DOI:10.19651/j.cnki.emt.2416358

基于改进蜣螂优化算法的无人机航迹规划*

刘文强 李 涛
(南京信息工程大学自动化学院 南京 210044)

摘 要:针对蜣螂优化算法(DBO)存在全局探索能力不足、容易陷入局部最优,致使无人机三维航迹规划效果不佳的问题,设计了一种改进的蜣螂优化算法(EDBO)。首先,从偷窃蜣螂种群中划分出一种融合灰狼领袖群思想的领袖蜣螂群,增强了算法的多样性和鲁棒性;其次,使用种群切换策略,同编号的蜣螂个体行为不再固定,提高了算法全局探索和局部挖掘的能力;最后,利用 Beta 分布动态反向学习策略帮助蜣螂更好的进化。将所提算法与 4 种优化算法进行了无人机三维航迹规划对比,仿真结果表明,在 3 个场景下,EDBO 可以更稳定的生成代价函数值更小的航迹,平均代价函数值相较于原始 DBO 算法分别减少了 9.8%,10.4%,16.5%。

关键词:无人机航迹规划;蜣螂优化算法;动态反向学习
中图分类号:V249;TN96 **文献标识码**:A **国家标准学科分类代码**:520.1040

UAV trajectory planning based on enhanced dung beetle optimization algorithm

Liu Wenqiang Li Tao
(School of Automation, Nanjing University of Information Engineering, Nanjing 210044, China)

Abstract: Aiming at the problem that the dung beetle optimization algorithm (DBO) has insufficient global exploration ability and easily falls into the local optimum, resulting in poor UAV 3D trajectory planning, an improved dung beetle optimization algorithm (EDBO) is designed. Firstly, a leader dung beetle group incorporating the idea of a grey wolf leader group is divided from the stealing dung beetle population to enhance the diversity and robustness of the algorithm; secondly, a population switching strategy is used, where the behaviour of dung beetle individuals with the same number is no longer fixed to improve the algorithm's ability of global exploration and local mining; and lastly, a Beta-distributed dynamic inverse learning strategy is used to help dung beetles to evolve better. The proposed algorithm is compared with 4 optimization algorithms for UAV 3D trajectory planning, and the simulation results show that EDBO can generate trajectories with smaller cost function values more stably in the 3 scenarios, and the average cost function values are reduced by 9.8%, 10.4%, and 16.5% compared to the original DBO algorithm, respectively.

Keywords: UAV track planning;dung beetle optimization algorithm;dynamic reverse learning

0 引 言

近年来,无人机在复杂和危险环境中的潜在应用引起了广泛关注。在无人机任务系统中,航迹规划设计是至关重要的一环,其核心目标是在特定约束条件下制定一条从起始位置到目的位置的安全、可行和平稳的飞行航迹[1]。

传统算法在设计无人机三维航迹时通常遇到复杂且效果不佳的问题,而群智能算法能够快速找到可行的解决方案[2]。例如粒子群算法(particle swarm optimization, PSO)[3]、麻雀算法[4]、沙丘猫算法(sand cat swarm optimization, SCSO)[5]等。但群智能算法仍有提升空间,已经有许多改进的群智能算法成功应用到了无人机航迹规划。王康等[6]提出了一种全新的非线性调整机制和自适应莱维飞行机制的改进 SCSO(levy flight improved SCSO, LVSCSO)算法,提高了算法的寻优能力。辛守庭等[7]提出了一种折线形惯性权重因子改进的 PSO(improved PSO, IPSO)改善了航迹规划的成功率的同时,减少了航迹长度。杨教等[8]将粒子群算法迭代的最优解用人工蜂群算法进行了二次寻优,提高了算法的寻优能力。

蜣螂优化算法(dung beetle optimizer, DBO)[9]是 2023

收稿日期:2024-07-02
*基金项目:国家自然科学基金(62373195)项目资助

年提出的一种新型群体智能优化算法,该算法通过一系列众所周知的数学测试函数展现了其搜索能力。Hu 等[10]运用 DBO 算法对 5 种机器学习模型中的最佳预测进行优化,潘志远等[11]将 DBO 用于优化定位算法,由此可见,该算法在处理参数优化问题上的有效性。DBO 算法具备全局探索和局部开发的特性,表现出快速收敛和高求解精度的优点。但也存在全局探索和局部开发能力不平衡,易陷入局部最优的问题。

目前,Zhang 等[12]设计了一种自适应更新滚球蜣螂的策略的 DBO(improved DBO,IDBO),通过考虑了种群交互,对部分滚球蜣螂的行为进行了变更,提高了 DBO 的搜索性能;潘劲成等[13]提出了一种改进正弦引导的 DBO(DBO guided by improved sine algorithm,MSADBO)算法提高了蜣螂的全局探索和局部开发能力;董奕含等[14]采用 Halton 序列初始化种群的位置(halton sequence initialised DBO,HDBO),提高了初始种群的质量,并应用于频散曲线反演。Shen 等[15]将一种多策略改进的 DBO(multi-strategy enhanced DBO,MDBO)算法应用于无人机航迹规划,极大的提高了算法的优化精度和稳定性,但增加了算法的时间开销。以上文献提升了 DBO 的寻优性能,但是未很好地利用种群之间的交互来帮助蜣螂进化。为此,本文为提高无人机航迹规划的效果,提出了一种改进的 DBO(enhanced DBO,EDBO),主要贡献如下:

首先,引入灰狼领袖群思想从偷窃蜣螂中划分出一种领袖蜣螂群,改变了以往只依赖最优或最差值引导的模式来带领蜣螂进化,增强了算法的多样性和鲁棒性;其次,设计了种群切换策略,改变了原本蜣螂个体的划分模式,同编号的蜣螂可以执行最多 5 种行为,促进了蜣螂个体之间行为的交流,为蜣螂群进化奠定了基础,提高了蜣螂群体全局探索和局部挖掘的能力;最后,利用 Beta 分布动态方向学习策略来探索更多的搜索空间,帮助算法更好的跳出局部最优。通过无人机航迹规划实验验证了改进后的蜣螂优化算法效果更佳。

1 无人机航迹规划代价函数设计

本文将环境中存在的危险源集合记为 K,每个障碍物简化设定为圆柱体模型。从以下 4 个方面设置代价函数。

1.1 航程代价约束

设搜索地图中的航迹节点坐标为 $P_i=(X_i,Y_i,Z_i)$。为了节约燃料成本和时间成本,规划的航段需满足最短航迹段约束为:

$$\begin{cases} F_1=\sum_{i=1}^{n-1} l_i \\ l_i=\sqrt{(X_{i+1}-X_i)^2+(Y_{i+1}-Y_{i+1})^2+(Z_{i+1}-Z_i)^2} \end{cases} \tag{1}$$

