题目:基于 matlab c-mex 的维诺图实现

学院：软件学院

老师：汤德佑

学生姓名：王琥

学号：201421031134

（附：可执行文件的使用说明见附录）

总体设计

名称	基于 matlab c-mex 的维诺图实现
设计目的	1. 合理设计数据结构，应用所学算法编写维诺图程序，并有良好的图形化界面； 2. 将所学的 c 语言知识和 matlab 结合起来，形象化的解决实际问题； 3. 培养综合运用专业知识，解决实际问题的能力。
编程环境	1. PC + Win7； 2. 安装 VS2010，MATLAB 等开发工具。
任务要求	1. 用 c 语言编写维诺图算法； 2. 实现鼠标点击的交互式界面（鼠标左键输入点集，右键结束输入）； 3. 实现交互式界面与 c 文件的 c-mex 接口对接。

一、问题描述

为了高质量的完成老师交给的维诺图编写任务，我查阅了很多资料，在周培德的《计算几何——算法设计与分析》一书中有一个很经典的比喻：假设图上有 n 个点，如果把每个点看做一个火源，如果把这 n 个点同时点燃，并以相同的速度向各个方向蔓延，那么燃烧熄灭处所形成的图便是 Voronoi 图。

这是我觉得最生动的一个比喻，通俗的理解维诺图就是空间或者平面的点集中，离一个定点最近的所有点的集合。如果用较严格一些的定义：对于一个空间 M，假定其中存在着一个结点集 S；S 中的任意一个结点都可以对其周围环境—

—属于空间 M 的点集，施加影响；对于空间 M 中的一组点集，在 S 的所有结点中，它们受一结点 p 的影响最为强烈，那么这组点集就构成了结点 p 的一个作用域。

二、算法设计与分析

既然要设计算法解决维诺图问题，就需要对维诺图的特性进行进一步的分析。据分析，维诺图的形成基本上就是三角形网的外心连线，这样问题就转化为求三角形网络。我决定采用 Delaunay 三角剖分算法进行三角形网络的构建。Delaunay 剖分具有以下一些优秀的特性：

1.最接近：以最近的三点形成三角形，且各线段(三角形的边)皆不相交。

2.唯一性：不论从区域何处开始构建，最终都将得到一致的结果。

3.最优性：任意两个相邻三角形形成的凸四边形的对角线如果可以互换的话，那么两个三角形六个内角中最小的角度不会变大。

4.最规则：如果将三角网中的每个三角形的最小角进行升序排列，则 Delaunay 三角网的排列得到的数值最大。

5.区域性：新增、删除、移动某一个顶点时只会影响临近的三角形。

6.四点不同圆特性。

（1）失败的经历

我开始根据以上特性写出了一个三角形刨分的暴力算法，算法的思想很简单，用户输入的所有点集中每三个点都形成一个三角形，然后判断每个三角形和除去顶点的所有点是不是同圆，如果同一个圆就去掉这个三角形。

这个算法根据定义得出，正确性是毋庸置疑的，但是算法的效率令人堪忧，算法在点数较少的情况下可以很迅速的得出答案，点数一旦增多，处理速度就急速变慢。我测试是 25 个点就要等几十秒。。。原因也很简单，因为这个算法的两步：1）构成所有三角形，这是一个 O（n^3）的一个复杂度　2）判断每个三角形的有效性是一个 O（n^2）的复杂度，这样复杂度就很高了。

