.; Wang, X.; Liu, Y. Industrial Robot Trajectory Optimization Based on Improved Sparrow Search Algorithm. *Machines* **2024**, *12*, 490. <https://doi.org/10.3390/machines12070490>

Academic Editors: Yong Tao, Hongxing Wei, Haiyuan Li and Guangzhe Zhao

Received: 28 June 2024  
Revised: 14 July 2024  
Accepted: 17 July 2024  
Published: 20 July 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

As industrialization continues to evolve, nations worldwide are actively advancing the modernization of industrial production and automation upgrades. Industrial robots, as the primary equipment for substituting manual labor to enhance production efficiency, have garnered extensive attention and research within industrial automated production lines. The advancement of artificial intelligence-related technologies has ushered in a new era in the field of industrial production, with industrial robots that incorporate intelligent technologies representing a significant direction for future development. To ensure the efficient and stable completion of production tasks during the operation of industrial robots, it is imperative to conduct a comprehensive trajectory optimization study on these machines.

In recent years, researchers have conducted extensive research on trajectory planning for robotic arms [1,2]. In recent years, researchers have conducted extensive research on trajectory planning for robotic arms. The trajectory planning of a robotic arm in joint space requires representing the joint variables as time-dependent functions and then constraining their angles, angular velocities, and angular accelerations. Sabarigirish et al. [3] presents the determination of the trajectory between the initial point and end point for a five-degree-of-freedom (DOF) robotic manipulator in the presence of obstacles. The path is determined using the cubic polynomial, which ensures the smooth motion of the robot. The nonlinear

nature of the robotic manipulator is taken into consideration as well. Fang et al. [4] performed trajectory planning of the robot arm using the cubic polynomial and the quintic polynomial function. The simulation results show that the quintic polynomial method effectively solves the problem of acceleration discontinuity and obtains a continuous smooth trajectory curve of each joint. Porawagama et al. [5] combined cubic and fifth-degree polynomials and used the 5-3-5-degree polynomial interpolation algorithm to make the generated motion trajectory continuously differentiable in position, velocity, and acceleration.

The objectives of trajectory planning vary across different applications of robotic arms, which can be categorized into optimal operation time, optimal energy consumption, optimal impact, and multi-objective optimization [6,7]. Notably, the majority of scholarly research focuses on optimal trajectory planning for operational time [8].

Concurrently, the integration of traditional trajectory planning methodologies with intelligent algorithms for robotic trajectory optimization is a pivotal concept in contemporary robot trajectory research. Markus [9] introduced a novel hybrid time-optimal flexible joint trajectory planning algorithm. This approach employed a smooth time-optimal switching strategy as an alternative to the acceleration mutation, yielding notable results. The foundation of this work was built upon optimal time trajectory planning. Joonyoung et al. [10] introduced a planning methodology designed for robots to execute spot welding tasks with the most efficient trajectory. This method not only efficiently computes the generated trajectory and dynamically approximates the optimal time but also ensures the integrity of path followed in high-frequency planning and control cycles. Luo et al. [11] proposed an improved quantum particle swarm optimization algorithm to enhance convergence speed and avoid local optima. Lan et al. [12] initially employed the seventh-order B-spline curve method to construct the joint trajectory of a robotic arm. They proposed a multi-objective particle swarm optimization algorithm for trajectory competition, aiming to solve the Pareto solution set for the optimal trajectory of the robotic arm in terms of time, energy, and impact. Experimental results demonstrated that this method effectively reduced the motion time, joint impact, and energy consumption of the robotic arm. However, it also increased the time complexity of the algorithm. Cao et al. [13] proposed an enhanced multi-objective particle swarm optimization algorithm designed to optimize the time, energy, and impact functions of a fruit-picking robotic arm. The experimental results indicated that this algorithm facilitated the smooth and efficient execution of fruit-picking operations by the robotic arm. Xu et al. [14] introduced an improved sparrow search algorithm (ISSA) with a fusion strategy to further improve the ability to solve challenging tasks. The results show that the proposed method is more effective, robust, and feasible in mobile robot path planning. Zhao et al. [15] introduced an enhanced whale optimization algorithm to address the optimal trajectory planning issue associated with robot time jitter. This was achieved by constructing a robot joint space fifth-order B-spline interpolation trajectory. Zhao et al. [16] introduced an optimization technique for the trajectory of a mechanical arm, focusing on minimizing both movement time and joint impact as objectives. This approach reframed the multi-objective optimization challenge into a single-objective problem by employing a weighted coefficient method. Additionally, they proposed a trajectory optimization method that integrates hybrid particle swarm optimization with whale optimization algorithms, leading to significant reductions in both movement time and joint impact of the mechanical arm. Wang et al. [17] introduced an enhanced elite nondominated sorting genetic algorithm for the multi-objective trajectory optimization of robotic arms, incorporating three novel genetic operators. This approach yielded superior efficiency and more consistent solutions for point-to-point tasks executed by robotic arms. Ekrem et al. [18] used the MATLAB program and particle swarm optimization (PSO) to carry out the trajectory planning of the robotic arm. Zhang et al. [19] investigated the trajectory-planning problem of a six-axis robotic arm based on deep reinforcement learning. The proposed method demonstrated its effectiveness in comparison with the RRT algorithm, as evidenced by the simulations and physical experiments.

This paper builds upon the principles of mixed polynomial interpolation motion trajectory planning to introduce an enhanced multi-strategy improved sparrow search algorithm. This algorithm is specifically designed for time-optimal trajectory optimization:

1. A two-dimensional logistic system is utilized to initialize the sparrow population, with the objective of enhancing initial diversity and achieving equilibrium.

2. The Levy flight strategy and the nonlinear weighting factor method are utilized to refine the position update algorithm of discoverers within the sparrow population. This approach is designed to optimize the search capability of the algorithm, bolster its robustness, and enhance both the accuracy and speed of convergence.

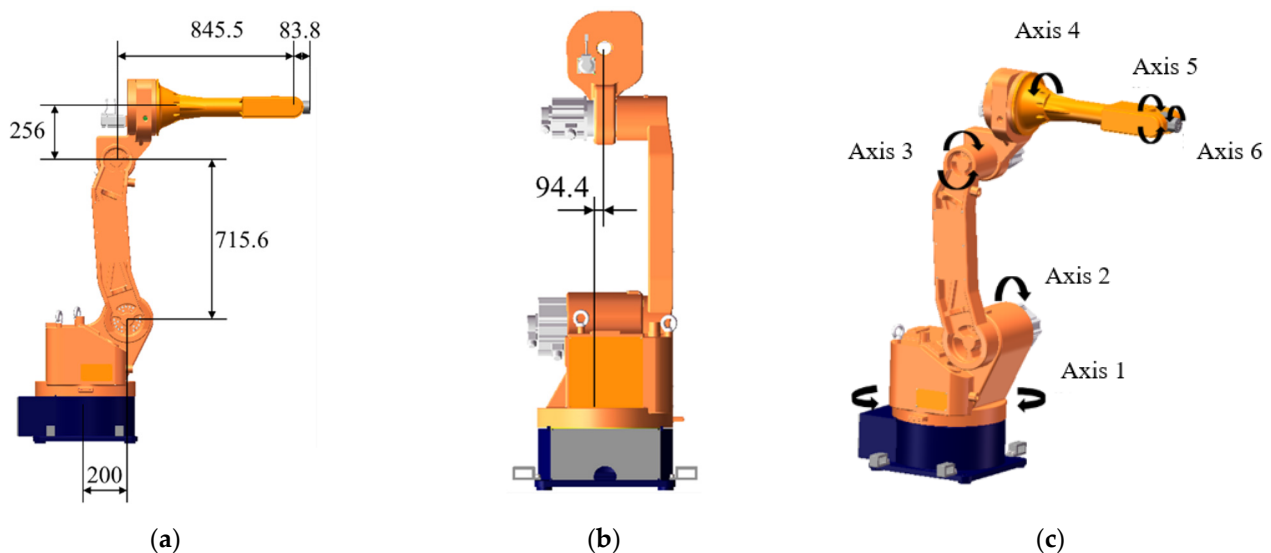
3. The vigilant update algorithm of the sparrow population is augmented with a reverse learning strategy, thereby enhancing both local and global search capabilities.

The findings indicate that the enhanced sparrow search algorithm demonstrates superior convergence speed and accuracy in optimizing industrial robot trajectories, outperforming both the standard sparrow algorithm and the particle swarm algorithm.

This paper is structured as follows: The initial section provides an introduction. The Section 2 presents the kinematic model of the robotic arm. The Section 3 delineates the hybrid polynomial interpolated kinematic trajectory planning. The Section 4 elaborates on the improved sparrow search algorithm. The Section 5 offers the experimental simulation results of the algorithm. Finally, the conclusions are summarized in the concluding section.