其中,l_i 为第 i 段的飞行航程。

1.2 碰撞威胁代价约束

计划航迹还需要通过引导无人机通过通常由操作空间中出现的障碍物引起的威胁来确保无人机的安全操作。威胁成本计算如下:

$$\begin{cases} F_2=\sum_{i=1}^{n-1}\sum_{k=1}^{K} T_k(l_i) \\ T_k(l_i)=\begin{cases} 0, & d_k>S+D+R_k \\ S+D+R_k-d_k, & D+R_k<d_k\leqslant S+D+R_k \\ \infty, & d_k\leqslant D+R_k \end{cases} \end{cases} \tag{2}$$

设每个其中 C_k 为威胁中心坐标为、R_k 为半径,对于给定的航迹段 l 到 C_k 的距离为 d_k,如图 1 所示。

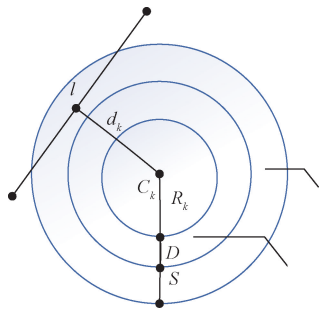


图 1 威胁成本的确定

Fig. 1 Determination of threat costs

1.3 飞行高度约束

在操作过程中,飞行高度通常被限制在最小高度和最大高度之间,分别记为 $h_{\min}$ 和 $h_{\max}$。成本计算为:

$$\begin{cases} F_3=\sum_{i=1}^{n} H_i \\ H_i=\begin{cases} \left| h_i-\dfrac{(h_{\max}+h_{\min})}{2}\right|, h_{\min}\leqslant h_i\leqslant h_{\max} \\ \infty \end{cases} \end{cases} \tag{3}$$

其中,h_i 为相对于地面的飞行高度。

1.4 平滑成本约束

平滑成本评估转弯和爬坡率,这对生成可行的航迹至关重要。P_i 航路点到 P_{i+1} 航路点的水平方向的投影可表示为向量 $\boldsymbol{k}_i=(X_{i+1}-X_i,Y_{i+1}-Y_i)$。最大偏转角记为 $\phi_{\max}$,最大俯仰角记为 $\varphi_{\max}$,则偏转角和俯仰角的计算如下:

$$\phi=\begin{cases} \arccos\left(\dfrac{\boldsymbol{k}_i^{\mathrm{T}}\boldsymbol{k}_{i+1}}{|\boldsymbol{k}_i||\boldsymbol{k}_{i+1}|}\right), \phi<\phi_{\max} \\ \phi_{\max} \end{cases} \tag{4}$$

$$\varphi=\begin{cases} \arccos\left(\dfrac{|Z_{i+1}-Z_i|}{\sqrt{(X_{i+1}-X_i)^2}+\sqrt{(Y_{i+1}-Y_i)^2}}\right), \varphi<\varphi_{\max} \\ \varphi_{\max} \end{cases} \tag{5}$$

平滑成本计算如下:

$$F_4 = a_1 \sum_{i=1}^{n-1} \phi_i + a_2 \sum_{i=1}^{n-1} \varphi_i \tag{6}$$

其中，a_1 和 a_2 分别为偏转角和俯仰角的惩罚系数。

1.5　总的航迹代价函数

$$F = \sum_{i=1}^{4} b_i F_i \tag{7}$$

其中，F 为航迹成本总代价，b_i 为每种代价函数的权重系数。本文无人机航迹规划部分采用此代价函数生成满足约束的离散航迹点，最后对航迹点进行平滑处理后得到最终航迹。

2　标准 DBO 算法

在 DBO 算法中，通过模拟了蜣螂的滚球、跳舞、觅食、繁殖和盗窃行为来更新候选解。每个蜣螂群由 4 种不同的代理蜣螂组成。

滚球蜣螂具备滚球和跳舞 2 种行为，由阈值决定滚球或跳舞，滚球蜣螂位置更新如下：

$$\boldsymbol{x}_i(t+1) = \begin{cases} \boldsymbol{x}_i(t) + \alpha \times k \times \boldsymbol{x}_i(t-1) + b \times |\boldsymbol{x}_i(t) - \boldsymbol{X}^w|, \lambda < r_d \\ \boldsymbol{x}_i(t) + \tan(\theta) |\boldsymbol{x}_i(t) - \boldsymbol{x}_i(t-1)| \end{cases} \tag{8}$$

其中，t 为当前迭代次数，$\boldsymbol{x}_i(t)$ 为第 i 只蜣螂在第 t 次迭代时的位置，$k \in (0, 0.2]$ 为定值。b 是属于 $(0,1)$ 的定值，α 赋值为 -1 或 1，$\boldsymbol{X}^w$ 为全局最差位置，$|\boldsymbol{x}_i(t) - \boldsymbol{X}^w|$ 用来模拟光强的变化，$\lambda \in [0,1)$ 的随机值，r_d 为滚球阈值决定了蜣螂滚球或跳舞，取值为 0.9。$\theta \in [0, \pi]$，如果 θ 取值 $0, \pi/2$ 或 π 时，蜣螂位置将不会进行更新。

为了可以安全繁殖后代，就必须选择合适的地方进行产卵，产卵区的下界和上界定义为：

$$\begin{aligned} Lb^* &= \max(\boldsymbol{X}^* \times (1-R), Lb), \\ Ub^* &= \min(\boldsymbol{X}^* \times (1+R), Ub) \end{aligned} \tag{9}$$

其中，$\boldsymbol{X}^*$ 表示当前局部最佳位置，$R = 1 - t/T_{\max}$，$T_{\max}$ 表示最大迭代次数，Lb 和 Ub 分别为优化问题的下界和上界。

繁育球位置更新如下：

$$\boldsymbol{x}_i(t+1) = \boldsymbol{X}^* + \boldsymbol{b}_1 \times (\boldsymbol{x}_i(t) - Lb^*) + \boldsymbol{b}_2 \times (\boldsymbol{x}_i(t) - Ub^*) \tag{10}$$

其中，$\boldsymbol{b}_1$ 和 $\boldsymbol{b}_2$ 表示 2 个大小为 $1 \times D$ 的独立随机向量，D 表示优化问题的维数。小蜣螂觅食区的下界和上界定义如下：

$$\begin{aligned} Lb^b &= \max(\boldsymbol{X}^b \times (1-R), Lb), \\ Ub^b &= \min(\boldsymbol{X}^b \times (1-R), Ub) \end{aligned} \tag{11}$$

其中，$\boldsymbol{X}^b$ 表示当前全局最佳位置。

小蜣螂的位置更新如下：

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + C_1 \times (\boldsymbol{x}_i(t) - Lb^b) + \boldsymbol{C}_2 \times (\boldsymbol{x}_i(t) - Ub^b) \tag{12}$$

