

# 基于凸包的视锥体裁剪精度优化

曾磊夫, 刘 爽

(电子科技大学 光电信息学院, 四川 成都 610054)

**摘 要:** 针对视锥体对空间中物体的裁剪, 提出一种新的基于凸包的精度优化方法, 该方法首先得到三维物体到二维平面的投影, 然后对投影坐标进行凸包组织, 最后根据凸包上的坐标点和摄像机的连线得到物体的可见性, 整个优化法只需一些额外的存储空间来存储物体的凸包信息即可. 实验结果表明, 该方法易于实现, 能减少通常的包围球、圆柱体检测法中出现的不合理结果, 进而提高整个三维渲染系统的效率.

**关键词:** 三维; 视锥体; 裁剪; 凸包; 投影

**中图分类号:** TP3

**文献标识码:** A

**文章编号:** 1000—7180(2016)01—0085—05

## Precision Optimeization of View Frustum Culling Based on Convex Hull

ZENG Lei-fu, LIU Shuang

(School of Optoelectronic Information, University of Electronic Science and  
Technology of China, Chengdu 610054, China)

**Abstract:** According to the space object frustum clipping, a kind of precision optimization method has been put forward that is based on the convex hull. First of all, the method get the 2d plane projection from 3d object, and organize projective coordinate to a convex hull, finally get the visibility of objects according to the line form the coordinate of convex hull and the camera, whole optimization needs any extra memory to save the data of convex hull only. The experimental results show that this method implement simply, it can reduce the unreasonable result from the sphere detection or cylinder detection, improving the whole efficiency of 3D rendering system.

**Key words:** 3D; view frustum; culling; convex hull; projection

### 1 引言

随着三维图形系统应用的广泛化、规模化, 对实时渲染速度的要求也越来越高, 三维图形渲染中, 待渲染物体的可见性判断是至关重要的一个环节, 它决定了后续步骤中数据处理量的多少, 直接影响整体效率, 是三维图形学中的重要研究领域.

物体的可见性判断主要集中在背面裁剪、遮挡裁剪、视锥体裁剪这三个领域. 其中视锥体裁剪利用摄像机视锥体和物体的相交情况来确定物体的可见性, 有相交则表示该物体可见需要进行绘制, 反之则不进行绘制. 视锥体裁剪的研究主要是速度和精度

两个方面<sup>[1-2]</sup>, 精度方面则侧重于结果的精确程度, 研究的是单个物体和视锥体的判断. 简单几何体当然容易判断, 但是其结果往往不精确, 不精确的结果会导致渲染效率下降. 针对包围球检测法的精度较低的情况, 文献[2]进行了改进, 使用圆柱体代替球体进行检测, 该方法在检测精度上比包围球更好一些, 但是依然有可能出现精度很低的结果. 文献[3]提出一种基于投影的裁剪算法, 这种方法把三维检测转换为二维检测降低了判断难度. 文献[4]介绍了一种基于逐边裁剪的检测法, 它通过计算裁剪边界内法线向量与多边形边的方向向量的夹角来判断顶点是否位于视锥之外, 而文献[5]的方法则是通过判

断顶点射线与平面相交情况来确定顶点与视锥体的相交情况。

针对这种情况,本文提出一种基于凸包的视锥体裁剪法,这种方法只需增加额外的凸包信息存储空间,就能够得到较高精度的判定结果,并且不会增加太多的计算量,同时兼顾了效果和效率。

## 2 优化原理

对于视锥体和物体的裁剪,可以直接在三维空间进行,但是这种处理运算复杂度较高,也可以将视锥体和物体一起先投影到平面上再用进行裁剪,这种处理可以把复杂的三维相交判断转换为相对简单的二维相交判断,然后再根据二维判断的结果确定物体的可见性。本方法的运算过程也是在二维平面上进行的,所以首先要将视锥体和物体都进行投影处理,但是不直接进行几何相交判断,而是通过视锥体投影线,以及物体与视锥体的连线来确定物体是否在视野范围之内,在视野之内即为可见,反之则不可见。如图 1 所示。

