Research on the inverse kinematics of manipulator using an improved self-adaptive mutation differential evolution algorithm

Qianqian Zhang¹,², Daqing Wang¹ and Lifu Gao¹

Abstract
To assess the inverse kinematics (IK) of multiple degree-of-freedom (DOF) serial manipulators, this article proposes a method for solving the IK of manipulators using an improved self-adaptive mutation differential evolution (DE) algorithm. First, based on the self-adaptive DE algorithm, a new adaptive mutation operator and adaptive scaling factor are proposed to change the control parameters and differential strategy of the DE algorithm. Then, an error-related weight coefficient of the objective function is proposed to balance the weight of the position error and orientation error in the objective function. Finally, the proposed method is verified by the benchmark function, the 6-DOF and 7-DOF serial manipulator model. Experimental results show that the improvement of the algorithm and improved objective function can significantly improve the accuracy of the IK. For the specified points and random points in the feasible region, the proportion of accuracy meeting the specified requirements is increased by 22.5% and 28.7%, respectively.

Keywords
DE, IK, adaptive mutation operator, adaptive scaling factor, orienting weighting factor

Date received: 8 December 2020; accepted: 14 April 2021

Introduction
The inverse kinematics (IK) of a manipulator is an important part of its design and control, which has guiding significance for motion analysis and trajectory planning. Traditional solution methods can be divided into two categories: analytic and numerical methods. When the manipulator does not meet Pieper criterion,¹ the analytic solution cannot be obtained. For PUMA-type manipulators, when the last three axes do not exactly intersect at a common point and the shoulder joint axes are not exactly orthogonal, there is an error in the analytic solution. In this case, the numerical method can be used to find the IK. Even in cases where an analytic solution does exist, numerical methods are often used to improve the accuracy of these solutions.²,³

Because of the limitations of traditional solutions, other algorithms, such as the Newton–Raphson method, neural network method, genetic algorithm (GA), and swarm intelligence algorithm, have been proposed to solve the IK of the manipulator. Yin et al.⁴ proposed to use an improved...
Davidon–Fletcher–Powell for IK of the manipulator. Köker, Al-Mashhadany, and Khaleel have proposed several methods of using neural networks to solve the IK of the manipulator. Tabandeh et al. proposed the adaptive niche GA to generate multiple solutions for manipulator IK. Ren et al. and Rui et al. proposed to use an improved particle swarm optimization algorithm to solve the IK of the manipulator. Zhang et al. used the chaotic and parallelized artificial bee colony algorithm to solve the manipulator IK problem. The Newton–Raphson method has the advantage of a simple calculation process, high efficiency, and high precision, but its optimization result is too dependent on the choice of initial value. The neural network method has the advantages of high efficiency and high precision, but a large amount of sample data is needed during training, and the generalization performance of the network cannot be guaranteed. The traditional swarm intelligence algorithm has no special requirements for the configuration of the manipulator, but the selection of parameters has a greater impact on the performance of the algorithm. Among them, the classic DE algorithm has the advantage of simple structure, few control parameters, and strong robustness. However, the improper selection of control parameters and mutation strategies leads to slower convergence and local optimization.

Aiming at the local optimization in the process of solving the IK, this article proposes an improved adaptive mutation differential evolution (ISAMDE) algorithm. An adaptive mutation operator (AMO) and adaptive scaling factor (AF) are used to adaptively adjust the mutation strategy of the algorithm and the degree of individual scaling according to the current iteration number and error value, and balance the exploration ability and exploitation ability of the algorithm. To further improve the accuracy of the IK of the manipulator, this article optimizes the objective function of the numerical solution and proposes an adaptive orientation error weight coefficient to normalize the corresponding posture error.

**Improvement of differential evolution algorithm and objective function**

**Classic differential evolution algorithm**

The DE algorithm is a heuristic parallel search algorithm based on population differences proposed by Storn and Price, including initialization, mutation, crossover, and selection. The flowchart of the DE algorithm is shown in Figure 1.

The standard DE algorithm mainly includes the following five steps:

1. **Initialization.** Assume that D is the individual dimension, NP is the population size, t is the evolutionary generation, and x(t) is the population of the t generation. The initial individual is
   \[ x_j(0) = x_j^L + r_1 \times (x_j^U - x_j^L) \]  
   where \( x_j^L \) and \( x_j^U \) are the upper and lower bounds of the j"th variable of the \( i \)"th individual, and \( r_1 \) is a uniformly distributed random number between 0 and 1.

2. **Mutation.** Two commonly used mutation strategies are DE/rand/1/bin and DE/best/1/bin. The mutation equations are shown in equations (2) and (3)
   \[ h_j(t + 1) = x_p(t) + F(x_{p2}(t) - x_{p3}(t)) \]  
   \[ h_j(t + 1) = x_j(t) + F(x_{p2}(t) - x_{p3}(t)) \]
   where \( F \) is the scaling factor, \( p_1, p_2, p_3, p_4 \) are the serial number of different individuals randomly selected from the population, and \( x_j(t) \) is the best individual of the current group.

