

歐洲图形国际會議文獻

上 冊



国家机械委武汉计算机外部设备研究所

一九八七年六月

1726
TP3

目 录

1. 一种 CSG表优先级隐面算法	(1)
2. 一种射线跟踪的照明模型	(14)
3. 一种改进的跟踪算法	(23)
4. 三维自然纹理的模型驱动合成法	(34)
5. 对于三维表面上纹理合成法局部坐标系的确定	(48)
6. 作为计算机动画分级过程模型的子演员数据类型	(55)
7. 利用二叉树转换描述实体模型	(63)
8. 用精确的八叉树表示多面体的几何模型	(75)
9. 便于航空电子显示软件开发的综合图形环境	(87)
10. BICEP--二维几何造型、排料、及气割技术的交互图形 辅助系统	(98)
11. 表示图象系统的流水线方式	(108)
12. 图象综合中轮廓跟踪的推广与应用	(118)
13. 二值位图图象的光栅旋转	(130)
14. 查找表图形插值	(144)
15. 贝齐三次曲线Timmer混合函线的推广	(157)
16. 建立在Peano空域填充曲线基础上的图形数据库	(161)
17. 多维图形	(175)

一种CSG表优先级隐面算法

荷兰 技术工业大学工业设计系

F · W Jansen

显示结构立体几何学 (CSG) 定义的几何模型的算法有多种,但都是建立在射线跟踪法,扫描线, Z-缓冲器这样的标准隐面算法的基础上。本文介绍的CSG表优先算法与一种射线跟踪法结合起来使用,这种射线跟踪法采用了空间再分技术和使用CSG相关性的CSG表结构。这种综合性的算法用来观察并用多面图元高质量地描绘出CSG模型。

引 言

光栅图形在几何造型中的两种用途是很容易区分的,这两种用途是:

1. 交互式观察: 为支持输入和交互设计模型而进行的物体直观表示。
2. 描绘高质量图形: 用于显示的逼真图形。

对于描绘高质量图形,射线跟踪法可获得最佳结果。因为图形用于显示,产生一张图象所需时间的长短不很重要,所以射线跟踪法的计算速度能够被人接受。

然而,对模型的交互式设计来说,图形的生成时间却是很重要的。因此往往采取舍弃质量而求得速度的办法并运用“预观察”的概念,即在设计过程的不同阶段简化图形,并以不同的误差值来逼近,用户可能会要求在设计过程的每一阶段产生若干图形,以便从不同的观点观察模型或用几张截面图检查外部结构和内部成分。如果预处理(预排序)能提高,同时产生几幅图的图形产生速度,倒也值得在预处理过程中花一定的时间。

鉴于以上这两种概念,我们为观察多面模型而研究出CSG表优先算法。本文后面几节将介绍此算法。

在第2节,我们评述一下现有的隐面算法。在第3节中介绍作为此算法数据结构基本的表结构。在第4节中介绍CSG表优先算法。这种算法结合射线跟踪算法,采用造型数据的预处理技术,CSG表结构并运用了一个加强射线跟踪处理的项目缓冲器,第5、6节给出一些实验结果和结论。

2、立体造型的隐面算法

2.1 立体造型

立体造型中的主要表示方法是结构实体几何表示法 (CSG rep), 这种表示法把物体定义成为一种初始模型的组合,以二叉树形式存贮,图元为树叶,布尔运算(合、交、差)为节点。这些图元可以是圆、块、圆柱、锥体圆凸形式等基本有限体,也可以是些由界面,扫描

