

实验：梯式 GIS 软件工程实践—小型桌面图形编辑系统

目录

- 1.需求分析 2
 - 1.1 题目要求 2
 - 1.2 需求分析 2
- 2.系统设计 2
 - 2.1 概要设计 2
 - 2.2 详细设计 3
- 3.软件开发 10
- 4.软件调试与测试 10
- 5.特点与不足 13
 - 5.1 技术特点 13
 - 5.2 不足和改进的建议 13
- 6.过程和体会 13
 - 6.1 遇到的主要问题和解决方法 13
 - 6.2 程序设计的体会 14
- 7.源码和说明 14
 - 7.1 文件清单及其功能说明 14
 - 7.2 用户使用说明书 14
 - 7.3 源代码 15

1.需求分析

1.1 题目要求

开发点、线、区等图形的编辑、存储、查询和显示等功能，实现一个小型桌面图形编辑系统。实现的功能具体如下：

- (1) 与文件相关：新建临时文件，保存点文件，保存线文件，保存区文件，打开点文件，打开线文件，打开区文件，另存线文件，另存线文件，另存区文件，退出。
- (2) 与窗口有关：放大，缩小，复位，移动，显示点，显示线，显示区。
- (3) 与点相关操作：造点，删除点，移动点，恢复点，显示删除点，修改点参数，修改点缺省参数。
- (4) 与线相关操作：造线，删除线，移动线，恢复线，显示删除线，连接线，线上加点，线上删点，修改线参数，修改线缺省参数。
- (5) 与区相关操作：造区，删除区，移动区，恢复区，显示删除区，修改区参数，修改区缺省参数。

1.2 需求分析

根据当前操作状态，显示状态等全局变量和鼠标左键按下，鼠标左键弹起，鼠标移动，鼠标右键弹起等指令确定当前的操作类型，并实现相应的功能。

2.系统设计

2.1 概要设计

(1) 定义点，线，区，节点，文件版本等结构体。其中点，线，区结构体需包括颜色，形状等内容。定义在 `MyDataType.h` 中。

(2) 定义相关的全局变量，如物理数，逻辑数，临时文件名，永久文件名，文件创建标志，读取临时文件的指针对象，当前操作状态等。

(3) 定义不同文件的读写函数到 `WriteOrRead.h` 中，实现文件的读取，写入，更新等。

(4) 定义不同图形的绘制函数到 `Paint.h` 中，实现不同图案的绘制。

(5) 有关计算的函数定义在 `Calculate.h` 中，实现查找点，查找线，查找区，计算距离等操作。

(6) 创建文件，修改点参数，修改线参数，修改线参数时需要弹出窗口，因此需要创建相应的窗口类。

(7) 为菜单的各个选项添加事件处理函数到 CMapEditorView 类中, 在各事件处理函数设置当前的操作状态及完成有关操作。

(8) 在左键按下, 左键弹起, 右键弹起, 鼠标移动等函数中根据当前操作状态及相关全局变量的情况实现有关的创建, 删除, 移动, 恢复等操作。

2.2 详细设计

(1) 定义结构体。

①点结构体 **PNT_STRU** 应包括的内容:

点型, 点颜色, 点的横纵坐标, 删除标志。

点型有: 十字型, 圆点型, 星型。

②线结构体 **LIN_NDX_STRU** 应包括:

线型, 线颜色, 线节点数, 线节点数据储存位置, 线删除标志。

线型有实线, 虚线, 点线。

③区结构体 **REG_NDX_STRU** 应包括:

区型, 区颜色, 区边界节点数, 区边界节点储存位置, 区删除标志。

④节点结构体 **D_DOT** 包括: 节点横纵坐标。

⑤文件版本结构体 **VERSION** 包括:

一个长度为 3 的 char 类型的一维数组 flag[3],

用来存储文件类型, 如 PNT, LIN, REG。

10, 可理解为 1.0 版本。

```
//点的数据结构
typedef struct {
    double x;//点位坐标 x
    double y;//点位坐标 y
    COLORREF color;//点颜色
    int pattern;//点图案号
    char isDel;//是否被删除
}PNT_STRU;
```

```
//线索引结构
typedef struct {
    char isDel;//是否被删除
    COLORREF color;//线颜色
    int pattern;//线型 ( 号)
    long dotNum;//线节点数
    long datOff;//线节点坐标数据存储位置
}LIN_NDX_STRU;
```

```
//区索引结构
typedef struct {
    char isDel;//是否被删除
    COLORREF color;//区颜色
    int pattern;//图案 ( 号)
    long dotNum;//边界节点数
    long datOff;//边界节点数据存储位置
}REG_NDX_STRU;
```

```
typedef struct {
    double x;//节点 x 坐标
    double y;//节点 y 坐标
} D_DOT;
```

```
//文件版本结构
typedef struct {
    char flag[3]; //标志符 PNT LIN REG
    int version; //10, 可理解为 1.0 版本
} VERSION;
```

(2) 全局变量:

为了更方便的表示和使用相关变量,需定义相关全局变量,如操作状态,图案物理数,逻辑数,指向临时文件的指针等。由于全局变量太多,一下仅列出一小部分,详见附件 MapEditor 中的 CMapEditorView.cpp.

```
///-----点数据相关的全局控制变量-----///  
bool GPntFCreated = false; //临时点文件是否创建  
CString GPntFName; //永久文件名 (含路径)  
CString GPntTmpFName = CString("tempPntF.dat"); //临时文件名 (含路径)  
bool GPntchanged = false; //是否更改  
int GPntNum = 0; //物理数  
int GPntLNum = 0; //逻辑数  
CFile* GPntTmpF = new CFile(); //读取临时文件的指针对象
```