3. **Crossover.** The reference individual \( x_{ij}(t) \) and the mutation individual \( h_{ij}(t+1) \) were hybridized to obtain the trial individual \( v_{ij}(t+1) \)
   \[ v_{ij}(t+1) = \begin{cases} h_{ij}(t+1), & r_1 \leq C \text{ or } j \neq \text{rand} \\ x_{ij}(t), & \text{otherwise} \end{cases} \]  

4. **Selection.** The methods crossing the boundary initialization include reinitialization, reflection, projection, wrapping, and resampling. Arabas et al. verified that the boundary constraint processing method has an important effect on the optimization performance of differential evolution.
where \( j_{\text{rand}} \) is an integer randomly selected in \([1, D]\), and \( C \) is the crossover probability.

**Selection.** The selection criteria can be described as

\[
x_i(t + 1) = \begin{cases} v_i(t + 1), & f(v_i(t + 1)) < f(x_i(t)) \\ x_i(t), & f(v_i(t + 1)) \geq f(x_i(t)) \end{cases}
\]

where \( f \) is the fitness value of the individual.

**Improved self-adaptive mutation differential evolution algorithm**

The core challenge of the DE algorithm is balancing the exploration and exploitation capabilities of the algorithm. In the literature, the authors\(^{20–23} \) studied the parameter setting method and optimization method of the algorithm, and proved that the setting of control parameters plays a vital role in the optimization performance, convergence speed, and stability of the algorithm. In the literature, Qin and Suganthan\(^ {24} \) and Qin et al.\(^ {25} \) studied the optimization method of mutation strategy and proved that mutation strategy is the key to the exploration and development capabilities of equilibrium algorithms. For example, the DE/rand/1/bin strategy emphasizes the exploration ability of the algorithm, and the DE/best/1/bin strategy emphasizes the development ability of the algorithm.

In this article, the control parameters and mutation strategy are improved, and the timing of the algorithm enters the DE/best/1/bin evolution mode and the degree of scaling of the mutated individual to achieve rapid global optimization of the algorithm. The flowchart of the ISAMDE is shown in Figure 2.

**Adaptive mutation operator for adaptive adjustment mutation strategy.** To improve the global optimization capability of the algorithm, an AMO is used in the ISAMDE algorithm to independently adjust the mutation strategy of the algorithm. After improvement, the algorithm’s mutation strategy and the AMO are defined as follows

\[
h_i(t) = \begin{cases} x_{i0}(t) + F(x_{i0}(t) - x_{i\text{rand}}(t)), & \text{rand} \leq \text{AMO} \\ x_{i0}(t) + F(x_{i0}(t) - x_{i\text{rand}}(t)), & \text{otherwise} \end{cases}
\]

Among them, \( \text{AMO} = 1 - (t/T)^N \), where AMO is related to a current number of iterations \( t \), a maximum number of iterations \( T \), and a power exponent \( N \). Figure 3 shows the relationship between \( N \) and the trend of the AMO function value. Figure 3 shows that the smaller the power exponent \( N \) is, the earlier the algorithm enters the DE/best/1/bin evolution mode, the stronger the exploitation capability, and the faster the convergence speed. The exponent \( N \) determines the selection principle of the mutation strategy of the DE algorithm. When \( N \) is greater than 1, the algorithm emphasizes exploration, and when \( N \) is less than 1, the algorithm emphasizes exploitation.

When the random number is less than the AMO, the DE/rand/1/bin mutation strategy is adopted. In the early stage of evolution, the AMO is relatively large, so the DE/rand/1/bin evolution mode is preferred, giving full play to the exploration ability of the algorithm. In the later stage of evolution, the algorithm enters the DE/best/1/bin evolution mode and is adjusted to be guided by the optimal individual, and the exploitation ability of the algorithm is given full play.

**Adaptive scaling factor for adaptively adjusting the degree of differential mutation scaling.** In the DE algorithm, the scaling factor \( F \) determines the degree of scaling of the mutated individual. Increasing \( F \) can increase the diversity of the population, but it will weaken the exploitation ability of the
algorithm. Reducing $F$ can improve the exploitation ability of the algorithm, but it will increase the risk of the algorithm falling into premature convergence. To achieve fast global optimization, this article proposes an error-related AF. At the beginning of the iteration, the scaling factor is 1, and the exploration ability is strong. When the error value under the current iteration number of the algorithm is less than the set value $E_0$, the AF in equation (7) is adopted to improve the exploration ability of the algorithm

$$\text{AF} = \begin{cases} f(t)^2 \times 10^{-6} + 0.5, & \varepsilon(t) < E_0 \\ 1, & \text{otherwise} \end{cases}$$  \hspace{1cm} (7)$$

where the AF is related to the error value under the current iteration number ($\varepsilon(t)$), and the value of $E_0$ can be adjusted according to the specific engineering problems. In this article, $E_0$ is 500. Figure 4 shows the relationship between the AF and $\varepsilon(t)$.

