Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Enhanced hippopotamus optimization algorithm for tuning proportional–integral–derivative controllers*#

Kailong MOU†1, Mengjian ZHANG†2, Deguang WANG†‡1, Ming YANG†1, Chengbin LIANG1

[1]*College of Electrical Engineering, Guizhou University, Guiyang 550025, China*

[2]*School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China*

†E-mail: gs.klmu23@gzu.edu.cn; 202111088258@mail.scut.edu.cn; dgwang@gzu.edu.cn; myang23@gzu.edu.cn

Received June 7, 2024; Revision accepted Oct. 4, 2024; Crosschecked Aug. 4, 2025

Abstract: Effectively tuning the parameters of proportional–integral–derivative (PID) controllers has persistently posed a challenge in control engineering. This study proposes enhanced hippopotamus optimization (EHO) to address this challenge. Latin hypercube sampling and adaptive lens reverse learning are used to initialize the population to improve population diversity and enhance global search. Additionally, an adaptive perturbation mechanism is introduced into the position update in the exploration phase. To validate the performance of EHO, it is benchmarked against hippopotamus optimization and four classical or state-of-the-art intelligent algorithms using the CEC2022 test suite. The effectiveness of EHO is further evaluated by applying it in tuning PID controllers for different types of systems. The performance of EHO is compared with five other algorithms and the classical Ziegler–Nichols method. Analysis of convergence curves, step responses, box plots, and radar charts indicates that EHO outperforms the compared methods in accuracy, convergence speed, and stability. Finally, EHO is used to tune the cascade PID controller for trajectory tracking in a quadrotor unmanned aerial vehicle to assess its applicability. The simulation results indicate that the integrals of the time absolute error for the position channels (x, y, z), when the system is optimized using EHO over an 80 s runtime, are 59.979, 22.162, and 0.017, respectively. These values are notably lower than those obtained by the original hippopotamus optimization and manual parameter adjustment.

Key words: PID controllers; Parameter tuning; Hippopotamus optimization; Latin hypercube sampling; Adaptive lens reverse learning; Adaptive perturbation mechanism

https://doi.org/10.1631/FITEE.2400492 **CLC number:** TP181

1 Introduction

Proportional–integral–derivative (PID) controllers are widely used in industrial systems due to their simple structure, straightforward implementation, and model independence, which contribute to their versatility and robustness. Achieving optimal performance with a PID controller necessitates precise tuning of its three parameters: proportional gain, integral gain, and derivative gain. Proper tuning can ensure desired performance characteristics, such as minimal overshoot, fast settling time, and robustness to disturbances. However, effective tuning of a PID controller presents significant challenges due to the intricate dynamics of the controlled systems and the inherent trade-offs in parameter adjustments. This study explores advanced optimization techniques to automate the PID controller tuning

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (Nos. 62341303, 62203132, and 52265066), the Guizhou Provincial Science and Technology Projects (No. [ZK[2022]Yiban103]), and the Guizhou Science and Technology Support Plan (No. 2024 General 136)

Electronic supplementary materials: The online version of this article (https://doi.org/10.1631/FITEE.2400492) contains supplementary materials, which are available to authorized users

ORCID: Kailong MOU, https://orcid.org/0009-0002-1576-8357; Mengjian ZHANG, https://orcid.org/0000-0001-8546-9972; Deguang WANG, https://orcid.org/0000-0003-4936-8773; Ming YANG, https://orcid.org/0000-0002-4470-3467; Chengbin LIANG, https://orcid.org/0000-0002-6094-018X

process, aiming to enhance control performance and robustness across various applications.

1.1 Brief literature review

Tuning PID controller parameters has been recognized as a challenging and significant issue, making it a popular research topic in both academic and industrial settings. Trial and error is one of the most intuitive and straightforward methods for tuning PID controllers. It involves manually adjusting the PID parameters, observing the system response, and iterating the process until the desired performance is achieved. Despite its simplicity, trial and error can be time-consuming, and requires a significant level of expertise to achieve optimal results. Traditional methods for tuning PID controller parameters are also widely used due to their simplicity and ease of implementation. These methods provide practical rules and guidelines to set the PID controller parameters based on the system response characteristics. The most well-known traditional methods include the Ziegler–Nichols (ZN) method (Ziegler and Nichols, 1942), Chien–Hrones–Reswick (CHR) method (Chien et al., 1952), and Cohen–Coon (CC) method (Cohen and Coon, 1953), providing general guidelines, but often requiring extensive manual adjustments and iterations to achieve acceptable performance. Despite their popularity, traditional methods have limitations, such as dependence on empirical rules and the necessity for continuous adjustments (Carlucho et al., 2020; Qi et al., 2020; Kommula and Kota, 2022). Additionally, traditional methods often fail to achieve precise regulation, leading to suboptimal performance in complex or highly nonlinear systems.

In recent years, various methods based on intelligent algorithms have emerged to optimize PID controller parameters for regulating closed-loop control systems and achieving objectives related to stability, applicability, and robustness. These methods leverage advanced computational techniques and artificial intelligence to enhance the tuning process, often outperforming traditional methods. Notable intelligent methods include expert systems (Shenassa and Khakpour, 2008), neural networks (Chen SY and Lin, 2013; Jing and Cheng, 2013; Liu et al., 2017; Preethi and Mamatha, 2022; Deng et al., 2024; Sun et al., 2024), fuzzy control (Carvajal et al., 2000; Mohan and Sinha, 2008; Wang YZ et al., 2017), and swarm intelligence algorithm (Wang YJ et al., 2015). Expert systems use artificial intelligence to mimic human decision-making. By employing predefined rules and a knowledge base, expert systems automatically adjust PID parameters, enhancing regulation efficiency, reducing reliance on human expertise, and adapting to various systems and environmental changes. However, their effectiveness depends on the quality of the knowledge base, which requires extensive domain knowledge and is difficult to update. Expert systems may not cover all scenarios, especially in complex, dynamic systems, and have high design and maintenance costs. Neural networks excel in handling high-dimensional, nonlinear, and complex systems, but face challenges like high computational costs, strong data dependency, and poor interpretability. Neural networks enhance response speed and stability without needing precise mathematical models, but depend on rule set and membership function selections, and thus require expert knowledge. Poor selection can degrade performance, and the number of rules can grow exponentially in complex systems, increasing system complexity. Fuzzy control uses human-language rules to tune PID parameters, and provides flexible responses to uncertainty and nonlinearity.

As a category of random search algorithms, swarm intelligence algorithms are inspired by biological intelligence or natural phenomena, and simulate foraging and reproduction behaviors observed in nature. These algorithms abstract these behaviors into measurable key indicators, and establish mathematical models to address various optimization problems (Reddy and Kumar, 2007; Deng et al., 2012; Parpinelli et al., 2012; Mortazavi et al., 2019; Tang et al., 2021; Gad, 2022). Classical swarm intelligence algorithms include particle swarm optimization (PSO) (Kennedy and Eberhart, 1995), ant colony optimization (ACO) (El Khoukhi et al., 2017), gray wolf optimizer (GWO) (Mirjalili et al., 2014), sparrow search algorithm (SSA) (Xue and Shen, 2020), golden jackal optimization (GJO) (Chopra and Ansari, 2022), and duck swarm algorithm (DSA) (Zhang and Wen, 2024). Kashyap and Parhi (2021) used PSO to adjust the PID gait controller of a humanoid robot, reducing stabilization time and eliminating overshoot. Mughees and Mohsin (2020) designed a magnetic levitation system and used ACO to adjust fractional order PID controller

parameters. Rana et al. (2019) employed GWO to tune the PID controller and to achieve efficient maximum power point tracking for proton exchange membrane (PEM) fuel cells. These examples demonstrate the effectiveness of swarm intelligence algorithms in tuning PID controllers owing to their powerful heuristic search strategies and excellent global search capabilities. However, original swarm intelligence algorithms often encounter issues such as slow convergence speed, low optimization precision, susceptibility to local optima, and difficulties in parameter tuning.

To enhance the performance of swarm intelligence algorithms, researchers have introduced various optimization mechanisms to improve the original algorithms. These enhancements primarily aim to increase convergence speed, optimize accuracy, and prevent the algorithms from getting trapped in local optima. By focusing on these aspects, the enhancements lead to substantial improvements in stability, accuracy, and computational efficiency of the algorithms when solving complex optimization problems (Mortazavi, 2022, 2024; Chen XY et al., 2024; Deng et al., 2024; Xie et al., 2024). Xu et al. (2024) developed a hybrid algorithm that integrates quantum particle swarm optimization and variable neighborhood search, providing fast convergence, good stability, and wide applicability. Feng et al. (2021) proposed an improved PSO to optimize PID controller parameters, enhancing the tracking accuracy of a robotic excavator electro-hydraulic position servo system. Liang et al. (2020) developed an improved genetic algorithm (GA) to optimize the intricate nonlinear fuzzy control rule, governing the relationship between input and response within a fuzzy PID controller. Mishra et al. (2020) combined PSO and GWO to tune the fractional order PID controller parameters. Despite these improvements, the enhanced algorithms still face challenges, including limited local search, lack of a global perspective, and individual clustering effects. To address these challenges in practical applications, it is essential to improve algorithm structures, adjust parameter settings, and incorporate global information.

1.2 Our contributions

Hippopotamus optimization (HO) (Amiri et al., 2024) draws inspiration from the natural behaviors of hippopotamus populations. This algorithm operates based on a three-phase model that includes position updates within their habitats, defense mechanisms against predators, and evasion tactics. Despite its success in various tests and engineering applications, HO suffers from slow convergence and a propensity to become trapped in local optima, which are common challenges in swarm intelligence algorithms. To address these limitations, this study proposes enhanced hippopotamus optimization (EHO). This improved version aims to enhance the performance of the original HO, and is specifically applied to the tuning of PID controller parameters in complex systems. The main contributions of this study are summarized as follows:

1. A novel initialization strategy for HO is designed, which combines Latin hypercube sampling (LHS) with adaptive lens reverse learning (ALRL). This strategy notably enhances the diversity of the initial population, and improves the global optimization performance of the algorithm.

2. An adaptive perturbation mechanism is integrated into the position update formula during the exploration phase of HO. This mechanism dynamically adjusts perturbations to maintain population diversity and effectively avoid local optima, thereby improving the search accuracy of the algorithm.

3. The performance of EHO is comprehensively validated on CEC2022 benchmark functions through statistical significance and convergence behavior analysis. Additionally, an ablation analysis is performed to assess the contribution of each improvement strategy on the optimization performance of EHO.

4. The effectiveness of EHO is evaluated through simulations conducted in MATLAB/Simulink. It is compared against five classical or state-of-the-art intelligent algorithms and the ZN method. The results demonstrate that EHO outperforms the compared methods in terms of accuracy, convergence speed, and stability, and showcase its superior performance in tuning PID controllers. To validate the practicality of EHO, it is applied to tune the cascade PID controller for trajectory tracking of a quadrotor unmanned aerial vehicle (UAV).

2 PID controller structure

This section presents the foundational aspects of PID controller design, beginning with an

examination of its structural components and their respective roles in shaping the controller's response. Subsequently, we discuss the critical task of defining an objective function, which is crucial for guiding the optimization process in determining optimal PID parameters (Ang et al., 2005).

The typical implementation of the PID controller is depicted in Fig. 1. The PID controller continuously calculates an error value as the difference between a desired setpoint and a measured process variable, and applies correction based on the PID terms.

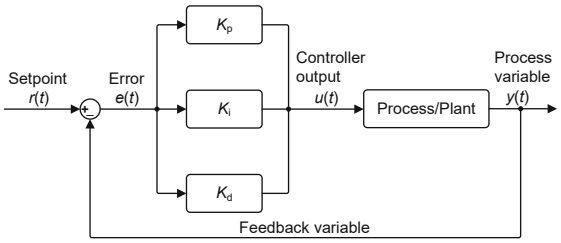

Fig. 1 Proportional–integral–derivative control structure

The PID control law can be expressed as follows:

$$u(t) = K_{\mathrm{p}} e(t) + K_{\mathrm{i}} \int e(t)\,\mathrm{d}t + K_{\mathrm{d}} \frac{\mathrm{d}e(t)}{\mathrm{d}t}, \tag{1}$$

where t is the time, $u(t)$ is the control signal (output of the PID controller), $e(t) = r(t) - y(t)$ is the error signal, which is the difference between the desired setpoint $r(t)$ and the actual process variable $y(t)$, K_{p} is the proportional gain, K_{i} is the integral gain, and K_{d} is the derivative gain.

