DOI:10.16356/j.1005-2615.2025.03.009

基于多策略改进蜣螂算法的三维无人机路径规划

王紫益[1],王 雷[1],徐浩然[1],张桐彬[1],夏强强[2]
(1.安徽工程大学机械与汽车工程学院,芜湖 241000;
2.长三角哈特机器人产业技术研究院,芜湖 241000)

摘要: 针对传统的蜣螂算法在三维无人机(Unmanned aerial vehicle, UAV)路径规划中存在求解精度低、收敛速度慢及容易陷入局部最优等问题,提出了一种基于多策略改进的蜣螂算法(Multi-strategy improved dung beetle optimizer, MSIDBO)。该算法首先采用空间金字塔匹配(Spatial pyramid matching,SPM)混沌映射与反向学习策略进行种群初始化,以提高初始种群的多样性和质量。其次,引入改进后的边界收敛因子,以实现算法全局探索与局部搜索能力的平衡。然后,融合海鸥优化算法的攻击机制,以提升收敛速度和求解精度。最后,采用*t*-distribution差分变异策略,以提高算法跳出局部最优解的能力。将改进的蜣螂算法与其他的启发式算法和相关的改进算法进行基准函数测试,MSIDBO算法相较于其他启发式算法和改进算法,在收敛速度与精度方面表现突出;此外,将改进的蜣螂算法应用于三维无人机路径规划仿真,实验仿真结果表明在不同的场景下MSIDBO算法生成的路径代价函数值更小,路径质量更高,平稳性更佳。

关键词: 蜣螂算法;空间金字塔匹配混沌映射;反向学习;海鸥优化算法;*t*-distribution差分变异
中图分类号: TP301.6;V249 **文献标志码:** A **文章编号:** 1005-2615(2025)03-0475-12

3D Path Planning of UAV Based on Multi-strategy Improved Dung Beetle Algorithm

WANG Ziyi[1], *WANG Lei*[1], *XU Haoran*[1], *ZHANG Tongbin*[1], *XIA Qiangqiang*[2]
(1. School of Mechanical and Automotive Engineering, Anhui Polytechnic University, Wuhu 241000, China;
2. Yangtze River Delta Hart Robot Industry Technology Research Institute, Wuhu 241000, China)

Abstract: Aiming at the problems of low accuracy, slow convergence and local optimality of traditional dung beetle algorithm in 3D unmanned aerial vehicle (UAV) path planning, a multi-strategy improved dung beetle optimizer (MSIDBO) is proposed. Firstly, spatial pyramid matching (SPM) chaotic mapping and reverse learning strategy are used to initialize the population to improve the diversity and quality of the initial population. Secondly, an improved boundary convergence factor is introduced to achieve the balance between global exploration and local search. Then, the attack mechanism of gull optimization algorithm is integrated to improve the convergence speed and solving accuracy. Finally, the *t*-distribution differential variation strategy is used to improve the ability of the algorithm to jump out of the local optimal solution. The improved Dung Beetle algorithm is compared with other heuristic algorithms and related improved algorithms by benchmark function test. Compared with other heuristic algorithms and improved algorithms, MSIDBO algorithm has outstanding performance in convergence speed and accuracy. In addition, the improved Dung Beetle algorithm is applied to 3D UAV path planning simulation. Experimental simulation results show that the path

基金项目: 安徽省高校优秀拔尖人才培育项目(gxbjZD2022023);安徽省机器视觉检测与感知重点实验室开放基金(KLMVI-2024-HIT-15)。
收稿日期: 2025-02-12;**修订日期:** 2025-03-19
通信作者: 王雷,男,教授,硕士生导师,E-mail:wangdalei2000@126.com。

引用格式: 王紫益,王雷,徐浩然,等. 基于多策略改进蜣螂算法的三维无人机路径规划[J]. 南京航空航天大学学报(自然科学版),2025,57(3):475-486. WANG Ziyi, WANG Lei, XU Haoran, et al. 3D path planning of UAV based on multi-strategy improved dung beetle algorithm[J]. Journal of Nanjing University of Aeronautics & Astronautics(Natural Science Edition),2025, 57(3):475-486.

cost function generated by MSIDBO algorithm is smaller, the path quality is higher, and the stability is better under different scenarios.

Key words: dung beetle algorithm; spatial pyramid matching (SPM) chaotic mapping; reverse learning; seagull optimization algorithm; *t*-distributed difference variation

近年来,无人机在多个领域的应用日益广泛。随着无人机技术的愈发成熟,其执行任务的复杂程度也越来越高,无论是在影视拍摄、地理测绘、货物配送还是在应急救援、军事侦察等场景下,科学、合理且高效的路径规划是保障无人机顺利完成任务的关键所在。无人机路径规划是一个多约束条件的复杂问题,智能优化算法为解决三维无人机路径规划提供了有效的途径。

智能优化算法不仅展现了卓越的全局搜索能力,而且在局部探索方面也表现出色,能够在复杂的三维空间中综合考虑各种约束因素,寻找到较优的无人机飞行路径。目前,常见的应用于三维无人机路径规划的智能优化算法包括遗传算法[1]、粒子群算法[2]、蚁群算法[3]和A*算法[4]等,这些算法在求解三维无人机路径规划时有一定的优势,但是仍存在求解精度不足并容易陷入局部最优解的问题。在此基础上,Xu等[5]为解决遗传算法易陷入局部最优的问题,引入专家指导策略,可以帮助算法跳出局部最优值,实现本应收敛的次优解,但是算法无法平衡局部搜索和全局搜索之间的关系。He等[6]结合粒子群优化算法与共生生物搜索策略,探索能力和利用能力高效结合,同时采用3次B样条曲线来生成平滑的飞行路径,但是算法的收敛精度不够高。Tian等[7]在传统蚁群算法的基础上引入麻雀优化算法的迭代解,以提供更好的初始信息素分布,提高了算法的求解精度,但当问题规模增大时,仍容易陷入局部最优解。Li等[8]通过利用新颖的启发式评估函数和均匀应用的二次B样条曲线平滑,能够规划出安全、平滑的无人机运行路径,但是算法的搜索时间过长。Wu等[9]提出了一种全局-局部平衡的鲸鱼优化算法,并结合随机梯度辅助优化方法,在迭代时确定负梯度方向,选择合适的步长来提高算法的探索能力。Yu等[10]采用灰狼优化算法和差分进化算法相结合的方式来求解无人机路径规划问题,位置方程更新,实现了基于等级的突变策略,平衡了算法开发和探索,但是算法的求解精度提高较少。

蜣螂优化算法(Dung beetle optimizer, DBO)是由Xue等[11]在2023年提出,是一种新型智能优化算法,通过实验证明,在多个工程设计中,该算法大大提升了问题求解的效率与质量,为工程领域的复杂问题提供了一个全新的有力工具。然而,基本的蜣螂算法在路径规划问题上仍存在全局搜索与局部搜索不平衡、寻优性能不稳定和搜索效率低等问题。针对这些问题,本文提出了一种基于多策略改进的蜣螂优化算法(Multi-strategy improved dung beetle optimizer, MSIDBO),并将其应用于三维无人机路径规划。在基准测试函数、CEC2019函数测试集和三维无人机路径仿真结果中表明,相较于传统DBO,MSIDBO算法在收敛速度和寻优精度方面均表现出显著优势。