**Figure 4.** Relationship between AF and $\varepsilon(t)$. AF: adaptive scaling factor.

### Kinematics and Improvement of Objective Function

The intelligent algorithm to solve the IK of the manipulator refers to transform the kinematics equation into the corresponding objective function and find the function extreme value.

The posture of the end effector relative to the first coordinate system is

$$T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix}$$

where the $R$ matrix is a $3 \times 3$ matrix and the $P$ matrix is a $3 \times 1$ matrix.

It can be seen in Figure 5 that the position error $E_P$ is related to the joint angles and the link parameters of the manipulator, and its value range is $(-1500, 1500)$, while orientation error $E_O$ is only the trigonometric function transformation of the joint angles, and its value range is $(-1, 1)$. The position error weight coefficient $\omega_p$ and the orientation error weight coefficient $\omega_o$ are introduced into the objective function to balance the weight of the two types of errors in the objective function so that the two types of errors are on the same order of magnitude.

**Figure 5.** Relationship between posture error and D-H parameters of manipulator: (a) Influence factors of $E_o$ and $E_P$ and (b) graph of $E_o$ and $E_P$ element values.

The objective function of the IK can be written as

$$F_{\text{obj}} = \omega_p \ast E_P + \omega_o \ast E_O$$

where $R_t$ and $P_t$ represent the orientation matrix and position matrix of the target point, respectively, $R_c$ and $P_c$ represent the orientation matrix and position matrix of the current population location, respectively. $E_P$ is calculated by the Euclidean distance between $P_t$ and $P_c$, $E_P = ||P_t - P_c ||$. $E_O$ is calculated by the Euclidean distance between $R_t$ and $R_c$, $E_O = ||R_t - R_c ||$. The orientation angles $\alpha, \beta, \gamma$ and the orientation matrix $R$ have one-to-one correspondence.$^{14}$

Tabandehe et al.$^8$ use the weight coefficients $\omega_p = 1$, $\omega_o = k \sqrt{\lambda} P_d / \pi$ in the objective function, where $k$ is the coefficient related to the rod length, $\lambda$ is the sum of the length of the connecting rod, and $P_d$ is the distance from the target position of the manipulator to the origin of the base coordinate.

In this article, $E_P$ is introduced as one of the influencing factors in $\omega_o$, and $\omega_o$ is adaptively adjusted according to the error value under the number of iterations of the current generation to ensure that the two types of errors are always at the same magnitude. In the simulation analysis of the 6 degree-of-freedom (DOF) manipulator, $k$ is 20. The position error weight $\omega_p = 1$ and the expression of $\omega_o$ are shown in equation (8)

$$\omega_o = \begin{cases} k \sqrt{\lambda} \ast E_p / \pi, & E_p \geq 0.1 \\ k \sqrt{\lambda} \ast 0.1 / \pi, & \text{otherwise} \end{cases}$$  \hspace{1cm} (8)$$

Figures 6 and 7 are the 6-DOF underwater manipulator and the 7-DOF manipulator (Cyton Gamma 300),
respectively. Tables 1 and 2 are their DH parameter tables, respectively.

### Experiments and results

To evaluate the performance of ISAMDE, three DE algorithms (DE, SAMDE, and ISAMDE) are used to solve the minimum of six standard benchmark functions.

#### Table 1. D-H parameters of 6-DOF underwater manipulator.

| Joint | \(a_i\) (mm) | \(\alpha_i\) (°) | \(d_i\) (mm) | \(\theta_i\) (°) | Range (°) |
|-------|---------------|----------------|--------------|----------------|-----------|
| 1     | 100           | -90           | 0            | \(\theta_1\)   | -90 to 180|
| 2     | 650           | 0             | 0            | \(\theta_2\)   | -60 to 60 |
| 3     | 520           | 0             | 0            | \(\theta_3\)   | -60 to 60 |
| 4     | 120           | 90            | 0            | \(\theta_4\)   | -100 to 100|
| 5     | 0             | -90           | 0            | \(\theta_5\)   | -100 to 100|
| 6     | 0             | 0             | 560          | \(\theta_6\)   | -180 to 180|

#### Table 2. D-H parameters of the Cyton Gamma 300.

| Joint | \(a_i\) (mm) | \(\alpha_i\) (°) | \(d_i\) (mm) | \(\theta_i\) (°) | Range (°) |
|-------|---------------|----------------|--------------|----------------|-----------|
| 1     | 0             | 90            | 0            | \(\theta_1\)   | 30–330    |
| 2     | 0             | -90           | 0            | \(\theta_2\)   | 70–280    |
| 3     | 0             | 0             | 0            | \(\theta_3\)   | 30–330    |
| 4     | 71.8          | 0             | 0            | \(\theta_4\)   | 70–280    |
| 5     | 71.8          | 90            | 0            | \(\theta_5\)   | 70–280    |
| 6     | 0             | -90           | 300          | \(\theta_6\)   | 75–285    |
| 7     | 0             | 0             | 260          | \(\theta_7\)   | 30–330    |

To test the effectiveness of the ISAMDE algorithm for the IK of the manipulator, simulation experiments were carried out on the IK of the 6-DOF and 7-DOF manipulators on the three DE algorithms. First, seek the IK for a fixed point in the workspace to verify the feasibility of the algorithm. Then, the IK is obtained for random target points in the workspace to verify the robustness of the algorithm.