The proportional term produces an output proportional to the current error. The proportional gain K_{p} dictates the response to the current error. A higher K_{p} value enhances the system responsiveness, but may result in overshoot and instability. The integral term concerns the accumulation of past errors. It integrates the error over time to eliminate the residual steady-state error associated with a pure proportional controller. The integral gain K_{i} determines the response based on the summation of past errors. Excessively high values can render the system unstable and oscillatory. The derivative term predicts the future trend of the error based on its rate of change. It provides a damping effect, reducing overshoot and enhancing system stability. The derivative gain K_{d} determines the response based on the rate of error change. A higher K_{d} value can aid in reducing overshoot, but may increase the susceptibility to noise.

Effective PID control relies on appropriate tuning of the K_{p}, K_{i}, and K_{d} parameters. The tuning process aims to achieve a desirable trade-off between responsiveness (speed), stability, and minimal steady-state error.

3 HO

This section provides a brief overview of HO, which draws inspiration from three key behavioral patterns observed in hippopotamus life: position update within rivers or ponds, defense strategy against the predators, and evasion tactics.

3.1 Population initialization

The population initialization involves generating the initial positions of the hippopotamuses randomly within the defined search space, given as follows:

$$\boldsymbol{x}_j = \mathbf{lb}_j + \boldsymbol{o}_1(\mathbf{ub}_j - \mathbf{lb}_j), \tag{2}$$

where $\boldsymbol{o}_1$ is an $N \times 1$ vector of random numbers with each element being drawn from a uniform distribution between 0 and 1, N is the population size, and $\mathbf{ub}_j$ and $\mathbf{lb}_j$ ($j = 1, 2, ..., m$, m is the dimension) are upper and lower bounds of the search space for each dimension, respectively.

The position matrix of the initial population is expressed as

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \\ \vdots \\ \boldsymbol{X}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_{1,1} & \boldsymbol{x}_{1,2} & \cdots & \boldsymbol{x}_{1,m} \\ \boldsymbol{x}_{2,1} & \boldsymbol{x}_{2,2} & \cdots & \boldsymbol{x}_{2,m} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{x}_{N,1} & \boldsymbol{x}_{N,2} & \cdots & \boldsymbol{x}_{N,m} \end{bmatrix}, \tag{3}$$

where $\boldsymbol{X}_i$ is the position of the i^{th} hippopotamus, and $\boldsymbol{x}_{i,j}$ is the position of the i^{th} hippopotamus in the j^{th} dimension ($i = 1, 2, ..., N$ and $j = 1, 2, ..., m$).

After generating the initial population, HO segments the population into two groups to simulate different behaviors: position update in a river or pond and defense against the predators. This segmentation strategy leverages the unique strengths of these behaviors to enhance the optimization process. The first 50% of the population is selected for position update in a river or pond. The other 50% of the population is used for defense against the predators.

3.2 Position update for hippopotamuses in a river or pond (exploration)

In HO, the population consists of different types of hippopotamuses: adult female hippopotamuses, baby hippopotamuses, adult male hippopotamuses, and the dominant male hippopotamus (leader of the herd). Typically, hippopotamuses gather closely together, with multiple female hippopotamuses positioned around the dominant male hippopotamus. Eq. (4) expresses the position calculation for male hippopotamuses $\boldsymbol{X}_i^{\mathrm{MH}}$:

$$\boldsymbol{X}_i^{\mathrm{MH}} = \boldsymbol{X}_i + r_1 \left(\boldsymbol{X}_i^{\mathrm{DH}} - I_1 \boldsymbol{X}_i\right), \tag{4}$$

where r_1 is a random number in the interval $(0,1)$, $\boldsymbol{X}_i^{\mathrm{DH}}$ is the position of the dominant male hippopotamus, and I_1 is a random integer in the interval $[1,2]$.

The variable set h is crucial for updating the positions of female or baby hippopotamuses. It is defined as a function of an index I and a random vector $\boldsymbol{o}$, with different branches being selected for subsequent location updates. The choices are determined by different conditions involving I and random components, which allow for diverse movement patterns within the search space:

$$h = \{I_2\boldsymbol{o}_2 + (\sim \alpha_1), 2\boldsymbol{o}_3 - \mathbf{1}, \boldsymbol{o}_4, I_3\boldsymbol{o}_5 + (\sim \alpha_2), r_2\}. \tag{5}$$

In Eq. (5), $\boldsymbol{o}_2$–$\boldsymbol{o}_5$ are $1 \times m$ vectors of random numbers uniformly distributed between 0 and 1, and $\mathbf{1}$ is a vector with all the elements being 1. α_1 and α_2 are random numbers drawn from a uniform distribution in the interval $[0,1]$, I_2 and I_3 are random integers in the interval $[1,2]$, and r_2 is a random number in the interval $(0,1)$.

Variable ω is defined as a probability threshold that exponentially decreases with the increasing current iteration number T, relative to the maximum iteration number $T_{\max}$:

$$\omega = \exp\left(-\frac{T}{T_{\max}}\right). \tag{6}$$

Eq. (7) describes the position calculation for female or baby hippopotamuses $\boldsymbol{X}_i^{\mathrm{FBH}}$:

$$\boldsymbol{X}_i^{\mathrm{FBH}} = \begin{cases} \boldsymbol{X}_i + h_1 \left(\boldsymbol{X}_i^{\mathrm{DH}} - I_2 \boldsymbol{X}_i^{\mathrm{MG}}\right), & \text{if } \omega > 0.6, \\ \boldsymbol{\beta}, & \text{otherwise,} \end{cases} \tag{7}$$

where h_1 is an element randomly selected from set h in Eq. (5), and $\boldsymbol{X}_i^{\mathrm{MG}}$ is the mean position of randomly selected hippopotamuses, including the current one. If $\omega > 0.6$, then the baby hippopotamus has moved away from its mother; otherwise, use $\boldsymbol{\beta}$ as an alternative position update:

$$\boldsymbol{\beta} = \begin{cases} \boldsymbol{X}_i + h_2 \left(\boldsymbol{X}_i^{\mathrm{MG}} - \boldsymbol{X}_i^{\mathrm{DH}}\right), & \text{if } r_3 > 0.5, \\ \mathbf{lb} + r_4(\mathbf{ub} - \mathbf{lb}), & \text{otherwise,} \end{cases} \tag{8}$$

where r_3 and r_4 are random numbers in the interval $(0,1)$, h_2 is an element randomly selected from set h in Eq. (5), and $\mathbf{lb}$ and $\mathbf{ub}$ are the upper and lower bounds in the m-dimensional search space, respectively. If $r_3 > 0.5$, it means that the baby hippopotamus has moved away from its mother, but is still within the group; otherwise, it is out of the group.

Eqs. (9) and (10) describe the position update for male hippopotamuses and female or baby hippopotamuses, respectively:

$$\boldsymbol{X}_i = \begin{cases} \boldsymbol{X}_i^{\mathrm{MH}}, & \text{if } F(\boldsymbol{X}_i^{\mathrm{MH}}) < F(\boldsymbol{X}_i), \\ \boldsymbol{X}_i, & \text{otherwise,} \end{cases} \tag{9}$$

$$\boldsymbol{X}_i = \begin{cases} \boldsymbol{X}_i^{\mathrm{FBH}}, & \text{if } F(\boldsymbol{X}_i^{\mathrm{FBH}}) < F(\boldsymbol{X}_i), \\ \boldsymbol{X}_i, & \text{otherwise,} \end{cases} \tag{10}$$

where F is the fitness function.

3.3 Hippopotamus defense against the predators (exploration)

The primary defense strategy employed by hippopotamuses is to deter predators from approaching them. At this stage, hippopotamuses exhibit behavior that involves approaching predators to prompt their retreat, thus effectively warding off potential threats. Eq. (11) represents the position of predators in the search space:

$$\boldsymbol{X}_{\mathrm{P}} = \mathbf{lb} + \boldsymbol{o}_6(\mathbf{ub} - \mathbf{lb}), \tag{11}$$

where $\boldsymbol{o}_6$ is a $1 \times m$ vector of random numbers between 0 and 1.

Eq. (12) expresses the distance from the i^{th} hippopotamus to the predators:

$$\boldsymbol{D} = \left|\boldsymbol{X}_{\mathrm{P}} - \boldsymbol{X}_i\right|. \tag{12}$$

If $F(\boldsymbol{X}_i^{\mathrm{HP}})$ is less than $F(\boldsymbol{X}_i)$, it indicates that the predators are close to the hippopotamus; otherwise, it indicates that the predators or intruders

are far away from the hippopotamus territory. The mathematical expression is as follows:

$$\boldsymbol{X}_i^{\mathrm{HP}}=\begin{cases}\boldsymbol{R}_{\mathrm{L}}\boldsymbol{X}_{\mathrm{P}}+\dfrac{A}{\boldsymbol{D}}, & \text{if } F(\boldsymbol{X}_i^{\mathrm{HP}})<F(\boldsymbol{X}_i),\\ \boldsymbol{R}_{\mathrm{L}}\boldsymbol{X}_{\mathrm{P}}+\dfrac{A}{2\boldsymbol{D}+\boldsymbol{v}}, & \text{otherwise,}\end{cases} \tag{13}$$

where $\boldsymbol{R}_{\mathrm{L}}$ is a $1\times m$ vector of random numbers with a Lévy distribution, and $\boldsymbol{v}$ is a $1\times m$ vector of random numbers between 0 and 1.

$$\begin{cases}A=\dfrac{f}{c-d\cos(2\pi g)},\\ \boldsymbol{R}_{\mathrm{L}}=0.05\mathrm{Levy}(\theta),\\ \mathrm{Levy}(\theta)=\dfrac{0.01\mu\sigma}{|\nu|^{\frac{1}{\theta}}},\\ \sigma=\left[\dfrac{\Gamma(1+\theta)\sin\left(\frac{\pi\theta}{2}\right)}{2^{\frac{\theta-1}{2}}\Gamma\left(\frac{1+\theta}{2}\right)\theta}\right]^{\frac{1}{\theta}}.\end{cases} \tag{14}$$

In Eq. (14), f, c, d, and g are normally distributed random numbers in the intervals $[2,4]$, $[1,1.5]$, $[2,3]$, and $[-2\pi,2\pi]$, respectively; μ and ν are random numbers obeying the normal distribution $N(0,\sigma^2)$, θ is the default constant set to 1.5, $\mathrm{Levy}(\theta)$ represents a random vector whose elements follow a Levy distribution with parameter θ, and Γ is the Gamma function.

If $F(\boldsymbol{X}_i^{\mathrm{HP}})$ is less than $F(\boldsymbol{X}_i)$, then the hunter will escape, and this hippopotamus will return to the herd; otherwise, the hippopotamus has been hunted, and another hippopotamus will replace it in the herd. The mathematical expression is as follows:

$$\boldsymbol{X}_i=\begin{cases}\boldsymbol{X}_i^{\mathrm{HP}}, & \text{if } F(\boldsymbol{X}_i^{\mathrm{HP}})<F(\boldsymbol{X}_i),\\ \boldsymbol{X}_i, & \text{otherwise.}\end{cases} \tag{15}$$

3.4 Escaping from the predators (exploitation)

When hippopotamuses encounter a group of predators or are unable to repel them through defensive actions, they seek refuge in the nearest lake or pond to avoid harm. This behavior is modeled by Eqs. (16)–(18).

The position of the hippopotamus that finds the nearest safe place is calculated as

$$\boldsymbol{X}_i^{\mathrm{HS}}=\boldsymbol{X}_i+r_5\left(\mathbf{lo}+q_1(\mathbf{hi}-\mathbf{lo})\right), \tag{16}$$

where r_5 is a random number obeying the uniform distribution in the interval $(0,1)$, $\mathbf{lo}=\mathbf{lb}/T$, $\mathbf{hi}=\mathbf{ub}/T$, and q_1 is an element randomly selected from the three scenarios of set q, which contains vector or number and is listed in Eq. (17):

$$q=\left\{2\boldsymbol{o}_7-\mathbf{1},r_6,r_7\right\}, \tag{17}$$

where $\boldsymbol{o}_7$ is a $1\times m$ vector of random numbers between 0 and 1, r_6 is a normally distributed random number in the interval $(0,1)$, and r_7 is a random number obeying the standard normal distribution in the interval $(0,1)$.

