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	上海海事大学
参赛队号	10254014
队员姓名	1.徐凤新
	2.朱若琪
	3.吴　方

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运用

摘 要：

本文对无人机在抢险救灾中的优化运用进行任务规划和研究。建立了无人机在在灾情巡查、生命迹象探测、灾区通信中继、对地数据传输多种限定条件下无人机相互配合完成任务的数学模型，解决了不同的组合规划问题，并对该数学模型根据条件变化进行相应改进，最后对该数学模型进行分析和讨论，能够显著提升无人机在抢险救灾中的勘测能力。

针对问题一，首先考虑将 3000 米以下海拔灾区的勘测问题简化，利用最优化模型对震区 7 个重点区域进行分析，通过分析每个重点区域中心目标点以及基地 H 的相对位置，简化为无人机从无人基地起飞遍历 7 个中心目标点的最优路径选择问题，将模型抽象为不确定数量的多起点开环多旅行商问题（MDO_MTSP），运用数学建模中的遗传算法模型，考虑遗传算法解决多旅行商问题这一思路，反复迭代后最终求得无人机在四小时的时间内尽可能覆盖最大灾区面积的最优飞行路径；

针对问题二，首先提取灾区的 3000 米以下地区部分，然后将所得图像进行灰度处理和二值化，所得到的图像仅有黑色和白色两部分，可将问题简化为，遍历所有白色区域所用的时间最短的问题。因为有 30 架飞机进行遍历，所以将整个需要遍历的区域划分为 30 块，然后再运用模拟退火算法和蚁群算法对某一区域进行路径规划，通过比较选择了相对更稳定的蚁群算法。将蚁群算法扩展到所

有的 30 个区域，最终得到所有无人机的飞行路径，以及完成任务所需的时间为 4.9 小时。

针对问题三，经过分析，综合考虑太阳能无人机在执行通信中继任务时，通信中继无人机与地面移动终端通信距离的限定范围，运用粒子群算法将 72 个地面移动终端划分为 11 个区域，然后在运用线性回归组合模型，反复验证后最终求得完成通信中继任务所需无人机的架次为 57，以及各个区域内无人机的飞行路线。

针对问题四，在问题三的路径规划基础上，将 11 个区域再次进行分配，保证地面移动终端平均分配给三架数据传输无人机。经过粒子群算法的迭代计算，最终确定各个数据传输无人机所遍历的终端个数分别为 27，25，20。经过对问题四所给条件的分析可以得出，问题四是一个多约束条件的非线性规划问题，需要综合考虑数据传输无人机的传输范围、飞行速度和地面终端的信道带宽进行无人机路径的规划。最终得到各个地面终端的功率分配、三架无人机的飞行路线、三架无人机完成任务所需时间。经过计算可得三架无人机完成任务分别需要 6.433 小时、3.687 小时和 3.511 小时。

关键词： 多无人机路径规划　粒子群算法　遗传算法　非线性规划

目 录

1　问题的重述

1.1　问题的背景

无人机（Unmanned Aerial Vehicle，UAV）是一种具备自主飞行和独立执行任务能力的新型勘测及信息传递平台，不仅能够执行对地攻击和目标轰炸等作战任务，而且还能够执行军事侦察、监视、搜索、目标指向等非攻击性任务。随着无人机技术的快速发展，越来越多的无人机应用在救灾现场。

无人机与传统人力信息传递相比，有以下较为突出的特点[1]：能够避免飞行员在危险环境下作业，具有伤亡率低甚至零伤亡的特点；无需考虑机载生命的影响，为进一步实现飞行器的机动性、低可探测性、持续作战能力等提供了可能；降低了飞行器系统的复杂性，使其研发、制造、装备、使用和维护的难度和成本大大低于有人机；激发和拓展更多样的任务形式和使用需求等。

基于无人机的以上特点，以及进入 21 世纪以来，随着计算机科学与技术、材料、传感器与数字图像处理技术的飞速发展，综合在近年来如叙利亚、伊拉克等战争使用无人机的次数和任务形式从侦察到空对地进行打击的转变，可以预见，在灾区营救方面，越来越多的无人机将介入。

根据目前的无人机发展趋势，虽然单架无人机的侦察、搜索能力越来越高，功能越来越强大，但面对救灾时间的紧急以及多样化的需求，单架无人机也会暴露容错性不足等短板，为弥补其局限性，无人机应当以集群的形式协同工作，即由多架无人机协同，提高任务执行效率，扩展任务执行方式。

1.2　要解决的问题

2017 年 8 月 8 日，四川阿坝州九寨沟县发生 7.0 级地震，造成了不可挽回的人员伤亡和重大的财产损失。由于预测地震比较困难，及时高效的灾后救援是减少地震损失的重要措施。无人机作为一种新型运载工具，能够在救援行动中发挥重要作用。附件 1 给出了震区的高程数据，共有 2913 列，2775 行。第一行第一列表示(0,0)点处的海拔高度值(单位：米)，相邻单元格之间的距离为 38.2 米，即第 m 行第 n 列单元格中的数据代表坐标(38.2(m-1), 38.2(n-1))处的高度值。本题中的无人机都假设平均飞行速度 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±15°，与其它障碍物(含地面)的安全飞行距离不小于 50 米，最大飞行高度为海拔 5000 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。

需要通过建立数学模型，解决以下几个问题：

问题一：大地震发生后，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。无人机飞行高度恒为 4200 米，将在地面某点看无人机的仰角大于 60° 且视线不被山体阻隔视为该点被巡查。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，在 4 小时之内利用最少的无人机使区域 S 内海拔 3000 米以下的地方尽可能多地被巡查到，使覆盖率最大，每架无人机飞行路线最短。并且在论文中画出相应的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。使用最少的无人机在最短时间和路线中完成任务，保证在 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(无人机均需从 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)。

问题二：使用无人机携带生命探测仪搜索生命迹象，给灾后救援提供准确的目标定位。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)，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，使附件 1 所给出的全区域内海拔 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：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，规划出在最短路线，利用最少架无人机，确保在白天 12 小时内，附件 2 中的任意两个地面终端之间都能实现不间断通信(作为中继的无人机之间的切换时间忽略不计，地面终端的移动距离不超过 2 千米)。

问题四：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(分布见附件 2)发送内容不同，总量均为 500M(1M 按 10^6 比特计算)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。数据传输过程可以简化为：当地面终端 i 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，如果无人机当前服务用户少于 10 个，则开始向 i 发送数据，并瞬间完成所有用户的功率再分配，否则，搁置 i 的需求，直到有地面用户退出，若此时 i 仍在可服务区域，则为 i 服务（先到先服务）。如果在一个服务时间区间（即无人机和终端之间满足可传输数据条件的时间范围）内不能传完全部数据，则以后区间可以续传。再设 i 用户在时刻 t 接收到无人机发送的信息速率为 $r_i(t)=B_i\log_2\left(1+\dfrac{p_i(t)}{\rho_0 d^2(u,i)}\right)$(比特/秒)，其中 B_i 表示无人机服务 i 的子信道带宽(取值见附件 2，单位 Hz)，$p_i(t)$ 表示 t 时刻无人机为第 i 个地面用户所在的子信道分配的功率,单位：w(瓦)，$d(u,i)$ 表示 t 时刻无人机与 i 之间的欧氏距离，单位：米。ρ_0 为信道特性参数，为简单起见，取为 4.314×10^{-10}(单位略)，假设无人机飞行速度在 60~100 千米/小时之间可调(水平面内最大加速度±5 米/秒 ^{2}，铅垂面内最大加速±2 米/秒 ^{2}，可同时在两个方向上加速)，为无人机设计恰当的航线、速度以及所服务的用户，并为每一个用户分配恰当的功率，使得无人机完成所有任务的时间总和尽量短。

2 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。

- **针对问题一：**

假设（1）：为了简化问题模型，在问题一，方圆 10 公里以半径为五公里来计算。

假设（2）：忽略无人机大小，将无人机简化为质点。无人机的巡查半径为 2500 米（原本半径为 2400 米）。

假设（3）：不存在灾区环境对无人机信号，视频采集等功能产生干扰。

假设（4）：4 小时内指的是无人机从基地 H 出发直至完成巡查任务（根据实际情况，续航时间为 8 小时，之多四小时用于巡查，至少四小时用于返回）。

假设（5）：无人机从 H 基地起飞后原地上升至 4200 米。

假设（6）：题目震区区域外的地方无人机均可顺利通过。

- **针对问题二：**

假设（1）： 飞机始终与被测地面保持 500 米的距离。

假设（2）： 飞行高度一定，尽量减少俯冲和攀升，转弯。

假设（3）： 当飞机在非被测区域飞行时始终以 5000 米高度飞行。

- **针对问题三：**

假设（1）： 忽略地面移动终端的移动，即保持静止。

假设（2）： 忽略天气环境对无人机的影响，即无人机一直保持有电状态，并且电量充足，飞行速度不变。

- **针对问题四：**

假设（1）： 无人机在 8 号、9 号、50 号、52 号和 63 号地面移动终端的上空 50 米处进行数据传输，与其余移动终端的数据传输距离为 3000 米。

假设（2）：无人机给 8 号、9 号、50 号、52 号和 63 号地面移动终端分配 5 瓦的功率，给其余点分配的功率均为 2.5 瓦。

3 符号说明

符号	含义
L_i	无人机飞行路径长度
H_i	每架无人机的飞行时间
F	地面终端数据的大小
t	地面终端完成所有数据接收的时间

4 问题的分析

4.1 对问题一的分析

4.1.1 问题描述及分析

问题一是研究无人机携带视频采集装置巡查 7 个重点区域中心方圆 10 公里以内的灾情。无人机在 4200 米的高空飞行，地面某点看无人机的仰角大于 60 度，此时无人机勘测的最大半径为 2400 米，无人机的起始点为 H，考虑在这种理想情况下无人机完成任务的最佳路线和无人机调度策略，主要考虑如何在顺利完成侦察任务的情况下，保证无人机勘测覆盖率最大、飞行路线最短。

根据题目和参考文献，给出最优路径及无人机调度策略如下：

（1）由于重点灾区的半径为 5000 米，为了尽可能完成灾区的覆盖，此时我们让无人机围绕以重点灾区中心为原点半径为 2500 米的圆周进行勘测。由于灾后抢险时间急迫，我们应该选择出最优路径，在有限的四小时时间内完成巡查。

（2）通过查询海拔在 3000 米以下的重点区域有 A、B、C、D、E，因此我们仅需考虑这五个灾区。此时我们通过建模选择最佳无人机数量和最优路线。

（3）通过查询海拔在 4000 米以下的重点区域，我们通过建模选择最佳无人机数量和最优路线。

（4）以上问题中，不考虑无人机大小，无人机为质点。

（5）比较在图中用来寻找优化路径的蚁群算法与模拟自然进化过程的遗传算法在解决组合优化问题的实际表现，最终选定遗传算法作为求解问题一的主要策略。

4.1.2 数据处理

根据附件 1 对灾区海拔数据进行处理。

（1） 震区 7 个重点区域的中心位置如下表所示：

表 1 震区 7 个重点区域的中心位置坐标

中心点	X 坐标	Y 坐标
A	30.3	89.8
B	66.0	84.7
C	98.4	76.7
D	73.7	61.0
E	57.9	47.6
F	86.8	22.0
G	93.6	48.8

（2）利用数据画出三维地形图：

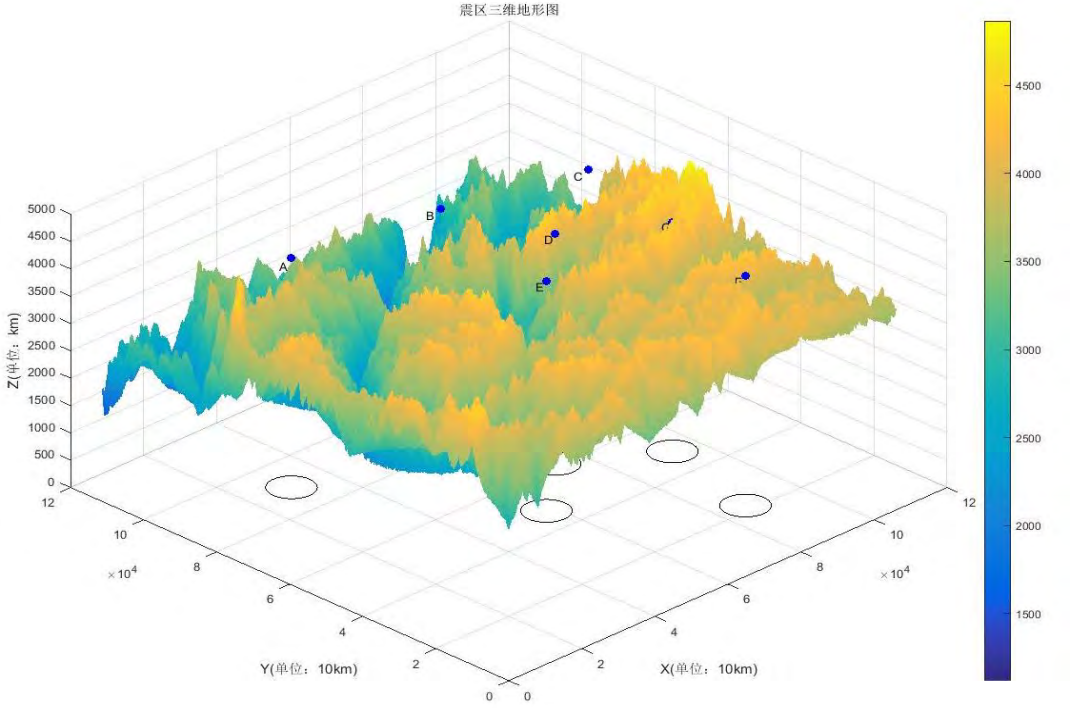


图 1　灾区三维地形图

（3）利用数据画出灾区三维地形加重图，并且画出以重灾区中心为圆心半径 5000 米的圆周：

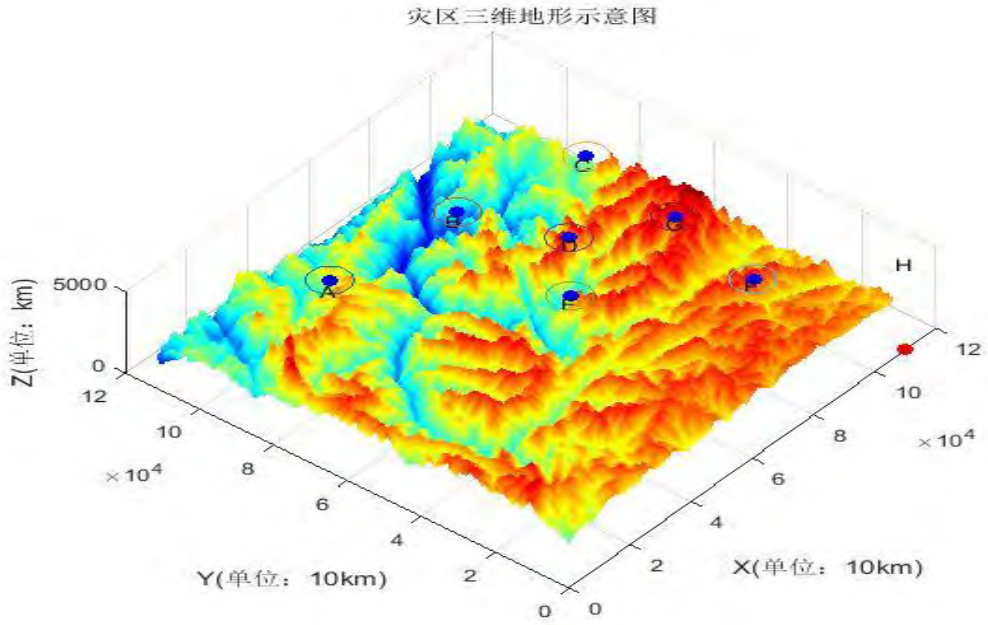


图 2 灾区三维加重图

（4）灾区地形信息，把数据进行海拔高度的分化，利于问题求解：

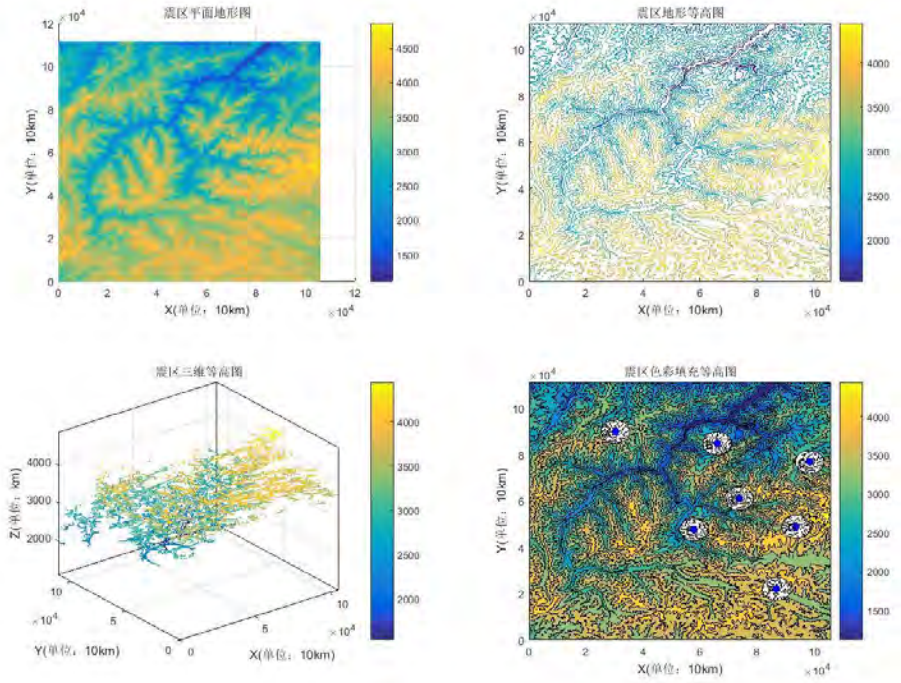


图 3 灾区地形信息

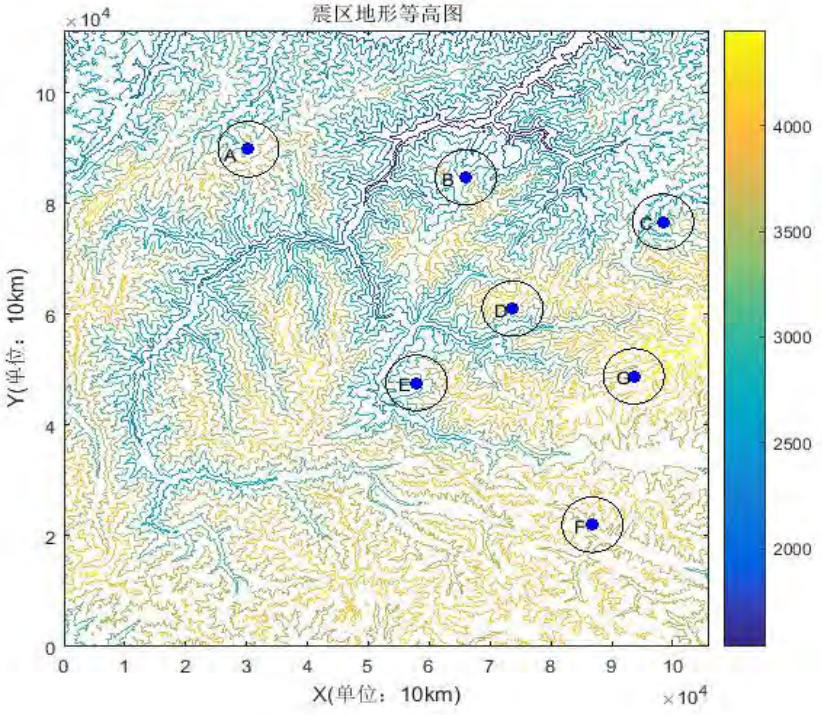


图 4 震区地形等高图

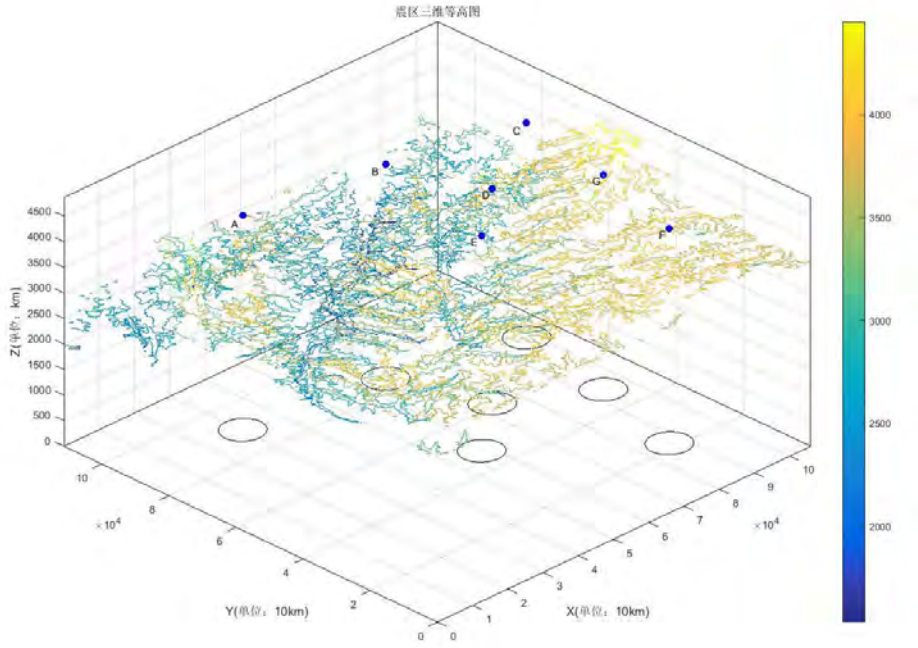


图 5 灾区三维等高图

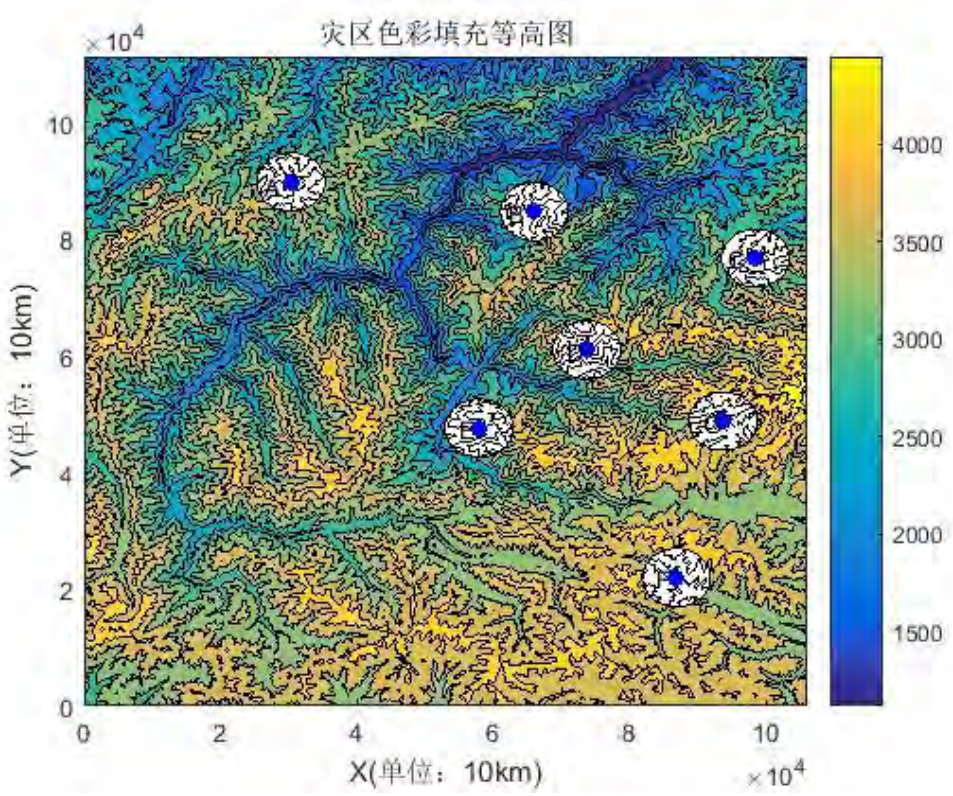


图 6 灾区色彩等高填充图

（5）由于无人机飞行的高度恒为 4200 米，与其他障碍物（含距离）的安全飞行距离不小于 50 米，因此 4150 米以上的海拔地区我们要绕过，数据处理如下：

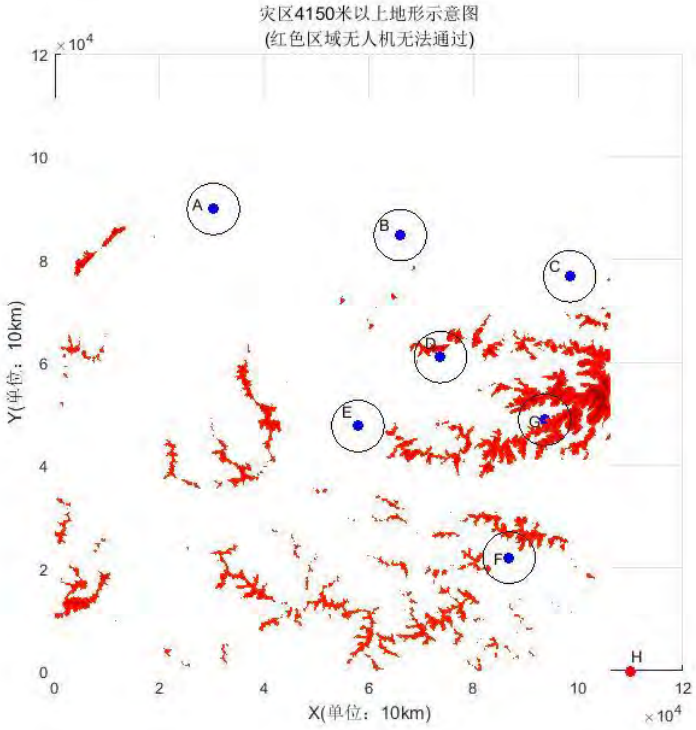


图 7 灾区 4150 米以上地形图（红色部分）

对于红色部分，我们假设无人机小幅度偏移巡查。由于 G 点绕过红色区域不方便，舍弃不进行考虑。

（6）问题一中第一小问是在海拔 3000 米以下进行的（区域 S 内），因此我们把 3000 米以上的数据进行着色处理，5 个重点灾区圆周内白色部分为要进行巡查的地方。

