machines

MDPI

Article

A Multi-Objective Trajectory Planning Method of the Dual-Arm Robot for Cabin Docking Based on the Modified Cuckoo Search Algorithm

Ronghua Liu [1,2] **and Feng Pan** [1,*]

[1] School of Automation, Beijing Institute of Technology, Beijing 100081, China; ronghua5416@163.com
[2] Jiangsu Automation Research Institute, Lianyungang 222061, China
* Correspondence: panfeng@bit.edu.cn

Abstract: During the assembly of mechanical systems, the dual-arm robot is always used for cabin docking. In order to ensure the accuracy and reliability of cabin docking, a multi-objective trajectory planning method for the dual-arm robot was proposed. A kinematic model of the dual-arm robot was constructed based on the Denavit–Hartenberg (D-H) method firstly. Then, in the Cartesian space, the end trajectory of the dual-arm robot was confirmed by the fifth-order B-spline curve. On the basis of a traditional multi-objective cuckoo search algorithm, a modified cuckoo algorithm was built using the improved initial population generation method and the step size. The total consumption time and joint impact were selected as the objective functions, the overall optimal solution for the modified cuckoo algorithm was obtained using the normalized evaluation method. The optimal trajectory planning was achieved. Finally, the feasibility and effectiveness of the trajectory planning method were verified with the experiments.

Keywords: multi-objective trajectory planning; modified cuckoo algorithm; dual-arm robot; cabin docking

check for updates

Citation: Liu, R.; Pan, F. A Multi-Objective Trajectory Planning Method of the Dual-Arm Robot for Cabin Docking Based on the Modified Cuckoo Search Algorithm. *Machines* **2024**, *12*, 64. https://doi.org/10.3390/machines12010064

Academic Editor: Augusto Luis Ballardini

Received: 15 November 2023
Revised: 11 January 2024
Accepted: 12 January 2024
Published: 16 January 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

In the manufacturing process of mechanical products, the assembly of cabin segments is very important, such as rockets, satellites and airplanes. The performance of the mechanical system is affected by the assembly quality [1]. Thus, there are various manipulative mechanisms for the compartment docking. Parallel mechanisms, mechanical mobile structures and decoupling control platform mechanisms can be found. Among them, the decoupling control platform mechanism is widely used [2]. From Figure 1, the cabin docking robot is often designed for the decoupling control platform mechanism, and there is a left robotic arm, a right robotic arm and a base. The left arm and the right arm are connected to the base, and all of the arms move axially through the gear and transmission. The structure of the left arm is the same as the right arm's structure.

The structure details of the right arm are given in Figure 2. The axial movement joint, vertical lifting joint and circumferential rolling joint are seen. Thus, there are six degrees of freedom for the dual arm: pitch, yaw, roll, lifting, front and rear. The axial movement joint is driven by the motor–gear–rack system, and the left movement joint and right movement joint are driven by the motor-screw systems. The vertical lifting joint is driven by the motor–spiral elevator for the lifting, and the circular rolling joint is driven by the motor–gear transmission system. There is an arc-shaped structure for the circular rolling joint, and the special rubber is used to increase the friction force among the workpieces.

On the basis of the above arm construct, trajectory planning is needed for the cabin docking. There are two steps for the trajectory planning: trajectory generation and trajectory optimization. The precondition for trajectory planning is provided by the former, while the trajectory performance is improved by the latter [3]. The trajectory is generated by the discrete path points, and the points are obtained by polynomial functions, parabolic functions or B-spline curve functions [4–8]. Therefore, in order to confirm the optimal trajectories, the multi-objective trajectory optimization algorithm has become an important aspect [9]. And some parameters are used for the multi-objective optimization parameters, such as the minimum execution time, minimum energy consumption and minimum disturbance. Firstly, an improved immune clone selection algorithm was proposed by Chen et al. [10]. In this algorithm, a multi-objective model for trajectory optimization was established using the total motion time, total energy consumption and actuator thrust. The trajectory was generated using a fifth-order B-spline curve, and the improved immune clone algorithm was used for the trajectory optimization. Lan et al. discussed a multi-objective trajectory planning method for collaborative robots [11]. Based on the collaborative kinematic model, the joint trajectories were constructed from the B-spline curves. The multi-objective particle swarm optimization algorithm was used for the Pareto optimal solution of the time-energy jump optimal trajectories. Then, the most suitable solution is selected using the normalized weighting function. A novel optimal trajectory planning method was constructed by Wang et al. [12]. A trajectory was created with fifth-order polynomials and cubic Bézier curves. The genetic operators were used for optimizing the travel time and torque functions, such as Rank-based Group Selection (RGS), Direction-based Crossover (DBX) and Adaptive Precision Controlled Mutation (APCM). Then, in virtue of Hermite cubic interpolation, the polynomial was used for the trajectory process of space species [4].

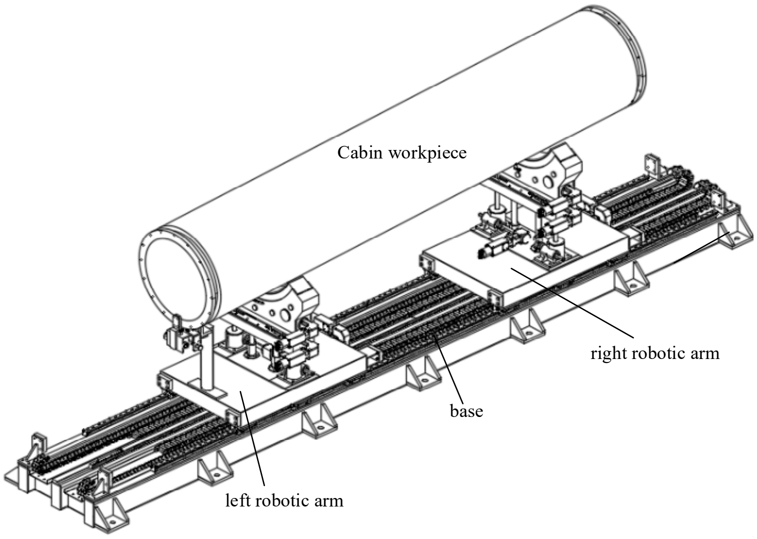

Figure 1. The cabin docking dual-arm robot.

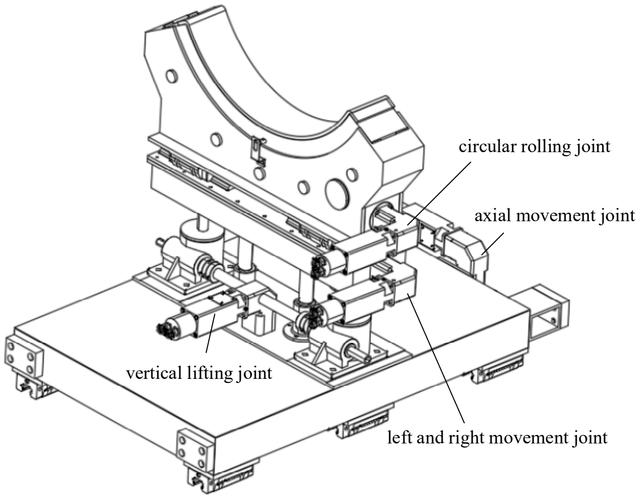

Figure 2. Components of the right robotic arm.

Compared with other algorithms, the Cuckoo Search (CS) algorithm is a new swarm optimization algorithm, which is heuristic and intelligent. There are also strong global search capabilities and fewer parameter settings [13–16]. Thus, a multi-objective Cuckoo Search Algorithm (MOCS) was proposed by Yang et al. [17]. The novel multi-objective improved Cuckoo optimization algorithm was built by Azizipanah et al. [18]. A set of non-dominated points were defined for the optimization problems. This algorithm was applied in the multi-objective static and dynamic optimization scheduling of a thermal-electric system, where the environmental objective was inconsistent with the economic objective. For the automotive sequencing problem of a mixed-model assembly line, the multi-objective Cuckoo Search algorithm was proposed by Wang et al. [19]. The algorithm was based on the record matrices. More factors were considered, such as part usage variation, workstation workload variation, idle time, overload time and model change cost. Meanwhile, two search strategies were also given, and the local and global search capabilities were enhanced. Later, the trajectory planning problems were solved by the Cuckoo Optimization Algorithm (COA), Cuttlefish Algorithm (CFA), Seagull Optimization Algorithm (SOA) and Tunicate Swarm Algorithm (TSA) [20].

From the above studies, less attention is paid to the optimization algorithm of the dual-arm robot. Therefore, on the basis of a modified Cuckoo Search algorithm, a multi-objective trajectory planning method is proposed. The trajectory is generated using a fifth-order B-spline curve, and the total consumption time and joint impact are minimized by the objective functions. The normalization comprehensive evaluation method is used for the solutions in the Pareto non-dominant, and the most comprehensive optimal solution is selected. The relative results are verified by the experiments. And the subsequent organization is given based on the above method. In Section 2, a kinematics model of the dual-arm robot is established. The trajectory is planned by the B-spline curve in Section 3. Later, a modified Cuckoo Search algorithm is proposed by the objective function and comprehensive optimal solution method, which can be found in Section 4. From Section 5, the effectiveness of the algorithm is verified by the simulation. The confirmatory experiment is given in Section 6. The conclusions are seen in Section 7.

2. Kinematic Analysis of Cabin Docking Dual-Arm Robot

2.1. Forward Kinematic Model

According to the organization, a simplified model of a dual-arm robot is constructed, and the relative coordinate systems are given in Figure 3. The relationship between the adjacent links of a dual-arm robot is established through the Denavit–Hartenberg (D-H) method. The base coordinate system is represented by $O_{base}x_{base}y_{base}z_{base}$. Two robotic arm coordinate systems $O_{L0}x_{L0}y_{L0}z_{L0}$ and $O_{R0}x_{R0}y_{R0}z_{R0}$ are established at the middle position. The axial movement joint, vertical lifting joint, left and right movement joint and circumferential rolling joint of the left robotic arm are, respectively, established with $O_{L1}x_{L1}y_{L1}z_{L1}$, $O_{L2}x_{L2}y_{L2}z_{L2}$, $O_{L3}x_{L3}y_{L3}z_{L3}$ and $O_{L4}x_{L4}y_{L4}z_{L4}$. The D-H parameters of the cabin docking dual-arm robot are given in Table 1.

On the basis of the D-H method, the terminal position matrix of the left robotic arm is as follows:

$$ {}^{L4}_{\text{base}}T = {}^{L0}_{\text{base}}\,T^{L1}_{L0}T^{L2}_{L1}T^{L3}_{L2}T^{L4}_{L3}T = \begin{bmatrix} 0 & 0 & 1 & -d_{L1} \\ -sin(\theta_L) & -cos(\theta_L) & 0 & -d_{L3} \\ cos(\theta_L) & -sin(\theta_L) & 0 & d_{L2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where ${}^{L0}_{base}T$ is the transition matrix from $O_{base}x_{base}y_{base}z_{base}$ to $O_{L0}x_{L0}y_{L0}z_{L0}$, ${}^{L1}_{L0}T$ is the transition matrix from $O_{L0}x_{L0}y_{L0}z_{L0}$ to $O_{L1}x_{L1}y_{L1}z_{L1}$, ${}^{L2}_{L1}T$ is the transition matrix from $O_{L1}x_{L1}y_{L1}z_{L1}$ to $O_{L2}x_{L2}y_{L2}z_{L2}$, ${}^{L3}_{L2}T$ is the transition matrix from $O_{L2}x_{L2}y_{L2}z_{L2}$ to $O_{L3}x_{L3}y_{L3}z_{L3}$, and ${}^{L4}_{L3}T$ is the transition matrix from $OO_{L3}x_{L3}y_{L3}z_{L3}$ to $O_{L4}x_{L4}y_{L4}z_{L4}$.