或程序表示定义的物体。

CSG是一种很适于输入和存贮的表示法，如果在实际运用中不使用几何模型，则需要把它转换成“单体”表示法。

我们把CSG表示法或组合体转换成单体表示法的过程称为CSG赋值。

常用的赋值方法有以下两种：

1. 边界赋值法：即使各图元的面相交而把有效的面归并成一个新模型，以此将DSG表示法转换成一致的边界表示法。

2. 空间赋值法：即把CSG表示法转换成一种空间枚举法或块模型。特殊算法可以通过询问具有点、线、面网格的组合物体而实现。

空间赋值法仅以给定的分辨力，对边界和物体抽样，而边界赋值法则要对实体的边界进行全面描述。

2.2. CSG模型的隐面算法：

对于组合物体以两种赋值方法，可使用以下两种实现CSG模型化的方法：

①首先应用边界赋值法，然后再用一个标准的隐面算法，描绘出最终产生出来的物体。

②应用一个实施空间赋值的隐面算法，局部地解决有效可见面的问题。

边界模型的隐面算法可分为投影法和非投影法两种，前者有扫描线，Z缓冲器，表优先算法等，后者有射线跟踪法等。

射线跟踪对于有阴影、镜面反射以及带折射的透明性的复杂图形来说，是一种很有用的技术，但是它很耗费时间，采用投影法图象达不到那么高的质量，但速度要快一个数量级。

这种方法的特点在于能对曲面元素〔5〕排序。在扫描线算法中，对于每条扫描线，在实行排序找出可见范围前须修改有效曲面元素表。

Z缓冲器算法按物体顺序描绘曲面元素，对每个象素作一次颜色浓度的排序。若采用专用硬件，几乎达到实时操作。

表格优先算法以优先递增顺序描绘帧缓冲器中的曲面元素。广为人知深度优先算法针对每个不同的观察点，动态地对曲面元素排序。用一种固定的优先排序〔7〕获得实时操作的性能。上述各方法都有一个预处理过程，对簇中与观察点无关的所有曲面元素，确定一个优先级顺序。硬件以给定的顺序处理各个元素。

以上介绍的方法都需要一种单独的边界表示法，在立体造型中，使用这些方法唯一办法是对CSG模型上首先实施边界赋值。

隐面消除和CSG赋值的结合运用最近已引人们极大注意。

由Roth〔8〕推广的CSG射线投射算法是一种很完善的算法，能够很容易地被应用到原始实体中，另外，用于分析，扫描以及程序定义物体的相交算法也已问世。

Atherlon〔1〕和Crocker的扫描线方法采用的是原始实体平面逼近法。Chung〔10〕已发表了类似二次曲面算法。

Atherlon早期时候发表了一种交互式观察曲面模型的算法。最近Okino等发表了此算法CSG的版本。在这两种方法中，对Z缓冲器进行扩展，为每个象素存贮物体所有的深度信息，Z缓冲器的每个登记项都是一个表的起始单元。在对射线穿透物体进行跟踪时，应采集的信息

存放在起始单元内。这种结构是逐渐地建立的,而且每增补一个因素得对数据结构作一次修改。重要的问题在于是否具有一个CSG表优先算法,即某个初始处理完毕,在帧缓冲器中用排序的最小值描绘多边形,我们先讨论多面模型的隐面算法,再详细谈这个问题。

2.3 多面CSG模型

立体造型中简便灵活的图元是逼近边界模型的多边形。

无论是用多边形逼近多面模型,还是棋盘形模型都能通过对封闭边界模型曲面再分的方法根据程序定义或产生出来。模型由大量的多边形构成,除了因素的关合框以外,不包括任何附加的拓朴或几何信息。整个CSG模型的所有多边形数目经常在1万个以上。

对这么大量数的多边形作射线跟踪,需要有一个辅助的数据结构,最近人们提出了几种空间划分方法。对CSG模型赋值或显现CSG模型。Woodwark和quinlan^[13]采用空间划分方法把空间分成两半,以便减少有限体模型评价的复杂程度;空间划分方法与八叉树结构相似,它是通过不断地反复对空间进行划分形成的。

Tammia^[14]、^[15]介绍了一种空间检索方法,它是把数据与物空间对分面产生出来的格或单元联系起来。现已用这种方法改进了射线跟踪算法^[16]。还有人提出与这种算法相似的算法^{[17][18]}。

射线跟踪的空间再分法是简单明了。具体做法是把物空间划成许多单元,再以桶方式把各个多边形存储到有关的单元中去。接着使射线与这些单元相交并检测每个单元中的各个多边形是否与射线相交。最后对相交的各个点进行排序。通过检测以找出有效的相交点。所谓相交点有效是指相交面为最终产生出来的CSG模型边界的一部份。

正如Atherton表明的那样,对于每个给出的象素来说,只有当遇到第一个有效面元时,才能作CSG赋值。探讨这个概念就会引出这样的结论:假如按优先级递减顺序描绘各元素。或者我们知道CSG赋值过程中涉及到其元素的每一个象素,就可能对每个象素作CSG赋值。

现在让我们回到CSG表优先算法的问题上来,我们可以以优先级递减顺序描绘出帧缓冲器里的多边形,并为每个象素保留那些表示已传送了哪些多边形已作描述,以及这些多边形中哪个是有效的等方面的信息。在解释这个算法之前,先介绍一下我们如何存储这些信息的。