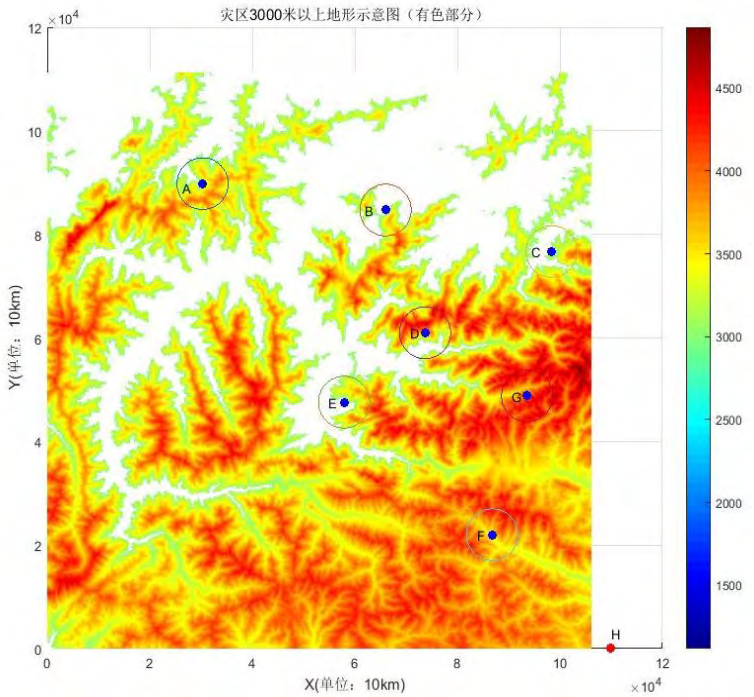


图 8 3000 米以上灾区地形图（有色部分）

（7）问题一中第二小问是在海拔 4000 米以下进行的（不限于区域 S），因此我们把 4000 米以上的数据进行着色处理。

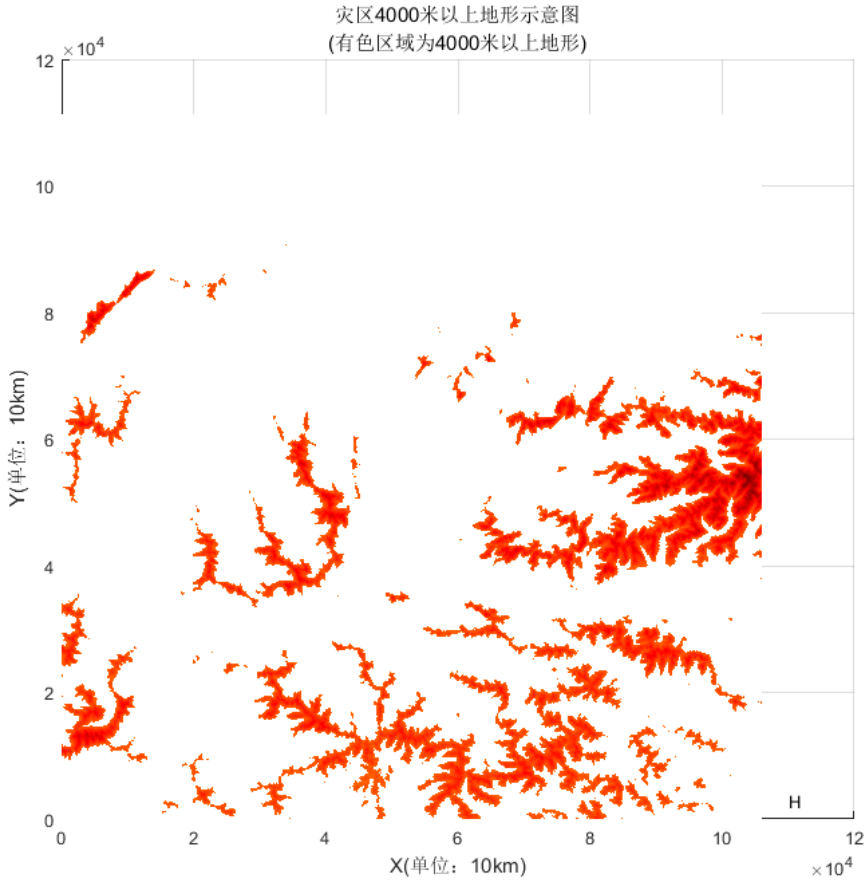


图 9 震区 4000 米以上地形示意图（有色部分）

4.1.3 模型准备

➢ **第一小问**

遗传算法[5]简介：

遗传算法（Genetic Algorithm）是 1975 年 Michigan 大学的 J.Holland 提出模拟达尔文生物进化论的自然选择和遗传学机理的生物进化过程的计算模型，是一种通过模拟自然进化过程搜索最优解的方法。遗传算法是从代表问题可能潜在的解集的一个种群开始的，而一个种群则由经过基因编码的一定数目的个体组成。每个个体实际上是染色体带有特征的实体。染色体作为遗传物质的主要载体，即多个基因的集合，其内部表现（即基因型）是某种基因组合，它决定了个体的形状的外部表现。因此，在一开始需要实现从表现型到基因型的映射即编码工作。由于仿照基因编码的工作很复杂，我们往往进行简化，如二进制编码，初代种群产生之后，按照适者生存和优胜劣汰的原理，逐代演化产生出越来越好的近似解，在每一代，根据问题域中个体的适应度大小选择个体，并借助于自然遗传学的遗传算子进行组合交叉和变异，产生出代表新的解集的种群。这个过程将导致种群像自然进化一样的后生代种群比前代更加适应于环境，末代种群中的最优个体经过解码，可以作为问题近似最优解[2]。总结来说，遗传算法模拟基因重组和进化

的自然过程，将问题的参数编码成为个体，个体构成种群，种群中的个体经过选择、突变、倒位的运算，反复经过迭代后得到一个较为优化的结果，本质实为一种并行的全局优化算法。

遗传算法的实现步骤[3]：

1、染色体编码

遗传算法的编码方式主要有浮点编码和二进制编码两种，通常在无人机路径规划中，采用二进制编码规则对多维空间的每个自变量编码，二进制编码不仅与计算机处理原理相符，也能同时实现染色体的遗传、编译和突变等操作。

设某一参数的取值范围为(A, B)，使用长度为 m 的二进制编码表示该参数，则它共有 2^m 种不同的编码，参数编码的对应关系为：

00000 = 0 → A
00001 = 1 → A+ μ
00011 = 2 → A+2 μ
……
11111 = 2^m-1 → B
由此可得：

$$\mu = \frac{B - A}{2^m - 1} \qquad \text{公式 1}$$

2、初始化群体的产生

遗传算法是对群体进行的进化操作，需要随机产生 N 个染色体组成起始搜索点作为初始群体数据。

3、适应度计算

遗传算法依照与个体适应度成正比的几率决定当前种群中各个体遗传到下一代种群中的机会。个体适应度大的个体更容易被遗传到下一代。

4、选择运算

选择运算是根据个体适应度大小决定其下代遗传的可能性。

若设种群中个体总数为 N，个体 i 的适应度为 f_i，则个体 i 被选中的几率为：

$$\mathrm{Pi} = \frac{f_i}{\sum_{k=1}^{N} f_k} \qquad \text{公式 2}$$

当个体选择的几率决定后，再产生[0,1]区间的均匀随机数来决定哪个个体参加复制。若个体适应度高，则被选中的几率Pi就大，有可能被多次选中，它的遗传基因就会在种群中扩散；若个体的选择几率小，则会被逐渐淘汰；

5、交叉运算

交叉运算是遗传算法中产生新个体的主要操作过程，使用单点或多点进行交叉的算子。交叉运算中首先使用随机数生成一个或多个交叉点位置，然后两个个体在交叉点位置互换部分基因码，形成两个子个体。

如两条染色体 R1 = 01000111，R2 = 10010100，交换其后 4 位基因，如图 10 所示，R1’ = 01000100，R2’ = 10010111 可被看做是原染色体 R1 和 R2 的子代染色体。

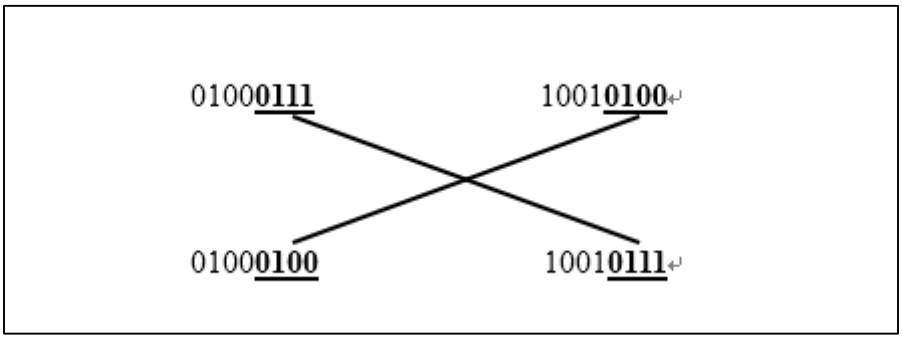


图 10 染色体基因交叉原理

6、变异运算

“变异运算”是使用基本位进行基因突变，是对个体的某一个或某一些基因码实行小概率翻转，将 0 变为 1，将 1 变为 0，以产生新个体的一种方法。例如染色体 R1 = 11101010，对第 2 位进行变异运算，可得染色体 R1’= 10101010，R1’即为原染色体 R 的子代染色体。

7、停止准则

当满足停止准则时，算法结束。

遗传算法优点：

- 搜索从群体出发，具有并行性，可以同时对多个个体进行比较。
- 与传统的枚举、启发等优化算法相比较，以生物进化为原型，收敛性强，计算耗时少。

➢ **第二小问**

环境建模简介

环境建模主要是将环境进行栅格[3]量化，这里将地图环境按照无人机探测范围进行栅格化处理，建立栅格地图，以此为基础进行路径规划算法研究。

1、建立量化栅格的环境地图

将工作环境栅格化后，整个区域分成 n*n 个栅格，每个栅格生成相应坐标 cell(x,y)其中，1≤x≤n，1≤y ≤n。然后针对每个栅格设定属性值：

$$Cell(x,y).block=\begin{cases}0 & \text{自由栅格}\\ 1 & \text{障碍物栅格}\end{cases} \qquad \text{公式 3}$$

当栅格属性 cell(x,y).block=0 时，说明该栅格为自由栅格，此时该栅格可以被覆盖，可以被覆盖的栅格还有一个覆盖属性 cell(x,y).visited，其初始值设置为 0，栅格每被覆盖一次，其覆盖属性值依次累加 1，即

$$Cell(x,y).visited=cell(x,y).visited+1 \qquad \text{公式 4}$$

图 11 画出了量化后的栅格地图。其中表示机器人，黑色表示障碍物，空白栅格表示自由栅格，机器人的任务就是要遍历所有的自由栅格。

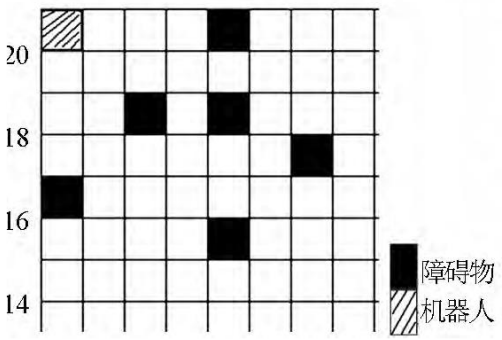


图 11 栅格地图

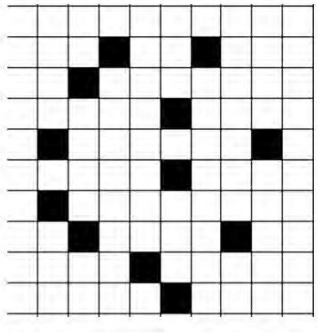


图 12 凸型障碍物

2、 环境的障碍物模型

障碍物种类可以分为凹型障碍物和凸型障碍物两种，凹型障碍物一般是半封闭型，比如 U 型、V 型等，凹型障碍物以外的障碍物归类为凸型障碍物。凸型障碍物，如上图 12 所示。凹型障碍物，比如U 型障碍物，如图 13 所示。在环境地图中，既有凹型障碍物又有凸型障碍物，如图 14 所示。

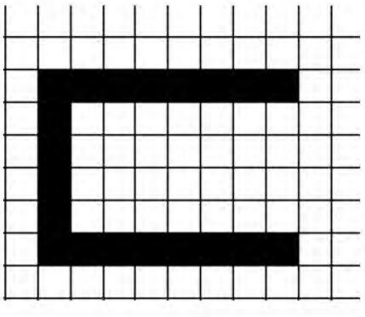


图 13 凹型障碍物

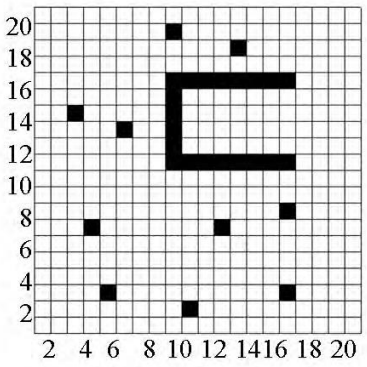


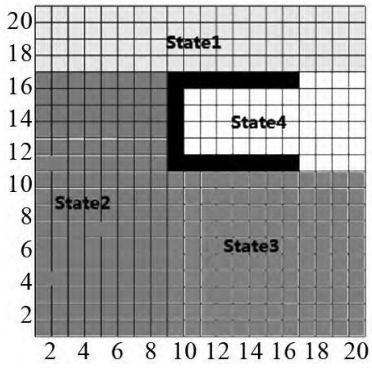
图 14 混合类型障碍物

基于栅格区域分解法的算法设计

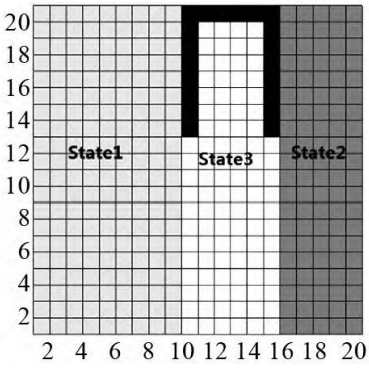
基于栅格的区域分解法包含区域分解、子区域连接和子区域遍历三部分。在含有混合障碍物的环境里，首先以凹型障碍物边缘和环境地图边界为标准进行区域划分，然后通过局部路径规划算法求相邻区域的最短路径，最后将子区域按照逆时针方向连接起来，形成遍历连通图。子区域遍历根据障碍物的不同类型采用不同的遍历方法。

1、 栅格的区域分解

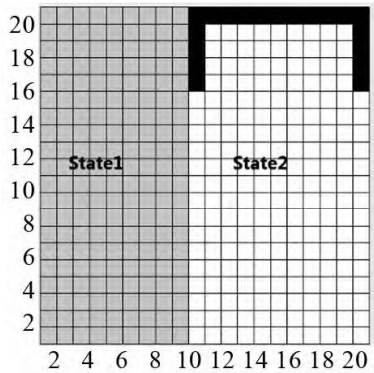
在含有混合障碍物环境中，这里以含有 U 型障碍物为例说明，在可行区域按照U 型障碍物边缘和环境边界进行分解，根据 U 型障碍物在环境地图的不同位置，如图 15 所示，有以下几种分解情况：



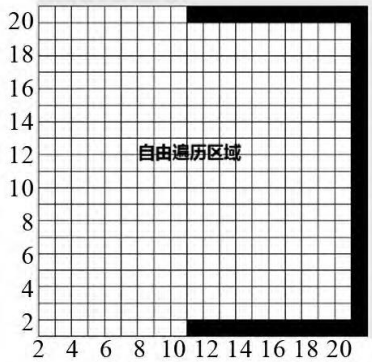
图(a) 一般 U 型障碍物单元分解图示



图(b) 障碍物一边靠在栅格地图边界



图(c) 障碍物两边靠在栅格地图边界



图(d) 障碍物三边均在栅格地图边界

图 15 栅格地图的区域分解

- 当 U 型障碍物三边与栅格地图边界均没有接触时为一般情况，可以分成四部分，分别为 State1, State2, State3, State4，如图 15（a）。
- 当 U 型障碍物一边与栅格地图边界接触时，可以分成三部分，分别为 State1, State2, State3, 如图 15（b）。
- 当 U 型障碍物两边与栅格地图边界接触时，可以分成两部分，分别为 State1, State2, 如图 15（c）。
- 当 U 型障碍物三边都与栅格地图边界接触时，除 U 型三边外，将整个自由栅格区域看做 U 型内部区域，如图 15（d）。

本文以图 15（a）为例说明算法设计过程。

2、 子区域之间连接

子区域之间的连接包括子区域起点、终点位置确定，相邻子区域连接方法设计和子区域连通图设计三部分。

- 子区域起点、终点位置确定

以图 15(a)为例，一共有四个子区域，分别为 State1、State2、State3、State4，相邻两个子区域连接的起点在子区域的边界。State1 作为遍历开始的子区域，设置机器人的起点为 cell(1,n)，遍历结束点

为 cell(x1,y1)．State2 起点有两个，cell(1, j)或者 cell(i, j)，通过计算结束点 cell(x1, y1)到 cell(1, j)或 cell(i,j)的距离，取最短距离的点为 State2 的起点，State2 遍历结束点为 cell(x2,y2)．State3 起点也有两个，cell(i,1)或者 cell(i,j2)，

通过计算结束点 cell(x2,y2)到 cell(i,1)或 cell(i,j2)的距离，取最短距离的点为 State3 起点，State3 遍历结束点为 cell(x3,y3)．State4 起点是确定的，为 U 型障碍物的右下角，设U型障碍物上下边缘长度为m，则起点为 cell(i+m,j2)，遍历结束点为 cell(i+m,j)。

State1 结束点到 State2 起点的最短距离：

$$d1 = \sqrt{(x1 - i)^2 - (y1 - j)^2} \qquad \text{公式 5}$$

$$d2 = \sqrt{(x1 - 1)^2 - (y1 - j)^2} \qquad \text{公式 6}$$

通过 mini(d1,d2)确定 State2 的起点。

State2 结束点到 State3 起点的最短距离：

$$d3 = \sqrt{(x2 - i)^2 - (y2 - 1)^2} \qquad \text{公式 7}$$

$$d4 = \sqrt{(x2 - i)^2 - (y2 - j)^2} \qquad \text{公式 8}$$

- 相邻子区域连接方法设计

根据确定的相邻两个子区域之间起点与结束点，按照两点法搜索策略进行局部路径规划，求出相对最短路径．两点法搜索策略简述为：移动机器人向目标栅格直线移动时，若遇到障碍物，就改变方向、移动到没有障碍物的位置，并再次确定目标栅格的方向、感知障碍物信息．如此反复，直至到达目标栅格。两点法搜索策略局部路径规划仿真如图 16 所示，表示起点， 表示终点，黑色表示障碍物， 表示规划的路径。

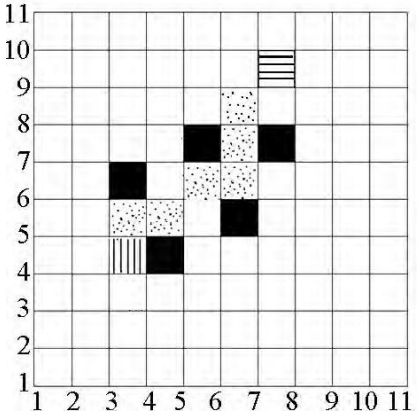


图 16 两点法局部路径规划

- 子区域连通图设计

四个子区域之间两两相邻，因此直接按照逆时针（顺时针）方向形成一张完全连通图，即移动机器人按照 State1→State2→State3→State4 顺序依次对每个子区域完成遍历。

3、 子区域遍历

子区域遍历根据障碍物的不同类型，在凸型障碍物区域采用内螺旋遍历方式，在凹型障碍物区域采用梳状遍历方式。

- 内螺旋遍历算法

在凸型障碍物的环境地图中，判断程序结束的覆盖率用 coverage 表示，最大覆盖率用 max 表示．根据设置的属性值判断机器人的动作目标：

G1　准目标栅格（即为前进方向的下一个栅格）

G2　障碍物栅格

G3　自由栅格（目标栅格相关的栅格）

具体算法如下：

Step1 初始化机器人位置坐标 cell(1,n)，最大覆盖率 max=100%，机器人运行方向为顺时针方向。

Step2 判断覆盖率 coverage<max 是否成立，否转移到 Step6，是则继续执行。

Step3 判断准目标栅格 G1 的属性值 cell(i,j).block=1 时，转移到 Step5，否则继续执行。

Step4 该栅格为自由栅格 G3，cell(x,y).visited=cell(x,y).visited+1，转移到 Step2。

Step5 判断障碍物栅格 G2 内侧的准目标栅格 G1 属性值 cell(i,j).block=0 时，转移到 Step4，否则继续执行 Step5。

Step6 退出循环，算法结束。

程序流程图如图 17 所示：

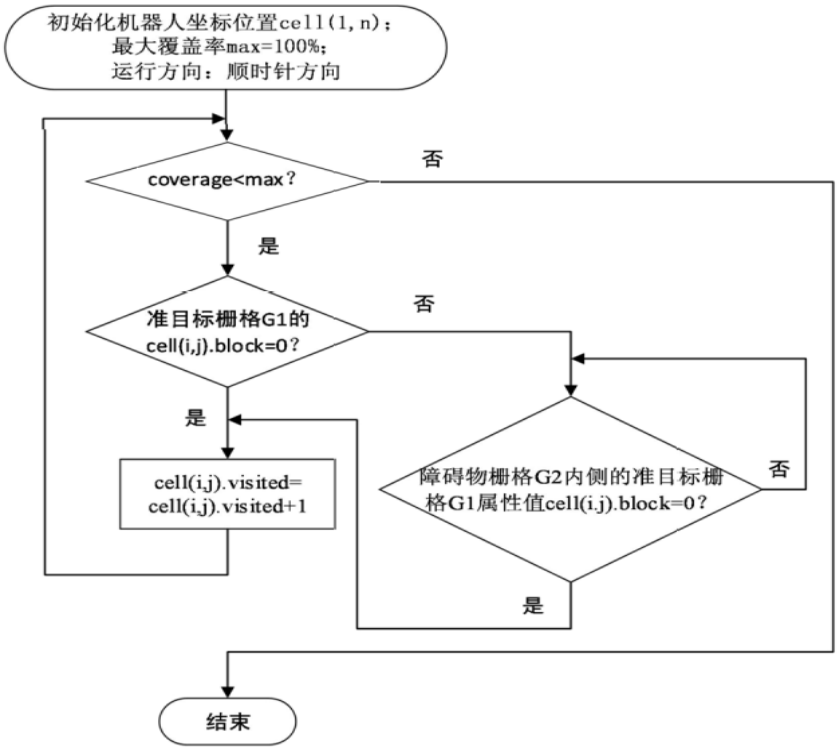


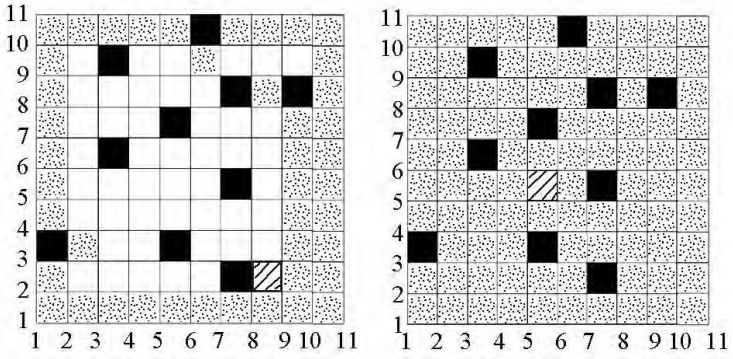
图 17　内螺旋程序流程图

算法仿真过程如图18所示。图18(a)为遍历中间过程，图18(b)是遍历结束，黑色表示障碍物， 表示机器人遍历过程中的位置和遍历结束的位置， 表示遍历路径。

- 梳状遍历算法

在U型障碍物栅格地图中，(i,j2)为U型障碍物左下角栅格坐标，(i,j)为U 型障碍物左上角栅格坐标。U 型区域遍历的初始位置横坐标表示为x=i+m，纵坐标表示为 y=j2，根据机器人起始点位置可知 U 型区域下边缘为起始行，定义其为奇数行，运行方向为横坐标增大的方向，相反的，在偶数行，运行方向为横坐标减小的方向。用r表示奇数行或者偶数行的计算，用r0表示奇偶数计算结果。r的计算方法如式(7)。

$$r = \frac{j-j2}{2} \qquad \text{公式 9}$$



(a) 遍历中间过程　　(b) 遍历结束

图 18 内螺旋算法仿真

具体算法如下：

Step1 初始化U型区域的起始坐标为cell(x,y)，r0=0，初始运行方向为横坐标增大的方向。

Step2　判断y=j，是则转移到Step7，否则继续执行。

Step3　判断r=r0，是则转移到Step9，否则继续执行。

Step4　机器人沿着当前方向横坐标 x=x+1，纵坐标不变，cell(x,y).visited=cell(x,y).visited+1。

Step5 判断x=n，否则转移到Step4，是则继续执行。

Step6 纵坐标y=y+1，并且转移到Step2。

Step7 判断 x=i+m，是则转移到 Step12，否则机器人沿着当前方向横坐标x=x-1，纵坐标不变，cell(x,y).visited=cell(x,y).visited+1，并继续执行Step7。

Step8 判断x=i+1并且y=j-1，是则转移到Step4，否则继续执行。

Step9 机器人沿着当前方向横坐标 x=x-1，纵坐标不变，cell(x,y).visited=cell(x,y).visited+1。

Step10 判断x=i+1，否则转移到Step6，是则继续执行。

Step11 判断x=i+1，是则转移到Step4，否则转移到Step6。

Step12 退出循环，算法结束。

程序流程图如 19 所示：

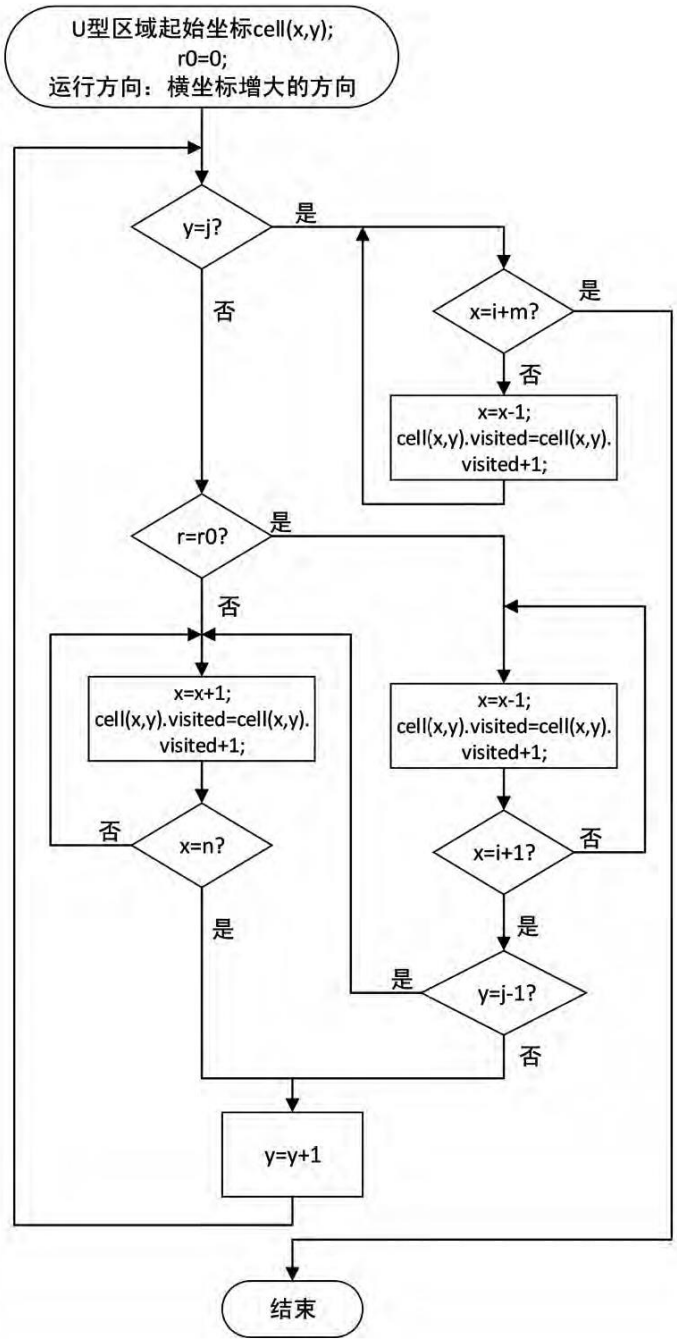
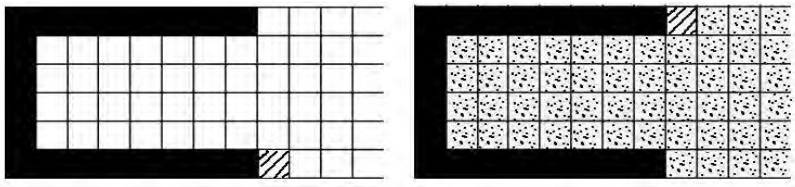


