

Time Optimal Trajectory Planning for a Robotic Manipulator Using an Improved Hippopotamus Optimization Algorithm

Shuaiqi Liu, Xiaojing Chen*

School of Electronic Information and Electrical Engineering, Yangtze University
Jingzhou 434023, China

*Corresponding author: Xiaojing Chen

Abstract: An improved Hippopotamus Optimization (IHO) algorithm is proposed to address the uneven initial population distribution and susceptibility to local optima of standard Hippopotamus Optimization (HO) in time-optimal trajectory planning for robotic manipulators. The algorithm combines Latin hypercube sampling, a nonlinear variable-step strategy, and stagnation-triggered genetic mutation. Latin hypercube sampling improves the uniformity of the initial population, the nonlinear variable-step strategy balances global exploration and local exploitation, and genetic mutation is activated when the search stagnates. Quintic B-splines are used to construct the joint trajectories of a UR5e manipulator. Segment durations are optimized to minimize the total motion time subject to joint velocity and acceleration constraints. Benchmark-function tests and trajectory-planning simulations are conducted to evaluate the proposed method. Compared with standard HO, particle swarm optimization, and a genetic algorithm, IHO achieves the shortest motion time, faster convergence, and the best run-to-run stability among the tested methods while satisfying the prescribed kinematic constraints.

Keywords: Robotic manipulator; time-optimal trajectory planning; hippopotamus optimization; quintic B-spline; Latin hypercube sampling; genetic mutation

1. INTRODUCTION

Trajectory planning directly affects the production cycle, motion smoothness, and operational safety of manufacturing systems. Joint-space planning has relatively low computational cost and permits joint constraints to be imposed directly, making it suitable for continuous tasks such as welding, material handling, assembly, and spraying. Time-optimal trajectory planning seeks to shorten task completion time while satisfying joint position, velocity, and acceleration limits, thereby improving the productivity of industrial robots [1].

Population-based optimization algorithms are widely used for this problem. Ran et al. [2] adjusted the convergence factor of the grey wolf optimizer using a cosine function, incorporated individual-best information from particle swarm optimization, and introduced Cauchy mutation and greedy selection to improve global search and escape from local optima. Zou et al. [3] combined uniform initialization with opposition-based learning and introduced a sine-cosine strategy and fuzzy adaptive inertia weights to improve the convergence speed and accuracy of the ivy algorithm. Ou et al. [4] used adaptive crossover and mutation to enhance the global search and local exploitation of a genetic algorithm. Zhao et al. [5] improved the search efficiency, global exploration, and repeatability of squirrel search through multi-population competitive iteration. Liu et al. [6] integrated chaotic mapping, imaging learning, and a golden-sine strategy into salp swarm optimization to jointly optimize motion time, energy consumption, and jerk. Wang et al. [7] improved particle swarm optimization using chaotic initialization, dynamic parameters, and genetic mutation. These studies indicate that population initialization and the balance between exploration and exploitation are central to effective trajectory optimization.

Hippopotamus Optimization (HO), proposed by Amiri et al. [8], has a simple structure, few control parameters, and diverse exploration mechanisms. Nevertheless, standard HO may exhibit uneven population initialization, random step sizes that do not suit both early and late search, and premature convergence to local minima. Improved HO variants have been applied to engineering design, feature selection, and photovoltaic prediction. Han et al. [9] enhanced global search using chaotic

mapping, a convergence factor, and opposition-based learning. Yuan et al. [10] improved feature selection using adaptive perturbation and optimal selection. Wang et al. [11] integrated Latin hypercube sampling, Jaya, and crossover-mutation mechanisms to improve photovoltaic output prediction. The present study applies an improved HO algorithm to quintic B-spline time-optimal trajectory planning for a UR5e manipulator. Segment durations are the decision variables, and total motion time is minimized under joint velocity and acceleration constraints. Latin hypercube sampling (LHS) [12] improves initial population coverage, a nonlinear variable-step strategy coordinates exploration and exploitation, and elite-guided genetic mutation is activated upon stagnation. Benchmark tests, algorithm comparisons, and independent dense-sampling checks are used to evaluate convergence, solution accuracy, stability, and trajectory feasibility.

2. KINEMATIC MODEL OF THE UR5e MANIPULATOR

2.1 UR5e Structure and D-H Parameters

The UR5e six-degree-of-freedom serial manipulator is used as the study object. It consists of six revolute joints. Figure 1 shows the manipulator and the Denavit-Hartenberg (D-H) coordinate frames used for kinematic modeling.

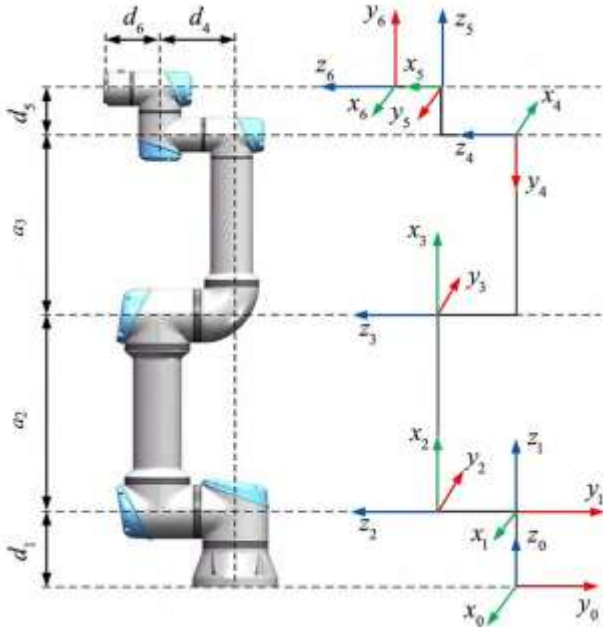


Figure 1. UR5e manipulator and D-H coordinate frames

The transformation from frame $\{i-1\}$ to frame $\{i\}$ is obtained by rotating about the z_{i-1} axis by θ_i , translating along z_{i-1} by d_i , translating along x_i by a_i , and rotating about x_i by α_i . The homogeneous transformation matrix between adjacent link frames is therefore

$${}^{i-1}T_i = \begin{bmatrix} c_i & -s_i c \alpha_i & s_i s \alpha_i & a_i c_i \\ s_i & c_i c \alpha_i & -c_i s \alpha_i & a_i s_i \\ 0 & s \alpha_i & c \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where ${}^{i-1}T_i$ is the homogeneous transformation from frame $\{i-1\}$ to frame $\{i\}$, and $i = 1, 2, \dots, 6$ denotes the joint index. The abbreviations are $c_i = \cos \theta_i$, $s_i = \sin \theta_i$, $c \alpha_i = \cos \alpha_i$, and $s \alpha_i = \sin \alpha_i$. Here, a_i is the link length, d_i is the link offset, α_i is the link twist, and θ_i is the joint angle. The adopted D-H parameters are listed in Table 1.

