中国研究生创新实践系列大赛
“华为杯”第十六届中国研究生数学建模竞赛

学　　校	西安电子科技大学
参赛队号	19107010009
队员姓名	1.郝奇甲
	2.杨芝
	3.李春雨

中国研究生创新实践系列大赛
“华为杯”第十六届中国研究生数学建模竞赛

题 目　　基于 A*算法与遗传算法的多约束航迹规划问题

摘　　要：

智能飞行器的航迹规划一直是飞行器任务规划系统中的重要组成部分。针对题目所要求为复杂环境下的智能飞行器进行快速航迹规划，本文对三种多约束情形下飞行器航迹规划问题建立**组合优化**模型，通过求解这些模型并综合优化目标要求，对模型的性能进行对应评价。

题目提供的两个数据集中，包含起点、终点、垂直校正点和水平校正点坐标值，统计结果如下：附件 1 和附件 2 中的数据集均包含唯一起点和终点。其中，附件 1 中数据集有垂直校正点 305 个，水平校正点 306 个；附件 2 中数据集有垂直校正点 158 个，水平校正点 167 个。

针对问题 1，要求规划出满足 1.3 节中多约束条件（1）~（7）的飞行器航迹，飞行器航迹尽可能小且经过校正次数尽可能少。首先通过使用 **A*算法**对可行解进行求解，为了避免维数灾难，**遗传算法**在 A*算法的基础上，引入**特殊的局部搜索方式**来调整个体，使得个体不断靠近可行解，并引入**模拟退火思想**设计适应度函数，加快收敛速度并最终得到全局最优解。最终依据改进算法模型的航迹规划情况统计如下：针对**优化目标一——航迹长度**，数据集 1 规划后航迹的长度为：**104496.91**，进过校正的次数为 **10**；数据集 2 规划后航迹的长度为：109342.28，进过校正的次数为 12。针对**优化目标二——校正点次数**，数据集 1 规划后航迹的长度为：**105160.54**，进过校正的次数为 **8**；数据集 2 规划后航迹的长度为：109342.28，进过校正的次数为 12。对应的航迹规划结果表详情见表 1-2。

针对问题 2，要求在问题 1 考虑的多约束条件下，增加对飞行器**无法完成即时转弯**时的最小转弯半径限制。由限制设定满足两个情况：（1）规划航迹中的任意两个点之间的距离应该**大于最小转弯直径**，即 400m;（2）求航迹中任意两点的距离（从起点出发除外），应该为转弯时的最短圆弧长度。问题 2 用到的算法思想与问题 1 并无太大差异，最终得到航迹规划情况统计如下：针对**优化目标一——航迹长度**，数据集 1 规划后航迹的长度为：**109842.52**，进过校正的次数为 **9**；数据集 2 规划后航迹的长度为：122784.24，进过校正的次数为 15。针对**优化目标二——校正点次数**，数据集 1 规划后航迹的长度为：**109842.52**，进过校正的次数为 **9**；数据集 2 规划后航迹的长度为：122784.24，进过校正的次数为 15。对应的航迹规划结果表详情见表 5-6。

针对问题 3，由于飞行环境等不可控因素影响，需要考虑飞行器的真实校正情况，即存在部分校正点成功将某个误差校正为 0 的概率是 80%，引入**蒙特卡洛方法**重新规划问题 1，对部分校正点进行随机采样并计算（x平均），以（x平均）与 80%的关系来确定该校正点

的校正效果。最终得到航迹规划情况统计如下：**针对优化目标一——航迹长度**，数据集 1 规划后航迹的长度为：**104496.91**，进过校正的次数为 10；数据集 2 规划后航迹的长度为：**109342.28**，进过校正的次数为 12。**针对优化目标二——校正点次数**，数据集 1 规划后航迹的长度为：**105160.54**，进过校正的次数为 8；数据集 2 规划后航迹的长度为：**109342.28**，进过校正的次数为 12。对应的航迹规划结果表详情见表 8-9。

关键词：多约束 航迹规划 A*算法 遗传算法 蒙特卡洛方法

目录

1. 问题重述

1.1 背景

随着智能飞行器的广泛应用，智能飞行器的航迹规划成为飞行器任务规划系统中的重要组成部分，也是保证飞行器圆满完成任务的重要技术支撑。航迹规划是指在充分考虑飞行器的机动性能、到达时间、飞行区域以及飞行误差等因素前提下，在飞行器出发点和目的地之间为其规划出一条最优的飞行航迹。

由于飞行器的定位系统无法进行实时精准定位，必须借助飞行区域里的安全位置（称之为校正点）进行校正，使得定位误差得到有效控制，继而完成飞行器任务。因此，在系统定位精度限制条件下进行智能飞行器的航迹快速规划成为亟需解决的问题。

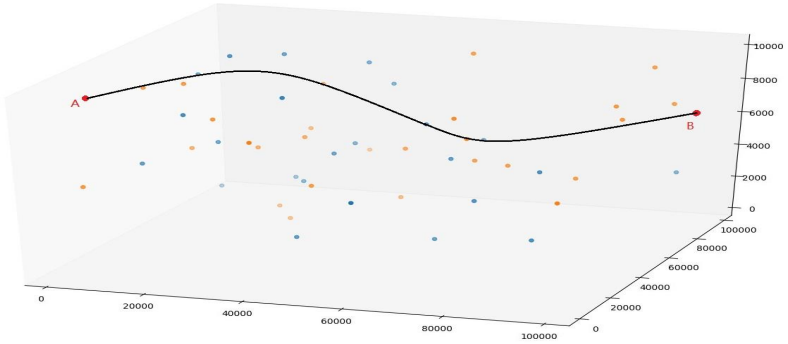


图 1 飞行器航迹规划区域示意图

1.2 待解决问题

本赛题要求就以下三种情况为智能飞行器建立从 A 点（出发点）$\delta = 0.001$ 到 B 点（目的地）的航迹规划一般模型和算法。

问题 1：针对附件 1 和附件 2 中的数据分别规划满足航迹约束条件（1）~（7）的飞行器的航迹，并且综合考虑以下优化目标：（A）航迹长度尽可能小；（B）经过校正区域进行校正的次数尽可能少。除此以外讨论算法的有效性和复杂度。

其中附件 1 数据的参数为： $\alpha_1 = 25, \alpha_2 = 15, \beta_1 = 20, \beta_2 = 25, \theta = 30, \delta = 0.001$。

附件 2 中数据的参数为： $\alpha_1 = 20, \alpha_2 = 10, \beta_1 = 15, \beta_2 = 20, \theta = 20, \delta = 0.001$。

请绘出两个数据集的航迹规划路径，并将结果（即飞行器从起点出发经过的误差校正点编号及校正前误差）依次填入航迹规划结果表。

问题 2：本问题是在第一问的基础上规划出满足航迹约束条件（8）的航迹。绘制两个数据集的航迹规划路径时注意（直线用黑色，圆弧用红色），并将结果填入航迹规划结果表。

问题 3：飞行器的飞行环境（例如天气等不可控因素影响）可能随时间动态变化，虽然校正点在飞行前已经确定，但是飞行器 在部分校正点进行误差校正时存在无法达到理想校正的情况（即将某个误 差精确校正为 0）。假设飞行器在部分校正点（附件 1 和附件 2 中 F 列标记为“1”的 数据）能够成功将某个误差校正为 0 的概率是 80%，如果校正失败，则校正后的剩余误差为 min(error，5)个单位（其中 error 为校正前误差，min 为取小函数），并且假设飞行器到达该校正点时即可知道在该点处是否能够校正成功，但不论校正成功与否，均不能改变规划路径。针对此情况重新规划问题 1 所要求的航迹，并要求成功到达终点的概率尽可能大。

绘制两个数据集的航迹规划路径，并将结果依次填入航迹规划结果表。

1.3 航迹约束描述

(1) 飞行器在空间飞行过程中需要实时定位，其定位误差包括垂直误差和水平误差。飞行器每飞行 1m，垂直误差和水平误差将各增加δ个专用单位，以下简称单位。到达终点时垂直误差和水平误差均应小于θ个单位，并且为简化问题，假设当垂直误差和水平误差均小于θ个单位时，飞行器仍能够按照规划路径飞行。

