International Journal for Housing Science and Its Applications
Publish August 3, 2025. Volume 46, Issue 3 Pages 377-393

https://doi.org/10.70517/ijhsa46327

Optimal trajectory planning method for robotic arm based on improved SMA

Min Sun[1,*] and Jingjing Hu[2]
[1] School of Mechanical and Electrical Engineering, Hubei Light Industry Technology Institute, Wuhan, Hubei, 430000, China
[2] School of Automotive Technology and Service, Wuhan City Polytechnic, Wuhan, Hubei, 430000, China
Corresponding authors: (e-mail: gguxingzi@163.com).

Abstract Industrial robotic arms have flexible maneuverability and have been widely popularized in various industries, therefore, the study of mobile robotic arm motion planning and control system has very important theoretical and practical value. In this paper, an optimal time-impact trajectory planning method for the robotic arm is proposed, which adopts five times non-uniform B-spline function to construct interpolation curves in the joint space, replaces the motion constraints of the robotic arm with the constraints of the control vertices of the B-spline curves of each order, and uses multi-strategy improved viscous bacterial algorithm to optimize the objective function. Simulation and experimental results show that the proposed improved mucilage algorithm can effectively improve the performance of the SMA algorithm, and the optimal trajectory planner is able to obtain a safe and smooth time-impact optimal trajectory under the premise of satisfying the joint constraints with the running time less than 15s and the impacts in the range of [10-7 rad, 10-3 rad]. This study ensures that smaller shocks are generated during the motion process, which makes the motion control performance of the robotic arm improved.

Index Terms robotic arm, 5 times B-spline curve, improved SMA algorithm, optimal trajectory planning

I. Introduction

In recent years, with the continuous development of automation, the application of robotic arms has become more and more widespread [1]. The robotic arm is a highly integrated space mechanical system with the integration of machine, electricity, heat and control [2]. With the development of science and technology, especially the birth of aviation aircraft, robots and other intelligent machinery, so that it has been widely used, the robotic arm as in the trajectory of the support, service, etc. to the attention of people [3]-[5]. And the trajectory planning of the robotic arm is the key link in its work [6].

The trajectory planning of the robotic arm refers to determining the movement trajectory of the robotic arm so that it can accomplish the predetermined tasks in a specific environment [7], [8]. The trajectory planning algorithm, mainly includes two aspects of path planning and velocity planning. Path planning refers to determining the path for the movement of the robotic arm, while velocity planning refers to determining the movement speed of the robotic arm on a specified path [9]-[11]. Reasonable robotic arm trajectory planning algorithms can enable robotic arms to complete work tasks efficiently and improve work efficiency, which are widely used in industrial, medical, and service fields [12], [13]. In industry, robotic arms can complete various complex work tasks such as assembly, handling and welding on production lines, and the application of robotic arm trajectory planning algorithms can help robotic arms accurately and efficiently complete various tasks and improve productivity [14]-[17]. In the medical field, robotic arms can assist doctors in surgical operations, reduce surgical risks and improve surgical accuracy [18], [19]. The application of robotic arm trajectory planning algorithms can enable the robotic arm to realize precise movement trajectories during the surgical process to ensure the safety and success of surgery [20], [21]. And in the service field, the robotic arm can accomplish tasks such as carrying and cleaning of dinner plates [22]. The application of robotic arm trajectory planning algorithms can enable the robotic arm to move flexibly in a narrow space to accomplish various service tasks and provide a better service experience [23], [24].

Literature [25] proposed a trajectory planning method based on the improved dynamic multiple swarm particle swarm optimization algorithm, verified the effectiveness of the algorithm through simulation results, and revealed that the improved dynamic multiple swarm particle swarm optimization algorithm effectively improves the work efficiency and convergence speed. Literature [26] discusses the six-axis robotic arm trajectory planning problem based on deep reinforcement learning, proposes a multi-objective optimization method based on deep reinforcement learning and optimal planning motivation, and illustrates the effectiveness of the proposed method through simulation experiments. Literature [27] describes the applications, roles and limitations of robotic arms, and

emphasizes the importance of trajectory planning for robotic arms, revealing the effectiveness of the method in robotic arm trajectory planning by means of a 6-degree-of-freedom robotic arm, as well as a trajectory planning method with a MATLAB program and particle swarm optimization. Literature [28] based on the traditional robotic arm trajectory planning method with low adaptability and other problems, proposed a method to plan the trajectory of robotic arm by using the trajectory learning and generalization properties of dynamic motion primitive, verified the feasibility of the method in robotic arm trajectory planning, and improved the adaptability and generalization performance of robotic arm trajectory planning. Literature [29] combined the Non-Superiority Sorting Genetic Algorithm-II (NSGA-II) and Achievement Scalarization Function (ASF) to propose a method for solving the joint trajectory of minimum hourly jumping moment of a six-axis industrial robot based on hybrid multi-objective optimization technique to obtain higher positioning accuracy, and the method obtained good optimization results. Literature [30] introduced a particle swarm optimization algorithm for robot trajectory planning to optimize the joint angles or paths of robotic arm motion, and improved it by introducing an adaptive weight strategy and a random perturbation term, which revealed that the algorithm improves the acceleration of the robotic arm and outperforms the traditional planning methods. Literature [31] introduced an optimal trajectory planning method for industrial robots, emphasizing the application of path tracking, and verified that the method can effectively solve the optimal trajectory of the path tracking problem, which is characterized by wide adaptability and high feasibility. Literature [32] used a polynomial curve-based trajectory design method to analyze the motion trajectory planning problem of a robotic arm, and showed by comparison that the trajectory optimization method is suitable for real-time trajectory planning. Literature [33] outlined the motion model and trajectory planning method of the robotic arm, and used genetic algorithm-improved particle swarm algorithm (PSO) to optimize the motion trajectory of the robotic arm, and simulation experiments pointed out that the improved PSO algorithm has fast convergence speed, and the lowest degree of adaptability after stable convergence. Literature [34] proposed a solution method to minimize the cost of moving the robotic arm along a specified path under input torque/force constraints based on the consideration of coupled nonlinear dynamics of the robotic arm, using dynamic programming to find the position, velocity, acceleration and torque that minimize the cost, and simulation experiments revealed the effectiveness of the method. Literature [35] explored an improved adaptive multi-objective particle swarm optimization method for time- and collision-based optimal trajectory planning of a multi-degree-of-freedom robotic arm, showing that the optimal trajectory planning method for the manipulator improves the efficiency of the movement of the manipulator, the tracking accuracy, as well as the operational efficiency and stability of the manipulator. Literature [36] proposed an optimal time trajectory planning method for robotic arm based on the improved tuna swarm optimization (TSO) algorithm, which is optimized on the basis of the standard TSO algorithm. Literature [37] introduces the role of space robotic arm, which can effectively replace humans to complete various on-orbit tasks, and describes the current research status of space obstacle avoidance trajectory planning and motion trajectory planning, and examines the basic principles and practical applications of space robotic arm trajectory planning methods. The above research emphasizes the importance of robotic arm trajectory planning and proposes methods for robotic arm trajectory optimization based on dynamic multiple swarm particle swarms, deep reinforcement learning, genetic algorithms, and improved TSOs, revealing that these methods effectively improve the efficiency and tracking accuracy of the robotic arm.

Aiming at the efficiency of industrial robotic arm execution as well as the vibration and mechanical wear generated by the impact during the motion process, this paper proposes an optimal time-impact trajectory planning method for robotic arm to optimize two coupled and contradictory motion performance indexes, namely, the robotic arm joint motion time and the added acceleration (impact). The motion trajectory executed by the robotic arm is converted into a position-time sequence in the joint space through inverse kinematics, the interpolation curve is constructed in the joint space using a 5-times inhomogeneous B-spline function and replaces the motion constraints of the robotic arm, and an improved SMA algorithm is proposed for solving. The solution process is as follows: firstly, the better population is sought as the initial population through the Tent mapping inverse learning strategy to improve the convergence speed of the algorithm, and secondly, the viscous bacteria are optimized by updating the position through the adaptive weights strategy and the perturbation strategy, adjusting the algorithm exploration ability and exploitation ability, so as to optimize the objective function. Finally, the improved SMA is compared with GWO and two other state-of-the-art metaheuristic algorithms on four benchmark functions, and the improved SMA algorithm is used to solve the time-impact optimal trajectory planning problem for a redundant seven-degree-of-freedom robotic arm.