（2）重新编码

吸取暴力法失败的经历之后，我思考了另外一种比较高效的算法。算法的基本步骤是：

1、构造一个超级三角形，包含所有散点（这些散点是用户一个个输入的），放入三角形链表。

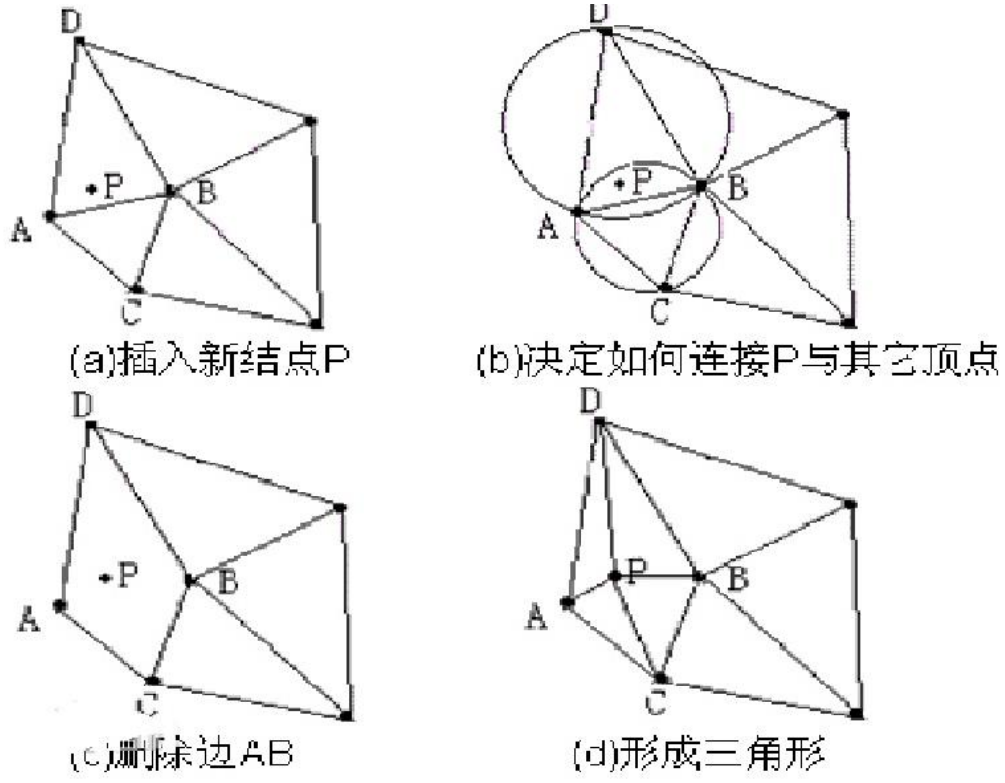
2、将点集中的散点依次插入，在三角形链表中找出外接圆包含插入点的三角形（称为该点的影响三角形）；这里有一个很重要的细节处理，将所有影响三角形的边存入一个边数组，删除影响三角形的公共边，将插入点同影响三角形的全部顶点连接起来，完成一个点在 Delaunay 三角形链表中的插入。

3、优化三角形，并将形成的三角形放入 Delaunay 三角形链表。

4、循环执行上述第 2 步，直到所有散点插入完毕。

5、连接外心，形成维诺图。

这一算法的关键的第 2 步图示如下：



（3）算法复杂度分析

算法主要的耗时步骤是在第 2 步，找出新插入点的影响三角形需要 O（n）的复杂度；将所有影响三角形的边存入一个边数组，删除影响三角形的公共边这

个需要 O（n^2）的复杂度。综合来说，这个算法的时间复杂度是 O（n^2）。

三、具体代码实现

（1）变量定义

为了代码运行的有更快的速度，我使用了 ANSI C 编写，所有数据结构都是自己定义，没有使用任何 c++的 STL 或者模板类。使用数组定义，虽然会使用多一些空间，但是都是采用下标查找的方式，在 O（1）时间内直接定位，速度的确得到了不错的提升。

回想编码的整个过程，虽然用自己定义的数据结构很辛苦，但是收获良多。

实现过程中各变量的定义说明如下：

1）结构体定义

```
typedef struct          //点结构体定义
{
  double x;                    //x，y 点坐标
  double y;
}Point;

typedef struct                 //边结构体定义
{
  //存放边的始点和终点下标
  int Start,End;

  //存放边两边的三角形下标
  int Left;
  int Right;
}Edge;


typedef struct       //三角形结构体定义
{
  //三角形的三个顶点
  //这里存放的是 point 数组的下标
  int A,B,C;

  //三角形外心坐标
  Point Center;
  double radius;//外接圆半径
```

```
  bool isValid;//判断是不是有效的
}Triangle;
```