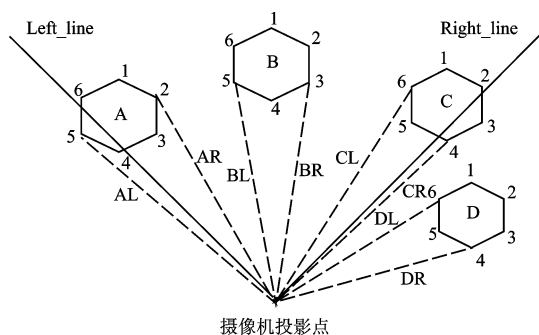


图 1 摄像机投影点和物体投影凸包连线示意图

图 1 所示为摄像机和物体都进行了二维平面投影之后的情况,其中 Left\_line 和 Right\_Line 表示的是摄像机的左右平面在二维平面上的投影线,空间中有 4 个物体分别是 A、B、C、D,图中的不规则几何体就是这 4 个物体的投影结果,另外 AL 和 AR、BL 和 BR、CL 和 CR、DL 和 DR 分别是这些二维投影相对于摄像机投影点的扇形边界线,L 和 R 分别表示左右,注意这些二维投影是按凸包进行组织的。

从图 1 中可以看出,Left\_Line 和 Right\_Line 所夹取的范围就是视野范围,完全在视野范围内的物体只有 B,因为 B 的两条边界线 BL 和 BR 都落在视野范围之内,而 A 和 C 只有部分可见,因为 A

的 AR 线在视野范围之内但是 AL 线不在,C 的 CL 线在视野范围之内但是 CR 线不在,所以算是部分可见,当然只要是可见无论是全部还是部分,都需要送入其后的渲染处理中,所以 A、B、C 三个物体将被绘制。观察物体 D,它的两条边界线 DL 和 DR 都在视野范围之外,所以 D 被判定为不可见,不被送入到后续的处理之中。

可见,只要确定了一个物体投影的左右扇形边界线,就可以通过和视野进行比较确定该物体是否具备可见性,这个比较是角度比较,使用向量的点积即可求得,而且物体的凸包边界相对于包围盒、包围圆等包裹体来说,更能体现物体的外形,可以使得到的判断结果更加精确。另外图中所示的每个物体凸包都有 6 个点,分别为 1 到 6,每个物体的扇形边界线都是这个物体凸包上的两个点到摄像机投影点的连线,但是各个物体对这两个点的选取不一定相同,比如物体 A 是 2、5 点,物体 C 是 4、6 点,具体选择哪两个点就需要通过角度计算得到,并且选取的点并不是一直不变的,因为摄像机和物体可能发生相对的位移,一旦位移就改变了它们的相对角度,所以必须在摄像机位移或者物体位移时,重新计算物体上的两个边界点,这个计算必须是快速的,否则这种方法将不具备实用性。

## 3 处理过程

### 3.1 投影和凸包

一般的图形应用中,都选择  $y=0$  平面作为系统中的水平面,所以要进行投影很简单,把物体上面的所有三维顶点的  $y$  坐标都设置为 0,即可得到该物体中所有顶点在水面上的投影。

接下来就是计算投影点的凸包结构,凸包的构造方法很多,常见的有 Graham 扫描法和 Jarvis 步进法<sup>[6-8]</sup>,进行凸包组织是为了将平面上的物体投影点用面积最小的几何多边形包裹起来,凸包相比其他包裹体更能体现物体的外形特征,如图 2 所示。

图 2 中有 3 个物体分别是 A、B、C,下方是摄像机的投影点 p 及其左右投影线,而中间带有箭头的线段表示摄像机的视线。在得到一个物体的凸包之后,就用双向循环链表结构将凸包上面的点按照一定顺序存储起来,顺序可以是顺时针或者逆时针,但不允许乱序存储。如果旋转存储顺序为顺时针,则物

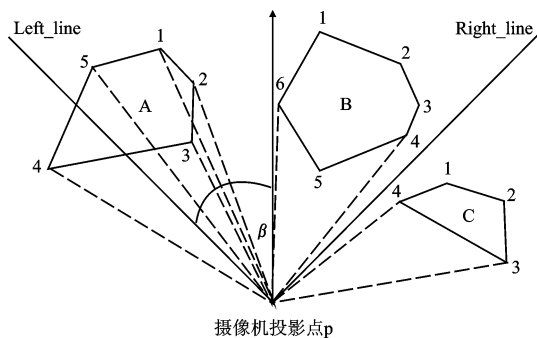


图2 物体的投影和凸包示意图

体 A 的投影凸包存储为 A1 到 A5, 物体 B 存储为 B1 到 B6, 物体 C 存储为 C1 到 C4, 其实选择哪个点作为起点、哪个点作为终点都无关紧要, 只要是按顺时针或逆时针的顺序存储即可。注意存储点的结构一定是双向循环的链表, 因为后面的操作使用这种链表会比较方便。