(2) 飞行器在飞行过程中到达校正点即能够根据该位置的误差校正类型进行误差校正。若垂直误差、水平误差都能得到及时校正，则飞行器可以按照预定航线飞行，通过若干个校正点进行误差校正后最终到达目的地。

(3) 在出发地 A 点，飞行器的垂直和水平误差均为 0。

(4) 飞行器在垂直误差校正点进行垂直误差校正后，其垂直误差将变为 0，水平误差保持不变。

(5) 飞行器在水平误差校正点进行水平误差校正后，其水平误差将变为 0，垂直误差保持不变。

(6) 当飞行器的垂直误差不大于α_1个单位，水平误差不大于α_2个单位时才能进行垂直误差校正。

(7) 当飞行器的垂直误差不大于β_1个单位，水平误差不大于β_2个单位时才能进行水平误差校正。

(8) 飞行器在转弯时受到结构和控制系统的限制，无法完成即时转弯(飞行器前进方向无法突然改变)，假设飞行器的最小转弯半径为 200m。

2. 模型假设和符号说明

2.1 模型假设

（1） 假设一：考虑到空间内任意两点间距离是直线最短，因此假设飞行器在每一步航迹规划过程中，都是以校正点或者终点 B 为下一步的飞行目标点。

（2） 假设二：问题 2 在模型处理上，仅仅和问题 1 的航迹点间距离求解方式不同。

（3） 假设三：飞行器不会以起点作为任意一段航迹的目标点，不会以终点作为任意一段航迹的起点。即假设飞行器从起点 A 出发后不会返回到 A 点，抵达终点 B 后立即终止航班。

2.2 符号说明

符号	符号含义
δ	飞行器飞行的误差专用单位大小
θ	到达飞行终点时误差边界单位大小
α_1	能够垂直误差校正时的垂直误差判断条件
α_2	能够垂直误差校正时的水平误差判断条件
β_1	能够水平误差校正时的垂直误差判断条件
β_2	能够水平误差校正时的水平误差判断条件
P_i	飞行器规划航迹上的点，$i=1,2,...,n$
m	给定数据集中校正点的个数
r_1,r_2,r_3	改进的估价函数的加权参数，$r_1\in[0,1],r_2\in[0,1],r_3\in[0,1]$
p^*	open 表里估价函数最小的结点
w_1,w_2	两个求解目标的加权参数，$w_1\in[0,1],w_2\in[0,1]$。
Arc_{P_{i-1},P_i}	点 p_{i-1} 到点 p_i 的弧长；
r	最小约束半径。
a,b	随机数的取值范围；
$f(x_i)$	随机分布概率函数。

3. 问题 1 分析、模型建立与求解

3.1 问题 1 的分析

问题 1 暂时不考虑飞行器方向突然改变时，由于飞行器的最小转弯半径约束而产生的路径规划限制。为了达到整个规划的航迹长度尽可能小，并且在经过校正区域时进行校正的次数尽可能少的目的，需要满足以下硬约束条件[1]：

（1）航迹点数量约束：规划后的航迹中航迹点的数量不小于两个，即航迹由出发地、目的地以及矫正区域的校正点（可为零）构成；

（2）飞行器正常飞行约束：飞行器满足 1.3 节中的条件（1）时可按照规划路径正常飞行；

（3）航迹点间距离约束：规划后的航迹中每两个航迹点之间的距离不为负；

（4）垂直误差校正约束：满足 1.3 节中的条件（6）；

（5）水平误差校正约束：满足 1.3 节中的条件（7）。

3.2 模型准备与建立

3.2.1 数据定义

（1）常量

δ ——到达飞行终点时误差边界单位大小；

θ ——到达飞行终点时误差边界单位大小；

α_1 ——能够垂直误差校正时的垂直误差判断条件；

α_2 ——能够垂直误差校正时的水平误差判断条件；

β_1 ——能够水平误差校正时的垂直误差判断条件；

β_2 ——能够水平误差校正时的水平误差判断条件；

m ——给定数据集中校正点的个数；

（2）变量

P_i ——飞行器规划航迹上的点，$i=1,2,...,n$；

r_1,r_2,r_3 ——改进的估价函数的加权参数，$r_1\in[0,1],r_2\in[0,1],r_3\in[0,1]$；

p^* ——open 表里估价函数最小的结点；

w_1,w_2 ——两个求解目标的加权参数，$w_1\in[0,1],w_2\in[0,1]$。

3.2.2 问题 1 模型建立

（1）目标函数：以达到整个规划的航迹长度 $F(P_1,P_2,...,P_n)$ 尽可能小，并且校正的次数 $n-2$ 尽可能少为目标进行建模，目标函数如下所示：

$$\min \begin{cases} F_1 = \sum_{i=2}^{n-2} dis\ (P_{i-1}, P_i) \\ F_2 = n \end{cases} \tag{3-1}$$

（2）航迹点数量约束，即规划后的航迹中航迹点的数量n肯定大于两个，航迹由出发地、目的地以及校正区域的校正点（数量可为零）构成：

$$n \geq 2 \tag{3-2}$$

（3）航迹点间距离约束，即规划后的航迹中每两个航迹点之间的距离（使用欧几里德公式）大于0：

$$dis(P_{i-1}, P_i) \geq 0, i = 2,3,...n \tag{3-3}$$

（4）飞行器正常飞行约束，即飞行器按照规划路径正常飞行需要满足1.3节中的条件（1）：

$$\max\{E_1(n), E_2(n)\} \leq \theta \tag{3-4}$$

$$E_1(j) = \sum_{i=2}^{j} Error_1(P_{i-1}, P_i), 2 \leq j \leq n \tag{3-5}$$

$$E_2(j) = \sum_{i=2}^{j} Error_2(P_{i-1}, P_i), 2 \leq j \leq n \tag{3-6}$$

（5）垂直误差校正约束：进行水平误差校正时必须满足1.3节中的条件（6）：

$$\mathrm{Error}_1 = \begin{cases} 0, & if \quad Error_1(P_1, P_u) \leq (\alpha_1 - dis(P_u, P_v) \times \delta) \quad and \quad Error_2(P_1, P_u) \leq (\alpha_2 - dis(P_u, P_v) \times \delta) \\ \sum_{i=2}^{u} Error_1(P_{i-1}, P_i) + dis(P_u, P_v) \times \delta, & else \end{cases} \tag{3-7}$$

（6）水平误差校正约束：进行水平误差校正时必须满足1.3节中的条件（7）：

$$\mathrm{Error}_2 = \begin{cases} 0, & if \quad Error_2(P_1, P_u) \leq (\beta_1 - dis(P_u, P_v) \times \delta) \quad and \quad Error_2(P_1, P_u) \leq (\beta_2 - dis(P_u, P_v) \times \delta) \\ \sum_{i=2}^{u} Error_2(P_{i-1}, P_i) + dis(P_u, P_v) \times \delta, & else \end{cases} \tag{3-8}$$

（7）综上，可建立问题1 的理论模型如下：

$$\min \begin{cases} F_1 = \sum_{i=2}^{n-2} dis\ (P_{i-1}, P_i) \\ F_2 = n \end{cases}$$

$$s.t \begin{cases} (1)\ n \geq 2 \\ (2)\ dis(P_{i-1}, P_i) \geq 0, i = 2,3,...n \\ (3) \max\{E_1(n), E_2(n)\} \leq \theta \end{cases}$$

对于条件（3），需要满足：

$$s.t.\begin{cases} E_1(j)=\sum_{i=2}^{j} Error_1(P_{i-1},P_i), 2\le j\le n \\ E_2(j)=\sum_{i=2}^{j} Error_2(P_{i-1},P_i), 2\le j\le n \\ \text{Error}_1=\begin{cases} 0, & if \quad Error_1(P_1,P_u)\le(\alpha_1-dis(P_u,P_v)\times\delta) \quad and \quad Error_2(P_1,P_u)\le(\alpha_2-dis(P_u,P_v)\times\delta) \\ \sum_{i=2}^{u} Error_1(P_{i-1},P_i)+dis(P_u,P_v)\times\delta, & else \end{cases} \\ \text{Error}_2=\begin{cases} 0, & if \quad Error_2(P_1,P_u)\le(\beta_1-dis(P_u,P_v)\times\delta) \quad and \quad Error_2(P_1,P_u)\le(\beta_2-dis(P_u,P_v)\times\delta) \\ \sum_{i=2}^{u} Error_2(P_{i-1},P_i)+dis(P_u,P_v)\times\delta, & else \end{cases} \end{cases}$$