1 蜣螂优化算法

蜣螂优化算法根据蜣螂的不同行为划分为4个子种群:滚球蜣螂、繁殖蜣螂、小蜣螂和偷窃蜣螂[11],每个种群都对应不同的位置更新公式。

1.1 滚球蜣螂

蜣螂为了生存会有一个滚粪球的行为,蜣螂会将自己所获得的粪球中的一部分用来产卵,以此来培养自己的下一代,剩下的部分用作食物。在蜣螂滚粪球的过程之中,蜣螂会尽可能快速有效地移动粪球避免与其他蜣螂竞争,因而蜣螂需要利用天体的线索(太阳、月亮、偏振光)来尽可能地让滚动的粪球保持直线的行走路径。滚球蜣螂的滚球行为在搜索空间的位置更新规则为

$$\begin{cases} \boldsymbol{x}_i(t+1)=\boldsymbol{x}_i(t)+\alpha\times k\times \boldsymbol{x}_i(t+1)+b\times \Delta\boldsymbol{x} \\ |\Delta\boldsymbol{x}|=|\boldsymbol{x}_i(t)-\boldsymbol{X}^w| \end{cases} \tag{1}$$

式中:t为当前的迭代次数;α为一个取值为1或者-1的自然系数,表示其是否偏离原来方向;$k\in(0,0.2)$表示的是偏转系数的一个常数;b表示(0,1)的一个常数;$\Delta\boldsymbol{x}$用来模拟光照强度的变化,$\boldsymbol{X}^w$为全局最差的位置。

当蜣螂的粪球遇到障碍物无法继续移动前行的时候,蜣螂会通过跳舞行为来进行重新定位,以此来选择下一步的方向获得新的路线。滚球蜣螂的跳舞行为在搜索空间的位置更新方式为

$$\boldsymbol{x}_i(t+1)=\boldsymbol{x}_i(t)+\tan\theta|\boldsymbol{x}_i(t)-\boldsymbol{x}_i(t-1)| \tag{2}$$

式中:$|\boldsymbol{x}_i(t)-\boldsymbol{x}_i(t-1)|$表示第$i$只蜣螂在第$t$次迭代时的位置与在$t-1$次迭代时所处的位置之差,$\theta\in[0,\pi]$,当$\theta$为0、$\pi/2$或者$\pi$时,滚球蜣螂的位置不进行更新。

1.2 繁殖蜣螂

为了给后代提供安全的产卵环境,蜣螂通常会将粪球转移至隐蔽区域并进行掩埋。基于蜣螂的这一繁殖行为,采用边界选择策略来模拟蜣螂的产卵区域分布。

$$\begin{cases} L_b^* = \max(X^* \times (1-R), L_b) \\ U_b^* = \min(X^* \times (1+R), U_b) \end{cases} \tag{3}$$

式中：X^*表示当前的局部最优位置；L_b^*为产卵区域的下界；U_b^*为产卵区域的上界；$R=1-t/T_{max}$，T_{max}为最大迭代次数；L_b为优化问题的下界；U_b表示优化问题的上界。在确定产卵区域后，雌性蜣螂会在该区域内进行产卵，其产卵区域的边界范围会随着迭代次数的变化而动态调整，所以繁殖蜣螂的位置也是动态变化的。繁殖蜣螂的位置更新策略为

$$B_i(t+1) = X^* + b_1 \times (B_i(t) - L_b^*) + b_2 \times (B_i(t) - U_b^*) \tag{4}$$

式中：$B_i(t)$为第i个雏球在第t次迭代时的位置；b_1和b_2为大小为$1\times D$的两个独立的随机变量，D为优化问题的维数。

1.3　小蜣螂

成熟的蜣螂会从地下爬出并开始寻找食物。在觅食过程中，蜣螂会构建一个最优觅食区域，该区域的数学表达式为

$$\begin{cases} L_b^b = \max(X^b \times (1-R), L_b) \\ U_b^b = \min(X^b \times (1+R), U_b) \end{cases} \tag{5}$$

式中：X^b表示全局最优位置，L_b^b表示最优觅食区域的下界，U_b^b表示最优觅食区域的上界。因此，小蜣螂的位置更新策略可表示为

$$x_i(t+1) = x_i(t) + C_1 \times (x_i(t) - L_b^b) + C_2 \times (x_i(t) - U_b^b) \tag{6}$$

式中：C_1为服从正态分布的随机变量，C_2为取值范围在(0,1)之间的随机向量。

1.4　偷窃蜣螂

有些蜣螂会从其他的蜣螂身上偷取粪球，这些蜣螂被称为偷窃蜣螂，X^b表示全局最优位置，偷窃蜣螂在全局最优位置附近进行食物的争夺。偷窃蜣螂的位置更新策略为

$$x_i(t+1) = X^b + S \times g \times (|x_i(t) - X^*| + |x_i(t) - X^b|) \tag{7}$$

式中：S为一个常数，g为一个大小为$1\times D$的服从正态分布的随机向量。

2　改进蜣螂优化算法

2.1　空间金字塔匹配混沌反向学习初始化

基本蜣螂算法在求解复杂优化问题时，通常采用随机种群初始化的方式。虽然这种方式简单直接，但存在明显的局限性。在搜索空间里，随机初始化难以确保初始种群能够全面覆盖到所有潜在的优势区域。此外，由于种群多样性较低，个体间差异不足，导致算法易陷入局部最优。为改善这一缺陷，本文提出了一种融合空间金字塔匹配(Spatial pyramid matching，SPM)混沌映射与反向学习的策略。首先，利用SPM混沌映射对种群进行初始化，其数学表达式为

$$x(t+1) = \begin{cases} \mathrm{mod}\left(\dfrac{x(t)}{\eta} + \mu\sin(\pi x(t)) + \varepsilon, 1\right) & 0 \leqslant |x(t)| < \eta \\ \mathrm{mod}\left(\dfrac{x(t)/\eta}{0.5-\eta} + \mu\sin(\pi x(t)) + \varepsilon, 1\right) & \eta \leqslant |x(t)| < 0.5 \\ \mathrm{mod}\left(\dfrac{(1-x(t))/\eta}{0.5-\eta} + \mu\sin(\pi(1-x(t))) + \varepsilon, 1\right) & 0.5 \leqslant |x(t)| < 1-\eta \\ \mathrm{mod}\left(\dfrac{(1-x(t))}{\eta} + \mu\sin(\pi(1-x(t))) + \varepsilon, 1\right) & 1-\eta \leqslant |x(t)| < 1 \end{cases} \tag{8}$$

式中：当$\eta\in(0,1)$和$\mu\in(0,1)$时，系统处于混沌的状态，ε为(0,1)之间的随机数。将SPM混沌映射与随机初始化方式各生成200个种群，种群的取值范围为[0,1]，实验结果如图1所示，SPM混沌映射在搜索空间中的频数分布更均匀，种群多样性更丰富。

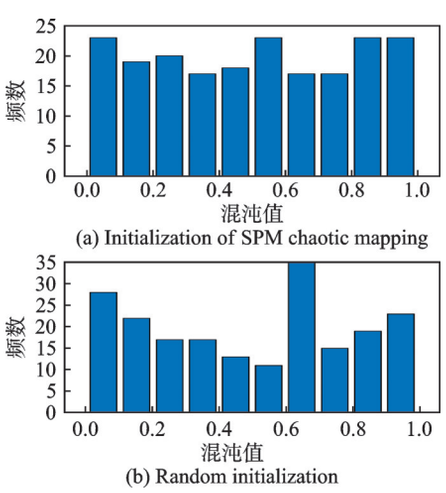


图1　频数分布直方图

Fig.1　Frequency distribution histogram

然后，将生成的SPM混沌映射到解空间，再对P_i进行反向学习得到反向解O_{P_i}

$$P_i = L_b + (U_b - L_b) \times X_i \tag{9}$$

$$O_{P_i} = K(L_b + U_b) - P_i \tag{10}$$

式中：L_b和U_b为优化问题的下界和上界，$K\in(0,1)$。最终，将SPM混沌映射生成的种群与反向学习生成的种群合并，并按照适应度排序，选择前N个最优个体作为最终的初始化种群。

