引入改进蝠鲼觅食优化算法的水下无人航行器三维路径规划

黄鹤[1,2]，李潇磊[1,2]，杨澜[3]，王会峰[1]，茹锋[1,2]
(1. 长安大学电子与控制工程学院，710064，西安；2. 长安大学西安市智慧高速公路信息融合与控制重点实验室，710064，西安；3. 长安大学信息工程学院，710064，西安)

摘要：针对复杂环境下传统群体智能优化算法在求解水下无人航行器(UUV)路径规划的过程中存在路径搜索能力不足、易陷入局部最优等问题，提出了一种引入改进蝠鲼觅食优化算法的 UUV 三维路径规划方法。首先，根据 UUV 在水下航行时的实际环境，建立相关地形模型和威胁源模型；其次，对传统的蝠鲼觅食优化算法进行改进，相关改进包括在初始化过程中加入局部反向学习机制优化种群的位置，提高了种群的多样性；根据每次迭代后种群个体适应度的不同，改进蝠鲼翻滚觅食的翻滚因子 S，由此实现一种自适应翻滚，有利于跳出局部最优；同时，在蝠鲼螺旋觅食过程中融合莱维飞行-柯西变异策略，扩大了搜索路径和种群搜索范围，提升了算法寻找全局最优的能力；最后，将改进的蝠鲼觅食优化算法引入到 UUV 的路径规划中，进行相应的实验模拟。实验结果表明：在地形 1 中采用改进的蝠鲼觅食优化算法所规划的路径相比于灰狼算法和蝠鲼觅食优化算法分别降低了 32.49 km 和 23.88 km，航迹代价分别降低了 9.68 和 4.04；在地形 2 中采用改进的蝠鲼觅食优化算法所规划的路径相较于灰狼算法和蝠鲼觅食优化算法分别降低了 20.83 km 和 29.95 km，航迹代价分别降低了 10.14 和 3.18；同时，所提路径规划方法能够使 UUV 有效地避开障碍物、威胁物等，较大地降低了风险成本，安全性更高。

关键词：水下无人航行器；路径规划；蝠鲼觅食优化算法；全局最优
中图分类号：TP181 **文献标志码**：A
DOI：10.7652/xjtuxb202207002 **文章编号**：0253-987X(2022)07-0009-10

OSID 码

Three Dimensional Path Planning of Unmanned Underwater Vehicle Based on Improved Manta Ray Foraging Optimization Algorithm

HUANG He[1,2], LI Xiaolei[1,2], YANG Lan[3], WANG Huifeng[1], RU Feng[1,2]
(1. School of Electronics and Control Engineering, Chang'an University, Xi'an 710064, China; 2. Xi'an Key Laboratory of Intelligent Expressway Information Fusion and Control, Chang'an University, Xi'an 710064, China; 3. School of Information Engineering, Chang'an University, Xi'an 710064, China)

Abstract: As the traditional swarm intelligence optimization algorithm is not capable enough in path search and is easy to fall into local optimum in complex environment, the improved manta ray foraging optimization (IMRFO) algorithm was proposed for three dimensional path planning of an unmanned underwater vehicle (UUV). First, terrain and threat source models were established based on real underwater environment. Second, the traditional MRFO algorithm was improved, including the application of local reverse learning mechanism during initialization to

收稿日期：2021-12-27。 作者简介：黄鹤(1979—)，男，教授，博士生导师；杨澜(通信作者)，女，博士，高级工程师，硕士生导师。 基金项目：国家重点研发计划资助项目(2021YFB2501200)；国家自然科学基金资助项目(52172324)；西安市智慧高速公路信息融合与控制重点实验室(长安大学)开放基金资助项目(300102321502)。
网络出版时间：2022-04-08 网络出版地址：http://kns.cnki.net/kcms/detail/61.1069.T.20220407.1639.002.html

optimize location of population and improve the population diversity. The individual fitness of the population after each iteration was based on to improve the tumbling factor S of the manta ray rolling foraging and thus realize an adaptive rolling which was conducive to getting out of local optimum. At the same time, integrating Levy flight-Cauchy mutation strategy in the manta ray spiral foraging process expanded the search path and population search range and improved the algorithm's ability to find the global optimum. Finally, the improved algorithm was applied to path planning of the UUV for experimental simulation. The results show that in terrain 1, the path planned with the improved algorithm reduces by 32.49 km and 23.88 km respectively and the track cost reduces by 9.68 and 4.04 respectively compared with the Wolf algorithm and the traditional MRFO algorithm and that in terrain 2, these reductions were 20.83 km and 29.95 km respectively and 10.14 and 3.18 respectively. Moreover, the proposed algorithm can enable the UUV to avoid obstacles, threats, etc., thus reducing risk cost and improving safety.

Keywords: underwater unmanned vehicle; path planning; manta ray foraging optimization algorithm; global optimum

水下无人航行器(unmanned underwater vehicle,UUV)可用于水下侦查敌情、海域巡逻、反击敌方潜艇等[1-2]。在 UUV 的研究中,路径规划问题非常重要,其目的是完成从航程起点到终点的规划,躲避障碍物和威胁物,同时尽量选择行驶路径短的路线。近年来关于 UUV 的路径规划问题,各国学者做了大量的研究工作,提出了很多 UUV 路径规划的算法,经典算法有人工势场法[3]、A* 算法[4]和 D* 算法[5]等。王健等[6]提出了一种基于混合势场法的 UUV 路径规化方法,用于 UUV 航向保持和避障,但是所设计的虚拟力比较难实现。相对于经典算法,智能算法收敛速度快,鲁棒性强,尤其采用群体智能优化算法求解 UUV 路径规划问题是目前的研究热点,常用的智能算法有粒子群算法[7]、遗传算法[8]、蚁群算法[9]、灰狼算法(grey wolf optimizer,GWO)[10]等。徐炜翔等[11]针对静态的地形和复杂环境下 UUV 的路径规划问题,提出了基于飞蛾扑火算法的 UUV 三维全局路径规划,但是搜索时间过长。张汝波等[12]针对水下可能存在不同等级的威胁源,提出了一种基于改进蚁群算法的 UUV 航路避障任务规划方法,但是搜索范围过大,搜索耗时较长。以上研究虽然能够实现 UUV 的路径规划,但是改进效果有待于进一步提高。在三维 UUV 路径规划中,地形和威胁的信息量比较大,更使得群体智能优化算法寻求最优路径时复杂程度会呈现指数上升[13]。蝠鲼觅食优化算法(manta ray foraging optimization algorithm,MRFO)[14]是 2020 年提出的一种新型群体智能优化算法,该算法模仿蝠鲼在海洋中觅食的过程,针对蝠鲼的链式觅食、螺旋觅食以及翻滚觅食进行相应的数学建模,描述了蝠鲼个体位置更新方式,具有收敛速度快,寻优能力强等特点。MRFO 算法因具有寻优能力强,收敛速度快等特点[15],在复杂环境下也可以保持良好的寻优能力,适用于三维 UUV 的路径规划,但是 MRFO 算法有时会陷入局部最优,需要进一步改进。因此,本文结合蝠鲼的海洋生物特性,提出了一种改进的蝠鲼觅食优化算法,并应用于 UUV 三维路径规划中,可以避免陷入局部最优,提高 UUV 寻优精度,有效避开水下威胁物和障碍物,快速获得最优路径。

