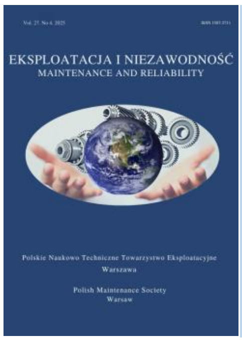


Eksploatacja i Niezawodnosc – Maintenance and Reliability
Volume 27 (2025), Issue 4
journal homepage: http://www.ein.org.pl

Article citation info:
Liu Q, Sun R, Bu X, Hanajima N, Ding W, Optimal Trajectory Planning Method for Handling Robots Based on Multi-objective Particle Swarm Optimization Guided by Evolutionary Information, Eksploatacja i Niezawodnosc – Maintenance and Reliability 2025: 27(4) http://doi.org/10.17531/ein/202990

Optimal Trajectory Planning Method for Handling Robots Based on Multi-objective Particle Swarm Optimization Guided by Evolutionary Information

Indexed by: Web of Science Group

Qunpo Liu[a], Rui Sun[a,*], Xuhui Bu[a], Naohiko Hanajima[b], Weiping Ding[c]

[a] Henan Polytechnic University, China
[b] Muroran Institute of Technology, Japan
[c] Nantong University, China

Highlights

- A novel EIGMOPSO optimizes handling robot trajectories for time, energy, and impact.
- Proposes a regionally dynamic stratification strategy enhancing performance stability.
- Designs region-guided layered optimization to maintain diversity and global search.
- Develops a two-stage archive strategy ensuring solution uniformity and convergence.

Abstract

This paper addresses the trajectory optimization and reliability challenges of 6-DOF handling robots by proposing a multi-objective particle swarm optimization method guided by evolutionary information (EIGMOPSO). The method optimizes trajectory planning in terms of time, energy consumption, and smoothness to enhance operational reliability and mechanical durability. To overcome the limitations of traditional MOPSO, a regionally dynamic stratification strategy based on evolutionary capability assessment is proposed, classifying the population into regions by evaluating fitness, diversity, and stability. A layered optimization mechanism dynamically adjusts exploration and exploitation processes, improving global search capability. Additionally, a dynamic two-stage archive maintenance strategy ensures high-quality solutions. Experimental results demonstrate that EIGMOPSO significantly improves operational efficiency, reduces mechanical wear and energy consumption, and enhances system maintainability, making it well-suited for handling robots in industrial environments.

Keywords

multi-objective optimization, particle swarm optimization, handling robot, evolutionary information guidance, trajectory planning

1. Introduction

6-DOF handling robots have become integral to modern industrial automation, particularly within the realms of intelligent manufacturing [1][2], high-efficiency production [3][4], and intelligent control [5][6]. Significant advancements have been made in these areas of research. Trajectory planning stands as a crucial issue in the field of robotics, involving the design of an optimal motion trajectory for a robot while satisfying boundary constraints. Due to their high flexibility and precision, handling robots are widely used in complex tasks such as material handling, assembly, and palletizing [7]. To accomplish these tasks efficiently, the objective is to optimize the motion trajectory of robot's six joints, achieving optimal performance metrics while adhering to physical constraints. A well-planned motion trajectory not only enhances operational precision but also minimizes mechanical wear, energy consumption, and the likelihood of system failure, contributing

(*) Corresponding author.
E-mail addresses: Q. Liu (ORCID: 0000-0002-2157-0278) lqpny@hpu.edu.cn, R. Sun (ORCID: 0009-0009-7190-2785) ruis121@163.com, X. Bu (ORCID: 0000-0001-5752-1091) buxuhui@gmail.com, N. Hanajima (ORCID: 0000-0002-4707-853X) hana@muroran-it.ac.jp, W. Ding (ORCID: 0000-0002-3180-7347) dwp9988@163.com,

to improved reliability and reduced maintenance costs. This process necessitates a comprehensive consideration of multiple factors—including path accuracy, energy consumption, motion smoothness, and real-time performance—and plays an irreplaceable role in industrial applications. Thus, research on trajectory planning is of great significance for achieving autonomous operations of handling robots [8].

With the continuous increase in industrial production demands, the adaptability requirements for handling robots under various complex operational conditions are also rising, particularly in scenarios such as precision assembly, complex material handling, and high-speed operations [9]. These conditions impose more stringent requirements on trajectory planning to ensure that handling robots can reliably perform tasks in complex environments. Trajectory planning for robots typically involves two key aspects: trajectory generation and trajectory optimization. The former provides the prerequisite conditions for trajectory planning, while the latter is an effective approach to enhance trajectory performance and fully utilize the capabilities of robotic arms. Trajectory generation is generally achieved by interpolating between any two given poses to establish a smooth trajectory for the robotic arm. Trajectory planning can be divided into Cartesian space planning and joint space planning, depending on the planning space. Liu et al. [10] proposed a trajectory planning method for obstacle avoidance in Cartesian space. However, because trajectory planning in Cartesian space involves real-time solving and the described paths have a continuous correspondence with paths in joint space, there is a problem of singularity. To avoid such singularity issues and to more intuitively evaluate the motion state of the manipulator, trajectory planning is often performed in joint space. In joint space, cubic polynomials [11][12] are popular due to their computational simplicity; however, their discontinuous joint angular acceleration can lead to mechanical system impacts. Higher-order polynomials, such as quintic and septic polynomials [13][14], provide smooth acceleration, avoiding impacts and vibrations, but their higher-order nature may result in trajectory distortion. Similarly, cubic B-spline curves [15][16] also suffer from non-smooth acceleration, which can lead to accumulated tracking errors and even resonance. In contrast, the seven times non-uniform B-spline [17] trajectory planning method ensures continuous and smooth jerk, while also maintaining low velocity and acceleration values, achieving high trajectory planning accuracy. This makes it a stable and reliable trajectory planning method.

In the field of trajectory optimization, obtaining a motion trajectory that satisfies both joint physical constraints and achieves optimal performance metrics has been a long-standing extensively studied topic. The initial strategies for robot trajectory optimization were time-oriented, aiming to enhance the efficiency of industrial robot production by minimizing the overall operational time [18]. With the continuous increase in industrial production demands, more optimization objectives have been proposed to evaluate robotic motion performance, including energy optimization aimed at reducing system energy consumption [19] and impact optimization to minimize joint wear and tear [20]. However, with the ever-increasing requirements for robotic arm operations, optimizing a single performance metric is no longer sufficient to meet technological needs. Therefore, optimization should be carried out according to real operating conditions, considering multiple performance metrics to ensure that trajectory planning aligns with practical application requirements. Early research has extensively explored energy-impact optimization, as both metrics are crucial in practical production settings. Xia [21] proposed a trajectory optimization method for energy and impact minimization for rehabilitation robots based on a multiple crow search algorithm (MCSA), significantly enhancing the energy efficiency and motion smoothness of rehabilitation robots. Time-impact optimization is a common multi-objective optimization approach aimed at improving efficiency, extending machine life. Wu et al. [22] aimed to minimize both the motion time and joint impact of robotic arms and proposed a trajectory optimization approach using an improved butterfly optimization algorithm (IBOA), effectively reducing both the motion time and joint impact of robotic arms. Time-energy-impact optimization better adapts to complex engineering applications and is classified under multi-objective trajectory optimization. The presence of multiple objectives and constraints increases the complexity of the planning process compared to single-objective and dual-objective optimizations. Traditional mathematical optimization methods often struggle to identify optimal solutions in such scenarios. Swarm intelligence algorithms have proven to be an effective approach to address these challenges. These algorithms can simultaneously optimize multiple objectives and select one or a set of optimal solutions as the final result based on actual operating conditions. Wang et al. [23] proposed an improved non-dominated sorting genetic algorithm (INSGA-II) for multi-objective trajectory optimization of robotic arms. This approach enhances the efficiency and stability in point-to-point tasks. Sun et al. [24], addressing the multi-objective problem in segmented assembly robots, proposed an infeasible-updating non-dominated sorting-based evolutionary algorithm (INSEA) to optimize the time, energy, and impact functions, enabling the robotic arm to perform assembly operations smoothly and efficiently.

Although the aforementioned studies have achieved significant progress in the field of multi-objective trajectory planning, the performance of these algorithms is often highly dependent on parameter configurations. Proper adjustment of key parameters is critical for optimizing algorithm performance. Compared to other algorithms, multi-objective particle swarm optimization (MOPSO) [25] demonstrates advantages such as low computational cost and fast convergence speed. The effectiveness of MOPSO in solving standard operational procedures has been validated, particularly in large and complex environments. For example, Han et al. [26] proposed a robust multimodal multi-objective particle swarm optimization algorithm (RMMPSO), which demonstrated outstanding performance in addressing data-driven multimodal optimization problems. Wang et al. [27] designed an improved MOPSO with an adaptive angular region partitioning algorithm, achieving significant improvements in convergence and solution diversity for UAV task allocation. Furthermore, MOPSO has shown its superiority in practical applications such as multi-objective load scheduling in microgrids [28], carpooling optimization [29], and job-shop scheduling [30]. It has also been successfully applied to robotic trajectory optimization. Huang et al. [31] first proposed a spatial robotic motion trajectory optimization method based on MOPSO. This method employs motion time, dynamic disturbance, and jerk as multi-objective functions to derive efficient and safe motion trajectories for spatial robots. Lan et al. [32] introduced a trajectory competition MOPSO (TCMOPSO) to solve for the Pareto optimal set of robotic arm trajectories considering time, energy, and impact, effectively reducing motion time, joint impact, and energy consumption.

Due to the complexity of trajectory planning problems for handling robots, there remains room for improvement in the efficiency and accuracy of existing solution methods. Thus, it is necessary to conduct an in-depth analysis and optimization of existing algorithms to achieve higher computational accuracy and faster processing speeds. Furthermore, although numerous trajectory planning methods have been proposed in previous literature, the outcomes typically provide a comprehensive trajectory without considering adaptability to actual operating conditions. To address this gap, it is crucial to study an integrated framework that can flexibly adjust optimization strategies based on different operating conditions and task requirements to ensure that handling robots can perform tasks efficiently under real-world operating conditions.

Building upon existing research, this paper proposes a MOPSO guided by evolutionary information (EIGMOPSO) for the multi-objective trajectory optimization problem of handling robots, and generates an optimal trajectory that meets real-world operating conditions through a comprehensive optimization of time, energy consumption, and joint impact. Compared to traditional methods, this approach establishes a normalized multi-objective function without merging different performance metrics, achieving a globally optimal design for time, energy, and impact while considering the kinematic constraints of handling robots. By avoiding the merging of performance metrics, the planned trajectory is made more practical and effectively avoids getting trapped in a single local optimum. the method enhances the reliability and maintainability of robotic systems without necessitating costly hardware upgrades.

The main contributions of this paper are as follows:

1. A novel multi-objective trajectory optimization method is proposed for handling robots based on an improved MOPSO. By adjusting the weights or priorities of the optimization objectives, the system can flexibly adapt to diverse task requirements, achieving an efficient, energy-saving, and stable motion scheme.