## 2. Kinematics Analysis

This paper focuses on a six-axis industrial robot, independently developed by our research team. The overall structure of the robot is depicted in Figure 1. Figure 1a,b illustrate the dimensions of the robot's connecting rod, while Figure 1c depicts the positioning of each joint axis within the robot. The coordinate system for the connecting rod, as presented in Figure 2, is constructed by utilizing the D-H parameter method. This construction also takes into account the mechanical structure's dimensions and the positioning of each joint axis. The robot's D-H parameters can be found in Table 1. These parameters represent the link length, link twist, link offset, and joint angle [20].



**Figure 1.** (a,b) Connecting rod dimensions of the robot; (c) the position of each joint axis of the robot.

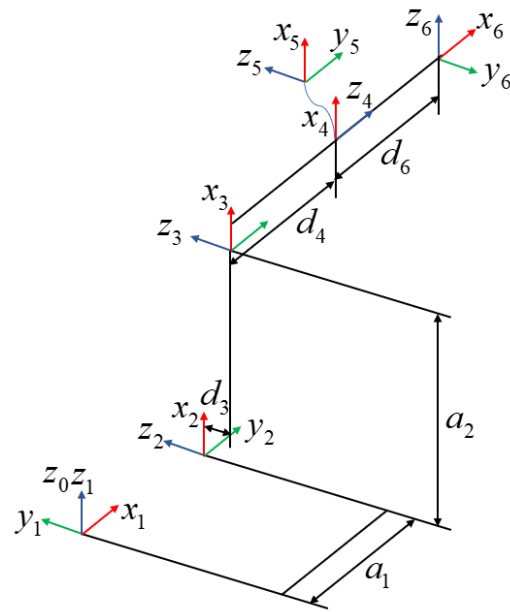


Figure 2. The coordinate system of the robot.

Table 1. The D-H parameter of the robot.

Joint <i>i</i>	$a_{i-1}/mm$	$\alpha_{i-1}/^\circ$	$d_i/mm$	$\theta_i/^\circ$	Joint Range/ $^\circ$
1	0	0	0	$\theta_1$	[−150~180]
2	200	−90	0	$\theta_2$	[−60~120]
3	715.6	0	−94.4	$\theta_3$	[−45~45]
4	256	−90	845.5	$\theta_4$	[−90~90]
5	0	90	0	$\theta_5$	[−90~90]
6	0	−90	83.8	$\theta_6$	[−180~180]

2.1. Forward Kinematic

By utilizing the D-H parametric method, the sub-transformation matrix between the robot’s adjacent linkage coordinate systems {*i* − 1} and {*i*} is represented in Equation (1) [20].

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

For the sake of the simplicity of expression,  $\sin \theta_i$  and  $\cos \theta_i$  in Equation (1) are denoted as  $s_i$  and  $c_i$ , respectively.  $\sin(\theta_m + \theta_n)$  and  $\cos(\theta_m + \theta_n)$  are denoted as  $s_{mn}$  and  $c_{mn}$ , respectively. In Equation (1),  $i, m, n \in \{1, 2, 3, 4, 5, 6\}$ .

The position matrix of the robot’s terminal point is depicted in Equation (2).

$${}^0T = {}^0_1T_1 {}^1_2T_2 {}^2_3T_3 {}^3_4T_4 {}^4_5T_5 {}^5_6T_6 T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} \tag{2}$$

In Equation (2), P and R represent the position and attitude of the robot’s end, respectively, in relation to the base coordinate system. This equation serves as the positive kinematic equation for a six-axis industrial robot. The elements within Equation (2) can be expressed as per Equation (3).

$$\left\{ \begin{array}{l} n_x = c_1 [c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6] + s_1 (s_4c_5c_6 + c_4s_6) \\ n_y = s_1 [c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6] - c_1 (s_4c_5c_6 + c_4s_6) \\ n_z = -s_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6 \\ o_x = c_1 [c_{23}(-c_4c_5s_6 - s_4c_6) + s_{23}s_5s_6] + s_1 (c_4c_6 - s_4c_5s_6) \\ o_y = s_1 [c_{23}(-c_4c_5s_6 - s_4c_6) + s_{23}s_5s_6] - c_1 (c_4c_6 - s_4c_5s_6) \\ o_z = -s_{23}(-c_4c_5s_6 - s_4c_6) + c_{23}s_5s_6 \\ a_x = -c_1 (c_{23}c_4s_5 + s_{23}c_5) - c_1s_4s_5 \\ a_y = -s_1 (c_{23}c_4s_5 + s_{23}c_5) + c_1s_4s_5 \\ a_z = s_{23}c_4s_5 - c_{23}c_5 \\ p_x = d_6 [-s_5(s_1s_4 + c_1c_4c_{23}) - c_1c_5s_{23}] - d_4c_1s_{23} - d_3s_1 - d_2s_1 + a_1c_1 + a_2c_1c_2 + a_3c_1c_{23} \\ p_y = d_6 (c_1s_4s_5 - c_5s_1s_{23} - c_2c_3c_4s_1s_5 + c_4s_1s_2s_3s_5) - d_4s_1s_{23} + d_3c_1 + a_3c_{23}s_1 + a_2c_2s_2 + a_1s_1 \\ p_z = d_6 (c_4s_5s_{23} - c_5c_{23}) - d_4c_{23} - a_3s_{23} - a_2s_2 \end{array} \right. \quad (3)$$

## 2.2. Inverse Kinematics

Robot inverse kinematics is the process of obtaining the angles of each joint of the robot by transforming the matrix based on the known position and posture of the robot's end effector. This study employs an analytical method [21] to address the inverse kinematics of the robot.

Equation (4) is derived from Equation (2):

$${}^1_6T = {}^0_1T^{-1}(\theta_1) {}^0_6T = {}^1_2T(\theta_2) {}^2_3T(\theta_3) {}^3_4T(\theta_4) {}^4_5T(\theta_5) {}^5_6T(\theta_6) \quad (4)$$

Given that the elements (2,4) at Equation (4) are equivalent as follows:

$$-s_1p_x + c_1p_y = d_3 \quad (5)$$

The value of  $\theta_1$  can be found by the following constant triangular transformation:

$$\theta_1 = \text{atan2}(p_y, p_x) - \text{atan2}\left(d_3, \pm \sqrt{(p_x^2 + p_y^2 - d_3^2)}\right) \quad (6)$$

In Equation (6), the plus and minus signs indicate that  $\theta_1$  has two solutions.

Upon establishing a solution for  $\theta_1$ , the elements (1,4) and (3,4) on both sides of Equation (4) are equal, and we can obtain the following:

$$\left\{ \begin{array}{l} c_1p_x + s_1p_y = a_1 + a_3c_{23} - d_4s_{23} + a_2c_2 \\ -p_z = a_3s_{23} + d_4c_{23} + a_2s_2 \end{array} \right. \quad (7)$$

Square both sides of Equation (5) and Equation (7), respectively, and subsequently add these equations to derive Equation (8).

$$a_3c_3 - d_4s_3 = k \quad (8)$$

where  $k = \frac{p_x^2 + p_y^2 + p_z^2 + a_1^2 - a_2^2 - a_3^2 - d_3^2 - d_4^2 - 2a_1c_1p_x - 2a_1s_1p_y}{2a_2}$ .

The  $\theta_2$  term has been eliminated from the equation, thereby enabling the calculation of Equation (9) for  $\theta_3$  through trigonometric substitution.

$$\theta_3 = \text{atan2}(a_3, d_4) - \text{atan2}\left(k, \pm \sqrt{a_3^2 + d_4^2 - k^2}\right) \quad (9)$$

The plus and minus signs in Equation (9) mean that  $\theta_3$  has two roots.