SPM混沌反向学习初始化策略，采用SPM混沌映射策略覆盖到搜索空间内的潜在优势区域，再

发挥反向学习的优势来生成更多的优质候选解,这种策略能够很好地提高初始种群的多样性,以及搜索空间的质量和精度。

2.2 改进边界收敛因子

基本蜣螂优化算法中,繁殖蜣螂和小蜣螂的活动范围是由动态的边界收敛因子 $R=1-t/T_{\max}$ 所确定的。其中 t 代表当前迭代次数,$T_{\max}$ 表示最大迭代次数。由此可见,边界收敛因子随着迭代次数的增加逐渐减小,从而使蜣螂的产卵区域和觅食区域逐步收缩。然而,线性下降的边界收敛因子存在一定的局限性:在早期阶段,较小边界收敛因子使得全局搜索能力不足;而在后期,较大的边界收敛因子又降低了局部搜索精度。因此,本文对基本的边界收敛因子进行改进,新的边界收敛因子为

$$R=\frac{\left(\left(\frac{2t}{T_{\max}}-1\right)^3-3\times\left(\frac{2t}{T_{\max}}-1\right)+2\right)}{4} \tag{11}$$

由图2可知,改进后的边界收敛因子在算法的早期阶段下降较为缓慢,有助于拓展搜索范围,从而增强全局搜索能力。而在后期,边界收敛因子的下降速度加快,提高了算法的局部搜索精度,并加速了收敛过程,同时有效平衡了全局搜索与局部搜索的能力。

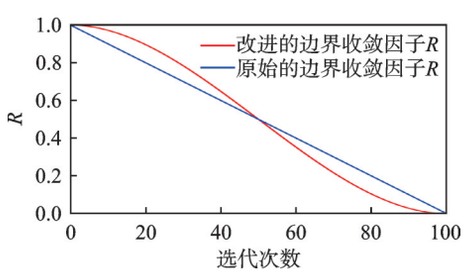


图2 边界收敛因子 R 对比图

Fig.2 Boundary convergence factor R comparison diagram

2.3 融合海鸥优化算法

在基本的蜣螂优化算法中,繁殖蜣螂逐渐向当前局部最优位置 $\boldsymbol{X}^*$ 靠拢。这种方式在一定程度上提高了搜索的精度,但是会降低种群多样性,同时在迭代过程中也会丧失种群的多样性,局部的搜索能力降低。因此,本文引入了海鸥优化算法的攻击机制[12],海鸥在迁徙过程中会持续调整其攻击角度与速度,以优化搜索策略,产生螺旋式的攻击机制,如图3所示,在迭代过程中,个体以螺旋轨迹逐步向新的迭代位置逼近,海鸥优化算法的攻击机制可描述为

$$r=u\times \mathrm{e}^{\beta v} \tag{12}$$

$$x'=r\times\cos\beta \tag{13}$$

$$y'=r\times\sin\beta \tag{14}$$

$$z'=r\times\beta \tag{15}$$

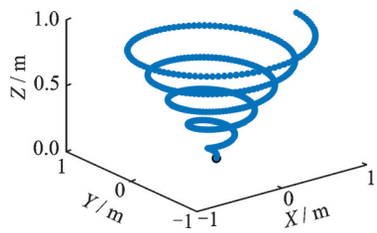


图3 螺旋攻击

Fig.3 Spiral attack

$$\boldsymbol{X}_i(t)=(\boldsymbol{D}_i(t)\times x'\times y'\times z')+\boldsymbol{X}_{\text{best}}(t) \tag{16}$$

式中:r 为螺旋攻击的半径,β 为区间[0,2π]内的随机攻击角度,u 和 v 为决定螺旋形状的常数,$\boldsymbol{D}_i(t)$ 为海鸥个体与最优海鸥个体之间的位置距离。融合海鸥优化算法攻击机制的繁殖蜣螂的位置更新公式为

$$\boldsymbol{B}_i(t+1)=\boldsymbol{X}^*+x'\times y'\times z'\times(\boldsymbol{b}_1\times(\boldsymbol{B}_i(t)-\boldsymbol{L}_{\text{b}}^*)+\boldsymbol{b}_2\times(\boldsymbol{B}_i(t)-\boldsymbol{U}_{\text{b}}^*)) \tag{17}$$

将海鸥优化算法中的螺旋攻击机制引入繁殖蜣螂的位置更新策略,不仅有效增强了种群的多样性,还加速了算法的收敛过程,同时提升了全局搜索能力与局部开发精度。

2.4 *t*-distribution差分变异

在求解多峰优化问题时,算法往往由于过早收敛而易陷入局部最优解,影响其全局优化能力。为增强算法的全局搜索能力,本文提出一种基于 t-分布(t-distribution)的差分变异策略,以提高跳出局部最优解的概率。该策略借鉴差分进化算法(Differential evolution, DE)[13]的思想,通过当前个体、全局最优个体及种群中的随机个体进行差分变异。但是传统的差分变异的收缩因子是区间[0,1]内的随机数,因此在变异过程中无法产生较大的扰动。t-distribution的概率密度函数在两端具有较长的尾部,使其能够施加更强的扰动,从而提升算法跳出局部最优的能力。位置更新公式为

$$\boldsymbol{X}_{\text{new}}(t)=w\times(\gamma_1\times(\boldsymbol{X}^{\text{b}}-\boldsymbol{X}_i(t))+\gamma_2\times(\boldsymbol{X}'(t)-\boldsymbol{X}_i(t))) \tag{18}$$

式中:w 为缩放因子,γ_1 和 γ_2 为服从自由度是当前迭代次数的 t-distribution 随机数,$\boldsymbol{X}_i(t)$ 为变异前蜣螂个体,$\boldsymbol{X}'(t)$ 为随机选择的蜣螂个体。

在应用 t-distribution 差分变异策略后,引入了贪婪选择机制,根据变异前后个体的适应度值来判断是否更新个体位置,从而提升优化效果。

$$\boldsymbol{X}_i=\begin{cases}\boldsymbol{X}_{\text{new}}(t) & f(\boldsymbol{X}_{\text{new}}(t))\leqslant f(\boldsymbol{X}_i(t))\\ \boldsymbol{X}_i(t) & f(\boldsymbol{X}_{\text{new}}(t))>f(\boldsymbol{X}_i(t))\end{cases} \tag{19}$$

式中:$\boldsymbol{X}_i(t)$ 为变异前的蜣螂个体,$\boldsymbol{X}_{\text{new}}(t)$ 为变异后的蜣螂个体,$f(x)$ 的值为对应的蜣螂个体的适应度。