2. To address the issue of MOPSO being prone to local optimum, this paper introduces a regionally dynamic stratification strategy based on evolutionary capability assessment. This strategy dynamically adjusts the population's exploration direction and exploitation depth by evaluating

fitness variation, diversity, and stability.

3. This paper further proposes a layered optimization strategy with region-guided and adaptive exploration, t dividing the population into three distinct regions based on their evolutionary capabilities. Different search mechanisms are designed for each region: the standard MOPSO strategy, a learning automaton strategy based on a hybrid grey wolf optimization algorithm, and a dual chaotic map mutation strategy.

4. A dynamic two-stage performance metric-based archive maintenance strategy is proposed. By incorporating comprehensive density estimation (CD) and convergence assessment function (CA) metrics, this strategy ensures that the solutions retained in the archive possess good distribution uniformity and convergence.

The remainder of this paper is organized as follows: Section 2 establishes the multi-objective optimization model for trajectory planning; Section 3 constructs the jerk-continuous trajectory for the handling robot; Section 4 provides a detailed description of EIGMOPSO and its performance testing; Section 5 validates the algorithm's effectiveness for practical application in optimizing handling robot trajectory; Section 6 concludes the paper.

2. Mathematical model of the multi-objective problem

Efficiency, energy consumption, and stability are key metrics for evaluating the performance of handling robots. To optimize these metrics during trajectory optimization, they are quantified using time, energy, and jerk [33]. Since handling robots are typically employed for repetitive tasks, incorporating these metrics into trajectory optimization can significantly improve their operational efficiency and stability while effectively reducing energy consumption. thereby increasing the reliability and maintainability of handling robots in repetitive industrial tasks. Based on these three evaluation metrics, this paper constructs the corresponding objective functions to achieve comprehensive optimization of the operating trajectory for handling robots. The specifics are as follows:

(1) Time objective function

$$f_1 = \sum_{i=0}^{n-1} \Delta t_i = \sum_{i=0}^{n-1} |t_{i+1} - t_i| \quad (1)$$

where, Δt_irepresents the time interval between two positions on the handling robot's path. The time objective function serves as a metric for evaluating the operational efficiency of the handling robot. To enhance its efficiency, the time objective function must be minimized.

(2) Energy consumption objective function

Energy consumption primarily arises from the electrical energy consumed by the servo motors. The accurate energy consumption function is given by:

$$E = \int_0^T \sum_{i=1}^{N} P_i(t)dt = \sum_{m=1}^{M} \sqrt{\frac{1}{T}\int_0^T (\theta_i \tau_i)^2 dt} \quad (2)$$

where $P_i(t)$ denotes the power at any given moment, and $\tau_i$represents the torque at any given moment.

Since obtaining the servo motor power at each instant is challenging, the energy consumption function can be simplified by using average acceleration as an energy consumption indicator. Considering the linear relationship between robot energy consumption and acceleration, the function can be expressed as:

$$f_2 = \sum_{m=1}^{M} \sum_{j=0}^{n-1} (\int_{t_j}^{t_{j+1}} T_i(\theta) d\theta_i) = \sum_{m=1}^{M} \sqrt{\frac{1}{T}\int_0^T (a_i)^2 dt} \quad (3)$$

where a_i represents the joint angular acceleration at any given moment. It should be noted that the energy consumption model presented provides a quantitative indicator of the energy consumed by all joints of the handling robot, intended for qualitative analysis.

(3) Smoothness objective function

The smoothness of a handling robot is primarily related to the joint jerk. The smoothness indicator can be expressed using the average joint jerk, as follows:

$$f_3 = \sum_{m=1}^{M} \sqrt{\frac{1}{T}\int_0^T (j_i)^2 dt} \quad (4)$$

where j_i denotes the joint jerk at any given moment. Essentially, abrupt changes in jerk equate to the application of excitation to the robot, leading to vibrations. Trajectory smoothness contributes to the stable operation of the joints and effectively reduces wear between them.

To generate trajectories that meet both performance requirements and practical application constraints, it is necessary to optimize multiple objectives and introduce appropriate constraints during the optimization process [34]. Focusing solely on a single optimization objective may lead to some parameters deviating excessively from reasonable ranges. For example, when minimizing motion time is the optimization objective, it often necessitates increasing joint velocities, which can result in infeasible solutions such as unrestricted joint speeds, clearly unreasonable in practical applications. Therefore, it is essential to impose appropriate constraints on multi-objective optimization to ensure that trajectory efficiency is improved without exceeding the system's physical limitations or violating task requirements. The introduction of these constraints effectively ensures the feasibility and safety of trajectory planning under real-world conditions [35].

The constraints are as follows:

$$\begin{cases} |v_m(t)| \le v_{mmax} \\ |a_m(t)| \le a_{mmax} \\ |j_m(t)| \le j_{mmax} \\ |\tau_m(t)| \le \tau_{mmax} \end{cases} \tag{5}$$

where v_m, a_m, j_m, τ_m represent the velocity, acceleration, jerk, and torque of the m -th joint at any given moment, respectively. v_{mmax}, a_{mmax}, j_{mmax}, τ_{mmax} denote the maximum velocity, maximum acceleration, maximum jerk, and maximum torque of the m -th joint, respectively.

Additionally, during operation, the joint angles of the handling robot should remain within the workspace limits. Therefore, the following constraint must be satisfied:

$$\theta_m \in \Omega \tag{6}$$

where $\theta_m$represents the joint position of the m -th joint at any given moment, and Ω denotes the allowable angular workspace for the joint.

3. Construction of jerk-continuous trajectories

A series of data points $P_0, P_1, \ldots, P_n$ is set along the desired trajectory in the joint space of the handling robot, dividing the B-spline trajectory into $n$segments. Due to the local support property of B-spline curves, altering a control point affects only the local trajectory around it, leaving the rest unchanged [36].

The expression for the i -th segment of the B-spline is given by:

$$P(u) = \sum_{i=0}^{n} d_i N_{i,k}(u) \tag{7}$$

where d_i denotes the i-th control vertex; u is the parameter of the curve; $N_{i,k}(u)$ denotes the B-spline basis function.

To generate curves of joint variables as a function of time, the cumulative chord length parameterization approach is used to normalize the time nodes t_i, resulting in the node vector U for the k -th B-spline curve.

$$u_0 = u_1 = \cdots = u_k = 0 \tag{8}$$

$$u_{n+k} = u_{n+k+1} = \cdots = u_{n+2k} = 1 \tag{9}$$

$$u_i = u_{i-1} + \frac{|\Delta t_{i-k-1}|}{\sum_{j=0}^{n-1} |\Delta t_j|}, i = k+1, \cdots, n+k+1 \tag{10}$$

Due to the non-uniform distribution of nodes along the parameter axis, a non-uniform B-spline curve will be generated. Given the node vector $U = [u_0, u_1, \cdots, u_{n+2k}]$, the n -th order normalized B-spline blending functions $N_{i,k}(u)$ are defined by the recursive Cox-de Boor equation as

$$\left. \begin{array}{l} N_{i,0}(u) = \begin{cases} 1, & u_i \le u < u_{i+1} \\ 0, & others \end{cases} \\ N_{i,k}(u) = \frac{u-u_i}{u_{i+k}-u_i} N_{i,k-1}(u) + \frac{u_{i+k+1}-u}{u_{i+k+1}-u_{i+1}} N_{i+1,k-1}(u) \end{array} \right\} \tag{11}$$

Based on the joint variable-time sequences, $(n+1)$ equations can be formulated to satisfy the interpolation conditions as

$$P(u_{i+k}) = \sum_{j=i}^{i+k} d_j N_{j,k}(u_{i+k}) = C_i \tag{12}$$

where $u_{i+k} \in [u_k, u_{n+k}], i = 0,1, \cdots n$.

Based on the constraints among the B-spline curve degree, number of nodes, and number of control points, it is evident that for a seventh-order non-uniform B-spline curve, an additional six equations are required, typically provided by boundary conditions. Since the end nodes have a multiplicity of k, the terminal control vertices coincide with the start and end data points. Therefore

$$\dot{P}_0 = \dot{P}(u_7) = \sum_{j=1}^{7} d_j^1 N_{j,6}(u_7) = v_0 \tag{13}$$

$$\dot{P}_n = \dot{P}(u_{n+7}) = \sum_{j=n+1}^{n+7} d_j^1 N_{j,6}(u_{n+7}) = v_e \tag{14}$$

Similarly,

$$\ddot{P}_0 = \ddot{P}(u_7) = \sum_{j=2}^{7} d_j^2 N_{j,5}(u_7) = a_0 \tag{15}$$

$$\ddot{P}_n = \ddot{P}(u_{n+7}) = \sum_{j=n+2}^{n+7} d_j^2 N_{j,5}(u_{n+7}) = a_e \tag{16}$$

$$\dddot{P}_0 = \dddot{P}(u_7) = \sum_{j=3}^{7} d_j^3 N_{j,4}(u_7) = j_0 \tag{17}$$

$$\dddot{P}_n = \dddot{P}(u_{n+7}) = \sum_{j=n+3}^{n+7} d_j^3 N_{j,4}(u_{n+7}) = j_e \tag{18}$$

where v_0, a_0, j_0 denote the velocity, acceleration, and jerk of the joint at the starting point, respectively, and v_e, a_e, j_e represent the velocity, acceleration, and jerk of the joint at the terminating point, respectively.

By solving the system of Eqs. (12)-(18), the $(n+1)$ control points can be determined. Finally, using the obtained control points $P_i$and the node vector U, the motion trajectory of a joint can be constructed. The motion trajectory for the handling robot's joints can be planned using the aforementioned method [37].

4. Proposed Method

MOPSO simulates the foraging behavior of bird flocks, exploring and optimizing multiple objective functions through the movement of particles [38]. Given that the optimization problem addressed in this paper involves multiple objectives, MOPSO is chosen as the benchmark algorithm due to its high operability in practical applications. However, MOPSO employs traditional particle swarm algorithm mechanisms for velocity and position updates, which can lead to suboptimal search capability. Additionally, the robot is a nonlinear, strongly coupled system, further complicating the optimization problem [39]. To address the issues of slow convergence speed encountered with traditional MOPSO in robot trajectory optimization, this paper proposes a MOPSO guided by evolutionary information (EIGMOPSO). This approach enables the algorithm to rapidly and accurately find the optimal trajectory within a complex solution space, thereby enhancing the performance and efficiency of the handling robots.

4.1. MOPSO

In MOPSO, each particle continuously adjusts its position in the solution space to explore new solutions. Particles update their positions under the guidance of two primary "leaders": the individual best solution $p_{i,d}$, and the global leader $g_{i,d}$. The velocity and position of the i -th particle are updated as

$$v_{i,d}(t+1) = \omega v_{i,d}(t) + c_1 r_1(p_{i,d}(t) - z_{i,d}(t)) + c_2 r_2(g_{i,d}(t) - z_{i,d}(t)) \quad (19)$$

$$z_{i,d}(t+1) = z_{i,d}(t) + v_{i,d}(t+1) \quad (20)$$

where $v_{i,d}(t)$ represents the velocity of the particle in the d -th dimension at the t -th iteration; $z_{i,d}(t)$ denotes the updated position of the particle; r_1 and r_2 are random numbers between 0 and 1; ωdenotes the inertia weight; c_1 and c_2 represent the learning factors.