Table 1. D-H parameters of the UR5e manipulator

Joint i	d_i (mm)	a_i (mm)	α_i (rad)	θ_i (rad)
1	162.5	0.0	$\pi/2$	θ_1
2	0.0	-425.0	0	θ_2
3	0.0	-392.2	0	θ_3
4	133.3	0.0	$\pi/2$	θ_4
5	99.7	0.0	$-\pi/2$	θ_5
6	99.6	0.0	0	θ_6

2.2 Forward Kinematics of the UR5e

Multiplying the six adjacent transformation matrices in joint order gives the forward kinematic equation from the base frame to the end-effector frame:

$${}^0T_6 = {}^0T_1 {}^1T_2 {}^2T_3 {}^3T_4 {}^4T_5 {}^5T_6 = \begin{bmatrix} r_x & r_y & r_z & p \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

where ${}^0T_6$ is the transformation matrix of the end-effector frame. The matrix $[r_x, r_y, r_z]$ and vector $p = [p_x, p_y, p_z]^T$ describe the end-effector orientation and position, respectively. Equation (2) maps the joint variables to the end-effector pose and is used to calculate and check the end-effector motion associated with a planned joint trajectory.

3. QUINTIC B-SPLINE TRAJECTORY AND TIME-OPTIMAL MODEL

Quintic B-spline interpolation is used to construct the UR5e joint-space trajectories, ensuring higher-order continuity and reducing abrupt motion changes at segment junctions [13].

3.1 Construction of the Quintic B-Spline Trajectory

Suppose that the manipulator passes through $M + 1$ joint-space waypoints in sequence. The r th waypoint is expressed as

$$\mathbf{Q}_r = [q_{r,1}, q_{r,2}, \dots, q_{r,6}]^T \quad (3)$$

where $\mathbf{Q}_r$ denotes the r th joint-space waypoint, $r = 0, 1, \dots, M$; $q_{r,j}$ is the angle of joint j at that waypoint, and $j = 1, 2, \dots, 6$. The interpolation problem is to construct B-spline functions that pass through all prescribed waypoints.

A trajectory segment is defined between each pair of adjacent waypoints. The segment-duration vector Δt , cumulative arrival time τ_r , and total motion time T are defined as follows:

$$\Delta t = [\Delta t_1, \Delta t_2, \dots, \Delta t_M]^T \quad (4)$$

$$\tau_r = \sum_{k=1}^r \Delta t_k, \tau_0 = 0 \quad (5)$$

$$T = \tau_M = \sum_{k=1}^M \Delta t_k \quad (6)$$

where Δt_k is the duration of segment k , τ_r is the cumulative arrival time at $\mathbf{Q}_r$, $r = 1, 2, \dots, M$, and τ_0 is the initial time, set to zero.

For joint j , $M + 5$ control points are introduced. Its quintic B-spline trajectory is written as

$$q_j(t) = \sum_{i=0}^{M+4} P_{i,j} N_{i,5}(t), \quad 0 \leq t \leq T \quad (7)$$

where $q_j(t)$ is the angle of joint j at time t , $P_{i,j}$ is its i th control point, and $N_{i,5}(t)$ is the quintic B-spline basis function. The basis functions are evaluated using the Cox-de Boor recursion [14]:

$$N_{i,0}(t) = \begin{cases} 1, & u_i \leq t < u_{i+1} \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

$$N_{i,p}(t) = \frac{t - u_i}{u_{i+p} - u_i} N_{i,p-1}(t) + \frac{u_{i+p+1} - t}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(t) \quad (9)$$

where p is the basis-function degree, $p = 1, 2, \dots, 5$, and u_i is the i th element of the knot vector $\mathbf{U}$, defined by

$$\mathbf{U} = \{0, \dots, 0, \tau_1, \dots, \tau_{M-1}, T, \dots, T\} \quad (10)$$

The initial knot τ_0 and final knot τ_M are each repeated six times. Interior knots are the cumulative arrival times at the intermediate waypoints. To interpolate all waypoints and achieve smooth starts and stops, position interpolation and zero endpoint velocity and acceleration are imposed for joint j :

$$q_j(\tau_r) = q_{r,j}, \quad r = 0, 1, \dots, M \quad (11)$$

$$\dot{q}_j(0) = \dot{q}_j(T) = 0 \quad (12)$$

$$\ddot{q}_j(0) = \ddot{q}_j(T) = 0 \quad (13)$$

Arranging the $M+1$ position interpolation conditions and the four endpoint derivative conditions gives the following linear system for joint j :

$$\mathbf{A}(\Delta \mathbf{t}) \mathbf{P}_j = \mathbf{b}_j, j = 1, 2, \dots, 6 \quad (14)$$

where $\mathbf{A}(\Delta \mathbf{t})$ is the collocation matrix formed from the quintic B-spline basis functions and their derivatives, $\mathbf{P}_j$ is the joint control-point vector, and $\mathbf{b}_j$ contains the prescribed positions and endpoint conditions. Here, $M = 5$, so the five segment durations are optimized and each joint has ten control points:

$$\mathbf{P}_j = [P_{0,j}, P_{1,j}, \dots, P_{9,j}]^T \quad (15)$$

$$\mathbf{b}_j = [q_{0,j}, q_{1,j}, \dots, q_{5,j}, 0, 0, 0, 0]^T \quad (16)$$

Let the quintic basis-function row vector be $\mathbf{n}_5(t) = [N_{0,5}(t), N_{1,5}(t), \dots, N_{9,5}(t)]$. The resulting 10th-order collocation matrix has the form

$$\mathbf{A}(\Delta \mathbf{t}) = \begin{bmatrix} \mathbf{n}_5(\tau_0) \\ \mathbf{n}_5(\tau_1) \\ \mathbf{n}_5(\tau_2) \\ \mathbf{n}_5(\tau_3) \\ \mathbf{n}_5(\tau_4) \\ \mathbf{n}_5(\tau_5) \\ \dot{\mathbf{n}}_5(0) \\ \dot{\mathbf{n}}_5(T) \\ \ddot{\mathbf{n}}_5(0) \\ \ddot{\mathbf{n}}_5(T) \end{bmatrix}_{10 \times 10} \quad (17)$$

The first six rows impose waypoint interpolation, and the last four rows impose zero velocity and acceleration at the initial and final times.

To evaluate the motion constraints, the first- and second-derivative control points are obtained from $\mathbf{P}_j$ as follows:

$$P_{i,j}^{(1)} = \frac{5}{u_{i+6} - u_{i+1}} (P_{i+1,j} - P_{i,j}) \quad (18)$$

$$P_{i,j}^{(2)} = \frac{4}{u_{i+6} - u_{i+2}} (P_{i+1,j}^{(1)} - P_{i,j}^{(1)}) \quad (19)$$

The first-derivative curve uses quartic B-spline basis functions with the first and last knots removed. The second-derivative curve uses cubic basis functions with the first two and last two knots removed. Joint velocity and acceleration are then expressed as

$$\dot{q}_j(t) = \sum_{i=0}^8 P_{i,j}^{(1)} N_{i,4}(t) \quad (20)$$

$$\ddot{q}_j(t) = \sum_{i=0}^7 P_{i,j}^{(2)} N_{i,3}(t) \quad (21)$$

For each candidate duration vector, cumulative arrival times and the knot vector are first calculated. The interpolation and stationary-endpoint conditions are then assembled to determine the control points for all six joints. Joint position, velocity, and acceleration are evaluated using Eqs. (7), (18), (19), (20), and (21).