The position update formula of the hippopotamus is as follows:

$$\boldsymbol{X}_i=\begin{cases}\boldsymbol{X}_i^{\mathrm{HS}}, & \text{if } F(\boldsymbol{X}_i^{\mathrm{HS}})<F(\boldsymbol{X}_i),\\ \boldsymbol{X}_i, & \text{otherwise.}\end{cases} \tag{18}$$

4 EHO

EHO incorporates several advanced techniques to improve the performance of the original HO. These enhancements aim to increase the diversity and quality of the initial population and improve the ability of the algorithm to balance exploration and exploitation throughout the search process. EHO employs LHS and ALRL during initialization. LHS ensures a more uniform coverage of the search space compared to random sampling, and ALRL enhances the diversity and quality of the initial population. After generating an initial population of N individuals using LHS and ALRL, EHO produces a combined population size of $2N$. All $2N$ individuals are then ranked based on their fitness values. The top N fittest individuals are selected for subsequent iterations. This strategy can accelerate the convergence of the algorithm by retaining the most promising solutions while discarding less effective ones. During the developmental stage, after the hippopotamus population has moved to a safer area, an adaptive t-distribution random perturbation strategy is employed. The perturbation size adapts based on the population size: smaller populations receive larger perturbations to promote exploration, whereas larger populations receive smaller perturbations to focus on exploitation. This adaptive mechanism enhances the ability of the algorithm to escape from local optima, increasing the likelihood of finding global optima. The flowchart of EHO is depicted in Fig. 2.

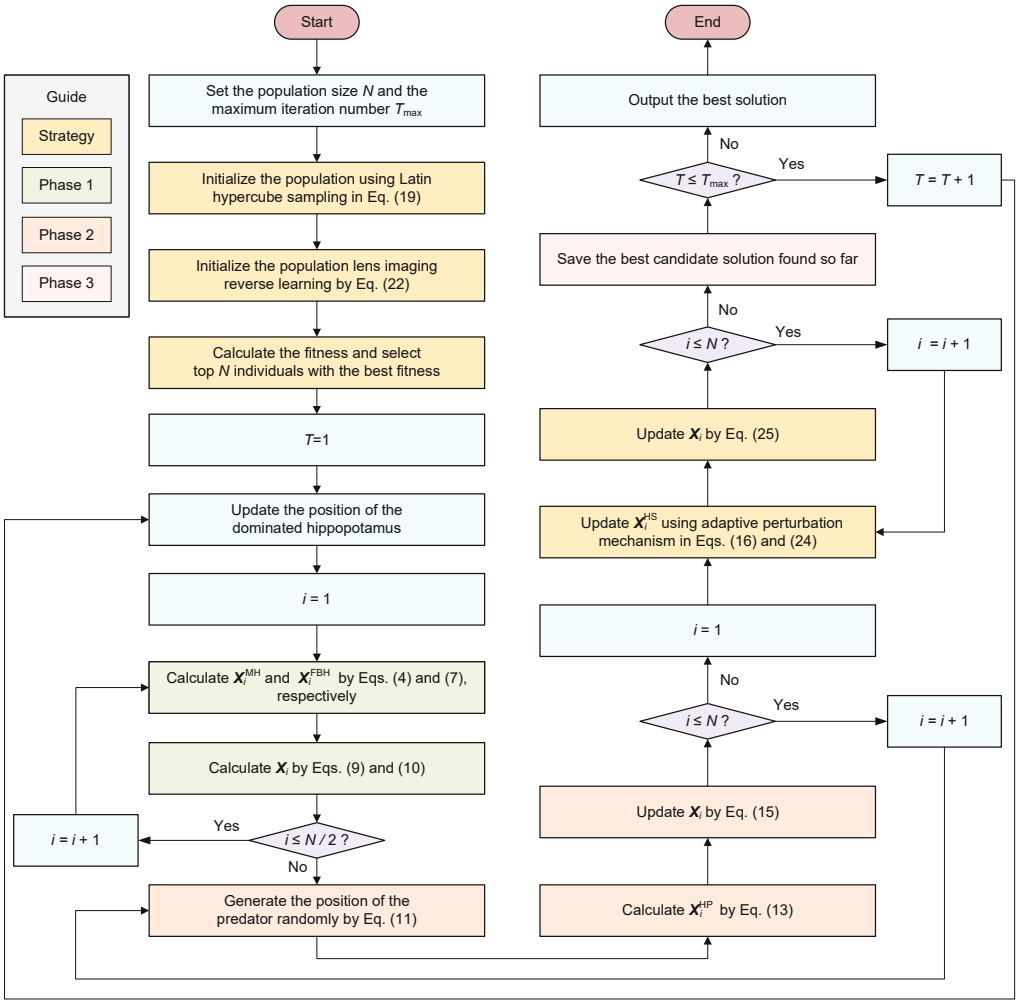

Fig. 2 Flowchart of the enhanced hippopotamus optimization

4.1 LHS

LHS is a powerful technique for generating a set of samples that are uniformly distributed within a given parameter space (Viana, 2016; Wang YP et al., 2024). In the context of optimizing PID parameters using EHO, LHS is employed to ensure that the initial population of solutions covers the search space. First, the range of values for each dimension of K_{p}, K_{i}, and K_{d} is evenly divided into several intervals. Next, one sample point is selected in each dimension to ensure that each interval contains exactly one sample point. This ensures a uniform distribution of sample points in each dimension. Then, the sample points on each dimension are randomly rearranged to introduce randomness and avoid any systematic biases. Finally, the sample points in each dimension are combined to form a sampling set, which constitutes the samples obtained through LHS.

EHO uses LHS to initialize the population positions uniformly distributed within the interval $[0, 50]$. The formula for generating sample points using LHS within this interval can be expressed as follows:

$$\boldsymbol{x}_j = (\mathbf{lb}_j - \boldsymbol{l}_j)\mathrm{LHS}(N, 1) + \boldsymbol{l}_j, \tag{19}$$

where $\boldsymbol{l}_j$ is the lower limit of the interval $[0, 50]$, which is $\mathbf{0}$ in this case with all elements being 0, and $\mathrm{LHS}(N, 1)$ is the LHS function that returns an $N \times 1$ matrix of sample points.

Eq. (19) generates sample points for each dimension of the population using LSH. By multiplying the difference between the lower bound of the dimension and the lower limit of the interval by the sampled values from LHS, and then adding the lower limit of

the interval, the equation ensures that the generated sample points are uniformly distributed within the interval $[0, 50]$.

LHS ensures that the samples are evenly distributed across the parameter space, preventing the population from clustering around certain regions and promoting a more thorough exploration. By generating a diverse set of initial solutions, LHS helps the algorithm avoid getting stuck in local optima during the initialization stage, thus improving its global search capability. The more uniform distribution of the initial solutions facilitated by LHS can lead to faster convergence and better overall performance of the algorithm.

The visual representations of the difference between the random initialization and LHS initialization are depicted in Fig. 3, highlighting the superior uniformity and coverage of the latter. This illustrates the effectiveness of LHS for population initialization in EHO.

4.2 ALRL

Traditional reverse learning strategies have a fixed reverse solution, which can hinder the algorithm's ability to escape from local optima, especially when the reverse solution is inferior to the current solution. ALRL addresses this limitation by dynamically adjusting the reverse solution. This technique can improve the diversity and quality of the initial population by expanding the search space and exploring new regions. Unlike traditional reverse learning, where the reverse solution remains fixed, ALRL adapts the reverse solution based on the current context and the individual's position within the solution space (He and Wang, 2024).

The principle of ALRL, as illustrated in Fig. 4, involves using the concept of lens imaging to dynamically adjust reverse solutions. In Fig. 4, the Y axis represents the convex lens. Eq. (20) represents the relationship between the object, its image, and the lens parameters:

$$\frac{(a+b)/2-x}{x^*-(a+b)/2}=\frac{l}{l^*}, \tag{20}$$

where x denotes the projection of object M on the X axis, x^* signifies the projection of the inverted real image M^* on the X axis, l stands for the height of object M, and l^* denotes the height of the inverted real image M^*.

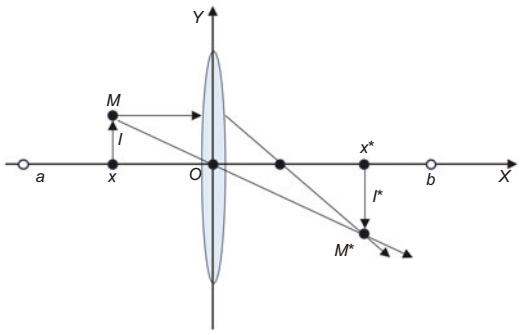

Fig. 4 Principle of adaptive lens reverse learning

This equation essentially describes the relationship between the distances of the object and its image from the midpoint of the lens. The ratio of these distances is equal to the ratio of the heights of the object and its image.

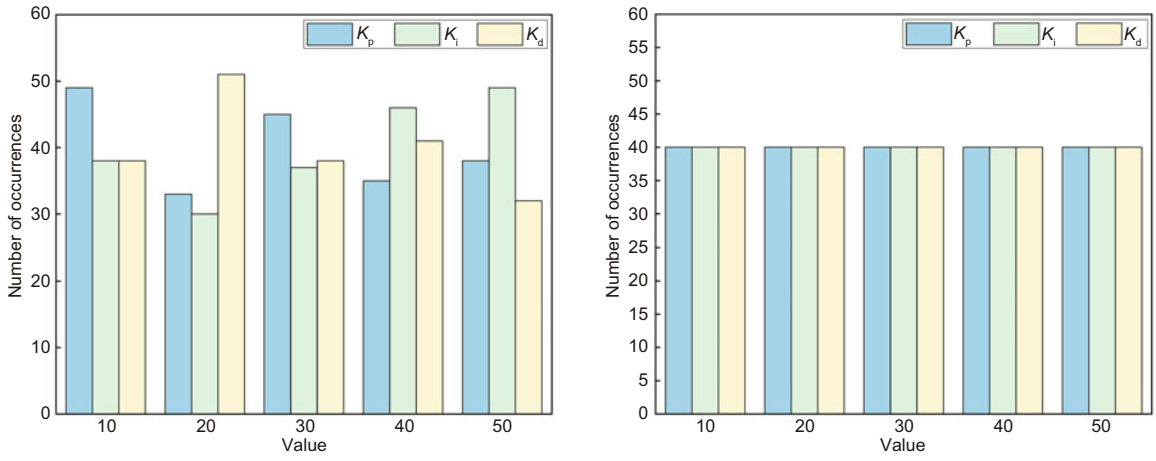

Fig. 3 Value distribution results of two initialization methods: (a) random initialization; (b) LHS initialization

Let $k = l/l^*$. Then Eq. (20) can be rewritten as

$$x^* = \frac{a+b}{2} + \frac{a+b}{2k} - \frac{x}{k}. \tag{21}$$

The parameter k determines how far the inverse solution deviates from the current solution. By adjusting its value, ALRL can dynamically modify the inverse solution, thereby enhancing the global search capability of the algorithm. In this study, the value of k is dynamically adjusted, and is set as the index of the current individual. The mathematical expression applied during the initialization phase of EHO, considering the dynamic adjustment of k, is as follows:

$$\boldsymbol{x}_i = \frac{\mathbf{ub}+\mathbf{lb}}{2} + \frac{\mathbf{ub}+\mathbf{lb}}{2i} - \frac{\boldsymbol{x}_{i,j}}{i}. \tag{22}$$

4.3 Initialization combining LHS and ALRL

To enhance the population generation during the initialization phase of HO, this study combines LHS and ALRL to generate the initial population. LHS ensures that the initial populations are uniformly distributed across the search space, reducing the chances of missing any potential solutions. ALRL increases the diversity of the initial populations, enabling the algorithm to explore a broader area of the search space. By combining LHS and ALRL and selecting the best populations based on fitness values, the algorithm starts with a high-quality initial population. This improves the efficiency of the optimization process and enhances the global performance of the algorithm.

The implementation steps of initializing the population in EHO are as follows:

1. Employ LHS to generate N initial individuals, and apply ALRL to generate another N individuals by taking the reverse solutions of the N individuals obtained through LHS.

2. Combine the N individuals generated by LHS and the N individuals generated by ALRL, resulting in a total of $2N$ individuals.

3. Evaluate the fitness values of all $2N$ individuals, and sort these individuals based on their fitness values in descending order, where individuals with higher fitness values rank higher.

4. Select the N individuals with the highest fitness values from the sorted list as the final initial population for the algorithm.

