
参赛密码 _____

(由组委会填写)



“华为杯”第十四届中国研究生 数学建模竞赛

学 校 上海交通大学

参赛队号 10248256

1. 安慧丽

队员姓名 2. 吴毅

3. 闫飞

参赛密码 _____

(由组委会填写)



“华为杯”第十四届中国研究生 数学建模竞赛

题 目

无人机在抢险救灾中的优化作用

摘要：

本文以图论、最优化理论为基础，研究了无人机山区救灾协同任务规划问题。首先，在使用尽量少的无人机巡查的情况下使覆盖率最高，对问题一第一小问的目标区域建立**避障路径最优化模型**。其次，将问题一第二小问的目标区域分解为满足特定条件的正方形区域，建立**区域覆盖约束优化模型**，即在每个正方形区域内建立以搜寻面积与正方形面积差最小为目标的分区路径规划策略。接着，对于无人机生命迹象探测问题，将问题二中的目标区域分成两个部分，分别建立以覆盖面积最大为目标的多无人机协同探测的**路径优化模型**。然后，对于无人机通信中继问题，建立最小生成树模型，求解给出最少无人机数目。最后针对数据传输问题，提出并论证**数据传输自适应策略**，在此基础上建立规划模型并求解。

对问题一第一小问，分两步求解避障路径最优化模型。首先将 S 内海拔 3000 米以下的网格利用**聚类**方法提取出多个中心点作为巡查路径上的必经点，将问题转化为每个区域内的旅行商(TSP)优化问题，逐步判断无人机是否满足飞行时间的约束来决定是否增加架数。使用模拟退火算法(Anneal Algorithm)求解，最后解得三架无人机即可满足题意，且达到 **79.93%** 的覆盖率，飞行路径详见图 7、图 8、图 9。

对第一题第二小问，基于上一问中分区统筹的思想，主要分两步求解区域覆盖约束模型。首先，为了充分利用无人机的续航能力，每三架为一组、飞行路径相同，持续巡逻一个正方形区域；其次每个正方形内的飞行路径按照是否含有障碍物分成两种，不含有障碍物的区域按照“直线螺旋”搜索方式，详见图 12；含有障碍物的区域按照上一问的方法求解飞行路径，详见图 12。满足上述搜索

方案共需要 87 架无人机。

对问题二，基于聚类选点的思想，且考虑两个基地的位置，将路径关键点分成两类，继而多无人机协同探测的路径优化模型转化为两个区域的多旅行商问题，利用遗传算法求解，得到 30 个无人机的规划路径，详见[表 3](#)。求解得到：**J、H 基地时间间隔分别为 3.48 小时和 4.11 小时，总的巡查覆盖率为 84.2%**。最终对比 100 个和 150 个聚类中心得出覆盖率与最短时间间隔两个优化目标的关系。

对问题三，需要考虑终端与无人机、无人机与无人机两层的通信要求，先考虑终端与无人机的通信联接，再利用 Prim 算法求解最小生成树，检验并调整不满足通信距离的情况，得出**87 架无人机就可以保障灾区通信中继的结论**。飞行路径示意图见[图 19](#)。

对问题四，为简化题目复杂度，提出**数据传输自适应策略**。经分析发现 1500m 为数据传输最合适距离，一台无人机可只负责地面一个终端的数据传输，通过合理调整无人机的飞行速度及高度，建立以时间最小化为目标的规划问题，接着使用遗传算法求解得到无人机的最优路径，详见[图 21](#)，**最短时间总和为 9.71h**。

关键词：旅行商问题 聚类 模拟退火 遗传算法 最小生成树

目录

目录	3
一 问题背景与地形可视化	5
1.1 问题重述	5
1.2 研究现状	6
1.3 主要符号说明	6
1.4 地形可视化	7
二. 问题一的建模与求解	7
2.1 问题假设	7
2.2 问题（一）的分析与模型建立	8
2.2.1 问题（一）分析	8
2.2.2 问题（一）模型建立	8
2.3 问题（一）模型求解	10
2.4 问题（二）分析与模型建立	13
2.5 问题（二）模型求解	15
2.5.1 区域覆盖约束模型求解结果	15
2.5.2 模型 3 分区路径规划策略求解结果	17
2.6 模型评价	17
三. 问题二的建模与求解	19
3.1 问题假设	19
3.2 问题分析	19
3.3 模型建立	19
3.3.1 “探测区 S3”飞行路径的必经点选择及区域划分	19
3.3.2 无人机探测面积计算	20
3.3.3 无人机飞行时间间隔	20
3.4 模型求解	21
3.4.1 求解思路	21
3.4.2 结果 1 “探测区 S3”飞行路径的关键点选择及分区	21
3.4.3 结果 2 无人机路径规划、覆盖率及时间间隔	22
3.5 敏感度分析	24
3.6 模型评价	25
四. 问题三的建模与求解	26
4.1 问题重述与分析	26
4.2 模型建立	27
4.3 模型求解	27
4.4 模型评价	29
五. 问题四的建模与求解	30

5.1 模型假设	30
5.2 问题分析与模型建立	30
5.3 模型求解	32
5.4 模型评价	34
六. 总结与展望	35
6.1 总结	35
6.2 展望	35
七. 参考文献	36
八. 附录	37
8.1 附录 1	37
8.2 附录 2	44

一 问题背景与地形可视化

1.1 问题重述

无人机性能数据如下：平均飞行速度 60 千米/小时，最大续航时间为 8 小时，飞行时的转弯半径不小于 100 米，最大爬升(俯冲)角度为 $\pm 15^\circ$ ，最大飞行高度为海拔 5000 米。

为保证飞行安全，无人机与其它障碍物(含地面)的安全飞行距离不小于 50 米。所有无人机均按规划好的航路自主飞行，无须人工控制，完成任务后自动返回原基地。

问题一：灾情巡查

大地震发生后，及时了解灾区情况是制订救援方案的重要前提。本题要求使用无人机携带视频采集装置巡查 7 个重点区域中心方圆 10 公里(并集记为 S)以内的灾情。假设无人机飞行高度恒为 4200 米，俯视角大于 60° 且视线不被山体阻隔的点才能被巡查。所有无人机均从基地 H(110,0)(单位：千米)处派出，且完成任务后再回到 H。黄金救援时间为 4 小时，在该时间段内希望尽可能多地巡查区域 S 内海拔 3000 米以下的地方。设计方案，给出无人机架数，覆盖率，每架无人机飞行路线图及巡查到的区域(不同的无人机的飞行路线图用不同的颜色表示)。

进一步，为及时发现次生灾害，使用无人机在附件 1 给出的高度低于 4000 米的区域(不限于 S)上空巡逻。设计方案：在 72 小时内，上述被巡查到的地方相邻两次被巡查的时间间隔不大于 3 小时(无人机均从 H 出发并在 8 小时内回到 H，再出发的时间间隔不小于 1 小时)。给出方案所需的无人机架数、每架无人机的飞行时间和路线。

问题二：生命迹象探测

从基地 H(110,0), J(110,55)(单位：千米)处总共派出 30 架无人机(各 15 架)携带生命探测仪搜索生命迹象，任务完成后回到各自的出发地。探测仪的有效探测距离不超过 1000 米，且最大侧视角(探测仪到可探测处的连线与铅垂线之间的夹角)为 60 度。规划它们的飞行路线，使全区域内海拔 3000 米以下部分能被探测到的面积尽可能大，且使从第一架无人机飞出到最后一架完成任务的无人机回到基地的时间间隔尽量短。

问题三：灾区通信中继

太阳能无人机由于在白天不受续航能力限制可以作为地面移动终端之间的通信中继，为灾区提供持续的通信保障。无人机在空中飞行时，可与距离 3000 米以内的移动终端通信，无人机之间的最大通信距离为 6000 米。为了使附件 2 中的任意两个地面终端之间都能实现不间断通信，给出最少需要的无人机架数和每架无人机的飞行路线。

问题四：无人机对地的数据传输

指挥中心从 H 派出 3 架无人机携带通信装备向灾区内的 72 个地面终端发送内容不同，总量均为 500M 的数据。每台通信装备的总功率是 5 瓦，可同时向不超过 10 个地面终端发送数据。只有当地面终端看无人机的仰角大于 30° 、距离不超过 3000 米且没有山体阻隔时，无人机才能传输数据。无人机性能参数如下：飞行速度在 60~100 千米/小时之间可调，水平面内最大加速度 ± 5 米/秒平方，铅垂面内最大加速 ± 2 米/秒平方，可同时在两个方向上加速。在满足题目中给出的

信息速率表达式的前提下，设计无人机的航线、速度以及所服务的用户，使得无人机完成所有任务的时间总和尽量短。

1.2 研究现状

目前，国内外学者对无人机搜救方面的论文较多，但大部分是关于无人机自身的航拍效果优化、航拍图像识别和无人机基于搜救的性能设计，仅有少部分牵涉到路径规划。在这少部分的相关文献中，较为成熟的无人机路径规划算法有3种：概略图规划算法，单元分解方法，类比航迹规划方法，数学形态学航迹规划。实际上，除了这几种算法以外，还有别的算法，大多是对目标区域人为分区、分层，进行优化。处理问题时，往往是将目标区域转化为图像进行骨架化处理，或是将三维空间简化为二维平面再对障碍物做多边形简化进行求解。

1.3 主要符号说明

表 1 主要符号说明表

符号	含义
X_{ijk}	第 i 台无人机是否从 j 点飞行到 k 点
h_1	无人机飞行高度(单位：米)
h_{jk}	j 点到 k 点的平均海拔高度 (单位：米)
l_{jk}	j 点到 k 点的直线距离长度 (单位：米)
v_1	无人机水平飞行速度
v_2	无人机垂直飞行速度
S_i	第 i 个无人机巡查飞过的总面积(单位：平方米)
S_{Eff_i}	第 i 个无人机侦查飞过有效面积(单位：平方米)
S_{Ineff_i}	第 i 个无人机侦察飞过的无效面积(单位：平方米)
S_{Goal}	待搜寻区域总面积(单位：平方米)
θ	地面对无人机的仰视角
Y	移动终端构成的顶点集
W	无人机构成的顶点 ji
$A = [a_{ijt}]$	移动终端间 t 时刻的邻接矩阵
T_i	无人机 i 的飞行时间 (单位：小时)
S_{fly}	飞行搜索面积
S_A	负责的正方形区域面积
D_{H-fly}	从 H 点到正方形区

1.4 地形可视化

根据附件一提供的震区内各个网格点高程数据以及每个网格点的坐标数据，利用 matlab 可绘制出震区的三维地形以及等高线分别如图 1 和图 2 所示。

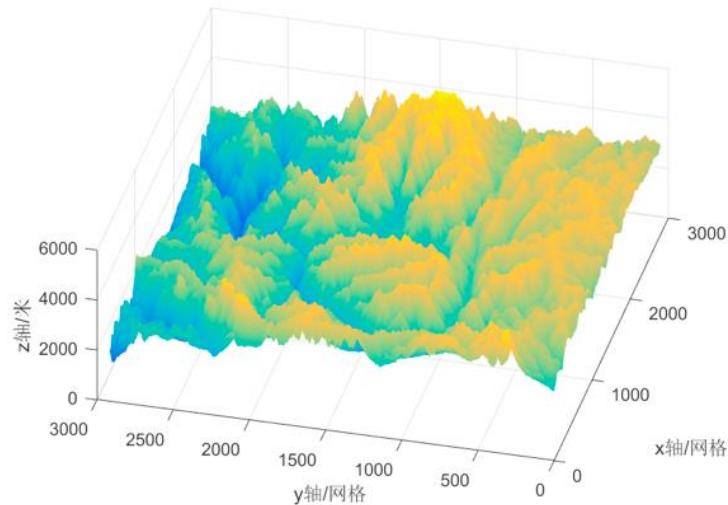


图 1 震区三维地形图

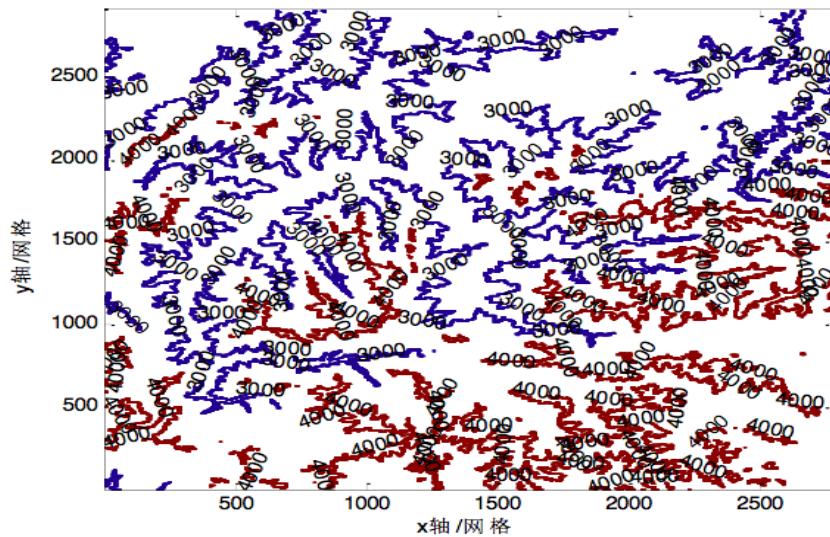


图 2 震区地形等高线图

二. 问题一的建模与求解

将问题一中灾情巡查的两个问题分解成问题（一）与问题（二），分别建立模型求解

2.1 问题假设

1. 山体的山峰到山脚的距离近似为直线，没有凸起或凹陷；
2. 不同无人机飞行路线规划时尽可能不重合，如有重合，忽略重合的有效搜寻面积；

2.2 问题（一）的分析与模型建立

2.2.1 问题（一）分析

该问题中要求在四小时之内使得区域 S 内海拔 3000 米以下的地方尽可能被巡查到，定义该待巡查区域为“巡查区 S1”且无人机飞行高度恒为 4200 米，又因为无人机安全飞行距离不小于 50 米，海拔高度为 4150 米以上的区域视为障碍区，无人机需要避开障碍区飞行。为辅助规划无人机的巡查路线，首先将“巡查区 S1”（黄色阴影）与障碍区（黑色阴影点）在 3000 米（黑色实线）和 4150 米（绿色实线）等高线地形图中表示如下图 3，其中七个重点区域方圆 10km 用色圆圈表示，圆圈的中心即为七个中心点的坐标位置。

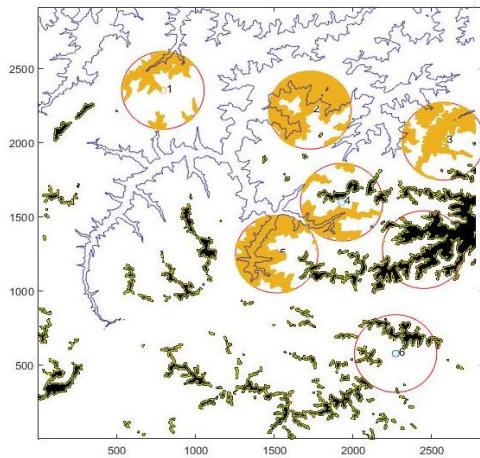


图 3 待巡查区域分布图

问题（一）即为安排无人机的数量及飞行路线使得“巡查区 S1”（黄色阴影）尽可能多的被无人机巡查到，且经过分析，重点区域 F、G 方圆十公里范围内不含任何“巡查区 S1”中的点。这里，我们对“巡查区 S1”内 A、B、C、D、E 内的单元格点 ($38.2\text{m} \times 38.2\text{m}$) 分别进行聚类，得到该 5 个重点区域内“巡查区 S1”的重心点（即为无人机飞行的必经点，且充分刻画了巡查区 S1 的轮廓），建立以覆盖率最高为目标的单目标优化问题。由于题目中要求在灾后黄金 4 小时内尽可能多的搜寻有效区域，我们只考虑从起飞点 H 处飞入搜寻区以及在灾区的搜寻时间，返回时间不计入要求的 4 小时内。进一步求解时，将问题转化为经典的“旅行商问题”（TSP）进行路径规划，利用模拟退火算法求解。

2.2.2 问题（一）模型建立

定义 1：有效飞行面积：无人机搜寻飞过的区域与待搜寻区域交集的面积；

定义 2：覆盖率：无人机有效的飞行面积 / 待搜寻区域总面积；

无人机运动时间计算：

设 $x_{ijk} = 1$ 代表无人机经过路径 (j,k) ， $x_{ijk} = 0$ 代表无人机 i 不经过路径 (jk) ，

定义路径 (j,k) 的直线距离为 l_{jk} ，水平飞行速度为 v_1 ，则无人机在通过路径 (j,k)

所需要的时间为 $\sum_{j,k} x_{ijk} \cdot l_{jk} v_1$ 。

任意两点间搜寻面积计算：

考虑无人机在某一时刻对地面的巡查面积，如图 4 所示。考虑点 j 与点 k 间的坡度对无人机搜寻遮挡的影响，取点 j 与点 k 的平均海拔 h_{jk} ，由三角公式，无人机在高度 h 下对山体 jk 的搜寻半径为 $r_{jk} = (h - h_{jk}) \cdot \tan \theta$ ，则搜寻面积为 $\pi \cdot r_{jk}^2$ 。考虑移动过程，无人机总的搜寻面积为 $x_{ijk} \cdot l_{jk} \cdot 2r_{jk} + \pi \cdot r_{jk}^2$ 。

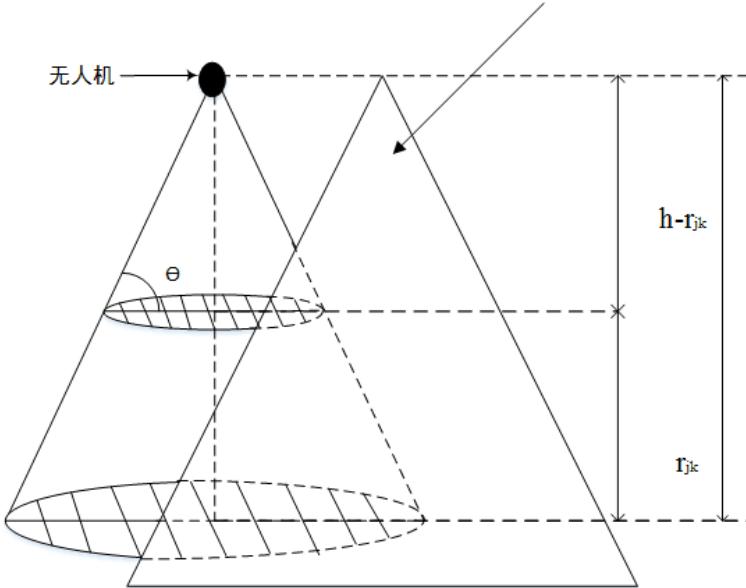


图 4 搜寻面积示意图-山体阻隔

综上考虑，可建立如下模型：

模型 1：避障路径最优化模型

目标：

$$\min_n (\max_i \sum_{i=1}^n \frac{S_{Eff_i}}{S_{goal}}) \quad (1)$$

约束：

$$\begin{cases} \sum_{j,k} x_{ijk} \cdot l_{jk} / v_1, i = 1, \dots, n \\ \theta \leq 30^\circ \end{cases} \quad (2)$$

