Multi-strategy improved electric eel foraging optimisation algorithm for UAV path planning

Zexin Zhang[a], Chengbiao Fu[a*], Hongwei Guo[c], Anhong Tian[b]

[a]Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China

E-mail：20242204083@kust.edu.cn

fcb@kust.edu.cn

[b]Faculty of Land Resource Engineering, Kunming University of Science and Technology, Kunming 650093, China

E-mail：ghw202@foxmail.com

[c]School of Engineering, Honghe University, Mengzi, Yunnan, China

E-mail：tah@kust.edu.cn

Abstracts

The Electric Eel Foraging Optimization (EEFO) algorithm is a metaheuristic optimization method based on swarm intelligence, designed to solve complex optimization problems by simulating the foraging behavior of electric eels. Although EEFO demonstrates strong global search capabilities, it still suffers from issues such as insufficient initial population diversity, premature convergence, a tendency to fall into local optima, and limited optimization accuracy. To address these limitations, this paper proposes an improved Electric Eel Foraging Optimization (IEEFO) algorithm that integrates multiple strategies. First, a good point set initialization strategy is adopted to enhance population diversity. Second, a differential evolution strategy is incorporated to improve global search ability. Third, a random boundary control mechanism is introduced to increase search flexibility. Finally, a solution quality enhancement strategy is implemented to refine local exploitation capabilities. Experimental results show that IEEFO outperforms the original EEFO and other comparison algorithms in terms of convergence accuracy and stability. Furthermore, its application to the UAV path planning problem further validates the algorithm's practical engineering value: IEEFO effectively handles 3D path planning under complex constraints, achieves a strong balance among multiple objectives such as threat avoidance, energy consumption minimization, and path smoothness, significantly improves path quality and algorithmic robustness, and demonstrates considerable potential for real-world applications.Keywords: Electric eel foraging optimisation algorithm; Population intelligence; Differential evolution; Boundary control; Path planning

1. Introduction

The Electric Eel Foraging Optimization (EEFO) algorithm is a bio-inspired metaheuristic algorithm proposed by Zhao and Wang, which draws inspiration from the foraging behavior of electric eel populations[1]. Although EEFO offers advantages such as simple parameter settings and high solution accuracy compared to some classical metaheuristic methods, its inherent limitations cannot be overlooked[2]. The main limitations of EEFO are as follows: (1) Insufficient diversity of initial solutions. Due to random initialization, the diversity of solutions in the early stage is inadequate, preventing the algorithm from quickly approaching the vicinity of the optimal solution and resulting in slower convergence. (2) Inadequate search breadth during the exploration phase. During exploration, the algorithm tends to search predominantly around the population mean, which may restrict its overall search scope. (3) Large randomness in out-of-bounds solutions after boundary control handling. Significant randomness in solutions may lead to wasted computational resources in non-critical regions, increasing time costs and computational inefficiency. (4) Insufficient depth in local search. As problem complexity increases, numerous nonlinear constraints make EEFO prone to falling into local optima, causing premature convergence and limiting further improvement in solution accuracy.Unmanned Aerial Vehicle (UAV) path planning is a typical high-dimensional, nonlinearly constrained optimization problem that requires the algorithm to possess strong global exploration capabilities and high local optimization precision. The problem involves identifying an optimal or near-optimal flight trajectory in a complex three-dimensional environment while considering multiple constraints such as threat avoidance, fuel consumption, path length, and smoothness. Traditional EEFO struggles with such problems, often converging slowly and becoming trapped in local optima. Motivated by these challenges, this paper proposes an Improved Electric Eel Foraging Optimization (IEEFO) algorithm that integrates multiple strategies, including differential evolution. The proposed IEEFO combines four enhancement mechanisms to collectively address issues such as slow convergence, lack of population diversity, and insufficient precision in high-dimensional spaces. Experimental validation demonstrates the effectiveness of IEEFO on both benchmark test functions and UAV path planning problems. The results show that the improved algorithm outperforms the original EEFO and other comparative algorithms in terms of convergence speed, global optimization capability, and stability. This study provides a more efficient optimization method for UAV path planning and extends the application of swarm intelligence algorithms in engineering optimization.