3.3 模型的求解与分析

从建立的模型可以看出：

（1）智能飞行器的航迹规划问题属于多约束下的优化问题，即是按照预先设定好的多个约束函数，通过一系列的算法搜索，找到一条满足目标函数的全局最优解。解决这一问题的方法有：动态规划法、Dijkstra 算法、遗传算法、A*搜索算法等。在这些算法中，A*算法充分利用启发函数，能针对本问题预判下一步的航迹动作，并且计算简单直接，易于实现，能保证找到一条最短的可行路径而得到广泛应用。

（2）这一问题同时也是 NP 难的双目标组合优化问题（在多项式计算时间内无法获得最优解），求解规模的扩大，会导致可行解的指数级增长，仅使用 A*算法会出现维数灾难[2]。

而遗传算法（GA）作为求解复杂组合问题的最优化全局解方法，可将 A*算法形成的可行解通过模仿生物进化和遗传原理的方式进行优化收敛到全局最优解。

3.3.1 算法设计

（1）A*算法

A*是一种启发式的图搜索算法，通过设定合适的启发函数，全面评估各扩展搜索节点的代价，通过比较代价值的大小，选择最有希望的点加以扩展，直到找到目标结点为止。算法的相关步骤如下：

STEP1 建图

Ⅰ.将所有校正点标号离散化成 1～m，其中 m 为校正点的个数。起点 A 标号 $Aid=m+1$，终点 B 标号 $Bid=m+2$；任意两个节点之间连接边关系如下（1 表示相连接，0 表示不连接）：

$$edge[i,j]=\begin{cases} 1, if \quad dis[P_i,P_j]=\theta/\delta \\ 0 \end{cases} \qquad (3\text{-}9)$$
$$s.t.\begin{cases} i\ne j \\ 1\le i,j\le m \quad or \quad i=Aid \quad or \quad j=Bid \end{cases}$$

这种建图方法保证了每一个结点的后继结点是有限的，且不超过 m；**实际数据为数据集一最大为 192（总个数 611），数据集二最大为 51（总个数 325）**，可见均远小于校正点总个数，极大程度上缩小了搜索解空间的范围。

Ⅱ. 传统估价函数 $f(m)=g(m)+h(m)$, g(m) 为从起点 A 到当前结点 m 的实际代价，h(m) 是从节点 m 到目标节点的估计代价，通常叫做启发函数。然而传统估价函数在约束问

题上有一定的弊端，通常无法满足多目标约束，最终无法找到一个可行解。

因此改进估价函数如下（r_1, r_2, r_3 ——改进的估价函数的加权参数，$r_1 \in [0,1], r_2 \in [0,1], r_3 \in [0,1]$）：

$$f(m) = r_1 \times g(m) + r_2 \times h(m) + r_3 \times l(m) \tag{3-10}$$

$$l_m = \max\{E_1(m), E_2(m)\} \tag{3-11}$$

针对问题 1 中优化目标一：$g(m)$为飞行器的实际轨迹长度；而 h(m) 为飞行器从结点 m 到终点 B 的估计轨迹长度，其函数定义如下：

$$h(m) = \min_{i=1\sim m \ or \ i=Bid} dis(P_m, P_i) + dis(P_i, B)$$
$$\text{s.t.}\begin{cases} P_i \text{不在} close \text{表} \\ m \neq i \end{cases} \tag{3-12}$$

针对问题 1 中优化目标一：$g(m)$为飞行器的实际校正点；而 h(m) 为飞行器从结点 m 到终点 B 的校正次数，其函数定义如下：

$$h(m) = \max dis(p_i, p_j) \tag{3-13}$$

STEP2 寻路

Ⅰ.寻找最短航迹长度路径。

a. 初始化将起点 A 加入 open 表格;

b. 对于 open 表中的所有结点，计算：

$$p^* = \min\{i \mid f(i), i \in \text{open}\} \tag{3-14}$$

c. 计算结点 P^* 的所有后继结点 P_k 的 l_{pk} ,并将 P_k 加入 open 表；

d. 将结点 P^* 加入 close 表;

e. 重复步骤 b-d，直到 open 表为空。

Ⅱ.寻找轨迹最少校正点路径。

Ⅲ. 随意寻找一条从起点到终点的简单路径。

Ⅳ. 通过参数加权对双目标问题进行求解。

（2）引入遗传算法原因

实际实验中，A*算法已经可以找到一个不错的可行解，但是仍存在以下弊端：

Ⅰ.实际的效果取决于 r1, r2, r3 的选取，针对不同的取值，估价函数影响不同，效果差异较大,因而无法找到一个确切的标准来评价参数的选取。

Ⅱ. 对于双目标问题，若通过参数 w_1，w_2 加权评价会使得结果更靠近某一个目标，而且在优化解的过程中，由于加权求和的原因，往往会忽略某一单独目标的优秀解。

Ⅲ. 虽然对评价函数进行了改进，使得每一个结点尽可能的考虑到到目标点的评价指标。但是和实际航迹还是有差距，实验过程中会发现丢失掉一些可选情况，而这些情况往

往有可能进化到最优解。

针对以上弊端，引入了遗传算法对 A*的解空间进行优化。

（3）遗传算法（GA）

遗传算法[3][4]是指将初始种群通过选择、交叉和变异等操作，进行优胜劣汰迭代产生新的子代，并以个体适应度为指导，不断逼近解区间内的近似最优解。因此，在 A*算法得到可行解的基础上又采用了遗传算法来求解问题 1，并引入特殊的局部搜索方式来调整个体，使得个体不断靠近可行解。并引入模拟退火思想设计适应度函数，该思想能够加快收敛速度并最终趋于稳定。算法步骤如下：

STEP1 个体编码设计

采用链表编码，其中链表的长度 n 表示飞行轨迹经过的点的数目，且 $n \geq 2$，满足飞行轨迹包含起点 A 和终点 B。链表中的每一个元素 P_i 表示轨迹中的一个点。数据结构如下：

$$\begin{cases} \text{fixIndex} \\ x \\ \text{y} \\ z \\ currentDis \\ verticalError \\ horizontalError \\ fixpoinsLable \end{cases}$$

如图 2 表示一个个体，描述了一条从 A-B 的路径：

$$A \to P_1 \to P_2 \to \ldots \to P_n \to B$$

图 2　一条从 A-B 的路径

STEP2 初始化种群

在航迹规划过程中，由于约束的原因，无法保证初始化的所有个体都满足约束条件。因此我们决定随机生成个体，使用 A*寻路Ⅲ中的寻找起点到终点的简单路径的方法。同时在初始化种群中，加入 A*启发式搜索求得的最短航迹长度路径以及最少校正点路径，保证解空间满足约束，可以朝着最优解的方向进化。并在算法中加入局部搜索来调整个体，使得个体靠近可行解。

STEP3 两点交叉

采用两点交叉算子，由一定的交叉概率选择交叉的两个个体作为父代，随机产生两个交叉点，再由父代交换两个交叉点之间的航迹点序号产生子代，交叉示意图如下图所示：

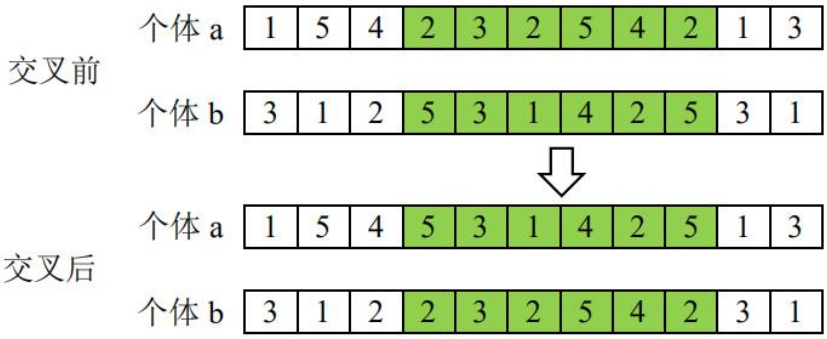


图 3　交叉操作示意图

STEP4 变异

对于种群中任意个体，选取该个体编码路径上的任意一点 p（不包括起点和终点），使点 p 以一定的随机概率变异到任意一个校正点。保证了个体可以以一定概率跳出局部最优解。

