引用格式:刘志阳,江涛,甄云卉. 基于改进ISPO算法的三维航迹规划方法[J]. 电光与控制,2018,25(7):48-53. LIU Z Y, JIANG T, ZHEN Y H. A 3D route planning method based on improved ISPO algorithm[J]. Electronics Optics & Control, 2018, 25(7):48-53.

基于改进ISPO算法的三维航迹规划方法

刘志阳[1], 江 涛[1], 甄云卉[2]
(1. 陆军工程大学石家庄校区,石家庄 050003; 2. 河北省军区,石家庄 050003)

摘 要: 提出了一种应用于无人机三维航迹规划的改进智能单粒子优化(ISPO)算法。把ISPO算法应用于航迹规划,并在此基础上引入子矢量之间的吸引效应,有效克服了算法易陷入局部最优解的缺陷。通过仿真实验,分别把改进ISPO算法、带动态惯性权系数的粒子群优化(PSO)算法及具备反向学习和局部学习的粒子群(RLPSO)算法应用于航迹规划。仿真结果表明,改进ISPO算法在航迹规划问题上有更强的寻优精度和能力,效率也高于另外两种算法。

关键词: 无人机; 航迹规划; 改进IPSO算法; 吸引效应
中图分类号: V279; TP242 **文献标志码**: A **doi**:10.3969/j.issn.1671-637X.2018.07.010

A 3D Route Planning Method Based on Improved ISPO Algorithm

LIU Zhi-yang[1], JIANG Tao[1], ZHEN Yun-hui[2]
(1. Shijiazhuang Campus, The Army Engineering University, Shijiazhuang 050003, China;
2. Hebei Military Area Command, Shijiazhuang 050003, China)

Abstract: An improved Intelligent Single Particle Optimizer (ISPO) algorithm is proposed for 3D route planning of Unmanned Aerial Vehicles (UAVs). The ISPO algorithm is applied to the route planning, and the attracting effect between sub-vectors is introduced to the improved algorithm, which effectively overcomes the shortcoming that the algorithm is easy to fall into the locally optimal solution. Simulation experiments are carried out, in which the improved ISPO algorithm, the Particle Swarm Optimizer (PSO) algorithm with dynamic inertia coefficient and the PSO with reverse-learning and local-learning capabilities (RLPSO) are applied to the route planning. The simulation results show that the improved ISPO algorithm has the best searching precision and ability in the route planning, and its efficiency is higher than that of the other two algorithms.

Key words: unmanned aerial vehicle; route planning; improved ISPO algorithm; attracting effect

0 引言

无人机航迹规划是任务规划的重要内容,可以有效地提升无人机的作战效能。因此,自20世纪80年代以来,世界各国都纷纷投入了大量的精力进行自动化航迹规划技术的研究,并取得了一定的成果[1-2]。

进化算法是一类基于群体的优化算法,粒子群优化(PSO)算法是其中的一种,具有实现容易、精度高、收敛快、不需要对很多参数进行调整等优点[3-4]。但PSO算法容易陷入局部最优,导致计算失败。针对这个问题,国内外学者提出了很多算法改进措施。SHI和EBERHART[5]提出将惯性权系数引入到PSO,并根据实验得出动态变化的惯性权系数比静态的惯性权系数性能优越的结论;文献[6]提出了具备反向学习和局部学习能力的粒子群算法(RLPSO),该算法利用反向学习和局部学习的交替作用避免算法陷入局部最优解,实验表明,在高维函数优化中该算法具有收敛速度快、求解精度高的特点;文献[7]提出了智能单粒子算法(ISPO),与传统的PSO算法不同,该算法在整个迭代过程中只有一个粒子参与计算,同时创新性地提出了子矢量的概念,有效地解决了高维度代价函数的优化问题。实验表明,此算法对大部分标准复合测试函数都具有很强的全局搜索能力。

本文在前人研究的基础上,把ISPO算法应用于无人机三维航迹规划,并在其基础上引入了航迹点(子矢

收稿日期:2017-06-20 修回日期:2017-08-21
作者简介:刘志阳(1993—),男,四川乐山人,硕士,研究方向为无人机协同作战航迹规划。

量)之间的吸引效应,就是把子矢量看作是具有物理特性的点,不同的子矢量之间会产生一个相互吸引的力。某个子矢量在某次更新中,全体子矢量会以该子矢量的移动步长为基础按照一定的比例移动相应的距离。这样的航迹更新过程从客观上符合实际航迹连续变化的规律,同时也增强了算法的全局搜索能力。通过 Matlab 仿真实验比较了改进 ISPO 算法与带动态变化惯性权系数的 PSO 算法和 RLPSO 算法的性能。仿真实验结果表明,改进 ISPO 算法能有效完成三维航迹规划的寻优计算,其规划结果优于带动态变化惯性权系数的 PSO 算法和 RLPSO 算法的规划结果。在后文中提到的 PSO 算法都是带动态变化惯性权系数的 PSO 算法。