II. Optimal trajectory planning for robotic arm with improved SMA

II. A.Kinematic modeling of the robotic arm

The kinematics of the robotic arm includes forward kinematics and inverse kinematics [38], and the forward and inverse kinematics of the robotic arm are illustrated in Fig. 1.FK refers to the computation of the position and attitude of the end-effector of the robotic arm based on the angle of each joint of the robotic arm, while IK refers to the computation of the corresponding joint angle based on the position and attitude of the end-effector of the robotic arm.

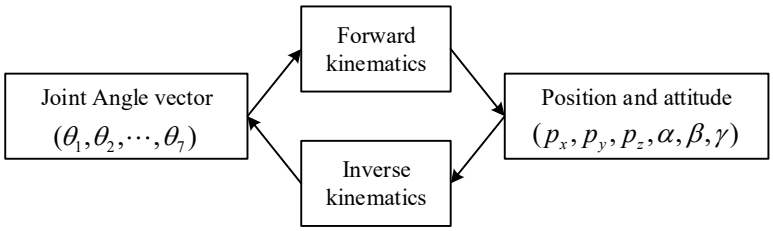


Figure 1: Kinematics of the manipulator

Before studying the inverse kinematics of a robotic arm, an FK model is required. The DH parameters can determine the structure of the robotic arm and are widely used to model the kinematics of robotic arms. Table 1 shows the DH parameters of the studied robotic arm, where $a_i,\alpha_i,d_i,\theta_i$ denote the linkage length, linkage torsion angle, linkage offset, and the range of joint angles, respectively.

Table 1: DH parameters of the manipulator

Arthrosis	a_i (m)	α_i (rad)	d_i (m)	θ_i (rad)
1	0.00	-π/2	d1=0.5	-π<θ1<π
2	l2=0.20	π/2	0	-π/2<θ2<π/6
3	L3=0.25	-π/2	0	-π/3<θ3<2π/3
4	L4=0.30	π/2	0	-π/2<θ4<π/2
5	L5=0.20	-π/2	0	-π/2<θ5<π/2
6	L6=0.20	0	0	-π/2<θ6<π/2
7	L7=0.10	0	d7=0.05	-π/6<θ7<π/2

The standard DH parameter method is used to model the FK of the robotic arm, and the chi-square transformation matrix of a single joint is:

$$ {}^{i}_{i-1}T = \begin{bmatrix} c\theta_i & -s\theta_i \cdot c\alpha_i & s\theta_i \cdot s\alpha_i & a_i \cdot c\theta_i \\ s\theta_i & c\theta_i \cdot c\alpha_i & -c\theta_i \cdot s\alpha_i & a_i \cdot s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1} $$

where ${}^{i}_{i-1}T$ is the chi-square transformation matrix from joint $i-1$ to i, and $s\theta_i$ and $o\theta_i$ denote $\sin(\theta_i)$ and $\cos(\theta_i)$, respectively.

The chi-square transformation matrix for each joint can be obtained by substituting each row of data in Table 1 into Eq. (1):

$$
\begin{aligned}
{}^{1}_{0}T &= \begin{bmatrix} c\theta_1 & 0 & -s\theta_1 & 0 \\ s\theta_1 & 0 & c\theta_1 & 0 \\ 0 & -1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix};\ {}^{2}_{1}T = \begin{bmatrix} c\theta_2 & 0 & s\theta_2 & l_2c\theta_2 \\ s\theta_2 & 0 & -c\theta_2 & l_2s\theta_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \\
{}^{3}_{2}T &= \begin{bmatrix} c\theta_3 & 0 & -s\theta_3 & l_3c\theta_3 \\ s\theta_3 & 0 & c\theta_3 & l_3s\theta_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix};\ {}^{4}_{3}T = \begin{bmatrix} c\theta_4 & 0 & s\theta_4 & l_4c\theta_4 \\ s\theta_4 & 0 & -c\theta_4 & l_4s\theta_4 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \\
{}^{5}_{4}T &= \begin{bmatrix} c\theta_5 & 0 & -s\theta_5 & l_5c\theta_5 \\ s\theta_5 & 0 & c\theta_5 & l_5s\theta_5 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix};\ {}^{6}_{5}T = \begin{bmatrix} c\theta_6 & -s\theta_6 & 0 & l_6c\theta_6 \\ s\theta_6 & c\theta_6 & 0 & l_6s\theta_6 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \\
{}^{7}_{6}T &= \begin{bmatrix} c\theta_7 & -s\theta_7 & 0 & l_7c\theta_7 \\ s\theta_7 & c\theta_7 & 0 & l_7s\theta_7 \\ 0 & 0 & 1 & d_7 \\ 0 & 0 & 0 & 1 \end{bmatrix}.
\end{aligned} \tag{2}
$$

The FK equation for the end-effector with respect to the base (base coordinate system) is generated by multiplying all the chi-square transformation matrices sequentially as shown in Equation (3):

$$
T_{End-Effect} = {}^{7}_{0}T = {}^{1}_{0}T \cdot {}^{2}_{1}T \cdot {}^{3}_{2}T \cdot {}^{4}_{3}T \cdot {}^{5}_{4}T \cdot {}^{6}_{5}T \cdot {}^{7}_{6}T \tag{3}
$$

where $T_{End-Effector}$ denotes the chi-square transformation matrix of the end-effector with respect to the base coordinate system.

When the values of a set of joint variables are given in Eq. (3), the alternative representation of $T_{End-Effector}$ can be written as:

$$
T_{End-Eifector} = \begin{bmatrix} \vec{n} & \vec{s} & \vec{a} & \vec{p} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}
$$

where $(p_x, p_y, p_z)^T$ denotes the position element of the end-effector in the base coordinate system, and $(\vec{n}, \vec{s}, \vec{a})$ denotes the attitude element.

Although the attitude matrix $(\vec{n}, \vec{s}, \vec{a})$ has nine elements, it has only three degrees of freedom, and it is a unitary orthogonal matrix with redundancy. Therefore, the Euler angles can be used to describe the attitude of the end-effector, calculated as:

$$
(\alpha, \beta, \gamma) = (\arctan(-a_y / a_z), \arctan(a_x / \sqrt{n_x^2 + s_x^2}), \arctan(-s_x / n_x)) \tag{5}
$$

Therefore, the position and pose of the end-effector can be expressed as $P = (p_x, p_y, p_z, \alpha, \beta, \gamma)$, where (p_x, p_y, p_z) is the position vector and (α, β, γ) is the pose vector expressed by the Euler angles. In summary, the kinematic model of the seven-degree-of-freedom robotic arm can be obtained by substituting Eqs. (2)~(5) into the simplification as:

$$
\begin{aligned}
&p_s = d_7(s_3 s_j - c_3 d) - l_7 s_7 o + n(l_6 + l_7 c_7) - l_3 i - l_4 l - l_3 t + l_2 c_{12} \\
&p_y = -d_7(s_3 s_j - c_5 d) - l_7 s_7 (c_6 k + s_6 p) - (l_6 + l_7 c_7)(s_6 k - c_6 p) + l_5 p + l_4 j + l_3 c + l_2 c_2 s_1 \\
&p_c = d_7 l + l_7 s_7 (s_6 m - c_6 f) - (l_6 + l_6 c_6 f) - l_3 m - l_4 e - l_3 c_3 s_2 - l_2 s_2 + l_1 \\
&\alpha = \arctan((s_3 j - c_6 d)/l) \\
&\beta = \arctan((s_3 h - c_3 b)/((c_7 n + s_7 o)^2 + (c_7 o - s_7 n)^2)^{0.5}) \\
&\gamma = \arctan((s_7 n - c_7 o)/(c_7 n + s_7 o)) \\
&where \quad a = s_{13} - c_{123}, b = c_3 s_1 + c_{12} s_3, c = c_1 s_3 - c_{23} s_1, d = c_{13} - c_2 s_{13}, \\
&\quad e = c_4 s_4 + c_{33} s_2, f = c_{24} - c_3 s_{24}, g = s_4 a - c_{14} s_2, h = c_4 a + c_1 s_{24}, \\
&\quad i = c_3 h + s_3 b, j = c_4 c - s_{12}, k = s_3 c + c_4 s_{12}, l = s_3 e + c_5 s_{23}, \\
&\quad m = c_5 e - s_{233}, n = s_6 g - c_6 l, o = c_6 g + s_6 l, p = c_3 k + s_5 d.
\end{aligned}
\tag{6}
$$

where s_i and c_i stand for $\sin(\theta_i)$ and $\cos(\theta_i)$, s_{ij} and c_{ij} stand for $\sin(\theta_i)\cdot\sin(\theta_j)$ and $\cos(\theta_i)\cdot\cos(\theta_j)$.

