http://bhxb.buaa.edu.cn jbuaa@buaa.edu.cn
DOI: 10.13700/j.bh.1001-5965.2022.0502

基于 ε-level 蝙蝠算法的无人机三维航迹规划

王福仪，孟秀云*，张海阔
(北京理工大学 宇航学院，北京 100081)

摘 要：针对复杂地形环境、存在各种威胁和约束的无人机航迹规划问题，提出了一种基于 ε-level 改进蝙蝠算法的航迹规划算法。根据无人机目标函数和约束条件，建立无人机三维航迹规划模型；其次，针对蝙蝠算法处理高维带约束问题存在的早熟现象，设计了自适应权重系数和迭代阈值以平衡算法的探索能力和开发能力，并结合 ε-level 比较策略提高算法对非凸非线性约束优化问题的处理能力；设计了转弯半径可变的 Dubins 曲线用来对航迹进行平滑处理并解决 2 个航迹点连线贯穿地形的问题。通过仿真实验，并与蝙蝠算法、粒子群优化算法、ε-level 粒子群优化算法和 ε-level 差分进化算法相比较，所提算法在开发能力、稳定性和成功率等方面都表现出较优的性能。

关 键 词：无人机航迹规划；蝙蝠算法；自适应权重系数；ε-level 比较策略；Dubins 平滑
中图分类号：V249.122+.3; TP301.6
文献标志码：A **文章编号**：1001-5965(2024)05-1593-11

在军事和民用领域，无人机以其低风险和高效率都在发挥更加重要的作用[1]。无论是执行运输、侦查还是打击任务，航迹规划都是无人机执行任务的关键组成部分，而全局航迹是无人机航迹规划任务的基础。在大范围复杂地形环境、存在防空火力威胁的环境以及受无人机性能约束的前提下，规划出一条从起始点到目标点的安全、高效三维航迹仍是无人机航迹规划的难点。

航迹规划是一个 NP-hard 问题，需要在有限的计算能力和较优的航迹之间寻求平衡，群智能优化算法内在的并行性和随机性特点使其在这种平衡中展现了优秀的寻优能力[2]，一系列群智能优化算法已经证明了可用于处理复杂多约束航迹规划问题。例如 Liu 等[3]将蚁群优化 (ant colony optimization, ACO) 算法进行改进提高其处理多约束问题的开发能力和收敛速度并将其用于无人机路径规划。陈侠等[4]将快速搜索随机树 (rapidly-exploring random tree, RRT) 和人工势场 (artificial potential field, APF) 算法相融合生成遗传算法 (genetic algorithm, GA) 的初始种群并将其用于无人机航迹规划。赵红超等[5]采用量子粒子群优化 (quantum particle swarm optimization, QPSO) 算法改进粒子的位置更新方式以提高算法在无人机路径规划问题的全局搜索能力。Shivgan 等[6]将无人机路径规划问题建模为旅行商问题并采用遗传算法进行优化以减少燃料消耗。汤安迪等[7]在鲸鱼优化算法 (whale optimization algorithm, WOA) 中引入混沌算子来提高初始种群的多样性以增强算法的开发能力和探索能力并将其用于无人机路径规划问题。Ghambari 等[8]将差分进化 (differential evolution, DE) 算法与 A*算法结合提高算法的探索能力并用于产生多条无人机子路径。高阳阳等[9]将轮盘赌和动态变异率引入传统共生生物搜索 (symbiotic organisms search, SOS) 算法中以提高算法的收敛速度和精度并将其用于无人机的空战机动决策。Jamshidi 等[10]采用改进的平行灰狼优化 (gray wolf optimization, GWO) 算法以降低实时无人机航迹规划的计算时间。以上算法显著提高了无人机航迹规划的能力，但是在复杂

收稿日期：2022-06-20；录用日期：2022-08-12；网络出版时间：2023-01-18 16:45
网络出版地址：link.cnki.net/urlid/11.2625.V.20230118.0946.001
*通信作者. E-mail：mengxy@bit.edu.cn

引用格式：王福仪，孟秀云，张海阔. 基于ε-level 蝙蝠算法的无人机三维航迹规划 [J]. 北京航空航天大学学报，2024，50（5）：1593-1603.
WANG F Y，MENG X Y，ZHANG H K. UAV three-dimensional path planning based on ε-level bat algorithm [J]. Journal of Beijing University of Aeronautics and Astronautics，2024，50（5）：1593-1603（in Chinese）.

地形环境、存在禁飞区约束的条件下，算法的收敛速度慢、开发能力偏弱、早熟现象能问题仍然存在。

蝙蝠算法(bat algorithm, BA)是由Yang[11]于2010年提出的一种元启发式优化算法，该算法的灵感来自于蝙蝠的回声定位机制，通过模拟蝙蝠利用超声波对猎物进行探索和定位，根据与食物的距离改变发射率和响度以调整算法的探索能力和开发能力。由于BA算法的原理简单，参数少，兼顾全局探索能力和局部开发能力，寻优效率高，自提出以来，该算法引起了广泛的关注。但是BA算法在处理高维约束问题时还是存在早熟、开发能力较弱等问题，为了提高算法的性能，许多研究者对其提出了一系列改进措施[12-15]并将其应用于工程优化问题[16]、旅行商问题[17]、控制器设计[18]和结构优化[19]等。在航迹规划领域，Wang等[20]将DE算法的变异策略引入到BA算法中，并将其用于三维航迹规划中，但是作者用BA算法进行航迹规划时没有对无人机的性能进行约束，同时航迹规划的地形环境较为简单。本文将BA算法应用于大范围、复杂地形环境，同时无人机性能受约束的三维全局无人机航迹规划问题。

本文将无人机三维航迹规划问题转化为高维带约束的连续优化问题；针对BA算法存在的早熟和局部开发能力较弱的问题，对BA算法的速度更新策略设计了自适应权重系数以平衡算法的前期探索和后期开发性能，引入迭代阈值来改善算法的早熟问题，并结合ε-level比较策略提高算法对约束优化问题的处理能力；同时分别设计威胁代价函数和转弯半径可变的Dubins曲线来解决2个连续的航迹点间贯穿威胁和地形的情况。

1 UAV航迹规划模型

1.1 问题描述

本文三维航迹规划的假设是飞行环境已经确定，并且已知所有的障碍物和威胁，根据目标函数和约束条件，规划出起始点到目标点的初始安全可飞行航迹。将原点O放在任务空间中某一点，建立初始三维惯性坐标系S_g-$Oxyz$，三维任务空间可表示为

$$\Omega=\{(x,y,z)\mid 0\leqslant x\leqslant X_{\max},0\leqslant y\leqslant Y_{\max},0\leqslant z\leqslant Z_{\max}\} \tag{1}$$

式中：x、y和z分别为任务空间中任意一点的横坐标、纵坐标和高度坐标；$X_{\max}$、$Y_{\max}$和$Z_{\max}$为任务空间的范围。

设无人机航迹$\boldsymbol{P}_{\mathrm{UAV}}=\{S,P_1,P_2,\cdots,P_N,T\}$且$P_i=(x_i,y_i,z_i)$。其中$S$为起始点；$T$为目标点；$P_i$为无人机第$i$个航路点；$N$为航路点个数，其取值根据任务空间信息、无人机起止点坐标、无人机性能和优化算法等综合确定；x_i、y_i和z_i分别为第i个航路点的横坐标、纵坐标以及高度坐标。则该航迹规划问题可由$3N$维向量$\{x_1,x_2,\cdots,x_N,y_1,y_2,\cdots,y_N,z_1,z_2,\cdots,z_N\}$组成，当航路点个数较多时，问题维度增加会使算法的计算时间急剧增加。