1　航迹表示与航迹代价函数

1.1　航迹表示

在运用 PSO 算法和 RLPSO 算法进行航迹规划时,每一个粒子表示一条完整的航迹,整个种群包含多少粒子,在规划空间中就有多少条可能的航迹,粒子中的相应分量表示航迹点的对应坐标。而在基于改进 ISPO 算法的航迹规划方法中只有一个粒子,也就是只对一条航迹进行优化。为了客观地比较 3 种算法的优劣性,本文采用相同的航迹表示方式,同样的初始化方法,同样的航迹代价函数,同样的约束条件以及同样的任务背景(也就是相同的规划空间)。

在基于 PSO 算法和 RLPSO 算法的航迹规划方法中,种群可表示为一个矩阵

$$\boldsymbol{X}=(\boldsymbol{X}_1 \quad \boldsymbol{X}_2 \quad \cdots \quad \boldsymbol{X}_N)^{\mathrm{T}} \tag{1}$$

式中:$\boldsymbol{X}_i$ 为粒子群的第 i 个粒子($i=1,2,\cdots,N$),N 为粒子群中的粒子个数。每条航迹有相同的起点和终点,假设由起点到终点的航迹用 n 个航迹点表示,在三维空间中每个航迹点需要一个三维坐标(x,y,z)来表示其空间位置信息。所以除去航迹的起点和终点,每条航迹可由一个 $3n$ 维的行向量来表示,即

$$\boldsymbol{X}_i=(x_{i1} \ \cdots \ x_{in} \ \ y_{i1} \ \cdots \ y_{in} \ \ z_{i1} \ \cdots \ z_{in}) \tag{2}$$

式中:$\boldsymbol{X}_i$ 为粒子群的第 i 个粒子($i=1,2,\cdots,N$);x_{i1},$x_{i2},\cdots,x_{in}$和 $y_{i1},y_{i2},\cdots,y_{in}$分别表示 n 个航迹点的横坐标和纵坐标,$z_{i1},z_{i2},\cdots,z_{in}$表示 n 个航迹点的飞行高度。所以第 i 条航迹的第 j 个航迹点的坐标可表示为(x_{ij},y_{ij},z_{ij})($j=1,2,\cdots,n$)[8]。

在运用改进 ISPO 算法进行航迹规划时,其航迹表示为

$$\begin{cases}\boldsymbol{X}=(x_1\cdots x_n \ \ y_1\cdots y_n \ \ z_1\cdots z_n)\\ \boldsymbol{G}_j=(x_j \ \ y_j \ \ z_j) \quad j=1,2,\cdots,n\\ \boldsymbol{X}=(\boldsymbol{G}_1 \ \ \boldsymbol{G}_2 \ \cdots \boldsymbol{G}_n)\end{cases} \tag{3}$$

式中:$\boldsymbol{G}_j$ 表示 $\boldsymbol{X}$ 的第 j 个子矢量,同时也是第 j 个航迹点的坐标,与 PSO 算法中的表示方法相同。

1.2　规划空间与初始化

航迹规划问题的实质就是在规划空间中搜索最优或者可接受的航迹,规划空间就相当于航迹的值域,航迹的选取不能超出规划空间。也就是航迹点的坐标有一定的取值范围,即

$$\begin{cases}0\leqslant x_j\leqslant x_{\max}\\ 0\leqslant y_j\leqslant y_{\max} \qquad j=1,2,\cdots,n\\ z_{\min}+h\leqslant z_j\leqslant z_{\max}\end{cases} \tag{4}$$

式中:$x_{\max}$,$y_{\max}$分别表示战场的长和宽,$x_{\max}=y_{\max}=200$ km;h 表示航迹点所对应位置的高程;$z_{\min}$,$z_{\max}$分别表示无人机的最小离地高度和最大飞行高度,在本文的计算中所有长度单位均为千米。在规划空间中的威胁用一个球心位于地面的半椭球表示。图 1 为规划空间的示意图。