2）变量定义

```
Triangle triangle[];   //存放所有出现的三角形
Point point[];         //存放输入点集

int pCnt=0;//记录输入点数
int triCnt=0;//现有三角形个数
int tmpAffTri[];   //记录当前受影响的三角形【下标】
int affectedTriCnt=0;     //当前受影响的三角形的数目
int affEdgeStart[];//受影响的三角形始点
int affEdgeEnd[];   //受影响的三角形终点
int outTri[];   //最后输出的三角形下标
int outTriCnt=0;     //输出 Delaunay 三角形的个数
Edge edge[];   //最后找链接外心用到的边数组
int edgeCnt=0;     //控制最后边数输入
```

（2）功能函数的实现（不一定按照出现顺序）

```
void init();                          //初始化整个函数
double distance(Point A,Point B);   //求两点的欧式距离
void    miniCircle(Point *p,Point&    center,double&    radius);    //求三角形的外心和外接圆半径
void freshTri(int sub);               //当三角形顶点改变时，更新三角形卫星数据（外心和外接圆半径）
bool isAffected(int n,int i);         //判断一个三角形是不是点 n 的影响三角形
void storeAffEdge(int n);             //存入所有影响三角形边（顺便去掉重边）
void pointProcess(int n);             //依次处理每个点
void storeNewTri(int n);              //存入所有新三角形
void offSupTri();                     //去掉超级三角形相关的三角形
void initEdge();                      //初始化边
void findEdgeTri();                   //找到维诺边两侧的邻接三角形
```

（3）c-mex 接口函数的实现

1）利用 MATLAB 中接口函数构建 c 语言在 matlab 中实现的接口

void mexFunction(int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[])

功能说明：进行 c 文件与 matlab 接口的实现

参数说明：

int nlhs:函数输出参数的个数

mxArray *plhs[]:输出参数为 mxArray 结构的指针数组变量

int nrhs:函数输入参数的个数

const mxArray *prhs[]:输入参数为 mxArray 结构的指针数组变量

2）pn= mxGetPr(prhs[0])

功能说明：获取数据数组中实数部的指针赋给 pn，代表获取所有点数

(3) plhs[0] = mxCreateDoubleMatrix(outTriCnt*3*2, 1, mxREAL)

y = mxGetPr(plhs[0]);

功能说明：获取数据数组中实数部的指针赋给 y

plhs[1] = mxCreateDoubleMatrix(1, 1, mxREAL);

y1 = mxGetPr(plhs[1]); //获取第二个输出指针，赋给 y1，传递连接 Delaunay 三角形的点下标（这些点的坐标在 matlab 中已获取，没有必要再传递一次）。