2. Multi-strategy improvement of electric eel foraging optimisation algorithms

2.1 Good point set population initialisation strategy

A high-quality initial solution distribution can effectively improve the efficiency of the algorithm, enhance the diversity of solutions, avoid falling into the local optimum at an early stage, and accelerate the convergence speed. In the original EEFO, the electric eel positions are initialised by simple random initialisation, which is difficult to ensure the population diversity and affects the search accuracy and efficiency. In order to improve the quality of the initial solution, this study adopts the good point set initialisation strategy to generate individuals in a more uniform and random way, and improve the global search capability and convergence speed[3]. The good point set strategy is characterised by the fact that the deviation order is only determined by the size of the solution set, independent of the problem dimension, and is suitable for high-dimensional optimisation. Its principle is as follows:

Let G_d be the unit cube in the d-dimensional Euclidean space, and given $r = (r_1, r_2, \dots, r_d) \in G_d$, defined better as:

$$X_i = \left\{ \left(\left\{ r_1^{(N)} \cdot i \right\}, \left\{ r_2^{(N)} \cdot i \right\}, \dots, \left\{ r_d^{(N)} \cdot i \right\} \right), 1 \le i \le N \right\} \tag{31}$$

$$r = \left\{ 2\cos\left(\frac{2\pi j}{p}\right), 1 \le j \le d \right\} \tag{32}$$

where X_i is the ith individual, r = $(r_1, r_2, r_3, \cdots, r_d)$ is a point in the d-dimensional space, r_j represents the value of the jth dimension of the point X_i as determined by Eq. (32), $\left\{ r_j^{(N)} \cdot i \right\}$ denotes the $r_j^{(N)} \cdot i$ fractional part, p is the smallest prime number that satisfies d ≤ $(p$-3$)/2$ the smallest prime number, i is an integer with a range of $1 \le i \le N$, denotes the index of a point in the set of good points, and N stands for the number of points in the set of good points, which in this case represents the number of populations. Suppose the population size is 100 and the search domain is [-100, 100]. **Fig.1** shows the comparison of the distribution of the population initialization generated by the good point set strategy and the population initialization generated by random numbers in the two-dimensional space. It can be clearly seen that the population generated by the random initialization strategy is chaotic and even has many overlapping parts. The population generated by the good point set initialization strategy is significantly more uniform and has more diversity in the search space.

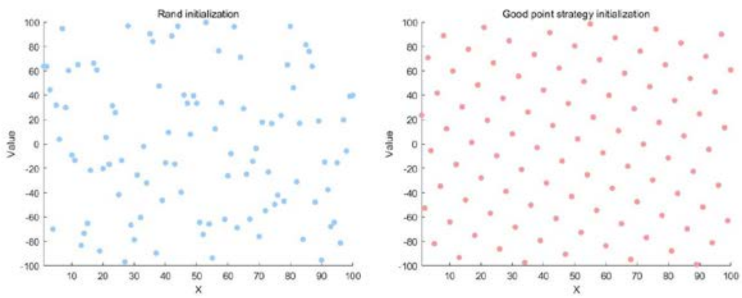

Fig.1 Comparison of random initialisation and good point set strategy initialisation distributions

2.2 Differential evolution strategy

Differential evolution (DE) is an efficient and deterministic global optimisation algorithm for solving nonlinear, nondifferentiable continuous space function problems. Compared with other global optimisation algorithms, DE possesses faster convergence speed and stronger search capability [4]. Its core lies in the continuous exploration of the search space through three major operations: mutation, crossover and selection. The mutation operation generates new individuals by constructing difference vectors for three randomly selected individuals and adding a third individual position to enhance diversity, which helps to jump out of the local optimum and expand the search scope, thus enhancing the global optimisation capability[5]. In order to enhance the performance of IEEFO, this paper introduces the differential evolution strategy and integrates its evolutionary process into IEEFO in the following steps:

(1) Variant operations:

$$\mathbf{X}_{new}(t) = \mathbf{X}_P(t) + F \times \left(\mathbf{X}_Q(t) - \mathbf{X}_M(t)\right), P \neq Q \neq M \tag{3}$$

where $\mathbf{X}_{new}(t)$ represents the mutant individuals, and P, Q and M are the three indices $\in \{1,2,\cdots,N\}$. F denotes the mutation scale factor, $F = F_0 \times 2e^{\frac{t}{1-t_{\max}}}$ where $\mathbf{X}_{new}(t)$ with F_0 is equal to 0.5.