4.4 Adaptive perturbation mechanism

To address the issue of reduced population diversity and susceptibility to local optima during the later iterations of HO, this study introduces an adaptive perturbation mechanism, which helps maintain diversity in the population and improves the algorithm's ability to escape from local optima by dynamically adjusting perturbations.

The perturbation probability density function is defined as

$$p(T) = \frac{Z}{\sqrt{\chi^2/\nu_{\mathrm{DoF}}}}, \tag{23}$$

where Z has a standard normal distribution and χ^2 has a chi-square distribution with degrees of freedom ν_{DoF}.

The adaptive perturbation mechanism leverages the t-distribution to adjust the perturbation size dynamically during the optimization process (Lange et al., 1989). This approach ensures larger perturbations for broader exploration during the early iterations and smaller perturbations for finer exploitation during the later iterations.

The position update formula using the adaptive perturbation mechanism is as follows:

$$\boldsymbol{X}_i^{\mathrm{HS}^*} = \boldsymbol{X}_i^{\mathrm{HS}} + \boldsymbol{X}_i^{\mathrm{HS}} p(T), \tag{24}$$

where $\boldsymbol{X}_i^{\mathrm{HS}}$ is the current hippopotamus individual position, and $p(T)$ is the perturbation probability density function that generates a random number in the interval $[0, 1]$.

The adaptive perturbation mechanism can effectively use the relevant information within the current population. This mechanism dynamically adjusts perturbations applied to individuals based on the current iteration number T. With lower degrees of freedom ν_{DoF}, significant perturbations will occur, enhancing global exploration. As the degrees of freedom ν_{DoF} increase, a normal distribution is approximated, balancing exploration and exploitation. With higher degrees of freedom ν_{DoF}, the perturbations become smaller, focusing on local exploitation and convergence accuracy.

To preserve the information from the original population and enhance the algorithm's exploitation capability, we implement the same fitness value ranking mechanism used during initialization, which is based on fitness value selection. This ensures that only beneficial perturbations are accepted, retaining

individuals that positively impact the fitness landscape. The formula for individual selection is as follows:

$$\boldsymbol{X}_{i,\text{new}} = \begin{cases} \boldsymbol{X}_i^{\text{HS}^*}, & \text{if } F(\boldsymbol{X}_i^{\text{HS}^*}) < F(\boldsymbol{X}_i^{\text{HS}}), \\ \boldsymbol{X}_i^{\text{HS}}, & \text{otherwise.} \end{cases} \tag{25}$$

4.5 Time complexity analysis

The time complexity (Li and He, 2020) of an algorithm is a critical measure of its performance, particularly when addressing complex optimization problems. Below is a detailed analysis of the time complexities for both HO and EHO.

In HO, the time complexity for population initialization is $\mathcal{O}(N)$, where N is the population size. During the exploration and exploitation phases, the time complexity is $\mathcal{O}(N \times T_{\max})$, with $T_{\max}$ representing the maximum number of iterations. Thus, the overall time complexity of HO can be expressed as $\mathcal{O}(N \times T_{\max})$. EHO employs advanced population initialization strategies, including LHS and ALRL. The time complexity for population initialization in EHO is $\mathcal{O}(N \times \log N)$ due to the sorting mechanism used to select the top N individuals from a pool of $2N$ candidates. This sorting process typically incurs a logarithmic overhead. Despite the increased time complexity of adaptive perturbation mechanism during the exploitation phase, the exploration and exploitation phases of EHO maintain the same complexity as HO, specifically $\mathcal{O}(N \times T_{\max})$. Consequently, the overall time complexity of EHO can be expressed as $\mathcal{O}(N \times \log N + N \times T_{\max})$.

Although the time complexity of EHO increases during the population initialization phase compared to HO, this additional complexity is offset by the improved performance characteristics of EHO. The efficient initialization strategies employed in EHO facilitate fast convergence and superior optimization accuracy, making it a more effective choice for complex optimization problems.

5 Results and discussion

This section presents a comprehensive evaluation of the proposed EHO across multiple scenarios. The initial step involves evaluating EHO on CEC2022 benchmark functions in a standardized optimization setting. This evaluation establishes a baseline for the performance of EHO in comparison to other algorithms, highlighting its capabilities in terms of solution quality, convergence speed, and robustness across different types of optimization landscapes. The next phase applies EHO to tune the PID controller parameters of a two-degree-of-freedom (2-DoF) robotic system. During the tuning process, various fitness functions are explored to assess their impact on the optimization process and system performance. The analysis focuses on justifying the selection of ITAE as the preferred fitness function for this study, given its ability to balance fast response, minimal overshoot, and stable system dynamics. Finally, EHO is applied to tune the PID parameters for transfer functions representing three different types of systems. These systems vary in complexity and dynamic behavior, providing a diverse testing ground for the effectiveness of EHO. The goals are to optimize the PID parameters using EHO and compare the results against those obtained from various traditional and intelligent tuning methods. All the experiments are performed using MATLAB on a personal computer with an Intel® Core™ i5-13500 HX CPU operating at 2.50 GHz with 16 GB of RAM and a 64-bit operating system.

5.1 CEC2022 benchmark functions and experimental setup

CEC2022 (Bujok and Kolenovsky, 2022) comprises 12 benchmark functions that are categorized into four distinct groups based on their characteristics: F_1 is a unimodal function, F_2–F_5 are basic multimodal functions, F_6–F_8 are hybrid functions, and F_9–F_{12} are composition functions. These functions vary in dimensionality and complexity, providing a comprehensive platform for evaluating the efficacy of optimization algorithms. Table S1 in the supplementary materials summarizes the properties and characteristics of each function in the CEC2022 test suite.

To comprehensively assess the proposed EHO, a comparative analysis is performed with five optimization algorithms, including HO, the GOOSE algorithm (Hamad and Rashid, 2024), GJO, PSO, and GA. These algorithms are selected because they represent either classical or recently developed metaheuristic algorithms widely used in solving optimization problems. For HO and EHO, key parameters are not fixed but generated based on random initialization within a specified range. For comparative

algorithms, parameters are chosen according to standard configurations from the literature to ensure a fair comparison. Additionally, we perform a series of preliminary experiments to fine-tune these parameters, ensuring optimal performance for the specific problem context. Detailed parameters for each algorithm are given in Table S2 in the supplementary materials. All algorithms are configured with consistent parameters: fixed population size, dimensionality, and maximum iteration count of 30, 20, and 500, respectively.

5.1.1 Statistical significance analysis

Table S3 in the supplementary materials presents the optimization results of EHO and five compared algorithms across 30 runs. The performance of the algorithms is evaluated using four metrics: best, worst, mean, and standard deviation (Std). EHO achieves the optimal solution for function F_1 with a Std of 0, outperforming all the compared algorithms. This indicates that EHO is highly consistent and accurate for this type of function. EHO shows exceptional performance on function F_2, demonstrating its capability to navigate complex landscapes with multiple local optima. For function F_3, EHO ranks the highest in terms of Std, highlighting its robustness in dealing with variability in solutions. For functions F_4 and F_5, EHO excels in both the worst value and Std, indicating a reliable performance in finding good solutions even in less favorable conditions. Although EHO does not always rank the first for all metrics in multimodal functions, it maintains the results close to the top-performing algorithms and ranks the second overall. EHO ranks the third for the best value on function F_6, but outperforms all other algorithms across the remaining metrics for all hybrid functions. This implies a balanced performance across varying types of optimization landscapes. EHO demonstrates superior performance on functions F_9 and F_{10}, achieving the best and the worst values, and mean value on function F_{10}. EHO ranks the second on function F_{12}, which shows its adaptability and effectiveness across complex and composite optimization problems. However, EHO does not achieve the best runtime performance among the algorithms, due to its incorporation of four improvement strategies that prioritize solution quality over speed. This trade-off results in superior optimization performance at the cost of longer runtime. The Friedman rank test provides an average ranking for the six optimization algorithms, where EHO is ranked the highest. The performance ranking order is EHO > HO > GJO > GOOSE > PSO > GA.

EHO demonstrates higher solution stability and accuracy compared to the other five algorithms on CEC2022 benchmark functions. Its superior performance in terms of solution quality and robustness makes it an ideal choice for optimization problems, especially in real-world applications where these attributes are critical.

5.1.2 Convergence curve analysis

Fig. S1a in the supplementary materials illustrates the convergence curves of six algorithms on the unimodal function F_1. As seen, EHO quickly identifies the optimal value early in the iterations. It maintains consistent performance until the maximum iteration limit, clearly outperforming the other algorithms. The other algorithms, by comparison, converge more slowly or struggle to avoid local optima, which highlights the superior convergence speed and robustness of EHO. Figs. S1b–S1e in the supplementary materials present the convergence curves of the six algorithms on basic multimodal functions F_2–F_5. EHO consistently outperforms other algorithms by surpassing local optima and converging to the optimal value with fewer iterations. The higher optimization accuracy of EHO compared to the other algorithms further emphasizes its effectiveness in finding global optima while avoiding premature convergence.

Figs. S1f–S1l in the supplementary materials provide the convergence curves of the six algorithms on hybrid F_6–F_8 and composite F_9–F_{12} functions. EHO generally demonstrates the highest optimization accuracy and the ability to overcome local optima, leading to global optima. The only exception is function F_{12}, where EHO shows slightly lower convergence speed and optimization accuracy compared to PSO. However, for all other functions, the performance of EHO remains superior. The LHS strategy enhances the exploration phase by ensuring a diverse sampling of the search space, which speeds up early convergence. The ALRL strategy improves exploitation by allowing EHO to refine its search in promising regions, boosting convergence speed and accuracy. The fitness ranking mechanism

helps in dynamically adjusting the search direction and preventing premature convergence to local optima. The adaptive perturbation mechanism allows the algorithm to escape from local optima, improving its ability to achieve global solutions throughout the iterations.

The analysis of convergence curves across different types of functions reveals that EHO demonstrates rapid convergence, high optimization accuracy, and a low tendency to become trapped in local optima. These attributes make EHO highly effective compared to other algorithms, particularly in scenarios where both speed and accuracy are critical.

5.1.3 Ablation analysis of improvement strategies

In this subsection, we analyze the synergistic effects achieved by integrating multiple strategies into the original HO. The four strategies mentioned in Section 4, including LHS, ALRL, the fitness ranking mechanism, and adaptive perturbation mechanism, are incorporated into HO individually. The resulting modified algorithms, referred to as LHS hippopotamus optimization (LHO), ALRL hippopotamus optimization (AHO), LHS and ALRL hippopotamus optimization (LAHO), and t-distribution hippopotamus optimization (THO), are then compared against the original HO and EHO. Representative CEC2022 benchmark functions are chosen to evaluate the effectiveness of these strategies, including F_1, F_2, F_8, F_9, F_{11}, and F_{12}. The experimental setup and parameters are consistent with those outlined in Section 5.1, ensuring a fair comparison among the algorithms.

Table S4 in the supplementary materials shows the impact of each improvement strategy on the optimization performance of HO. The optimization difficulty of the algorithms increases with the increasing complexity of the functions, leading to a decrease in solution accuracy. For functions F_1 and F_2, LHO, AHO, LAHO, and THO achieve higher optimization accuracy compared to the original HO. This demonstrates that all four strategies effectively enhance the search accuracy of HO in simpler optimization tasks. For functions F_8 and F_9, AHO, LAHO, and THO outperform both HO and LHO. The ALRL strategy, fitness ranking mechanism, and adaptive perturbation mechanism show a significant improvement in balancing exploration and exploitation. These strategies enable HO to avoid local optima more effectively and achieve global optima. For functions F_{11} and F_{12}, THO excels in finding optimal values for these complex functions. The superior performance is attributed to the adaptive perturbation mechanism, which enhances the ability of HO to escape from local optima and continue exploring the search space during iterations. EHO, which integrates all four strategies, demonstrates the best performance across all benchmark functions. It achieves the highest optimization accuracy, and effectively balances the exploration and exploitation. Although incorporating a single strategy into the original HO improves optimization accuracy, convergence speed, and global search capability, relying on a single strategy is insufficient for comprehensively enhancing the performance of HO in solving complex optimization problems.

5.2 Effect analysis of objective functions

The optimization of PID parameters is a complex task due to the trade-offs between response speed, system stability, and overshoot/undershoot behavior. To effectively tune the PID parameters, it is crucial to select an appropriate objective or fitness function that reflects the control objectives. Here are some commonly used fitness functions in the optimization of PID controllers.