(3) 与文件读写有关:

本部分函数定义在 WriteOrRead.h 中,根据文件名、数据储存位置、Seek、Write、Read 等实现文件的读取、写入、更新。

通过文件调用 Seek 函数定位到文件中数据的位置;

通过文件调用 Write 函数实现数据的写入;

通过文件调用 Read 函数实现数据的读取。

函数	说明及实现
WritePntToFile	将点数据写入临时文件
ReadTempFileToPnt	从临时点文件读取点数据
ReadPntPermanentFileToTemp	从永久文件读取点数据到临时文件
UpdatePnt	更新点临时文件中点数据
WriteLinNdxToFile	将线索引数据写入线临时索引文件
WriteLinDatToFile	将线节点数据写入线临时数据文件
ReadTempFileToLinDat	从临时线数据文件中读取线的点数据
ReadTempFileToLinNdx	从临时线索引文件中读取线索引
WriteTempToLinPermanentFile	将线的索引和点数据写入永久文件
ReadLinPermanentFileToTemp	从永久文件读取线数据到临时文件
UpdateLin	更新线临时索引文件中线数据
UpdateLin	重载函数更新线的点数据到临时文件
AlterStartLin	修改第一条线索引
AlterEndLin	修改第二条线索引
WriteRegNdxToFile	将区索引数据写入区临时索引文件
WriteRegDatToFile	将区节点数据写入区临时数据文件
ReadTempFileToRegNdx	从区临时文件中读取区索引
ReadTempFileToRegDat	从区临时文件中读取区节点数据
WriteTempToRegPermanentFile	将区的索引和点数据写入永久文件
ReadRegPermanentFileToTemp	从永久文件读取区数据到临时文件
UpdateReg	更新区数据

(4) 绘制相关的函数:

此类函数都定义在 paint.h 中。

DrawPnt, DrawSeg, DrawReg 三个函数利用 dc, 画刷, 画笔及各种绘制函数分别实现点 (十字行, 圆形, 五角星形), 线 (实线, 虚线, 电线), 区 (实心, 空心) 的绘制。

ShowAllPnt, ShowAllLin, ShowAllReg 实现从对应的点文件, 线文件, 区文件读取数据并把相应的点, 线, 区显示在窗口中。

(5) 与计算有关的函数:

此类函数定义在 Calculate.h 中, 具体如下。

函数	功能及实现
Distance	根据两点的坐标计算两点间的距离
FindPoint	在点文件中遍历查找距离鼠标距离在 10 像素以内最近的一个点 (未删除)
DisPntToSeg	计算鼠标到到线段的最短距离, 涉及海伦公式以及角度的判断
FindLine	在线文件中遍历查找距离鼠标距离在 10 像素以内最近的一条线 (未删除)
DotToPnt	此函数有两个, 是重载函数, 分别实现批量的和单独的 D_DOT 类型变量转成 POINT 类型变量。
PntToDot	此函数有两个, 是重载函数, 分别实现批量的和单独的 POINT 类型变量转成 D_DOT 类型变量。
PntDPtoVP	实现点从数据坐标系转成窗口坐标系, 参数有缩放系数, 横纵坐标偏移量以及需要转换的点。
PntVPtoDP	实现点从窗口坐标系转成数据坐标系, 参数有缩放系数, 横纵坐标偏移量以及需要转换的点。
GetCenter	计算并返回矩形的中心点
modulusZoom	计算放大倍数 (拉框放大时)
AlterLindot	修改线的点数据 (用于线连接时将两条线的节点数据合并到一条线)
PtinPolygon	判断鼠标左键单击时鼠标是否在区内
FindReg	查找鼠标所在的最近的区 (未删除)
UpdateReg	通过重新写入的方式实现区索引数据的更新。
FindDeletePnt	在点文件中遍历查找距离鼠标距离在 10 像素以内最近的一个点 (已删除的)
FindDeleteline	在线文件中遍历查找距离鼠标距离在 10 像素以内最近的一条线 (已删除的)
FindDeleteReg	查找鼠标所在的最近的区 (已删除的)

FindPntOnLin	查找线上距鼠标最近的点
DelPntOnLin	删除线上距鼠标最近的点

(6) 对话框窗口:

新建文件, 修改点参数, 修改线参数, 修改区参数时都要弹出窗口, 应创建四个窗口类: CCreateFileDlg, CPointParameterDlg, CLinParameterDlg, CRegParameterDlg。这四个类都由 CDialogEx 类派生。创建对话框资源后, 通过 MFC 添加类向导添加相应的类。

(7) 菜单各项的消息处理函数:

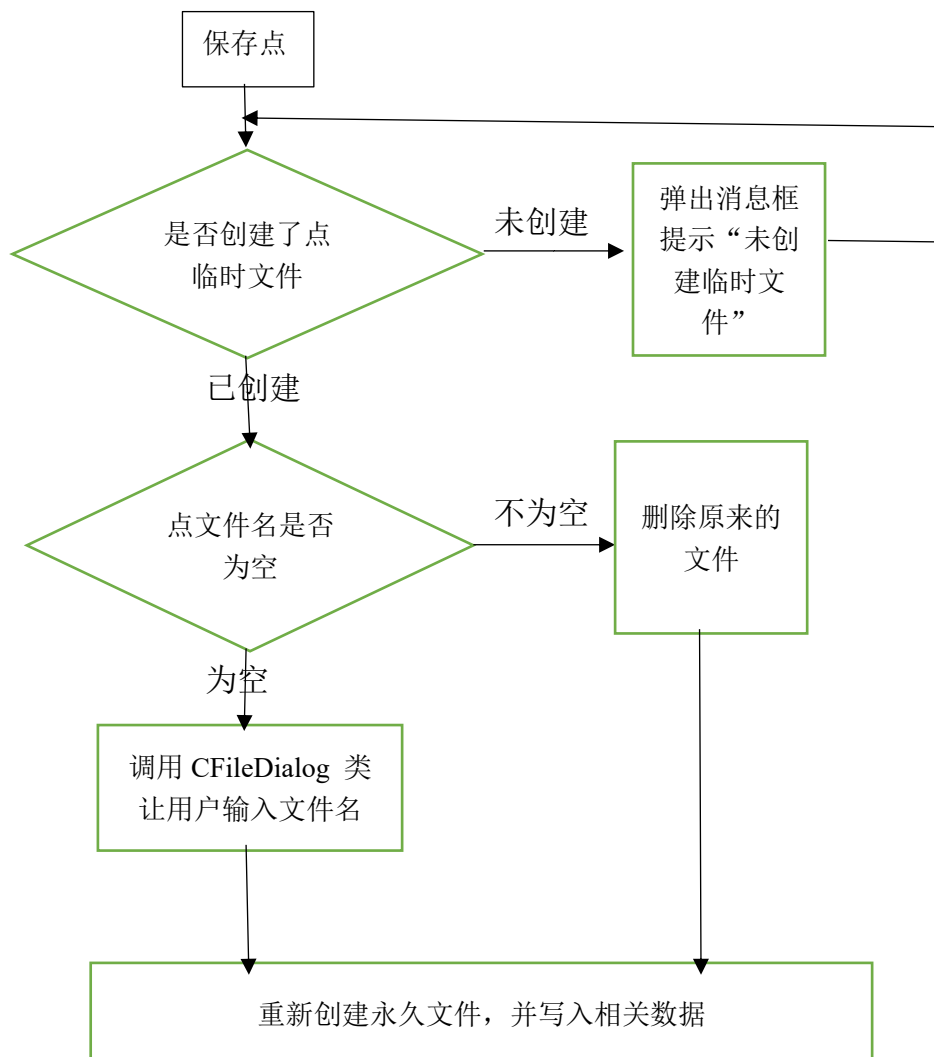
为菜单中的的各项功能添加消息处理函数到 CMapEditorView 中。

① 文件:

新建 (OnFileNew): 首先判断临时文件是否已创建, 若临时文件未创建, 则创建“新建临时对象”对话框对象, 选择文件位置并点击“确定”后, 在指定位置创建 tempLinF.dat(线临时数据文件), tempLinF.ndx(线临时索引文件), tempPntF.dat(点临时数据文件), tempRegF.dat(区临时数据文件), tempRegF.ndx(区临时索引文件)五个临时文件。

退出 (OnAppExit): 首先判断文件是否发生更改, 若发生了更改, 提醒是否保存更改。最后调用父窗口 CMainFrame 的销毁窗口函数 DestoryWindow()关闭窗口。

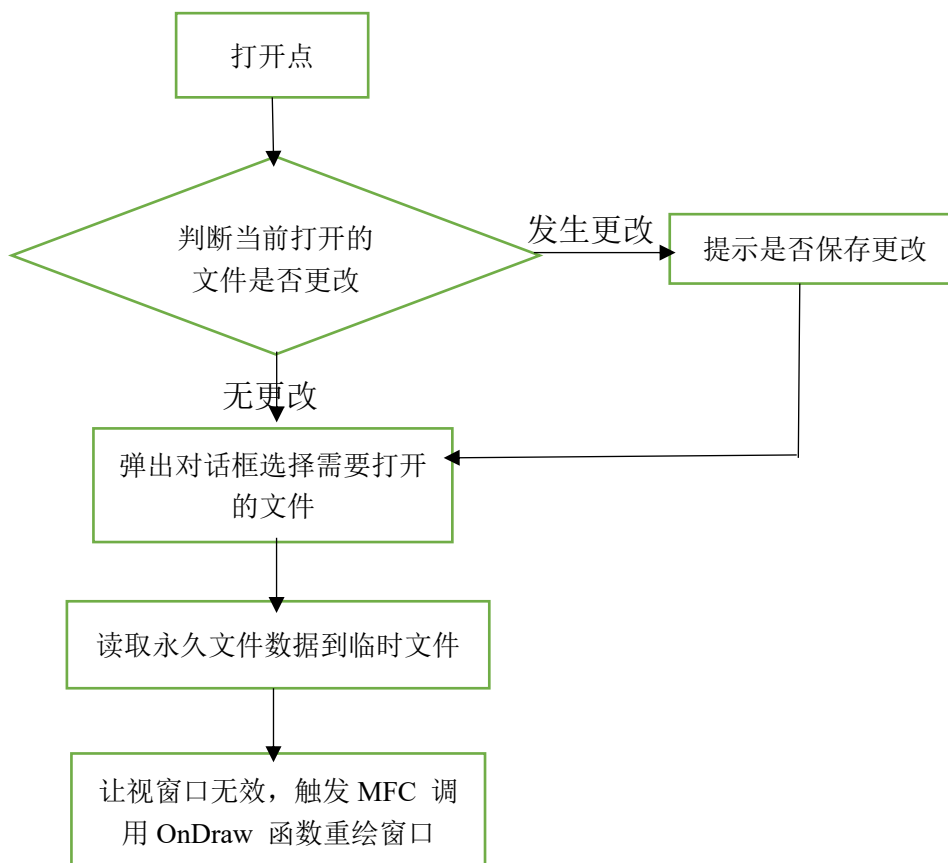
保存: OnFileSavePoint, OnFileSaveLine, OnFileSaveRegion