图 19 梳状遍历的流程图

算法仿真过程如图 20 所示。其中 20(a)为遍历开始，20(b)是遍历结束，黑色为障碍物， 分别为机器人在U型遍历的起点和终点， 是遍历路径。



（a）遍历开始 （b）遍历结束

图 20 梳状遍历仿真

算法评价

根据移动机器人全覆盖遍历路径规划的要求，一是提高覆盖率，二是降低重复率，通过计算得到覆盖率和重复率的值，判断该算法的可行性和有效性。

覆盖率计算：

$$coveragerate = \frac{\text{num}}{\text{n}*\text{n}-\text{ObsNumber}} * 100\% \qquad \text{公式 10}$$

$$crepeat = \frac{\text{numb}}{\text{n}*\text{n}-\text{ObsNumber}} * 100\% \qquad \text{公式 11}$$

式中：num 为覆盖栅格个数；numb 为重复覆盖栅格个数；n 为环境区域边长；ObsNumber 为障碍物个数。

4.1.4 模型建立与求解

在多无人机协同任务规划中，采用基本遗传算法固定长度二进制编码方式，遗传操作采用交叉和变异算子，基于一定的环境要素和任务需求，在完成任务最大化的同时，保证无人机在灾区巡查路径最短，所以，从本质上说，问题一可以看作是一个遗传算法求解多旅行商以获得无人机飞行的最佳路径问题。

问题一的数学描述：

1. 问题一第一小问

根据前期数据准备可以看出，第一小问无人机需巡查 5 个重点区域中心方圆 10 公里以内的海拔为 3000 米以下的区域。现构建问题一的数学模型，假设有 n 个需要遍历的中心目标点，一个起飞基地 H，变量定义如下：

n 为给定的需要遍历的中心目标点（$0 \le n \le 11$）；

m 为起飞基地数（$1 \le m \le 4$）；

N = n+m-1 为总遍历数，i，j 为遍历序号，i,j∈{1,2,…N}；

C_{ij}为遍历点 i 到遍历点 j 的距离，其中虚拟遍历点位之间以及虚拟遍历点位和起始遍历点之间的距离设为无穷大；

D_k为单架 FY-1 型飞机从无人机基地出发所经过的路径长度，k∈{1,2,…m}；

L_k为每架无人机遍历的点位数，k∈{1,2,…m}；

L_{max}为每架无人机允许遍历的点数上限；

S = {S_1, S_2, …, S_N}为一条遍历所有中心目标点的环路；

则转化后的数学模型为：

总的路径最短：

$$T(S) = \min(\sum_{i=1}^{N-1} C_{S_i,S_{i+1}} + C_{S_N,S_1}) \qquad \text{公式 12}$$

目标函数式（公式 12）表示无人机经过的总的路径之和最短，所有中心目标点经过一次，且只经过一次；

最长路径最短：

$$F(S) = \min(\max(D_k)), k \in \{1,2,\dots m\} \qquad \text{公式 13}$$

目标函数式（公式 13）表示最长的路径最短

Step 1：根据遗传算法原理计算每组无人机遍历点数与其最优路径，基于遗传算法的多无人机协同规划最佳路径的算法流程如图 21 所示：

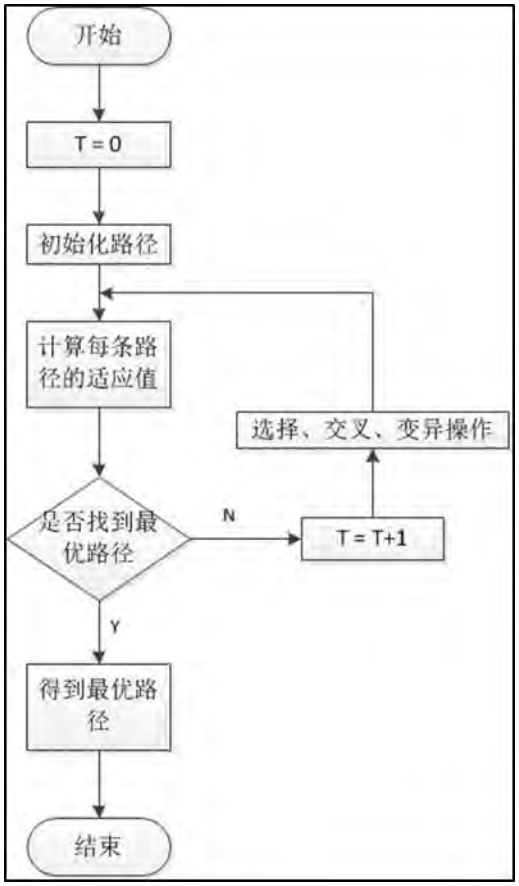


图 21　基于遗传算法的多无人机协同规划流程

算法的部分参数设置如下：

popsize（种群数量）= 80

numIter（算法迭代次数）= 5000

minTour（最少遍历点数）= {1,2,3…,5}

showProg（遗传算法进程，如果正确则置为 1）= 1

showResult（遗传算法结果，如果正确则置为 1）= 1

通过运行程序，设置不同的最小遍历点数，则可得出不同情况下的问题一的最优路径。

Step 2：分析需要派出的无人机与无人机至少遍历点数的关系

表 2 圈数与每组无人机最少遍历点数的关系

封闭圈数	无人机至少遍历点数
1	5
1	4
1	3
2	2
3	2
5	1

在考虑第一小问时，可有多种飞行方案，因为题设中未给出使用无人机的数量限制，故可以使用 1 架无人机遍历所有中心目标点，也可以使用 2-4 架无人机；

- 希望在四小时之内使 S 区域海拔 3000 米以下的地方尽可能多地被巡查到，因此本文不考虑返回时间。

- 方案一：根据算法结果无人机遍历 5 个点、4 个点、3 个点时都是一个封闭圈即使用一架无人机进行巡查，且路径是一致的：

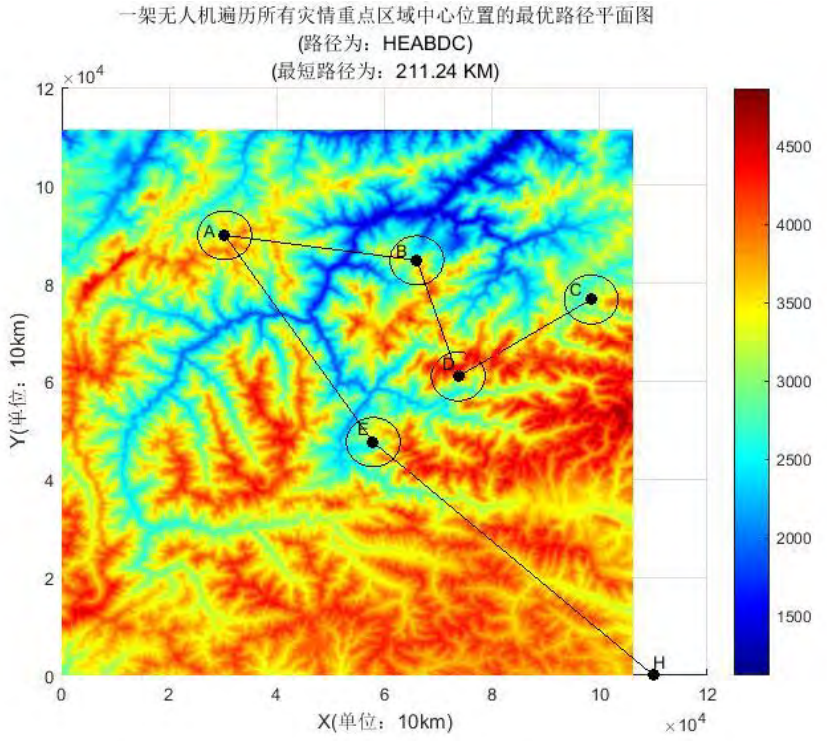


图 22 一架无人机巡查最优路径平面图

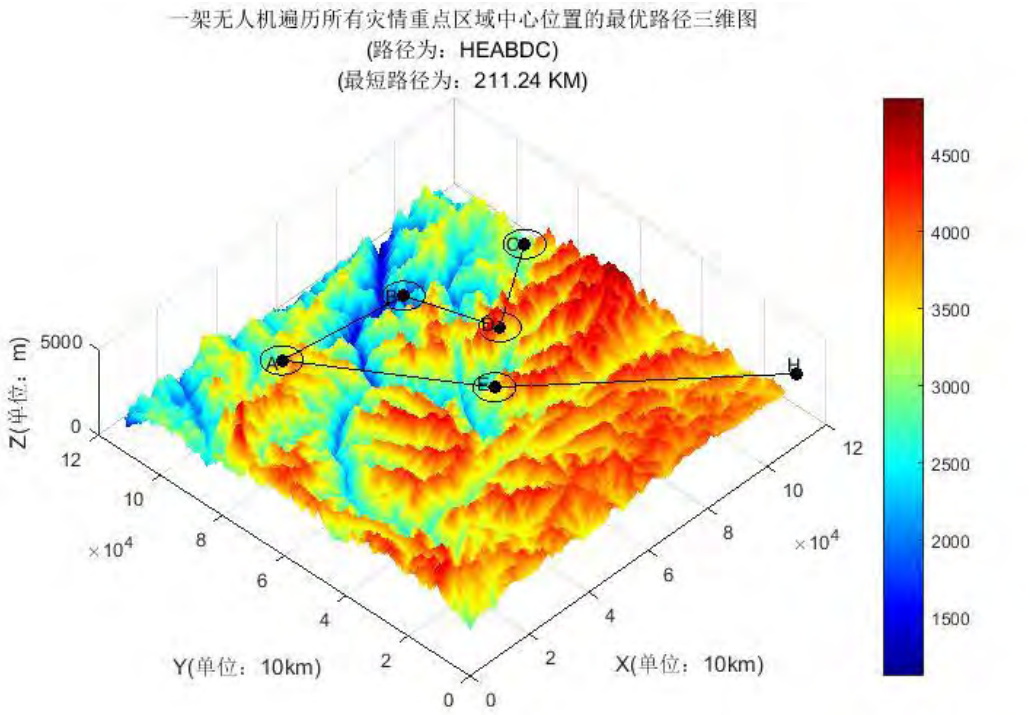


图 23 一架无人机巡查最优路径三维图

- 方案二：根据算法结果无人机遍历 2 个点，形成两个封闭圈时，路径如下：

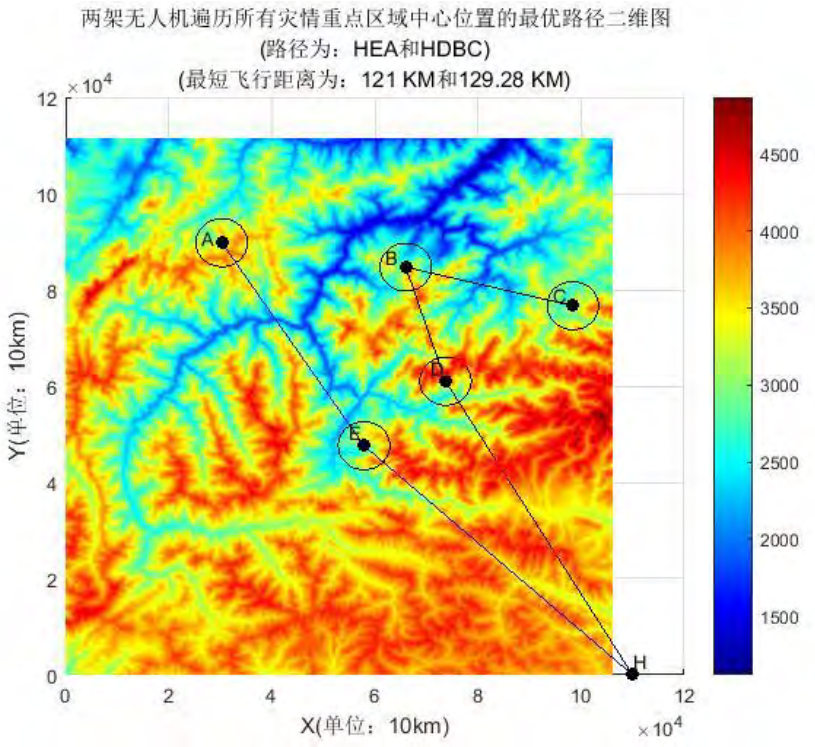


图 24 两架无人机巡查最优路径平面图

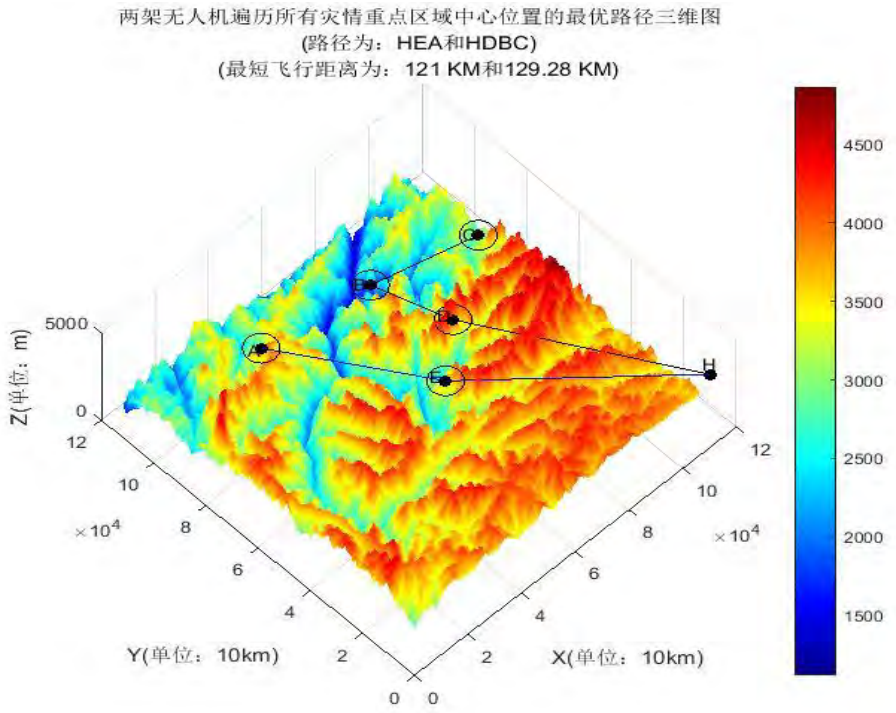


图 25 两架无人机巡查最优路径三维图

- 方案三：根据算法结果无人机遍历 2 个点，形成三个封闭圈时，路径如下：

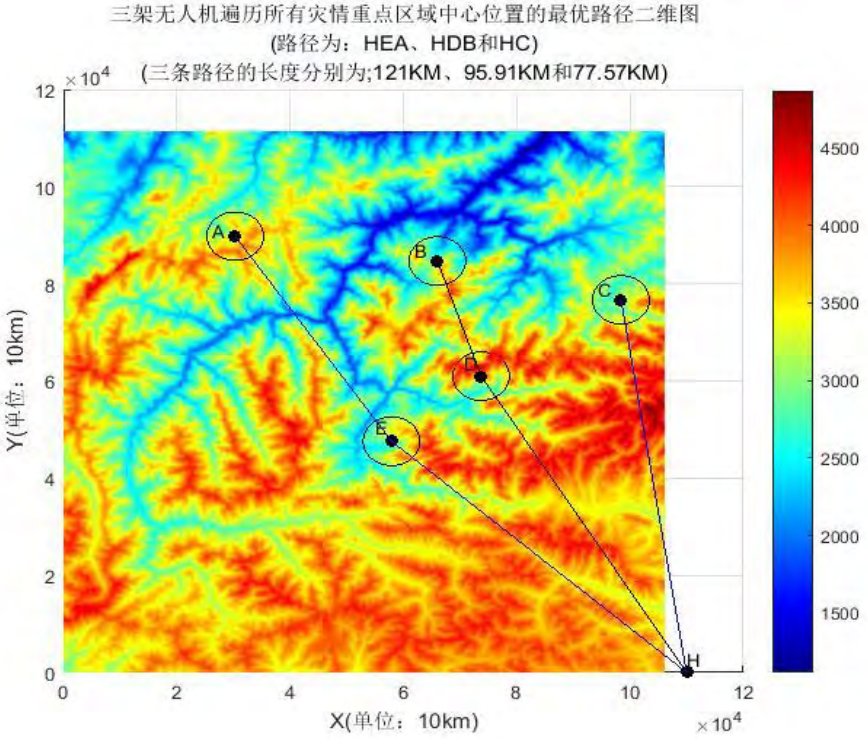


图 26 三架无人机巡查最优路径平面图

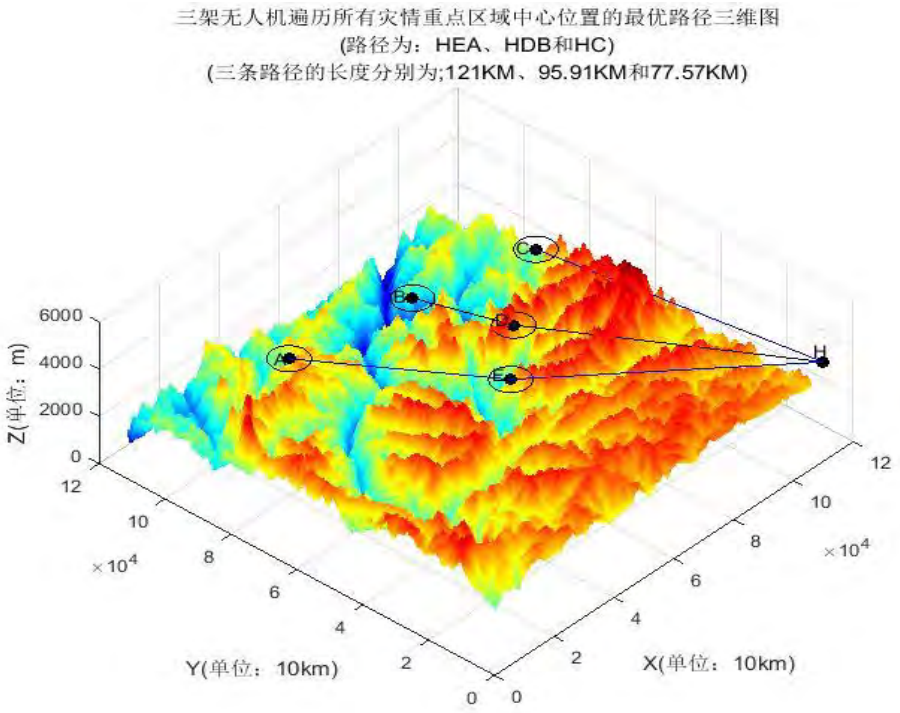


图 27 三架无人机巡查最优路径三维图

- 方案四：根据算法结果无人机遍历 1 个点，形成五个封闭圈时，路径如下：

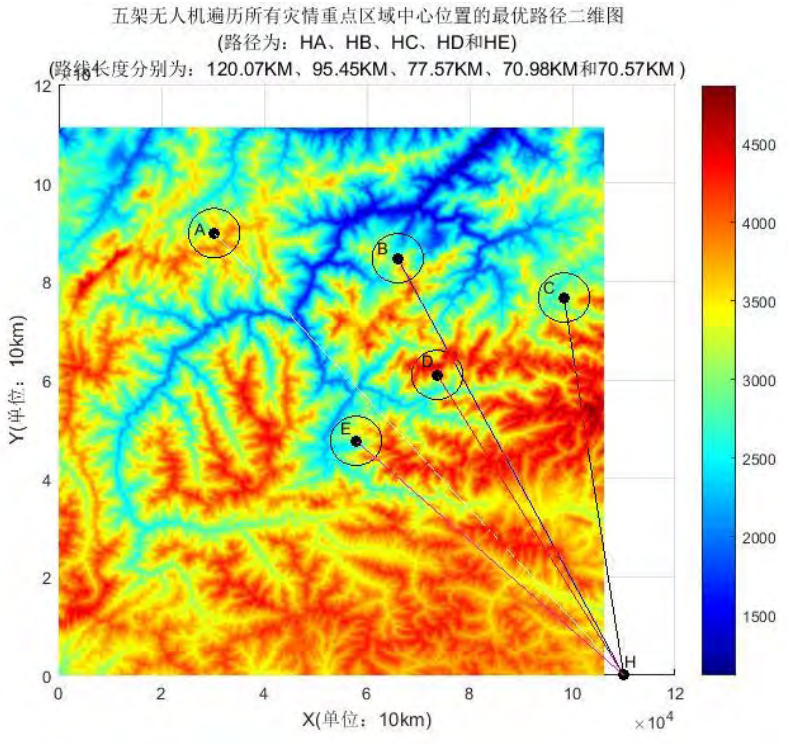


图 28 五架无人机巡查最优路径平面图

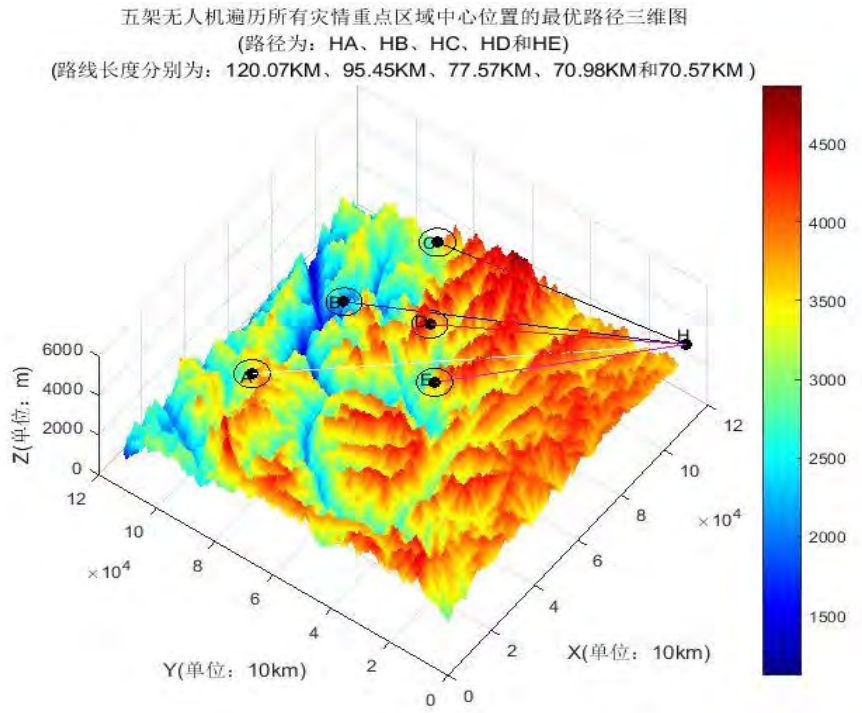


图 29 五架无人机巡查最优路径三维图

➢ 根据计算可得，四种方案的路径数据如下表所示：

表 3 各方案及其路径长短

方案	路线	各路径长度（km）	最长路径(km)	总路径（km）
一	HEABDC	211.24	211.24	211.24
二	HEA、HDBC	121、129.28	129.28	250.28
三	HEA、HDB、HC	121、95.91、77.57	121	294.48
四	HA、HB、HC、HD、HE	120.07 、 95.45 、 77.57 、 70.98 、 70.57	120.07	434.64

综合考虑总路径最短及最长路径最短，选定方案三为待优化路径（避开 4150 米海拔区域）

2. 问题一第二小问

由分析可得，为保证在 72 小时内，低于 4000 米海拔的区域（不限于 S）相邻两次被巡查到的时间间隔不大于 3 小时，并且无人机均需从 H 出发并在 8 小时内回到 H，路径长度与无人机数量有存在下表关系：

表 4 路径长度与所需无人机数量关系

路径长度 L_1(km)	所需时间 H(h)	无人机数 M
L<=180	H<=3	M=1
180<L<=360	3<H<=6	M=2
360<L<=480	6<H<=9	M=3

- 根据环境建模中栅格的区域分解，本文把 4000 米以下海拔区域遍历一遍，需 9 条路径，在不考虑 72 小时内的循环（即仅考虑相邻两次巡查不超过 3 小时）的情况下：**需 23 架无人机（见下表）**。

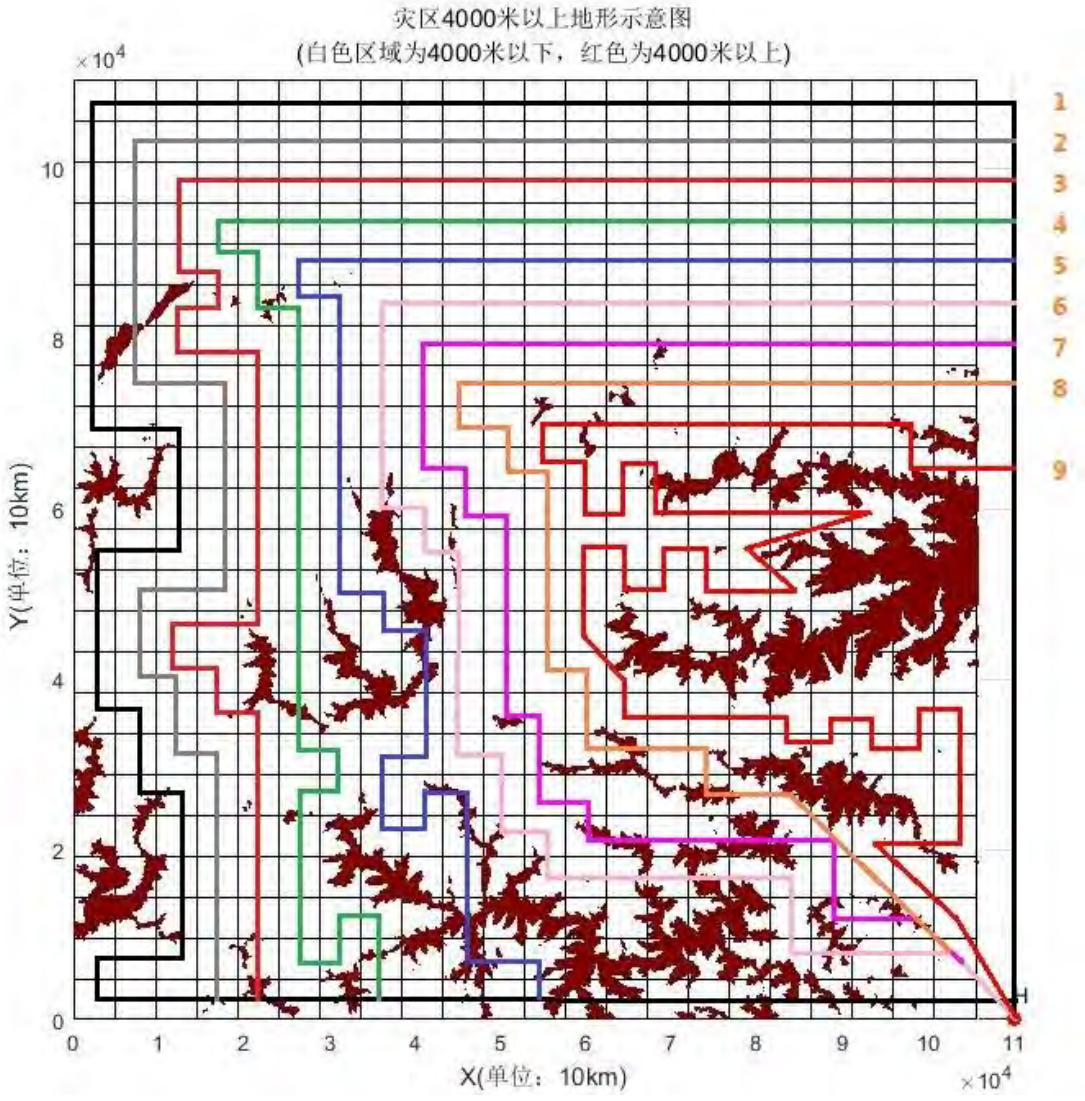


图 30 巡逻 4000 米以下区域无人机路径图

表 5 路径与所需无人机及其每架无人机飞完路径所需时间

路径编号	路径长度 L_2(km)	每架无人机所需飞行时间 H_2（h）	无人机架数
①	460.8	7.68	3
②	422.4	7.04	3
③	408.9	6.815	3
④	384	6.4	3
⑤	369.6	6.16	3
⑥	316.8	5.28	2
⑦	297.6	4.96	2
⑧	259.2	4.32	2
⑨	355.2	5.92	2