1. Integral of the time absolute error (ITAE):

$$\mathrm{ITAE} = \int_0^\infty t\,|e(t)|\,\mathrm{d}t. \tag{26}$$

2. Integral time squared error (ITSE):

$$\mathrm{ITSE} = \int_0^\infty te^2(t)\mathrm{d}t. \tag{27}$$

3. Integral of the absolute error (IAE):

$$\mathrm{IAE} = \int_0^\infty |e(t)|\,\mathrm{d}t. \tag{28}$$

4. Integral of the squared error (ISE):

$$\mathrm{ISE} = \int_0^\infty e^2(t)\mathrm{d}t. \tag{29}$$

To justify the choice of ITAE in this study, a 2-DoF robot system (Silva et al., 2023) is modeled using the transfer function:

$$G(s) = \frac{8.32s^2 + 179.7s + 1754}{s^3 + 5.119s^2 + 253.5s + 23.15}, \tag{30}$$

where s is a complex frequency domain variable, derived from the Laplace transformation.

To optimize the PID parameters for this system, the simulation model using MATLAB/Simulink is created, and its block diagram is shown in Fig. 5. The PID parameters and various response metrics for each fitness function are summarized in Table S5 in the supplementary materials. The convergence curves of optimization processes and the step responses using four optimized PID controllers are presented in Fig. S2 in the supplementary materials. It is evident that when ITAE is used as the fitness function, EHO demonstrates fast convergence, high search accuracy, good robustness, and stable system response. To sum up, using ITAE as the fitness function for optimizing PID parameters provides a balanced approach that accounts for both steady-state accuracy and dynamic performance. This ensures that the control system is fine-tuned to achieve optimal responses.

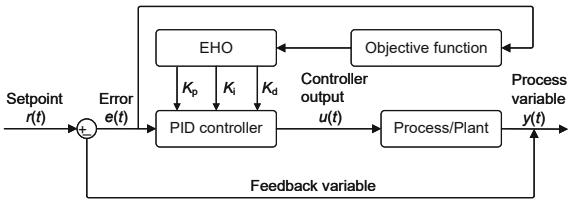

Fig. 5 Block diagram of the proportional–integral–derivative controller optimized by the enhanced hippopotamus optimization

5.3 Typical system simulation

To validate the feasibility of EHO for tuning PID parameters, we compared it against the HO, GOOSE, GJO, PSO, GA, and ZN methods. A circuit simulation model was designed using MATLAB/Simulink, as depicted in Fig. 6. This model includes the benchmark systems, PID controllers, and various performance measurement blocks.

The following benchmark transfer functions were used for simulations:

1. Overdamped system

$$G_1(s) = \frac{1}{s^2 + 4s + 4}. \tag{31}$$

2. Underdamped system

$$G_2(s) = \frac{10}{0.04s^3 + 0.54s^2 + 1.5s + 1}. \tag{32}$$

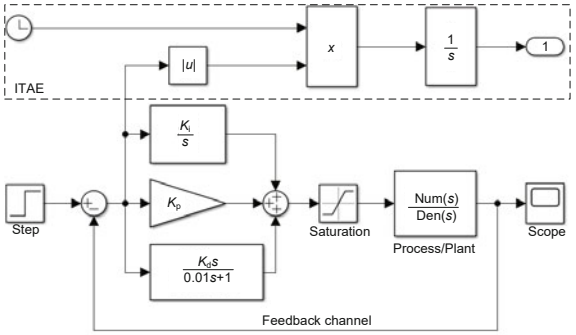

Fig. 6 A circuit simulation model

3. Unstable system

$$G_3(s) = \frac{s + 2}{s^4 + 8s^3 + 4s^2 - s + 0.4}. \tag{33}$$

Each algorithm initialized a population of 50 individuals, and ran for a maximum of 100 iterations. The ranges for K_{p}, K_{i}, and K_{d} were $[0, 50]$. The saturation ranges for $G_1(s)$, $G_2(s)$, and $G_3(s)$ were $[-5, 5]$, $[-5, 5]$, and $[-10, 10]$, respectively. The input signal was a unit step signal, and the simulation duration was 20 s. The parameter configurations for the six algorithms are listed in Table S2 in the supplementary materials. For each algorithm, 10 independent experiments were conducted.

5.3.1 Convergence behavior analysis

Fig. 7 presents the convergence curves for six different algorithms optimizing the PID controller applied to transfer functions $G_1(s)$, $G_2(s)$, and $G_3(s)$. Compared to HO, EHO exhibits faster convergence and lower fitness throughout the iteration process. EHO converges more quickly than the other algorithms, indicating that it can reach near-optimal solutions in fewer iterations. This efficiency is primarily due to the combination of LHS and ALRL, ensuring a more uniform and diverse initial population. The initial population screening based on fitness value ranking helps in selecting the most promising individuals, accelerating the convergence of the algorithm by starting the optimization process with a better initial population. EHO achieves lower fitness compared to the other algorithms, suggesting higher search accuracy and better optimization performance. The adaptive perturbation mechanism allows EHO to balance between the global exploration in the early stages and local exploitation in the later stages of the optimization process. By adapting the

degree of perturbations based on the iteration number, the algorithm maintains diversity and improves the convergence accuracy.

Overall, the convergence curves in Fig. 7 illustrate that EHO not only outperforms the other algorithms in terms of convergence speed, but also achieves significantly better search accuracy across all tested systems. This indicates that EHO improves substantial performance in optimizing PID controller parameters, making it a robust and efficient choice for PID tuning in various control systems.

5.3.2 Step response analysis

Fig. 8 presents the step responses of $G_1(s)$, $G_2(s)$, and $G_3(s)$ using six algorithms. Statistical results including overshoot, settling time, peak time, mean, and Std for $G_1(s)$, $G_2(s)$, and $G_3(s)$ are summarized in Tables 1–3, respectively. Overshoot represents the maximum peak value of the system response curve as a percentage over the desired response. Settling time is the time required for the system to remain within a certain error band around the final value, indicating how quickly the system stabilizes. Peak time is the time when the system reaches its first peak. Mean of ITAE is a measure of the overall system performance, with lower values indicating better performance. Std of ITAE indicates the consistency of the algorithm. A lower Std suggests more stable performance on multiple runs.

For $G_1(s)$, EHO achieved an overshoot of only 0.11%, which is slightly higher than that of the GA and ZN methods. However, the GA and ZN methods exhibit the slowest settling among the compared methods, indicating that EHO outperforms the GA and ZN methods. Although EHO does not have the shortest settling time or peak time, the difference compared to GJO is only a matter of milliseconds, which is negligible for an overdamped system with inherently slow response.

For $G_2(s)$, EHO achieved an overshoot of 0.62% and the shortest settling time of 0.774 s among the compared methods. Although its peak time ranks fourth, the compared methods with lower peak time have the overshoot $\geq 40\%$, which is very dangerous

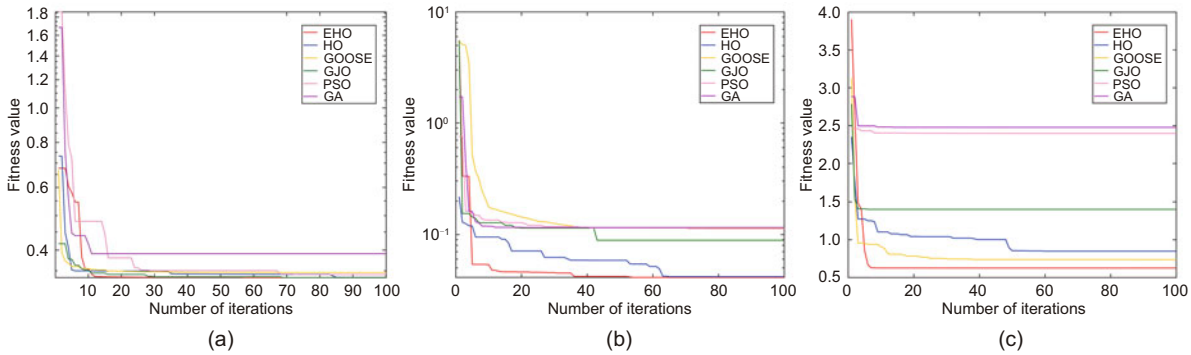

Fig. 7 ITAE convergence curves of six algorithms for $G_1(s)$ (a), $G_2(s)$ (b), and $G_3(s)$ (c)

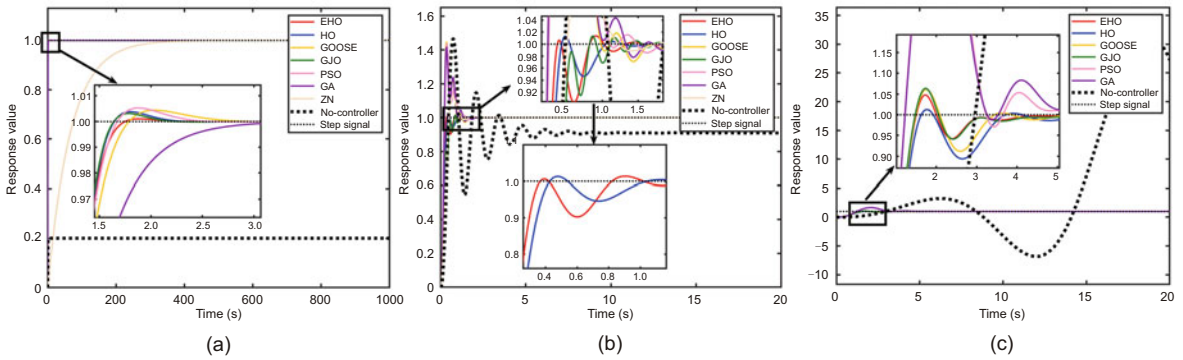

Fig. 8 Comparison of step responses using six algorithms and the ZN method for $G_1(s)$ (a), $G_2(s)$ (b), and $G_3(s)$ (c)

Table 1 Statistical results of six algorithms and the ZN method for $G_1(s)$

Method	K_p	K_i	K_d	Overshoot (%)	Settling time (s)	Peak time (s)	Mean of ITAE	Std of ITAE
EHO	49.894	5.539	7.875 00	0.110	1.498	1.868	**0.337 02**	**0.001 85**
HO	41.546	5.554	6.029 00	0.390	1.485	1.802	0.339 26	0.005 37
GOOSE	15.787	5.524	3.024 00	0.450	1.565	2.053	0.390 06	0.076 60
GJO	50.000	5.552	6.953 00	0.320	**1.480**	**1.772**	0.343 81	0.011 98
PSO	25.045	5.552	4.092 00	0.550	1.508	1.868	0.338 88	0.002 34
GA	38.755	5.308	12.956 00	0.004	1.912	5.140	0.461 36	0.318 44
ZN	0.016	0.059	0.001 44	**0**	263.084			

The best results are in bold

Table 2 Statistical results of six algorithms and the ZN method for $G_2(s)$

Method	K_p	K_i	K_d	Overshoot (%)	Settling time (s)	Peak time (s)	Mean of ITAE	Std of ITAE
EHO	3.115	0.460	0.604	**0.62**	**0.774**	0.391	**0.043 65**	**0.001 63**
HO	1.920	0.437	0.405	1.34	0.924	0.479	0.044 20	0.001 82
GOOSE	5.714	18.377	0.922	44.3	1.520	0.371	0.650 42	1.584 62
GJO	8.557	0.410	1.724	1.57	1.059	0.864	0.110 65	0.007 91
PSO	4.550	12.026	0.721	43.64	1.056	0.383	0.136 11	0.068 56
GA	5.933	16.476	0.919	41.65	1.477	**0.350**	1.823 78	2.205 39
ZN	0.850	0.467	0.156	12.52	1.111	0.716		

The best results are in bold

Table 3 Statistical results of six algorithms and the ZN method for $G_3(s)$

Method	K_p	K_i	K_d	Overshoot (%)	Settling time (s)	Peak time (s)	Mean of ITAE	Std of ITAE
EHO	47.710	0.216	41.973	4.93	**2.811**	**1.726**	**0.630 12**	**0.008 24**
HO	22.971	0.195	21.809	**1.41**	3.438	1.765	0.663 77	0.065 04
GOOSE	29.771	0.211	26.381	6.45	3.208	1.771	2.157 45	0.609 39
GJO	50.000	0	43.188	6.54	2.840	1.745	0.697 41	0.234 73
PSO	50.000	29.612	43.647	69.60	4.495	2.090	1.748 38	0.812 22
GA	38.657	21.279	33.451	66.55	4.823	2.090	1.485 19	0.850 02
ZN								

The best results are in bold. We cannot obtain the results for PID parameter tuning using the ZN method for $G_3(s)$ because of instability

for underdamped systems. It is generally known that excessive overshoot will lead to severe instability. Overall, EHO demonstrates the best performance on $G_2(s)$.