3、CSG表格结构

我们可区别CSG隐法算法中以下两个动作:

1. 找出每个象素的投射点或相交点,并对其排序。
2. 通过对CSG树求值,确定第一个有效点。

CSG树求值仅仅是取决于深度级,因而可以同具有相同相交1投影图素序列的象素一起共享。这样要为缓冲器里的每一个图元序列(表)存储对CSG树求出的值,并对类似的图元作一次查对。我们称这种缓冲器为表结构。表结构见图1。

该结构的根是最大元表元素指针的陈列。图元号是这种陈列的索引,表格可以从任何一项开始,它从阵列索引的图元号开始记录图元号序列。为了使不同的图元序列具有相同起始

的图素。表还可以分支。图2示出了6—7—10(有效), 6—5—8(有效)以及6—8(无效)等序列的实例。

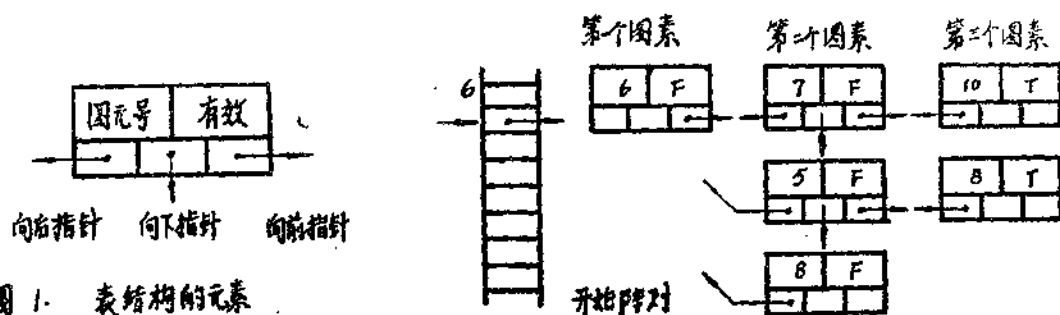


图1. 表结构的元素

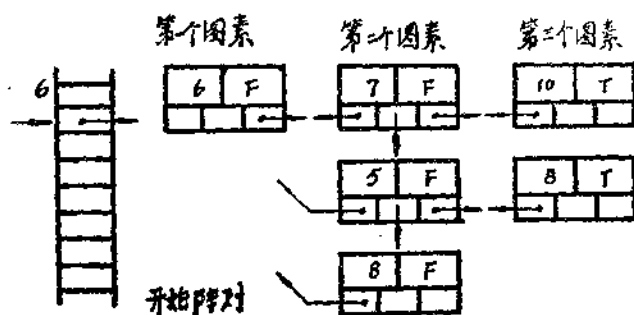


图2. 表结构

表格用第一次出现的图元序列进行修改, CSG树每扩展一次, 就要作一次CSG树的求值。CSG表的求值评价与标准树遍历方法相似, 不同之处是对每个组合只作一次求值。

表结构可用于各个不同观察点。只有当更改CSG定义时才需对表结构重新初始化。对每个象素需存贮一个指向表结构中某一单元的指针。指针最初为零, 当处理完某相交点之后, 更新指针, 使其指向相应的表元素。每处理完一个相交点, 表只要移动一个位置, 直到遇见有效元素为止。对射线跟踪来说, 我们只保留一个指针, 这是因为一次只处理一个象素。而对CSG表优先算法, 我们必须以简单方式存贮每个象素的指针。我们称这个指针缓冲器为C缓冲器。用表结构作象素处理的过程如下: 如果扫描一个多边形和检测一个象素, 可以出现两种情况。第一种情况是: 如果以前对该象素未作检测, 那么这个象素的指针为零。这时, 在起始阵列中进行一次查找。假如指针仍然为零, 图素象第一象素那样以前也未经检测过, 这时产生一个新元素。然后在表元素中进行一次查找, 若该象素有效, 必须要描绘出来, 若无效则跳过去。在上述两种情况下, 指针作修改, 使其指向那个表格元素。