(2) Crossover operation:

$$Z_{i,j}(t) = \begin{cases} X_{new,j}(t) & rand(0,1) \leqslant CR \text{ or } j = j_{rand} \\ X_{i,j}(t) & otherwise \end{cases} \tag{4}$$

where $CR \in [0,1]$ denotes the crossover probability. In this study CR=0.3 and how this value is reached will be proved in Section 4. j_{rand} denotes a random integer $\in \{1,2,\cdots,D\}$ which ensures that $Z_i(t)$ gets at least one parameter from $\mathbf{X}_{new}(t)$.

(3) Selection operation:

$$X_i(t+1) = \begin{cases} Z_{i,j}(t) & fit(Z_{i,j}(t)) \leqslant fit(X_i(t+1))) \\ X_i(t) & fit(Z_{i,j}(t)) > fit(X_i(t+1))) \end{cases} \tag{5}$$

If the fitness value of the test vector $Z_i(t)$ is better than $X_i(t+1)$, then $Z_i(t)$ will be retained as the new individual $X_i(t+1)$ in subsequent iterations. Otherwise, the existing target vector $X_i(t+1)$ is retained.

By combining IEEFO with differential evolution strategy, the search range of the algorithm is further expanded and the population diversity is significantly enhanced. It not only effectively avoids the situation of falling into local optimum, but also significantly improves the convergence accuracy of the algorithm.

2.3 Stochastic boundary control strategy

During the iteration process, some individuals may cross the boundary and enter the invalid solution space, affecting the efficiency and stability of the algorithm. The traditional practice usually forces them back to the boundary range, which can ensure the validity of the solution, but does not make full use of the high-quality solution information in the population. For this reason, this paper proposes an improved boundary control strategy: when an individual crosses the boundary, it is no longer simply truncated, but combined with the position of the optimal and suboptimal individuals in the current population, and guidedly adjusted with a certain probability. This method not only preserves the effectiveness of the solution space, but also improves the efficiency of the use of high-quality solutions, enhances the global search ability, reduces the risk of falling into the local optimum, and improves the overall performance of the algorithm. Its mathematical expression is shown as follows:

$$X_i(t+1) = \begin{cases} X_{BEST\,\mathrm{II}}(t) + R_0 \times \left(X_{prey}(t) - X_{BEST\,\mathrm{II}}(t)\right) & rand \leqslant 0.5 \\ R_1(Up - Low) + Low & rand > 0.5 \end{cases} \tag{6}$$

Where $X_{prey}(t)$, $X_{BEST\,\mathrm{II}}(t)$ are the position of the best solution and the position of the second best individual in the current iteration, respectively, R_0, R_1 are both random numbers of [0,1], and Up and Low are the upper and lower bounds, respectively. This method allows the probability of individuals exceeding the boundary to obtain better fitness values so that they have better initial positions in the next iteration, while the method effectively avoids the problem of premature convergence. Therefore, the introduction of this new stochastic boundary control strategy can effectively improve the performance of the algorithm. The principle of the method is shown in **Fig. 2.**

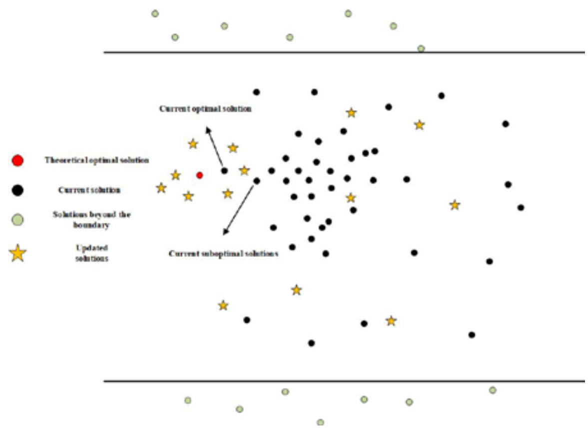

Fig.2 Schematic diagram of boundary control

2.4 Solution quality enhancement strategy

In order to address the shortcomings of EEFO in terms of convergence accuracy and the tendency to fall into local optimums, we introduce a solution quality enhancement strategy after generating a new solution [6], which was first proposed by ImanAhmadianfar in 2021 as a way to extend the search depth of the bounds, enhance the algorithm's local search capability, and enable the diversity of solutions to be improved.The IEEFO algorithm, through the use of a solution quality enhancement strategy can move each solution to a better position, further refining the optimal solution while minimising the risk of falling into a local optimum.

