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	华东理工大学
参赛队号	19102510081
队员姓名	1.郝家岗
	2.张欢欢
	3.戴瑞勇

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　多约束条件下智能飞行器航迹快速规划

摘　　要：

本文研究了多约束条件下智能飞行器航迹快速规划问题，这是一个多目标约束问题。本文首先针对附件中的校正节点数据进行数据处理，构建从起点 A 到终点 B 的邻接距离网络，将航迹快速规划问题转化为 0-1 多目标整数规划问题。接着通过系统建模建立 0-1 多目标整数规划模型，并通过自适应改进型 Dijkstra 算法和自适应型蚁群算法，综合求解多目标规划模型，给出多约束条件下智能飞行器航迹快速规划的方案。

针对问题一，本文通过构架 0-1 多目标整数规划模型，以航迹长度尽可能小和经过校正区域进行校正的次数尽可能少为目标，通过动态规划中的分阶段优化方法，给出航迹快速规划的方案。在第一阶段利用自适应改进型 Dijkstra 算法和蚁群算法得出当前满足约束条件的最优路径和最佳误差校正点。第二阶段，在满足约束条件的基础上，应用贪婪算法在实际情况中对航行轨迹进一步优化。针对问题一，本文求出附件一的最优航行轨迹为：起点 A → 503 → 69 → 237 → 155 → 338 → 457 → 555 → 436 → 终点 B，飞行器最短的航迹长度为104.9×10^3m，经过校正区域进行校正的次数为 8 次；附件二的最优航行轨迹为：起点 A → 163 → 114 → 8 → 309 → 305 → 123 → 45 → 160 ⟶ 92 → 93 ⟶ 61 ⟶ 292 ⟶ 终点 B，飞行器最短的航迹长度为109.34×10^3m，经过校正区域进行校正的次数为 12 次。

针对问题二，与第一问不同的是，问题二增加了飞行器在实际飞行过程中有 200 米的最小转弯半径约束。本文通过系统分析最小转弯半径约束对飞行器实际飞行路程和能否成功到达的影响，重新构建邻接距离网络和多目标规划模型。接着，通过问题一算法模型与优化思路，求到附件一重新满足新约束条件的最优航行轨迹为：起点 A → 503 → 69 → 237 → 155 → 338 → 457 → 555 → 436 → 终点 B，飞行器最短的航迹长度为$104.903 \times 10^3 m$，经过校正区域进行校正的次数为 8 次；附件二的最优航行轨迹为：起点 A → 163 → 114 → 8 → 309 → 305 → 123 → 45 → 160 → 92 ⟶ 93 ⟶ 61 ⟶ 292 ⟶ 终点 B。飞行器最短的航迹长度为$109.464 \times 10^3 m$，经过校正区域进行校正的次数为 12

次。

针对问题三，与问题一不同的是，在飞行器飞行的实际情况中，问题三增加了部分可能发生误差校正失败的故障点。本文根据故障点分布和误差调整类型，重新构建邻接距离网络和多目标规划模型，增加目标函数和约束条件。接着，应用问题一的自适应改进型 Dijkstra 算法和蚁群算法，重新规划航行轨迹。

与前两问相比，为了使飞行器成功到达的概率尽可能大，问题三在满足约束条件的基础上，通过不断放松和改变校正节点的方法，求得附件一航行轨迹为：起点 A $\rightarrow 503 \rightarrow 294 \rightarrow 91 \rightarrow 607 \rightarrow 61 \rightarrow 250 \rightarrow 369 \rightarrow 566 \rightarrow 400 \longrightarrow$ 终点 B，飞行器最短的航迹长度为$105.77 \times 10^3 m$，经过校正区域进行校正的次数为 9 次，成功概率为 100%；接着综合考虑实际情况，当把针对附件二中的数据，求得成功到达概率目标约束为 100%、最优航行路径有：起点 A $\rightarrow 140 \rightarrow 226 \rightarrow 288 \rightarrow 306 \rightarrow 237 \rightarrow 280 \rightarrow 65 \rightarrow 142 \rightarrow 310 \longrightarrow 7 \longrightarrow 145 \longrightarrow 293 \longrightarrow 156 \longrightarrow 213 \longrightarrow 164 \longrightarrow 50 \longrightarrow 247 \longrightarrow 38 \longrightarrow 110 \longrightarrow 99 \longrightarrow 292 \longrightarrow$ 终点 B。飞行器最短的航迹长度为155.07×10^3m，经过校正区域进行经过的校正节点个数为为 21 个，成功概率为 100%。

关键词：邻接距离网络，自适应改进型 Dijkstra 算法，蚁群算法，0-1 多目标整数规划，多目标规划模型，贪婪算法，复杂网络，分阶段优化

目录

一、 问题重述

1.1 问题背景

复杂环境下航迹快速规划是智能飞行器控制的一个重要课题。由于系统结构限制，这类飞行器的定位系统无法对自身进行精准定位，一旦定位误差积累到一定程度可能导致任务失败。因此，在飞行过程中对定位误差进行校正是智能飞行器航迹规划中一项重要任务。

1.2 需要解决的问题

本题目是研究智能飞行器在系统定位精度限制下的航迹快速规划问题。假设飞行器的飞行区域如图 1 所示，出发点为 A 点，目的地为 B 点。其航行需要满足的约束如下：

(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需要对定位误差进行校正。飞行区域中存在一些安全位置（称之为校正点）可用于误差校正，当飞行器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。校正垂直和水平误差的位置可根据地形在航迹规划前确定（如图 1 为某条航迹的示意图, 黄色的点为水平误差校正点，蓝色的点为垂直误差校正点，出发点为 A 点，目的地为 B 点，黑色曲线代表一条航迹）。可校正的飞行区域分布位置依赖于地形，无统一规律。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。

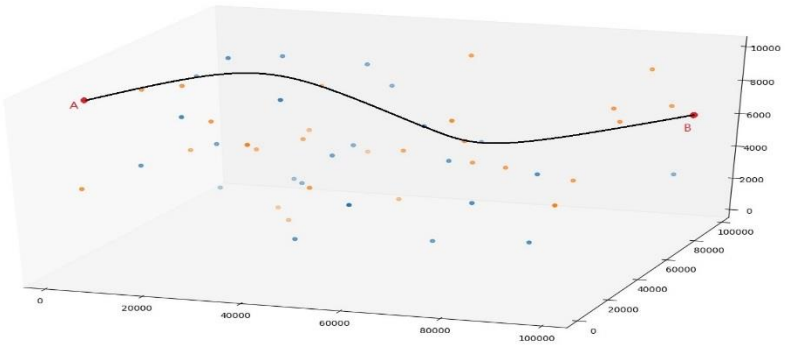

图 1-1：飞行器航迹规划区域示意图

(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。

需要通过建立数学模型，为上述智能飞行器建立从 A 点飞到 B 点满足上述约束条件

的航迹规划一般模型和算法。解决以下几个问题：

问题 1. 针对附件 1 和附件 2 中的数据分别规划满足条件（1）~（7）时飞行器的航迹，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。

讨论算法的有效性和复杂度，并绘出两个数据集的航迹规划路径。

问题 2. 针对附件 1 和附件 2 中的数据（参数与第一问相同）分别规划满足条件（1）~（8）时飞行器的航迹，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。

讨论算法的有效性和复杂度，并绘出两个数据集的航迹规划路径（直线用黑色，圆弧用红色）。

问题 3.飞行器的飞行环境可能随时间动态变化，虽然校正点在飞行前已经确定，但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），例如天气等不可控因素导致飞行器到达校正点也无法进行理想的误差校正。现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，要求成功到达终点的概率尽可能大，并绘出两个数据集的航迹规划路径。

二、 模型假设

为了便于问题的研究，对题目中某些条件进行简化及合理的假设。

1、忽略无人机自身大小对路径长度的影响，将无人机视为质点；

2、无人机总能恰好经过校正区点；

3、无人机在飞行途中不会遇到障碍物；

4、无人机到达一个校正区后立刻转弯飞向下一个校正区或终点；

5、为使路径长度尽可能小,无人机在转弯时只以最小转弯半径进行转弯。

三、符号说明

符号	意义
α	能够进行垂直误差校正的误差阈值
β	能够进行水平误差校正的误差阈值
θ	能够到达终点的垂直误差和水平误差阈值
δ	飞行器每飞行 1m, 飞行误差增加δ个单位
S	飞行器飞行网络图
A	飞行器飞行起点
B	飞行器飞行终点
C	校正点集:$C=[c_1,c_2,c_3\cdots,c_i,\cdots c_n]$， n为校正点个数
U	水平校正点集:$U=[u_1,u_2,u_3\cdots,u_i,\cdots u_m]$，m为水平校正点个数
V	垂直校正点集：$V=[v_1,v_2,v_3\cdots,v_i,\cdots v_Q]$，Q为垂直校正点个数
G	故障校正点集:$G=[g_1,g_2,g_3\cdots,g_i,\cdots g_r]$，其中r为总故障校正点个数
a_{ij}	邻接距离矩阵元素
s_{ij}	校正节点c_i，c_j间的距离
P_{ij}	校正节点c_i，c_j间的误差增量
P_i	校正节点c_i的水平或者垂直误差
$\boldsymbol{l}_{v_{k-1},v_k}$	垂直校正点v_{k-1},v_k间的水平或者垂直误差
x_{ij}	0-1 决策变量
$p^k{}_{ij}(t)$	蚂蚁k从城市i转移到城市j的概率
$b_{i,\ i+1}$	校正节点c_i，c_{i+1}间的边
γ	边$b_{i,\ i+1}$和$b_{i+1,\ i+2}$间的夹角
φ	最小半径转弯调整角度
w	为经过误差校正故障点的路径能够成功到达终点的概率
w_{ij}	校正节点c_j点的水平或垂直误差理想校正概率

四、问题一

4.1 问题描述与分析

问题一是研究智能飞行器在系统定位精度限制下的航迹快速规划问题。要针对附件 1 和附件 2 中的地点位置数据和地点安全类型，分别从 A 点出发到达目的地 B 点，规划满足约束条件（1）~（7）时飞行器的航行轨迹，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。讨论算法的有效性和复杂度，并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中。

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：

$$\alpha_1 = 20，\alpha_2 = 10，\beta_1 = 15，\beta_2 = 20，\theta = 20，\delta = 0.001$$

问题一的目的是在使得航迹长度尽可能小的同时，经过校正区域进行校正的次数也尽可能少，即为多目标动态规划问题。由于智能飞行器在飞行过程中每飞行 1m，垂直误差和水平误差将各增加δ个专用单位使得智能飞行器在飞行过程中必须选择误差校正位置进行垂直或者水平误差调整，而不能直接选择从起点到终点的航行轨迹。智能飞行器的航迹长度大小与飞行器在飞行过程中选择的校正点位置、校正点的垂直或水平误差校正类型有关。经过校正区域进行校正的次数与飞行器在飞行过程中不断累加的飞行定位误差值、每个校正点允许的垂直或水平误差阈值α,β个单位和终点 B 允许到达的垂直误差和水平误差θ个单位。需要注意的是飞行器在飞行过程中需要不断调整飞行方向以达到航迹长度尽可能小的目的。

根据题目和参考文献，给出智能飞行器轨迹规划策略如下：

（1）对附件中的起点A、校正节点和终点B的三维坐标以及校正点类型进行数据处理，构建飞行器飞行邻接距离矩阵，并在邻接距离矩阵中剔除不满足条件的校正节点关系，从而构建飞行器飞行网络图。