其中：

$$\begin{cases} S_{Eff_i} = S_i - S_{Ineff_i} \\ S_i = \sum_{j,k} (x_{ijk} \cdot l_{jk}) \cdot 2r_{jk} + \pi \cdot r_{jk}^2 \\ r_{jk} = (h - h_{jk}) \tan \theta \\ h_{jk} = (h_j + h_k) / 2 \end{cases} \quad (3)$$

目标函数即为求出尽量少的无人机架数，使得覆盖率尽可能大。第一个约束条件是对每台无人机在实行任务前往待寻区域以及在待寻区域中搜寻时的飞行时间小于等于 4 小时的约束。

2.3 问题（一）模型求解

求解思想概述：

模型（1）为单目标优化问题，由于附件 1 中给出的灾区地形复杂，障碍物分布极不规律，因此考虑将问题（一）的模型简化为旅行商问题进行求解，主要分为两个步骤：

步骤 1:先对待搜寻区域进行“骨骼化”，利用聚类算法求出 5 个重点区域方圆 10km 内的待搜寻区“巡查区 S1”内的单元网格进行聚类，利用聚类得到的“巡查区 S1”的多个重心对其进行区域刻画，及无人机的搜寻路线必须经过该重心点，且重心点数越多，飞行覆盖的有效面积越大，但路径较长时可能不能满足飞行时间 4 小时的约束；

步骤 2:利用步骤 1 中得到的重心作为路径上的必经点进行最短路径的规划问题，先对一台无人机时，判断是否满足飞行时间约束以及较高的覆盖率，若不满足，增加无人机至两台，采用“区域覆盖”的思想，将 5 个重点区域内“巡查区 S1”分成两个区域，分别求解，判断飞行时间约束及高的覆盖率；若不满足条件，增加无人机至三台，分成三个区域类似求解判断，直至满足条件。

其中，步骤 2 可用如下流程图 5 表示：

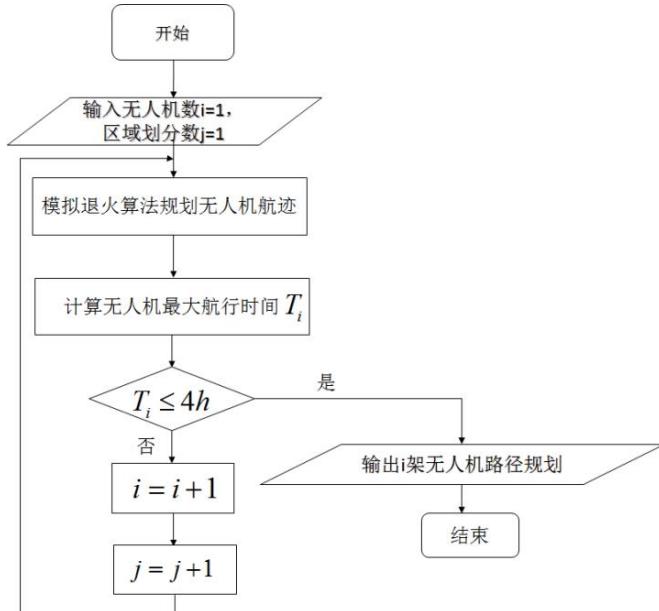


图 5 步骤 2 流程图

结合本题，二元素优化法（模拟退火算法）用于求解模型（1）的基本步骤如下：

算法 1：二元素优化法

step1:任选一个可行路径 s , 并假设 s 是最优路径;

step2:随机选取两点 i 和 k , 将 i 之前的路径不变添加到新路径中, 将 i 到 k 之间的路径翻转其编号后添加到新路径中, 将 k 之后的路径不变添加到新路径中, 从而获得一个新的可行路径;

step3:将新的可行路径与最优路径进行比较, 取两者更优的路径最优路径;

step4:重复 step2 与 step3, 直到找不到更优的路径为止。

模型求解结果:

结果 1: 飞行路径关键点的选取结果

重点区方圆 10 公里“巡查区 S1”单位网格由 matlab 聚类函数 kmeans()可以计算得到分布的重心, 为尽可能准确描述区域形状且又能避免过于复杂的飞行路径, 5 个区域分别用 20 个聚类后的重心点描述, 且作为无人机飞行路径中的必经关键点。各重心坐标详见附录 1, 图 6 中以区域 A 为例, 展示了聚类算法的到的“巡查区 S1”路径关键点分布情况。

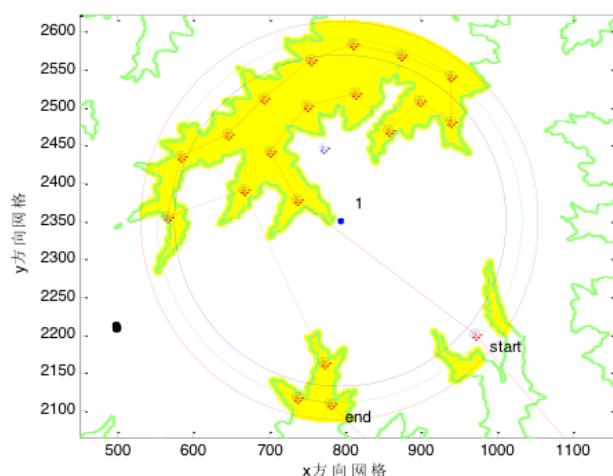


图 6 巡查区 S1 路径图

结果 2: 避障路径最优化模型（1）求解结果

(1) 当只有一台无人机巡查时, 5 个区域的路径关键点作最短路径规划问题, 采用模拟算法 Matlab 迭代求解得到, 最短路径长度为 534.8 公里, 飞行时间为 $534.8 / 60=8.9$ 小时>4 小时, 飞行路径见附录图一。所以一台无人机无法实现要求;

(2) 当有两台无人机巡查时, 将“巡查区 S1”中的五个重点区域划分为两个大区, 每个大区内分别指派一台无人机进行巡查。由于五个重点区的地理位置分布特点较为明显 (A 区距离出发点最远且基本无障碍物影响飞行, B 区 C 区地形分布相邻, D 区 E 区周围障碍物较多), 所有可能分配方式有 $C_5^2=10$ 种方案, 对所有方案进行求解, 最后得到最短路径的分配方式为 B+C 为一区, 记为 BC 区, A+D+E 为一区, 记为 ADE 区, 且从起点巡查 BC 区最短路径为 232.6 公里, 飞行时间 3.88 小时, 从起点巡查 ADE 区最短路径为 261.67 公里, 飞行时间为 4.36 小时>4 小时, 所以两台无人机同时巡查也不能满足飞行时间 4 小时的约束。;

(3) 当有三台无人机巡查时, 将将“巡查区 S1”中的五个重点区域划分为

三个大区，每个大区内分别指派一台无人机进行巡查。类似于（2）中提到的思想，最后求得最优路径的区域的分配方式为 A 区、BC 区，DE 区，其中从起点巡查 A 区的最短路径为 176.13 公里，飞行时间 2.94 小时，巡查 BC 区的最短路径为 232.63 公里，飞行时间 3.88 小时，巡查 DE 区的最短路径为 192.86 公里，飞行时间为 3.21 小时，满足飞行时间约束，且覆盖率为 $507.47/634.89=79.93\%$ ，满足我们的预期，所以三台无人机巡查时即可满足问题（一），三条飞行路线分别见图 7、8、9。

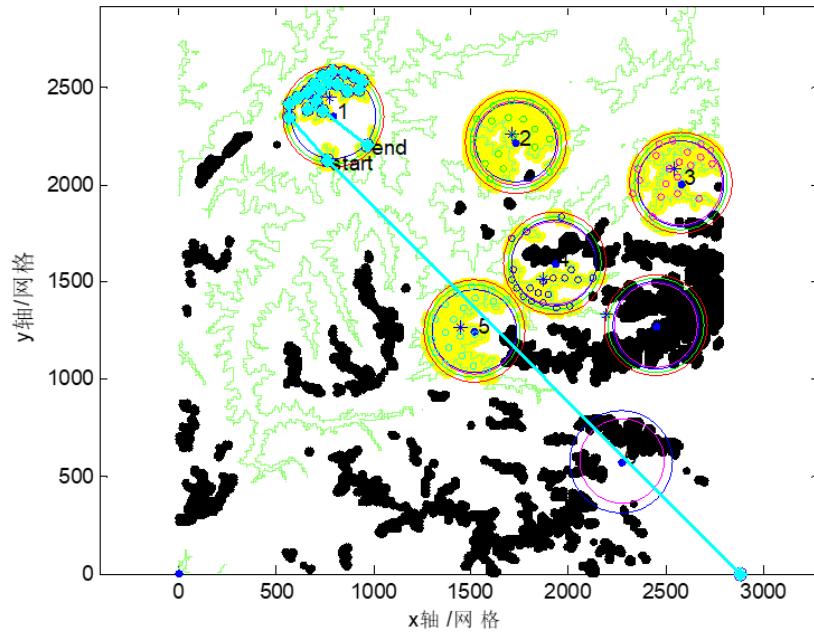


图 7 A 区无人机搜索路径

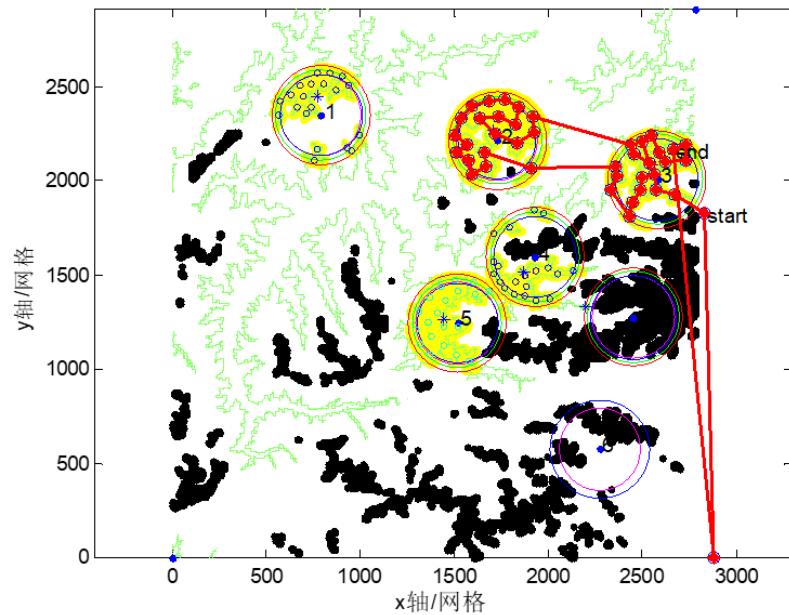


图 8 BC 区无人机搜索路径

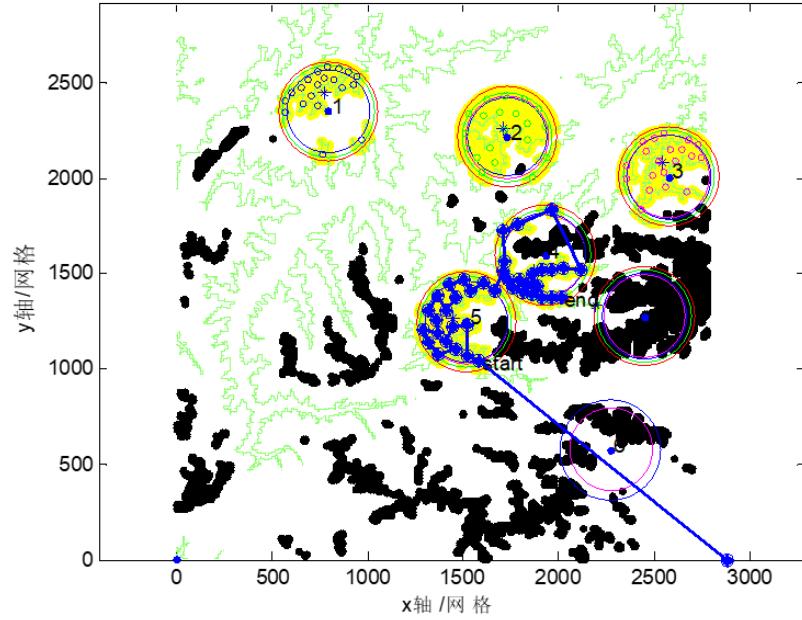


图 9 DE 区无人机搜索路径

2.4 问题（二）分析与模型建立

问题（二）分析：

该题目中要求无人机在高度低于 4000 米的区域巡逻，（为下述方便，我们定义海拔高度低于 4000 米的区域为“巡查区 S2”），并保证在 72 小时之内“巡查区 S2”里被巡查到的地方至少在 3 小时之内被巡查过两次，且无人机每次外出飞行只有 8 小时的续航，返回基地 H 出后最快 1 小时以后才可以再次外出飞行巡逻。

首先将“巡查区 S2”（黄色区域）以及海拔高度高于 4150 米的地区即“障碍区”（黑色区域）在附件一提供的单元格数据中可视化，如图 10 所示。由于题目中要求巡查到的地方至少在 3 小时之内被巡查过两次，且一台无人机从基地出发至返回基地尽量满足 8 小时的飞行时间，结合问题（一）中“分区治理”的思想提示，我们考虑将“巡查区 S2”用面积不等的正方形分区，每个分区由 3 台无人机循环负责搜寻。

巡查方式：以某一个正方形分区为例，为保证区域内相邻两次巡查时间间隔不超过 3 小时，第一台无人机出发后，每隔三小时发出下一台无人机，正方形区域面积及其位置的确定可以保证每台无人机尽可能地在 8 小时的飞行时间内路径不重复搜寻完毕（包含出发及返回时间）。在 72 小时之内，每个正方形区域需要 24 个批次的无人机出发，考虑无人机全部可以重复使用（飞回基地 1 小时候再出发），只需要三台无人机负责一个区域，每台无人机需要飞行 8 次。不包含障碍物的正方形区域采用“扩展方形”巡查路径，包含障碍物的正方形区域参考问题（一）中模型 1 的方法求解即可。

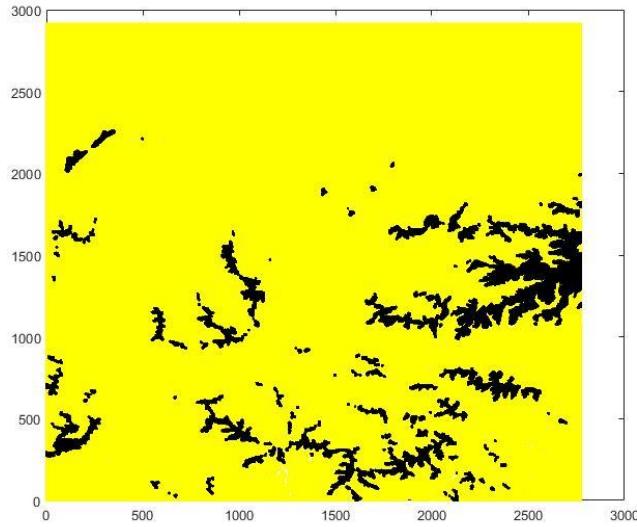


图 10 巡查区示意图

问题（二）模型建立：

为了满足上述要求，利用等式：

$$S_{fly} = S_A + D_{H-fly}^2$$

S_{fly} — 飞行搜索面积； S_A — 负责的正方形区域面积； D_{H-fly} — 从 H 点到正方形区域右下角入口距离

求解每个正方形区域的相应边长进而得到位置坐标。

第 1 列第 1 行的正方形区域的边长满足：

$$\min \left| \left\{ v_l t 2r - [2 \cdot c_{1,1} \cdot 2r + d_{1,1}^2] \right\} \right| \quad (4)$$

$$s.t \quad c_{1,1}^2 = (a - d_{1,1})^2 + (b - d_{1,1})^2 \quad (5)$$

类似的，第 1 列所有正方形区域的边长满足：

$$\min \left| \left\{ v_l t 2r - [2c_{1,1} \cdot 2r + d_{1,1}^2] \right\} \right| \quad (6)$$

$$s.t \begin{cases} c_{i,1}^2 = (a - d_{i,1})^2 + \left(b - \sum_{k=1}^{i-1} d_{k,1} \right)^2 \\ b - \sum_{k=1}^{i-1} d_{k,1} \geq 0 \end{cases} \quad (7)$$

类似的，整个区域对角线上各列第一格（这里只考虑对角线一侧的覆盖情况，对角线另外一侧转置）：

对角线上正方形左上角顶点坐标：

$$\left(a - \sum_{k=1}^{j-1} d_{2,k}, b - \sum_{k=1}^{j-1} d_{2,k} \right) \quad (8)$$

目标及约束:

$$\min \left| \left\{ vt \cdot 2r - [2c_{i,j} \cdot 2r + d_{i,j}^2] \right\} \right| \quad (9)$$

$$s.t \begin{cases} c_{i,j}^2 = \left(a - \sum_{k=1}^{j-1} d_{2,k} - d_{i,j} \right)^2 + \left(b - \sum_{k=1}^{j-1} d_{2,k} - d_{i,j} \right)^2 \\ a - \sum_{k=1}^{j-1} d_{2,k} - d_{i,j} \geq 0 \\ b - \sum_{k=1}^{j-1} d_{2,k} - d_{i,j} \geq 0 \end{cases} \quad (10)$$