In industrial control systems, the control time for $G_3(s)$ is crucial to quickly stabilize the system and reduce economic losses. EHO has the shortest settling time and peak time among the compared methods, with an overshoot of only 4.93%. Particularly, the ZN method is an empirical approach that is generally more effective for the simpler, lower-order systems, such as first- and second-order systems. It relies on accurately determining the critical gain and critical oscillation period to adjust PID parameters. However, for the high-order system $G_3(s)$, which has complex pole distributions and exhibits nonlinear behavior, the ZN method becomes unreliable. During the simulations, we observe that $G_3(s)$ has two poles located in the right half-plane, indicating instability. This instability makes it impossible to achieve the required oscillation waveforms (e.g., a 4:1 or 10:1 ratio), which are fundamental for the ZN method. Consequently, we cannot obtain the results for PID parameter tuning using the ZN method for $G_3(s)$.

Overall, EHO exhibits superior performance on different types of systems, and demonstrates its robustness and effectiveness in PID parameter tuning. Its ability to balance the overshoot, settling time, and peak time makes it particularly suitable for a variety of control applications. The combination of LHS, ALRL, and the adaptive perturbation mechanism significantly enhances its optimization capabilities, ensuring better convergence, stability, and overall performance compared to traditional and other intelligent tuning methods.

5.3.3 Box plot analysis

The performance of EHO and the compared algorithms is assessed using box plots in Fig. 9, offering a visual representation of the distribution and variability of the ITAE.

For $G_1(s)$, the median for EHO is 0.337, which

is second only to 0.336 achieved by GJO. The interquartile range (IQR) for EHO is also narrow, second only to that of PSO. Notably, EHO exhibits the smallest range between its maximum and minimum values, indicating the least variability and highest stability among the compared algorithms.

For $G_2(s)$, EHO not only achieves the smallest median, but also has the smallest range between the maximum and minimum values among the compared algorithms. Although its IQR is larger than that of HO, EHO still demonstrates superior performance by maintaining a small overall range, which suggests consistent and reliable optimization results.

For $G_3(s)$, EHO excels by having the smallest median, the second smallest IQR, and the smallest range between maximum and minimum values. This comprehensive robustness underscores the ability of EHO to handle a wide variety of system dynamics effectively.

The box plot analysis confirms that EHO consistently outperforms the other algorithms across all the tested systems. This superior performance, characterized by low median and minimal variability, highlights the effectiveness and reliability of EHO in optimizing PID controller parameters.

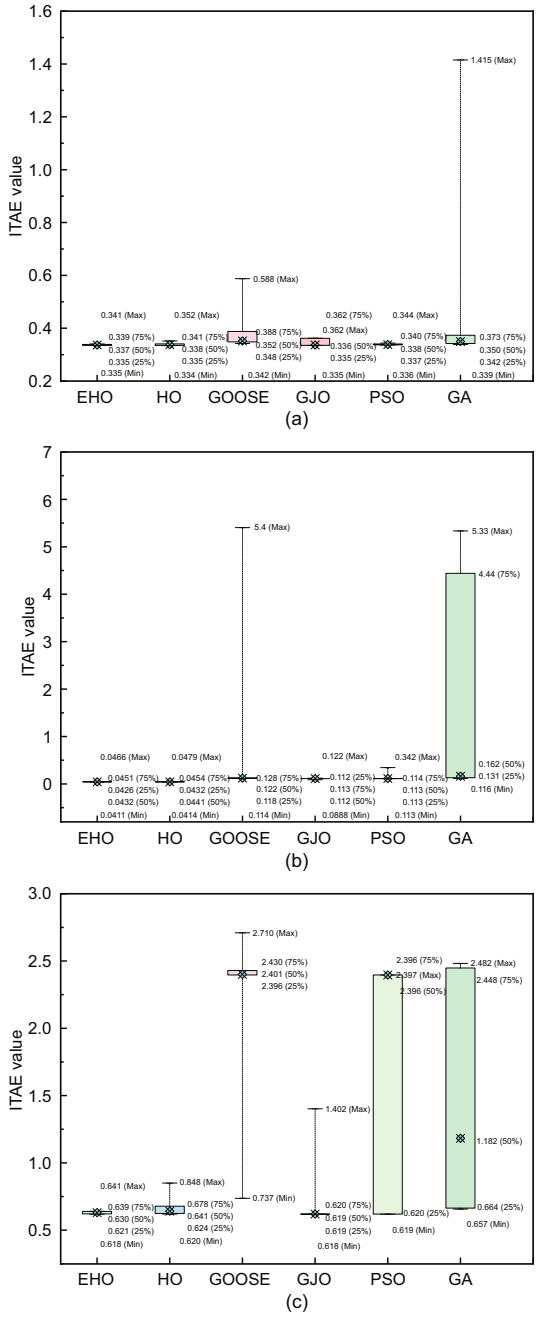

Fig. 9 Box plots of six algorithms for $G_1(s)$ (a), $G_2(s)$ (b), and $G_3(s)$ (c)

5.3.4 Radar chart analysis

To compare the effectiveness of different tuning methods more intuitively, we rank the optimization results of the compared methods, and use radar charts to visually display the rankings. Fig. 10 presents these radar charts, showing the performance rankings of six algorithms and ZN method on $G_1(s)$, $G_2(s)$, and $G_3(s)$. The endpoints of the radar chart, arranged in a clockwise direction, represent the mean, Std, overshoot, peak time, and settling time. EHO demonstrates a significant advantage over the other methods, indicating a substantial performance improvement. Moreover, to further illustrate the performance of EHO, we average the rankings of six algorithms and the ZN method on $G_1(s)$, $G_2(s)$, and $G_3(s)$ to derive a comprehensive ranking. As shown in Fig. 10, EHO ranks the first among all the compared methods.

By using radar charts, we provide an intuitive and comprehensive comparison of the algorithms, clearly demonstrating that EHO is the most effective method for PID controller tuning on different system types. This comprehensive performance improvement indicates the robustness and efficiency of EHO in optimizing PID parameters.

5.4 Discussion

EHO incorporates several improvement strategies to enhance its ability to escape from local optima and improve convergence speed. Although these strategies increase the overall complexity of

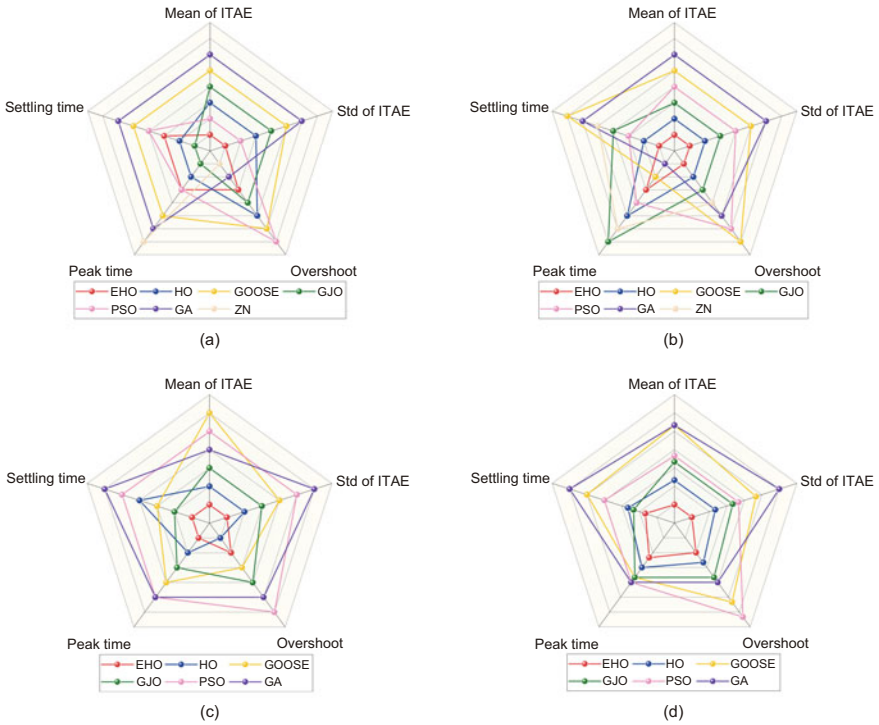

Fig. 10 Radar charts of six algorithms and the ZN method for $G_1(s)$ (a), $G_2(s)$ (b), $G_3(s)$ (c), and comprehensive comparison (d)

the algorithm, the resulted increase in runtime is not exponential for the original algorithm HO. For CEC2022 benchmark functions, the improvement strategies in EHO lead to a moderate increase in runtime. However, this increase is relatively small when compared to the substantial gains in optimization accuracy and convergence reliability, particularly in challenging optimization tasks. The additional computational cost is balanced by the superior performance of EHO in finding high-quality solutions and its ability to navigate complex and multi-dimensional search spaces effectively.

When optimizing PID controller parameters for three typical systems, the MATLAB compiler is used in conjunction with Simulink for interactive simulations. The runtime for a single execution of all algorithms on the same equipment ranges from 10 to 12 min. This duration is consistent across the algorithms. The primary focus of this study is to optimize PID controller parameters to enhance the dynamic performance response metrics of the controlled system. For this reason, the priority is placed on the optimization accuracy and reliability of the algorithm rather than its runtime. Despite the increased computation time, EHO demonstrates superior performance in terms of convergence speed, stability, and robustness. These attributes are crucial for real-world applications where the quality and reliability of the solution are more important than the computational time. For practitioners, it is important to balance the benefits of the enhanced search accuracy and robustness against the potential increase in computational time and resources. In applications (e.g., robot control and industrial process control) where high precision, stability, and robustness are paramount, the slight increase in computational overhead is justified by the substantial performance benefits offered by EHO. On the other hand, for less demanding tasks where computational efficiency is paramount, simpler algorithms may suffice.

6 Trajectory tracking for quadrotor UAVs

To further evaluate the effectiveness of EHO in real-world applications, this section explores the use of EHO in optimizing the cascade PID parameters of a quadrotor UAV for the trajectory tracking problem, and compares EHO with HO and manual

parameter adjustment (MPA). A quadrotor UAV is a highly dynamic, multi-input, strongly coupled, multivariable, and underactuated system. The optimization task involves tuning the cascade PID parameters to achieve efficient trajectory tracking while minimizing overshoot, settling time, and steady-state error.

6.1 Quadrotor UAV modeling

The control of a quadrotor UAV is fundamentally based on the manipulation of its four rotors, which generate thrust forces that allow the UAV to perform maneuvers, such as taking off, hovering, and landing. By adjusting the speed of each rotor, the quadrotor can control its motion along various axes. Fig. 11 illustrates a quadrotor UAV in an X-configuration, a commonly used setup where the four rotors are positioned in a cross-like arrangement. The forces generated by the rotors are denoted as F_1–F_4. The rotational speeds of the four rotors are denoted as Ω_1–Ω_4. The center of the UAV is labeled as O_b, with φ representing the roll angle, θ the pitch angle, and ψ the yaw angle. The body coordinate system B moves with the quadrotor and is centered at O_b, and the world coordinate system E provides a global reference frame for the position and orientation of the UAV.