（2）飞行器在飞行过程中需要满足飞行路程最短的同时，尽可能经过少的飞行误差校正节点。此时我们通过建模选择最优路线和最佳误差点数量和。

（3）根据最优路线和最佳误差点数量和带入实际问题中综合考虑多目标规划问题的最优解。

（4）以上问题中，不考虑无人机大小，无人机为质点。

（5）应用不同算法，应用自适应改进型 Dijkstra 算法与基于蚁群寻优的智能优化蚁群算法在解决组合优化问题，通过调整数据，应用贪婪算法对航行轨迹进一步优化。

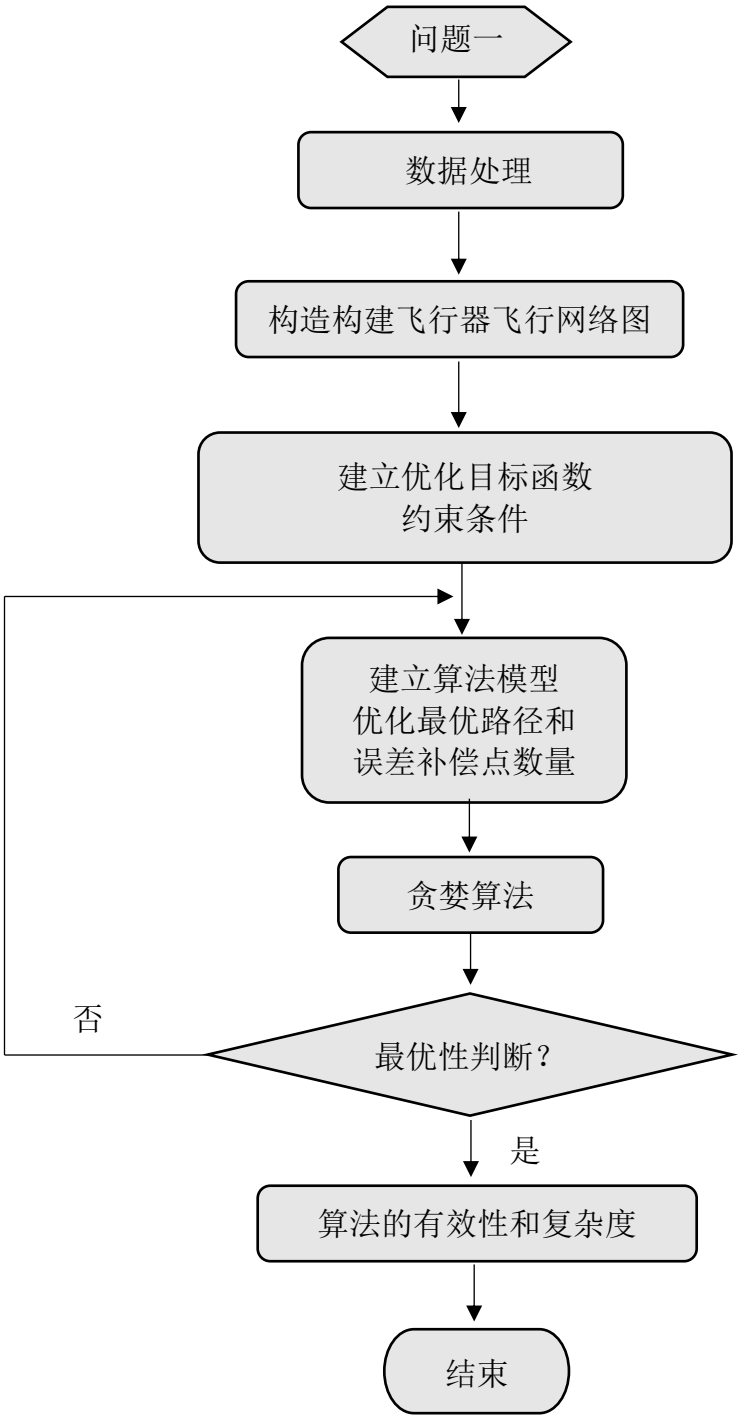

图 4-1：问题一求解思路及算法

4.2 模型建立

4.2.1 数据处理

根据附件中的起点 A、校正节点和终点 B 的三维坐标以及校正点类型，以附件一为例，从起点 A 到终点 B 共有 611 个水平和垂直节点，如下图 4-2 所示。

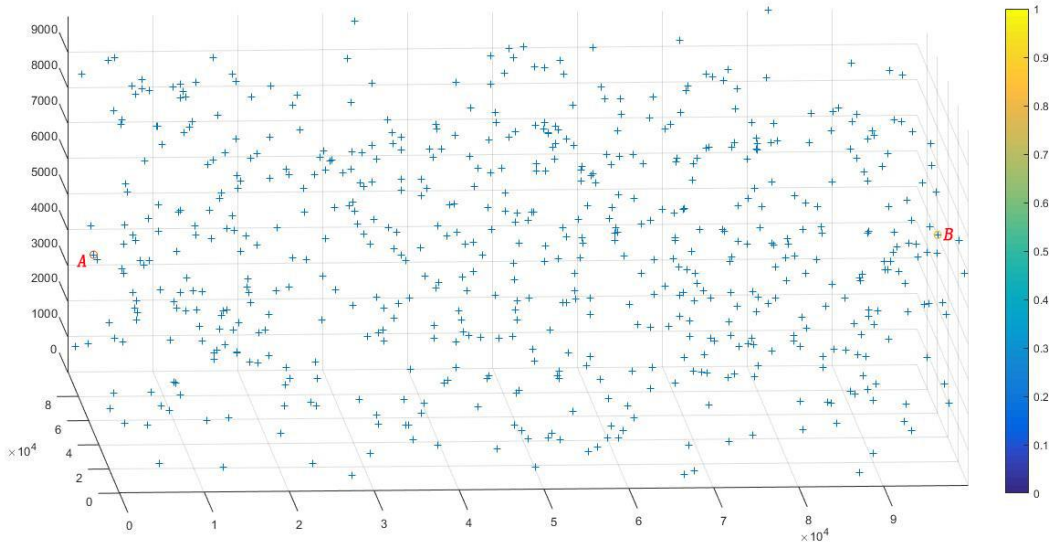

图 4-2：附件一校正节点分布图

首先进行数据处理，构建飞行器飞行邻接距离矩阵。邻接距离矩阵的基本原理是将图中 n 个顶点的数据存放在一组一维数组中，用一个 n × n 矩阵的形式来表示各个顶点间的邻接关系。图可分为有向图和无向图，在飞行器的航迹规划中，涉及到的是无向图。其中网络图的实质是结构较为复杂的一种数据结构，图的基本信息由两个部分组成：图中所有顶点的数据和各个顶点之间的关系，即边或者弧的关系。并在邻接距离矩阵中剔除不满足条件的校正节点关系，从而构建飞行器飞行网络图。

设D′为无向图，根据D′的各个顶点之间是否可以直接连接，编写一个矩阵，1 表示可以直接连接，0 表示不能直接连接，其定义如下：

$A = (a_{ij})_{n\times n}$称作无向图D′的邻接距离矩阵，其中n为校正节点个数加 2，

$$a_{ij} = \begin{cases} 1, & \text{起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ i \neq j \\ 0, & \text{未从起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ i \neq j \end{cases}$$

因为任何顶点自身不能相连，所以在任何无向图的邻接矩阵中，主对角线一定皆为 0，其余可以相连的顶点皆为 1。由此可得，每个无向图的邻接矩阵皆为关于主对角线对称的 n×n 的矩阵。邻接矩阵中包含了图的一切性质，邻接矩阵与它所唯一对应的图成一一对应关系。

针对于本题的航迹规划问题，将无向图转化为飞行器飞行网络图S，也就是测量出各个可以相连的校正点之间的距离，将无向图邻接矩阵中的“1”用实际距离代替，并在邻接距离矩阵中剔除不满足条件的校正节点关系，从而构建飞行器飞行网络图。即其中针对相连的各个顶点之间的距离不满足最大允许调整的水平和垂直误差，用“∞”表示，即将无向图邻接矩阵中的“0”用“∞”代替。

$S=(s_{ij})_{n\times n}$称作飞行器飞行网络图，$s_{ij}, i, j = 1,2,3\cdots, n$其中n为校正节点个数加 2，

$$s_{ij}=\begin{cases} s_{ij}, s_{ij} \le \text{最大误差阈值} \\ \infty, s_{ij} \le \text{最大误差阈值} \end{cases}$$

经过数据处理得到的飞行器飞行网络图如图 4-3 所示：

4.2.2 航迹规划模型建立

根据校正节点的数据处理得到的飞行器飞行网络图，飞行器在从起点 A 到终点 B 飞行过程中需要满足飞行路程最短的同时，尽可能经过少的飞行误差校正节点。我们通过建模选择最优路线和最佳误差点数量和，这是一个多目标动态规划问题。飞行器在飞行过程中经过校正节点c_i到校正节点c_j, 则选中飞行器飞行网络图中的边s_{ij}，加入最优飞行路径中。否则，s_{ij}取零。因此航迹规划问题可以转化为 0–1 整数规划问题。

为了达到整体最优的情况，我们将航迹规划的整体过程分为两步，进行分阶段优化。先规划从起点 A 到终点 B 的最短路径，再找出满足约束条件的最优路径使得飞行器在从起点 A 到终点 B 飞行过程中需要满足飞行路程最短的同时，尽可能经过少的飞行误差校正节点。这也是一个组合网络优化问题。

对所有校正点进行编号：

（1）校正点集:$\mathrm{C}=[c_1,c_2,c_3\cdots,c_i,\cdots c_n]$, $\mathrm{i}=1,2,3\cdots,\mathrm{n}$，其中n为校正点个数；

（2）水平校正点集:$\mathrm{U}=[u_1,u_2,u_3\cdots u_i\cdots u_m]$, $\mathrm{i}=1,2,3\cdots,\mathrm{m}$，其中m为水平校正点个数；

（3）垂直校正点集：$\mathrm{V}=[v_1,v_2,v_3\cdots v_i\cdots v_Q]$, $\mathrm{i}=1,2,3\cdots,\mathrm{Q}$，其中Q为垂直校正点个数；

其中校正点集C、水平校正点集U和垂直校正点集V之间的关系为$\mathrm{U}\cup\mathrm{V}=\mathrm{C}$，且$\mathrm{U}\cap\mathrm{V}=\emptyset$，

对于任意一点$c_i,\mathrm{i}=1,2,3\cdots,\mathrm{n}$根据附件中给出的该点水平或垂直误差调整类型信息，进行判定该点属于水平点集U或者垂直点集V，其判定步骤如下：（程序见附件 1）

Step1：校正点初始化。根据附件中数据，将起点 A 到终点 B 中的点按附件数据依次标号到$c_i,\mathrm{i}=1,2,3\cdots,\mathrm{n}$。

Step2:校正点类型判断。根据附件中标出的校正点$c_i,\mathrm{i}=1,2,3\cdots\mathrm{n}$的类型，分别将水平或者垂直校正点$c_i$分别依次标号到水平校正点集U垂直校正点集V，且对:

(1)$\forall u_k,u_{k+1}\in$水平校正点集U，u_k,u_{k+1}对应到校正点集C中的c_i和c_j，$\mathrm{i},\mathrm{j}\epsilon\{1,2,3\cdots,\mathrm{n}\}$有：$\mathrm{i}<j$。

(2)$\forall v_k,v_{k+1}\in$垂直校正点集V，v_k,v_{k+1}对应到校正点集C中的c_i和c_j，$\mathrm{i},\mathrm{j}\epsilon\{1,2,3\cdots,\mathrm{n}\}$有：$\mathrm{i}<j$；

Step3:直到完成所有校正节点的编号和校正节点判断。

首先根据问题一的规定模型假设：首先假设飞行器从一个校正节点到另一个校正节点之间始终匀速直线运动，忽略转弯引起的轨迹变化。飞行器在从起点 A 到终点 B 的飞行过程中需要依次选择水平校正节点和垂直校正节点进行误差校正，以满足能够在误差允许的

范围内到达各个水平或者垂直校正节点进行水平和垂直误差校正。其中需要注意的是飞行器到达终点 B 的允许误差和到达各校正节点的允许误差值是不一致的，这是一个多目标航迹规划问题。多目标规划的优化目标是获得在从起点A到终点B的飞行中，航迹路程最小的同时，经过的飞行误差调整点u_i or v_i ,$\mathrm{i}=1,2,3\cdots,\mathrm{m}$ or Q尽可能的少。

飞行器飞行路程目标函数为：

$$\min\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}\cdot s_{ij} \tag{4-1}$$

飞行器经过水平或者垂直误差校正点的目标函数为：

$$\min\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij} \tag{4-2}$$

式中，$0-1$ 变量：$x_{ij}=\begin{cases}1, & \text{当飞行器从起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ \mathrm{i}\neq\mathrm{j}\\ 0, & \text{当飞行器未从起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ \mathrm{i}\neq\mathrm{j}\end{cases}$，

起点 or 校正点c_i到终点 or 校正点c_j的路程长度：

$$s_{ij}=\sqrt[2]{(\mathrm{X}_i-\mathrm{X}_j)^2+(\mathrm{Y}_i-\mathrm{Y}_j)^2+(\mathrm{Z}_i-\mathrm{Z}_j)^2} \tag{4-3}$$

约束条件包括：

⑴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，则飞行器从起点或者校正点c_i到终点或者下一个校正点c_j的误差增量P_{ij}为：

$$\mathrm{P}_{ij}=\begin{pmatrix}s_{ij}\cdot\delta\\ s_{ij}\cdot\delta\end{pmatrix} \tag{4-4}$$