保存线，保存区与上述流程类似。

打开：OnFileOpenPoint, OnFileOpenLine, OnFileOpenRegion



打开线，打开区与上述流程类似。

另存：OnFileSaveAsPoint, OnFileSaveAsLine, OnFileSaveAsRegion。

以另存点为例：首先将永久文件名设为空，再调用 OnFileSavePoint 函数，若另存失败则恢复原本的文件名。

② 窗口：

放大，缩小，移动，复位四个对应得函数的内容都是将系统操作状态 **GCurOperState** 设置为相应的状态。

显示点，显示线，显示区通过更改 **GShowPnt, GShowLin, GShowReg** 的状态，来实现不同的显示状态。

以显示点为例：首先，若当前显示状态是显示删除状态,则先把所有显示开关关闭，并将显示状态显示为“显示未删除”。若当前已显示点，则关掉开关，不显示点；否则打开开关，显示点。

显示线，显示区与上述同理。

③ 点编辑:

造点, 删除点, 移动点, 修改点参数, 恢复点对应函数的内容都是将系统操作状态 **GCurOperState** 和显示状态 **GCurShowState** 设置为相应的状态。

显示删除点对应函数设置当前显示状态后刷新窗口, 使窗口显示删除的点。

设置点缺省参数对应函数调用“修改点参数”对话框并更改点默认参数。

④ 线编辑:

造线, 删除线, 移动线, 连接线, 恢复线, 线上删点, 线上加点, 修改线参数对应的函数的内容都是将系统操作状态 **GCurOperState** 和显示状态 **GCurShowState** 设置为相应的状态。

显示删除线对应函数设置当前显示状态后刷新窗口, 使窗口显示删除的线。

设置线缺省参数对应函数调用“修改线参数”对话框并更改线默认参数。

⑤ 区编辑:

造区, 删除区, 移动区, 恢复区, 修改区参数对应函数的内容都是将系统操作状态 **GCurOperState** 和显示状态 **GCurShowState** 设置为相应的状态。

显示删除区对应函数设置当前显示状态后刷新窗口, 使窗口显示删除的区。

设置区缺省参数应函数调用“修改区参数”对话框并更改区默认参数。

(8) 鼠标状态对应函数:

鼠标对应状态有左键按下 (**OnLButtonDown**), 左键弹起 (**OnLButtonUp**), 右键弹起 (**OnRButtonUp**), 鼠标移动 (**OnMouseMove**)。

OnLButtonDown: 涉及的操作状态有移动点, 移动线, 移动区, 放大。

在移动点, 移动线, 移动区中主要实现都是查找最近的点/线/区。

OnLButtonUp: 涉及操作状态比较多, 有绘制点, 删除点, 移动点, 恢复点, 修改点, 绘制线, 删除线, 移动线, 连接线, 恢复线, 修改线, 线上加点, 线上删点, 绘制区, 删除区, 移动区, 恢复区, 修改区, 放大, 缩小。

OnRButtonUp: 涉及操作状态有绘制线, 线上加点, 绘制区。

OnMouseMove: 涉及操作状态有移动点, 绘制线, 移动线, 线上加点, 绘制区, 移动区, 拉框放大。

接下来分点, 线, 区三部分讨论:

① 点:

绘制点: 鼠标左键弹起后, 可根据鼠标位置设置点的坐标并进行坐标系转换, 同时设置点的物理数和逻辑数都加一, 将点数据写入临时文件。后调用 **DrawPnt** 函数将点绘制在窗口上。

移动点: 首先, 根据鼠标左键弹起的位置, 调用 **FindPoint** 函数查找到

移动的点；然后鼠标移动时，不断在鼠标上一位置绘制点和在鼠标目前位置绘制点，实现点的移动，直至鼠标左键再次按下。

删除点：根据鼠标左键弹起的位置，调用 FindPoint 函数查找到删除的点，将点的删除标志设为 true，并调用 UpdatePnt 函数更新该点的数据，同时，点逻辑数减一。刷新重绘。

恢复点：显示状态转换为显示删除点，根据鼠标左键弹起的位置，调用 FindDeletePnt 函数查找到需要恢复的点，将点的删除标志设为 false，并调用 UpdatePnt 函数更新该点的数据，同时，点逻辑数加一。刷新重绘。

修改点参数：根据鼠标左键弹起的位置，调用 FindPoint 函数查找到需要修改参数的点，调用“修改点参数对话框”，更改点数据，并调用 UpdatePnt 更新该点数据，刷新重绘。

② 线：

绘制线：鼠标左键弹起则确定一个节点，连接相邻两个节点形成折线，鼠标移动时不断重绘上一节点与鼠标上一位置之间连线，以清除上一条连线，同时连接上一节点与鼠标目前位置，直至鼠标右键弹起。连线时调用 DrawLin 函数。将线索引和线节点数据写入相应文件中。