### Benchmark function tests

\(D\) and \(P\) refer to search space dimension and population size, respectively, \(N\) means the power exponent of the AMO, \(F_0\) is the original scaling factor, \(N_{\text{max}}\) is the maximum number of iterations, and symmetric mapping is used after the mutated individual crosses the boundary.

The benchmark functions and their simulation results are shown in Table 3 and Table 4. Figure 8 shows the convergence curves of the three algorithms on the benchmark functions. The optimal value of the algorithm is presented in bold in Table 4. In the solution of \(f_1\), the SAMDE algorithm performs best with an average accuracy of \(10^{-78}\), while the average accuracy of ISAMDE is slightly lower, which is \(10^{-74}\). In the solution of \(f_2\), ISAMDE and SAMDE have the same solution accuracy. In the solution of \(f_3\), \(f_4\), and \(f_6\), the ISAMDE algorithm has the highest solution accuracy. In the solution of \(f_5\), the calculation results of ISAMDE and SAMDE are in an order of magnitude, and SAMDE is better.

ISAMDE algorithm has better convergence speed on \(f_2\)–\(f_4\), ISAMDE has better convergence speed on \(f_1\) and \(f_5\), and DE algorithm has better convergence speed on \(f_6\).

### Inverse kinematics simulation of manipulators

Aiming at the IK of the manipulator, the 6-DOF and 7-DOF manipulators are used as experimental objects, and three DE algorithms are used for simulation experiments to verify the effectiveness of the algorithm improvement and objective function improvement.

The initial parameters of the algorithm are set as follows: \(F_0 = 0.5\), \(C = 0.9\), \(N = 0.9\), \(P = 70\), \(N_{\text{max}} = 600\), and random initialization is used after the mutated individual crosses the boundary.

Termination condition of algorithm iteration: meet the given accuracy (position: \(10^{-3}\) mm and orientation: \(5 \times 10^{-4}\) rad) or reach the maximum number of iterations. The
### Table 3. Basic unimodal and multimodal benchmark functions.\textsuperscript{14}

| No. | Function | $f(x^{(0)})$ | Search range | Initialization range | Formulae |
|-----|----------|--------------|--------------|----------------------|----------|
| 1   | Sphere   | $f(0) = 0$   | $[-100,100]^D$ | $[-100,50]^D$  | $f(x) = \sum_{i=1}^{D-1} x_i^2$ |
| 2   | Rosenbrock| $f(1) = 0$   | $[2.0482,2.048]^D$ | $[-2.048,2.048]^D$ | $f(x) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$ |
| 3   | Ackley   | $f(0) = 0$   | $[-32.768,32.768]^D$ | $[-32.768,16]^D$ | $f(x) = -20\exp\left(-0.2 \sqrt{\sum_{i=1}^{D} x_i^2 / D}\right) - \exp\left(\sum_{i=1}^{D} \cos(2\pi x_i) / D\right) + 20 + \varepsilon$ |
| 4   | Griewank | $f(0) = 0$   | $[-600,600]^D$ | $[-600,200]^D$  | $f(x) = \sum_{i=1}^{D} x_i^2 / 400 - \prod_{i=1}^{D} \cos(x_i / \sqrt{i}) + 1$ |
| 5   | Rastrigin | $f(0) = 0$   | $[-5.125,12]^D$ | $[-5.12,2]^D$   | $f(x) = \sum_{i=1}^{D} x_i^2 - 10\cos(2\pi x_i) + 10$ |
| 6   | Schwefel | $f(420.96) = 0$ | $[-500,500]^D$ | $[-500,500]^D$ | $f(x) = 418.9829xD - \sum_{i=1}^{D} x_i \sin(\sqrt{|x_i|})$ |

### Table 4. Results obtained through 50 independent runs on benchmark functions: $N_{\text{max}}$: 600, $D$: 10, $F_{\text{G}}$: 0.5, $C$: 0.9, $P$: 140, $N$: 0.8.