1 水下无人航行器三维路径规划环境建模

1.1 地形约束

水下海拔为负数,但是为了计算方便,这里仿照同类研究文献[16],将水下海拔都以海底为 0 开始按正数赋值,根据水下海拔的不同,可以将水下地形分为多种地区。这里面影响比较大的因素是山峰。首先,水下山峰建模如下式

$$Z(X,Y)=h\exp\left(\left(-\frac{(X-X_0)^2}{m_1}-\frac{(Y-Y_0)^2}{m_2}\right)^2\right) \tag{1}$$

式中:(X,Y)为水底平面以上山峰对应的坐标;(X_0,Y_0)为山峰中心点对应的坐标;h 为高度参数;m_1 和 m_2 为山峰的陡峭程度。规划的路径用航迹点来表示。第 i 个航迹点对应的地形威胁代价表示为

$$f_{Z_i}=\begin{cases}K_z, & h_i-Z_i<0\\ 0, & h_i-Z_i\geqslant 0\end{cases} \tag{2}$$

式中:K_z 为地形威胁系数;h_i 为第 i 个航迹点对应

的海拔高度；Z_i 为第 i 个航迹点距离海底的垂直高度。此外，UUV 路径规划需考虑下潜的深度，下潜的深度必须处于最大和最小下潜深度之间，即为深度约束，第 i 个航迹点对应的深度威胁代价函数表示为

$$f_{\mathrm{H}_i}=\begin{cases}0, & h_{\min}<h_i<h_{\max}\\ K_{\mathrm{w}}, & h_i<h_{\min}\parallel h_i>h_{\max}\end{cases} \tag{3}$$

式中：$h_{\min}$ 和 $h_{\max}$ 分别表示 UUV 相对海底的最小高度和最大高度；K_{w} 表示深度威胁系数。

1.2　威胁模型约束

(1)声呐威胁。声呐通过声波探测敌方目标的距离、速度和相对位置等信息，UUV 路径规划过程中必须避开其探测范围。声呐威胁区近似由一个椭球形表示，在三维平面上表示的范围如下

$$S_{\mathrm{th}}=\begin{Bmatrix}P(X,Y,Z)\mid P(X,Y,Z)\in S,\\ (Y_1\in[0,h]\cap[(X-X_1)^2/a^2+(Z-Z_1)/b^2]\leqslant R^2)\end{Bmatrix} \tag{4}$$

式中：S_{th} 表示声呐威胁区；X、Y、Z 分别指三维平面的 X 轴、Y 轴和 Z 轴；X_1、Y_1、Z_1 为对应的球心坐标；a、b 分别为椭球体的长半径和短半径；R 为相应的球体的半径。

(2)洋流威胁。UUV 在水下航行时，要尽量避免航线经过强洋流地带。本文强洋流地带可以用下式表示

$$S_{\mathrm{o}}=\begin{Bmatrix}P(X,Y,Z)\mid P(X,Y,Z)\in S,\\ (Y_1\in[0,h]\cap[(X-X_1)^2+(Z-Z_1)\leqslant R^2)\end{Bmatrix} \tag{5}$$

式中：S_{o} 表示洋流威胁区。

(3)水雷威胁。敌方水雷的威胁范围内被称作禁游区，可以近似为一个半球体，表示为

$$P_{\mathrm{M}}=P_{\mathrm{v}}P_{\mathrm{k/v}} \tag{6}$$

$$P_{\mathrm{v}}=K_0\frac{\Delta h_{\mathrm{AS}}}{R_{\mathrm{S}}}=K_0\sin\theta,\ 0^\circ\leqslant\theta\leqslant 90^\circ \tag{7}$$

式中：$P_{\mathrm{k/v}}$ 表示 UUV 在禁游区半径内被摧毁的概率，是常数；K_0 为比例系数；R_{S} 为水雷威胁中心与 UUV 之间的斜距；θ 为视线俯视角。则

$$P_{\mathrm{M}}=K_0\sin\theta\cdot P_{\mathrm{k/v}} \tag{8}$$

在三维模型创建的过程中，威胁源半径比较大，因此可以将所有的威胁源等效为圆柱地形处理。威胁代价表示如下

$$f_{\mathrm{T},ij}=\begin{cases}K_{\mathrm{T}j}/r_{ij}^4, & r_{ij}\leqslant R_{\mathrm{T}j}\\ 0, & r_{ij}\geqslant R_{\mathrm{T}j}\end{cases} \tag{9}$$

式中：$K_{\mathrm{T}j}$ 为威胁源 j 的威胁系数；r_{ij} 为第 i 个航迹点到威胁源 j 的直线距离；$R_{\mathrm{T}j}$ 表示的是威胁源的威胁半径。

1.3　UUV 自身物理约束

UUV 航行过程中会受到自身转弯角 α_0 和爬升角 β_0 的约束。同时，还有自身的燃油代价，可以用路程来表示。设最大转弯角为 $\alpha_{\max}$，最大爬升角为 $\beta_{\max}$。转弯角、爬升角物理约束分别表示如下

$$J_{\mathrm{a}}=\begin{cases}0, & \alpha_0<\alpha_{\max}\\ K_{\mathrm{h}}, & \alpha_0>\alpha_{\max}\end{cases} \tag{10}$$

$$J_{\mathrm{v}}=\begin{cases}0, & \beta_0<\beta_{\max}\\ K_{\mathrm{v}}, & \beta_0>\beta_{\max}\end{cases} \tag{11}$$

式中：K_{h} 和 K_{v} 分别为转弯角和爬升角威胁系数；J_{a} 和 J_{v} 为第 i 个航迹点对应 α_0 和 β_0 的代价函数。综合各代价函数，可得出第 i 个航迹点的 UUV 物理约束代价函数为

$$f_{\mathrm{J}i}=J_{\mathrm{a}}+J_{\mathrm{v}}+J_{\mathrm{L}i} \tag{12}$$

路程代价物理约束为

$$J_{\mathrm{L}i}=\begin{cases}l_i, & i>1\\ 0, & i=1\end{cases} \tag{13}$$

1.4　航迹代价函数

将所有地形约束、深度代价、威胁模型约束以及 UUV 的自身物理约束加权综合起来，构成最终 UUV 的代价函数，如下式

$$F(R)=\sum_{i=1}^{n}\left(\omega_1 f_{Z_i}+\omega_2 f_{\mathrm{H}_i}+\omega_3\sum_{j=1}^{n_j}f_{\mathrm{T},ij}+\omega_4 f_{\mathrm{J}i}\right) \tag{14}$$