删除线：通过 FindLine 函数查找线，其他操作与删除点类似，不再说明。

移动线：通过 FindLine 函数查找线，其他操作与移动点类似，不再说明。

恢复线：与恢复点类似，不做说明。

修改线参数：与修改点参数类似，不做说明。

连接线：首先根据鼠标左键按下位置确定第一条线并在第一条线两个端点画上圆圈标记，鼠标左键再次按下，找到第二条线，通过调用 AlterLinDot 确定要连接的两个端点，并将两条线的数据合并，线逻辑数减一，更新线数据，刷新重绘。

线上删点：根据鼠标左键按下的位置找到对应的线，调用 FindPntOnLine 找到线上要删除的点。（在删除点之前，先判断线节点数是否大于 2，只有线节点数大于 2 才可删点。）随后，调用 DelObtOnLine 删除线上对应的节点，删除节点的方法是将删除点后的节点都往前移动一个位置，并将节点数减一。刷新重绘。

线上加点：根据鼠标左键按下的位置找到相应的线上的相应的线段，鼠标移动时，重绘线段两个端点与鼠标上一位置的连线，绘制线段两个端点与鼠标目前位置的连线，线节点数加一，并将新的节点数写入文件。刷新重绘。**注意：**若加点的线在文件中的位置不是最后，则将在文件最后重新添加一条线，将要加点的线的节点以及所加的节点写入到对应位置中，将原本要加点的线设为已删除。

③ 区：

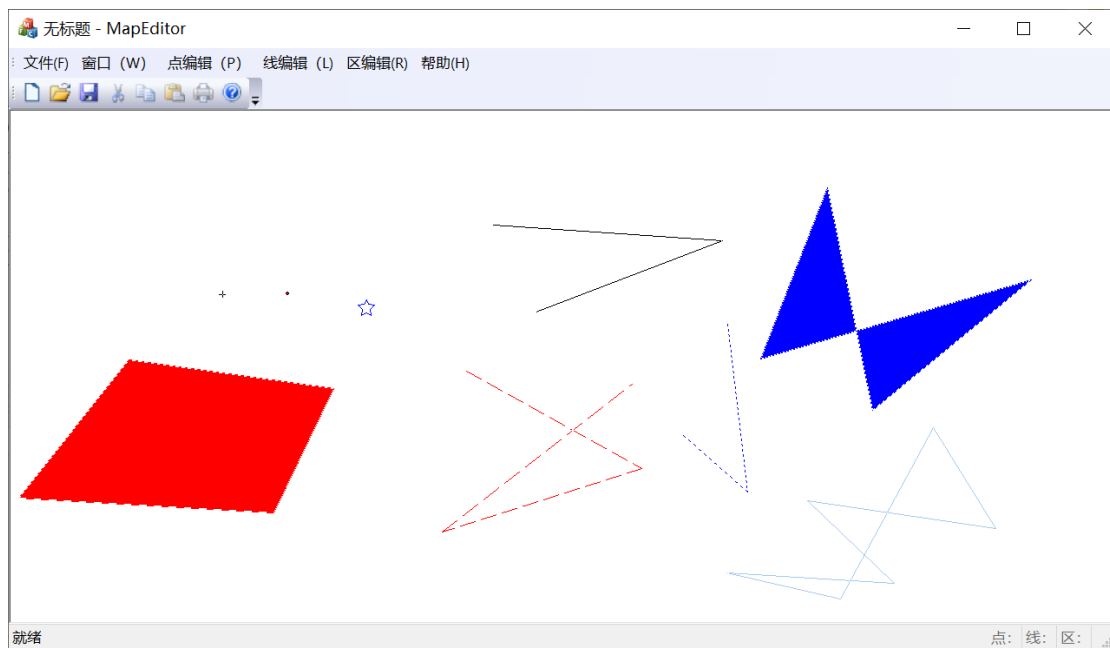
造区，删除区，移动区，恢复区，修改区参数可结合上述点模块和线模块得知，这里不再作说明。