| No. | Algorithm | Best        | Worst       | Mean         | Standard deviation |
|-----|-----------|-------------|-------------|--------------|--------------------|
| $f_1$ | DE       | $4.93 \times 10^{-21}$ | $8.91 \times 10^{-20}$ | $3.75 \times 10^{-20}$ | $2.19 \times 10^{-20}$ |
| $f_1$ | UM-SAMDE  | $1.87 \times 10^{-78}$ | $2.90 \times 10^{-75}$ | $2.02 \times 10^{-76}$ | $4.39 \times 10^{-76}$ |
| $f_2$ | ISAMDE    | $5.59 \times 10^{-74}$ | $1.03 \times 10^{-66}$ | $5.85 \times 10^{-68}$ | $1.91 \times 10^{-67}$ |
| $f_3$ | DE        | $1.15 \times 10^{-9}$  | $1.81 \times 10^{-2}$  | $6.73 \times 10^{-4}$  | $3.31 \times 10^{-3}$  |
| $f_3$ | UM-SAMDE  | $0.00 \times 10^{0}$  | $0.00 \times 10^{0}$  | $0.00 \times 10^{0}$  | $0.00 \times 10^{0}$  |
| $f_3$ | ISAMDE    | $0.00 \times 10^{0}$  | $0.00 \times 10^{0}$  | $0.00 \times 10^{0}$  | $0.00 \times 10^{0}$  |
| $f_4$ | DE        | $2.61 \times 10^{-11}$ | $1.82 \times 10^{-10}$ | $9.73 \times 10^{-11}$ | $3.53 \times 10^{-11}$ |
| $f_4$ | MM-SAMDE  | $4.44 \times 10^{-15}$ | $4.44 \times 10^{-15}$ | $4.44 \times 10^{-15}$ | $4.44 \times 10^{-15}$ |
| $f_5$ | ISAMDE    | $8.88 \times 10^{-16}$ | $4.44 \times 10^{-15}$ | $4.37 \times 10^{-15}$ | $5.02 \times 10^{-16}$ |
| $f_6$ | DE        | $1.80 \times 10^{-1}$  | $4.76 \times 10^{-1}$  | $3.32 \times 10^{-1}$  | $7.19 \times 10^{-2}$  |
| $f_6$ | MM-SAMDE  | $0.00 \times 10^{0}$  | $9.35 \times 10^{-2}$  | $3.94 \times 10^{-2}$  | $2.26 \times 10^{-2}$  |
| $f_6$ | ISAMDE    | $7.39 \times 10^{-3}$  | $9.53 \times 10^{-2}$  | $4.05 \times 10^{-2}$  | $2.23 \times 10^{-2}$  |
| $f_7$ | DE        | $1.80 \times 10^{0}e+00$ | $2.84 \times 10^{1}$  | $2.15 \times 10^{1}$  | $3.77 \times 10^{1}$  |
| $f_7$ | MM-SAMDE  | $9.95 \times 10^{0}e+00$ | $7.96 \times 10^{2}$  | $3.30 \times 10^{2}$  | $1.81 \times 10^{1}$  |
| $f_7$ | ISAMDE    | $1.23 \times 10^{-1}e-04$ | $8.95 \times 10^{-2}$  | $3.64 \times 10^{-2}$  | $1.77 \times 10^{1}$  |
| $f_8$ | DE        | $2.20 \times 10^{-5}$  | $4.72 \times 10^{-2}$  | $1.08 \times 10^{-2}$  | $1.04 \times 10^{-2}$  |
| $f_8$ | MM-SAMDE  | $1.42 \times 10^{-14}$ | $3.70 \times 10^{-2}$  | $8.26 \times 10^{-3}$  | $9.06 \times 10^{-3}$  |

MM: multimodal; UM: unimodal; ISAMDE: improved self-adaptive mutation differential evolution; DE: differential evolution; SAMDE: self-adaptive mutation differential evolution. The optimal value of the algorithm are shown in bold in the table.

Results were achieved on MATLAB R2018a on a 2.8 GHz intel i5 processor.

**6-DOF underwater manipulator.** The joint vector corresponding to the target point of the 6-DOF underwater manipulator is $[\pi/6, -\pi/3, \pi/6, \pi/18, -\pi/3, -\pi/2]$. In the algorithm, the solution of each population represents a set of joint angles of the manipulator, corresponding to a set of posture at the end of the manipulator.

Aiming at the fixed target point, using three DE algorithms, the simulation results of the IK of the manipulator are shown in Figure 9 and Table 5.

In the table, DE-O, SAMDE-O, and ISAMDE-O represent the optimal results of the three algorithms under the original objective function. DE-I, SAMDE-I, and ISAMDE-I represent the optimal results of the three algorithms under the improved function. Best_P, Worst_P, Best_O, and Best_O represent the optimal value of the position error, the worst value of the position error, the optimal value of the orientation error, and the worst value of the orientation error, respectively. SR is the ratio of the posture error to the specified accuracy in 1000 independent runs.

The abscissa in the figure is the number of repeated positioning of the algorithm, and the ordinate is the number of algorithm iterations, position error (mm), orientation error (rad), and algorithm running time (s).

As shown in Figure 9 and Table 5, aiming at the fixed target point, for the 6-DOF underwater manipulator, before and after improving the objective function, the SR of the three DE algorithms for IK of the manipulator is 77.5%, 84.9%, 98.7% and 100%, 100%, respectively.

Aiming at the fixed target point, using Newton–Raphson method, where $E_{\text{obj}} = E_P + E_O$, the units of $E_P$ and $E_O$ are meter and radians, the IK simulation results are shown in Table 6. $N_{\text{max}}$ is 1000.
Table 6 shows that the SR of Newton–Raphson method is 88%.

For 50 random target points in the workspace, using three DE algorithms, the simulation results of the IK are shown in Figure 10 and Table 7.