In the adopted solution quality improvement strategy, the average value X_{mean} of three random solutions is calculated first, and then it is combined with the current optimal solution X_{prey} to generate a new candidate solution X_{new1}. Generate X_{new2} through the following formula:

$$X_{new1} = rand \times X_{mean} + (1 - rand) \times X_{prey} \tag{37}$$

$$X_{mean} = \frac{x_{r1} + x_{r2} + x_{r3}}{3} \tag{38}$$

$$w = rand(0,2).exp\left(-c\left(\frac{It}{MaxIt}\right)\right) \tag{39}$$

$$X_{new2} = \begin{cases} X_{new1} + r.w.|(X_{new1} - X_{mean}) + randn| & rand < 0.5 \; and \; w < 1 \\ (X_{new1} - X_{mean}) + r.w.|(u.X_{new1} - X_{mean}) + randn| & other \end{cases} \tag{40}$$

The resulting solution X_{new2} may not be better than the current solution (i.e. $f(X_{new2}) > f(X_i)$). For this reason, another new candidate solution X_{new3} is further constructed, which is defined as follows:

$$X_{new3} = (X_{new2} - rand.X_{new2}) + SF.\left(rand.X_i + \left(v.X_{prey} - X_{new2}\right)\right) rand < w \tag{41}$$

Where

$$SF = 2.(0.5 - rand) \times f \tag{42}$$

$$f = a \times exp\left(-b \times rand \times \left(\frac{It}{MaxIt}\right)\right) \tag{43}$$

X_{new3} is an update of X_{new2}, where SF is mainly used to control the balance between exploration and development. the value of v is $2 \times rand$, and X_i is the current solution.

3. The UAV Path Planning Problem

In this section, an improved electric eel foraging optimisation algorithm is used to solve the UAV path planning problem. The path planning cost function is based on the model proposed by Manh Duong Phung (2021) [7] and contains four components: path cost, threat cost, altitude cost and smoothing cost. The path cost is the sum of Euclidean distances between waypoints:

(1) The path cost is the sum of the Euclidean distances between waypoints:

$$F_1(X_i) = \sum_{j=1}^{n-1} \left\| \overrightarrow{Z_{ij} Z_{i,j+1}} \right\| \tag{14}$$

(2) The threat cost takes into account the obstacle cylinder area and the safe distance from the UAV and is calculated as:

$$\begin{cases} F_2(X_i) = \sum_{j=1}^{n-1} \sum_{k=1}^{K} T_k(\overrightarrow{Z_{ij} Z_{i,j+1}}), \\ T_k(\overrightarrow{Z_{ij} Z_{i,j+1}}) = \begin{cases} 0, & \text{if} d_k > S + D + R_k \\ (S + D + R_k) - d_k, & \text{if} D + R_k < d_k \le S + D + R_k \\ \infty, & \text{if} d_k \le D + R_k \end{cases} \end{cases} \tag{15}$$

(3) Altitude cost penalises path points whose flight altitude is outside the limit:

$$H_{ij} = \begin{cases} \left| h_{ij} - \frac{(h_{max} + h_{min})}{2} \right|, & \text{if} h_{min} \le h_{ij} \le h_{max} \\ \infty, & \text{otherwise,} \end{cases} \tag{16}$$

The total height cost is:

$$F_3(X_i) = \sum_{j=1}^{n} H_{ij} \tag{17}$$

(4) The smoothing cost combines the turning angle $\alpha_{i,j}$ and the flat slope angle $\beta_{i,j}$, where the projection vector can be computed as:

$$\overrightarrow{Z'_{i,j} Z'_{i,j+1}} = \vec{k} \times \left(\overrightarrow{Z_{i,j} Z_{i,j+1}} \times \vec{k}\right) \tag{18}$$

The turning angle is calculated as：

$$\alpha_{i,j} = \arctan\left(\frac{\left\|\overrightarrow{Z'_{i,j}Z'_{i,j+1}} \times \overrightarrow{Z'_{i,j+1}Z'_{i,j+2}}\right\|}{\overrightarrow{Z'_{i,j}Z'_{i,j+1}} \cdot \overrightarrow{Z'_{i,j+1}Z'_{i,j+2}}}\right) \tag{19}$$

The angle of climb is calculated as：

$$\beta_{i,j} = \arctan\left(\frac{z_{i,j+1} - z_{ij}}{\left\|\overrightarrow{Z'_{i,j}Z'_{i,j+1}}\right\|}\right) \tag{20}$$

So the smoothing cost is calculated as：

$$F_4(X_i) = a_1 \sum_{j=1}^{n-2} \alpha_{i,j} + a_2 \sum_{j=1}^{n-1} \mid \beta_{i,j} - \beta_{i,j-1} \mid \tag{21}$$

where a_1 and a_2 are the penalty coefficients for turning angle and climb angle, respectively. The final total cost function is defined as:

$$F(X_i) = AF_1(X_i) + BF_2(X_i) + CF_3(X_i) + DF_4(X_i) \tag{23}$$

where A, B, C, and D are the weight coefficients for each consideration.