As mentioned above, the FK equations for a seven-degree-of-freedom robotic arm can be found by the DH coordinate method. If the joint angle vectors $(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6,\theta_7)$ are given, the positional attitude of the robotic arm $(p_x,p_y,p_z,\alpha,\beta,\gamma)$ can be computed directly by Equation (6). However, if the positional attitude $(p_x,p_y,p_z,\alpha,\beta,\gamma)$ of the robotic arm is given, it is used to calculate the joint angle vectors $(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6,\theta_7)$ The IK equations are highly nonlinear and are considered challenging optimization problems. Since the analytical method is used up the objective of the non-IK problem is to optimize the joint angle vectors of the robotic arm $\vec{\theta}_i = (\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6,\theta_7)$ in order to eliminate the end-effector's positional error.The FK equation is used to calculate the position and attitude of the end-effector. For the desired positional attitude $P_0 = (p_{x0}, p_{y0}, p_{z0}, \alpha_0, \beta_0, \gamma_0)$, the fitness of the candidate joint angle vectors $\vec{\theta}_i$ is defined as:

$$
f_{error}(\vec{\theta}_i) = w_1 \cdot f_1(\vec{\theta}_i) + w_2 \cdot f_2(\vec{\theta}_i) \tag{7}
$$

$$
f_1(\vec{\theta}_i) = \left[(p_{xi} - p_{x0})^2 + (p_{yi} - p_{y0})^2 + (p_{zi} - p_{z0})^2\right]^{\frac{1}{2}} \tag{8}
$$

$$
f_2(\vec{\theta}_i) = \left[(\alpha_i - \alpha_0)^2 + (\beta_i - \beta_0)^2 + (\gamma_i - \gamma_0)^2\right]^{\frac{1}{2}} \tag{9}
$$

where $w_1 + w_2 = 1$ denotes the weights of position and pose errors, and $P_i = (p_{xi}, p_{yi}, p_{zi}, \alpha_i, \beta_i, \gamma_i)$ denote the the bit position of the end-effector corresponding to the joint angle vector $\vec{\theta}_i$, which can be calculated by Equation (6).

The fitness function defined in Eq. (7) consists of position and attitude errors. It is worth noting that in previous studies, many researchers only considered the position and not the attitude, reducing the complexity of the IK problem. Although these algorithms obtained high accuracy, they deviated from many real-world application scenarios. The position and attitude of the end-effector are considered together to obtain the complete position of the robotic arm.

II. B.Optimal trajectory planning model of robotic arm based on time shock

II. B. 1) Description of the problem

Discretizing the motion trajectory to be executed by the robotic arm in Cartesian space, a sequence of spatial position matrices T_i can be obtained, and through inverse kinematics the spatial position sequence can be converted into a sequence of joint positions of the robotic arm $p_i = \left[p_{1,i} p_{2,i} \cdots p_{j,i} \cdots p_{N,i}\right]^T, i = 0,1,2\cdots,n$ and $1 \le j \le N$, N denotes the number of joints of the robotic arm, and $p_{j,i}$ denotes the position of the j th joint at time node t_i. The joint position-time node sequence is:

$$
Q = \{(p_i, t_i) \mid i = 0,1,\cdots,n\} \tag{10}
$$

By interpolating the above joint position-time node sequence by a polynomial or spline function, the joint trajectory profile can be obtained, i.e:

$$\begin{cases} p_j = f_j(t), v_j = \dot{f}_j(t) \\ a_j = \ddot{f}_j(t), p_j = \dddot{f}_j(t) \\ st. \quad p_j = f_j(t_i) \\ \quad |v_j(t)| \le VC_j \\ \quad |a_j(t)| \le AC_j \\ \quad |j_j(t)| \le JC_j \end{cases} \tag{11}$$

where VC_j, AC_j and JC_j denote the maximum velocity VC, acceleration AC, and shock (plus acceleration) JC allowed for joint j.

Optimize the two coupled and contradictory motion performance indexes of robotic arm joint motion time and acceleration (impact), the former to improve the execution efficiency of the robotic arm, the latter to ensure that the impact of the robotic arm in the execution of the motion trajectory is reduced as much as possible. The impact is bounded to ensure the tracking accuracy of the trajectory, to obtain a higher operating speed, for the joint actuator, the smaller the joint impact accumulation, so that the driving torque change accumulation will be smaller, the smoother the movement of the robotic arm.

The traditional time-optimal trajectory planning scheme is mainly used to optimize the robotic arm joint running time under polynomial or B-spline interpolation curves, and the impact is only used as a constraint, which can only guarantee that the impact is bounded during the motion process, and cannot guarantee that the impact does not change abruptly. The polynomial-based joint space interpolation is not locally controllable. 3-times B-spline interpolation does not guarantee that the shock curves are continuous, and it will result in the sudden change of shocks.

In view of the above problems, 5 times B spline function is used to construct the trajectory curves of each joint motion of the robotic arm, and the joint motion time and impact are taken as optimization indexes at the same time, so that the optimal time impact planning problem of the robotic arm is summarized as a multi-objective optimization problem, that is:

$$\begin{cases} \min \quad S(x) = \min\left[S_1(x), S_2(x), \cdots, S_f(x)\right] \\ s.t. \quad\quad g_j(x) \le 0 \quad j = 1, 2\cdots, p \\ \quad\quad\quad h_k(x) = 0 \quad k = 1, 2\cdots, l \end{cases} \tag{12}$$

where the objective function $S_i(x)(i = 1, 2\cdots, f)$ and the constraint functions $g_j(x) \le 0$ and $h_k(x) = 0$ are functions of the decision variables $x = (x_1, x_2, \cdots, x_n)$.

Given a feasible solution $x^* \in A$ with $\forall x \in A$ with $S(x^*) < S(x)$, x^* is known as the absolute optimal solution to the multi-objective planning problem. If there is no $x \in A$ such that $S(x) < S(x^*)$, then x^* is called the efficient solution of the multi-objective planning problem, which is also called the Pareto optimal solution.

In order to improve the operating efficiency of the robotic arm and reduce the impact, the following optimization objective is defined:

$$S_1 = \sum_{i=1}^{n-1} h_i \tag{13}$$

$$S_2 = \sum_{j=1}^{N} \sqrt{\frac{1}{t_f}\int_0^{t_f} (\dddot{f}_j(t))^2 dt} \tag{14}$$

where $h_i = t_{i+1} - t_i$ denotes the time interval between neighboring nodes; and t_f denotes the total time of the joint motion of the robotic arm. Consider the motion constraints of the robotic arm and define the following constraints:

$$g_j = |\dot{p}_j| - VC_j, j = 1, 2, \cdots, N \tag{15}$$

$$g_{j+N} = |\ddot{p}_j| - AC_j, j = 1,2,\cdots,N \tag{16}$$

$$g_{j+2N} = |\dddot{p}_j| - JC_j, j = 1,2,\cdots,N \tag{17}$$

To satisfy the impact continuous trajectory, each joint trajectory curve is at least third-order geometrically continuous, and $k \geq 4$ is required since the k th order B-spline curve has the property of C^{k+1} continuity. In this paper, the interpolated curve construction is carried out by using the 5th order B spline function [39].

II. B. 2) B-spline interpolation trajectory construction

The k times B-spline curve is described uniformly as:

$$p(u) = \sum_{i=0}^{n} d_i N_{i,k}(u) \tag{18}$$

where $N_{i,k}(u)(i = 0,1\cdots,n)$ denotes the spline base of the k th B-spline; $d_i(i = 0,1\cdots,n)$ denotes the control vertex of the curve.

and:

$$\begin{cases} N_{i,0}(u) = \begin{cases} 1 & u_i < u < u_{i+1} \\ 0 & \text{other} \end{cases} \\ N_{i,k}(u) = \dfrac{u - u_i}{u_{i+k} - u_i} N_{i,k-1}(u) + \dfrac{u_{i+k+1} - u}{u_{i+k+1} - u_{i+1}} N_{i+1,k-1}(u) \\ \text{And} := \dfrac{0}{0} = 0 \end{cases} \tag{19}$$

where $N_{i,k}(u)(i = 0,1\cdots,n)$ is the basis function, and the basis function interval $u \in [u_i, u_{i+k+1}]$ is defined as $U = [u_0, u_1, \cdots, u_{n+2k}]$ as a node vector. For a point $u \in [u_i, u_{i+k+1}]$, there are only up to $k+1$ non-zero k th B-splines $N_{r,k}(u)(r = i-k, i-k+1\cdots, i)$, and the other k th B-splines are 0. Thus the B-spline curve can be expressed also as:

$$p(u) = \sum_{r=i-k}^{i} d_r N_{r,k}(u) \quad u \in [u_i, u_{i+k+1}] \tag{20}$$

Convert the domain of definition of the interpolating curve to the domain of canonical parameters with $k+1$ first and last repetition, i.e., $u \in [u_k, u_{n+k}] = [0,1], u_0 = u_1 = \cdots = u_k = 0, u_{n+k} = u_{n+k+1} = \cdots = u_{n+2k} = 1$.

