IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 7 April 2023, accepted 30 April 2023, date of publication 3 May 2023, date of current version 10 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3272835

RESEARCH ARTICLE

Robot Time Optimal Trajectory Planning Based on Improved Simplified Particle Swarm Optimization Algorithm

XIAO HU[1], HENG WU[1], QIANLAI SUN[1], AND JUN LIU[2]
[1]School of Electronic Information Engineering, Taiyuan University of Science and Technology, Taiyuan, Shanxi 030024, China
[2]Innovation Research Institute, CITIC Heavy Industry Machinery Company Ltd., Luoyang, Henan 471033, China

Corresponding author: Xiao Hu (xiaoxiaoshine@126.com)

This work was supported in part by the Key Research and Development Program of Shanxi Province under Grant 202102020101005, in part by the Scientific and Technological Innovation Programs of Higher Education Institutions in Shanxi under Grant 2020L0343, and in part by the Major Science and Technology Project of Shanxi Province under Grant 20191102009.

ABSTRACT In order to tackle the robot trajectory planning problem with the short running time as the optimization goal, a time-optimal trajectory planning algorithm was presented based on improved simplified particle swarm optimization (ISPSO). The robot's trajectory was constructed by 3-5-3 polynomial interpolation in the joint space of the robot. Under the condition of satisfying the velocity constraint, the objective function was constructed by the sum of the time intervals between each node. ISPSO was used to optimize the objective function. The algorithm was improved by optimizing the inertia weight updating method and introducing a golden sine segmentation algorithm as an optimization operator. Compared with other particle swarm optimization algorithms, ISPSO had higher search velocity and accuracy. The effectiveness of the proposed algorithm was demonstrated through simulations using the PUMA 560 industrial robot, which resulted in a 19% reduction in time compared to the simplified particle swarm algorithm. The simulation results show that ISPSO achieved time optimization under the condition of velocity constraint, which proved its superiority in trajectory planning.

INDEX TERMS Trajectory planning, time-optimal, PSO, polynomial interpolation.

I. INTRODUCTION

Path planning is the basic field of robot kinematic research. The terminal trajectory planning of industrial robots determines whether the robot can accurately complete the task. Trajectory planning of industrial robots has very important practical significance because it determines the robot's energy consumption, production efficiency, and service life [1], [2], [3].

The trajectory planning of robots is a highly complex and nonlinear control problem. According to different optimization objectives, trajectory planning can be divided into time optimization [4], operational stability [5], mechanical life [6], and optimal combinations [7], [8], [9]. The main purpose of optimal path planning is to obtain the optimal trajectory through the path points according to the path points given by the task under the condition of satisfying the motion constraints [10]. To meet the constraint conditions of path points, it is necessary to introduce a trajectory planning algorithm. The commonly used trajectory planning algorithm is an interpolation algorithm divided into polynomial functions and b-spline curves. Su and Zou [11] proposed an algebraic and trigonometric polynomial that can replicate the continuity of displacement, velocity, and acceleration in the joint space of the robot and realize the continuous and smooth operation of the robot. Sidobre and Desormeaux [12] proposed a solution method based on a cubic polynomial function segment sequence. Liu et al. [13] proposed a design method combining polynomial and b-spline planning to ensure the smooth motion of the robot and the tracking performance.

The associate editor coordinating the review of this manuscript and approving it for publication was Yingxiang Liu.

Wang et al. [14] proposed an improved b-spline interpolation method to improve the tracking accuracy and motion smoothness of the b-spline by adding correction functions. Despite the progress reported in the literature, these traditional algorithms have high computational order and lack of convex, which is difficult to solve using the traditional optimization method. Thus, intelligent optimization algorithms for problem solving and trajectory planning [15], [16] have gained popularity and represent a promising area for future research. The commonly used intelligent optimization algorithms [17] include the genetic algorithm [18], particle swarm algorithm [19], and ant colony algorithm [20], among others. Because of its few parameters, simplicity and easy implementation, particle swarm algorithm has been a research hotspot [21], [22].

Mazhoud et al. [23] proposed a new particle swarm constraint mechanism to transform constrained problems into unconstrained two-objective optimization problems through arithmetic intervals for global optimal and historically optimal particles. Zhang and Xing [24] proposed a fuzzy particle swarm algorithm for the uncertainty of conditions in industrial engineering, which can effectively solve the problem of time and quality optimization of engineering, but it is limited because it can only generate a single solution according to preference. Literature [25] proposed a particle swarm algorithm based on the mixed coding method, which randomly selects excellent particles by the CR method. This improved the convergence and diversity of populations. However, the correlation between discrete and continuous variables needs further study. Through multiple strategies, Xia improved the performance of particle swarm algorithms. A particle swarm algorithm with multiple adaptive strategies has also been proposed, effectively improving the comp algorithm's comprehensive performance through various rm and population adaptive strategies [26]. Three new renewal strategies were introduced to improve the population's exploration and development capacity, and the population's ability to escape the local optimal was improved [27].

The time-optimal trajectory planning problem can be expressed by an objective mathematical function [28] and then the established objective function can be optimized using optimization tools. In a recent study, Sadhu et al. [29] explored a trajectory planning algorithm using the synergistic effect of the firefly algorithm and Q-learning, which improved the velocity and stability of the robot. Wang et al. [30] reported a smoothed point-to-point trajectory planning method for industrial robots, which obtains the near-time-optimal trajectory by maximizing the isokinetic part; compared with the classical description, the arduous stage of solving many polynomial coefficients could be simplified. Wang et al. [31] proposed a trajectory optimization algorithm inspired by Beetle, which has good search velocity and control performance without increasing algorithm complexity. Finally, Zhao et al. [32] used a hybrid improved whale optimization and PSO algorithm to effectively reduce the jitter of the robot and improve the working efficiency of the robot. All the above methods achieve high quality and efficient solutions to a certain extent, but they need many iterations and complex parameter debugging.

An optimal-time 3-5-3 polynomial interpolation trajectory planning algorithm based on an improved, simplified particle swarm optimization (ISPSO) algorithm under velocity constraints is proposed in this paper. First, the ISPSO uses random inertia weight to balance the development and utilization of simplified PSO. Then, the golden sine algorithm is introduced into the latter stage of PSO as an optimization operator to improve the convergence accuracy of PSO to improve convergence velocity and accuracy. Finally, an ISPSO algorithm is applied to joint space trajectory planning, which reduces the running time of the robot and improves production efficiency.

The research content of this paper includes four parts: the first part reviews and introduces the current relevant research. The second part mainly introduces the polynomial interpolation trajectory of the robot. The third part proposes a new simplified particle swarm algorithm. The fourth part verified by testing the function and establishing the robot model. Finally, the fifth part summarizes the article and discusses the result.

II. PROBLEM DESCRIPTION AND TRAJECTORY CONSTRUCTION

A. PROBLEM DESCRIPTION

This paper deals with the time optimization of a point-to-point trajectory in joint robot space. First, according to the robot's activity range and application scenario, several spatial pose points must pass through the Cartesian space. Then, the joint angles and nodes in the joint space can be obtained through the inverse kinematics of the robot. If the time for the robot to reach each joint node is t_m, m is the number of nodes ($m = 0,1,2\ldots$); the optimization goal is to enable the robot to pass through each joint node in the shortest time under the condition that the constraint conditions are met. The interval between adjacent nodes is denoted as $h_i = t_{i+1} - t_i$. The objective optimization function is the minimum running time of each joint satisfying the velocity constraint condition. The objective function is shown in equation (1) and the velocity constraint condition is shown in equation (2).

$$\min T = \sum_{i=0}^{m-1} h_i \tag{1}$$

$$|\dot{\theta}_{\mathrm{i}}| \leq v_{max} \tag{2}$$

where T is the total time to run, v_{max} is the limiting velocity of the joint, and $\dot{\theta}_{\mathrm{i}}$ is the real-time velocity of the polynomial trajectory segment.

B. TRAJECTORY CONSTRUCTION

If we use only cubic polynomials for trajectory planning, the angular acceleration is discontinuous, and using only

five-degree polynomials for trajectory planning will result in more computation. Xu et al. [33] proposed a 3-5-3 spline splicing method, which can specify the angular velocity at the intermediate target point arbitrarily, and the angular acceleration is continuous. This paper uses a 3-5-3 polynomial curve to fit the starting, intermediate, and ending points. The general formula of the 3-5-3 spline polynomial is shown in the following equation.

$$\theta_{i1} = a_{i13}t^3 + a_{i12}t^2 + a_{i11}t^1 + a_{i10} \tag{3}$$
$$\theta_{i2} = a_{i25}t^5 + a_{i24}t^4 + a_{i23}t^3 + a_{i22}t^2 + a_{i21}t^1 + a_{i20} \tag{4}$$
$$\theta_{i3} = a_{i33}t^3 + a_{i32}t^2 + a_{i31}t^1 + a_{i30} \tag{5}$$

where the coefficient $a_{i1j}, a_{i2j}, a_{i3j}$ is the j-th coefficient of the interpolation function at the first, second, and third sections of the trajectory of the i-th joint and θ_{ik} represents the polynomial trajectory at the k-th section of the i-th joint.

