两个简单多边形求交的算法

宋立明，闫浩文，王邦松，方爱玲

（兰州交通大学 数理与软件工程学院，甘肃 兰州 730070）

摘　要：采用了双向链表这种数据结构，对两简单多边形的顶点及交点进行存储，在对交点进行插入时，可直接插入链表中，避免了利用单向链表或数组进行存储时对点的重复查找。通过遍历两个顶点、交点混合表，可得到两多边形的交及多边形的顶点，从而得到两多边形的交集。相对其他类似的算法，该算法具有较快的计算速度和较高的效率。

关键词：双向链表；多边形；顶点；交点；交集

中图分类号：TP311　　**文献标识码**：B　　**文章编号**：1672－5867(2011)06－0258－03

An Algorithm to Compute the Intersection of Two Simple Polygons

SONG Li－ming, YAN Hao－wen, WANG Bang－song, FANG Ai－ling

(School of Mathematics, Physics and Software Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China)

Abstract: This algorithm utilizes the two－way list data structure to store the vertices and nodes of two simple polygons. When being inserted, the node can be directly inserted into the list, which can avoid a repeat search comparing with the one－way list and the array structures. The vertex of the intersection polygons can be acquired after the two mixed－point lists are searched, so that the intersection polygons can be obtained. The algorithm owns the rapid calculation speed and high efficiency comparing with the similar ones.

Key words: Two－way list; Polygon; Vertices; Nodes; Intersection

0　引　言

多边形求交，对于地理信息系统中空间叠置分析具有重要的应用价值[1]。目前，国内外不少研究人员对此问题进行了深入研究，提出了很多解决方法，但这些解决方法在时间复杂度、空间复杂度及结构复杂度等方面，都有各自的优、缺点。双向链表与 weiler 算法的树形结构相比，降低了数据结构的复杂性；shamos 算法和 O'Romke 算法只能处理凸多边形，而本算法还可处理凹多边形。因此，继续深入研究多边形求交的问题，具有重要的理论意义和实际意义。

1　算法的相关数据结构

利用双向链表这种数据结构，容易实现对交点的快速插入及对交集多边形顶点的快速搜索，本算法中用到的数据结构如下。

1) 存储多边形顶点的结构如下：

```
typedef struct tagMYPOINT
{
    CPoint p; //点的坐标
    long next; //静态链表中指向后一元素的指针
    long front; //静态链表中指向前一元素的指针
} CMyPoint;
```

通过此结构，能够确定每一个点与相邻点的关系，即点按逆时针排序时，可确定某一个点前、后的点。

2) 存储多边形的结构如下：

```
typedef struct tagMYPOLYGON
{
    CmyPoint*  pVertex; //存储多边形的顶点
    long size; //记录多边形的顶点数目
} MYPOLYGON;
```

3) 存储交点的结构如下：

```
typedef struct tagMYINTERSECTION
{
    CPoint P; //交点坐标
    POINTTYPE type; //交点类型，即是入点还是出点
    long posA; //交点在 A 多边形中的位置
    long posB; //交点在 B 多边形中的位置
} MYINTERSECTION;
```

此结构记录了交点的坐标、类型及分别在多边形 A，B

收稿日期：2010－10－25

作者简介：宋立明(1984－)，男，山东青岛人，地图学与地理信息系统专业硕士研究生，主要研究方向为近景摄影测量与数字图像处理。

中的位置。pos$A$,pos$B$ 分别和 A,B 顶点、交点混合表中的 next,front 存在对应关系。

4) 存储交点集合的结构,用来存储计算所得的交点

```
typedef struct tagINTERSECTIONLIST
{
    MYINTERSECTION*  pIntersection; //交点集合
    long size;  //交点集合大小
} INTERSECTIONLIST;
```

2　算法思想

两多边形的交是由它们的顶点及交点依次连接所围成的闭和多边形。若将两个多边形的顶点分别按逆时针排序,从一个是入点的交点按逆时针方向行进,碰到交点改沿另一多边形的边前进,最终回到起始交点,便得到一个交。再从剩下的没有遍历过的交点,选择一个是入点的交点,继续进行上述操作,直至所有交点都被遍历过,最终可得到两个多边形的交集[2]。