The time node t_i is normalized using the cumulative chord length parameterization, i.e:

$$u_i = u_{i-1} + \frac{|\Delta t_{i-k-1}|}{\sum_{r=0}^{n-1} |\Delta t_r|}, i = k+1, k+2, \cdots, k+n \tag{21}$$

The derivative vectors of each order for each joint of the robotic arm are calculated from the DeBoor-Cox recursive formula, which is given by the recursive formula:

$$\begin{cases} p^l(u) = \sum_{r=i-k+l}^{i} d_r^l N_{r,k-l}(u), u_i \leq u < u_{i+1} \\ d_r^l = \begin{cases} d_j & l = 0 \\ (k+1-l)\dfrac{d_r^{l-1} - d_{r-1}^{l-1}}{u_{r+k+1-l} - u_r} & l = 1,2,\cdots,r \\ & r = i-k+l,\cdots,i \end{cases} \end{cases} \tag{22}$$

In order to make each joint motion trajectory pass through $n+1$ position nodes in Q, it is necessary to invert the control points $d_r \in R^{(n+k)\times 1}$ of the B-spline trajectory equations, and from the time-position sequences, one can list $n+1$ equations, i.e.:

$$p(u_{i+k}) = \sum_{r=i}^{i+k} d_r N_{r,k}(u) = p_i, i = 0,1,2,\cdots,n \tag{23}$$

Therefore it is also necessary to add $k-1$ conditions, and in this paper we take $k=5$ and obtain the remaining boundary conditions by configuring the boundary conditions, i.e:

$$\begin{cases} \dot{p}(u_0)=v_0, \ddot{p}(u_0)=a_0 \\ \dot{p}(u_e)=v_e, \ddot{p}(u_e)=a_e \end{cases} \tag{24}$$

where and a_0 denote the joint starting velocity and acceleration, and v_e and a_e denote the joint termination velocity and acceleration, respectively.

After obtaining a system of $n+5$ equations containing $n+5$ control vertices, the control vertex equations for the j th joint are described in matrix form, i.e:

$$A_j d_j = p_j \tag{25}$$

In the formula:

$$\begin{cases} A_j \in R^{(n+k)\times(n+k)} \\ d_j = \left[d_{j,0} \ \ d_{j,1} \cdots d_{j,(n+1)} \ \ d_{j,(n+2)} \ \ d_{j,(n+3)} \ \ d_{j,(n+4)} \right]^T \\ p_j = \left[p_{j,0} \ \ p_{j,1} \cdots p_{j,n} \ \ v_{j0} \ \ v_{jc} \ \ a_{j0} \ \ a_{js} \right]^T \end{cases} \tag{26}$$

One can invert the control vertex d_j:

$$d_j = A_j^{-1} p_j \tag{27}$$

The trajectory profile of joint j on $t \in [t_0, t_n]$ is derived from the solved control vertex vectors and temporal node vectors, and its derivative profiles of all orders can be deduced by DeBoor-Cox recursive formulae, viz:

$$\begin{cases} v(t)=\dot{p}(u)=\sum_{r=i-k+1}^{i} d_r^1 N_{r,k-1}(u) \\ a(t)=\ddot{p}(u)=\sum_{r=i-k+2}^{i} d_r^2 N_{r,k-2}(u) \\ j(t)=\dddot{p}(u)=\sum_{r=i-k+3}^{i} d_r^3 N_{r,k-3}(u) \end{cases} \tag{28}$$

II. B. 3) Motion constraints

The derivative curves of each order of each joint deduced from 5 times non-uniform B-splines need to satisfy the relevant kinematic constraints, i.e., the point with the largest absolute value on the velocity, acceleration, and acceleration curves of each joint is required to satisfy the relevant constraints, and the analytical solving method usually adopts the golden section method, but the analytical method of solving for the point with the largest absolute value on the curves is more difficult. Then:

$$\begin{cases} \max\{|\, d_{j,i}^1 \,|\} \le VC_j, i=1,2,\cdots,n+6 \\ \max\{|\, d_{j,i}^2 \,|\} \le AC_j, i=2,3,\cdots,n+6 \\ \max\{|\, d_{j,i}^3 \,|\} \le JC_j, i=3,4,\cdots,n+6 \end{cases} \tag{29}$$

Replacing the kinematic constraints of each joint with the constraints of each control vertex of the B spline curve reduces the difficulty of the solution.

Based on the above, the optimal time-impact planning problem for the robotic arm can be converted into a multi-objective nonlinear constrained problem:

$$\begin{cases} \min & S_1 = \sum_{i=1}^{n-1} h_i \\ & S_2 = \sum_{j=1}^{N} \sqrt{\frac{1}{t_f}\int_0^{t_f} (\dddot{f}_j(t))^2 dt} \\ s.t. & g_j(X) = \max\limits_{i=1,2,\cdots,n+6} \{|d_{j,i}^1|\} - VC_j \le 0 \\ & g_{j+N}(X) = \max\limits_{i=2,3,\cdots,n+6} \{|d_{j,i}^2|\} - AC_j \le 0 \\ & g_{j+2N}(X) = \max\limits_{i=3,4,\cdots,n+6} \{|d_{j,i}^3|\} - JC_j \le 0 \end{cases} \tag{30}$$

Eq. $t_f = \sum_{i=1}^{n-1} h_i$.

II. C.Model Solving for Improved Mucilage Algorithm

II. C. 1) Basic slime mold algorithm

The SMA algorithm is a stochastic optimization method that simulates the three phases of the food finding process of slime molds, i.e., food discovery, approaching food, and encircling food [40].

(1) Food discovery phase

When the food concentration satisfies the condition, the weight near the region is larger; when the food concentration is lower, the weight of the region will be reduced, thus shifting to other regions to explore the rest. Then:

$$X(t+1) = \begin{cases} X_b(t) + v_b \cdot (W \cdot X_A(t) - X_B(t)), r < p, \\ v_c \cdot X(t), r \ge p. \end{cases} \tag{31}$$

In the formula:

$$\begin{cases} v_b = [-a, a] \\ a = \arctan h\left(1 - \left(\frac{t}{t_{\max}}\right)\right) \\ p = \tanh |S(i) - F_0|, i \in 1, 2, \cdots, N \end{cases} \tag{32}$$

The W -weight update strategy is:

$$\overrightarrow{W(\text{SmellIndex}(i))} = \begin{cases} 1 + r \cdot \log\left(\frac{F_b - S_i}{F_b - F_\omega} + 1\right), \text{condition}, \\ 1 - r \cdot \log\left(\frac{F_b - S_i}{F_b - F_\omega} + 1\right), \text{others}; \end{cases} \tag{33}$$

$$\text{SmollIndax} = sort(S) \tag{34}$$

In Eq. (31), r denotes a random value in the interval [0, 1], $X_b(t)$ denotes the currently obtained best position, W is the weight, and $X_A(t)$ and $X_n(t)$ are 2 randomly selected individuals. v_b is the control parameter, v_c decreases linearly from 1 to 0, t represents the current number of iterations, $X(t)$ denotes the current bit, and p is the selection switch. In Eq. (33), t represents the current number of iterations, $t_{\max}$ represents the maximum number of iterations, $S(i)$ is the population after performing the ranking, and F_D is the best value among all iterations. The vector F_b denotes the optimal fitness value obtained during the current iteration, F_ω denotes the worst fitness value obtained during the current iteration, and condition denotes that $S(i)$ is the first half of the population, $i = N/2$. SmellIndex in Eq. (34) ranks the adaptation values.