4.2. The proposed EIGMOPSO

To address the issues of slow convergence speed encountered with traditional MOPSO, this paper proposes a MOPSO guided by evolutionary information (EIGMOPSO). This algorithm effectively enhances search efficiency and the ability to discover global optima by comprehensively considering the evolutionary information of the population after each iteration and partitioning the population into different regions based on this information. By employing different search mechanisms for particles in various regions, providing more flexibility in determining the final optimal solution. Additionally, a two-stage adaptive archive maintenance strategy is introduced to ensure that the solutions retained in the final archive exhibit both good convergence and sufficient diversity, thereby improving the overall performance of the algorithm.

4.2.1. Regionally dynamic stratification strategy

To more comprehensively assess the evolutionary state of particles, this paper proposes an evolutionary capability formula that integrates fitness variation, diversity evaluation, and stability assessment. The specific definition is as follows:

(1) Fitness variation: $E_i^{fitness}(t)$

Fitness variation is used to measure the change in a particle's fitness between two consecutive iterations. It is defined as

$$E_i^{fitness}(t) = \sqrt{\sum_{d=1}^{D}(f_{i,d}^{pbest}(t) - f_{i,d}^{pbest}(t-1))^2} \quad (21)$$

where $f_{i,d}^{pbest}(t)$ represents the fitness value of the individual best solution of particle i in the d-th dimension. This formula reflects the evolutionary magnitude of the particle during the optimization process and effectively captures the trend of changes in the particle's fitness.

(2) Diversity evaluation: $E_i^{diversity}(t)$

Diversity evaluation is used to measure the relative position of a particle within the population. The specific formula is as

$$E_i^{diversity}(t) = \frac{1}{N-1}\sum_{j=1,j\neq i}^{N} \parallel \boldsymbol{x}_i(t) - \boldsymbol{x}_j(t) \parallel \quad (22)$$

where $\boldsymbol{x}_i(t)$ and $\boldsymbol{x}_j(t)$ represent the position vectors of particle $i$and particle j at the t-th iteration, respectively. The diversity is evaluated by calculating the average distance between a particle and other particles.

(3) Stability assessment: $E_i^{stability}(t)$

Stability assessment is used to measure the stability of a particle's position changes over a period of time. It is defined as

$$E_i^{stability}(t) = exp\left(-\frac{\Vert x_i(t) - x_i(t-1)\Vert}{\sigma}\right) \quad (23)$$

where $\boldsymbol{x}_i(t)$ and $\boldsymbol{x}_i(t-1)$ represent position changes of particle i between two consecutive iterations. σ is a tuning parameter used to control the sensitivity of the changes. This formula reflects the stability of a particle by calculating an exponential function of its position changes.

Based on the three criteria mentioned above, the comprehensive evolutionary capability is defined as follows:

$$E_i(t) = \alpha \cdot E_i^{fitness}(t) + \beta \cdot E_i^{diversity}(t) + \gamma \cdot E_i^{stability}(t) \quad (24)$$

where α, β, γ are adaptive weight coefficients that can be adjusted dynamically to meet the requirements of the optimization process, and can be defined as

$$\alpha(t) = \frac{D_{fitness}(t)}{D_{total}(t)} \quad (25)$$

$$\beta(t) = \frac{D_{diversity}(t)}{D_{total}(t)} \quad (26)$$

$$\gamma(t) = \frac{D_{stability}(t)}{D_{total}(t)} \quad (27)$$

where $D_{\text{total}}(t) = D_{\text{fitness}}(t) + D_{\text{diversity}}(t) + D_{\text{stability}}(t)$.

Based on the comprehensive evolutionary capabilities of each particle during the iteration process, the ranking for each particle is determined according to its comprehensive evolutionary capability:

$$ER = Rank(E(t)) \quad (28)$$

Based on the descending order ranking ER of each particle within the population, the particles are assigned to specific regions as follows:

$$Region(x_i) = \begin{cases} \text{I}, ER(\text{x}_i) < 0.2 \\ \text{II}, 0.2 \le ER(\text{x}_i) < 0.8 \\ \text{III}, ER(\text{x}_i) \ge 0.8 \end{cases} \quad (29)$$

where Region I represents the high evolutionary capability region, comprising the top 20% of particles based on their comprehensive evolutionary capability ranking. Region II represents the medium evolutionary capability region, consisting of particles whose comprehensive evolutionary capability ranks between 20% and 80%. Region III represents the low evolutionary capability region, containing the bottom 20% of particles based on their comprehensive evolutionary capability ranking.

Algorithm 1: Regionally dynamic stratification strategy

```
Input: The population P, the population size N, $\omega$, $c_1$, $c_2$.
Output: The sub-populations: Region I, Region II, Region III.
for $i = 1$ to N do
    Calculate $E_i^{fitness}(t)$ by Eq. (21);
    Calculate $E_i^{diversity}(t)$ by Eq. (22);
    Calculate $E_i^{stability}(t)$ by Eq. (23);
    Compute the Comprehensive Evolutionary Performance $E_i(t)$ by Eq. (24);
end for
Sort P by $E_i(t)$ in descending order;
Region I = Top 20% of P;
Region II = Next 60% of P;
Region III = Bottom 20% of P;
Return sub-populations.
```

4.2.2. Layered optimization strategy

This paper develops differentiated search strategies designed for particles in different regions. This method maintains high search efficiency and solution diversity at all evolutionary stages. The specific strategies are as follows:

1) Particles in Region I exhibit significant fitness variation, high diversity, and stable positions, already demonstrating good optimization performance. To preserve these favorable characteristics, the standard MOPSO velocity and position update formulas are applied. This allows particles in Region I to effectively utilize information from their personal best and global best solutions, maintaining a high convergence rate and solution quality, thereby ensuring the stability and reliability of the optimization process.

2) Particles in Region II have a certain level of optimization capability but still require further exploration. To improve the search direction and depth of particles in this region and enhance the overall optimization efficiency, a strategy inspired by the alpha-wolf strategy of the grey wolf optimization is employed. Three optimal particles are selected as leaders to guide the search. Additionally, a Learning Automata strategy is introduced, allowing other particles to perform adaptive learning and optimization. By assigning different learning probabilities to each particle and dynamically adjusting these probabilities based on their performance in each iteration, the guiding role of the optimal particles is further strengthened. This mechanism provides diverse search directions, enhancing the ability to discover global optimal solutions.

The evolution formula for the particle position is:

$$\vec{D} = \left|\vec{C} \cdot \vec{P_o}(t) - \vec{P}(t)\right| \quad (30)$$

$$\vec{P}(t+1) = \vec{P_o}(t) - \vec{A} \cdot \vec{D} \quad (31)$$

where $\vec{D}$ is the distance between the particle and the target; $\vec{P}(t)$ and $\vec{P_o}(t)$ represent the position vectors of the particle and the target at the t-th iteration, respectively; $\vec{A}$ and $\vec{C}$ are coefficient vectors that vary with the number of iterations, defined as

$$\vec{A} = 2a \cdot \vec{r_1} - a \tag{32}$$

$$\vec{C} = 2 \cdot \vec{r_2} \tag{33}$$

$$a = 2 - \frac{2t}{T_{max}} \tag{34}$$

where $\vec{r_1}$ and $\vec{r_2}$ are random vectors ranging from 0 to 1, T_{max} denotes the maximum number of iterations, and a is a convergence factor.

When the optimal solution is found, the three optimal particles lead the other particles in the search. The positions of the three leaders are updated as

$$\vec{D_\alpha} = \left|\vec{C_1} \cdot \vec{P_\alpha}(t) - \vec{P}(t)\right| \tag{35}$$

$$\vec{D_\beta} = \left|\vec{C_2} \cdot \vec{P_\beta}(t) - \vec{P}(t)\right| \tag{36}$$

$$\vec{D_\delta} = \left|\vec{C_3} \cdot \vec{P_\delta}(t) - \vec{P}(t)\right| \tag{37}$$

The positions of the other particles are given by

$$\begin{cases} \vec{P_1}(t+1) = \vec{P_\alpha}(t) - \vec{A_1} \cdot \vec{D_\alpha} \\ \vec{P_2}(t+1) = \vec{P_\beta}(t) - \vec{A_2} \cdot \vec{D_\beta} \\ \vec{P_3}(t+1) = \vec{P_\delta}(t) - \vec{A_3} \cdot \vec{D_\delta} \end{cases} \tag{38}$$

$$\vec{P}(t+1) = \frac{\vec{P_1}(t+1) + \vec{P_2}(t+1) + \vec{P_3}(t+1)}{3} \tag{39}$$

The obtained optimal position values are incorporated into the updates of particle velocity and position. Adaptive optimization is then performed through learning probabilities, resulting in the following equations:

$$v_{i,d}(t+1) = \omega_{i,d}(t) \cdot v_{i,j}(t) + c_1 r_1 (g_{i,d}(t) - z_{i,d}(t)) + P_{i\alpha} c_2 r_2 \cdot [p_1(t) - z_{i,d}(t)] + P_{i\beta} c_3 r_3 \cdot [p_2(t) - z_{i,d}(t)] + P_{i\delta} c_4 r_4 \cdot [p_3(t) - z_{i,d}(t)] \tag{40}$$

$$z_{i,d}(t+1) = z_{i,d}(t) + v_{i,d}(t+1) \tag{41}$$

where $P_{i\alpha}, P_{i\beta}, P_{i\delta}$ represent the probabilities of particle i learning from the three optimal particles. The initial values are set to $1/3$, with a range of [0,1] and are defined as

$$P_{i\alpha} = P_{i\alpha} + \Delta P \cdot (1 - P_{i\alpha}) \tag{42}$$

$$P_{i\beta} = P_{i\beta} + \Delta P \cdot (1 - P_{i\beta}) \tag{43}$$

$$P_{i\delta} = P_{i\delta} + \Delta P \cdot (1 - P_{i\delta}) \tag{44}$$

where ΔP represents the increment of the learning probability, which is set to 0.1 in this paper.

3) Particles in Region III have weaker optimization capabilities, so enhancing particle diversity should be the highest priority. Unlike the mutation strategies used in most existing MOPSO, this paper proposes a method based on a dual chaotic mapping approach to improve the optimization performance. This method utilizes the high randomness and strong chaotic properties of Sine and Logistic chaotic maps. By alternating these two chaotic maps during the iteration process, more diverse mutation sequences are generated. This approach not only increases the complexity of mutations but also effectively avoids the periodic behavior that could result from using a single chaotic map, thereby further enhancing population diversity. Additionally, this paper randomly selects particles from the swarm that are non-dominated by, or even dominate, the current particle to act as perturbation particles. This can prevent a slowdown in the optimization process or even degeneration when particles escape from local optima. The introduction of perturbation factors injects new, positive elements into the variation of the particles' flight trajectories.