Similar methods are used for the right robotic arm, and the terminal position matrix can be obtained as follows:

$$ {}^{R4}_{\text{base}}T = \begin{bmatrix} 0 & 0 & -1 & d_{R1} \\ sin(\theta_R) & cos(\theta_R) & 0 & -d_{R3} \\ cos(\theta_R) & -sin(\theta_R) & 0 & d_{R2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2) $$

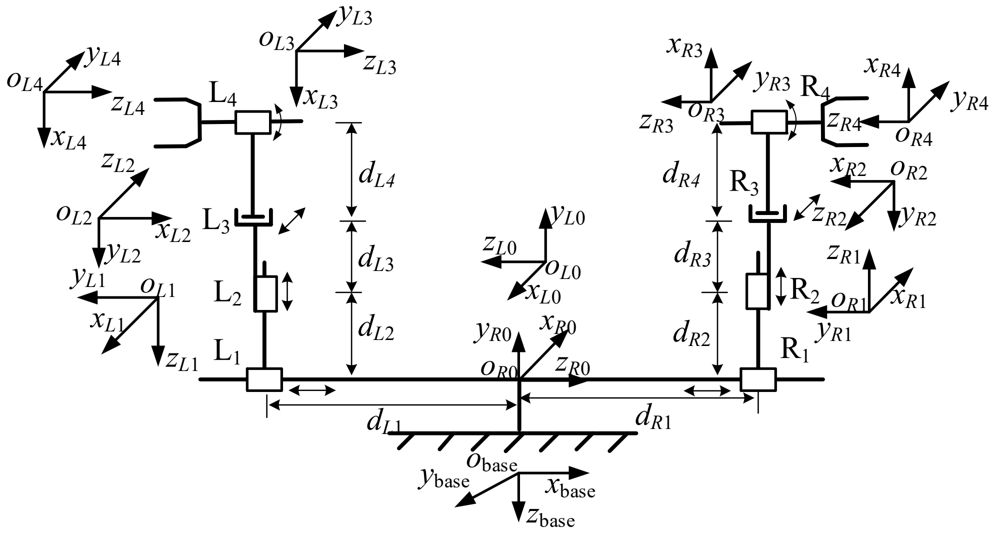

Figure 3. The simplified model and joint coordinate systems of the cabin docking dual-arm robot.

Table 1. D-H parameters for the cabin docking dual-arm robot.

Link	Joint Angle θ_i	Link Offset d_i	Link Length a_i	Link Angle α_i
Base-L$_0$	$\pi/2$	0	0	$-\pi/2$
L_1	0	d_{L1}	0	$\pi/2$
L_2	$-\pi/2$	d_{L2}	0	$\pi/2$
L_3	$\pi/2$	d_{L3}	d_L	$\pi/2$
L_4	θ_L	d_{L4}	0	0
Base-R$_0$	$\pi/2$	0	0	$\pi/2$
R_1	0	d_{R1}	0	$-\pi/2$
R_2	$\pi/2$	d_{R2}	0	$-\pi/2$
R_3	$-\pi/2$	d_{R3}	d_R	$-\pi/2$
R_4	θ_R	d_{R4}	0	0

2.2. Inverse Kinematics Model

The inverse kinematics model of the dual-arm rotor can be solved with the formula Paul inverse transformation method. The pose matrix equation of the right robotic arm is

$$ {}^{R4}_{\text{base}}T = \begin{bmatrix} n_{Rx} & o_{Rx} & a_{Rx} & p_{Rx} \\ n_{Ry} & o_{Ry} & a_{Ry} & p_{Ry} \\ n_{Rz} & o_{Rz} & a_{Rz} & p_{Rz} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3) $$

According to Equations (2) and (3), the forward kinematics formula for the right robotic arm can be obtained as follows:

$$ \begin{aligned} p_{\text{Rx}} &= d_{R1} \\ p_{\text{Ry}} &= -d_{R3} \\ p_{\text{Rz}} &= d_{R2} \\ yaw_R &= -\theta_R \end{aligned} \quad (4) $$

The same method is also used for the left robotic arm, the forward kinematics equation is

$$\begin{aligned} p_{\mathrm{L}x} &= -d_{L1} \\ p_{\mathrm{L}y} &= -d_{L3} \\ p_{\mathrm{L}z} &= d_{L2} \\ yaw_L &= \pi + \theta_L \end{aligned} \tag{5}$$

In view of the mapping relationship of Equations (4) and (5), the inverse kinematics transformation formulas of the right robotic arm and the left robotic arm can be established as follows:

$$\begin{aligned} d_{R1} &= p_{\mathrm{R}x} \\ d_{R3} &= -p_{\mathrm{R}y} \\ d_{R2} &= p_{\mathrm{R}z} \\ \theta_R &= -yaw_R \end{aligned} \tag{6}$$

$$\begin{aligned} d_{L1} &= -p_{\mathrm{L}x} \\ d_{L3} &= -p_{\mathrm{L}y} \\ d_{L2} &= p_{\mathrm{L}z} \\ \theta_L &= yaw_L \end{aligned} \tag{7}$$

3. Trajectory Planning Method Based on B-Spline Interpolation

3.1. Trajectory Planning Scheme

In Figure 4, in order to analyze the attitude adjustment process of the dual-arm robot, the left center coordinate system $O_1x_1y_1z_1$ of the cabin workpiece is established, and the right center coordinate system is $O_2x_2y_2z_2$. The axial movement direction of the cabin docking robot is defined as the Z-axis direction. The movement direction of the cabin docking robot is defined as the X-axis direction. The movement direction of the cabin docking robot along the vertical lifting joint is defined as the Y-axis. The distance between $O_{L4}x_{L4}y_{L4}z_{L4}$ and $O_1x_1y_1z_1$ is D_1, the distance between $O_{R4}x_{R4}y_{R4}z_{R4}$ and $O_2x_2y_2z_2$ is D_3, and the distance from $O_{L4}x_{L4}y_{L4}z_{L4}$ to $O_{R4}x_{R4}y_{R4}z_{R4}$ is D_2. According to the established coordinate system, the Pitch angle α, the Yaw angle β and the Roll angle γ are defined, which can be found in Figure 5. As shown in Figure 4, the dashed line is the cabin workpiece axis.

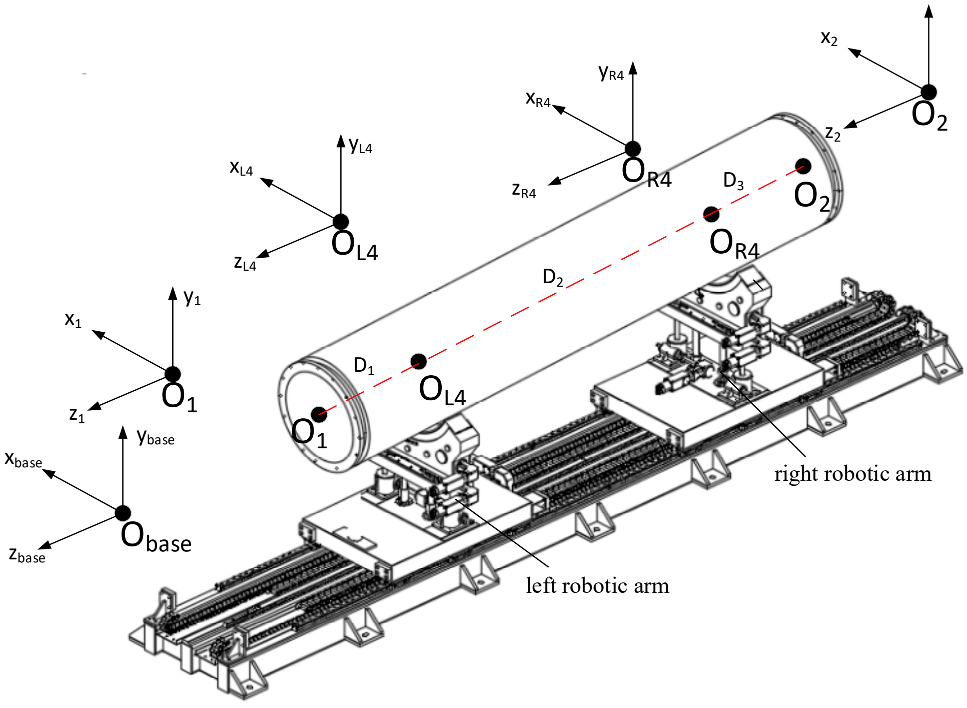

Figure 4. Distribution of coordinate systems for dual-arm robot docking in cabin section.

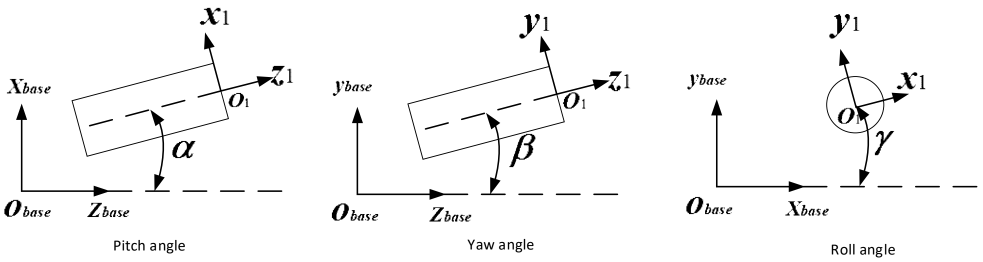

Figure 5. The attitude parameters of the cabin section workpiece.

For analyzing the relationship between the workpiece and the cabin docking robot, there are four situations: the initial equilibrium position, 0° pitch angle, 0° yaw angle and 0° roll angle. Their detailed descriptions are as follows:

(1) Initial equilibrium position

In the actual production process, it is generally required that the cabin docking robot is in the balanced spatial posture. $O_1x_1y_1z_1$, $O_{L4}x_{L4}y_{L4}z_{L4}$, $O_2x_2y_2z_2$ and $O_{R4}x_{R4}y_{R4}z_{R4}$ are in a straight line. The pitch angle α, the yaw angle β and the roll angle γ are set as 0°, as shown in Figure 6. The positional relationship in the z direction is

$$\begin{cases} (z_1 - z_{L4})^2 = D_1^2 \\ (z_{L4} - z_{R4})^2 = D_2^2 \\ (z_{R4} - z_2)^2 = D_3^2 \end{cases} \tag{8}$$

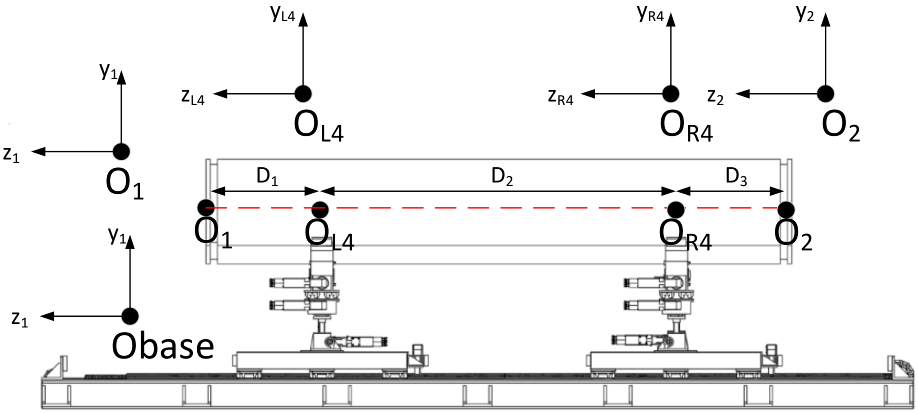

Figure 6. Coordinate system situation of the initial balance position.

(2) 0° pitch angle

Due to frictional force between the cabin docking robot and the workpiece, the vertical lifting of the docking robot is limited, and the value is often ±50 mm. When the pitch angle of the workpiece is changing, the end of the docking robot still contacts with the workpiece. Therefore, $O_1x_1y_1z_1$, $O_{L4}x_{L4}y_{L4}z_{L4}$, $O_2x_2y_2z_2$ and $O_{R4}x_{R4}y_{R4}z_{R4}$ are also on a straight line. As seen in Figure 7, the positional relationship of the coordinate system is

$$\begin{cases} (y_1 - y_{L4})^2 + (z_1 - z_{L4})^2 = D_1^2 \\ (y_{L4} - y_{R4})^2 + (z_{L4} - z_{R4})^2 = D_2^2 \\ (y_{R4} - y_2)^2 + (z_{R4} - z_2)^2 = D_3^2 \end{cases} \tag{9}$$

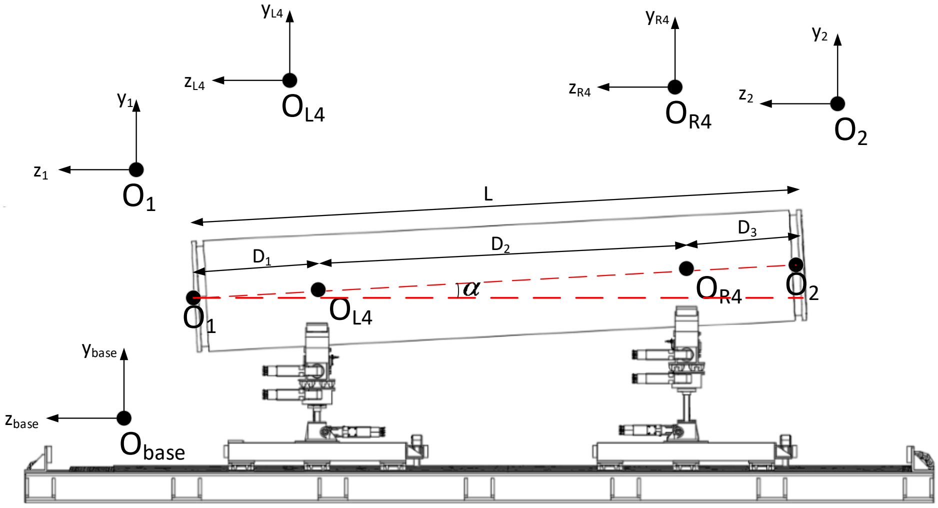

Figure 7. Coordinate system situation when the pitch angle is not at 0°.

