判断点是否位于多边形内的算法的简要介绍

杨则瑞

（计算机科学与技术学院，计科 2401 班 24281024）

摘要：简单介绍了判断点是否位于三角形内、多边形内的算法和对应算法的思路。介绍了从判断点是否位于三角形到是否位于多边形内的拓展过程的的思路，以及判断点是否位于多边形内多边形算法的一些具体实现过程。也介绍了一些对特殊多边形。

关键词：图形学；计算机几何学；多边形；

判断点是否位于多边形内是计算机几何学的重要内容。多边形的光栅化、物理仿真的刚体碰撞检测都需要涉及这个判断。多边形光栅化常被用于绘制多边形矢量图到点阵屏幕、绘制 3D 图形程序的三角面，在这些过程中，计算机需要根据指定程序对每一个像素判断像素是否位于多边形内进行判断，然后根据结果对像素进行着色[1]；而物理仿真中，通常需要判断节点是否位于碰撞多边形内，从而判断是否发生碰撞，计算节点的受力情况。

但是，在计算机世界里，多边形的内与外并不是先决地存在的。因为在点只记录坐标和多边形只记录节点和顺序时，点的位置信息里面并不包含着是否位于多边形内，所以需要算法进行判断。本文将分别简单介绍有关三角形和多边形的绘制方法。

一、三角形内部的点的判定

三角形是边最少的多边形，因而具有一些好的性质。为寻找三角形如何把平面分为内外两部分，从而找到三角形内部的点具有的性质，不妨把三边延长，同时取三顶点依次连接的方向（不妨取顺时针方向）作为直线的方向，如图 1 所示。注意到，各个直线分别把整个平面分成了以它为界的两个部分，每半个部分都可以用方向直线的左或右描述。

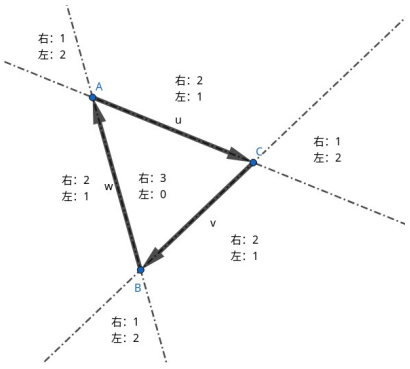

图 1　三角形三边向量划分的区域

三角形的内部的点总是位于三条边的右侧，而不位于任何一边的左侧。这一性质使得我们可以使用外积(叉积)的正负性计算点是否位于三角形内，实现的伪代码如下。

```
if (NumberOfPoints() == 3){
        bool A = ((point - getPoint(0)) X (getEdge(0)) > 0);
        bool B = ((point - getPoint(1)) X (getEdge(1)) > 0);
        bool C = ((point - getPoint(2)) X (getEdge(2)) > 0);
        return !((A != B) or (B != C));
}
```

判断点是否位于三角形内的三角形伪代码

因为判断点之于三角形的位置如此简便，再加之三角形的许多其他性质，使这个算法十分广为人知。许多图形程序，特别是 3D 程序，都利用这样的三角形算法绘制图形(虽然主要是因为三个点确定一个面)；算法研究者也在寻求将边数大于四的多边形拆分为若干三角形的算法[2]，从而简化一些过程。

然而，三边以上的任意多边形并不能相似地简单推广为点位于所有边的同一侧，这可能会使得多边形内的区域不全满足条件(图 2)。事实上，只有边的延长线不相交于除顶点外的点特殊多边形(凸多边形)才可以使用点位于同一侧来判断。

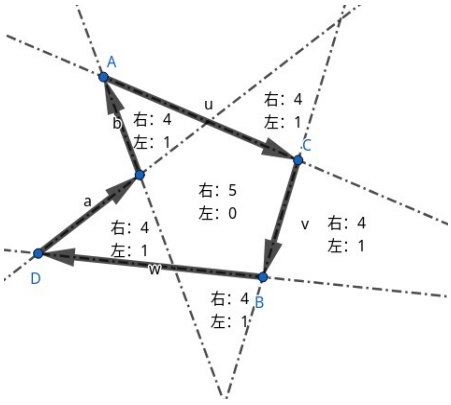

图 2　判断点是否位于多边形内错误情况

二、任意多边形内部的点的判断

1. 思路引入

给定平面上的一个闭合图形，不妨维持一个点 P，使其在平面上移动。此时，点将会多次经过多边形的边界。如果此点经过了偶数次边界，则该点位于多边形之外；如果该点经过了奇数次边界，则该点位于多边形之内。

2. 过程简化

如果需要判断指定点 S 是否位于多边形内，只需让动点 P 从多边形外开始运动，连续运动到 S，统计动点经过的边界数量。为了更加利于计算机程序进行判断，可以简化这个过程。

不妨让动点运动轨迹为一直线，而动点的起点横坐标则始于多边形节点横坐标的最小值。

设：

待判断的点 $S = (x_{dest}, y_{dest})$.