After sorting out Equation (4), Equation (10) can be obtained:

$${}^0_3T^{-1}(\theta_1, \theta_2, \theta_3) {}^0_6T = {}^3_4T(\theta_4) {}^4_5T(\theta_5) {}^5_6T(\theta_6) \quad (10)$$

Given that the elements (1,4) and (2,4) at Equation (10) are equivalent

$$c_1c_{23}p_x + c_{23}s_1p_y - s_{23}p_z - a_1c_{23} - a_2c_3 = a_3 \quad (11)$$

$$-c_1s_{23}p_x - s_1s_{23}p_y - c_{23}p_z + a_1s_{23} + a_2s_3 = d_4 \quad (12)$$

$\theta_2$  can be obtained as follows:

$$\theta_2 = \text{atan2}(s_{23}, c_{23}) - \theta_3 \quad (13)$$

Here

$$s_{23} = \frac{(-a_3 - a_2c_3)p_z + (c_1p_x + s_1p_y - a_1)(a_2s_3 - d_4)}{p_z^2 + (c_1p_x + s_1p_y - a_1)^2} \quad (14)$$

$$c_{23} = \frac{(-d_4 + a_2s_3)p_z - (c_1p_x + s_1p_y - a_1)(-a_2c_3 - a_3)}{p_z^2 + (c_1p_x + s_1p_y - a_1)^2} \quad (15)$$

Given that the elements (1,3) and (3,3) at Equation (10) are equivalent

$$a_xc_1c_{23} + a_y c_{23}s_1 - a_zs_{23} = -c_4s_5 \quad (16)$$

$$-a_x s_1 + a_y c_1 = s_4s_5 \quad (17)$$

$\theta_4$  can be solved

$$\theta_4 = \text{atan2}(-a_x s_1 + a_y c_1, -a_x c_1 c_{23} - a_y c_{23} s_1 + a_z s_{23}) \quad (18)$$

If  $s_5 = 0$ , the robot will assume a singularly shaped position. For singular forms, any value of  $\theta_4$  can be selected to calculate the corresponding  $\theta_6$ .

After sorting out Equation (10), Equation (19) can be obtained:

$${}^0_4T^{-1}(\theta_1, \theta_2, \theta_3, \theta_4) {}^0_6T = {}^4_5T(\theta_5) {}^5_6T(\theta_6) \quad (19)$$

Given that the elements (1,3) and (3,3) at Equation (19) are equivalent

$$a_x(s_1s_4 + c_1c_4c_{23}) + a_y(s_1c_{23}c_4 - c_1s_4) - a_zs_{23}c_4 = -s_5 \quad (20)$$

$$a_x(-c_1s_{23}) + a_y(-s_1s_{23}) + a_z(-c_{23}) = c_5 \quad (21)$$

$\theta_5$  can be solved

$$\theta_5 = \text{atan2}(s_5, c_5) \quad (22)$$

After sorting out Equation (19), Equation (23) can be obtained:

$${}^0_5T^{-1}(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) {}^0_6T = {}^5_6T(\theta_6) \quad (23)$$

Given that the elements (3,1) and (1,1) at Equation (23) are equivalent

$$-n_x(c_1c_{23}s_4 - s_1c_4) - n_y(s_1c_{23}s_4 + c_1c_4) + n_z(s_{23}s_4) = s_6 \quad (24)$$

$$\begin{aligned} & n_x[(c_1c_{23}c_4 + s_1s_4)c_5 - c_1s_{23}s_5] \\ & + n_y[(s_1c_{23}c_4 - c_1s_4)c_5 - s_1s_{23}s_5] \\ & - n_z(s_{23}c_4c_5 + c_{23}s_5) = c_6 \end{aligned} \quad (25)$$

The closed-form solution for  $\theta_6$  is given by Equation (26).

$$\theta_6 = \text{atan2}(s_6, c_6) \quad (26)$$

### 3. Hybrid Polynomial Interpolation in Motion Trajectory Planning

#### 3.1. 3-5-3 Mixed Polynomial Interpolation Functions

While conventional cubic and quintic polynomial interpolation algorithms are highly effective for motion paths between two points, robots often traverse more than two path points during operation [22]. To enhance the efficiency and accuracy of robotic tasks, this paper introduces a hybrid trajectory planning method that combines cubic and quintic polynomial interpolations for multiple path point scenarios. The specific implementation method is tailored to fit the robot under study.

1. Two predetermined path points are determined to designate the initial and terminal positions of the robot arm's movement.
2. The initial and terminal path points are divided into three distinct segments, as illustrated in Figure 3. Within the segment  $0 \sim t_1$ , cubic polynomial interpolation is employed, followed by quintic polynomial interpolation within the segment  $t_1 \sim t_1 + t_2$ . Cubic polynomial interpolation is utilized again in the segment  $t_1 + t_2 \sim t_1 + t_2 + t_3$ . These steps collectively constitute a hybrid "3-5-3" polynomial interpolant.

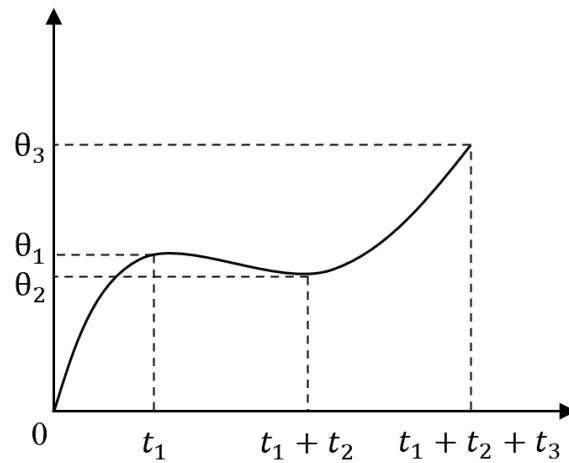


Figure 3. The schematic of hybrid polynomial interpolation.

The relationship between the position of joint  $j$  and the motion time varies across different path segments.

- (1) During the time period  $0 < t < t_1$ , it can be expressed as in Equation (27):

$$\begin{cases} \theta_{j1}(t) = a_{j13}t^3 + a_{j12}t^2 + a_{j11}t + a_{j10} \\ \dot{\theta}_{j1}(t) = 3a_{j13}t^2 + 2a_{j12}t + a_{j11} \\ \ddot{\theta}_{j1}(t) = 6a_{j13}t + 2a_{j12} \end{cases} \quad (27)$$

- (2) During the time period  $t_1 < t < t_1 + t_2$ , it can be expressed as in Equation (28).

$$\begin{cases} \theta_{j2}(t) = a_{j25}(t - t_1)^5 + a_{j24}(t - t_1)^4 + a_{j23}(t - t_1)^3 + a_{j22}(t - t_1)^2 + a_{j21}(t - t_1) + a_{j20} \\ \dot{\theta}_{j2}(t) = 5a_{j25}(t - t_1)^4 + 4a_{j24}(t - t_1)^3 + 3a_{j23}(t - t_1)^2 + 2a_{j22}(t - t_1) + a_{j21} \\ \ddot{\theta}_{j2}(t) = 20a_{j25}(t - t_1)^3 + 12a_{j24}(t - t_1)^2 + 9a_{j23}(t - t_1) + 2a_{j22} \end{cases} \quad (28)$$

- (3) During the time period  $t_1 + t_2 < t < t_1 + t_2 + t_3$ , it can be expressed as in Equation (29).

$$\begin{cases} \theta_{j3}(t) = a_{j33}(t - t_1 - t_2)^3 + a_{j32}(t - t_1 - t_2)^2 + a_{j31}(t - t_1 - t_2) + a_{j30} \\ \dot{\theta}_{j3}(t) = 3a_{j33}(t - t_1 - t_2)^2 + 2a_{j32}(t - t_1 - t_2) + a_{j31} \\ \ddot{\theta}_{j3}(t) = 6a_{j33}(t - t_1 - t_2) + 2a_{j32} \end{cases} \quad (29)$$

In Equations (27)–(29),  $t$  represents the current motion time. The variables  $t_1, t_2,$  and  $t_3$  denote the motion times of joints within the three-segment path trajectories. Here,  $a_{j1k}, a_{j2k},$  and  $a_{j3k}$  (where  $j$  ranges from 1 to 6) signify the  $k$ -th interpolation coefficients of the function associated with the  $j$ -th joint in these trajectories. The terms  $\dot{\theta}_{j1}(t), \dot{\theta}_{j2}(t),$  and  $\dot{\theta}_{j3}(t)$  represent the velocity of joint  $j$  in the three-segment path trajectories. Conversely,  $\ddot{\theta}_{j1}(t), \ddot{\theta}_{j2}(t),$  and  $\ddot{\theta}_{j3}(t)$  indicate the acceleration of joint  $j$  in the same trajectories.