综上, 建立**模型 2: 区域覆盖约束模型 (ACCM)**, 任意一个小正方形的边长需要满足如下:

$$\min \left| \left\{ vt \cdot 2r - [2c_{i,j} \cdot 2r + d_{i,j}^2] \right\} \right| \quad (11)$$

$$s.t \begin{cases} c_{i,j}^2 = \left(a - \sum_{k=1}^{j-1} d_{i+1,k} - d_{i,j} \right)^2 + \left(b - \sum_{k=1}^{j-1} d_{i+1,k} - d_{i,j} - \sum_{k=1}^{i-1} d_{k,j} \right)^2 \\ a - \sum_{k=1}^{j-1} d_{i+1,k} - d_{i,j} \geq 0 \\ b - \sum_{k=1}^{j-1} d_{i+1,k} - d_{i,j} - \sum_{k=1}^{i-1} d_{k,j} \geq 0 \end{cases} \quad (12)$$

在每个小区域中, 按是否含有障碍物分两种情况进行处理, 建立**模型 3: 分区路径规划策略**如下:

1. 当小正方形区域中含有障碍物时, 按照问题 (一) 中建立的避障路径最优化模型进行路径规划;
2. 当小正方形区域中不含障碍物之, 按照“直线螺旋”型路径进行区域巡逻

2.5 问题 (二) 模型求解

2.5.1 区域覆盖约束模型求解结果

对上述建立的模型 2, 利用 Matlab 中的 `fsolve()` 函数先求解出第一列第一行的正方形边长, 及方程组 (2), 得到边长 $d_{1,1}=(16.01\text{km}, 95.27\text{km})$, 见 图 11 (左上)。同样方式求解方程组 (3), 得

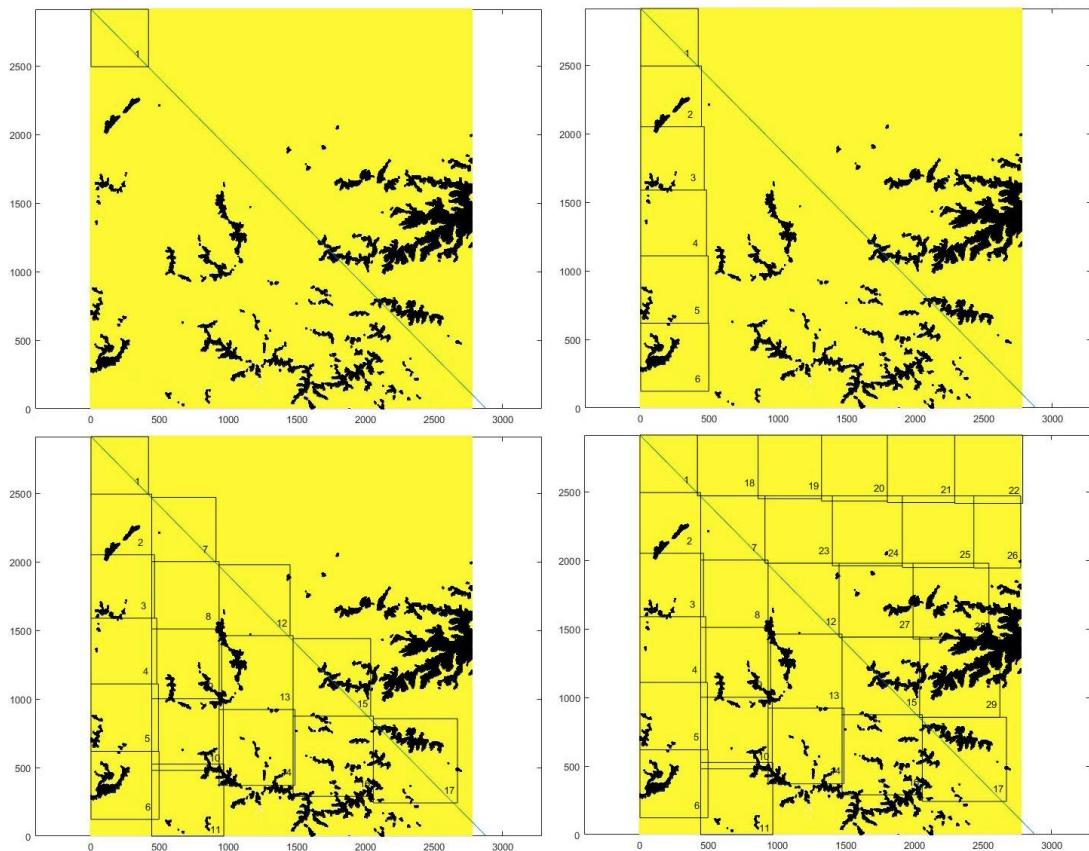


图 11 区域覆盖约束模型求解

到第一列不等边的 6 个小正方形，其排列方式如图 11（右上），每个小正方形的左下角坐标在表 2 中给出。类似，通过求解方程组（5）得到黄色区域对角线下面的全部小正方形边长跟位置，如图 11（左下）所示。将黄色大区域近似为正方形，沿着对角线对折，得到对角线右上角小正方形的覆盖情况，如图 11（右下）所示。综上，整个区域需要 29 个小正方形覆盖，每个正方形的左下角坐标见表 2。

表 2 区域划分坐标

序号	坐标 x (千米)	坐标 y (千米)
1	16.01	95.27
2	16.90	78.37
3	17.68	60.69
4	18.31	42.38
5	18.75	23.63
6	18.95	4.68
7	34.80	76.48
8	35.66	57.72
9	36.36	38.25
10	36.82	18.33
11	36.97	-1.74

12	55.43	55.85
13	56.25	35.26
14	56.78	14.13
15	77.87	33.41
16	78.58	11.07
17	102.03	9.25
18	32.87	94.34
19	50.55	93.56
20	68.86	92.92
21	87.61	92.49
22	106.56	92.29
23	53.52	75.58
24	72.99	74.88
25	92.91	74.41
26	112.98	74.27
27	75.98	54.99
28	97.11	54.45
29	100.16	32.65

2.5.2 模型 3 分区路径规划策略求解结果

对于第一种情况，即小正方形区域中含有障碍物时，以编号为 2 的小正方形为例进行说明。由于黄色区域面积是无人机在该区域巡逻时间内扫过面积的 100 倍，所以在按聚类函数中制定 100 个聚类中心点，然后对这些聚类中心点使用模拟退火算法求解 TSP 问题，得到第一个小正方形区域内的路径规划如图 12（左）表示。

对于第二种情况，即小正方形区域中不含障碍物时，“直线螺旋形法”对区域进行地毯式搜索。以编号为 1 的小正方形为例进行说明，得到如图 12（右）所示的路径规划图。

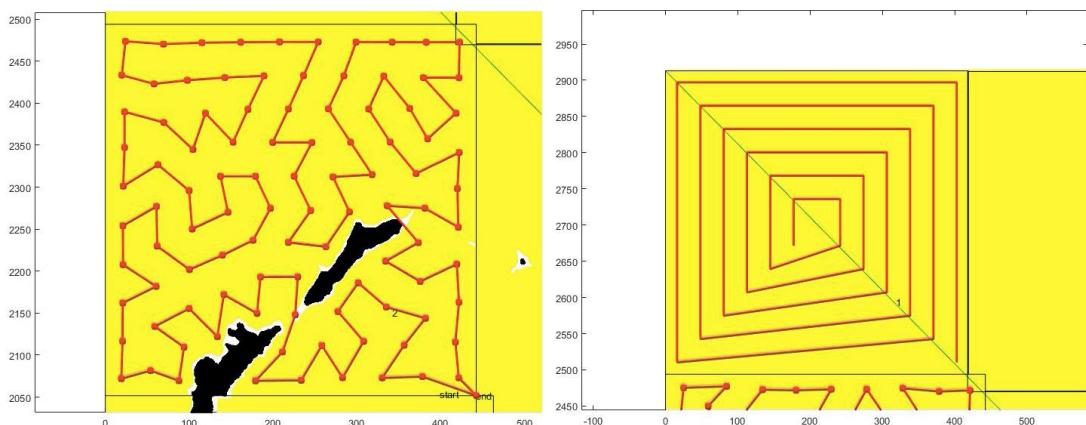


图 12 模拟退火算法路径（左）；直线螺旋形法路径（右）

2.6 模型评价

问题一的两个模型分别适用于定向搜索和非定向搜索的情况。对于有指定搜

索目标的区域，K-means 提取出的关键点相比传统上直接人为标注搜寻点的做法要来得更为科学和高效，同时模拟退火法规划出的路径无交叉、搜寻效率高。需要注意的是，该模型仍有不符实际的情况，需要根据无人机距障碍物的安全距离和飞行时的最小转弯半径对轨迹上距障碍物较近的区域进行圆弧形修正。对于无指定搜索目标的区域，由于任何聚类方法效果都不佳，本文原创性地给出了**ACCM** 模型，该方法理论上对任何区域及地形均适用，是一种自适应的区域划分模型。由于该模型建立之初的思想就是充足使用续航时间，对搜寻的有效时间没有进行优化，导致的结果是无人机可能会在飞往待搜寻区域的路上花费较多时间。

三. 问题二的建模与求解

3.1 问题假设

- 1、两个基地的 30 台无人机同时从基地出发执行任务；
- 2、不考虑无人机执行任务过程中的意外情况；

3.2 问题分析

基地 $H(110,0)$, $J(110,55)$ (单位: 千米) 两处各自派出 15 架无人机对灾区进行声明迹象探测, 结束任务后返回各自基地。探测仪有效探测距离不超过 1000 米, 最大侧视角为 60 度。规划无人机飞行路径, 使得海拔 3000 米以下被探测到的面积尽可能大, 且第一架无人机飞出到最后一架无人机飞回基地的时间间隔尽可能短。为下述方便, 将此问题中的海拔 3000 米目标探测区域定义为“探测区 S3”, 并如图 13 所示。

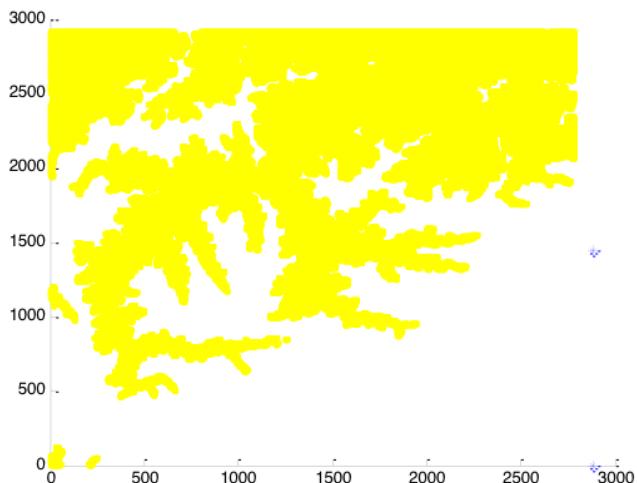


图 13 探测区 S3 示意图

根据基地坐标的大致分布, 利用分区治理思想, 我们将“探测区 S3”划分为两部分, 分配给两个基地负责生命迹象探测。在每个分区内, 利用聚类思想选出“探测区 S3”内网格点的重心, 将问题转化为多旅行商路径规划问题进行求解。

3.3 模型建立

3.3.1 “探测区 S3” 飞行路径的必经点选择及区域划分

由于无人机要尽量多的探测到目标区域中的每一个位置, 选择区域的多个重心点(及飞行路径必经点)仅可能精确的描绘“探测区 S3”, 使得无人机沿着这些散点飞行时能够尽量全面覆盖到目标区域; 为合理分配上述重心点给两个基地, 利用欧式距离度量重心点与两个基地 J, H 的距离关系进行区域划分, $x_{i,J} = 1$

代表第 i 个重心点分配给基地 H 的无人机负责巡逻, 即:

$$\begin{aligned}x_{i,H} &= \begin{cases} 1, & \text{distance}(x_i, x_H) < \text{distance}(x_i, x_J) \\ 0, & \text{otherwise} \end{cases} \\x_{i,J} &= \begin{cases} 1, & \text{distance}(x_i, x_J) < \text{distance}(x_i, x_H) \\ 0, & \text{otherwise} \end{cases}\end{aligned}\quad (13)$$

3.3.2 无人机探测面积计算

考虑山体阻隔无人机探测有效区域，在问题（一）2.1.2 中下任意两点间搜寻面积计算的基础上可以得到每个无人机在任意两点间飞行时的有效搜索面积，那么两个基地各 15 架无人机总的探测面积为

$$\sum_{p=1}^2 \sum_{i_p=1}^{15} S_{Eff_{i_p}} \quad (14)$$

其中，

$$\begin{cases} S_{Eff_{i_p}} = S_{i_p} - S_{Ineff_{i_p}} \\ S_{i_p} = \sum_{j,k} (x_{i_p} \cdot l_{jk}) \cdot 2r_{jk} + \pi \cdot r_{jk}^2 \\ r_{jk} = (h - h_{jk}) \cdot \leq \tan \theta \\ h_{jk} = (h_j + h_k) / 2 \end{cases}\quad (15)$$

3.3.3 无人机飞行时间间隔

考虑无人机匀速飞行，每台无人机的飞行时间为：

$$t_{i_p} = \sum_{j,k} \frac{x_{i_p,jk} \cdot l_{jk}}{v_1}, i_p = 1, \dots, 15, p = 1, 2 \quad (16)$$

每个基地的无人机同时发射，则从第一架无人机飞出至最后一台完成任务的无人机飞回基地的时间即为最后一台返回基地的无人机从出发到返回的总时间，若按题目要求时间间隔最短，则有如下表达：

$$\min \left| \max_{i_p=1, \dots, 15} t_{i_p} \right|, p = 1, 2 \quad (17)$$

综上所述，建立问题二的多无人机协调探测的路径优化模型（模型三）

$$\begin{cases} \max \sum_{p=1}^2 \sum_{i_p=1}^{15} S_{Eff_{i_p}} \\ \min \left| \max_{i_p=1, \dots, 15} t_{i_p} \right|, p = 1, 2 \end{cases}\quad (18)$$

$$s.t \begin{cases} 0 < t_{i_p} \leq 8 \\ \theta \geq 30^\circ \end{cases}\quad (19)$$

其中，

$$\left\{ \begin{array}{l} S_{Eff_{i_p}} = S_{i_p} - S_{Ineff_{i_p}} \\ S_{i_p} = \sum_{j,k} (x_{i_p} \cdot l_{jk}) \cdot 2r_{jk} + \pi r_{jk}^2 \\ r_{jk} = (h - h_{jk}) \cdot \tan \theta \\ h_{jk} = (h_j + h_k) / 2 \\ t_{i_p} = \sum_{j,k} \frac{(x_{i_p,jk} \cdot l_{jk})}{v_1}, i_p = 1, \dots, 15, p = 1, 2 \end{array} \right. \quad (20)$$

3.4 模型求解

3.4.1 求解思路

我们将上述模型三中的双目标分解成两部分，覆盖面积的大小由飞行路径中必经点的数决定，在单位网格聚类时，选择尽可能多聚类后的类别个数，使得“探测区 S3”的区域被这些聚类中心点所描绘，再根据聚类中心距离两个基地 H, J 的距离将其分配给基地 H, J；第二个优化目标时间间隔最小由多旅行商问题解决最短路径规划，该问中假定每台无人机的飞行速度恒为 60km/h，则最短路径决定了时间间隔最小。

该问题的多旅行商问题描述如下：

15 台无人机从一个基地（H 或 J）出发，途径上述聚类中心点的一部分，使得每个聚类中心点必须被某一台无人机探测过且只能探测一次，最后返回基地。求最优路径规划，使得总的飞行距离最小。

将模型三转化成两个多旅行商问题之后，采用遗传算法求解规划最优飞行路径。

遗传算法：

step1:对于有 n 个终端的 TSP 问题，染色体个体为 1 到 n 的全排列

step2:利用改良圈算法求得较好的初始种群，将路径距离总和设为目标函数

step3:利用单点交叉操作产生新解，同时变异操作能保证算法跳出局部最优解

step4:在父代和子代中选择优良品种作为新的父代，加速收敛

step5:重复 step3、step4 直到满足迭代次数

3.4.2 结果 1 “探测区 S3” 飞行路径的关键点选择及分区

选用 150 个聚类中心点描述“巡查区 S3”时，Matlab 软件 kmeans() 函数对原题中提供的网格点进行聚类，得到聚类中心如下图 14 所示的分布，蓝色点代表该点分配给基地 J 的 15 架无人机负责巡查，红色点代表分配给 H 的 15 架无人机。每个路径关键点的位置分布见图 14，具体坐标详见附录 8.1 表 14.