式中(a,b)为1×2 维矩阵，因为任意一个误差校正点c_i，$\mathrm{i}=1,2,3\cdots\mathrm{n}$在误差允许范围内可调整水平或者垂直误差。则在任意一个校正点c_i，$\mathrm{i}=1,2,3\cdots\mathrm{n}$的剩余误差为：

$$\mathrm{P}_j=\begin{pmatrix}\mathrm{P}_{ju}\\ \mathrm{P}_{jv}\end{pmatrix}=\begin{pmatrix}\mathrm{P}_{iu}\\ \mathrm{P}_{iv}\end{pmatrix}+x_{ij}s_{ij}\delta\cdot\begin{pmatrix}1\\1\end{pmatrix}=\begin{cases}\begin{pmatrix}\mathrm{P}_{iu}+x_{ij}s_{ij}\delta\\0\end{pmatrix}, & \mathrm{v}_j\in\mathrm{V}\\ \begin{pmatrix}0\\\mathrm{P}_{iv}+x_{ij}s_{ij}\delta\end{pmatrix}, & \mathrm{u}_j\in\mathrm{U}\end{cases} \tag{4-5}$$

式中P_{ju}表示：校正点c_i，$\mathrm{i}=1,2,3\cdots\mathrm{n}$的水平剩余误差；$\mathrm{P}_{jv}$表示：校正点$c_i$，$\mathrm{i}=1,2,3\cdots\mathrm{n}$的垂直剩余误差。

要求到达终点时垂直误差和水平误差均应小于θ个单位，即：

$$\mathrm{P}_{iu}+x_{iB}s_{iB}\delta\leq\theta\text{且}\mathrm{P}_{iv}+x_{iB}s_{iB}\delta\leq\theta \tag{4-6}$$

⑵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，即：

$$P_{iv} + x_{ij}s_{ij}\delta \le \alpha_1 ，\ v_j \in V \qquad (4-7)$$

$$P_{iu} + x_{ij}s_{ij}\delta \le \alpha_2 ，\ v_j \in V \qquad (4-8)$$

附件一中$\alpha_1 = 25$，$\alpha_2 = 15$, 附件二中$\alpha_1 = 20$，$\alpha_2 = 10$。

⑶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。

$$P_{iv} + x_{ij}s_{ij}\delta \le \beta_1 ，\ u_j \in U \qquad (4-9)$$

$$P_{iu} + x_{ij}s_{ij}\delta \le \beta_2 ，\ u_j \in U \qquad (4-10)$$

附件一中$\beta_1 = 20$，$\beta_2 = 25$，附件二中$\beta_1 = 15$，$\beta_2 = 20$。

⑷飞行器在飞行过程中会根据最优路径和节点数选择经过水平误差补偿和垂直误差补偿点，在算法的实际计算中常常需要直接约束任意两个水平或者垂直误差的阈值，约束条件如下所示：

$$\boldsymbol{l}_{v_{k-1},v_k} \cdot \delta \le \alpha_1 \qquad (4-11)$$

$$\boldsymbol{l}_{C_{\max\{i|C_i \notin V, i<ord(v_k)\}},v_k} \cdot \delta \le \alpha_2 \qquad (4-12)$$

$$\boldsymbol{l}_{u_{k-1},u_k} \cdot \delta \le \beta_1 \qquad (4-13)$$

$$\boldsymbol{l}_{C_{\max\{i|C_i \notin V, i<ord(v_k)\}},v_k} \cdot \delta \le \beta_2 \qquad (4-14)$$

综上所述，飞行器飞行航迹的多目标规划为：

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} \cdot s_{ij}$$

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} \qquad (4-15)$$

$$\text{s.t}\begin{cases} 0-1\text{变量}x_{ij}=\begin{cases}1, & \text{当飞行器从起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ \mathrm{i}\neq\mathrm{j}\\ 0, & \text{当飞行器未从起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ \mathrm{i}\neq\mathrm{j}\end{cases}\\ \mathrm{P}_{iv}+x_{ij}s_{ij}\delta\le\alpha_1,\ \mathrm{v}_j\in\mathrm{V}\\ \mathrm{P}_{iu}+x_{ij}s_{ij}\delta\le\alpha_2,\ \mathrm{v}_j\in\mathrm{V}\\ \mathrm{P}_{iv}+x_{ij}s_{ij}\delta\le\beta_1,\ \mathrm{u}_j\in\mathrm{U}\\ \mathrm{P}_{iu}+x_{ij}s_{ij}\delta\le\beta_2,\ \mathrm{u}_j\in\mathrm{U}\\ \mathrm{P}_{iu}+x_{iB}s_{iB}\delta\le\theta,\mathrm{B}\text{为终点}\\ \mathrm{P}_{iv}+x_{iB}s_{iB}\delta\le\theta,\mathrm{B}\text{为终点}\end{cases}$$

4.3 求解算法建立

针对轨迹规划问题的算法研究已经有很多的成果了，其中传统的航迹规划算法有 prim 算法，Dijkstra 算法和动态规划方法。传统航迹规划算法适用于问题环境相对简单且规模较小的情况，其具有运算速度快、复杂性低、有效性高，能够根据实际问题快速给出最优解等优点，应用范围广泛而被广泛使用。针对多目标航迹规划问题，本文使用多阶段 Dijkstra 算法求解，算法的最大迭代次数为 1440 次，最长执行时间为 42.23S，由此可见，该算法时间复杂度较低。其具体方法如下：

4.3.1 最短路径数学表达

飞行网络图中有n个顶点，现需要求从校正节点 A 至校正节点 B 的最短路径。设$\mathrm{S}=\left(s_{ij}\right)_{n\times n}$为邻接距离矩阵，其分量为：

$$s_{ij}=\begin{cases}s(c_i,c_j), & c_j\in C\\ \infty, & others\end{cases}\qquad(4-16)$$

决策变量为x_{ij}，当$x_{ij}=1$，说明c_i,c_j位于校正节点 A 至校正节点 B 的路上；否则$x_{ij}=0$。其数学规划表达式为：

$$\min\sum_{v_i,v_j\in E}S_{ij}\cdot x_{ij}\qquad(4—17)$$

$$\text{s.t.}\sum_{j=1}^{n}x_{ij}-\sum_{j=1}^{n}x_{ji}=\begin{cases}1, & \mathrm{i}=1\\ -1, & \mathrm{i}=\mathrm{m}\\ 0, & i\neq 1,n\end{cases}\qquad(4—18)$$

$$x_{ij}=0\text{ or }1$$

4.3.2 自适应改进型 Dijkstra 算法求解步骤

轨迹规划问题的常用算法根据解法思想条件可以主要分为传统算法和近代优化算法两类，其中传统的航迹规划算法有 prim 算法，Dijkstra 算法和动态规划方法。传统航迹规划算法适用于问题环境相对直接情况，其具有运算速度快、复杂性低、有效性高，适用范围广的特点，能够快速根据实际问题得出结果。

传统的航迹规划算法 Dijkstra 算法具有有效性高和较强的全局搜索能力。本文在 Dijkstra 算法原理的基础上进行自适应改进，求解情况复杂的多目标优化问题中，并将改进自适应的 Dijkstra 算法作为求解的本题适用算法。

该算法的具体步骤如下：

Step1: 令 $l(c_0)$，对$c_i \neq c_0$，令 $S(c_i) = \infty, S_0 = \{c_0\}$，$i = 0$；

Step2: 对每个$c_i \in \bar{C}(\bar{C} = C \backslash c_i)$,用$min_{c_i \in C}\{l(c_i), l(c_i) + s_{ij}\}$代替 $l(c_i)$。

计算$min_{c_i \in C}\{l(c_i)\}$，把达到这个最小值的一个定点$c_i$记为$y_{i+1}$，

令$S_{i+1} = S_i \cup \{y_{i+1}\}$；

Step3: 若 $i = n - 1$，停止；若 $i < n - 1$,用 $i + 1$ 替代 i,转 Step2。

Step4:将得到的路径中的每一个c_i，按水平或者垂直校正节点类型分别标记为水平校正点集 $U = [u_1, u_2, u_3 \cdots u_k \cdots u_m]$，$i = 1,2,3 \cdots, m$和垂直校正点集：

$V = [v_1, v_2, v_3 \cdots v_i \cdots v_Q]$，$i = 1,2,3 \cdots, Q$，其中m 和 Q为分别路径中的水平校正点和垂直校正点个数；

Step5:对路径中的水平或者垂直误差校正点进行约束优化，约束优化条件如下：

$$\boldsymbol{l}_{v_{k-1}, v_k} \cdot \delta \leq \alpha_1 \qquad (4-19)$$

$$\boldsymbol{l}_{C_{\max\{i|C_i \notin V, i<ord(v_k)\}}, v_k} \cdot \delta \leq \alpha_2 \qquad (4-20)$$

$$\boldsymbol{l}_{u_{k-1}, u_k} \cdot \delta \leq \beta_1 \qquad (4-21)$$

$$\boldsymbol{l}_{C_{\max\{i|C_i \notin V, i<ord(v_k)\}}, v_k} \cdot \delta \leq \beta_2 \qquad (4-22)$$

Step6:若满足约束条件，则转Step7，否则转Step2；

Step7：结束

算法流程为：

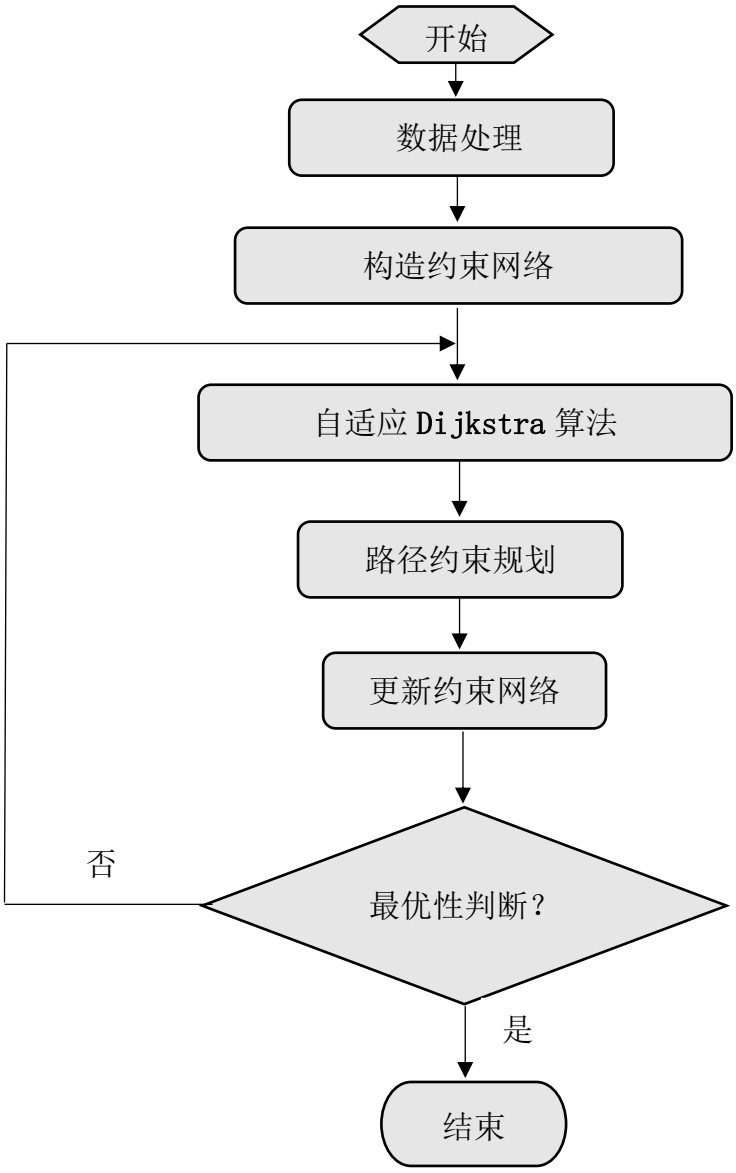

图 4-3：自适应改进型 Dijkstra 算法流程图

4.3.3 自适应基于蚁群算法的飞行器航迹规划求解

蚁群算法作为一种近现代优化算法，和绝大部分的现代近似优化算法一样，蚁群算法也是基于自然规律或者人体基因遗传进化原理而来的算法。蚁群算法是一种用于解决复杂组合优化问题的适当的方法，它具有鲁棒性强、适应性好、全局搜索和较好的全局最优等优点。应用在大部分环境中有较好的结果。

蚁群算法的算法思想是基于蚁群在自然界中能够在未知的条件下找到从巢穴到食物源的最短路径的寻优原理。通过一种正反馈机制，不断将搜索方向趋于最短路径上。本文根据基于蚁群算法的搜索原理，不断调整搜索起点 A 到终点 B 的满足优化约束条件的路径，并根据正反馈机制，最终找到从起点 A 到终点 B，均匀经过水平和垂直校正节点的最短路径，实现飞行器航迹规划求解。