The definition of chaotic mapping is

$$x_{\text{chaotic},t} = \begin{cases} 4 \cdot sin(\pi \cdot x_{\text{chaotic},t}), & \text{if } t \ mod\ 2 \neq 0 \\ 3.9 x_{\text{chaotic},t}(1 - x_{\text{chaotic},t}), & \text{if } t \ mod\ 2 = 0 \end{cases} \tag{45}$$

The leader in Region III is defined as the local best (Lbest). In each iteration, the leader's current position is re-positioned using chaotic mapping:

$$x_{Lbest}(t+1) = x_{Lbest}(t) + x_{Lbest}(t) \times x_{\text{chaotic},t} \tag{46}$$

where x_{Lbest} represents the current position of the leader in Region III.

The velocity and position updates for other particles are as

$$v_{i,d}(t+1) = \omega v_{i,d}(t) + c_1 r_1 (p_{i,d}(t) - z_{i,d}(t)) + c_2 r_2 (x_{Lbest}(t) - z_{i,d}(t)) + c_3 r_3 (p_{j,d}(t) - z_{i,d}(t)) \tag{47}$$

$$z_{i,d}(t+1) = z_{i,d}(t) + v_{i,d}(t+1) \tag{48}$$

where $p_{j,d}(t)$ represents the position of the disturbance particle j in the d-th dimension, randomly selected from the t-th generation of the particle swarm. It is required that particle j either dominates the current particle i or is non-dominated with respect to particle i.

Algorithm 2: Layered optimization strategy

```
Input: Region I, Region II, Region III.
Output: The updated particles.
for $i$ in Region I do
Update the particles by Eq. (19) and Eq. (20);
end for
for $i$ in Region II do
Select the three best particles $\alpha$, $\beta$, $\delta$ as leaders;
```

```
    Calculate distance vectors $\overrightarrow{D_\alpha}, \overrightarrow{D_\beta}, \overrightarrow{D_\delta}$;
    Update the particles by Eq. (40) and Eq. (41);
end for
for $i$ in Region III do
    Apply chaotic mapping for mutation by Eq. (45);
    Update the Lbest particle by Eq. (46);
    Introduce disturbance from randomly selected particle $j$;
    Update the particles by Eq. (47) and Eq. (48);
end for
Return particles.
```

4.2.3. Dynamic two-stage performance metric-based archive maintenance strategy

In MOPSO, maintaining the external archive is crucial. When the external archive reaches its maximum capacity, it needs to be maintained [40]. The objective is to select solutions with good diversity and convergence to retain while removing the inferior non-dominated solutions. To this end, this paper proposes a dynamic two-stage performance metric-based archive maintenance strategy. In the first stage, comprehensive density estimation (CD) is used to evaluate the position of the solutions and their density distribution. In the second stage, a convergence assessment (CA) function is employed to measure the proximity of solutions to the optimal front. This two-stage maintenance strategy ensures that the solutions retained in the final archive exhibit both good convergence and sufficient diversity, thereby enhancing the algorithm's overall performance. The pseudocode for this strategy is presented in Algorithm 3.

Algorithm 3: Archive maintenance

```
Input: The updated population N, the archive A.
Output: The updated archive A.
while size(A)>N do
    Stage I: Calculate the CD of each nondominated solution by Eq. (49);
        Sort the nondominated solution;
        Remove the particle with high density;
    Stage II: Calculate the CA of each nondominated solution by Eq. (50);
        Sort the nondominated solutions;
        Remove the particle with poorly convergent solutions;
end while
Return A.
```

In the first stage, the CD is used to evaluate the distribution of individuals. First, the nearest solution point to particle i is identified as particle m, and the Euclidean distance $dis(x_i, x_m)$ between them is calculated. Then, the nearest solution point to particle m is identified as particle n, and the Euclidean distance $dis(x_m, x_n)$ between them is calculated, ensuring that $x_i \neq x_m \neq x_n$. Finally, the CD is defined as

$$CD = dis(x_i, x_m) + dis(x_m, x_n) - |dis(x_i, x_m) - dis(x_m, x_n)| \tag{49}$$

The smaller the distance value obtained using the evaluation method in Eq. (49), the more densely distributed the individuals are, indicating poorer diversity. Therefore, solutions with smaller CD values need to be removed to maintain the uniformity of the solution distribution in the archive.

In the second stage, the CA is used to comprehensively evaluate the convergence degree of individuals within the solution space. A smaller CA value indicates that a solution is closer to the Pareto front, thus exhibiting better convergence. Therefore, in this stage, solutions with larger CA values are removed to ensure the population convergence. The definition of CA is as follows:

$$CA = \sum_{j=1}^{M} \left(\frac{f_j(x_i) - f_j^*}{f_j^{max} - f_j^{min}} \right)^2 \tag{50}$$

where $f_j(x_i)$ represents the fitness value of particle i on the j -th objective, f_j^* is the optimal value for the j -th objective, f_j^{min} and f_j^{max} are the minimum and maximum fitness values or the j -th objective in the current population, respectively.

5. Numerical simulation

To demonstrate the feasibility and practicality of the trajectory planning method and to obtain the optimal motion trajectory, the proposed EIGMOPSO is compared with several state-of-the-art algorithms. The EIGMOPSO algorithm is implemented in MATLAB to solve the multi-objective trajectory optimization model.

5.1. Performance metrics

To evaluate the convergence and uniformity of the approximate Pareto front obtained by the algorithm, the Inverted

Generational Distance (IGD) [41] and Spacing (SP) [42] metrics are employed.

(1) IGD is a metric that measures the distance between the approximate Pareto front generated by the algorithm and the true Pareto front. IGD is calculated by

$$IGD(P^*, Q) = \frac{\sum_{v \in P^*} d(v,Q)}{|P^*|} \tag{51}$$

where P^* is the set of points uniformly distributed on the true Pareto front, $|P^*|$ is the length of set P^*, Q is the approximate Pareto front, and $d(v,Q)$ is the minimum Euclidean distance of an individual $v$in P^* to the set Q.

(2) SP measures the standard deviation of the minimum distance of each solution to the others, serving as an important indicator of the variability among neighboring solutions within a given range. SP is calculated by

$$SP = \sqrt{\frac{1}{|P|}\sum_{i=1}^{|P|}(\bar{d} - d_i)^2} \tag{52}$$

where d_i represents the minimum Euclidean distance between an individual i on the approximate front and its neighboring individuals, and $\bar{d}$ is the average of these distances.

5.2. Test problems and comparative algorithms

To evaluate the effectiveness of the proposed EIGMOPSO, performance tests are conducted using the bi-objective ZDT1–4 and ZDT6 series of multi-objective test functions [43], as well as the tri-objective DTLZ1–7 [44]. The decision vectors for both ZDT and DTLZ are 30-dimensional. Five representative multi-objective optimization algorithms are selected as comparison algorithms: dMOPSO (decomposition-based MOPSO) [45], CMOPSO (Competitive Mechanism-Based MOPSO) [46], MODE-RMO (multi-objective differential evolution with ranking-based mutation operator) [47], SPEA2 [48], and NSGA-II [49]. The parameter settings for all comparison algorithms are kept consistent with those in the original references. The population size is set to 100 for all algorithms, and the global external archive has a maximum capacity of 100. The maximum number of evaluations for all test functions is set to 50,000, and the maximum iterations is set to 500 for all algorithms.

To minimize the influence of randomness in the performance analysis, each algorithm is independently run 30 times on all test functions. The experimental environment consists of an AMD Ryzen 7 5800X 8-core processor with 16GB of memory, running on the Windows 11 operating system, with MATLAB 2020a as the software environment.

This study evaluates six algorithms on 12 test problems, recording the mean (Mean) and standard deviation (Standard Deviation, Std) of their IGD and SP metrics. Tables 1 and 2 provide detailed statistical results of these metrics, offering a quantitative basis for comparison. In the tables, the symbols "+", "=", and "-" indicate that EIGMOPSO is significantly better than, equal to, or worse than the corresponding algorithm in the respective column based on a two-tailed t-test at a 5% significance level for the test problem in the corresponding row. Additionally, the minimum values for each test problem among all algorithms are highlighted in bold.

Table 1. Performance IGD comparison of different algorithms on 12 test problem.

Problem	IGD	EIGMOPSO	dMOPSO	CMOPSO	MODE-RMO	SPEA2	NSGA-II
ZDT1	Mean	**1.45E-3**	2.72E-3	3.05E-3	3.81E-3	4.52E-3	2.88E-3
	Std	**3.45E-4**	5.47E-3	1.08E-3	7.02E-4	1.85E-3	4.12E-3
ZDT2	Mean	**1.25E-3**	2.55E-3	2.93E-3	3.54E-1	4.18E-2	4.62E-2
	Std	**3.20E-5**	4.75E-4	5.52E-5	6.45E-1	7.04E-2	2.55E-3
ZDT3	Mean	2.83E-3	**1.78E-3**	2.95E-3	3.25E-2	4.68E-3	4.05E-2
	Std	3.85E-4	**3.65E-4**	4.24E-4	5.95E-3	7.32E-4	3.75E-4
ZDT4	Mean	**2.12E-3**	3.14E-2	3.52E-2	4.25E-3	4.85E+0	3.22E-1
	Std	**4.25E-4**	5.72E-2	2.60E-3	7.12E-4	6.75E-2	2.08E-1
ZDT6	Mean	**1.73E-3**	2.84E-3	3.34E-2	2.74E-1	1.87E-3	3.72E+0
	Std	**3.65E-5**	6.05E-4	5.85E-3	3.65E-2	8.12E-4	4.62E-1

Problem	IGD	EIGMOPSO	dMOPSO	CMOPSO	MODE-RMO	SPEA2	NSGA-II
DTLZ1	Mean	**6.52E-3**	8.82E-1	9.53E-3	1.09E-2	1.16E-1	1.24E-2
	Std	**1.05E-3**	1.23E-1	1.32E-3	1.42E-3	1.54E-2	1.62E-2
DTLZ2	Mean	4.25E-3	5.11E-2	**1.23E-3**	6.25E-3	1.95E-1	7.39E-2
	Std	6.06E-5	7.92E-3	**4.12E-5**	8.03E-4	5.17E-2	9.04E-3
DTLZ3	Mean	**2.05E-3**	8.85E-3	9.55E-3	3.15E-3	7.26E-2	2.32E-1
	Std	**9.24E-5**	1.14E-3	1.22E-4	6.79E-4	1.05E-2	5.67E-2
DTLZ4	Mean	**3.85E-3**	4.90E-3	5.25E-3	5.85E-3	6.45E-2	7.92E-3
	Std	**5.62E-4**	6.51E-4	7.06E-3	7.57E-3	3.02E-3	2.45E-3
DTLZ5	Mean	**3.48E-3**	4.65E-3	5.07E-3	5.65E-3	6.25E-1	4.32E-2
	Std	**5.05E-4**	6.23E-4	5.75E-4	7.29E-4	4.76E-3	5.66E-3
DTLZ6	Mean	4.12E-3	5.24E-3	**2.25E-3**	6.45E-2	7.06E-1	7.67E-1
	Std	6.49E-5	7.38E-4	**5.11E-5**	8.39E-3	4.85E-2	9.31E-2
DTLZ7	Mean	**4.52E-3**	3.15E-2	5.05E-3	5.62E-3	6.21E-2	2.81E-2
	Std	**6.23E-4**	5.12E-3	7.12E-4	7.56E-4	3.09E-2	4.15E-3
+/=/-		——	10/1/1	9/1/2	11/1/0	11/1/0	10/2/0

As seen in Table 1, EIGMOPSO achieves the optimal IGD value in 10 out of the 12 test problems, while CMOPSO achieves the optimal IGD value in 2 test problems, and dMOPSO achieves the optimal IGD value in 1 test problem. Although EIGMOPSO does not achieve the optimal IGD value on the ZDT3, DTLZ2, and DTLZ6 problems, the IGD value obtained by EIGMOPSO on these test functions are in the same order of magnitude as those obtained by the algorithms that achieve the optimal IGD value on the same test problems. This indicates that on these three test problems, the results obtained by EIGMOPSO are close to the optimal values. The IGD means and variances presented in Table 1 demonstrate that the EIGMOPSO exhibits the best IGD performance across the 12 test problems.