STEP5 局部搜索

对于任意一个个体$U = P_1 \to P_2 \to P_3 \to ... \to P_n$；如果$E(n) > \theta$。则说明该个体不满足可行解，此时我们进行局部搜索调整该个体。步骤如下：

Ⅰ. 求$c^* = \min\{c \mid 2 \le c \le m \text{ and } \max\{E_1(c), E_2(c)\} > \theta\}$；

Ⅱ. 随机生成$[c^*, n]$之间的整数mid；

Ⅲ. 使用A^*启发式搜索求的P_{c^*-1}到P_m的可行路径$Path_1$（轨迹长度较短），以及P_m到P_n的可行轨迹$Path_2$

Ⅳ. 调整个体编为 $P_1 \to P_2 \to P_{c^*-1} + \text{Path}_1 + \text{Path}_2$

此时可以保证解空间朝着满足约束的方向进化。

STEP6 选择

本问题是个双目标优化问题，因此我们采用 NSGAⅡ（**快速非支配排序**）**算法**的思想进行选择。

Ⅰ. 将生成的子代和父代进行合并，进行快速非支配排序。

Ⅱ. 对于每一个非支配层中的个体进行拥挤度计算。

Ⅲ. 根据非支配关系以及个体适应度选择合适的个体组成新的父代种群

选择操作如图所示：

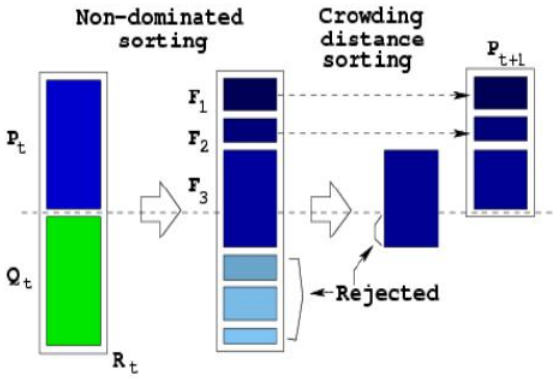


图 4　选择操作示意图

STEP7 终止条件

重复进行 STEP3-6，直到迭代代数超过给定的迭代上界为止。人为从非支配排序层最靠内部的一层的解空间选取一个解最为此问题的求解方案。

3.3.2 问题 1 求解结果与分析

（1）首先预处理数据。

将起点和终点的标号置为$m+1, m+2$，其中m未校正点的个数。初始化常量参数

$\alpha_1, \alpha_2, \beta_1, \beta_2, \delta, \theta$。

（2）对于 A*算法以及 A*算法融合遗传算法后的改进算法（以下简称改进算法）进行评价时，需要根据问题 1 的相关要求选取合适的指标。针对问题 1 的相关要求，一是要考虑优化目标；二是要考虑算法有效性和复杂度。最终选取的评价指标如下图所示：

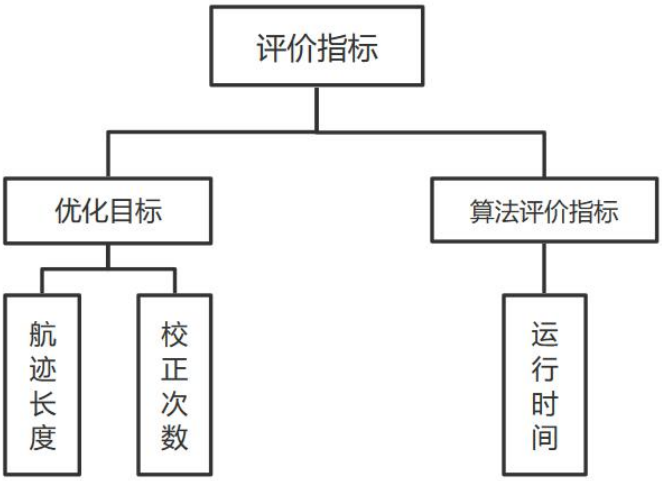


图 5　评价指标图

（3）按照 3.3.1 节中的算法设计流程，对数据集 1 和数据集 2 分别应用 A*算法以及改进算法并针对优化目标一（航迹长度）和优化目标二（校正次数）进行了编程求解，得到了对应的航迹结果显示图如下所示：