蚁群算法原理可以表述如下：

Step1：路径构建。

在算法的初始时刻，将 m 只蚂蚁随机的放在 n 个城市中，同时，将每只蚂蚁的搜索禁忌表 tabu 的第一个元素设置为它当前所在城市。此时各路径上的信息素量相等，设$\tau_{ij}(0) =$

c,（c为一个较小的常数），接下来，每只蚂蚁根据路径上残留的启发式信息（两城市间的距离）独立的选择下一座城市，在时刻 t，蚂蚁 k 从城市 i 转移到城市 j 的概率$p^k{}_{ij}(t)$ 为：

$$p^k{}_{ij}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}\cdot[\gamma_{ij}(t)]^{\beta}}{\sum_{s\epsilon J_k(i)}[\tau_{ij}(t)]^{\alpha}\cdot[\gamma_{ij}(t)]^{\beta}}, & \text{当 j}\epsilon J_k(i)\\ 0, & \text{其他}\end{cases} \qquad (4—23)$$

式中，$J_k(i)=\{1,2,\cdots,n\}-tabu_k$表示蚂蚁 k 下一步允许选择的城市集合。禁忌表$tabu_k$记录了蚂蚁 k 当前走过的城市。当蚂蚁走到终点 B 时，蚂蚁 k 便完成了一次周游。γ_{ij}是一个启发式因子，代表蚂蚁从城市 i 到城市 j 的期望大小，常取距离的倒数，α、β分别代表信息素和期望启发式因子的相对重要程度。

信息素更新：

$$\tau_{ij}(t+n)=(1-\rho)\cdot\tau_{ij}(t)+\Delta\tau_{ij} \qquad (4-24)$$

其中，ρ（$0<\rho<1$）表示路径上信息素的蒸发系数，$1-\rho$表示信息素的持久系数；$\Delta\tau_{ij}$表示本次迭代中边ij上信息素的增量：

$$\Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij} \qquad (4-25)$$

其中

$$\Delta^k\tau_{ij}=\begin{cases}\dfrac{Q}{L_k}, & \text{第 k 只蚂蚁在次迭代中留在边 ij上的信息量}\\ 0, & \text{其他}\end{cases} \qquad (4-26)$$

其中，Q为正常数，L_k表示第k只蚂蚁在本次周游中所走过路径的长度。

基本蚁群算法的具体实现步骤如下：

Step1参数初始化。令时间$t=0$和循环次数$N_c=0$，设置最大循环次数G，将将 m 只蚂蚁放在 n 个元素上，令有向图每条边(i, j)的初始化信息量$\tau_{ij}(t)=c$，其中c表示常数，且初始时刻$\Delta\tau_{ij}(0)=0$。

Step2：循环次数$N_c=N_c+1$。

Step3：蚂蚁的禁忌表索引号$K=1$。

Step4：蚂蚁数量$K=k+1$。

Step5：蚂蚁根据转移概率（3—5）式，选择元素j前进，j$\epsilon\{J_k(i)\}$。

Step6：修改禁忌表指针，即选择好之后将蚂蚁移动到新的元素，并把该元素移动到该蚂蚁个体的禁忌表中。

Step7：若蚂蚁未到终点 B，则跳转到第Step4步；否则执行Step8。

Step8：记录本次最佳路线。

Step9：根据式 $(3-6)$和$(3-7)$ 更新每条路径上的信息量。

Step10：若满足条件，即如果循环次数$N_c \geq G$，则循环结束并输出程度优化结果，否则清空禁忌表跳转Step2

蚁群算法的运算流程如图所示：

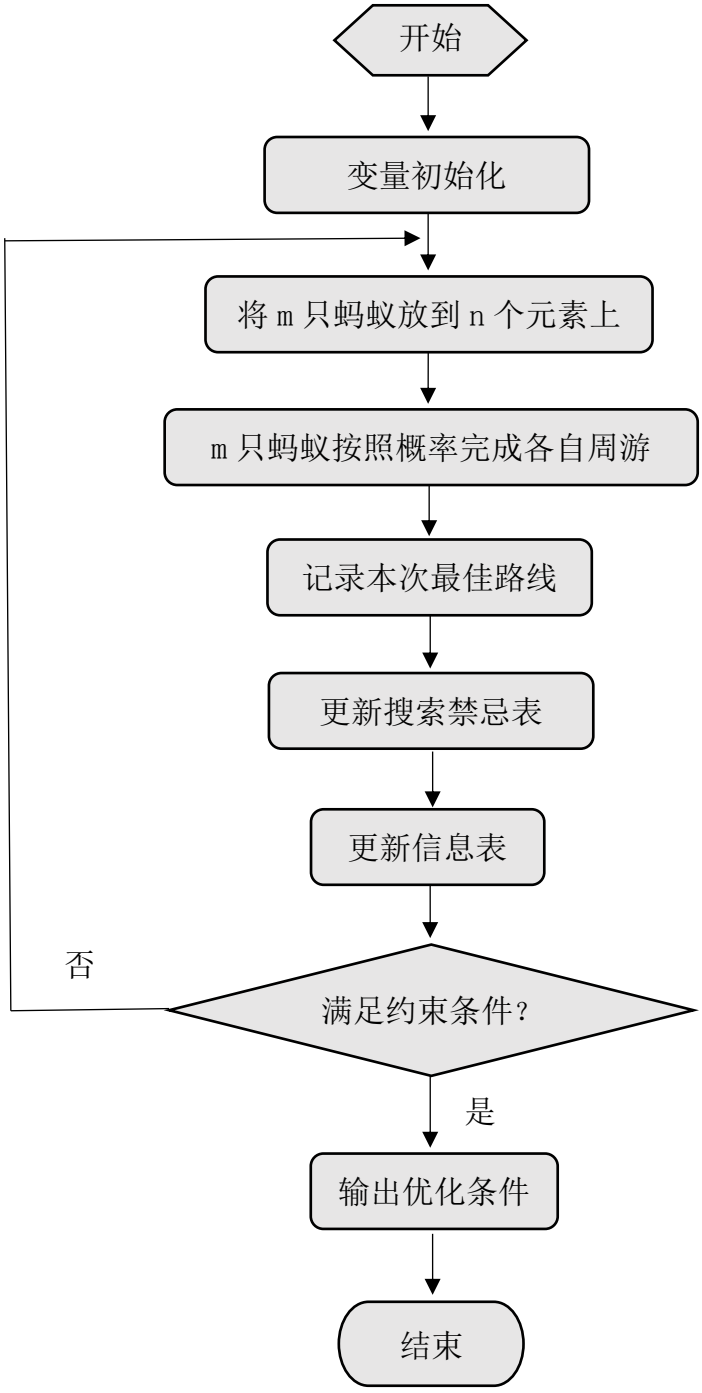

图 4-4：自适应改进型 Dijkstra 算法流程图

4.4 模型求解及结果分析

根据以上建立的多目标优化模型和自适应改进型算法，得到结果如下（程序见附件）。

附件一：

附件一中共有 610 个校正节点，其中水平校正节点 306 个，垂直校正节点 305 个。在通过自适应改进 Dijkstra 算法和蚁群算法得到的最优航行轨迹的基础上，通过贪婪便利算法得到一条从起始点 A 到终点 B 的最优路径长度、最佳校正节点数的航迹规划路径为：起点 A → 503 → 69 → 237 → 155 → 338 → 457 → 555 → 436 → 终点 B。则多目标规划的值分别为：

$$\min\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}\cdot s_{ij}=104.9\times10^{3}\mathrm{m}，\ \min\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}=8。$$

即在问题一的七个约束条件（1）~（7）下，飞行器最短的航迹长度为$104.9\times10^{3}m$；经过校正区域进行校正的次数为8次。飞行器在飞行过程中经过的校正节点编号、校正前垂直误差、校正前水平误差和校正点类型如下表所示：

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	13.388	13.388	11
69	8.8073	22.195	01
237	21.307	12.499	11
155	11.201	23.7	01
338	23.387	12.186	11
457	12.813	24.999	01
555	24.503	11.69	11
436	7.3559	19.046	01
612	22.314	14.958	终点B

表 4-1：航迹规划路径(附件一)

该路径经过 8 个校正点，总路程为 104.9 公里，算法迭代次数为 425 次，执行时间为 23.25S。飞行器的最优航迹规划路径如下图所示：

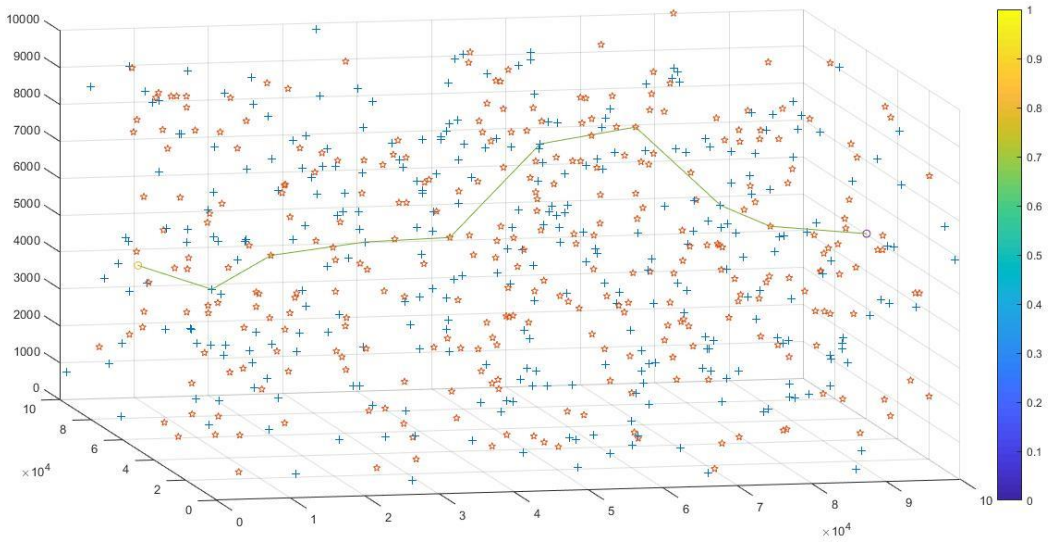

图 4-5：航迹规划路径(附件一)

附件二：

附件二中共有 325 个校正节点，其中水平校正节点 167 个，垂直校正节点 158 个。在通过自适应改进 Dijkstra 算法和蚁群算法得到的最优航行轨迹的基础上，通过贪婪便利算法综合实际情况得到一条从起始点 A 到终点 B 的最优路径长度、最佳校正节点数的航迹规划路径为：起点 A → 163 → 114 → 8 → 309 → 305 → 123 → 45 → 160 ⟶ 92 → 93 ⟶ 61 ⟶ 292 ⟶ 终点 B。则多目标规划的值分别为：

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} \cdot s_{ij} = 109.34 \times 10^3 \mathrm{m}，\ \min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} = 12。$$

即在问题一的七个约束条件（1）~（7）下，飞行器最短的航迹长度为$109.34 \times 10^3 \mathrm{m}$；经过校正区域进行校正的次数为12次。飞行器在飞行过程中经过的校正节点编号、校正前垂直误差、校正前水平误差和校正点类型如下表(4-2)所示：

校正点编号	校正后垂直误差	校正后水平误差	校正点类型
0	0	0	出发点A
163	13.288	13.288	01
114	18.62205012	5.334148463	11
8	13.92198957	19.25613803	01
309	19.44630952	5.524319956	11
305	5.96871847	11.49303843	01
123	15.1731167	9.204398231	11
45	10.00615708	19.21055531	01
160	17.49129148	7.485134406	11
92	5.776165548	13.26129995	01
93	15.26088101	9.484715466	11
61	9.834210374	19.31892584	01
292	16.38812355	6.55391318	11
326	6.960509412	13.51442259	终点B

表 4-2 ：航迹规划路径(附件二)

该路径经过 12 个校正点，总路程为 109.34 公里。算法迭代次数为 92 次，执行时间为 42.23S。

飞行器的最优航迹规划路径如下图所示：

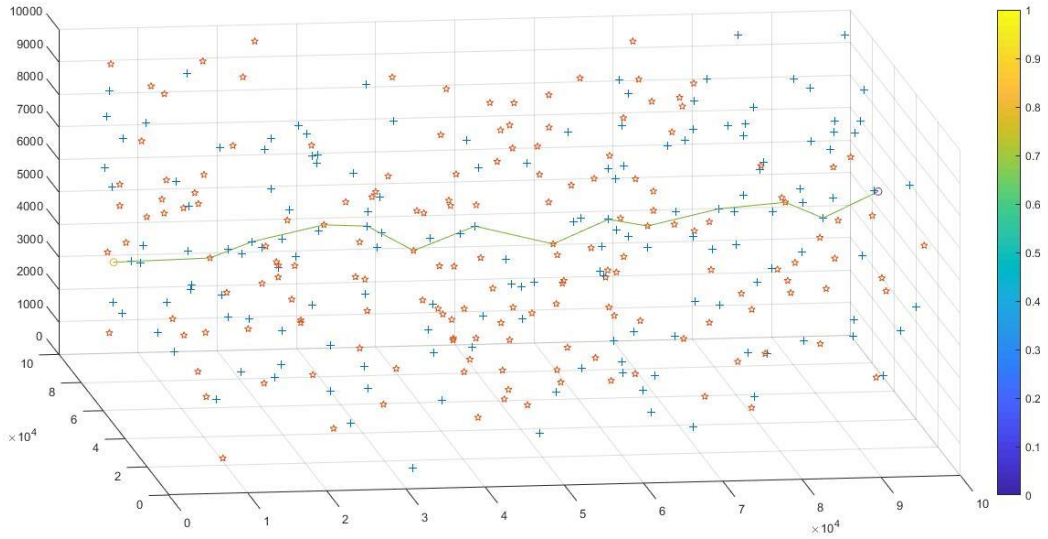

图 4-6：航迹规划路径(附件二)

