基于改进的活性边表填充算法的任意多边形填充

李桂清　（南宁广西大学计算机与信息工程学院　530004）

摘要：本文通过对重迭自相交的多边形区域进行分析，提出一种改进的活性边表填充算法，既保留了活性边表算法充分利用点和扫描线相关性的特点，又扩充了原算法的适用范围。

关键字：多边形区域　填充　边表　活性边表填充算法

一、引言

总的说来，传统区域填充算法大都只适用于简单多边形，因此在实际应用中都或多或少需要作改进。为了解决轮廓汉字填充问题[1]提出了一种基于描线求交的填充算法，用以处理轮廓汉字笔画相交的问题，例如图 1 中空心笔画构成的多边形是自相交的，传统的填充算法将会出现漏填。本文通过改进活性边表算法，解决上述问题，并与[1]的算法作简单比较。

(a) 自相交的多边形　(b) 一般填充算法漏填情形

图 1　自相交多边形漏填实例

二、活性边表算法

多边形区域填充最自然的方法是取显示区域的每一个点判断是否在区域内来决定是否填充。点在区域内判断方法则是从这一点引一条射线，计算射线与多边形所有边的交点，交点个数为奇数，点在区域内，否则在区域外。这种方法没有充分利用区域内点的连贯性，即某点在区域时其邻点也可能在区域内的可能性也很大。因此提出了一批基于扫描线求交的算法，比如对每条 y 扫描线求出它与多边形边的所有交点，并按 x 坐标大小排序，按奇偶性法则，即只对处在奇数位和偶数位的相邻两交点连线填充。活性边表算法是这类算法中最有效的方法之一。

活性边表算法还注意到上下扫描线的相关性。设当前扫描线 y 与某边交点坐标为(x, y)，则下一条扫描线 y－1 与此边的交点 x 坐标为 x－1/k，k 为边的斜率。从而避免了求交运算。算法可以分为两大部分(以下设多边形顶点按逆时针排列依次为 P1, P2, . . . Pn, Pi = (xi, yi))：

1. 建立边表

对每条边建立如下节点信息：

```
struct edgeInfo{
  int x;                    // 保存两端点中，y 坐标较大点的 x 坐标
  double deltaX; // －1/k
  double deltaY; // 与之相交的扫描线条数
  struct edgeInfo *next; // 指向另一条边的节点
};
```

把边节点放入与之相交的最大 y 扫描线桶中。实际处理时节点信息需按如下三条原则修正。

(1) 水平边不产生节点。

(2) 当 Pi－1, Pi, Pi＋1 的 y 坐标严格单调并且扫描线经过 Pi 时用 Pi'Pi＋1 代替 PiPi＋1 来求边的节点信息，其中单调减时 Pi' = (xi－1/k, yi －1)，单调增时 Pi' = (xi＋1/k, yi＋1)。如果 Pi－1Pi 是水平边，则看 Pi－2, Pi, Pi＋1 的单调性，依此类推。

(3) 为了避免因象素具有大小而引起的失真，往往采用象素中心扫描线 y＋0.5 代替扫描线 y 进行求交，相应地成员 x 修正为 x＋0.5/k；而在后面对[x1, x2]的填充区间段，只填充满足 x1≤x＋0.5≤ x2 的(x, y)象素。

例如图 2(b)的多边形相应边表如图 2(a)所示。

2. 扫描线填充

填充线段的数据通过建立称为活性表的动态链表来

存放。算法的类 C 语言描述如下：Ymax, Ymin 为 y 扫描线的最大最小值；Bucket 是扫描线桶；AEL 是活性表指针。

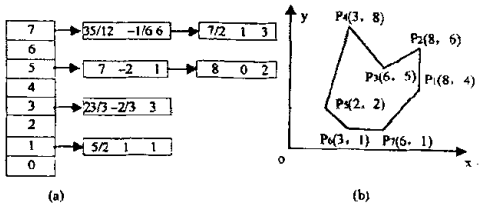

图 2　多边形及其边表

```
struct edgeInfo * Bucket [Ymax], * AEL;
y = Ymax;
while (y≥Ymin){
    append(AEL, Bucket[y]);  // 把 Bucket[y]链到 AEL 之后
    sort(&AEL) ; // 对 AEL 按 x 成员递增排序;
    fillScan(&AEL) // 按顺序逐对取出节点填充扫描线段;
    update(AEL); // 修改 AEL;
    deleteZeroNode(&AEL) // 删除 deltaY 成员为零的节点;
    y = y - 1;
}
```