第二种情况象素以前已被检测过, 这时要对表元素作一次查找, 看看有效区域是否为真。如果为真而且象素已被描绘, 就跳过去, 如果有效域为假, 则按照表再作一次查对, 并有必要对分支进行查找, 以便找出具有相同图元号的表元素。若表元素不具有相同的图元数, 则向分支增加一个新的元素, 为了尽可能限制搜索, 把最后搜索的元素重新连接起来, 使之成为下次访问中被测试的第一个表元素。在附录给出一个表处理程序的表格。

4、CSG表优先算法及射线跟踪算法

4.1 概述

我们可以把所有的程序块拼在一起。空间细分法可以被用来改进射线跟踪过程, 并可以用作表优先算法中独立于观察点的三维桶(bucket)排序法。对于每个观察点, 我们须对每个桶中的多边形进行动态排序。

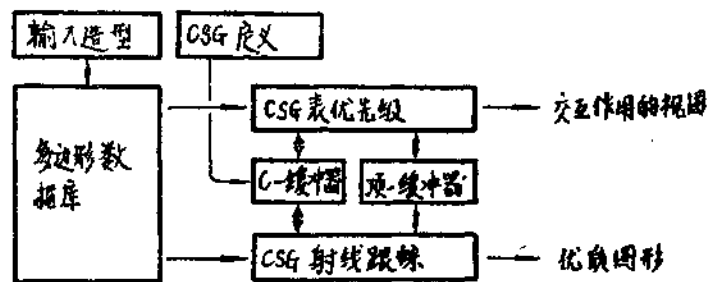


图 3. 图形系统框图

模型输出以及图形产生过程包括：

——预处理

——多边形独立于观察点的桶式排序

——对每个桶作多边形的预处理，以便尽可能地避免相交及优先级冲突。

——表优先算法的观察操作包括

——如果有必要，把顶点转换成眼空间坐标或屏坐标

——从最靠近的桶开始，并以递减顺序进行多边形扫描转换。

一个桶中所有的多边形用简单的深度排序法进行排序，在CSG表格结构中，对每个象素作一次测试。

——每个象素列出的射线跟踪算法：

——单元结构与射线相交，多边形与射线相交，深度排序以及对表结构逐步查对直到找出有效点为止。

设计阶段要重复地使用视见算法，只有当造型数据变化时，才对预处理作修改。在这过程中，多面模型，允许有些偏差。

在最后的图象产生过程中，对画面作最初的预观察后，可使用一个更精确的模型近似值。

4.2 预处理：

对于空间细分，可使用三维EXCELL方法。EXCELL几何数据结构是对物空间作对分所产生的单元结构假设给出一个物体箱，对该箱递归地对分成若干个小格即单元，这些单元可容纳的多边形数目有限，单元内的多边形存贮在一个桶内，当得到的单元达到最小时，则停止对空间的划分。这时，要产生出一个溢出桶存贮该单元的附加的多边形，参考文献^(15, 16)介绍了数据结构，产生这种结构和输出数据的算法。

■预处理过程是从物体箱的计算开始的，物体箱包含所有的图元，然而对图元定比例，实施变换，并用多边形网格逼近多边形。最后把多边形输入到单元结构中。此外，我们还作了相当大的预处理，尽可能避免优先级冲突。对于每个单元，要按单元平面对多边形作剪裁，

并且检查其是否与桶中其它图元多边形相交。具有相同图元的多边形被认为是非相交的多边形。如果要想改变图元实体,则找出数据结构中与图元箱重叠的单元,然后消去具有该图元号的所有多边形。图元可以插入在单元结构的其它位置。

注:预处理只与图元的定义有关,在以下方面不限制视见算法:

- 控制视见参数:从用户规定视点的透视投影和正交投影。
- 控制可见分析的工具:横截面的视图半透明地审查组件及内部成份。
- 控制CSG定义,单个图元实体的视图,视图或对子组件用不用CSG运算都行。
- 控制浓淡参数:用图元或子组件的伪彩色观察,或随光源的不同位置作观察。

4.3 表优先算法:

视见图运算从窗口剪取平面和眼空间变换矩阵的计算开始,由于对单元结构是作正交对分,所以桶的输出顺序很容易被确立,废弃视见锥体面和后面的单元。单元中只有部份在窗口内的多边形要作剪取,每个桶的多边形要进行排序,然后扫描多边形,并在CSG表格结构中对每个象素作一次查找。