式中：$F(R)$ 就是整条航迹的代价；ω_1、ω_2、ω_3、ω_4 为各代价的权重。

2　改进的蝠鲼觅食算法

2.1　经典 MRFO 算法

2.1.1　链式觅食[17]

链式觅食更新方式的数学模型如下

$$x_i^d(t+1)=\begin{cases}x_i^d(t)+r(x_{\mathrm{best}}^d(t)-x_i^d(t))+\\ \quad\alpha(x_{\mathrm{best}}^d-x_i^d(t)),i=1\\ x_i^d(t)+r(x_{i-1}^d(t)-x_i^d(t))+\\ \quad\alpha(x_{\mathrm{best}}^d-x_i^d(t)),i>1\end{cases} \tag{15}$$

$$\alpha=2r\sqrt{|\lg r|} \tag{16}$$

式中：$x_i^d(t)$ 表示的是第 t 代第 i 个个体在 d 维上的位置；r 为[0,1]上的随机数；$x_{\mathrm{best}}^d(t)$ 为第 t 代最优个体在 d 维上的位置。

2.1.2 螺旋觅食[18]

螺旋觅食更新方式的数学模型如下。

(1)当 $t/T>r$ 时,蝠鲼螺旋觅食方程为

$$x_i^d(t+1)=\begin{cases}x_{\text{best}}^d(t)+r(x_{\text{best}}^d(t)-x_i^d(t))+\\ \quad\beta(x_{\text{best}}^d-x_i^d(t)),i=1\\ x_{\text{best}}^d(t)+r(x_{i-1}^d(t)-x_i^d(t))+\\ \quad\beta(x_{\text{best}}^d-x_i^d(t)),i>1\end{cases}\tag{17}$$

$$\beta=2\exp(r_1\frac{T-i+1}{T})\sin(2\pi r_1)\tag{18}$$

式中:T 为总迭代次数;t 为当前迭代次数;r 和 r_1 都表示在[0,1]上的随机数。

(2)当 $t/T\leqslant r$ 时,蝠鲼螺旋觅食方程为

$$x_i^d(t+1)=\begin{cases}x_r^d(t)+r(x_r^d(t)-x_i^d(t))+\\ \quad\beta(x_r^d-x_i^d(t)),i=1\\ x_r^d(t)+r(x_{i-1}^d(t)-x_i^d(t))+\\ \quad\beta(x_r^d-x_i^d(t)),i>1\end{cases}\tag{19}$$

$$x_r^d=L^d+r(U^d-L^d)\tag{20}$$

式中:$x_r^d(t)$ 为第 t 代 d 维的随机位置;U^d 和 L^d 为变量的上下界。

2.1.3 翻滚觅食

翻滚觅食时,蝠鲼个体会以当前最优解的位置作为翻滚的支点,数学表达式如下

$$x_i^d(t+1)=x_i^d(t)+S(r_2x_{\text{best}}^d-r_3x_i^d(t)),\quad i=1,2,\cdots,N\tag{21}$$

$$S=2\tag{22}$$

式中:r_2 和 r_3 都是在[0,1]上的随机数;S 为翻滚因子;N 为个体的数量。

2.2 改进的蝠鲼觅食优化算法

2.2.1 局部反向学习

反向学习是一种群体智能的改进策略,随机初始化所生成的种群可能位置不够好,不利于全局搜索。因此,初始化过程中需加入反向解搜索[19],增加种群的多样性,有利于寻找全局最优解。

设 $x_i^j=[x_i^1,x_i^2,\cdots,x_i^d](i=1,2,\cdots,N)$ 为 d 维搜索空间的种群个体,其相应的反向种群定义为

$$(x_i^j)'=L_j+U_j-x_i^j\tag{23}$$

式中:x_i^j 为第 i 个个体在 j 维上的位置,$j=1,2,\cdots,d$;U_j 和 L_j 分别为第 j 维搜索空间的上边界和下边界。

在实际搜索过程中,如果对所有初始化的种群都生成其反向解,会增加迭代搜索的时间,因此在经典反向学习的基础上,本文设计了一种局部反向学习方法,只对初始化位置较差的一半种群进行反向学习,其余的保持不变。

局部反向学习策略的具体公式为

$$(x_{ip}^j)'=L_j+U_j-x_{ip}^j\tag{24}$$

$$x_{if}^j=x_{ib}^j+(x_{ip}^j)'\tag{25}$$

式中:x_{ip}^j 表示初始化适应度较差的一半种群;$(x_{ip}^j)'$ 表示生成对应的反向种群;x_{ib}^j 为初始化适应度较优的一半种群;x_{if}^j 为最终迭代的种群。

局部反向学习策略的具体步骤为:

(1)随机生成初始化种群 $x_i^j=[x_i^1,x_i^2,\cdots,x_i^d]$;

(2)计算所生成种群的适应度;

(3)对种群的适应度进行 sort 排序,筛选出适应度较差的一半种群 x_{ip}^j 和适应度较优的一半种群 x_{ib}^j;

(4)对所筛选出的适应度较差的一半种群进行反向学习,求出相应的反向个体 $(x_{ip}^j)'$;

(5)将种群 $(x_{ip}^j)'$ 和 x_{ib}^j 合并,作为最终迭代的种群。

这样相比于经典的反向学习,提出的局部反向学习不仅能够获得位置较优的种群,提高全局搜索能力。同时,只对适应度较差的一半种群进行反向求解,使得搜索时间更短,提高了寻优速度。

2.2.2 自适应翻滚

MRFO 算法中翻滚觅食是一个比较频繁的动作,可以提高蝠鲼的觅食效率。在其数学模型中有一个翻滚因子 S,决定了翻滚的距离。在传统的 MRFO 算法中 S 一般取 2,但是因为翻滚因子 S 在蝠鲼的迭代更新过程中不会发生变化,所以搜索过程中易陷入局部最优。因此,本文提出一种新的自适应翻滚方式,根据每次迭代后种群的适应度不同,在翻滚因子中加入一个变化的翻滚系数 ρ,翻滚系数 ρ 随着蝠鲼种群位置的适应度变化而变化,使得翻滚因子 S 能够随着种群适应度的变化而做出动态调整,有效地扩大了搜索范围,有利于跳出局部最优。加入 ρ 后蝠鲼种群的变化方式变得更灵活,有利于寻求全局最优,提升寻优精度。翻滚系数的更新方式如下式

$$\rho=5+(0.25+\mathrm{e}^{(f_{\max}^i/f_{\text{best}}^i)})^{-1}\tag{26}$$

式中:$f_{\max}^i$ 为第 i 次迭代时蝠鲼最大的适应度;f_{best}^i 为第 i 次迭代时蝠鲼个体的最优适应度。在翻滚因子 S 中加入翻滚系数 ρ 后,S 的变化如图 1 所示。