The main objective of the problem is to reduce the total cost as much as possible within the constraints, having 10 coordinates on each path other than the start and end points. Here we use the weighting coefficients (5, 1, 10, 1) from the original paper and perform the following experiments when targeting 4 threats in scenario 3, with a population size of 500 and an iteration number of 200, where each result is the average of the values obtained from 10 independent runs. Table.1 shows the solution and minimum total cost corresponding to the total cost function solved by each algorithm, and **Fig.3** shows the convergence curve of IEEFO with different algorithms to solve the minimum total cost.

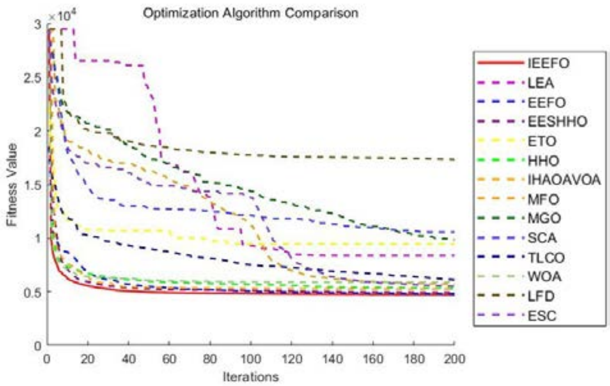

Fig.3 Comparison of convergence curves of different functions for solving the minimum total cost

Table.1 Experimental results for the UAV road planning problem

Algorithm	Valueofoptimalvariable											Minimumcost	Rank
		1	2	3	4	5	6	7	8	9	10		
IEEFO	$X_{1\sim10}$	269.041	225.548	225.627	226.024	226.302	226.413	226.420	226.464	226.439	226.580	4665	1
	$Y_{1\sim10}$	225.548	225.627	226.023	226.302	226.412	226.420	226.464	226.439	226.579	226.952		
	$Z_{1\sim10}$	149.880	149.860	149.870	149.871	149.889	149.848	149.797	149.846	149.875	149.952		
HHO	$X_{1\sim10}$	267.649	274.220	279.957	279.199	274.295	273.294	275.591	277.558	278.848	281.901	5381	5
	$Y_{1\sim10}$	231.543	236.158	235.062	233.654	233.789	234.874	234.854	235.404	234.747	238.116		
	$Z_{1\sim10}$	149.382	148.517	148.907	149.290	150.399	150.483	149.159	148.858	149.152	149.251		
EEFO	$X_{1\sim10}$	262.111	264.254	264.826	265.415	265.544	265.421	265.472	265.953	267.015	268.390	4818	2