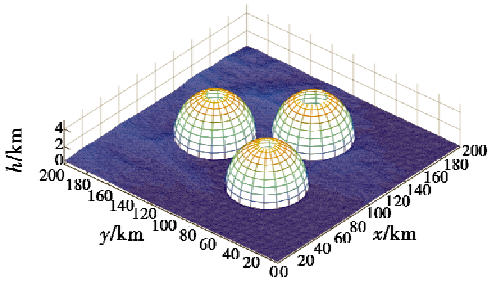

图 1　规划空间示意图

Fig. 1　Configuration space

在算法开始运行的时候,需要对种群进行初始化。本文采用的是一种基于航迹最短的近似随机初始化思路。在不考虑飞行高度的前提下,连接起始点和终点的直线是最短的航迹。在此原则下,以连接起点和终点的直线为公共对角线,均分出 n 个矩形,对应的第 j 个航迹点可能出现在第 j 个矩形内的任意位置,如图 2 所示。飞行高度的初始化均设为固定值,该固定值为无人机的最佳飞行高度。

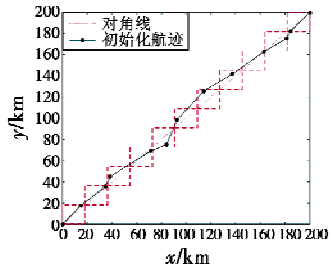

图 2　初始化航迹

Fig. 2　The initialization of the route

1.3　约束条件和航迹代价函数

受限于无人机自身的物理限制和使用要求,无人机在飞行过程中需要满足一定的航迹基本约束条件,

主要包括以下方面:最小航迹段长度、最大拐弯角、最大爬升/俯冲角、最大航迹长度、最低飞行高度、最高/最低飞行速度和目标进入方向[9]。

在本文的研究中,把约束条件划分为三类:时效性约束、安全性约束和机动性约束。

1) 时效性约束。包括最小航迹段长度、最大航迹长度和最高/最低飞行速度。在本文中假设整个规划过程中出现的航迹总长度始终在无人机的最大航程以内。飞行速度取在允许范围内的定值。在本文中时效性指标可表示为

$$L = k_L \sum_{i=1}^{n+1} \left(l_i - \frac{L_0}{n+1} \right)^2 \tag{5}$$

式中:L 表示总的时效性指标;k_L 为常数;n 为航迹点的个数;l_i 表示每条航迹段的长度;L_0 表示起点到终点的直线距离。

2) 安全性约束。安全性约束主要考虑最低飞行高度和环境对无人机的威胁。安全性指标为

$$F = \sum_{i=1}^{n+1} \int_{l_i} f \mathrm{d}s + F_{地} \quad f = \begin{cases} K_f / r^4 & \text{航迹点被覆盖} \\ 0 & \text{航迹点未被覆盖} \end{cases} \tag{6}$$

式中:F 表示整个规划空间中的威胁总强度,它由每一段航迹的威胁强度求和得到,每一段的威胁强度又是由每一点的威胁强度沿航迹段积分求得;每一点的威胁强度可由 f 求得,其中,K_f 为常数,表示威胁的强度,r 表示航迹段上的点距球心的距离,地形威胁 $F_{地}$ 的求取和防空威胁类似,可表示为

$$F_{地} = \sum_{i=1}^{n+1} \int_{l_i} f_{地} \mathrm{d}s \quad f_{地} = \begin{cases} K_{地} & z - z_{\min} < h \\ 0 & z - z_{\min} \geqslant h \end{cases} \tag{7}$$

式中:$F_{地}$ 表示整个规划空间中地形威胁的强度,正常情况下其值应该为 0;$f_{地}$ 表示航迹段中某一点的威胁强度;z 表示无人机在该点的飞行高度;当无人机在该点的飞行高度减去无人机的最小离地高度小于该点的高程时,地形威胁强度为 $K_{地}$,这里的 $K_{地}$ 为一个远大于航迹总代价的常数。

3) 机动性约束。机动性约束包括最大拐弯角、最大爬升/俯冲角和目标进入方向。在本文中假设无人机向任何方向机动的角度都不得超过 $\theta_{\max}$。由于本文的研究中目标为一个点目标,所以不用考虑目标进入方向。同时为了保证无人机尽量保持水平飞行,在机动性能约束中加入了航迹点之间的高度差限制。所以机动性指标可表示为

$$D = \sum_{j}^{n} [K_\theta \theta + K_z(|z_j - z_{j-1}|)] \quad \theta = \begin{cases} \theta_j^2 & \theta_j \leqslant \theta_{\max} \\ K_\theta \theta_j & \theta_j > \theta_{\max} \end{cases} \tag{8}$$

式中:D 表示总的机动性指标,它是通过每个航迹点的机动性指标求和得到;θ_j 为第 j 个航迹点所连接的两条航迹之间的角度;K_θ 和 K_z 均为常系数;z_j, z_{j-1} 分别为第 j 和第 $j-1$ 个航迹点的高度。