As shown in Figure 10 and Table 7, before and after improving the objective function, the SR of the three DE algorithms for IK of the manipulator is 73.6%, 71.3%, 78.9% and 91.6%, 90.2%, 92.3%, respectively.

For 50 random target points in the workspace, the simulation result of Newton–Raphson method to solve the IK is provided in Table 8.

Table 8 shows that the SR of Newton–Raphson method is 76%.

Seven degrees-of-freedom Cyton Gamma 300 manipulator.
The joint vector corresponding to the target point of the Cyton Gamma 300 manipulator is \([\pi/3, \pi/2, \pi/4, \pi/2, \pi/2, \pi/6]\).

For the fixed target point, using three DE algorithms, the IK simulation results are shown in Figure 11 and Table 9.

As shown in Figure 11 and Table 9, aiming at the fixed target point, for the Cyton Gamma 300 manipulator, before

Figure 8. Convergence performance diagram of three algorithms optimized on the benchmark function. (a) Sphere, (b) Rosenbrock, (c) Ackley, (d) Griewank, (e) Rastrigin, and (f) Schwefel.

Figure 9. 6-DOF underwater manipulator fixed target point simulation results (50 independent runs): (a) Original objective function and (b) improved objective function. DOF: degree-of-freedom.

| Case   | Best-P | Worst-P | Best-O | Worst-O | SR (%) |
|--------|--------|---------|--------|---------|--------|
| DE_O   | 10^-3  | 10^-7   | 10^-3  | 10^-3   | 77.5   |
| DE_I   | 10^-3  | 10^-5   | 10^-4  | 10^-4   | 100    |
| SAMDE_O| 10^-3  | 10^-12  | 10^-4  | 10^-4   | 84.9   |
| SAMDE_I| 10^-3  | 10^-5   | 10^-5  | 10^-5   | 100    |
| ISAMDE_O| 10^-3 | 10^-7   | 10^-5  | 10^-5   | 98.7   |
| ISAMDE_I| 10^-3 | 10^-5   | 10^-5  | 10^-5   | 100    |

DOF: degree-of-freedom; ISAMDE: improved self-adaptive mutation differential evolution; DE: differential evolution; SAMDE: self-adaptive mutation differential evolution. SR is the ratio of the posture error to the specified accuracy in 1000 independent runs.
and after improving the objective function, the SR of the three DE algorithms for IK of the manipulator is 77.5%, 84.9%, 98.7% and 100%, 100%, 100%, respectively.

For 50 random target points in the workspace, using three DE algorithms, the simulation results of the IK are shown in Figure 12 and Table 10.

As shown in Figure 12 and Table 10, before and after improving the objective function, the SR of the three DE algorithms for IK of the manipulator is 65.0%, 52.4%, 21.0% and 92.8%, 91.8%, and 86.2%, respectively.

![Figure 10. Simulation results of random target points of the 6-DOF underwater manipulator: (a) Original objective function and (b) improved objective function. DOF: degree-of-freedom.](image)

![Figure 11. Simulation results of the Cyton Gamma 300 manipulator with fixed target points (50 independent runs): (a) Original objective function and (b) improved objective function.](image)

| Algorithm | DOF | Best-P | Worst-P | Best-O | Worst-O | SR (%) |
|-----------|-----|--------|---------|--------|---------|--------|
| N_L       | 6   | 10^2   | 10^-6   | 10^1   | 10^-15  | 88     |

DOF: degree-of-freedom; IK: inverse kinematic.

Table 6. Newton–Raphson algorithm fixed target point IK results.

Table 7. Statistical results of random target points of the 6-DOF underwater manipulator.

| Case      | Best-P | Worst-P | Best-O | Worst-O | SR (%) |
|-----------|--------|---------|--------|---------|--------|
| DE_O      | 10^-3  | 10^3    | 10^-15 | 10^-3   | 73.6   |
| DE_I      | 10^-5  | 10^1    | 10^-15 | 10^-3   | 91.6   |
| SAMDE_O   | 10^-3  | 10^3    | 10^-12 | 10^-2   | 71.3   |
| SAMDE_I   | 10^-6  | 10^2    | 10^-5  | 10^1    | 90.2   |
| ISAMDE_O  | 10^-3  | 10^3    | 10^-14 | 10^-2   | 78.9   |
| ISAMDE_I  | 10^-3  | 10^5    | 10^-5  | 10^-4   | 92.3   |

DOF: degree-of-freedom; ISAMDE: improved self-adaptive mutation differential evolution; DE: differential evolution; SAMDE: self-adaptive mutation differential evolution.
Discussion

The simulation results of benchmark functions in Table 4 and Figure 8 show that ISAMDE performs well in most benchmark functions both in terms of computational accuracy and convergence speed. The experimental results show that under the guidance of the optimal individual $x_{best}$, ISAMDE algorithm quickly approaches the optimal solution, which better balances the exploration and exploitation ability of the algorithm.