In the process of trajectory planning, the constraints include the velocity and acceleration (generally 0) of the initial point X_{i0}, the intermediate point X_{i1} and X_{i2}, and the termination point X_{i3} of each joint. The velocity and acceleration between the path points are continuous. According to the above conditions, the relationship equations (6)-(8) between the coefficient a_{ij} and interpolation point are deduced, where, X_{im} represents the interpolation position of the i-th joint and m represents the serial number of interpolation points, m = 0,1,2,3.

$$A = \begin{bmatrix} B & C & 0 \\ 0 & D & E \\ 0 & 0 & F \\ G & 0 & 0 \\ 0 & H & I \end{bmatrix} \tag{6}$$
$$X = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & X_{i3} & 0 & 0 & X_{i0} & 0 & 0 & X_{i1} & X_{i2} \end{bmatrix}^T \tag{7}$$
$$a = A^{-1}X = \begin{bmatrix} A_1 & A_2 & A_3 \end{bmatrix}^T \tag{8}$$

where

$$B = \begin{bmatrix} t_{i1}^3 & t_{i1}^2 & t_{i1}^1 & 1 \\ 3t_{i1}^2 & 2t_{i1} & 1 & 0 \\ 6t_{i1} & 2 & 0 & 0 \end{bmatrix}$$
$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -2 & 0 & 0 \end{bmatrix}$$
$$D = \begin{bmatrix} t_{i2}^5 & t_{i2}^4 & t_{i2}^3 & t_{i2}^2 & t_{i2} & 1 \\ 5t_{i2}^4 & 4t_{i2}^3 & 3t_{i2}^2 & 2t_{i2} & 1 & 0 \\ 20t_{i2}^3 & 12t_{i2}^2 & 6t_{i2} & 2 & 0 & 0 \end{bmatrix}$$
$$E = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 0 & -2 & 0 & 0 \end{bmatrix},$$
$$F = \begin{bmatrix} t_{i3}^3 & t_{i3}^2 & t_{i3} & 1 \\ 3t_{i3}^2 & 3t_{i3} & 1 & 0 \\ 6t_{i3} & 2 & 0 & 0 \end{bmatrix}$$
$$G = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$
$$H = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$
$$I = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$
$$A_1 = \begin{bmatrix} a_{i13} & a_{i12} & a_{i11} & a_{i10} \end{bmatrix}$$
$$A_2 = \begin{bmatrix} a_{i25} & a_{i24} & a_{i23} & a_{i22} & a_{i21} & a_{i20} \end{bmatrix}$$
$$A_3 = \begin{bmatrix} a_{i33} & a_{i32} & a_{i31} & a_{i30} \end{bmatrix}$$

III. IMPROVED SIMPLIFIED PARTICLE SWARM OPTIMIZATION(ISPSO)

A. SIMPLIFIED PARTICLE SWARM OPTIMIZATION

Particle swarm optimization (PSO) is a swarm-based stochastic optimization algorithm derived from the simulation of group predation behavior of birds. Each particle evolves in search of an optimal solution, guided by group and individual experience. The updating formula of position and velocity of the standard particle swarm with inertial weight is shown in equation (9)-(10).

$$v_{id}^{k+1} = w \times v_{id}^k + c_1 \times rand \times (p_{best} - x_{id}^k) + c_2 \times rand \times (g_{best} - x_{id}^k) \tag{9}$$
$$x_{id}^{k+1} = x_{id}^k + v_{id}^{k+1} \tag{10}$$

where k is the number of iterations. w is inertial weight. c_1 and c_2 are learning factors. rand is a random number between [0,1]. p_{best} is the individual historical optimal position of the particle. g_{best} is the optimal global position of the particle.

In their study, Hu et al. conducted an analysis of the velocity term of the standard particle swarm algorithm [34]. They demonstrated that this term was not a necessary component for the evolutionary process of PSO, and that, in fact, could lead to the particle deviation from the correct evolution direction. Therefore, the simplified particle swarm algorithm (SPSO) was proposed and the particle position is updated through the following expression in equation (11).

$$x_{id}^{k+1} = wx_{id}^k + c_1 rand(p_{best} - x_{id}^k) + c_2 rand(g_{best} - x_{id}^k) \tag{11}$$

By comparing equations (9), (10), and (11), it can be concluded that there is no particle velocity parameter in the renewal method of equation (11), which simplifies the particle control process and avoids the influence of an artificially determined velocity parameter $[V_{min}, V_{max}]$ on the particle convergence velocity and accuracy (V_{max} and V_{min} refer to the upper and lower limits of the particle's flight velocity).

B. IMPROVEMENT OF INERTIA WEIGHT

Inertia weight is one of the important parameters of the simplified PSO algorithm. From equation (11), it can be seen that the size of w determines the influence of the position of the last particle in this iteration. For a large size w, the early stage of search is conducive to global search, while for a small size w the late stage is conducive to improving local mining ability. Thus far, many improved strategies for inertia weight have been proposed, but it has been shown that among the total 18 inertial weight strategies, random inertial weight is superior to all other inertial strategies except constant inertial weight [35].

Aiming at the tedious parameter debugging for the inertia weights of constant terms, this paper proposes a new random inertia weight strategy based on swarm information updating. Using the characteristics of random variables to adjust the inertia weight can make the algorithm jump out of the local optimal quickly, which is beneficial to maintain the diversity of the algorithm and improve the global search ability of the algorithm. When the optimal global value g_{best} does not change within a certain searching algebraic range, a larger value w is given randomly to expand the searching range and jump out of the optimal local. When it can be updated constantly, the weight is expected to balance the global search ability and local development ability. In this paper, the average fitness information of the swarm, the optimal, and the worst particle information are introduced into the inertial weight, so that the particles can use the swarm information to balance the global search and local search ability in the iterative process. Finally, an updating method of random weight is proposed based on the above analysis.

When the algorithm is judged to be trapped in the local optimal, the updating mode of inertia weight is shown in equation (12).

$$w = rand(r_2 - r_1) + r_1 \tag{12}$$

When the algorithm does not fall into an optimal local, the updating mode of inertia weight is shown in equation (13).

$$w = \frac{f_{mean} - f_{best}}{f_{wrost} - f_{best}} \tag{13}$$

where f_{mean} is the average fitness value of the current iteration swarm. f_{best} is the minimum fitness value in the current iteration. f_{wrost} is the worst fitness target value in the current iteration. The selection interval of r_1 and r_2 is [0.5, 1.2]. When the optimal global value g_{best} does not change for five consecutive times, it is judged to be trapped in the optimal local value and the weights are updated using Equation 12; otherwise Equation 13 is used.

C. GOLD SINE GUIDING MECHANISM

The golden Sine algorithm [36] (GSA) is a meta-heuristic algorithm that constructs a mathematical model based on a sine function to solve optimization problems. The strong global search ability of GSA arises from its ability to traverse all points on the unit circle through the geometric relationship between the unit circle and the sine function. This, coupled with its convergence characteristics, few control parameters, strong robustness, and few operator numbers, has made GSA a widely used in algorithm improvement and engineering applications. At the same time, the golden section coefficient is introduced to accelerate the convergence velocity of the algorithm, allowing for each iteration to explore the region near the optimal solution and promoting strong local development ability. The location update formula for GSA is shown in equation (14).

$$X_i^{t+1} = X_i^t |sin(R_1)| + R_2 sin(R_2) |x_1 P_i^t - x_2 X_i^t| \tag{14}$$

where, t is the number of iterations. X_i^t is the position of the i-th individual in the t iteration. R_1 and R_2 are random numbers, $R_1 \in [0, 2\pi], R_2 \in [0, \pi]$. R_1 determines the moving distance of the individual in the next iteration. R_2 determines the moving direction of the individual in the next iteration. P_i^t represents the optimal position of the optimal individual i in the t iteration. x_1 and x_2 are gold sinusoidal segmentation coefficients used to refine the search space. To balance search and development, the Golden Segmentation achieves the following:

$$x_1 = a(1-h) + bh \tag{15}$$

$$x_2 = ah + b(1-h) \tag{16}$$

where: a and b are the initial values of the golden section, the golden sine section is divided by the standard sine interval, since the period of the sine division function $\sin x$ is 2π, according to its definition and the relationship with the unit circle, to traverse the whole search space, here take $a = p$, $b = -\pi$. h is the golden section ratio, usually $h = \left(\sqrt{5} - 1\right)/2 \approx 0.61833$. A specific derivation can be found in [36].