plhs[2] = mxCreateDoubleMatrix(edgeCnt*2*2, 1, mxREAL); //传递维诺边的起止坐标

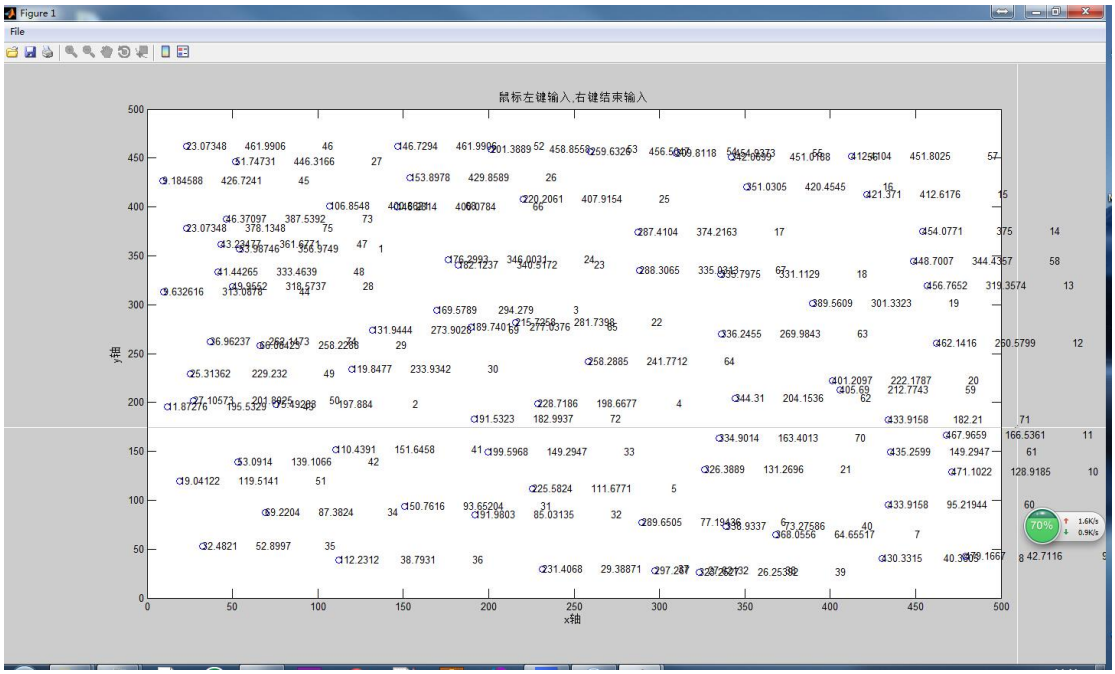
y2 = mxGetPr(plhs[2]); //获取第 3 个输出指针，赋给 y2

plhs[3] = mxCreateDoubleMatrix(1, 1, mxREAL);

y3 = mxGetPr(plhs[3]); //获取第 4 个输出指针，赋给 y3

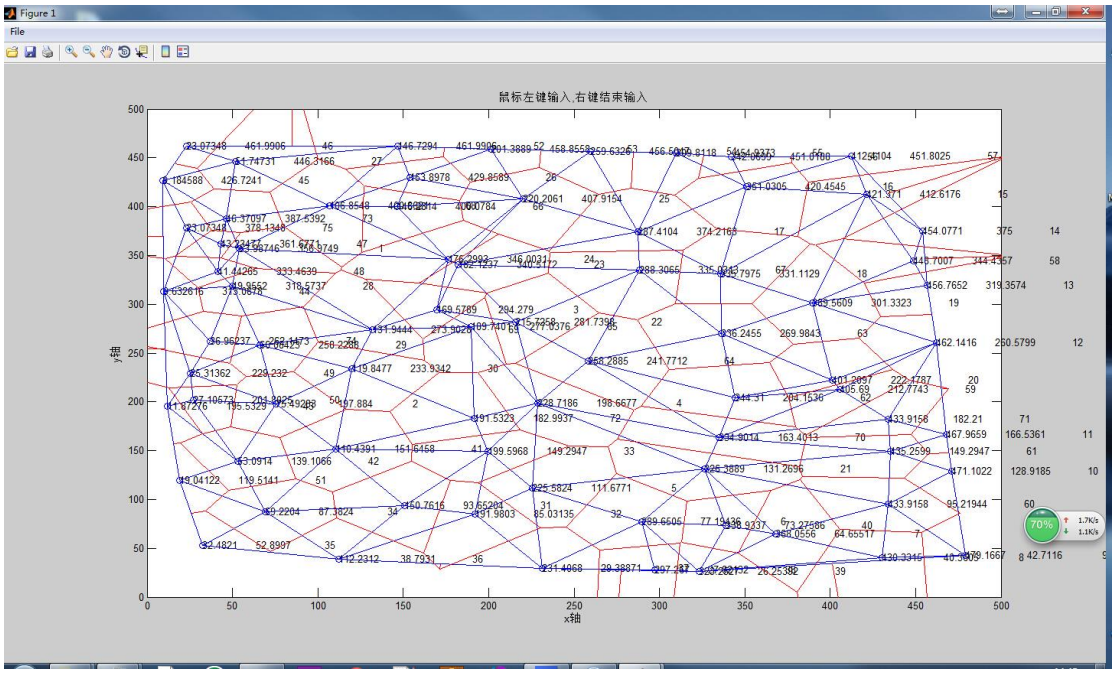
y3[0]=edgeCnt;