从图 1 可以看出,翻滚因子 S 随着迭代的进行不断变化,在迭代前期、中期,因种群个体适应度与最优种群适应度差异较大,赋予翻滚系数 ρ 一个变化较大的值,导致翻滚因子 S 波动剧烈,扩大了蝠鲼种群的翻滚幅度,可以使蝠鲼种群在下一次迭代

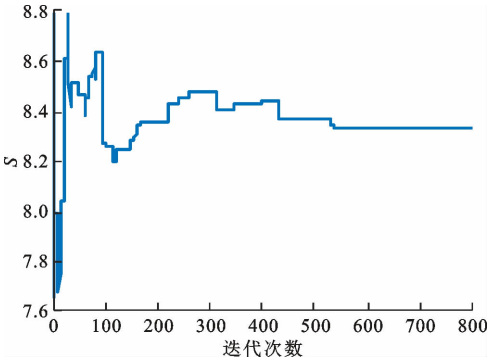


图 1　翻滚因子 S 的变化

Fig. 1　Variation of rolling factor S

过程中位置多样性增强，有利于寻找全局最优。随着迭代的进行，蝠鲼种群逐渐搜索到最优解，这样适应度的变化幅度也相应减小，S 也逐渐趋于稳定，蝠鲼聚集在最优解周围增强了局部的搜索能力。加入翻滚系数 ρ 后蝠鲼翻滚觅食的更新方式转换为

$$x_i^d(t+1)=x_i^d(t)+\rho s(r_2x_{\text{best}}^d-r_3x_i^d(t)),\quad i=1,2,\cdots,N \tag{27}$$

2.2.3　融合莱维飞行-柯西变异策略

本文提出一种融合莱维飞行-柯西变异的策略，首先在蝠鲼的螺旋觅食中加入莱维飞行策略，增加了种群位置的多样性，扩大了搜索范围。莱维飞行是一种随机游走策略[20-21]，结合长距离游走和短距离游走。执行过程中，通过长距离游走扩大搜索范围，跳出局部最优；通过短距离搜索加快收敛速度。这样，在蝠鲼的螺旋觅食中加入莱维飞行，可以扩大搜索范围，同时也加快了收敛速度。相较于其他的改进方法，莱维飞行的随机游走性更强[22]。同时在翻滚觅食后，对当前蝠鲼的最优位置进行柯西变异，柯西函数因为其分布范围广，可以进一步提升蝠鲼的全局搜索能力。莱维飞行的具体描述如下

$$L(\lambda)=\frac{\mu}{|\nu|^{1/\delta}} \tag{28}$$

式中：$\delta=1.5$；μ 和 ν 则服从正态分布

$$\mu\sim N(0,\sigma_\mu^2) \tag{29}$$

$$\nu\sim N(0,\sigma_\nu^2) \tag{30}$$

$$\sigma_\mu=\left\{\frac{\Gamma(1+\delta)\sin(\pi\delta/2)}{2^{(\delta-1)/2}\Gamma[(1+\delta)/2]\delta}\right\}^{1/\delta} \tag{31}$$

式中：$\Gamma(\cdot)$ 为伽马函数；$\sigma_\nu=1$。

增加莱维飞行后，式(17)和(19)可转换为

$$x_i^d(t+1)=\begin{cases}x_{\text{best}}^d(t)+r(x_{\text{best}}^d(t)-x_i^d(t))+\\ \quad\beta(x_{\text{best}}^d-x_i^d(t))L(\lambda),i=1\\ x_{\text{best}}^d(t)+r(x_{i-1}^d(t)-x_i^d(t))+\\ \quad\beta(x_{\text{best}}^d-x_i^d(t))L(\lambda),i>1\end{cases} \tag{32}$$

$$x_i^d(t+1)=\begin{cases}x_r^d(t)+r(x_r^d(t)-x_i^d(t))+\\ \quad\beta(x_r^d-x_i^d(t))L(\lambda),i=1\\ x_r^d(t)+r(x_{i-1}^d(t)-x_i^d(t))+\\ \quad\beta(x_r^d-x_i^d(t))L(\lambda),i>1\end{cases} \tag{33}$$

采用的柯西分布[23]和变异函数如下式

$$\text{Cauchy}(x_0)=\frac{1}{\pi(1+x_0^2)} \tag{34}$$

$$x_{\text{newbest}}=x_{\text{best}}+\text{Cauchy}(x_0)x_{\text{best}} \tag{35}$$

式中：x_{newbest} 为更新后蝠鲼的位置；x_{best} 为更新前蝠鲼的位置；柯西函数中 x_0 取值为 0～1 的随机数。柯西分布函数的中间峰值比较小而两边分布的范围比较广，可以使蝠鲼在当前范围产生较大的扰动，帮助跳出局部最优。

2.2.4　本文算法流程

本文算法流程如图 2 所示。

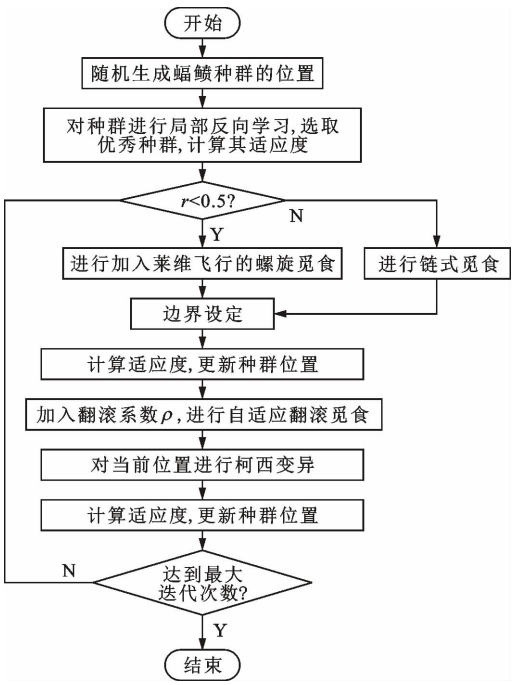


图 2　本文算法流程图

Fig. 2　The flow chart of this algorithm

2.3　性能仿真实验

2.3.1　测试函数

根据路径规划需要，利用测试函数对本文算法的性能进行分析比较。其中，单峰值函数可以测试算法的收敛精度，多峰值函数可以测试算法的寻优能力。分别选取 3 个单峰值函数和 3 个多峰值函数进行测试。