数据集 1&优化目标一：

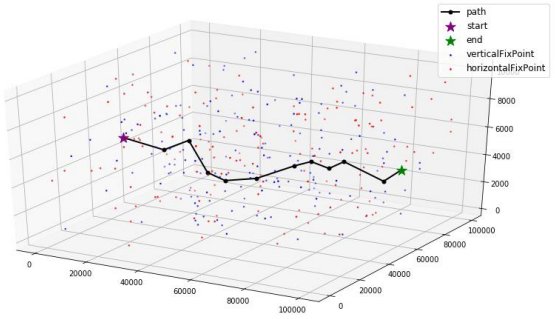


图 6　A*算法航迹结果图

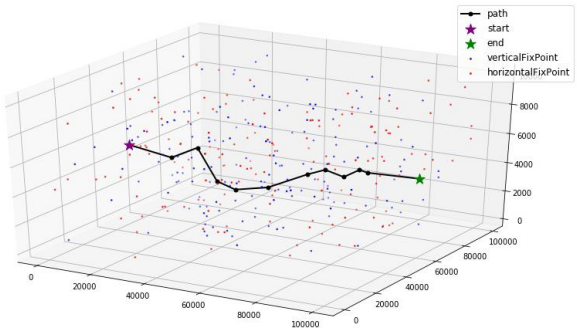


图 7　改进算法航迹结果图

数据集 1&优化目标二：

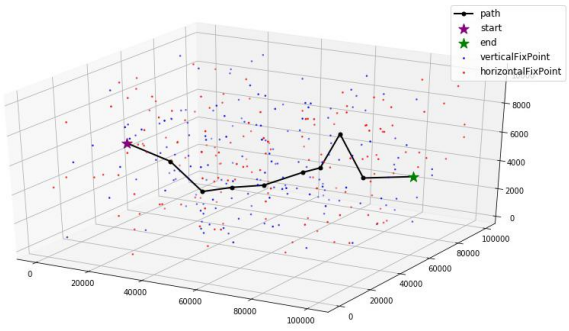


图 8　A*算法航迹结果图

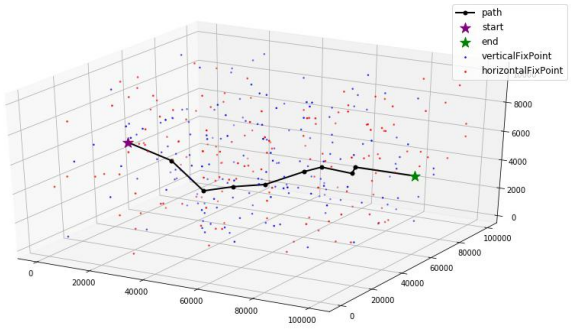


图 9　改进算法航迹结果图

数据集 2&优化目标一：

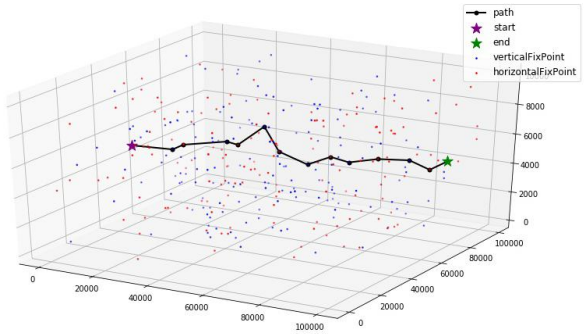


图 10　A*算法航迹结果图

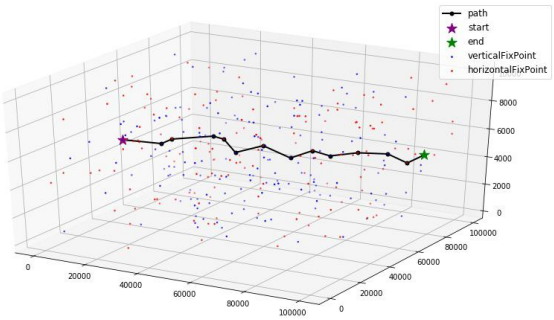


图 11　改进算法航迹结果图

数据集 2&优化目标二：

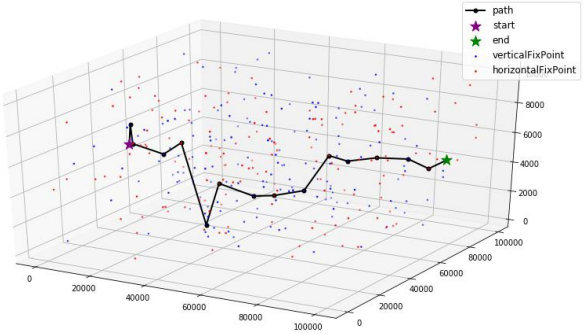


图 12　A*算法航迹结果图

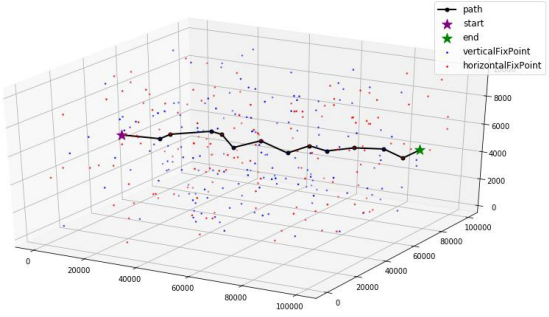


图 13　改进算法航迹结果图

（4）根据航迹显示图可得航迹误差表如下：

表 1 航迹规划结果表（数据集 1）

校正点编号	校正前垂直误差	校正前水平误差	校正点类型
0	0	0	出发点 A
503	13.39	13.39	1，1
69	8.81	22.20	0，1
237	21.31	12.50	1，1
155	11.20	23.70	0，0
540	17.91	6.71	1，0
250	11.43	18.14	0，0
340	24.20	12.77	1，1
277	12.00	24.77	0，0
612	28.35	16.35	终点 B

表 2 航迹规划结果表（数据集 2）

校正点编号	校正前垂直误差	校正前水平误差	校正点类型
0	0	0	出发点 A
163	13.29	13.29	0,1
114	18.62	5.33	1,1
8	13.92	19.26	0,1
309	19.45	5.52	1,1
305	5.97	11.49	0,1
123	15.17	9.20	1,1
45	10.01	19.21	0,1
160	17.49	7.49	1,1
92	5.78	13.26	0,1
93	15.26	9.48	1,1
61	9.83	19.32	0,0
292	16.39	6.55	1,0
326	6.96	13.51	终点 B

（5）对 A*与改进算法按照评价目标进行比较，得到相关表格：

表 3 算法评价结果表

数据集类型	算法名称	评价目标一			评价目标二		
		航迹长度	校正次数	运行时间	航迹长度	校正次数	运行时间
数据集 1	A*	104562.94	10	856.67ms	115482.17	8	605.67ms
	改进算法	**104496.91**	**10**	**1389.53ms**	**105160.54**	**8**	**1489.53ms**
数据集 2	A*	110772.81	12	889.78ms	135923.40	14	667.78ms
	改进算法	**109342.28**	**12**	**1456.03ms**	**109342.28**	**12**	**1778.03ms**

从表格中的结果可以看出：通过对 A*算法与改进算法针对优化目标一和二的比较，对于数据集 1 和 2 来说，在对应的目标下**航迹长度普遍变小，校正次数得到降低**，但要产生较好的效果，就要消耗较多的算法运行时间。

4. 问题 2 分析、模型建立与求解

4.1 问题 2 的分析

问题 2 是在问题 1 的基础上要求考虑飞机转弯时会受到最小转弯半径（200m）的限制。优化目标依旧与问题 1 相同，要求最小化航迹和进行的校正次数较少。因此问题 2 的航迹规划模型的硬约束条件只需要增加最小转弯半径约束。

对于最小转弯半径约束条件需要考虑两个情况：

（1）规划航迹中的任意两个点之间的距离应该大于最小转弯直径，即 400m;

（2）航迹中任意两点的距离不再是直线距离（从起点出发除外），而要根据前一时刻的方向和点的位置，求出下一时刻转弯时的最短圆弧长度。

4.2 模型准备与建立

4.2.1 数据定义

Arc_{P_{i-1},P_i}——点 p_{i-1} 到点 p_i 的弧长；

r——最小约束半径。

4.2.2 航迹点间的距离

考虑到飞机转弯时的诸多限制，题目中对飞行器的最小转弯半径进行了相关约束。利用这一约束条件，我们首先对两个数据集 1 和 2 中每两个校正点之间的空间直线距离进行了计算，计算结果显示如下：

表 4 校正点间直线距离显示表

数据集类型	总点数	总边数	最小边长	最大边长
数据集 1	613	72255	528.49	29999.94
数据集 2	327	10055	567.22	19997.39

由此看出，对于数据集 1 和 2 来说，任意两个校正点之间的距离均满足 4.1 节情况 1。

对于 4.1 节涉及到的情况 2，假设上一时刻飞行路径的两个端点为 P_1和P_2，下一时刻的校正点为 P_3。三个点之间的关系满足：

（1）P_1P_2 连线方向应为飞行器下一时刻转弯所形成圆的切线方向；

（2）P_2P_3 中垂线 MO 上的点与 P_2和P_3 距离相等；

（3）垂直于 P_1P_2 并过点 P_2 的直线 P_2O 与直线 $M0$ 相较于点 O；

（4）点 O 即为飞行器下一时刻转弯所形成圆的圆心。

以上描述可表示为下图：

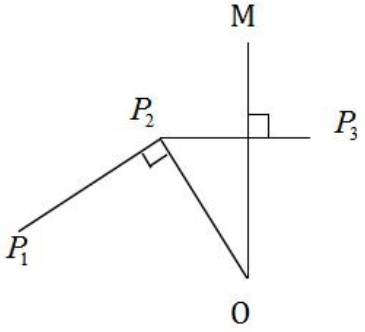


图 14 航迹点间最短弧计算示意图

由上图可得，航迹点间的最短弧长具体计算步骤如下：

STEP1:求 $P_1(x_1,y_1,z_1),P_2(x_2,y_2,z_2),P_3(x_3,y_3,z_3)$ 三点组成的平面方程，方程一般式为：$ax+by+cz+d=0$，带入 P_1,P_2,P_3 三点得：

$$\begin{cases} a=(y_2-y_1)\times(z_3-z_1)-(z_2-z_1)\times(y_2-y_1) \\ b=(z_2-z_1)\times(x_3-x_1)-(x_2-x_1)\times(z_3-z_1) \\ c=(x_2-x_1)\times(y_3-y_1)-(y_2-y_1)\times(x_3-x_1) \\ d=0-(a\times x_1+b\times y_1+c\times z_1) \end{cases} \tag{4-1}$$

STEP2:求圆心 O；

STEP3:根据圆心 O 及半径 r 可以求得弧长 Arc_{P_2,P_3}。

4.2.3 问题 2 模型建立

问题 2 目标函数与问题 1 相同，而满足的约束除了问题 1 的类似相关约束以外，航迹点间距离约束发生变化，并且还需增加最小转弯半径约束：

（1）航迹点间距离约束，即规划后的航迹中每两个航迹点之间的距离（根据弧长计算）大于 0：

$$dis'(P_{i-1},P_i)=Arc_{p_{i-1},p_i}\geq 0, i=2,3,...n \tag{4-2}$$

（2）最小转弯半径约束：飞行器的最小转弯半径为 200m:

$$\mathrm{r}\geq 200 \tag{4-3}$$

（3）问题 2 的理论模型如下：

$$\min\begin{cases} F_1=\sum_{i=2}^{n-2} dis\ (P_{i-1},P_i) \\ F_2=n \end{cases}$$

$$s.t\begin{cases} (1)\ \ n\geq 2 \\ (2)\ \ dis'(P_{i-1},P_i)=Arc_{i-1,i}\geq 0, i=2,3,...n \\ (3)\max\{E_1(n),E_2(n)\}\leq\theta \\ (4)\ \ \mathrm{r}\geq 200 \end{cases}$$