3.2 Time-Optimal Model and Constraint Handling

The segment-duration vector $\Delta \mathbf{t}$ is used as the decision variable. The objective $J(\Delta \mathbf{t})$ minimizes total motion time T :

$$J(\Delta \mathbf{t}) = T = \sum_{k=1}^M \Delta t_k \quad (22)$$

The duration constraint is $\Delta t_{k,\min} \leq \Delta t_k \leq \Delta t_{k,\max}$, where $\Delta t_{k,\min}$ and $\Delta t_{k,\max}$ are the lower and upper bounds for segment k , respectively.

Joint position, velocity, and acceleration are checked at N_s discrete times t_s . The maximum normalized constraint violations for joint j are defined as

$$\begin{aligned} g_{q,j} &= \max_{1 \leq s \leq N_s} \left(\frac{|q_j(t_s)|}{q_{j,\max}} - 1 \right) \\ g_{v,j} &= \max_{1 \leq s \leq N_s} \left(\frac{|\dot{q}_j(t_s)|}{\dot{q}_{j,\max}} - 1 \right) \\ g_{a,j} &= \max_{1 \leq s \leq N_s} \left(\frac{|\ddot{q}_j(t_s)|}{\ddot{q}_{j,\max}} - 1 \right) \end{aligned} \quad (23)$$

where N_s is the number of evaluation points (60 in this study). The quantities $q_{j,\max}$, $\dot{q}_{j,\max}$, and $\ddot{q}_{j,\max}$ are the position, velocity, and acceleration limits of joint j ; $g_{q,j}$, $g_{v,j}$, and $g_{a,j}$ are their maximum normalized violations. An exterior penalty function is used to construct the HO fitness function $f(\Delta \mathbf{t})$:

$$f(\Delta \mathbf{t}) = J(\Delta \mathbf{t}) + \rho \sum_{j=1}^6 \Phi_j \quad (24)$$

$$\Phi_j = \varphi_{q,j}^2 + \varphi_{v,j}^2 + \varphi_{a,j}^2 \quad (25)$$

$$\begin{aligned} \varphi_{q,j} &= \max(0, g_{q,j}) \\ \varphi_{v,j} &= \max(0, g_{v,j}) \\ \varphi_{a,j} &= \max(0, g_{a,j}) \end{aligned} \quad (26)$$

where $f(\Delta \mathbf{t})$ is the fitness value, $J(\Delta \mathbf{t})$ is the total-time objective, ρ is the penalty coefficient, and Φ_j is the sum of squared positive violations for joint j . The quantity $\varphi_{\mu,j}$ is the positive violation of constraint type μ for joint j , with $\mu \in \{q, v, a\}$. When all constraints are satisfied, the penalty is zero and $f(\Delta \mathbf{t}) = J(\Delta \mathbf{t})$.

4. IMPROVED HIPPOPOTAMUS OPTIMIZATION ALGORITHM

4.1 Standard Hippopotamus Optimization

HO is a population-based optimization method introduced in 2024 that imitates the behavior of hippopotamuses in their natural environment [8]. After initialization, standard HO performs three phases: position updating in a river or pond, defense against predators, and escape from predators.

The initial population is first sampled uniformly within the search bounds:

$$\begin{aligned} x_{i,j} &= lb_j + r_{i,j}(ub_j - lb_j) \\ i &= 1, 2, \dots, N, \quad j = 1, 2, \dots, D \end{aligned} \quad (27)$$

where $x_{i,j}$ is the initial position of hippopotamus i in dimension j , N is the population size, and D is the number of decision variables. The quantities lb_j and ub_j are the lower and upper bounds in dimension j , and $r_{i,j}$ denotes mutually independent uniform random numbers in $[0, 1]$.

4.1.1 Position Updating in a River or Pond

For the first half of the population, $\mathbf{X}_i$ ($i = 1, 2, \dots, \frac{N}{2}$), an adult-male candidate is generated by moving toward the dominant individual:

$$\mathbf{X}_i^m = \mathbf{X}_i + r_1(\mathbf{X}_{\text{best}} - I_1 \mathbf{X}_i) \quad (28)$$

where $\mathbf{X}_i^m$ is the adult-male candidate position, $\mathbf{X}_i$ is the current position vector of individual i , $\mathbf{X}_{\text{best}}$ is the current global best position, r_1 is uniformly distributed in $[0, 1]$, and I_1 is randomly set to 1 or 2. To generate female or juvenile candidates, the random-coefficient collection h and the time-dependent variable τ are defined as

$$h = \begin{Bmatrix} I_2 \mathbf{r}_1 + (1 - \delta_1) \\ 2\mathbf{r}_2 - \mathbf{1} \\ \mathbf{r}_3 \\ I_1 \mathbf{r}_4 + (1 - \delta_2) \\ r_5 \end{Bmatrix} \quad (29)$$

$$\tau = \exp\left(-\frac{t}{T_{\max}}\right) \quad (30)$$

where h contains five possible random coefficients; I_1 and I_2 are independently set to 1 or 2. The quantities $\mathbf{r}_1 \sim \mathbf{r}_4$ are uniform random vectors in $[0, 1]$, and r_5 is a uniform random scalar in $[0, 1]$. The variables δ_1 and δ_2 independently take 0 or 1. Here, t is the current iteration and $T_{\max}$ is the maximum number of iterations.

$$\mathbf{X}_i^f = \begin{cases} \mathbf{X}_i + h_1 \odot (\mathbf{X}_{\text{best}} - I_2 \mathbf{M}_i), & \tau > 0.6 \\ E, & \text{otherwise} \end{cases} \quad (31)$$

$$E = \begin{cases} \mathbf{X}_i + h_2 \odot (\mathbf{M}_i - \mathbf{X}_{\text{best}}), & r_6 > 0.5 \\ \mathbf{lb} + \mathbf{r}_7 \odot (\mathbf{ub} - \mathbf{lb}), & \text{otherwise} \end{cases} \quad (32)$$

Here, $\mathbf{X}_i^f$ is the female or juvenile candidate; h_1 and h_2 are independently selected from the five cases in Eq. (29); $\mathbf{M}_i$ is the mean position of a randomly selected subgroup; and I_2 randomly takes 1 or 2. The quantity r_6 is a uniform random scalar and $\mathbf{r}_7$ a uniform random vector in $[0, 1]$. The vectors $\mathbf{lb}$ and $\mathbf{ub}$ are the global lower and upper bounds, and $\odot$ denotes elementwise multiplication. When $\tau > 0.6$, the candidate is guided by the best individual and subgroup mean. When $\tau \leq 0.6$, the method switches between subgroup-difference search and uniform resampling over the full domain.