为了减小三维全局航迹规划问题的维度以降低问题的求解难度，建立以起始点到目标点在xOy平面的投影为横坐标轴的坐标系S_r-$O_rx_ry_rz_r$，如图1所示。坐标系S_r-$O_rx_ry_rz_r$通过式(2)绕z轴顺时针转θ角，并根据起始点平移后可得到惯性坐标系S_g-$Oxyz$：

$$\begin{bmatrix}x\\y\\z\end{bmatrix}=\begin{bmatrix}\cos\theta&-\sin\theta&0\\\sin\theta&\cos\theta&0\\0&0&1\end{bmatrix}\begin{bmatrix}x^{\mathrm{r}}\\y^{\mathrm{r}}\\z^{\mathrm{r}}\end{bmatrix}+\begin{bmatrix}x_{\mathrm{s}}\\y_{\mathrm{s}}\\0\end{bmatrix} \tag{2}$$

式中：$(x_{\mathrm{s}},y_{\mathrm{s}})$为起始点$S$的横、纵坐标；$(x^{\mathrm{r}},y^{\mathrm{r}},z^{\mathrm{r}})$为航路点在$S_r$-$O_rx_ry_rz_r$坐标系的横坐标、纵坐标和高度坐标，控制节点为$\{w_1,w_2,\cdots,w_N\}$且$w_k=\left(x_k^{\mathrm{r}},y_k^{\mathrm{r}},z_k^{\mathrm{r}}\right)$；$\theta$为方向角。

如图1所示，首先将起始点与目标点的连线ST以$\Delta l=\|ST\|/(N+1)$平均分成$(N+1)$等份，则每个控制节点的横坐标可表示为

$$x_k^{\mathrm{r}}=k\Delta l\qquad k=1,2,\cdots,N \tag{3}$$

式中：Δl为等分后的长度。

因此，UAV的航迹可转换为$2N$维向量，即$\left\{y_1^{\mathrm{r}},y_2^{\mathrm{r}},\cdots,y_N^{\mathrm{r}},z_1^{\mathrm{r}},z_2^{\mathrm{r}},\cdots,z_N^{\mathrm{r}}\right\}$，经过坐标转换后可得到所需航迹，从而将航迹规划问题转化为高维度带约束的连续优化问题。

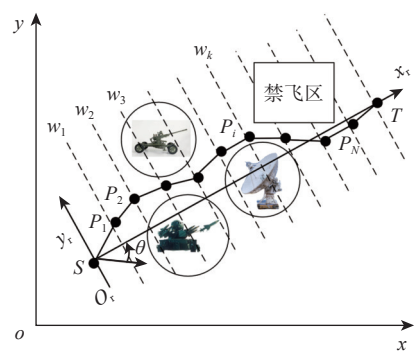


图1 坐标转换

Fig. 1 Coordinate conversion

1.2 目标函数和约束条件设计

三维全局航迹规划的目标函数主要由3部分组成：

$$\begin{cases}\min J=\sum_{k=1}^{3}\lambda_kJ_k\\ \text{s.t.}\quad g_i\leqslant 0\qquad i=1,2,3,4\\ \qquad\ h=0\end{cases} \tag{4}$$

式中: J_1为燃油代价, J_2为威胁代价, J_3为飞行高度代价; λ_k为代价权重系数且$\lambda_1+\lambda_2+\lambda_3=1$, 其目的主要是在无人机的安全和性能消耗等方面进行平衡, 可根据不同的任务需求进行设置; g为不等式约束; h为等式约束。

1) 燃油代价

假设无人机在任务空间内以恒定速率飞行, 则燃油代价可以通过航迹的欧氏距离进行等价表示:

$$J_1=\sum_{j=0}^{N}\left\|\left(x_{j+1},y_{j+1},z_{j+1}\right)-\left(x_j,y_j,z_j\right)\right\|_2 \tag{5}$$

式中: x_j、y_j和z_j分别为第j个航迹点的横坐标、纵坐标和飞行高度坐标。

2) 威胁代价

为了减少群优化算法的计算时间, 需要规划的航路点个数尽可能少, 但是在复杂地形环境中, 虽然规划的各个中间航路点满足地形和约束条件, 但是存在两个连续的航路点连线贯穿地形或威胁的情况。为了保证规划的航路点和航路点的连线避开威胁区域, 本文将其作为一种代价加入目标函数中, 贯穿地形的情况将在航迹点平滑策略中处理。航迹的威胁代价可通过式(6)计算:

$$J_2=\sum_{i=0}^{N}\sum_{j=1}^{M}T_j \tag{6}$$

式中: T_j为第j个威胁相对于全局航迹的总威胁代价; M为威胁个数。利用平均策略对 2 个连续航迹点间的连线进行m等分, 如图 2 所示, 第j个威胁的代价可通过式(7)计算:

$$\begin{cases}T_j=\dfrac{L_i}{m}\displaystyle\sum_{k=1}^{m}\Delta T_{j,k}\\ \Delta T_{j,k}=\begin{cases}0 & d_{k/m,j}>R_j\\ R_j^4/\left(d_{k/m,j}^4+R_j^4\right) & \text{其他}\end{cases}\end{cases} \tag{7}$$

式中: L_i为航迹段的欧氏距离; $\Delta T_{j,k}$为第j个威胁对于第k个航路点的威胁代价; R_j为第j个威胁的威胁半径; $d_{k/m,j}$为第k/m个点到第j个威胁的欧氏距离, 综合考虑航路点个数和任务空间范围, 本文将m设置为 5。

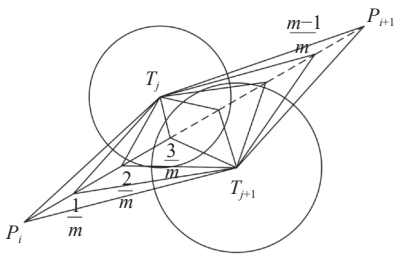


图 2　威胁代价计算

Fig. 2　Calculation of threat cost

3) 飞行高度代价

较低的飞行高度有利于 UAV 利用地形进行隐藏, 降低被雷达发现的概率, 因此本文将飞行高度作为其中一种飞行代价, 同时为了保证其与其他代价函数在同一数量级[21], 将任务空间横纵坐标范围的欧氏距离与高度范围的比值作为高度代价的系数, 如式 (8) 和式 (9) 所示:

$$J_3=\sum_{j=0}^{N+1}z_j\cdot Z_f \tag{8}$$

$$Z_f=\sqrt{X_{\max}^2+Y_{\max}^2}\Big/Z_{\max} \tag{9}$$

4) 爬升/下滑坡度约束

受无人机性能的限制, 规划的航迹在爬升或下滑时的斜率$\boldsymbol{s}$受到最大爬升坡度$\boldsymbol{\alpha}$和最小下滑坡度$\boldsymbol{\beta}$的约束:

$$g_1=\max\left(s_j-\alpha_j\right)\leqslant 0 \tag{10}$$

$$g_2=\max\left(\beta_j-s_j\right)\leqslant 0 \tag{11}$$

式中: s_j为第j个航迹点的斜率; α_j和β_j分别为第j个航迹的最大爬升坡度和最小下滑坡度, 其值由高度z_j确定。

5) 最大转向角约束

为了避免急转弯, 考虑到 UAV 机身性能的限制, 需要在每个航迹点处对其转向角进行限制, 最大转向角约束可表示为

$$g_3=\max\left(\Delta\gamma_i-\gamma_{\max}\right)\leqslant 0 \tag{12}$$

式中: 转向角$\Delta\gamma_i=\gamma_{i+1}-\gamma_i$; $\gamma_{\max}$为 UAV 的最大转向角。γ_i按式 (13) 计算:

$$\gamma_i=\operatorname{anctan}(\Delta y/\Delta x) \tag{13}$$

式中: $\Delta x=x_{j+1}-x_j$, $\Delta y=y_{j+1}-y_j$。

6) 飞行高度约束

考虑到飞行的安全性, UAV 应在预先设定的安全高度内飞行, 即

$$g_4=H_{\mathrm{s}}-\min\left(z_i-H\left(x_i,y_i\right)\right)\leqslant 0 \tag{14}$$