对于条件（3），需要满足：

$$
s.t.\begin{cases}
E_1(j)=\sum_{i=2}^{j} Error_1(P_{i-1},P_i), 2\le j\le n \\
E_2(j)=\sum_{i=2}^{j} Error_2(P_{i-1},P_i), 2\le j\le n \\
\mathrm{Error}_1=\begin{cases} 0, & if \quad Error_1(P_1,P_u)\le(\alpha_1-dis(P_u,P_v)\times\delta) \quad and \quad Error_2(P_1,P_u)\le(\alpha_2-dis(P_u,P_v)\times\delta) \\ \sum_{i=2}^{u} Error_1(P_{i-1},P_i)+dis(P_u,P_v)\times\delta, & else \end{cases} \\
\mathrm{Error}_2=\begin{cases} 0, & if \quad Error_2(P_1,P_u)\le(\beta_1-dis(P_u,P_v)\times\delta) \quad and \quad Error_2(P_1,P_u)\le(\beta_2-dis(P_u,P_v)\times\delta) \\ \sum_{i=2}^{u} Error_2(P_{i-1},P_i)+dis(P_u,P_v)\times\delta, & else \end{cases}
\end{cases}
$$

4.3 模型的求解与分析

4.3.1 算法说明

问题 2 的约束问题与问题 1 的差别在于航迹点间距离约束的变化，这一步增加了计算量，除此以外，增加最小转弯半径约束条件；目标函数相比无差别，依旧要求航迹长度尽可能小以及校正次数尽可能少。问题 2 依旧使用 A*融合遗传算法后的改进算法。

4.3.2 问题 2 求解结果与分析

（1）首先预处理数据。

（2）问题 2 选取的评价指标与问题 1 相同，仍然是航迹长度、校正次数以及运行时间。

（3）按照 4.3.1 节中的算法设计流程，对数据集 1 和数据集 2 分别应用 A*算法以及改进算法并针对优化目标一（航迹长度）和优化目标二（校正次数）进行了编程求解，得到了对应的航迹结果显示图如下所示：

数据集 1&优化目标一：

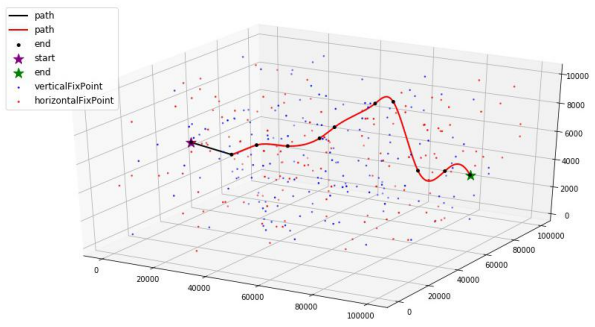


图 15　A*算法航迹结果图

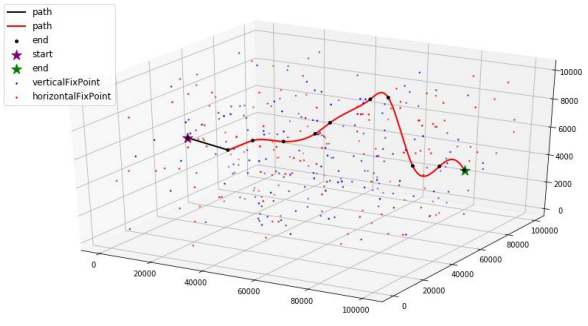


图 16　改进算法航迹结果图

数据集 1&优化目标二：

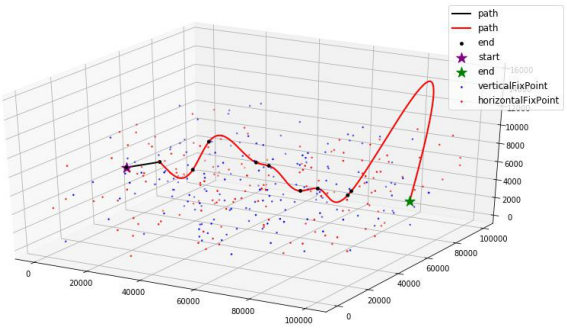


图 17　A*算法航迹结果图

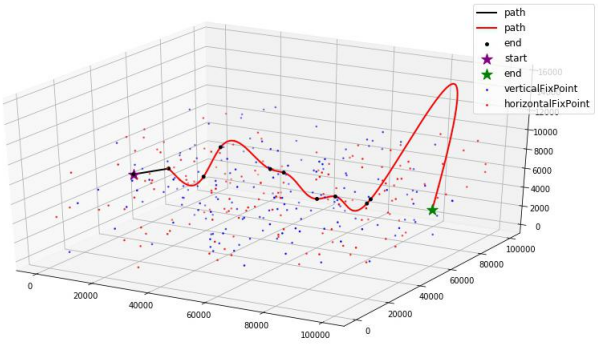


图 18 改进算法航迹结果图

数据集 2&优化目标一：

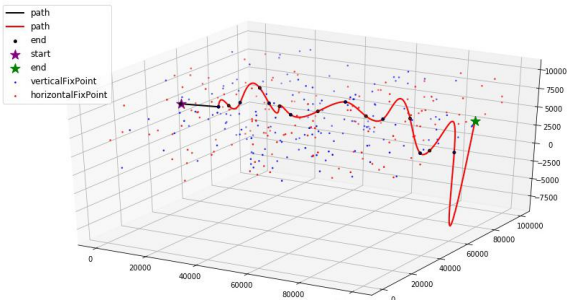


图 19　A*算法航迹结果图

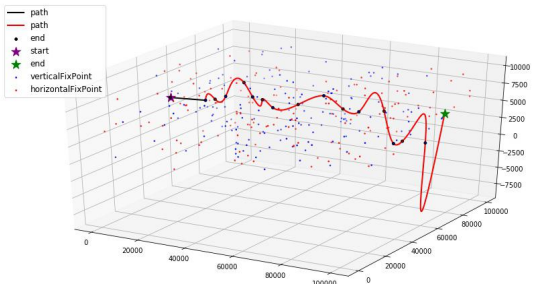


图 20 改进算法航迹结果图

数据集 2&优化目标二：

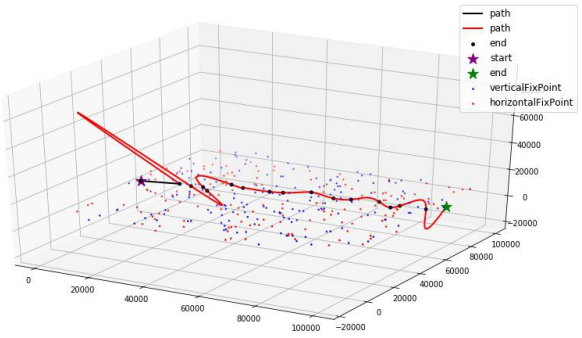


图 21　A*算法航迹结果图

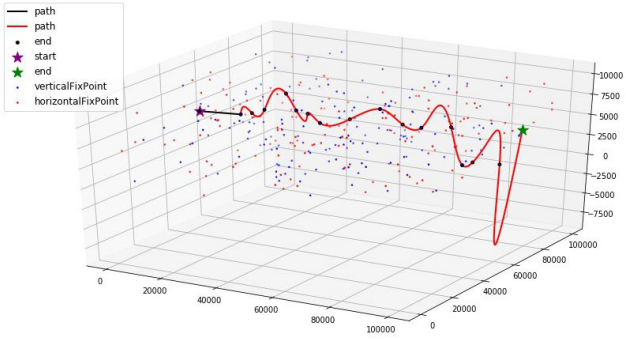


图 22 改进算法航迹结果图

（4）根据航迹显示图可得航迹误差表如下：

表 5 航迹规划结果表（数据集 1）

校正点编号	校正前垂直误差	校正前水平误差	校正点类型
0	0	0	出发点 A
503	13.39	13.39	1,1
69	8.81	22.20	0,1
237	21.31	12.50	1,1