(1)Schwefel 1.2 函数表达式为

$$f(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_j\right)^2 \tag{36}$$

式中：$n=30$；$x_j \in [-100,100]$；全局最小值为 0。

(2)Schwefel 2.21 函数表达式为

$$f(x) = \max_i\{|x_i|, 1 \leqslant i \leqslant n\} \tag{37}$$

式中：$n=30$；$x_i \in [-100,100]$；全局最小值为 0。

(3)Step 函数表达式为

$$f(x) = \sum_{i=1}^{n}(x_i + 0.5)^2 \tag{38}$$

式中：$n=30$；$x_i \in [-100,100]$；全局最小值为 0。

(4)Ackley 函数表达式为

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right) + 20 + \mathrm{e} \tag{39}$$

式中：$n=30$；$x_i \in [-32,32]$；全局最小值为 0。

(5)Griewank 函数表达式为

$$f(x) = \frac{1}{4\,000}\sum_{i=1}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i - 100}{\sqrt{i}}\right) + 1 \tag{40}$$

式中：$n=30$；$x_i \in [-600,600]$；全局最小值为 0。

(6)Shekel 10 函数表达式为

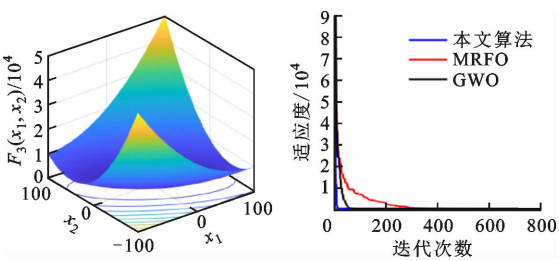
$$f(x) = -\sum_{i=1}^{10}\left|(\boldsymbol{x}_{ki} - \boldsymbol{a}_{ki})(\boldsymbol{x}_{ki} - \boldsymbol{a}_{ki})^{\mathrm{T}} + \boldsymbol{c}_{ki}\right|^{-1} \tag{41}$$

式中：$\boldsymbol{x}_{ki}$ 为矩阵 $\boldsymbol{X}_k$ 中第 i 行的向量；$\boldsymbol{a}_{ki}$ 为矩阵 $\boldsymbol{A}_k$ 中第 i 行的向量；$\boldsymbol{c}_{ki}$ 为向量 $\boldsymbol{C}_k$ 的第 i 个元素；矩阵 $\boldsymbol{X}_k$ 和矩阵 $\boldsymbol{A}_k$ 均为 10 行 4 列的矩阵；$\boldsymbol{x}_{ki} \in [0,10]$；$\boldsymbol{a}_{ki} \in [0,10]$；$\boldsymbol{c}_{ki} \in [0,1]$；该函数的全局最小值为 −10.53。

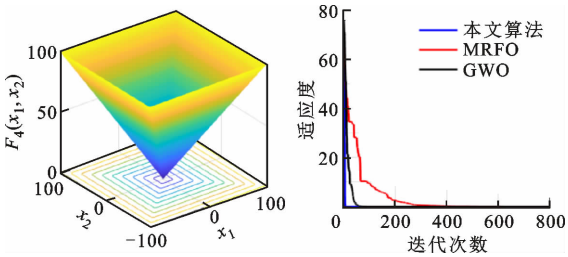
2.3.2 实验验证

采用上述测试函数评估 MRFO 算法、GWO 算法[24]及本文算法的寻优性能。其中，种群规模为 100，迭代次数为 800。图 3 为各函数的三维参数空间以及各算法在不同测试函数上的适应度曲线，表 1 为经过 30 次测试的适应度均值和最优值。

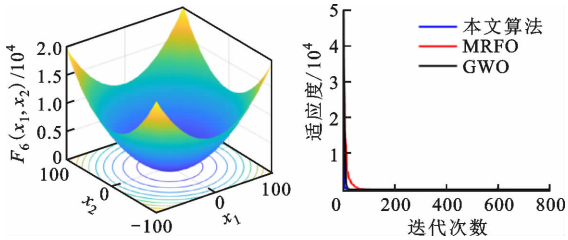
由图 3 和表 1 可知，在 Schwefel 1.2 函数和 Schwefel 2.21 函数中，MRFO 算法的收敛精度不如 GWO 算法，但是本文算法收敛精度却是最好的，强于 GWO 算法。在 Step 函数中 MRFO 算法的收敛精度优于 GWO 算法，本文算法在 MRFO 算法基础上有了一定提升。在 Ackley 函数中，本文算法寻优能力和速度都优于 GWO 和 MRFO 算法。说明



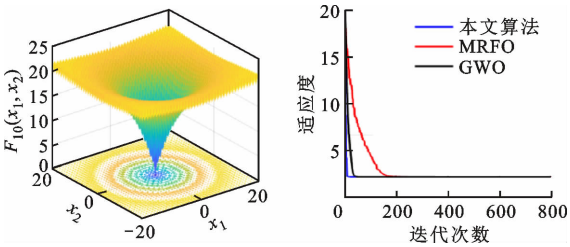
(a)Schwefel 1.2 函数



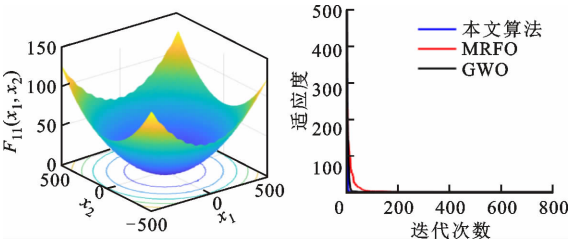
(b)Schwefel 2.21 函数



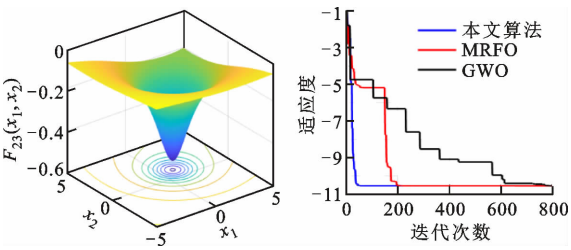
(c)Step 函数



(d)Ackley 函数



(e)Griewank 函数



(f)Shekel 10 函数

图 3 不同函数三维参数空间和各算法适应度曲线

Fig. 3 3D parameter space of different functions and adaptation curves of each algorithm

本文算法可以有效地跳出局部最优，寻找全局最优。在 Griewank 函数中，GWO 和本文算法寻优能力大致相同，但是寻优速度不如本文算法。在 Griewank 函数中，GWO 算法的寻优能力和寻优速度都不如 MRFO 算法，本文算法最高。总体来说改进后的本文算法可以提高 MRFO 算法的寻优精度、寻优能力以及寻优的速度，有利于跳出局部最优。