2.5　MSIDBO算法流程

根据上文的改进策略，改进后的MSIDBO的算法流程图如图4所示。

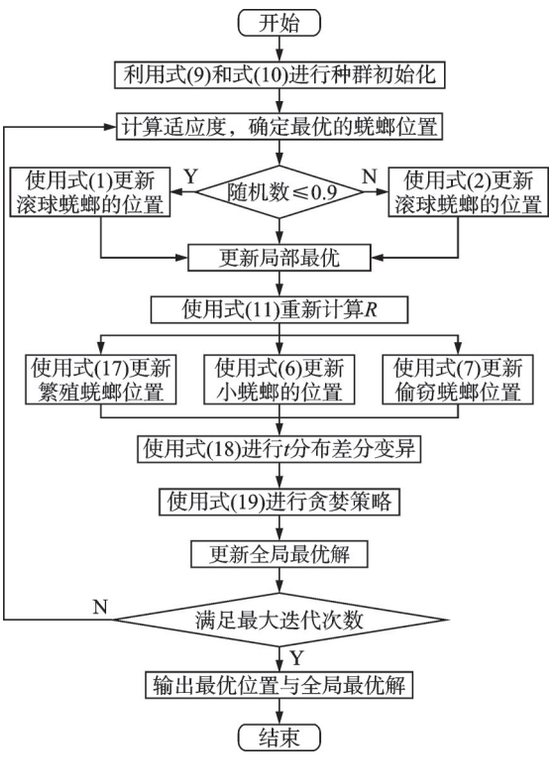


图4　MSIDBO算法流程图

Fig.4　Flowchart of MSIDBO algorithm

3　函数性能测试实验

3.1　标准测试函数和实验环境

为了评估本文改进算法的性能，选取了15个经典标准测试函数进行实验仿真。其中，单模态测试函数(F_1～F_6)仅包含一个全局最优解，主要用于评估算法的局部搜索能力；多模态测试函数(F_7～F_{11})包含一个全局最优解和多个局部最优解，以衡量算法的全局探索能力；复合测试函数(F_{12}～F_{15})则具有固定的维度，用于考察算法在全局探索与局部搜索之间的平衡性。各测试函数的数学表达式详见表1。

3.2　参数分析

在本文改进的MSIDBO算法中3个控制参数(u,v,w)对算法的搜索范围和精度具有重要影响。为探讨这些参数对算法性能的影响，本节固定其他参数为文献推荐值，并设置$u=0.5$、1，$v=0.5$、1，$w=0.5$、0.8、1进行实验。测试函数选取F_1～F_{11}，实验设定种群规模为30，最大迭代次数为500，并进行30次独立运行。以平均适应度值作为性能评估标准，实验结果详见表2。实验结果表明，当$u=0.5$、$v=1$、$w=0.5$时，算法在测试的11个函数中的10个函数上取得最优。因此，后续实验采用该参数组合。

表1　基准测试函数

Table 1　Benchmarking functions

函数表达式	维度	范围	最优值
$F_1(x)=\sum_{i=1}^{n}x_i^2$	30	[−100,100]	0
$F_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	30	[−10,10]	0
$F_3(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_i\right)^2$	30	[−100,100]	0
$F_4(x)=\max_i\{\lvert x_i\rvert,1\leqslant i\leqslant n\}$	30	[−100,100]	0
$F_5(x)=\sum_{i=1}^{n}(\lvert x_i+0.5\rvert)^2$	30	[−100,100]	0
$F_6(x)=\sum_{i=1}^{n}ix_i^4+\mathrm{random}[0,1)$	30	[−1.28,1.28]	0
$F_7(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	30	[−5.12,5.12]	0
$F_8(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+\mathrm{e}$	30	[−32,32]	0
$F_9(x)=\frac{1}{4\,000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	30	[−600,600]	0
$F_{10}(x)=\frac{\pi}{n}\left\{10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right]+(y_n-1)^2\right\}+\sum_{i=1}^{n}u(x_i,10,100,4)$ $y_i=1+\frac{(x_i+1)}{4}$；$u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\0 & -a<x_i<a\\k(-x_i-a)^m & x_i<-a\end{cases}$	30	[−50,50]	0

续表

函数表达式	维度	范围	最优值
$F_{11}(x)=0.1\left\{\sin^2(3\pi x_1)+\sum_{i=1}^{n}(x_i-1)^2[1+\sin^2(3\pi x_i+1)]+(x_n-1)^2[1+\sin^2(2\pi x_n)]\right\}+\sum_{i=1}^{n}u(x_i,5,100,4)$	30	$[-50,50]$	0
$F_{12}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2$	4	$[-5,5]$	0.000 3
$F_{13}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	$[-5,5]$	0.398
$F_{14}(x)=\left[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)\right]\times\left[30+(2x_1-3x_2)^2\times(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)\right]$	2	$[-2,2]$	3
$F_{15}(x)=\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{3}a_{ij}(x_j-p_{ij})^2\right)$	3	$[0,1]$	−3.86

表2 不同参数对MSIDBO算法性能的影响

Table 2 Influence of different parameters on the performance of MSIDBO algorithm

u	0.5	0.5	0.5	0.5	0.5	0.5	1	1	1	1	1	1
v	0.5	0.5	0.5	1	1	1	0.5	0.5	0.5	1	1	1
w	0.5	0.8	1	0.5	0.8	1	0.5	0.8	1	0.5	0.8	1
F_1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_5	2.40E−02	1.47E−02	3.20E−02	1.19E−02	2.08E−01	3.43E−01	4.10E−01	7.51E−01	1.51E+00	2.87E−01	6.33E−01	7.34E−01
F_6	2.98E−04	7.67E−04	8.34E−04	4.25E−04	3.53E−04	5.97E−04	6.05E−04	9.53E−04	1.28E−03	6.52E−04	8.11E−04	1.17E−03
F_7	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_8	4.44E−16	4.44E−16	4.44E−16	4.44E−16	4.44E−16	4.44E−16	4.44E−16	4.44E−16	4.44E−16	4.44E−16	4.44E−16	4.44E−16
F_9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_{10}	8.78E−05	4.18E−04	5.36E−04	1.66E−05	1.22E−02	1.10E−02	1.87E−02	3.20E−02	4.52E−02	2.12E−02	3.35E−02	4.89E−02
F_{11}	1.88E−03	7.03E−03	2.34E−02	8.24E−04	2.91E−02	6.38E−02	8.88E−02	2.04E−02	3.08E−01	3.77E−02	1.93E−01	3.32E−01

3.3 算法性能测试实验

为验证本文提出的MSIDBO算法的性能，将MSIDBO算法与基本的DBO、金枪鱼群算法(Tuna swarm optimization，TSO)[14]、徒步优化算法(Hiking optimization algorithm，HOA)[15]、灰狼优化算法(Grey wolf optimization，GWO)[16]、鲸鱼优化算法(Whale optimization algorithm，WOA)[17]、海鸥优化算法(Seagull optimization algorithm，SOA)进行对比实验。对比实验中的各算法参数均按照相关文献中的设置进行调整。实验中，种群规模设为30，最大迭代次数为200，每个算法独立运行30次，并以最优值、平均值和标准差作为评价指标。实验结果如表3所示，其中各项评价指标的最优值以粗体显示。在单模态测试函数F_1～F_6的测试中，MSIDBO算法在F_1、F_2、F_3、F_4中找到了理论最优值、理论最优标准差以及理论最优的平均最优值。