233	10.82	23.32	0,0
33	15.99	5.17	1,0
315	15.46	20.63	0,0
403	23.49	8.03	1,0
594	11.03	19.06	0,0
501	22.23	11.20	1,0
612	8.49	19.69	终点 B

表 6 航迹规划结果表（数据集 2）

校正点编号	校正前垂直误差	校正前水平误差	校正点类型
0	0	0	出发点 A
163	13.29	13.29	0,1
114	18.62	5.33	1,1
234	4.53	9.87	0,1
222	11.81	7.28	1,1
8	4.63	11.91	0,1
309	10.15	5.52	1,1
305	5.97	11.49	0,1
123	15.17	9.20	1,1
231	9.44	18.64	0,1
160	18.15	8.72	1,1
92	5.78	14.49	0,1
93	15.26	9.48	1,1
38	6.30	15.78	0,0
110	10.22	3.93	1,0
99	9.20	13.13	0,0
326	17.85	8.65	终点 B

（5）对 A*与改进算法按照评价指标进行比较，得到相关表格：

表 7　算法评价结果表

数据集类型	算法名称	评价目标 1			评价目标 2		
		航迹长度	校正次数	运行时间	航迹长度	校正次数	运行时间
数据集 1	A*	109842.52	9	898.12ms	112565.54	9	787.23ms
	改进算法	**109842.52**	**9**	**1678.56ms**	**109842.52**	**9**	**1789.25ms**
数据集 2	A*	122784.24	15	897.54ms	127608.16	15	634.34ms
	改进算法	**122784.24**	**15**	**1587.57ms**	**122784.24**	**15**	**1588.2ms**

从表格中的结果可以看出：通过对 A*算法与改进算法进行优化目标的比较，对于数据集 1 和 2 来说，针对优化目标 2 **两者变动不大，航迹长度得到一定降低**，改进算法的运行也消耗较多时间。

5. 问题 3 分析、模型建立与求解

5.1 问题 3 的分析

问题 3 是在问题 1 的基础上对某些校正点是否能成功起到校正作用，增加概率条件约束。并满足校正失败时，校正的剩余误差需要取校正前误差与 5 之间的较小值。除此以外，校正成功与否并不影响飞机的规划路径。问题 3 的目标函数要求尽可能大概率保证成功到达终点。

问题 3 满足的硬约束条件相较于问题 1 来说增加了校正成功概率约束。

5.2 模型准备与建立

5.2.1 数据定义

a,b——随机数的取值范围；

$f(x_i)$——随机分布概率函数。

5.2.2 蒙特卡洛方法说明

蒙特卡罗方法，1940 年代中期由于科技发展和电子计算发明而被提出，是一种以概率统计理论为指导的数值计算方法。它的一种形式是将求解问题转化为某种随机分布的特征数，比如随机事件出现的概率。通过随机抽样的方法，以随机事件出现的频率估计其概率.

5.2.3 问题 3 模型建立

当在$[a,b]$间随机取一点x时，它对应的函数值为$f(x)$。粗略估计$f(x)$曲线下方面积时可以用$f(x)\times(b-a)$，即积分值。

因为问题三涉及对部分校正点增加校正成功的概率，所以可以将蒙特卡洛思想结合到问题三中。我们采取四次随机采样，得到了四个随机样本x_1,x_2,x_3,x_4，并且得到了这四个样本的$f(x_i)$值分别为$f(x_1,x_2,x_3,x_4)$，每个样本能求一个近似的面积值，大小可以表示为：

$$f(x_i)\times(b-a) \tag{5-1}$$

针对问题三的校正成功概率约束，$f(x)$的值为$[0,1.0]$，随意数$a=0,b=1.0$;因为$f(x)$函数满足随机分布，因此求四次随机样本的平均值（x平均）。

如果（x平均）<= 0.8，则我们认为该校正点可以正确校正，否则校正失败。

5.3 模型的求解与分析

5.3.1 算法设计

在实际算法的求解过程中，每一次用到校正点的时候，我们都会进行上述随机采样，由蒙特卡洛模拟法保证随机性。

算法大致思路与问题 1 相似，部分差异如下：

（1）针对 A*算法中的求解后继结点以及遗传算法中对个体的交叉，变异，局部搜索；

（2）若路径中的校正点包括不确定性校正点，对该校正点进行上述随机采样并计算（x平均）；

（3）若（x平均）<= 0.8;则可以调整对应垂直误差或者水平误差为 0，否则调整到 $\min(error,5)$ 。

5.3.2 问题 3 求解结果与分析

（1）首先预处理数据。

（2）问题 3 选取的评价指标与问题 1 相同，仍然是航迹长度、校正次数以及运行时间。

（3）按照 5.3.1 节中的算法设计流程，对数据集 1 和数据集 2 分别应用 A*算法以及改进算法并针对优化目标一（航迹长度）和优化目标二（校正次数）进行了编程求解，得到了对应的航迹结果显示图如下所示：

数据集 1&优化目标一：

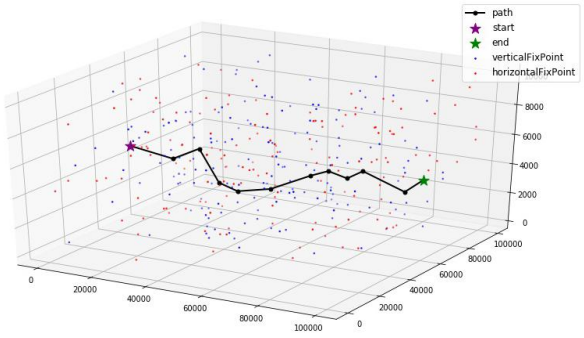


图 22　A*算法航迹结果图

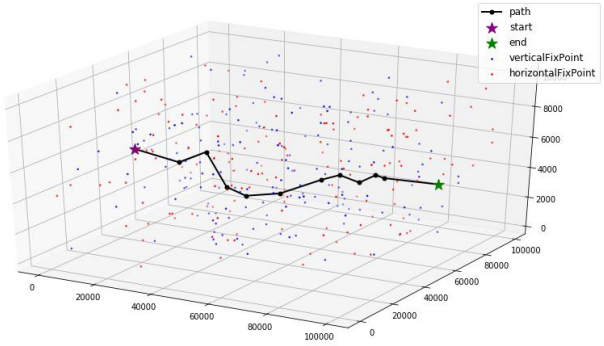


图 23 改进算法航迹结果图

数据集 1&优化目标二：

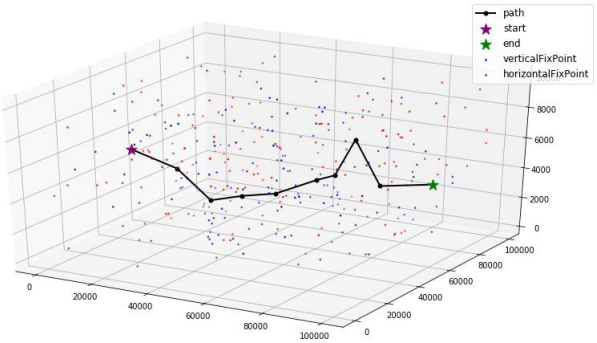


图 24 A*算法航迹结果图

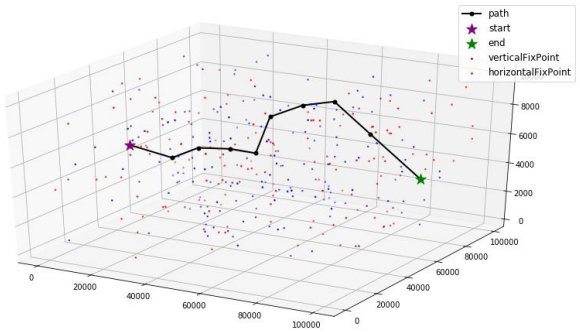
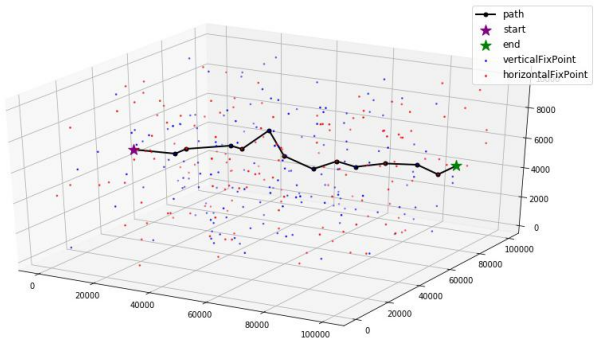