4.1.2 Defense Against Predators

For the second half of the population, $\mathbf{X}_i$ ($i = \frac{N}{2} + 1, \frac{N}{2} + 2, \dots, N$), a predator position is sampled in the search domain and its distance from the current individual is calculated:

$$\mathbf{P} = \mathbf{lb} + \mathbf{r}_8 \odot (\mathbf{ub} - \mathbf{lb}), \quad \mathbf{R}_i = |\mathbf{P} - \mathbf{X}_i| \quad (33)$$

where $\mathbf{P}$ is the predator position sampled within the global bounds, $\mathbf{r}_8$ is a uniform random vector in $[0, 1]$, and $\mathbf{R}_i = |\mathbf{P} - \mathbf{X}_i|$ is the elementwise distance from individual i to the predator. A Lévy random vector $\mathbf{L}$ is generated using the Mantegna method:

$$\mathbf{L} = 0.05 \frac{\mathbf{u} \sigma_u}{|\mathbf{v}|^{1/\beta}} \quad (34)$$

$$\sigma_u = \left[\frac{\Gamma(1 + \beta) \sin(\pi\beta/2)}{\Gamma((1 + \beta)/2) \beta 2^{(\beta-1)/2}} \right]^{1/\beta} \quad (35)$$

where $\mathbf{L}$ is D -dimensional, 0.05 is a fixed scale factor, and $\mathbf{u}$ and $\mathbf{v}$ are independent D -dimensional standard normal vectors. The function Γ is the Gamma function, β is the Lévy exponent (1.5 here), and σ_u is the scale parameter in Eq. (35). Standard HO samples $b \in [2, 4]$, $c \in [1, 1.5]$, $d \in [2, 3]$, and $\ell \in [-1, 1]$, and constructs the defense candidate as follows:

$$\kappa = \frac{b}{c - d \cos(2\pi\ell)} \quad (36)$$

$$\mathbf{X}_i^d = \begin{cases} \mathbf{L} \oplus \mathbf{P} + \kappa \mathbf{R}_i^{-1}, & f(\mathbf{P}) < f(\mathbf{X}_i) \\ \mathbf{L} \oplus \mathbf{P} + \kappa (2\mathbf{R}_i + \mathbf{r}_9)^{-1}, & f(\mathbf{P}) \geq f(\mathbf{X}_i) \end{cases} \quad (37)$$

Here, κ is the defense movement coefficient; $\mathbf{X}_i^d$ is the defense candidate; $\oplus$ denotes elementwise addition; $\mathbf{r}_9$ is a uniform random vector in $[0, 1]$; and $\mathbf{R}_i^{-1}$ and $(2\mathbf{R}_i + \mathbf{r}_9)^{-1}$ denote elementwise reciprocals. The function $f(\cdot)$ evaluates fitness. Equation (37) models defense and relocation under a predator threat. The candidate combines a Lévy random-displacement term with a distance-dependent response. Lévy perturbations produce mostly small moves with occasional long jumps, expanding global search coverage. A smaller predator distance increases the response magnitude. When the predator has better fitness, the stronger reciprocal-distance response promotes perturbation near a promising region. Otherwise, a larger denominator and a random vector reduce guidance from the poorer location. The coefficient κ further varies the defense response across individuals.

4.1.3 Escape from Predators

The escape phase (phase 3) is applied to all individuals. The local search bounds contract in proportion to $1/t$, and an escape candidate is generated near the current position:

$$\mathbf{lb}^{loc} = \frac{\mathbf{lb}}{t}, \quad \mathbf{ub}^{loc} = \frac{\mathbf{ub}}{t} \quad (38)$$

$$\mathbf{s} \in \{2\mathbf{r}_{11} - \mathbf{1}, \mathbf{r}_{12}, \mathbf{r}_{13}\} \quad (39)$$

$$\mathbf{X}_i^e = \mathbf{X}_i + r_{10} [\mathbf{lb}^{loc} + \mathbf{s} \odot (\mathbf{ub}^{loc} - \mathbf{lb}^{loc})] \quad (40)$$

where $\mathbf{lb}^{loc}$ and $\mathbf{ub}^{loc}$ are the local lower and upper bounds at iteration t , with $t = 1, 2, \dots, T_{\max}$. The quantities r_{10} and r_{13} are uniform random scalars in $[0, 1]$, $\mathbf{r}_{11}$ is a uniform random vector in $[0, 1]$, and r_{12} is a standard normal scalar. Equation (38) reduces the local neighborhood as t increases, supporting finer exploitation in later iterations. The coefficient $\mathbf{s}$ is selected with equal probability from the three forms in Eq. (39), and $\mathbf{X}_i^e$ is the escape candidate.

4.1.4 Population Updating

After a candidate is generated in any phase, the individual is updated using the greedy rule

$$\mathbf{X}_i(t+1) = \begin{cases} \mathbf{X}_i^c, & f(\mathbf{X}_i^c) < f(\mathbf{X}_i(t)) \\ \mathbf{X}_i(t), & f(\mathbf{X}_i^c) \geq f(\mathbf{X}_i(t)) \end{cases} \quad (41)$$

where $\mathbf{X}_i(t)$ and $\mathbf{X}_i(t+1)$ are the positions before and after selection, respectively, and $\mathbf{X}_i^c$ is the candidate. The label $c \in \{m, f, d, e\}$ identifies the male, female or juvenile, defense, and escape candidates. A candidate is accepted only if it has a lower fitness value than the current position.

4.2 Improved Hippopotamus Optimization

4.2.1 LHS Initialization

Standard HO initializes the population by uniform random sampling. With a finite population, this can concentrate individuals in some regions while leaving others sparsely sampled, limiting early global exploration. LHS is therefore used to improve initial coverage of the search domain.

Let N be the population size and D the decision-space dimension. The lower and upper bounds in dimension j are lb_j and ub_j , respectively. Each dimension is divided into N non-overlapping, equal-probability strata, and one sample is drawn randomly from each stratum. The initial position of individual i in dimension j is given by

$$x_{i,j}^{\text{LHS}} = lb_j + \frac{\pi_{i,j} - r_{i,j}}{N} (ub_j - lb_j) \quad (42)$$

$\pi_{i,j} \in \text{Perm}(1, 2, \dots, N)$

where $\pi_{i,j}$ is a random permutation of $1, 2, \dots, N$ in dimension j , with $\pi_{i,j}$ denoting its i th element, and $r_{i,j}$ is uniform in $[0, 1]$. Each of the N samples in a given dimension therefore occupies a different equal-probability stratum.

An initialization comparison uses the same population size and dimension as the trajectory-planning experiment: $N = 30$ and $D = 5$, corresponding to the five segment durations. Each dimension is normalized to $[0, 1]$, with $lb_i = 0$ and $ub_j = 1$ for $j = 1, 2, \dots, 5$. Each initialization therefore produces $N \times D = 150$ normalized variable components. To avoid aligning the counting bins exactly with the 30 LHS strata, $[0, 1]$ is divided into 12 equal-width bins. The theoretical uniform frequency is 12.5 per bin. LHS and standard HO random initialization are each repeated independently 30 times, and the mean bin frequencies are calculated.

Figures 2(a) and 2(b) compare the mean frequency distributions over the 30 trials. LHS exhibits small fluctuations around the theoretical uniform frequency, whereas standard random initialization shows larger variations, indicating local clustering and sparse coverage. Stratified sampling thus improves the initial coverage of a finite population and reduces local sampling bias, providing a more evenly distributed starting population for global search.