其中，C_1 为服从正态分布的随机数，$\boldsymbol{C}_2$ 为属于 $(0,1)$ 的随机向量。

盗窃蜣螂的位置更新如下：

$$\boldsymbol{x}_i(t+1) = \boldsymbol{X}^b + \mathrm{S} \times \boldsymbol{g} \times (|\boldsymbol{x}_i(t) - \boldsymbol{X}^*| + |\boldsymbol{x}_i(t) - \boldsymbol{X}^b|) \tag{13}$$

其中，$\boldsymbol{g}$ 为 $1 \times D$ 服从正态分布的随机向量；S 为常数。

原蜣螂位置更新模式为：

$$x_i(t+1) = \begin{cases} \begin{cases} \boldsymbol{x}_i(t) + \alpha \times k \times \boldsymbol{x}_i(t-1) + b \times |\boldsymbol{x}_i(t) - \boldsymbol{X}^w|, \lambda < r_d \\ \boldsymbol{x}_i(t) + \tan(\theta) |\boldsymbol{x}_i(t) - \boldsymbol{x}_i(t-1)| \end{cases}, & 0 < i \leqslant m_1 \\ \boldsymbol{x}_i(t+1) = \boldsymbol{X}^* + \boldsymbol{b}_1 \times (\boldsymbol{x}_i(t) - Lb^*) + \boldsymbol{b}_2 \times (\boldsymbol{x}_i(t) - Ub^*), & m_1 < i \leqslant m_2 \\ \boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + C_1 \times (\boldsymbol{x}_i(t) - Lb^b) + \boldsymbol{C}_2 \times (\boldsymbol{x}_i(t) - Ub^b), & m_2 < i \leqslant m_3 \\ \boldsymbol{x}_i(t+1) = \boldsymbol{X}^b + \mathrm{S} \times \boldsymbol{g} \times (|\boldsymbol{x}_i(t) - \boldsymbol{X}^*| + |\boldsymbol{x}_i(t) - \boldsymbol{X}^b|), & m_3 < i \leqslant N \end{cases} \tag{14}$$

式中：m_1, m_2, m_3 为控制不同蜣螂职能编号的参数，种群的个体总数为 N，文献[9]中 N 取值为 30，m_1, m_2, m_3 取值分别 6,12,19，即蜣螂编号为 1～6 为滚球蜣螂，进行滚球或跳舞行为，蜣螂编号为 7～12 为繁育蜣螂，进行繁育，蜣螂编号为 13～19 为小蜣螂，进行觅食，蜣螂编号为 20～30 为盗窃蜣螂，进行偷窃行为。

3　改进的 DBO 算法

DBO 虽有着不同搜索机制，但仍存在全局探索能力欠缺、易陷入局部最优的问题。EDBO 采用了领袖蜣螂群思想、种群切换策略以及基于 Beta 分布的动态方向学习策略，在兼顾算法的探索与开发的同时，提高了算法的寻优性能。

3.1　融合灰狼领袖思想的领袖蜣螂群

因为蜣螂行为中往往仅依赖于最差值或者最优值，容易陷入局部最优，灰狼算法是根据 3 匹头狼的位置来更新个体的位置，类似的取出蜣螂的全局前三优解来构建一个领袖蜣螂群。然而，种群总数固定，增加一个种群必然会使其他种群的蜣螂减少。其中盗窃蜣螂的主要是借鉴最优值对解空间进行开发，而蜣螂优化算法的探索机制不缺乏最优值之间的开发，故本文牺牲偷窃蜣螂数量来划分出一个融合灰狼思想的种群。领袖蜣螂群采取以下位置更新公式：

$$\boldsymbol{x}_i(t+1) = \varepsilon \times \boldsymbol{x}_i(t) + q \times (1-\varepsilon) \times (\boldsymbol{X}^b(t) + \boldsymbol{X}^s(t) + \boldsymbol{X}^t(t)) \times \eta \tag{15}$$

其中，$\boldsymbol{X}^s$、$\boldsymbol{X}^t$ 为当前全局次优解和第三优解的位置，$\varepsilon = t/T_{\max}$，q 为权重系数，η 为扰动系数定义如下：

$$\eta = \tan[(r_2 - 0.5) \times \pi] \tag{16}$$

引入上述策略改进后，蜣螂种群不再仅仅依赖最优或最差位置的信息，还多了一组由蜣螂领袖组成的群体，领袖群会对蜣螂种群有一个指引作用，在迭代初期所占的权重大，有利于给当前蜣螂提供一个更强的导向作用，在后期，个体能力被逐渐发掘，有利于避免蜣螂后期快速趋于同化，同时引入了扰动系数，帮助蜣螂更好的进化。

3.2 种群切换策略

文献[9]中各比例固定,其中蜣螂各司其职,缺乏彼此间的交流,使得算法的性能有所降低。通过动态调整生产者数量,并根据一定的策略控制生产者之间的竞争与合作,可以增强算法的多样性和稳定性。文献[12]中虽对编号为 7～12 的滚球蜣螂进行了行为变更,但是随着迭代的进行各种群的交互会逐渐减弱,所以本文提出一种种群切换策略,在开始迭代时,并不将蜣螂个体行为固定,而是随机选择一个种群进行探索,基于种群切换蜣螂位置更新模式为:

$$x_i(t+1)=\begin{cases}\begin{cases}x_i(t)+\alpha\times k\times x_i(t-1)+b\times\mid x_i(t)-X^w\mid, & \lambda<r_d\\ x_i(t)+\tan(\theta)\mid x_i(t)-x_i(t-1)\mid\end{cases}, & 0<R_r(1,i)\leqslant\omega_1\\ x_i(t+1)=X^*+b_1\times(x_i(t)-Lb^*)+b_2\times(x_i(t)-Ub^*), & \omega_1<R_r(1,i)\leqslant\omega_2\\ x_i(t+1)=x_i(t)+C_1\times(x_i(t)-Lb^b)+C_2\times(x_i(t)-Ub^b), & \omega_2<R_r(1,i)\leqslant\omega_3\\ x_i(t+1)=X^b+S\times g\times(\mid x_i(t)-X^*\mid+\mid x_i(t)-X^b\mid), & \omega_3<R_r(1,i)\leqslant\omega_4\\ x_i(t+1)=\varepsilon\times x_i(t)+q\times(1-\varepsilon)\times(X^b(t)+X^s(t)+X^t(t))\times\eta, & \omega_4<R_r(1,i)\leqslant1\end{cases}\tag{17}$$