表 1　3 种算法在测试函数上的实验对比

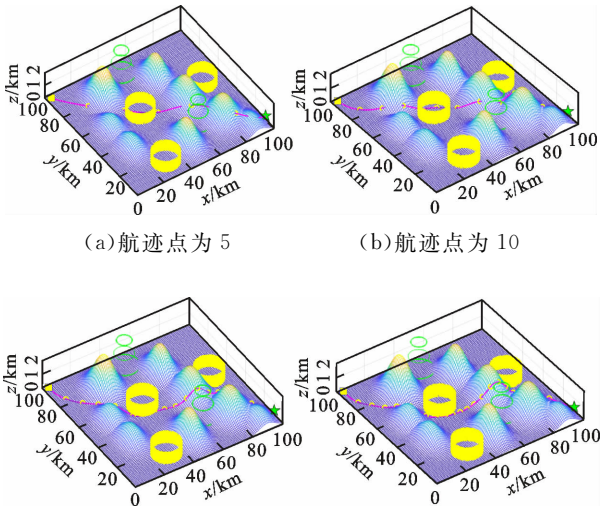
Table 1　Experimental comparison of three algorithms on Test Functions

测试函数	算法	适应度最优值	适应度均值
Schwefel 1.2 函数	MRFO 算法	0.05	0.29
	GWO 算法	1.13×10^{-25}	2.14×10^{-19}
	本文算法	5.17×10^{-191}	5.62×10^{-162}
Schwefel 2.21 函数	MRFO 算法	1.77×10^{-3}	8.57×10^{-3}
	GWO 算法	1.64×10^{-18}	2.54×10^{-17}
	本文算法	6.58×10^{-103}	6.19×10^{-92}
Step 函数	MRFO 算法	6.28×10^{-28}	6.26×10^{-26}
	GWO 算法	1.31×10^{-5}	0.27
	本文算法	3.32×10^{-30}	2.41×10^{-28}
Ackley 函数	MRFO 算法	1.51×10^{-14}	2.93×10^{-14}
	GWO 算法	7.99×10^{-15}	1.23×10^{-14}
	本文算法	8.88×10^{-16}	8.88×10^{-16}
Griewank 函数	MRFO 算法	0	1.53×10^{-2}
	GWO 算法	0	0
	本文算法	0	0
Shekel 10 函数	MRFO 算法	−10.53	−10.00
	GWO 算法	−10.53	−9.72
	本文算法	−10.53	−10.53

3　引入改进蝠鲼觅食优化算法的 UUV 三维路径规划

本文算法中每条路径由每只蝠鲼来代表，定义种群中的一只蝠鲼即为一条路径 $X_i^n=[x_i^1,x_i^2,\cdots,x_i^n]$，一共有 n 个航迹点。每条路径都是由多个航迹点连接而成的线。每个航迹点 x_i^n 具有空间三维属性(x,y,z)。利用本文算法规划 UUV 三维的突防路径时，确定航迹点数 n，x 方向的航迹点坐标固定不变，将三维等效为二维，通过对 y 和 z 坐标的寻优确定最优路线。

航迹点数 n 对于 UUV 的路径规划很重要，相关研究表明，路径的精度随着 n 的增加变得更精确，但是如果 n 的数量过多，不仅会提高计算复杂度，还影响路径优化的效率[25]。本文设置了航迹点数分别为 5、10、15、20，结合本文提出的改进蝠鲼觅食优化算法对路径进行预先规划，以确定合适的航迹点数。图 4 为不同航迹点下利用改进蝠鲼觅食优化算法所规划的路径。



(a)航迹点为 5　(b)航迹点为 10

(c)航迹点为 15　(d)航迹点为 20

图 4　不同航迹点下利用改进蝠鲼觅食优化算法所规划的路径

Fig. 4　Path planning by improved manta ray foraging optimization algorithm at different track points

从图 4 中可以看出，当航迹点为 5 时，规划路径不能有效规避水下山脉等障碍物，与理想路径有较大差距。当航迹点为 10 时，比航迹点为 5 时规划的路径避障能力更强，但是精度仍然不够，不能规避所有障碍物。航迹点为 15 时，规划的路径准确，可以较好地实现水下避障，寻找较短的路径，实现较理想的路径规划。航迹点为 20 时，规划路径效果相比于航迹点为 15 时规划的路径并没有明显的提升，但是耗时增加很多，因此综合比较可以看出，选择航迹点为 15、可以在耗时较少的情况下，规划出精准的路径。

4　路径规划仿真实验及分析

仿真环境中采用 CPU 为 Intel Core i7-10750H CPU 2.60 GHz、内存为 16 GB 的计算机，操作系统为 Windows10，编译软件为 Matlab R2018b。为了测试本文算法在 UUV 路径规划中的效果，采用两种不同的地形。地形 1 中地形区域大小为 100 km ×100 km，测试中 UUV 距离海底的最大高度为 4 km。UUV 的起始点与终止点分别为(1,3,0.5) km 及(99,98,0.5) km，敌方水雷距离中心为(20,13.5,0) km、(85,60,0) km、(40,59,0) km，分布在半径为 9 km 的区域。声呐距离中心为(65,30,0) km、(50,90,0) km，作用半径为 9 km。路径规划过

程中的主要参数如表 2 所示。地形 1 中 UUV 路径轨迹、代价曲线分别如图 5 和图 6 所示，UUV 的航程和最优代价如表 3 所示。地形 2 中区域大小和地形 1 相同，UUV 的起始点为(1，93，0.5) km，终止点为(99，8，0.5) km。水下水雷的位置为(30，13.5，0) km、(85，55，0) km、(40，55，0) km。声呐的位置为(65，30，0) km、(50，90，0) km。用正方形、星号和圆圈分别表示 MRFO、GWO 和本文算法求得 3 条 UUV 路径的航迹点。在实验模拟中各算法种群数设为 100，最大迭代次数为 800 次。地形 2 中 UUV 的路径轨迹、代价曲线分别如图 7 和图 8 所示，UUV 的航程和最优代价如表 4 所示。

表 2 路径规划过程中的主要参数

Table 2 The main parameters in the path planning

参数	数值
地形威胁系数 K_z	1 000
威胁源威胁系数 K_{Tj}	1 000
深度威胁系数 K_w	10
转弯角威胁系数 K_h	10
俯仰角威胁系数 K_v	10
地形约束权值 ω_1	0.25
深度约束权值 ω_2	0.4
威胁源约束权值 ω_3	0.2
物理约束权值 ω_4	0.15
航迹点数 n	15