The comprehensive path trajectory comprises four interpolation points, namely  $X_{j1}, X_{j2}, X_{j3},$  and  $X_{j4},$  which are further segmented into three distinct sections. The kinematic Equations (30)–(33) can be derived from Equations (27)–(29).

$$\mathbf{A}\mathbf{a} = \mathbf{A}^{-1}\boldsymbol{\theta} \tag{30}$$

$$\boldsymbol{\theta} = [0\ 0\ 0\ 0\ 0\ 0\ X_{j1}\ 0\ 0\ X_{j2}\ 0\ 0\ X_{j3}\ X_{j4}] \tag{31}$$

$$\mathbf{a} = [a_{j13}\ a_{j12}\ a_{j11}\ a_{j10}\ a_{j25}\ a_{j24}\ a_{j23}\ a_{j22}\ a_{j21}\ a_{j20}\ a_{j33}\ a_{j32}\ a_{j31}\ a_{j30}] \tag{32}$$

$$\mathbf{A} = \begin{bmatrix} t_1^3 & t_1^2 & t_1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 3t_1^2 & 2t_1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 6t_1 & 2 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & t_2^5 & t_2^4 & t_2^3 & t_2^2 & t_2 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 5t_2^4 & 4t_2^3 & 3t_2^2 & 2t_2 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 20t_2^3 & 12t_2^2 & 6t_2 & 2 & 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_3^3 & t_3^2 & t_3 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3t_3^2 & 2t_3 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6t_3 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{33}$$

In the 3-5-3 mixed polynomial, the coefficients  $a_{j1k}, a_{j2k},$  and  $a_{j3k}$  (where  $j$  ranges from 1 to 6) can be determined by substituting them into Equations (30) through (33), based on the time points  $t_1, t_2,$  and  $t_3$  of each path trajectory. This process allows the derivation of the relationship between the three path trajectories associated with joint  $j$  and time  $t$ .

### 3.2. Time-Optimal Fitness Function under Constraints

Within the confines of maximum speed and acceleration restrictions for robot joints, a fitness function is formulated with an emphasis on time optimization. Initially, the optimal value is determined within these constraints by utilizing a mathematical model, as depicted in Equation (34).

$$\begin{cases} F(x) = (\min f_1(x), \min f_2(x), \dots, \min f_m(x)) \\ \text{s.t. } Q_i(x) \leq |Q_{\max}| \\ J_i(x) \leq |J_{\max}| \end{cases} \tag{34}$$

In Equation (34),  $x$  represents a vector  $[x_1, x_2, x_3, \dots, x_n],$  with  $Q_i(x)$  and  $J_i(x)$  serving as constraints. The optimization objective function  $F(x)$  is constructed from  $f_m(x).$

In the context of mixed polynomial interpolation, the fitness function is delineated in Equations (35) and (36), taking into account the three stages of motion time.

$$f(t) = \min \sum_{i=0}^n (t_{i1} + t_{i2} + t_{i3}) \tag{35}$$



$$\begin{cases} |v_{ij}(t)| \leq v_{\max} \\ |a_{ij}(t)| \leq a_{\max} \\ |J_{ij}(t)| \leq J_{\max} \end{cases} \quad (36)$$

In Equations (35) and (36),  $t_{i1}$ ,  $t_{i2}$ , and  $t_{i3}$  represent the motion times of each joint across the three distinct motion paths, while  $v_{ij}(t)$  denotes the joint motion speed. The term  $v_{\max}$  signifies the maximum constraint speed, which is determined by the joint motion acceleration  $a_{ij}(t)$ . Additionally,  $a_{\max}$  represents the maximum allowable constraint acceleration.

#### 4. Improved Sparrow Search Algorithm

##### 4.1. Sparrow Search Algorithm

The sparrow search algorithm (SSA) is a novel intelligent search algorithm, initially proposed by Xue [23] in 2020. This algorithm primarily simulates sparrow foraging and anti-predator behavior to identify the optimal solution in multidimensional space. The sparrow population is categorized into three distinct groups based on their characteristics: discoverers, followers, and vigilantes. Individuals with higher fitness within the population function as discoverers, responsible for food searching. Followers, who trail the discoverers, are accountable for maintaining foraging vigilance within the population. Vigilantes, who sound the alarm upon detecting danger, update the population's location and relocate to other areas for foraging when the vigilance value surpasses a certain threshold.

Supposing that  $n$  sparrows are distributed in the  $D$ -dimensional solution space, then the initial population distribution can be represented in matrix form as Equation (37).

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,\text{dim}} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,\text{dim}} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,\text{dim}} \end{bmatrix} \quad (37)$$

The fitness of the  $n$ th individual sparrow is represented by Equation (38).

$$F_n = [f(x_{i,1}), f(x_{i,2}), f(x_{i,3}), \cdots, f(x_{i,\text{dim}})] \quad (38)$$

The fitness value of each individual sparrow within the initial population is enhanced through a continuous search in space. The update of the discoverer position is represented by Equation (39).

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^t \cdot \exp\left(\frac{-i}{\alpha \cdot N\_gen}\right), R_2 < ST \\ X_{i,j}^t + Q \cdot L, R_2 \geq ST \end{cases} \quad (39)$$

Here,  $X_{i,j}^t$  denotes the position of the  $i$ -th sparrow in  $j$ -dimensional space during the  $t$ -th iteration. The variable  $\alpha$  signifies a random number that adheres to a uniform distribution within the range of 0 to 1.  $N\_gen$  represents the maximum number of iterations, while  $Q$  is a random number that follows a normal distribution within the same range.  $L$  symbolizes a matrix of  $1 \times d$  with elements equal to 1.  $R_2$  stands for the warning value, and  $ST$  signifies the safety value. If the warning value surpasses the safety value, the discoverer will execute random position updates based on the current position.

The equation for the follower to update their position based on the discoverer's position is presented in Equation (40).

$$X_{i,j}^{t+1} = \begin{cases} Q \cdot \exp\left(\frac{X_w^t - X_{i,j}^t}{i^2}\right), i > \frac{NP}{2} \\ X_p^{t+1} + |X_{i,j}^t - X_p^{t+1}| \cdot A^+ \cdot L, i \leq \frac{NP}{2} \end{cases} \quad (40)$$

Here,  $X_w^t$  is the position of the worst-fitted sparrow at iteration  $t$ ,  $X_p^{t+1}$  is the position of the best-fitted sparrow at iteration  $t + 1$ ,  $A$  is a  $1 \times d$  matrix with elements randomly set to 1 or  $-1$ ,  $L$  is a  $1 \times d$  matrix with elements set to 1, and  $NP$  is the population size.

The guardians, constituting 10% to 20% of the population, are tasked with performing vigilance behavior during sparrow foraging activities. Upon detecting potential danger, these guardians will cease their current foraging activity and relocate to the next available foraging location. The mathematical formula used to update the location of these guardians is presented in Equation (41).

$$X_{i,j}^{t+1} = \begin{cases} X_b^t + \beta \cdot |X_{i,j}^t - X_b^t|, f_i > f_g \\ X_{i,j}^t + K \cdot \left( \frac{|X_{i,j}^t - X_w^t|}{(f_i > f_w) + \varepsilon} \right), f_i = f_g \end{cases} \quad (41)$$

Here,  $X_b^t$  denotes the position of the sparrow with the highest global fitness at iteration  $t$ . The variable  $\beta$  is a random number drawn from a normal distribution  $(0,1)$ ,  $K$  is a uniformly distributed random number in the range  $(-1,1)$ , and  $\varepsilon$  is a minimum value to prevent the denominator from becoming zero.  $f_i$ ,  $f_g$ , and  $f_w$  represent the fitnesses of the current sparrow, the globally highest individual, and the globally lowest individual, respectively. If the vigilant individual occupies the global optimal position, it will move based on its current location. Conversely, if it does not occupy the global optimal position, it will shift towards the current optimal position.

#### 4.2. Enhanced Multi-Strategy Sparrow Search Algorithm

The conventional sparrow search algorithm initializes the population at random within the search space. However, as the algorithm iterates, there is a gradual decrease in both the diversity and balance of the population distribution. This can negatively impact the convergence speed and stability of the algorithm. Furthermore, according to Equation (39), within the warning value range, the discoverer updates its position based on a small range of its current location. Consequently, the positions of sparrow individuals after iteration tend to converge to a single point, significantly increasing the likelihood of the algorithm falling into local optimization [24]. When the warning value is exceeded, the discoverer employs a random strategy to update its position. Similarly, as the iteration progresses, the population richness also gradually diminishes. To address these issues, this paper proposes an improved sparrow search algorithm. This algorithm initially uses a two-dimensional Logistic chaotic system to initialize the population, thereby ensuring the richness and balance of the initial population. It then introduces a Levy flight strategy and a nonlinear adaptive weight to enhance the position update operator of the discoverer. Additionally, it incorporates a reverse learning strategy to improve the position update operator of the vigilant. This approach ensures that in the early stages of algorithm iteration, the weight increases significantly to boost global search ability. In contrast, in the later stages of iteration, the weight decreases slightly to strengthen local search ability. Simultaneously, this method avoids issues such as the “premature” convergence and local optimization of the algorithm, thereby enhancing its robustness, as well as its convergence accuracy and speed.