As seen in Figure 7, there are two dashed lines in red, one is the current cabin workpiece axis, the other is the cabin axis at the balance position, and the angle between the two lines is attitude angle. The relationship between the pitch angle and the coordinate system parameters is

$$sin\alpha = \frac{y_{R4} - y_{L4}}{D_2} \tag{10}$$

(3) 0° yaw angle

The same method used for the pitch angle is also used for the yaw angle. From Figure 8, the positional relationship of the coordinate system is

$$\begin{cases} (x_1 - x_{L4})^2 + (z_1 - z_{L4})^2 = D_1^2 \\ (x_{L4} - x_{R4})^2 + (z_{L4} - z_{R4})^2 = D_2^2 \\ (x_{R4} - x_2)^2 + (z_{R4} - z_2)^2 = D_3^2 \end{cases} \tag{11}$$

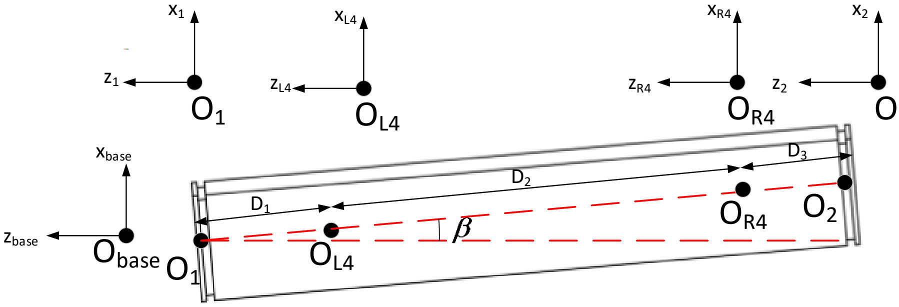

Figure 8. Coordinate system situation when the yaw angle is not at 0°.

The relationship between the pitch angle and the coordinate system parameters is

$$sin\beta = \frac{x_{R4} - x_{L4}}{D_2} \tag{12}$$

(4) 0° roll angle

In view of the workpiece, the positional relationships of various coordinate systems are not affected by the changing roll angle. Thus, all the coordinates are also on a straight line, which can be seen in Figure 9. However, in consideration of the bolt connection between the workpieces, the roll angle is also adjusted with a slight change in the cabin space pose.

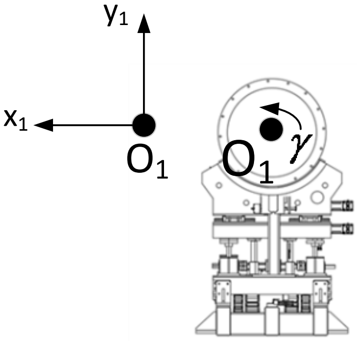

Figure 9. Coordinate system situation when roll angle γ is not at 0°.

Based on the above analysis, the adjustment processes for the position and attitude of the workpiece are independent of each other. It is assumed that the trajectory for the end of the left robotic arm is $K_{L4}(x_{L4}(t), y_{L4}(t), z_{L4}(t))$, and the end of the right arm is $K_{R4}(x_{R4}(t), y_{R4}(t), z_{R4}(t))$. The first formula for the trajectory planning scheme is

$$(x_{L4}(t) - x_{R4}(t))^2 + (y_{L4}(t) - y_{R4}(t))^2 + (z_{L4}(t) - z_{R4}(t))^2 = D_2^2 \tag{13}$$

In Equation (13), on the basis of the motion trajectory for the left robotic arm, the spatial position is aligned, the end of the right robotic arm is changed with the action of the left robotic arm. The distance from the end of the left robotic arm to the end of the right robotic arm is constant. In addition to participating in spatial position adjustment, the right robotic arm is also used to adjust the posture of the cabin segment. Thus, the motion trajectory of the right robotic arm can be divided into two parts: ${K_{R4}}^1({x_{R4}}^1(t), {y_{R4}}^1(t), {z_{R4}}^1(t))$ and ${K_{R4}}^2({x_{R4}}^2(t), {y_{R4}}^2(t), {z_{R4}}^2(t))$. The relationship is as follows:

$$K_{R4}(x_{R4}(t), y_{R4}(t), z_{R4}(t)) = K_{R4}^1(x_{R4}(t), y_{R4}(t), z_{R4}(t)) + K_{R4}^2(x_{R4}(t), y_{R4}(t), z_{R4}(t)) \tag{14}$$

where ${K_{R4}}^1({x_{R4}}^1(t), {y_{R4}}^1(t), {z_{R4}}^1(t))$ is used to follow the action of the left robotic arm, while ${K_{R4}}^2({x_{R4}}^2(t), {y_{R4}}^2(t), {z_{R4}}^2(t))$ reflects the attitude change of the cabin segment, and the relative formula is given as follows:

$$\begin{cases} x_{R4}^2(t) = x_{L4}(t) + D_2 cos\alpha(t) cos\beta(t) \\ y_{R4}^2(t) = y_{L4}(t) + D_2 cos\alpha(t) sin\beta(t) \\ z_{R4}^2(t) = z_{L4}(t) + D_2 cos\alpha(t) \end{cases} \tag{15}$$

where $\alpha(t)$ and $\beta(t)$ represent the pitch angle and yaw angle trajectory of the workpiece, respectively.

3.2. B-Spline Curve Construction

The B-spline curves are widely used for the robot trajectory planning due to the advantages, such as derivative continuity, piecewise processing and local support. The first and second derivatives of the B-spline curve are both continuous, which can meet the requirements of trajectory planning. The definition of a k-th degree B-spline curve is as follows:

$$p(u) = \sum_{i=0}^{n} c_i N_{i,k}(u) \tag{16}$$

where c_i is the control point, and $N_{i,k}(u)$ is a k-th degree B-spline basis function. There are several approaches for representing B-Spline base functions, which are often defined as the de Boor–Cox recursion formula [21,22]. The order of the B-spline curve is selected as five, the number of intermediate control points is five, three expresses the endpoint repetitions of control points, and the endpoint repetitions of node vectors is six. Then, the total number of control points is 11, and the element number in the node vector is 17. The initial and final pose of the cabin segment is expressed as follows:

$$\begin{aligned} p_o &= \left[p_o^x, p_o^y, p_o^z, p_o^\alpha, p_o^\beta, p_o^\gamma\right] \\ p_e &= \left[p_e^x, p_e^y, p_e^z, p_e^\alpha, p_e^\beta, p_e^\gamma\right] \end{aligned} \tag{17}$$

There are three repetitions for the control points at the beginning and end positions:

$$\begin{aligned} d_0^p = d_1^p = d_2^p = p_0 \\ d_8^p = d_9^p = d_{10}^p = p_e \end{aligned} \tag{18}$$

Therefore, the vector can be represented as

$$\mathbf{d} = \left[d_0^p, d_1^p, d_2^p, d_3^p, d_4^p, d_5^p, d_6^p, d_7^p, d_8^p, d_9^p, d_{10}^p\right] \tag{19}$$

The time nodes of the initial and final pose points have a duplicate degree of six. Therefore, the time node vector can be represented as follows:

$$\mathbf{t} = [0, 0, 0, 0, 0, 0, t_1, t_2, t_3, t_4, t_5, t_6, t_6, t_6, t_6, t_6, t_6] \tag{20}$$

The B-spline curve of pose motion trajectory is differentiated as velocity, acceleration and jerk. The relative expression is as follows:

$$\begin{aligned} v(t) &= \sum_{i=0}^{9} d_i^v \cdot N_{i,4}(t) \quad (i = 0, 1, \ldots, 9) \\ a(t) &= \sum_{i=0}^{9} d_i^a \cdot N_{i,3}(t) \quad (i = 0, 1, \ldots, 8) \\ j(t) &= \sum_{i=0}^{9} d_i^a \cdot N_{i,2}(t) \quad (i = 0, 1, \ldots, 7) \end{aligned} \tag{21}$$

The sequence of control points are

$$\begin{aligned} d_i^v &= \frac{k \cdot \left(d_{i+1}^p - d_i^p\right)}{t_{i+k+1}^p - t_{i+1}^p} & (i = 0, 1, \ldots, 9) \\ d_i^a &= \frac{(k-1) \cdot \left(d_{i+1}^v - d_i^v\right)}{t_{i+k}^v - t_{i+1}^v} & (i = 0, 1, \ldots, 8) \\ d_i^j &= \frac{(k-2) \cdot \left(d_{i+1}^a - d_i^a\right)}{t_{i+k-1}^a - t_{i+1}^a} & (i = 0, 1, \ldots, 7) \end{aligned} \tag{22}$$

k is the order of the position B-spline curve, and the value is five. The node vector corresponding to each control point is as follows:

$$\begin{aligned} \mathrm{t}^v &= [0, 0, 0, 0, 0, t_1, t_2, t_3, t_4, t_5, t_6, t_6, t_6, t_6, t_6] \\ \mathrm{t}^a &= [0, 0, 0, 0, t_1, t_2, t_3, t_4, t_5, t_6, t_6, t_6, t_6] \\ \mathrm{t}^j &= [0, 0, 0, t_1, t_2, t_3, t_4, t_5, t_6, t_6, t_6] \end{aligned} \tag{23}$$

According to Equations (21)–(23), the velocity, acceleration and jerk curves can be computed. The trajectory planning process of the fifth degree B-spline curve is shown in Figure 10.

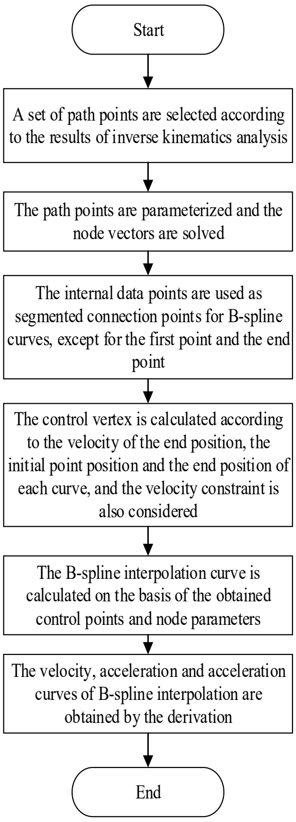
Start
A set of path points are selected according to the results of inverse kinematics analysis
The path points are parameterized and the node vectors are solved
The internal data points are used as segmented connection points for B-spline curves, except for the first point and the end point
The control vertex is calculated according to the velocity of the end position, the initial point position and the end position of each curve, and the velocity constraint is also considered
The B-spline interpolation curve is calculated on the basis of the obtained control points and node parameters
The velocity, acceleration and acceleration curves of B-spline interpolation are obtained by the derivation
End

Figure 10. Calculation process of trajectory planning for 5th degree B-spline curves.

4. The Modified Multi-Objective Cuckoo Search Algorithm

4.1. The Objective Function and Comprehensive Optimal Solution

During the docking process of the workpieces, the docking time should be shortened, which can ensure the efficiency. Therefore, the total consumption time of the dual-arm robot is selected as the first objective evaluation function:

$$S_1 = T = \sum t_i \tag{24}$$

where t_i is the node vector for the parametric variable of the B-spline.

In addition to the high work efficiency, the stability is also very important. However, the impacts are often generated when the dual-arm robot is working. In order to keep the motion trajectory smoother and more stable, the impact must be contained within a small range. Therefore, the second objective evaluation function is defined as follows:

$$S_2 = \sum_{j=1}^{N} \sqrt{\frac{1}{T}\int_0^T \left[\dddot{q}_j(t)\right]^2 dt} \tag{25}$$

where $\dddot{q}_j(t)$ refers to the jerk of the j-th joint.