其中,$\boldsymbol{R}_r$ 表示大小为 $1\times N$ 的随机向量,ω_1、ω_2、ω_3、ω_4 为决定蜣螂行为的阈值,不妨设选中 5 个种群的概率均为 0.2,ω_1、ω_2、ω_3、ω_4 取值为 0.2、0.4、0.6、0.8,与式(14)最大的不同在于编号为 i 的个体职责不再固定,随着迭代的进行,编号同为 i 的蜣螂个体可以参与到 5 种搜索机制。从此,蜣螂内部具有了社会性质,个体迭代方式不再固定,每个个体都兼顾了局部探索和全局开发的能力,为蜣螂群体共同进化奠定了基础。

3.3 基于 Beta 动态分布的反向学习

随着迭代的进行,蜣螂个体会逐渐趋于同化,想要发生进化的难度会增大。本文利用 Beta 分布生成反射解,将 Beta 分布函数定义为:

$$\begin{cases}f(x)=B(\alpha,\beta)^{-1}x^{\alpha-1}(1-x)^{\beta-1}, x\in[0,1]\\ B(\alpha,\beta)=\int t^{\alpha-1}(1-t)^{\beta-1}\mathrm{d}t\end{cases}\tag{18}$$

式中:α 和 β 均取 0.5,假设维数为 D,最大迭代次数为 $T_{\max}$,第 j 维的上界和下界为 $[Lb^j,Ub^j]$,逆解为 $\boldsymbol{u}$,根据以下公式生成:

$$u_{i,j}(t)=\begin{cases}r_{i,j}(t), & x_{i,j}(t)\geqslant s_j\\ p_{i,j}(t), & x_{i,j}(t)<s_j\end{cases}\tag{19}$$

其中,$i\in[1,N]$,$j\in[1,D]$,$t\in[1,T_{\max}]$,$x_{i,j}(t)$ 为第 i 个解第 j 维在第 t 次迭代时的迭代值;$u_{i,j}(t)$ 是第 i 个反射解第 j 维在第 t 次迭代时的迭代值。

$$r_{i,j}(t)=\begin{cases}\Phi(Ub^j+Lb^j-x_{i,j}(t),s_j), & S<\Phi(0,1)\\ \Phi(Lb^j,Ub^j+Lb^j-x_{i,j}(t)), & \text{其他}\end{cases}\tag{20}$$

$$p_{i,j}(t)=\begin{cases}\Phi(s_j,Ub^j+Lb^j-x_{i,j}(t)), & S<\Phi(0,1)\\ \Phi(Ub^j+Lb^j-x_{i,j}(t),Ub^j), & \text{其他}\end{cases}\tag{21}$$

$$S=\frac{\left|x_{i,j}(t)-\frac{1}{2}(Ub^j+Lb^j)\right|}{\mid Ub^j-Lb^j\mid}\tag{22}$$

$$s_j=\frac{1}{2}(Ub^j+Lb^j)\tag{23}$$

$$\Phi(a,b)=a+(b-a)\times\mathrm{betarnd}(\alpha,\beta)\tag{24}$$

$$\boldsymbol{x}_i(t)=\begin{cases}\boldsymbol{x}_i(t), & f(\boldsymbol{x}_i(t))\leqslant f(\boldsymbol{u}_i(t))\\ \boldsymbol{u}_i(t)\end{cases}\tag{25}$$

s_j 为上下边界的域中心;$\Phi(a,b)$在(a,b)内生成一个遵循 Beta 分布的随机数;*betarnd* 是一个函数,它随机生成一个服从 Beta 分布的数字。将生成的反向解与原位置进行比较,保留较优值。本文每 10 次迭代后选出 10 个蜣螂生成反向解,将本文算法增加的复杂度与文献[13]进行比较,文献[13]对每一代的最优值进行高斯柯西变异后择优,多增加了 $T_{\max}$ 次位置更新,本文增加了 $T_{\max}/10\times10=T_{\max}$ 次,两者是相同的,同时为了进一步定量分析,本文在后续航迹规划实验中加入了运行时间。

3.4 EDBO 算法流程图

EDBO 算法流程图如图 2 所示。

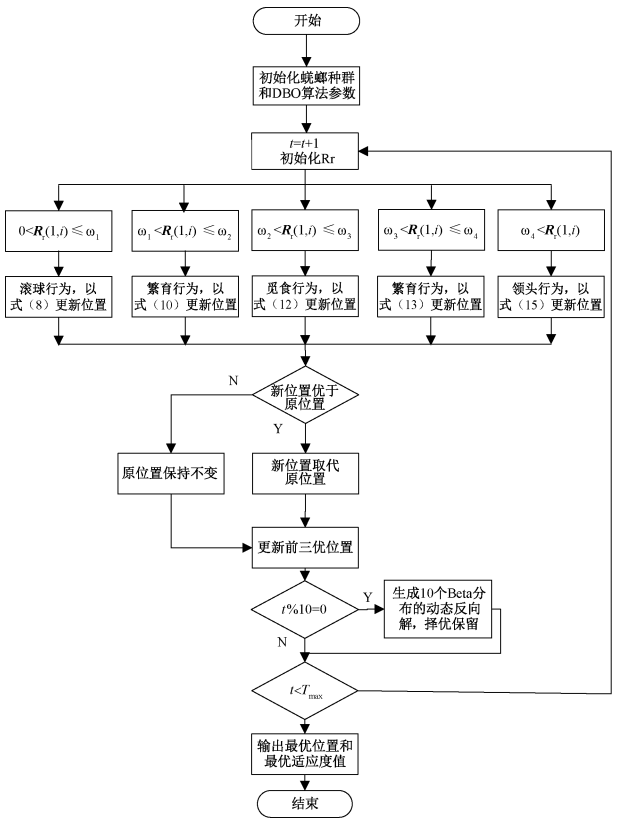


图 2 EDBO 算法流程图

Fig. 2 Flowchart of EDBO algorithm

1)初始化蜣螂种群的相关参数:种群规模 N、最大迭代次数 $t_{\max}$、空间维度 dim、随机向量 $\boldsymbol{R}_r$ 等;

2)编号为 i 的蜣螂通过 $\boldsymbol{R}_r$ 与给定概率比较来决定执行式(8)、(10)、(12)、(13)或(15)更新其位置;