##### 4.2.1. Improved Two-Dimensional Logistic Chaotic System

In recent years, the rapid advancement of chaos theory has expanded its application scope. The defining characteristics of chaotic systems include high randomness, divergence, and ergodicity [25]. Logistic mapping, as depicted in Equation (42), is a classic example of such a system.

$$x_{n+1} = \mu x_n (1 - x_n) \quad (42)$$

Here,  $x_n$  and  $x_{n+1}$  are chaotic sequences, and  $\mu$  is a chaotic control parameter. When  $3.55 < \mu < 4.0$ , the chaotic characteristics of the mapping sequence begin to emerge.

Despite the robust chaotic properties of the classic Logistic chaotic mapping, its small mapping space and suboptimal randomness limit its effectiveness. To address these issues, an enhanced two-dimensional Logistic chaotic mapping [25] approach is employed to replace the initial population in the sparrow search algorithm with a generated chaotic sequence. This substitution results in superior chaotic performance, thereby enhancing the global convergence of the algorithm and preventing it from becoming trapped in local optimal problems. The improved two-dimensional Logistic chaotic system [26] is defined in Equation (43).

$$\begin{cases} x_{n+1} = \sin(\pi(4ax_n(1-x_n)) + (1-a)\sin(\pi y_n)) \\ y_{n+1} = \sin(\pi(4ay_n(1-y_n)) + (1-a)\sin(\pi(x_{n+1}^2))) \end{cases} \quad (43)$$

Here,  $y_n$  and  $y_{n+1}$  are chaotic sequences. When  $0 < a < 1$ , the system is in a chaotic state.

#### 4.2.2. Discoverer Update Algorithm Based on Levy Flight and Nonlinear Weighting Factors

Levy flight, a strategy that employs both short-range and long-range random transformations for spatial movement, is characterized by its unique randomness and leaps. This method is particularly well suited to the position updates of the sparrow algorithm's discoverer. It iteratively selects movements based on the current position, utilizing either short- or long-range steps. This approach enhances the algorithm's search capabilities and augments the diversity of spatial population distribution.

Levy flight denotes a non-Gaussian stochastic variation process. The Monte Carlo algorithm is employed to simulate the calculation method of random step length during this flight process [27], as depicted in Equation (44).

$$\text{Levy} = \frac{\mu}{|\nu|^{\frac{1}{\beta}}} \quad (44)$$

$$\begin{cases} \sigma_{\mu} = \left\{ \frac{\Gamma(1+\beta) \times \sin(\pi\beta/2)}{\Gamma[(1+\beta)/2] \times \beta \times 2^{(\beta-1)/2}} \right\}^{1/\beta} \\ \sigma_{\nu} = 1 \end{cases} \quad (45)$$

Here,  $\Gamma$  is Standard Gamma Function,  $\beta$  is a constant with value 1.5, and  $\mu$  and  $\nu$  are parameters that follow a normal distribution  $\mu \sim N(0, \sigma_{\mu}^2)$ ,  $\nu \sim N(0, \sigma_{\nu}^2)$ .

In the algorithm for updating the discoverer position, a nonlinear weight factor [28] can be employed when the warning value exceeds the safety value ( $R_2 \geq ST$ ). This ensures that the weight factor is substantial during the initial stages of algorithm iteration, with a gradual reduction in its rate of decrease. This strategy broadens the search range of the sparrow, thereby enhancing global search capabilities and increasing population richness. As the algorithm progresses to later stages, the weight factor diminishes progressively with each iteration. This is advantageous for accelerating convergence and improving local search capabilities, as demonstrated in Equation (46).

$$\begin{cases} \frac{d\omega}{dt} = \frac{2 \cdot (\omega_{\max} - \omega_{\min})}{t_{\max}^2} \cdot t \\ \omega = \omega_{\max} - \frac{\omega_{\max} - \omega_{\min}}{t_{\max}^2} \cdot t^2 \end{cases} \quad (46)$$

Here,  $\omega_{\max}$  and  $\omega_{\min}$  represent the maximum and minimum values of the nonlinear weight factor, respectively.  $t$  denotes the current iteration number, while  $t_{\max}$  signifies the maximum iteration number.

By integrating Equations (44)–(46), we present an enhanced discoverer position update algorithm, which is based on Levy flight and nonlinear weight factors, as depicted in Equation (47).

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^t + X_{i,j}^t \cdot \text{Levy}_j, R_2 < ST \\ X_{i,j}^t + Q \cdot L \cdot \omega, R_2 \geq ST \end{cases} \quad (47)$$

#### 4.2.3. Vigilant Update Algorithm Based on Reverse Learning Strategy

The reverse learning strategy generates a reverse solution derived from the original one. This is subsequently compared to yield an enhanced solution. When integrated into the vigilant update algorithm, the process entails determining the optimal position between the original and reverse solution positions, followed by position updates. The relevant equations are presented in Equations (48) and (49).

$$X_b^*(t) = ub + r \oplus (lb - X_b(t)) \quad (48)$$

$$X_{i,j}^{t+1} = X_b^t + b_1 \oplus (X_b(t) - X_b^*(t)) \quad (49)$$

Here,  $X_b^t(t)$  denotes the inverse solution of the optimal solution at iteration  $t$ . The variables  $ub$  and  $lb$  signify the upper and lower bounds of the solution space, respectively.  $r$  is a random matrix that follows a uniform distribution within the range of (0,1). Additionally,  $b_1$  serves as an information control parameter, which can be expressed using Equation (50).

$$b_1 = \left( t_{\max} - \frac{t}{t_{\max}} \right)^t \quad (50)$$

#### 4.2.4. Process of Improved Sparrow Search Algorithm

In summary, the methodology of the “3-5-3” hybrid polynomial interpolation trajectory planning multi-strategy improved sparrow search algorithm for time-optimal six-axis robots, as proposed in this article, is depicted in Figure 4.

The main procedures are as follows:

Step 1: Ascertain the angle values associated with the six joints and identify the four interpolation points based on the inverse kinematics solution of the robot arm.

Step 2: The initialization of parameters for the enhanced sparrow search algorithm encompasses the number of populations, the maximum iteration count, and the allocation of roles such as discoverers, followers, vigilantes, among others.

Step 3: Utilize a two-dimensional Logistic chaotic system to generate a random initial population and subsequently initialize the population.

Step 4: Determine the optimal and least favorable individual fitnesses within the constraints. Subsequently, update the positions of the discoverer, follower, and guard by utilizing the position update operator of the enhanced sparrow search algorithm.

Step 5: Upon attaining the maximum iteration count, the optimal time for trajectory planning is output. Should the maximum iteration count not be met, the process reverts to step 4 until this threshold is satisfied.

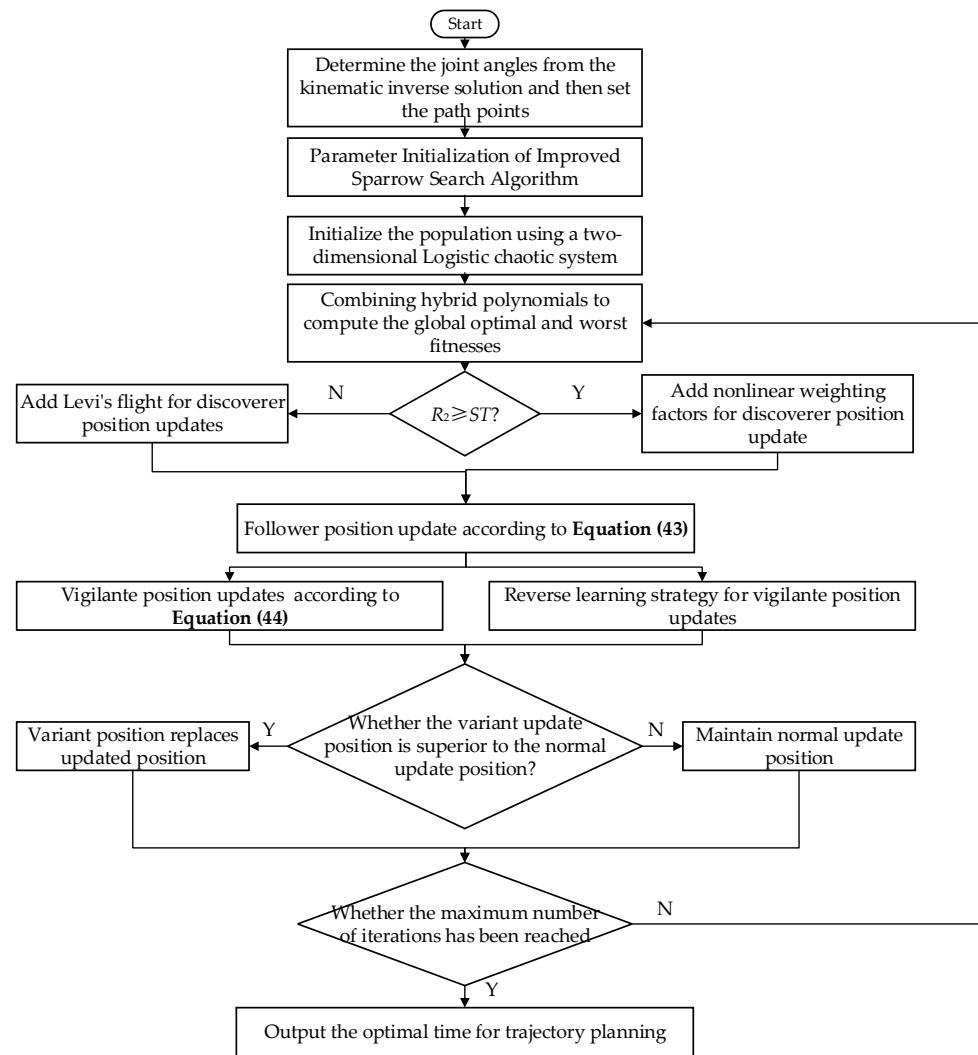


Figure 4. The flowchart of improved sparrow search algorithm.