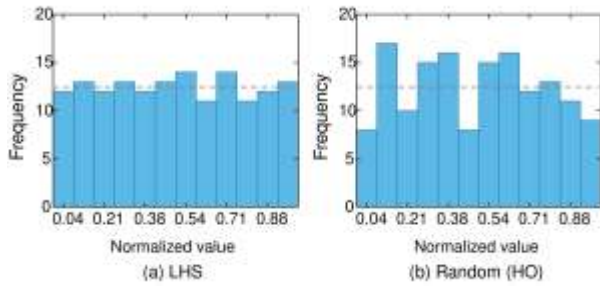


Figure 2. Initial population distributions using LHS and standard random sampling

4.2.2 Nonlinear Variable-Step Search

Candidates generated in the standard HO phases have different random step scales. Direct updates may provide insufficient early exploration or overly large late-stage movements near constraint boundaries. The proposed strategy preserves the standard HO search direction and applies a nonlinear scale factor to the displacement between the current and candidate positions:

$$\lambda(t) = \lambda_{\min} + (\lambda_{\max} - \lambda_{\min}) \left[1 - \left(\frac{t}{T_{\max}} \right)^{\gamma} \right] \quad (43)$$

$$\bar{X}_i^c = X_i + \lambda(t)(X_i^c - X_i) \quad (44)$$

where $\lambda(t)$ is the step coefficient at iteration t ; $\lambda_{\max}$ and $\lambda_{\min}$ are its upper and lower bounds; $T_{\max}$ is the iteration limit; and γ is the nonlinear decay exponent. The positions X_i^c , $\bar{X}_i^c$, and X_i denote the original HO candidate, scaled candidate, and current individual, respectively. As iteration proceeds, $\lambda(t)$ decreases nonlinearly from $\lambda_{\max}$ to $\lambda_{\min}$. Larger early steps support exploration, while smaller late steps improve local search precision.

4.2.3 Stagnation-Triggered Genetic Mutation

If the global best fitness does not improve for several consecutive iterations, the population may have concentrated near a local

optimum. Elite individuals are retained, while poorer non-elite individuals are perturbed around the global best using a decaying Gaussian term and a differential term formed from two randomly selected individuals:

$$\sigma(t) = \sigma_{\min} + (\sigma_{\max} - \sigma_{\min}) \left(1 - \frac{t}{T_{\max}} \right)^2 \quad (45)$$

$$X_i^{\text{mut}} = X_{\text{best}} + \sigma(t)(ub - lb) \odot z + F_d(X_{k_1} - X_{k_2}) \quad (46)$$

where $\sigma(t)$ is the Gaussian mutation scale, and $\sigma_{\max}$ and $\sigma_{\min}$ are its maximum and minimum values. The vector z is D -dimensional and standard normal; F_d is the differential perturbation factor. The vectors X_{k_1} and X_{k_2} correspond to two distinct individuals selected from the full population, and X_i^{mut} is the mutation candidate. The settings are $\sigma_{\max} = 0.025$, $\sigma_{\min} = 10^{-5}$, and $F_d = 0.10$. Mutation is triggered for the worst 40% of non-elite individuals when the global best fitness shows no effective improvement for eight consecutive iterations. This threshold avoids excessively frequent mutation.

$$X_i(t+1) = \begin{cases} X_i^{\text{mut}}, & f(X_i^{\text{mut}}) < f(X_i(t)) \\ X_i(t), & f(X_i^{\text{mut}}) \geq f(X_i(t)) \end{cases} \quad (47)$$

Here, $X_i(t)$ and $X_i(t+1)$ are the positions before and after mutation selection, respectively; X_i^{mut} is generated by Eq. (46); and $f(\cdot)$ is the fitness function. The mutation candidate is accepted only if it improves fitness; otherwise, the current position is retained.

4.2.4 Procedure of the Improved Algorithm

Segment durations in quintic B-spline planning are often assigned empirically, which does not ensure minimum motion time. The proposed IHO algorithm optimizes these durations under joint velocity and acceleration constraints. Its procedure is shown in Figure 3.

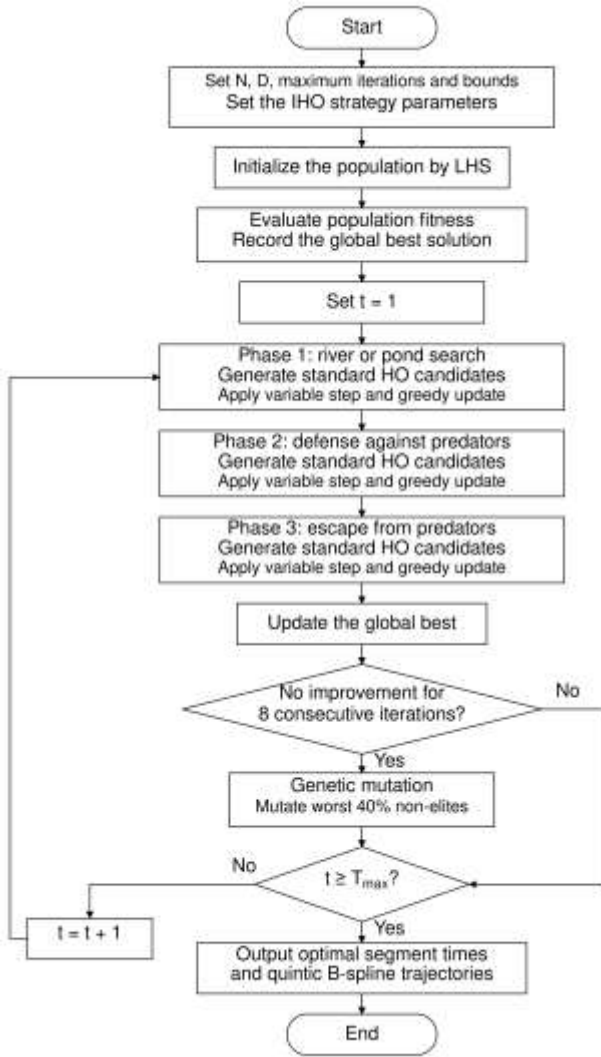


Figure 3. Flowchart of the improved HO algorithm

4.3 Benchmark Tests

4.3.1 Test Functions and Experimental Settings

Four benchmark functions, Sphere, Ackley, Griewank, and Rastrigin, are selected to assess convergence speed, solution accuracy, and run-to-run stability. IHO is compared with standard HO, particle swarm optimization (PSO) [15], and a genetic algorithm (GA) [16]. The functions and their characteristics are described below.

(1) Sphere function

$$f_1(x) = \sum_{i=1}^D x_i^2, \quad x_i \in [-100, 100] \quad (48)$$

Sphere is unimodal and separable and is used to assess basic convergence speed and solution accuracy.

(2) Ackley function

$$f_2(x) = -20 \exp \left[-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2} \right] - \exp \left[\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i) \right] + 20 + e \quad (49)$$

$x_i \in [-32, 32]$

Ackley is multimodal with many local minima and assesses the ability to converge from a broad search domain toward the optimal region.

(3) Griewank function

$$f_3(x) = 1 + \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right) \quad (50)$$

$x_i \in [-600, 600]$

Griewank is multimodal and assesses global exploration over a large domain and the ability to escape local optima.

(4) Rastrigin function

$$f_4(x) = 10D + \sum_{i=1}^D [x_i^2 - 10 \cos(2\pi x_i)] \quad (51)$$

$x_i \in [-5.12, 5.12]$

Rastrigin is multimodal and separable and assesses population diversity and global search capability.