五、问题二

5.1 问题描述与分析

问题二是在问题一的飞行误差补偿约束条件的基础上，考虑飞行器在实际飞行过程中的 200 米的最小转弯半径，实现航行轨迹的重新规划。要针对附件 1 和附件 2 中的校正节点位置和校正节点类型，从 A 点出发到达目的地 B 点，建模规划满足约束条件（1）~（8）时飞行器的航行轨迹，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，并讨论算法的有效性和复杂度。

约束条件中的误差补偿阈值同问题一。

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：

$$\alpha_1 = 20，\ \alpha_2 = 10，\ \beta_1 = 15，\ \beta_2 = 20，\ \theta = 20，\ \delta = 0.001$$

问题二的目的也是在使得航迹路程尽可能小的同时，经过校正区域进行校正的次数尽可能少，即为多目标动态规划问题。根据题目和参考文献，给出新的飞行器轨迹规划策略如下：

（1）增加考虑飞行器在实际飞行中有最小转弯半径约束，在问题一中飞行器能够直线到达的校正节点，问题二中飞行器需要有一个飞行方向调整，最小转弯半径轨迹为一个半径为 200m 的弧。

（2）当两个校正节点之间的距离s_{ij}与最小转弯半径 200m 的相差不大的情况下，飞行器在飞行过程中很难以接近距离s_{ij}的飞行路程到达下一个校正节点。

（3）飞行器在飞行过程中需要满足飞行路程最短的同时，尽可能经过少的飞行误差校正节

点。我们通过建模选择最优路线和最佳误差点数量和。
（4）根据最优路线和最佳误差点数量和带入实际问题中综合考虑多目标规划问题的最优解。
（5）应用不同算法，应用自适应改进型 Dijkstra 算法与基于蚁群寻优的智能优化蚁群算法在解决组合优化问题，通过调整数据，应用贪婪算法对航行轨迹进一步优化。

5.2 模型建立

5.2.1 飞行器转弯过程分析

问题二是在问题一的飞行误差补偿约束条件的基础上，增加考虑实际飞行过程中，飞行器在转弯过程中存在 200 米的最小转弯半径约束。一方面在问题一中飞行器能够直线到达的校正节点，问题二中飞行器需要有一个飞行方向调整，最小转弯半径轨迹为一个半径为 200m 的弧。如图所示：

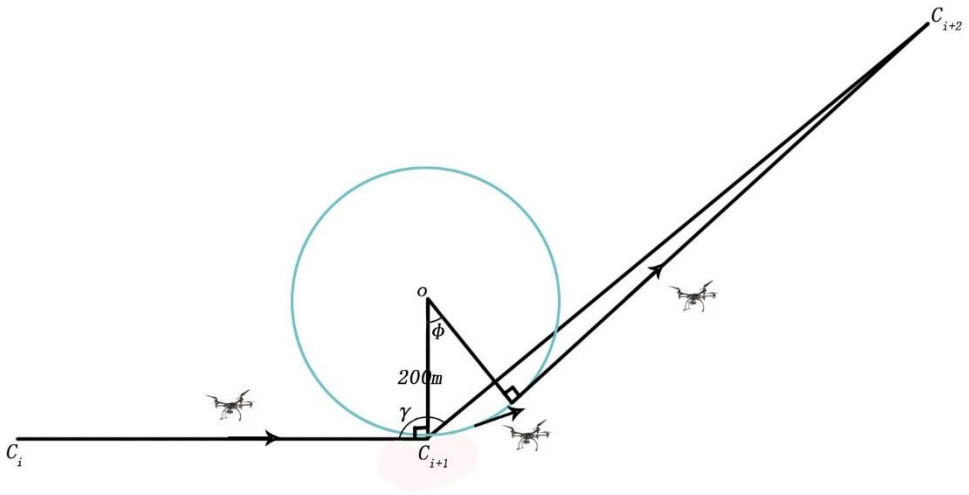

图 5-1：转弯轨迹图一

图中，飞行器从校正节点c_i到校正节点c_{i+1}，再到校正节点c_{i+2}。在实际情况中，当校正节点c_i到c_{i+1}的飞行长度$s_{i,\ i+1}$，再从c_{i+1}到c_{i+2}的飞行长度$s_{i+1,\ i+2}$满足：

$$s_{i+1,\ i+2} \geq 2 \times 200 \times \sin\frac{\varphi}{2} \qquad (5-1)$$

时，飞行器能够按照 200m 的最小转弯半径调整飞行方向，到达校正节点c_{i+2}。式中角φ为飞行方向最小调整角度。

边$b_{i,\ i+1}$和$b_{i+1,\ i+2}$的夹角γ有：

$$\gamma = \arccos\frac{{s_{i,\ i+1}}^2 + {s_{i+1,\ i+2}}^2 - {s_{i,\ i+2}}^2}{2 \cdot s_{i,\ i+1} \cdot s_{i+1,\ i+2}} \qquad (5-2)$$

在实际情况中，当$s_{i+1,\ i+2} \gg 200\text{m}$时，边$b_{i+1,\ i+2}$和飞行器调整完成的飞行方向之间的夹角很小，则有：

$$\varphi \approx 180^\circ - \gamma \qquad (5-3)$$

此时，飞行器从校正节点c_{i+1}到c_{i+2}的最短路程为半径为 200m 的圆的一段弧长和方向调整完成后的直线距离，即为：

$$s_{i+1,\ i+2}=\frac{\varphi}{360^{\circ}}\times 2\pi 200+\sqrt[2]{[(s_{i+1,\ i+2}-200)\times sin\varphi]^2-(200-200cos\varphi)^2} \quad (5-4)$$

综合考虑飞行方向的调整，飞行器从校正节点c_i到校正节点c_{i+2}飞行轨迹为先从沿最小转弯半径200m的轨迹飞行，当方向调整到校正节点c_{i+2}的方向时，飞行器将按照直线飞行。

另一方面，当两个校正节点之间的距离s_{ij}与最小转弯半径200m的相差不大的情况下，飞行器在飞行过程中难以接近距离s_{ij}的飞行路程到达下一个校正节点。如图5-3所示：

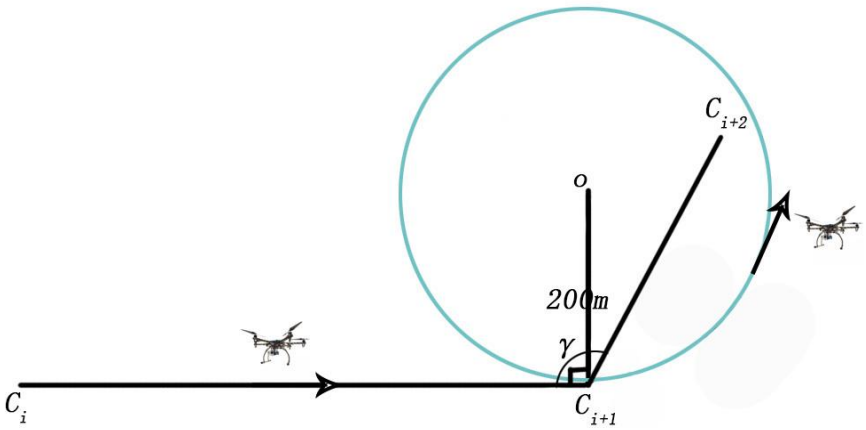

图5-2：转弯轨迹图二

图中，飞行器从校正节点c_i到校正节点c_{i+1}，再到校正节点c_{i+2}。在实际情况中，当校正节点c_i到c_{i+1}的飞行长度$s_{i,\ i+1}$，再从c_{i+1}到c_{i+2}的飞行长度$s_{i+1,\ i+2}$满足：

$$s_{i+1,\ i+2}<2\times 200\times sin\frac{\varphi}{2} \quad (5-5)$$

时，飞行器将不再能够按照200m的最小转弯半径调整飞行方向。在这种情况下，飞行器将按照更大的转弯半径，绕更大的弧形轨迹并中转更多的校正节点才能到达校正节点c_{i+2}，不满足最优的路径和最佳的误差补偿点数。本文假设此种情况为不能够到达。

5.2.2 最小转弯半径约束

飞行器在实际飞行过程中，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)，飞行器在从一个误差校正点到下一个误差校正点的过程中需要调整飞行方向，具体情况如以上分析。则飞行器的最小转弯半径约束为：

$$x_{ij}=\begin{cases}1, & \text{当飞行器从校正点}c_i\text{到校正点}c_j,\ s_{ij}\geq 2\times 200\times \sin\varphi\\ 0, & \text{当飞行器未从校正点}c_i\text{到校正点}c_j,\ s_{ij}<2\times 200\times sin\varphi\end{cases} \quad (5-6)$$

$$s_{i+1,\ i+2}=\begin{cases}\dfrac{\varphi}{360^{\circ}}\times 2\pi 200+\sqrt[2]{(200-200cos\varphi)^2+(s_{ij}\cdot 200sin\varphi)^2}, & x_{ij}=1\\ s_{ij}, & x_{ij}=0\end{cases} \quad (5-7)$$

综上所述，飞行器飞行航迹的多目标规划为：

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}\cdot s_{ij}$$

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} \qquad (5-8)$$

$$\text{s.t}\begin{cases} 0-1\text{变量}x_{ij}=\begin{cases}1,\quad \text{当飞行器从起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ \text{i}\neq\text{j}\\ 0,\ \text{当飞行器未从起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ \text{i}\neq\text{j}\end{cases} \\ \text{P}_{iv}+x_{ij}s_{ij}\delta\le\alpha_1,\ \text{v}_j\in\text{V} \\ \text{P}_{iu}+x_{ij}s_{ij}\delta\le\alpha_2,\ \text{v}_j\in\text{V} \\ \text{P}_{iv}+x_{ij}s_{ij}\delta\le\beta_1,\ \text{u}_j\in\text{U} \\ \text{P}_{iu}+x_{ij}s_{ij}\delta\le\beta_2,\ \text{u}_j\in\text{U} \\ \text{P}_{iu}+x_{iB}s_{iB}\delta\le\theta,\text{B 为终点} \\ \text{P}_{iv}+x_{iB}s_{iB}\delta\le\theta,\text{B 为终点} \\ x_{ij}=\begin{cases}1,\quad \text{当飞行器从校正点}c_i\text{到校正点}c_j,\ s_{ij}\ge 2\times 200\times\sin\varphi\\ 0,\ \text{当飞行器未从校正点}c_i\text{到校正点}c_j,\ s_{ij}<2\times 200\times sin\varphi\end{cases} \\ s_{ij}=\begin{cases}\dfrac{\varphi}{360^\circ}\times 2\pi 200+\sqrt[2]{(200-200cos\varphi)^2+(s_{ij}\cdot 200sin\varphi)^2},\ x_{ij}=1\\ s_{ij},\ x_{ij}=0\end{cases}\end{cases}$$