- 考虑到在 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(无人机均需从 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**)，并且保证无人机可以循环使用，因此原所需飞行时间均需加 1。

表 6 考虑 72 小时内存在循环后的数据改进

路径编号	原所需飞行时间 H（h）	原无人机架数	现所需飞行时间 H（h）	现无人机架数
①	7.68	3	8.68	3
②	7.04	3	8.04	3
③	6.815	3	7.815	3
④	6.4	3	7.4	3
⑤	6.16	3	7.16	3
⑥	5.28	2	6.28	3
⑦	4.96	2	5.96	2
⑧	4.32	2	5.32	2
⑨	5.92	2	6.92	3

从表中我们可以看出路径 6 和路径 9 的无人机架数变为 3 架，原因见上表 6

4.1.5 求解结果

1. 问题一第一小问

- 理想情况下，方案三路径如下：

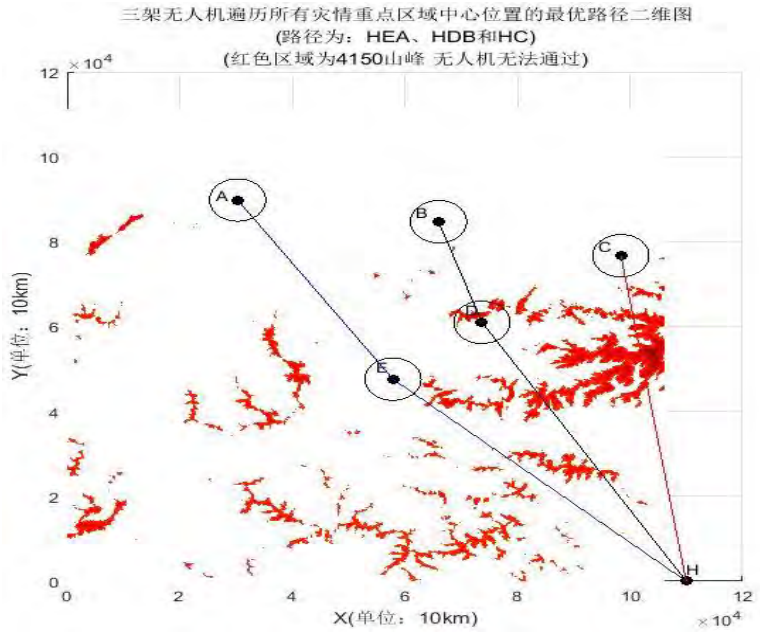


图 31 理想情况下方案三最优路径

- 考虑 4150 米海拔无法通过，经过修正后的路径如下图所示：

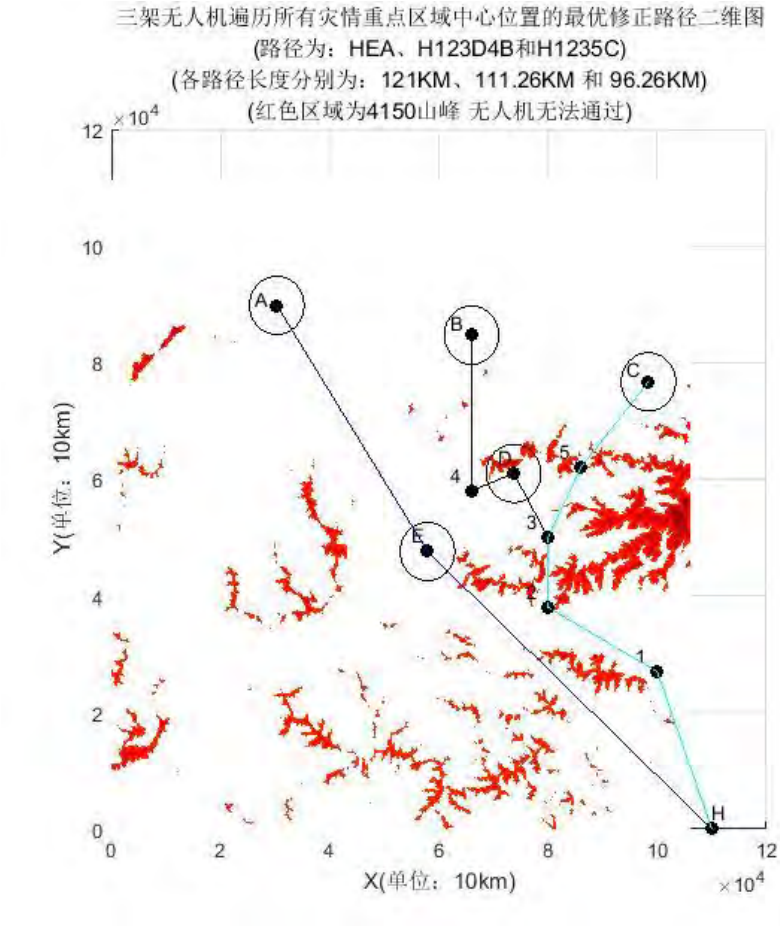


图 32 修正后路径图

- 无人机围绕以重点区域中心为圆心半径为 2500 米的圆周进行巡查（在假设无人机巡查半径为 2500 米时），考虑到 4150 米以上海拔无法通过的最终最优路径图如下所示：

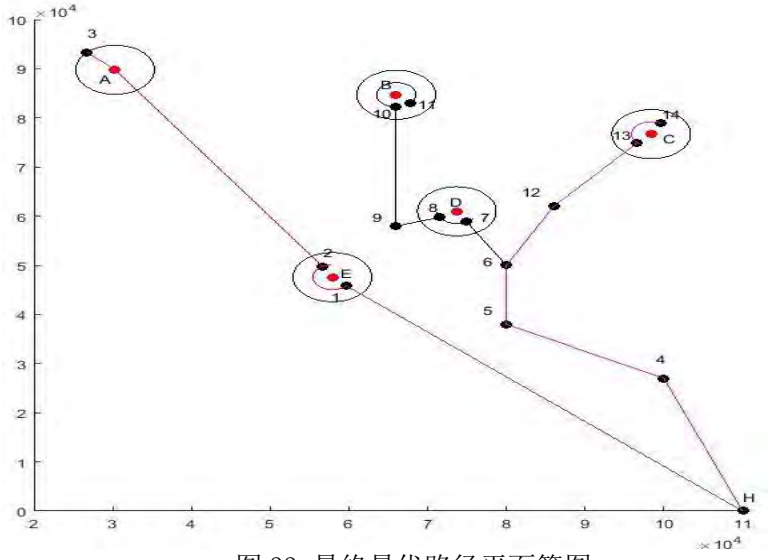


图 33 最终最优路径平面简图

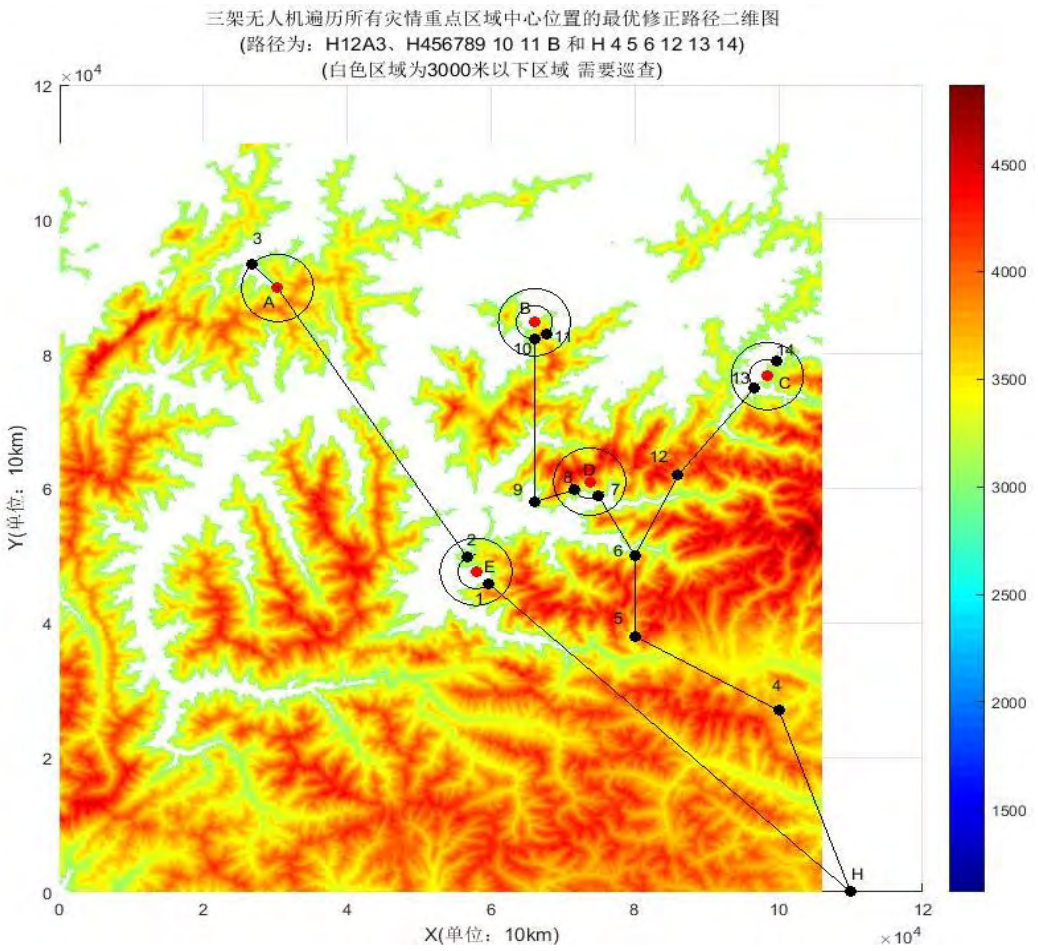


图 34 全貌最终最优路径平面简图（白色部分为 3000 米以下海拔）

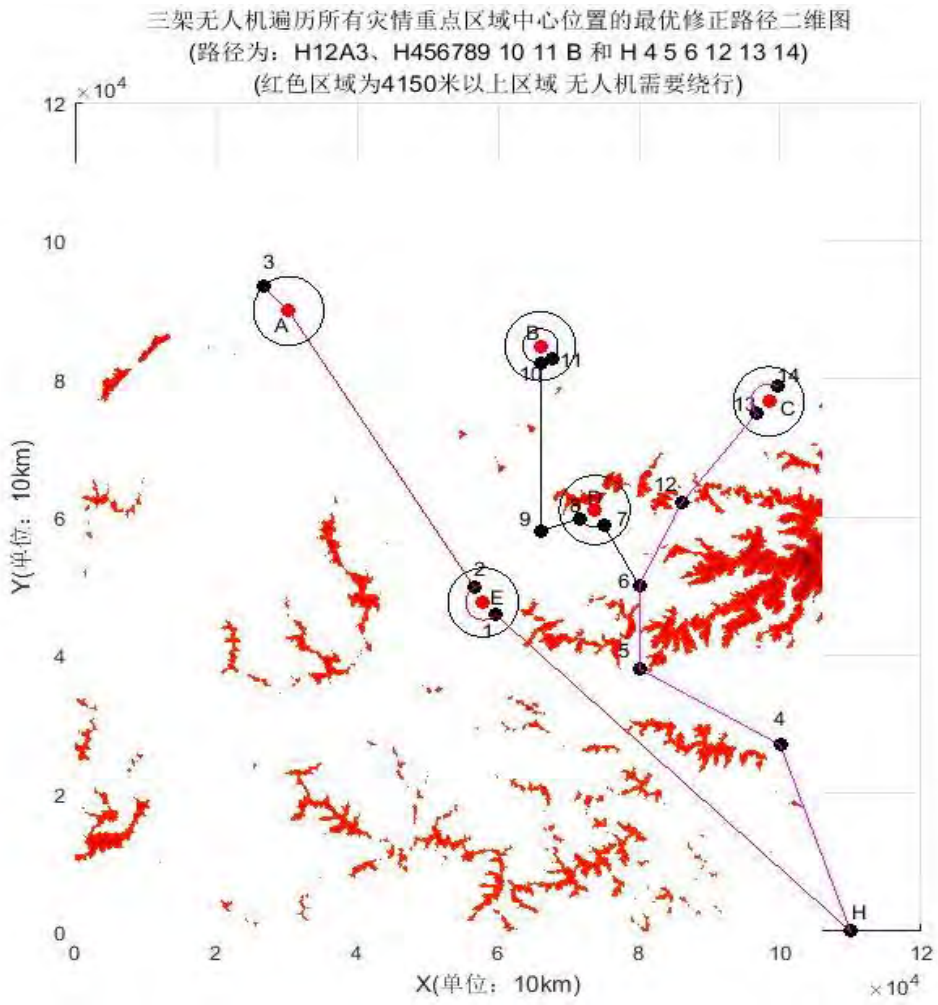


图 35 方案三最终最优路径平面图（红色部分海拔 4150 米以上）

由上图可知三架无人机的路径分别为：H12A3、H456789 10 11 B、H456 12 13 14，对应的路径长度分别为：128.83km、125.25km、101.61km，因此在四小时之内可以完成区域 S 内海拔 3000 米以下地方的巡查。

由于无人机的巡查半径实际为 2400 米，为了题目计算方便本文假设无人机半径为 2500 米，因此产生误差，实际覆盖率为 99.84%。如下图所示，以重点区域中心为圆心，2600 米为半径的圆周进行巡查，白色区域为可巡查到的区域。

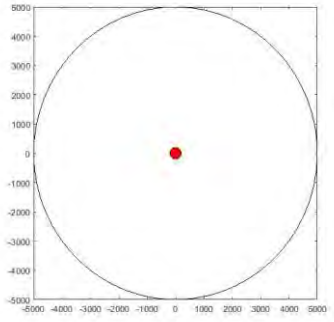


图 36 扫描覆盖面积图

2. 问题一第二小问

由以上分析，本文得出了最优路径，如下图所示：

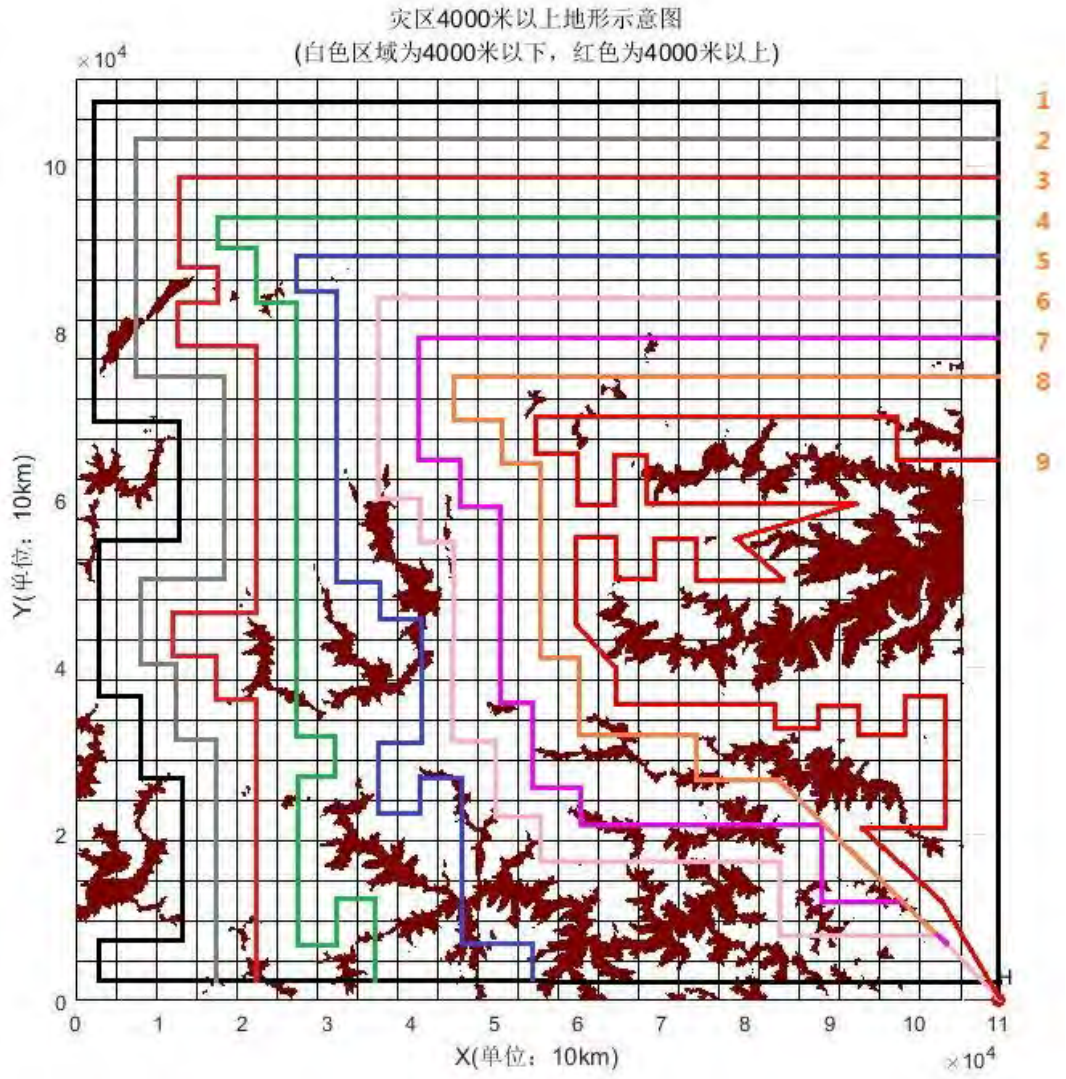


图 37 最优路径图

此过程最少使用 25 架无人机，才能保证在 72 小时内，低于 4000 米海拔的区域（不限于 S）相邻两次被巡查到的时间间隔不大于 3 小时。各路径无人机飞行时间和每个路径所需无人机架次如表 7 和表 8：

表 7 每个路径上无人机飞行所需时间

路径编号	每架无人机所需飞行时间 H（h）
①	7.68
②	7.04
③	6.815
④	6.4
⑤	6.16
⑥	5.28
⑦	4.96
⑧	4.32
⑨	5.92

表 8 巡逻 4000 米以下区域所需的无人机架次

路径编号	原所需飞行时间 H(h)	原无人机架数	现所需飞行时间 H(h)	现无人机架数
①	7.68	3	8.68	3
②	7.04	3	8.04	3
③	6.815	3	7.815	3
④	6.4	3	7.4	3
⑤	6.16	3	7.16	3
⑥	5.28	2	6.28	3
⑦	4.96	2	5.96	2
⑧	4.32	2	5.32	2
⑨	5.92	2	6.92	3

4.2 对问题二的分析

4.2.1 问题描述及分析

路径规划是无人机一个重要研究内容。常规的路径规划是指点到点的最优路径规划,其目标是寻找一条从起始点到目标点的无碰撞最优路径。全区域覆盖是指在某个区域中,按一定的评价标准, 无人机能够遍历整个工作区域,在区域遍历过程中自主可靠地完成指定的操作任务。因此全区域覆盖的路径规划其目标是产生一条有效路径来遍历工作环境的每个可达区域。

知道某一范围，标注出关键搜寻点，将所有关键搜寻点遍历一次的最快路径，可以类似的看作旅行商问题（TSP）。旅行商是一个相对而言十分成熟的问题，但在普通的旅行商问题的解决方案中，没有考虑各个“城市”间有没有障碍物能不能直接到达。然而无人机山区搜寻，某些区域障碍物繁多，我们不仅要找出最快的覆盖整个搜寻范围的路径，还要避开海拔 3000 米以上的山峰。

本题首先引入邻接矩阵，并且为了做到无人机躲避障碍物，将关键搜寻点抽象为邻接矩阵。介绍了遗传、蚁群、模拟退火 3 种经典优化算法解决旅行商问题的基本思路。分别对三种算法进行 MATLAB 计算仿真，对比搜寻路径的长短、算法稳定性、运行时间，得出无人机山区搜寻路径规划的最优算法，从而得到无人机山区搜寻路径规划方案。

4.2.2 数据处理

（1）已知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，可得知飞机与地面投影距离最高 500 米。则无人机探测距离最大直径为 1732 米。

将灾区域栅格化，如图：

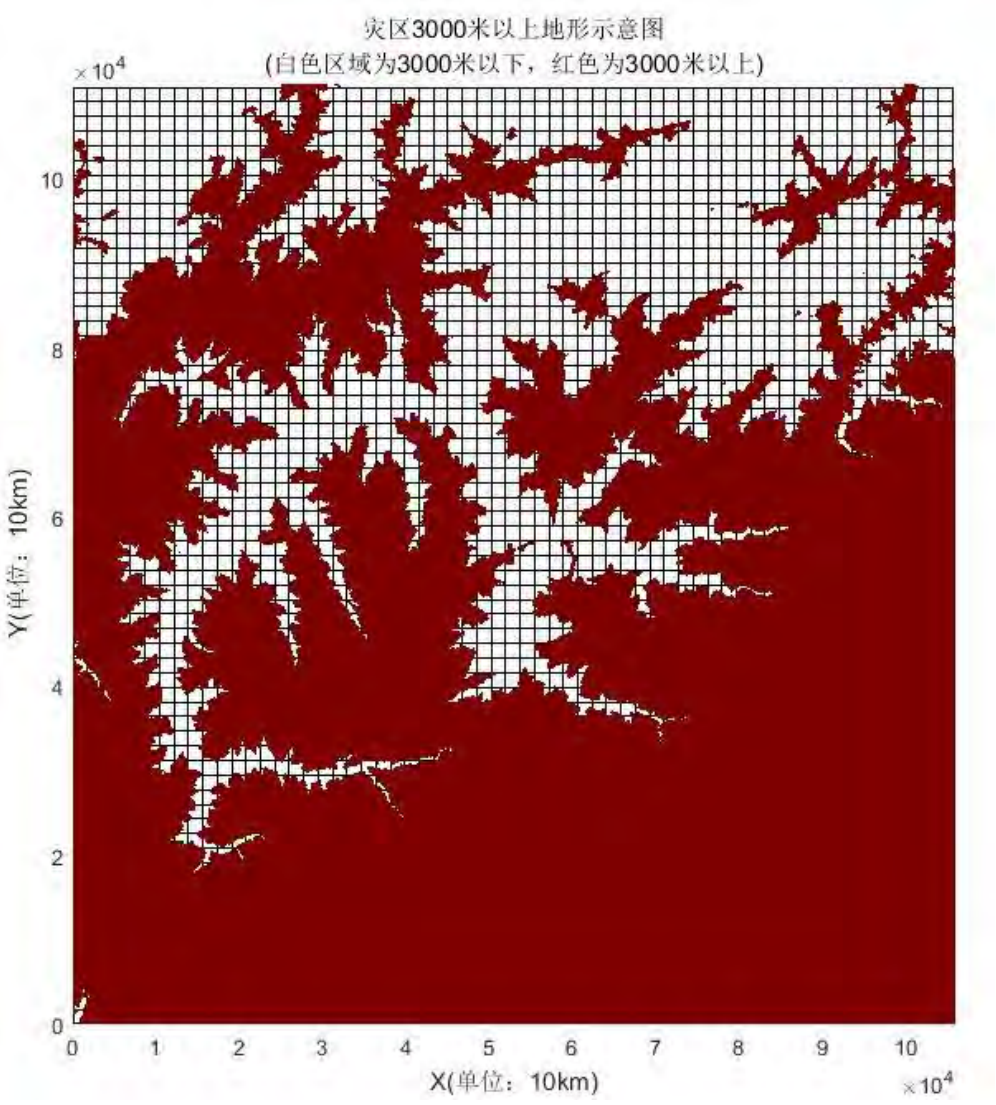


图 38 白色区域为 3000 米以下海拔，红色为 3000 米以上海拔

（2）然后将灾区地形图进行二值化和膨胀处理如下图所示

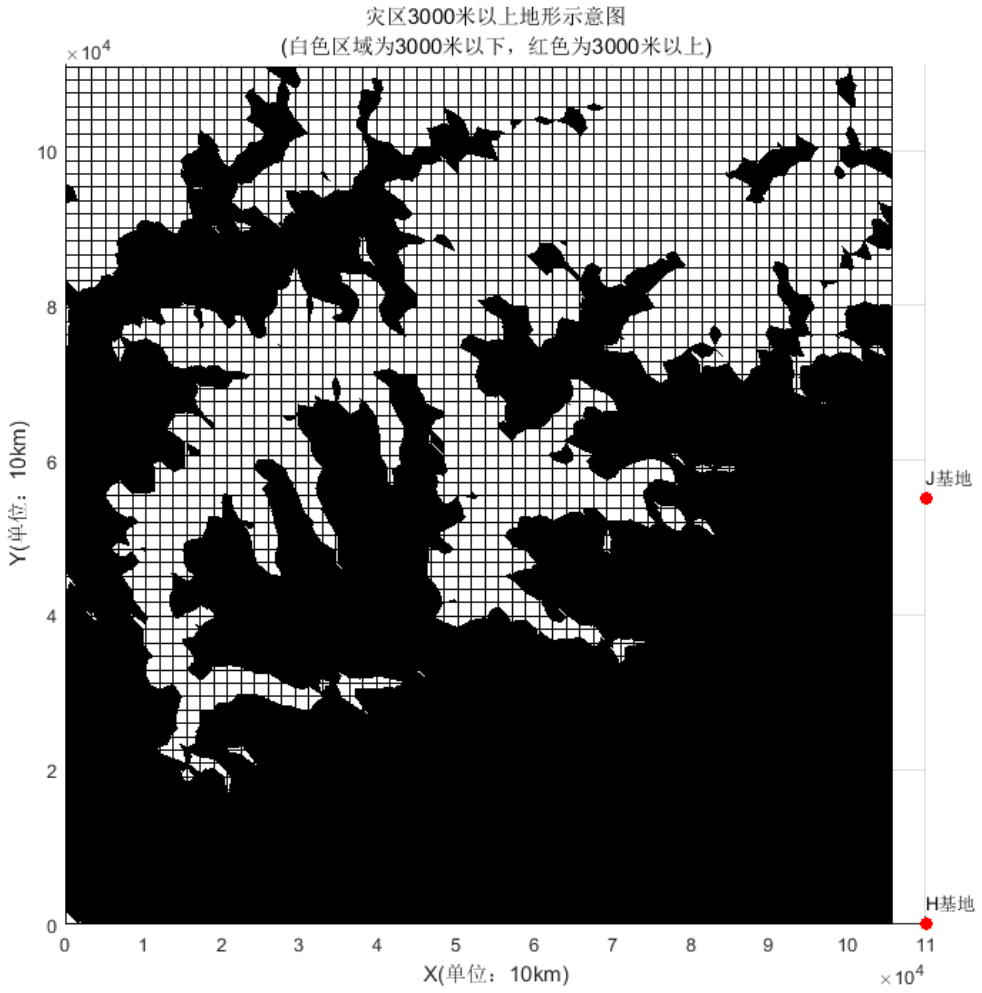


图 39 灰度二值化膨胀图

（3）在此基础上，在等高线障碍物图中选取关键搜寻点，将关键搜寻点抽象为邻接矩阵根据无人机山区搜寻的特点与邻接矩阵的性质，在选取关键搜寻搜寻点时需要注意以下几个方面：