Additionally, from the t-test results in Table 1, it is evident that the net win scores of EIGMOPSO compared to the other five algorithms are all positive. Specifically, the net win score of EIGMOPSO compared to MODE-RMO and SPEA2 is 11, compared to NSGA-II is 10, and compared to dMOPSO is 9, while the net win score compared to CMOPSO is 7. The t-test results indicate that IGD performance metrics achieved by the EIGMOPSO algorithm are significantly better than those of the other five comparison algorithms across all test problems.

Table 2. Performance SP comparison of different algorithms on 12 test problems.

Problem	SP	EIGMOPSO	dMOPSO	CMOPSO	MODE-RMO	SPEA2	NSGA-II
ZDT1	Mean	3.36E-3	3.28E-2	**2.75E-3**	4.32E-3	4.93E-2	1.21E-2
	Std	3.58E-4	4.25E-3	**3.16E-4**	5.35E-4	5.87E-3	5.32E-3
ZDT2	Mean	3.14E-3	**3.12E-3**	3.42E-3	4.12E-3	4.62E-3	2.31E-2
	Std	5.49E-5	**2.27E-5**	4.55E-4	5.17E-4	5.75E-3	4.65E-3
ZDT3	Mean	**2.85E-3**	6.52E-3	4.18E-2	4.78E-2	5.25E-1	3.12E-2
	Std	**3.68E-4**	4.22E-4	5.84E-3	2.30E-2	6.22E-2	7.29E-4
ZDT4	Mean	**3.72E-3**	4.08E-3	3.98E-3	5.12E-3	3.65E+0	2.35E-1

	Std	**3.94E-4**	2.85E-3	5.35E-4	7.94E-4	6.48E-2	5.61E-2
ZDT6	Mean	**2.75E-3**	3.75E-3	1.45E-2	4.65E-3	2.92E-3	1.35E+0
	Std	**3.07E-5**	4.85E-4	5.23E-4	1.88E-3	5.78E-4	4.84E-1
DTLZ1	Mean	**6.85E-2**	2.34E-1	7.05E-2	2.65E-1	8.16E-2	2.75E-1
	Std	**4.10E-4**	5.48E-4	1.12E-3	1.72E-2	1.22E-2	6.54E-3
DTLZ2	Mean	4.73E-2	**3.22E-2**	1.85E-1	6.14E-2	5.95E-1	9.83E-2
	Std	5.69E-3	**4.67E-3**	7.93E-2	8.49E-3	3.15E-2	2.02E-2
DTLZ3	Mean	**4.83E-2**	6.06E-2	8.38E-2	4.14E-1	6.78E-2	7.66E-1
	Std	**7.62E-3**	3.29E-2	4.56E-2	9.62E-3	4.36E-2	8.41E-2
DTLZ4	Mean	**3.17E-2**	5.30E-1	4.38E-2	1.07E-1	4.79E-1	5.97E-1
	Std	**6.26E-4**	2.74E-2	5.51E-3	3.35E-2	8.09E-3	6.79E-2
DTLZ5	Mean	**2.29E-2**	7.52E-2	8.49E-2	4.63E-2	3.38E-2	1.92E-1
	Std	**4.62E-4**	6.83E-4	5.85E-3	1.06E-2	5.52E-3	6.42E-3
DTLZ6	Mean	3.84E-2	8.93E-2	4.38E-2	1.69E-1	4.21E-1	**4.03E-3**
	Std	2.06E-3	2.91E-2	6.54E-3	4.30E-3	8.83E-2	**1.35E-3**
DTLZ7	Mean	**2.27E-2**	3.63E-2	6.98E-2	4.52E-2	7.75E-2	6.81E-2
	Std	**5.74E-4**	7.10E-3	8.80E-2	7.32E-3	5.08E-2	3.17E-2
+/=/-		——	10/0/2	10/1/1	12/0/0	10/2/0	11/0/1

From Table 2, it is observed that the proposed EIGMOPSO achieves the optimal SP value in 8 out of the 12 test problems. In the ZDT1, ZDT2, DTLZ2, and DTLZ6 test problems, the SP metric of the proposed algorithm is slightly inferior to the algorithm that obtained the optimal value but is superior to the others. For the remaining test problems, the proposed algorithm achieves the optimal SP value. Further analysis of the t-test results in Table 2 reveals that the net win scores of EIGMOPSO compared to the other five algorithms are all positive. Specifically, the net win score is 12 compared to MODE-RMO, 10 compared to both SPEA2 and NSGA-II, 9 compared to CMOPSO, and 8 compared to dMOPSO. This indicates that the EIGMOPSO significantly outperforms the other five comparison algorithms in SP performance across all test problems.

5.3. Multi-objective trajectory optimization experiment verification

To verify the effectiveness of the proposed multi-objective trajectory optimization method, the joint space trajectory is planned for an example involving 8 path points, based on the teaching path points of a handling robot in its actual workspace. The joint position sequences for each path point are obtained through inverse kinematics, as shown in Table 3. To ensure the accuracy of trajectory planning, the constraints on the velocity, acceleration, and jerk of each joint of the handling robot are set as $v_c = 30/s$、$a_c = 20/s^2$、$j_c = 40/s^3$, respectively, and the constraint amplification factors are all set to 1.8 based on the results of multiple experiments to ensure the feasibility of the trajectory optimization.

Table 3. Joint position sequences for each path point.

Path point	Joint1/(°)	Joint 2/ (°)	Joint 3/ (°)	Joint 4/ (°)	Joint 5/ (°)	Joint 6/ (°)
P_0	43.35	7.37	130.55	0	39.05	-46.65
P_1	43.33	-18.00	152.09	0	45.89	48.02
P_2	50.04	-41.85	170.66	0	51.20	-32.35
P_3	62.62	-53.76	179.41	0	54.01	-5.01
P_4	78.01	-57.32	182.70	0	54.62	33.02
P_5	94.40	-52.74	178.45	0	53.38	75.93
P_6	104.13	-42.42	173.54	0	50.32	74.76
P_7	111.91	6.79	132.77	0	40.42	112.16

Based on the given joint path, the time intervals of the path points are used as decision variables for the optimization problem. The number of iterations is set to 100, and the population size is set to 200. For the multi-objective optimization problem of time-energy-impact, EIGMOPSO is employed for optimization.

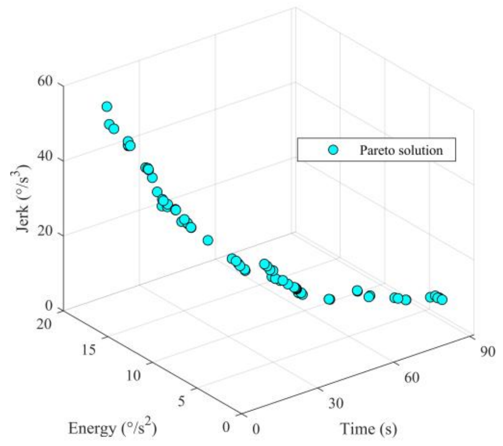


Fig. 1. Optimal Pareto front Optimal Pareto front.

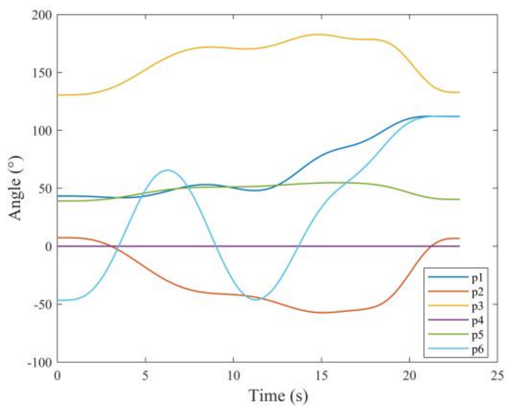
The generated Pareto front, shown in Fig. 1, presents a three-dimensional plot where the axes represent time, energy, and impact, respectively, clearly illustrating the trade-offs among different optimization objectives. The results indicate a significant positive correlation between the energy consumption index and the impact index, while a significant negative correlation exists between these indices and the time performance metric.

The Pareto front is a set of non-dominated solutions. To select different optimal solutions from this set according to different working conditions, the dimensional expressions of each objective are converted into dimensionless expressions. The normalized weight function is defined as

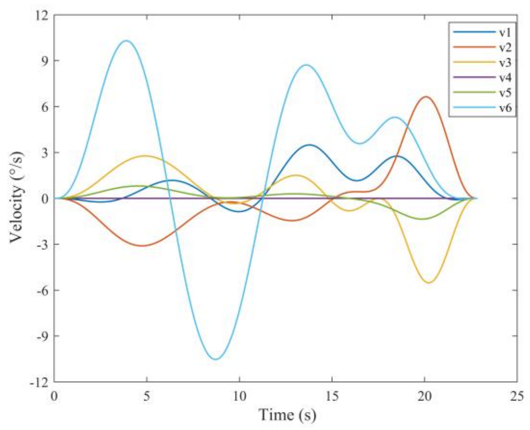
$$f = \frac{\beta_1 f_1}{N_1} + \frac{\beta_2 f_2}{N_2} + \frac{\beta_3 f_3}{N_3} \tag{53}$$

where $\beta_1, \beta_2, \beta_3$are the objective coefficients, and satisfy $\beta_1 + \beta_2 + \beta_3 = 1$; N_1, N_2, N_3 are used to normalize the objective functions to the same range.

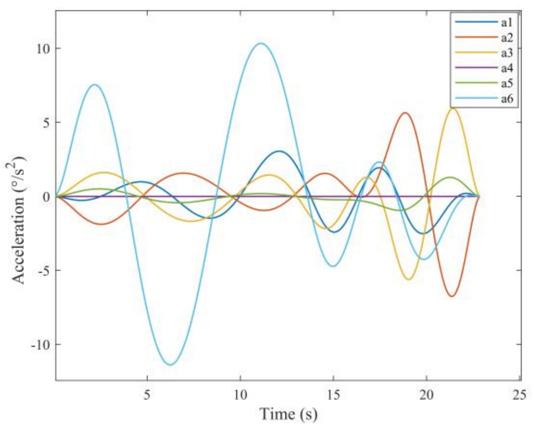
By adjusting the values of parameters $\beta_1, \beta_2, \beta_3$, diverse optimization objectives can be achieved. Specifically, setting $\beta_1 = 1, \beta_2 = 0, \beta_3 = 0$ enables time optimization; setting $\beta_1 = 0, \beta_2 = 1, \beta_3 = 0$ achieves energy optimization; and setting $\beta_1 = 0, \beta_2 = 0, \beta_3 = 1$ leads to impact optimization. Within this framework, a larger value of $\beta_x (x = 1,2,3)$ indicates a higher degree of optimization in the corresponding dimension. From a practical application perspective, the key is to identify and adopt the model that best suits the current requirements. In the absence of specific requirements, a comprehensive optimal setting of $\beta_1 = 1/3, \beta_2 = 1/3, \beta_3 = 1/3$ is used to simplify the parameter adjustment process and achieve balanced optimization.