### 5. Simulation Analysis

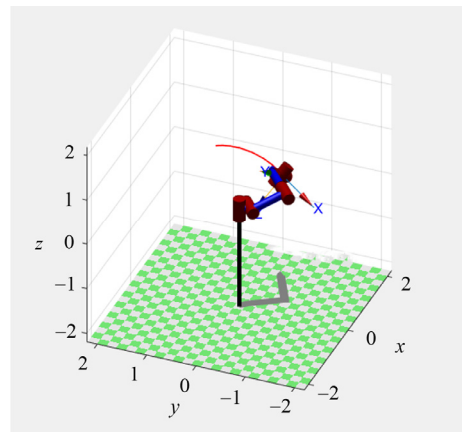
The six-axis industrial robot, designed for this study, is kinematically modeled using MATLAB. A trajectory for the robot’s end point is planned in Cartesian space, as depicted in Figure 5. Within this Cartesian space, four path points are selected from the robot’s motion path, which extends from the start to the end point. The interval between these four path points is set at 2 s. Subsequently, an inverse kinematic solution is applied to these four path points to determine the joint angle values corresponding to each of the robot’s six joints. These values are presented in Table 2. The kinematic constraints associated with each joint of the robot are detailed in Table 3.

Table 2. The corresponding angle values for each joint interpolation point.

Joint <i>i</i>	Path Point 1 /°	Path Point 2 /°	Path Point 3 /°	Path Point 4 /°
1	0	0	0	0
2	0	−30	−45	−50
3	−45	−30	−20	−20
4	0	30	45	60
5	0	0	30	60
6	0	0	0	30

**Table 3.** TKinematic constraints of joints.

Joint $i$	Maximum Angular Velocity/(°/s)	Maximum Angular Acceleration/(°/s <sup>2</sup> )
1	150	93
2	180	120
3	210	160
4	360	170
5	360	260
6	450	330

**Figure 5.** The Cartesian spatial planning trajectories.

To ascertain the efficacy of the improved sparrow algorithm (ISSA) proposed in this study, it was compared with the standard sparrow algorithm (SSA) and the standard particle swarm optimization algorithm (PSO) for optimizing the “3-5-3” mixed polynomial interpolation trajectory. Subsequently, a comparative analysis of the optimization outcomes derived from these three algorithms was conducted. During the optimization phase, the ISSA and SSA algorithms were initialized with the following parameters: a sparrow population size of 30, a maximum iteration count of 1000, a discoverer proportion of 20%, a vigilant proportion of 10%, and a vigilance value of 0.8. Conversely, the PSO algorithm was initialized with the following parameters: a population size of 30, a maximum iteration count of 1000, an initial inertia weight of 0.9, and a final inertia weight of 0.4.

The convergence curves for the three algorithms, which are based on time-optimal trajectory planning for the robot’s six joints, are depicted in Figure 6. As observed from this figure, the ISSA algorithm demonstrates superior performance in terms of both convergence speed and accuracy when compared to the PSO and SSA algorithms. Upon comparing the convergence curves of joints 2, 3, 4, 5, and 6, the ISSA algorithm significantly outperforms both the PSO and SSA algorithms in terms of convergence accuracy, as shown in Table 4. Specifically, for joint 2, the ISSA algorithm demonstrates a convergence accuracy that is 13% higher than that of the SSA algorithm and 33% higher than that of the PSO algorithm. For joint 3, the ISSA algorithm demonstrates a similar convergence accuracy to the SSA algorithm and exhibits a convergence accuracy that is 6% higher than that of the PSO algorithm. In the case of joint 4, the convergence accuracy achieved through the implementation of the ISSA algorithm surpasses those of the SSA and PSO algorithms by 18.8% and 48%, respectively. For joint 5, the ISSA algorithm outperforms both the SSA and PSO algorithms by 6.5% and 9.3%, respectively. In the instance of joint 6, the ISSA algorithm demonstrates a 2.4% increase in convergence accuracy compared to the PSO algorithm. On a broader scale, it is evident that the ISSA algorithm consistently delivers superior convergence accuracy, exceeding those of the SSA and PSO algorithms by up to 18.8% and 48%, respectively.

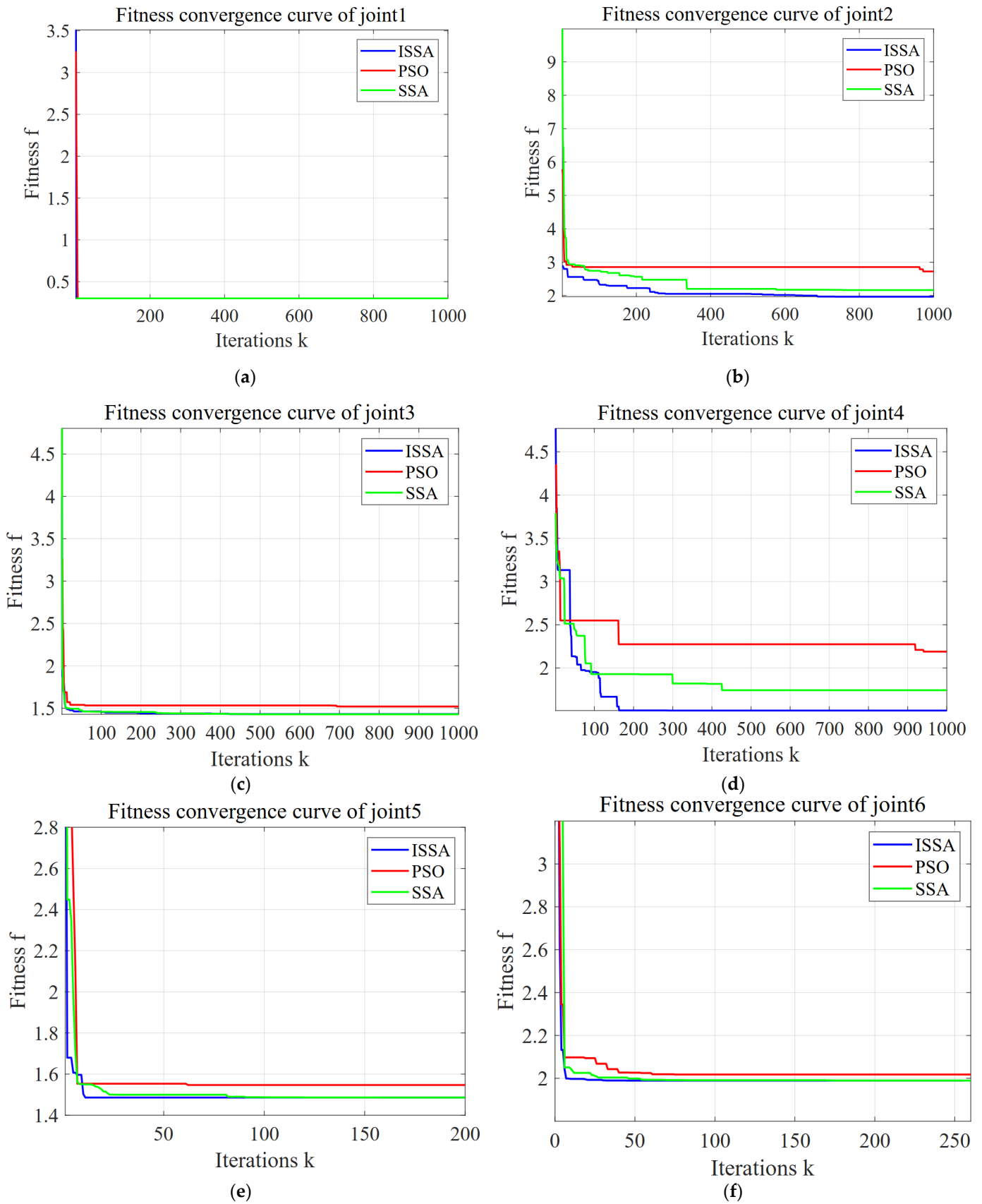


Figure 6. (a–f) fitness convergence curve of the robot joints 1–6.