In the ISAMDE algorithm, the introduction of AMO enables the algorithm to switch timely between the two evolutionary modes of DE/rand/1/bin and DE/best/1/bin. The introduction of the current generation error value $\varepsilon(t)$ in AF realizes the adaptive adjustment of the search step length of the algorithm. The above parameters all achieve a good balance between algorithm exploration ability and exploitation ability. The simulation results of the IK of the 6-DOF manipulator show that among the three DE algorithms, the ISAMDE algorithm has the best performance on the IK problem.

For random points in the workspace, the SR of the ISAMDE algorithm can reach 92.3% instead of 100%, which may be related to factors such as the randomness of the swarm intelligence algorithm, improper selection of the initial point, the location of the random point too close to the boundary, and the large search space. To improve the accuracy of the algorithm, follow-up research can be carried out around how to reasonably select the location of the initial point of the algorithm and the optimization of the boundary points.

### Table 8. Newton–Raphson algorithm random points IK results.

| Algorithm | DOF | Best-P | Worst-P | Best-O | Worst-O | SR (%) |
|-----------|-----|--------|---------|--------|---------|--------|
| N_L       | 6   | $10^2$ | $10^{-10}$ | $10^1$ | $10^{-6}$ | 76     |

DOF: degree-of-freedom.

### Table 9. Statistical results of fixed target points of the Cyton Gamma 300 manipulator (50 independent runs).

| Case    | Best-P | Worst-P | Best-O | Worst-O | SR (%) |
|---------|--------|---------|--------|---------|--------|
| DE_O    | $10^{-3}$ | $10^1$ | $10^{-8}$ | $10^{-5}$ | 71.2   |
| DE_I    | $10^{-3}$ | $10^{-3}$ | $10^{-6}$ | $10^{-4}$ | 100    |
| SAMDE_O | $10^{-3}$ | $10^2$ | $10^{-15}$ | $10^0$ | 50.9   |
| SAMDE_I | $10^{-3}$ | $10^{-3}$ | $10^{-8}$ | $10^{-4}$ | 100    |
| ISAMDE_O| $10^{-3}$ | $10^2$ | $10^{-15}$ | $10^{10}$ | 45.4   |
| ISAMDE_I| $10^{-3}$ | $10^{-3}$ | $10^{-7}$ | $10^{-4}$ | 99.9   |

ISAMDE: improved self-adaptive mutation differential evolution; DE: differential evolution; SAMDE: self-adaptive mutation differential evolution.

### Table 10. Statistical results of random target points of the Cyton Gamma 300.

| Case    | Best-P | Worst-P | Best-O | Worst-O | SR |
|---------|--------|---------|--------|---------|----|
| DE_O    | $10^{-3}$ | $10^2$ | $10^{-8}$ | $10^{-4}$ | 65.0% |
| DE_I    | $10^{-3}$ | $10^1$ | $10^{-9}$ | $10^{-4}$ | 92.8% |
| SAMDE_O | $10^{-3}$ | $10^2$ | $10^{-14}$ | $10^9$ | 52.4% |
| SAMDE_I | $10^{-3}$ | $10^2$ | $10^{-16}$ | $10^9$ | 91.8% |
| ISAMDE_O| $10^{-3}$ | $10^2$ | $10^{-13}$ | $10^9$ | 21.0% |
| ISAMDE_I| $10^{-3}$ | $10^2$ | $10^{-8}$ | $10^9$ | 86.2% |

ISAMDE: improved self-adaptive mutation differential evolution; DE: differential evolution; SAMDE: self-adaptive mutation differential evolution.

### Figure 12. Simulation results of random target points of the Cyton Gamma 300. (a) Original objective function and (b) improved objective function.
The 7-DOF manipulator is a redundant manipulator. In practical applications, for different optimization goals, different constraints need to be added to find the IK of the manipulator. The ISAMDE algorithm is inferior to the 6-DOF in the inverse solution of the 7-DOF algorithm, which is related to the increase in the extreme value peaks of the objective function caused by the increase of the algorithm dimension $D$. The ISAMDE algorithm does not perform well in the solution of multimodal functions. Follow-up research can continue to optimize the algorithm for multimodal functions.

This article proposes an error-related objective function and introduces the position error $E_p$ to realize the adaptive adjustment of the orientation error weight coefficient $\omega_o$, which ensures that the position error and orientation error are always at the same magnitude. Both the 6-DOF and 7-DOF manipulator IK experimental results show that after using the improved objective function, the performance of the three DE algorithms has been greatly improved, so the improvement effect of the objective function is significant.

Conclusions
In this article, an IK method of a serial manipulator based on an ISAMDE algorithm is proposed. To better balance the exploration and exploitation ability of the mutation DE algorithm, the AMO and the error-related AF are proposed, which can enable a quick approach to the optimal solution when solving the IK of the manipulator, improve the convergence speed, avoid premature convergence, and enhance global optimization ability. The experimental results show that the proposed IK method is suitable for 6-DOF and 7-DOF serial manipulators and has the advantage of high precision and high efficiency. This provides a new research concept for the general IK method of serial manipulators.