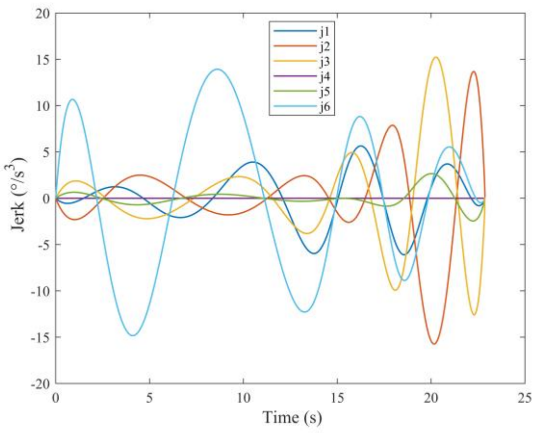
(a) angle



(b) velocity

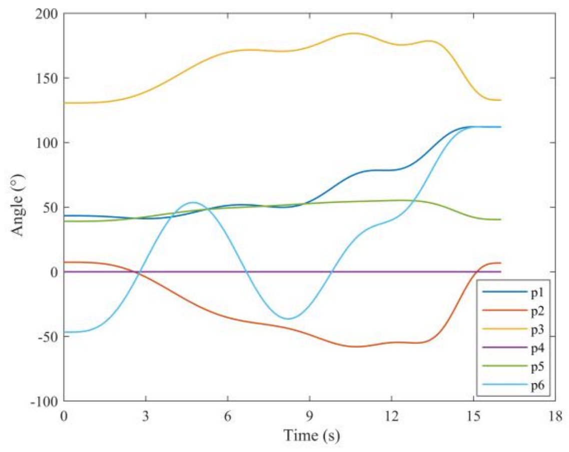


(c) acceleration

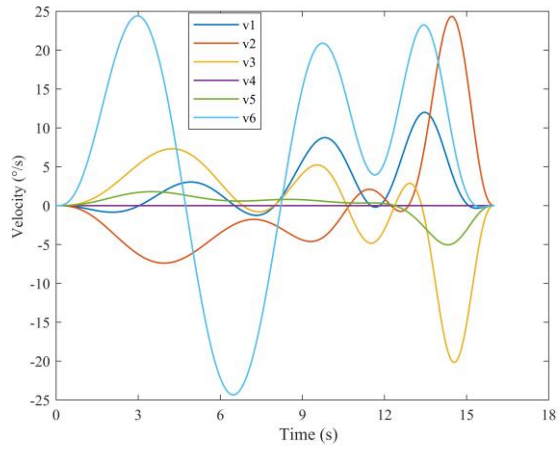


(d) jerk

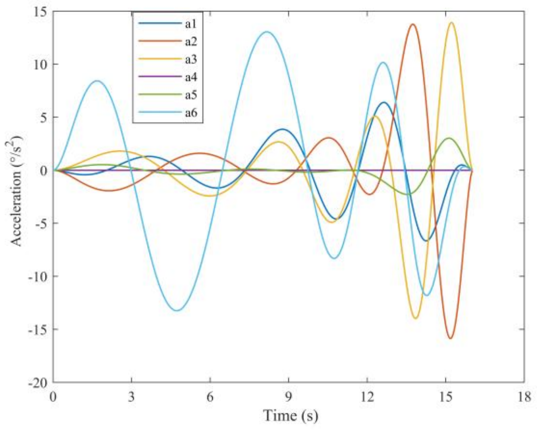
Fig. 2. Comprehensive optimization of each joint using the proposed multi-objective algorithm.



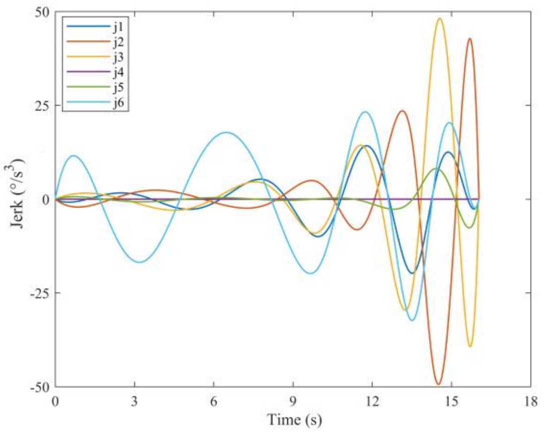
(a) angle



(b) velocity

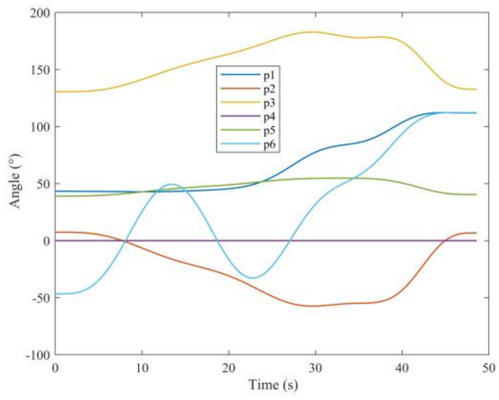


(c) acceleration

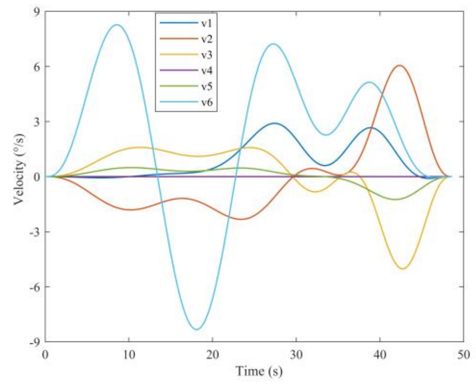


(d) jerk

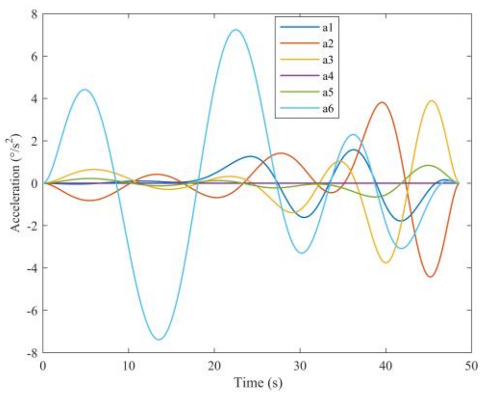
Fig. 3. Time optimization of each joint using the proposed multi-objective algorithm.



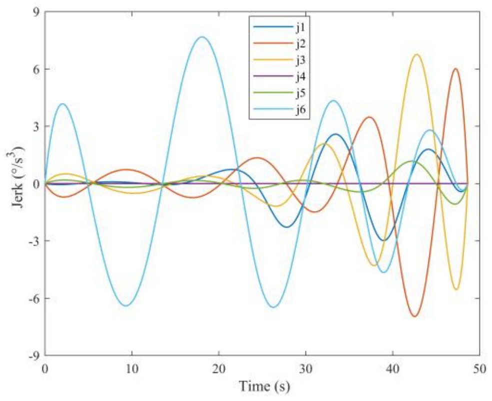
(a) angle



(b) velocity

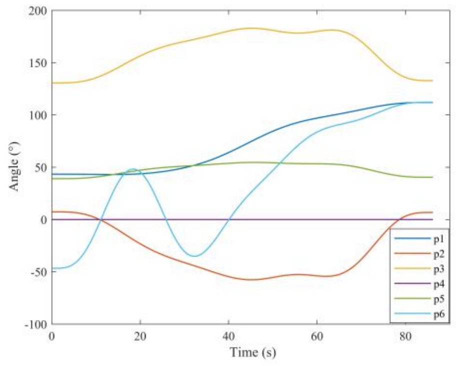


(c) acceleration

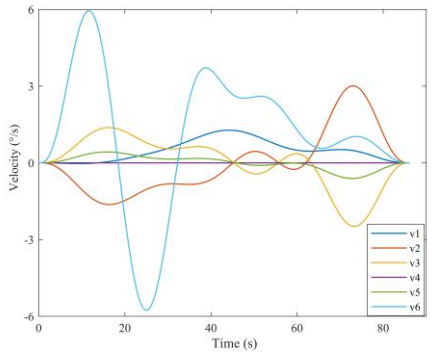


(d) jerk

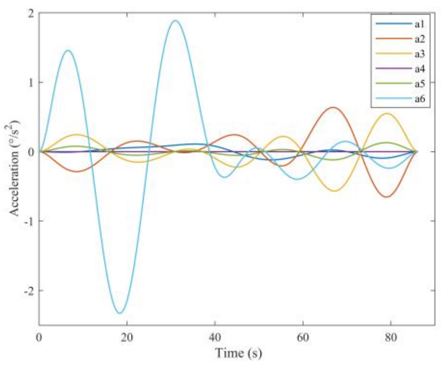
Fig. 4. Energy optimization of each joint using the proposed multi-objective algorithm.