In this paper, an ISPSO algorithm is proposed to improve the convergence accuracy and accelerate the convergence velocity of the swarm. Furthermore, the golden sine segmentation algorithm is introduced into the later stage of the PSO as an optimization operator. In ISPSO, the search agent is updated twice in equation (14).

The golden sine guiding mechanism continuously splits the search space to obtain the best search space for the search agent optimization process, resulting in two large improvements:

(1) The particle search is carried out within the optimal range, and thus it is easier for the particle to find the optimal global position;

(2) The search range of particles is reduced by the golden ratio, which enables the particle swarm to approach the optimal area quickly and improves the convergence velocity of the algorithm.

D. ISPSO ALGORITHM STEPS

The algorithm steps of ISPSO are summarized as follows:

TABLE 1. Test functions.

Function	Range	Min
$F_1(x)=\sum_{i=1}^{n} x_i^2$	[−100,100]	0
$F_2(x)=\sum_{i=1}^{n} \lvert x_i \rvert+\prod_{i=1}^{n} \lvert x_i \rvert$	[−10,10]	0
$F_3(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i} x_j)^2$	[−100,100]	0
$F_4(x)=\max_i\{\lvert x_i \rvert, 1\le i\le n\}$	[−100,100]	0
$F_5(x)=-\sum_{i=1}^{n} x_i \sin(\sqrt{\lvert x_i \rvert})$	[−500,500]	−418.95*n
$F_6(x)=0.1*\{\sin^2 3\pi x_i+\sum_{i=1}^{n}(x_i-1)^2[1+\sin^2(3\pi x_i+1)]+(x_i-1)^2*[1+\sin^2(2\pi x_i)]\}+\sum_{i=1}^{n} u(x_i,5,100,4)$	[−50,50]	0
$F_7(x)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2})-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i))+20+e$	[−32,32]	0
$F_8(x)=\frac{\pi}{n}\{10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_{i-1}^2)*[1+\sin^2(\pi y_{i+1})]+(y_n-1)^2\}$ $y_i=1+\frac{x_i}{4}, u=\begin{cases}k(x_i-a)^m, x_i>a\\ 0, -a<x_i<a\\ k(-x_i-a)^m, x_i<a\end{cases}$	[−50,50]	0

Step1: Set parameters. The maximum number of iterations T, the number of particles N, and the dimension Dim of the fitness function.

Step2: Initialization. The particle position is initialized randomly, the fitness value of each particle is calculated, and the optimal particle is recorded.

Step3: Update of the inertia weight coefficient of the swarm according to equation (12).

Step4: Update of the position of the particle swarm according to equation (11), calculation of the particle's fitness, and update of the historical position of the optimal swarm particle and the individual optimal particle.

Step5: For the particles generated in step 4, generation of a new particle swarm guided by the golden sine according to equation (14).

Step6: Calculation of the fitness of the swarm again and update of the swarm's optimal particle and the individual's historical optimal particle.

Step7: Repetition of steps 3-6 until the maximum number of iterations is reached.

IV. SIMULATION AND ANALYSIS

A. TEST AND ANALYSIS OF SEARCH ABILITY

1) PARAMETER SETTINGS AND SIMULATION RESULTS

The simulation environment is a 64-bit window 10 operating system, Intel Core i5-6300hq CPU, Matlab 2018a. To verify the optimization ability of the algorithm, PSO [37], SPSO, GSA, double-swarm particle swarm optimization (DPSO) [38], dynamically adjusted simplified particle swarm optimization (DSMPSO) [39], and the ISPSO proposed in this paper are compared by 8 test functions [40] in Tabl. 1, which include single-peak (F_1–F_4) and multi-peak (F_5–F_8) functions.

TABLE 2. Parameter settings.

Algorithm	Parameter settings
PSO	$c_1=c_2=0.05$, $w=0.5(1+rand)$
SPSO	$c_1=c_2=2$, $w=0.8$
DPSO	$c_1=c_2=2$
DSMPSO	$w_{max}=0.9$, $w_{min}=0.2$, $\sigma=0.1$
GSA	$a=\pi$, $b=-\pi$
ISPSO	$c_1=c_2=2$

To ensure the fairness of the experiment, the swarm size of all algorithms was set to 30, the maximum number of iterations was set to 1,000, and the dimensions were set to 3 classes for testing (i.e., $D=10$, 20 and 30). The settings of various algorithms are presented in Table 2. After 30 independent operations, the mean value and standard deviation were calculated as the evaluation indexes. The experimental results are shown in Table 3 to 5, and the convergence curve of the test function is shown in Figure 1.

TABLE 3. Results obtained by different algorithms (D = 10).

FUNCTION		PSO	SPSO	DPSO	DSMPSO	GSA	ISPSO
F_1	MEAN	9.86	5.33E-25	1.57E-87	5.46E-101	**0**	**0**
	STD	1.99E + 01	2.01E-24	6.09E-87	1.55E-100	**0**	**0**
F_2	MEAN	4.10E-01	2.63E-16	2.01E-54	1.09E-46	2.08E-299	0
	STD	3.80E-01	8.6E-16	7.80E-54	4.16E-46	**0**	**0**
F_3	MEAN	4.57E+01	2.73E-21	6.99E-119	3.01E-77	**0**	**0**
	STD	4.34E+01	1.055E-20	2.67E-118	1.16E-76	**0**	**0**
F_4	MEAN	3.21	8.009E-16	9.79E-50	2.041E-44	2.65E-296	**0**
	STD	1.96	1.45E-15	3.79E-49	7.90E-44	0	**0**
F_5	MEAN	−2.381E+03	−4.18979E+03	-2.352E+03	−4.18981E+03	−4.1837E+03	**−4.18982E+03**
	STD	2.82E+02	8.09E-02	2.38E+02	1.03E-02	1.52E+01	**2.10E-03**
F_6	MEAN	3.53	2.32E-05	1.52E-1	2.02E-05	1.40E-2	**1.74E-05**
	STD	3.90	2.65E-05	2.67E-2	2.54E-05	8.36E-3	**2.05E-05**
F_7	MEAN	3.49	1.64E-13	4.44E-15	**8.88E-16**	**8.88E-16**	**8.88E-16**
	STD	1.31	3.58E-13	0	**0**	**0**	**0**
F_8	MEAN	3.45	1.74E-06	3.2E-2	7.99E-06	5.9E-3	**3.46E-06**
	STD	4.16	2.39E-06	1E-2	1.05E-05	3.8E-3	**3.88E-3**

TABLE 4. Results obtained by different algorithms (D = 20).

FUNCTION		PSO	SPSO	DPSO	DSMPSO	GSA	ISPSO
F_1	MEAN	5.12E+02	1.04E-26	6.84E-102	1.031E-85	**0**	**0**
	STD	3.03E+02	3.95E-26	2.65E-101	3.99E-85	**0**	**0**
F_2	MEAN	4.69	3.28E-16	7.03E-58	2.28E-43	6.30E-302	**0**
	STD	1.60	1.14E-15	2.72E-57	8.86E-43	**0**	**0**
F_3	MEAN	1.99E+03	8.18E-17	1.22E-89	2.38E-84	**0**	**0**
	STD	1.11E+03	3.16E-16	4.76E-89	8.75E-84	**0**	**0**
F_4	MEAN	1.61E+01	6.28E-15	2.69E-48	9.47E-48	1.12E-297	**0**
	STD	2.90	1.63E-14	1.04E-47	3.58E-47	**0**	**0**
F_5	MEAN	−4.218E+03	−8.37962E+03	−3.153E+03	−8.3796E+03	−8.376E+03	**−8.379649E+03**
	STD	4.17E+02	4.03E-02	2.86E+02	6.25E-02	6.14	**8.249E-03**
F_6	MEAN	4.21E+03	4.69E-05	8.19E-01	4.07E-05	6.8E-02	**6.74E-06**
	STD	1.38E+04	8.99E-05	8.0E-02	5.10E-05	2.65E-02	**5.49E-06**
F_7	MEAN	8.81	1.36E-15	4.91E-15	**8.81E-16**	**8.81E-16**	**8.81E-16**
	STD	1.143	1.25-15	1.25E-15	**0**	**0**	**0**
F_8	MEAN	1.27E+01	2.63E-06	0.121	5.45E-06	8.11E-03	**1.76E-06**
	STD	1.05E+01	5.62E-06	3.09E-02	8.55E-06	4.04E-03	**1.45-06**

2) SIMULATION RESULT ANALYSIS

The results obtained were synthesized in conjunction with Tables 3 to 5. For the single-peak test function F_1 to F_4, it can be seen that when D = 10, 20, 30, the optimization effect of GSA is almost the same as that of ISPSO, but combined with the optimization curve of D = 30, the convergence speed of only ISPSO's algorithm is much better than that of GSA. Moreover, it can be seen from the results in the table that the optimization effect of ISPSO in each dimension of the single-peak function is far better than that of several other algorithms. From the optimization results of the multi-peak function from F_5 to F_8, when the dimension D = 10, 20, 30, the optimization accuracy of ISPSO is better than that produced by other algorithms; only on the function F_7, GSA, DSMPSO, and ISPOS have the same optimization accuracy. From comparing the test function results in the three dimensions, IMOPSO emerges as being generally better than several other algorithms.