(2) Proximity to food stage

The proximity to food phase simulates the contraction method within the mucus vein structure, and the position is adjusted according to the quality of the food, i.e., the higher the concentration of the food, the greater the weight

of the region. Otherwise, the weights of the region are converted to other regions according to equation (35). The position update strategy is:

$$X=\begin{cases} R\cdot(B_u-B_1)+B_1, R<z; \\ X_b(t)+v_b\cdot(W\cdot X_A(t)-X_B(t)), r<p; \\ v_c\cdot X(t), r\geq p \end{cases} \tag{35}$$

where R denotes a random value in the interval [0, 1], B_1 and B_u denote the lower and upper bounds of the range of the search space, and z denotes the switching probability, which determines whether the SMA is exploring other food sources or searching around the best individual; the other variables are the same as in Eq. (31).

(3) Surrounding food phase

The encircling food phase is to simulate the behavior of v_b, where v_b floats in the interval $[-a,a]$ in a random manner and gradually decreases to zero as the number of iterations increases. The value of v_c floats in the interval [0, 1] and eventually reaches zero.

II. C. 2) Improved Mucilage Algorithm Solving with Multiple Strategies

(1) Initialization population strategy based on Tent mapping reverse learning

To address the shortcomings of the SMA algorithm, the Tent mapping [41] reverse learning strategy is used to reinitialize the population in order to obtain a better initial population.

1) Tent mapping

It has been shown that the advantages of randomness, regularity and traversal of chaotic motion can be utilized to produce rich and diverse initial populations. The Tent chaotic mapping expression is:

$$\begin{cases} x_{k+1}=x_k/\varphi, 0<x_k<\varphi, \\ x_{k+1}=(1-x_k)/(1-\varphi), \varphi\leq x_k<1。 \end{cases} \tag{36}$$

When $\varphi\in(0,1)$ and $x_k\in[0,1]$, the system is in a chaotic state.

2) Reverse learning strategy

Reverse learning strategy is a method to improve the search efficiency, using the idea of seeking the reverse solution to its initial population to increase the diversity of the search population and thus improve the quality of the best point. The basic idea is: if there exists a point on the D -dimensional space $X=(X_1,X_2,\cdots,X_p)$ and $x_i(i=1,2,\cdots,D)$ is distributed in the interval $[c,d]$, the inverse point $x_i^{'}=c+d-x_i, i\in[0,D]$. So the population X_i is the reverse population of X_i, and the reverse population is given by:

$$X_i^{'}=L_i+U_i-X_i \tag{37}$$

where L_i and U_i are the upper and lower bounds, X_i is the original initial population, and X_i is the reverse population. The original and reverse populations are combined into a new population $X=(X_i\cup X_i)$, and then the fitness values are calculated and ranked, and the top N points are selected as the initial population X.

(2) Adaptive weighting strategy

This paper introduces the adaptive weights strategy of nonlinear change, balances the algorithm exploration and development capacity, fully guarantees the effectiveness of the algorithm, adaptive weights as in equation (38):

$$\omega=(\omega_{initial}-\omega_{final})\times\left(1-\sin\left(\frac{\pi}{2}\left(\frac{t}{T}\right)^2\right)\right) \tag{38}$$

where $\omega_{aiitial},\omega_{final}$ denotes the initial and final values, ω is in the range of [0, 1], t is the current iteration number, and T is the maximum iteration number.

The improved position update formula is:

$$X=\begin{cases} \omega\cdot(B_u-B_1)+B_1, \omega<z; \\ X_b(t)+v_b\cdot(W\cdot X_A(t)-X_B(t)), r<p; \\ v_c\cdot X(t), r\geq p。 \end{cases} \tag{39}$$

In this paper, we introduce the adaptive weighting strategy for position update, which dynamically changes the weights nonlinearly as the number of iterations increases. Larger weights in the early delivery can obtain a strong exploration ability and quickly converge to the global optimum, thus improving the convergence speed of the algorithm, and smaller weights are selected in the later iterations to improve the ability to jump out of the local optimum.

(3) Perturbation strategy

When the viscous bacteria update the position, the current optimal position is generally selected for updating, which makes the search scope narrower and the number of iterations reduced. In order to improve the efficiency of global search, a perturbation is chosen to add a perturbation to the current optimal position, and a greedy strategy is judged on the position information to determine whether the current position is optimal or not. A randomized perturbation of the current position is given by the formula:

$$\hat{X}(t)=\begin{cases}X(t)+0.5r_1X(t), rand<0.5,\\ X(t), rand\geq 0.5\end{cases} \tag{40}$$

The greedy mechanism strategy is used to make the judgment of whether to keep the perturbation or not, and the formula is:

$$X(t)=\begin{cases}\hat{X}(t), f(\hat{X}(t))<f(X(t)),\\ \hat{X}(t), f(\hat{X}(t))\geq f(X(t)).\end{cases} \tag{41}$$

The steps of the improved SMA algorithm proposed in this paper are as follows:

Stepl Initialize each parameter and generate the initial population X based on Tent chaotic mapping and reverse learning strategy.

Step2 Calculate the fitness value and sort it.

Step3 Update p, v_b, v_c according to the iteration conditions.

Step4 Update position, weights in each iteration by Eq.

Step5 Recalculate the fitness value and select the optimal position by choosing the adapted update position formula at the same time.

Step6 Perturbation update the current optimal position by using the formula.

Step7 Judge whether the set termination condition is satisfied, if yes, output the global optimal value and the algorithm ends, otherwise go back to Step2.

III. Simulations and experiments

III. A. Performance analysis based on benchmark functions

In order to verify the feasibility and superiority of the improved SMA algorithm, the CEC2005 test functions are used.The CEC2005 test functions of single-peak functions are evaluated for the local search capability and the multi-peak functions are evaluated for the global search capability.In this section, 4 test functions are selected from these 2 categories for evaluation, respectively.

III. A. 1) Performance Comparison of Algorithms

In order to further test the optimization accuracy of the improved SMA algorithm, the Particle Swarm Algorithm (PSO), Gray Wolf Algorithm (GWO), and Slimy Mushroom Algorithm (SMA) are selected for a comprehensive comparison based on the CEC2005 test function, which includes single-peak function and multiple-peak function. In order to fairly verify the effectiveness of the improved SMA algorithm, the test is carried out in the same running bad environment, and MATLAB is used to complete the simulation, and the population number of all algorithms is set to 100, and the number of iterations is 2000. In order to reduce the influence of chance factors on the experiment, each algorithm is run 50 times individually, and the optimal value, average value and variance are used as the final evaluation indexes to test the algorithm's equal-optimization ability, optimization search speed and stability. The expression, search space and optimal value of each test function are shown in Table 2, and the parameter settings of each algorithm are shown in Table 3.

Table 2: Reference test function

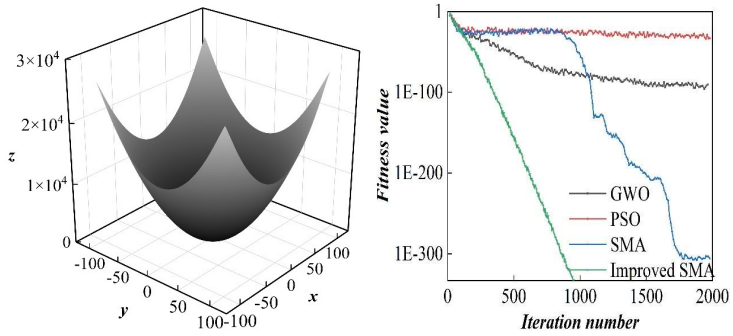
Function	Type	Dimension	Search space	Theoretical optimal solution
F1	Sphere	30	$[-100,100]^2$	0
F2	Schwefel's 2.22	30	$[-10,10]^n$	0
F6	Ackley	30	$[-32,32]^n$	0
F7	Penalized1	30	$[-50,50]^n$	0

Table 3: Algorithm parameter setting

Algorithm	Parameter setting
PSO	$c_1 = c_2 = 1.5, w = 0.8$
GWO	$a \downarrow 0,\ r_1, r_2 \in [0,1]$
SMA	$z = 0.03, vb \in [-a, a]$
Improved SMA	$z = 0.03, Cr = 0.5, q = 0.9$

