用游动法作多边形的 BOOLEAN 运算

杨学平　（中国科学院软件所　100080）

摘要：本文提出用游动算法代替求交算法来做两个多边形的 BOOLEAB 运算的方法，以此避开了通常算法中难以处理的退化情况。

关键词：多边形 BOOLEAN 运算　多边形的和、差、并　游动算法　计算机图形学

一、前言

在体素造型中，BOOLEAN 运算是核心部分之一，从原理上讲，这是集合论的问题，但在计算机上实现时，考虑到正则性问题，我们多用求交的方法，这些方法在许多论著中均有叙述，如[1]。这些算法存在的一个问题是难以处理退化情况，如图 5，在一些算法中都会得出错误的结果。因此本文讨论一种改进算法：使用计算机图形学中善长的游动算法。

现设有两个平面多边形 A 与 B，要求作 BOOLEAB 运算，即两个多边形的和、差、并。为了讨论原理，我们假设 A 与 B 都是单连通、无洞的简单多边形，因为从以下算法可以看出，算法本身扩展到多连通、有洞的多边形进行运算是毫无困难的。为了一般性，我们对 A 和 B 的凹凸性不作限制，但要求它们在顺序排列顶点时按相同的方向排列，要么都按顺时针排列，要么都按逆时针排列。

二、予备

两条直线段 $L_1(x_1, y_1, x_2, y_2)$ 和 $L_2(u_1, v_1, u_2, v_2)$：

$$\begin{cases} x = x_1 + t(x_2 - x_1) \\ y = y_1 + t(y_2 - y_1) \end{cases} \quad 0 \le t \le 1 \qquad (1)$$

$$\begin{cases} x = u_1 + s(u_2 - u_1) \\ v = v_1 + s(v_2 - v_1) \end{cases} \quad 0 \le s \le 1 \qquad (2)$$

令

$$\triangle = (y_2 - y_1)(u_2 - u_1) - (x_2 - x_1)(v_2 - v_1) \qquad (3)$$

当 $\triangle \neq 0$ 时，(1)和(2)联立方程组有解，相应地可计算出交点的参数 t 和 s。若 $0 \leqslant t \leqslant 1$ 并且 $0 \leqslant s \leqslant 1$，则线段 L_1 与 L_2 有一个交点，否则，尽管 L_1 与 L_2 所在的直线有交，但 L_1 与 L_2 却不相交。

当 $\triangle = 0$ 时，这两条直线段所在的直线或者重合或者平行，这只须将 (x_1, y_1) 代入(2)检查是否满足(2)即可得出结论点：若 (x_1, y_1) 不满足(2)，则 L_1 与 L_2 平行；若 (x_1, y_1) 满足(2)，此时 L_1 与 L_2 同在一直线上，我们将 (x_1, y_1)、(x_2, y_2) 分别代入(2)计算对应的参数 s_1、s_2，若

$$\min(s_1, s_2) > 1 \qquad (4)$$

或

$$\max(s_1, s_2) < 0 \qquad (5)$$

(5)则 L_1 与 L_2 实际上无交。若有一个 $s_i = 0$ 而另一个 $s_j < 0$，或者一个 $s_i = 1$ 而另一个 $s_j > 1$，则 L1 与 L2 有一个交点(在端点上)。除以上情况外，L_1 与 L_2 部分或全部重合，此时我们进而将 (u_1, v_1)、(u_2, v_2) 分别代入(1)得参数值 t_1、t_2。若令

$$\bar{t}_i = \begin{cases} 0 & \text{若 } t_i < 0 \\ t_i & \text{若 } 0 \le t_i \le 1 \\ 1 & \text{若 } t_i > 0 \end{cases} \qquad (6)$$

$$\bar{s}_i = \begin{cases} 0 & \text{若 } s_i < 0 \\ s_i & \text{若 } 0 \le s_i \le 1 \\ 1 & \text{若 } s_i > 0 \end{cases} \qquad (7)$$

那末我们说，L_1 与 L_2 有两个交点，对应的参数值分别是 $\bar{t}_1$、$\bar{t}_2$，$\bar{s}_1$、$\bar{s}_2$。

定义 1 直线段 L_1 与 L_2 如果存在一个交点，则称存在单交点；若部分或全部重合，则称存在双交点，对应的参数值由(6)、(7)确定；其他情况为无交点。一条直线与其他所有直线求交时产生的单交点和双交点的全体称为此线段的全部交点。