TABLE 5. Results obtained by different algorithms (D = 30).

FUNCTION		PSO	SPSO	DPSO	DSMPSO	GSA	ISPSO
F_1	MEAN	1.648E+03	1.40E-24	3.25E-78	9.70E-92	**0**	**0**
	STD	5.97E+02	7.01E-24	1.63E-77	4.82E-91	**0**	**0**
F_2	MEAN	1.49E+01	4.24E-14	1.46E-55	5.04E-43	1.59E-303	**0**
	STD	3.9E+00	2.05E-13	7.28E-55	2.51E-42	**0**	**0**
F_3	MEAN	5.232E+03	1.91E-20	1.85E-94	4.12E-67	**0**	**0**
	STD	1.608E+03	9.54E-20	9.16E-94	2.04E-66	**0**	**0**
F_4	MEAN	2.15E+01	2.14E-14	7.98E-39	1.80E-48	1.34E-298	**0**
	STD	4.04E+00	1.01E-13	3.99E-38	9.01E-48	**0**	**0**
F_5	MEAN	−5.389E+03	−1.25693E+03	−3.9184E+03	−1.25693E+03	−1.25115E+03	**−1.25695E+03**
	STD	7.59E+02	1.73E-01	2.78E+02	3.12E-01	7.91E+01	**1.6892E-02**
F_6	MEAN	6.62E+03	2.86E-05	2.04E+00	2.72E-05	1.58E-1	**1.33E-05**
	STD	1.168E+05	4.22E-05	2.68E-01	2.88E-05	4.42E-2	**1.88E-05**
F_7	MEAN	1.06E+01	1.69E-13	5.86E-15	**8.88E-16**	**8.88E-16**	**8.88E-16**
	STD	1.096E+00	6.89E-13	1.78E-15	**0**	**0**	**0**
F_8	MEAN	1.57E+02	3.49E-06	2.21E-01	4.56E-06	1.32E-02	**6.97E-07**
	STD	4.28E+02	5.74E-06	3.42E-02	6.73E-06	6.844E-03	**8.10E-07**

TABLE 6. Results obtained for different algorithms.

Algorithm	Dim	PSO	SPSO	DPSO	DSMPSO	GSA	ISPSO
Average Rank	30	5.3750	4.3125	3.7500	3.1875	2.8750	**1.5**
Average Rank	20	5.875	3.875	4.125	3.25	2.6250	**1.2500**
Average Rank	10	5.875	4.000	4.125	3.000	2.625	**1.375**

TABLE 7. Algorithm run time (D = 30).

FUNCTION	PSO	SPSO	DPSO	DSMPSO	GSA	ISPSO
F_1	0.301	0.335	0.479	0.325	0.304	0.496
F_2	0.326	0.365	0.501	0.329	0.312	0.464
F_3	0.321	0.326	0.459	0.320	0.247	0.490
F_4	0.309	0.338	0.484	0.351	0.30	0.493

Because the article is limited in space, the particular case of D = 30 and the curve iteration are analyzed in detail. As seen from the experimental results in Table 5 and Figur. 1, for all test functions, the mean and standard deviation of the ISPSO algorithm is significantly better than that of PSO, SPSO, DSMPSO, and DPSO. The seeking value of some GSA test functions is tied for first place with ISPSO, but the convergence speed of this part of the function is much lower than that of ISPSO. In the tables, the best results of the experiment are marked in bold.

Each single-peak function has only one optimal solution, which is used to evaluate the algorithm's development performance and convergence velocity. For ISPSO, the single-peak test functions F_1 to F_4 produce the theoretical optimal value, the success rate of the optimization reaches 100%, and the standard deviation is 0. This shows that the ISPSO algorithm has good robustness and stability, and the accuracy and convergence velocity improve by about 10 orders of magnitude compared with the SPSO algorithm. Compared with the performance of PSO, DPSO, and DSMPSO comparison algorithms from F_1 to F_4, ISPSO is significantly better in velocity and accuracy. For the test functions, GSA has the same mean and standard deviation as ISPSO; both are 0. However, it can be seen from the iterative comparison curves in Figur. 1(a) and Figur. 1(c) that the convergence velocity and accuracy of ISPSO are

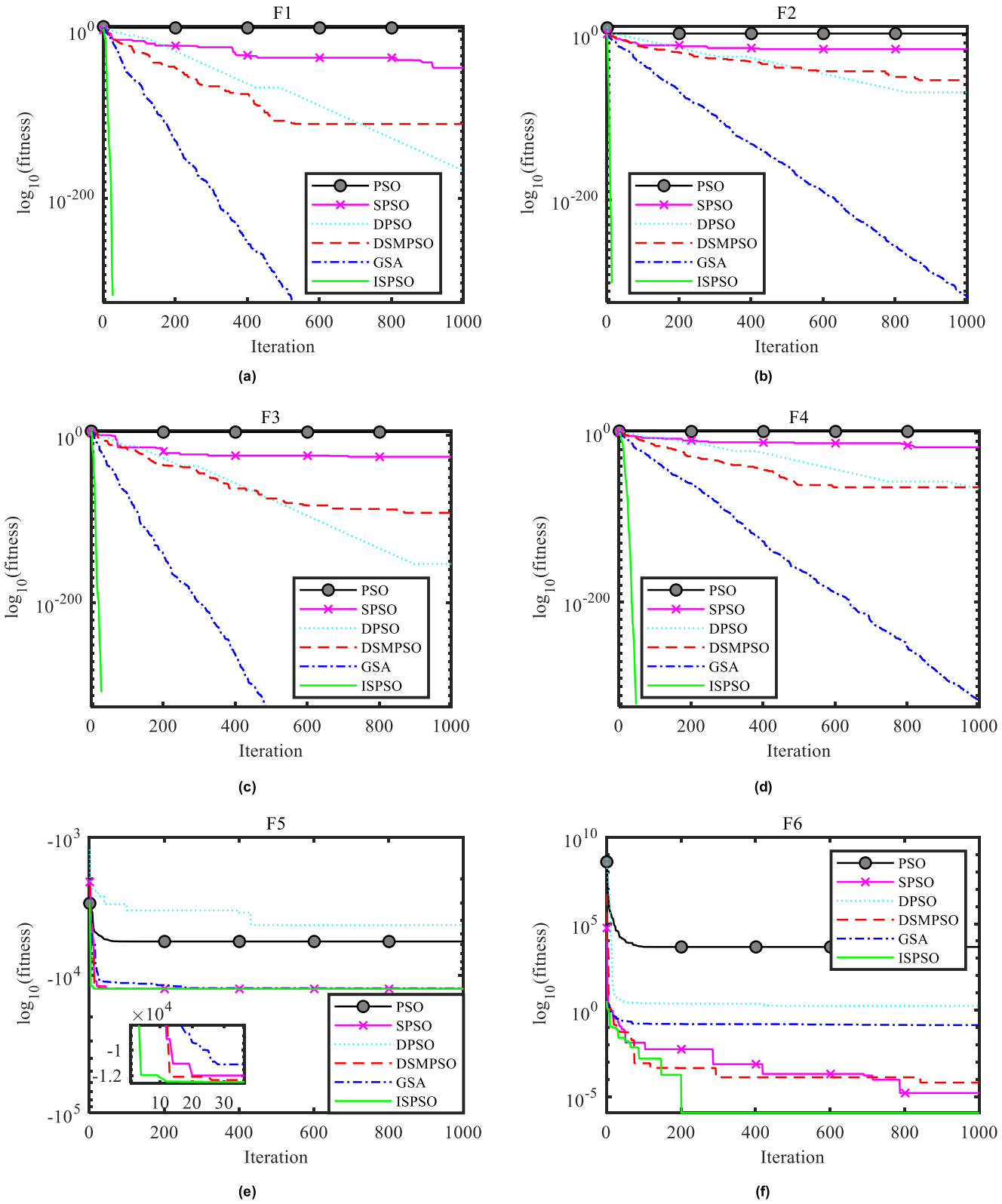
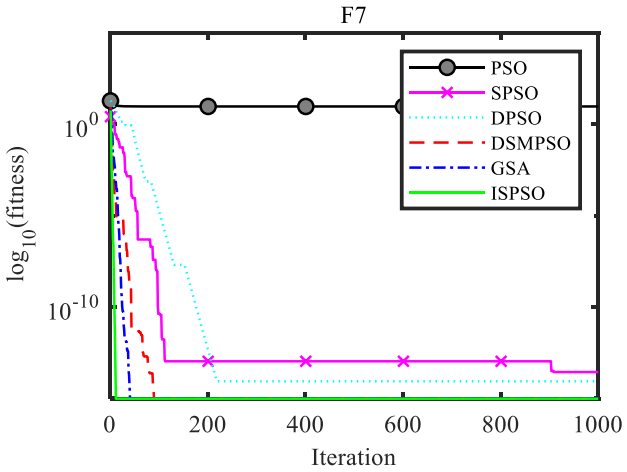