**Table 4.** ISSA convergence accuracy compared to SSA and PSO.

Convergence Accuracy	ISSA Compared to SSA	ISSA Compared to PSO
Joint 2	13%	33%
Joint 3	-	6%
Joint 4	18.8%	48%
Joint 5	6.5%	9.3%
Joint 6	2.4%	18.8%

Upon comparing joints 4, 5, and 6, it is evident that the ISSA algorithm exhibits a superior convergence rate in comparison to both the PSO and SSA algorithms, as shown in Table 5. Specifically, for joint 4, the convergence rate achieved through the ISSA algorithm surpasses that of the SSA algorithm by 25% and outperforms the PSO algorithm by 16.7%. For joint 5, the ISSA algorithm demonstrates an 80% increase in convergence rate compared to the PSO algorithm. In the case of joint 6, the ISSA algorithm's convergence rate exceeds that of the SSA algorithm by 44.4% and surpasses the PSO algorithm by 56%. On a broader scale, the ISSA algorithm's convergence rate can be up to 73% higher than that of the SSA algorithm and up to 80% higher than that of the PSO algorithm.

**Table 5.** ISSA convergence speed compared to SSA and PSO.

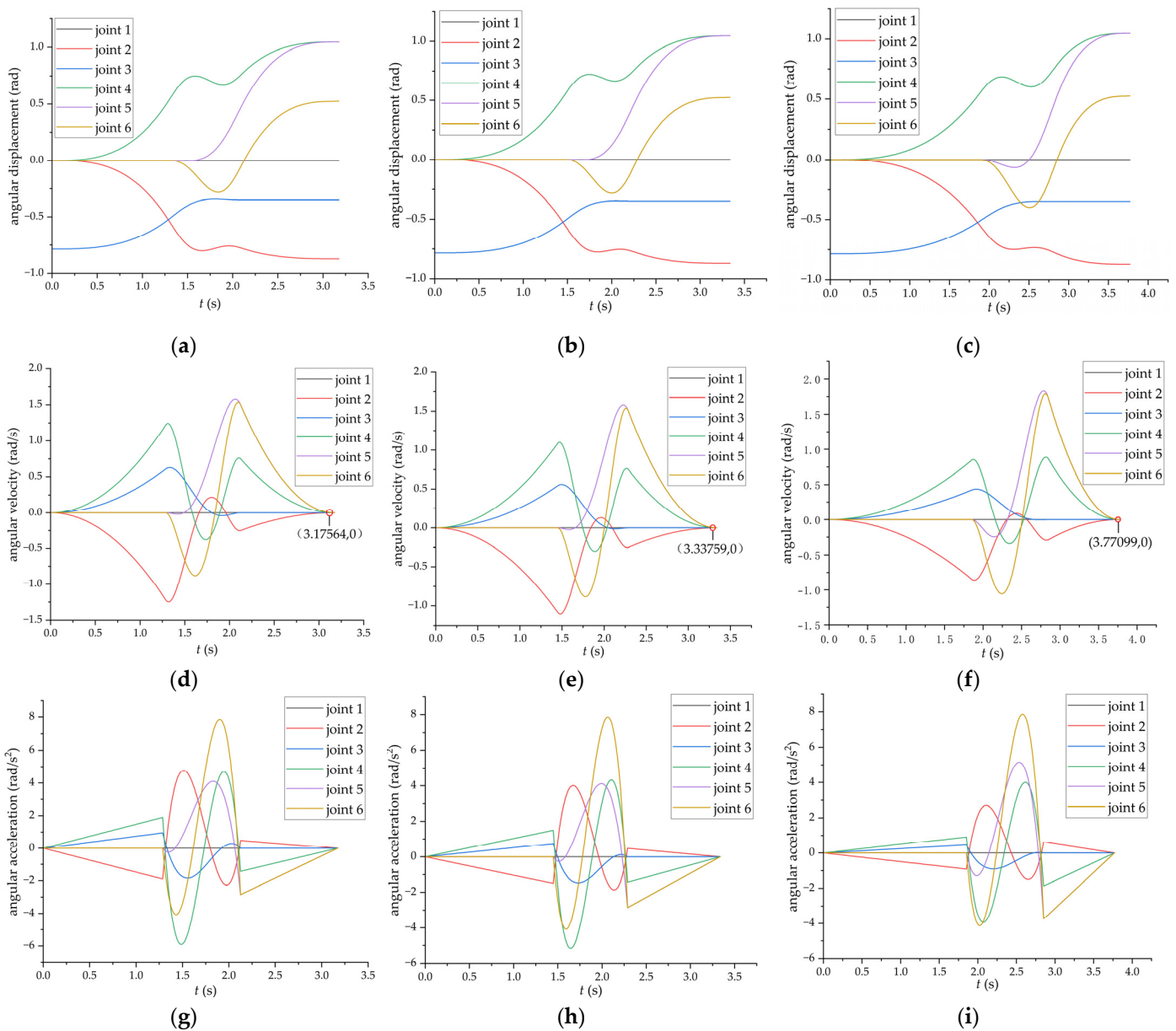
Convergence Speed	ISSA Compared to SSA	ISSA Compared to PSO
Joint 4	25%	16.7%
Joint 5	52%	80%
Joint 6	44.4%	56%

Given the minimal interval between path points, the fitness curves of the two algorithms exhibit negligible differences in joint 1. Consequently, employing the ISSA algorithm for time-optimal trajectory planning of six-axis industrial robots is both feasible and efficient. Through the application of the ISSA algorithm, it becomes possible to effectively accomplish trajectory planning in the shortest timeframe, thereby enhancing the robot's work efficiency and optimizing its performance. Simultaneously, the efficiency of the ISSA algorithm demonstrates strong practicality and promising application prospects in real-world engineering scenarios.

Figure 7 illustrates the curves for angular displacement, angular velocity, and angular acceleration obtained following the application of the ISSA, SSA, and PSO algorithms in time-optimal trajectory planning for an industrial robot with six joints. This process ensures compliance with kinematic constraints.

As illustrated in Figure 7, the industrial robot optimized by the PSO algorithm operates at a time of approximately 3.77099 s. In contrast, the robot optimized by the SSA algorithm functions at an average time of 3.33759 s, while the one optimized by the ISSA algorithm operates at around 3.17564 s. The ISSA-optimized robot's running time is notably shorter than that of the unoptimized model by 2.82436 s, which represents a reduction of 47%. Furthermore, the ISSA-optimized robot's running time is 0.16195 s or 4.9% shorter than that of the robot optimized by the SSA algorithm, and it is 0.59535 s or 15.8% faster than that of the robot optimized by the PSO algorithm. This comparison indicates that each joint trajectory of the unoptimized robot experiences a significant reduction in running time when optimized by either the ISSA, SSA, or PSO algorithms. However, the optimization effect of the ISSA algorithm surpasses that of both the SSA and PSO algorithms, effectively enhancing the robot's operational efficiency. Specifically, each joint's running time is reduced by 0.59535 s compared to the unoptimized model. Post-optimization, the curves representing angular displacement, velocity, and acceleration for each joint of the robot are smoother and exhibit no abrupt changes, thereby mitigating any rigid or flexible impact on the robot's performance.





**Figure 7.** (a–c) The angular displacement curve obtained after using the ISSA, SSA, and PSO algorithms; (d–f) the angular velocity curve obtained after using the ISSA, SSA, and PSO algorithms; (g–i) the angular acceleration curve obtained after using the ISSA, SSA, and PSO algorithms.

### 6. Conclusions

This paper introduces an enhanced multi-strategy sparrow search algorithm, which is based on the 3-5-3 trajectory planning. The aim is to achieve time-optimal trajectory planning for industrial robots.

This paper proposes an enhanced multi-strategy sparrow search algorithm based on a 3-5-3 hybrid polynomial interpolation. The specific improvement strategies include initializing the population using a two-dimensional Logistic chaotic system to ensure diversity and balance, implementing a Levy flight strategy and nonlinear adaptive weighting to enhance the discoverer position updating operator, and introducing an inverse learning strategy to refine the vigilante position-updating operator. This approach ensures that during the early iterations of the algorithm, a larger weight boosts the global search capability. Conversely, in later iterations, a smaller weight augments the local search ability. This

method simultaneously mitigates the issues of “precocity” and local optimization, thereby enhancing the robustness of the algorithm, as well as its convergence accuracy and speed.