式中: $H(x_i,y_i)$为第 i 个航迹点的地形高度; H_{s}为最低安全飞行高度。

7) 禁飞区约束

在 UAV 执行任务过程中, 需要避免进入人为定义的禁飞区域 (no-fly zone, NFZ), 例如高危区域、气候恶劣区域、未知区域以及高层建筑等。本文设置长方形区域$\left\{x_{\mathrm{low}},x_{\mathrm{up}},y_{\mathrm{low}},y_{\mathrm{up}}\right\}$来表示禁飞区域, 同时为了提高该约束的计算精度, 利用图 2 所示的平均计算策略进行处理, 等式约束h如式 (15) 所示:

$$h=\sum_{i=0}^{N}\sum_{j=1}^{K}F_j=0 \tag{15}$$

$$\begin{cases} F_j = \dfrac{1}{m}\displaystyle\sum_{k=1}^{m} F_{j,k} \\ F_{j,k} = \begin{cases} 1 & x_{\mathrm{low},j} \leqslant x_{k/m,i,k} \leqslant x_{\mathrm{up},j} \\ & y_{\mathrm{low},j} \leqslant y_{k/m,i,k} \leqslant y_{\mathrm{up},j} \\ 0 & \text{其他} \end{cases} \end{cases} \tag{16}$$

式中：K为禁飞区域个数；F_j为第j个禁飞区域的平均威胁程度；m为2个相邻航路点间的平均分段数；x_{up}和x_{low}分别为禁飞区域的横坐标上、下界；y_{up}和y_{low}分别为禁飞区域的纵坐标上、下界；$\left(x_{k/m,i,k}, y_{k/m,i,k}\right)$为第$i$个航路点所在航迹段$P_iP_{i+1}$被$m$等分后的第$k$个小航迹段。

2 蝙蝠算法

BA算法的灵感来源于蝙蝠通过回声定位搜寻食物和规避威胁[11]，自然界中的蝙蝠大多可以向周围环境发射超声波并通过回声来捕食和导航。BA算法的迭代分为全局搜索部分和局部搜索部分，过程中通过调整频率$\boldsymbol{f}$、响度$\boldsymbol{A}$和发射率$\boldsymbol{r}$来更新每个蝙蝠的位置。

全局搜索部分，D维空间中蝙蝠的速度$\boldsymbol{v}_k$和位置$\boldsymbol{x}_k$更新策略如式(17)～式(19)所示：

$$f_k = f_{\min} + (f_{\max} - f_{\min})\beta_1 \tag{17}$$

$$v_k^t = v_k^{t-1} + \left(x_k^{t-1} - x^*\right) f_k \tag{18}$$

$$x_k^t = x_k^{t-1} + v_k^t \tag{19}$$

式中，β为[0,1]之间服从均匀分布的随机变量；t为当前迭代次数；$f_{\min}$和$f_{\max}$分别为频率的最小值和最大值，根据搜索空间尺寸调整取值，如果不同维度的搜索范围不同，其相应的频率取值范围也不同，本文中$f_{\min}=0$，$f_{x\max}=f_{y\max}=0.001$，$f_{z\max}=0.2$；$x^*$为根据式(4)计算得到的当前种群的最优个体。

局部搜索部分，当产生的随机值小于发射率r_k时，通过随机游走在由目标函数确定的当前最优解周围产生新的蝙蝠个体x_{new}，如式(20)所示：

$$x_{\mathrm{new}} = x^* + \mu A^t \tag{20}$$

式中：μ为[−1,1]之间均匀分布的随机数；$A^t = \langle A_k^t \rangle$为当前所有蝙蝠的平均响度，其取值跟任务空间尺寸相关，用于控制随机产生粒子距离最优解的步长。此外，发射率r_k和平均响度A^t根据迭代次数更新。

$$\begin{cases} A_k^{t+1} = \alpha A_k^t \\ r_k^{t+1} = r_k^0(1 - \exp(-\gamma t)) \end{cases} \tag{21}$$

其中：α和γ设置为常值。由式(21)可知，当蝙蝠接近食物源时，会通过增大发射率r_k并减小超声波响度A_k来寻找食物，本文中$A_x^0 = A_y^0 = 100\,000$，$A_z^0 = 300$，$r^0 = 0.5$，$\alpha = 0.9$，$\gamma = 0.3$。

从BA算法的更新过程可以看出，其全局搜索过程的个体更新策略类似于粒子优化群(particle swarm optimization, PSO)算法，同时采用贪心策略决定是否采纳新的个体；局部搜索过程更新策略类似于模拟退火算法。与其他算法相比，BA算法兼顾探索能力和开发能力，搜索效率高。

3 ε-level比较策略改进蝙蝠算法

BA算法在处理约束优化问题方面具有优越性，然而利用该算法求解航迹规划问题仍存在以下缺点：①BA算法处理高维优化问题时存在早熟问题；②BA算法处理约束优化问题的效率不高。本文提出如下基于ε-level比较策略的改进BA算法(ε-level improved bat algorithm, ε-IBA)：①设计自适应权重系数以平衡算法的探索能力和开发能力；②在局部搜索过程中针对当前最优蝙蝠个体设计了迭代阈值，以保证种群的多样性，提高算法的探索能力防止早熟；③结合ε-level比较策略，在BA算法中引入约束违反度函数以更好的求解约束优化问题。

3.1 改进BA算法

在BA算法的速度更新策略和局部搜索策略中，都利用了当前种群最优解信息，使种群会向当前最优解收敛，算法存在早熟问题。为了提高算法的探索能力，本文提出了2项改进措施设计改进蝙蝠算法。

3.1.1 自适应权重系数

BA算法的速度和位置更新策略类似于PSO算法，而PSO算法的速度更新策略包含3部分：速度项，个体历史最优解项和全局历史最优解项，PSO算法通过调整速度项的权重来调节算法的探索能力和开发能力。因此本文设计自适应权重系数并将其引入BA算法的速度更新策略来提高种群的多样性，平衡算法的全局和局部优化能力，权重系数如式(22)所示：

$$w_k^t = w_l^t - \delta h_k^t + \beta_2 s^t \tag{22}$$

式中：自适应权重系数w_i^t主要由线性因子w_l^t、进化因子h_i^t和聚合因子s^t[22]三部分确定，且$0 \leqslant w_i^t \leqslant 1$，$\delta = 0.3$，$\beta_2 = 0.4$。

线性因子由式(23)确定：

$$w_l^t = \frac{t_{\max} - t}{t_{\max}}(w_{\max} - w_{\min}) + w_{\min} \tag{23}$$