FIGURE 1. The function evolution curve.

much better than those of GSA. For the test functions F_2 and F_4, the mean value of ISPSO is slightly better than that of GSA, which corresponds to an improvement of about 3 orders of magnitude, and the number of iterations required is about 5% of the interactions for SPSO, which is a significant increase in speed. The performance of ISPSO in test



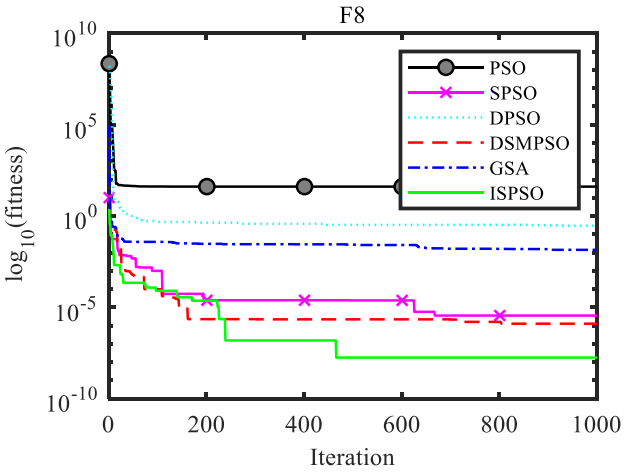


FIGURE 1. *(Continued.)* **The function evolution curve.**

TABLE 8. D-H parameters of the robot.

Joint no.	a_i / mm	α_i / (°)	d_i / mm	θ_i
1	0	90	0	θ_1
2	431.8	0	0	θ_2
3	20.3	−90	150.1	θ_3
4	0	90	431.8	θ_4
5	0	−90	0	θ_5
6	0	0	0	θ_6

TABLE 9. Cartesian space path points.

Starting point	Intermediate point 1	Intermediate point 2	Ending point
(548, −150, 300)	(419, −201, 396)	(161, −304, 589)	(32, −356,685)

functions demonstrates the good development performance of ISPSO.

The multi-peak function has only one global optimal solution and the rest are local optimal solutions, which can be used to evaluate the global search ability of the algorithm. For the multi-peak test functions F_5 to F_8, the convergence curves of ISPSO are shown in Figure 1 (e)-(h). For the test functions F_5 and F_6, the average searching value of SPSO is basically the same as ISPSO. For the test function F_7, the average seeking values of ISPSO and SPSO are 8.88E-16 and 1.69E-13, and ISPSO is improved by three orders of magnitude relative to SPSO. The average searching value is 3.49E-06 and 6.97E-07, respectively, increasing by 1 order of magnitude. Overall, ISPSO ranks first or is tied for first place in the average search value. This shows that the improved algorithm somewhat avoids the problem of falling into local optimal solutions.

3) FRIEDMAN TEST

To further illustrate the comprehensive performance of the algorithm, the Friedman test was performed and is described

TABLE 10. Joint space interpolation point.

	Position Angle /°			
	Starting point	Intermediate point 1	Intermediate point 2	Ending point
Joint 1	0	−6.85	−36.3	−60
Joint 2	−15	−4.75	22.6	36
Joint 3	0	6.1	−8.09	−30

TABLE 11. Joint interpolation results of ISPSO.

Time /s	T_{i1}	T_{i2}	T_{i3}
Joint 1	0.231699	0.268667	0.627397
Joint 2	0.334496	0.265567	0.387221
Joint 3	0.37329	0.299207	0.621277

TABLE 12. Optimal robot time of different algorithms.

Algorithm	PSO	SPSO	DPSO	DSMPSO	GSA	ISPSO
Optimal time /s	1.9395	1.6178	1.6567	1.3186	1.469	1.299894

in this section. The test results are listed in Table 6, where the algorithm test results for dimensions d = 10, 20, and 30 are given. Again, the algorithm's results are given in descending order, and the lower the result, the better.

The IMOPSO results are optimal for all dimensions of the test results, so the overall performance of IMPSO is optimal for other algorithms (Table 6).

The ISPSO algorithm generally has high convergence accuracy, good robustness, and more stable optimization performance. Based on the analysis results of the above two kinds of test functions, it can be concluded that the ISPSO algorithm has good development performance and global searchability.

However, ISPSO also increases the time complexity of the algorithm to a certain extent, increases the execution time of the algorithm, and obtains the execution time of function F_1–F_4 1,000 times under each algorithm in the case of D=30 dimensions (Table 7). It can be seen that the execution time

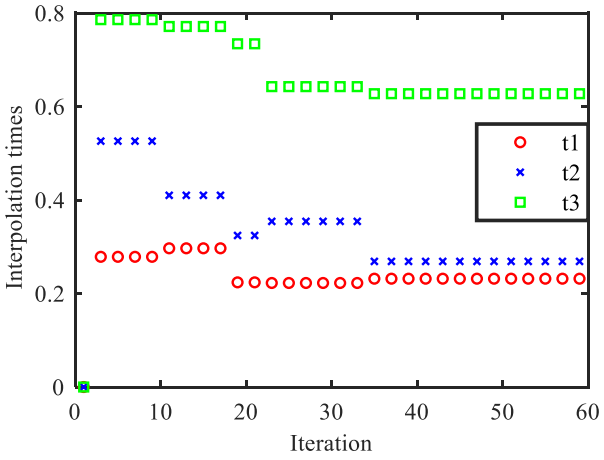


FIGURE 2. Joint 1 optimal particle iteration diagram.

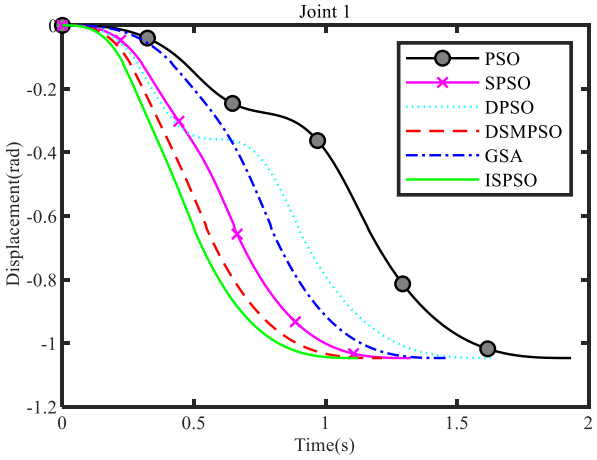


FIGURE 3. Joint 1 displacement curve.