遍历过程中的关键是碰到交点改沿另一多边形的边行进。虽然指定的是逆时针方向,但一些复杂的多边形,从总体上可将其顶点按逆时针方向排序; 局部上,可能是顺时针方向排序的。因此,交点转向时,存在着两个方向。周培德的 Z_{5-4} 算法[3] 中,需要判断这一交点与所在边的一端点组成的线段与另一多边形的交点的个数,当这一交点所在边的一端点在另一多边形内部或者交点个数为偶数个时,便沿着交点到此端点的方向行进。此算法能够得到交集多边形正确的行进路线。但这需要每次遇到交点时都要进行判断,增加了计算量,降低了运算效率。

本算法通过两个双向链表来分别存储两多边形的顶点及交点,并记录它们在表中的位置关系。从一个表中是入点的交点开始行进,碰到交点跳到另一个表中行进,同样碰到交点跳到另一表中行进,直至回到起始交点,得到一个交集。若交点未遍历完,继续进行上述操作。最终,可得到两多边形的交集。

3　实现步骤

以图 1 为例,多边形 A 的顶点按逆时针方向排列顺序为 A_1,A_2,A_3,A_1; 多边形 B 的顶点按逆时针方向排列顺序为 B_1,B_2,B_3,B_4,B_5,B_1。

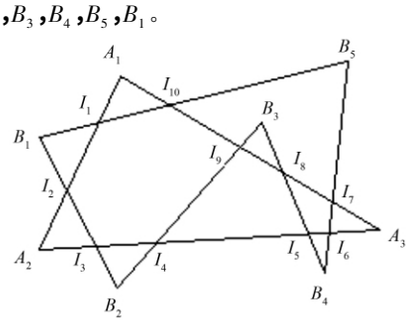

图 1　两个简单多边形

Fig. 1　Two simple polygons

表 1　A 顶点链表

Tab. 1　The vertices list of polygon A

点的位置	0	1	2	3	4	…	…
多边形顶点	A_1	A_2	A_3	A_1	0	…	0
front	-1	0	1	2	-1	…	-1
next	1	2	3	-1	-1	…	-1

表 2　B 顶点链表

Tab. 2　The vertices list of polygon B

点的位置	0	1	2	3	4	5	6	…
多边形顶点	B_1	B_2	B_3	B_4	B_5	B_1	0	…
front	-1	0	1	2	3	4	-1	…
next	1	2	3	4	5	-1	-1	…

1) 将两多边形的顶点按逆时针方向排列的顺序存储到双向链表中,表 1 和表 2 分别表示 A 顶点链表和 B 顶点链表,表中的 front 和 next 分别代表某点的前面一点及后面一点在链表中的位置,如果某点在链表中前面或后面没点,则对应的 front 或 next 记为为 -1。

2) 先做两多边形的包围盒(即最小投影矩形) ,判断两包围盒是否相交,若不相交,则判断两者是包含还是相离关系; 若相交进行以下操作。

3) 为减少计算量,彼此用包围盒过滤掉对方多边形在包围盒外面的边[4]。

4) 从多边形 A 的第一条边开始,依次判断与多边形 B 每一条边是否相交(其中,在第(3) 步中被过滤掉的边可不予判断) ,若相交,计算出多边形 A 的这条边与多边形 B 的交点。若有两个以上的交点,对交点按到多边形 A 的这条边的起始端点的距离从大到小进行排序,依次插入到 A 顶点链表中。直至求出所有交点,并按照上述操作插入到 A 顶点链表中(见表 3) 。

A 表中顶点、交点按逆时针方向排列的顺序为 A_1,I_1,I_2,A_2,I_3,I_4,I_5,I_6,A_3,I_7,I_8,I_9,I_{10},A_1。

5) 从多边形 B 的起始边开始,依次判断哪些所求出的交点在该边上,同样,将在该边上的交点,按到该边起始端点的距离从小到大进行排序,并且插入到 B 顶点链表中(见表 4) 。

B 表中顶点、交点按逆时针方向排列的顺序为 B_1,I_2,I_3,B_2,I_4,I_9,B_3,I_8,I_5,B_4,I_6,I_7,B_5,I_{10},I_1,B_1。

6) 判断多边形 A 的起始点是否在多边形 B 的内部,以确定第一个入点(当其中一多边形从交点处进入另一多边形时该交点的状态为入点[5]) ,本例中第一个入点为 I_1(如图 1 所示) 。