After the objective evaluation functions are confirmed, the comprehensive optimal solution should also be given. However, the comprehensive optimal solution is closely linked to the multi-objective optimization. The significant feature of multi-objective optimization is that each objective function is mutually constrained. If the value of one objective function is optimal, the performance of other objective functions is often lost. In the feasible solution set of multi-objective optimization problems, there is no traditional absolute optimal solution, which can ensure that all objective functions are the optimal value simultaneously. Then, it is often determined by the dominance relationship, and the non-dominated solution is considered as the final effective solution, which is also called as the Pareto optimal solution [23]. The normalization of various indicators is used for the Pareto optimal solution. In this method, all the indicators are normalized, the influence of units is eliminated, and a comprehensive evaluation value for the non-dominated solutions can be obtained. Each set of Pareto solutions is calculated for the satisfaction indicators of all objective functions. And the satisfaction indicator is minimized as the optimal solution, which is defined as follows:

$$\delta = \frac{1}{m}\sum_{i=1}^{m} w_i \left(\frac{f_{i\max} - f_i}{f_{i\max} - f_{i\min}} \right) \tag{26}$$

where m represents the number of objective functions, and f_{imax} and f_{imin} are the maximum and minimum values of the i-th objective function. $w = [w_1, w_2]$ represents the weight coefficient, which serves as a trade-off criterion between time optimality and impact optimality. By adjusting the ratio between w_1 and w_2, different weights for time and impact can be achieved.

4.2. Traditional Multi-Objective Cuckoo Search Algorithm

The Cuckoo Search algorithm (CS) is an optimization algorithm, and it is used to simulate the special habit of cuckoo laying eggs. The Multi-Objective Cuckoo Search algorithm (MOCS) is an improvement of the CS algorithm, where a single objective function is replaced by multiple conflicting objectives for the global search [24]. The search path and location update equations of the algorithm are as follows:

$$X_i^{t+1} = X_i^t + \alpha \oplus \mathrm{Le'vy}(\beta) \tag{27}$$

where X_i^{t+1} represents the location of the i-th nest in generation $t + 1$, $\alpha > 0$ denotes the step size, X_i^t is the random location of the nest in generation t, and $\oplus$ denotes the point-to-point multiplication. $\mathrm{Le'vy}(\beta)$ is the randomized step size, which can be calculated as

$$\mathrm{Le'vy} \sim u = t^{-1-\beta}, 0 \le \beta \le 2 \tag{28}$$

In MOCS, the complete $\mathrm{Le'vy}$ equation is as follows:

$$\alpha_0 \left(x_j^{(t)} - x_i^{(t)} \right) \oplus \mathrm{levy}(\beta) \sim 0.01 \frac{u}{|\ v|^{1/\beta}} (x_j^{(t)} - x_i^{(t)}) \tag{29}$$

where u and v are normally distributed.

4.3. Modified Multi-Objective Cuckoo Search Algorithm

Compared with other multi-objective optimization algorithms, there are some advantages of the MOCS algorithm, such as few parameters and strong optimization ability. However, the drawbacks are not ignored, such as slow convergence speed in the later period and uneven distribution of optimal solutions. Aiming at the above drawbacks, a modified multi-objective Cuckoo algorithm (IMOCS) is proposed for chaotic initialization of the population. The diversity and uniformity of the initial population are ensured, the adaptive step size is introduced during the Levi flight process, which can effectively develop the algorithm's global and local optimization capabilities, and the improvement plan will be described in detail.

4.3.1. Improvement of Initial Population Generation Method

In the traditional MOCS algorithm, the initial population is generated randomly, and the equation is as follows:

$$X_i^0 \sim N(0,1) \tag{30}$$

where X_i^0 represents the position of the i-th bird's nest at the initial moment.

The random initialization may lead to an uneven population distribution, a large number of poor solutions are generated, and the convergence speed of the algorithm becomes slower. Thus, the characteristics of chaos principles are integrated, the population diversity is improved without losing individual randomness, a foundation is provided for a further effective global search. Bernoulli chaotic mapping is used to initialize the population for chaos, and its expression is

$$Z_{k+1} = \begin{cases} Z_k/(1-\rho) & Z_k \in (0, 1-\rho] \\ (Z_k - 1 + \rho)/\rho & Z_k \in (1-\rho, 1) \end{cases} \tag{31}$$

where Z_k represents the current value of the chaotic sequence generated in the k-th generation, and ρ is the control parameter.

The steps of chaos initialization are as follows:

Step 1: For the locations of S nests in n-dimensional space, a random n-dimensional vector is generated as the location of the first nest. Among them, $y_i \in (0, 1), 1 \leq i \leq d$.

Step 2: each dimension of $Y_1{}^{(0)}$ is iterated for S-1 times, S-1 chaotic variables $Y_2{}^{(0)}, \ldots, Y_{\text{S-1}}{}^{(0)}, Y_{\text{S}}{}^{(0)}$ are generated.

Step 3: the positions of the nests in the solution space are generated by using the S chaotic variables, which is described as

$$x_{id} = L_d + y_{id}\,(U_d - L_d) \tag{32}$$

where U_d and L_d represent the upper and lower bounds of the d-th dimension of the search space.

4.3.2. Improvement of Step Size in Cuckoo Algorithm

There is a fixed step size factor α_0 in the traditional cuckoo algorithm, the adaptive adjustment of step size cannot be achieved during the optimization process, and the convergence speed of the algorithm is slow. In the optimization process, if the step size factor α_0 is large, the algorithm has strong exploration ability, but the high-precision global optimal solutions cannot be obtained. If the step size factor α_0 remains small, the algorithm will require more iterations for target accuracy. In addition, biological heuristic algorithms originate from the characterization of natural biological behavior habits, and there is significant randomness.

In response to the above shortcomings, an adaptive Levi flight mechanism is used for the global search, and the Levy flight step size continuously decreases with iteration. The improved algorithm has a larger step size factor in the early stage of optimization, thereby the search space in the early stage of the algorithm is expanded, and the global search ability is improved. During the optimization process, the step size is reduced, and the local search performance of the algorithm is improved. The step size updating equation is shown as follows:

$$\alpha_0 = \alpha_{max}\left(1 - \left(\frac{iter - 1}{Maxiter}\right)^2\right) \tag{33}$$

where α_{max} represents the maximum step length, *iter* denotes the current iteration number, and Maxiter is the total iteration number.

4.4. Flowchart of Trajectory Optimization Algorithm

Firstly, the parameterized B spline curve is used to express the trajectory. According to the optimization objective and the range of adjusting mechanism parameters, the optimization algorithm adaptation functions and constraints are constructed to establish the multi-objective optimization trajectory. The multi-objective optimization algorithm is used for the set of Pareto optimal solutions, and the index normalized evaluation method is given for the comprehensive optimal solution. The relative flowchart is shown in Figure 11.

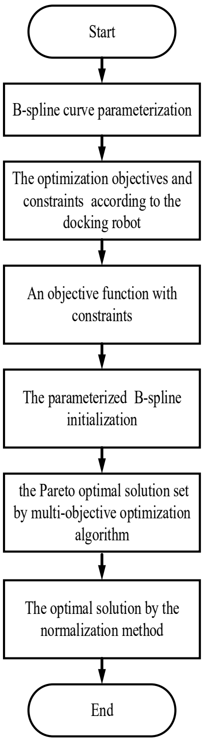
Start
B-spline curve parameterization
The optimization objectives and constraints according to the docking robot
An objective function with constraints
The parameterized B-spline initialization
the Pareto optimal solution set by multi-objective optimization algorithm
The optimal solution by the normalization method
End

Figure 11. Flow chart of the trajectory optimization.

5. Simulation Analysis

To verify the effectiveness of the multi-objective trajectory optimization algorithm, the trajectory planning simulation is conducted on the docking robot. At the same time, in order to verify the modified multi-objective cuckoo search algorithm, the NSGA-II algorithm, MOPSO algorithm, MOCS algorithm and IMOCS algorithm are, respectively, used in the simulation experiments, comparative simulation experiments are carried out, and the experimental results are analyzed.

5.1. Parameter Configuration

The start point and end point of the position for the simulation process are set as

$$\begin{aligned} P_o &= [775, -35, -1315, 0.2, -0.2, 190] \\ P_e &= [745, -5, 1515, 0, 0, 180] \end{aligned} \tag{34}$$

The elements from left to right in the vector are the X-axis position, the Y-axis position, the Z-axis position, the pitch angle, the yaw angle and the roll angle. The parameters of a docking robot are selected as the constraint conditions, and the relative value is given in Table 2.

5.2. Results and Discussion

Compared with the performance of different Multi-Objective Evolutionary Algorithms (MOEAs), the indicators are divided into counting indicators, convergence indicators, diversity indicators and comprehensive indicators [25]. The counting index is used to count the number or proportion of non-dominated solutions, which is based on the Overall Non-dominated Vector Generation (ONVG) [26]. The convergence index is chosen as the Generational Distance (GD) [27], and the average distance from the solution to the closest reference point is calculated. The diversity selection spatial indicator spacing (SP) [28] is used to reflect the diversity of the optimal solution in the spatial distribution. To ensure the rationality of the simulation, the population size is set to 150, and after 100 iterations, the Pareto front is shown in Figure 12.

Table 2. Design parameters of cabin docking robot.

Number	Parameter	Value
1	Roll angle range	±15°
2	Z-axis direction travel range	±600 mm
3	Y-axis direction travel range	±50 mm
4	X-axis direction travel range	±50 mm
5	Roll angle speed range	0~6.8°/s
6	Z-axis direction speed range	0~400 mm/s
7	Y-axis direction speed range	0~25 mm/s
8	X-axis direction speed range	0~2.5 mm/s
9	Roll angle acceleration range	0~240°/s^2
10	Z-axis direction acceleration range	0~100 mm/s^2
11	Y-axis direction acceleration range	0~100 mm/s^2
12	X-axis direction acceleration range	0~100 mm/s^2

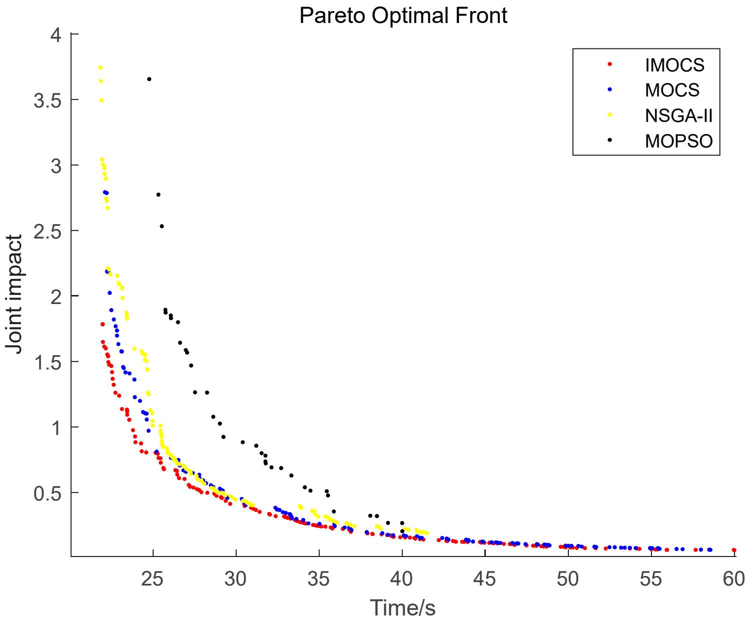

Figure 12. Pareto frontier for robot arm trajectory planning.

(1) Comparing the number of non-dominated solutions, the non-dominated solutions of the NSGA-II algorithm, the MOPSO algorithm, the MOCS algorithm and the IMOCS algorithm are 150, 36, 133 and 150, respectively. Therefore, in terms of the number for non-dominated solutions, the IMOCS algorithm is superior to the MOPSO and MOCS algorithms, and it is comparable to the NSGA-II algorithm.

(2) Comparing the convergence of non-dominated solutions, the GD values of the NSGA-II algorithm, the MOPSO algorithm, the MOCS algorithm and the IMOCS algorithm are 0.0065, 0.0220, 0.0041 and 0.0019, respectively. As a result, the convergence of the IMOCS algorithm is better.

(3) Comparing the spacing of non-dominated solutions, the SP values of the NSGA-II algorithm, the MOPSO algorithm, the MOCS algorithm and the IMOCS algorithm are 0.0049, 0.0420, 0.0064 and 0.0037, respectively, so the spacing of the IMOCS algorithm is better.

In order to analyze the time shock distribution relationship in the non-dominated solution set more intuitively, 11 sets of solutions are uniformly selected from the non-dominated solution set obtained by the IMOCS optimization algorithm, as shown in Table 3. The total time of cabin docking is increasing from 21.5743 s to 59.5678 s. With the increasing total time, the joint impact decreases from 3.9448 to 0.0753. The normalization method is used to solve the optimal solutions in these 11 groups. The weight coefficients are set according to Equation (26), which is shown in Table 4.