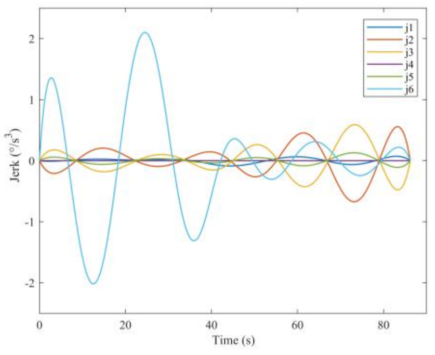
(a) angle



(b) velocity

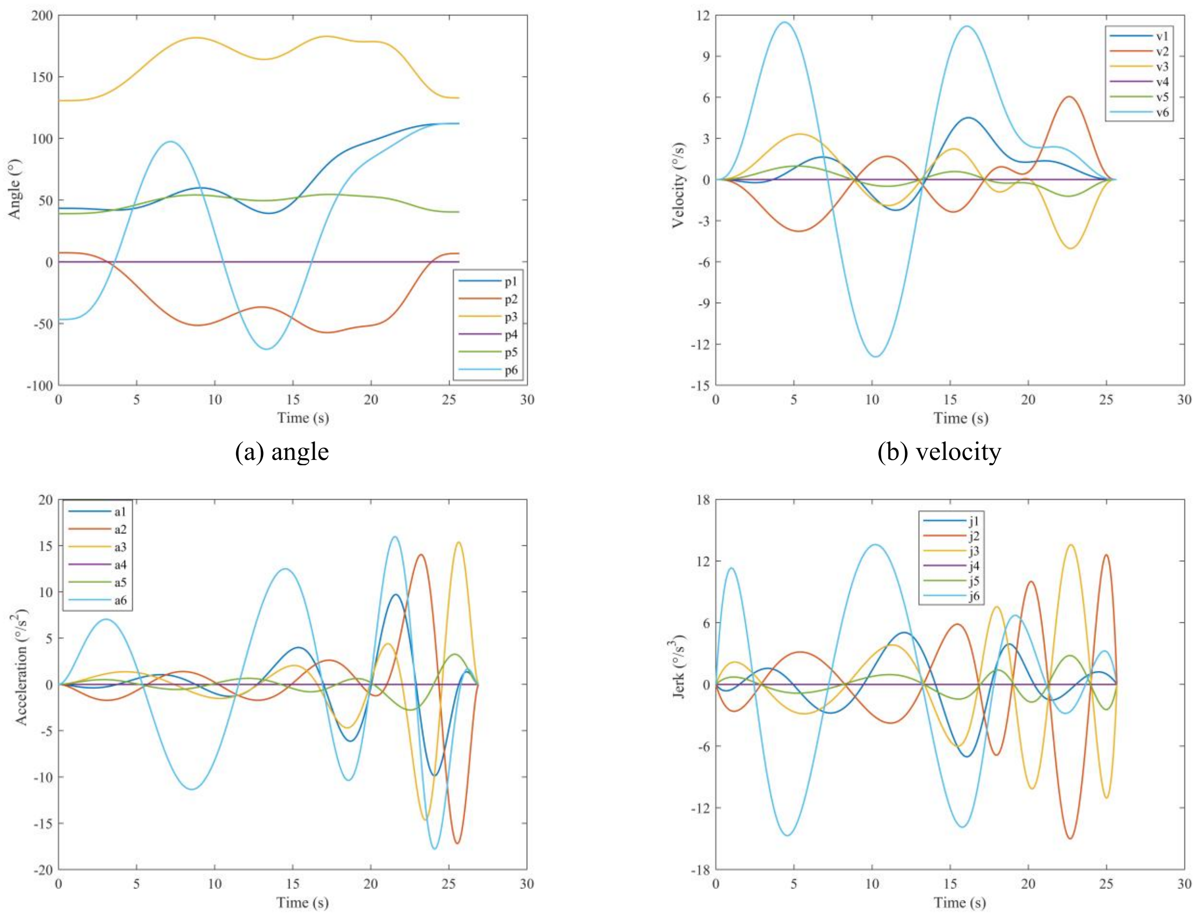


(c) acceleration



(d) jerk

Fig. 5. Impact optimization of each joint using the proposed multi-objective algorithm.



(a) angle (b) velocity

(c) acceleration (d) jerk

Fig. 6. Comprehensive optimization of each joint using the MODE-RMO.

Fig. 2 shows the iterative curve of the proposed EIGMOPSO under comprehensive optimal conditions, with a run time of 22.85 s and an impact range of [−15.74,15.24], demonstrating a balance among time, energy, and impact. Under this optimization strategy, the changes in angle, velocity, acceleration, and jerk of each joint of handling robots are relatively balanced without significant abrupt changes. The comprehensively optimized trajectory design is more robust and suitable for handling tasks that require trade-offs between different performance metrics, allowing it to better adapt to various constraints in complex environments during actual operation.

Fig. 3-5 present the results of joint angle, velocity, acceleration, and jerk optimization under time, energy, and impact optimization using the proposed EIGMOPSO. Under the time-optimal trajectory condition, the curves exhibit a distinct time-priority characteristic, with a runtime of only 15.96 s, significantly improving operational efficiency. However, this also leads to increased peak values of acceleration and jerk, with an impact range of [−49.35,48.26], which may reduce the machine's lifespan and increase mechanical wear and energy consumption. This optimization strategy is suitable for time-sensitive handling tasks but may compromise energy efficiency and mechanical durability.

The energy optimization results show that the changes in angle, velocity, and acceleration of each joint are slower and more stable, indicating minimized energy consumption. However, this optimization strategy typically increases the task completion time, extending the runtime to 48.58 s, which may not meet the requirements of time-sensitive tasks. Therefore, energy optimization is suitable for handling tasks in applications that require long durations or are energy-constrained.

Under the impact-optimal trajectory condition, fluctuations in the jerk curve are significantly reduced, with an impact range of only [−2.02,2.10], greatly minimizing the impact on the mechanical structure and helping to extend equipment life. However, this also increases the motion time to 86.21 s. Hence, this optimization strategy is suitable for handling tasks that require long life and high reliability, especially in high-load or frequently used scenarios.

In summary, different optimization strategies provide specific advantages and trade-offs for particular application needs. Selecting an appropriate optimization strategy based on specific task requirements can effectively enhance the overall performance and lifespan of the handling robot, ensuring better alignment with industrial reliability and maintenance goals.

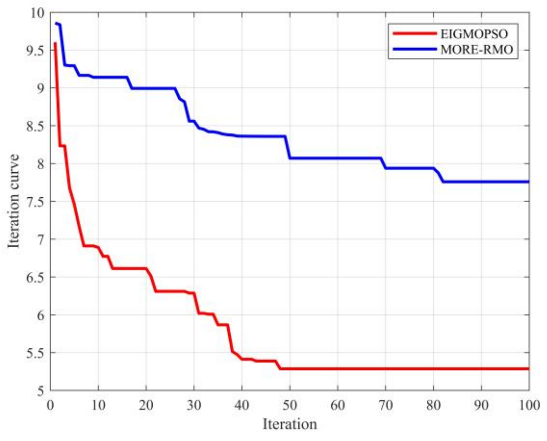


Fig. 7. Iteration curves with the MODE-RMO and the proposed multi-objective algorithm at the comprehensive optimal condition.

To further validate the superiority of the proposed EIGMOPSO in multi-objective trajectory planning for robots, the MODE-RMO, which demonstrates strong performance under IGD and SP metrics, is selected for comparison. Fig. 2 and Fig. 6 present the trajectories of six joints generated by the proposed multi-objective algorithm and MODE-RMO under comprehensive optimization conditions, respectively. The results indicate that the proposed algorithm outperforms MODE-RMO, with the required execution time reduced from 25.73 s to 22.85 s, indicating partial optimization in terms of time. Execution efficiency is improved by 11.21%, and more stable motion control is achieved in angular optimization. For velocity optimization, the velocity curve exhibits smoother transitions at the start and end phases, effectively reducing impact risks. In acceleration optimization, the proposed method significantly decreases the fluctuation range of acceleration curves, minimizing mechanical stress caused by sudden changes. Additionally, the proposed algorithm reduces joint impact, enhancing operational stability. These advantages collectively enable the robot to operate more smoothly and precisely, while also reducing energy consumption and mechanical wear, thus achieving a better balance among multiple objectives. Experimental results demonstrate that the proposed multi-objective trajectory planning algorithm is feasible and excels in trajectory performance, meeting the requirements of practical handling tasks.

Fig. 7 illustrates the iteration curves of MODE-RMO and EIGMOPSO under comprehensive optimization conditions. Compared with MODE-RMO, the proposed algorithm achieves a significantly faster convergence speed. Specifically, while MODE-RMO converges after 82 iterations, EIGMOPSO only requires 48 iterations to reach convergence. Furthermore, the final objective value obtained by EIGMOPSO is 5.286, which is superior to the 7.759 achieved by MODE-RMO. This represents a 31.87% improvement in system efficiency. These results highlight that EIGMOPSO effectively leverages population information, optimizes regional partitioning, and dynamically adjusts search strategies to rapidly identify optimal solutions in complex solution spaces. Consequently, under the same computational resources and time constraints, it delivers high-quality trajectory planning schemes that better satisfy practical handling requirements.

6. Conclusions

This paper addresses the complex multi-objective trajectory optimization problem in handling robot trajectory planning by proposing a MOPSO guided by evolutionary information (EIGMOPSO). The algorithm comprehensively evaluates the population's evolutionary state and divides it into different regions. Differentiated search strategies are designed for particles in different regions, overcoming the shortcomings of traditional MOPSO. Additionally, this paper proposes a dynamic two-stage performance metric-based archive maintenance strategy, effectively improving the solution quality. Experimental validation demonstrates that EIGMOPSO significantly outperforms five existing representative multi-objective optimization algorithms (dMOPSO, CMOPSO, MODE-RMO, SPEA2, NSGA-II) across multiple standard test functions. Notably, in terms of IGD and SP metrics, EIGMOPSO achieves optimal values in most test problems, highlighting the algorithm's strong performance and stability.

In the multi-objective trajectory optimization experiments, this study validates the proposed approach by applying it to the teaching path points within the actual workspace of a handling robot. Without integrating any specific performance metrics, a normalized multi-objective function is established, and the EIGMOPSO is employed to achieve comprehensive multi-objective optimization. This approach simultaneously considers constraints on joint velocity, acceleration, and jerk. The results demonstrate that the proposed multi-objective trajectory optimization approach effectively balances time, energy, and impact. By adjusting the weights or priorities of the optimization objectives, the method demonstrates flexibility in adapting to diverse optimization requirements and application scenarios. It offers effective solutions for enhancing the reliability, efficiency, and maintainability of handling robots, while also providing new insights and approaches for trajectory optimization in industrial applications.

In future research, we intend to further integrate advanced intelligent control technologies to address multi-objective optimization problems in dynamic environments, achieving intelligent trajectory planning that includes real-time environmental perception, dynamic path adjustment, and adaptive optimization. In dynamic environments, handling robots may encounter challenges such as moving obstacles and complex terrains, making multi-objective trajectory planning under these conditions a critical research topic. Through these studies, we aim to develop robust solutions to meet the demands of increasingly complex and diverse industrial applications, further contributing to the reliability and operational efficiency of robotic systems.