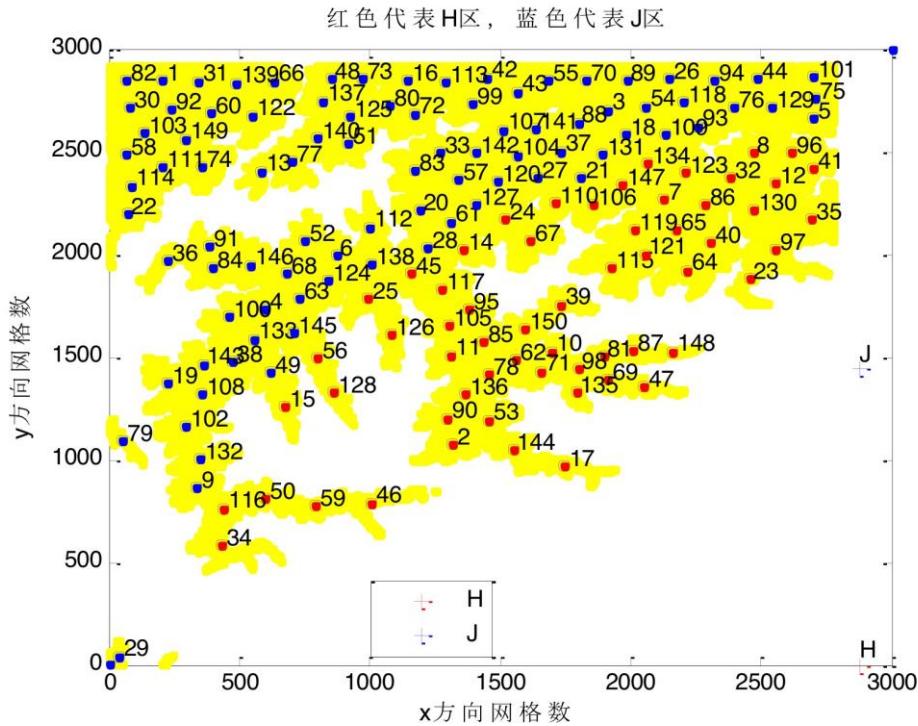


图 14 150 个聚类中心点位置分布

3.4.3 结果 2 无人机路径规划、覆盖率及时间间隔

由遗传算法求解问题二的多旅行商问题，表 3 给出了 H 基地和 J 基地各自 15 架无人机的飞行方式，其中每个路径上节点的坐标详见表 2。图 15 从二维坐标图中给出了飞行路径的可视化结果。

表 3 无人机飞行路径

30 架无人机飞行路径 (150 个聚点)			
无人机 (1~15)	飞行路径 (H 基地)	无人机 (16~30)	飞行路径 (J 基地)
1	H-43-76-109-82-29-H	1	J-65-89-70-50-45-J
2	H-148-51-44-69-83-37-104-H	2	J-71-63-24-95-146-J
3	H-21-55-12-H	3	J-135-28-41-5-137-42-J
4	H-112-150-H	4	J-92-123-122-27-106-62-J
5	H-22-79-H	5	J-54-68-9-91-138-100-14-74-J
6	H-136-96-49-80-H	6	J-87-18-134-31-32-J
7	H-142-114-19-39-121-30-103-H	7	J-126-115-17-38-6-149-111-78-147-97-J
8	H-66-48-107-1-64-H	8	J-124-33-86-140-94-72-J
9	H-52-25-40-H	9	J-20-110-67-88-81-130-133-7-J
10	H-77-61-119-H	10	J-15-132-98-127-84-16-J
11	H-35-H	11	J-125-2-102-60-J
12	H-93-4-101-47-H	12	J-34-117-85-26-J
13	H-128-10-118-36-56-H	13	J-8-99-73-139-113-141-J

14 H-13-145-H

15 H-116-53-75-11-108-90-57-H

14 J-23-131-3-144-J

15 J-120-58-143-105-46-129-59-J

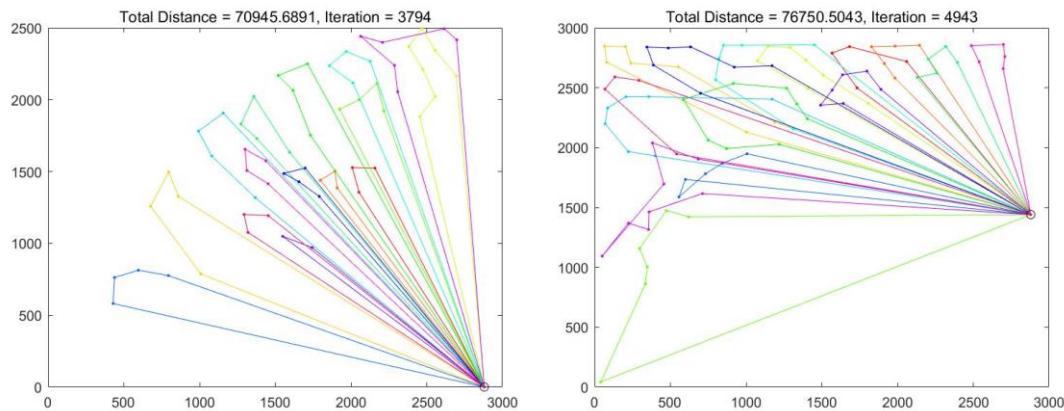


图 15 二维可视化路径

飞行最优路径规划后，两个基地 30 个无人机总的飞行距离见表 4 所示。从而求得 J 基地中飞行路径长度最长的为 208.98 千米，从而第一架无人机起飞到最后一台无人机完成任务归队时间间隔为 **3.48** 小时；H 基地飞行路径最长的长度为 246.95 千米，时间间隔为 **4.11** 小时。且 30 架无人机的有效覆盖面积为 **3218.41** 平方千米，有效覆盖率为 $3218.41 / 3821.87 = \mathbf{84.2\%}$ 。

表 4 无人机飞行距离

H 基地 (150 个聚类点)		J 基地 (150 个聚类点)	
序号	路径长度 (单位: Km)	序号	路径长度 (单位: Km)
1	12.41	1	92.46
2	142.36	2	140.09
3	208.98	3	246.95
4	199.50	4	176.49
5	19.42	5	64.70
6	26.18	6	71.31
7	31.00	7	21.98
8	19.84	8	59.28
9	30.86	9	65.34
10	20.95	10	28.13
11	16.40	11	54.29
12	65.10	12	29.95
13	44.56	13	92.02
14	18.09	14	21.34
15	11.02	15	38.26

3.5 灵敏度分析

对比模 3.4.3 中给出的结果，当考虑用 100 个（之前使用 150 个）聚类中心点描述“巡逻区 S3”作为无人机飞行路径的必经点进行多旅行商路径规划时，H 和 J 基地各 15 架无人机的飞行路径规划见图 16，100 个路径必经点的分布具体坐标及 30 台无人机的飞行路径坐标规划详见附录 8.1 表 13。

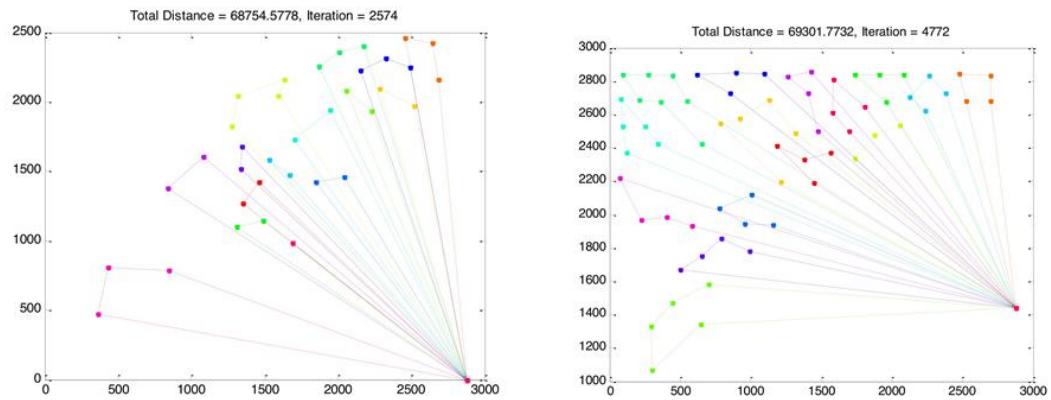


图 16 无人机飞行路径规划图-H 基地（左），J 基地（右）

当设 100 个聚类中心点时，30 台无人机的飞行总路程见表 5。H 基地 15 架无人机里程最远相差 92.86km，第一架无人机飞出到最后一架完成任务的无人机回到基地的时间间隔 1.55h；J 基地最远里程相差 123.85km，所求时间间隔为时间间隔 2.06h。

对比 3.4.3 中 150 个聚类中心点的结果，当聚类中心点增加时，30 架无人机的有效覆盖率由显著提升，但由于飞行路径长度的增加，从无人机出发到最后一架无人机完成任务飞回基地所需要的时间间隔 J 基地和 H 基地大致都增加了 2 小时左右。

表 5 无人机飞行路程

2 个基地共 30 架无人机飞行路程			
H 基地		J 基地	
编号	路程 (Km)	编号	路程 (Km)
1	159.16	1	151.25
2	195.16	2	117.76
3	169.48	3	187.27
4	210.56	4	122
5	171.48	5	213.83
6	148.78	6	143.62
7	202.82	7	241.61
8	173.9	8	241.39
9	158.74	9	122.54
10	138.53	10	168.85
11	190.14	11	206.19
12	175.38	12	185.39

13	198. 3	13	166. 52
14	209. 43	14	225. 87
15	117. 7	15	146. 29

表 6 结果对比

	100 个聚类中心点	150 个聚类中心点
扫描面积 (平方千米)	1221.67	3218.41
覆盖率	32%	84%
所求时间间隔 (小时)	J:2.06; H:1.55	J:4.11; H:3.48

3.6 模型评价

问题二与 MTSP 模型较为贴近，为了规划 30 架无人机的路径，我们使用了 Genetic Algorithm 进行求解。相比于问题一中使用的模拟退火算法，该方法算法收敛速度快，全局搜索能力强，求出的路径能满足尽可能不交叉同时搜索的区域最大化的要求。遗传算法也有其缺点：编程较为困难，适应度值标定方式多种多样，没有一个简洁、通用的方法。

四. 问题三的建模与求解

4.1 问题重述与分析

地震灾害后，灾区电力设施受损导致通信中断，继而抢险救灾人员间的相互通信受阻，因此需要无人机作为通信中继为配备通信终端实施搜救的工作人员提供持续的通信保障，提高搜救效率。其中，72名搜救人员作为地面移动终端可移动的最大距离为2000m，无人机续航能力为12小时，无人机与地面移动终端有效通信距离为3000m、无人机间的有效通信距离为6000m，结合实际问题，上述距离皆为三维空间欧氏距离。考虑到搜救成本，在上述约束条件下，最小化无人机数量并最优化每架无人机飞行路线。

观察72个移动终端空间位置分布(如图17)可得，有3个移动终端(50,52,63)不在震区高程数据中，故舍弃，只考虑余下69个移动终端，由图18可知，69个移动终端空间分布较分散，因此须先考虑单独分配无人机的移动终端，再考虑为距离较近的移动终端规划无人机航迹。

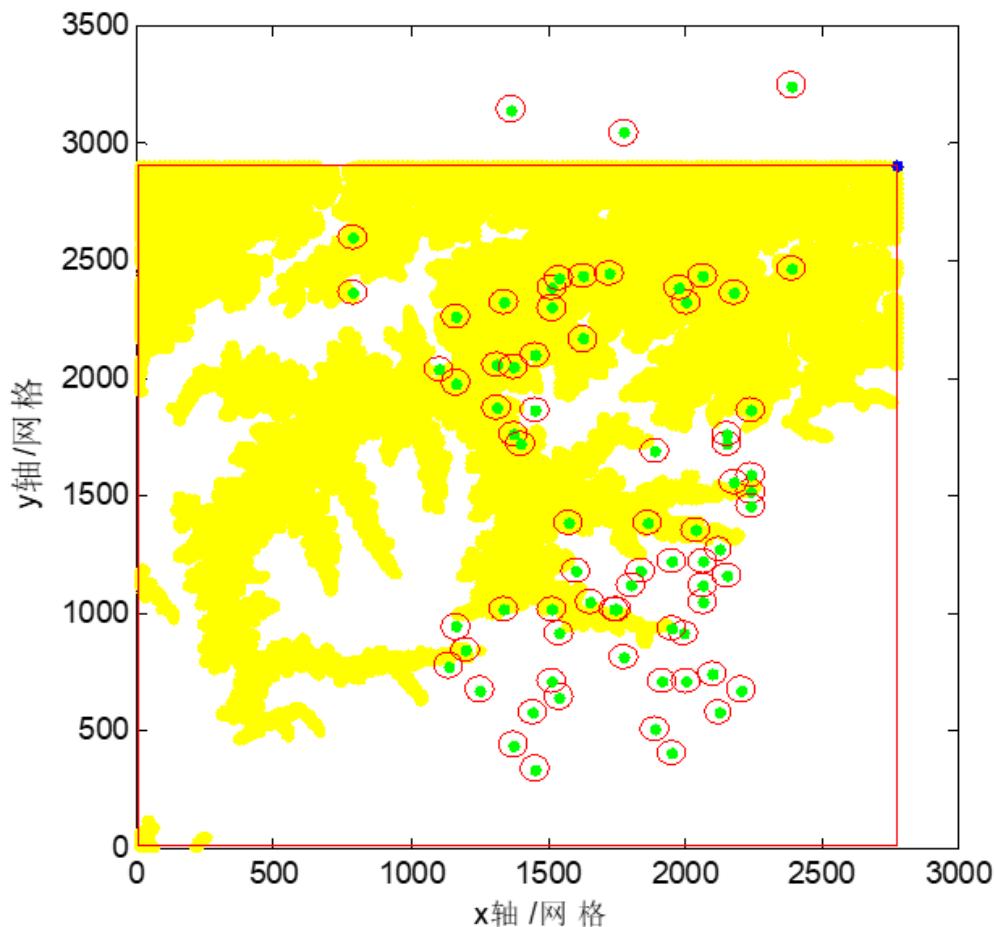


图 17 72 个终端平面俯视图

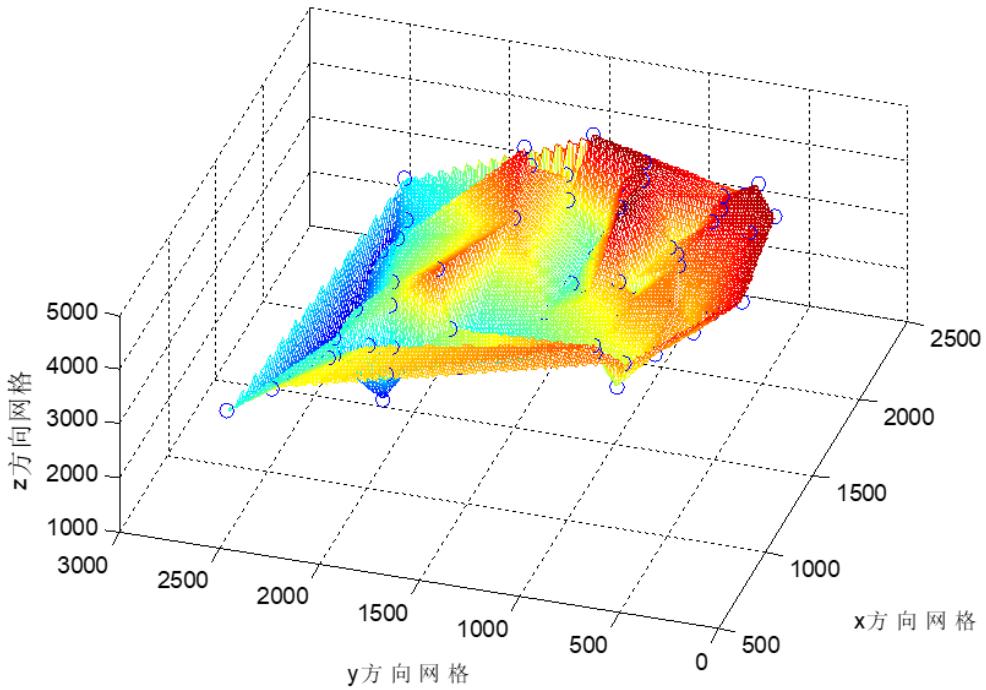


图 18 72 个移动终端三维位置示意图

4.2 模型建立

模型假设:

1. 假设无人机与移动终端通信不计山体阻隔;
2. 无人机与移动终端为质点;

模型建立: 问题目标可简化为在求解最小无人机数量 n , 使得任意 t 时刻移动终端 i,j 之间的邻接矩阵 $A = [a_{ijt}]$ 为元素全为 1 的矩阵, 即:

$$\min\{n\} \quad \text{s.t.} \quad A = [a_{ijt}]_{69 \times 69} = [1]_{69 \times 69}, \forall i, j \in Y \quad (21)$$

且满足如下约束:

$$\begin{cases} l_{ij} \leq 3000, \forall i \in Y, j \in W \\ l_{ij} \leq 6000, \forall i, j \in W \\ T_i \leq 12, \forall i \in W \end{cases} \quad (22)$$

目标函数求解最小的无人机架数使得任意时刻两个移动终端可通信, 第一个约束条件保证无人机与移动终端间的通讯距离小于 3000 米, 第二个约束保证无人机与无人机之间的通信距离保证 6000 米, 第三个约束条件保证无人机的续航时间小于 12 小时。

4.3 模型求解

求解思路概述: 考虑到震区内的 69 个移动终端空间分布十分分散, 模型求解可分为 3 个步骤。

步骤 1: 求出移动终端之间可安排一架无人机维持通信的最小距离 d_{thresh} ;

步骤 2: 对集合 $\text{Set}_1 = \{i \mid l_{ij} \leq d_{thresh}, \forall j \in W \setminus i\}$ 中的移动终端两两间安排一架无人机在中心处以半径 100m 盘旋，保证持续通信；

对集合 $\text{Set}_2 = \{i \mid l_{ij} > d_{thresh}, \forall j \in W \setminus i\}$ 中的每个终端安排一架无人机跟踪其移动；

步骤 3: 验证步骤 1、2 得出的无人机航迹是否满足 $T_i \leq 12, \forall i \in W$ ；

步骤 4: 验证上述步骤规划的无人机航迹是否满足 $l_{ijt} \leq 6000, \forall i, j \in W \text{ 且 } i \neq j$ ，利用图论中最小生成树的思想，对以无人机为顶点，

两点间距离为边权重连通赋权图 $G = (W, E)$ ，其最小生成树可使得连通无人机的距离最小。利用 Prim 算法可求得连通无人机的最短距离矩阵，对距离大于 6000 米的无人机，各为其安排 1 架无人机保持通信。

其中，用 Prim 算法构造最小生成树方法如下：

算法 3: Prim 算法

设置集合 P 和 Q，P 用于存放 G 的最小生成树的顶点，Q 存放 $G = (W, E)$ 最小生成树的边。算法思想是，从所有 $p \in P, v \in W - P$ 的边中，选取具有最小权的边 pv ，将顶点 v 加入集合 P 中，将边 pv 加入集合 Q 中，如此不断重复，直到 $P = V$ 时，最小生成树构造完毕，此时集合 Q 中包含了最小生成树的所有边。

算法步骤如下：

step1: $P = \{v_1\}$ ， $Q = \emptyset$ ；

step2: while $P \neq W$

 找最小边 pv ，其中 $p \in P, v \in W - P$

$P = P + \{v\}$

$Q = Q + \{pv\}$

end

求解结果：

- 考虑到移动终端最远可移动 2000m，无人机与移动终端的通信距离须控制在 3000m 以内，无人机在两移动终端中点处盘旋，所以 d_{thresh} 须满足：

$$d_{thresh} / 2 + 2000 = 3000 \quad (23)$$

解得 $d_{thresh} = 2000m$ 。

- 通过计算并判断移动终端间距离可得，

$\text{Set}_1 = \{4, 11, 17, 19, 21, 42, 45, 63, 66, 69\}$ 共 10 移动终端，因此为其分配 5 架无

无人机在两个终端中心处盘旋，无人机分配情况如表 7; Set_2 中共 59 个点，因此须安排 59 个无人机跟随集合中的点。

表 7 无人机分配情况

移动终端	移动终端间距离 (米)
4, 11	1726.5
17,66	1881.7
21,19	1336.5
42,72	1823.0
45,69	547.1

3. 由于无人机追踪地面终端，而地面终端只在白天 12 小时内工作，故无人机工作时间满足要求。
4. 由最小生成树得到的距离矩阵可知，有 23 个移动终端距离在 6000 到 12000 米之间，故还需安排 23 架无人机追踪此 23 个移动终端，终端编号如下表，最小生成树的权矩阵如附录 8.1 表 16 所示。