表 3　基准函数测试结果

Table 3　Test results of benchmark functions

函数	评价标准	TSO	HOA	GWO	WOA	SOA	DBO	MSIDBO
	最优值	9.20E−112	1.16E−19	7.46E−10	1.05E−34	6.44E−31	3.88E−62	0.00E+00
F_1	平均值	3.56E−92	2.48E−16	9.00E−09	6.78E−28	2.48E−12	1.03E−39	0.00E+00
	标准差	1.97E−86	1.61E−16	9.27E−09	1.12E−26	2.30E−16	6.36E−33	0.00E+00
	最优值	4.81E−57	2.89E−09	2.56E−06	2.61E−23	4.34E−36	6.28E−34	0.00E+00
F_2	平均值	3.27E−48	1.61E−08	5.04E−06	4.37E−20	1.09E−25	1.12E−20	0.00E+00
	标准差	3.42E−46	1..56E−08	2.92E−06	2.03E−19	2.52E−25	2.39E−23	0.00E+00
	最优值	1.20E−112	1.42E−13	3.02E−01	5.00E+04	4.87E−03	1.68E−64	0.00E+00
F_3	平均值	3.27E−85	4.32E−11	5.94E+00	7.83E+04	1.07E+05	1.42E−05	0.00E+00
	标准差	1.61E−81	1.71E−10	7.35E+00	2.37E+04	2.91E+04	1.39E−01	0.00E+00
	最优值	1.94E−57	4.10E−09	1.09E−02	3.67E+00	4.51E+01	1.07E−36	0.00E+00
F_4	平均值	5.98E−46	2.23E−08	3.10E−02	6.47E+01	7.10E+01	6.81E−18	0.00E+00
	标准差	6.56E−46	1.45E−08	1.79E−02	2.31E+01	9.45E+00	2.26E−20	0.00E+00
	最优值	1.90E−04	3.86E+00	3.82E−04	6.06E−01	2.85E+00	4.39E−02	1.13E−04
F_5	平均值	4.23E−02	5.28E+00	1.19E+00	1.35E+00	3.92E+00	2.36E−01	1.10E−03
	标准差	6.67E−02	5.62E−01	4.65E−01	4.82E−01	7.67E−01	2.20E−01	1.13E−02
	最优值	2.63E−05	1.20E−04	1.46E−03	3.50E−04	3.84E−06	2.53E−04	3.42E−06
F_6	平均值	7.95E−04	1.54E−03	7.92E−03	5.02E−03	1.10E−03	1.36E+00	4.46E−04
	标准差	7.70E−04	1.06E−03	3.45E−03	1.18E−02	1.27E−01	2.84E−03	2.56E−04
	最优值	0.00E+00	0.00E+00	2.72E−07	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_7	平均值	0.00E+00	1.96E+01	1.40E+01	2.08E−14	1.41E−01	1.36E+00	0.00E+00
	标准差	0.00E+00	3.61E+01	5.60E+00	2.28E−14	1.48E+00	5.38E+00	0.00E+00
	最优值	4.44E−16	1.27E−09	6.32E−06	3.99E−15	4.44E−16	4.44E−16	4.44E−16
F_8	平均值	4.44E−16	5.79E−09	2.01E−05	2.35E−14	1.41E−09	1.03E−15	4.44E−16
	标准差	0.00E+00	6.03E−09	6.97E−06	2.52E−14	9.14E−10	9.01E−16	0.00E+00
	最优值	0.00E+00	0.00E+00	2.64E−09	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F_9	平均值	0.00E+00	0.00E+00	1.16E−02	1.24E−02	2.01E−02	0.00E+00	0.00E+00
	标准差	0.00E+00	0.00E+00	1.60E−02	9.85E−02	1.34E−01	0.00E+00	0.00E+00
	最优值	2.71E−06	3.05E−01	1.34E−02	1.37E−02	3.38E−01	5.96E−04	4.31E−05
F_{10}	平均值	2.59E−03	6.98E−01	1.27E−01	1.42E−01	1.28E+03	3.63E−02	2.18E−04
	标准差	6.16E−04	6.16E−04	1.16E−01	7.89E−02	9.48E+02	7.84E−03	1.01E−04
	最优值	2.62E−05	5.92E−02	3.43E−01	3.54E−01	2.56E+00	3.14E−01	5.90E−06
F_{11}	平均值	2.29E−02	1.09E−01	1.05E+00	1.05E+00	1.23E+02	1.85E+00	4.41E−03
	标准差	2.01E−02	3.38E−02	3.28E−01	3.78E−01	1.88E+03	5.82E−01	1.63E−02
	最优值	3.07E−04	3.07E−04	3.15E−04	3.23E−04	3.41E−04	3.07E−04	3.07E−04
F_{12}	平均值	5.29E−04	4.12E−04	3.87E−03	1.25E−03	9.44E−04	8.59E−04	3.56E−04
	标准差	3.07E−04	5.09E−04	8.00E−03	3.45E−03	2.19E−03	4.81E−04	1.99E−04
	最优值	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01
F_{13}	平均值	3.98E−01	3.98E−01	3.98E−01	3.98E−01	4.18E−01	3.98E−01	3.98E−01
	标准差	0.00E+00	2.89E−04	1.05E−05	7.09E−04	2.35E−01	0.00E+00	0.00E+00
	最优值	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00
F_{14}	平均值	3.00E+00	7.42E+00	5.70E+00	3.90E+00	1.33E+01	3.00E+00	3.00E+00
	标准差	7.33E−15	2.12E+01	4.49E−04	1.31E−03	1.32E+01	2.72E−04	3.67E−15
	最优值	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00
F_{15}	平均值	−3.86E+00	−3.85E+00	−3.86E+00	−3.84E+00	−3.75E+00	−3.86E+00	−3.86E+00
	标准差	2.18E−15	4.10E−03	2.50E−03	5.20E−02	2.00E−01	3.37E−03	3.20E−03

虽然在F_5、F_6上未能找到理论最优解,但其整体表现仍优于其他对比算法。因此,在单模态的测试函数中,MSIDBO算法的搜索寻优能力十分的优异,局部搜索能力强、收敛速度快。

在多模态测试函数F_7~F_{11}的测试中,MSIDBO算法在F_7、F_9中找到了理论最优值、理论最优标准差以及理论最优平均最优值。在F_8、F_{10}、F_{11}函数中MSIDBO算法的优化精度亦优于其他对比算法。所以在多模态的测试函数中,MSIDBO算法展现出良好的性能,具备强大的全局搜索能力且不易陷入局部最优。在复合测试函数F_{12}~F_{15}的测试中,MSIDBO算法都能找到理论最优值,有效平衡了固定维度条件下的全局搜索与局部开发。