四、运行结果



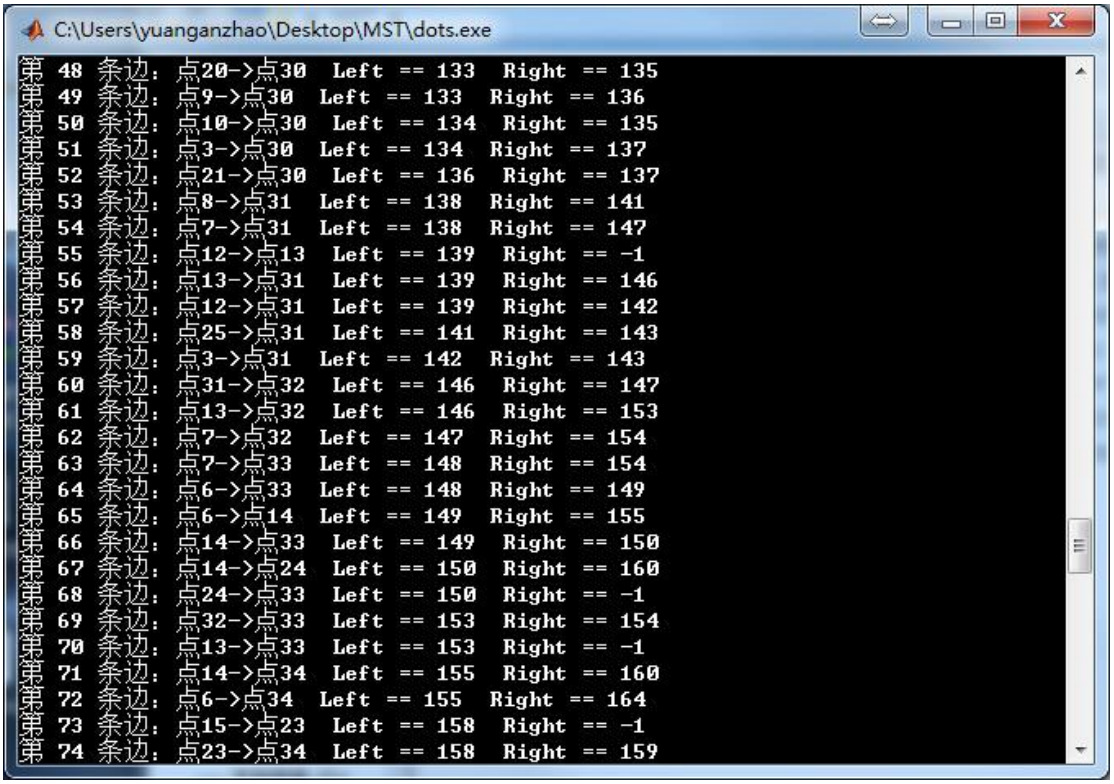
鼠标左键输入点集

图上会显示点的坐标和第几个点



鼠标右键结束输入

输出 **Delaunay** 三角形和维诺边



输出的辅助信息

附录：可执行文件使用说明

由于我是用的 matlab 和 c-mex 接口写的程序，所以在没有安装 MATLAB R2013a win64 的机器上是直接跑不起来的，需要有 MATLAB R2013a win64 的 runtime 库才行。。

这样就有两种方法：

直接安装 MATLAB R2013a win64 bit 的程序，执行 MST 包里面的 dots.exe 或者直接在 matlab 界面下 run dots.m 文件都可以。

安装 MCRInstaller.exe（一定要 MATLAB R2013a win64 bit），然后就可以运行 MST 包里面的 dots.exe 了。

如果没有 MCRInstaller.exe 的程序（413MB），我已经上传到 360 云盘，地址是：http://yunpan.cn/cgwIFgVLefaSA　提取码 1249，提供下载。