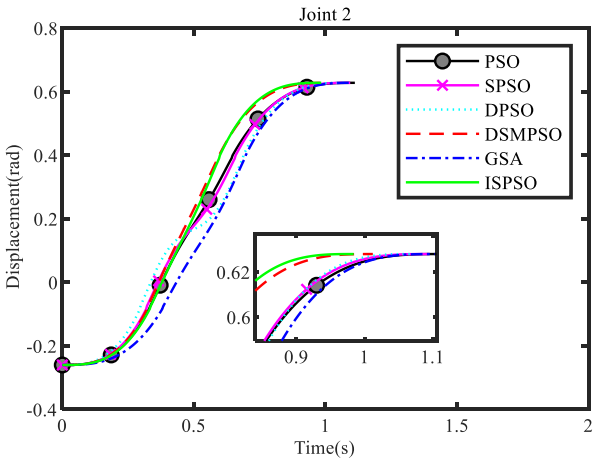


FIGURE 4. Joint 2 displacement curve.

of the algorithm has indeed increased, but compared with the improved accuracy and convergence speed of the algorithm, combined with this paper is an offline trajectory planning, this cost is considered acceptable.

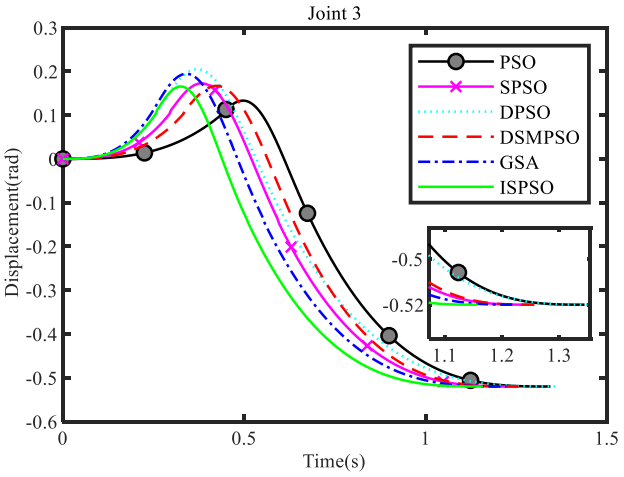


FIGURE 5. Joint 3 displacement curve.

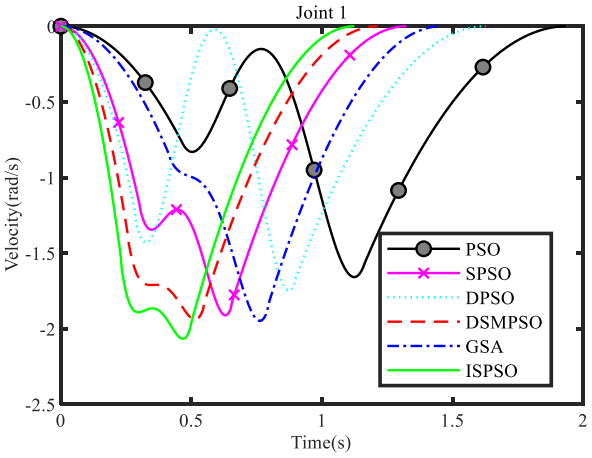


FIGURE 6. Joint 1 velocity curve.

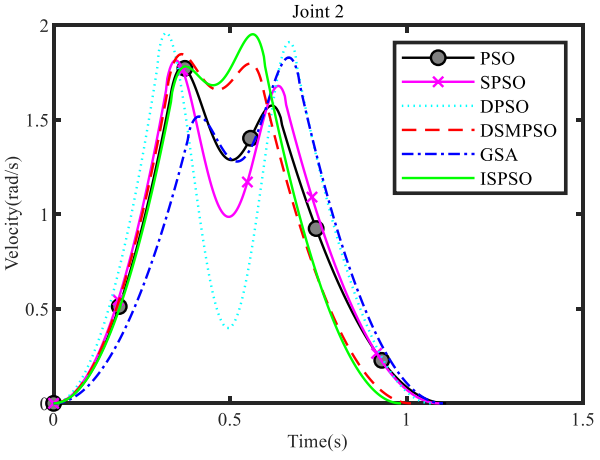


FIGURE 7. Joint 2 velocity curve.

B. TIME-OPTIMAL TRAJECTORY PLANNING BASED ON ISPSO

Combined with the 3-5-3 polynomial trajectory interpolation proposed in Section II. and the improved algorithm mentioned above, the trajectory interpolation model is introduced

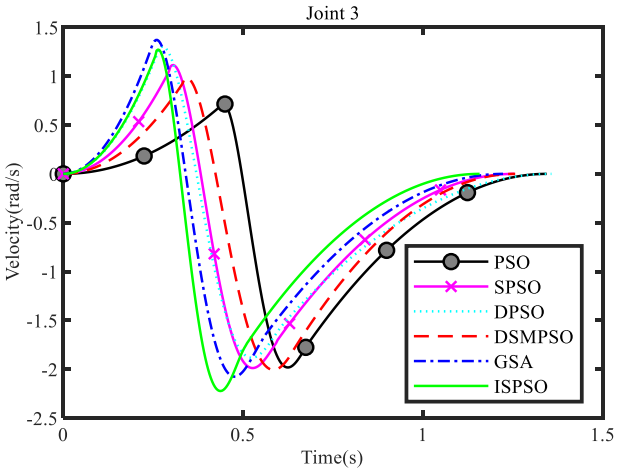


FIGURE 8. Joint 3 velocity curve.

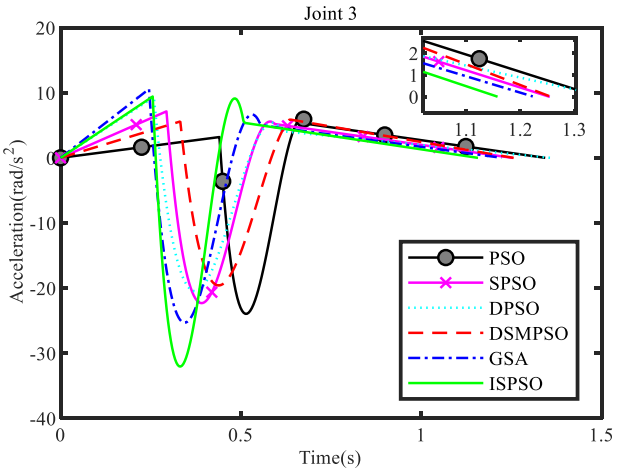


FIGURE 11. Joint 3 acceleration curve.

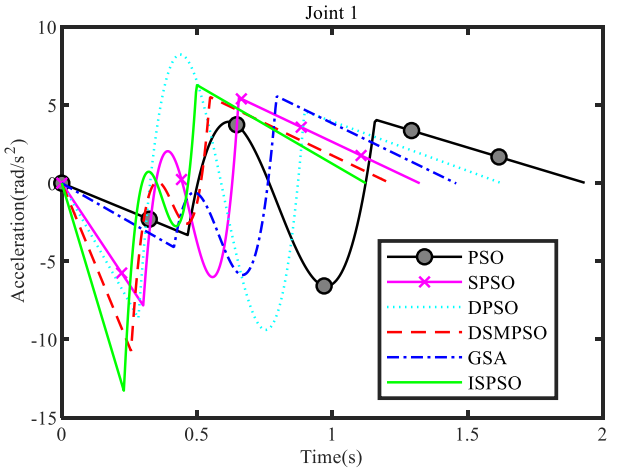


FIGURE 9. Joint 1 acceleration curve.

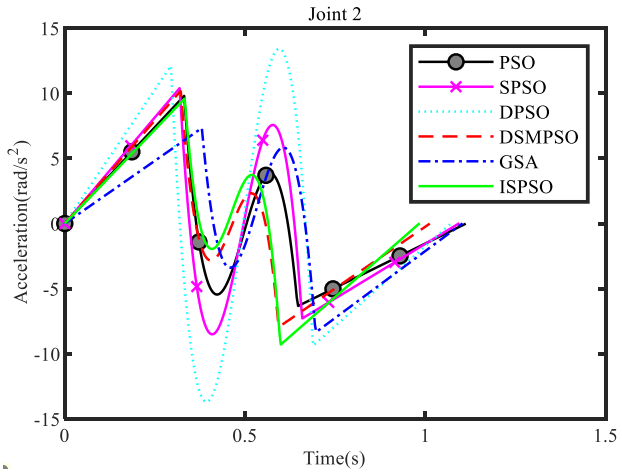


FIGURE 10. Joint 2 acceleration curve.

to the simulation verification algorithm. The key formula of the algorithm finds three interpolation times and optimal values. The time t and matrix a of equation (8) are variables in the text. If matrix a is used as the independent variable, then the dimension of the algorithm is 14, and the algorithm is computationally large and complex, so this article directly searches the time and reduces the dimension to 3 dimensions to reduce the amount of calculation. The algorithm steps were as follows:

Step1: Population produces three-dimensional particle variables t_1, t_2 and t_3.

Step2: Substitution of the time variable combination into equation (6) - equation (8) to solve the matrix a.