7) 对两表进行遍历,在 A 表中,从第一个入点 I_1 开始行进,碰到交点跳到 B 表中行进。只要碰到交点,就跳到另一个表中行进,直至回到起始交点,便得到第一个交,为点 I_1,I_2,I_3,I_4,I_9,I_{10},I_1 顺次连接构成的闭合多边形; 再从 A 表中未遍历的交点,找到一入点,继续进行遍历,直

至所有交点都被遍历过一次，可得到另一个交，为 I_5，I_6，I_7，I_8，I_5顺次连接构成的闭合多边形。得交集(阴影部分)如图 2 所示。

表 3 A 表

Tab. 3 Tabulation A

点的位置	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	…	…
多边形顶点及交点	A_1	A_2	A_3	A_1	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	0	…	0
front	-1	5	9	13	0	4	1	6	7	8	2	10	11	12	-1	…	-1
next	4	6	10	-1	5	1	7	8	9	2	11	12	13	3	-1	…	-1

表 4 B 表

Tab. 4 Tabulation B

点的位置	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	…
多边形顶点及交点	B_1	B_2	B_3	B_4	B_5	B_1	I_2	I_3	I_4	I_9	I_8	I_5	I_6	I_7	I_{10}	I_1	0
front	-1	7	9	11	13	15	0	6	1	8	2	10	3	12	4	14	-1
next	6	8	10	12	14	-1	7	1	9	2	11	3	13	4	15	5	-1

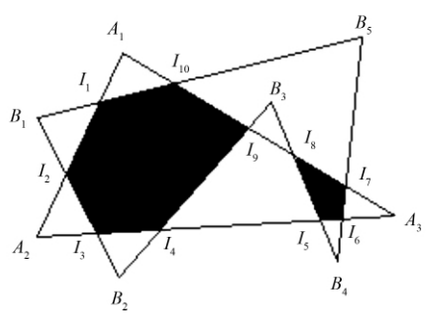

图 2 两个简单多边形的交集

Fig. 2 Intersection of two simple polygons

4 结束语

本算法采用双向链表结构，容易实现对多边形的顶点及交点快速存储及对交集多边形顶点的快速搜索，从而得到交集多边形。本算法中，对求交的两个多边形的要求是：两多边形的顶点不相互重合；一多边形的顶点不在另一多边形的边上。考虑到在实际应用中，两多边形的顶点相互重合和一多边形的顶点在另一多边形的边上这两种情况不多，但这两种情况的类型较多，应用本算法暂时还不能解决此两种情况，需要在进一步研究中实现。

参考文献：

[1] 于雷易，边馥苓，万丰. 一种多边形交、并、差运算的有效算法 [J]. 武汉大学学报(信息科学版)，2003，25(5)：615 - 618.

[2] 吴立新，史文中. 地理信息系统原理与算法 [M]. 北京：科学出版社，2003.

[3] 周培德. 计算几何(第二版) [M]. 北京：清华大学出版社，2005.

[4] 李海姣，张维锦. 用 VC ++ 实现的任意多边形剪裁算法 [J]. 计算机应用，2005，25(12)：421 - 423.

[5] 杜爽，陈成永. 以节点操作实现多边形求交的算法 [J]. 测绘通报，2007(10)：21 - 24.

[编辑：宋丽茹]

(上接第 257 页)

4 结束语

本文推导了全站仪坐标法监测基坑水平位移的精度计算公式，并进行了计算与分析。通过分析结果与工程实践，验证了对于不同等级要求的基坑水平位移监测，只要选择适当的全站仪进行作业，即能保证精度符合要求，提高作业效率。随着测量技术的发展和仪器的不断更新，全站仪在基坑水平位移监测中必将得到越来越广泛的应用。

参考文献：

[1] 金建平，赵仲荣. 自由设站法在深基坑水平位移监测中的应用与分析 [J]. 勘察科学技术，2008(5)：55 - 58.

[2] 华锡生，黄腾. 精密工程测量技术及应用 [M]. 南京：河海大学出版社，2002.

[3] 中国有色金属工业协会. GB 50026 - 2007 工程测量规范 [S]. 北京：中国计划出版社，2007.

[责任编辑：王丽欣]