式中：$t_{\max}$为最大迭代次数；$w_{\max}$和$w_{\min}$为权重系数的最大值和最小值，$w_{\max}=1$，$w_{\min}=0.2$。随着迭代次数增加，w_l^t线性减小，速度项的影响逐渐减小，算

法由全局探索逐渐转为局部开发。

每个蝙蝠个体的进化因子h_i^t由式 (24) 确定:

$$h_i^t = 1 - \frac{\min(f(x_{\text{best},i}^{t-1}), f(x_{\text{best},i}^t))}{\max(f(x_{\text{best},i}^{t-1}), f(x_{\text{best},i}^t))} \tag{24}$$

式中: $0 \leqslant h_i^t \leqslant 1$, $f\left(x_{\text{best},i}^t\right)$为第 i 个个体的历史最优适应度函数值。h_i^t=0 时, 表明蝙蝠个体适应度值没有变小, 需要增大个体权重系数增强蝙蝠个体的探索能力; 反之h_i^t越大, 需要减小权重系数增强个体的开发能力。整个种群的聚合因子由式 (25) 确定:

$$s^t = f_{t\text{Best}} \big/ \overline{f_t} \tag{25}$$

其中: $0 \leqslant s^t \leqslant 1$; $f_{t\text{Best}}$为当前种群最优个体的适应度值; $\overline{f_t}$为当前种群所有个体的平均适应度值。s^t 越大, 说明种群的聚集度越高, 需要增大权重系数增强种群多样性, 反之需减小权重系数增强算法的局部优化能力。

综上, 随着算法迭代进行, 线性因子线逐渐减小, 并且通过个体进化因子和整个种群聚合因子自适应调节速度权重系数, 进而调节速度大小, 既保证了种群的多样性, 同时提高算法的开发能力。改进后的速度更新公式为

$$v_k^t = w_k^t v_k^{t-1} + \left(x_k^{t-1} - x^*\right) f_k \tag{26}$$

3.1.2　局部最优解迭代次数阈值

在局部搜索中, 发射率随迭代次数变化曲线如图 3 所示, 当产生的随机值小于发射率r_i时, 蝙蝠个体会被当前最优蝙蝠个体周围产生的新蝙蝠个体替代, 并且随着迭代次数增加, 蝙蝠个体被替换的概率越来越大, 通过调整发射率, BA 算法增强了在当前最优解周围的局部开发能力, 但会使整个种群过早的收敛于局部最优解, 降低了算法对全局空间的探索能力。对此, 本文针对当前种群最优解设计了迭代次数阈值$I_{\max}$, 如果当前种群最优蝙蝠个体在连续的$I_{\max}$代内与其历史最优位置相比没有改进, 则将当前最优蝙蝠个体用随机产生的新的蝙蝠个体替换。针对局部搜索的迭代阈值更新操作既能够有效的防止算法早熟, 同时由于在当前最有个体被替换前局部最优解周围已经聚集了一群蝙蝠个体使其对局部开发的影响较小。迭代阈值根据发射率初值及其变化规律综合确定, 本文中$I_{\max} = 10$。

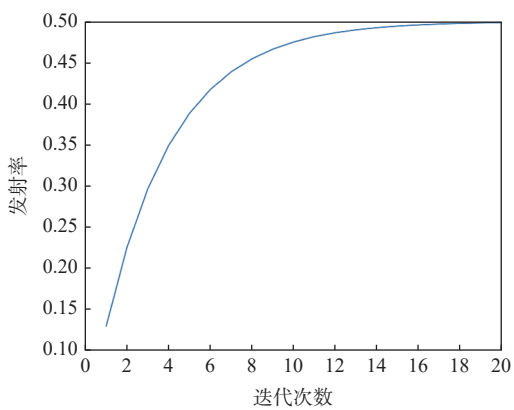


图 3　$r^0 = 0.5$, $\gamma = 0.3$时发射率随迭代次数变化曲线

Fig. 3　Emission rate changes with iterations when $r^0 = 0.5$, $\gamma = 0.3$

3.2　ε-level 比较策略

为了提高 BA 算法求解约束优化问题的效率, 本文采用 ε-level 比较策略将目标函数值与约束违反度相结合, 提高 BA 算法处理约束优化问题的效率。ε-level 比较策略是文献 [23] 针对约束优化问题提出的约束处理策略, 并成功应用于 PSO 算法[24] 和 DE 算法[25] 等。在不改变优化算法基本结构的前提下, 该策略能够有效的将约束优化问题转化为非约束优化问题, 具有广泛的适应性。一般情况下, 含有等式、不等式和上下边界的约束优化问题可以描述为

$$\begin{cases} \min f(x) \\ \text{s.t.} \quad g_k(x) \leqslant 0 \quad k = 1,2,\cdots,q \\ \qquad\quad h_j(x) = 0 \quad j = 1,2,\cdots,m \\ \qquad\quad x_{\text{low}_i} \leqslant x_i \leqslant x_{\text{up}_i} \quad i = 1,2,\cdots,N \end{cases} \tag{27}$$

式中: $f(x)$为目标函数; x为N维空间的一个点; h_j为第 j 个等式约束; g_k为第 k个不等式约束; x_{up_i}和x_{low_i}为变量x_i的上下边界。

在 ε-level 比较策略中, 约束违反度$\phi(x)$的定义为所有约束条件的和, 如式 (28) 所示:

$$\phi(x) = \sum_k \max\{0, g_k(x)\} + \sum_j \left|h_j(x)\right| \tag{28}$$

由约束违反度的定义可知, 如果某个点的约束违反度值大于 0, 则这个点为非可行解, 其价值较低。不同于其他约束处理策略, ε-level 比较策略在目标函数值与约束条件之间定义了次序关系, 约束违反度值的比较要优先于目标函数值。令$f_1(x)$、$f_2(x)$和$\phi_1(x)$、$\phi_2(x)$分别为x_1和x_2的目标函数值及其约束违反度, 则对于任意$\varepsilon > 0$, (f_1,ϕ_1)和(f_2,ϕ_2)之间的 ε-level 比较$\overset{\varepsilon}{<}$可定义为

$$(f_1,\phi_1) \overset{\varepsilon}{<} (f_2,\phi_2) \Leftrightarrow \begin{cases} f_1 < f_2 & \phi_1,\phi_2 < \varepsilon \\ f_1 < f_2 & \phi_1 = \phi_2 \\ \phi_1 < \phi_2 & \text{其他} \end{cases} \tag{29}$$

从式(29)可以看出, 只有当 2 个约束函数值均小于ε时, 比较结果才有目标函数值的大小决定。当$\varepsilon = \infty$时, 该比较策略等效为无约束的目标函数比较策略, 当$\varepsilon = 0$时, 该比较策略等效为完全满足约束条件的目标函数比较策略。ε一般通过式(30)确定:

$$\begin{cases}\varepsilon(0)=\phi(x_\vartheta)\\ \varepsilon(t)=\begin{cases}\varepsilon(0)(1-t/T_c)^\tau & 0<t<T_c\\ 0 & t\geqslant T_c\end{cases}\end{cases} \tag{30}$$

式中：初始值$\varepsilon(0)$为前ϑ个较优个体约束违反度且$\vartheta=0.2n$，n为种群规模；T_c为控制参数，一般设置为最大迭代次数的1/5；参数τ一般设置为5。式(30)表示在迭代次数小于T_c时，算法主要是寻找满足约束条件的可行解；而当迭代次数大于T_c时，则主要在可行解中寻找目标函数值较小的个体。