Table 3. The 11 sets of IMOCS trajectory planning non-dominated solutions.

Serial Number	Total Time	Joint Impact
1	21.5743	3.9448
2	23.4057	1.5797
3	24.7217	1.0077
4	27.5831	0.6339
5	31.9816	0.3965
6	34.3913	0.3104
7	38.1978	0.2363
8	42.6796	0.1724
9	44.9217	0.1483
10	51.1223	0.1025
11	59.5678	0.0753

Table 4. Normalized weight coefficient setting table.

Type	w_1	w_2
1	1	1
2	0	1
3	1	0

From Table 4, three coefficient types are set using the normalized evaluation method. Type 1 is a weight coefficient $[w_1, w_2] = [1, 1]$, and the comprehensive optimal solution of the total time and joint impact objective functions in the non-dominated solution are selected. Type 2 is a weight coefficient $[w_1, w_2] = [0, 1]$, and the solution with the smallest total time among non-dominated solutions is selected. Type 3 is the weight coefficient $[w_1, w_2] = [1, 0]$, and the solution with the smallest joint impact among non-dominated solutions is selected. The corresponding solutions can be obtained from Table 4 based on the selection method of these solutions. Then, the simulations are made for the end X-axis motion trajectory of the left robotic arm, the end Y-axis motion trajectory of the left robotic arm, and the end Z-axis motion trajectory of the left robotic arm.

5.2.1. Simulation of X-axis End Motion Trajectory of Left Robotic Arm

The X-axis displacement, X-axis velocity and X-axis acceleration for the end of the left robotic arm are shown in Figures 13–15, respectively. The displacement in the X-axis direction is uniformly reducing from 775 mm to 745 mm. The X-axis direction speed of Type 2 is relatively high, and a maximum value 2.39 mm/s can be found, but the maximum value is not more than 2.5 mm/s. The X-axis acceleration of Type 2 is relatively high, and a maximum value for 1.39 mm/s^2 appears but not exceeding 100 mm/s^2.

5.2.2. Simulation of Y-axis End Motion Trajectory of Left Robotic Arm

The Y-axis displacement, Y-axis velocity and Y-axis acceleration for the end of the left robotic arm are shown in Figures 16–18, respectively. The displacement in the Y-axis direction is uniformly reducing from -35 mm to -5 mm. The Y-axis direction speed of Type 2 is relatively high, and a maximum value 3.09 mm/s can be seen but not exceeding 25 mm/s. The Y-axis acceleration of Type 2 is relatively high, and there is a maximum value 0.91 mm/s^2, but the value is not more than 100 mm/s^2.

5.2.3. Simulation of Z-axis End Motion Trajectory of Left Robotic Arm

The Z-axis displacement, Z-axis velocity and Z-axis acceleration for the end of the left robotic arm are shown in Figures 19–21, respectively. The displacement in the Z-axis direction is uniformly reduced from -1315 mm to -1515 mm. The Z-axis direction speed of Type 2 is relatively high, and a maximum value 18.19 mm/s is reaching but not exceeding 400 mm/s. The Z-axis acceleration of Type 2 is relatively high, and a maximum value 4.81 mm/s^2 can be found but not more than 100 mm/s^2.

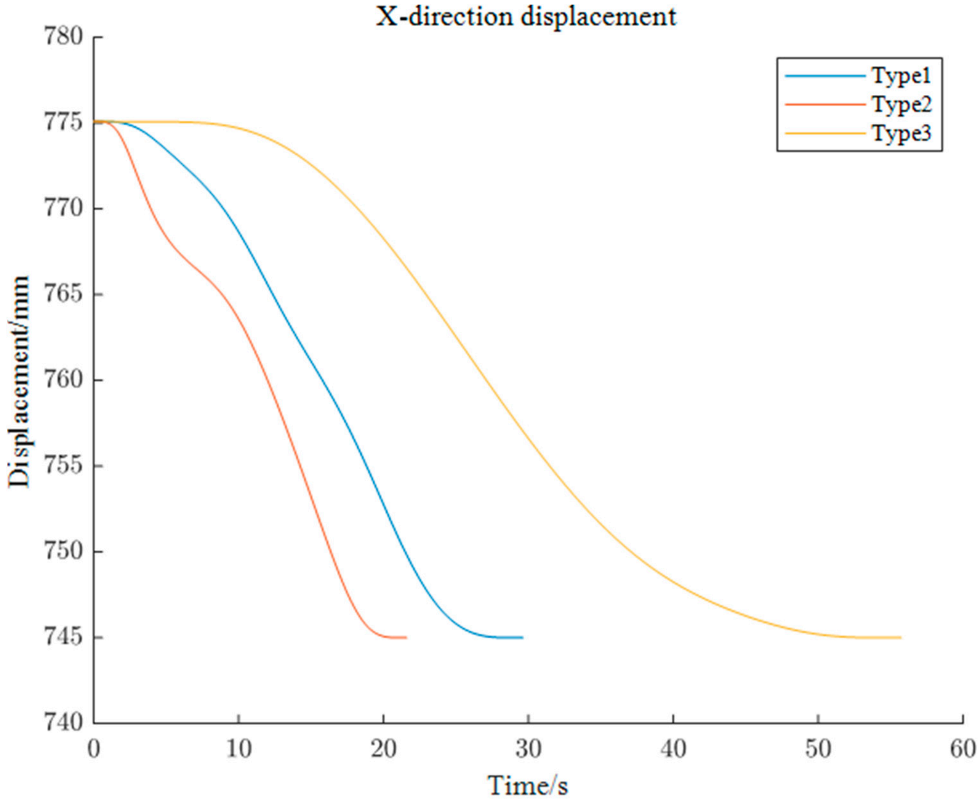

Figure 13. Left robotic arm X-axis displacement curve.

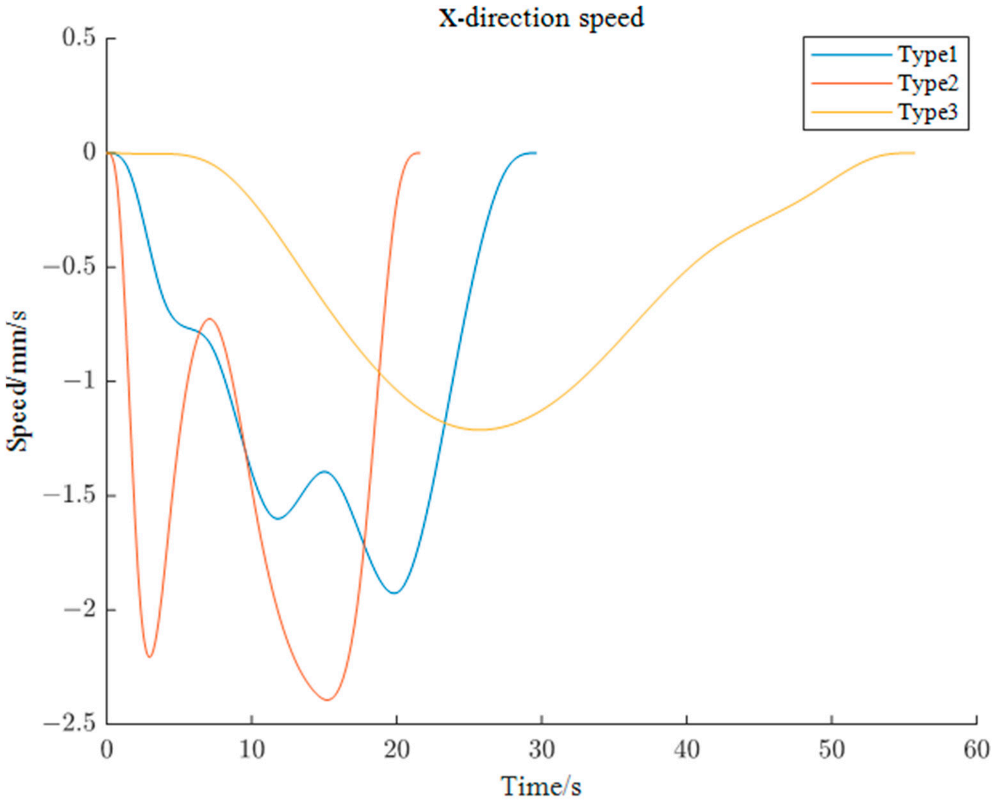

Figure 14. Left robotic arm X-axis speed curve.

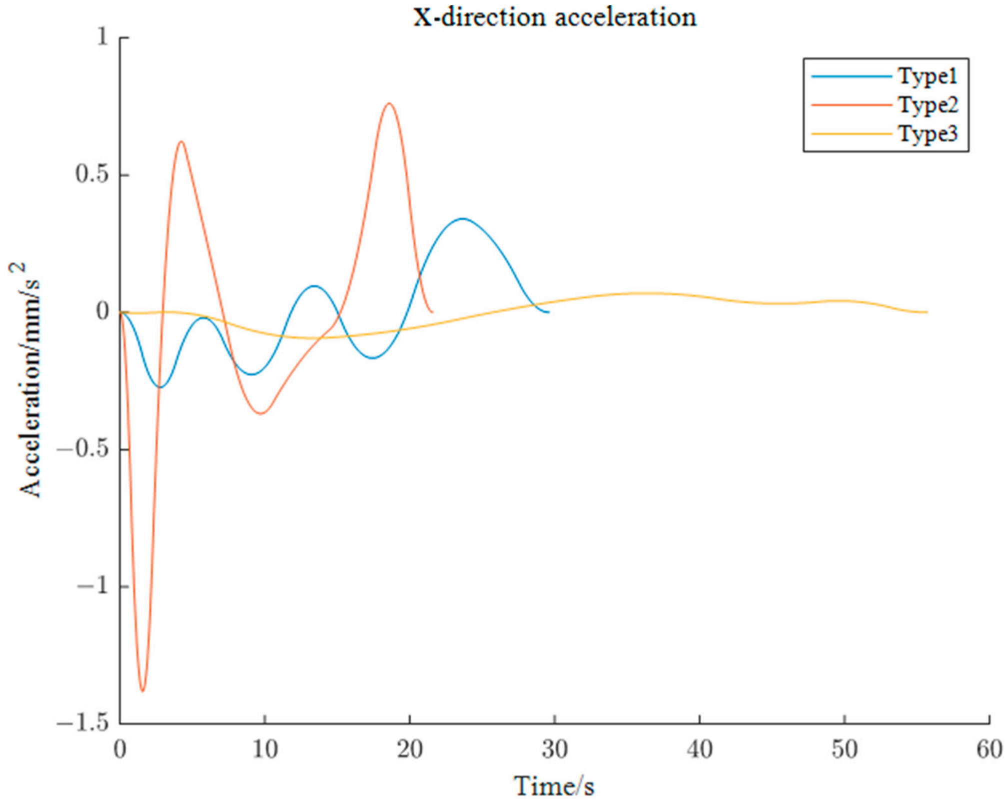

Figure 15. Left robotic arm X-axis acceleration curve.

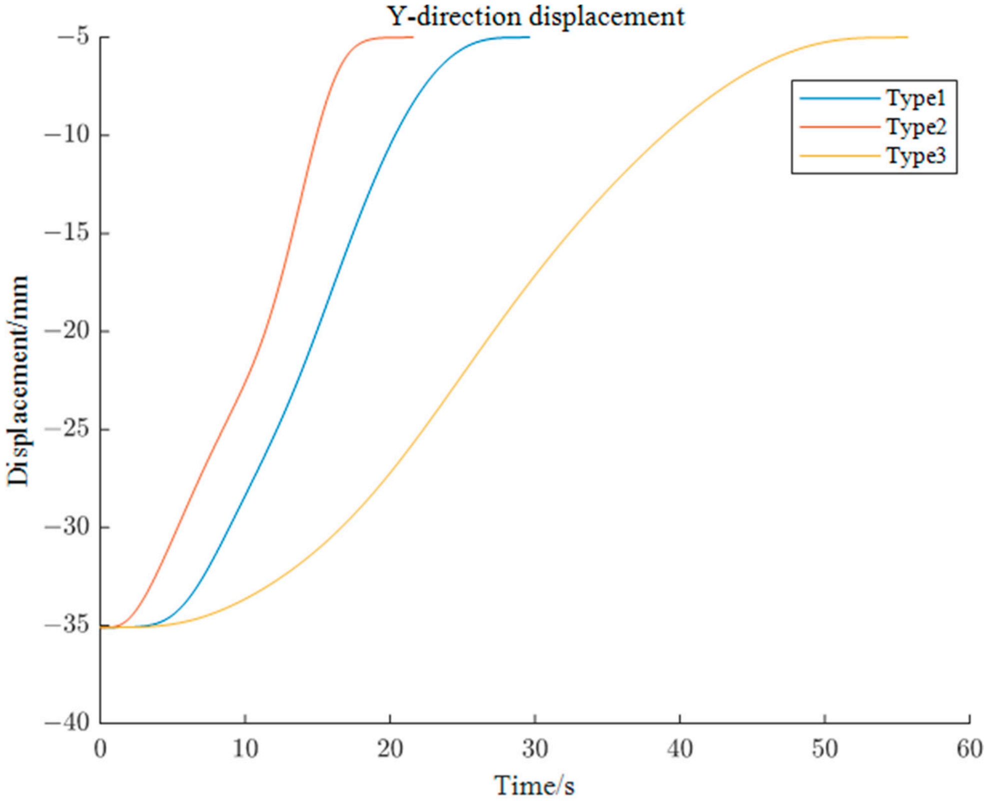

Figure 16. Left robotic arm Y-axis displacement curve.