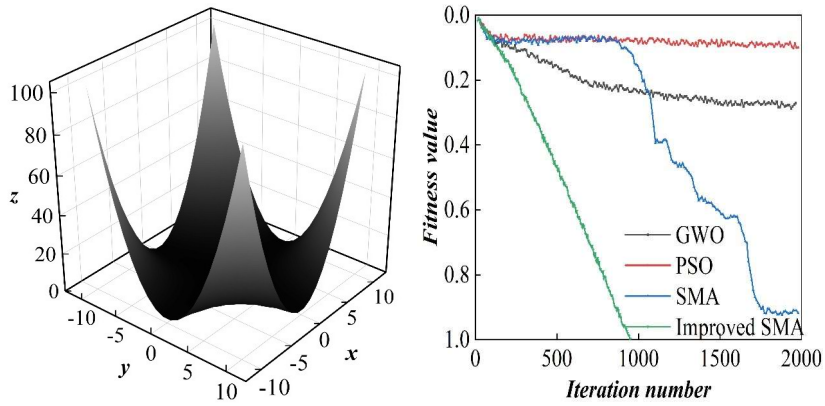
III. A. 2) Convergence curve analysis

The convergence curves of benchmark test functions can clearly reflect the convergence speed and convergence accuracy of each algorithm. The convergence curves of the four optimization algorithms on the four benchmark test functions (single-peak function F1, F2 and multi-peak function F6, F7) are shown in Fig. 2~Fig. 5, and (a)~(b) are the convergence accuracy and convergence curves, respectively. Where the horizontal axis represents the number of iterations, and the vertical axis represents the adaptation value, when the curve is no longer shown with the increase of the number of iterations, it means that the algorithm has found the optimal value, and the improved SMA algorithms are all able to achieve the optimal adaptation.



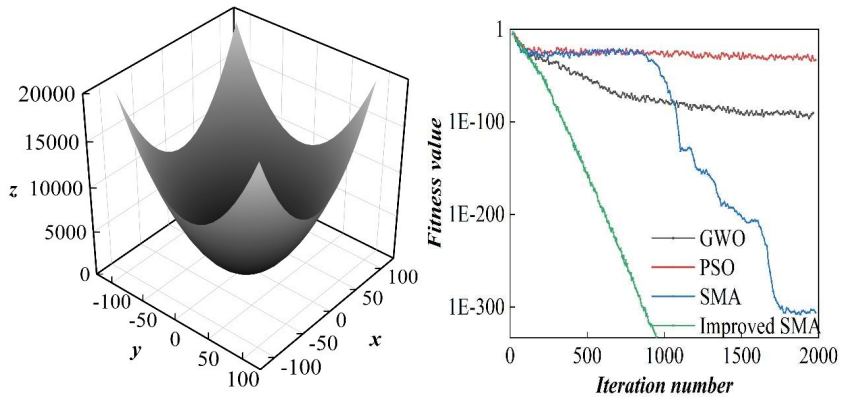
(a) Convergence accuracy (b) Convergence curve

Figure 2: Single peak function F1



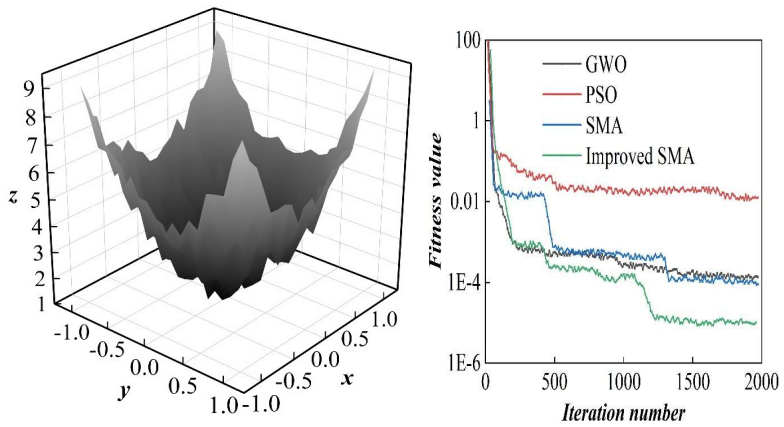
(a) Convergence accuracy (b) Convergence curve

Figure 3: Single peak function F2



(a) Convergence accuracy (b) Convergence curve

Figure 4: Multimodal function F6



(a) Convergence accuracy (b) Convergence curve

Figure 5: Multimodal function F7

III. A. 3) Algorithm optimization comparison

For each test function, the four algorithms are independently carried out 50 times, record the optimal value of each function in each group of results, and find the average and standard deviation of the results of the 50 operations, where the optimal value reflects the quality of the understanding, the average reflects the accuracy that the algorithm can achieve, and the standard deviation reflects the robustness and stability of the algorithm. The comparison results are shown in Table 4.

For the single-peak functions F1 and F2, PSO and GWO did not find the optimal value, and SMA only achieved the optimal value in F1, while the improved slime mold algorithm achieved the theoretical optimal value in F1 and F2. For the multi-peak functions F6 and F7, although the improved mucilage algorithm does not obtain the optimal value, the solution quality is higher than that of GWO and SMA in F6, and higher than that of PSO, GWO, and SMA in F7, which reflects the strong ability to find the optimal value.The results of the four test letters show that the improved SMA algorithm is significantly better than the remaining three algorithms in terms of stability and robustness, and also illustrates that the introduction of Tent mapping, backward learning strategy, adaptive weighting strategy, and the introduction of the Tent mapping are more effective than the other three algorithms. It also shows that the introduction of the Tent mapping, the inverse learning strategy, the adaptive weights strategy and the perturbation strategy has a positive effect on the global optimization ability of the SMA algorithm, and reduces the chances of the SMA falling into the local optimum. The synthesized data shows that the algorithm has a certain improvement effect.

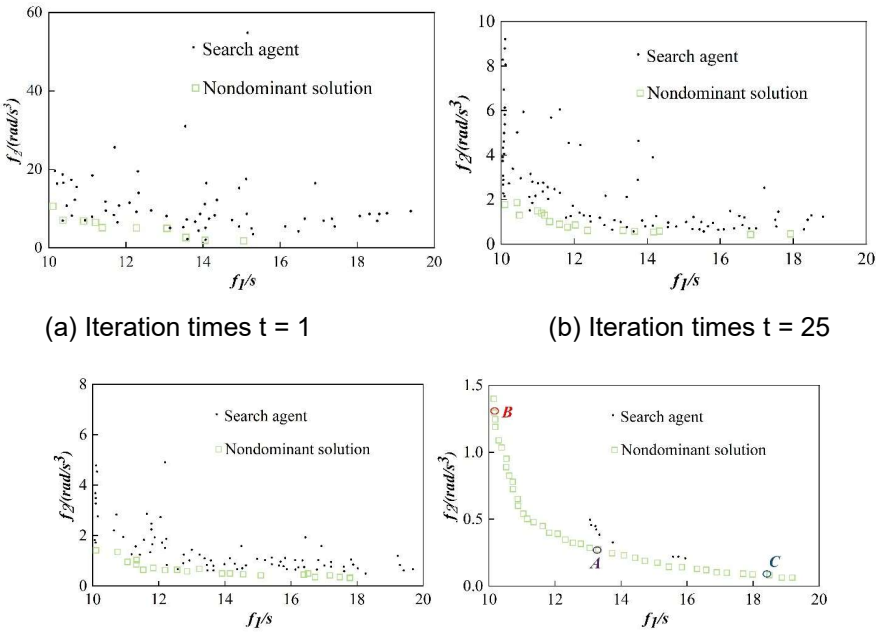
Table 4: Algorithm comparison results

Test function	Algorithm category	Optimal solution	Mean value	Standard deviation
F1	PSO	2.6885	5.0112	1.2579
	GWO	6.5921e-61	9.1982e-59	2.2438e-59
	SMA	0.0000	0.0000	0.0000
	Improved SMA	0.0000	0.0000	0.0000
F2	PSO	6.5383	10.0115	1.5984
	GWO	4.7844e-36	9.6921e-35	1.0885e-35
	SMA	1.8845e-305	4.7641e-185	0.0000
	Improved SMA	0.0000	0.0000	0.0000
F3	PSO	1.555	2.0258	0.1396
	GWO	7.4454e-15	1.4845e-15	2.8972e-15
	SMA	8.1154e-09	1.9523e-05	1.1522e-05
	Improved SMA	5.6251e-19	1.3345e-13	6.4012e-13
F4	PSO	1.9552	2.1456	1.5624e-01
	GWO	6.5382e-03	4.0012e-02	1.3945e-02
	SMA	3.1578e-06	7.3216e-05	8.305e-05
	Improved SMA	2.6642e-06	7.1434e-05	5.8219e-05

III. B. Time-shock optimal trajectory planning

In the simulations in this section, two configurations are randomly selected as the start and end points for the robotic arm trajectory planning. In particular, the joint configurations of the starting point are [0.1, 0.2, 0.5, -0.1, 0.2, 0.4, 0.1, -0.2, 0.1] (in rad) and the joint configurations of the termination point are [1.5, 1.3, 1.7, -1.5, 1.0, 1.2, 1.8, -1.8, 2.0] (in rad).