综上所述,航迹代价函数可由以上 3 个指标加权求和得到,即

$$J(\boldsymbol{X}) = w_1 L + w_2 F + w_3 D \tag{9}$$

式中:J 表示航迹代价;$\boldsymbol{X}$ 表示一条完整的航迹;w_1, w_2, w_3 为 3 个权系数,分别表示其所对应的指标的重要程度,在本文中假设它们同等重要,所以 $w_1 = w_2 = w_3 = 1/3$。通过分析可得,在没有威胁的情况下,如果起点和目标点处在同一水平高度,航迹代价最小可为 0,这时航迹为连接起点和终点的直线。

2 改进 ISPO 算法

2.1 算法原理

文献[8]中详细介绍了 PSO 算法的计算过程,其新速度和位置为

$$\boldsymbol{V}_i^{k+1} = \omega \boldsymbol{V}_i^k + c_1 r_1 (\boldsymbol{P}_i^k - \boldsymbol{X}_i^k) + c_2 r_2 (\boldsymbol{P}_g - \boldsymbol{X}_i^k) \tag{10}$$

$$\boldsymbol{X}_i^{k+1} = \boldsymbol{X}_i^k + \boldsymbol{V}_i^{k+1} \tag{11}$$

$$\omega = \omega_{\max} - \frac{\omega_{\max} - \omega_{\min}}{k_{\max}} \times k \tag{12}$$

式中:$i = 1, 2, \cdots, N$;k 是迭代次数;ω 为惯性权系数;$\omega_{\max}$ 为初始权重;$\omega_{\min}$ 为终止权重;$k_{\max}$ 为最大迭代次数;r_1, r_2 为[0,1]之间的随机数,这两个参数可以保证群体的多样性;c_1, c_2 为学习因子。

相较于 PSO 算法,RLPSO 算法增强了种群最优个体的局部搜索能力,同时使种群具有反向学习能力,保证了种群的多样性。

RLPSO 算法的局部搜索方法:同样是利用式(10)、式(11)进行算法迭代,只是式(10)中的 $\boldsymbol{P}_g$ 是由种群历史最优位置 $\boldsymbol{P}_{g1}$ 和历史次优位置 $\boldsymbol{P}_{g2}$ 的差分结果与 $\boldsymbol{P}_{g1}$ 比较得到。

$$\boldsymbol{P}'_{g1} = \boldsymbol{P}_{g1} + r_3 d_k (\boldsymbol{P}_{g1} + \boldsymbol{P}_{g2}) \tag{13}$$

$$d_{k+1} = d_k (1 - k/k_{\max}) \tag{14}$$

式中:r_3 为[-1,1]之间的随机数;k 为迭代次数;$k_{\max}$ 为最大迭代次数;$\boldsymbol{P}_g$ 通过比较 $\boldsymbol{P}'_{g1}$ 与 $\boldsymbol{P}_{g1}$ 的航迹代价的大小决定。第 i 个粒子的历史最差位置为 $\boldsymbol{w}_i = (w_1 \quad w_2 \quad \cdots \quad w_m)$。初始种群中满足排异距离的 m 个粒子构成初始最差粒子个体的位置集合 $\boldsymbol{W} = (W_1 \quad W_2 \quad \cdots \quad W_m)$[6]。第 i 个粒子反向学习的速度更新公式为

$$\boldsymbol{V}_i^{t+1} = \omega \boldsymbol{V}_i^t + c_3 r_4 (\boldsymbol{X}_i^t - \boldsymbol{w}_i^t) + c_4 r_5 (\boldsymbol{X}_i^t - \boldsymbol{W}) \tag{15}$$

式中:$i = 1, 2, \cdots, m$;t 是迭代次数;ω 为惯性权系数;r_4, r_5 为[0,1]之间的随机数;c_3, c_4 为学习因子。

RLPSO 算法的反向学习:当迭代一定的次数发现 $\boldsymbol{P}_g$ 没有更新,选取前 m 个粒子进行 s 次的反向学习过程,其他 $N-m$ 个粒子的迭代方式不变。