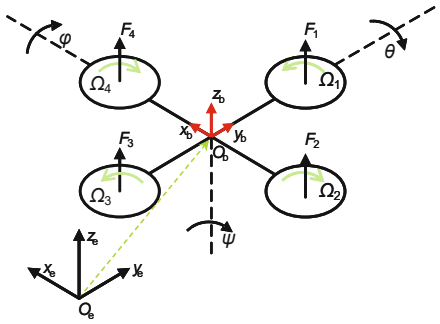

Fig. 11 Structure of the quadrotor unmanned aerial vehicle

The rotation matrix in Eq. (34) at the bottom of this page from systems B to E describes the orientation of the quadrotor UAV in a three-dimensional space (Gupta et al., 2016). The relationship between the Euler angles (φ,θ,ψ) and the angular velocities of the UAV body $(\omega_{x_\mathrm{b}},\omega_{y_\mathrm{b}},\omega_{z_\mathrm{b}})$ is given by the following:

$$\begin{bmatrix}\dot{\varphi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix}=\begin{bmatrix}1 & \tan\theta\sin\varphi & \tan\theta\cos\varphi\\ 0 & \cos\varphi & -\sin\varphi\\ 0 & \sin\varphi/\cos\theta & \cos\varphi/\cos\theta\end{bmatrix}\begin{bmatrix}\omega_{x_\mathrm{b}}\\ \omega_{y_\mathrm{b}}\\ \omega_{z_\mathrm{b}}\end{bmatrix}. \quad (35)$$

From the relationship between angular velocities and Euler angles, we derive the position and velocity dynamics of the quadrotor in the world (Earth-fixed) coordinate system:

$$\begin{cases}\dot{x}=v_{x_\mathrm{e}},\\ \dot{y}=v_{y_\mathrm{e}},\\ \dot{z}=v_{z_\mathrm{e}},\\ \dot{\varphi}=\omega_{x_\mathrm{b}}+(\tan\theta\sin\varphi)\omega_{y_\mathrm{b}}+(\tan\theta\cos\varphi)\omega_{z_\mathrm{b}},\\ \dot{\theta}=(\cos\varphi)\omega_{y_\mathrm{b}}-(\sin\varphi)\omega_{z_\mathrm{b}},\\ \dot{\psi}=\dfrac{\sin\varphi}{\cos\theta}\omega_{y_\mathrm{b}}+\dfrac{\cos\varphi}{\cos\theta}\omega_{z_\mathrm{b}},\end{cases} \quad (36)$$

where (x,y,z) are the coordinates of the quadrotor in the Earth-fixed frame and $(v_{x_\mathrm{e}},v_{y_\mathrm{e}},v_{z_\mathrm{e}})$ represent its linear velocities in the same frame.

The linear acceleration in each axis of the world coordinate system is derived from Newton's second law:

$$\begin{cases}\ddot{x}=\dot{v}_{x_\mathrm{e}}=\dfrac{f}{m}\left(\cos\psi\sin\theta\cos\varphi+\sin\psi\sin\varphi\right),\\ \ddot{y}=\dot{v}_{y_\mathrm{e}}=\dfrac{f}{m}\left(\sin\psi\sin\theta\cos\varphi-\cos\psi\sin\varphi\right),\\ \ddot{z}=\dot{v}_{z_\mathrm{e}}=\dfrac{f}{m}\left(\cos\varphi\cos\theta\right)-g,\end{cases} \quad (37)$$

where f denotes the total thrust generated by the rotors in the body-fixed frame, m is the mass of the quadrotor, and g is the gravitational acceleration.

For practical controller design, the UAV dynamics is often simplified by assuming slow changes in the attitude angles, neglecting gyroscopic moments and linearizing the equations. The moment $\boldsymbol{\tau}$ generated by the propeller on the fuselage axis is related to the moment of inertia $\boldsymbol{J}$ as follows:

$$\boldsymbol{\tau}=\begin{bmatrix}\tau_x\\ \tau_y\\ \tau_z\end{bmatrix}=\begin{bmatrix}J_x & 0 & 0\\ 0 & J_x & 0\\ 0 & 0 & J_x\end{bmatrix}\begin{bmatrix}\dot{\omega}_{x_\mathrm{b}}\\ \dot{\omega}_{y_\mathrm{b}}\\ \dot{\omega}_{z_\mathrm{b}}\end{bmatrix}=\boldsymbol{J}\dot{\boldsymbol{\omega}}_\mathrm{b}. \quad (38)$$

$$\boldsymbol{R}_\mathrm{b}^\mathrm{e}=\begin{bmatrix}\cos\theta\cos\psi & \cos\psi\sin\theta\sin\varphi-\sin\psi\cos\varphi & \cos\psi\sin\theta\cos\varphi+\sin\psi\sin\varphi\\ \cos\theta\sin\psi & \sin\psi\sin\theta\sin\varphi+\cos\psi\cos\varphi & \sin\psi\sin\theta\cos\varphi-\cos\psi\sin\varphi\\ -\sin\theta & \sin\varphi\cos\theta & \cos\theta\cos\varphi\end{bmatrix}. \quad (34)$$

The rigid body model for the quadrotor UAV is expressed by

$$\begin{cases} \ddot{x} = \frac{\sin\theta\cos\varphi\cos\psi+\sin\varphi\sin\psi}{m}U_1, \\ \ddot{y} = \frac{\sin\theta\cos\varphi\cos\psi-\sin\varphi\sin\psi}{m}U_1, \\ \ddot{z} = \frac{\cos\varphi\cos\theta}{m}U_1 - g, \\ \ddot{\varphi} = \frac{1}{J_x}U_2, \\ \ddot{\theta} = \frac{1}{J_y}U_3, \\ \ddot{\psi} = \frac{1}{J_z}U_4, \end{cases} \tag{39}$$

where $U_1 = f$ is the total lift force, and $U_2 = \tau_x$, $U_3 = \tau_y$, and $U_4 = \tau_z$ are the control inputs representing the torques around the body axes.

The lift force f and moment $\boldsymbol{\tau}$ acting on the airframe are related to the rotational speeds Ω of the rotors, as expressed in Eq. (40). Here, c_T is the tension coefficient, and c_M is the moment coefficient.

$$\begin{bmatrix} f \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} c_\mathrm{T} & c_\mathrm{T} & c_\mathrm{T} & c_\mathrm{T} \\ -\frac{\sqrt{2}}{2}c_\mathrm{T} & \frac{\sqrt{2}}{2}c_\mathrm{T} & \frac{\sqrt{2}}{2}c_\mathrm{T} & -\frac{\sqrt{2}}{2}c_\mathrm{T} \\ \frac{\sqrt{2}}{2}c_\mathrm{T} & -\frac{\sqrt{2}}{2}c_\mathrm{T} & \frac{\sqrt{2}}{2}c_\mathrm{T} & -\frac{\sqrt{2}}{2}c_\mathrm{T} \\ c_\mathrm{M} & c_\mathrm{M} & c_\mathrm{M} & c_\mathrm{M} \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix}. \tag{40}$$

6.2 Results and analysis

The cascade PID controller is particularly well-suited for quadrotor UAV control due to its simplicity, tuning flexibility, and strong disturbance resistance. By employing a hierarchical control structure with inner and outer loops, the controller ensures the stability and responsiveness of the UAV, enabling precise attitude control and trajectory tracking during flight, as illustrated in Fig. 12. The inner loop manages the UAV's attitude, while the outer loop handles position control. In this part, EHO is employed to simultaneously tune all the parameters of the cascade PID controller for a quadrotor UAV to track a complex helical trajectory. The simulation results are then compared with those obtained using HO and the manual PID tuner from MATLAB/Simulink.

In the simulations, the PID parameter range is restricted to $[0, 10]$. EHO is configured with a population size of 20, a maximum iteration number of 100, and a dimensionality of 6, reflecting the six PID parameters to be optimized. The selected trajectory to evaluate the EHO-optimized UAV PID controller is a complex helical trajectory, which challenges the UAV's maneuvering capabilities with intricate geometric features. The mathematical expression for this trajectory is

$$\begin{cases} x_\mathrm{ref}(t) = \cos(0.6t) - \cos(0.5t)^3, \\ y_\mathrm{ref}(t) = \sin(0.2t) - \sin(0.4t)^3, \\ z_\mathrm{ref}(t) = 0.3t. \end{cases} \tag{41}$$

Given that the quadrotor is an underactuated system, it lacks direct control inputs to stabilize the x and y positions. Therefore, changes in these positions are achieved by controlling the UAV's attitude angles. In this study, the desired yaw angle is set to $\psi_\mathrm{d} = 0°$. By calculating the dynamic equations of the UAV, the relationship between the desired position and the desired attitude angles can be determined as follows:

$$\begin{cases} \theta_\mathrm{d} = \arctan\frac{\ddot{x}_\mathrm{d}\cos\psi_\mathrm{d}+\ddot{y}_\mathrm{d}\sin\psi_\mathrm{d}}{\ddot{z}_\mathrm{d}+g}, \\ \varphi_\mathrm{d} = \arctan(\cos(\theta_\mathrm{d})\frac{\ddot{x}_\mathrm{d}\sin\psi_\mathrm{d}-\ddot{y}_\mathrm{d}\cos\psi_\mathrm{d}}{\ddot{z}_\mathrm{d}+g}). \end{cases} \tag{42}$$

Table S6 in the supplementary materials lists the parameters of the used quadrotor UAV. Tables S7 and S8 in the supplementary materials provide detailed PID tuning parameters for each channel using EHO, HO, and MPA for outer and inner loops, respectively. The simulation results of the position variables x, y, and z, and the attitude angles φ and θ over an 80-s complex helical trajectory are presented in Fig. S3 in the supplementary materials, along with their deviations from the ideal positions. As seen in Fig. S3 in the supplementary

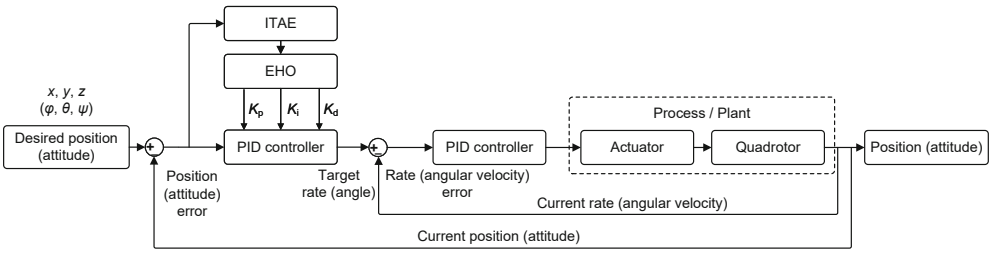

Fig. 12 Cascade PID controller design of the quadrotor unmanned aerial vehicle using enhanced hippopotamus optimization

materials, optimization results of EHO and HO in the x-channel are nearly identical, while MPA exhibits the largest deviation. In the y-channel, EHO achieves a smaller error than HO during the UAV's takeoff phase. In the z-channel, EHO demonstrates a higher response speed compared to MPA and HO. EHO and HO exhibit similar tracking performance in the φ-channel. Specifically, EHO achieves a significantly smaller error in the θ-channel compared to HO, with the deviation from the desired value diminishing over time. The three-dimensional flight path of the quadrotor UAV is visualized in Fig. S4 in the supplementary materials.

Table 4 presents the ITAE values for each channel, which are used to evaluate the three control methods. The results show that all three methods effectively optimize the cascade PID controller of the quadrotor UAV, confirming their stability and effectiveness in the cascade PID control design. The ITAE values presented in Table 4 also indicate that EHO outperforms both HO and MPA across all channels.

Overall, EHO achieves substantial improvements over HO in optimizing complex systems. Both EHO and HO provide more efficient optimization results compared to MPA, which requires more time and effort for parameter tuning. This demonstrates that EHO is a promising approach for the automated tuning of PID controllers in UAV applications, particularly for complex trajectories requiring precise control.

7 Conclusions

Tuning PID controllers is a critical aspect of control systems, because it directly impacts accuracy, stability, and response speed. Although HO offers simplicity, adaptability, and efficiency, its original form struggles with slow convergence and a tendency to get trapped in local optima, especially in complex optimization tasks. Consequently, this study introduces an EHO to improve PID controller tuning. EHO enhances the optimization process by incorporating LHS, ALRL, fitness value sorting, and an adaptive perturbation mechanism. In comparison with five classical or state-of-the-art algorithms using CEC2022 benchmark functions, EHO exhibits superior performance in terms of optimization accuracy, convergence speed, exploration–exploitation balance, and resistance to stagnation. To further validate the effectiveness of EHO, MATLAB/Simulink simulations on three typical systems are performed. The results indicate that EHO outperforms the compared methods, underscoring its superiority in tuning PID controllers. Finally, EHO is applied to optimize the cascade PID controller parameters for a quadrotor UAV, demonstrating its practical benefits in trajectory tracking. The EHO-optimized UAV shows superior alignment with the desired trajectory, and achieves the lowest integral of time-weighted absolute error across all position channels, outperforming both the original HO and MPA. These findings highlight the robustness and effectiveness of EHO for real-world control applications.