1	5	9	10	11	12	15	17	20	21	22	23	37	38	39	40	44	49	51	57	59	63	66
---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

最终可得，总共需要安排 87 架无人机在无人机与地面终端通信距离小于 3000 米，无人机间通信距离小于 6000 米的条件下，完成任意两地面移动终端间持续不间断通信任务，无人机位置分布如下图 19，所有无人机皆在此位置上以 100 米半径盘旋。

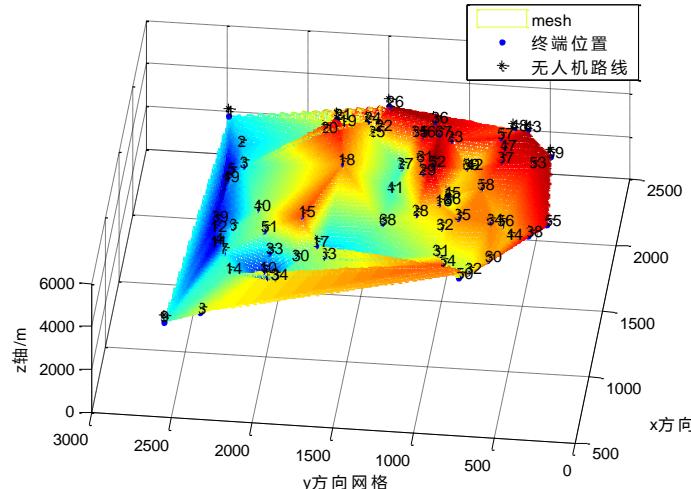


图 19 无人机位置分布

4.4 模型评价

在第三个问题规划所有 72 个移动终端通过无人机的连通问题时，我们使用了最小生成树 (Minimum Spanning Tree) 来进行建模，使得每个移动终端都能通过无人机进行互连，同时又使得满足条件时所使用的无人机架数最少。在对模型所得结果验证之后，我们发现，对于存在不满足约束条件的无人机，需要额外安排一架无人机在该无人机负责连通的两个移动终端的中心处以半径 100 米盘旋。经过这样的调整，才能完全满足题目的所有要求。相对来说，模型不够智能，需要人为调整的地方较多。

五. 问题四的建模与求解

5.1 模型假设

1. 无人机飞行高度可变，总可以找到合适位置对移动终端传输数据，不受山体阻隔；
2. 忽略终端移动的影响；
3. 无人机执行完数据传输任务后需返回至基地；
4. 三台无人机同时从基地出发；

5.2 问题分析与模型建立

分析 1：数据传输策略分析

由于无人机与终端的有效数据传输距离为 3000 米，且终端看无人机仰角大于 30 度，考虑极端情况，即仰视角为 30 度时，在某一时刻 t_1 到 t_2 （暂时忽略与地表的 50 米的安全距离）无人机对单个终端传输数据的情况：参考图 20 当无人机与终端从开始满足有效传输距离 a 点到刚刚飞离终端的不满足数据有效传输距离 b 点之间的飞行路径长度为 $3000 \cdot \sin 30^\circ \cdot 2 = 3000\sqrt{3} \approx 5196m$ 。考察 72 个终端之间的距离，两两距离小于 5196 米的终端群在无人机飞过时可能会存在共同分配 5 瓦总功率同时传输数据的情况，统计满足该条件的终端占比较小。考虑到多终端同时从一个无人机传输数据的比较复杂，涉及到功率分配，以及由于飞行路径的关系使得每个终端与无人机距离不同则传输速度也有变化，最终导致不同终端需要完全传递完需要的时间有差别，需要其他无人机或该无人机续传，极大地增加了该问题的复杂度。所以，为了减小“牵一发而动全身”的连环影响，这里我们固定某些分配参数，提出如下传输策略，并进一步验证这一传输策略的可行性。

策略 2 数据传输自适应策略：

1. 一台无人机只允许同时只对一个移动终端传输数据，直至 500M 数据传输完毕为止，继续下一台终端的传输任务；
2. 每台终端 500 数据完整传输的方式：无人机在对某一个终端进行任务传输时以一恒定速率飞行（但针对每个终端的恒定速度可能是不同的），数据传输任务完成后直至开始下一台终端传输任务过程中调节速度，以最短用时飞行；
3. 关于每个终端所需要的无人机均匀飞行速度及路径形状的要求：无人机按一定路径（直线或其他）和某一均匀速度飞过终端上空，保证从开始进入某终端的数据有效传输区域到离开有效传输区域的时间内可以传输完毕该终端的 500M 数据。

下面论证上述策略的可行性，即

分析 2：相关命题的提出与证明

命题 1：当无人机飞行高度下降时，无人机与待传输数据的终端距离减小，但数据传输速率没有显著提高，即无人机传输速率对无人机与移动终端间的距离不敏感，牺牲无人机飞行高度并不能提高数据传输效率。

命题 1 证明如下：

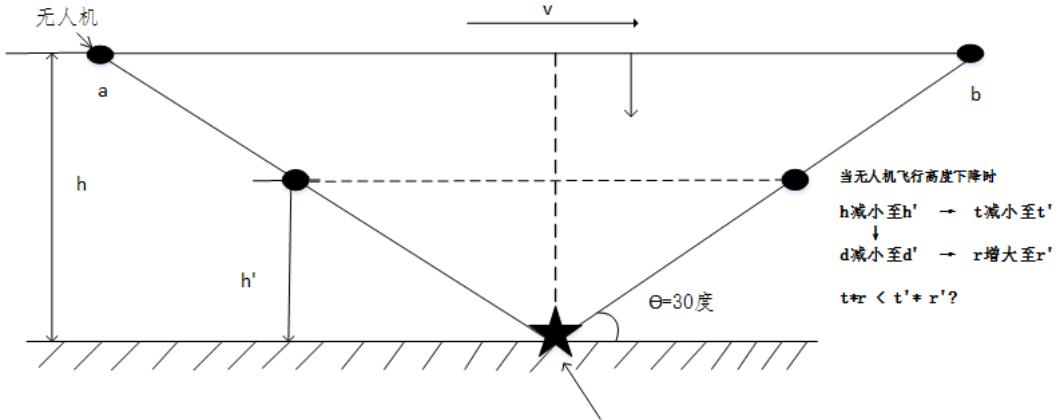


图 20 命题 1 证明示意图

参考图 20, 讨论等式

$$r_i(t) = B_i \log_2 \left(1 + \frac{p_i(t)}{\rho_0 d^2(u, i)} \right) \quad (24)$$

因为 $\rho_0 = 4.314 \times 10^{-10}$ 非常小的数值作为分母, 可以近似

$$\begin{aligned} r_i(t) &= \log_2 \left(1 + \frac{5}{\rho_0 d^2(u, i)} \right) \approx \log_2 \left(\frac{5}{\rho_0 d^2(u, i)} \right) \\ &= \log_2 5 - \log_2 \rho_0 - 2 \log_2 d(u, i) \end{aligned} \quad (25)$$

若减小无人机与终端间距离 $d^2(u, i)$, 简单记为 d 减小至 $d' = d/n$, 则 $\log_2 d$ 减小至 $\log_2(d/n)$,

$$\text{又 } \log_2 \frac{d}{n} = -\log_2 n + \log_2 d \quad (26)$$

将 d 的变化带入信息速率公式, 得到数据传输速率增加为

$$r_i(t) = \log_2 n \quad (27)$$

即, 降低无人机飞行高度并不能显著提高其间的数据传输速率。

证毕。

分析 3: 策略 2 可行性分析与模型建立

下面探讨策略 2 中的第 2 条, 第 3 条要求如何满足, 即可以证明策略 2 的可行性。基于上述命题 1, 我们发现无人机应牺牲距离, 在仅可能高且又满足数据传输的有效距离内的空间飞行, 使得在飞出数据有效传输距离前的飞行时间尽可能长, 能够将 500M 数据完全传输完毕, 根据问题四中提供的数据及要求, 可以将问题描述为下述规划模型五:

$$\min t_i, \forall i \in Y$$

$$s.t \quad \int_0^{\frac{3000\sqrt{3}}{v_i}} B_i \log_2 \left(1 + \frac{5}{\rho_o \left[1500^2 + (1500\sqrt{3} - v_i t)^2 \right]} \right) dt \geq 500 \quad (28)$$

其中，

$$t_i = \frac{3000\sqrt{3}}{r_i}$$

5.3 模型求解

根据 5.2 中的数据传输自适应策略以及规划模型 5，利用 Matlab 及优化算法，我们先对无人机全程直线型路径讨论其能否满足在终端的有效数据传输区域中飞行时将 500M 数据全部传输完毕：无人机在海拔 1500 米处以最低速度 60 千米 / 时且直线型路径飞过终端有效数据传输区域，利用多旅行商算法求解出此时三架无人机的飞行轨迹，得到其飞行总时间如表 8，进一步检验策略 2 中的目标 3，我们发现对于子信道带宽在 140KHZ 以上的移动终端，无人机在 1500 米高空直线穿过移动终端的信号接收范围即可完成对移动终端的 500M 数据传输；但对于子信道带宽在 140KHZ 以下的移动终端，并不能在信号接收范围内完成接受任务，因此考虑进一步优化求解。

表 8 最小速度-60km/h 耗时图

无人机序号	总时间 (单位: s)
1	9477.95
2	14173.04
3	20565.83

针对上述不满足条件的移动终端特点，调整对应无人机在其信号接受范围内的飞行时间，及适当调整其飞行路径即完成进一步的优化目标。通过尝试，我们发现对于子信道带宽在 140KHZ 以下的移动终端，无人机可在 1500 米高空，绕行一个半径为 2598 米的半圆轨迹进行数据传输即可在信号接受范围内，终端完全接收完 500M 数据，飞行路径见表 9。

表 9 飞行路径

无人机 序号	路径
1	H-44-48-38-59-57-65-36-63-17-67-46-53-30-70-43-58-H
2	H-24-68-47-28-42-16-18-64-69-29-32-55-51-33-61-45-39-56-54-60-H
3	H-49-62-19-41-52-34-14-31-35-11-9-10-15-8-7-5-12-13-40-6-4-50-3-2-21-22-20-25-26-23-27-66-37-H

为进一步减小飞行时间，为缩短在非信号接受范围内的飞行时间，通过增加其速度至最大 100 千米 / 小时全力飞行，直至接近下一个目标终端的信号接受范围前调整其速度至合适。

综上，本问题的结论如下。

结果 1:无人机的飞行速度及分配功率说明

每架无人机的飞行时间由三部分组成：一是无人机在非信号传输区域以最大速度 100 千米/小时飞行与无人机在信号传输区域以自适应最大速度飞行之间的加速、减速时间；二是无人机在非信号传输区域以最大速度 100 千米/小时飞行的时间；三是无人机在信号传输区域以自适应最大速度飞行的时间。根据每个终端的频率数不同，表 10 给出了每个频率的终端对应其信号传输区域中所需要的无人机飞行速度及数据传输时间。

基于 5.2 中数据传输自适应策略及可行性分析，每个无人机服务的用户数不超过 1 个，即每个用户都分配 5 瓦的功率。

表 10 飞行速度及传输时间

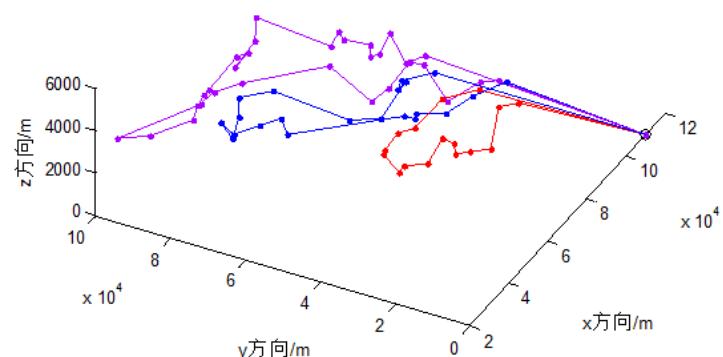
飞行路径为半圆		
频率(KHZ)	速度 m/s	时间 (s)
100	16.87	483.94
110	18.55	439.95
120	20.24	403.29
130	21.93	372.26

飞行路径为直线		
频率(KHZ)	速度 m/s	时间 (s)
140	16.77	309.91
150	17.97	289.21
160	19.17	271.10
170	20.37	255.13
180	21.47	242.06
190	22.67	229.24
200	23.87	217.72

结果 2:空间中三台无人机的三维飞行路径示意图（见图 21）

图 21 三台无人机飞行路径三维图

Total Distance = 752457.3609, Iteration = 7991



结果 3：传输任务完成的总时间

优化后的三架无人机飞行各自总时间见表 11，由于假设 5.1 中规定无人机全部同时从基地废除，则无人机完成所有任务的时间总和为耗时最大的飞行总时间，即 **17784.81 秒（4.94 小时左右）**。

表 11 优化后耗时

无人机序号	总时间（单位：s）
1	7559.44
2	11333.37
3	16079.31

5.4 模型评价

问题四的三架无人机的飞行路线规划使用 Genetic Algorithm 去进行求解。我们使用自适应数据传输算法（ADTA）给出了无人机在移动终端数据接收范围内的最大速度。该方法在保证无人机能将 500M 数据全部传输给相应移动终端的同时，又大大优化了无人机在移动终端数据接收范围内飞行速度较慢的问题。由于需要考虑的因素较多，本题对该模型进行了较多简化，未考虑终端移动的情形，另外无人机在不同移动终端间变向所用的时间也忽略不计。由于我们对问题背景不是很熟悉，理解该问题时可能有偏差。如果能有实际体会，我们会对模型做进一步优化。

六. 总结与展望

6.1 总结

本文分析了无人机在三维山区地形中扫描搜索、通信中继、数据传输等实际场景，将问题进一步抽象为图论中 TSP、MTSP 和最小生成树等问题，并综合运用 kmeans 聚类思想、二元素交换法和基于改良圈的遗传算法解决实际难题，通过对问题的仔细分析，建立下列模型：

1. 避障路径最优化模型
2. 区域覆盖约束优化模型（ACCM）
3. 多无人机协同探测路径优化模型
4. 最小生成树模型
5. 数据传输自适应策略

6.2 展望

1. 由于实际三维地形复杂，本文做出了一些适当假设，如：在低矮山区中认为地形 z 方向起伏较少，近似理解为固定海拔的开阔地，从而导致部分模型精度损失。
2. 在搜索面积计算中，我们简化了零散区域的计算，导致模型计算结果可能偏小。
3. 在计算旅行商距离问题上，我们较多使用直线长度代替轻微曲线的长度，使得计算结果偏小。
4. 由于时间仓促，我们没有深入研究移动终端移动较大范围的情况。
5. 由于我们对问题背景不是很熟悉，理解该问题时可能有偏差。如果能有实际体会，我们会对模型做进一步优化。

七. 参考文献

- [1] Kiraly, A. and J. Abonyi (2011). Optimization of Multiple Traveling Salesmen Problem by a Novel Representation Based Genetic Algorithm. *Studies in Computational Intelligence*. M. Koppen, G. Schaefer and A. Abraham. **366**: 241-269.
- [2] Zhu, Y. and Z. Cai, et al. (2011). "Hybrid genetic algorithm for multiple-objective TSP." *Computer Engineering and Applications* **47** (7): 52-6.
- [3] 符小卫,程思敏,高晓光. 无人机协同中继过程中的路径规划与通信优化 [J/OL]. 系统工程与电子技术,2014,36(05):890-894. (2014-03-12)[2017-09-20].
- [4] 卓星宇. 无人机山区搜寻方法研究[D].中国民用航空飞行学院,2017.
- [5] 陈通. 救援直升机航迹规划研究[D].中国民用航空飞行学院,2011.
- [6] Li, J. and Q. Sun, et al. (2014). "Colored Traveling Salesman Problem and Solution." *IFAC PAPERSONLINE* **47** (3): 9575-9580.
- [7] 杨国兴.一种多出发点多旅行商问题到旅行商问题的转换[J]. 系统工程理论方法应用,1993,2(3):66-80
- [8] 史峰. MATLAB 智能算法30个案例分析 [M]. 北京航空航天大学出版社, 2011.
- [9] 卢厚清,王辉东,黄杰,李波.任务均分的多旅行商问题[J].系统工程,2005,(02):19-21. [2017-09-20].