以下考虑的两线段求交点都在此意义下进行。

现在考虑满足我们在前言中要求的两个多边形 A 和 B。将 A 和 B 的每条棱边逐条相互计算交点，对每一条棱边，收集其所有交点，剔除重点(包括与端点重合的交点)，按各点参数值由小到大排序，并嵌入原多边形相应棱边中，我们得到原来多边形的一个新表示，这个过程称为多边形的重新分割。如图 1，多边形 $A = A_1A_2A_3A_4$，经 B 的重复分割后，就变成 $A^* = A_1P_1P_2A_2A_3P_3P_4A_4$。

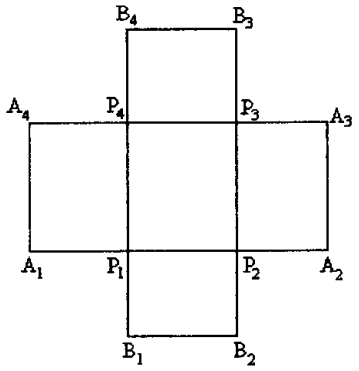

图 1　多边形的重新分割

定义 2 多边形 A* 被称为多边形 A 对 B 的分段多边形，指的是用 B 重新分割 A 后形成的包括 A 的原有顶点及分割点的多边形。

为简单起见，称 A* 为 A 的分段多边形。图 1 中多边形 $A_1A_2A_3A_4$ 的分割多边形为 $A_1P_1P_2A_2A_3P_3P_4A_4$。重新分割后多边形的顶点顺序直接继承了原多边形的顶点顺序。

三、A∪B 算法

A∪B 的算法如下：

1. 计算 A 和 B 所有棱边的相互交点；

2. 若 A 和 B 所有棱边不存在交点，那末：

①若 B 的第一顶点在 A 内部（不包括边界，下同），则 B⊂A（内含），于是 A∪B = A（图 2）

②若 A 的第一顶点在 B 内部，则 A⊂B（内含），于是 A∪B = B（图 3）

③若以上两种测试均不成立，则 A 与 B 分离，于是 A∪B = A 和 B（图 4）

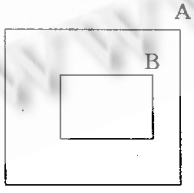

图 2　B⊂A

3. 如果 A 和 B 所有棱边均不存在单交点只存在双交点，那末：

①若 B 中有一棱边中点在 A 内部，则 B⊂A（内接），于是 A∪B = A

②若 A 中有一棱边中点在 B 内部，则 A⊂B（内接），于是 A∪B = B

③若以上两种测试均不成立，则 A 与 B 外接，具体结果需转入以下正常求和计算。

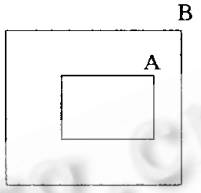

图 3　A⊂B

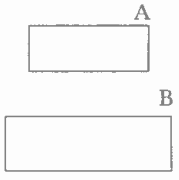

图 4　A 与 B 分离

4. 计算 A 的分段多边形 A* 和 B 的分段多边形 B*；

5. 顺序地取 A* 的每一条棱边中点，检测它是否在 B 外，形成一个棱边属性记录 ICHAIN，每一个记录对应一条棱边，若该棱边中点在 B 外，则相应的记录为 1，否则为 0；对 B* 也同样形成棱边属性 JCHAIN：若棱边中点在 A 外，则记录为 1，否则为 0；

6. 在 ICHAIN 中寻找第一个为 1 的记录，取其对应的棱边的第一点作为游动的起始点（x0，y0），依 A* 的顶点顺序向前游向第二点，并将对应的 ICHAIN 中的记录清为 0；

7. 在 A* 中游动过程中，如果没有碰到 A*、B* 的公共顶点，则连续地在 A* 中依顶点顺序向前游动，并同时将经过的 A* 中的棱边属性记录清为 0。

将经过的 A* 或 B* 的棱边收入 A∪B 的结果中（下同）

8. 在 A* 游动过程中，如果碰到 A*、B* 的公共顶点，则去检查 B* 的下一条棱边的属性记录，若为 1，则游动转入 B* 中进行：依 B* 的顶点顺序向前游动，若为 0，则继续在 A* 中向前游动。