- 无人机在搜寻过程中，并不知道伤员在何处，需要覆盖整个区域，因此顶点要遍及全部搜寻区域。为了做到这一点，选取关键搜寻点的基本思想是：空隙处须有顶点；障碍物凸起处须有顶点；障碍物凹陷处须有顶点；空旷处须有顶点。
- 根据邻接矩阵的固有特点，任意一个顶点必须至少能与另外两个顶点连通，构成一个回路。
- 在以上 2 个条件都满足的情况下，顶点数量尽可能少。

兼顾上述 3 方面的要求，对某灾区的一部分进行了关键搜寻点的选取，并将搜寻点进行编号，如图 40 所示：

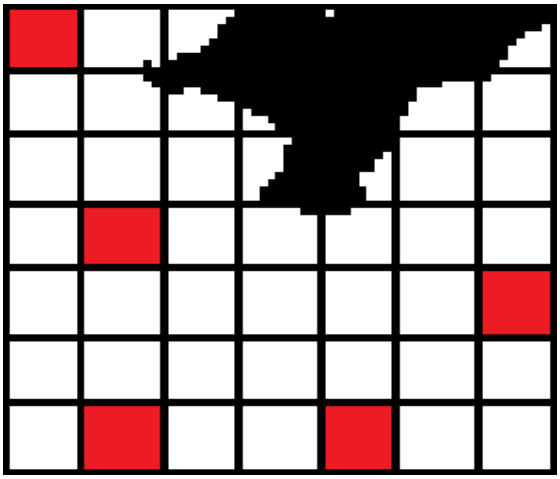


图 40 关键搜寻点示意图

（4）顶点选取完成后，将没有进行过灰度化，二值处理的原始等高线图在 Auto CAD 中打开，利用其自带的测距工具表示出所有能够连通的点之间的关系，用
“0”表示，不能不能连通的点之间的距离用“1”表示。该区域一共选择了 5 个顶点。生成的 0-1 矩阵如图 41 所示：

0	0	1	1	1	1	1
0	0	0	1	1	0	0
0	0	0	1	1	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

图 41 关键搜寻点的 0-1 矩阵表示

（5）将等高线障碍物图抽象成邻接矩阵后，在利用优化算法寻找最短路径时，将邻接矩阵的数据输入到算法的程序中，形成的搜寻路径将会避开障碍物，为无人机在障碍物繁多区域搜寻提供了重要的安全保障。

4.2.3 模型准备

由于初始种群的随机性，遗传算法的稳定性不如模拟退火和蚁群，数次迭代以后，运行结果并不是完全稳定，偶尔会有一些细微的波动，因此，在无人机生命迹象探测飞行路径规划中，我们将放弃采用遗传算法。

模拟退火[5]算法发展至今，逐渐成为了迭代自适应启发式的搜索算法，且求出全局优化解的概率比较大。模拟退火算法有众多优点：它具有较强的鲁棒性和较好的全局收敛性与隐含并行性，以及广泛的适应性。它不仅能够处理离散的优化设计参数变量，还能够处理连续的和混合的优化设计参数变量，对于目标函数和约束函数没有任何要求，也不需要借助其他的辅助信息。如果能够合理利用 Metropolis 法则来计算接受概率，并且较为精确的控制好温度下降过程，对于

旅行商问题的求解，模拟退火算法具有很强的竞争力。模拟退火算法的流程如图所示：

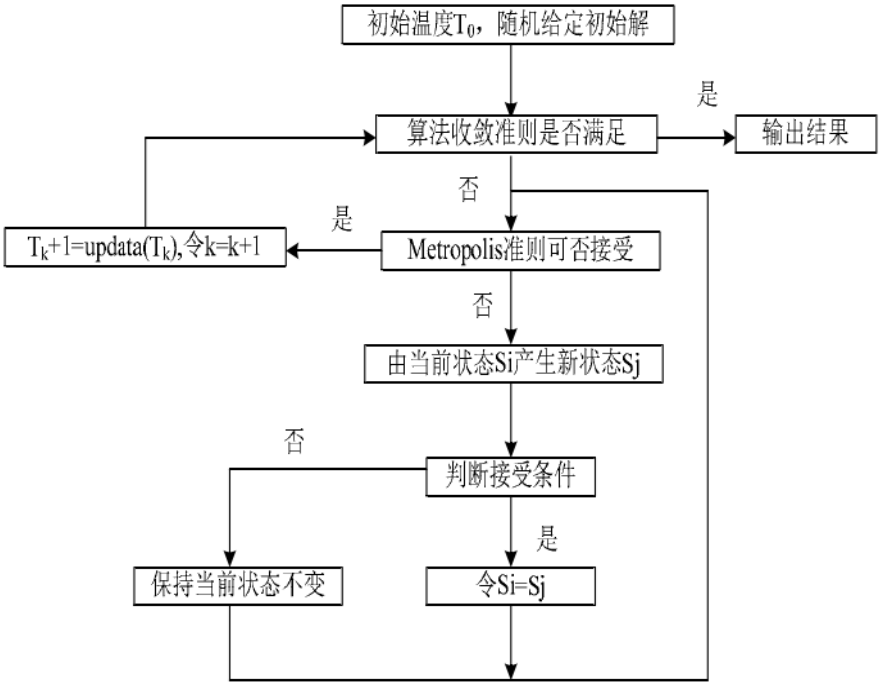


图 42 模拟退火算法求解过程

加温过程、等温过程、冷却过程组成了物理退火三大部分，模拟退火算法正是起源于这样的物理退火过程。为了让粒子热运动更佳剧烈，使粒子偏离原本的平衡位置，所以设立加温过程。从物理学角度看来，等温是必然发生的过程，系统的自由能达到最小时，系统的状态达到平衡。冷却过程是使粒子热运动减弱，系统能量逐步下降，进而得到低能晶体结构。模拟退火是照着物理退火依葫芦画瓢，两者的过程十分相似，如表所示：

表 9 模拟退火与物理退火对应关系

物理退火	模拟退火
粒子状态	解
能量最低态	最优解
溶解过程	设定初温
等温过程	Metropolis 采样过程
冷却	控制参数下降
能量	目标函数

模拟退火算法有 6 大要素，分别为：

1、状态空间和领域函数。状态空间由经过编码的可行解的集合组成；领域函数由两部分组成：生成候选解的方式、候选解生成的概率分布，它最大限度地保证了所生成的候选解遍布整个解空间。

2、状态转移概率，即接受概率，一般采用 Metropolis 准则：

$$p=\begin{cases}1,\Delta f\le 0\\ e^{\frac{-\Delta f}{t}},\Delta f>0\end{cases} \qquad \text{公式 14}$$

3、冷却表进度；指从高温状态 T_0逐渐向低温状态冷却的降温管理表。模拟退火算法降温方式分为经典和快速 2 种：经典模拟退火算法降温方式为：

$$T(t)=\frac{T_0}{\lg(1+t)} \qquad \text{公式 15}$$

快速模拟退火算法的降温方式为：

$$T(t)=\frac{T_0}{1+t} \qquad \text{公式 16}$$

其中，T_0为初始温度，t 为某一时刻；

4、初始温度；

5、Metropolis 准则也被称作内循环终止准则，它决定了各个温度将要生成候选解的个数；

用模拟退火算法对旅行商问题进行求解，其解空间 S 是当且仅当遍历 n 个城市一次的所有路径，数学方法表示为 1 到 n 的所有循环排列集合，即 S=[S_{ij}]是(1，……，n)的排列。其中，Si 表示第 i 个城市被第 Si 个访问。在旅行商问题中，模拟退火算法的目标函数为遍历全区域内海拔 3000 米以下区域的路径长度之和:

$$f(x)=\sum_{x=1}^{n-1} d\left[S(x),S(x+1)\right] \qquad \text{公式 17}$$

- **基于蚁群算法的无人机山区搜寻路径规划**[5-7]

蚁群算法跟遗传算法有异曲同工之妙，同属于启发式算法。蚁群算法的基本原理可归结为：受到了蚂蚁由觅食过程中产生的群体智能来寻找觅食路径这一基本事实的启发。蚂蚁虽然视觉不算发达，但是昆虫学家们发现，它们有一种特殊的能力可以使其借助其他同伴释放的信息素便找到从巢穴到食物源的最短路径，并且即便环境发生了改变，例如有障碍物出现，它们同样能够自动搜寻新的最佳路径。蚁群算法被认为是用于解决组合优化问题的又一种适当的方法，它具有鲁棒性强、适应性好、全局搜索等优点。蚁群活动、信息素发挥、信息素增强三大部分构成了蚁群算法的核心。蚁群算法的求解过程如图所示：

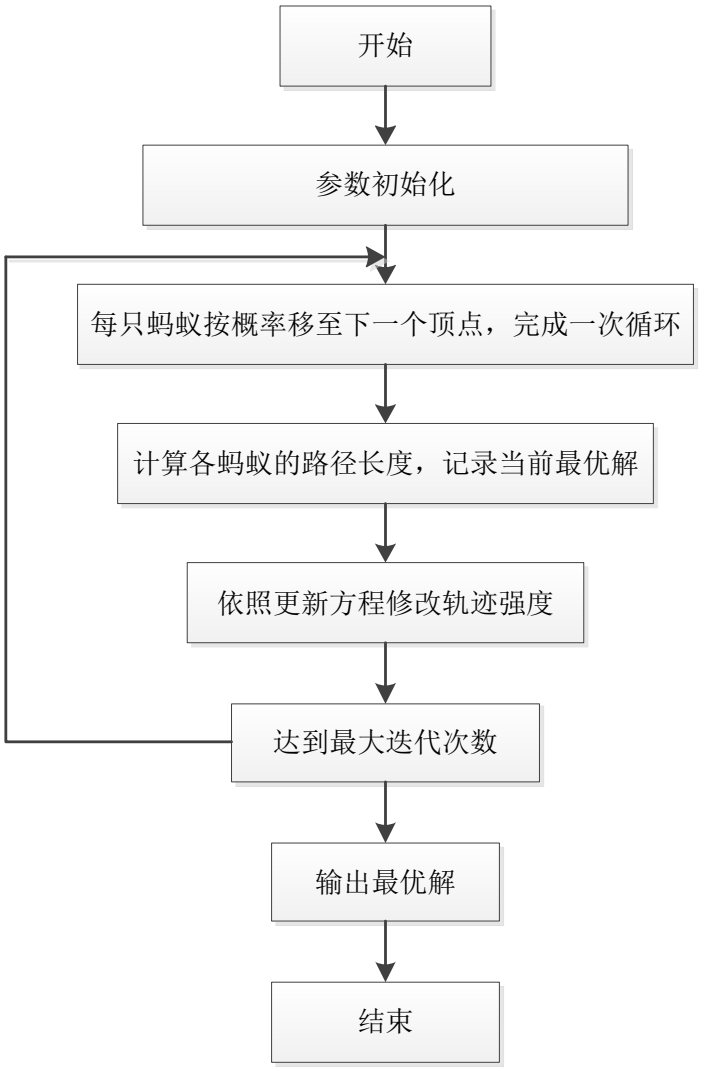


图 43 蚁群算法求解过程

用蚁群算法对旅行商问题进行求解，有 2 大步骤：路径构建和信息素更新

1、路径构建。每只蚂蚁在 n 个城市中随机选择一个城市作为它的出发点，行进中按照随机比例规则进行下一个要去城市的选择。随机比例规则如下：

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}*[n_{ij}]^{\beta}}{\sum_{k\epsilon allowed_k}[\tau_{ik}(t)]^{\alpha}*[n_{ik}(t)]^{\beta}}, & j\in allowed_k\\ 0, & j\notin allowed_k\end{cases}$$ 公式 18

其中，i 为起点，j 为终点；

η_{ij} 为能见度，是两城市 i、j 之间距离的倒数；

$\tau_{ij}(t)$ 为时间 t 从 i 到 j 的信息素强度；

$allowed_k$ 为还没访问过的城市集合；

α 为常数，是信息数加权值；

β 为常数，是能见度加权值；

2、信息素更新。假设有 m 只蚂蚁，n 个城市，信息素初始浓度如下：

$$\tau_{ij} = C \qquad \text{公式 19}$$

$$\mathrm{C} = \frac{m}{c^m} \qquad \text{公式 20}$$

如果 C 过于小，蚂蚁能够发现的信息素过大，将会很快找到一条局部最优路径；如果 C 过于大，蚂蚁能够发现的信息素过小，将会影响算法的性能，因为信息素起到的指导作用太小。为了蚂蚁能够将更多的信息素留在走过的路径上，在所有蚂蚁都到达终点时，需将个路径的信息素浓度再重新更新一次，更新分为两个步骤：第一，每一轮后所有路径上的信息素全部蒸发；第二，所有蚂蚁根据自己每轮所走过的路径长度在该路径边上释放出信息素。数学语言表示为：

$$\tau_{ij}(t) = (1-\rho)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \qquad \text{公式 21}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} (C_k)^{-1}, \text{当第 K 只蚂蚁经过 i、j} \\ 0, \qquad\qquad \text{否则} \end{cases} \qquad \text{公式 22}$$

其中， ρ 为信息素蒸发率，0＜ρ≤1；

$\Delta\tau_{ij}^{k}$为第 k 只蚂蚁从 i 至 j 留下的信息素；

C_k为第 k 只蚂蚁所走过的整条路径的总长度；

4.2.4 模型建立与求解

考虑到无人机的有效探测距离为 1732，将整片区域的 x 轴 y 轴分别除以 1732，将整片区域规划为 62*65 的网格图。其中 3000 米以下区域（下图白色区域）共有 1300 格。将 3000 米以下区域按照方格分成 30 等份，即每等份为 44 格，考虑到黑色区域也可以飞行，将每份的格子数量增加到 49 格。由 30 架无人机分别搜索每一块区域。

将区域按下图进行分块：

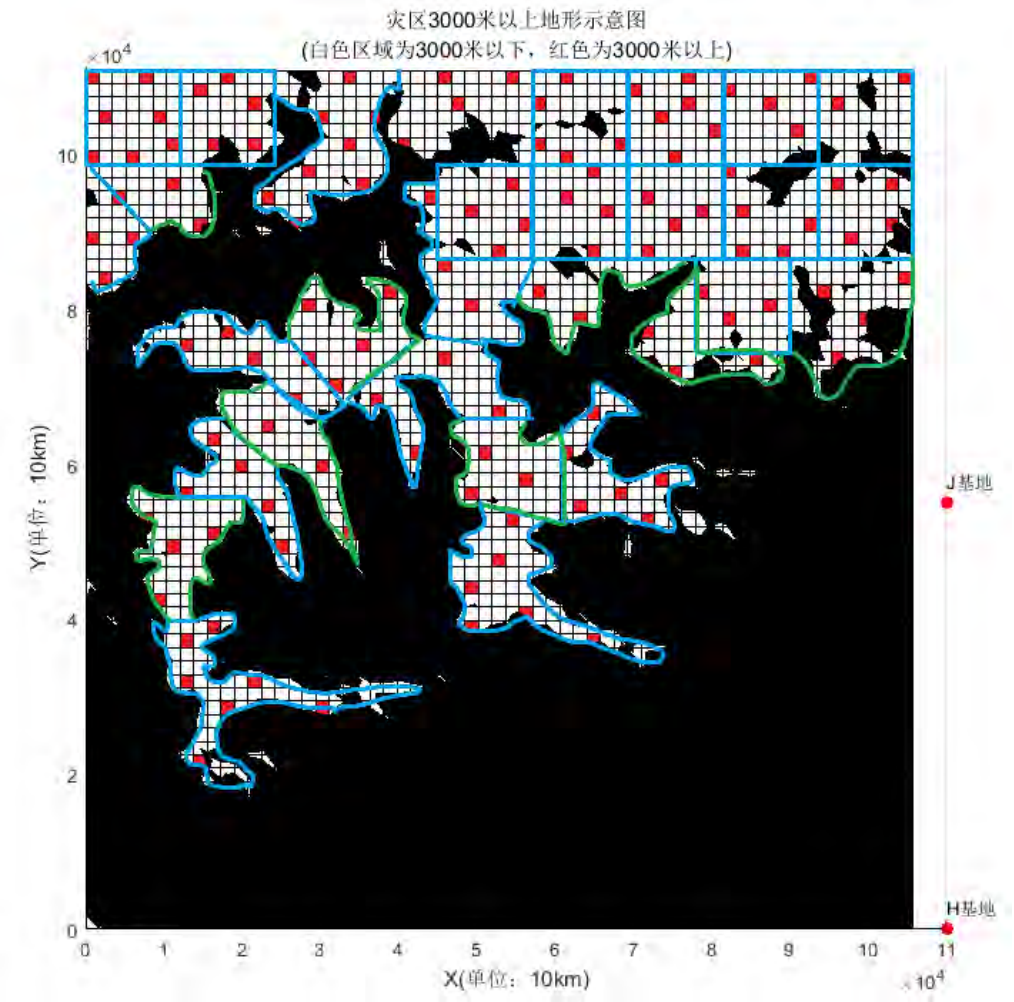


图 44 3000 米以下海拔区域分块图

- **模拟退火算法实现无人机山区路径规划**

用模拟退火算法为无人机作山区搜寻作路径规划分为如下 7 个步骤：

1、设置一个相对较高的起始温度 T，再令迭代次数 k=1；

$$f(x)=\sum_{x=1}^{n-1} d[S(x),S(x+1)] \qquad \text{公式 23}$$

2、求解目标函数的最小函数值，从而得到初始解 S0；

3、对当前解 S0 进行变换，产生全新的路径数，得到一个全新的解。变换方法有如下两种：

①任选顶点序号 u，v，交换 u 和 v 之间访问顺序；

②任选顶点序号 u，v，w，将 u，v 之间的路径插入到 w 之后访问。

4、计算新的目标函数最小值 $f(x_n)$；

5、按照 Metropolis 准则确定的概率计算$\Delta f(x)=f(x_n)-f(x)$；

6、依照退火时间表降低温度 T；

7、增加迭代次数 k，当达到最大迭代次数便停止迭代，否则返回第 3 步；在 MATLAB 中编写程序，将初始温度 t 设置为 1500℃，经过不断的训练和调试，选取图？作为例子，采用模拟退火算法得到的最短路径与等高线先障碍图叠加效果如下图？所示：

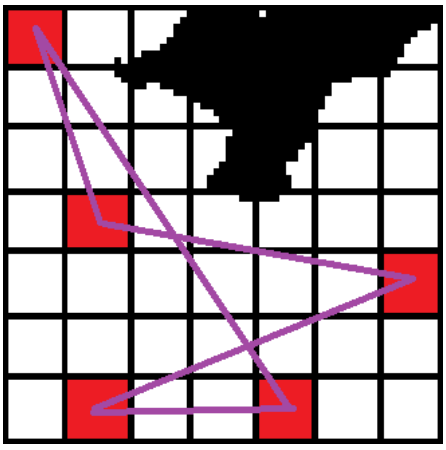


图 45 模拟退火算法所得最短搜索路径

蚁群算法实现无人机山区路径规划

蚁群算法实现无人机山区路径规划分为如下 5 个步骤：

1.令最大迭代次数 NC=0；每条边上的 τ_{ij}=C(常数)，且$\Delta\tau_{ij}$=0；放 m 只蚂蚁到 n 个关键搜寻点上；

2.将每只蚂蚁选择的初始关键搜寻点都放置于当前解集 S_k 中，对每只蚂蚁 k 按照概率 P_{ij}^k 移到下一个关键搜寻点 j，再将关键搜寻点 j 放入当前解集 S_k 中；

3.经过 n 个时刻，蚂蚁 k 便可以将所有关键搜寻点遍历一次，完成一次完整的循环。此时，计算出每只蚂蚁走过的路径总长度 d_k，更新出最短路径；

4.更新每条边上的信息素 τ_{ij}（t+n)；对于每条边，$\Delta\tau_{ij}$=0；NC+1；

5.如果 NC 小于预定的迭代次数 NCmax，那么调到第 2 步；否则，输出最短路径，整个程序终止；

采用蚁群算法得到的无人机最短搜寻路径与等高线先障碍图叠加效果如下：

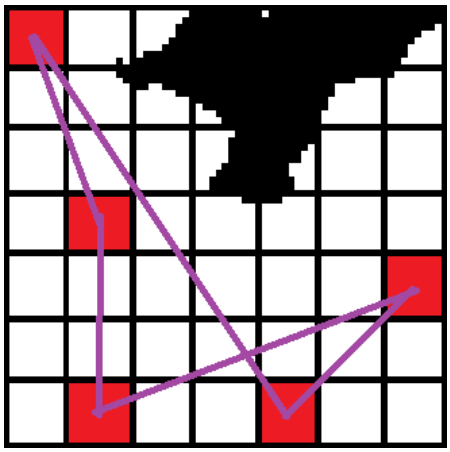


图 46 蚁群算法所得最短搜索路径

为了更好更高效的规划出无人机山区搜寻路径，本题将从上述两种算法中，选取一种相对更加优化的算法作为无人机山区搜寻的固定算法。在模拟退火算法和蚁群算中，蚁群算法得到的最短路径，比模拟退火算法的最短路径短了 3.5 千米，但是程序实际运行时间比模拟退火算法多了 0.4h，对于山区搜寻的无人机来说，几乎没有任何影响，而且覆盖的 3000 米以下的区域更广，且在实际运行过程中，蚁群算法的稳定性强于模拟退火算法，模拟退火算法有时会陷入局部最优解。

综上所述，在无人机山区路径规划中，本题更看重蚁群算法的稳定性、高质量的解、通用易实现以及初始值鲁棒性（Robustness）强的特点。最终，将蚁群算法定为无人机山区搜寻路径规划的最优算法。

4.2.5 求解结果

使用蚁群算法得到的无人机最优飞行路径如下图所示：

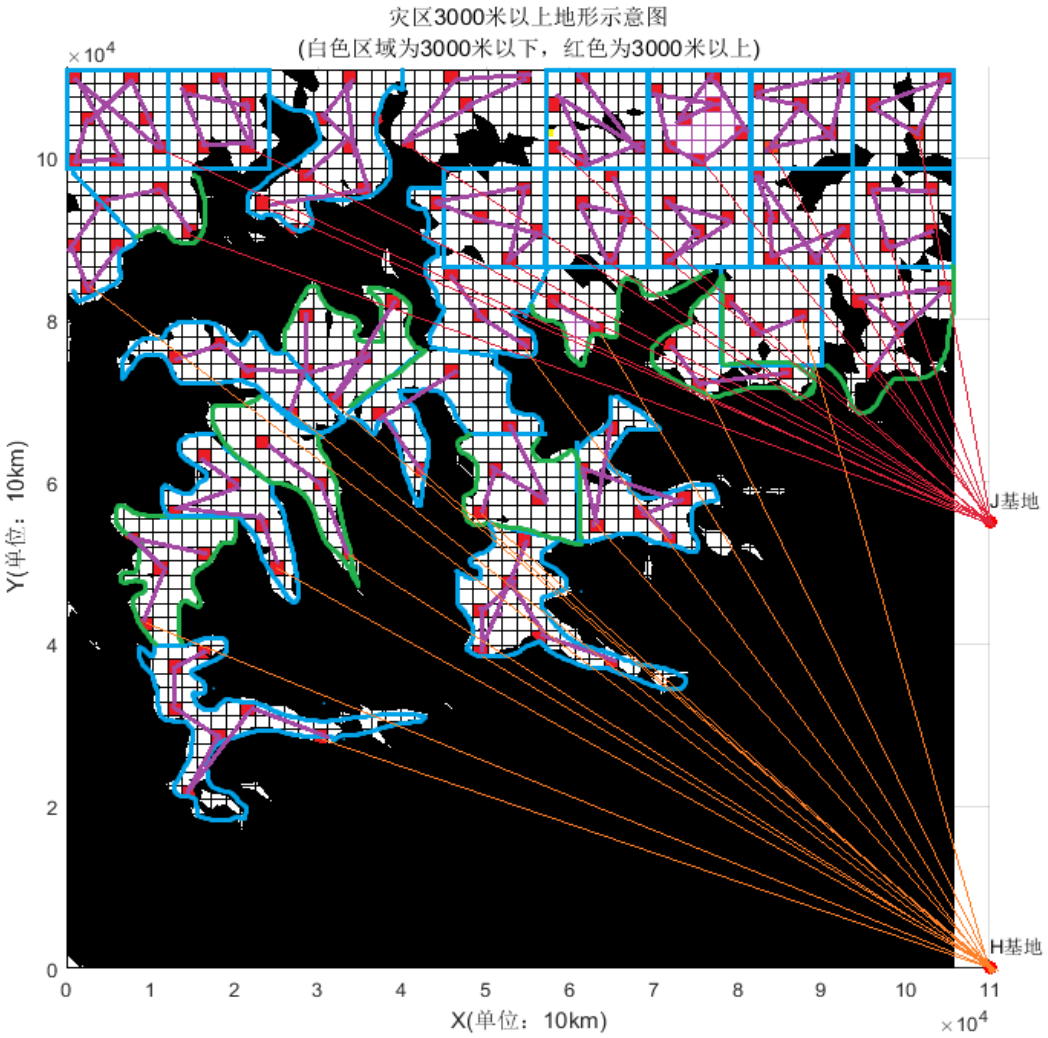


图 47 无人机最优飞行路径

从第一架无人机飞出到最后一架无人机完成任务回到基地的时间间隔为 4.9 小时。

4.3 对问题三的分析

4.3.1 问题描述及分析

问题三的研究背景是当大地震发生后，地面电力设备被破坏，灾区通信中断。太阳能无人机（白天不受续航能力限制，其余条件同前述）可以作为通信中继，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。通过建模确定无人机的数量和路线，保证在白天 12 小时内，附件 2 中的任意两个地面中断之间都能实现不间断通信。

根据题目和参考文献，给出问题三的求解策略如下：

（1）本文假设地面移动终端保持静止（忽略移动距离），那么我们根据附件 2 可以得知地面终端的地理位置。

（2)根据对附件 2 的数据处理，本题利用粒子群算法对地面终端进行分区。

（3）使用粒子群算法对各个分区求最短路径。

（4）无人机之间的最大通信距离为 6000 米，为了减少无人机的使用，本文考虑将所有的地面移动终端都包含在以无人机为圆心半径为 6000 米的圆周内（无人机此时是静止的)。