### 3.2 边界点和可见性

在确定了物体的凸包结构之后, 就需要根据初始摄像机的位置和朝向计算每个物体投影的扇形边界点, 这个信息需要为每个物体都存储单独的一个, 记为 XL 和 XR, X 表示物体的标记, L 和 R 表示顺时针顺序下的左侧边界和右侧边界。

还是观察图 2 所示情况, 图中物体 A 的凸包存储了 A1 到 A5 一共 5 个点的信息, 我们需要的边界点一定在这 5 个点中, 可以通过如下步骤得到。

首先遍历所有物体投影上的点, 得到这些点和摄像机投影点 p 之间连接的线段。如图所示物体 A 就可以得到 p-A1、p-A2、p-A3、p-A4、p-A5 这 5 条线段。

然后将这些线段视为向量, 通过向量的点积分别计算这些向量和摄像机视线向量的夹角, 得到一组角度值, 并将这些角度值进行排序。按照图 2 所示, 物体 A 凸包中 5 个点构成的向量和摄像机视线向量构成的夹角从小到大分别是:  $\angle p-A2$ 、 $\angle p-A1$ 、 $\angle p-A3$ 、 $\angle p-A5$ 、 $\angle p-A4$ 。

在得到的夹角顺序中, 最小的和最大的两个夹角所属的点就是物体凸包的扇形边界点, 所以物体 A 的扇形边界点是 A2 和 A4。

把边界点向量和摄像机视线向量构成的夹角, 记为  $\angle left$ 、 $\angle right$ , 和摄像机左右投影线构成的夹角的一半, 记为  $\beta$  的角度进行比较, 如果这两个角度中有一个角度小于或两个角度都小于  $\beta$ , 那

么这个物体就部分或者全部位于视线之内, 即是可见的。

图 2 中, 物体 A 的  $\angle left$  和  $\angle right$  分别是  $\angle p-A4$  和  $\angle p-A2$ , 其中  $\angle p-A2$  小于  $\beta$  而  $\angle p-A4$  大于  $\beta$ , 所以物体 A 为部分可见。物体 B 的  $\angle left$  和  $\angle right$  分别是  $\angle p-B6$  和  $\angle p-B4$ , 这两个角度都小于  $\beta$ , 所以 B 为完全可见。物体 C 的  $\angle left$  和  $\angle right$  分别是  $\angle p-C4$  和  $\angle p-C3$ , 这两个角度都大于  $\beta$ , 所以 C 为不可见。

### 3.3 摄像机的变换

基于三维渲染系统的应用中, 很少有摄像机位置和朝向固定不变的, 一般情况下都允许用户控制摄像机的位置和朝向, 以确定如果和三维场景进行观察, 另外三维空间中的物体可能根据业务需求, 会发生位移或者旋转的。

一旦发生摄像机或者物体发生变化, 那么在初始化阶段计算得来的物体扇形边界点就有可能失效, 也就是说物体投影凸包上面的点和摄像机投影点的相对位置发生了变化, 这个变化可能是摄像机产生位移或朝向变化造成的, 也可能是物体位移或旋转造成的。再观察图 1 中的 A、B、C、D 这 4 个投影凸包, 这些凸包的形状完全一样, 若假设摄像机的位置和朝向没有变化, 那么也可以将 A、B、C、D 这 4 个投影凸包看做是同一物体在空间中不同位置下的投影形成的, 处于 A 位置时扇形边界点为 A5 和 A2, 处于 B 位置时扇形边界点为 B5 和 B3, 所以必须在渲染系统运行过程中, 根据物体或者摄像机的变化及时更新物体的扇形边界点, 以保证视锥体裁剪的正确性, 如图 3 所示。

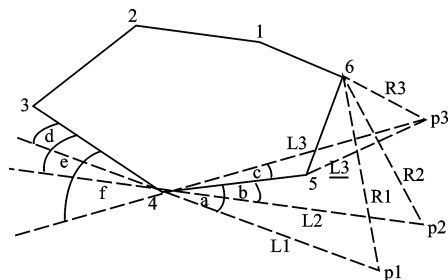


图3 摄像机位移中投影点与凸包点连线示意图

图中 p1、p2、p3 三个点分别代表摄像机在三处不同位置上的平面投影, 设其移动顺序是从 p1 移动到 p2, 再从 p2 移动到 p3。移动过程中 L1、L2、L3 三条线段分别表示顺时针情况下物体投影凸包左侧的