3.软件开发

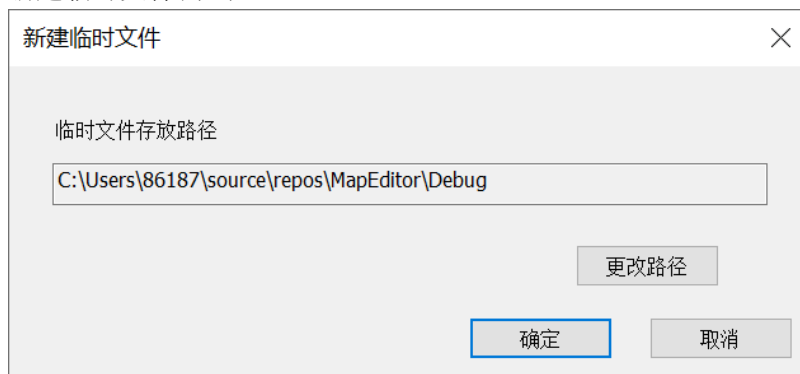
在 Visual Studio2019 环境下，使用 C++语言编写。
















4.软件调试与测试

造点，造线，造区测试：



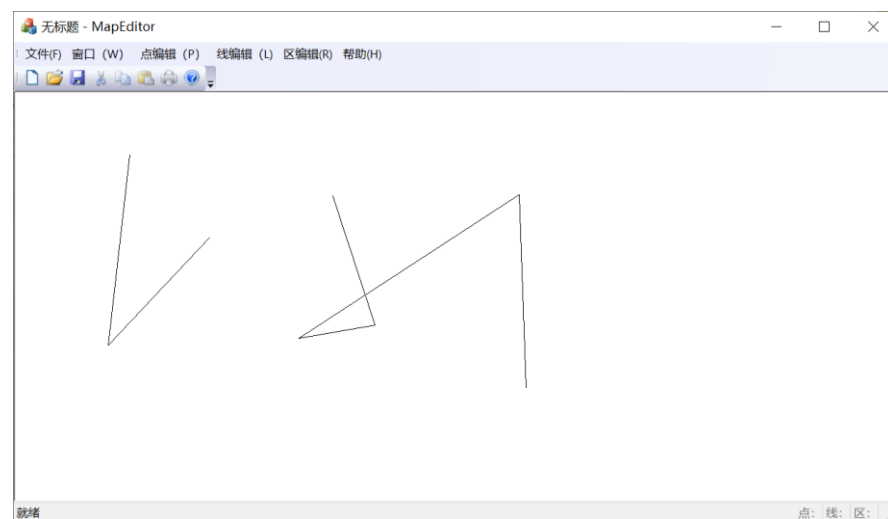
新建临时文件测试：



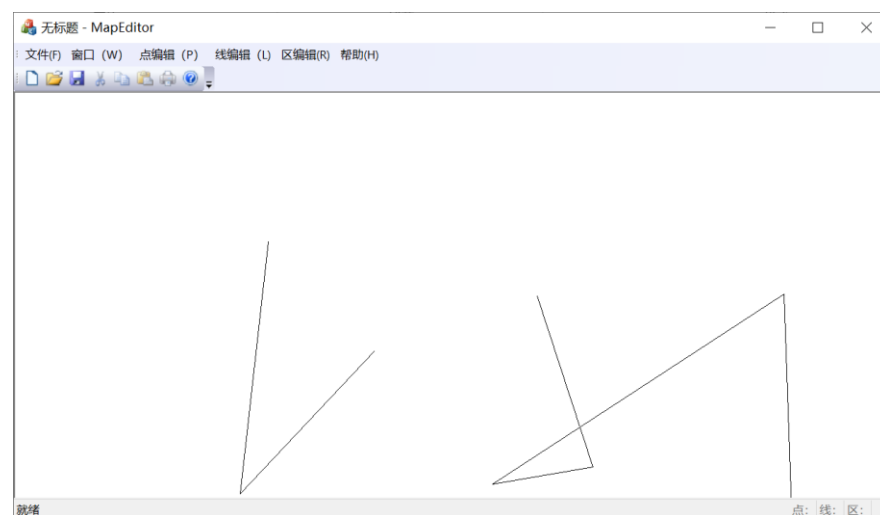
« source » repos » MapEditor » Debug		搜索"Debug"
名称	修改日期	类型
 tempLinF.dat	2020/6/6 19:47	DAT 文件
 tempLinF.ndx	2020/6/6 19:47	NDX 文件
 tempPntF.dat	2020/6/6 19:47	DAT 文件
 tempRegF.dat	2020/6/6 19:47	DAT 文件
 tempRegF.ndx	2020/6/6 19:47	NDX 文件
 MapEditor	2020/6/6 16:17	应用程序
 MapEditor.ilkk	2020/6/6 16:17	Incrementa
 MapEditor.pdb	2020/6/6 16:17	Program D
 test8.lin	2020/6/3 19:59	LIN 文件
 test7.lin	2020/5/29 18:12	LIN 文件
 test.lin	2020/5/27 20:53	LIN 文件
 test6.lin	2020/5/25 20:38	LIN 文件
 test.pnt	2020/5/25 12:51	PNT 文件
 test5.lin	2020/5/22 19:31	LIN 文件
 test7	2020/5/22 19:27	注册表项