（5）经过（4)，发现有的地面移动终端还是不能进行通信，此时可以考虑加无人机。

（6）上述解决之后开始考虑无人机的运动带来的影响。

4.3.2 数据处理

（1）根据附件 2 的数据我们可以得到 72 个地面移动终端的分布如下：

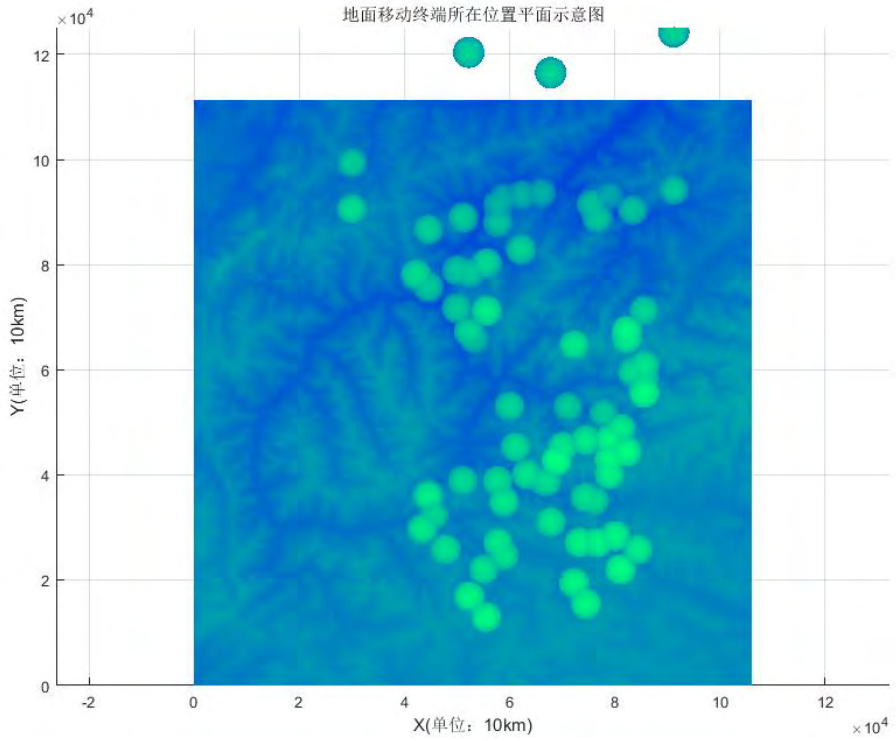


图 48 地面移动终端平面图

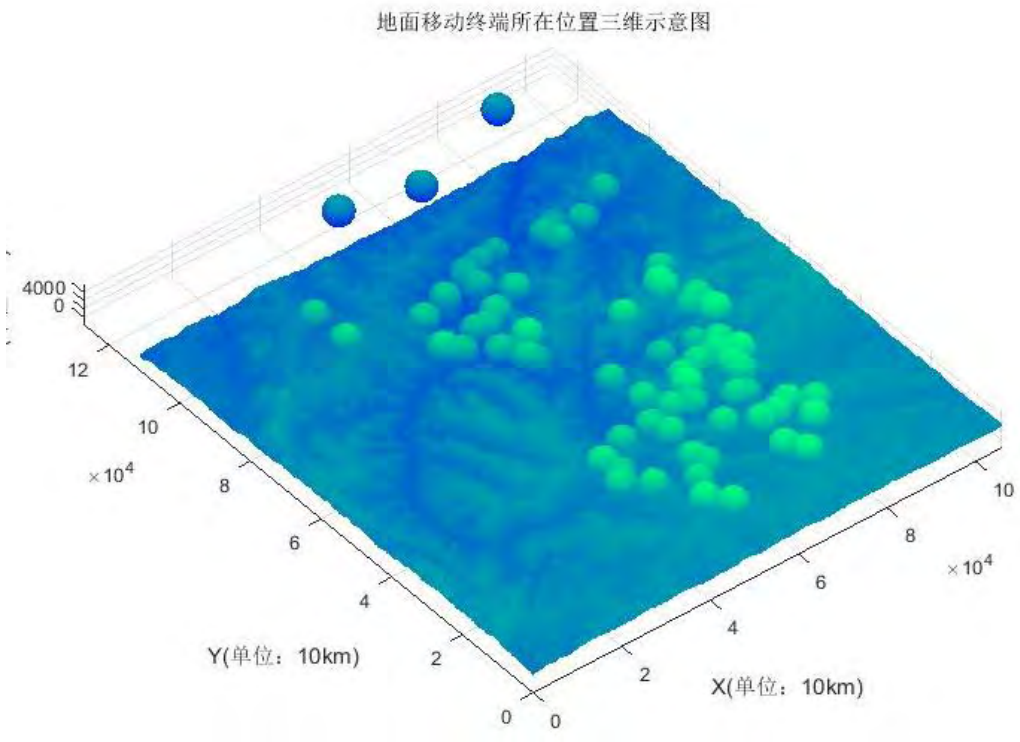


图 49 地面移动终端三维图

（2）以地面移动终端为圆心半径为 3000 米的圆周：

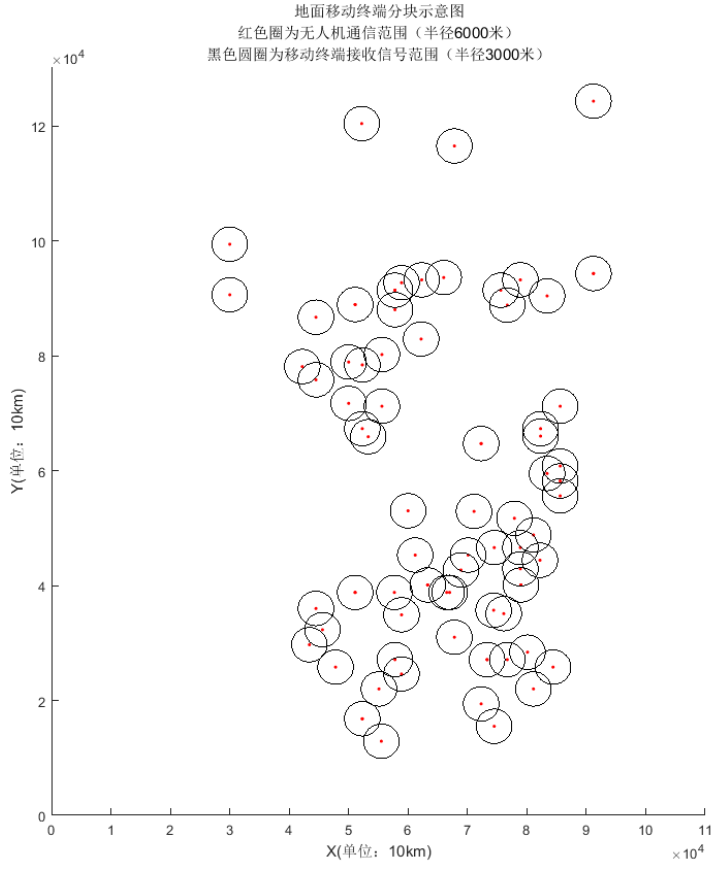


图 50 以地面移动终端为圆心半径为 3000 米的圆周平面图

4.3.3 模型准备

在问题三中，我们主要运用粒子群算法和一元线性回归组合模型对问题进行建模求解。

粒子群算法[5]：

现采用粒子群算法实现这一最优规划。粒子群算法（Particle Swarm Optimization，PSO）是一种基于群体的随机优化技术[2]，它将每个可能产生的解表述为群中的每一个微粒，每个微粒都具有自己的位置向量和速度向量，以及一个由目标函数决定的适应度，所有微粒在搜索空间中以一定速度飞行，在初始化一组随机解后通过迭代搜寻最优解。

粒子群算法解决该问题的过程为[2]：

（1）初始化一个群体规模 m=40 的二维粒子群，随机初始化粒子的位置 xis 和速度 vis；

（2）计算每个粒子的适应值；

（3）对每个粒子将其适应值和其经历过的最好位置 pis 的适应值进行比较，若较好，则将其作为当前的最好位置；

（4）对每个粒子将其适应值和全局经历过的最好位置 pgs 的适应值进行比较，若较好，则将其作为当前的全局最好位置；

（5）根据以下两个式子对粒子的速度和位置进行更新；

$$v_{is}(t+1) = w * v_{is}(t) + c_1 r_1[p_{is}(t) - x_{is}(t)] + c_2 r_2[p_{gs}(t) - x_{gs(t)}]$$

$$x_{is}(t+1) = x_{is}(t) + v_{is}(t+1)$$

其中，惯性权重 w 取 0.7298，学习因子 c1=2、c2=2，r1、r2 为服从[0,1]上均

匀分布的相互独立的伪随机数。

（6）如果达到设定的最大迭代次数 45，则输出解，否则返回步骤（2）。

一元线性回归模型简介[4]：

一元线性回归方程反映一个因变量与一个自变量之间的线性关系，当直线方程 Y = a + bx 的 a 和 b 确定时，即为一元回归线性方程。

经过相关分析后，在坐标系中将大量数据绘制成散点图，这些点不在一条直线上，但可以从中找到一条合适的直线，使各散点到这条直线的纵向距离之和最小，这条直线即为回归直线，直线的方程即为直线回归方程。

回归分析研究的主要问题[4]：

- 确定 Y 与 X_1, X2, …, Xp 间的定量关系表达式，这种表达式即为回归方程；
- 对求得的回归方程的可信度进行检验；
- 利用所求得的回归方程进行预测和控制；

一元线性回归的数学模型：

将问题一数据准备中所得的中心目标点坐标放入直角坐标系中，从图 3 散点图发现，中心目标点基本集中在几条斜率不同的直线附近，从而可以认为 Y 与 X 的关系基本是分组呈线性相关，而点与直线的偏离是由其他一切不确定因素所造成，为此可以做如下假定：

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

其中，$\beta_0 + \beta_1 X$表示 Y 随 X 的变化而线性变化的部分；ε是随机误差，它是一切不确定影响因素的总和，其值不可观测，通常假定$\varepsilon \sim \mathrm{N}(0, \sigma^2)$，$\beta_0$为回归常数，$\beta_1$为回归系数，统称为回归参数，X 为回归自变量（回归因子），Y 为回归因变量（或响应变量）。

若$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$是(X,Y)的一组观测值，则一元线性回归模型可表示为：

$$y_i = \beta_0 + \beta_1 x_i + \in_i, i = 1,2, \ldots, n \qquad \text{公式 24}$$

回归方程的显著性检验：

从回归参数的估计可知，在计算过程中并不一定要知道 Y 与 X 是否有线性相关的关系，但如果不存在这种关系，那么求得的回归方程毫无意义，因此，需要对回归方程进行检验。

从统计学考虑，β_1是 E(Y)随 X 线性变化的变化率，若$\beta_1 = 0$，则 E(Y)实际上并不随 Y 作线性变化，仅当$\beta_1 \neq 0$时，E(Y)才随 X 作线性变化，也仅在这时一元线性回归方程才有意义，因此假设检验为

$$H_0: \beta_1 = 0, H_1: \beta_1 \neq 0 \qquad \text{公式 25}$$

通常采用三种方法：

（1）t 检验法. 当H_0成立时，统计量：

$$T = \frac{\widehat{\beta_1}}{sd(\widehat{\beta_1})} = \frac{\widehat{\beta_1}\sqrt{S_{xx}}}{\hat{\sigma}} \sim t(n-2) \qquad \text{公式 26}$$

对于给定的显著性水平σ，检验的拒绝域为：

$$|T| \geq t_{\frac{\sigma}{2}}(n-2) \qquad \text{公式 27}$$

（2）F 检验法. 当H_0成立时，统计量：

$$F = \frac{\widehat{\beta_1}^2 S_{xx}}{\hat{\sigma}^2} \sim F(1, n-2) \qquad \text{公式 28}$$

对于给定的显著性水平σ，检验的拒绝域为:

$$F \geq F_\sigma(1, n-2) \qquad \text{公式 29}$$

（3）相关系数检验法. 相关系数 R 定义为：

$$R = \frac{S_{xy}}{\sqrt{S_{xx} S_{yy}}} \qquad \text{公式 30}$$

对于给定的显著性水平σ，查相关系数临界值表可得 $r_{\alpha}(n-2)$，则检验的拒绝域为：

$$|R| > r_{\alpha}(n-2) \qquad \text{公式 31}$$

当拒绝 H_0 时，认为线性回归方程是显著的。

4.3.4 模型建立与求解

Step 1:利用粒子群算法把地面移动终端划分为 11 个区域，并且用粒子算法得出各个区域内的最优路径。

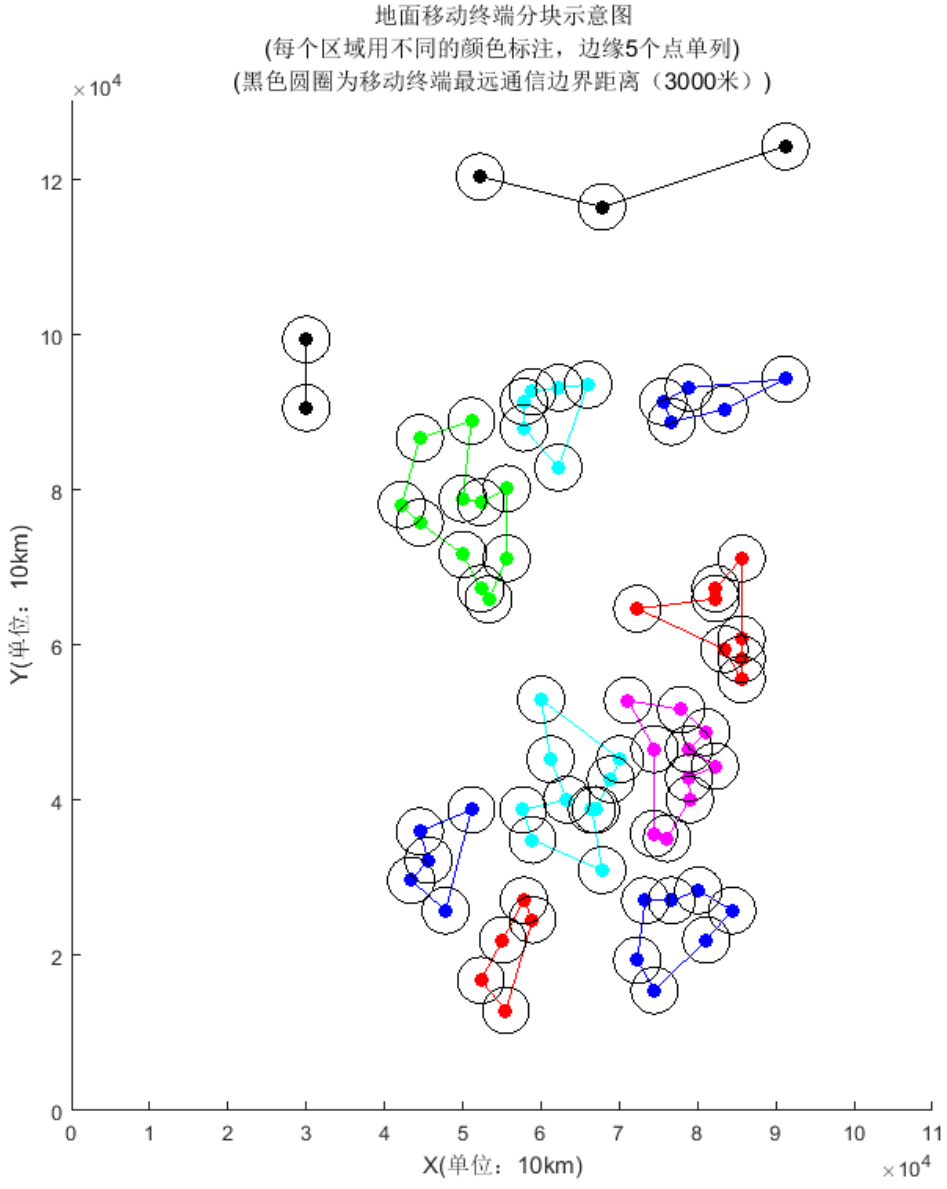


图 51 地面移动终端分块示意图

Step 2:本文使用一元线性回归确定了无人机在静止时候的位置。以无人机为圆心半径为 6000 米的圆周，为保证通信不中断，本文考虑红色圆周将黑色圆周包含在内（无人机此时静止）。红色圆周相切或者相连可以使无人机之间保持

通信，其中 A、B、C 三个圆周负责把分散的区域接连在一起，确保通信完整。，此时无人机的架数为 43

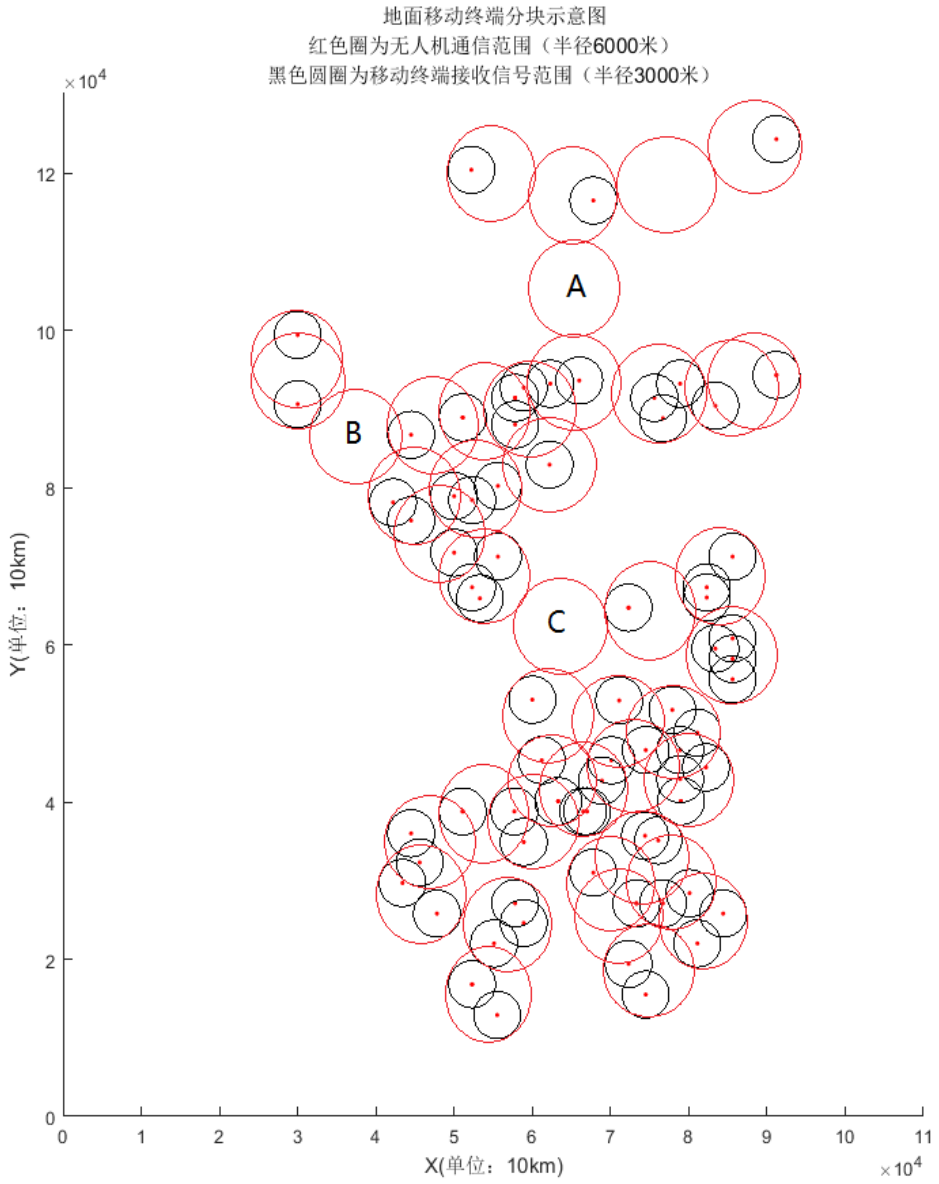


图 51 理想包含状态示意图

可是在实际中无人机并不是静止不动的，因此本文考虑各个分区内的无人机循环飞行。经过计算，无人机的架数分布如下图所示：

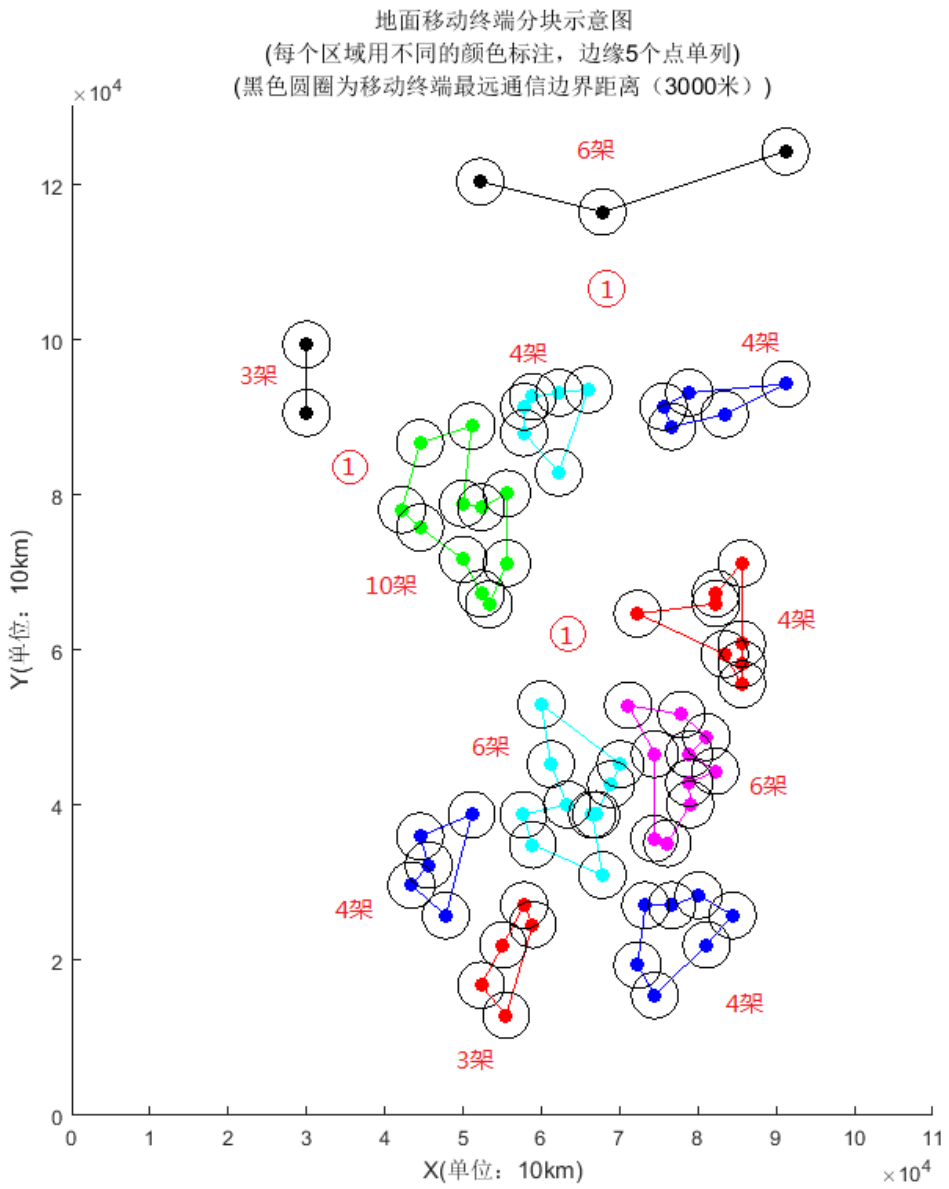


图 52 每个分区内的无人机架数分布图

4.3.5 求解结果

根据以上的分析，每架无人机的飞行路线如下图所示：

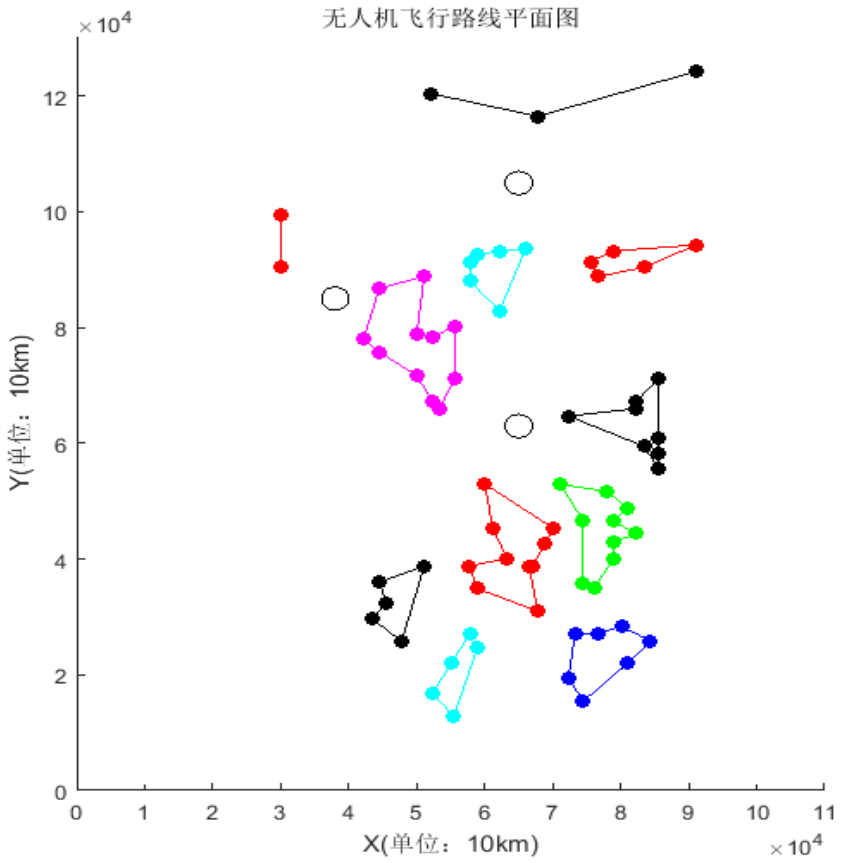


图 53 无人机飞行路线平面图

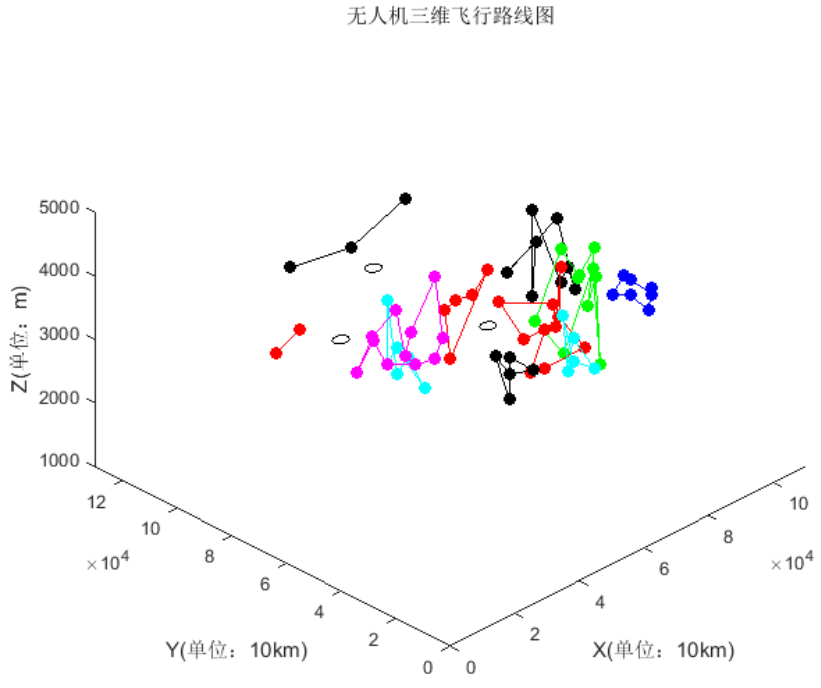


图 54 无人机飞行路线三维图

图中的三个小黑圈代表三架无人机，负责不相连分区的信息传递，黑圈的半径为100米（无人机飞行时的转弯半径不小于100米）。**因此最少需要57架无人机，才能保证在白天12小时内，附件2中的任意两个地面移动终端之间都能实现不间断通信。**