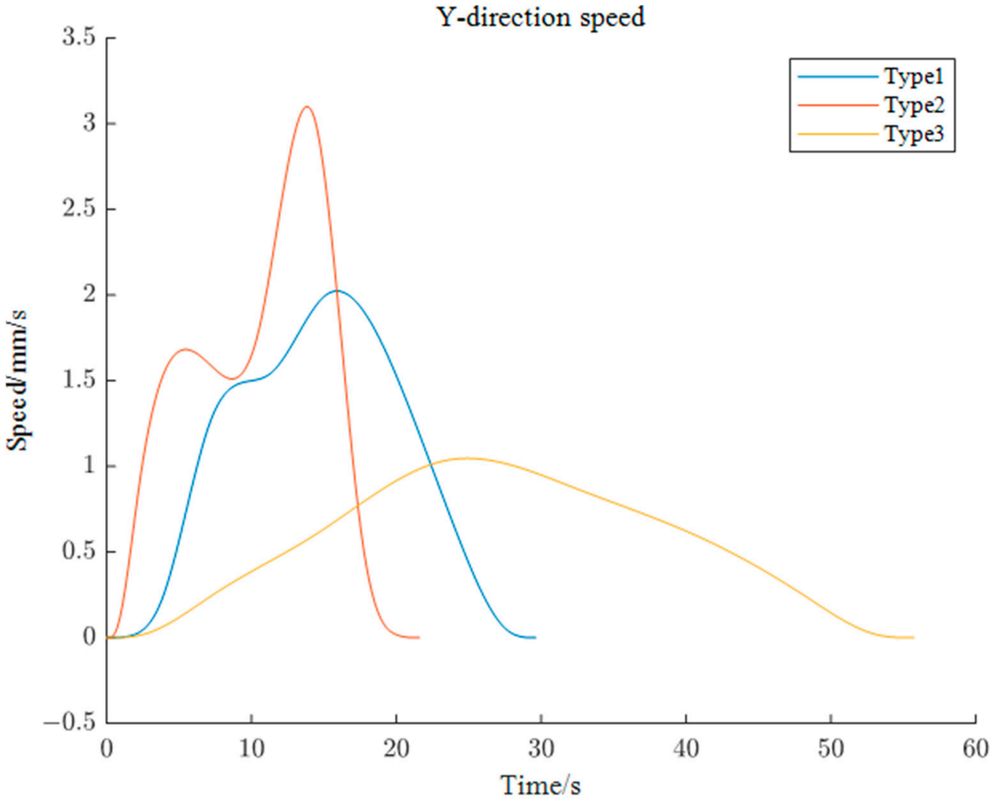

Figure 17. Left robotic arm Y-axis speed curve.

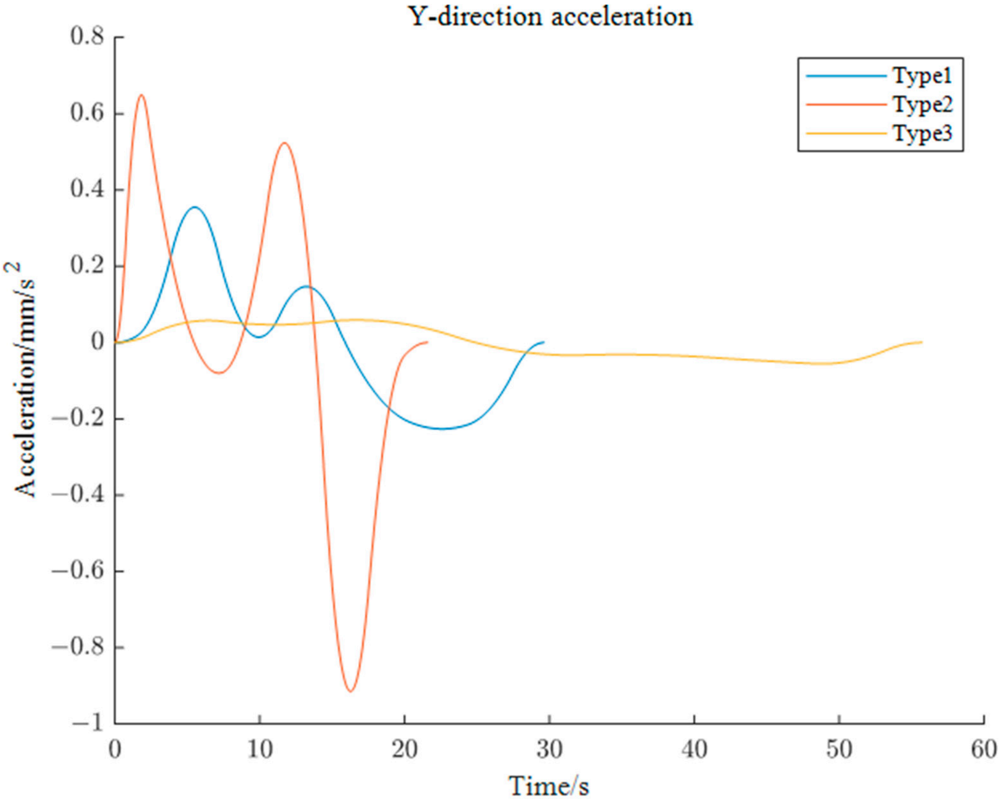

Figure 18. Left robotic arm Y-axis acceleration curve.

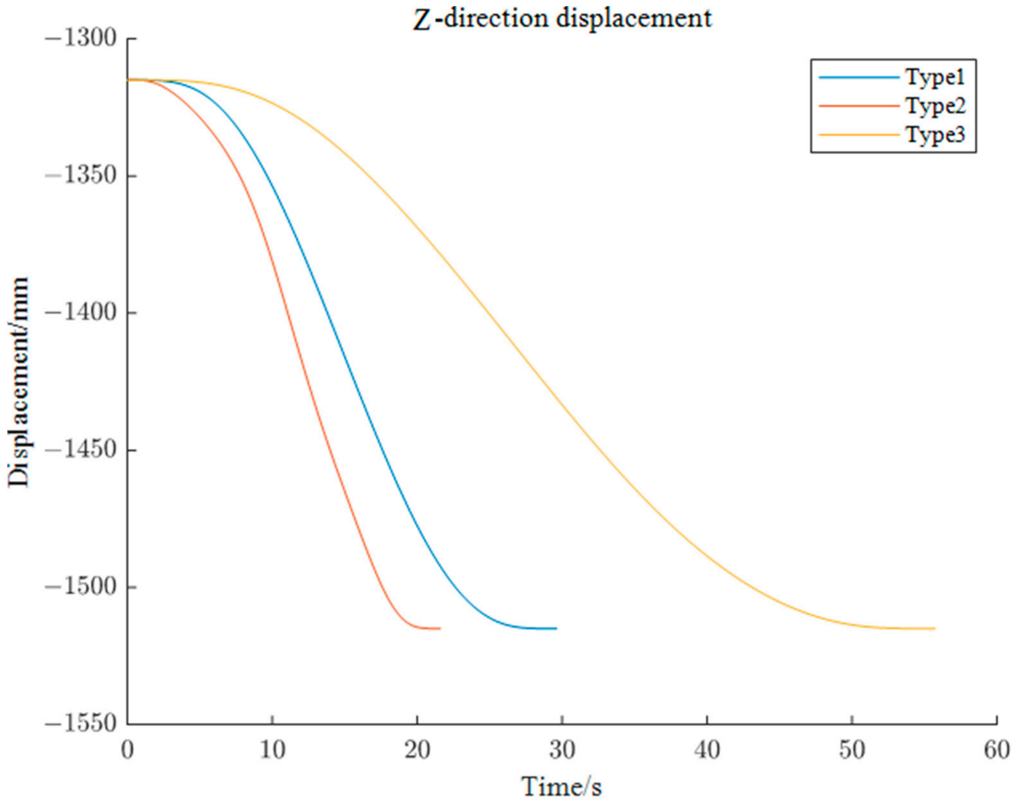

Figure 19. Left robotic arm Z-axis displacement curve.

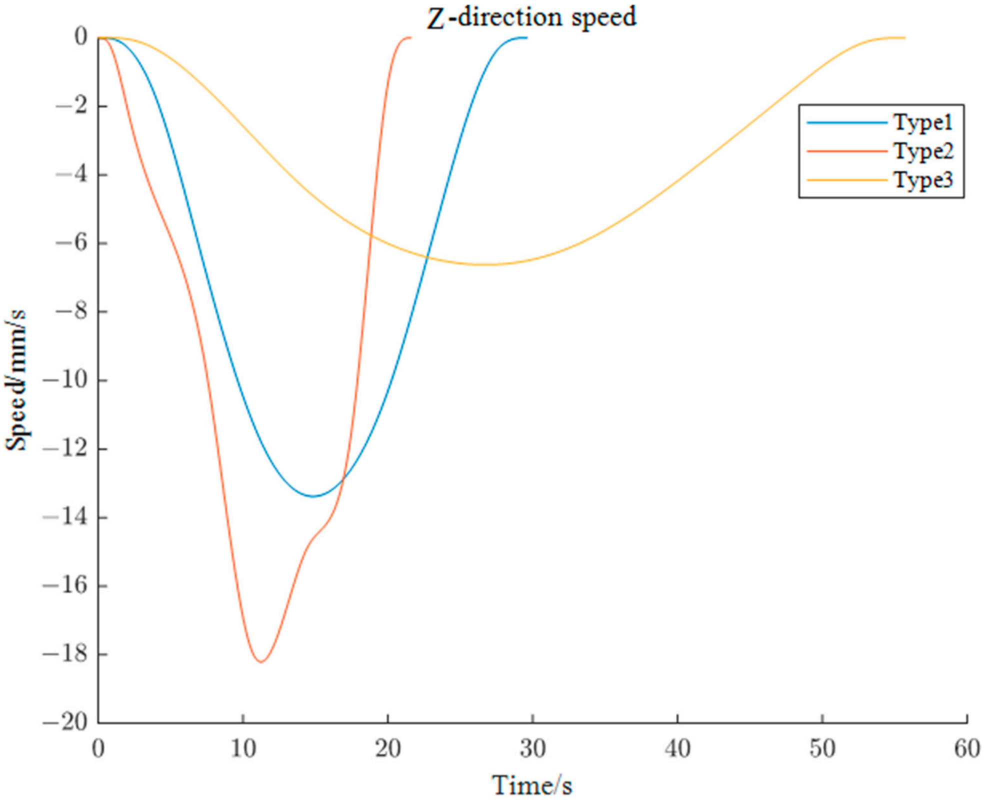

Figure 20. Left robotic arm Z-axis speed curve.