图5展示了MSIDBO算法与其他对比算法的

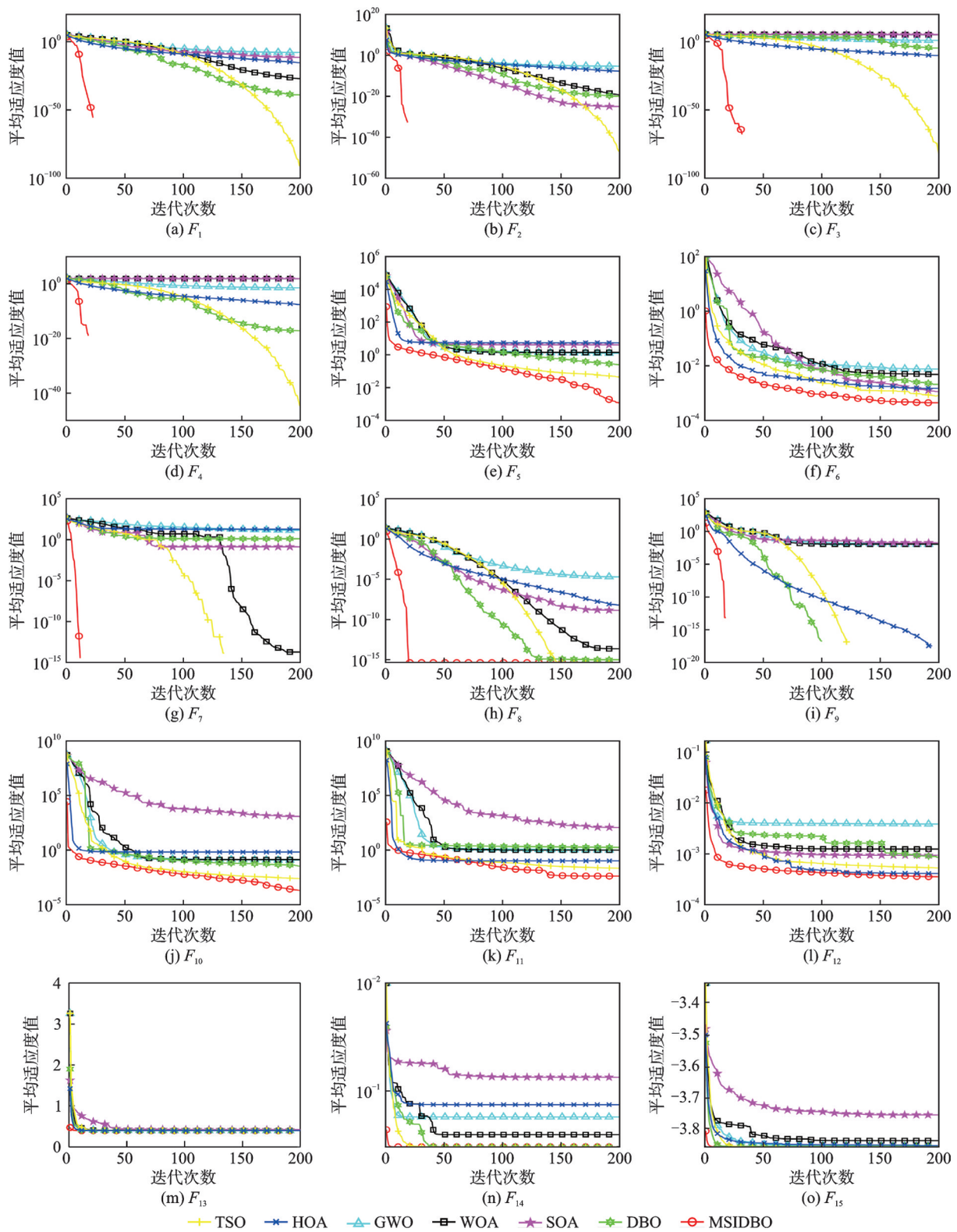


图5 基准函数测试的收敛曲线

Fig.5 Convergence curves of benchmark function test

平均适应度收敛曲线。可以看出本文所提出的MSIDBO算法展现出了多方面的显著优势。该算法运用SPM混沌反向学习策略，利用混沌系统的遍历性和随机性，生成一个较优的初始化种群。通过改进的边界收敛因子和融合海鸥优化算法的攻击策略使得算法能够平衡全局探索和局部搜索之间的关系，提高开发的精度。最后采用*t*-distribution差分变异策略和贪婪策略，帮助算法能够跳出局部最优解。综合对比实验结果，MSIDBO算法在基准函数测试中展现出卓越的全局优化性能和收敛速度，同时具备有效规避局部最优陷阱的能力。

3.4　与其他改进算法对比

为进一步验证MSIDBO算法解决复杂问题的有效性，将MSIDBO算法和文献[18]中提出的多策略改进的蜣螂优化算法（Multi-strategy improved dung beetle optimizer，MIDBO）、文献[19]中提出的融合多策略改进的鲸鱼优化算法（Multi-strategy improved whale optimization algorithm，MSWOA）在CEC2019函数测试集上进行对比实验。该测试集包含10个单目标测试函数，这些函数具有复杂的空间特征，且其理论最优值均为1，函数的具体信息如表4所示。

表4　CEC2019测试函数

Table 4　CEC2019 benchmark functions

编号	函数名称	维度	范围
CEC01	Storn's Chebyshev polynomial fitting problem	9	[−8 192，8 192]
CEC02	Inverse hilbert matrix problem	16	[−16 384，16 384]
CEC03	Lennard-Jones minimum energy cluster	18	[−4，4]
CEC04	Rastrigin's function	10	[−100，100]
CEC05	Griewangk's function	10	[−100，100]
CEC06	Weierstrass function	10	[−100，100]
CEC07	Modified Schwefel's function	10	[−100，100]
CEC08	Expanded Schaffer's F_6 function	10	[−100，100]
CEC09	Happy cat function	10	[−100，100]
CEC10	Ackley function	10	[−100，100]

为确保实验的公平性，各算法均采用统一参数设置。种群规模设为30，最大迭代次数为500，维度为30，并分别独立运行30次。实验结果详见表5，各项评价标准最优的用粗体表示。在CEC2019函数测试集实验中，MSIDBO算法对比另外两种改进算法在CEC01、CEC03、CEC04、CEC08、CEC09上的所有评价指标都是最优的，在CEC02、CEC06、CEC10上的平均值是最优的。这一结果充分表明MSIDBO算法在多数情况下能够更可靠地收敛到高质量的解，展现出其在解决复杂函数优化问题上的优势。

表5　与其他改进算法对比

Table 5　Comparisons with the improved algorithm

函数编号	评价指标	MIDBO	MSWOA	MSIDBO
CEC01	平均值	1.00E+00	1.21E+07	**1.00E+00**
	标准差	0.00E+00	1.72E+07	**0.00E+00**
CEC02	平均值	4.93E+00	5.14E+03	**4.55E+00**
	标准差	**2.15E−01**	2.18E+03	3.69E−01
CEC03	平均值	5.40E+00	5.02E+00	**2.35E+00**
	标准差	1.32E+00	2.32E+00	**1.09E+00**
CEC04	平均值	5.00E+01	2.53E+01	**2.41E+01**
	标准差	1.51E+01	1.17E+01	**1.07E+01**
CEC05	平均值	**1.05E+00**	1.35E+00	1.20E+00
	标准差	**4.23E−02**	1.95E−01	1.80E−01
CEC06	平均值	6.40E+00	5.25E+00	**4.66E+00**
	标准差	**1.08E+00**	1.61E+00	1.29E+00
CEC07	平均值	1.52E+03	**7.78E+02**	9.79E+02
	标准差	2.56E+02	**1.25E+02**	3.46E+02
CEC08	平均值	4.43E+00	3.91E+00	**3.80E+00**
	标准差	1.82E−01	1.06E−01	**1.01E−01**
CEC09	平均值	1.36E+00	1.52E+00	**1.33E+00**
	标准差	1.29E−01	2.15E−01	**1.10E−01**
CEC10	平均值	2.00E+01	2.10E+01	**1.91E+01**
	标准差	2.23E+00	**2.19E−02**	5.62E+00

4　基于MSIDBO无人机路径规划

4.1　地形环境模型

在运用智能优化算法进行无人机三维路径规划时，由于无人机在现实场景中往往需要穿梭于复杂多变的地形之间，面临着各种各样的障碍物，因此模拟环境必须贴近于真实情况，才能确保规划出的路径具备实际应用价值。本文采用指数函数生成山峰作为无人机运行的障碍物区域，通过调整指数函数的参数，可以灵活控制山峰的高度、坡度及分布密度，从而生成多样化的地形场景。这种建模方法不仅能有效反映真实环境中的地形特征，还能为无人机路径规划提供更具挑战性的测试环境。

$$Z(x,y)=\sum_{i=1}^{n}h_i\exp\left[-\left(\frac{x-x_i}{x_{si}}\right)^2-\left(\frac{y-y_i}{y_{si}}\right)^2\right] \tag{20}$$

式中:n为山峰的数量,Z为山峰的地形高度,(x_i,y_i)表示第i个山峰的中心点坐标,h_i为第i个山峰的高度,x_{si}和y_{si}为山峰的坡度。环境模型如图6所示。