4.4 对问题四的分析

4.4.1 问题描述及分析

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(1M按10^6比特计算)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。数据传输过程可以简化为：当地面终端i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，如果无人机当前服务用户少于10个，则开始向i发送数据，并瞬间完成所有用户的功率再分配，否则，搁置i的需求，直到有地面用户退出，若此时i仍在可服务区域，则为i服务（先到先服务）。如果在一个服务时间区间（即无人机和终端之间满足可传输数据条件的时间范围）内不能传完全部数据，则以后区间可以续传。再设i用户在时刻t接收到无人机发送的信息速率为$r_i(t)=B_i\log_2\left(1+\dfrac{p_i(t)}{\rho_0 d^2(u,i)}\right)$(比特/秒)，其中$B_i$表示无人机服务$i$的子信道带宽(取值见附件2，单位Hz)，$p_i(t)$表示$t$时刻无人机为第$i$个地面用户所在的子信道分配的功率,单位：w(瓦)，$d(u,i)$表示t时刻无人机与i之间的欧氏距离，单位：米。ρ_0为信道特性参数，为简单起见，取为4.314×10^{-10}(单位略)，假设无人机飞行速度在60~100千米/小时之间可调(水平面内最大加速度±5米/秒^2，铅垂面内最大加速±2米/秒^2，可同时在两个方向上加速)，为无人机设计恰当的航线、速度以及所服务的用户，并为每一个用户分配恰当的功率，使得无人机完成所有任务的时间总和尽量短。

根据题目和参考文献，给出问题四的求解策略如下：

（1）无人机在8号、9号、50号、52号和63号地面移动终端的上空50米处进行数据传输，与其余移动终端的数据传输距离为3000米。

（2）无人机给8号、9号、50号、52号和63号地面移动终端分配5瓦的功率，给其余点分配的功率均为2.5瓦。

（3)根据对附件2的数据处理,本题利用粒子群算法对地面终端进行分区。

（4）无人机和地面移动终端之间的最大通信距离为3000米，最小通信距离为50米。

4.4.2 数据处理

（1） 根据附件2的数据我们可以得到72个地面移动终端的分布如下：

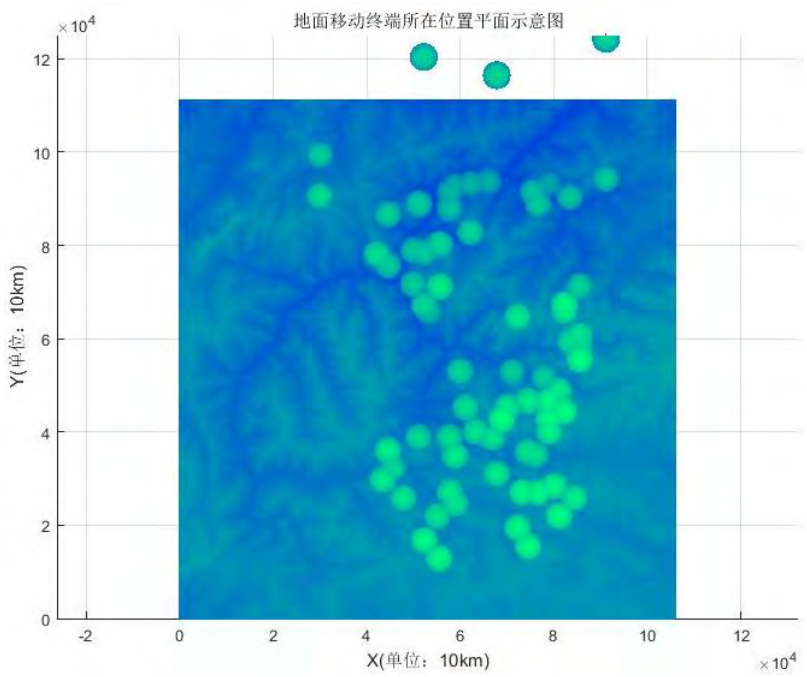


图 55 地面移动终端平面图

（2）以地面移动终端为圆心半径为 3000 米的圆周，如下图所示：

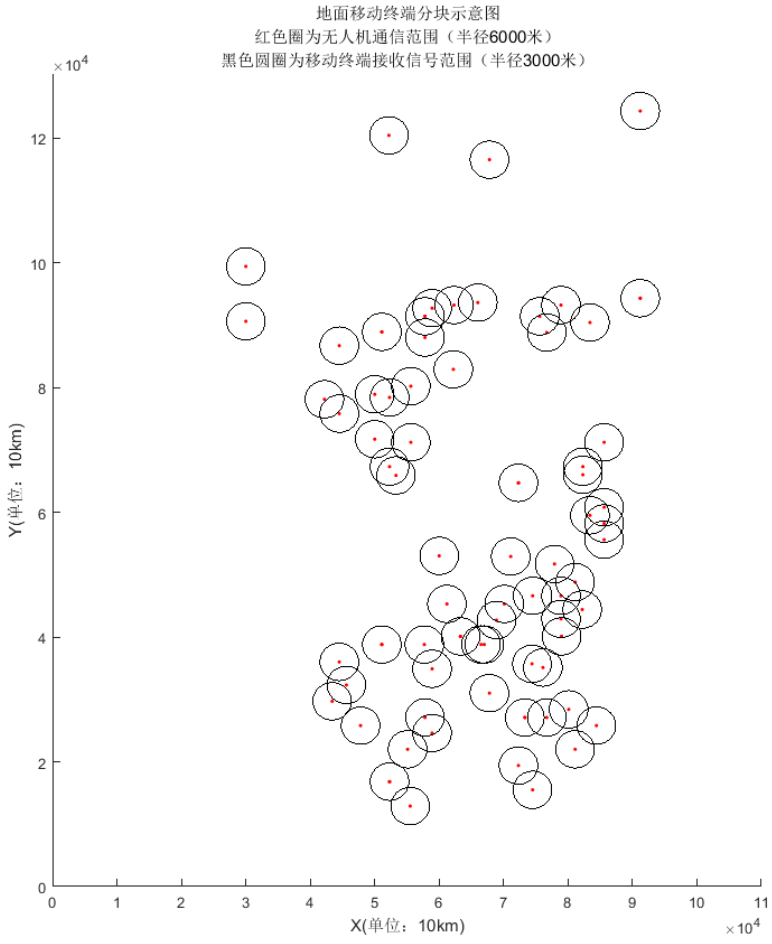


图 56　地面移动终端的通信边界距离（3000 米）

4.4.3 模型准备

在问题四中，我们主要运用粒子群算法和非线性规划模型对问题进行建模求解。

粒子群算法见问题三。

非线性规划模型：

非线性规划方法是一种数学规划方法，它具有非线性的约束条件或者目标函数，可以求解非线性的最优化问题。非线性规划的优化方法，是在线性规划方法上的发展出来的。

求解非线性规划的算法有二次规划，一般非线性规划等方法。本文采用一般非线性规划方法，其标准模型如下：

$$\begin{aligned} &\min F(X)\ \ Aeq.X = beq \\ &s.t\ AX \le b\ \ G(X) \le 0 \\ &Ceq(X)=0\ \ VLB \le X \le 0 \end{aligned} \qquad \text{公式 32}$$

公式中为维变元向量，为线性不等式的系数矩阵，与均

为非线性函数或线性函数组成的向量，与分别为的下界与上界

对于一个实际的移动机器人路径规划问题，在把它的路劲规划问题分析成非线性规划问题时，一般要注意如下几点：

1、确定路径规划的方案：首先要分析通信无人机和地面移动终端有关的资料和数据，在全面熟悉通信无人机运动原理的基础上，确认通信无人机本体的运动约束条件，并用一组变量来表示它们。

2、提出明确的地面终端划分方案：经过数学原理分析，根据实际通信无人机通信目标所需要的要求，以及地面移动终端的位置信息，对 72 个地面终端进行划分。

3、寻求环境的限制条件：由于通信无人机的通信对象一般都要在一定的环境约束下取得路径最短或者最优的效果，因此还需要寻找出移通信无人机在通信时的所有限制条件，这些限制条件通常用变量之间的一些不等式或等式来表达。

此基础上进行下一步数据计算，来确定无人机的航线、完成虽有任务的时间总和尽量短。

4.4.4 模型建立与求解

Step 1:利用粒子群算法把地面移动终端划分为 11 个区域，并且用粒子算法得出各个区域内的最优路径。

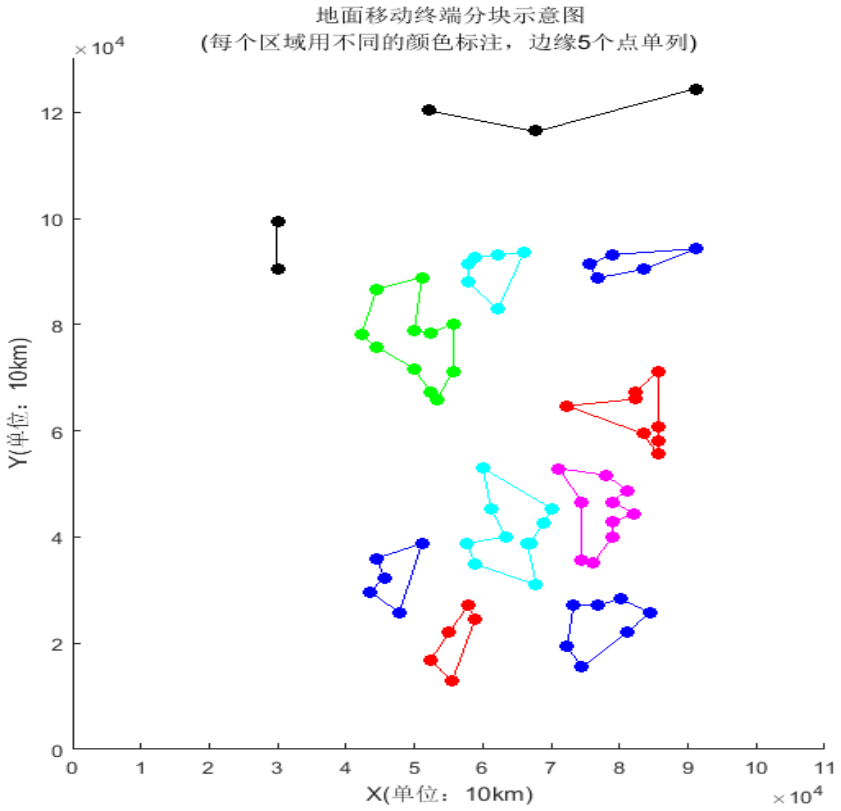


图 57 地面移动终端区域一次划分示意图

Step 2:跟据上述区域的划分，再将地面移动终端平均分配给三架通信无人机，二次划分结果如下图所示：

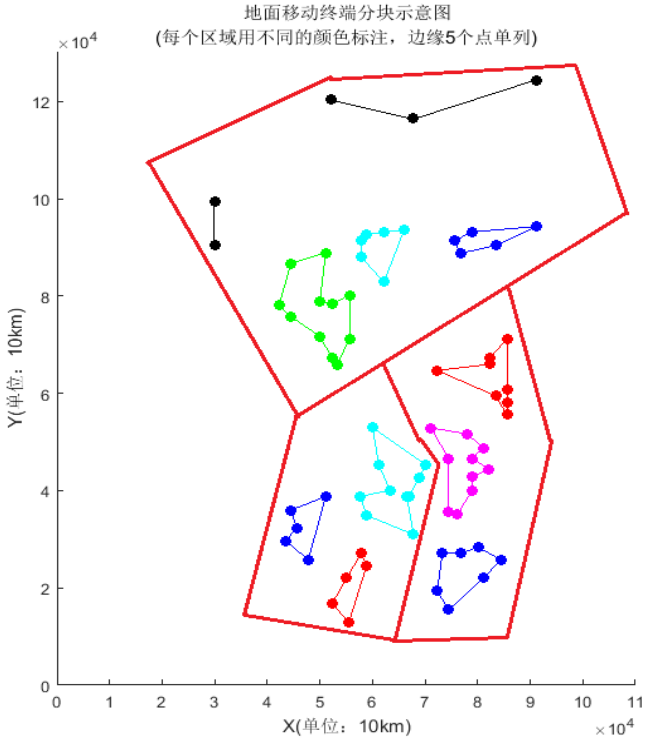


图 58 地面移动终端区域二次划分示意图

Step 3:根据 $r_i(t)=B_i\log_2\left(1+\frac{p_i(t)}{\rho_0 d^2(u,i)}\right)$(比特/秒)公式，计算出无人机和地面移动终端相距 50 米、1500 米和 3000 米的传输速率，如下表所示：

表 9 无人机和移动终端相距 50 米、1500 米和 3000 米时的速率表达式

无人机和移动终端距离 D（m）	速率表达式
50	$r_i(t)_{50}=B_i\log_2\left(1+\frac{p_i(t)}{1.079*10^{\wedge-6}}\right)$
1500	$r_i(t)_{1500}=B_i\log_2\left(1+\frac{p_i(t)}{0.000971}\right)$
3000	$r_i(t)_{3000}=B_i\log_2\left(1+\frac{p_i(t)}{0.0038826}\right)$

然后再分别计算每个用户功率为 5 瓦和 2.5 瓦时的传输速率，如下表所示：

表 10 无人机和移动终端相距 3000 米 且功率为 2.5 瓦时的速率表达式

无人机和移动终端距离 D（m）	功率为 2.5 瓦时的速率表达式
3000	$r_i(t)_{(3000\&2.5)}=B_i\log_2\left(1+\frac{p_i(t)}{0.0038826}\right)$=9.33 Bi

表 11 无人机和移动终端相距 50 米且功率为 5 瓦时的速率表达式

无人机和移动终端距离 D（m）	功率为 5 瓦时的速率表达式
50	$r_i(t)_{(50,5)}=B_i\log_2\left(1+\frac{p_i(t)}{1.079*10^{\wedge-6}}\right)$=22.14 Bi

Step 4：由上述计算结果，根据 $t=\frac{F}{r_i(t)}$ 计算出各点完成 500M 数据传输所需要的时间，以及无人机经过各点的先后顺序，如下表所示：

表 12 区域 1 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（min）	无人机经过该点顺序
18	7.44	1
19	8.93	2
20	7.44	4
21	5.95	3
22	4.47	6
24	8.93	5
25	5.25	8
26	7.44	7

表 13 区域 2 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（min）	无人机经过该点顺序
37	8.93	1
43	6.87	3
47	5.25	6
48	5.95	4
55	7.44	2
59	4.70	7
61	6.38	5

表 14 区域 3 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（min）	无人机经过该点顺序
38	4.47	1
44	8.93	2
57	6.38	5
58	5.58	4
67	8.93	3

表 15 区域 4 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（h）	无人机经过该点顺序
31	5.95	1
32	6.38	4
51	6.87	3
56	7.44	2
62	4.96	5

表 16 区域 5 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（h）	无人机经过该点顺序
16	4.47	1
28	6.38	2
29	5.58	10
35	6.38	3
45	4.96	6
54	5.25	5
60	4.70	9
65	4.96	7
69	8.12	4
71	5.25	8

表 17 区域 6 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（min）	无人机经过该点顺序
23	7.44	1
27	6.38	5
36	6.38	10
41	6.38	7
42	5.95	4
46	8.93	2
64	8.12	8
68	5.25	6
70	5.25	3
72	5.25	9

表 18 区域 7 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（min）	无人机经过该点顺序
7	5.25	1
10	6.87	3
13	4.47	8
14	5.58	10
15	5.95	5
17	6.38	11
30	4.47	6
33	4.47	7
34	5.95	9
53	4.96	2
66	5.58	4

表 19 区域 8 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（min）	无人机经过该点顺序
1	4.96	1
2	5.58	2
3	4.96	3
5	7.44	4
49	4.70	5

表 20 区域 9 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（min）	无人机经过该点顺序
4	4.70	1
6	4.96	2
11	5.58	6
12	4.47	5

39	6.38	4
40	7.44	3

表 21 区域 10 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（min）	无人机经过该点顺序
50	2.9	2
52	2.51	1
63	3.14	3

表 22 区域 11 内各点完成数据传输需要的时间及无人机飞行顺序

移动终端序号	完成数据传输需要的时间 T（min）	无人机经过该点顺序
8	2.35	1
9	1.98	2

Step 5：根据上述计算结果，结合粒子群算法得出无人机经过各区域完成传输任务所需时间（50、52、63、8 和 9 号基地单独遍历）单位为 min，如下图所示：

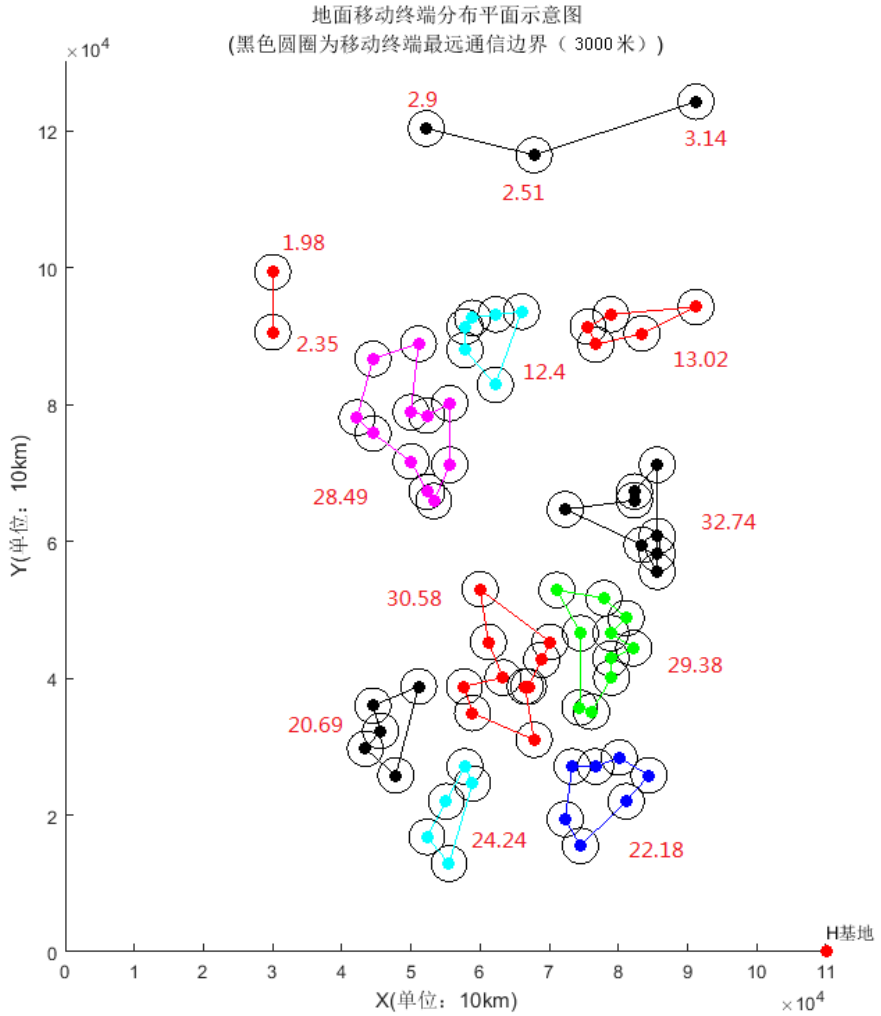


图 59 无人机经过各区域完成传输任务所需时间

4.4.5 求解结果

根据以上的分析，每架无人机的飞行路线如下图所示：

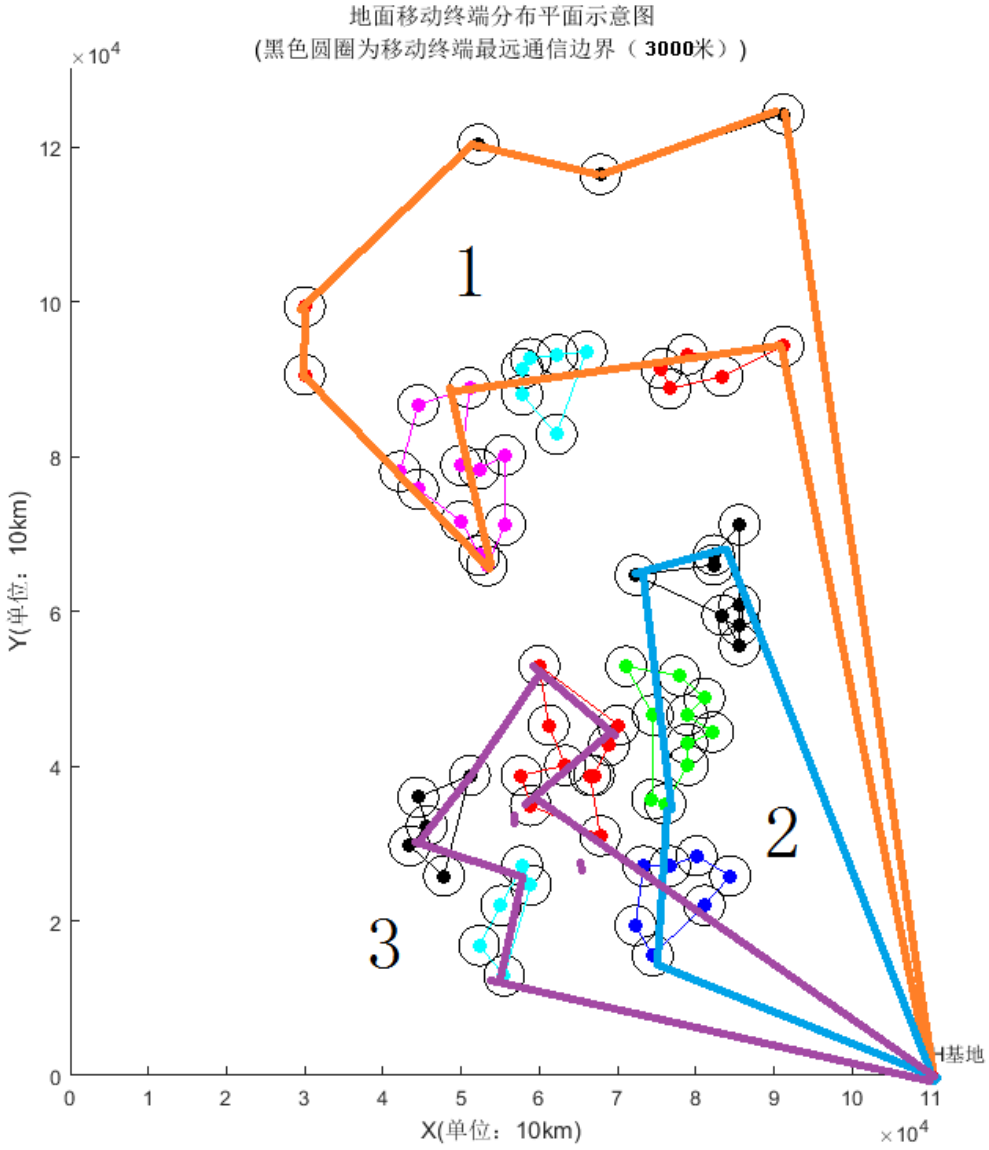


图 60 三架无人机的飞行路线

三架无人机执行任务所需要的时间如下表所示：

表 23 三架无人机完成任务所需要的时间

无人机编号	完成任务需要的时间 T（h）
1	6.433
2	3.697
3	3.511

5 模型的评价

5.1 模型的优点

（1）本文充分考虑了模型的现实条件，对模型的多种可能情况进行了分析讨论，如在问题 1 中，绘制了整个灾区的地形图，考虑高海拔的地区和无人机无法飞行的高度，得到了 4 种飞行路径方案，并通过算法实现最优；除此之外，在每一步得出结果之后，都对结果进行反复验证，确保结果的真实准确；