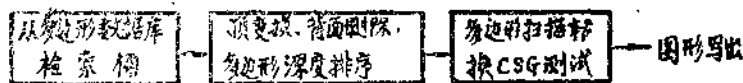


图4. CSG表优先算法

删除那些只有在联运算中才用得上的图元背面多边形,可以减少被扫描多边形的数目。在CSG定义输入过程中对图元进行编码。并用这些编码,在排序之前,作完背面删除。扫描前作一次最初CSG检测后,可根据CSG定义删除更多的多边形。

遗憾的是,这种算法的结果显示出了以下两点不足:

——这种算法不能正确地处理重合面,假如两个面重合,则必须同时考虑这两个重合面,而不是把它们当作赋予最高优先级的一个面来看待。所以在预处理中,对这些表面打上标记,用不同的方法加以处理。

——尽管把曲面分割成许多小块,但仅作简单的深度测试,还不足以消除所有的排序错误。我们只能通过反复测试来改进排序。

然而我们也可通过保留一个附加的Z缓冲器来改变上述的结果,然后,测检两个表面是否重合,或两个多边形是否相交,假设有三个多边形有排序错误,那就不能保证是否能确定正确的顺序,因为我们只存有一个深度值。

最好是表优先算法当作预视见图,把Z缓冲器当作以额外费用进行的一种改进。

4.4 项缓冲器

参考文献^[10]对项缓冲器作了介绍,它是由投影的隐面算法所产生的,对于每个象素,在项缓冲器中存入五个指向该象素上可见的多边形的指针由于主要的射线(从视点看)是投影射线的顶相射线,所以此多边形是与射线相交的第一个多边形。这样,我们对第一条射线不作

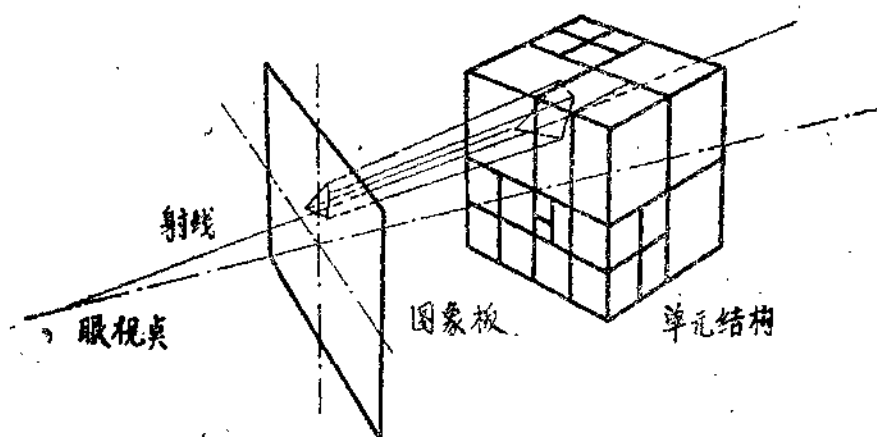


图5 CSG表优先标法为投影的形式
CSG射线跟踪标法为非投影形式

跟踪，只需找出第二条射线的相交点就行了。

对于由多面图元确定的CSG模型来说，只有在用到有关找到第一个(有效)交点前与射线相交的图元序列的信息时才使用项缓冲器。这种信息已被存入在C缓冲器中。这时，CSG表结构提供附加的信息，使该缓冲器可用于CSG射线跟踪算法。

5、实现和结果

5.1 测试结果

为了得到这种算法性能的某个指标，我们用PASCAL语言编制了该算法的程序，微机(带2Mb存贮，不用浮点处理器)上运行。由于有C缓冲器，附加的项缓冲器，Z缓冲器以及多边形的数据，对存贮器容量的要求是很大的。一个复杂的模型至少需要4Mb存贮容量因此我们不得不限制物体的复杂程度，下面示出一些测试物体。

- | | |
|---------------------|----------------------|
| a 图号 | 1. 预处理 |
| b 建模偏差 | 2. 顶变换 |
| c 多边形的最初数目 | 3. 桶检索和多边形排序 |
| d 经过预处理后、数据库中的多边形数目 | 4. 扫描转换和CSG测试 |
| 子图象尺寸: 256 × 256 | 5. 任选的Z缓冲器测试 |
| 桶最大尺寸: 24个多边形 | 6. 总数: 2 + 3 + 4 + 5 |

包括缓冲器初始化不包括写入图形

表1 CSG表优先算法(处理时间以占用cpu秒数计算)