放大，缩小，复位测试：

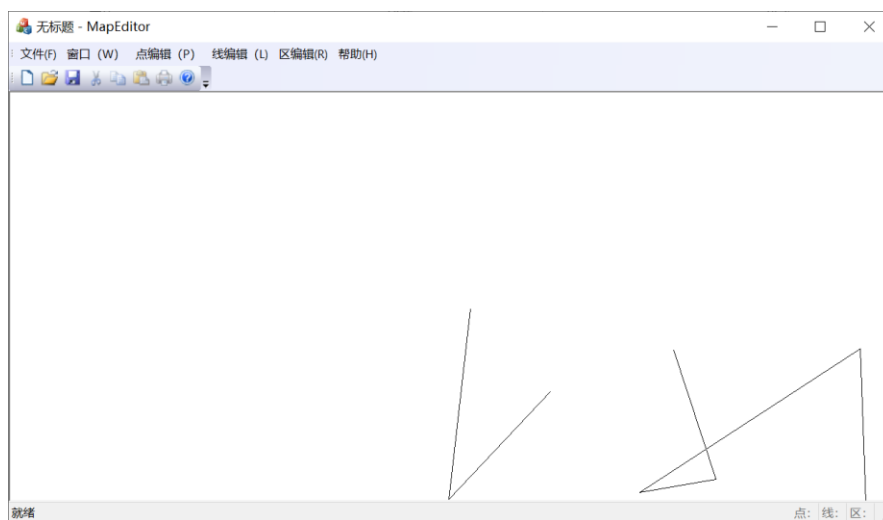
原图：



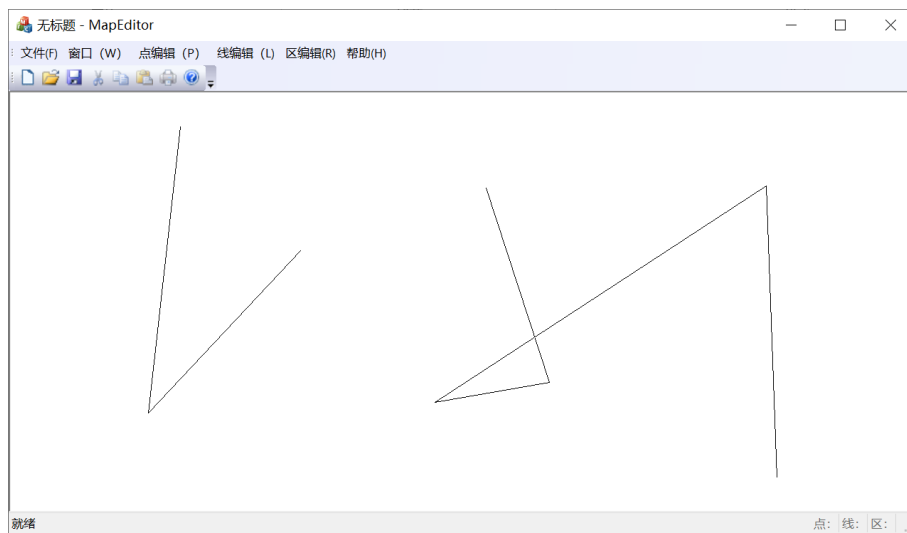
放大后：



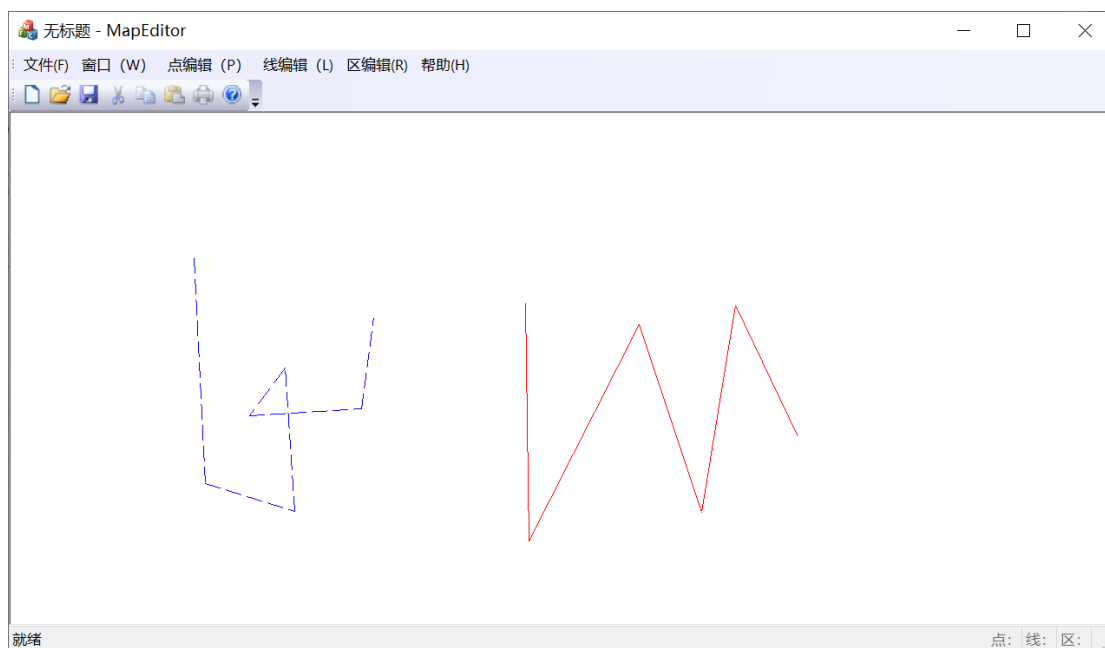
缩小后：

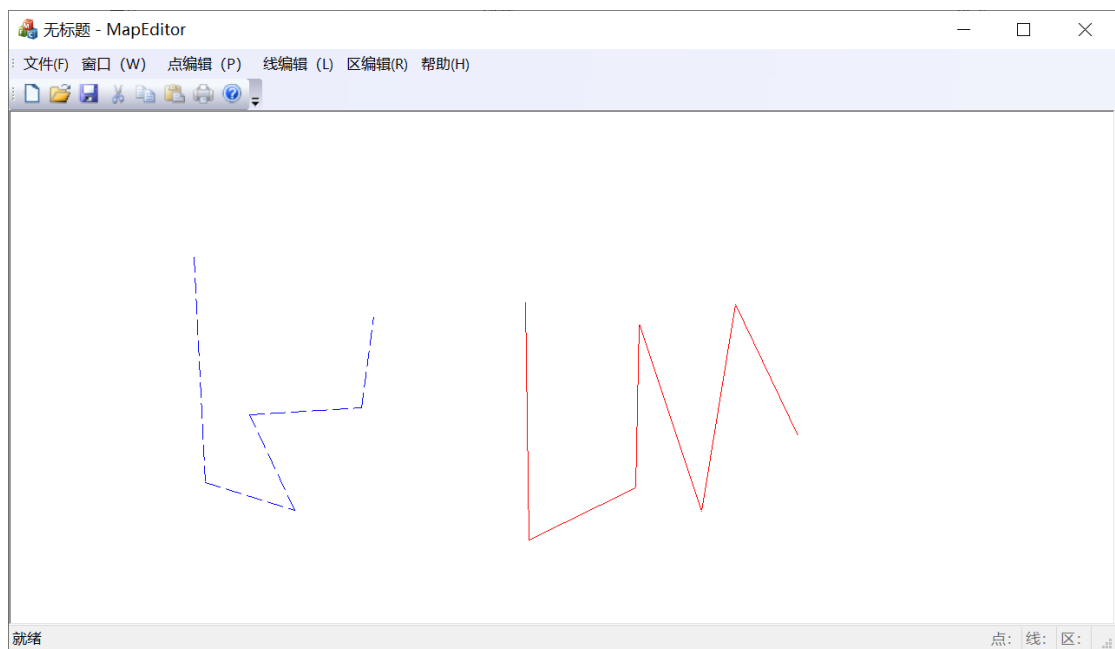


复位后：



线上加点，线上删点测试：





其他功能测试不方便通过图片看出，可自行测试程序。

5.特点与不足

5.1 技术特点

书上的练习运用了结构体，全局变量，异或画图等，我觉得用得非常巧妙。而对于后面 31, 32 自己完成得代码，感觉中规中矩，没有什么技术特点。

5.2 不足和改进的建议

无法做到像书上示例代码那样巧妙、简介；没有实现显示几何图形数量，部分删除，增加线型的图案等功能。

6.过程和体会

6.1 遇到的主要问题和解决方法

在实习过程中，遇到过缺漏代码等问题，如点参数改变后忘记更新点参数等。后通过逐步调试的方法找到问题所在。

除此之外，还存在未解决的问题，线上加点在放大时存在显示的问题，但最终画出来的结果没有问题。保存区文件文件类型总显示为注册表项，但其他打开等操作都没有问题。

6.2 程序设计的体会

在程序设计的过程中，脑子里必须要有一个完整的设计体系，不然容易造成各种缺少代码，考虑不周引发的问题和错误。

除此之外，调试是发现问题和解决问题最实际的办法。

7.源码和说明

7.1 文件清单及其功能说明

MapEditor：源码

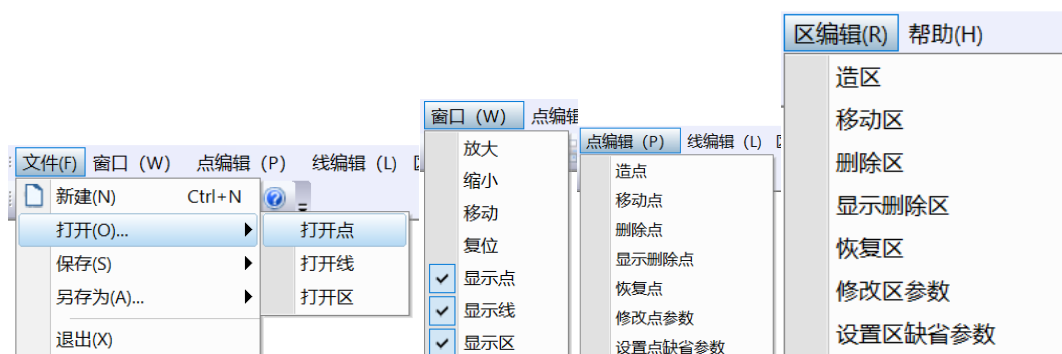
Release→MapEditor.exe：可执行文件