结合ε-level比较策略，改进后的ε-IBA算法的流程如算法1所示：

算法1 ε-level改进BA算法。

目标函数$f(x)$

初始化$x_k(k=1,2\cdots,n)$和v_k

定义个体x_k的频率f_k

初始化响度和发射率r_k

初始化权重系数$\omega_{\max},\omega_{\min}$和迭代阈值$I_{\max}$

While (t < Max number of generations)

For k from 1 to n

$w_k^t=w_l^t-\delta h_k^t+\beta_2 s^t$

$f_k=f_{\min}+(f_{\max}-f_{\min})\beta_1$

$v_k^t=\omega_k^t v_k^{t-1}+(x_k^{t-1}-x^*)f_k$

$x_{k\text{new}}^t=x_k^{t-1}+v_k^t$

If (rand<r_k)

$x_{k\text{new}}^t=x^*+\mu A^t$

End If

计算新个体的适应度值f_{new}和约束违反度值ϕ_{new}

If (rand < A_k&$(f_{\text{new}},\phi_{\text{new}})\overset{\varepsilon}{<}\left(f\left(x_k^{t-1}\right),\phi\left(x_k^{t-1}\right)\right)$)

$x_k^t=x_{k\text{new}}^t$

增大t_k，减小A_k

End If

If $((f_{\text{new}},\phi_{\text{new}})<(f(x^*),\phi(x^*)))$

$x^*=x_{k\text{new}}^t$

I=0

End If

End For

I=I+1

If ($I>I_{\max}$)

将当前最优个体用随机产生的新个体替换

End If

End While

4 转弯半径可变的Dubins航迹平滑方案设计

对于优化算法规划出的一系列控制节点，需要采用相应的策略对其进行平滑处理；同时对于大范围复杂地形航迹规划问题，尽管规划出的所有航迹点满足地形、威胁和无人机的性能约束，但是还是有可能出现2个航迹点的连线贯穿地形的情况出现，如图4所示。针对上述问题，主要有2种解决方案，第1种是以连续的2个控制节点为起始点和目标点对局部航迹进行重规划[26]，但重规划会增加算法计算时间；第2种方法是采用航迹平滑处理，常用的航迹平滑算法有贝塞尔曲线，B样条曲线，Dubins曲线等。与贝塞尔曲线和B样条曲线相比，Dubins曲线经过控制节点，并且曲线的长度只跟起始点和最终点转弯半径和速度方向有关[27]，因此本文采用Dubins曲线来平滑航迹，并通过调整转弯半径或者更改转弯方式来避免2个航迹点的连线贯穿地形的情况出现。

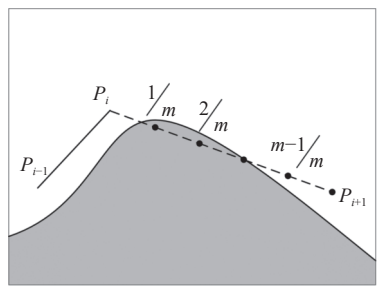


图4 航迹贯穿地形

Fig. 4 Track penetrates the terrain

4.1 Dubins曲线

Dubins曲线由首尾各是圆弧段中间是直线段连接(CSC)或者三段圆弧(CCC)连接而成，“C”代表了圆弧段，可以左转(L)也可以右转(R)，“S”代表了直线段。因此Dubins曲线有6种机动方式{LSL, LSR, RSL, RSR, RLR, LRL}。RSL类型的Dubins曲线如图5所示，$P_i(x_i,y_i,\theta_i)$和$P_f(x_f,y_f,\theta_f)$分别为起始点和最终点，x和y为坐标，θ表示相应的速度方向角。文献[27]通过微分几何的原理证明了只要起始点和最终点的转弯半径和速度方向确定，CSC机动方式的最短航迹由式(31)直接确定：

$$L=L_i+L_s+L_f=\alpha_i R_i+\sqrt{(\Delta X)^2+(\Delta Y)^2}+\alpha_f R_f \tag{31}$$

式中：L_i、L_s和L_f分别代表起始圆弧段、直线段和最终圆弧段长度；$\sqrt{(\Delta X)^2+(\Delta Y)^2}$表示起始圆弧段结束点和最终圆弧段开始点坐标间的欧氏距离；α_i和α_f分别为起始和目标圆弧角度；R_i和R_f分别为起始和目标半径。α_i通过式(32)和式(33)计算：

$$\begin{bmatrix}\cos\alpha_i\\ \sin\alpha_i\end{bmatrix}=\frac{1}{c}\begin{bmatrix}\sqrt{c^2-[(\pm R_f)-(\pm R_i)]} & -(\pm R_f)-(\pm R_i)\\ (\pm R_f)-(\pm R_i) & \sqrt{c^2-((\pm R_f)-(\pm R_i))}\end{bmatrix}\boldsymbol{t}_{ct} \tag{32}$$

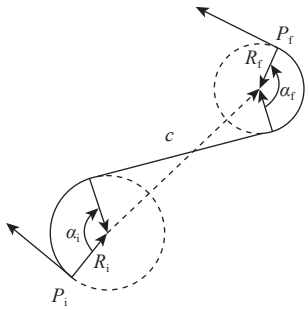


图 5　Dubins 曲线

Fig. 5　Dubins curve

$$\alpha_i = \tan^{-1}(\sin\alpha_i, \cos\alpha_i) \tag{33}$$

式中: c 为 2 个圆圆心之间的距离; $\boldsymbol{t}_{ct}$ 为两圆圆心方向的单位向量; R_i 和 R_f 前的正负号表示相应的机动方式, 正号为右拐, 负号为左拐。

目标圆弧段的角度即可通过式(34)计算:

$$\alpha_f = \theta_f - \theta_i - \alpha_i \tag{34}$$

求得初始圆弧段和最终圆弧段的角度之后, 即可求相应的切点坐标, 从而求得中间直线段的长度。分别求取 4 种机动方式的路径长度, 选择其中最短路径作为当前最优 Dubins 机动方式。

将 Dubins 曲线应用于三维航迹平滑策略中, 二维 Dubins 曲线起始点和最终点的速度向量位于同一个平面内, 但是在三维空间中, 起始点和最终点的速度方向通常情况下是异面的。

为了简化问题难度, 同时根据三维航迹的特点, 对于任意 2 个连续的航迹点 P 和 P_{i+1}, 规定点 P_{i+1} 的速度方向与向量 $\overrightarrow{P_iP_{i+1}}$ 同向, 这样起始点速度向量 $\overrightarrow{V_{p_i}}$、最终点速度向量 $\overrightarrow{V_{p_i+1}}$ 以及由起始点指向最终点的向量 $\overrightarrow{P_iP_{i+1}}$ 3 个向量将共面, 这样三维航迹平滑问题转化为空间中斜平面内二维 Dubins 最短航迹求解问题。

如图 6 所示, 建立由空间中斜平面为 xOy 平面的三维坐标系 S_d-$O_dx_dy_dz_d$, 以 P_i 为坐标原点 O, 以 $\overrightarrow{P_iP_{i+1}}$ 为 x 轴方向, 俯视斜平面沿 x 轴逆时针旋转 90°为 y 轴方向, z 轴由右手定则确定。求取初始点 P_i 在 S_d-$O_dx_dy_dz_d$ 坐标系中的速度方向 θ_i, 即可知起始点和最终点在 S_d-$Oxyz$ 坐标系中的坐标和速度方向角分别为 $P_i(0,0,\theta_i)$、$P_f\left(\overrightarrow{RP_{l+1}}|,0,0\right)$, 代入式 (31) 求得 Dubins 路径后通过相应的坐标转换再将路径转到 S_g-$Oxyz$ 坐标系中。