图 25 改进算法航迹结果图

数据集 2&优化目标一：



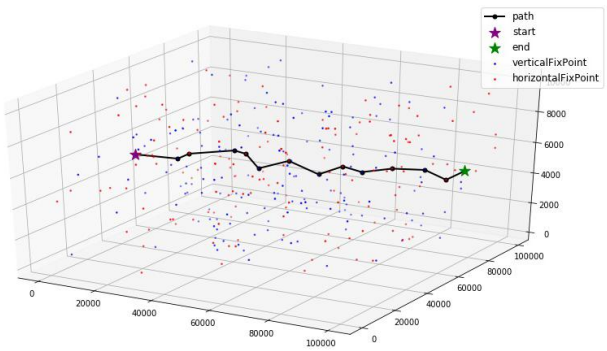
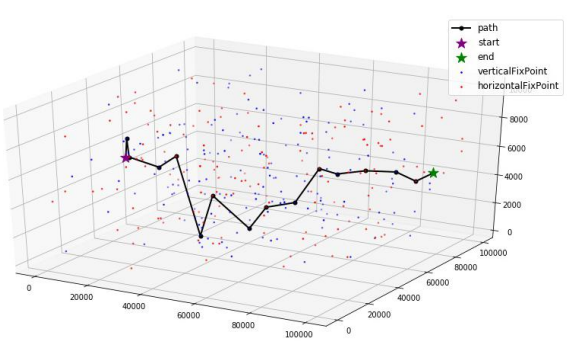


图 26　A*算法航迹结果图　　图 27 改进算法航迹结果图

数据集 2&优化目标二：



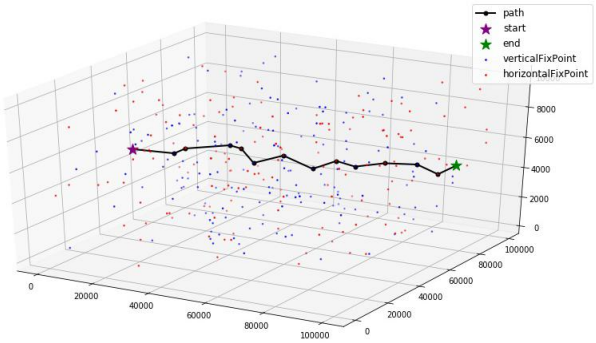


图 28 A*算法航迹结果图　　图 29 改进算法航迹结果图

（4）根据航迹显示图可得航迹误差表如下：

表 8　航迹规划结果表（数据集 1）

校正点编号	校正前垂直误差	校正前水平误差	校正点类型
0	0	0	出发点 A
503	13.39	13.39	1，1
69	8.81	22.20	0，1
237	21.31	12.50	1，1
155	11.20	23.70	0，0
540	17.91	6.71	1，0
250	11.43	18.14	0，0
340	24.20	12.77	1，1
277	12.00	24.77	0，0
612	28.35	16.35	终点 B

表 9　航迹规划结果表（数据集 2）

校正点编号	校正前垂直误差	校正前水平误差	校正点类型
0	0	0	出发点 A
163	13.29	13.29	0，1
114	18.62	5.33	1，1
8	13.92	19.26	0，1
309	19.45	5.52	1，1
305	5.97	11.49	0，1

123	15.17	9.20	1，1
45	10.01	19.21	0，1
160	17.49	7.49	1，1
92	5.78	13.26	0，1
93	15.26	9.48	1，1
61	9.83	19.32	0，0
292	16.39	6.55	1，0
326	6.96	13.51	终点 B

（5）对 A*与改进算法按照评价指标进行比较，得到相关表格：

表 10 算法评价结果表

数据集类型	算法名称	评价目标 1			评价目标 2		
		航迹长度	校正次数	运行时间	航迹长度	校正次数	运行时间
数据集 1	A*	104562.94	10	787.67ms	107217.33	8	604.67ms
	改进算法	**104496.91**	**10**	**1564.53ms**	**105160.54**	**8**	**1576.53ms**
数据集 2	A*	110772.81	12	897.78ms	122992.68	14	786.78ms
	改进算法	**109342.28**	**12**	**1478.03ms**	**109342.28**	**12**	**1876.03ms**

从表格中的结果可以看出：通过对 A*算法与改进算法针对优化目标进行比较，对于数据集 1 和 2 来说，**航迹长度普遍变小，校正次数得到降低**，但要产生较好的效果，还是要以消耗较多的算法运行时间为代价。

6. 模型评价与展望

6.1 模型优点

（1）本文提出了一种基于 A*算法融合遗传算法后的改进算法，用于规划飞行器的最优化航迹，能充分满足问题中要求的优化目标。

（2）改进算法避免了 A*算法会出现的维数灾难，以及其难以满足直飞限制以及无人机最小转弯半径等约束的局限性，从而利用遗传算法（GA）作为求解复杂组合问题的最优化全局解方法的特性，可将 A*算法形成的可行解通过模仿生物进化和遗传原理的方式进行优化收敛到全局最优解。

（3）对于数据，模型具有很强的鲁棒性，且能够找到全局的相对最优解，克服局部最优解的缺陷。

6.2 模型的缺点及改进方向

（1）遗传算法作为模型里的一种全局优化算法, 一般可以很快地收敛到最优解附近, 但是接近最优解后, 收敛速度可能会变得很慢。

（2）本文所有模型均为 NP 难问题，虽然我们使用 A*算法和进化算法能得到较好的结果，但实际中想要求得精确解的难度很大。

（3）后期可以进一步与其他搜索方法结合并比较得出更加优化的求解结果。

参考文献

[1] 李季, 孙秀霞. 基于改进 A-Star 算法的无人机航迹规划算法研究[J]. 兵工学报, 2008, 29(7):788-792.
[2] 辛培源. 基于三维环境复杂约束条件的无人机航迹规划方法研究[D]. 首都师范大学, 2014.
[3] 贾广芝. 基于遗传算法和稀疏 A*算法的无人机三维航迹规划研究[D].
[4] 李思海, 白存儒, LISi-hai, et al. 基于遗传算法的飞行器航迹规划研究[J]. 华东交通大学学报, 2007, 24(4):147-151.

附录

附录 1：C++编写算法主模块介绍

（1）点类

```
// 向着目标点 nxtP 移动 len 个距离后的点
Point moveToPoint(const Point &nxtP, const double &len);
// 获取两点距离
double getDis(const Point &preP, const Point &nxtP) const;
// 求解平面方程
void calFlatEquation(const Point &p1, const Point &p2, const Point P3,
                     double &A, double &B, double &C, double &D);
// 根据两点求直线方程
Line calLine(const Point &p1, const Point &p2);

// 过该点，垂直与直线 L 的垂线方程
Line calVerLine(const Line &L);

// 两条直线求交点
Point calComPoint(const Line &L1, const Line &L2);
```

（2）遗传算法：

```
// 单元测试
bool test();
// 入口函数
void execute();
// 种群初始化
void popuInit();
// 交叉
void cross();
// 变异
void variation();
// 校正
void fix();
// 局部搜索
void partSearch();
// 选择
void select();
// 迭代到下一代的处理
void initialize();
```

（3）A*寻路

```
// 路径最短情况下可行路径(满足约束)
std::vector<Point> findShortDisPath(const Point &mStart, const Point &mEnd);
```

```
// 校正点最少情况下可行路径(满足约束)
std::vector<Point> findMinFixCountPath(const Point &mStart,
                                       const Point &mEnd);

// 初始化种群,随机选取简单路径 (不一定满足约束)
std::vector<Point> randomSimplePath(const Point &mStart, const Point &mEnd);

// 简单路径(不一定满足约束)
std::vector<Point> simplePath(const Point &mStart, const Point &mEnd);
```