多边形的节点是有序点集 $E = (P_1, P_2, P_3, \ldots, P_n)$.

从而就有令 E 中每个点的横坐标最小值为 x_{min}.

可以让动点 P 从(x_{min}, y_{dest})的位置开始运动，连续的横向运动到 S 。然后统计到 S 点时动点经过了几次边界，便可以判断 S 点之于多边型的内外了。

P 从始点到 S 连续运动的过程中若干次经过边界

有：

对于经过边界的次数：
当其为奇数，
- 说明点在多边形内。

当其为偶数，
- 说明点在多边形外。

3. 进一步简化

然而为了求解经过边界的次数无需声明一个动点，我们只是要想知道动点轨迹有没有经过边，所以需要知道的是动点已经过的边的数量，这个过程可以分为两个步骤。

1. 判断边是否与运动轨迹直线相交:

   动点的运动轨迹所在的直线将平面分成两个部分。现顺次遍历这一个多边形的每一个节点。当节点和上一个节点 不同时位于平面被分成的一个部分的时候，这个边就有可能是动点曾经路过或将要路过的边。

2. 判断与路径相交的边否是动点曾经路过的:

待检测点和出发点如果位于此边的同一侧，那么动点未曾经过此边。

判断过程可以可视化为图 3，伪代码如下。

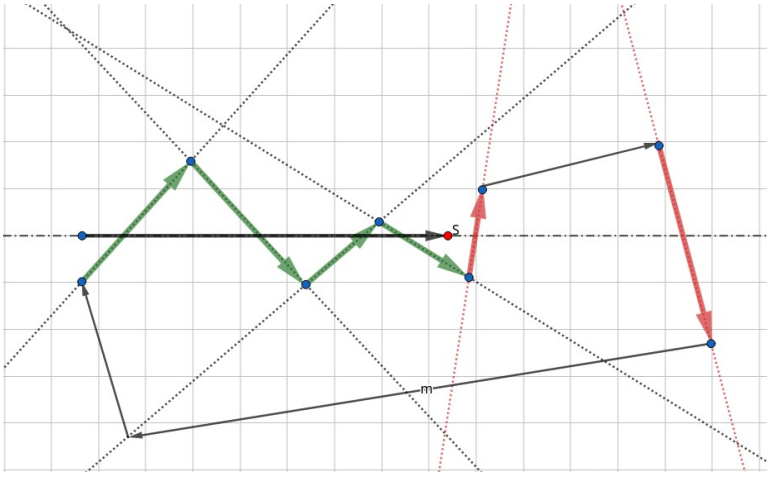

图 3　判断方法图示

```
if (NumberOfPoints() > 3){
    int intersects = 0;
    for (size_t i = 0; i < NumberOfPoints(); i++)
    {
        if ((getPoint(i).y > point.y)
             != ((getPoint(i).y + getPoint(i).y) > point.y)){
            if ((point - getPoint(i)) X getEdge(i) >= 0
                !=( Vector2D( outline.x, point.y)
                   - getPoint(i)) X getEdge(i) > 0){
                intersects++;
            }
        }
    }
    if (intersects % 2 == 1){
        return true;
    }
}
```

判断点是否位于多边形的伪代码

三、特殊多边形

利用这样的算法，可以定义一些特殊多边形的内部的区域。将图形内部着色，可以可视化特殊多边形的内部。

1. 内部有空穴的多边形

对于一个内部有空穴的多边形，可以使用共用边的方法绘制，相当于挖去了多边形的一条厚度为 0 的区域。如下图的字母 b。

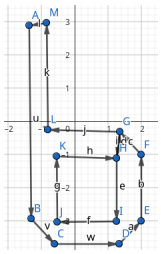

图 4-1　多边形表示的字母“b”

图 4-2　被绘制出来的字母“b”（y 轴反转）

2. 非简单多边形

一个边发生交叉（不包括重合）的多边形并不属于简单多边形。

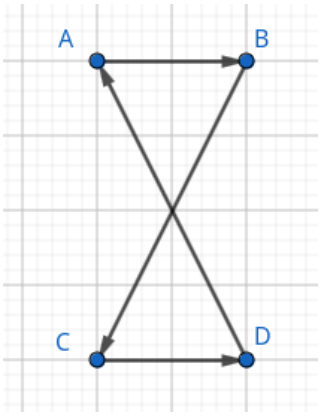

图 5-1　非简单多边形

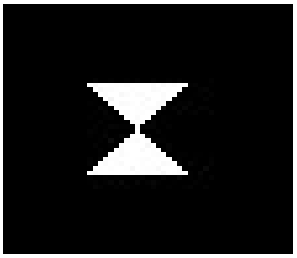

图 5-2　绘制的多边形（y 轴反转）

参考文献：

[1] 闫令琪,计算机图形学入门,https://sites.cs.ucsb.edu/~lingqi/teaching/games101.html.2020

[2] 陈小雕,彭宇,刘玉身.有效的简单多边形三角形化算法的认识与实,dsa.cs.tsinghua.edu.cn.2000.1.16