八. 附录

8.1 附录 1

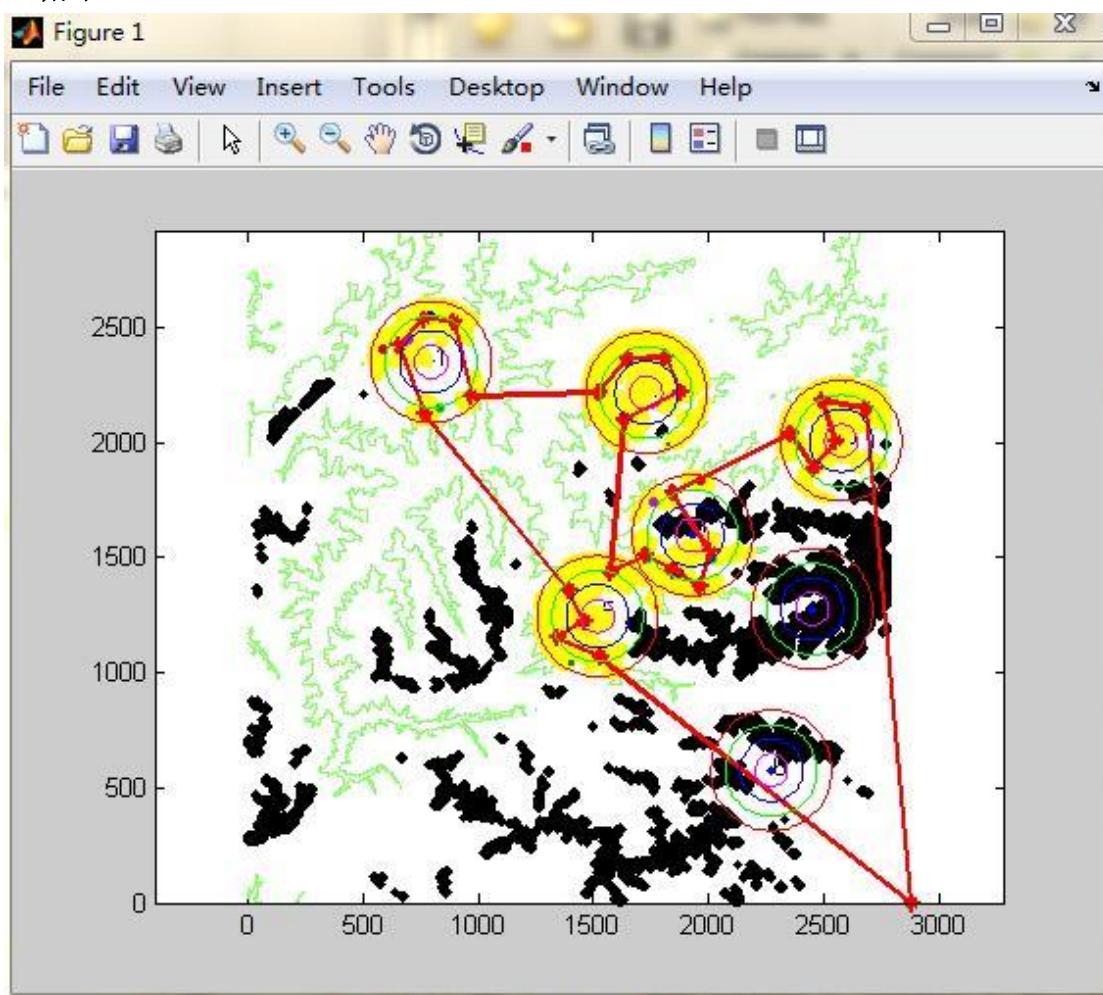


图 22 问题一（一）-单无人飞行轨迹

表 12 重心坐标

序号	坐标 x (千米)	坐标 y (千米)
1	16.01	95.27
2	16.90	78.37
3	17.68	60.69
4	18.31	42.38
5	18.75	23.63
6	18.95	4.68
7	34.80	76.48
8	35.66	57.72
9	36.36	38.25
10	36.82	18.33

11	36. 97	-1. 74
12	55. 43	55. 85
13	56. 25	35. 26
14	56. 78	14. 13
15	77. 87	33. 41
16	78. 58	11. 07
17	102. 03	9. 25
18	32. 87	94. 34
19	50. 55	93. 56
20	68. 86	92. 92
21	87. 61	92. 49
22	106. 56	92. 29
23	53. 52	75. 58
24	72. 99	74. 88
25	92. 91	74. 41
26	112. 98	74. 27
27	75. 98	54. 99
28	97. 11	54. 45
29	100. 16	32. 65

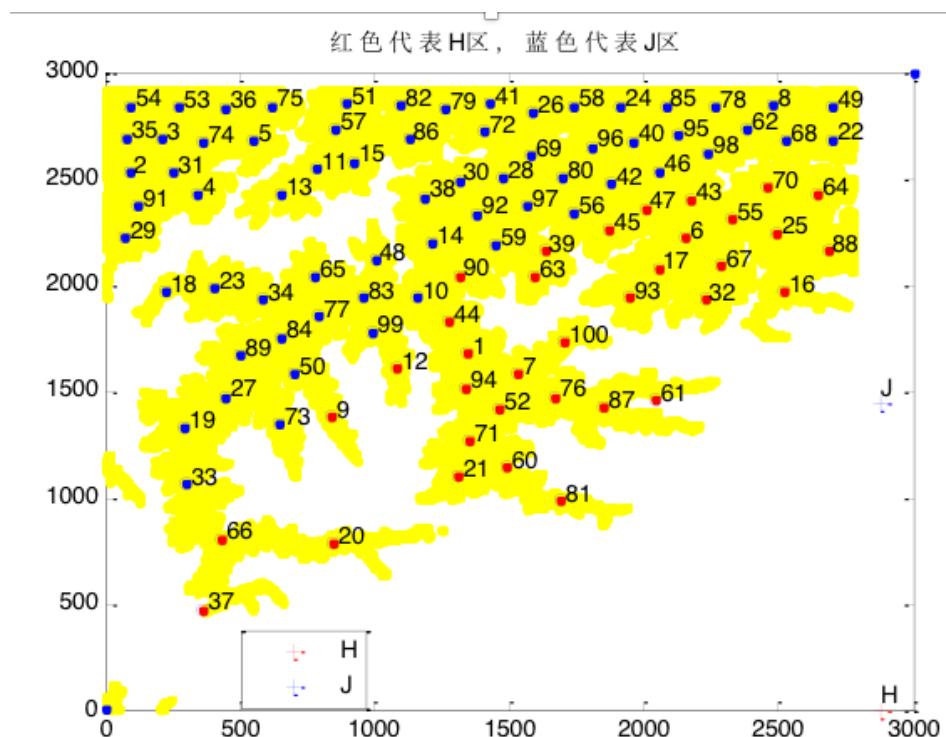


图 23 100 路径关键点

表 13 100 个聚类中心坐标

序号	坐标 x (Km)	坐标 y (Km)	序号	坐标 x(Km)	坐标 y (Km)
1	51. 42	64. 19	51	34. 3	108. 95

2	3. 44	96. 66	52	55. 96	54. 18
3	8. 16	102. 7	53	10. 45	108. 41
4	13. 03	92. 58	54	3. 51	108. 53
5	20. 91	102. 33	55	88. 8	88. 35
6	82. 28	84. 91	56	66. 51	89. 24
7	58. 43	60. 51	57	32. 64	104. 24
8	94. 77	108. 79	58	66. 48	108. 52
9	32. 13	52. 62	59	55. 25	83. 57
10	44. 21	74. 1	60	56. 86	43. 63
11	29. 98	97. 24	61	78. 14	55. 68
12	41. 29	61. 39	62	91. 03	104. 24
13	24. 96	92. 62	63	60. 92	77. 95
14	46. 44	83. 81	64	101. 05	92. 7
15	35. 27	98. 51	65	29. 71	77. 95
16	96. 26	75. 29	66	16. 5	30. 79
17	78. 43	79. 35	67	87. 19	79. 99
18	8. 69	75. 09	68	96. 61	102. 43
19	11. 16	50. 79	69	60. 39	99. 72
20	32. 41	29. 89	70	93. 83	93. 83
21	49. 95	42. 04	71	51. 63	48. 44
22	103. 15	102. 52	72	53. 71	104. 1
23	15. 36	75. 78	73	24. 76	51. 28
24	73. 01	108. 52	74	13. 99	102. 1
25	95. 19	85. 77	75	23. 6	108. 54
26	60. 5	107. 34	76	63. 73	56. 25
27	17	56. 22	77	30. 24	70. 89
28	56. 29	95. 57	78	86. 6	108. 34
29	2. 69	84. 83	79	48. 13	107. 92
30	50. 19	95. 13	80	64. 86	95. 61
31	9. 74	96. 66	81	64. 5	37. 54
32	85. 04	73. 76	82	41. 95	108. 62
33	11. 49	40. 7	83	36. 64	74. 19
34	22. 3	73. 74	84	24. 96	66. 95
35	2. 93	102. 82	85	79. 5	108. 56
36	17. 03	108. 16	86	43. 06	102. 7
37	13. 83	18. 05	87	70. 65	54. 39
38	45. 36	92. 14	88	102. 46	82. 56
39	62. 34	82. 51	89	19. 21	63. 67
40	74. 92	102. 11	90	50. 18	78. 03
41	54. 61	109. 05	91	4. 58	90. 66
42	71. 71	94. 62	92	52. 76	89. 06
43	82. 99	91. 79	93	74. 26	74. 18
44	48. 74	69. 77	94	51. 05	57. 96
45	71. 34	86. 28	95	81. 2	103. 3

46	78. 55	96. 79	96	69. 02	101. 13
47	76. 58	90. 11	97	59. 82	90. 63
48	38. 29	80. 88	98	85. 42	100. 14
49	103. 15	108. 29	99	37. 97	67. 99
50	26. 88	60. 34	100	65. 18	66. 02

表 14 150 个聚类中心坐标

150 个聚类中心坐标 (单位: Km)	
(203. 05, 2842. 77)	(1145. 15, 2846. 45)
(1319. 89, 1076. 46)	(1743. 28, 970. 65)
(1913. 37, 2701. 45)	(1983. 42, 2580. 09)
(599. 45, 1734. 05)	(223. 35, 1368. 9)
(2697. 49, 2658. 86)	(1189. 02, 2213. 04)
(871. 53, 1992. 09)	(1805. 19, 2369. 89)
(2127. 32, 2267. 01)	(69. 24, 2198. 79)
(2467. 76, 2497. 17)	(2454. 37, 1883. 43)
(335. 39, 863. 07)	(1519. 08, 2168. 42)
(1697. 2, 1524. 64)	(992. 59, 1780. 72)
(1312. 93, 1507. 02)	(2145. 12, 2854. 45)
(2555. 24, 2344. 12)	(1641. 7, 2368. 55)
(585. 99, 2401. 92)	(1218. 04, 2028. 03)
(1358. 31, 2023. 76)	(40. 93, 43. 02)
(677. 26, 1258. 72)	(80. 4, 2712. 59)
(345. 35, 2839. 87)	(1008. 7, 787. 6)
(2382. 19, 2369. 54)	(2052. 75, 1356. 65)
(1268. 39, 2496. 07)	(850. 78, 2854. 5)
(429. 49, 581. 22)	(621. 71, 1421. 99)
(2694. 82, 2166. 77)	(596. 42, 813. 28)
(223. 48, 1966. 39)	(915. 82, 2535. 19)
(1733. 13, 2497. 92)	(749. 92, 2062. 05)
(474. 33, 1473. 93)	(1454. 02, 1192. 94)
(1731. 99, 1752. 88)	(2059. 75, 2718. 76)
(2308. 44, 2054. 68)	(1683. 12, 2843. 36)
(2697. 7, 2416. 44)	(796. 65, 1496. 71)
(1451. 48, 2857. 39)	(1335. 11, 2364. 31)
(1567. 01, 2789. 37)	(68. 06, 2488. 75)
(2487. 85, 2851. 04)	(794. 4, 776. 04)
(1156. 25, 1906. 56)	(388. 36, 2688. 74)
(1311. 94, 2155. 76)	(2395. 33, 2711. 23)
(1556. 93, 1486. 48)	(699. 95, 2453. 27)
(732. 09, 1781. 43)	(1452. 54, 1414. 87)
(2217. 63, 1920. 54)	(49. 74, 1093. 9)
(2171. 73, 2114. 26)	(1074. 68, 2723. 3)

(633. 04, 2841. 3)	(1894. 33, 1502. 77)
(1617. 26, 2064. 7)	(66. 58, 2846. 45)
(683. 73, 1903. 71)	(1172. 33, 2403. 98)
(1908. 7, 1386. 35)	(399. 31, 1930. 72)
(1827. 73, 2842. 62)	(1436. 75, 1575. 63)
(1656. 82, 1428. 57)	(2287. 01, 2238. 13)
(1172. 05, 2682. 89)	(2009. 2, 1527. 21)
(972. 47, 2854. 03)	(1797. 8, 2638. 16)
(358. 19, 2425. 08)	(1984. 09, 2846. 81)
(2708. 64, 2759. 71)	(1293. 78, 1201. 66)
(381. 99, 2039. 04)	(1857. 03, 2237. 01)
(239. 52, 2704. 93)	(1509. 92, 2603. 05)
(2259. 94, 2622. 02)	(356. 52, 1315. 86)
(2318. 23, 2844. 51)	(2129. 9, 2585. 26)
(1378. 09, 1729. 87)	(1712. 88, 2249. 69)
(2615. 8, 2493. 33)	(206. 6, 2424. 36)
(2556. 32, 2022. 48)	(1002. 26, 2129. 26)
(1797. 59, 1439. 38)	(1290. 64, 2835. 98)
(1394. 26, 2730. 55)	(85. 58, 2330. 93)
(457. 18, 1695. 37)	(1924. 63, 1934. 41)
(2699. 47, 2860. 77)	(438. 86, 762. 32)
(296. 6, 1158. 42)	(1273. 78, 1830. 38)
(134. 15, 2590. 09)	(2203. 14, 2739. 06)
(1568. 55, 2480. 07)	(2013. 25, 2116. 27)
(1300. 48, 1654. 97)	(1491. 92, 2354. 69)
(2053. 16, 1999. 04)	(1367. 38, 1318. 5)
(552. 94, 2673. 93)	(821. 1, 2744. 44)
(2207. 87, 2399. 25)	(1005. 9, 1948. 17)
(842. 89, 1868. 66)	(487. 34, 2831. 56)
(922, 2672. 05)	(798. 08, 2566. 52)
(1080. 36, 1607. 64)	(1637. 13, 2607. 75)
(1405. 51, 2240. 44)	(1409. 83, 2492. 07)
(859. 57, 1327. 49)	(359. 53, 1463)
(2538. 75, 2716. 42)	(1549. 02, 1048. 98)
(2474. 13, 2212. 05)	(711. 96, 1616. 25)
(1890. 64, 2486. 02)	(541. 63, 1946. 91)
(347. 76, 1003. 54)	(1968. 27, 2335. 7)
(556. 89, 1587. 5)	(2158, 1524. 13)
(2064. 36, 2441. 12)	(292. 5, 2561. 26)
(1790. 81, 1326. 38)	(1594. 55, 1634. 2)

表 15 30 架无人机飞行路径 (100 个聚点)

无人机 (1~15)	飞行路径 (H 基地)	无人机 (16~30)	飞行路径 (J 基地)
1	H-8-9-H	1	J-64-61-27-41-J
2	H-16-27-30-H	2	J-14-33-6-44-J
3	H-10-18-H	3	J-10-8-11-58-21-J
4	H-24-31-35-26-H	4	J-67-31-30-38-J
5	H-19-11-H	5	J-47-23-13-18-34-J
6	H-5-7-H	6	J-28-40-16-57-J
7	H-29-33-32-H	7	J-9-48-3-37-36-26-5-J
8	H-13-14-H	8	J-60-2-25-22-4-J
9	H-12-16-H	9	J-42-51-62-65-J
10	H-4-3-H	10	J-32-43-55-7-J
11	H-21-23-22-H	11	J-54-35-49-39-J
12	H-15-20-H	12	J-66-50-56-59-J
13	H-28-25-H	13	J-19-46-52-29-J
14	H-17-36-34-H	14	J-20-12-15-24-J
15	H-2-37-H	15	J-53-45-17-63-J

表 16 最小生成数的权

序号 i	序号 j	距离
1	2	8721.619
2	42	5380.114
42	4	3864.532
4	3	2829.372
4	34	9873.068
34	10	3724.117
10	60	4114.008
60	5	4178.12
5	35	6740.475
5	6	6760.341
6	12	6958.564
35	44	7131.662
44	28	3790.277
28	11	2492.469
11	29	6381.17
29	9	3317.591
29	25	6860.418
25	13	5731.773
13	61	5510.586
12	7	15022.16
7	8	8817.171

61	59	15391. 51
59	23	7798. 565
23	14	5612. 462
14	64	3697. 877
64	45	4612. 993
45	24	2941. 358
24	54	4588. 29
54	39	4418. 784
39	57	3130. 671
39	58	3701. 144
58	18	2813. 006
58	31	3625. 273
57	22	4438. 77
14	55	5752. 932
55	30	4092. 97
18	63	6049. 854
55	26	6601. 75
22	36	6905. 175
26	47	7200. 155
47	43	3937. 675
43	27	3462. 388
27	53	5880. 361
30	56	7877. 335
56	49	2740. 146
49	38	4604. 425
38	33	5941. 81
33	48	5044. 847
64	51	7881. 769
51	32	6756. 875
32	40	3400. 001
40	50	3640. 47
50	41	5030. 477
41	37	5056. 592
32	46	7764. 671
46	52	4480. 873
57	21	8175. 241
21	17	2834. 544
17	20	2555. 626
20	19	2568. 579
19	62	6734. 072
62	16	5717. 587
62	15	10194. 96

8.2 附录 2

第一题代码

```
clear;clc;close all
load stepThreeMat.mat
plot(1,1,'.');
hold on
plot(m,n,'.');

%画 3000 米以下区域 (标黄), 4150 禁飞区域 (标黑), 做等高线 (3000 米)
[a,b]=find(final == 1);
plot(a,b,'y.');
[c,d]=find(height>=4150);
plot(c,d,'k.');
hold on
x=1:2775;
y=1:2913;
[X,Y] = meshgrid(x,y);
contour(X,Y,height',[3000,9000]);

%聚类
label=NaN(length(a),2,7);%7 个中心圈的点放在 7 张表里, 两列放 x,y 坐标,
逐点写入, 最后把 NaN 删去
for i=1:7
    for j=1:length(a)
        if (a(j) - stdPoint(i,1))^2 + (b(j) - stdPoint(i,2))^2 <= area^2
            label(j,1,i)=a(j);
            label(j,2,i)=b(j);
        end
    end
end
circle1=label(:,:,1);%将 circle 中非数值行去掉
circle1(isnan(circle1(:,:,1)),:)=[];
circle2=label(:,:,2);
circle2(isnan(circle2(:,:,1)),:)=[];
circle3=label(:,:,3);
circle3(isnan(circle3(:,:,1)),:)=[];
circle4=label(:,:,4);
circle4(isnan(circle4(:,:,1)),:)=[];
circle5=label(:,:,5);
circle5(isnan(circle5(:,:,1)),:)=[];
circle6=label(:,:,6);
circle6(isnan(circle6(:,:,1)),:)=[];
circle7=label(:,:,7);
circle7(isnan(circle7(:,:,1)),:)=[];
```

```

%每一个 circle 中黄色区域的平均高度
averageheight=NaN(1,7);
height1=NaN(length(circle1),1);
height2=NaN(length(circle2),1);
height3=NaN(length(circle3),1);
height4=NaN(length(circle4),1);
height5=NaN(length(circle5),1);
height6=NaN(length(circle6),1);
height7=NaN(length(circle7),1);
for i=1:length(circle1)
    height1(i)=height(circle1(i,1),circle1(i,2));
end
for i=1:length(circle2)
    height2(i)=height(circle2(i,1),circle2(i,2));
end
for i=1:length(circle3)
    height3(i)=height(circle3(i,1),circle3(i,2));
end
for i=1:length(circle4)
    height4(i)=height(circle4(i,1),circle4(i,2));
end
for i=1:length(circle5)
    height5(i)=height(circle5(i,1),circle5(i,2));
end
for i=1:length(circle6)
    height6(i)=height(circle6(i,1),circle6(i,2));
end
for i=1:length(circle7)
    height7(i)=height(circle7(i,1),circle7(i,2));
end
averageheight(1)=mean(height1);
averageheight(2)=mean(height2);
averageheight(3)=mean(height3);
averageheight(4)=mean(height4);
averageheight(5)=mean(height5);
averageheight(6)=mean(height6);
averageheight(7)=mean(height7);
r=(4200-averageheight)*sqrt(3)/3/38.2;
r(6)=0;

bench = round(0.86*2*1000/38.2);
%做三个圆
for i = 1:size(stdPoint,1)