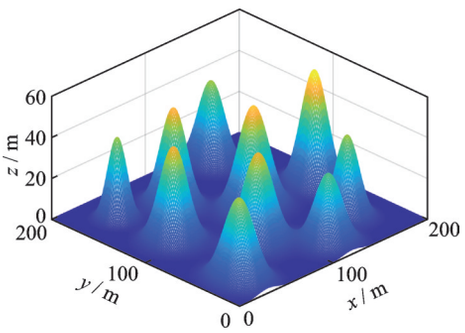


图6 地形环境模型

Fig.6 Terrain environment model

4.2 代价函数

4.2.1 路径长度代价

在三维无人机路径规划的问题中,路径的长度是需要考虑的重要因素。对于无人机而言,较短的路径长度具备多重显著优势,不仅可以极大地节省无人机的飞行时间,使其能够更高效地完成各项任务,还能显著降低能耗,减少电池的消耗,延长无人机的续航时间。路径长度是由每两个相邻节点之间的欧几里德距离相加而成的,路径长度代价定义如下

$$f_1=\sum_{i=1}^{n-1}\sqrt{(x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2+(z_{i+1}-z_i)^2} \tag{21}$$

式中:n为路径节点的个数,(x_i,y_i,z_i)为第i个路径节点的坐标。

4.2.2 飞行高度稳定性代价

在无人机执行任务时,飞行高度的稳定性至关重要。稳定的飞行高度可以使无人机保持良好的飞行姿势,同时减少无人机的能耗。无人机飞行高度的稳定性代价为

$$f_2=\sqrt{\sum_{i=1}^{n}\frac{(h_i-h_{\text{mean}})^2}{n}} \tag{22}$$

式中:h_i为路径节点对应的高度,h_{mean}为所有路径节点的高度平均值。

4.2.3 转向角代价

在无人机飞行转向过程中,合适的转向角度能确保其平稳过渡,使机身保持稳定的姿态,不会引发机身的剧烈晃动,避免因姿态失控而导致飞行事故,确保飞行过程的安全性与稳定性。转向角代价定义如下

$$f_3=\sum_{i=1}^{n-2}\arccos\left(\frac{\boldsymbol{l}_i^{\mathrm{T}}\boldsymbol{l}_{i+1}}{|\boldsymbol{l}_i||\boldsymbol{l}_{i+1}|}\right) \tag{23}$$

式中$\boldsymbol{l}_i=[x_{i+1}-x_i,y_{i+1}-y_i,z_{i+1}-z_i]$。

4.2.4 无人机路径代价函数

将上述的路径长度代价、飞行的高度稳定性代价、转向的角度代价进行综合加权,得到的就是无人机的路径代价函数,用式(24)表示为

$$F_{\text{cost}}=\rho_1 f_1+\rho_2 f_2+\rho_3 f_3 \tag{24}$$

式中:ρ_i为上述函数权重,$\rho_i\geqslant 0,\sum_{i=1}^{3}\rho_i=1$。

4.3 无人机三维路径规划仿真

根据地形环境模型,进行无人机环境模拟,相关参数设置如表6所示。分别采用DBO、SOA、WOA以及MSIDBO算法解决三维无人机路径规划问题,为了降低随机性对实验结果的影响,每个算法均进行了20次独立试验。

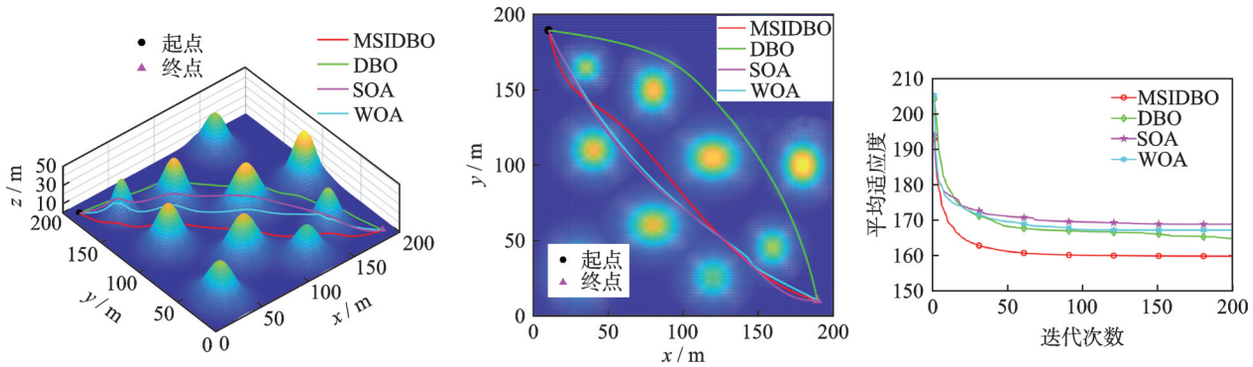
表6 环境参数设置

Table 6 Environment parameter setting

参数	参数值
执行空间	200×200×100
起点坐标	(10,190,1)
终点坐标	(190,10,1)
山峰数量	10、15
种群大小	100
迭代次数	200

图7显示了4种算法在山峰数量为10时的路径规划结果以及其适应度收敛曲线,从图7(a,b)中可见,MSIDBO算法所规划的路径长度最短,高度稳定性最优。相比之下,SOA和WOA算法的高度稳定性较差,可能增加了飞行的不确定性和风险。DBO算法的路径长度较长,这无疑会提高飞行成本,降低任务执行效率。从图7(c)中可以看出在10处山峰的环境下,各算法均能较快收敛,但除MSIDBO外,其他算法普遍存在较低的收敛精度,并更容易陷入局部最优解。相较之下,MSIDBO算法具备更强的寻优能力,求解精度更高,这表明t-distribution差分变异策略有效帮助了算法跳出局部最优。

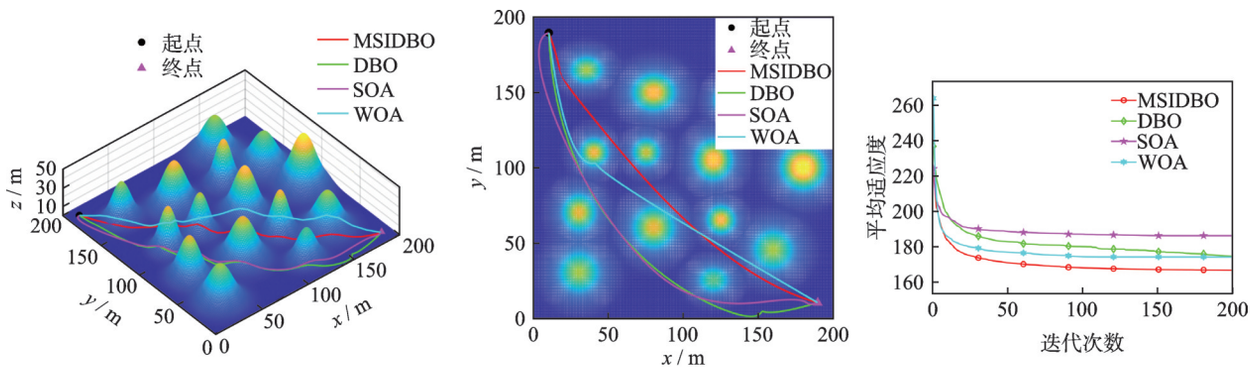
图8是各个算法在15处山峰的环境下运行的对比图。从图8(a,b)中可见WOA算法的路径高度稳定性代价较大,而SOA和DBO算法的路径长度较长,MSIDBO算法不仅路径长度最短,同时高度稳定性与路径平滑性均表现最佳。从图8(c)可以看出,在15处山峰的环境下,MSIDBO算法在前期的收敛速度不快,这是因为改进的边界收敛因子使得算法在迭代前期拓展了全局搜索能力,而在迭代后期算法寻优精度更高,所以MSIDBO算法的寻优精度比其他算法要好。