在 ISPO 算法中,整个迭代过程只有一个粒子参与

计算。该算法不是同时更新速度矢量或者位置矢量，而是把粒子的位置矢量分成若干子矢量，依次对每一个子矢量进行更新，更新完全部子矢量后再从第一个子矢量开始新一轮更新。在整个更新过程中，引入一种新的学习策略，使得算法可以分析之前的速度更新情况，使粒子在搜索空间中能够动态地调整速度和位置，从而向全局最优靠近。在更新第 $j(j=1,2,\cdots,n)$ 个子矢量时，按以下公式更新速度和位置。

$$\boldsymbol{V}_j^k=\frac{a}{k^p}\times\boldsymbol{r}+b\times\boldsymbol{L}_j^{k-1} \tag{16}$$

$$\boldsymbol{G}_j^k=\begin{cases}\boldsymbol{G}_j^{k-1}+\boldsymbol{V}_j^k & J(\boldsymbol{x}_1^k)>J(\boldsymbol{x}_2^k)\\ \boldsymbol{G}_j^{k-1} & J(\boldsymbol{x}_1^k)\leqslant J(\boldsymbol{x}_2^k)\end{cases} \tag{17}$$

$$\boldsymbol{L}_j^k=\begin{cases}\boldsymbol{L}_j^{k-1}/s & J(\boldsymbol{x}_1^k)\leqslant J(\boldsymbol{x}_2^k)\\ \boldsymbol{V}_j^k & J(\boldsymbol{x}_1^k)>J(\boldsymbol{x}_2^k)\end{cases} \tag{18}$$

式中：$\boldsymbol{V}_j^k=(v_{xj}^k\ \ v_{yj}^k\ \ v_{zj}^k)$，$j=1,2,\cdots,n$；$\boldsymbol{x}_1^k=(\boldsymbol{G}_1\cdots\boldsymbol{G}_j^{k-1}\cdots\boldsymbol{G}_n)$，$\boldsymbol{x}_2^k=(\boldsymbol{G}_1\cdots\boldsymbol{G}_j^{k-1}+\boldsymbol{V}_j^k\cdots\boldsymbol{G}_n)$，$k$ 表示迭代次数；L 为学习变量，可以分析之前迭代的速度更新状况，从而对下一次速度产生影响；$\boldsymbol{r}$ 为在 $[-0.5,0.5]$ 之间均匀分布的随机矢量，其维数与子矢量的维数相同；a 为多样性因子；p 为下降因子；s 为收缩因子；$b(b\geqslant 1)$ 为加速度因子；J 为航迹代价函数[10]。

2.2　算法改进

航迹规划问题是一种典型的高维度的优化问题，而 ISPO 算法通过划分子矢量的方法有效解决了传统的 PSO 算法不能兼顾所有维度优化方向的问题。在 ISPO 算法中，不会同时更新所有航迹点，而是轮流更新每一个航迹点。假设 $\boldsymbol{G}_j^k$ 和其前后两个航迹点的分布情况如图 3 所示，在迭代的过程中可出现 $\boldsymbol{V}_j^k$ 这样一个速度矢量，使得 $\boldsymbol{G}_j^k$ 更新到了 $\boldsymbol{G}_j^{k+1}$ 的位置，则可以通过计算得到更新后的代价优于之前的代价，所以接受更新后的结果为历史最优。

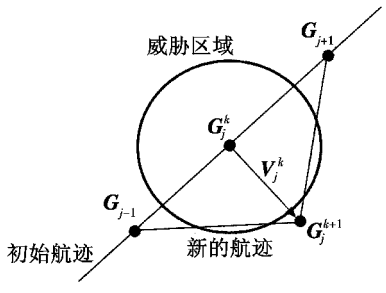

图 3　航迹更新示意图

Fig. 3　Route updating

为了提高航迹的精度，航迹点的个数不能过少。然而过多的航迹点将会导致每一次迭代对整体航迹的影响过小，从而导致规划算法不能逃脱局部最小解。因此本文在 ISPO 算法的基础上引入了航迹点之间的吸引效应，这样既保证了航迹的精度，也提高了算法的全局搜索能力。具体的算法迭代公式为

$$\boldsymbol{V}_j^k=\frac{a}{k^p}\times\boldsymbol{r}+b\times\boldsymbol{L}_j^{k-1} \tag{19}$$

$$\boldsymbol{V}_i^k=\boldsymbol{V}_j^k\times k_G^{|j-i|}\quad i=1,2,\cdots,n \tag{20}$$

$$\begin{cases}\boldsymbol{X}^k=\boldsymbol{X}^{k-1}+\boldsymbol{V}^k & J(\boldsymbol{x}_1^k)>J(\boldsymbol{x}_2^k)\\ \boldsymbol{X}^k=\boldsymbol{X}^{k-1} & J(\boldsymbol{x}_1^k)\leqslant J(\boldsymbol{x}_2^k)\end{cases} \tag{21}$$