```

```

plot(stdPoint(i,1),stdPoint(i,2),'.');
axis equal
pos1 = [stdPoint(i,1) - shift stdPoint(i,2) - shift 2*shift 2*shift];
rectangle('Position',pos1,'Curvature',[1 1],'EdgeColor',[1 0 0]);
posBen = [stdPoint(i,1) - shift + bench, stdPoint(i,2) - shift + bench,
2*(shift - bench), 2*(shift - bench)];
rectangle('Position',posBen,'Curvature',[1 1],'EdgeColor',[1 0 1]);
pos2 = [stdPoint(i,1) - shift+r(i) stdPoint(i,2)+r(i) - shift 2*(shift-r(i))
2*(shift-r(i))];
rectangle('Position',pos2,'Curvature',[1 1],'EdgeColor',[0 1 0]);
pos3 = [stdPoint(i,1) - shift+2*r(i) stdPoint(i,2)+2*r(i) - shift
2*(shift-2*r(i)) 2*(shift-2*r(i))];
rectangle('Position',pos3,'Curvature',[1 1],'EdgeColor',[0 0 1]);
text(stdPoint(i,1)+20,stdPoint(i,2)+20,int2str(i));
end

% 聚类分析找点
species=20;
hold on
X = circle1;
[~,cmeans,~,~] = kmeans(X,species,'dist','sqEuclidean');% 欧式距离聚类
for i =1:species
    plot(cmeans(i,1),cmeans(i,2),'bo','markersize',2);
end
cluster(:,:,1)=cmeans;

hold on
X = circle2;
[~,cmeans,~,~] = kmeans(X,species,'dist','sqEuclidean');
for i =1:species
    plot(cmeans(i,1),cmeans(i,2),'go','markersize',2);
end
cluster(:,:,2)=cmeans;

hold on
X = circle3;
[~,cmeans,~,~] = kmeans(X,species,'dist','sqEuclidean');
for i =1:species
    plot(cmeans(i,1),cmeans(i,2),'mo','markersize',2);
end
cluster(:,:,3)=cmeans;

hold on
X = circle4;

```

```

[~,cmeans,~,~] = kmeans(X,species,'dist','sqEuclidean');
for i =1:species
    plot(cmeans(i,1),cmeans(i,2),'bo','markersize',2);
end
cluster(:,:,4)=cmeans;

hold on
X = circle5;
[~,cmeans,~,~] = kmeans(X,species,'dist','sqEuclidean');
for i =1:species
    plot(cmeans(i,1),cmeans(i,2),'co','markersize',2);
end
cluster(:,:,5)=cmeans;

%4,5 面积估算
num = 40;
d = zeros(num+2);
zoneCenter = zeros(num,2);
zoneCenter(1:20,:) = cluster(:,:,4);
zoneCenter(21:40,:) = cluster(:,:,5);
wholeCoor = [[round(110*1000/38.2) 0]; zoneCenter;[round(110*1000/38.2) 0]]
for i = 1:num+2
    for j = i+1:num+2
        d(i,j) = norm(wholeCoor(i,:)- wholeCoor(j,:));
    end
end
d = d + d';
S0=[];Sum=inf;
rand('state',sum(clock));
for j=1:10000
    S=[1 1+randperm(num), num + 2];
    temp=0;
    for i=1:num+1
        temp=temp+d(S(i),S(i+1));
    end
    if temp<Sum
        S0=S;Sum=temp;
    end
end
e=0.1^30;L=200000;at=0.99999;T=1;
%退火过程
for k=1:L
    %产生新解
    cc=2+floor(num*rand(1,2));

```

```

cc=sort(cc);
c1=cc(1);c2=cc(2);
%计算代价函数值

df=d(S0(c1-1),S0(c2))+d(S0(c1),S0(c2+1))-d(S0(c1-1),S0(c1))-d(S0(c2),S0(c2+1));
%接受准则
if df<0
    S0=[S0(1:c1-1),S0(c2:-1:c1),S0(c2+1:num+2)];
    Sum=Sum+df;
elseif exp(-df/T)>rand(1)
    S0=[S0(1:c1-1),S0(c2:-1:c1),S0(c2+1:num+2)];
    Sum=Sum+df;
end
T=T*at;
if T<e
    break;
end
end
% 输出巡航路径及路径长度
S0;
Sum = Sum - d(S0(num+1),1) - d(S0(2),1);

for i = 1:size(wholeCoor,1)
    xx = wholeCoor(S0(i),1);
    yy = wholeCoor(S0(i),2);
    plot(xx,yy,'o');
    hold on
    if(i == 2)
        text(xx+20,yy-20,'start');
    elseif (i == size(wholeCoor,1) - 1)
        text(xx+20,yy-20,'end');
    end
end
clear xx yy
xx = wholeCoor(S0(1:end-1),1);
yy = wholeCoor(S0(1:end-1),2);
plot(xx,yy,'c-*','LineWidth',2)
% Sum*38.2/1000

xishu = 38.2;
xx0 = wholeCoor(S0(2),1) * xishu;
yy0 = wholeCoor(S0(2),2) * xishu;
sumLine = 0;
for i = 3:size(wholeCoor,1)-1

```

```

if(i==4 || i == 5 || i == 20 || i == 9)
    i = i + 1;
    continue;
end
xx0 = wholeCoor(S0(i-1),1) * xishu;
yy0 = wholeCoor(S0(i-1),2) * xishu;
xx = wholeCoor(S0(i),1) * xishu;
yy = wholeCoor(S0(i),2) * xishu;
distance = norm([xx0 - xx,yy0 - yy]);
sumLine = sumLine + distance
end
sumArea = sumLine * 1.6*1000;
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
clear;clc;close all
load 1_1.mat
%第一题第二问
figure(2)
axis equal
[g,h]=find(height<=4000);
plot(g,h,'y.');
hold on
axis equal
[r,s]=find(height>=4150);
plot(r,s,'k.');

%最远正方形求解
height_yellow=NaN(length(g),1);
for i=1:length(g)
    height_yellow(i)=height(g(i),h(i));
end
averageheight_yellow=mean(height_yellow);% 第一题第二问黄色部分平均高度
r_yellow=(4200-averageheight_yellow)*sqrt(3)/3/38.2;% 平均扫描半径

%下三角作图
a=110000;
n=2913;
b=(n-1)*38.2;
yy=n;
z=zeros(1,6);
zuobiao=NaN(36,2);
plot([a/38.2,0],[0,n]);
for i=1:6

```

```

fs=@(x)(x^2+2*sqrt((a-x)^2+(b-x)^2)*2*r_yellow*38.2-8*60000*2*r_yellow*38.2);
% 定义句柄函数
x0=10000;
x=fsolve(fs,x0);
z(i)=x;
b=b-z(i);
hold on
axis equal
yy=yy-z(i)/38.2;
rectangle('Position',[0,yy,z(i)/38.2,z(i)/38.2]);
zuobiao(i,1)=x;
zuobiao(i,2)=yy*38.2;
text(zuobiao(i,1)/38.2-100,zuobiao(i,2)/38.2+100,int2str(i));
end

a=110000-zuobiao(2,1);
b=(n-1)*38.2-zuobiao(2,1);
yy=n-zuobiao(2,1)/38.2;
z=zeros(1,5);
for i=1:5

fs=@(x)(x^2+2*sqrt((a-x)^2+(b-x)^2)*2*r_yellow*38.2-8*60000*2*r_yellow*38.2);
% 定义句柄函数
x0=10000;
x=fsolve(fs,x0);
z(i)=x; % 要手动
b=b-z(i);
hold on
axis equal
yy=yy-z(i)/38.2;
if yy>0
    rectangle('Position',[zuobiao(2,1)/38.2,yy,z(i)/38.2,z(i)/38.2]);
else
    rectangle('Position',[zuobiao(2,1)/38.2,0,z(i)/38.2,z(i)/38.2]);
end
zuobiao(i+6,1)=x+zuobiao(2,1);
zuobiao(i+6,2)=yy*38.2;
text(zuobiao(i+6,1)/38.2-100,zuobiao(i+6,2)/38.2+100,int2str(i+6));
end

a=110000-zuobiao(8,1);
b=(n-1)*38.2-zuobiao(8,1);
yy=n-zuobiao(8,1)/38.2;

```

```

z=zeros(1,5);
for i=1:3

fs=@(x)(x^2+2*sqrt((a-x)^2+(b-x)^2)*2*r_yellow*38.2-8*60000*2*r_yellow*38.2);
% 定义句柄函数
    x0=10000;
    x=fsolve(fs,x0);
    z(i)=x; % 要手动
    b=b-z(i);
    hold on
    axis equal
    yy=yy-z(i)/38.2;
    rectangle('Position',[zuobiao(8,1)/38.2,yy,z(i)/38.2,z(i)/38.2]);
    zuobiao(i+11,1)=x+zuobiao(8,1);
    zuobiao(i+11,2)=yy*38.2;
    text(zuobiao(i+11,1)/38.2-100,zuobiao(i+11,2)/38.2+100,int2str(i+11));
end

a=110000-zuobiao(13,1);
b=(n-1)*38.2-zuobiao(13,1);
yy=n-zuobiao(13,1)/38.2;
z=zeros(1,5);
for i=1:2

fs=@(x)(x^2+2*sqrt((a-x)^2+(b-x)^2)*2*r_yellow*38.2-8*60000*2*r_yellow*38.2);
% 定义句柄函数
    x0=10000;
    x=fsolve(fs,x0);
    z(i)=x; % 要手动
    b=b-z(i);
    hold on
    axis equal
    yy=yy-z(i)/38.2;
    rectangle('Position',[zuobiao(13,1)/38.2,yy,z(i)/38.2,z(i)/38.2]);
    zuobiao(i+14,1)=x+zuobiao(13,1);
    zuobiao(i+14,2)=yy*38.2;
    text(zuobiao(i+14,1)/38.2-100,zuobiao(i+14,2)/38.2+100,int2str(i+14));
end

a=110000-zuobiao(16,1);
b=(n-1)*38.2-zuobiao(16,1);
yy=n-zuobiao(16,1)/38.2;
z=zeros(1,5);
for i=1:1

```

```

fs=@(x)(x^2+2*sqrt((a-x)^2+(b-x)^2)*2*r_yellow*38.2-8*60000*2*r_yellow*38.2);
% 定义句柄函数
x0=10000;
x=fsolve(fs,x0);
z(i)=x; % 要手动
b=b-z(i);
hold on
axis equal
yy=yy-z(i)/38.2;
rectangle('Position',[zuobiao(16,1)/38.2,yy,z(i)/38.2,z(i)/38.2]);
zuobiao(i+16,1)=x+zuobiao(16,1);
zuobiao(i+16,2)=yy*38.2;
text(zuobiao(i+16,1)/38.2-100,zuobiao(i+16,2)/38.2+100,int2str(i+16));
end

% 上三角转置
for i=1:5
    zuobiao(i+17,1)=(n-1)*38.2-zuobiao(i+1,2);
    zuobiao(i+17,2)=(n-1)*38.2-zuobiao(i+1,1);
end
for i=1:5

rectangle('Position',[zuobiao(i+17,1)-zuobiao(i+1))/38.2,zuobiao(i+17,2)/38.2,zuobia
o(i+1)/38.2,zuobiao(i+1)/38.2]);
    text(zuobiao(i+17,1)/38.2-100,zuobiao(i+17,2)/38.2+100,int2str(i+17));
end

for i=1:4
    zuobiao(i+22,1)=(n-1)*38.2-zuobiao(i+7,2);
    zuobiao(i+22,2)=(n-1)*38.2-zuobiao(i+7,1);
end
for i=1:4
    if zuobiao(i+22,1)/38.2<=m

rectangle('Position',[n-zuobiao(i+6,2)/38.2,zuobiao(i+22,2)/38.2,(zuobiao(i+7,1)-zuob
iao(2,1))/38.2,(zuobiao(i+7,1)-zuobiao(2,1))/38.2]);

text(zuobiao(i+22,1)/38.2-100,zuobiao(i+22,2)/38.2+100,int2str(i+22));
    else

rectangle('Position',[n-zuobiao(i+6,2)/38.2,zuobiao(i+22,2)/38.2,m-(n-zuobiao(i+6,2)/
38.2),(zuobiao(i+7,1)-zuobiao(2,1))/38.2]);
    text(m-100,zuobiao(i+22,2)/38.2+100,int2str(i+22));

```

```

    end
end

for i=1:2
    zuobiao(i+26,1)=(n-1)*38.2-zuobiao(i+12,2);
    zuobiao(i+26,2)=(n-1)*38.2-zuobiao(i+12,1);
end
for i=1:2

rectangle('Position',[n-zuobiao(i+11,2)/38.2,zuobiao(i+26,2)/38.2,(zuobiao(i+12,1)-zu
obiao(8,1))/38.2,(zuobiao(i+12,1)-zuobiao(8,1))/38.2]);
text(zuobiao(i+26,1)/38.2-100,zuobiao(i+26,2)/38.2+100,int2str(i+26));
end

for i=1:1
    zuobiao(i+28,1)=(n-1)*38.2-zuobiao(i+15,2);
    zuobiao(i+28,2)=(n-1)*38.2-zuobiao(i+15,1);
end
for i=1:1

rectangle('Position',[n-zuobiao(i+14,2)/38.2,zuobiao(i+28,2)/38.2,(zuobiao(i+15,1)-zu
obiao(13,1))/38.2,(zuobiao(i+15,1)-zuobiao(13,1))/38.2]);
text(zuobiao(i+28,1)/38.2-100,zuobiao(i+28,2)/38.2+100,int2str(i+28));
end

% 处理 1 区域
d1=zuobiao(1,1)/38.2;
square_01=zeros(m,n);
for i = 1:m
    for j = 1:n
        if i<=d1 && j>=n-d1
            square_01(i,j) = 1;
        end
    end
end
[a,b]=find(square_01 == 1);
square1=NaN(length(a),2);
for i=1:length(a)
    square1(i,1)=a(i);
    square1(i,2)=b(i);
end

sq1_center=mean(square1);
number=floor(d1/r_yellow/2);

```

```

line_x=linspace(0,d1,number+1);
linex=mean([line_x(1:end-1);line_x(2:end)],1);
line_y=linspace(n-d1,n,number+1);
liney=mean([line_y(1:end-1);line_y(2:end)],1);
lujing1=NaN(24,2);
for i=1:24
    if mod(i,4)==1
        lujing1(i,1)=linex(13-ceil(i/4)+1);
        lujing1(i,2)=liney(ceil(i/4));
    elseif mod(i,4)==2
        lujing1(i,1)=linex(13-ceil(i/4)+1);
        lujing1(i,2)=liney(13-ceil(i/4)+1);
    elseif mod(i,4)==3
        lujing1(i,1)=linex(ceil(i/4));
        lujing1(i,2)=liney(13-ceil(i/4)+1);
    elseif mod(i,4)==0
        lujing1(i,1)=linex(ceil(i/4));
        lujing1(i,2)=liney(ceil(i/4));
    end
end
hold on
plot(lujing1(:,1),lujing1(:,2),'r','LineWidth',2);

%处理 2 区域
height_4000=NaN(m,n);
for i = 1:m
    for j = 1:n
        if(height(i,j)<=4150)
            height_4000(i,j) = 1;
        end
    end
end
d2=zuobiao(2,1)/38.2;
square_2=zeros(m,n);
for i = 1:m
    for j = 1:n
        if      i<=zuobiao(2,1)/38.2      &&      j<=zuobiao(1,2)/38.2      &&
j>zuobiao(2,2)/38.2 && height_4000(i,j)==1
            square_2(i,j) = 1;
        end
    end
end
[c,d]=find(square_2 == 1);
square2=NaN(length(c),2);

```

```

for i=1:length(c)
    square2(i,1)=c(i);
    square2(i,2)=d(i);
end

species=100;
hold on
X = square2;
[~,cmeans,~,~] = kmeans(X,species,'dist','sqEuclidean');%欧式距离聚类
for i =1:species
    plot(cmeans(i,1),cmeans(i,2),'bo','markersize',2);
end

%路径规划
num = 100;
d = zeros(num+2);
zoneCenter = cmeans;
wholeCoor = [[zuobiao(2,1)/38.2 zuobiao(2,2)/38.2];
zoneCenter;zuobiao(2,1)/38.2 zuobiao(2,2)/38.2]];
for i = 1:num+2
    for j = i+1:num+2
        d(i,j) = norm(wholeCoor(i,:)- wholeCoor(j,:));
    end
end
d = d + d';
S0=[];Sum=inf;
rand('state',sum(clock));
for j=1:10000
    S=[1 1+randperm(num), num + 2];
    temp=0;
    for i=1:num+1
        temp=temp+d(S(i),S(i+1));
    end
    if temp<Sum
        S0=S;Sum=temp;
    end
end
e=0.1^30;L=200000;at=0.99999;T=1;
%退火过程
for k=1:L
    %产生新解
    cc=2+floor(num*rand(1,2));
    cc=sort(cc);
    c1=cc(1);c2=cc(2);