扇形边界,  $R1$ 、 $R2$ 、 $R3$  则表示的是右侧的扇形边界。

当摄像机在位置  $p1$  的时候, 物体的扇形边界点分别是凸包点 4 和凸包点 6, 此时  $L1$  和物体凸包上的线段 4-5、3-4 的夹角分别为  $a$ 、 $d$ , 当摄像机从位置  $p1$  移动到位置  $p2$  的时候, 扇形边界线  $L2$  和物体凸包上的线段 4-5、3-4 的夹角分别为  $b$ 、 $e$ , 可以看出随着位置的变化扇形边界线和凸包上线段的夹角也会产生变化, 夹角  $a$  变成了夹角  $b$ , 夹角  $d$  变成了夹角  $e$ , 虽然夹角大小有改变, 但是夹角的同侧性没有变,  $a$ 、 $b$  相对于线段 4-5 来说是属于同一侧,  $d$ 、 $e$  相对线段 3-4 也是属于同一侧, 这次位移并没有造成扇形边界点的变化, 无论是  $p1$  位置还是  $p2$  位置都是 4、6 两个点作为扇形边界点。

当摄像机从位置  $p2$  移动到位置  $p3$  的时候, 扇形边界线  $L3$  和凸包线段 4-5、3-4 的夹角分别为  $c$ 、 $f$ , 这次的变化和前次有不同的地方, 从角度变换上看, 摄像机在位置  $p2$  时的夹角  $b$ 、 $e$  分别变成了  $c$ 、 $f$ ,  $e$  和  $f$  两个夹角依然是在线段 3-4 的同侧, 但是  $b$  和  $c$  两个夹角变成了在线段 4-5 的异侧, 这次  $b$ 、 $c$  的角度变化就改变了摄像机投影点和物体凸包的左侧扇形边界, 夹角从线段的一侧变为线段的另一侧意味着原有的边界点不再成立, 因为夹角异侧意味着扇形边界和凸包上的线段有相交的情况产生, 这是和扇形边界线的定义违背的。

如图 3 中所示, 当摄像机位于位置  $p3$  的时候, 应该选择点 5 作为左侧边界点, 左侧边界线也应该从  $L3$  修改为  $L3$ , 而 3-4 线段上的角度变化没有引起异侧的情况, 所以不会把点 3 作为新的左侧边界点。

根据上述情况的观察和分析, 可以得到扇形边界点更新的步骤如下:

首先存储每个物体的扇形边界点, 记为  $border$ , 并设边界点的两侧线段的另一端端点分别为  $border-left$  和  $border-right$ , 然后扇形边界线和扇形边界点两侧线段的夹角, 分别记为  $\angle left$  和  $\angle right$ ;

在渲染系统运行过程中, 一旦有摄像机或者空间中的物体产生位移或者角度的变化, 就重新计算  $\angle left$  和  $\angle right$ , 一旦新的夹角和旧的夹角相对于其所属线段产生了异侧的情况, 就更改扇形边界点为产生异侧情况的那条线段的另一端端点, 比如旧

的  $\angle left$  和新的  $\angle left$  有异侧情况产生, 就要将  $border$  更新为  $border-left$ . 如果两侧线段都没有产生异侧情况就不用更新;

角度的同侧异侧情况可以用三维向量的叉积来判断, 平面中的点只需设置  $z$  坐标为 0 即可, 若两次扇形边界线和线段的叉积结果中  $z$  值同号, 就表示同侧, 若  $z$  值异号, 就表示异侧, 此时更新扇形边界点即可。

## 4 实验与分析

本文所述投影凸包裁剪法和圆柱体投影裁剪法进行比较实验。

从执行的精度上看, 本文方法远远高于圆柱体投影裁剪法, 因为凸包相比包围圆, 更能体现物体的外形特征, 更高的精度将更有效地阻止不可见物体进入后续渲染步骤, 而后续步骤要到光栅化才会再进行可见性判断, 中间包含了顶点变换、背面拣选、光照运算等步骤, 在这些步骤中处理不被视锥体看见的物体将极大降低整体效率。