3)更新目前的最优解与适应度值；

4)每 10 次迭代后随机选取 10 个当代蜣螂进行反向学习，并与原位置进行比较，择优保留；

5)重复步骤 2)～4)直到达到最大迭代次数，并将最优参数输出，即所求问题的最优解，算法终止。

4　实验仿真及分析

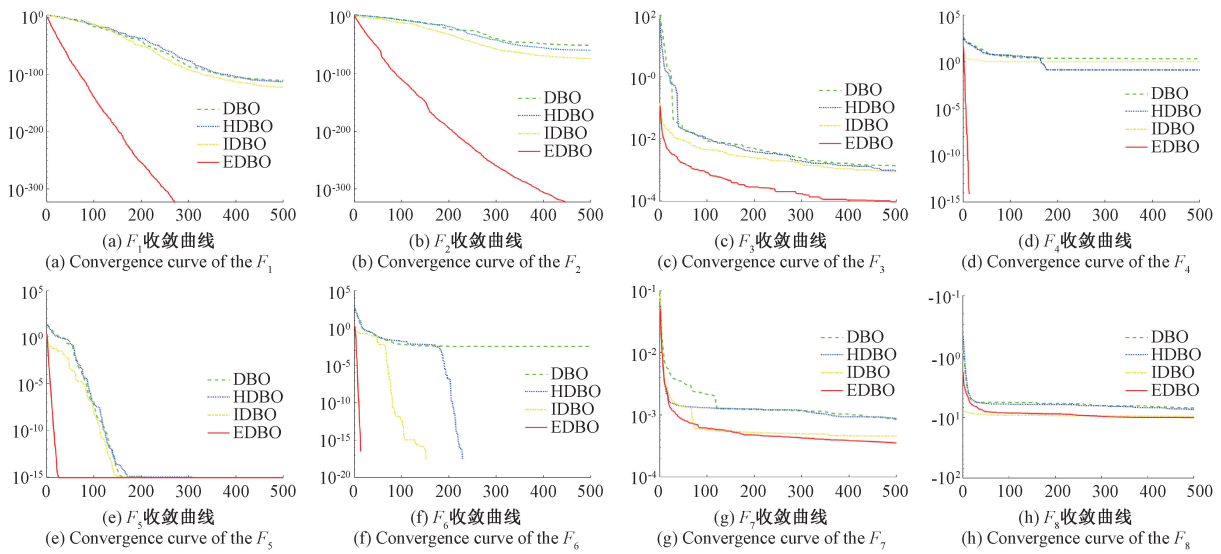
4.1　EDBO 算法性能分析

为了测试本文算法的性能，在 Matlab2019a 平台选用 8 个经典的标准测试函数测试算法进行仿真实验，其中，F_1～F_3 为单峰优化问题，F_4～F_6 为多峰优化问题，F_7、F_8 为固定维优化问题，测试函数其余特征如表 1 所示。因为该部分对比的文献算法都是 DBO 的改进版本，所以为了公平起见，算法中 k 统一取 0.1，u 统一取 0.3，所有算法种群规模设置为 30，最大迭代次数设置为 500，各独立运行 30 次。将本文所提的 EDBO 算法与原始 DBO 算法、用 *Halton* 映射初始化种群的 HDBO[14]、带有种群交互思想的 IDBO[12] 两个文献算法在测试函数上进行比较，4 个算法在测试函数的迭代曲线如图 3 所示，同时为了进一步突出本算法的寻优能力，在对比结果中加入了改进正弦算法引导的蜣螂优化算法[13]，其中文献[13]数据均来自于原文献。表 3 使用平均值和标准差作为实验结果评价标准，将最优值进行了加粗。

表 1　基准测试函数

Table 1　Benchmarking functions

函数表达式	维度	搜索空间	最优值
$F_1(x)=\sum_{i=1}^{n}x^2$	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
$F_3(x)=\sum_{i=1}^{n}ix_i^4+\mathrm{random}[0,1)$	30	[−1.28,1.28]	0
$F_4(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	30	[−5.12,5.12]	0
$F_5(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	30	[−32,32]	0
$F_6(x)=\frac{1}{4\,000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	30	[−600,600]	0
$F_7(x)=\sum_{i=1}^{11}[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}]^2$	4	[−5,5]	0.000 307 5
$F_8(x)=-\sum_{i=1}^{5}[(x-a_i)(x-a_i)^T+c_i]^{-1}$	4	[0,10]	−10.152 3



(a) F_1收敛曲线 (a) Convergence curve of the F_1　(b) F_2收敛曲线 (b) Convergence curve of the F_2　(c) F_3收敛曲线 (c) Convergence curve of the F_3　(d) F_4收敛曲线 (d) Convergence curve of the F_4

(e) F_5收敛曲线 (e) Convergence curve of the F_5　(f) F_6收敛曲线 (f) Convergence curve of the F_6　(g) F_7收敛曲线 (g) Convergence curve of the F_7　(h) F_8收敛曲线 (h) Convergence curve of the F_8

图 3　4 种算法测试收敛图

Fig. 3　Convergence plots of the 4 algorithms tested

由图 3 可以看出,在这些测试函数中,EDBO 算法在迭代初期就展现出较强的全局搜索能力;随着迭代的进行,结合表 2 的统计结果可看出,在 8 个经典测试函数实验中,相较于原始 DBO 算法,3 个文献改进算法在收敛精度和稳定性上都有不同幅度的提升。对于测试函数 F_1、F_4,只有 MSADBO 算法和 EDBO 算法寻得了最优解,表明 MSADBO 算法相较于其他两个文献算法具有一定的优势。在测试函数 F_2、F_8 中只有 EDBO 寻得了理论最优解,说明了 EDBO 算法的寻优能力更强,在测试函数 F_3、F_7 中,EDBO 虽未寻得理论最优值,但 EDBO 算法展现出更高的收敛精度和更小的标准差。F_5 中各算法均寻得了最优解,说明了 DBO 算法本身具有一定的优势。综上所述,EDBO 平均值更优,标准差更小,说明 EDBO 更加稳定,具有更强的寻优能力。EDBO 算法更是比其他算法有更高的收敛精度。