7.2 用户使用说明书

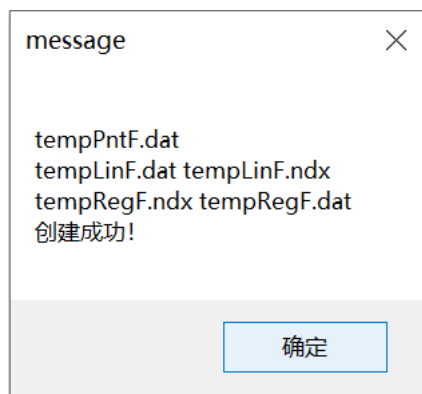
初始界面如下：



各菜单详细内容：



进行绘制前：首先需要新建临时文件，操作步骤为文件→新建，在弹出的对话框中选择好路径并确认后，弹出以下窗口则表示创建成功。



造点：选择点编辑→造点后，鼠标左键单击便可造点。

造线：选择线编辑→造线后，鼠标左键单击一次确定一个节点，单击右键结束造线。（节点数需在两个或两个以上才能造线）

造区：选择区编辑→造区后，鼠标左键单击一次确定一个节点，单击右键结束造区。（节点数需在三个或三个以上才能造区）

删除点/线/区：选择删除后，鼠标左键单击要删除的点/线/区即可删除。

移动点/线/区：选择移动后，鼠标左键单击要移动的点/线/区再移动鼠标即可实现移动。

恢复点/线/区：选择恢复后，窗口显示已删除的点/线/区，鼠标左键单击需要恢复的点/线/区，再切换为显示未删除状态，即可看到恢复后的点/线/区。

修改点/线/区参数：选择修改后，鼠标左键单击需要修改参数的点/线/区，在弹出的对话框中选择想要的参数并确定后即可修改参数。

设置点/线/区缺省参数：与修改参数一样，在弹出的对话框选择好参数并确定后，再绘制即可。

放大：选择窗口→放大后，可通过鼠标左键单击客户区或者拉框的方式放大窗口。

缩小：选择窗口→缩小后，鼠标左键单击客户区即可缩小。

复位：选择窗口→复位后，即可复位。

线上删点：选择线编辑→线上删点后，鼠标左键单击需要删除的线上的点即可实现线上删点。

线上加点：选择线编辑→线上加点后，鼠标左键单击需要加点位置的线段再移动鼠标到要加点的位置，单击右键即可。

7.3 源代码

由于源代码量过大，所以不粘贴，已经将源代码存在附件 MapEditor 文件夹中。