5.3 算法流程

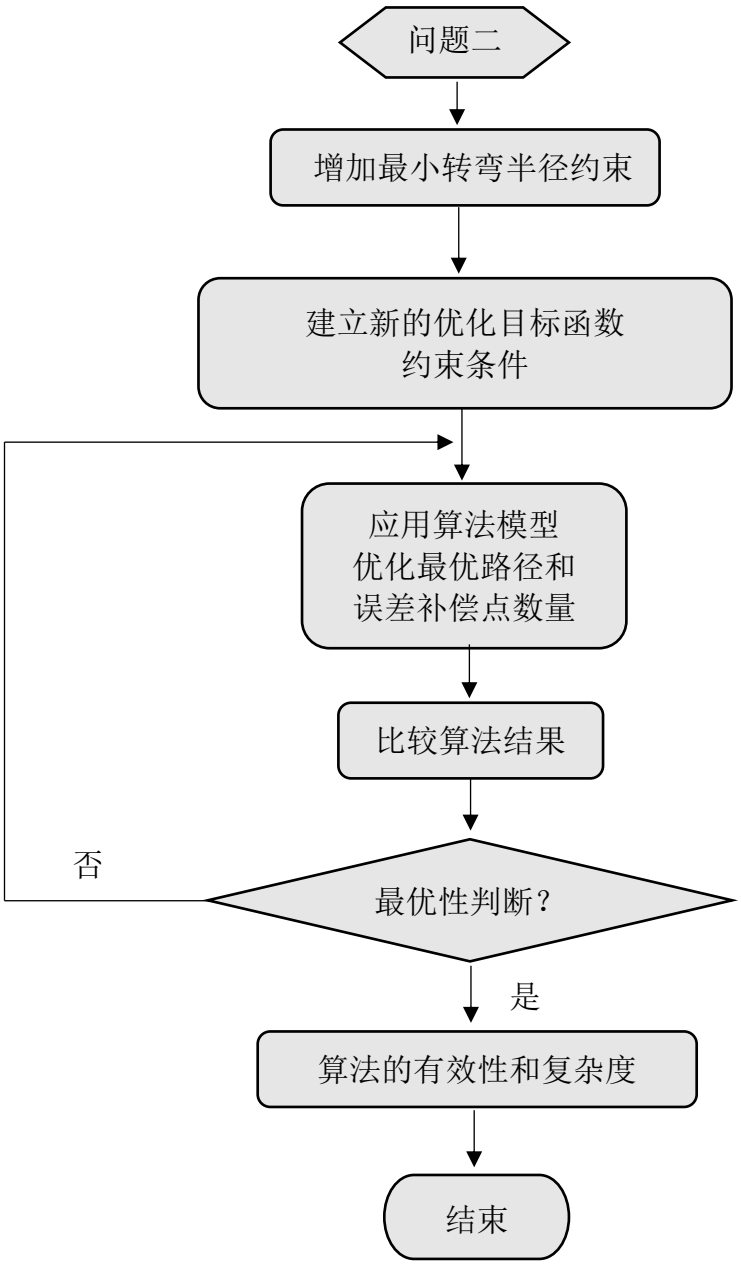

图 5-3：算法流程图

5.4 模型求解

根据以上建立的多目标优化模型，应用问题一中的自适应改进型算法，得到结果如下。

附件一：

通过应用问题一中的自适应改进 Dijkstra 算法和蚁群算法得到的最优航行轨迹的基础上，通过贪婪便利算法综合考虑实际情况得到一条从起始点 A 到终点 B 的最优路径长度、最佳校正节点数的航迹规划路径为：起点 A → 503 → 69 → 237 → 155 → 338 → 457 → 555 → 436 → 终点 B。则多目标规划的值分别为：

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} \cdot s_{ij} = 104.903 \times 10^{3}\text{m}，\ \min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} = 8。$$

即在问题二的八个约束条件（1）~（8）下，飞行器最短的航迹长度为$104.903 \times 10^3 m$；经过校正区域进行校正的次数为8次。飞行器在飞行过程中经过的校正节点编号、校正前垂直误差、校正前水平误差和校正点类型如下表所示：

校正点编号	校正后垂直误差	校正后水平误差	校正点类型
0	0	0	出发点A
503	13.388	13.388	11
69	8.808406697	22.1963267	01
237	21.30917535	12.50076865	11
155	11.20102786	23.70179651	01
338	23.3871117	12.18608384	11
457	12.81427133	25.00035517	01
555	24.50487141	11.69060008	11
436	7.356435741	19.04703582	01
612	22.31432013	14.95788439	终点B

表 5-1：航迹规划路径(附件一)

该路径经过 8 个校正点，总路程为 104.903 公里。算法迭代次数为 425 次，执行时间为 23.25S。

飞行器的最优航迹规划路径如下图所示：

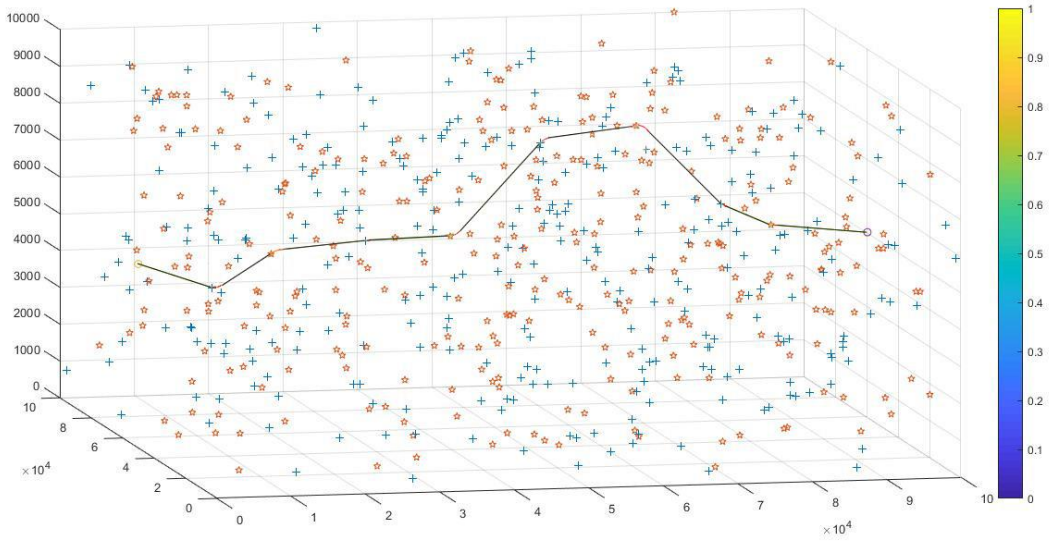

图 5-4：航迹规划路径(附件一)

附件二：

针对附件二中的校正节点坐标的类型数据，通过应用问题一中的自适应改进 Dijkstra 算法和蚁群算法得到的最优航行轨迹的基础上，通过贪婪便利算法综合考虑实际情况得到一条从起始点 A 到终点 B 的最优路径长度、最佳校正节点数的航迹规划路径为：

起点 A → 163 → 114 → 8 → 309 → 305 → 123 → 45 → 160 → 92 ⟶ 93 ⟶ 61 ⟶ 292 ⟶ 终点 B。多目标规划的值分别为：

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} \cdot s_{ij} = 109.464 \times 10^3 \mathrm{m}，\ \min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} = 12。$$

即在问题二的八个约束条件（1）~（8）下，飞行器最短的航迹长度为$109.464m$；经过校正区域进行校正的次数为 12 次。飞行器在飞行过程中经过的校正节点编号、校正前垂直误差、校正前水平误差和校正点类型如下表所示：

校正点编号	校正后垂直误差	校正后水平误差	校正点类型
0	0	0	出发点A
163	13.288	13.288	01
114	18.62933053	5.341428872	11
8	13.95683802	19.29826689	01
309	19.51972224	5.562884219	11
305	5.968775564	11.53165978	01
123	15.190511	9.221735437	11
45	10.00822457	19.22996001	01
160	17.49381934	7.485594772	11
92	5.776484714	13.26207949	01
93	15.26166264	9.485177925	11
61	9.834694415	19.31987234	01
292	16.3889462	6.55425179	11
326	6.980052408	13.5343042	终点B

表 5-2：航迹规划路径(附件二)

该路径经过 12 个校正点，总路程为 109.464 公里。算法迭代次数为 92 次，执行时间为 42.23S。

飞行器的最优航迹规划路径如下图所示：

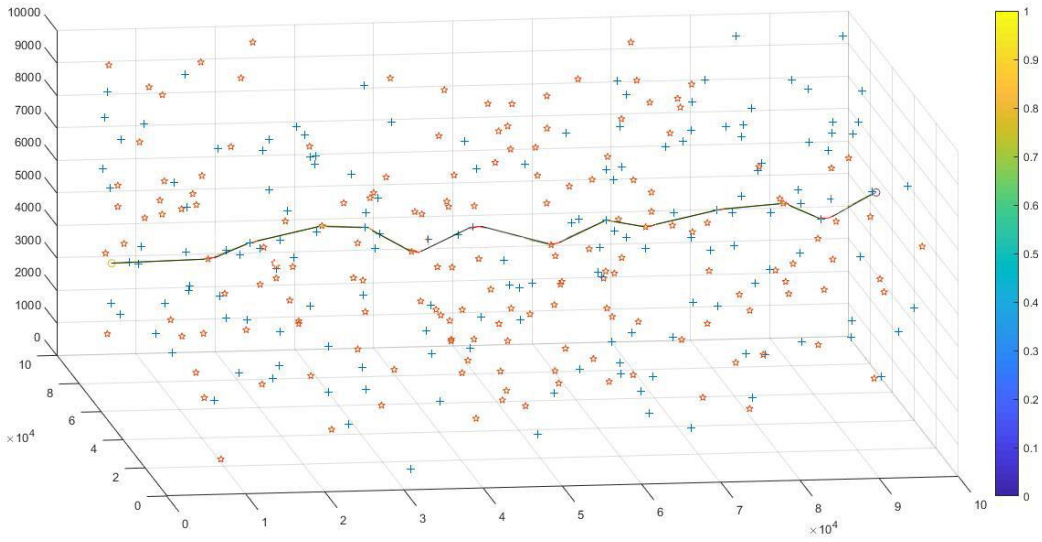

图 5-5：航迹规划路径(附件二)

六、问题三

6.1 问题描述与分析

问题三是在问题一的飞行误差补偿约束条件的基础上，考虑飞行器在实际飞行过程中飞行环境可能随时间动态变化的影响。虽然最优路径和最佳误差校正点在飞行前已经确定，但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。考虑部分故障校正

点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 $\min(P_i, 5)$的情况，重新规划问题一所要求的航迹，实现航行轨迹的重新规划。在综合考虑问题一的约束条件的情况下进行多目标航迹规划，优化目标如下：

（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；

（C）要求成功到达终点的概率尽可能大；

约束条件中的误差补偿阈值同问题一。

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：

$$\alpha_1 = 20，\alpha_2 = 10，\beta_1 = 15，\beta_2 = 20，\theta = 20，\delta = 0.001$$

问题三在增加部分可能发生误差校正故障点的基础上，重新规划航行轨迹使得航迹路程尽可能小、经过校正区域进行校正的次数尽可能少的同时，要求成功到达终点的概率尽可能大。即为多目标动态规划问题。根据题目和参考文献，给出新的飞行器轨迹规划策略如下：

（1）约束调整：增加考虑飞行器在实际飞行中经过误差校正故障点未能进行完全误差修正的条件约束。在问题一中飞行器能够直解到达的校正节点，问题三中飞行器需要在故障点按概率判断误差修正情况。