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences under grant no. XDA22040303, the key projects supported by “the 13th five year plan” of Hefei Institute of Physical Sciences, Chinese Academy of Sciences, under grant no. KP-2019-14, and Anhui Natural Science Foundation of China under grant no. 1808085QF214.

ORCID iD
Qianqian Zhang https://orcid.org/0000-0003-3039-5712

References
1. Craig JJ. Introduction to robotics: mechanics and control, 4th ed. Beijing: China Machine Press, 2018.
2. Lynch KM and Park FC. Modern robotics: mechanics, planning, and control. Beijing: China Machine Press, 2020.
3. Baolin Y, Bingbing Y, Yushuo W, et al., eds. Modeling and numerical simulation for the kinematics of six revolute joint robot. In: 2015 2nd international conference on electrical, computer engineering and electronics, Delhi, India, June 2015, Atlantis Press.
4. Yin F, Wang YN, and Wei SNJAAS. Inverse kinematic solution for robot manipulator based on electromagnetism-like and modified DFP algorithms. Acta Autom Sin 2011; 37(1): 74–82.
5. Koker R. A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization. Inform Sci 2013; 222: 528–543.
6. Al-Mashhadany YI, ed. Inverse kinematics problem (IKP) of 6-DOF manipulator by locally recurrent neural networks (LRNNs). In: 2010 international conference on management and service science, Wuhan, China, 24–26 August 2010.
7. Khaleel H. Inverse kinematics solution for redundant robot manipulator using combination of GA and NN. Al-Khawarizmi Eng J 2018; 14: 136–144.
8. Tabandeh S, Melek WW, and Clark CM. An adaptive niching genetic algorithm approach for generating multiple solutions of serial manipulator inverse kinematics with applications to modular robots. Robotica 2010; 28: 493–507.
9. Ren Z, Wang Z, and Sun L. A global harmony search algorithm and its application to inverse kinematics problem for humanoid arm. Control Theory Appl 2012; 029(007): 867–876.
10. Rui T, Zhu J W, Zhou Y, et al. Inverse kinematics analysis of multi-DOFs serial manipulators based on PSO. J Syst Simul 2009; 21: 2930–2932.
11. Zhang L and Xiao NJSC. A novel artificial bee colony algorithm for inverse kinematics calculation of 7-DOF serial manipulators. Soft Computing 2019; 23(10): 3269–3277.
12. Xie X, Fan S, Zhou X, et al. Inverse kinematics of manipulator based on the improved differential evolution algorithm. Robot 2019; 41: 50–57.
13. Pena CA, Guzman MA, and Cardenas PF, eds. Inverse kinematics of a 6 DOF industrial robot manipulator based on bio-inspired multi-objective optimization techniques. In: Robotics & automation, Bogota, Colombia, 29–30 September 2016.
14. Bilal , Pant M, Zaheer H, et al. Differential evolution: a review of more than two decades of research. Eng Appl Artif Intell 2020; 90: 103479.
15. Leon M, Xiong N, Molina D, et al. A novel memetic framework for enhancing differential evolution algorithms via combination with alopex local search. Int J Comput Intell Syst 2019; 12: 795–808.
16. Qingfeng D and Xiaoyu Y. Research survey of differential evolution algorithms. *CAAI Trans Intell Syst* 2017; 12: 431–442.
17. Opara KR and Arabas J. Differential evolution: a survey of theoretical analyses. *Swarm Evol Comput* 2019; 44: 546–558.
18. Storn R and Price K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *J Glob Optim* 1997; 11(4): 341–359.
19. Arabas J, Szczechankiewicz A, and Wroniak T. Experimental comparison of methods to handle boundary constraints in differential evolution. In: Schaefer R, Cotta C, Kolodziej J, and Rudolph G (eds) *Parallel problem solving from nature-Ppsn Xi, Pt Ii*. Lecture Notes in Computer Science. Vol. 6239, Berlin, Heidelberg: Springer, 2010, p. 411–420.
20. Gamperle R, Muller S, and Koumoutsakos A. A parameter study for differential evolution. *Adv Intell Syst Fuzzy Syst Evol Comput* 2002; 10: 293–298.
21. Ronkkonen J, Kukkonen S, and Price KV, eds. Real-parameter optimization with differential evolution. In: *Proceedings of IEEE congress on evolutionary computation*, Edinburgh, UK, 2–5 September 2005.
22. Elsayed SM, Sarker RA, and Ray T, eds. Differential evolution with automatic parameter configuration for solving the CEC2013 competition on real-parameter optimization. In: *Evolutionary Computation*, Cancun, Mexico, 20–23 June 2013.
23. Das S, Konar A, and Chakraborty U. Two improved differential evolution schemes for faster global search. In: *Proceedings of genetic and evolutionary computation conference 2005*, Washington DC, USA, 25–29 June 2005.
24. Qin AK and Suganthan PN, eds. Self-adaptive differential evolution algorithm for numerical optimization. In: *IEEE congress on evolutionary computation*, Edinburgh, UK, 2–5 September 2005.
25. Qin AK, Huang VL, and Suganthan PN. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans Evol Comput* 2009; 13(2): 398–417.