表 2 5 种算法测试函数的对比结果

Table 2 Comparison results of 5 algorithmic test functions

函数	指标	DBO	HDBO	IDBO	MSADBO	EDBO
F_1	平均值	1.14×10^{-110}	1.13×10^{-112}	9.50×10^{-123}	**0**	**0**
	标准差	6.23×10^{-110}	6.19×10^{-112}	5.20×10^{-122}	**0**	**0**
F_2	平均值	1.59×10^{-51}	5.29×10^{-60}	1.52×10^{-74}	4.10×10^{-295}	**0**
	标准差	8.68×10^{-51}	2.19×10^{-59}	7.15×10^{-74}	**0**	**0**
F_3	平均值	1.32×10^{-3}	9.04×10^{-4}	8.30×10^{-4}	9.61×10^{-5}	$\mathbf{8.72\times10^{-5}}$
	标准差	1.18×10^{-3}	9.03×10^{-4}	6.33×10^{-4}	6.78×10^{-5}	$\mathbf{5.61\times10^{-5}}$
F_4	平均值	2.09×10^{0}	1.33×10^{-1}	1.01×10^{0}	**0**	**0**
	标准差	1.15×10^{1}	7.27×10^{1}	5.51×10^{0}	**0**	**0**
F_5	平均值	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$	$\mathbf{8.88\times10^{-16}}$
	标准差	**0**	**0**	**0**	**0**	**0**
F_6	平均值	3.27×10^{-3}	**0**	**0**	**0**	**0**
	标准差	1.79×10^{-2}	**0**	**0**	**0**	**0**
F_7	平均值	8.50×10^{-4}	8.68×10^{-4}	4.57×10^{-4}	3.73×10^{-4}	$\mathbf{3.54\times10^{-4}}$
	标准差	4.98×10^{-4}	3.66×10^{-4}	4.65×10^{-4}	1.21×10^{-4}	$\mathbf{8.22\times10^{-5}}$
F_8	平均值	-6.69×10^{-1}	-2.34×10^{-1}	-9.98×10^{-1}	-1.01×10^{1}	$\mathbf{-1.02\times10^{1}}$
	标准差	2.53×10^{0}	2.38×10^{0}	9.23×10^{-1}	5.23×10^{-2}	$\mathbf{5.12\times10^{-3}}$

4.2 无人机三维航迹规划实验

为验证本文 EDBO 算法航迹规划的可行性及优越性,分别在多个地图上进行无人机航迹规划仿真实验。并与 DBO[9]、IPSO[7]、MDBO[15]、LVSCSO[6]算法航迹规划结果进行比较,各算法种群数、迭代次数以及 DBO、MDBO、EDBO 算法参数设置同测试函数,其中 IPSO 折线斜率是原文献的 2 倍是因为原文献的迭代次数为 1 000,IPSO 中 $\omega_{max}=0.7$, $\omega_{min}=0.3$, $\mu=0.002$, $c_1=2.8$, $c_2=1.3$。LVSCSO 中 $r_G\in[2,0]$, $R\in[-r_G,r_G]$。

在规划空间内进行无人机航迹规划仿真实验,首先构建实验三维环境,为了更加逼近真实环境,设置了较多的威胁区域。其中,环境的面积为 1 000×1 000×400,设置任务的起点为(50,50,100),终点为(750,750,100),无人机的直径 D 设置为 10,威胁成本 S 设置为 20,飞行高度限制在[100,200]。最大偏转角为 ϕ_{max} 和最大俯仰角 φ_{max} 均设置为 45°,4 种代价权重设置分别为 $b_1=0.2$, $b_2=0.4$, $b_3=0.2$, $b_4=0.2$,给定的威胁模型数量为 6,4,9,参数如表 3 所示。

图 4 显示了 5 种算法在场景 1 中的航迹规划结果以及各自的收敛曲线,由图 3(a)和(b)可以明显看出 DBO 算法所获得的航迹最长、最曲折,同时由图 3(c)可以看出 DBO 算法得到的代价函数值最大,IPSO 算法、MDBO 算法 LVSCSO 算法和 EDBO 算法,规划的航迹相近,从图 3(c)可以看出 LVSCSO 算法的收敛速度最快,但最后得到的代价函数值仅优于 DBO,IPSO 和 MDBO 算法的收敛的速度虽然稍慢些,但规划出的代价函数值更小。由于地形简单,各算法前期寻得的航迹代价航迹值接近,EDBO 的前期优势并不明显,但在迭代到 33 次时已经领先于其他 4 个算法,并且随着迭代的进行,仍有寻得更优解的能力。

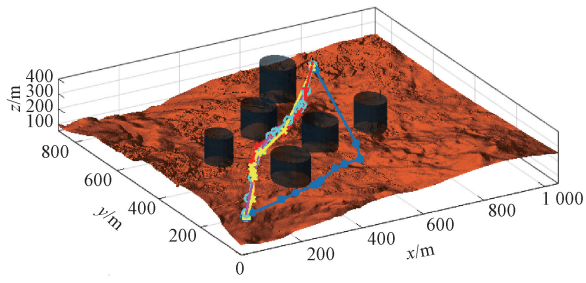
结合图 5 的航迹规划结果,可以看出在场景 2 中,虽然障碍物的数量比场景 1 有所减少,但由于障碍物直接分布在起点和终点的连线上,增大了算法在航迹规划的难度,其中,DBO 算法和 LVSCSO 算法所生成航迹较为冗长,但 LVSCSO 算法规划的航迹较为平滑,航迹成本得到进一步的减少。IPSO 算法和 MDBO 算法在图 3 中所得航迹结果接近,相较于 LVSCSO 的函数代价值得到进一步的减少,EDBO 算法规划的航迹最为平滑的同时,航程代价最小,所以函数代价值最小。

表 3　威胁模型参数设置

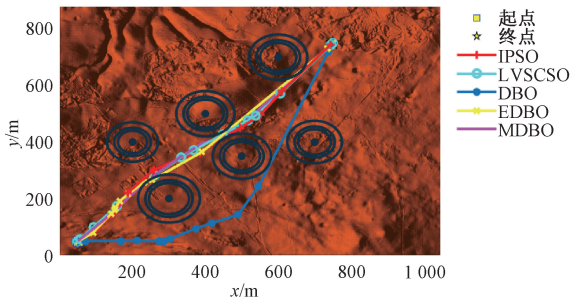
Table 3　Threat model parameter settings

模型	威胁区域中心/m	威胁区域高度和半径/m
场景 1	(200,400)	(100,40)
	(300,200)	(150,55)
	(400,500)	(150,50)
	(500,350)	(200,50)
	(600,700)	(150,50)
	(700,400)	(200,45)
场景 2	(200,200)	(150,50)
	(300,500)	(100,40)
	(500,300)	(200,55)
	(600,600)	(200,55)
场景 3	(100,200)	(150,30)
	(200,100)	(100,20)
	(300,300)	(100,40)
	(300,500)	(200,30)
	(400,150)	(100,30)
	(550,300)	(150,50)
	(500,500)	(100,60)
	(700,600)	(200,30)
	(600,700)	(150,30)

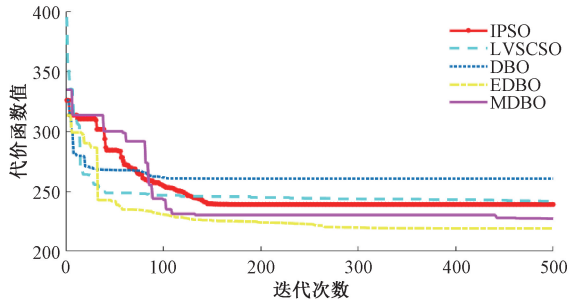
为了进一步比较各算法航迹规划的能力，本文设计了场景 3，场景 3 比场景 1 有着更多的障碍物，比场景 2 障碍物的分布更加复杂，使得航迹规划的难度大幅提升。图 6 显示了 5 种算法在场景 3 中的航迹规划结果以及各自的收敛曲线，可以看出 DBO 算法规划的航迹最长也最曲折，