```

```

%计算代价函数值

df=d(S0(c1-1),S0(c2))+d(S0(c1),S0(c2+1))-d(S0(c1-1),S0(c1))-d(S0(c2),S0(c2+1));
%接受准则
if df<0
    S0=[S0(1:c1-1),S0(c2:-1:c1),S0(c2+1:num+2)];
    Sum=Sum+df;
elseif exp(-df/T)>rand(1)
    S0=[S0(1:c1-1),S0(c2:-1:c1),S0(c2+1:num+2)];
    Sum=Sum+df;
end
T=T*at;
if T<e
    break;
end
end
% 输出巡航路径及路径长度
S0;
Sum = Sum - d(S0(num+1),1);

for i = 1:size(wholeCoor,1)
    xx = wholeCoor(S0(i),1);
    yy = wholeCoor(S0(i),2);
    plot(xx,yy,'o');
    hold on
    if(i == 2)
        text(xx+20,yy-20,'start');
    elseif (i == size(wholeCoor,1) - 1)
        text(xx+20,yy-20,'end');
    end
end
clear xx yy
xx = wholeCoor(S0,1);
yy = wholeCoor(S0,2);
plot(xx,yy,'r-*','LineWidth',2)
% Sum*38.2/1000

xishu = 38.2;
xx0 = wholeCoor(S0(2),1) * xishu;
yy0 = wholeCoor(S0(2),2) * xishu;
sumLine = 0;
for i = 3:size(wholeCoor,1)-1
    if(i==4 || i == 8 || i == 9)
        i = i + 1;

```

```

        continue;
    end
    xx0 = wholeCoor(S0(i-1),1) * xishu;
    yy0 = wholeCoor(S0(i-1),2) * xishu;
    xx = wholeCoor(S0(i),1) * xishu;
    yy = wholeCoor(S0(i),2) * xishu;
    distance = norm([xx0 - xx,yy0 - yy]);
    sumLine = sumLine + distance;
end
sumArea = sumLine * 1.6*1000;

```

第二题代码

```

function varargout =
mtspf_ga(xy,dmat,salesmen,min_tour,pop_size,num_iter,show_prog,show_res)

nargs = 8;
for k = nargin:nargs-1
    switch k
        case 0
            xy = 10*rand(40,2);
        case 1
            N = size(xy,1);
            a = meshgrid(1:N);
            dmat = reshape(sqrt(sum((xy(a,:)-xy(a',:)).^2,2)),N,N);
        case 2
            salesmen = 5;
        case 3
            min_tour = 2;
        case 4
            pop_size = 80;
        case 5
            num_iter = 5e3;
        case 6
            show_prog = 1;
        case 7
            show_res = 1;
        otherwise
    end
end

% Verify Inputs
[N,dims] = size(xy);
[nr,nc] = size(dmat);
if N ~= nr || N ~= nc

```

```

    error('Invalid XY or DMAT inputs!')
end
n = N - 1; % Separate Start/End City

% Sanity Checks
salesmen = max(1,min(n,round(real(salesmen(1))))));
min_tour = max(1,min(floor(n/salesmen),round(real(min_tour(1)))));
pop_size = max(8,8*ceil(pop_size(1)/8));
num_iter = max(1,round(real(num_iter(1)))); 
show_prog = logical(show_prog(1));
show_res = logical(show_res(1));

% Initializations for Route Break Point Selection
num_brks = salesmen-1;
dof = n - min_tour*salesmen; % degrees of freedom
addto = ones(1,dof+1);
for k = 2:num_brks
    addto = cumsum(addto);
end
cum_prob = cumsum(addto)/sum(addto);

% Initialize the Populations
pop_rte = zeros(pop_size,n); % population of routes
pop_brk = zeros(pop_size,num_brks); % population of breaks
for k = 1:pop_size
    pop_rte(k,:) = randperm(n)+1;
    pop_brk(k,:) = randbreaks();
end

% Select the Colors for the Plotted Routes
clr = [1 0 0; 0 0 1; 0.67 0 1; 0 1 0; 1 0.5 0];
if salesmen > 5
    clr = hsv(salesmen);
end

% Run the GA
global_min = Inf;
total_dist = zeros(1,pop_size);
dist_history = zeros(1,num_iter);
tmp_pop_rte = zeros(8,n);
tmp_pop_brk = zeros(8,num_brks);
new_pop_rte = zeros(pop_size,n);
new_pop_brk = zeros(pop_size,num_brks);
route = zeros(salesmen,size(xy,1));

```

```

if show_prog
    pfig = figure('Name','MTSPF_GA' | Current Best
Solution','Numbertitle','off');
end
for iter = 1:num_iter
    % Evaluate Members of the Population
    for p = 1:pop_size
        d = 0;
        p_rte = pop_rte(p,:);
        p_brk = pop_brk(p,:);
        rng = [[1 p_brk+1];[p_brk n]]';
        for s = 1:salesmen
            d = d + dmat(1,p_rte(rng(s,1))); % Add Start Distance
            for k = rng(s,1):rng(s,2)-1
                d = d + dmat(p_rte(k),p_rte(k+1));
            end
            d = d + dmat(p_rte(rng(s,2)),1); % Add End Distance
        end
        total_dist(p) = d;
    end

    % Find the Best Route in the Population
    [min_dist,index] = min(total_dist);
    dist_history(iter) = min_dist;
    if min_dist < global_min
        global_min = min_dist;
        opt_rte = pop_rte(index,:);
        opt_brk = pop_brk(index,:);
        rng = [[1 opt_brk+1];[opt_brk n]]';
        if show_prog
            % Plot the Best Route
            figure(pfig);
            for s = 1:salesmen
                rte = [1 opt_rte(rng(s,1):rng(s,2)) 1];
                route(s,1:length(rte)) = rte;
                if dims == 3,
                    plot3(xy(rte,1),xy(rte,2),xy(rte,3),'-','Color',clr(s,:));
                else plot(xy(rte,1),xy(rte,2),'-','Color',clr(s,:));end
                title(sprintf('Total Distance = %1.4f, Iteration
= %d',min_dist,iter));
            hold on
        end
        if dims == 3, plot3(xy(1,1),xy(1,2),xy(1,3),'ko');
        else plot(xy(1,1),xy(1,2),'ko'); end
    end

```

```

        hold off
    end
end

% Genetic Algorithm Operators
rand_grouping = randperm(pop_size);
for p = 8:8:pop_size
    rtes = pop_rte(rand_grouping(p-7:p),:);
    brks = pop_brk(rand_grouping(p-7:p),:);
    dists = total_dist(rand_grouping(p-7:p));
    [ignore,idx] = min(dists);
    best_of_8_rte = rtes(idx,:);
    best_of_8_brk = brks(idx,:);
    rte_ins_pts = sort(ceil(n*rand(1,2)));
    I = rte_ins_pts(1);
    J = rte_ins_pts(2);
    for k = 1:8 % Generate New Solutions
        tmp_pop_rte(k,:) = best_of_8_rte;
        tmp_pop_brk(k,:) = best_of_8_brk;
        switch k
            case 2 % Flip
                tmp_pop_rte(k,I:J) = fliplr(tmp_pop_rte(k,I:J));
            case 3 % Swap
                tmp_pop_rte(k,[I J]) = tmp_pop_rte(k,[J I]);
            case 4 % Slide
                tmp_pop_rte(k,I:J) = tmp_pop_rte(k,[I+1:J I]);
            case 5 % Modify Breaks
                tmp_pop_brk(k,:) = randbreaks();
            case 6 % Flip, Modify Breaks
                tmp_pop_rte(k,I:J) = fliplr(tmp_pop_rte(k,I:J));
                tmp_pop_brk(k,:) = randbreaks();
            case 7 % Swap, Modify Breaks
                tmp_pop_rte(k,[I J]) = tmp_pop_rte(k,[J I]);
                tmp_pop_brk(k,:) = randbreaks();
            case 8 % Slide, Modify Breaks
                tmp_pop_rte(k,I:J) = tmp_pop_rte(k,[I+1:J I]);
                tmp_pop_brk(k,:) = randbreaks();
            otherwise % Do Nothing
        end
    end
    new_pop_rte(p-7:p,:) = tmp_pop_rte;
    new_pop_brk(p-7:p,:) = tmp_pop_brk;
end
pop_rte = new_pop_rte;

```

```

pop_brk = new_pop_brk;
end

if show_res
    % Plots
    figure('Name','MTSPF_GA | Results','Numbertitle','off');
    subplot(2,2,1);
    if dims == 3, plot3(xy(:,1),xy(:,2),xy(:,3),'k.');
    else plot(xy(:,1),xy(:,2),'k.'); end
    title('City Locations');
    subplot(2,2,2);
    imagesc(dmat([1 opt_rte],[1 opt_rte]));
    title('Distance Matrix');
    subplot(2,2,3);
    rng = [[1 opt_brk+1];[opt_brk n]]';
    for s = 1:salesmen
        rte = [1 opt_rte(rng(s,1):rng(s,2)) 1];
        if dims == 3, plot3(xy(rte,1),xy(rte,2),xy(rte,3),'.-' , 'Color',clr(s,:));
        else plot(xy(rte,1),xy(rte,2),'.-' , 'Color',clr(s,:)); end
        title(sprintf('Total Distance = %1.4f,min_dist));
        hold on;
    end
    if dims == 3, plot3(xy(1,1),xy(1,2),xy(1,3),'ko');
    else plot(xy(1,1),xy(1,2),'ko'); end
    subplot(2,2,4);
    plot(dist_history,'b','LineWidth',2);
    title('Best Solution History');
    set(gca,'XLim',[0 num_iter+1],'YLim',[0 1.1*max([1 dist_history])]);
end

% Return Outputs
if narginout
    varargout{1} = opt_rte;
    varargout{2} = opt_brk;
    varargout{3} = min_dist;
    varargout{4} = route;
end

% Generate Random Set of Break Points
function breaks = randbreaks()
    if min_tour == 1 % No Constraints on Breaks
        tmp_brks = randperm(n-1);
        breaks = sort(tmp_brks(1:num_brks));
    else % Force Breaks to be at Least the Minimum Tour Length

```

```

num_adjust = find(rand < cum_prob,1)-1;
spaces = ceil(num_brks*rand(1,num_adjust));
adjust = zeros(1,num_brks);
for kk = 1:num_brks
    adjust(kk) = sum(spaces == kk);
end
breaks = min_tour*(1:num_brks) + cumsum(adjust);
end
end
%% %%%%%%
load cmeans_150
num = 150;
dlow = zeros(num+2);
dhigh = zeros(num+2);
zoneCenter = cmeans;

wholeCoor = [[round(110*1000/38.2) round(55*1000/38.2)]; ...
    zoneCenter;[round(110*1000/38.2) round(55*1000/38.2)]]]
for i = 1:num+2
    for j = i+1:num+2
        dhigh(i,j) = norm(wholeCoor(i,:)- wholeCoor(j,:));
    end
end
dhigh = dhigh + dhigh';

wholeCoor = [[round(110*1000/38.2) 0]; ...
    zoneCenter;[round(110*1000/38.2) 0]]]
for i = 1:num+2
    for j = i+1:num+2
        dlow(i,j) = norm(wholeCoor(i,:)- wholeCoor(j,:));
    end
end
dlow = dlow + dlow';
disH = dhigh(2:151,1);
disL = dlow(2:151,1);
[val,index] = sort(disL);
numL = index(1:60);
numH = find(~ismember(1:150,numL)==1);
lengthL = length(numL) +2;
lengthH = length(numH) +2;
disLow = zeros(lengthL);
disHigh = zeros(lengthH);

```

```

wholeCoorLow = [[round(110*1000/38.2) 0]; ...
    zoneCenter(numL,:);[round(110*1000/38.2) 0]];
for i = 1:lengthL
    for j = i+1:lengthL
        disLow(i,j) = norm(wholeCoorLow(i,:)-wholeCoorLow(j,:));
    end
end
disLow = disLow + disLow';

wholeCoorHigh = [[round(110*1000/38.2) round(55*1000/38.2)]; ...
    zoneCenter(numH,:);[round(110*1000/38.2) round(55*1000/38.2)]];
for i = 1:lengthH
    for j = i+1:lengthH
        disHigh(i,j) = norm(wholeCoorHigh(i,:)-wholeCoorHigh(j,:));
    end
end
disHigh = disHigh + disHigh';

%
mtspf_ga(xy,dmat,salesmen,min_tour,pop_size,num_iter,show_prog,show_res)
    [trail1,brk1,minDis1,route1]
mtspga(wholeCoorLow,disLow,15,2,80,5000,true,true);
    [trail2,brk2,minDis2,route2]
mtspga(wholeCoorHigh,disHigh,15,4,80,5000,true,true);
    %plot(0,0,'.');
    hold on
    %%%%%%%%%%%%%%
load cmeans_150
load stepThreeMat
load num

hx = [110*1000/38.2 110*1000/38.2];
hy = [0 55*1000/38.2];
plot(hx(1),hy(1),'r+');
hold on
plot(hx(2),hy(2),'b+');
legend('H','J')
text(hx(1),hy(1)+60,'H');
text(hx(2),hy(2)+60,'J');
[m,n] = find(height <= 3000);
plot(m,n,'y.');
title('红色代表 H 区, 蓝色代表 J 区')
X = [m,n];

```

```

num = 150;
% [cidx,cmeans,sumd,D] = kmeans(X,num,'dist','sqEuclidean');
% ptsymb = {'bo','ro','go','ko','co'};
% MarkFace = {[0 0 1],[.8 0 0],[0 .5 0],[1,0,1],[0,0.5,0.5]};
for i = 1:num
    % clust = find(cidx == i);
    plot(cmeans(i,1),cmeans(i,2),'bo','markersize',5);
    hold on
end
zoneCenter = cmeans;
plot(1,1,'.');
hold on;
plot(3000,3000,'.');
for i = 1:size(zoneCenter,1)
    if sum(ismember(numL,i)) == 1
        plot(zoneCenter(i,1),zoneCenter(i,2),'r','markersize',5);
    else
        plot(zoneCenter(i,1),zoneCenter(i,2),'b','markersize',5);
    end
    text(zoneCenter(i,1)+20,zoneCenter(i,2)+20,int2str(i));
end
xlabel('x 方向网格数');
ylabel('y 方向网格数');

```

第三题代码

```

load terminal
load height
terminal = terminal.*1000/38.2;
idx = find(terminal(:,2)>2913);
terminal(idx,:) = []
for i = 1:length(terminal)
    terH(i) = height(round(terminal(i,1)),round(terminal(i,2)));
end
terminal = [terminal terH'];
pos = terminal;
pos(:,1:2) = pos(:,1:2).*38.2;

thDis = zeros(69);
for i = 1:69
    for j = i+1:69
        thDis(i,j) = norm(pos(i,:)-pos(j,:));
    end
end

thDis = thDis + thDis';

```

```

thDis = thDis + 100000*eye(69);

[inda,indb] = find(thDis<=2000);
indb = [indb inda];
indstrip=indb;
nixu = find(indstrip(:,2)<indstrip(:,1));
indstrip(nixu,:) = []

indf = indstrip;
% 第一类中心
clsf = find(~ismember(1:69,unique(indf))==1);
clsf = clsf';
% 第二类中心 可直接算出位置
uavpos = pos(clsf,:);

for i = 1:5
    mid = (pos(indstrip(i,1),:) + pos(indstrip(i,2),:))/2;
    uavpos = [uavpos;mid];
end

[ms,ns] = size(uavpos);
costmtx = zeros(ms);
for i = 1:ms
    for j = i+1:ms
        costmtx(i,j) = norm(uavpos(i,:)-uavpos(j,:));
    end
end
costmtx = costmtx + costmtx';

a = costmtx;a(find(a==0))=inf;
result=[];p=1;tb=2:length(a);
while length(result)~>length(a)-1
    temp=a(p,tb);temp=temp(:);
    d=min(temp);
    [jb,kb] = find(a(p,tb)==d);
    j=p(jb(1));k=tb(kb(1));
    result=[result,[j;k;d]];p=[p,k];tb(find(tb==k))=[];
end
idx6 = find(result(3,:)>6000);
idx6 = idx6';
[X,Y]=meshgrid(500:10:2500,250:10:2800);
Z=griddata(terminal(:,1),terminal(:,2),terminal(:,3),X,Y);
mesh(X,Y,Z)
hold on

```

```

for i = 1:size(terminal,1)
    plot3(terminal(i,1),terminal(i,2),terminal(i,3),'','markersize',6);

    plot3(terminal(i,1),terminal(i,2),terminal(i,3)+300,'k*','markersize',6);
    text(terminal(i,1)+20,terminal(i,2)+20,terminal(i,3)+100,int2str(i));
end
xlabel('x 方向网格');
ylabel('y 方向网格');
zlabel('z 轴/m');

```

第四题代码 solveFour.m

```

function [v t] = montecarlo2(b)
syms v t
rou0 = 4.314*1e-10;
% b = 140*1e3;
b = b*1e3;
to = 3000*sqrt(3)/v;
rt = b*log2(1+5/(rou0*(1500^2 + (v*t -1500*sqrt(3))^2)));
rtold = rt;

for v = 60/3.6:0.1:100/3.6
    t1 = subs(to,v);
    rt = subs(rt,v);
    if(int(rt,0,t1)>=500*1e6)
        rt = rtold;
        continue;
    else
        break;
    end
end
if (v ~= 60/3.6)
    v;
    t = subs(to);
end
%%%%%%%%%%%%%
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
load stepThreeMat
load terminalHeight
terminal = [terminal(:,1:2)*38.2 terminal(:,3)];
terminal = [[110*1000 0 0];terminal;[110*1000 0 0]];
[m,n] = size(terminal);
indexd = [50 52 63];
fre(indexd) = [];

```

```

unifre = unique(fre);
thDis = zeros(m);
for i = 1:m
    for j = i+1:m
        thDis(i,j) = norm(terminal(i,:)-terminal(j,:));
    end
end
thDis = thDis + thDis';
for i = 1:size(fre,1)
    if(fre(i)>=140)
        [v,t] = montecarlo2(fre(i));
        fre(i,2) = v;
        fre(i,3) = t;
    else
        [v,t] = lower140(fre(i));
        fre(i,2) = v;
        fre(i,3) = t;
    end
end
[opt_rte,opt_brk,min_dist,route] = mtspga(terminal,thDis,3,12,100,8000,1,1);
%%%%%
function [v t] = lower140(b)
    b = b * 1e3;
    rou0 = 4.314*1e-10;
    rt = b*log2(1+5/(rou0*3000*3000));
    t = 500*1e6/rt;
    v = 1500*sqrt(3)*pi/t;
end
%%%%%
disIndividualLow = zeros(3,1);
for i = 1:3
    single = route(i,:);
    index = find(single == 1);
    singleRoute = single(index(1):index(2));
    for j = 2:length(singleRoute)
        disIndividualLow(i) = disIndividualLow(i) +
            thDis(singleRoute(j-1),singleRoute(j));
        if(thDis(singleRoute(j-1),singleRoute(j)) <= 3*sqrt(3))
            disp(i);
            disp(j);
        end
    end
end

```