9. 在 B* 中游动过程中，如果碰到 A*、B* 的公共顶点，检查 A* 的下一条棱边属性记录，若为 1，则游动转入 A* 中进行：依 A* 的顶点顺序向前游动，若为 0，则继续

在 B* 中向前游动；

10. 当游动回到(x0, y0)时，则 A∪B 的这一分枝结束；

11. 扫描 A* 的棱边属性记录 ICHAIN，看是否还存在"1"的记录，若还有，找到其在 A* 中对应的棱边，从以上第 6 款开始游动，计算 A∪B 的第二分枝，…，继续这过程，直至 ICHAIN 全为 0；

12. 很显然，上述游动结果，如果得到的一个分枝其顶点排列顺序与 A 的排列顺序相反，则表明这是一个洞。

这个算法很像等值线图格网游动算法[2]。

四、A－B 和 A∩B 的算法

以上算法很容易变形用来作 A－B 和 A∩B。

对 1、2、3 款很容易写出相应的结果，不再赘述。

对 A－B，上述算法唯一需要进行修改的是，在 B* 中的游动时不再是按 B* 原来的顺序，而是反着原来的顺序来游动的即可得到 A－B 的结果。

对 A∩B，只需对 A∪B 算法作两点重要修正即可得到结果：

1. 第 5 款中形成棱边属性记录的原则改为：当 A* 的一条棱边中点在 B 内时，相应的棱边记录为 1，否则为 0；当 B* 的一条棱边中点在 A 内时，相应的棱边记录为 1，否则为 0。

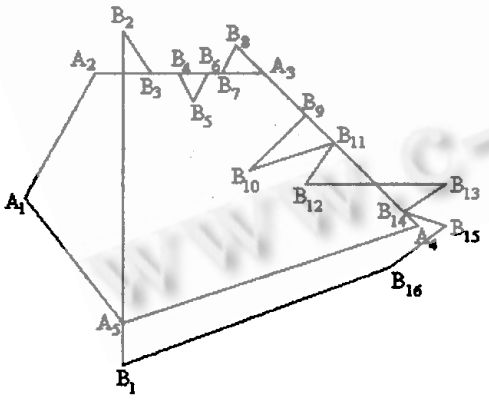

图 5　一个例子的多边形 A 与 B

2. 如果第 5 款中形成的 A* 棱边属性记录全部为 0，即 A⊂B，此时我们将 A 和 B 的角色对调(即从计算 A∩B 变为计算 B∩A)，从第 6 款开始重新计算。

五、例子

多边形 A 由以下 5 个顶点定义：

(30,150),(80,240),(200,240),(310,130),(100,60)

多边形 B 由以下 16 个顶点定义：

(100,30),(100,270),(120,240),(140,240),(150,220),(160,240),(170,240),(180,260),(230,210),(190,170),(250,190),(230,160),(330,160),(300,140),(330,130),(290,100)

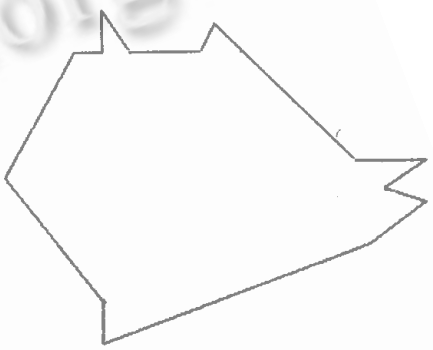

图 6　A∪B

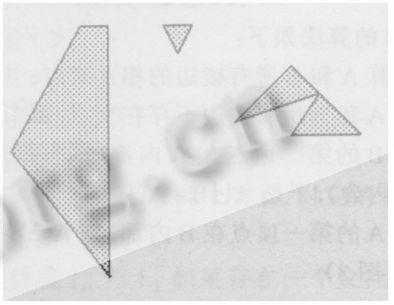

图 7　A－B

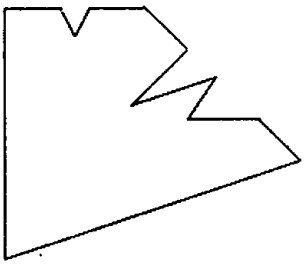

图 8　A∩B

(来稿时间：1998 年 1 月)