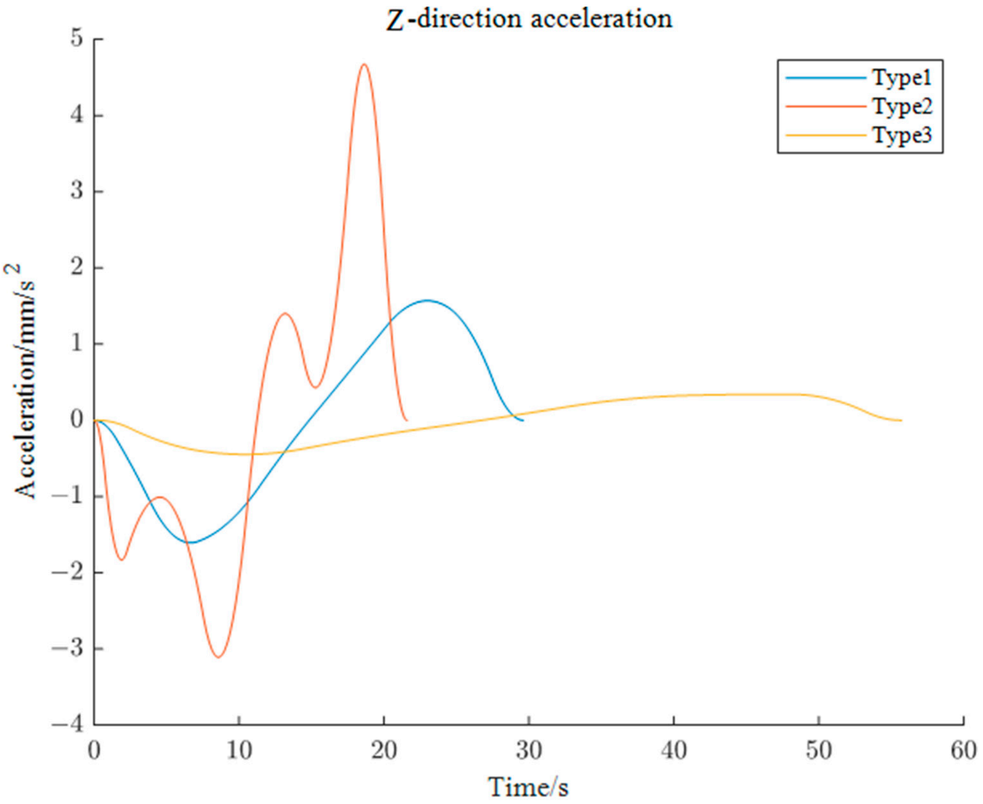

Figure 21. Left robotic arm Z-axis acceleration curve.

6. Experimental Confirmation

In order to verify the effectiveness of the algorithm, trajectory planning and tracking experiments of the cabin docking robot are conducted based on the cabin docking experimental platform. The cabin docking experimental platform is shown in Figure 22. One of the cabin-docking robots is selected as the research object for trajectory planning and trajectory following experiments.

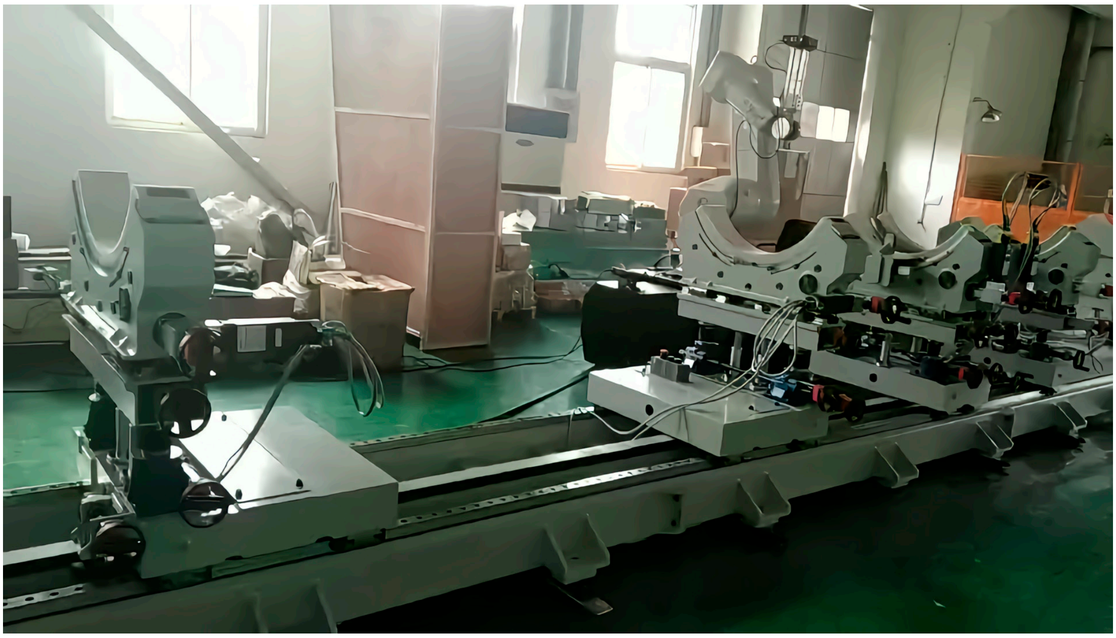

Figure 22. Docking experiment platform.

In order to ensure the accuracy of the trajectory following control, explosion-proof servo motors are selected for driving the cabin docking experimental platforms. Based on the experimental platform, the trajectory generated by the modified multi-objective cuckoo algorithm is taken as the target trajectory, and the actual position is used as the target trajectory. Then, the experiments for the X-axis trajectory planning and the Z-axis trajectory planning are carried out.

6.1. X-axis Trajectory Planning

The starting position in the X-axis direction of the cabin docking robot is set as 4.04 mm, and the stopping position is −1.514 mm. The planned target trajectory is compared with the collected actual trajectory, which is given in Figure 23. A comparison between the planned target speed and the collected actual speed is shown in Figure 24. It is found that the maximum and minimum tracking errors of the X-axis trajectory are 0.014 mm and 0 mm, respectively. The maximum tracking error of the X-axis speed is 0.465 mm/s, and the minimum value is 0 mm. The error is very small.

6.2. Z-axis Trajectory Planning

The starting position in the Z-axis direction of the cabin docking robot is set as −1591.03 mm, and the stopping position is −1601.47 mm. The comparison between the planned target trajectory and the collected actual trajectory is shown in Figure 25. The relative results for the speed can be seen in Figure 26. The maximum and minimum tracking errors of the Z-axis trajectory are 0.018 mm and 0 mm, respectively. The maximum tracking error of the speed is 0.364 mm/s, and the minimum value is 0 mm. The error is also very small, and the effectiveness of the modified cuckoo search algorithm is proven.

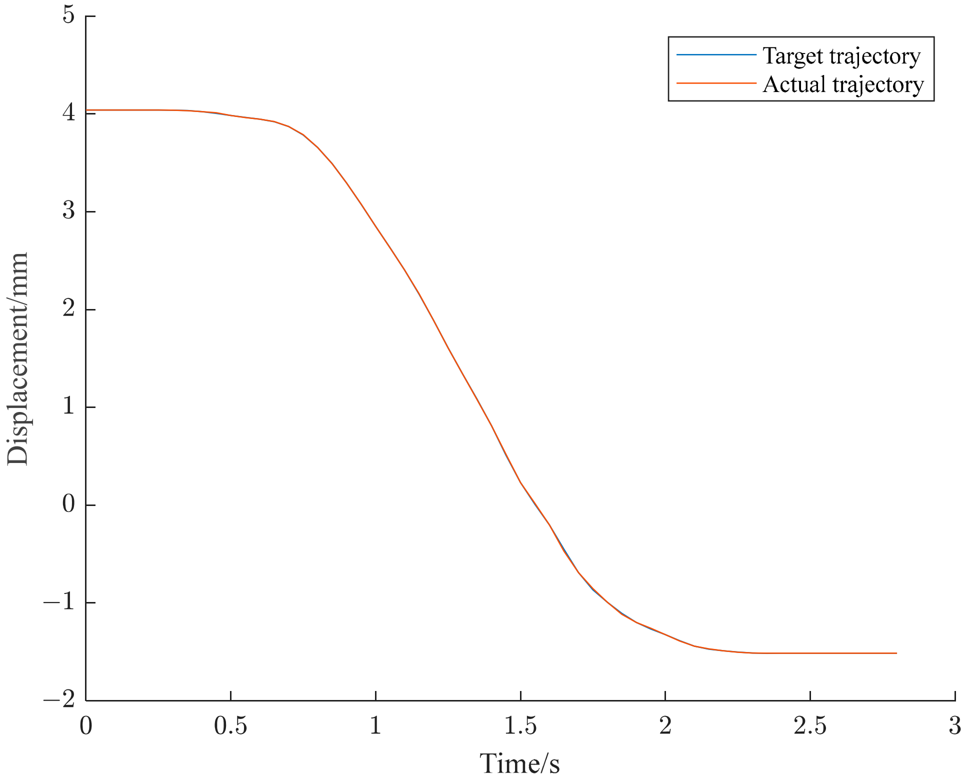

Figure 23. X-axis trajectory planning curve of cabin docking robot.

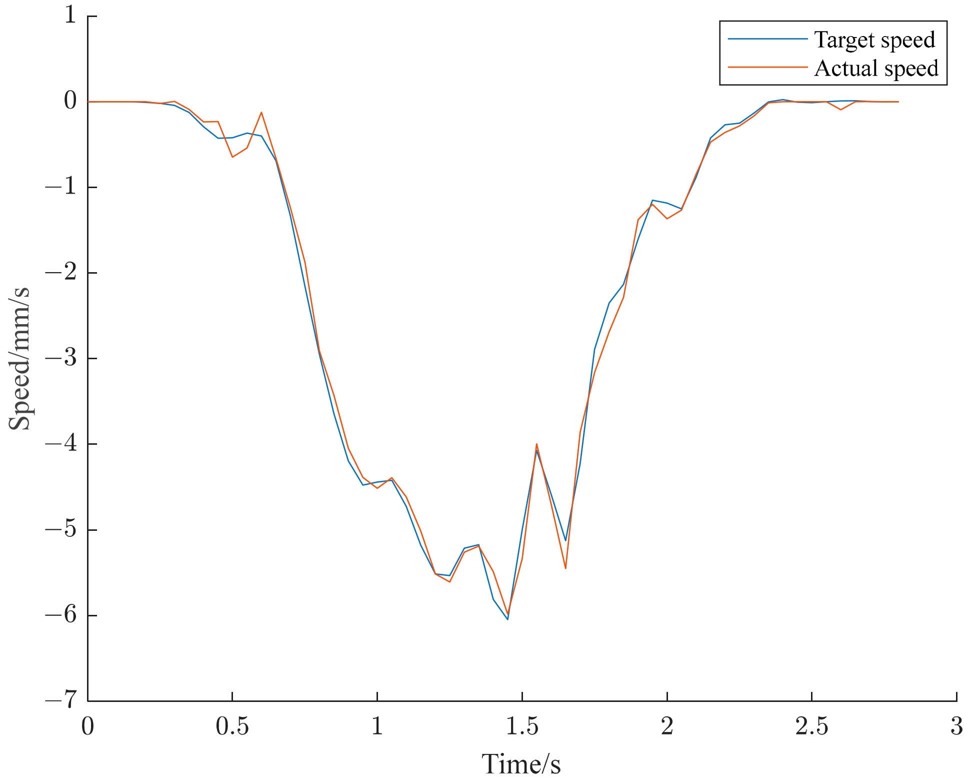

Figure 24. Speed planning curve in the X-axis direction for the docking robot.

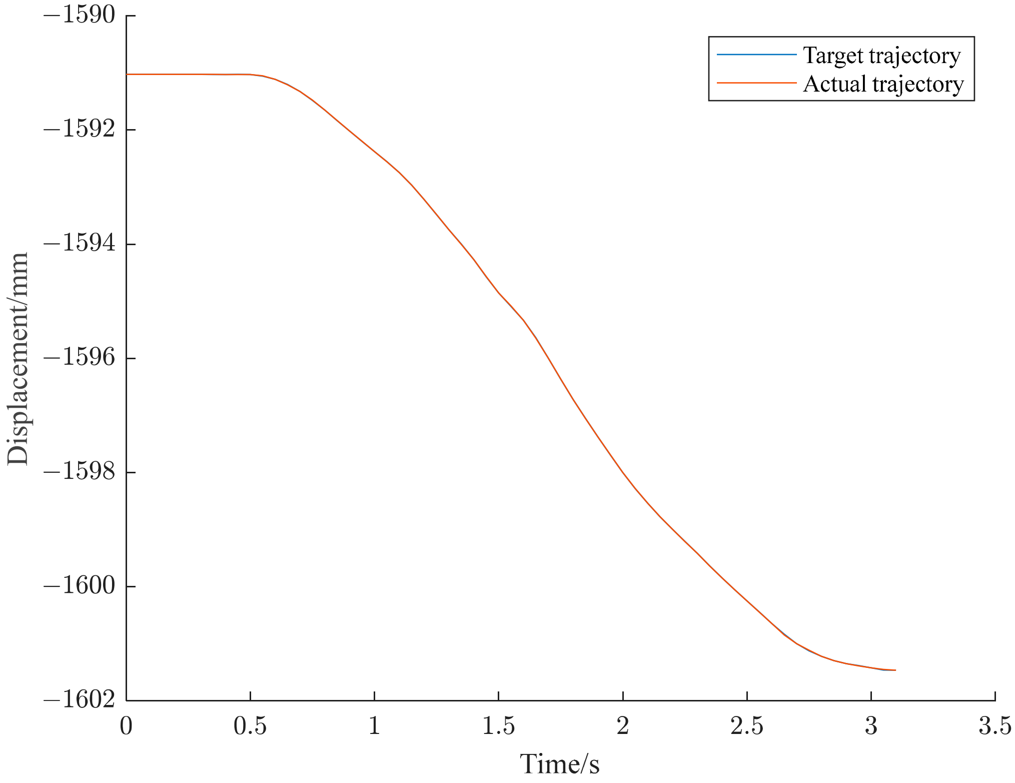

Figure 25. Z-axis trajectory planning curve of cabin docking robot.