Step3: Substitution of the coefficient of matrix a into the 3-5-3 polynomial, and at the same time, determination of whether it meets the speed constraint.

Step4: Introduction of the speed constraint, update: If the speed constraint is not satisfied, this paper selects a large fitness value, and the algorithm will replace it during iteration.

Step5: Used until the end of the algorithm.

Simulation experiments were conducted on the first three joints of the PUMA 560 robot to verify the performance of the proposed algorithm. The D-H parameters of the manipulator are shown in Tabl. 8. The Matlab Robotics Toolbox was used to model the robot [41], [42]. PSO, SPSO, GSA, DPSO, DSMPSO, and ISPSO were selected for trajectory planning simulation. For all algorithms, the swarm size was set to 20, the number of iterations was set to 50, the particle position was constrained at [0.1, 4], and the maximum joint angular velocity was set to 115°/s (2.0071 rad/s). The specific parameter settings for each algorithm are shown in Table 2.

In the Cartesian coordinate system, the position of the robot's starting point, intermediate points, and ending point are given (Table 9).

Through the inverse kinematics of the robot, the starting point, the intermediate point 1, the intermediate point 2, and the ending point in Cartesian space were transformed into the angle values in joint space (Tabl. 10).

Under the velocity constraint, the three interpolation times of joint 1 were solved by ISPSO and the optimal interpolation time g_{best} in each iteration was recorded to obtain the interpolation time g_{best} evolution of joint 1 (Figure 2). It can be

observed that the optimal particle of joint 1 converges rapidly after about 40 iterations at most. The time required for joint 1 to run the three-segment interpolation is denoted as.

The above method was also used to optimize other joints to obtain the optimal operation time of each joint under the velocity constraint (Table 11).

Since each joint of the robot moves at the same time, the maximum interpolation time of each joint segment was selected to ensure the arrival of each joint at the same time. $T_1 = \max\{T_{i1}\}$, $T_2 = \max\{T_{i2}\}$, $T_3 = \max\{T_{i3}\}$, $(i = 1, 2, 3)$. Then $T_1 = 0.37329s$, $T_2 = 0.299207s$, $T_3 = 0.627397s$, the total running time of the robot based on ISPSO was $T = T_1 + T_2 + T_3 = 1.299894s$. Similarly, PSO, SPSO, DPSO, DSMPSO, and GSA were used to solve the trajectory interpolation time of the robot joints. The optimal running time obtained by each algorithm is shown in Tabl. 12.

The ISPSO trajectory planning method can shorten the running time of the robot and improve the working efficiency (Table 12). Compared with SPSO and GSA, the ISPSO algorithm increased by 19.6% and 11.5%, respectively, and the time decreased from 1.617s and 1.469s to 1.299s. PSO and the two-particle swarm improvement algorithms DPSO and DSMPSO trajectory planning times were 1.93395s, 1.6567s, and 1.3086s, respectively, while that of ISPSO was 1.299s. ISPSO improved about 33%, 14.1%, and 1.15% over PSO, DPSO, and DSMPSO, respectively.

Figures 3 to Figure 11 show the displacement, velocity, and acceleration curves of each robot joint obtained by various algorithms. The displacement, velocity, and acceleration of each joint are smooth and without mutation. From the comparison of the diagrams of the joint velocity in Figures 6 and 8, it can be concluded that ISPSO satisfies the velocity constraint conditions, and the optimized joint velocity tends to the ultimate velocity but does not exceed it. Compared with the other five algorithms, ISPSO can run at the maximum velocity for longer without exceeding the limit velocity in joints 1 and 2. At the same time, it can be seen from the comparison of the acceleration of each joint in Figures 9,10, and 11 that the acceleration at the starting and ending joint is 0. Continuous acceleration during movement not only ensured the stability of the joint movement but also reduced the vibration of the joint.

In summary, according to the analysis of the above simulation results, ISPSO trajectory planning can effectively shorten the running time of the robot and improve work efficiency while ensuring the robot's safety.

V. CONCLUSION

Aiming at the optimal-time trajectory planning problem of robot joint space, a 3-5-3 polynomial interpolation curve model was established and ISPSO was proposed to solve the problem. Furthermore, the following conclusions were obtained through testing and verification.

(1) ISPSO is proposed as an improvement of SPSO by introducing random inertia weights and GSA. PSO, SPSO, GSA, DPSO, DSMPSO, and ISPSO were applied to eight groups of test functions (including single-peak functions and multi-peak functions). In the four groups of single-peak functions from F_1-F_4, ISPSO converged to the theoretical optimal value, and the success rate of the search was 100%. In four groups of multi-peak functions from F_5-F_8, the average search value and standard deviation of ISPSO were ranked first in the three groups and tied for first in one group. The results obtained by the test function demonstrated that ISPSO improved the convergence accuracy and efficiency of the algorithm and solved the problem of low convergence accuracy and slow speed.

(2) The ISPSO was applied to joint space trajectory planning and compared with the optimal-time trajectory planned by the above five algorithms. Compared with SPSO and GSA, the simulation result time of ISPSO was reduced by about 19.6% and 11.5%, respectively, and also improved compared to some improved particle swarm algorithms. It is verified that the algorithm could make the robot's trajectory smooth and optimal, and that the displacement, velocity, and acceleration curves had no mutation and met the kinematic constraints. The joint optimized by ISPSO could operate at a maximum velocity for a long time without exceeding the limit velocity. It was proved that the algorithm has strong practicability in time-optimal trajectory planning and can improve the working efficiency of the robot.

In this paper, the algorithm of time optimal trajectory is studied and simulated under off-line condition. The next step of this paper is to improve the real-time performance of the algorithm in trajectory planning, and the research direction is real-time online trajectory planning algorithm research.

REFERENCES

[1] F. J. Abu-Dakka, F. Rubio, F. Valero, and V. Mata, "Evolutionary indirect approach to solving trajectory planning problem for industrial robots operating in workspaces with obstacles," *Eur. J. Mech. A/Solids*, vol. 42, pp. 210–218, Nov. 2013.

[2] L. Tang, X. Tang, X. Jiang, and C. Gosselin, "Dynamic trajectory planning study of planar two-DOF redundantly actuated cable-suspended parallel robots," *Mechatronics*, vol. 30, pp. 187–197, Sep. 2015.

[3] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Trajectory planning in robotics," *Math. Comput. Sci.*, vol. 6, no. 3, pp. 269–279, 2012.

[4] F. J. Abu-Dakka, I. F. Assad, R. M. Alkhdour, and M. Abderahim, "Statistical evaluation of an evolutionary algorithm for minimum time trajectory planning problem for industrial robots," *Int. J. Adv. Manuf. Technol.*, vol. 89, nos. 1–4, pp. 389–406, Mar. 2017.

[5] J. Kim and W. K. Chung, "Real-time zero moment point compensation method using null motion for mobile manipulators," *Adv. Robot.*, vol. 20, no. 5, pp. 581–593, Jan. 2006.

[6] L. Cui, H. Wang, and W. Chen, "Trajectory planning of a spatial flexible manipulator for vibration suppression," *Robot. Auto. Syst.*, vol. 123, Jan. 2020, Art. no. 103316.

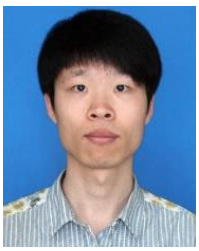
[7] A. Abe, "An effective trajectory planning method for simultaneously suppressing residual vibration and energy consumption of flexible structures," *Case Stud. Mech. Syst. Signal Process.*, vol. 4, pp. 19–27, Dec. 2016.

[8] J. Huang, P. Hu, K. Wu, and M. Zeng, "Optimal time-jerk trajectory planning for industrial robots," *Mechanism Mach. Theory*, vol. 121, pp. 530–544, Mar. 2018.

[9] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: A general overview," *Motion Operation Planning Robotic Syst.*, vol. 29, pp. 3–27, Jan. 2015.

[10] A. Gasparetto and V. Zanotto, "Optimal trajectory planning for industrial robots," *Adv. Eng. Softw.*, vol. 41, no. 4, pp. 548–556, Apr. 2010.