式中：k_G 为常系数；$\boldsymbol{V}^k=(\boldsymbol{V}_1^k\ \ \boldsymbol{V}_2^k\cdots\boldsymbol{V}_n^k)^{\mathrm{T}}$，$k$ 为迭代次数；$\boldsymbol{x}_1^k=\boldsymbol{X}^{k-1}$，$\boldsymbol{x}_2^k=\boldsymbol{X}^{k-1}+\boldsymbol{V}^k$。其航迹更新如图 4 所示。

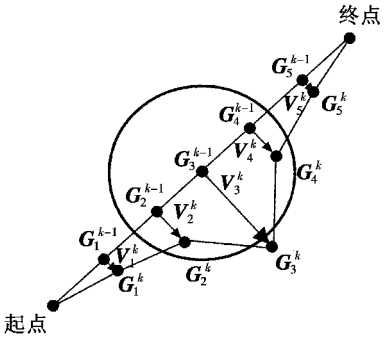

图 4　改进算法航迹更新示意图

Fig. 4　The route updating of the improved algorithm

3　实验结果与分析

3.1　实验数据与参数

实验所用的地图为 200 km × 200 km 一共 40 000 个点的数字高程地图。威胁覆盖范围用长轴和短轴分别为 30 km 和 5 km 的半椭球表示。

本实验所涉及的参数：无人机的最小离地高度为 0.05 km，最大飞行高度为 2 km，$\theta_{\max}$ 为 30°。在 PSO 算法中，最大惯性权系数 $\omega_{\max}$ 为 1.4，最小惯性权系数 $\omega_{\min}$ 为 0.8，学习因子 c_1 为 2，c_2 为 2；在 RLPSO 算法中，反向学习因子 $c_3=0.7$，$c_4=0.3$，反向学习代数 $s=10$，反向学习种群规模 $m=28$，扰动系数初始值 $d_0=40$，反向学习条件为种群最优解连续 10 代没有变化；在改进 ISPO 算法中，当确定多样性因子时，$\boldsymbol{A}$ 为(160　160　0.02)，下降因子 p 为1.55，收缩因子 s 为 4，加速度因子 b 为 2，k_G 取 0.3，飞行的起点坐标为(0,0,1)，终点坐标为(200,200,1)。

实验 1　综合考虑航迹计算精度与航迹表示精度，通过大量的实验，设定 PSO 算法和 RLPSO 算法的种群数量 N 为 40，航迹点为 10，最大迭代次数为 100。改进 ISPO 算法的航迹迭代次数 N_X 为 30，航迹点迭代次数为 10，航迹的表示同 PSO 算法。每种算法进行 50 次运算。实验数据如表 1 所示。航迹规划结果和航迹代价分别如图 5 和图 6 所示。

表1 实验1的3种算法的航迹代价比较

Table 1 Route costs of the three algorithms in Experiment 1

算法	最小值	最大值	均值	方差	极差
PSO	803.1	7722	2353	1753	6919
RLPSO	871.4	5882	1756	1296	5010
改进 ISPO	801.5	5038	1435	679	4236

在实验1中,PSO算法、RLPSO算法和改进ISPO算法的平均耗时分别为37.05 s,48.61 s,25.49 s。图5中的4条航迹表示4类可接受的航迹。

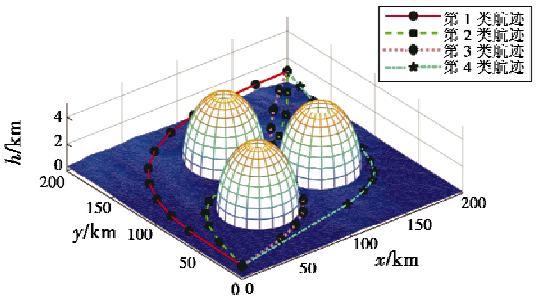

图5 实验1的航迹规划结果

Fig. 5 Route planning result of Experiment 1

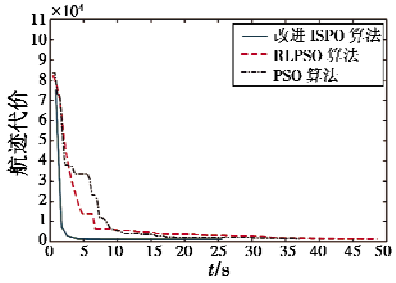

图6 实验1的航迹代价变化曲线

Fig. 6 Route cost of Experiment 1

在PSO算法中,计算50次,成功45次,失败5次。其中:第1类航迹34次,平均航迹代价1453;第2类航迹6次,平均航迹代价5231;第3类航迹3次,平均航迹代价4787;第4类航迹2次,平均航迹代价5374。