	$Y_{1\sim10}$	218.135	222.133	223.951	224.614	224.712	224.489	224.802	225.484	227.305	228.286		
	$Z_{1\sim10}$	149.532	149.727	149.582	149.576	149.836	149.776	149.802	149.831	149.761	149.691		
ESC	$X_{1\sim10}$	222.601	246.473	297.140	355.864	398.672	447.447	496.945	557.137	605.889	675.401	5490	7
	$Y_{1\sim10}$	295.013	411.299	495.275	552.422	589.067	616.598	639.472	650.076	678.847	720.718		
	$Z_{1\sim10}$	151.363	150.277	150.807	148.911	150.766	150.244	149.785	149.693	149.900	150.184		
LFD	$X_{1\sim10}$	173.148	282.037	245.376	424.923	486.942	410.093	539.838	510.968	552.979	617.785	17329	
	$Y_{1\sim10}$	238.811	390.385	506.435	593.759	542.180	567.710	504.010	421.428	512.991	560.111		
	$Z_{1\sim10}$	152.284	153.544	159.776	148.213	163.000	154.394	146.526	148.954	136.507	154.921		
MFO	$X_{1\sim10}$	246.794	288.863	339.323	394.560	472.211	517.688	572.731	637.886	696.349	751.429	5706	6
	$Y_{1\sim10}$	205.135	317.434	418.281	493.993	564.936	588.499	624.927	671.226	705.586	750.757		
	$Z_{1\sim10}$	149.836	149.608	149.952	150.239	149.036	150.118	150.416	149.377	149.193	149.188		
ETO	$X_{1\sim10}$	106.061	1.645	1.742	1.333	1.011	1.150	85.575	144.752	257.680	534.867	7487	10
	$Y_{1\sim10}$	106.274	101.575	108.503	132.389	195.102	202.752	352.138	481.547	563.760	668.402		
	$Z_{1\sim10}$	139.399	1151.131	143.946	149.937	149.622	148.318	150.000	142.432	138.576	158.6831		
WOA	$X_{1\sim10}$	216.299	230.979	233.946	239.587	240.639	241.010	240.342	240.043	244.286	271.548	5888	8
	$Y_{1\sim10}$	154.095	197.076	205.341	206.715	206.174	202.548	197.979	200.714	212.065	293.029		
	$Z_{1\sim10}$	149.932	150.065	150.085	149.950	149.968	149.866	150.014	149.968	150.024	149.871		
SCA	$X_{1\sim10}$	49.247	6.724	6.597	9.777	4.655	15.948	4.016	5.152	19.488	110.122	10501	13
	$Y_{1\sim10}$	29.677	3.652	7.627	17.479	14.900	12.086	13.764	9.883	5.375	63.928		
	$Z_{1\sim10}$	142.158	155.377	159.175	152.647	151.143	153.099	153.391	147.790	156.964	153.096		

4. Conclusion

In this study, an improved optimisation algorithm for electric eel foraging is proposed, which significantly improves the algorithm's global search ability, convergence speed and ability to get rid of local optimal solutions. In practical engineering applications, IEEFO shows good competitiveness and practicability, and can effectively solve high-dimensional nonlinear multi-constraint optimisation problems.

5. Acknowledgements

1.This study was supported by the National Natural Science Foundation of China (42361007), by Reserve Talents Project for Young and Middle-aged Academic and Technical Leaders in Yunnan Province (202205AC160005), and by Yunnan Province "Xing Dian Ying Talent Support Programme" Young Talents Project (KKXX202303001).

2.This work was supported in part by the National Natural Science Foundation of China under Grant 62461023; in part by the Key Industry Science and Technology Service Project of Yunnan Provincial Universities under Grant FWCY-ZNT2024025.

6. References

[1]Weiguo Zhao, Liying Wang, Zhenxing Zhang, Honggang Fan, Jiajie Zhang, Seyedali Mirjalili, Nima Khodadadi, Qingjiao Cao,Electric eel foraging optimization: A new bio-inspired optimizer for engineering applications,Expert Systems with Applications,Volume 238, Part F,2024,122200,ISSN 0957-4174,https://doi.org/10.1016/j.eswa.2023.122200.

[2] Haiyang Liu, Xingong Zhang, Hanxiao Zhang, Chunyan Li, Zhaohui Chen,A reinforcement learning-based hybrid Aquila Optimizer and improved Arithmetic Optimization Algorithm for global optimization,Expert Systems with Applications,Volume 224,2023,119898,ISSN 0957-4174,https://doi.org/10.1016/j.eswa.2023.119898.

[3] Guang He, Xiao-li Lu,Good point set and double attractors based-QPSO and application in portfolio

with transaction fee and financing cost,Expert Systems with Applications,Volume 209,2022,118339,ISSN 0957-4174,https://doi.org/10.1016/j.eswa.2022.118339.

[4] Storn, R., Price, K. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization* **11**, 341–359 (1997). https://doi.org/10.1023/A:1008202821328

[5] Hu, G., Chen, L., Wang, X. et al. Differential Evolution-Boosted Sine Cosine Golden Eagle Optimizer with Lévy Flight. J Bionic Eng 19, 1850–1885 (2022). https://doi.org/10.1007/s42235-022-00223-y

[6] Iman Ahmadianfar, Ali Asghar Heidari, Amir H. Gandomi, Xuefeng Chu, Huiling Chen,RUN beyond the metaphor: An efficient optimization algorithm based on Runge Kutta method,Expert Systems with Applications,Volume 181,2021,115079,ISSN 0957-4174,https://doi.org/10.1016/j.eswa.2021.115079.

[7] Manh Duong Phung, Quang Phuc Ha,Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization,Applied Soft Computing,Volume 107,2021,107376,ISSN 1568-4946,https://doi.org/10.1016/j.asoc.2021.107376.