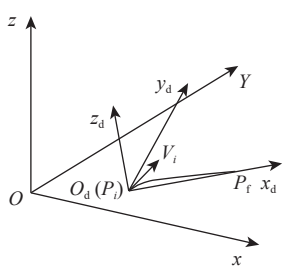


图 6　三维 Dubins 曲线

Fig. 6　Three -dimensional Dubins curve

4.2　转弯半径调整策略

为了解决两航迹点连线贯穿地形的问题, 本文设计了 Dubins 曲线转弯半径调整策略。受无人机性能的约束, 假设其最小转弯半径为 R_{min}, 同时为了保证 Dubins 曲线存在可行解, 必须保证式 (35) 成立, 在此基础上在铅垂面上对起始点的转弯半径采用 CSC 机动方式动态调整。

$$1 - \frac{1}{c^2}(\pm R_f - (\pm R_f))^2 > 0 \tag{35}$$

Dubins 航迹平滑策略的流程如图 7 所示。首先, 以最小转弯半径 $R_i = R_f = R_{min}$ 对航迹段进性平滑处理, 采用平均处理策略依次检测平滑后的所有 m 个点是否满足飞行高度约束, 若所有点满足, 则继续对下一个航迹段进行平滑处理, 若第 k 个点不满足, 则采用式 (36) 调整转弯半径 R_i 重新对航迹段进行平滑处理, 在保证 R_i 使 Dubins 曲线存在可行解的前提下重新检测所有点是否满足飞行高度约束, 若还不满足, 则选择次优的 Dubins 机动方式, 直至满足地形约束条件。

$$R_i = \frac{k}{m}(P_{i+1} - P_i) \tag{36}$$

平滑后的航迹曲线如图 8 所示。

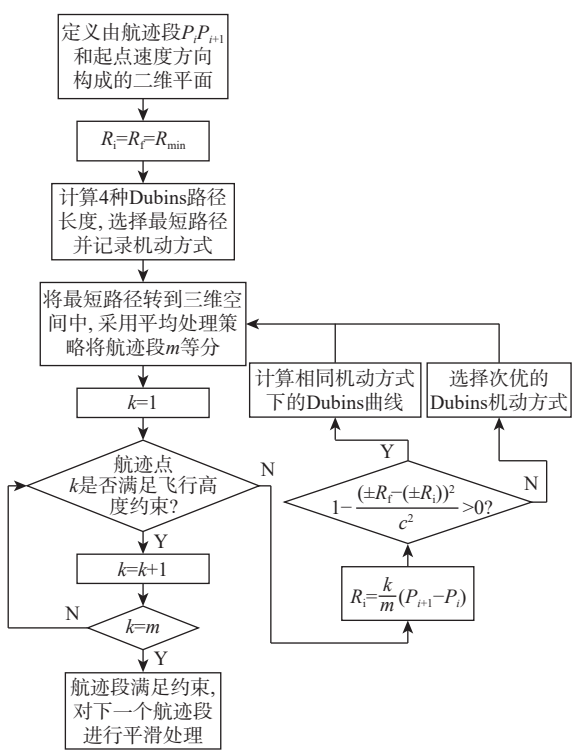


图 7　转弯半径可变的 Dubins 航迹平滑流程

Fig. 7　Dubins track smoothing process with variable turning radius

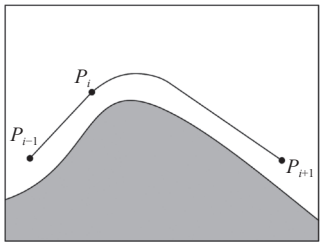


图 8 平滑后的航迹

Fig. 8 Track after smooth

5 实验评估与比较

在本节中，将评估ε-IBA算法与其他优化算法（BA算法、PSO算法、ε-PSO算法和ε-DE算法）相比在解决三维全局迹规划问题方面的性能。为了BA算法和PSO算法适用于求解约束优化问题，定义罚函数：

$$f_b(y,z)=J+10^6\left(\sum_{k=1}^{q}\max(0,g_k)+\sum_{j=1}^{m}|h_j|\right) \tag{37}$$

采用MATLAB进行仿真，运行环境为11th Gen Intel(R) Core(TM) i5-1135G7处理器，16 G内存。每种算法独立运行100次，设置最大迭代次数为2 000次，代价的权重因子$\lambda_1=1/3$，$\lambda_2=1/3$，$\lambda_3=1/3$，种群规模设置为$n=40$，问题维度$D=2N=50$。

各算法的其他参数设置如下：

1）ε-IBA：响度初值$A=0.5$，$r^0=0.5$，$\lambda=0.01$，$T_c=0.2$；

2）BA：$A=0.5$，$r^0=0.5$，$\lambda=0.01$；

3）ε-PSO：速度惯性权重初值和终值为$w_0=0.9$和$w_T=0.1$，个体学习因子$C_1=2$，社会学习因子$C_2=2$，$T_c=0.2$；

4）PSO：$w_0=0.9$，$w_T=0.1$，$C_1=2$，$C_2=2$；

5）ε-DE：缩放因子$F\in[0.6,0.8]$，交叉概率$p_{CR}\in[0.6,0.25]$，$T_c=0.2$。

UAV的相关性能参数设置如下：

1）飞行速度：$v=200$ m/s；

2）最大过载：$n_{max}=2$；

3）航路点：$N=25$；

4）最大转向角：$\varphi_{max}=\pi/4$；

5）安全飞行高度：$H_{max}=3\,000$ m，$H_{safe}=100$ m；

6）任务空间：1 000 km×1 000 km的方形空间。

起始点为（20,30）km、目标点为（900,800）km、各种类型威胁和禁飞区如表1所示。其中，导弹、高射炮和雷达信息为威胁圆心和半径，禁飞区信息分别为禁飞区x方向的下上边界和y方向的下上边界。

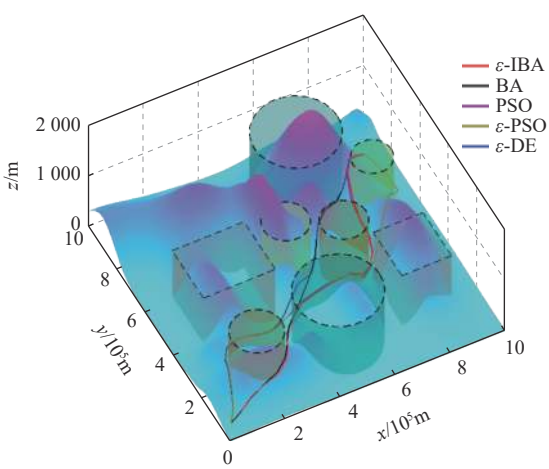
表 1 任务和威胁信息

Table 1 Task and threat information km

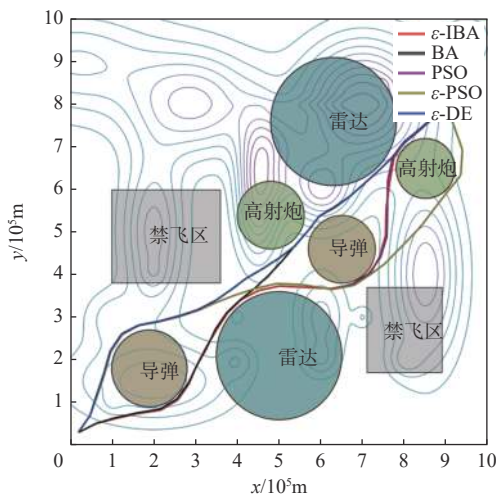
威胁类型	威胁范围
导弹	([190,180],90); ([650,460],80)
高射炮	([480,540],80); ([850,650],70)
雷达	([500,210],150); ([630,760],150)
禁飞区	([100,360],[380,600]); ([710,890],[170,370])