在RLPSO算法中,计算50次,成功50次。其中:第1类航迹45次,平均航迹代价1423;第2类航迹3次,平均航迹代价4306;第3类航迹1次,航迹代价5882;第4类航迹1次,航迹代价4971。

在改进ISPO算法中,计算50次,成功50次。其中:第1类航迹15次,平均航迹代价1152;第2类航迹12次,平均航迹代价1637;第3类航迹7次,平均航迹代价1670;第4类航迹16次,平均航迹代价1446。

实验2 选取更为复杂的规划空间进行实验,所涉及的参数不变。每种算法进行50次运算。实验数据如表2所示,航迹规划结果和航迹代价分别见图7和图8。

在实验2中,PSO算法、RLPSO算法和改进ISPO算法的平均耗时分别为62.49 s,83.71 s,43.25 s。图7中的4条航迹表示4类可接受的航迹。可以看出,第2和第4类航迹无人机机动幅度较大、航迹较长、因此航迹代价会高于第1和第3类航迹。

表2 实验2的3种算法的航迹代价比较

Table 2 Route costs of the three algorithms in Experiment 2

算法	最小值	最大值	均值	方差	极差
PSO	1135	8278	2898	2286	7143
RLPSO	987.6	7838	2397	1695	6850
改进 ISPO	890.5	6357	1630	1020	5466

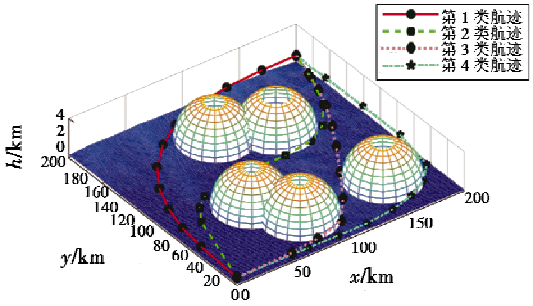

图7 实验2的航迹规划结果

Fig. 7 Route planning result of Experiment 2

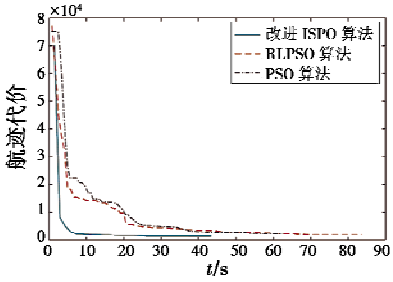

图8 实验2的航迹代价变化曲线

Fig. 8 Route cost of Experiment 2

在PSO算法中,计算50次,成功43次,失败7次。其中:第1类航迹38次,平均航迹代价2224;第2类航迹5次,平均航迹代价8023;没有第3和第4类航迹。

在RLPSO算法中,计算50次,成功50次。其中:第1类航迹45次,平均航迹代价2022;第2类航迹4次,平均航迹代价6459;没有第3类航迹;第4类航迹1次,航迹代价3013。

在改进ISPO算法中,计算50次,成功50次。其中:第1类航迹42次,平均航迹代价1499;第2类航迹4次,平均航迹代价2898;第3类航迹2次,平均航迹代价1527;第4类航迹2次,平均航迹代价1957。

3.2 实验结果分析与算法比较

本文主要从2个方面来分析与比较2种不同的算法:1)寻优能力,也就是全局寻优能力和寻优精度;2)寻优效率,也就是计算耗时和收敛效率。

从表1、表2中可以看出,平均航迹代价、方差和极

差由大到小排列都是：PSO 算法、RLPSO 算法、改进 ISPO 算法。说明这 3 种算法的寻优精度逐渐提高。

在实验 1 中可以得到 4 类可飞行的航迹，由这 3 种算法所计算的 4 类航迹的平均航迹代价之间的对比也能证明上述结论。由于 3 种算法都能找到全部 4 类航迹，全局搜索能力得不到充分的比较。

在实验 2 中也可以得到 4 类可飞行的航迹，由这 3 种算法所计算的 4 类航迹的平均航迹代价之间的对比也能证明关于寻优精度的结论。但是 PSO 算法只能找到 2 种航迹，RLPSO 算法能找到 3 种航迹，而改进 ISPO 算法能找到全部 4 种航迹。这说明全局搜索能力改进 ISPO 算法强于 RLPSO 算法，RLPSO 算法又强于 PSO 算法。

从算法耗时上来看，改进 ISPO 算法耗时最少，PSO 算法次之，RLPSO 算法又次之。从图 6 和图 8 中可以看出，改进 ISPO 算法的收敛速率明显快于其他 2 种算法。PSO 算法和 RLPSO 算法收敛到最优解所耗费的时间接近，但反向学习机制的引入增加了算法运算的总时间。从收敛情况上看，RLPSO 算法在运算初期收敛速率略快于 PSO 算法。

