# 设计说明书

目录

[设计说明书 1](#_Toc137668675)

[一、 系统需求分析 1](#_Toc137668676)

[二、 总体设计 1](#_Toc137668677)

[三、 详细设计 2](#_Toc137668678)

[四、 执行流程 3](#_Toc137668679)

[五、 结果分析 3](#_Toc137668680)

## 系统需求分析

在本项目中，我们的目标是实现一个寻找最短路径的算法，并在图形界面上显示出障碍物和最终路径。系统需要满足以下需求：

* 随机产生二维障碍物环境，包括路径随机起点、终点，以及障碍物和边界信息（形状、个数、大小、位置都随机，要避免太简单环境）。
* 设计算法，实现从起点到终点的最短路径的求解。
* 输入输出障碍物环境信息、最短路信息。

## 总体设计

整个系统由四个类构成：Point、Map、Findpath和Show。

* Point类：表示二维平面上的点，具有x和y两个坐标属性。
* Map类：表示地图，包括障碍物点集合obstacles和标记是否为障碍物的哈希表is\_obstacles，以及地图的宽度w和高度h。Map类还有P2I方法用于将点转换为索引，GetRandObstacles方法用于随机生成障碍物，GetIs\_obstacles方法用于将障碍物加入到is\_obstacles集合中。
* Findpath类：表示路径查找类，包括地图的宽度w和高度h，以及一个Map对象mymap。Findpath类有pass\_path属性表示经过的路径集合，final\_path属性表示最终路径。该类还包含Find方法用于寻找最短路径，Isxline、Isyline和Isslash三个方法分别用于判断是否可以向右、向下、斜向移动。
* Show类：表示展示类，包括地图的宽度w和高度h。Show类有Show\_back方法用于显示背景网格，Show\_obsandline方法用于显示障碍物和路径，Show\_all方法用于显示全部内容。

## 详细设计

* **Point类:**
* **数据成员:**

x（横坐标）

y（纵坐标）

* **成员函数:**

构造函数：用于初始化Point对象的坐标。

* **Map类:**
* **数据成员:**

obstacles（障碍物点集合）：用vector<Point>存储障碍物点的坐标。

is\_obstacles（标记是否为障碍物的哈希表）：使用unordered\_set<int>存储障碍物点的索引。

w（地图宽度）

h（地图高度）

* **成员函数:**

P2I(Point p)：将Point对象转换为索引，使用公式key = p.y \* w + p.x。

GetRandObstacles()：根据地图尺寸，在合适的范围内生成随机的障碍物坐标，并存储到obstacles中。

GetIs\_obstacles()：遍历obstacles，将障碍物点的索引存储到is\_obstacles中。

* **Findpath类:**
* **数据成员:**

w（地图宽度）

h（地图高度）

mymap（Map对象）

pass\_path（经过的路径集合）：使用unordered\_set<int>存储已经经过的路径点的索引。

final\_path（最终路径）：用vector<Point>存储最短路径上的点坐标。

* **成员函数:**

Find()：在地图上寻找从起点(0,0)到终点(w-1,h-1)的最短路径。使用广度优先搜索算法进行遍历，通过判断是否可以向右移动、向下移动或斜向移动来确定下一个点，直到找到终点或无法继续搜索。

Isxline(Point now)：判断当前点是否可以向右移动，即判断右侧的点是否为障碍物并且未经过。

Isyline(Point now)：判断当前点是否可以向下移动，即判断下方的点是否为障碍物并且未经过。

Isslash(Point now)：判断当前点是否可以斜向移动，即判断右下方的点是否为障碍物并且未经过。

* **Show类:**
* **数据成员:**

w（地图宽度）

h（地图高度）

* **成员函数:**

Show\_back()：使用图形库初始化画布，并绘制背景网格和边框。

Show\_obsandline()：创建Findpath对象，调用Find方法获取最短路径，并在画布上绘制障碍物和路径。

Show\_all()：调用Show\_back()和Show\_obsandline()，显示全部内容。

## 执行流程

* 首先创建一个Show对象test，并传入地图的宽度和高度（这里是20x20）。
* 调用Show对象的Show\_all方法。
* 在Show\_all方法中，首先调用Show\_back方法来绘制背景网格。initgraph函数用于初始化图形窗口，设置窗口大小为地图宽度乘以30和地图高度乘以30。setbkcolor函数设置背景颜色为白色，cleardevice函数清空窗口内容。接着使用setcolor和rectangle函数绘制一个黑色的矩形边框作为地图的边界。利用setlinestyle函数设置绘制网格线的样式，然后使用for循环绘制水平和垂直的网格线。
* 接下来调用Show\_obsandline方法来显示障碍物和路径。首先创建一个Findpath对象find，传入地图的宽度和高度。然后调用find的Find方法来寻找最短路径。
* Find方法中，首先调用mymap的GetRandObstacles方法来随机生成一些障碍物点，并将它们存储在obstacles集合中。然后调用mymap的GetIs\_obstacles方法将障碍物点加入到is\_obstacles哈希表中。
* **在while循环中，通过回溯法遍历所有可能的路径。首先将当前路径存储在temp\_path中，然后清空path集合。接着遍历temp\_path中的每条路径，获取最后一个点now。根据Isxline、Isyline和Isslash方法判断是否可以向右、向下或斜向移动，并将新的点加入到当前路径中。同时将该点的索引添加到pass\_path集合中。**
* **在遍历完所有路径后，检查路径的最后一个点是否为终点（即坐标为(w-1, h-1)）。如果是，则将该路径设为最终路径并跳出循环。**
* 最后，调用Show类的Show\_obsandline方法来绘制障碍物和路径。使用setfillcolor函数设置障碍物的填充颜色为黑色，然后利用fillrectangle函数绘制每个障碍物点的矩形。使用setcolor函数设置路径的颜色为红色，并利用line函数连接路径上的每两个相邻点。
* Show\_all方法中的system("pause")暂停程序，等待用户点击任意键继续执行。然后调用closegraph函数关闭图形窗口。

## 结果分析

结果分析，实现了以下功能：

* 随机二维障碍物环境的产生、显示。
* 从起点到终点的最短路径的求解。
* 最短路径的显示。
* 输入输出障碍物环境信息、最短路信息。