图9列出了每种算法独立运行100次，然后利用三维Dubins航迹平滑策略后得到的最优航迹，其中图9(a)为三维数字地形图下的航迹，图9(b)为相应等高线图下的二维航迹。通过100次的独立优化搜索，并结合Dubins航迹平滑策略，所有5种算法都能找到满足地形约束，同时规避威胁的安全飞行航迹。

表2列出了5种算法独立运行100次的性能统计结果，成功率表示100次运行中能够规划出满足约束、规避威胁的安全飞行路径的次数。从统计结



(a) 三维数字地形图下的航迹



(b) 相应等高线图下的二维航迹

图 9 不同算法得到的UAV航迹

Fig. 9 UAV track obtained by different algorithms

表 2　算法性能统计结果

Table 2　Algorithm performance statistics

算 法	最优/10^6	期望/10^6	最差/10^6	标准差/10^5	成功率/%
ε-IBA	**1.249**	**1.388**	**1.558**	**0.785 0**	**98**
BA	1.377	2.632	3.359	5.906	36
PSO	1.264	2.726	3.658	6.525	28
ε-PSO	1.260	1.523	1.869	1.583	64
ε-DE	1.412	1.755	2.039	2.329	54

果可以看出, 本文所提出的ε-IBA算法规划出的最优路径所需代价为 1.249×10^6, 比其余 4 种比较算法规划出的最优路径代价都小, 充分证明了ε-IBA算法具有较强的开发能力; 同时ε-IBA算法的期望、最差以及标准差路径代价等各项指标都远远优于其他算法, 通过对ε-IBA算法规划后的航路点成像进行分析, 100 次实验结果规划的所有离散航路点和采用 m 等分后的点都满足要求, 成功避开了威胁和禁飞区, 但是有两次实验结果中依然存在两个航路点连线间小部分经过威胁区域的情况, ε-IBA算法的规划成功率为 98%, ε-IBA算法在解决三维全局航迹规划方面有较好的稳定性。

图 10 列出了 5 种算法最优航迹的相应剖面图, L 为航迹路程。从高度剖面图中可以看出, 所有算法规划出的最优航迹在高度方向上都保持近地飞行, 充分说明了本文目标函数设计以及代价权重系数选取的合理性; 同时所有航迹两点间连线都没有贯穿地形的情况出现, 说明了通过变转弯半径的三维 Dubins 航迹平滑策略的有效性。

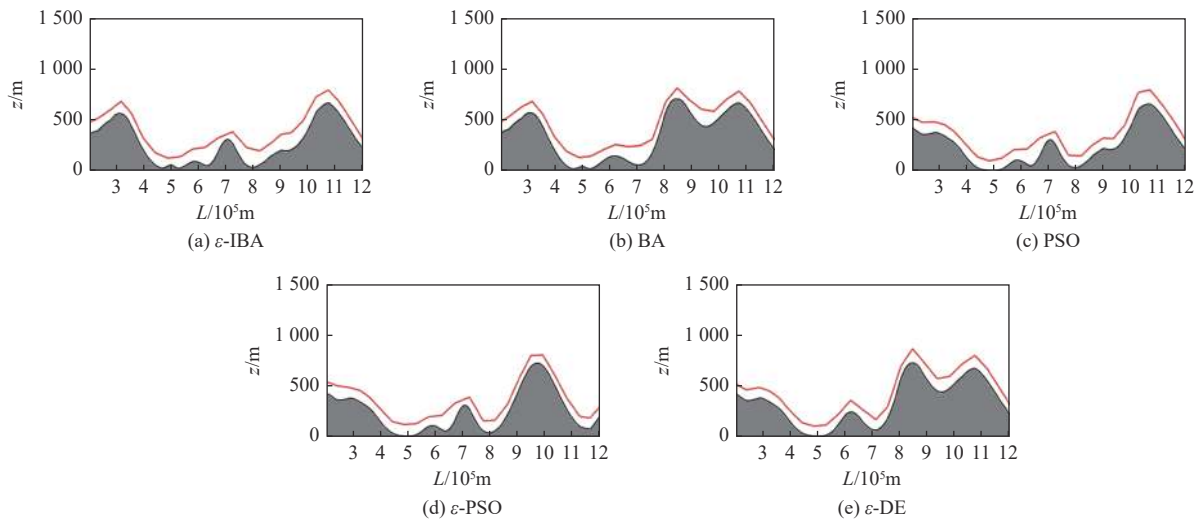


图 10　航迹剖面图

Fig. 10　Track profile

图 11 为 5 种算法求解航迹规划模型的最优航迹代价收敛曲线图。通过比较ε-IBA和 BA 算法、ε-PSO和 PSO 算法, 可以看出 ε-level 比较策略可以有效的加快算法的收敛速度; 同时改进后的 BA 算法开发能力更强, 精度更高。

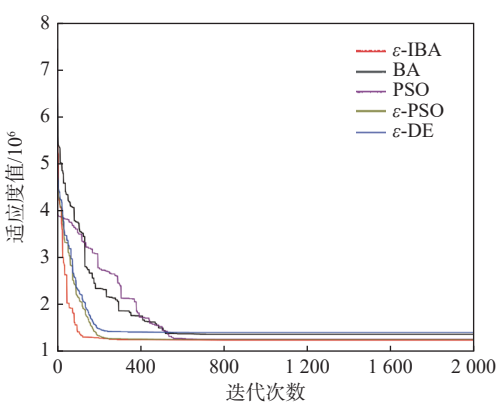


图 11　航迹代价收敛曲线

Fig. 11　Convergence curves of path cost

6　结　论

本文提出的ε-IBA算法能够较好的规划出满足任务要求的航迹, 同时与 BA、PSO、ε-PSO和ε-DE算法相比, ε-IBA算法在开发能力、稳定性和成功率等方面都表现出较好的性能。

下一步将结合天气、无人机的隐身性等不确定因素, 同时考虑动力学模型, 设计更加符合作战和控制要求的航迹规划模型; 同时多无人机协同作战已成为空战的主要作战方式, 作者将进一步研究无人机协同航迹规划和任务分配问题。

参考文献（References）

[1] SANTOSO F, GARRATT M, ANAVATTI S. State-of-the-art intelligent flight control systems in unmanned aerial vehicles[J]. IEEE Transactions on Automation Science and Engineering, 2018, 15(2):

613-627.

[2] ZHAO Y J, ZHENG Z, LIU Y. Survey on computational-intelligence-based UAV path planning[J]. Knowledge-Based Systems, 2018, 158: 54-64.

[3] LIU H, ZHANG N, LI Q. UAV path planning based on an improved ant colony algorithm[C]//Proceedings of 2021 4th International Conference on Intelligent Autonomous Systems (ICoIAS). Piscataway: IEEE Press, 2021: 357-360.