(a) UAV flight path　(b) Top view of UAV flight path　(c) Four algorithm converge curves

图 7　4 种算法在 10 处山峰中的路径规划对比图

Fig.7　Comparison diagrams of path planning of four algorithms in ten mountain peaks



(a) UAV flight path　(b) Top view of UAV flight path　(c) Four algorithm converge curves

图 8　4 种算法在 15 处山峰中的路径规划对比图

Fig.8　Comparison diagrams of path planning of four algorithms in fifteen mountain peaks

表 7 对比了各算法在不同障碍物数量环境下的路径规划结果，以每种算法的最优适应度、最差适应度、平均适应度值和标准差为评价标准。各项评价标准最优的用粗体表示。

表 7　4 种算法在不同环境下的路径规划结果对比

Table 7　Comparison of path planning results of four algorithms in different environments

山峰数量	算法	最优值	最差值	平均值	标准差
10	MSIDBO	**159.291**	**164.707**	**159.791**	**1.416**
	DBO	159.307	171.504	164.784	3.910
	SOA	164.172	176.881	168.823	3.703
	WOA	163.540	173.472	167.181	2.502
15	MSIDBO	**164.575**	**175.072**	**166.875**	**2.999**
	DBO	164.990	228.830	174.759	14.000
	SOA	170.896	228.878	186.286	14.502
	WOA	168.364	184.835	174.331	4.375

相较于其他 3 种算法，MSIDBO 算法的最优适应度、最差适应度、平均适应度及标准差均为最小值。随着地图障碍物的增多，MSIDBO 算法在无人机路径规划问题中所展现出的优势也就越大。这表明，在融合海鸥优化算法的攻击机制后，MSIDBO 算法在无人机路径规划中的局部搜索能力得到了显著增强。此外，在迭代初期，MSIDBO 算法的搜索种群适应度值较低，表明引入 SPM 混沌映射和反向学习策略不仅丰富了种群多样性，还优化了初始种群的质量，从而有效加速了算法的收敛过程。

综上所述，在三维无人机路径规划问题中，MSIDBO 算法表现出较快的收敛速度、更高的求解精度和更优的稳定性，能够在不同障碍物环境下规划出稳定且平滑的无人机飞行路径。

5　结　　论

本文针对三维无人机路径规划问题，提出了 MSIDBO。该算法在传统蜣螂优化算法的基础上，引入 SPM 混沌映射和反向学习初始化策略，以提升种群多样性和初始化质量；结合非线性边界收敛因子策略，有效平衡了全局搜索与局部搜索的能力。融合海鸥算法的攻击机制，以增强局部搜索能力；并结合 t-distribution 差分变异策略，进一步提高寻优精度。在基准函数测试及 CEC2019 函数集对比实验中，MSIDBO 算法相较于其他优化算法和改进算法表现优异。实验结果显示，该算法不仅在稳定性方面表现优异，而且具备迅速的收敛速度和出色的寻优精度。最后将 MSIDBO 算法应用在三维无人机路径规划问题中，综合考虑路径长度成

本、高度成本和转向角成本。实验仿真结果表明，MSIDBO算法在不同环境的无人机路径规划的过程中，在求解稳定性和寻优性能方面均具备优势，能够生成更加稳定且代价更低的无人机运行路径。

参考文献：

[1] DEBNATH D, VANEGAS F, BOITEAU S, et al. An integrated geometric obstacle avoidance and genetic algorithm tsp model for uav path planning[J]. Drones, 2024, 8(7): 302-330.

[2] PHUNG M D, HA Q P. Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization[J]. Applied Soft Computing, 2021, 107: 107376.

[3] JIA Y, ZHOU S, ZENG Q, et al. The UAV path coverage algorithm based on the greedy strategy and ant colony optimization[J]. Electronics, 2022, 11(17): 2667.

[4] LI J, LIAO C, ZHANG W, et al. UAV path planning model based on R5DOS model improved A-star algorithm[J]. Applied Sciences, 2022, 12(22): 11338.

[5] XU H, NIU Z, JIANG B, et al. ERRT-GA: Expert genetic algorithm with rapidly exploring random tree initialization for multi-uav path planning[J]. Drones, 2024, 8(8): 367.

[6] HE W, QI X, LIU L. A novel hybrid particle swarm optimization for multi-UAV cooperate path planning [J]. Applied Intelligence, 2021, 51(10): 7350-7364.

[7] TIAN Y, ZHANG J, WANG Q, et al. Application of hybrid algorithm based on ant colony optimization and sparrow search in UAV path planning[J]. International Journal of Computational Intelligence Systems, 2024, 17(1): 286.

[8] LI Y, DONG X, DING Q, et al. Improved A-STAR algorithm for power line inspection UAV path planning [J]. Energies, 2024, 17(21): 5364.

[9] WU Q, TAN W, ZHAN R, et al. GLBWOA: A global-local balanced whale optimization algorithm for UAV path planning[J]. Electronics, 2024, 13(23): 4598.

[10] YU X, JIANG N, WANG X, et al. A hybrid algorithm based on grey wolf optimizer and differential evolution for UAV path planning[J]. Expert Systems with Applications, 2023, 215: 1-13.

[11] XUE J, SHEN B. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization[J]. The Journal of Supercomputing, 2023, 79(7): 7305-7336.

[12] DHIMAN G, KUMAR V. Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems[J]. Knowledge-Based Systems, 2019, 165: 169-196.

[13] DAS S, MANDAL A, MUKHERJEE R. An adaptive differential evolution algorithm for global optimization in dynamic environments[J]. IEEE Transactions on Cybernetics, 2013, 44(6): 966-978.

[14] XIE L, HAN T, ZHOU H, et al. Tuna swarm optimization: A novel swarm-based metaheuristic algorithm for global optimization[J]. Computational intelligence and Neuroscience, 2021, 2021: 1-22.

[15] OLADEJ O, SUNDAY O, STEPHEN O, et al. The hiking optimization algorithm: A novel human-based metaheuristic approach[J]. Knowledge-Based Systems, 2024, 296: 111880.

[16] MIRJALILI S, MIRJALILI S M, LEWIS A. Grey wolf optimizer[J]. Advances in Engineering Software, 2014, 69: 46-61.

[17] MIRJALILI S, LEWIS A. The whale optimization algorithm[J]. Advances in Engineering Software, 2016, 95: 51-67.

[18] 郭琴，郑巧仙.多策略改进的蜣螂优化算法及其应用[J].计算机科学与探索，2024，18(4)：930-946.
GUO Qin, ZHENG Qiaoxian. Dung beetle optimization algorithm with multi-strategy improvement and its application[J].Journal of Computer Science and Exploration, 2024, 18(4): 930-946.

[19] 王玉芳，程培浩.融合多策略改进的鲸鱼优化算法[EB/OL]. [2025-03-06]. http://kns.cnki.net/kcms/detail/11.2127.TP.20241120.1704.010.html.
WANG Yufang, CHENG Peihao. Fusion strategy more whales optimization algorithm[EB/OL]. [2025-03-06]. http://kns.cnki.net/kcms/detail/11.2127.TP.20241120.1704.010.html.

（编辑：陈珺）