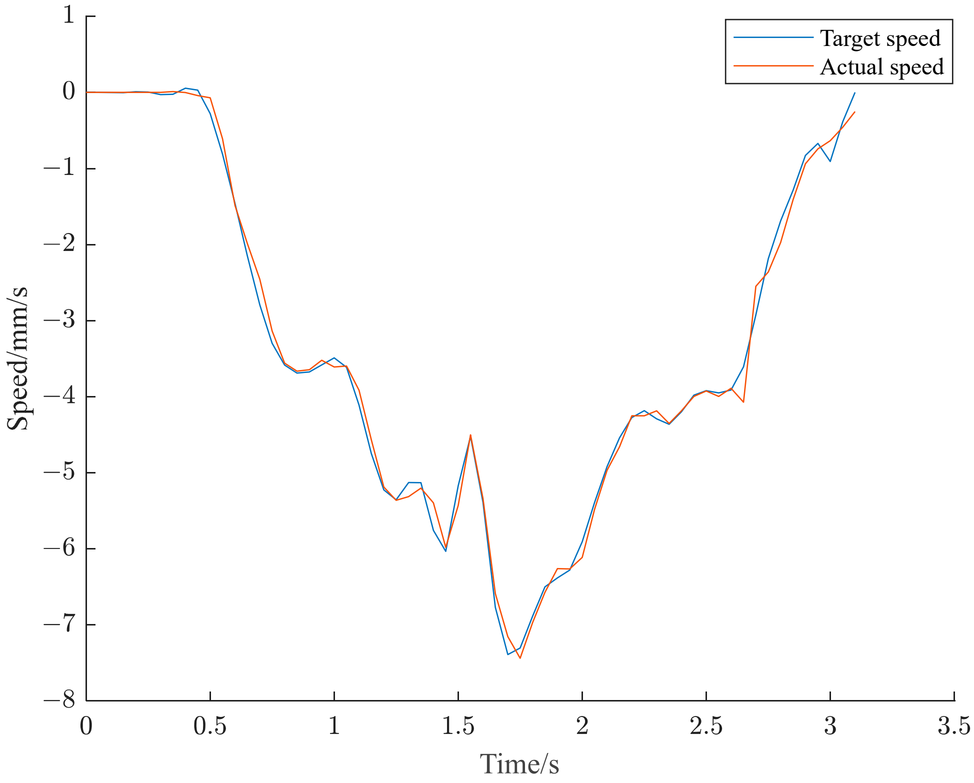

Figure 26. Speed planning curve in the Z-axis direction for the docking robot.

7. Conclusions

In order to ensure the accuracy and reliability of cabin docking, a multi-objective trajectory planning method for the dual-arm robot is proposed. The relative results are summarized as follows:

(1) A forward kinematics model of the cabin docking dual-arm robot is established based on the D-H modeling method, and the inverse kinematics analysis is also conducted. Based on the analysis of four different cabin attitudes, the relationship between the end trajectory and attitude parameters of the cabin-docking robot is studied. A trajectory planning scheme for the robotic arm is designed, and B-spline curve construction and parameterization methods are introduced.

(2) The selection method of objective function and comprehensive optimal solution is introduced. On the basis of the traditional multi-objective cuckoo algorithm, a modified multi-objective cuckoo search algorithm is proposed from two aspects: initial population generation method and cuckoo algorithm step size.

(3) The performance of the NSGA-II algorithm, MOPSO algorithm, MOCS algorithm and IMOCS algorithm are compared using simulations. Three types of normalization methods are selected: the optimal total time and joint target synthesis, the minimum total time and the minimum joint impact. The simulation analysis is conducted on the end effector of the cabin docking robot in the X-axis and Y-axis directions. The trajectory planning effect in the Z-axis direction is smooth in all three directions, and the maximum speed and acceleration in the X-axis direction are 2.39 mm/s and 1.39 mm/s^2. The maximum operating speed in the Y-axis direction is 3.09 mm/s, and the maximum operating acceleration in the X-axis direction is 0.91 mm/s^2. The maximum operating speed in the Z-axis direction is 18.19 mm/s, and the maximum operating acceleration in the Z-axis direction is 4.81 mm/s^2. The simulation results can meet expectations, and the values are within the constraint range.

(4) On the basis of the cabin docking experimental platform, trajectory planning and tracking time of the end effector of the cabin docking robot in the X-axis and Z-axis directions are carried out. The maximum trajectory tracking error in the X-axis direction

of the cabin docking robot is 0.014 mm, and the maximum velocity tracking error is 0.465 mm/s. The maximum trajectory tracking error in the Z-axis direction of the cabin docking robot is 0.018 mm, and the maximum velocity tracking error is 0.364 mm/s, which proves the effectiveness and feasibility of the modified algorithm.

Author Contributions: Conceptualization, R.L. and F.P.; methodology, R.L.; software, R.L.; validation, R.L.; formal analysis, R.L.; investigation, R.L.; resources, R.L.; data curation, R.L.; writing—original draft preparation, R.L.; writing—review and editing, F.P.; visualization, F.P.; supervision, F.P.; project administration, F.P.; funding acquisition, F.P. All authors have read and agreed to the published version of the manuscript.

Funding: We would like to express our gratitude to the National Natural Science Foundation of China (No.61973036) and the Science and Technology Cooperation Project of Yunnan Province (202003AD150002).

Data Availability Statement: Data are contained within the article and are available from the corresponding author on reasonable request.

Conflicts of Interest: There are no any personal circumstances or interest that may be perceived as inappropriately influencing the representation or interpretation of reported research results in this paper.

References

1. Rüscher, O.; Mayländer, H. Automated alignment and marry-up of aircraft fuselage sections with a final assembly line. *SAE Trans.* **2001**, *110*, 173–178. [CrossRef]
2. Bond, J. Mission critical: Precision in the air and on the ground. *Mod. Mater. Handl.* **2013**, *68*, 48–50.
3. Kim, J.-J.; Lee, J.-J. Trajectory optimization with particle swarm optimization for manipulator motion planning. *IEEE Trans. Ind. Inform.* **2015**, *11*, 620–631. [CrossRef]
4. Tian, L.; Collins, C. An effective robot trajectory planning method using a genetic algorithm. *Mechatronics* **2003**, *14*, 455–470. [CrossRef]
5. Wang, H.; Wang, H.; Huang, J.; Zhao, B.; Quan, L. Smooth point-to-point trajectory planning for industrial robots with kinematical constraints based on high-order polynomial curve. *Mech. Mach. Theory* **2019**, *139*, 284–293. [CrossRef]
6. Wang, S.; Wu, S.; Kang, C.; Li, X. Trajectory planning of a parallel manipulator based on kinematic transmission property. *Intel. Serv. Robot.* **2015**, *8*, 129–139. [CrossRef]
7. Liu, L.; Chen, C.; Zhao, X.; Li, Y. Smooth trajectory planning for a parallel manipulator with joint friction and jerk constraints. *Int. J. Control Autom. Syst.* **2016**, *14*, 1022–1036. [CrossRef]
8. Chen, Z.; Wang, J.; Li, S.; Ren, J.; Wang, Q.; Cheng, Q.; Li, W. An Optimized Trajectory Planning for Welding Robot. *IOP Conf. Ser. Mater. Sci. Eng.* **2018**, *324*, 012009. [CrossRef]
9. Gasparetto, A.; Zanotto, V. A technique for time-jerk optimal planning of robot trajectories. *Robot. Comput. Integr. Manuf.* **2007**, *24*, 415–426. [CrossRef]
10. Chen, D.; Li, S.; Wang, J.; Feng, Y.; Liu, Y. A multi-objective trajectory planning method based on the improved immune clonal selection algorithm. *Robot. Comput. Integr. Manuf.* **2019**, *59*, 431–442. [CrossRef]
11. Lan, J.; Xie, Y.; Liu, G.; Cao, M. A Multi-Objective Trajectory Planning Method for Collaborative Robot. *Electronics* **2020**, *9*, 859. [CrossRef]
12. Wang, Z.; Li, Y.; 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*, 81–95. [CrossRef]
13. Wang, Y.-J.; Xing, K.; She, B.; Liu, Y.; Chen, T.; Hu, G.; Wu, Q. Mixed Gaussian stochastic inversion based on a hybrid of cuckoo algorithm and Markov chain Monte Carlo. *Chin. J. Geophys.* **2021**, *64*, 2540–2553. [CrossRef]
14. Jiang, Y.; Shuai, L.; Wu, Q.; Cao, Y. Optimization of resonant parameters of bidirectional DC electronic transformer based on cuckoo algorithm. *Acta Energiae Solaris Sin.* **2021**, *42*, 485–494. [CrossRef]
15. Huang, B.Y.; Zhang, Y.X.; Zhao, L. Research on Fault Diagnosis Method of Rolling Bearings Based on Cuckoo Search Algorithm and Maximum Second Order Cyclostationary Blind Deconvolution. *J. Mech. Eng.* **2021**, *57*, 99–107. [CrossRef]
16. Lai, Y.W.; Zhang, J. Urban Bus Scheduling Optimization Based on Simulated Anneal-adaptive Cuckoo Search Algorithm. *J. Transp. Syst. Eng. Inf. Technol.* **2021**, *21*, 183–189. [CrossRef]
17. Yang, X.S.; Deb, S. *Multiobjective Cuckoo Search for Design Optimization*; Elsevier Science Ltd.: Amsterdam, The Netherlands, 2013.
18. Niknam, T.; Zare, M.; Gharibzadeh, M.; Azizipanah-Abarghooee, R. Multi-objective short-term scheduling of thermoelectric power systems using a novel multi-objective improved cuckoo optimization algorithm. *Iet Gener. Transm. Distrib.* **2014**, *13*, 1546–1550. [CrossRef]
19. Wang, H.; Sheng, B.; Lu, Q.; Luo, R.; Fu, G. A Multi-Objective Cuckoo Search Algorithm Based on the Record Matrix for a Mixed-Model Assembly Line Car-Sequencing Problem. *IEEE Access* **2020**, *8*, 76453–76470. [CrossRef]

20. Singh, G.; Banga, V.K. Kinematics and trajectory planning analysis based on hybrid optimization algorithms for an industrial robotic manipulators. *Soft Comput.* **2022**, *26*, 11339–11372. [CrossRef]
21. De Boor, C. On Calculation with B-spline. *J. Approx. Theory* **1972**, *6*, 50–62. [CrossRef]
22. Cox, M.G. The numerical evaluation of B-spline. *J. Inst. Maths. Appl.* **1972**, *10*, 134–149. [CrossRef]
23. Kahagalage, S.; Turan, H.H.; Jalalvand, F.; El Sawah, S. A novel graph-theoretical clustering approach to find a reduced set with extreme solutions of Pareto optimal solutions for multi-objective optimization problems. *J. Glob. Optim.* **2023**, *86*, 467–494. [CrossRef]
24. Zhang, X.S.; Liu, J.; Luo, S.B. An improved multi-objective cuckoo search algorithm for airfoil aerodynamic optimization design. *Acta Aeronaut. Astronaut. Sin.* **2019**, *40*, 122500. (In Chinese) [CrossRef]
25. Wang, L.P.; Ren, Y.; Qiu, Q.C.; Qiu, F.Y. Survey on Performance Indicators for Multi-objective Evolutionary Algorithms. *Chin. J. Comput.* **2021**, *44*, 1590–1619. (In Chinese) [CrossRef]
26. Van Veldhuizen, D.A.; Lamont, G.B. On measuring multi-objective evolutionary algorithm performance. In Proceedings of the 2000 Congress on Evolutionary Comutation, La Jolla, CA, USA, 16–19 July 2000; Volume 204, p. 211.
27. Van Veldhuizen, D.A. Lamont G B. Evolutionary computation and convergence to a Pareto front. In Proceeding of the Late-breaking Paper at the Genetic Programming 1998 Conference, Madison, WI, USA, 22–25 July 1998; pp. 221–228.
28. Schoxtt, J.R. Fault-Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's Thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH, USA, 1995.

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.