（2）考虑飞行器在飞行过程中，经过一个误差修正点发生故障，未能完全修正的情况下，依旧能够在当前飞行误差的条件下完成飞行目标的概率。

（3）飞行器在飞行过程中需要满足飞行路程短的同时，尽可能经过少的飞行误差校正节点。我们通过建模选择最优路线和最佳误差点数量。

（4）根据最优路线和最佳误差点数量和带入实际问题中，比较误差故障点和安全点，综合考虑多目标规划问题的最优解。

（5）应用不同算法，应用自适应改进型 Dijkstra 算法与基于蚁群寻优的智能优化蚁群算法在解决组合优化问题，通过调整数据，应用贪婪算法对航行轨迹进一步优化。

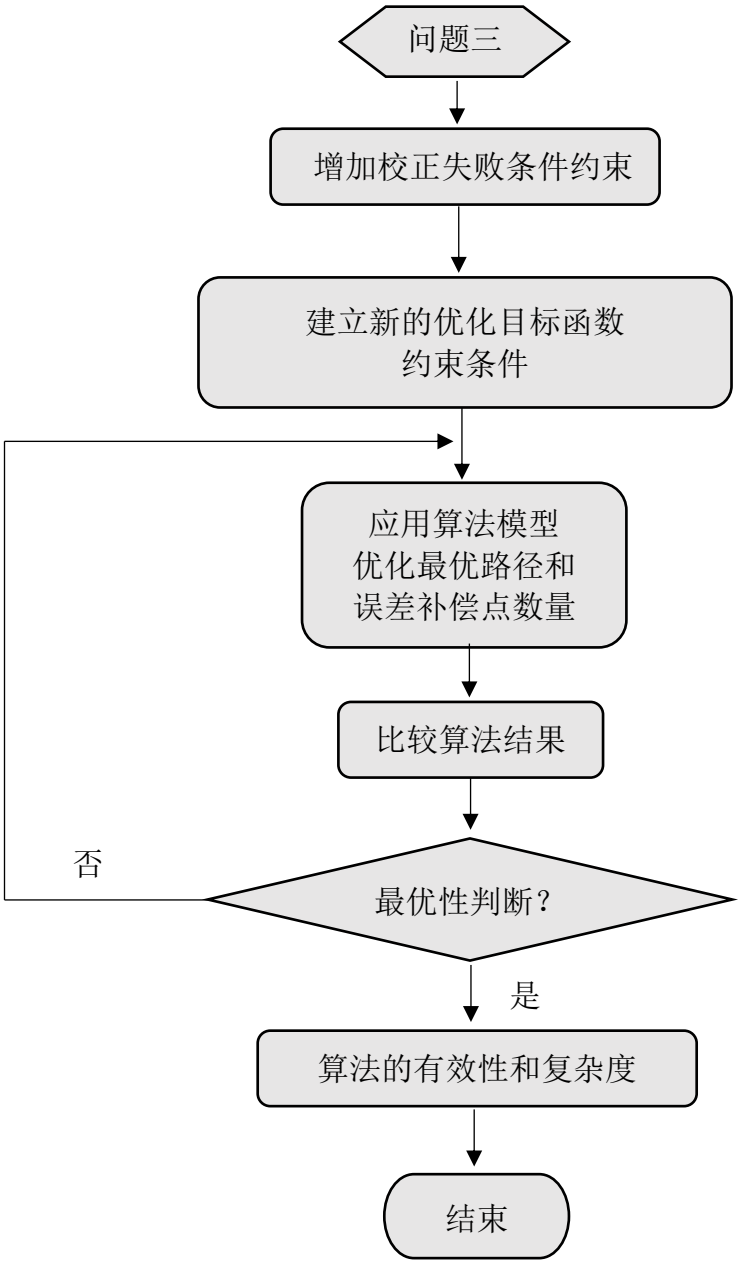

图 6-1：算法流程图

6.2 模型建立

6.2.1 误差校正情况分析

在飞行器的实际飞行过程中，飞行环境更加复杂多变。在问题一中经过的垂直或者水平误差校正点时，在满足误差修正阈值的条件下，均能够达到理想的修正结果。但是在问题三中存在部分不可控校正点存在 20%的不能完全修正的情况。分别根据附件一、附件二中的校正节点的故障节点标志，将故障校正点进行编号：

故障校正点集:$\mathrm{G}=[g_1,g_2,g_3\cdots,g_i,\cdots g_r]$, $\mathrm{i}=1,2,3\cdots,\mathrm{r}$，其中r为总故障校正点个数；

6.2.2 校正失败条件约束

问题三在问题一的基础上增加考虑飞行器在实际飞行中经过误差校正故障点未能进行完全误差修正的条件约束。在问题一中飞行器能够直接到达的校正节点，在问题三中飞行器需要在误差故障点按概率判断误差修正的实际情况，并根据剩余误差规划路径。

根据问题一中的约束条件分析有：

在任意一个校正点c_i，$\mathrm{i}=1,2,3\cdots\mathrm{n}$，加入误差校正故障点的约束后，剩余误差为：

$$\mathrm{P}_j=\begin{pmatrix}\mathrm{P}_{ju}\\ \mathrm{P}_{jv}\end{pmatrix}=\begin{pmatrix}\mathrm{P}_{iu}\\ \mathrm{P}_{iv}\end{pmatrix}+x_{ij}s_{ij}\delta\cdot\begin{pmatrix}1\\ 1\end{pmatrix}=\begin{cases}\begin{pmatrix}\mathrm{P}_{iu}+x_{ij}s_{ij}\delta\\ 0\end{pmatrix}, & \mathrm{v}_j\in\mathrm{V},\ \mathrm{c}_j\notin\mathrm{G}\\ \begin{pmatrix}0\\ \mathrm{P}_{iv}+x_{ij}s_{ij}\delta\end{pmatrix}, & \mathrm{u}_j\in\mathrm{U},\ \mathrm{c}_j\notin\mathrm{G}\\ \begin{pmatrix}\mathrm{P}_{iu}+x_{ij}s_{ij}\delta\\ \min(\mathrm{P}_{jv},\ 5)\end{pmatrix}, & \mathrm{v}_j\in\mathrm{V},\mathrm{c}_j\in\mathrm{G}\\ \begin{pmatrix}\min(\mathrm{P}_{ju},\ 5)\\ \mathrm{P}_{iv}+x_{ij}s_{ij}\delta\end{pmatrix}, & \mathrm{u}_j\in\mathrm{U},\mathrm{c}_j\in\mathrm{G}\end{cases}\qquad(6-1)$$

当要求到达终点时垂直误差和水平误差均应小于θ个单位 ，即：

$$\mathrm{P}_{iu}+x_{iB}s_{iB}\delta\le\theta\text{且}\mathrm{P}_{iv}+x_{iB}s_{iB}\delta\le\theta\qquad(6-2)$$

⑵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，即：

$$\mathrm{P}_{iv}+x_{ij}s_{ij}\delta\le\alpha_1\ ,\ \mathrm{v}_j\in\mathrm{V}\qquad(6-3)$$

$$\mathrm{P}_{iu}+x_{ij}s_{ij}\delta\le\alpha_2,\ \mathrm{v}_j\in\mathrm{V}\qquad(6-4)$$

附件一中$\alpha_1=25$，$\alpha_2=15$, 附件二中$\alpha_1=20$，$\alpha_2=10$。

⑶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。

$$\mathrm{P}_{iv}+x_{ij}s_{ij}\delta\le\beta_1\ ,\ \mathrm{u}_j\in\mathrm{U}\qquad(6-5)$$

$$\mathrm{P}_{iu}+x_{ij}s_{ij}\delta\le\beta_2,\ \mathrm{u}_j\in\mathrm{U}\qquad(6-6)$$

附件一中$\beta_1=20$，$\beta_2=25$，附件二中$\beta_1=15$，$\beta_2=20$。

⑷飞行器在飞行过程中会根据最优路径和节点数选择经过水平误差补偿和垂直误差补偿点，在算法的实际计算中常常需要直接约束任意两个水平或者垂直误差的阈值，约束条件如下所示：

$$\boldsymbol{l}_{v_{k-1},v_k}\cdot\delta\le\alpha_1\qquad(6-7)$$

$$\boldsymbol{l}_{C_{\max\{i|C_i\notin V,i<ord(v_k)\}},v_k}\cdot\delta\le\alpha_2\qquad(6-8)$$

$$\boldsymbol{l}_{u_{k-1},u_k}\cdot\delta\le\beta_1\qquad(6-9)$$

$$\boldsymbol{l}_{C_{\max\{i|C_i\notin V,i<ord(v_k)\}},v_k}\cdot\delta\le\beta_2 \qquad (6-10)$$

问题三的目标是在增加部分可能发生误差校正故障点的基础上，规划航行轨迹使得航迹路程尽可能小、经过校正区域进行校正的次数尽可能少的同时，要求成功到达终点的概率尽可能大。则在问题一的基础上增加成功到达终点的概率目标函数为：

$$\min \mathrm{w}=\mathrm{w}_{ij}{}^{\mathrm{q}} \qquad (6-11)$$

式中，w为经过误差校正故障点的路径能够成功到达终点的概率；w_{ij}为无人机在c_j点误差校正成功的概率，则根据题目有：$\mathrm{w}_{ij}=80\%$。q表示飞行途中一旦校正失败，就会导致无法到达 B 点的可能出现故障的校正点数目。

综上所述，飞行器飞行航迹的多目标规划为：

$$\begin{cases}\min\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}\cdot s_{ij}\\ \min\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}\\ \min \mathrm{w}=\mathrm{w}_{ij}{}^{\mathrm{q}}\end{cases} \qquad (4-12)$$

$$\text{s.t}\begin{cases}0-1\text{ 变量}x_{ij}=\begin{cases}1,\ \text{当飞行器从起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ \mathrm{i}\neq\mathrm{j}\\0,\ \text{当飞行器未从起点 or 校正点}c_i\text{到终点 or 校正点}c_j,\ \mathrm{i}\neq\mathrm{j}\end{cases}\\ \mathrm{P}_{iv}+x_{ij}s_{ij}\delta\le\alpha_1,\ \mathrm{v}_j\in\mathrm{V}\\ \mathrm{P}_{iu}+x_{ij}s_{ij}\delta\le\alpha_2,\ \mathrm{v}_j\in\mathrm{V}\\ \mathrm{P}_{iv}+x_{ij}s_{ij}\delta\le\beta_1,\ \mathrm{u}_j\in\mathrm{U}\\ \mathrm{P}_{iu}+x_{ij}s_{ij}\delta\le\beta_2,\ \mathrm{u}_j\in\mathrm{U}\\ \mathrm{P}_{iu}+x_{iB}s_{iB}\delta\le\theta,\mathrm{B}\text{ 为终点}\\ \mathrm{P}_{iv}+x_{iB}s_{iB}\delta\le\theta,\mathrm{B}\text{ 为终点}\end{cases}$$

6.3 模型求解

根据以上建立的多目标优化模型，应用问题一中的自适应改进型算法，得到结果如下。

附件一：

通过应用问题一中的自适应改进 Dijkstra 算法和蚁群算法得到的最优航行轨迹的基础上，通过贪婪便利算法在综合考虑实际情况，得到一条从起始点 A 到终点 B 的最优路径长度、最佳校正节点数的航迹规划路径为：起点 A → 503 → 294 → 91 → 607 → 61 → 250 → 369 → 566 → 400 ⟶ 终点 B。则多目标规划的值分别为：

$\min\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}\cdot s_{ij}=105.77\times10^3\mathrm{m}$，$\min\sum_{i=1}^{n}\sum_{j=1}^{n}x_{ij}=9$，$\min \mathrm{w}=\mathrm{w}_{ij}{}^{\mathrm{q}}=1$。

即在问题三的七个约束条件（1）~（7）下，飞行器最短的航迹长度为105.77×10^3m；经过校正区域进行校正的次数为 9 次；成功概率为 100%。飞行器在飞行过程中经过的校

正节点编号、校正前垂直误差、校正前水平误差和校正点类型如下表所示：

校正点编号	校正后垂直误差	校正后水平误差	校正点类型
0	0	0	出发点A
503	13.388	13.388	11
294	10.18080701	23.56872701	01
91	17.535509	7.354701996	11
607	8.353022313	15.70772431	01
61	16.50631475	8.153292433	11
250	16.12382364	24.27711607	01
369	18.14644728	2.022623639	11
566	14.51248202	16.53510566	01
400	17.87590386	3.363421844	11
612	22.31888741	25.68230925	终点B

表 6-1：航迹规划路径(附件一)

该路径经过 9 个校正点，总路程为 105.77 公里。算法迭代次数为 586 次，执行时间为 13.03S。

飞行器的最优航迹规划路径如下图所示：

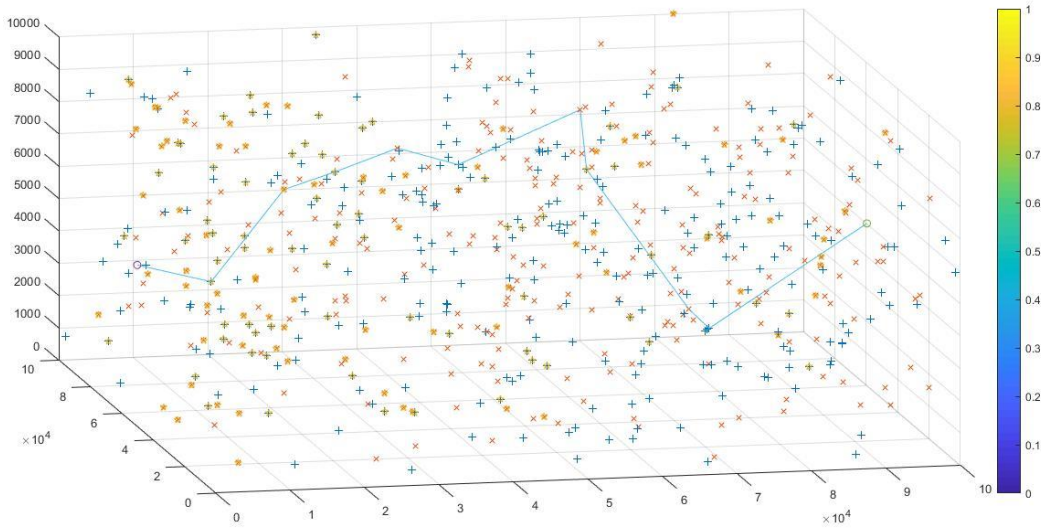

图 6-2：航迹规划路径(附件一)

附件二：

针对附件二中的校正节点坐标的类型数据，通过应用问题一中的自适应改进 Dijkstra 算法和蚁群算法得到的最优航行轨迹的基础上，通过贪婪便利算法综合考虑实际情况得到一条从起始点 A 到终点 B 的航迹规划路径一为：