(a) 飞行航迹3D图
(a) 3D maps of flight tracks



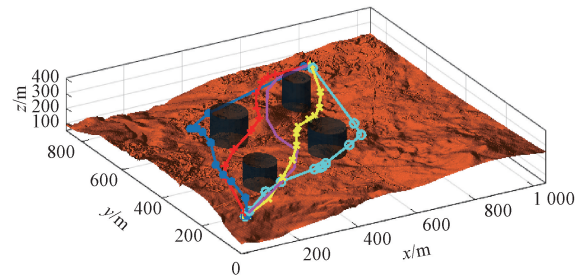
(b) 飞行航迹俯视图
(b) Top view of flight path



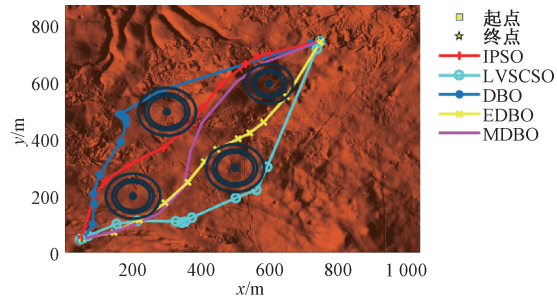
(c) 5种算法收敛图
(c) Convergence diagrams for the 5 algorithms

图 4　5 种算法在场景 1 中的航迹规划对比结果和收敛曲线

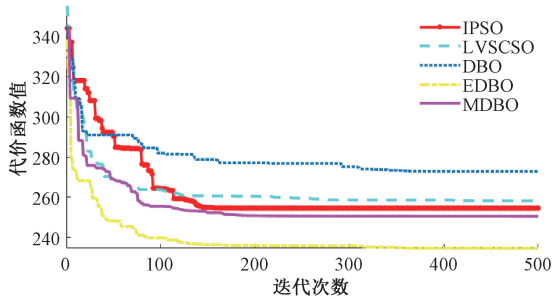
Fig. 4　Comparison results and convergence curves of 5 algorithms for path planning in scene 1



(a) 飞行航迹3D图
(a) 3D maps of flight tracks



(b) 飞行航迹俯视图
(b) Top view of flight path

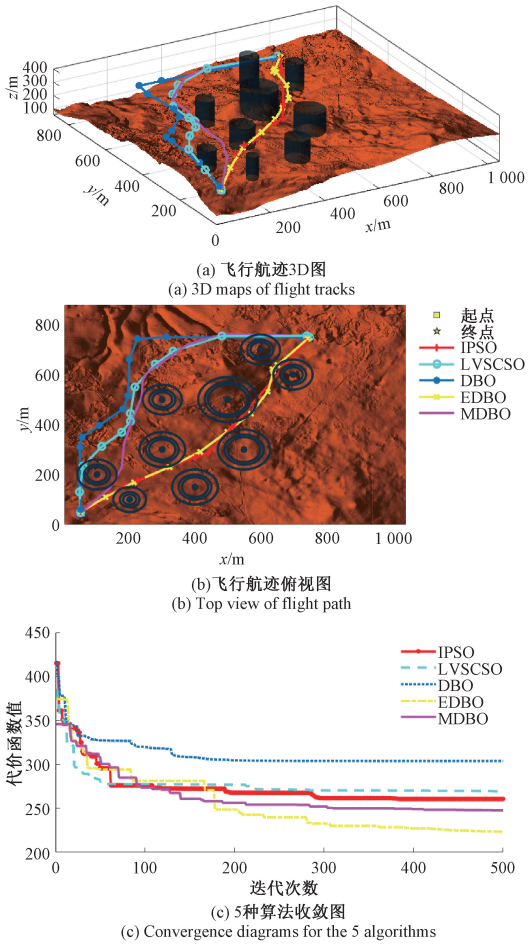


(c) 5种算法收敛图
(c) Convergence diagrams for the 5 algorithms

图 5　5 种算法在场景 2 中的航迹规划对比结果和收敛曲线

Fig. 5　Comparison results and convergence curves of 5 algorithms for path planning in scene 2

MDBO 算法和 LVSCSO 算法规划的航程相近，但 LVSCSO 承受着更多的威胁代价，IPSO 比 MDBO 航程



(a) 飞行航迹3D图
(a) 3D maps of flight tracks

(b)飞行航迹俯视图
(b) Top view of flight path

(c) 5种算法收敛图
(c) Convergence diagrams for the 5 algorithms

图 6 5 种算法在场景 3 中的航迹规划对比结果和收敛曲线

Fig. 6 Comparison results and convergence curves of 5 algorithms for path planning in scene 3

短,但也承受着更多的威胁代价,EDBO 借助适当的转弯率和飞行接近率,更安全地穿越危险区域附近,规划出的三维航迹路线更为合理。

将每个场景的实验均独立运行 30 次,把航迹规划结果记录在表 4 中,分别有最差值,最优值,平均值,标准差和运行时间,用来更好地分析和验证算法的性能。各项指标中的排名第一的值用粗体显示。

其中,DBO 算法除了运行时间上是最优的,其余各项指标均不如其他算法;MDBO 的运行时间最长,其余各项指标仅次于 EDBO,这是由于该算法对每一代每一只的蜣螂都进行了反向学习使得算法复杂度翻倍,同时还对边界控制采取莱维飞行策略,使得每一代的种群随机性都得到了提升,但极大的提升了算法的复杂度;IPSO 算法和 LVSCSO 算法的得到的结果类似,但 IPSO 运行时间比 LVSCSO 较短,这是由于 IPSO 算法并未加入任何贪婪策略,LVSCSO 算法对不同行为的沙猫进行莱维飞行游走后进行了贪婪选择,增加了运行时间成本。EDBO 算法相较于其他 4 种算法除了运行时间都是最优的,说明 EDBO 算法可以稳定地获得最优航迹,随着地图复杂度加大,EDBO 算法相较于其他算法的优势也更大。其平均函数代价值相较于原始 DBO 算法分别提高了 9.8%,10.4%,16.5%。