3.3　算法时间复杂度与空间复杂度分析

在本文的算法编码中，PSO 算法的时间复杂度为 $(4000m+12\,000)n+96\,100$，其中，m 为威胁的个数，n 为航迹点个数。假设 m 为定值，渐进时间复杂度为 $O(n)$；RLPSO 算法的时间复杂度为 $(4480m+13\,440)n+99\,164$，同样设 m 为定值，渐进时间复杂度为 $O(n)$；改进 ISPO 算法的时间复杂度为 $(100m+1300/3)n^2+2110n+120$，同样设 m 为定值，渐进时间复杂度为 $O(n^2)$。从量级上看，改进 ISPO 算法的时间复杂度差于前 2 种算法，但在航迹规划中，n 一般在保证能有效表示航迹的前提下取尽量小，以减少计算量。在 n 取较小的值时，改进 ISPO 算法所花费的时间低于另外 2 种算法。

图 9 和图 10 分别对应实验 1 和实验 2 的情况，由图可以得出与仿真实验相似的结论，在 $n=10$ 的情况下，花费的时间由小到大排列为：改进 ISPO 算法、PSO 算法、RLPSO 算法。

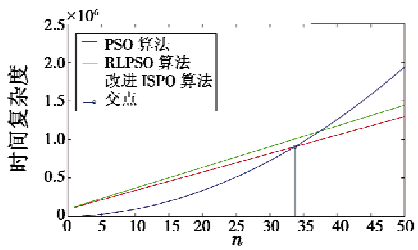

图 9　$m=3$ 的时间复杂度变化曲线

Fig. 9　The time complexity curve when $m=3$

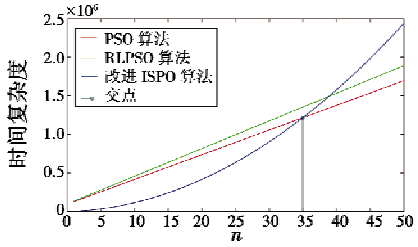

图 10　$m=5$ 的时间复杂度变化曲线

Fig. 10　The time complexity curve when $m=5$

3 种算法的空间复杂度都为 $O(n)$，因为改进 ISPO 算法只有 1 个粒子，而其他 2 种算法为粒子群，所以改进 ISPO 算法的空间复杂度优于其他 2 种算法。

4　结论

本文把 ISPO 算法引入航迹规划，并对其加以改进，有效提高了算法的性能。由实验结果可得，改进 ISPO 算法能够有效地应用于三维航迹规划，相较于 PSO 算法和 RLPSO 算法有较强的全局搜索能力，其规划结果也有较高的品质和较高的精度。改进 ISPO 算法还具有较高的计算效率和航迹代价收敛速率，但是在计算中仍存在航迹代价较大的航迹。在下一步的改进工作中可以考虑进一步提高算法的稳定性。

参 考 文 献

[1]　毛红保，田松，晁爱农. 无人机任务规划[M]. 北京：国防工业出版社，2015.

[2]　沈林成，陈璟，王楠. 飞行器任务规划技术综述[J]. 航空学报，2014，35(3)：594-606.

[3]　刘衍民. 一种解约束优化问题的混合粒子群算法[J]. 清华大学学报：自然科学版，2013，53(2)：242-246.

[4]　夏学文，刘经南，高柯夫，等. 具备反向学习和局部学习能力的粒子群算法[J]. 计算机学报，2015，38(7)：1398-1407.

[5]　SHI Y, EBERHART R C. A modified particle swarm optimizer[C]//IEEE World Congress on Computational Intelligence, Anchorage, 1998：69-71.

[6]　夏学文，刘经南，高柯夫，等. 具备反向学习和局部学习能力的粒子群算法[J]. 计算机学报，2015，38(7)：1398-1407.

[7]　纪震，周家锐，廖惠连，等. 智能单粒子优化算法[J]. 计算机学报，2010，33(3)：557-561.

[8]　傅阳光，周成平，丁明跃. 基于混合量子粒子群优化算法的三维航迹规划[J]. 宇航学报，2010，31(12)：2657-2664.

[9]　郑昌文，严平，丁明跃，等. 飞行器航迹规划[M]. 北京：国防工业出版社，2008：32-36.

[10]　纪震，廖慧连，吴青华. 粒子群算法及应用[M]. 北京：科学出版社，2009：64-70.