表 3 地形 1 中 UUV 的航程及最优代价

Table 3 Range and optimal cost of UUV in terrain 1

算法	航程/km	最优代价
MRFO 算法	158.64	25.37
GWO 算法	167.25	31.01
本文算法	134.76	21.33

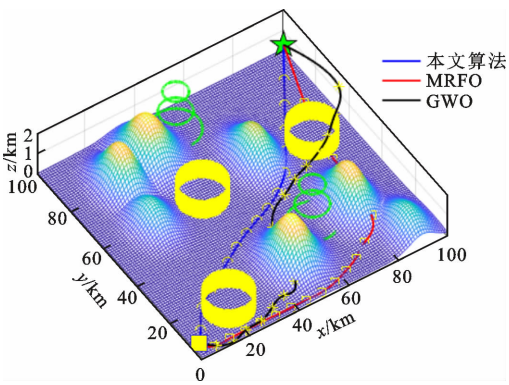


图 5 地形 1 中 UUV 的路径轨迹

Fig. 5 Path tracks of UUV in terrain 1

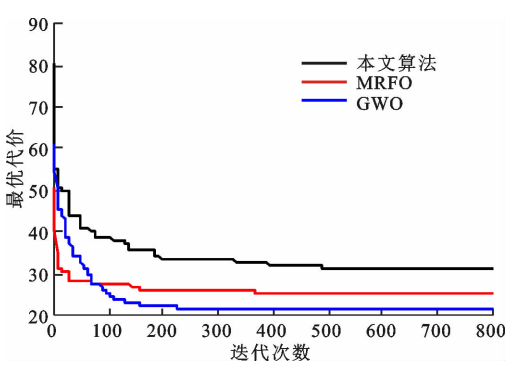


图 6 地形 1 中最优路线代价变化

Fig. 6 Variation map of optimal route cost in terrain 1

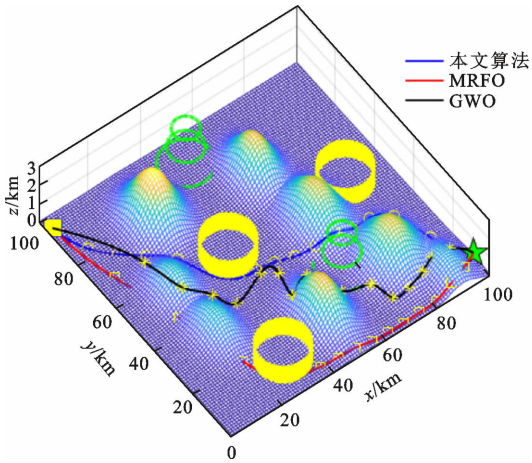


图 7 地形 2 中 UUV 的路径轨迹

Fig. 7 Path tracks of UUV in terrain 2

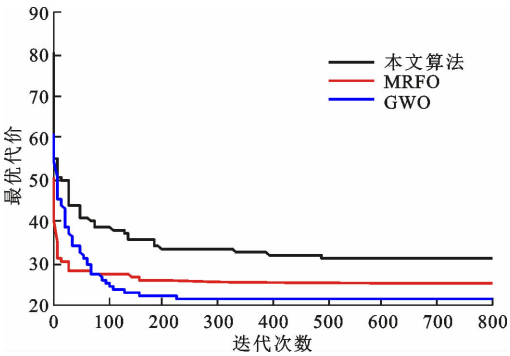


图 8 地形 2 中最优路线代价变化

Fig. 8 Change map of optimal route cost in terrain 2

表 4 地形 2 中 UUV 的航程及最优代价

Table 4 Range and optimal cost of UUV in Terrain 2

算法	航程/km	最优代价
MRFO 算法	168.28	25.59
GWO 算法	159.16	32.55
本文算法	138.33	22.41

在图 5 和图 7 中，两种地形中各有相应的威胁

源和障碍物。根据航迹代价函数式(14)可知，UUV 路径规划时，需要对各地形及水雷、洋流等威胁源同步规避，而最终路径取决于航迹代价函数及优化算法的寻优能力。如果优化算法寻优能力较弱，路径点易陷入局部最优，导致 UUV 进入威胁区或者与障碍物发生碰撞。从图 5 可以看出，地形 1 中 GWO 算法的规划路径会与水下的山脉相撞；MRFO 算法的规划路径在迭代初期可以绕过山脉，但是迭代过程中无法跳过威胁区寻到一条更优的路径，而且后续路径还是会与山脉相撞，这是因为 MRFO 算法寻优能力不足所导致的。本文算法的路径可以有效地避开水下声呐、水雷等威胁区域，同时也能避开水下山脉等障碍物，主要原因是本文算法的寻优能力更强。相对于 GWO 和 MRFO 算法，本文算法规划的路线最优。由表 3 可以看出，GWO 算法的航程最长，代价最大，MRFO 算法相对有所减小，而本文算法的航程和代价最小。相对于 GWO 和 MRFO 算法，本文算法航程分别减少了 32.49 km 和 23.88 km，代价损失分别减小了 9.68 和 4.04。由图 7 可知，UUV 按照 GWO 和 MRFO 算法在地形 2 中规划的路径会与水下山脉相撞，同时不能规避声呐和水雷威胁，而采用本文算法所规划的路径则可以有效规避障碍物和水下威胁，不会与山脉等相撞。由表 4 可以看出，地形 2 中本文算法的航程最短，相对于 GWO 和 MRFO 算法所规划的航程分别减少了 20.83 km 和 29.95 km，代价损失分别减小了 10.14 和 3.18。同时，由图 6 和图 8 还可以看出，GWO 和 MRFO 算法收敛过快，易陷入局部最优，本文算法则可以有效地跳出局部最优，寻优能力更强。这是由于本文算法采用自适应翻滚和莱维飞行-柯西变异策略，面对复杂地形时各路径点寻优能力较强，在威胁源中或者发生碰撞前，可迅速规避并寻得最优路径。

5　结　论

针对 UUV 的路径规划问题，构建了相关的水下地形、威胁以及自身约束组成的三维模型；提出了一种改进的蝠鲼觅食优化算法，解决了 MRFO 算法容易陷入局部最优，收敛精度低的问题；将改进的蝠鲼觅食优化算法应用到 UUV 三维路径规划中，可以有效地减小 UUV 路径规划的航程和代价，躲避水下障碍物和威胁，选择最优路径。

参考文献：

[1] SÁNCHEZ-PÉREZ J F, MASCARAQUE-RAMÍREZ C, MORENO NICOLÁS J A, et al. Study of the application of PCM to thermal insulation of UUV hulls using network simulation method [J]. Alexandria Engineering Journal, 2021, 60(5): 4627-4637.

[2] GALYAEV A A, LYSENKO P V, YAKHNO V P. 2D optimal trajectory planning problem in threat environment for UUV with non-uniform radiation pattern [J]. Sensors, 2021, 21(2): 396.

[3] ZHU Danjie, YANG S X. Path planning method for unmanned underwater vehicles eliminating effect of currents based on artificial potential field [J]. Journal of Navigation, 2021, 74(5): 955-967.

[4] MAHMOUDZADEH S, POWERS D M W, YAZDANI A M, et al. Efficient AUV path planning in time-variant underwater environment using differential evolution algorithm [J]. Journal of Marine Science and Application, 2018, 17(4): 585-591.