For a consistent comparison, all algorithms use the same population size N , dimension D , and iteration limit. The historical best fitness is recorded at each iteration. The final mean and sample standard deviation are calculated over 30 independent runs. The common settings are given in Table 2.

Table 2. Settings for the benchmark-function tests

Parameter	Value
Population size N	30
Dimension D	30
Maximum iterations	150
Independent runs	30

For IHO, the upper step bound $\lambda_{\max}$, lower step bound $\lambda_{\min}$, and nonlinear exponent γ are 1.0, 0.5, and 1.8, respectively. After eight iterations without effective improvement, mutation is applied to the worst 40% of non-elite individuals. The Gaussian scale σ decreases from 0.025 to 1×10^{-5} , and $F_d = 0.10$. For PSO, the inertia weight ω decreases linearly from 0.9 to 0.4, both learning factors c_1 and c_2 are 2.0, and the maximum velocity ratio k_v is 20%. GA uses real-valued encoding, crossover probability $p_c = 0.9$, mutation probability $p_m = 1/D$, and an elite count $N_e = 1$.

4.3.2 Results and Analysis

Table 3 and Figure 4 present the final statistics and mean convergence curves for the four functions over 30 independent runs. The mean indicates solution accuracy, and the standard deviation indicates run-to-run stability. All four functions have a theoretical minimum of zero.

Table 3. Benchmark results over 30 runs (mean $\pm$ standard deviation)

Function	IHO	HO	PSO	GA
Ackley	5.62×10^{-12} $\pm 1.31 \times 10^{-12}$	6.02×10^0 $\pm 9.71 \times 10^{-1}$	4.76×10^0 $\pm 4.25 \times 10^{-1}$	2.83×10^0 $\pm 2.81 \times 10^{-1}$
Griewank	1.54×10^{-2} $\pm 1.63 \times 10^{-2}$	1.06×10^0 $\pm 5.03 \times 10^{-2}$	2.34×10^0 $\pm 5.39 \times 10^{-1}$	1.24×10^0 $\pm 7.58 \times 10^{-2}$
Rastrigin	2.05×10^{-13} $\pm 1.92 \times 10^{-13}$	2.69×10^1 $\pm 8.49 \times 10^0$	1.17×10^2 $\pm 2.90 \times 10^1$	7.64×10^1 $\pm 1.57 \times 10^1$
Sphere	7.44×10^{-22} $\pm 5.88 \times 10^{-22}$	1.69×10^1 $\pm 1.04 \times 10^1$	1.54×10^2 $\pm 7.48 \times 10^1$	3.50×10^1 $\pm 1.27 \times 10^1$

IHO has the lowest mean fitness on all four functions. Its mean values on Ackley, Rastrigin, and Sphere are 5.6152×10^{-12} , 2.0464×10^{-13} , and 7.4442×10^{-22} , respectively, approaching the theoretical optima. On Griewank, its mean fitness is 1.5419×10^{-2} , lower than those of HO, PSO, and GA. IHO also has the smallest standard deviation on each function, indicating less variation between runs. As shown in Figure 4, IHO decreases rapidly during the search and maintains high accuracy in later iterations, with faster overall convergence than the other tested methods.

The benchmark results support the effectiveness of the proposed strategies in improving solution accuracy, convergence speed, and run-to-run stability under the tested settings.

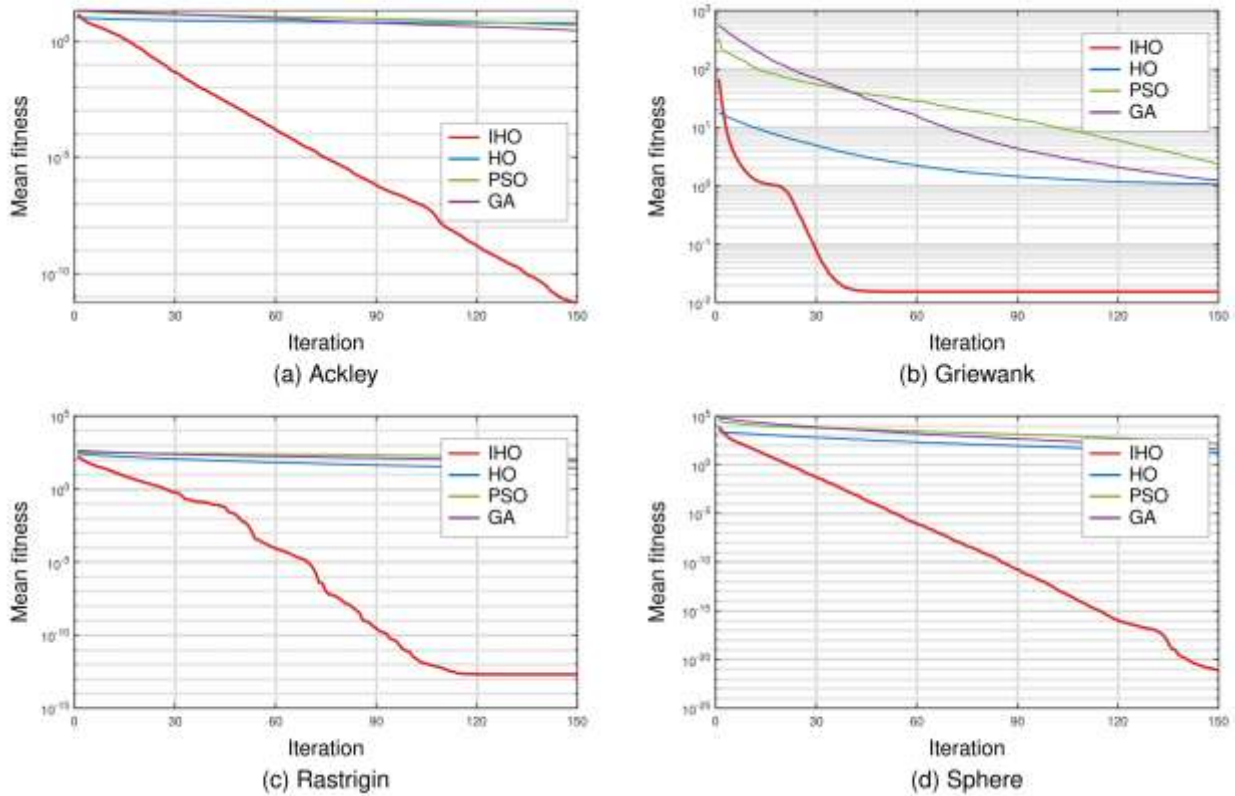


Figure 4. Mean fitness convergence curves for the benchmark functions

5. SIMULATION RESULTS AND ANALYSIS

5.1 Simulation Environment and Trajectory Settings

Simulations are conducted in MATLAB R2024a on a computer with an eighth-generation Intel Core i5 processor. Planning is performed in joint space. The path consists of a start point, four intermediate points, and an end point, yielding six joint-space waypoints $Q_0 - Q_5$ (Table 4) and five durations $\Delta t_1 - \Delta t_5$ to optimize. Quintic B-splines interpolate these waypoints, with the segment durations $\Delta t_1 - \Delta t_5$ serving as the optimization variables.