上述程序段 append()把放在扫描线 y 桶中的边的信息加入到活性边表中；sort()函数排序保证第 1 和第 2 节点、第 3 和第 4 节点等等奇节点和偶节点间的区段落在区域内。Update()修改 AEL 每个节点的 x 和 deltaY 成员：x ← x + deltaX ；deltaY ← deltaY − 1 ；活性表中 deltaY= 0 的节点对应边的扫描线已填充完毕，调用 deleteZeroNode()删除。

三、算法的改进

活性边表算法可以填充凸、凹甚至带孔的多边形，但不能填充大部分自交的多边形。其原因是它采用奇偶法则填充扫描线，即对同一条扫描线所有交点排序，奇偶位置交点间的线段在区域内，偶奇位置交点线段在区域外。对任意多边形上述法则不成立。图 3 P1P2P3P4P5 是一个自交的多边形，扫描线 L 的四个交点间的线段都在区域内，但用上述方法，中间粗线段将不被填充，这是因交点配对不合理造成的，事实上 P1P2 和 P4P5 的右侧属于区填充域，不妨称为左侧边，相应的交点称为左交点；P2P3 和 P5P1 的左侧属于填 充区域，这些边和相应交点分别称为右侧边和右交 点。按新的原则进行填充：从左到右依次取左交点与最近的右交点匹配进行填充。

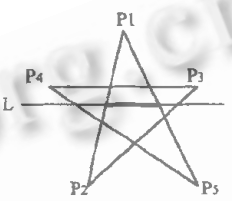

图 3

如果规定多边形顶点按逆时针顺序排列，边 PiPi + 1，Pi = (xi, yi)，Pi + 1 = (xi + 1, yi + 1)，满足 yi < yi + 1 则是右侧 边，否则为左侧边。建立左右两个边表分别存放左右侧边的信息，做法与 2. 1 一样。

算法的填充部分可以改为：

Ymax, Ymin 为 y 扫描线的最大最小值；BucketLeft, BucketRight 分别是左右边表；

AELLeft, AELRight 分别是与左右边表相对应的活性表指针。

```
struct edgeInfo * BucketLeft[Ymax], * AELLeft, * BucketRight[Ymax], * AELRight;
y = Ymax;
while (y≥Ymin){
    append(AELLeft, BucketLeft[y]); append(AELRight, BucketRight[y]);
    sort(&AELLeft) ; sort(&AELRight) ;
    fillScan(&AELLeft, &AELRight) ;// 按顺序逐对取出节点填充扫描线段;
    update(AELLeft); update(AELRight);
    deleteZeroNode (&AELLeft); deleteZeroNode (&AELLeft);
    y = y - 1;
}
```

函数 fillScan()分别取出 AELLeft 和 AELRight 处在相同位置的节点配对进行填充，其他函数同前。算法了

多一个桶的内存开销，两个活性表加起来与原来的相等，对两个较小的链表排序比原来效率更高。改进后的算法可填充任意多边形，而填充凸、凹、带孔的多边形时效率与活性边表算法一样。

四、算法比较

首先列出[1]中提出的算法。

二维数组 A[Ymax][n]，B[Ymax][n]，Ymax 扫描线最大值，n 为与扫描线的最大交点数。

```
y=Ymax;
while (y≥Ymin){ // 扫描线最小值
  求扫描线与每条边的交点 x 值，与左侧边的交点存 A[y]，与右侧边交点存 B[y]，
  A[y]和 B[y]相当于 y 桶，插入桶时按 x 递增排序；
  y=y+1;
}
y=Ymax;
while (y>=Ymin){ // Ymin 是扫描线最小值
  for( i = 0;i<n; i++)
    填充(A[y][i],y)与(B[y][i],y)之间的区段
  y=y+1;
}
```

上述算法求交点部分同样需要注意 2.1 提到的三个原则，以避免出现不完全匹配的错误。这个算法与简单有序边表算法一样也是计算所有边与所有扫描线的交点并存储起来，只不过它把交点分成两类存放。另外它在求交点时用到的快速递推算法与活性边表算法的 x 增量法的思想是一致的。与[1]相比，本文的算法有如下优点：节省大量空间；且具有更简洁的形式；由于考虑到失真问题，避免了漏填水平棱边的情况。但它们在排序开销方面基本相同。

参考文献

[1] 杨学平，从空心汉字到实心汉字—任意区域填充算法，计算机系统应用，1997.11

[2] David F. Rogers，计算机图形学的算法基础，科学出版社，1987

（来稿时间：1997 年 12 月）