[5] 朱蟋蟀，孙兵，朱大奇. 基于改进 D* 算法的 AUV 三维动态路径规划 [J]. 控制工程，2021，28(4)：736-743.
ZHU Xixi, SUN Bing, ZHU Daqi. Three-dimensional dynamic path planning of AUV based on improved D* algorithm [J]. Control Engineering of China, 2021, 28(4): 736-743.

[6] 王健，张华，徐令令，等. 基于混合势场法的无人潜航器路径规划及编队方法研究 [J]. 舰船科学技术，2020，42(12)：97-100.
WANG Jian, ZHANG Hua, XU Lingling, et al. Research on path planning and obstacle avoidance method of underwater unmanned vehicle based on hybrid potential field method [J]. Ship Science and Technology, 2020, 42(12): 97-100.

[7] LIU Xiaohuan, ZHANG Degan, ZHANG Jie, et al. A path planning method based on the particle swarm optimization trained fuzzy neural network algorithm [J]. Cluster Computing, 2021, 24(3): 1901-1915.

[8] CAO Yan, WEI Wanyu, BAI Yu, et al. Multi-base multi-UAV cooperative reconnaissance path planning with genetic algorithm [J]. Cluster Computing, 2019, 22(3): 5175-5184.

[9] 刘贵杰，刘鹏，穆为磊，等. 采用能耗最优改进蚁群算法的自治水下机器人路径优化 [J]. 西安交通大学学报，2016，50(10)：93-98.
LIU Guijie, LIU Peng, MU Weilei, et al. A path optimization algorithm for AUV using an improved ant colony algorithm with optimal energy consumption [J]. Journal of Xi'an Jiaotong University, 2016, 50(10): 93-98.

[10] 王永琦，江潇潇. 基于混合灰狼算法的机器人路径规划 [J]. 计算机工程与科学，2020，42(7)：1294-1301.
WANG Yongqi，JIANG Xiaoxiao. Robot path planning using a hybrid grey wolf optimization algorithm [J]. Computer Engineering and Science，2020，42(7)：1294-1301.

[11] 徐炜翔，朱志宇. 基于飞蛾火焰算法的 AUV 三维全局路径规划 [J]. 上海理工大学学报，2021，43(2)：148-155.
XU Weixiang，ZHU Zhiyu. Three-dimension global path planning for AUV based on moth-flame algorithm [J]. Journal of University of Shanghai for Science and Technology，2021，43(2)：148-155.

[12] 张汝波，李建军，杨玉. 基于改进蚁群算法的 AUV 航路避障任务规划 [J]. 华中科技大学学报(自然科学版)，2015，43(z1)：428-430.
ZHANG Rubo，LI Jianjun，YANG Yu. AUV route planning study for obstacle avoidance task based on improved ant colony algorithm [J]. Journal of Huazhong University of Science and Technology (Natural Science Edition)，2015，43(z1)：428-430.

[13] 黄鹤，吴琨，王会峰，等. 基于改进飞蛾扑火算法的无人机低空突防路径规划 [J]. 中国惯性技术学报，2021，29(2)：256-263.
HUANG He，WU Kun，WANG Huifeng，et al. Path planning of UAV low altitude penetration based on improved moth-flame optimization [J]. Journal of Chinese Inertial Technology，2021，29(2)：256-263.

[14] ZHAO Weiguo，ZHANG Zhenxing，WANG Liying. Manta ray foraging optimization：an effective bio-inspired optimizer for engineering applications [J]. Engineering Applications of Artificial Intelligence，2020，87：103300.

[15] 刘永利，朱亚孟，晁浩. 多策略 MRFO 算法的卷积神经网络超参数优化 [J]. 北京邮电大学学报，2021，44(6)：83-88.
LIU Yongli，ZHU Yameng，CHAO Hao. Hyperparameter optimization of convolutional neural network based on multi-strategy MRFO algorithm [J]. Journal of Beijing University of Posts and Telecommunications，2021，44(6)：83-88.

[16] 朱佳莹，高茂庭. 融合粒子群与改进蚁群算法的 AUV 路径规划算法 [J]. 计算机工程与应用，2021，57(6)：267-273.
ZHU Jiaying，GAO Maoting. AUV path planning based on particle swarm optimization and improved ant colony optimization [J]. Computer Engineering and Applications，2021，57(6)：267-273.

[17] 李璟楠，乐美龙. 多种群蝠鲼觅食优化求解多跑道机场航班排序 [J]. 航空计算技术，2020，50(6)：47-51.
LI Jingnan，LE Meilong. Multi-group manta ray foraging optimization for multi-runway airport flights sequencing [J]. Aeronautical Computing Technique，2020，50(6)：47-51.

[18] 叶剑华，罗凤章，杨理. 基于改进蝠鲼觅食优化 SVM 的配电网拓扑辨识 [J]. 电力系统及其自动化学报，2021，33(10)：43-50.
YE Jianhua，LUO Fengzhang，YANG Li. Distribution network topology identification based on SVM optimized by improved manta ray foraging optimization algorithm [J]. Proceedings of the CSU-EPSA，2021，33(10)：43-50.

[19] GUO Zhaolu，SHI Jinxiao，XIONG Xiaofeng，et al. Chaotic artificial bee colony with elite opposition-based learning [J]. International Journal of Computational Science and Engineering，2019，18(4)：383-390.

[20] YILDIZ B S，PHOLDEE N，BUREERAT S，et al. Comparison of the political optimization algorithm，the Archimedes optimization algorithm and the Levy flight algorithm for design optimization in industry [J]. Materials Testing，2021，63(4)：356-359.

[21] PANDEY A，BANERJEE S. Test data generation and selection using levy flight-based firefly algorithm [J]. International Journal of Software Innovation，2021，9(2)：18-34.

[22] WANG Zhongbin，WU Ziqing，SI Lei，et al. A novel path planning method of mobile robots based on an improved bat algorithm [J]. Proceedings of the Institution of Mechanical Engineers：Part C Journal of Mechanical Engineering Science，2021，235(16)：3071-3086.

[23] LI Jun，LUO Yangkun，WANG Chong，et al. Simplified particle swarm algorithm based on nonlinear decrease extreme disturbance and Cauchy mutation [J]. International Journal of Parallel，Emergent and Distributed Systems，2020，35(3)：236-245.

[24] ZHANG Xiaoqing，ZHANG Yuye，MING Zhengfeng. Improved dynamic grey wolf optimizer [J]. Frontiers of Information Technology & Electronic Engineering，2021，22(6)：877-890.

[25] PANDEY P，SHUKLA A，TIWARI R. Three-dimensional path planning for unmanned aerial vehicles using glowworm swarm optimization algorithm [J]. International Journal of System Assurance Engineering and Management，2018，9(4)：836-852.

（编辑 武红江 刘杨）