Table 4. Joint-space waypoints of the UR5e manipulator (deg)

Waypoint	q_1 (deg)	q_2 (deg)	q_3 (deg)	q_4 (deg)	q_5 (deg)	q_6 (deg)
Q0	0.0	-20.0	15.0	0.0	-10.0	0.0
Q1	25.0	-45.0	40.0	20.0	-35.0	30.0
Q2	40.0	-27.5	52.5	2.5	-10.0	42.5
Q3	55.0	-10.0	65.0	-15.0	15.0	55.0
Q4	67.5	5.0	50.0	7.5	10.0	65.0
Q5	80.0	20.0	35.0	30.0	5.0	75.0

Time reduction must preserve feasible manipulator motion. The prescribed joint position, velocity, and acceleration limits are listed in Table 5. If the quintic B-spline trajectory associated with a candidate duration vector violates any limit, the violation is included in the fitness penalty.

Table 5. Joint motion limits used in the simulation

Joint i	Position (deg)	Velocity limit (deg/s)	Acceleration limit (deg/s ²)
1	[-180,180]	120	300
2	[-180,180]	120	300
3	[-180,180]	120	300
4	[-180,180]	180	500
5	[-180,180]	180	500
6	[-180,180]	240	600

Each segment duration is bounded within [0.2, 4.0] s. The 0.2 s lower bound excludes excessively short segments that can produce large velocity and acceleration peaks. The 4.0 s upper bound excludes unnecessarily long allocations, reducing the search domain and improving optimization efficiency.

5.2 Settings for the Algorithm Comparison

IHO, HO, PSO, and GA are compared using the same fitness function, variable bounds, population size, constraint evaluation, iteration limit, and constraint-sampling density. Each algorithm is run independently 20 times. The historical best feasible time is recorded in every iteration. The best value, mean, standard deviation, and feasibility rate are used to evaluate performance. The settings are listed in Table 6.

Table 6. Settings for the manipulator trajectory-planning comparison

Parameter	Value
Decision variables	$\Delta t_1 - \Delta t_5$
Duration bounds	[0.2, 4.0] s
Population size	30
Maximum iterations	100
Independent runs	20
Optimization sampling	60 points/segment
Penalty coefficient p	1×10^4

5.3 Comparison of Algorithm Performance

Table 7 summarizes the 20 independent runs of each algorithm. Averaging the historical best feasible times across the 20 runs gives the mean convergence curves in Figure 5.

The best feasible time measures the best single-run solution, while the mean and standard deviation describe overall solution quality and repeatability. The feasibility rate is the percentage of runs that obtain a solution satisfying the velocity and acceleration constraints. HO and IHO achieve mean feasible times of 2.4117 s and 2.0572 s, respectively, compared with 2.7548 s for PSO and 2.7763 s for GA. Their standard deviations, 0.0484 s and 0.0212 s, are also smaller than those of PSO and GA. These results indicate better search performance and repeatability for HO and IHO on this problem.

Table 7. Time-optimization results for the four algorithms

Algorithm	Best feasible time (s)	Mean feasible time (s)	Standard deviation (s)	Feasible runs (%)
IHO	2.0416	2.0572	0.0212	90
HO	2.2984	2.4117	0.0484	85
PSO	2.4110	2.7548	0.3326	65
GA	2.3717	2.7763	0.3210	85

IHO reduces the mean feasible time by 14.70% relative to HO and has a smaller standard deviation, showing that shorter trajectories are obtained more consistently. Its feasibility rate is 90%, compared with 85% for HO, indicating that the proposed strategies improve the ability to find and retain feasible solutions.

As shown in Figure 5, both HO and IHO rapidly reduce the mean best feasible time during early iterations. IHO continues to improve in the middle and later stages, while HO reaches a slowly changing region earlier. PSO and GA converge more slowly and finish with longer motion times.

Under the tested settings, IHO outperforms the other three algorithms in convergence speed, solution quality, and run-to-run stability.

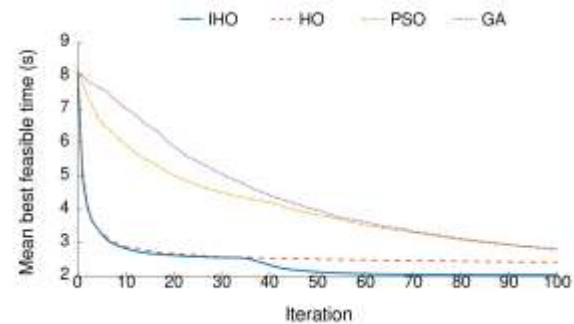


Figure 5. Mean convergence curves of the four algorithms

5.4 Optimal Trajectory and Motion-Constraint Validation

The best feasible IHO solution from the main comparison is used to reconstruct the trajectory and check the kinematic constraints. Its five segment durations are listed in Table 8.

Table 8. Segment durations of the best IHO solution (s)

Segment	Δt_1	Δt_2	Δt_3	Δt_4	Δt_5	Total
Time (s)	0.6637	0.3955	0.3284	0.2986	0.3554	2.0416

The best total motion time is 2.0416 s. The segment durations differ: Δt_1 is the longest, whereas Δt_3 and Δt_4 are shorter. This reflects the nonlinear effects of joint displacement and the velocity and acceleration constraints on the time allocation.

To check for constraint violations between the 60 optimization-stage samples per segment, the best IHO trajectory is independently evaluated at 400 uniformly spaced points per segment. The new sampling interval is 14.8% of the original interval. The optimized durations, knot vector, and control points are kept fixed. Joint velocity and acceleration are recalculated over the continuous B-spline trajectory, and their peaks are extracted, as shown in Table 9.

Table 9. Joint motion-constraint validation for the best trajectory

Joint	Peak velocity (deg/s)	Velocity limit (deg/s)	Peak acceleration (deg/s ²)	Acceleration limit (deg/s ²)
1	54.70	120.00	253.58	300.00
2	64.12	120.00	299.84	300.00
3	71.20	120.00	299.97	300.00
4	104.23	180.00	434.63	500.00
5	94.89	180.00	346.13	500.00
6	65.72	240.00	207.58	600.00

All joint velocity peaks are below their respective limits. The acceleration peaks of joints 2, 3, and 4 approach but do not exceed the specified limits. Figures 6, 7, and 8 show the joint position, velocity, and acceleration profiles for the best IHO solution.

The six joint position profiles in Figure 6 are continuous and smooth and pass through the prescribed waypoints Q_0-Q_5 . Figures 7 and 8 show continuous velocity and acceleration over the motion interval, with zero endpoint values. The sampled velocity and acceleration peaks remain within the specified limits.

IHO therefore shortens motion time while maintaining the prescribed kinematic constraints and trajectory smoothness. The benchmark tests, trajectory-planning comparisons, and independent dense-sampling check assess the search performance, time reduction, and sampled feasibility of the resulting trajectory, respectively.