References

1. Vaz JC, Kosanovic N, Oh P. ART: Avatar robotics telepresence—the future of humanoid material handling loco-manipulation. Intelligent Service Robotics. 2024;17:237–250. https://doi.org/10.1007/s11370-023-00499-x.
2. Ding I, Su JL. Designs of human–robot interaction using depth sensor-based hand gesture communication for smart material-handling robot operations. Proceedings of the Institution of Mechanical Engineers Part B: Journal of Engineering Manufacture. 2022;237:392–413. https://doi.org/10.1177/09544054221102247.
3. Aleboyeh M, Urbanic J. Development of robotic automation solutions for limp flexible material handling leveraging a finite element modelling technique. The International Journal of Advanced Manufacturing Technology. 2024;132. https://doi.org/10.1007/s00170-024-13229-z.
4. Liu Z, Chen Y, Song H, Xing Z, Tian H, Shan X. High-speed handling robot with bionic end-effector for large glass substrate in clean environment. Sensors (Basel Switzerland). 2021;22. https://doi.org/10.3390/s22010149.
5. Gürel S, Gultekin H, Emiroglu N. Scheduling a dual gripper material handling robot with energy considerations. Journal of Manufacturing Systems. 2023;67:265–280. https://doi.org/10.1016/j.jmsy.2023.01.011.
6. Gultekin H, Gürel S, Taspinar R. Bicriteria scheduling of a material handling robot in an m-machine cell to minimize the energy consumption of the robot and the cycle time. Robotics and Computer-Integrated Manufacturing. 2021;72:102207. https://doi.org/10.1016/j.rcim.2021.102207.
7. Feng H, Yin CB, Cao DH. Trajectory tracking of an electro-hydraulic servo system with a new friction model-based compensation. IEEE/ASME Transactions on Mechatronics. 2023;28(1):473–482. https://doi.org/10.1109/TMECH.2022.3201283.
8. Li J, Shi X, Liang P, Li Y, Lv Y, Zhong M, Han Z. Research on gait trajectory planning of wall-climbing robot based on improved PSO algorithm. Journal of Bionic Engineering. 2024;21(4):1747–1760. https://doi.org/10.1007/s42235-024-00538-y.
9. Yang T, Zhang B, Hong H, Chen Y, Yang H, Wang T, Cao D. Motion control for earth excavation robot based on force pre-load and cross-coupling compensation. Automation in Construction. 2022;141:104402. https://doi.org/10.1016/j.autcon.2022.104402.
10. Liu Y, Li X, Jiang P, Du Z, Wu Z, Sun B, Huang X. Evolutionary multi-objective trajectory optimization for a redundant robot in Cartesian space considering obstacle avoidance. Mechanical Sciences. 2022;13(1):41–53. https://doi.org/10.5194/ms-13-41-2022.
11. Li Y, Wang J, Ji Y. Function analysis of industrial robot under cubic polynomial interpolation in animation simulation environment. International Journal of Interactive Multimedia and Artificial Intelligence. 2020;6(4):105–112. https://doi.org/10.9781/ijimai.2020.11.012.
12. Ding H, Sang Z, Li Z, Shi J, Wang Y, Meng D. Trajectory planning and control of large robotic excavators based on inclination-displacement mapping. Automation in Construction. 2024;158. https://doi.org/10.1016/j.autcon.2023.105209.

13. Ali A, Kazemi R, Azadi S. Scr-normalize: A novel trajectory planning method based on explicit quintic polynomial curves. Proceedings of the Institution of Mechanical Engineers Part K Journal of Multi-body Dynamics. 2020;234(4):1464419320924196. https://doi.org/10.1177/1464419320924196.
14. An J, Li X, Zhang Z, Man W, Zhang G. Joint trajectory planning of space modular reconfigurable satellites based on kinematic model. International Journal of Aerospace Engineering. 2020;2020. https://doi.org/10.1155/2020/8872788.
15. Ji Z, Zhang G, Cheng M, Kong M, Li R. A convex optimization method to time-optimal trajectory planning with jerk constraint for industrial robotic manipulators. IEEE Transactions on Automation Science and Engineering. 2023. https://doi.org/10.1109/TASE.2023.3346693.
16. Kang M, Ni F, Liu H. Robotic abrasive belt grinding of complex curved blades based on a novel force control architecture integrating smooth trajectories. Journal of Manufacturing Processes. 2023;107:447–458. https://doi.org/10.1016/j.jmapro.2023.10.048.
17. Shi Q, Wang Z, Ke X, Zheng Z, Zhou Z, Wang Z, Fan Y, Lei B, Wu P. Trajectory optimization of wall-building robots using response surface and non-dominated sorting genetic algorithm III. Automation in Construction. 2023;155. https://doi.org/10.1016/j.autcon.2023.105035.
18. Zhang X, Xiao F, Tong X, Yun J, Liu Y, Sun Y, Tao B, Kong J, Xu M, Chen B. Time optimal trajectory planning based on improved sparrow search algorithm. Frontiers in Bioengineering and Biotechnology. 2022;10:852408. https://doi.org/10.3389/fbioe.2022.852408.
19. Nonoyama K, Liu Z, Fujiwara T, Alam MM, Nishi T. Energy-efficient robot configuration and motion planning using genetic algorithm and particle swarm optimization. Energies. 2022;15(6):2074–2074. https://doi.org/10.3390/en15062074.
20. Song L, Ding B, Li Y. Minimum-jerk trajectory planning pertaining to a translational 3-degree-of-freedom parallel manipulator through piecewise quintic polynomials interpolation. Advances in Mechanical Engineering. 2020;12(3):1687814020913667–1687814020913667. https://doi.org/10.1177/1687814020913667.
21. Xia L. Trajectory planning application of rehabilitation robots based on improved csa algorithm. IEEE Access. 2023;11:144617–144630. https://doi.org/10.1109/ACCESS.2023.3343734.
22. Wu P, Wang Z, Jing H, Zhao P. Optimal time–jerk trajectory planning for delta parallel robot based on improved butterfly optimization algorithm. Applied Sciences. 2022;12(16):8145–8145. https://doi.org/10.3390/app12168145.
23. 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. Chinese Journal of Mechanical Engineering. 2022;35(1). https://doi.org/10.1186/s10033-021-00669-x.
24. Sun H, Tao J, Qin C, Dong C, Xu S, Zhuang Q, Liu C. Multi-objective trajectory planning for segment assembly robots using a b-spline interpolation- and infeasible-updating non-dominated sorting-based method. Applied Soft Computing. 2024;152:111216. https://doi.org/10.1016/j.asoc.2023.111216.
25. Coello CA, Lechuga MS. MOPSO: A proposal for multiple objective particle swarm optimization. IEEE Service Center. 2002. https://doi.org/10.1109/CEC.2002.1004388.
26. Han H, Liu Y, Hou Y, et al. Data-driven robust multimodal multiobjective particle swarm optimization. IEEE Trans Syst Man Cybern Syst. 2025;54. https://doi.org/10.1109/TSMC.2024.3357872.
27. Wang YF, Zhang L. Improved multi-objective particle swarm optimization algorithm based on area division with application in multi-UAV task assignment. IEEE Access. 2023;11:123519 – 123530. https://doi.org/10.1109/ACCESS.2023.3328344.
28. Zhang X, Wang Z, Lu Z. Multi-objective load dispatch for microgrid with electric vehicles using modified gravitational search and particle swarm optimization algorithm. Appl Energy. 2022;306:118018. https://doi.org/10.1016/j.apenergy.2021.118018.
29. Su S, Xiong D, Yu H, Dong X. A multiple leaders particle swarm optimization algorithm with variable neighborhood search for multiobjective fixed crowd carpooling problem. Swarm Evol Comput. 2022;72:101103. https://doi.org/10.1016/j.swevo.2022.101103.
30. Zhou W, Chen F, Ji X, Li H, Zhou J. A Pareto-based discrete particle swarm optimization for parallel casting workshop scheduling problem with fuzzy processing time. Knowl-Based Syst. 2022;256(28):109872. https://doi.org/10.1016/j.knosys.2022.109872.
31. Huang P, Liu G, Yuan J, et al. Multi-objective optimal trajectory planning of space robot using particle swarm optimization. DBLP. 2008;:20. https://doi.org/10.1007/978-3-540-87734-9_20.
32. Lan J, Xie Y, Liu G, Cao M. A multi-objective trajectory planning method for collaborative robot. Electronics. 2020;9(5). https://doi.org/10.3390/electronics9050859.

33. Chen W, Wang H, Liu Z, Jiang K. Time-energy-jerk optimal trajectory planning for high-speed parallel manipulator based on quantum-behaved particle swarm optimization algorithm and quintic b-spline. Engineering Applications of Artificial Intelligence. 2023;126(PC):107223. https://doi.org/10.1016/j.engappai.2023.107223.
34. Cao X, Yan H, Huang Z, Ai S, Xu Y, Fu R, Zou X. A multi-objective particle swarm optimization for trajectory planning of fruit picking manipulator. Agronomy. 2021;11(11):2286–2286. https://doi.org/10.3390/agronomy11112286.
35. Wu J, Wei C, Zhang H, Liu Y, Li K. Learning-based spacecraft multi-constraint rapid trajectory planning for emergency collision avoidance. Aerospace Science and Technology. 2024;149:109112. https://doi.org/10.1016/j.ast.2024.109112.
36. Lyche T, Winther R. A stable recurrence relation for trigonometric b-splines. Journal of Approximation Theory. 1979;25(3):266–279. https://doi.org/10.1016/0021-9045(79)90017-0.
37. Phan HT, Nguyen VH, Konigorski U. A time-optimal trajectory generation approach with non-uniform b-splines. International Journal of Control Automation and Systems. 2021;19(12):3947–3955. https://doi.org/10.1007/s12555-020-0497-3.
38. Xu XF, Wang K, Ma WH, Wu CL, Huang XR, Ma ZX, Li ZH. Multi-objective particle swarm optimization algorithm based on multi-strategy improvement for hybrid energy storage optimization configuration. Renewable Energy. 2024;223:120086. https://doi.org/10.1016/j.renene.2024.120086.
39. Zhao X, Zhang Y, Ding W, Tao B, Ding H. A dual-arm robot cooperation framework based on a nonlinear model predictive cooperative control. IEEE/ASME Transactions on Mechatronics. 2023. https://doi.org/10.1109/TMECH.2023.3263357.
40. Sun Y, Chang Y, Yang S, Wang F. Dynamic niching particle swarm optimization with an external archive-guided mechanism for multi-modal multi-objective optimization. Information Sciences. 2024;653. https://doi.org/10.1016/j.ins.2023.119794.
41. Neri F, Cotta C. Memetic algorithms and memetic computing optimization: A literature review. Swarm and Evolutionary Computation. 2011;2:1–14. https://doi.org/10.1016/j.swevo.2011.11.003.
42. Schott JR, Ramon J. Fault tolerant design using single and multicriteria genetic algorithm optimization. Thesis (M.S.), Massachusetts Institute of Technology, Dept. of Aeronautics and Astronautics. 2005;199–200.
43. Zitzler E, Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: empirical results. Evolutionary Computation. 2000;8(2):173–195. https://doi.org/10.1162/106365600568202.
44. Huband S, Hingston P, Barone L, While L. A review of multiobjective test problems and a scalable test problem toolkit. IEEE Transactions on Evolutionary Computation. 2006;10(5):477–506. https://doi.org/10.1109/TEVC.2005.861417.
45. Zapotecas Martínez S, Coello Coello CA. A multi-objective particle swarm optimizer based on decomposition. CINVESTAV-IPN, México, DF, México. 2011.
46. Zhang X, Zheng X, Cheng R, Qiu J, Jin Y. A competitive mechanism based multi-objective particle swarm optimizer with fast convergence. Information Sciences. 2018;427:63–76. https://doi.org/10.1016/j.ins.2017.10.037.
47. Chen X, Du W, Qian F. Multi-objective differential evolution with ranking-based mutation operator and its application in chemical process optimization. Chemometrics and Intelligent Laboratory Systems. 2014;136:85–96. https://doi.org/10.1016/j.chemolab.2014.05.007.
48. Zitzler E, Laumanns M, Thiele L. Spea2: Improving the strength pareto evolutionary algorithm. Technical Report Gloriastrasse. 2001. https://doi.org/10.3929/ethz-a-004284029.
49. Deb K, Agrawal S, Pratap A, Meyarivan T. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. Springer. 2000. https://doi.org/10.1007/3-540-45356-3_83.