综上所述,EDBO 算法具有良好的性能并且在不同的场景中可以获得不仅寻优精度更高,适应性也更强的最优航迹。

表 4 航迹规划实验统计结果

Table 4 Statistical results of the trajectory planning experiment

地图	算法	最差值	最优值	平均值	标准差	运行时间/s
场景 1	DBO	295.357	223.208	248.004	18.090	**2.661**
	IPSO	287.236	218.719	237.531	13.039	3.035
	MDBO	269.548	213.670	227.298	10.226	10.379
	LVSCSO	285.943	219.308	239.560	14.774	3.987
	EDBO	**244.563**	**210.065**	**223.798**	**5.665**	3.213
场景 2	DBO	316.704	242.712	265.910	19.270	**2.276**
	IPSO	292.649	236.542	257.142	16.047	2.437
	MDBO	275.201	226.603	250.478	10.415	12.683
	LVSCSO	303.767	228.737	258.140	17.421	3.314
	EDBO	**254.778**	**222.002**	**240.953**	**6.743**	2.749
场景 3	DBO	331.297	247.814	285.261	23.419	**3.345**
	IPSO	329.337	226.801	269.372	18.350	4.045
	MDBO	285.493	230.560	258.089	12.254	14.438
	LVSCSO	324.892	244.261	274.251	19.174	4.896
	EDBO	**264.456**	**222.221**	**244.825**	**8.213**	4.533

5　结　　论

针对蜣螂优化算法(DBO)存在全局探索能力不足、容易陷入局部最优,致使三维航迹规划效果不佳等问题,设计了一种改进的蜣螂优化算法(EDBO)。从偷窃蜣螂种群中划分出一种融合灰狼领袖群思想的领袖蜣螂群,增强了算法的多样性和鲁棒性;使用种群切换策略,建立起不同职能蜣螂之间的联系,提高了算法全局探索和局部挖掘能力;最后利用基于 Beta 分布的动态反向学习策略,进一步提高了算法跳出局部最优的能力。通过与原始 DBO 算法以及 3 个改进 DBO 的文献算法在测试函数上的对比,验证了 EDBO 算法寻优精度和稳定性有了进一步的提高,并且相较于对比文献具有一定优势,又在不同场景下的无人机三维航迹规划实验中与 3 个应用于无人机路径规划的文献算法进行对比,证实了改进算法能够更稳定生成更安全避障且总代价更低的航迹。

参考文献

[1] 张宏宏,甘旭升,李双峰,等. 复杂低空环境下考虑区域风险评估的无人机航路规划[J]. 仪器仪表学报,2021,42(1):257-266.
ZHANG H H, GAN X SH, LI SH F, et al. UAV route planning considering regional risk assessment under complex low altitude environment[J]. Chinese Journal of Scientific Instrument, 2021, 42(1): 257-266.

[2] 李安醌,武丁杰,李诚龙. 低空无人机自主避障算法综述[J]. 电光与控制,2021,28(8):59-64.
LI AN T, WU D J, LI CH L. A survey on autonomous collision avoidance algorithms for UAVs at low altitude[J]. Electronics Optics & Control, 2021, 28(8): 59-64.

[3] 邢燕好,于昊,张佳,等. 基于粒子群参数优化的 O-VMD 数据处理方法研究[J]. 仪器仪表学报,2023,44(4):304-313.
XING Y H, YU H, ZHANG J, et al. Research on the O-VMD thickness measurement data processing method based on particle swarm optimization[J]. Chinese Journal of Scientific Instrument, 2023, 44(4): 304-313.

[4] 朱菊香,谷卫,任明煜,等. 基于 SWT-ISSA-LSTM 的地铁空气质量预测建模[J]. 国外电子测量技术,2023,42(7):164-174.
ZHU J X, GU W, REN M Y, et al. Modeling of subway air quality prediction based on SWT-ISA-LSTM[J]. Foreign Electronic Measurement Technology, 2023, 42(7): 164-174.

[5] SEYYEDABBASI A, KIANI F. Sand cat swarm optimization: A nature-inspired algorithm to solve global optimization problems[J]. Engineering with Computers, 2022, 39(4): 2627-2651.

[6] 王康,司鹏,陈莉,等. 基于改进沙猫群算法的无人机三维航迹规划[J]. 兵工学报,2023,44(11):3382-3393.
WANG K, SI P, CHEN L, et al. 3D path planning of unmanned aerial vehicle based on enhanced sand cat swarm optimization algorithm[J]. Acta Armamentarii, 2023, 44(11): 3382-3393.

[7] 辛守庭,赵冠宇,王晓光,等. 基于改进粒子群算法的旋翼无人机三维航迹规划[J]. 飞行力学,2022,40(5):47-52.
XIN SH T, ZHAO G Y, WANG X G, et al. 3D trajectory planning of rotor UAV based on improved PSO algorithm[J]. Flight Dynamics, 2022, 40(5): 47-52.

[8] 杨教,陆安江,彭熙舜,等. 基于改进粒子群算法的三维路径规划研究[J]. 电子测量技术,2023,46(12):92-97.
YANG J, LU AN J, PENG X SH, et al. Research on 3D path planning based on improved particle swarm optimization[J]. Electronic Measurement Technology, 2023, 46(12): 92-97.

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

[10] HU T Y, ZHANG H, ZHOU J T. Prediction of the debonding failure of beams strengthened with FRP through machine learning models[J]. Buildings, 2023, 13(3): 608.

[11] 潘志远,卜凡亮. 基于蜣螂算法优化的 DV-Hop 定位算法[J]. 电子测量与仪器学报,2023,37(7): 33-41.
PAN ZH Y, BU F L. DV-Hop localization algorithm optimized based on dung beetle optimizer[J]. Electronic Measurement and Instrumentation, 2023, 37(7): 33-41.

[12] ZHANG R Z, ZHU Y J. Predicting the mechanical properties of heat-treated woods using optimization-algorithm-based BPNN[J]. Forests, 2023, 14(5): 935.

[13] 潘劲成,李少波,周鹏,等. 改进正弦算法引导的蜣螂优化算法[J]. 计算机工程与应用,2023,59(22):92-110.
PAN J CH, LI SH B, ZHOU P, et al. Dung beetle optimization algorithm guided by improved sine algorithm[J]. Computer Engineering and Applications, 2023, 59(22): 92-110.

[14] 董奕含,喻志超,胡天跃,等. 基于改进蜣螂优化算法的瑞雷波频散曲线反演方法[J]. 油气地质与采收率,2023,30(4):86-97.
DONG Y H, YU ZH CH, HU T Y, et al. Inversion of rayleigh wave dispersion curve based on improved dung beetle optimizer algorithm[J]. Petroleum Geology and Recovery Efficiency, 2023, 30(4): 86-97.

[15] SHEN Q W, ZHANG D, XIE M SH, et al. Multi-strategy enhanced dung beetle optimizer and its application in three-dimensional UAV path planning[J]. Symmetry, 2023, 15(7): 1432.

作者简介

刘文强,硕士研究生,主要研究方向为智能算法、路径规划等。

E-mail:1553083643@qq.com

李涛,教授,博士生导师,主要研究方向为非线性系统控制理论及应用、智能控制等。

E-mail:litaojia@163.com