起点 A → 140 → 226 → 288 → 306 → 237 → 280 → 65 → 142 → 310 ⟶ 7 ⟶ 145 ⟶ 293 ⟶ 156 ⟶ 213 ⟶ 164 ⟶ 50 ⟶ 247 ⟶ 38 ⟶ 110 ⟶ 99 ⟶ 292 ⟶ 终点 B。多目标规划的值分别为：

$\min \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} \cdot s_{ij} = 155.07 \times 10^3 \mathrm{m}$，$\min \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} = 12$，$\min \mathrm{w} = \mathrm{w}_{ij}{}^{\mathrm{q}} = 1$。

即在问题三的七个约束条件（1）~（7）下，飞行器最短的航迹长度为155.07×10^3m；经过校正区域进行校正的次数为 21 次，成功概率为 100%。飞行器在飞行过程中经过的校正节点编号、校正前垂直误差、校正前水平误差和校正点类型如下表(5-2)所示：

针对附件二中的校正节点坐标的类型数据，算法结果为：

校正点编号	校正后垂直误差	校正后水平误差	校正点类型
0	0	0	出发点A
140	5.6558	5.6558	11
226	11.118361	16.77411907	01
288	13.86700288	2.748641876	11
306	5.744628189	8.493270065	01
237	11.43769045	5.693062258	11
280	6.324475645	12.0175379	01
65	10.70270496	4.378229318	11
142	9.717830215	14.09605953	01
310	14.55964303	4.841812814	11
7	12.8206075	17.66242032	01
145	18.38024624	5.559638737	11
293	12.42013874	17.97977748	01
156	19.33178376	6.911645014	11
213	9.136410069	16.04805508	01
164	12.96196	3.825549931	11
50	10.90687245	14.73242239	01
247	17.08333383	6.176461371	11
38	6.997345054	13.17380642	01
110	10.92388525	3.926540195	11
99	9.199885147	13.12642534	01
292	13.2023839	4.002498752	11
326	6.960509412	10.96300816	终点B

表 6-2：航迹规划路径(附件二)

该路径经过 21 个校正点，总路程为 155.07 公里。算法迭代次数为 1440 次，执行时间为 5.61S。

飞行器的最优航迹规划路径如下图所示：

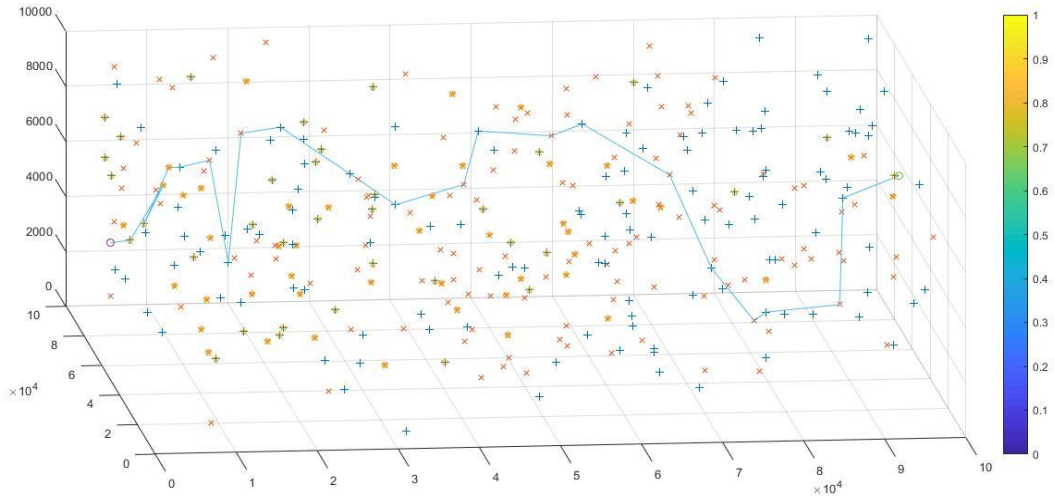

表 6-1：航迹规划路径(附件二)

根据上述结果，当飞行器的成功概率为100%时，飞行器需要飞行的航迹长度和需要经过的误差补偿点个数都相对问题一有了很大增加。

根据问题一中的算法，在适当放松成功率时，飞行器的最优航迹如下：

（1）当成功率降低到80%时：

从起始点A到终点B的航迹规划路径二为：

起点 A → 169 → 322 → 270 → 89 → 236 ⟶ 132 → 53 → 112 → 268 → 250 ⟶ 243 ⟶ 167 ⟶ 76 ⟶ 6 ⟶ 12 ⟶ 216 ⟶ 279 ⟶ 302 ⟶ 161 ⟶ 终点 B。

多目标规划的值分别为：

$\min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} \cdot s_{ij} = 158 \times 10^3 \mathrm{m}$，$\min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} = 19$，$\min \mathrm{w} = \mathrm{w}_{ij}{}^{\mathrm{q}} = 0.8$。

即在问题三的八个约束条件（1）~（8）下，飞行器最短的航迹长度为$158 \times 10^3 \mathrm{m}$；经过校正区域进行校正的次数为19次，成功概率为80%。

（2）当成功率降低到64%时：

从起始点A到终点B的航迹规划路径三为：

起点 A → 184 → 136 → 259 → 237 → 280 ⟶ 65 → 142 → 310 → 7 → 145 ⟶ 293 ⟶ 156 ⟶ 213 ⟶ 164 ⟶ 50 ⟶ 323 ⟶ 61 ⟶ 292 ⟶ 终点 B。

多目标规划的值分别为：

$\min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} \cdot s_{ij} = 146.61 \times 10^3 \mathrm{m}$，$\min \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} = 18$，$\min \mathrm{w} = \mathrm{w}_{ij}{}^{\mathrm{q}} = 0.64$。

即在问题三的八个约束条件（1）~（8）下，飞行器最短的航迹长度为$146.61 \times 10^3 \mathrm{m}$；经过校正区域进行校正的次数为18次，成功概率为64%。

因此综合上述结果来看，放松成功概率的情况下，飞行器航行轨迹长度和经过的节点数都有相应程度的下降，在考虑飞行成功率尽可能大的情况下，我们优先选择航迹路径一。但是在实际情况中，我们可以在相应成功率要求下，选择更优的飞行航迹路程和更佳的节点数，即航迹路径二、三。

七、参考文献

[1] 江泽强. 改进 RRT 算法在无人机航迹规划中的应用[J]. 计算机与数字工程, 2019, 47(5):1131-1135.
[2] 俞琪. 基于遗传算法的快速航迹规划方法研究[D]. 华中科技大学, 2011.
[3] 杨杰. 具有端点方向约束的快速航迹规划方法研究[D]. 华中科技大学, 2013.
[4] 李自杰. 改进粒子群算法在航迹规划中的应用[D]. 华中科技大学, 2012.
[5] 无人飞行器航迹规划算法研究[D]. 哈尔滨工程大学, 2010.
[6] 韩攀, 陈谋, 陈哨东,等. 基于改进蚁群算法的无人机航迹规划[J]. 吉林大学学报(信息科学版), 2013, 31(1):66-72.
[7] 孟祥恒, 王社伟, 陶军. 基于改进蚁群算法的多无人机航路规划研究[J]. 计算机仿真, 2008, 25(11):56-59.
[8] 吴蕊, 赵敏, 李可现. 基于改进蚁群算法的无人机协同航迹规划研究[J]. 电光与控制, 2011, 18(11):12-16.
[9] 彭文敏, 胡书, 张莉,等. 基于粒子群算法的无人机航迹规划问题[J]. 北京电子科技学院学报, 2009, 17(4):70-73.
[10] 刘平[1], 彭建亮[1]. 基于 Voronoi 图和离散粒子群优化的无人机航迹规划[J]. 现代导航, 2011, 02(6):412-416.

八、附录

```
%%%%%%%%%%%%%%%%%%%%%%%%%%

PATH=cell(10000,1);
SENDNEW=SSSNEW;

for i=1:10^4
  b=sparse(SENDNEW);
  [dist,path,pred] = graphshortestpath(b,1,613);


%起点 1,2,3,约束
if ismember(path(2),verticaldata)==1
   if S(path(1),path(2))<=25/0.001 & S(path(1),path(2))<=15/0.001
      if S(path(2),path(3))<=20/0.001 & ( S(path(1),path(2))+S(path(2),path(3)) ) <=25/0.001
      path=path;
      else
         SENDNEW(path(2),path(3))=0;SENDNEW(path(3),path(2))=0;
         %path=[];
         continue;
      end
   else
       SENDNEW(path(1),path(2))=0;SENDNEW(path(2),path(1))=0;
       %path=[];
       continue;
   end
elseif ismember(path(2),standarddata)==1
   if S(path(1),path(2))<=20/0.001 & S(path(1),path(2))<=25/0.001
      if S(path(2),path(3))<=15/0.001 &    (S(path(1),path(2))+S(path(2),path(3)) ) <=25/0.001
      path=path;
      else
         SENDNEW(path(2),path(3))=0;SENDNEW(path(3),path(2))=0;
         %path=[];
         continue;
      end
   else
       SENDNEW(path(1),path(2))=0;SENDNEW(path(2),path(1))=0;
       %path=[];
         continue;
   end
end
```

```
%中间点约束
flag=0;
for m=3:length(path)-1
   if ismember(path(m),verticaldata)==1
      if  (  S(path(m-2),path(m-1))  +  S(path(m-1),path(m))  )<=25/0.001  &  S(path(m-1),path(m))<15/0.001
      path=path;

      else
          SENDNEW(path(m-1),path(m))=0;SENDNEW(path(m),path(m-1))=0;
          %path=[];
          flag=1;
          break;

      end

   elseif ismember(path(m),standarddata)==1
      if  (S(path(m-2),path(m-1))  +  S(path(m-1),path(m)))<=25/0.001  &  S(path(m-1),path(m))<20/0.001
      path=path;
      else
          SENDNEW(path(m-1),path(m))=0;SENDNEW(path(m),path(m-1))=0;
          %path=[];
          flag=1;
          break;

      end
   end
end

if flag==1
   continue;
end

 %终点约束
 if (   S( path(size(path,2)-2),path(size(path,2)-1) )+S( path(size(path,2)-1),613 )   )<30/0.001
      path=path;
 else
      SENDNEW(613,path(size(path,2)-1))=0;
      SENDNEW(path(size(path,2)-1),613)=0;
      %path=[];
      continue;
 end
```

```
 %交叉约束
 flagbb=0;
 for j=2:length(path)-1
    if ( ismember(path(j),verticaldata) & ismember(path(j+1),verticaldata) ) |
( ismember(path(j),standarddata) & ismember(path(j+1),standarddata) )
        SENDNEW(path(j),path(j+1))=0;
        SENDNEW(path(j+1),path(j))=0;
        %path=[];
        flagbb=1;
        break;
      end
   end
if flagbb==1
 continue;
end



%拐弯问题 2 约束
%在误差较小时误差成功概率大
%一般角度是钝角
flagcc=0;
for nv=2:length(path)-1
   cosalpha=( (S(path(nv-1),path(nv)))^2+(S(path(nv),path(nv+1)))^2-(S(path(nv-
1),path(nv+1)))^2 )/( 2*S(path(nv-1),path(nv))*S(path(nv),path(nv+1)) );
   sinbeita=abs(cosalpha);
   if 200*sinbeita*2<=S(path(nv),path(nv+1))
      path=path;
   else
      SENDNEW(path(nv),path(nv+1))=0;SENDNEW(path(nv+1),path(nv))=0;
      flagcc=1;
      break;
   end
end
if flagcc==1
 continue;
end



%%%%%%%%%%%%%%%%%%故障点从 3 到 n-2 个点的约束
flagdd=0;
for ok=3:length(path)-3
```

```
 if ismember(path(ok),wrong_point)
    if S(path(ok),path(ok+1))+S(path(ok+1),path(ok+2))<=20/0.001
      path=path;
    else
      SENDNEW(path(ok),path(ok+1))=0;SENDNEW(path(ok+1),path(ok))=0;
      flagdd=1;
      break;
    end
   end
end
if flagdd==1
 continue;
end
%%%%%%%%%%%%%故障点在 n-2 点的约束
%if                                  S(path(length(path)-2),path(length(path)-1))+S(path(length(path)-
1),path(length(path)))<25/0.001
 % path=path;
%else
 %            SENDNEW(path(length(path)-2),path(length(path)-1))=0;SENDNEW(path(length(path)-
1),path(length(path)-2))=0;
   %continue;
%end


%最优路径的故障点保存
wrong_save=[];
for okj=1:length(path)
   if ismember(path(okj),wrong_point)
      wrong_save=[wrong_save,okj];%
   end
end


%能到达这里的 path 这里已经经过了重重考验
PATH{i}=path;
countt=0;
 for k=1:size(PATH,1)
   countt=countt+~isempty(PATH{k});
 end
 if countt~=0
    break;
 end
```

```
end
```