（2）模型的扩展性和可移植性比较强，当题目中样本的性状和位点的数量和复杂性发生变化时，此模型仍然适用。

（3）模型的实用性较强，模型易于理解，符合实际，计算也相对简便。

5.2 模型的缺点

（1）模型在一些条件上进行了假设，对某些信息考虑不足，模型还不够完善。

（2）模型有些条件都采取了理想化的处理，实际中，应该更加细致的去讨论。

6 参考文献

[1] 邓启波，多无人机协同任务规划技术研究，北京理工大学，无，摘要，2014。

[2] 赵明，李涛，苏小红,等,三维多无人机系统协同任务规划关键问题综述，智能计算机与应用，6（1）：31-34，2016。

[3] 于红斌，李孝安，基于栅格法的机器人快速路径规划，微电子学与计算机，22(6)：98-100，2005。

[4] 科普中国，线性回归，http://baike.baidu.com/view/449540.htm.2016/9/18，2017.9.27。

[5] 卓金武，MATLAB 在数学建模中的应用，北京：北京航空航天大学，79-176，2014。

[6] 袁利平，夏洁，陈宗基，多无人机协同路径规划研究综述，飞行力学，27（5）：1-5，2010。

[7] 王振华，章卫国，李广文，基于改进多目标蚁群算法的无人机路径规划，计算机应用研究，26（6）：2105-2106，2009。

7 附录

附录为本文构造数学模型时所用的代码：

粒子群算法求解问题代码如下：

```
%-------------------------------------------------------------------------%

clear
clc
t0 = clock;


% P01=xlsread('data.xlsx','E70:F72');
% P01=[72.3 82.3 85.6 82.3 85.6 85.6 83.4 85.6;
%      64.7 66 71.2 67.3 58.2 60.8 59.5 55.6]';
P01=[1 1 1 2 2 3 4 4 5 6 7 7 9 9 10 11 13 13;
     4 10 16 2 14 7  4 10 16 13 11 8 15 5 11 16 14 11]';
%-------------------------------------------------------------------------%
n=size(P01,1);
D=zeros(n,n);
for i=1:n
    for j=1:n
        if i~=j
            D(i,j)=sqrt( sum( ( P01(i,:)-P01(j,:) ).^2 ) );
        else
            D(i,j)=1e-4;
        end
    end
end

%-------------------------------------------------------------------------%
%%初始化参数
m = 100;                                % 蚂蚁数量
alpha = 1;                              % 信息素重要程度因子
beta = 5;                               % 启发函数重要程度因子
vol = 0.2;                              % 信息素挥发(volatilization)因子
Q = 10;                                 % 常系数
Heu_F = 1./D;                           % 启发函数(heuristic function)
Tau = ones(n,n);                        % 信息素矩阵
Table = zeros(m,n);                     % 路径记录表
iter = 1;                               % 迭代次数初值
iter_max = 200;                         % 最大迭代次数
Route_best = zeros(iter_max,n);         % 各代最佳路径
Length_best = zeros(iter_max,1);        % 各代最佳路径长度
Length_ave = zeros(iter_max,1);         % 各代路径的平均长度
Limit_iter = 0;                         % 程序收敛时迭代次数
%-------------------------------------------------------------------------
%% 迭代寻找最佳路径
while iter <= iter_max
    % 随机产生各个蚂蚁的起点目标
      start = zeros(m,1);
      for i = 1:m
          temp = randperm(n);
          start(i) = temp(1);
      end
      Table(:,1) = start;
      %构建解空间
      PO1_index = 1:n;
      %逐个蚂蚁路径选择
      for i = 1:m
          % 逐个城市路径选择
         for j = 2:n
             tabu = Table(i,1:(j - 1));          % 已访问的目标集合
```

```
            allow_index = ~ismember(PO1_index,tabu);
            allow = PO1_index(allow_index);  % 待访问的城市集合
            P = allow;
            % 计算城市间转移概率
            for k = 1:length(allow)
                P(k) = Tau(tabu(end),allow(k))^alpha *
Heu_F(tabu(end),allow(k))^beta;
            end
            P = P/sum(P);
            % 轮盘赌法选择下一个访问目标
           Pc = cumsum(P);
           target_index = find(Pc >= rand);
           target = allow(target_index(1));
           Table(i,j) = target;
        end
    end
    % 计算各个蚂蚁的路径距离
    Length = zeros(m,1);
    for i = 1:m
        Route = Table(i,:);
        for j = 1:(n - 1)
            Length(i) = Length(i) + D(Route(j),Route(j + 1));
        end
        Length(i) = Length(i) + D(Route(n),Route(1));
    end
    %计算最短路径距离及平均距离
    if iter == 1
        [min_Length,min_index] = min(Length);
        Length_best(iter) = min_Length;
        Length_ave(iter) = mean(Length);
        Route_best(iter,:) = Table(min_index,:);
        Limit_iter = 1;

    else
        [min_Length,min_index] = min(Length);
        Length_best(iter) = min(Length_best(iter - 1),min_Length);
        Length_ave(iter) = mean(Length);
        if Length_best(iter) == min_Length
            Route_best(iter,:) = Table(min_index,:);
            Limit_iter = iter;
        else
            Route_best(iter,:) = Route_best((iter-1),:);
        end
    end
    % 更新信息素
    Delta_Tau = zeros(n,n);
    % 逐个目标计算
    for i = 1:m
        % Öð¸öÄ¿±ê¼ÆËã
        for j = 1:(n - 1)
            Delta_Tau(Table(i,j),Table(i,j+1)) =
Delta_Tau(Table(i,j),Table(i,j+1)) + Q/Length(i);
        end
        Delta_Tau(Table(i,n),Table(i,1)) = Delta_Tau(Table(i,n),Table(i,1))
+ Q/Length(i);
    end
    Tau = (1-vol) * Tau + Delta_Tau;
   % 迭代次数加 1，情况路径记录
   iter = iter + 1;
   Table = zeros(m,n);
end
%-------------------------------------------------------------------------
```

```
%%结果显示
[Shortest_Length,index] = min(Length_best);
Shortest_Route = Route_best(index,:);
Time_Cost=etime(clock,t0);
disp(['最短距离:' num2str(Shortest_Length)]);
disp(['最短路径:' num2str([Shortest_Route Shortest_Route(1)])]);
disp(['收敛时迭代次数:' num2str(Limit_iter)]);
disp(['程序执行时间:' num2str(Time_Cost) 'Ãë']);
%--------------------------------------------------------------------------

plot([P01(Shortest_Route,1);P01(Shortest_Route(1),1)],...
     [P01(Shortest_Route,2);P01(Shortest_Route(1),2)],'o-');
grid on
for i = 1:size(P01,1)
    text(P01(i,1),P01(i,2),['   ' num2str(i)]);
end
text(P01(Shortest_Route(1),1),P01(Shortest_Route(1),2),'       起点');
text(P01(Shortest_Route(end),1),P01(Shortest_Route(end),2),'       终点');
xlabel('通信基站横坐标')
ylabel('通信基站纵坐标')
title(['遍历所有点的最优化路径（最短距离）:' num2str(Shortest_Length) 'km)'])
figure(2)
plot(1:iter_max,Length_best,'b')
legend('最短距离')
xlabel('迭代次数')
ylabel('距离')
title('算法收敛轨迹')


%--------------------------------------------------------------------------
```

遗传算法求解问题程序代码如下

```
function varargout =
A_Mtspv_GA(xy,dmat,minTour,popSize,numIter,showProg,showResult)

% Process Inputs and Initialize Defaults
nargs = 7;
for k = nargin:nargs-1
    switch k
        case 0
%            xy = [30.3  57.9 110;
%                  89.8  47.6   0]';
             xy = [ 66.0 98.4 73.7  110;
                    84.7 76.7 61.0    0]';
        case 1
            N = size(xy,1);
            a = meshgrid(1:N);
            dmat = reshape(sqrt(sum((xy(a,:)-xy(a',:)).^2,2)),N,N);
        case 2
            minTour = 4;
        case 3
            popSize = 80;
        case 4
            numIter = 5e3;
        case 5
            showProg = 1;
        case 6
            showResult = 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;

% Sanity Checks
minTour = max(1,min(n,round(real(minTour(1)))));
popSize = max(8,8*ceil(popSize(1)/8));
numIter = max(1,round(real(numIter(1))));
showProg = logical(showProg(1));
showResult = logical(showResult(1));

% Initialize the Populations
popRoute = zeros(popSize,n);    % population of routes
popBreak = cell(popSize,1);     % population of breaks
for k = 1:popSize
    popRoute(k,:) = randperm(n);
    popBreak{k} = rand_breaks();
end

% Select the Colors for the Plotted Routes
pclr = ~get(0,'DefaultAxesColor');
clr = hsv(floor(n/minTour));

% Run the GA
globalMin = Inf;
totalDist = zeros(1,popSize);
distHistory = zeros(1,numIter);
tmpPopRoute = zeros(8,n);
tmpPopBreak = cell(8,1);
newPopRoute = zeros(popSize,n);
newPopBreak = cell(popSize,1);
if showProg
    pfig = figure('Name',遗传算法（GA） | ÊµÊ±×îÓÅ½â','Numbertitle','off');
end
for iter = 1:numIter
    % Evaluate Each Population Member (Calculate Total Distance)
    for p = 1:popSize
        d = 0;
        pRoute = popRoute(p,:);
        pBreak = popBreak{p};
        nSalesmen = length(pBreak)+1;
        rng = [[1 pBreak+1];[pBreak n]]';
        for s = 1:nSalesmen
            d = d + dmat(pRoute(rng(s,2)),pRoute(rng(s,1)));
            for k = rng(s,1):rng(s,2)-1
                d = d + dmat(pRoute(k),pRoute(k+1));
            end
        end
        totalDist(p) = d;
    end

    % Find the Best Route in the Population
    [minDist,index] = min(totalDist);
    distHistory(iter) = minDist;
    if minDist < globalMin
        globalMin = minDist;
```

```
        optRoute = popRoute(index,:);
        optBreak = popBreak{index};
        nSalesmen = length(optBreak)+1;
        rng = [[1 optBreak+1];[optBreak n]]';
        if showProg
            % Plot the Best Route
            figure(pfig);
            for s = 1:nSalesmen
                rte = optRoute([rng(s,1):rng(s,2) rng(s,1)]);
                if dims > 2,
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(['ÔÖÇøÑ²²é×Üº½³Ì = %1.4f KM, ÎÞÈË»úÊýÁ¿ = %d ¼Ü,
' ...
                    'µü´ú = %d'],minDist,nSalesmen,iter));
                hold on
            end
            hold off
        end
    end


text(26.300,89.100,'A');
text(62.000,84.700,'B');
text(94.400,76.700,'C');
text(70.700,61.000,'D');
text(54.900,47.600,'E');
text(110.000,0,'H');


    % Genetic Algorithm Operators
    randomOrder = randperm(popSize);
    for p = 8:8:popSize
        rtes = popRoute(randomOrder(p-7:p),:);
        brks = popBreak(randomOrder(p-7:p));
        dists = totalDist(randomOrder(p-7:p));
        [ignore,idx] = min(dists); %#ok
        bestOf8Route = rtes(idx,:);
        bestOf8Break = brks{idx};
        routeInsertionPoints = sort(ceil(n*rand(1,2)));
        I = routeInsertionPoints(1);
        J = routeInsertionPoints(2);
        for k = 1:8 % Generate New Solutions
            tmpPopRoute(k,:) = bestOf8Route;
            tmpPopBreak{k} = bestOf8Break;
            switch k
                case 2 % Flip
                    tmpPopRoute(k,I:J) = tmpPopRoute(k,J:-1:I);
                case 3 % Swap
                    tmpPopRoute(k,[I J]) = tmpPopRoute(k,[J I]);
                case 4 % Slide
                    tmpPopRoute(k,I:J) = tmpPopRoute(k,[I+1:J I]);
                case 5 % Change Breaks
                    tmpPopBreak{k} = rand_breaks();
                case 6 % Flip, Change Breaks
                    tmpPopRoute(k,I:J) = tmpPopRoute(k,J:-1:I);
                    tmpPopBreak{k} = rand_breaks();
                case 7 % Swap, Change Breaks
                    tmpPopRoute(k,[I J]) = tmpPopRoute(k,[J I]);
                    tmpPopBreak{k} = rand_breaks();
                case 8 % Slide, Change Breaks
                    tmpPopRoute(k,I:J) = tmpPopRoute(k,[I+1:J I]);
```

```
                    tmpPopBreak{k} = rand_breaks();
                otherwise % Do Nothing
            end
        end
        newPopRoute(p-7:p,:) = tmpPopRoute;
        newPopBreak(p-7:p) = tmpPopBreak;
    end
    popRoute = newPopRoute;
    popBreak = newPopBreak;
end

if showResult
    % Plots
    figure('Name','ÒÅ´«Ëã·¨£¨GA£© | ×îÖÕ¼á¹û','Numbertitle','off');
    subplot(2,2,1);
    if dims > 2, plot3(xy(:,1),xy(:,2),xy(:,3),'.','Color',pclr);
    else plot(xy(:,1),xy(:,2),'.','Color',pclr); end

text(26.300,89.100,'A');
text(62.000,84.700,'B');
text(94.400,76.700,'C');
text(70.700,61.000,'D');
text(54.900,47.600,'E');
text(110.000,0,'H');

    title('ÖØµãÇøÓòÎ»ÖÃ×ø±ê');
    subplot(2,2,2);
    imagesc(dmat(optRoute,optRoute));
    title('¾àÀë¾ØÕó');
    nSalesmen = length(optBreak)+1;
    subplot(2,2,3);
    rng = [[1 optBreak+1];[optBreak n]]';
    for s = 1:nSalesmen
        rte = optRoute([rng(s,1):rng(s,2) rng(s,1)]);
        if dims > 2,
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
        text(26.300,89.100,'A');
        text(62.000,84.700,'B');
        text(94.400,76.700,'C');
        text(70.700,61.000,'D');
        text(54.900,47.600,'E');
        text(110.000,0,'H');
        title(sprintf('ÔÖÇéÑ²²é×Üº½³Ì = %1.4f KM',minDist));
        hold on;
    end
    subplot(2,2,4);
    plot(distHistory,'b','LineWidth',2)
    title('ÀúÊ·×îÓÅ¼â');
    set(gca,'XLim',[0 numIter+1],'YLim',[0 1.1*max([1 distHistory])]);
end



% Return Outputs
if nargout
    varargout{1} = optRoute;
    varargout{2} = optBreak;
    varargout{3} = minDist;
end

    % Generate Random Set of Breaks
    function breaks = rand_breaks()
```

```
        nSalesmen = ceil(floor(n/minTour)*rand);
        nBreaks = nSalesmen - 1;
        dof = n - minTour*nSalesmen;    % degrees of freedom
        addto = ones(1,dof+1);
        for kk = 2:nBreaks
            addto = cumsum(addto);
        end
        cumProb = cumsum(addto)/sum(addto);
        nAdjust = find(rand < cumProb,1)-1;
        spaces = ceil(nBreaks*rand(1,nAdjust));
        adjust = zeros(1,nBreaks);
        for kk = 1:nBreaks
            adjust(kk) = sum(spaces == kk);
        end
        breaks = minTour*(1:nBreaks) + cumsum(adjust);
    end
end
```

绘制第一题地形和最优路径代码如下：

```
clc;
clear;

% x=0:38.2:105966.8;y=0:38.2:111238.4;
% z1=xlsread('data.xlsx');
% z=z1';
% zz=ones(2913,2775)*4150;
% % subplot(2,2,1),mesh(x,y,z);
% mesh(x,y,z);
% hold on;
%
mesh(x,y,zz,'EdgeColor','w','FaceColor','w','MarkerEdgecolor','w','MarkerFa
cecolor','w');
% colormap(jet);
% hold on;
%
% xlabel('X(µ¥Î»£º10km)');
% ylabel('Y(µ¥Î»£º10km)');
% zlabel('Z(µ¥Î»£ºm)');%ÉèÖÃXÖáµÄÃû³ÆÏÔÊ¾
% % x3=[100000 80000 80000  66000 86000];
% % y3=[27000  38000 50000  58000 62000];
% % title({'ÔÖÇø4150Ã×ÒÔÉÏµØÐÎÊ¾ÒâÍ¼';'(ºìÉ«ÇøÓòÎÞÈË»úÎÞ·¨Í¨¹ý)'});
%
title({'Èý¼ÜÎÞÈË»ú±éÀúËùÓÐÔÖÇéÖØµãÇøÓòÖÐÐÄÎ»ÖÃµÄ×îÓÅÐÞÕýÂ·¾¶Î¬Í¼';'(Â·¾¶Î
ª£ºH12A3¡¢H456789 10 11 B ºÍ H 4 5 6 12 13 14)';'(ºìÉ«ÇøÓòÎª4150Ã×ÒÔÉÏÇøÓò
ÎÞÈË»úÐèÒªÈÆÐÐ)'});
% % %
title({'Ò»¼ÜÎÞÈË»ú±éÀúËùÓÐÔÖÇéÖØµãÇøÓòÖÐÐÄÎ»ÖÃµÄ×îÓÅÂ·¾¶';'(Â·¾¶Îª£ºHEABDC)
';'(ºìÉ«ÇøÓòÎª4200É¼·å ÎÞÈË»úÎÞ·¨Í¨¹ý)'});
% view([0,90]);
% % view([-45,45]);
% hold on;


%%


% subplot(2,2,2),contour(x,y,z,8);
% contour(x,y,z,8);
% xlabel('X(µ¥Î»£º10km)');%ÉèÖÃXÖáµÄÃû³ÆÏÔÊ¾
```

```
% ylabel('Y(µ¥Î»£º10km)');%ÉèÖÃXÖáµÄÃû³ÆÏÔÊ¾
% title ÔÖÇøµØÐÎµÈ¸ßÍ¼;%ÉèÖÃ±êÌâ
%  view([-45,45]);
% hold on;
% subplot(2,2,3),contour3(x,y,z,8);
% contour3(x,y,z,8);
% xlabel('X(µ¥Î»£º10km)');%ÉèÖÃXÖáµÄÃû³ÆÏÔÊ¾
% ylabel('Y(µ¥Î»£º10km)');%ÉèÖÃXÖáµÄÃû³ÆÏÔÊ¾
% zlabel('Z(µ¥Î»£ºkm)');%ÉèÖÃXÖáµÄÃû³ÆÏÔÊ¾
% title ÔÖÇøÈýÎ¬µÈ¸ßÍ¼;%ÉèÖÃ±êÌâ
% view([-45,45]);
% hold on;
% subplot(2,2,4),contourf(x,y,z,8);
% contourf(x,y,z,8);
% xlabel('X(µ¥Î»£º10km)');%ÉèÖÃXÖáµÄÃû³ÆÏÔÊ¾
% ylabel('Y(µ¥Î»£º10km)');%ÉèÖÃXÖáµÄÃû³ÆÏÔÊ¾
% title ÔÖÇøÉ«²ÊÌî³äµÈ¸ßÍ¼;%ÉèÖÃ±êÌâ
% hold on;


%%
%%¡ª¡ª-------------------------ÕðÇø7¸öÖØµãÇøÓò×ø±ê¼°·¶Î§-----------------
-------------------------%%
%
rectangle('Position',[25300,84800,10000,10000],'Curvature',[1,1],'facecolor
','w'),%ÖÐÐÄµãA
%
rectangle('Position',[61000,79700,10000,10000],'Curvature',[1,1],'facecolor
','w'),%ÖÐÐÄµãB
%
rectangle('Position',[93400,71700,10000,10000],'Curvature',[1,1],'facecolor
','w'),%ÖÐÐÄµãC
%
rectangle('Position',[68700,56000,10000,10000],'Curvature',[1,1],'facecolor
','w'),%ÖÐÐÄµãD
%
rectangle('Position',[52900,42600,10000,10000],'Curvature',[1,1],'facecolor
','w'),%ÖÐÐÄµãE
%
rectangle('Position',[81800,17000,10000,10000],'Curvature',[1,1],'facecolor
','w'),%ÖÐÐÄµãF
%
rectangle('Position',[88600,43800,10000,10000],'Curvature',[1,1],'facecolor
','w'),%ÖÐÐÄµãG
h=5000; % ¸ß¶È
t1=0:0.1:(2*pi);
t1=[t1,0];
plot3(30300+5000*sin(t1),89800+5000*cos(t1), h*ones(size(t1)),'k');
hold on;
plot3(66000+5000*sin(t1),84700+5000*cos(t1), h*ones(size(t1)),'k');
plot3(98400+5000*sin(t1),76700+5000*cos(t1), h*ones(size(t1)),'k');
plot3(73700+5000*sin(t1),61000+5000*cos(t1), h*ones(size(t1)),'k');
plot3(57900+5000*sin(t1),47600+5000*cos(t1), h*ones(size(t1)),'k');
% plot3(86800+5000*sin(t),22000+5000*cos(t), h*ones(size(t)),'k');
% plot3(93600+5000*sin(t),48800+5000*cos(t), h*ones(size(t)),'k');
hold on;




%%
¡ª¡ª-------------------------ÕðÇø7¸öÖØµãÇøÓò×ø±ê¼°·¶Î§-------------------
------------------------%%
```

```
%% ËùÓÐµãµÄ×ø±ê
% x1=[30300 66000 98400 73700 57900 86800 93600];
% y1=[89800 84700 76700 61000 47600 22000 48800];
% z2=ones(7,7)*4200;
% plot3(x1,y1,z2,'bO','MarkerEdgecolor','b','MarkerFacecolor','b');
% hold on;
%
% x2=110000;
% y2=0    ;
% z3=ones(1,1)*4200;
% plot3(x2,y2,z3,'rO','MarkerEdgecolor','r','MarkerFacecolor','r');
% hold on;
%%  Ò»¼ÜÎÞÈË»úÂ·¾¶Ê¾ÒâÍ¼£¨HEABDC£©
% % x1=[30300 66000 98400 73700 57900 86800 93600];
% % y1=[89800 84700 76700 61000 47600 22000 48800];

% x2=[110000 57900 30300 66000 73700 98400];
% y2=[0     47600 89800 84700 61000 76700];
% [X,Y]=meshgrid(x2,y2);
% % z2=ones(3,3)*4200;
% z2=ones(6,6)*5000;
% plot3(x2,y2,z2,'-bO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;


%%  Á½¼ÜÎÞÈË»úÂ·¾¶Ê¾ÒâÍ¼£¨HEAºÍHDBC£©

% x2=[110000 57900 30300];
% y2=[0     47600 89800];
% [X,Y]=meshgrid(x2,y2);
% % z2=ones(3,3)*4200;
% z2=ones(3,3)*5000;
% plot3(x2,y2,z2,'-bO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;
%
% x3=[110000 73700 66000 98400];
% y3=[0     61000 84700 76700];
% [X1,Y1]=meshgrid(x3,y3);
% z3=ones(4,4)*5000;
% % z3=ones(7,7)*4200;
% plot3(x3,y3,z3,'-kO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;



%%  Èý¼ÜÎÞÈË»úÂ·¾¶Ê¾ÒâÍ¼(HEA¡¢HDBºÍHC)


% x2=[110000 57900 30300];
% y2=[0     47600 89800];
% [X,Y]=meshgrid(x2,y2);
% % z2=ones(3,3)*4200;
% z2=ones(3,3)*5000;
% plot3(x2,y2,z2,'-bO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;
%
% x3=[110000 73700  66000];
% y3=[0     61000  84700];
% [X1,Y1]=meshgrid(x3,y3);
% z3=ones(3,3)*5000;
% % z3=ones(7,7)*4200;
% plot3(x3,y3,z3,'-kO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;
```

```
%
% x5=[110000 98400];
% y5=[0     76700];
% [X2,Y2]=meshgrid(x5,y5);
% z5=ones(2,2)*5000;
% % z5=ones(6,6)*4200;
% plot3(x5,y5,z5,'-rO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;

%%  Îå¼ÜÎÞÈË»úÂ·¾¶Ê¾ÒâÍ¼£¨HA¡¢HB¡¢HC¡¢HDºÍHE£©

% x2=[110000 30300];
% y2=[0     89800];
% [X,Y]=meshgrid(x2,y2);
% z2=ones(2,2)*4200;
% % z2=ones(2,2)*5000;
% plot3(x2,y2,z2,'-cO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;
%
% x3=[110000 66000];
% y3=[0     84700];
% [X1,Y1]=meshgrid(x3,y3);
% % z3=ones(2,2)*5000;
% z3=ones(2,2)*4200;
% plot3(x3,y3,z3,'-bO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;
%
% x5=[110000 98400];
% y5=[0     76700];
% [X2,Y2]=meshgrid(x5,y5);
% % z5=ones(2,2)*5000;
% z5=ones(2,2)*4200;
% plot3(x5,y5,z5,'-kO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;
%
% x6=[110000 73700];
% y6=[0     61000];
% [X3,Y3]=meshgrid(x6,y6);
% % z6=ones(2,2)*5000;
% z6=ones(2,2)*4200;
% plot3(x6,y6,z6,'-rO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;
%
% x7=[110000 57900];
% y7=[0     47600];
% [X4,Y4]=meshgrid(x7,y7);
% % z7=ones(2,2)*5000;
% z7=ones(2,2)*4200;
% plot3(x7,y7,z7,'-mO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;




%% ×îÓÅÂ·¾¶¹æ»®£¨½ûÖ¹É¾³ý£©

% % x1=[30300 66000 98400 73700 57900 86800 93600];
% % y1=[89800 84700 76700 61000 47600 22000 48800];
% x2=[110000 57900 30300];
% y2=[0     47600 89800];
% [X,Y]=meshgrid(x2,y2);
% % z2=ones(3,3)*4200;
% z2=ones(3,3)*5000;
```

```
% plot3(x2,y2,z2,'-bO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;
%
% x3=[110000 100000 80000 80000  73700 66000 66000];
% y3=[0     27000  38000 50000  61000 58000 84700];
% [X1,Y1]=meshgrid(x3,y3);
% z3=ones(7,7)*5000;
% % z3=ones(7,7)*4200;
% plot3(x3,y3,z3,'-kO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;
%
% x5=[110000 100000 80000 80000 86000 98400];
% y5=[0     27000  38000 50000 62000 76700];
% [X2,Y2]=meshgrid(x5,y5);
% z5=ones(6,6)*5000;
% % z5=ones(6,6)*4200;
% plot3(x5,y5,z5,'-cO','MarkerEdgecolor','k','MarkerFacecolor','k');
% hold on;

%% Ñ²²é3000Ã×ÒÔÏÂÇøÓòÂ·¾¶
x1=[30300 66000 98400 73700 57900 86800 93600];
y1=[89800 84700 76700 61000 47600 22000 48800];
x2=[110000 59670];
y2=[0     45830];

[X,Y]=meshgrid(x2,y2);
% z2=ones(3,3)*4200;
z2=ones(2,2)*5000;
plot3(x2,y2,z2,'-rO','MarkerEdgecolor','k','MarkerFacecolor','k');
hold on;

x3=[56650 30300 26765];
y3=[49765 89800 93335];
[X1,Y1]=meshgrid(x3,y3);
% z2=ones(3,3)*4200;
z3=ones(3,3)*5000;
plot3(x3,y3,z3,'-rO','MarkerEdgecolor','k','MarkerFacecolor','k');
hold on;




x4=[110000 100000 80000 80000  73700+1250];
y4=[0     27000  38000 50000  61000-1250*1.732];
[X3,Y3]=meshgrid(x4,y4);
z4=ones(5,5)*5000;
% z4=ones(7,7)*4200; 66000   58000
plot3(x4,y4,z4,'-kO','MarkerEdgecolor','k','MarkerFacecolor','k');
hold on;







x5=[73700-1250*1.732 66000 66000];
y5=[61000-1250      58000 82200];
[X4,Y4]=meshgrid(x5,y5);
z5=ones(3,3)*5000;
% z4=ones(7,7)*4200; 66000   58000
plot3(x5,y5,z5,'-kO','MarkerEdgecolor','k','MarkerFacecolor','k');
plot3(67768,82932,5000,'-kO','MarkerEdgecolor','k','MarkerFacecolor','k');
```

```
hold on;

text(63000,81000,4500,'10');
text(68768,82932,4500,'11');

x6=[110000 100000 80000 80000 86000 98400-1767];
y6=[0      27000  38000 50000 62000 76700-1767];
[X5,Y5]=meshgrid(x6,y6);
z6=ones(6,6)*5000;
% z6=ones(6,6)*4200;
plot3(x6,y6,z6,'-mO','MarkerEdgecolor','k','MarkerFacecolor','k');
plot3(99650,78865,5000,'-mO','MarkerEdgecolor','k','MarkerFacecolor','k');
hold on;




x7=[30300 66000 98400 73700 57900];
y7=[89800 84700 76700 61000 47600];
[X6,Y6]=meshgrid(x7,y7);
z7=ones(5,5)*5000;
% z6=ones(6,6)*4200;
plot3(x7,y7,z7,'rO','MarkerEdgecolor','r','MarkerFacecolor','r');
hold on;



%% »-Ô²»¡
h=5000; % ¸ß¶È
t1=(3/4*pi):0.1:(2*pi);
t2=(0.7*pi):0.1:(1.5*pi);
t3=(1.2*pi):0.1:(2.2*pi);
t4=(1*pi):0.1:(2.8*pi);
% plot3(30300+5000*sin(t),89800+5000*cos(t), h*ones(size(t)),'k');
plot3(66000+2500*sin(t4),84700+2500*cos(t4), h*ones(size(t4)),'k');
plot3(98400+2500*sin(t3),76700+2500*cos(t3), h*ones(size(t3)),'m');
plot3(73700+2500*sin(t2),61000+2500*cos(t2), h*ones(size(t2)),'k');
plot3(57900+2500*sin(t1),47600+2500*cos(t1), h*ones(size(t1)),'r');
% plot3(86800+5000*sin(t),22000+5000*cos(t), h*ones(size(t)),'k');
% plot3(93600+5000*sin(t),48800+5000*cos(t), h*ones(size(t)),'k');
hold on;


%% A-EµãµÄ±ê¼Ç
text(28300,88000,4500,'A');
text(64000,87000,4500,'B');
text(99900,76000,4500,'C');
text(72700,63000,4500,'D');
text(58900,48600,4500,'E');
% text(83800,22000,5000,'F');
% text(90600,48800,5000,'G');
text(110000,3000,4500,'H');

text(57670,43830,4500,'1');
text(56650,52700,4500,'2');
text(26765,97335,4500,'3');
text(99000,31000,4500,'4');
text(77000,41000,4500,'5');
text(77000,51000,4500,'6');
text(76700,60000,4500,'7');
text(70000,62000,4500,'8');
text(63000,60000,4500,'9');
```

```
text(82000,65000,4500,'12');
text(93400,76700,4500,'13');
text(99650,80865,4500,'14');
% x3=[100000 80000 80000  66000 86000];1¡¢2¡¢3¡¢4¡¢5µÄ×ø±ê
% y3=[27000  38000 50000  58000 62000];



view([0,90]);
```