The above trajectory planning problem is solved using the improved SMA algorithm in the framework of the trajectory planner presented in Section 2. The population size, the maximum number of iterations and the maximum number of members in the archive of the algorithm are set to 100, and the number of variables to be searched is 30. The convergence process of the improved SMA algorithm is shown in Fig. 6, with (a) to (d) representing the number of iterations of 1, 25, 50, and 100, respectively. Obviously, as the number of iterations increases, the slime mold individuals successfully converge to the concave curve (Pareto front) composed of the objective functions f1 and f2.



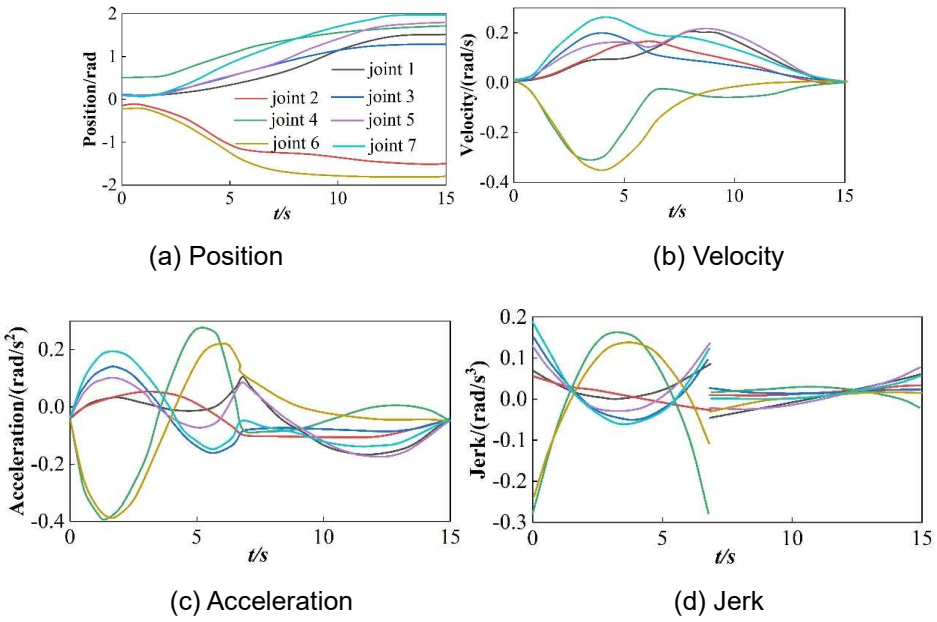
(a) Iteration times t = 1 (b) Iteration times t = 25

(c) Iteration times t = 50 (d) Iteration times t =100

Figure 6: The convergence of the improved SMA algorithm

Based on the solution A of the more compromise measure of f1 and f2 in the archive of Fig. 6, the joint motion trajectory curves obtained from the computation are shown in Fig. 7, with (a)~(d) being the position, velocity, acceleration and impact curves, respectively. Each joint accurately accomplishes the point-to-point motion under the premise of satisfying the constraints. Its position, velocity and acceleration curves are continuous and smooth, the running time is less than 15s, and the impact is between -0.30 rad/s³~0.20 rad/s³. Fig. 8 shows the conformational changes presented by the robotic arm executing the above motion trajectory, and the motion trajectory of the end of the robotic arm calculated by positive kinematics. In the figure, the end trajectory of the robotic arm is continuous and smooth, indicating that solution A ensures the safety and smoothness of the end motion of the robotic arm.

The same conclusion can be obtained at other solutions in the archive, which will not be repeated due to space limitation. In practical applications, the optimal solution can be selected in the archive according to the demand, taking the archive shown in Fig. 6(d) as an example, if it is hoped to obtain a smaller running time, solution B can be selected; if it is hoped to obtain a smaller impact, solution C can be selected.



(a) Position (b) Velocity

(c) Acceleration (d) Jerk

Figure 7: The trajectory of A

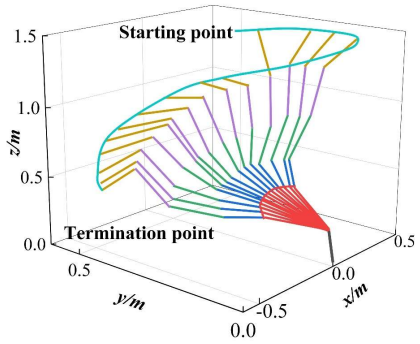
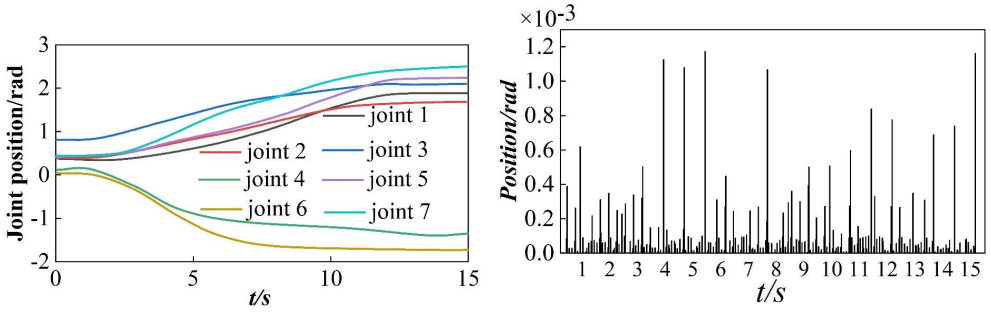


Figure 8: The configuration of solution A

Finally, the motion trajectories of solution A are experimentally verified on the experimental platform, and for each joint, the planned position profile is input to the driver as a control command via a network cable with a sampling time of 5 ms. Fig. 9 shows the motion trajectories of the joints and the trajectory tracking errors measured by the host computer in the experiments, and (a) and (b) are the distribution of the joint position profiles and the joint position errors, respectively. The error between its motion trajectory and the simulation result is roughly between [10-7, 10-3] rad, indicating that the robotic arm can accurately track the optimal trajectory.

In summary, the time-impact optimal trajectory searched by the trajectory planner can enable the robotic arm to complete the specified point-to-point motion safely, smoothly and accurately.



(a) Joint position curve (b) Joint position error

Figure 9: The motion trajectory and tracking error of the experimental measurement

IV. Conclusion

In order to solve the time-impact optimal trajectory planning problem for redundant robotic arms, this paper proposes an optimal trajectory planner based on an improved viscous bacteria algorithm. The initial population is optimized by Tent chaotic mapping reverse learning strategy, which improves the convergence speed of the algorithm. The adaptive weights strategy avoids the phenomenon of local extremes of the algorithm and improves the speed of slime molds approaching and acquiring food, and the perturbation strategy updates the optimal position with perturbations to find a better global optimal value, and the algorithm is able to avoid the phenomenon of premature maturity.

The performance of the improved SMA was tested using using the CEC2005 test function, and the trajectory planner was simulated and experimentally validated on a seven-degree-of-freedom redundant robotic arm. The experimental results show that the improved sticky mushroom algorithm effectively enhances the performance of the SMA with strong superiority. The trajectory planner successfully searches for efficient and smooth motion trajectories with a running time of less than 15 s and impacts between [10-7 rad, 10-3 rad]. The study enables the robotic arm to complete the specified point-to-point motion safely, smoothly and accurately.

Funding

2022 Guiding Project of Scientific Research Plan of Hubei Provincial Department of Education, China.

References

[1] Ghadge, K., More, S., Gaikwad, P., & Chillal, S. (2018). Robotic arm for pick and place application. International Journal of Mechanical Engineering and Technology, 9(1), 125-133.

[2] Kareemullah, H., Najumnissa, D., Shajahan, M. M., Abhineshjayram, M., Mohan, V., & Sheerin, S. A. (2023). Robotic Arm controlled using IoT application. Computers and Electrical Engineering, 105, 108539.

[3] Juang, J. G., Tsai, Y. J., & Fan, Y. W. (2015). Visual recognition and its application to robot arm control. Applied Sciences, 5(4), 851-880.