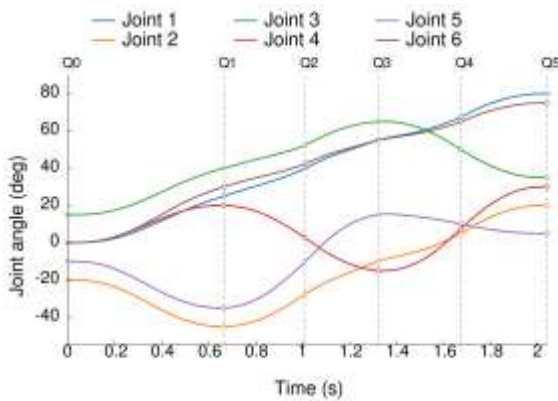


Figure 6. Joint position profiles of the IHO-optimized trajectory

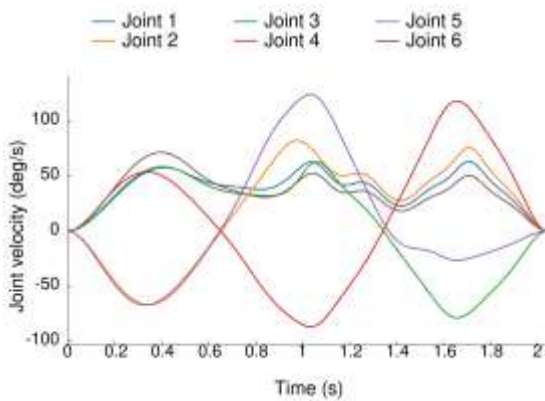


Figure 7. Joint velocity profiles of the IHO-optimized trajectory

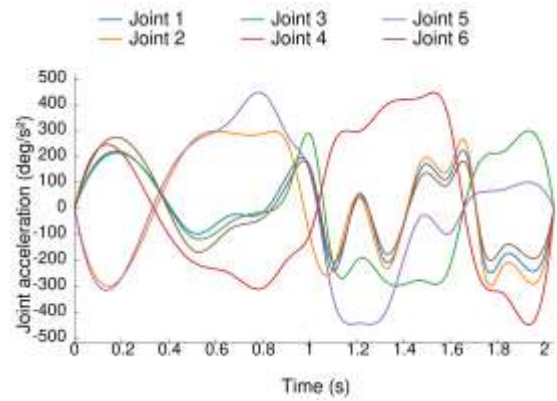


Figure 8. Joint acceleration profiles of the IHO-optimized trajectory

6. CONCLUSIONS

An improved HO algorithm combining LHS initialization, nonlinear variable-step search, and stagnation-triggered genetic mutation is proposed to address uneven population coverage, limited step-size adaptation, and premature convergence. The method is applied to quintic B-spline time-optimal planning for a UR5e manipulator. The main findings are as follows.

(1) On the four benchmark functions, IHO achieves faster convergence, higher accuracy, and better run-to-run stability than standard HO. Its mean fitness and standard deviation are also lower than those of PSO and GA, supporting the effectiveness of the combined strategies in global exploration, late-stage exploitation, and repeatability.

(2) In the manipulator comparison, IHO achieves a mean feasible time of 2.0572 s, a standard deviation of 0.0212 s, and a feasibility rate of 90%. Its overall performance is better than that of HO, PSO, and GA. The results show that the proposed strategies improve the initial search coverage and convergence behavior and yield shorter and more consistent motion times.

(3) The optimized quintic B-spline position, velocity, and acceleration profiles are continuous, with zero initial and final velocity and acceleration. The sampled velocity and acceleration peaks satisfy the prescribed limits, supporting motion smoothness and kinematic feasibility under the adopted numerical checks.

This work considers time optimization at the kinematic level. Future studies may introduce joint torque, energy, jerk, and obstacle constraints, extend the formulation to multi-objective optimization, and conduct experiments on a physical UR5e manipulator to assess practical applicability.

7. ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant No. 62473064) and a technology development project of China Petroleum Logging Co., Ltd. (Project No. 2025-1422).

8. REFERENCES

- [1] Li, L., Shang, J. Y., Feng, Y. L., et al. 2018. A review of trajectory planning for articulated industrial robots. Computer Engineering and Applications 54(5), 36–50. (in Chinese)

- [2] Ran, R., Liang, K., and Liang, K. 2025. Time-optimal trajectory planning for a robotic manipulator based on an improved grey wolf optimizer. *Modular Machine Tool and Automatic Manufacturing Technique* (12), 21–26. (in Chinese)
- [3] Zou, F. L. and Zhao, T. 2026. Time-optimal trajectory planning for a robotic manipulator based on an improved ivy algorithm. *Modular Machine Tool and Automatic Manufacturing Technique* (3), 31–36. (in Chinese)
- [4] Ou, X., Zhou, L., Zhang, Z., et al. 2022. Time-optimal trajectory planning for a serial manipulator on an electricity-meter testing platform. *Computer Engineering and Applications* 58(6), 250–255. (in Chinese)
- [5] Zhao, Y. H., Liu, D. X., Liu, Z. Y., et al. 2022. Time-optimal trajectory planning for a robotic manipulator based on a multi-population competitive squirrel search algorithm. *Journal of Zhejiang University (Engineering Science)* 56(12), 2321–2329. (in Chinese)
- [6] Liu, J. L., Huang, H. S., Fan, Q. S., et al. 2025. Multi-objective trajectory planning for robotic manipulators based on an improved salp swarm algorithm. *China Mechanical Engineering* 36(9), 2047–2056. (in Chinese)
- [7] Wang, G. R., Ni, Z. Q., Zhou, K., et al. 2025. Time-optimal trajectory planning for robotic manipulators using a multi-strategy improved particle swarm optimization algorithm. *China Mechanical Engineering* 36(5), 1044–1053. (in Chinese)
- [8] Amiri, M. H., Mehrabi Hashjin, N., Montazeri, M., et al. 2024. Hippopotamus optimization algorithm: a novel nature-inspired optimization algorithm. *Scientific Reports* 14, 5032.
- [9] Han, T., Wang, H., Li, T., et al. 2025. MHO: A modified hippopotamus optimization algorithm for global optimization and engineering design problems. *Biomimetics* 10(2), 90.
- [10] Yuan, X., Yao, L., Zhang, T., et al. 2025. An effective method for solving feature selection problems based on multi-strategy hippo optimization algorithm. *Journal of Computational Design and Engineering* 12(8), 193–251.
- [11] Wang, H., Binti Mansor, N. N., and Mokhlis, H. B. 2024. Novel hybrid optimization technique for solar photovoltaic output prediction using improved hippopotamus algorithm. *Applied Sciences* 14(17), 7803.
- [12] McKay, M. D., Beckman, R. J., and Conover, W. J. 1979. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* 21(2), 239–245.
- [13] Wang, M. X., Yan, L., and Li, Y. F. 2025. Time-optimal trajectory planning for a 6R manipulator based on an improved particle swarm algorithm. *Manufacturing Technology and Machine Tool* (2), 36–42. (in Chinese)
- [14] de Boor, C. 2001. *A Practical Guide to Splines*. Revised edition. Springer, New York, 89–90.
- [15] Kennedy, J. and Eberhart, R. 1995. Particle swarm optimization. In *Proceedings of ICNN'95—International Conference on Neural Networks*, Perth, IEEE, 1942–1948.
- [16] Holland, J. H. 1975. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press, Ann Arbor, 89–118.