Contributors

Kailong MOU and Deguang WANG conceived and designed the study, performed the experiments, and drafted the paper. All the authors reviewed the paper. Deguang WANG finalized the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Amiri MH, Mehrabi Hashjin N, Montazeri M, et al., 2024. Hippopotamus optimization algorithm: a novel

Table 4 Comparison of the ITAE values for attitude position channel

Method	ITAE				
	x-channel	y-channel	z-channel	φ-channel	θ-channel
EHO	**59.9790**	**22.1615**	**0.0170**	**2.9579**	**2.8064**
HO	60.2891	23.4227	0.2275	**2.9579**	6.8240
MPA	416.6531	262.0463	0.0381	3.0366	7.0969

The best results are in bold

nature-inspired optimization algorithm. *Sci Rep*, 14(1):5032. https://doi.org/10.1038/s41598-024-54910-3

Ang KH, Chong G, Li Y, 2005. PID control system analysis, design, and technology. *IEEE Trans Contr Syst Technol*, 13(4):559-576. https://doi.org/10.1109/TCST.2005.847331

Bujok P, Kolenovsky P, 2022. Eigen crossover in cooperative model of evolutionary algorithms applied to CEC 2022 single objective numerical optimisation. IEEE Congress on Evolutionary Computation, p.1-8. https://doi.org/10.1109/CEC55065.2022.9870433

Carlucho I, De Paula M, Acosta GG, 2020. An adaptive deep reinforcement learning approach for MIMO PID control of mobile robots. *ISA Trans*, 102:280-294. https://doi.org/10.1016/j.isatra.2020.02.017

Carvajal J, Chen GR, Ogmen H, 2000. Fuzzy PID controller: design, performance evaluation, and stability analysis. *Inform Sci*, 123(3-4):249-270. https://doi.org/10.1016/S0020-0255(99)00127-9

Chen SY, Lin FJ, 2013. Decentralized PID neural network control for five degree-of-freedom active magnetic bearing. *Eng Appl Artif Intell*, 26(3):962-973. https://doi.org/10.1016/j.engappai.2012.11.002

Chen XY, Zhang MJ, Yang M, et al., 2024. A multistrategy improved beluga whale optimization algorithm for constrained engineering problems. *Cluster Comput*, 27(10):14685-14727. https://doi.org/10.1007/s10586-024-04680-4

Chien KL, Hrones JA, Reswick JB, 1952. On the automatic control of generalized passive systems. *J Fluids Eng*, 74(2):175-183. https://doi.org/10.1115/1.4015724

Chopra N, Ansari MM, 2022. Golden jackal optimization: a novel nature-inspired optimizer for engineering applications. *Expert Syst Appl*, 198:116924. https://doi.org/10.1016/j.eswa.2022.116924

Cohen GH, Coon GA, 1953. Theoretical consideration of retarded control. *J Fluids Eng*, 75(5):827-834. https://doi.org/10.1115/1.4015451

Deng W, Chen R, He B, et al., 2012. A novel two-stage hybrid swarm intelligence optimization algorithm and application. *Soft Comput*, 16(10):1707-1722. https://doi.org/10.1007/s00500-012-0855-z

Deng W, Cai X, Wu DQ, et al., 2024. MOQEA/D: multiobjective QEA with decomposition mechanism and excellent global search and its application. *IEEE Trans Intell Transp Syst*, 25(9):12517-12527. https://doi.org/10.1109/TITS.2024.3373510

El Khoukhi F, Boukachour J, El Hilali Alaoui A, 2017. The "dual-ants colony": a novel hybrid approach for the flexible job shop scheduling problem with preventive maintenance. *Comput Ind Eng*, 106:236-255. https://doi.org/10.1016/j.cie.2016.10.019

Feng H, Ma W, Yin CB, et al., 2021. Trajectory control of electro-hydraulic position servo system using improved PSO-PID controller. *Autom Constr*, 127:103722. https://doi.org/10.1016/j.autcon.2021.103722

Gad AG, 2022. Particle swarm optimization algorithm and its applications: a systematic review. *Arch Computat Methods Eng*, 29(5):2531-2561. https://doi.org/10.1007/s11831-021-09694-4

Gupta L, Jain R, Vaszkun G, 2016. Survey of important issues in UAV communication networks. *IEEE Commun Surv Tutor*, 18(2):1123-1152. https://doi.org/10.1109/COMST.2015.2495297

Hamad RK, Rashid TA, 2024. GOOSE algorithm: a powerful optimization tool for real-world engineering challenges and beyond. *Evolving Syst*, 15(4):1249-1274. https://doi.org/10.1007/s12530-023-09553-6

He Y, Wang MR, 2024. An improved chaos sparrow search algorithm for UAV path planning. *Sci Rep*, 14(1):366. https://doi.org/10.1038/s41598-023-50484-8

Jing XJ, Cheng L, 2013. An optimal PID control algorithm for training feedforward neural networks. *IEEE Trans Ind Electron*, 60(6):2273-2283. https://doi.org/10.1109/TIE.2012.2194973

Kashyap AK, Parhi DR, 2021. Particle swarm optimization aided PID gait controller design for a humanoid robot. *ISA Trans*, 114:306-330. https://doi.org/10.1016/j.isatra.2020.12.033

Kennedy J, Eberhart R, 1995. Particle swarm optimization. Int Conf on Neural Networks, p.1942-1948. https://doi.org/10.1109/ICNN.1995.488968

Kommula BN, Kota VR, 2022. Design of MFA-PSO based fractional order PID controller for effective torque controlled BLDC motor. *Sustain Energy Technol Assess*, 49:101644. https://doi.org/10.1016/j.seta.2021.101644

Lange KL, Little RJA, Taylor JMG, 1989. Robust statistical modeling using the t distribution. *J Am Stat Assoc*, 84(408):881-896. https://doi.org/10.1080/01621459.1989.10478852

Li AD, He Z, 2020. Multiobjective feature selection for key quality characteristic identification in production processes using a nondominated-sorting-based whale optimization algorithm. *Comput Ind Eng*, 149:106852. https://doi.org/10.1016/j.cie.2020.106852

Liang HB, Zou JL, Zuo K, et al., 2020. An improved genetic algorithm optimization fuzzy controller applied to the wellhead back pressure control system. *Mech Syst Sig Process*, 142:106708. https://doi.org/10.1016/j.ymssp.2020.106708

Liu YJ, Xu H, Zhang YG, 2017. Burner-electrode position control of calcium carbide furnace based on BP-PID controller. IEEE Int Conf on Mechatronics and Automation, p.810-815. https://doi.org/10.1109/ICMA.2017.8015920

Mirjalili S, Mirjalili SM, Lewis A, 2014. Grey wolf optimizer. *Adva Eng Softw*, 69:46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Mishra AK, Das SR, Ray PK, et al., 2020. PSO-GWO optimized fractional order PID based hybrid shunt active power filter for power quality improvements. *IEEE Access*, 8:74497-74512. https://doi.org/10.1109/ACCESS.2020.2988611

Mohan BM, Sinha A, 2008. Analytical structure and stability analysis of a fuzzy PID controller. *Appl Soft Comput*, 8(1):749-758. https://doi.org/10.1016/j.asoc.2007.06.003

Mortazavi A, 2022. Interactive fuzzy Bayesian search algorithm: a new reinforced swarm intelligence tested on engineering and mathematical optimization problems. *Expert Syst Appl*, 187:115954. https://doi.org/10.1016/j.eswa.2021.115954

Mortazavi A, 2024. A fuzzy reinforced Jaya algorithm for solving mathematical and structural optimization problems. *Soft Comput*, 28(3):2181-2206. https://doi.org/10.1007/s00500-023-09206-5

Mortazavi A, Toğan V, Moloodpoor M, 2019. Solution of structural and mathematical optimization problems using a new hybrid swarm intelligence optimization algorithm. *Adv Eng Softw*, 127:106-123. https://doi.org/10.1016/j.advengsoft.2018.11.004

Mughees A, Mohsin SA, 2020. Design and control of magnetic levitation system by optimizing fractional order PID controller using ant colony optimization algorithm. *IEEE Access*, 8:116704-116723. https://doi.org/10.1109/ACCESS.2020.3004025

Parpinelli RS, Teodoro FR, Lopes HS, 2012. A comparison of swarm intelligence algorithms for structural engineering optimization. *Int J Num Methods Eng*, 91(6):666-684. https://doi.org/10.1002/nme.4295

Preethi P, Mamatha HR, 2022. Region-based convolutional neural network for segmenting text in epigraphical images. *Artif Intell Appl*, 1(2):119-127. https://doi.org/10.47852/bonviewAIA2202293

Qi Z, Shi Q, Zhang H, 2020. Tuning of digital PID controllers using particle swarm optimization algorithm for a CAN-based DC motor subject to stochastic delays. *IEEE Trans Ind Electron*, 67(7):5637-5646. https://doi.org/10.1109/TIE.2019.2934030

Rana KPS, Kumar V, Sehgal N, et al., 2019. A novel $\frac{dP}{dI}$ feedback based control scheme using GWO tuned PID controller for efficient MPPT of PEM fuel cell. *ISA Trans*, 93:312-324. https://doi.org/10.1016/j.isatra.2019.02.038

Reddy MJ, Kumar DN, 2007. An efficient multi-objective optimization algorithm based on swarm intelligence for engineering design. *Eng Optim*, 39(1):49-68. https://doi.org/10.1080/03052150600930493

Shenassa MH, Khakpour K, 2008. Knowledge base expert system for tuning PID controllers using wireless technology. Int Conf on Computer and Communication Engineering, p.310-313. https://doi.org/10.1109/ICCCE.2008.4580618

Silva F, Batista J, Souza D, et al., 2023. Control and identification of parameters of a joint of a manipulator based on PID, PID 2-DOF, and least squares. *J Braz Soc Mech Sci Eng*, 45(6):327. https://doi.org/10.1007/s40430-023-04251-5

Sun QY, Chen JM, Zhou L, et al., 2024. A study on ice resistance prediction based on deep learning data generation method. *Ocean Eng*, 301:117467. https://doi.org/10.1016/j.oceaneng.2024.117467

Tang J, Liu G, Pan QT, 2021. A review on representative swarm intelligence algorithms for solving optimization problems: applications and trends. *IEEE/CAA J Automat Sin*, 8(10):1627-1643. https://doi.org/10.1109/JAS.2021.1004129

Viana FAC, 2016. A tutorial on Latin hypercube design of experiments. *Qual Reliab Eng Int*, 32(5):1975-1985. https://doi.org/10.1002/qre.1924

Wang YJ, He HY, Qu ZW, 2015. PSO-PID based temperature control method for bifilar helix calculable resistor. 12th IEEE Int Conf on Electronic Measurement & Instruments, p.722-725. https://doi.org/10.1109/ICEMI.2015.7494317

Wang YP, Zhang JX, Zhang MJ, et al., 2024. Enhanced artificial ecosystem-based optimization for global optimization and constrained engineering problems. *Cluster Comput*, 27(7):10053-10092. https://doi.org/10.1007/s10586-024-04488-2

Wang YZ, Jin QB, Zhang RD, 2017. Improved fuzzy PID controller design using predictive functional control structure. *ISA Trans*, 71:354-363. https://doi.org/10.1016/j.isatra.2017.09.005

Xie GD, Zhang MJ, Yang M, et al., 2024. Economic dispatch of isolated microgrids based on enhanced sparrow search algorithm. *Eng Lett*, 32(4):753-760.

Xu YX, Zhang MJ, Yang M, et al., 2024. Hybrid quantum particle swarm optimization and variable neighborhood search for flexible job-shop scheduling problem. *J Manuf Syst*, 73:334-348. https://doi.org/10.1016/j.jmsy.2024.02.007

Xue JK, Shen B, 2020. A novel swarm intelligence optimization approach: sparrow search algorithm. *Syst Sci Contr Eng*, 8(1):22-34. https://doi.org/10.1080/21642583.2019.1708830

Zhang MJ, Wen GH, 2024. Duck swarm algorithm: theory, numerical optimization, and applications. *Cluster Comput*, 27(5):6441-6469. https://doi.org/10.1007/s10586-024-04293-x

Ziegler JG, Nichols NB, 1942. Optimum settings for automatic controllers. *Trans Am Soc Mech Eng*, 64(8):759-765. https://doi.org/10.1115/1.4019264

List of supplementary materials

Table S1 Description of CEC2022 benchmark functions

Table S2 Parameter settings of six algorithms

Table S3 Optimization results of EHO and five compared algorithms on CEC2022 benchmark functions

Table S4 Comparison results of the original HO and its improved variants

Table S5 PID controller parameter optimization results by EHO using different fitness functions

Table S6 Parameters of the quadrotor UAV

Table S7 PID controller parameters for outer loop

Table S8 PID controller parameters for inner loop

Fig. S1 Convergence curves of EHO and five compared algorithms on CEC2022 benchmark functions

Fig. S2 Convergence curves of EHO and step responses of the system using different fitness functions

Fig. S3 Tracking effect of each channel

Fig. S4 3D simulation results for the complex helical trajectory