A six-axis industrial robot independently developed by the laboratory is taken as the research object, and the improved sparrow search algorithm, traditional sparrow search algorithm, and particle swarm algorithm are, respectively, used to carry out time-optimal trajectory planning simulation experiments based on 3-5-3 hybrid polynomial interpolation. The numerical results demonstrate that the enhanced sparrow search algorithm outperforms both the standard sparrow algorithm and the particle swarm algorithm in terms of convergence speed and accuracy when optimizing a robot’s trajectory. This improved algorithm not only converges more rapidly but also achieves greater precision, thereby increasing the robot’s work efficiency and overall performance.

**Author Contributions:** Conceptualization, F.M., W.S., and X.W.; Data curation, F.M.; Formal analysis, X.W. and Y.L.; Funding acquisition, F.M. and Z.J.; Investigation, W.S.; Methodology, F.M. and Z.J.; Project administration, F.M. and Y.L.; Resources, W.S. and Y.L.; Software, F.M. and X.W.; Supervision, Z.J. and S.S.; Validation, W.S. and Y.L.; Visualization, W.S.; Writing—original draft, X.W.; Writing—review and editing, S.S. and Y.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Young Backbone Teacher Support Plan of the Beijing Information Science and Technology University (number 5112411115) and the General Program of the National Natural Science Foundation of China (grant number 52175452).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data are contained within the article.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Yudha, H.M.; Dewi, T.; Risma, P.; Oktarina, Y. Arm Robot Manipulator Design and Control for Trajectory Tracking; a Review. In Proceedings of the 2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), Malang, Indonesia, 16–18 October 2018; pp. 304–309.
2. Ma, C.; Zhang, Y.; Cheng, S. Time Optimal Trajectory Planning Based on Redundant Manipulator. In Proceedings of the 2021 6th International Conference on Control, Robotics and Cybernetics (CRC), Shanghai, China, 9–11 October 2021; pp. 125–129.
3. Sabarigirish, S.; Mija, S.J. Obstacle avoiding trajectory planning for 5 degree of freedom robot. In Proceedings of the 2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), Delhi, India, 4–6 July 2016; pp. 1–5.
4. Fang, S.; Ma, X.; Zhao, Y.; Zhang, Q.; Li, Y. Trajectory Planning for Seven-DOF Robotic Arm Based on Quintic Polynomial. In Proceedings of the 2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 24–25 August 2019; pp. 198–201.
5. Porawagama, C.D.; Munasinghe, S.R. Reduced jerk joint space trajectory planning method using 5-3-5 spline for robot manipulators. In Proceedings of the 7th International Conference on Information and Automation for Sustainability, Colombo, Sri Lanka, 22–24 December 2014; pp. 1–6.
6. Zhang, T.; Zhang, M.H.; Zou, Y.B. Time-optimal and smooth trajectory planning for robot manipulators. *Int. J. Control Autom. Syst.* **2021**, *19*, 521–531. [[CrossRef](#)]
7. Wang, L.; Wu, Q.; Lin, F.; Li, S.; Chen, D. A new trajectory-planning beetle swarm optimization algorithm for trajectory planning of robot manipulators. *IEEE Access* **2019**, *7*, 154331–154345. [[CrossRef](#)]
8. Wang, N.G.; Dong, Y.C.; Chen, H.Q. A Time Optimal Trajectory Planning of the Metamorphic Industrial Robot. *J. Mech. Eng.* **2023**, *59*, 100–111.
9. Markus, R. Near Time-optimal Flexible-joint Trajectory Planning Algorithm for Robotic Manipulators. In Proceedings of the IEEE Conference on Control Technology and Applications, Copenhagen, Denmark, 21–24 August 2018; IEEE: New York, NY, USA, 2018; pp. 265–272.
10. Joonyoung, K.; Croft, E.A. Online Near Time-optimal Trajectory Planning for Industrial Robots. *Robot. Comput. Integr. Manuf.* **2019**, *58*, 158–171.
11. 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](#)]

12. Lan, J.Y.; Xie, Y.G.; Liu, G.J.; Cao, M.X. A Multi-objective Trajectory Planning Method for Collaborative Robot. *Electronics* **2020**, *9*, 859. [[CrossRef](#)]
13. Cao, X.M.; Yan, H.S.; Huang, Z.Y.; Ai, S.; Xu, Y.; Fu, R.; Zou, X. A Multi-objective Particle Swarm Optimization for Trajectory Planning of Fruit Picking Manipulator. *Agronomy* **2021**, *11*, 2286. [[CrossRef](#)]
14. Xu, Y.; Sang, B.; Zhang, Y. Application of Improved Sparrow Search Algorithm to Path Planning of Mobile Robots. *Biomimetics* **2024**, *9*, 351. [[CrossRef](#)] [[PubMed](#)]
15. Zhao, J.; Zhu, X.; Song, T.; Meng, X. An Improved Whale Optimization Algorithm for Robot Time-jerk Optimal Trajectory Planning. *J. Phys. Conf. Ser.* **2022**, *2170*, 012008. [[CrossRef](#)]
16. Zhao, J.; Zhu, X.J.; Song, T.J. Serial Manipulator Time-Jerk Optimal Trajectory Planning Based on Hybrid IWOA-PSO Algorithm. *IEEE Access* **2022**, *10*, 6592–6604. [[CrossRef](#)]
17. Wang, Z.S.; Li, Y.B.; Shuai, K.; Zhu, W.; Chen, B.; Chen, K. Multi-objective Trajectory Planning Method Based on the Improved Elitist Non-dominated Sorting Genetic Algorithm. *Chin. J. Mech. Eng.* **2022**, *35*, 7. [[CrossRef](#)]
18. Ekrem, Ö.; Aksoy, B. Trajectory planning for a 6-axis robotic arm with particle swarm optimization algorithm. *Eng. Appl. Artif. Intell.* **2023**, *122*, 106099. [[CrossRef](#)]
19. Zhang, S.; Xia, Q.; Chen, M.; Cheng, S. Multi-Objective Optimal Trajectory Planning for Robotic Arms Using Deep Reinforcement Learning. *Sensors* **2023**, *23*, 5974. [[CrossRef](#)] [[PubMed](#)]
20. Craig, J.J. *Introduction to Robotics*, 4th ed.; Pearson Education, Inc.: London, UK, 2018.
21. Feng, M.; Dai, J.; Zhou, W.; Xu, H.; Wang, Z. Kinematics Analysis and Trajectory Planning of 6-DOF Hydraulic Robotic Arm in Driving Side Pile. *Machines* **2024**, *12*, 191. [[CrossRef](#)]
22. Tonan, M.; Bottin, M.; Doria, A.; Rosati, G. Motion Planning of Differentially Flat Planar Underactuated Robots. *Robotics* **2024**, *13*, 57. [[CrossRef](#)]
23. Xue, J.K.; Shen, B. A Novel Swarm Intelligence Optimization Approach: Sparrow Search algorithm. *Syst. Sci. Control Eng.* **2020**, *8*, 22–34. [[CrossRef](#)]
24. Su, Y.Y.; Wang, S.X. Adaptive Hybrid Strategy Sparrow Search Algorithm. *Comput. Eng. Appl.* **2023**, *59*, 75–85.
25. Hraoui, S.; Gmira, F.; Jarar, A.O.; Satori, K.; Saaïdi, A. Benchmarking AES and Chaos Based Logistic Map for Image Encryption. In Proceedings of the Acs International Conference on Computer Systems & Applications, Ifrane, Morocco, 27–30 May 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 1–4.
26. Fang, P.F.; Huang, L.G.; Lou, M.M.; Jiang, K. An Image Encryption Algorithm Based on Combining Two-dimensional Logistic Chaotic Map and DNA Sequence Operation. *China Sci.* **2021**, *16*, 247–252.
27. Amirsadri, S.; Mousavirad, S.J.; Ebrahimpour-Komleh, H. A Levy Flight-based Grey Wolf Optimizer Combined with Back-propagation Algorithm for Neural Network Training. *Neural Comput. Appl.* **2018**, *30*, 3707–3720. [[CrossRef](#)]
28. Qiu, B.; Li, X.B.; Shi, Z.X.; Luo, Y.F. Application of Multi-strategy Improved Sparrow Algorithm in Time Optimal Trajectory Planning of Manipulator. *Mech. Sci. Technol. Aerosp. Eng.* **2024**, 1–10. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.