[4] Mohammed Ali, H., Hashim, Y., & A AL-Sakkal, G. (2022). Design and implementation of Arduino based robotic arm. International Journal of Electrical and Computer Engineering, 12(2), 1411-1411

[5] Long, D. T., Binh, T. V., Hoa, R. V., Anh, L. V., & Toan, N. V. (2020). Robotic arm simulation by using matlab and robotics toolbox for industry application. International Journal of Electronics and Communication Engineering, 7(10), 1-4.

[6] Yang, A., Chen, Y., Naeem, W., Fei, M., & Chen, L. (2021). Humanoid motion planning of robotic arm based on human arm action feature and reinforcement learning. Mechatronics, 78, 102630.

[7] Mirrazavi Salehian, S. S., Figueroa, N., & Billard, A. (2018). A unified framework for coordinated multi-arm motion planning. The International Journal of Robotics Research, 37(10), 1205-1232.

[8] Zhao, J., Wang, C., & Xie, B. (2023). Human-like motion planning of robotic arms based on human arm motion patterns. Robotica, 41(1), 259-276.

[9] Tamizi, M. G., Yaghoubi, M., & Najjaran, H. (2023). A review of recent trend in motion planning of industrial robots. International Journal of Intelligent Robotics and Applications, 7(2), 253-274.

[10] Pambudi, W. S., Alfianto, E., Rachman, A., & Hapsari, D. P. (2019). Simulation design of trajectory planning robot manipulator. Bulletin of Electrical Engineering and Informatics, 8(1), 196-205.

[11] Carvalho, J., Le, A. T., Baierl, M., Koert, D., & Peters, J. (2023, October). Motion planning diffusion: Learning and planning of robot motions with diffusion models. In 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 1916-1923). IEEE.

[12] Xia, X., Li, T., Sang, S., Cheng, Y., Ma, H., Zhang, Q., & Yang, K. (2023). Path planning for obstacle avoidance of robot arm based on improved potential field method. Sensors, 23(7), 3754.

[13] Dobiš, M., Dekan, M., & Kohút, M. (2021). The Comparison of Motion Planners for Robotic Arms. Journal of Control Engineering and Applied Informatics, 23(2), 87-94.
[14] Gao, F., Jiang, T., & Yu, Z. (2024, June). Time-optimal trajectory planning for robotic arm based on improved particle swarm optimization. In Proceedings of the 2024 4th International Conference on Control and Intelligent Robotics (pp. 269-274).
[15] Völz, A., & Graichen, K. (2018, October). An optimization-based approach to dual-arm motion planning with closed kinematics. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 8346-8351). IEEE.
[16] Huang, J., Hu, P., Wu, K., & Zeng, M. (2018). Optimal time-jerk trajectory planning for industrial robots. Mechanism and Machine Theory, 121, 530-544.
[17] Pellegrinelli, S., Orlandini, A., Pedrocchi, N., Umbrico, A., & Tolio, T. (2017). Motion planning and scheduling for human and industrial-robot collaboration. CIRP Annals, 66(1), 1-4.
[18] Xie, Y., Zhao, X., Jiang, Y., Wu, Y., & Yu, H. (2024). Flexible control and trajectory planning of medical two-arm surgical robot. Frontiers in Neurorobotics, 18, 1451055.
[19] Dehghani, H., Sun, Y., Cubrich, L., Oleynikov, D., Farritor, S., & Terry, B. (2020). An optimization-based algorithm for trajectory planning of an under-actuated robotic arm to perform autonomous suturing. IEEE Transactions on Biomedical Engineering, 68(4), 1262-1272.
[20] Tian, X., & Xu, Y. (2020). Low delay control algorithm of robot arm for minimally invasive medical surgery. IEEE Access, 8, 93548-93560.
[21] Xie, Y. G., Liu, G. J., Lan, J. Y., & Gao, B. B. (2020). Abdominal needle assisted robotic arm motion control and trajectory planning. J Computers, 31(2), 298-311.
[22] Gasparetto, A., Boscariol, P., Lanzutti, A., & Vidoni, R. (2015). Path planning and trajectory planning algorithms: A general overview. Motion and operation planning of robotic systems: Background and practical approaches, 3-27.
[23] Jeon, J., Jung, H. R., Yumbla, F., Luong, T. A., & Moon, H. (2022). Primitive action based combined task and motion planning for the service robot. Frontiers in Robotics and AI, 9, 713470.
[24] Liu, W., Huang, Z., Qu, Y., & Chen, L. (2024). Path planning for robotic arms based on an improved RRT algorithm. Open Journal of Applied Sciences, 14(5), 1214-1236.
[25] Wu, R., Yang, Y., Yao, X., & Lu, N. (2024). Optimal Trajectory Planning for Robotic Arm Based on Improved Dynamic Multi-Population Particle Swarm Optimization Algorithm. International Journal of Advanced Computer Science & Applications, 15(5).
[26] Zhang, S., Xia, Q., Chen, M., & Cheng, S. (2023). Multi-objective optimal trajectory planning for robotic arms using deep reinforcement learning. Sensors, 23(13), 5974.
[27] Ekrem, Ö., & Aksoy, B. (2023). Trajectory planning for a 6-axis robotic arm with particle swarm optimization algorithm. Engineering Applications of Artificial Intelligence, 122, 106099.
[28] Jia, X., Zhao, B., Liu, J., & Zhang, S. (2024). A trajectory planning method for robotic arms based on improved dynamic motion primitives. Industrial Robot: the international journal of robotics research and application, 51(5), 847-856.
[29] Rout, A., BBVL, D., Biswal, B. B., & Mahanta, G. B. (2021). Optimal trajectory planning of industrial robot for improving positional accuracy. Industrial Robot: the international journal of robotics research and application, 48(1), 71-83.
[30] Wu, N., Jia, D., Li, Z., & He, Z. (2024). Trajectory planning of robotic arm based on particle swarm optimization algorithm. Applied Sciences, 14(18), 8234.
[31] Usubamatov, R. (2025). Physics of gyroscopic inertial torques nullification. TK Techforum Journal (ThyssenKrupp Techforum), 2025(2), 6–14.
[32] Wu, G., & Zhang, S. (2022). Real-time jerk-minimization trajectory planning of robotic arm based on polynomial curve optimization. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 236(21), 10852-10864.
[33] Miao, X., Fu, H., & Song, X. (2022). Research on motion trajectory planning of the robotic arm of a robot. Artificial Life and Robotics, 27(3), 561-567.
[34] Shin, K., & McKay, N. (1986). A dynamic programming approach to trajectory planning of robotic manipulators. IEEE Transactions on Automatic Control, 31(6), 491-500.
[35] Guoyu, Z. U. O., Mi, L. I., & Banggui, Z. H. E. N. G. (2024). Optimal trajectory planning for robotic arms based on an improved adaptive multiobjective particle swarm algorithm. Experimental Technology & Management, 41(3).
[36] Jichun, W. U., Zhaiwu, Z. H. A. N. G., Yongda, Y. A. N. G., Ping, Z. H. A. N. G., & Dapeng, F. A. N. (2024). Time optimal trajectory planning of robotic arm based on improved tuna swarm algorithm. Computer Integrated Manufacturing System, 30(12), 4292.
[37] Dai, Y., Xiang, C., Zhang, Y., Jiang, Y., Qu, W., & Zhang, Q. (2022). A review of spatial robotic arm trajectory planning. Aerospace, 9(7), 361.
[38] Chen Yanlin, Zhang Xianmin, Huang Yanjiang, Wu Yanbin & Ota Jun. (2023). Kinematics optimization of a novel 7-DOF redundant manipulator. Robotics and Autonomous Systems,163.
[39] Lucia Romani & Alberto Viscardi. (2025). Algebraic characterization of planar cubic and quintic Pythagorean-Hodograph B-spline curves. Journal of Computational and Applied Mathematics,465,116592-116592.
[40] Lin Haiping, Ahmadianfar Iman, Amiri Golilarz Noorbakhsh, Jamei Mehdi, Heidari Ali Asghar, Kuang Fangjun... & Chen Huiling. (2022). Adaptive slime mould algorithm for optimal design of photovoltaic models. Energy Science & Engineering,10(7),2035-2064.
[41] Yanying Qi, Aipeng Jiang & Yuhang Gao. (2024). A Gaussian convolutional optimization algorithm with tent chaotic mapping. Scientific Reports,14(1),31027-31027.