在 CPU 为 P4-1.6 GHz, 内存 1 GB 的硬件环境下, 对两种视锥体裁剪法进行比较实验, 实验结果如图 4 所示,  $x$  轴为三维场景中的三角形数, 单位是个,  $y$  轴为渲染对应数量三角形的时间, 单位为毫秒。

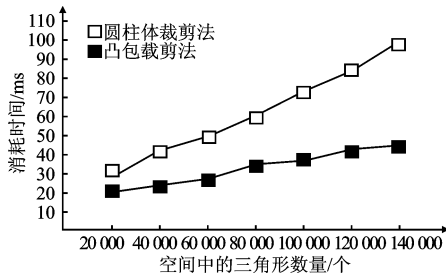


图 4 对比实验结果数据示意图

从实验结果可以看出, 随着空间中物体三角形分布越来越多, 圆柱体裁剪法的误判断就越多, 其后的渲染也就越慢, 凸包裁剪法因为相对精确的结果, 阻止了更多的不可见物体进入后续处理, 虽然单独比较二者, 圆柱体裁剪法的速度高于凸包裁剪法, 但是从最终整体结果看, 凸包裁剪法的精确裁剪结果带来的效率提升远高于它自身运算中导致的效率开销, 总的来说, 它为整体渲染流程带来的好处明显优于圆柱体裁剪法。

5 结束语

综上所述,凸包裁剪法只需在初始化阶段计算物体的投影凸包,然后用少量的额外存储空间来存储扇形边界点,在其后的系统运行过程中,只需要根据扇形边界线和视线的夹角就可以判断物体的可见性,一旦摄像机和物体发生变化,也只需要两次角度同侧异侧判断即可更新扇形边界点,消耗的处理时间很短。

从时间复杂度上看,凸包裁剪法比圆柱体裁剪法更复杂,但是在判断结果的精度上比圆柱体判断要高,这就意味着在同等条件下,凸包裁剪法能够过滤掉更多的不可见物体,阻止更多的无效三角形进入后续渲染步骤,提升渲染的整体效率。

参考文献:

[1] 王竹,束搏,邱显杰,等. 基于场景结合的大规模动态群体可见性计算方法[J]. 计算机辅助设计与图形学学报,2009,21(3):331-338.

[2] Andre LaMothe. Tricks of the 3D game programming

gurus:advanced 3D graphics and rasterization[M]. Indiana,USA;Sams Publishing,2003;531-538.

[3] 王超一,丁莹,张鑫,等. 一种基于投影的视锥体裁剪算法[J]. 系统仿真学报,2014,26(10):2412-2417.

[4] 吴思,郭立,郭安泰. 3D图形的裁剪算法及硬件实现技术研究[J]. 电子技术,2008,46(2):44-48.

[5] 黄建国,黄穗. 一种新的视截体裁剪算法[J]. 微计算机应用,2009,30(12):18-22.

[6] 吴文周,李利番,王结臣. 平面点集凸包 Graham 算法的改进[J]. 测绘科学,2010,35(6):123-125.

[7] 王杰臣. 2 维空间数据最小凸包生成算法优化[J]. 测绘学报,2002,46(1):82-86.

[8] 刘光惠,陈传波. 求解简单多边形和平面点集凸包的新算法[J]. 计算机科学,2007,34(12):222-226.

作者简介:

曾磊夫 男,(1981-),硕士研究生. 研究方向为计算机图形、光电通信. E-mail:278984134@qq. com.

刘 爽 女,(1970-),博士生导师. 研究方向为光通信与集成光学、光电探测、光电能源技术

(上接第 84 页)

[5] Hong Jiang, YuQing-song, Gong Yu. An improved ant colony clustering algorithm[C] // Biomedical Engineering and Informatics (BMEI), 2010 3rd International Conference on pages. Yantai;IEEE;2010; 2368-2372 .

[6] 王小华,沈杰,王荣波. 一种新的基于蚁群和凝聚的混合聚类算法[J]. 杭州电子科技大学学报,2012, 30

(1):26-29.

作者简介:

张海涛 男,(1990-),硕士生. 研究方向为数据挖掘. E-mail:937885321@qq. com

周爱武 女,(1965-),副教授,硕士生导师. 研究方向为智能计算、信息系统安全与数据库与 WEB 技术.