[11] B. Su and L. Zou, "Manipulator trajectory planning based on the algebraic-trigonometric Hermite blended interpolation spline," *Proc. Eng.*, vol. 29, pp. 2093–2097, Jan. 2012.
[12] D. Sidobre and K. Desormeaux, "Smooth cubic polynomial trajectories for human-robot interactions," *J. Intell. Robotic Syst.*, vol. 95, nos. 3–4, pp. 851–869, Sep. 2019.
[13] H. Liu, X. Lai, and W. Wu, "Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematic constraints," *Robot. Comput.-Integr. Manuf.*, vol. 29, no. 2, pp. 309–317, Apr. 2013.
[14] X. Wang, A. Wang, D. Wang, W. Wang, B. Liang, and Y. Qi, "Repetitive control scheme of robotic manipulators based on improved B-spline function," *Complexity*, vol. 2021, pp. 1–15, Apr. 2021.
[15] L. Zhang, Y. Wang, X. Zhao, P. Zhao, and L. He, "Time-optimal trajectory planning of serial manipulator based on adaptive cuckoo search algorithm," *J. Mech. Sci. Technol.*, vol. 35, no. 7, pp. 3171–3181, Jul. 2021.
[16] Y. Huang and M. Fei, "Motion planning of robot manipulator based on improved NSGA-II," *Int. J. Control, Autom. Syst.*, vol. 16, no. 4, pp. 1878–1886, Aug. 2018.
[17] W. Li, G.-G. Wang, and A. H. Gandomi, "A survey of learning-based intelligent optimization algorithms," *Arch. Comput. Methods Eng.*, vol. 28, no. 5, pp. 3781–3799, Aug. 2021.
[18] S. Forrest, "Genetic algorithms," *ACM Comput. Surv.*, vol. 28, no. 1, pp. 77–80, 1996.
[19] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. ICNN Int. Conf. Neural Netw.*, vol. 4, Nov. 1995, pp. 1942–1948.
[20] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Comput. Intell. Mag.*, vol. 1, no. 4, pp. 28–39, Nov. 2006.
[21] M. Pluhacek, R. Senkerik, A. Viktorin, T. Kadavy, and I. Zelinka, "A review of real-world applications of particle swarm optimization algorithm," in *AETA 2017 Recent Advances in Electrical Engineering and Related Sciences: Theory and Application* (Lecture Notes in Electrical Engineering), vol. 465, V. H. Duy, T. T. Dao, I. Zelinka, S. B. Kim, and T. T. Phuong, Eds. Cham, Switzerland: Springer, 2018, pp. 115–122.
[22] N. K. Jain, U. Nangia, and J. Jain, "A review of particle swarm optimization," *J. Inst. Eng. India Ser. B*, vol. 99, no. 4, pp. 407–411, Aug. 2018.
[23] I. Mazhoud, K. Hadj-Hamou, J. Bigeon, and P. Joyeux, "Particle swarm optimization for solving engineering problems: A new constraint-handling mechanism," *Eng. Appl. Artif. Intell.*, vol. 26, no. 4, pp. 1263–1273, Apr. 2013.
[24] H. Zhang and F. Xing, "Fuzzy-multi-objective particle swarm optimization for time–cost–quality tradeoff in construction," *Autom. Construction*, vol. 19, no. 8, pp. 1067–1075, Dec. 2010.
[25] F. Wang, H. Zhang, and A. Zhou, "A particle swarm optimization algorithm for mixed-variable optimization problems," *Swarm Evol. Comput.*, vol. 60, Feb. 2021, Art. no. 100808.
[26] B. Wei, X. Xia, F. Yu, Y. Zhang, X. Xu, H. Wu, L. Gui, and G. He, "Multiple adaptive strategies based particle swarm optimization algorithm," *Swarm Evol. Comput.*, vol. 57, Sep. 2020, Art. no. 100731.
[27] X. Xia, L. Gui, and Z.-H. Zhan, "A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting," *Appl. Soft Comput.*, vol. 67, pp. 126–140, Jun. 2018.
[28] W. Wang, Q. Tao, Y. Cao, X. Wang, and X. Zhang, "Robot time-optimal trajectory planning based on improved cuckoo search algorithm," *IEEE Access*, vol. 8, pp. 86923–86933, 2020.
[29] A. K. Sadhu, A. Konar, T. Bhattacharjee, and S. Das, "Synergism of firefly algorithm and Q-learning for robot arm path planning," *Swarm Evol. Comput.*, vol. 43, pp. 50–68, Dec. 2018.
[30] H. Wang, H. Wang, J. Huang, B. Zhao, and L. Quan, "Smooth point-to-point trajectory planning for industrial robots with kinematical constraints based on high-order polynomial curve," *Mechanism Mach. Theory*, vol. 139, pp. 284–293, Sep. 2019.
[31] L. Wang, Q. Wu, F. Lin, S. Li, and D. Chen, "A new trajectory-planning beetle swarm optimization algorithm for trajectory planning of robot manipulators," *IEEE Access*, vol. 7, pp. 154331–154345, 2019.
[32] J. Zhao, X. Zhu, and T. Song, "Serial manipulator time-jerk optimal trajectory planning based on hybrid IWOA-PSO algorithm," *IEEE Access*, vol. 10, pp. 6592–6604, 2022.
[33] X. Xiang-Rong, W. Xiao-Gang, and Q. Feng, "Trajectory planning of robot manipulators by using spline function approach," in *Proc. 3rd World Congr. Intell. Control Autom.*, Jun. 2000, pp. 1215–1219.
[34] W. Hu and Z. S. Li, "A simpler and more effective particle swarm optimization algorithm," *J. Softw.*, vol. 18, no. 4, pp. 861–868, 2007.
[35] K. R. Harrison, A. P. Engelbrecht, and B. M. Ombuki-Berman, "Inertia weight control strategies for particle swarm optimization: Too much momentum, not enough analysis," *Swarm Intell.*, vol. 10, no. 4, pp. 267–305, Dec. 2016.
[36] E. Tanyildizi and G. Demir, "Golden sine algorithm: A novel math-inspired algorithm," *Adv. Electr. Comput. Eng.*, vol. 17, no. 2, pp. 71–78, 2017.
[37] S. Han, X. Shan, J. Fu, W. Xu, and H. Mi, "Industrial robot trajectory planning based on improved PSO algorithm," *J. Phys., Conf. Ser.*, vol. 1820, no. 1, Mar. 2021, Art. no. 012185.
[38] L. Shen, X. Huang, and C. Fan, "Double-group particle swarm optimization and its application in remote sensing image segmentation," *Sensors*, vol. 18, no. 5, p. 1393, May 2018.
[39] Y. Huang, H. Lu, K. Xu, and W. Shen, "Simplified mean particle swarm optimization algorithm with dynamic adjustment of inertia weight," *J. Chin. Comput. Syst.*, vol. 39, no. 12, pp. 2590–2595, 2018.
[40] G. I. Sayed, G. Khoriba, and M. H. Haggag, "A novel chaotic SALP swarm algorithm for global optimization and feature selection," *Int. J. Speech Technol.*, vol. 48, no. 10, pp. 3462–3481, Oct. 2018.
[41] P. I. Corke, "A robotics toolbox for MATLAB," *IEEE Robot. Autom. Mag.*, vol. 3, no. 1, pp. 24–32, Mar. 1996.
[42] J. Xiao, W. Han, and A. Wang, "Simulation research of a six degrees of freedom manipulator kinematics based on MATLAB toolbox," in *Proc. Int. Conf. Adv. Mech. Syst. (ICAMechS)*, Dec. 2017, pp. 376–380.



XIAO HU was born in Puyang, Henan, China, in 1986. He received the B.S. and M.S. degrees in control theory and control engineering and the Ph.D. degree in materials processing engineering from Northeastern University, in 2009 and 2015.

From 2015 to 2017, he was a Lecturer with the Taiyuan University of Science and Technology. Since 2018, he has been an Associate Professor with the School of Electronic Information Engineering, Taiyuan University of Science and Technology. He is the author of more than 20 articles and more than five inventions. His research interests include intelligent equipment and control system research and application.

HENG WU was born in Changzhi, Shanxi, China, in 1998. He received the bachelor's degree from the Taiyuan University of Science and Technology, in 2016, where he is currently pursuing the master's degree.

His research interest includes industrial robot trajectory planning.

QIANLAI SUN received the B.S., M.S., and Ph.D. degrees in automation from the Taiyuan University of Science and Technology, in 2000, 2003, and 2018, respectively. Since 2018, he has been an Associate Professor with the Taiyuan University of Science and Technology.

JUN LIU was born in Luohe, Henan, China, in 1985. He received the master's degree in mechanical design and theory from the University of Science and Technology Beijing.

He is currently the Deputy Director of the Technology Center and the President of the Innovation Research Institute, CITIC Heavy Industry Machinery Company Ltd., Luoyang. He has published 12 articles and has obtained six invention patents and 21 utility model patents. His research interests include big data, the Internet of Things, and artificial intelligence.