[4] 陈侠, 刘奎武, 毛海亮. 基于APF-RRT算法的无人机航迹规划[J]. 电光与控制, 2022, 29(5): 17-22.
CHEN X, LIU K W, MAO H L. The path planning of UAV based on APF-RRT algorithm[J]. Electronics Optics & Control, 2022, 29(5): 17-22(in Chinese).

[5] 赵红超, 周洪庆, 王书湖. 无人机三维航迹规划的量子粒子群优化算法[J]. 航天控制, 2021, 39(1): 40-45.
ZHAO H C, ZHOU H Q, WANG S H. Quantum particle swarm optimization algorithm of three-dimensional path planning of unmanned aerial vehicle[J]. Aerospace Control, 2021, 39(1): 40-45(in Chinese).

[6] SHIVGAN R, DONG Z. Energy-efficient drone coverage path planning using genetic algorithm[C]//Proceedings of 2020 IEEE 21st International Conference on High Performance Switching and Routing (HPSR). Piscataway: IEEE Press, 2020: 1-6.

[7] 汤安迪, 韩统, 徐登武, 等. 混沌多精英鲸鱼优化算法[J]. 北京航空航天大学学报, 2021, 47(7): 1481-1494.
TANG A D, HAN T, XU D W, et al. Chaotic multi-leader whale optimization algorithm[J]. Journal of Beijing University of Aeronautics and Astronautics, 2021, 47(7): 1481-1494(in Chinese).

[8] GHAMBARI S, LEPAGNOT J, JOURDAN L, et al. UAV path planning in the presence of static and dynamic obstacles[C]//Proceedings of 2020 IEEE Symposium Series on Computational Intelligence (SSCI). Piscataway: IEEE Press, 2020, 465-472.

[9] 高阳阳, 余敏建, 韩其松, 等. 基于改进共生生物搜索算法的空战机动决策[J]. 北京航空航天大学学报, 2019, 45(3): 429-436.
GAO Y Y, YU M J , HAN Q S, et al. Air combat maneuver decision-making based on improved symbiotic organisms search algorithm[J]. Journal of Beijing University of Aeronautics and Astronautics, 2019, 45(3): 429-436(in Chinese).

[10] JAMSHIDI V, NEKOUKAR V, REFAN M H. Real time UAV path planning by parallel grey wolf optimization with align coefficient on CAN bus[J]. Cluster Computing-The Journal of Networks Software Tools and Applications, 2021, 24(3): 2495-2509.

[11] YANG X. A new metaheuristic bat-inspired algorithm[C]//Proceedings of Nature Inspired Cooperative Strategies for Optimization (NICSO 2010). Berlin: Springer, 2010: 65-74.

[12] GANDOMI A, YANG X. Chaotic bat algorithm[J]. Journal of Computational Science, 2014, 5(2): 224-232.

[13] LIU Q, WU L, XIAO W S, et al. A novel hybrid bat algorithm for solving continuous optimization problems[J]. Applied Soft Computing, 2018, 73: 67-82.

[14] RAUF H, MALIK S, SHOAIB U, et al. Adaptive inertia weight bat algorithm with sugeno-function fuzzy search[J]. Applied Soft Computing, 2020, 90: 106159.

[15] LYU S L, LI Z, HUANG Y L, et al. Improved self-adaptive bat algorithm with step-control and mutation mechanisms[J]. Journal of Computational Science, 2019, 30: 65-78.

[16] YANG X, GANDOMI A. Bat algorithm: a novel approach for global engineering optimization[J]. Engineering Computations, 2012, 29(5): 464-483.

[17] OSABA E, YANG X, DIAZ F, et al. An improved discrete bat algorithm for symmetric and asymmetric traveling salesman problems[J]. Engineering Applications of Artificial Intelligence, 2016, 48: 59-71.

[18] TALBI N. Design of fuzzy controller rule base using bat algorithm[J]. Energy Procedia, 2019, 162: 241-250.

[19] HASANÇEBI O, TEKE T, PEKCAN O. A bat-inspired algorithm for structural optimization[J]. Computers & Structures, 2013, 128: 77-90.

[20] WANG G G, CHU H C, SEYEDALI M. Three-dimensional path planning for UCAV using an improved bat algorithm[J]. Aerospace Science and Technology, 2016, 49: 231-238.

[21] HOLUB J, FOO J, KILIVARAPU V, et al. Three dimensional multi-objective UAV path planning using digital pheromone particle swarm optimization[C]//Proceedings of 53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference. Reston: AIAA, 2012: 1525.

[22] DADGAR M, JAFARI S, HAMZEH A. A PSO-based multi-robot cooperation method for target searching in unknown environments[J]. Neurocomputing, 2016, 177: 62-74.

[23] TAKAHAMA T, SAKAI S. Constrained optimization by the ε constrained differential evolution with an archive and gradient-based mutation[C]//Proceedings of IEEE Congress on Evolutionary Computation. Piscataway: IEEE Press, 2010: 1-9.

[24] TAKAHAMA T, SAKAI S. Solving constrained optimization problems by the ε constrained particle swarm optimizer with adaptive velocity limit control[C]//Proceedings of IEEE Conference on Cybernetics & Intelligent Systems. Piscataway: IEEE Press, 2006: 1-7.

[25] WANG L, LI P. An effective differential evolution with level comparison for constrained engineering design[J]. Structural and Multidisciplinary Optimization, 2010, 41(6): 947-963.

[26] ALLAIRE FCJ, TARBOUCHI M, LABONTÉ G, et al. Real-time UAV path-terrain collision evaluation on FPGA[C]//Proceedings of 2018 4th International Conference on Optimization and Applications (ICOA). Piscataway: IEEE Press, 2018: 1-5.

[27] SHANMUGAVEL M, TSOURDOS A, ZBIKOWSKI R, et al. Path planning of multiple UAVs using dubins sets[C]//Proceedings of AIAA Guidance, Navigation, and Control Conference and Exhibit. Reston: AIAA, 2005: 5827.

UAV three-dimensional path planning based on ε-level bat algorithm

WANG Fuyi，MENG Xiuyun*，ZHANG Haikuo

(School of Aerospace Engineering，Beijing Institute of Technology，Beijing 100081，China)

Abstract: To address the problem of complex terrain environment and various threats and constraints, this article proposes a path planning algorithm for UAV based on ε-level improved bat algorithm. First, according to the drone target function and constraints, a three-dimensional path planning model of the UAV is established. Second, in response to the precocious phenomenon in handling the high-dimensional constraints problem of the bat algorithm, the adaptive weight coefficient and iteration threshold are designed to balance the exploration and exploitation capabilities of bat algorithms. Furthermore, by integrating an ε-level comparative strategy, the algorithm's capability to handle issues of non-convex and non-linear constraints is enhanced. Additionally, a three-dimensional Dubins curve with variable turning radius is designed to smooth the path and solve the problem of penetrating the terrain of the two trails. Through simulation experiments and compared with BA, PSO, ε-PSO and ε-DE, the algorithm proposed in this paper shows superior performance in terms of exploitation ability, stability and success rate.

Keywords: UAV path planning; bat algorithm; adaptive weight coefficient; ε-level comparison strategy; Dubins smoothing

Received：2022-06-20；**Accepted**：2022-08-12；**Published Online**：2023-01-18 16:45

URL：link.cnki.net/urlid/11.2625.V.20230118.0946.001

* **Corresponding author**. E-mail：mengxy@bit.edu.cn