有关EXCELL方法的性能请参见参考文献[16]，在我们的例子中，线性的性能因多边形分割处理而含有偏差。预处理完全取决于初始多边形的数目，处理时间取决于图象大小以及模型的复杂程度，随着模型精确度的增加，总开销会因额外的多边形而有所增长。多边形排序时间随多边形数目多寡成线性增加或缩减。由于对排序一个桶的时间是恒定的，所以桶的数

目完全取决于多边形的数目,因此,我们估计,描述过程长短是完全取决于最初多边形的数目。

为了得到多个复杂模型(10,000个多边形以上)作图时间的真实印象,必须把表1的某些数字乘上50—100,即使我们考虑到VAX780与68,000之间硬件上的差异,很显然,我们表优先算法程序的处理时间并没有对扫描线算法(1)(9)有所改进。

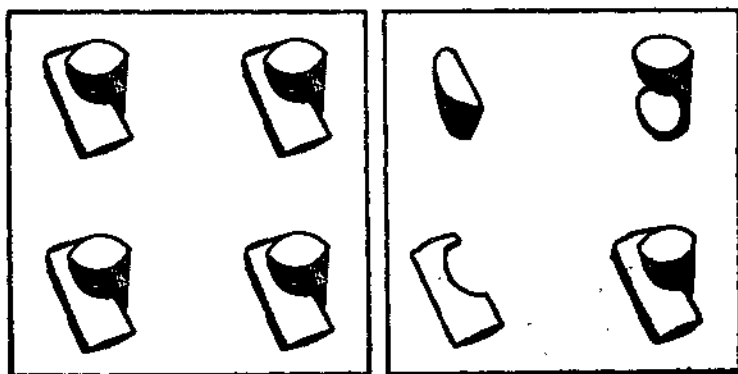


图6. 图7两个圆柱体,以不同精度表示(右边)和用柱圆柱体中的LSG的表示(左)

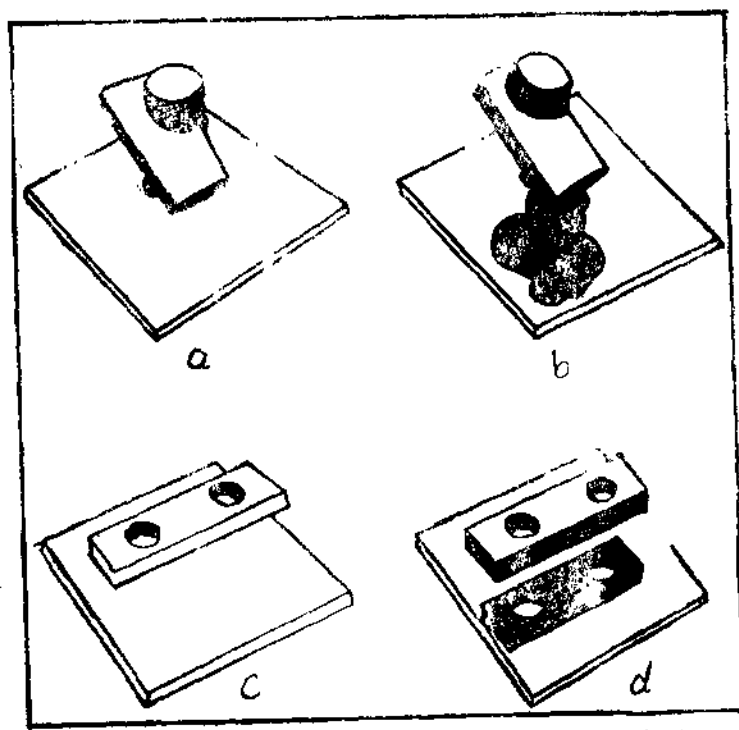


图8: a和c CSG最优先算法 b和d CSG射线跟踪表示

1. 只是主要的射线
2. 带阴影
3. 带阴影以及镜面反射
 - a. 使用/不用项缓冲器, 为每条射线测出的多边形平均数。
 - b. 使用/不用项缓冲器, 与每条射线相交的多边形平均数。
 - c. 使用/不用项缓冲器, 对 256×256 图象的处理时间。

表 2 CSG 射线跟踪算法处理时间 (以占用 cdu 秒数计算)

用项缓冲器得到期望的结果

5.2 讨论:

如果我们把表优先算法与扫描线算法进行比较, 也会注意到, 若不是象使用 Z 缓冲器算法那样在图素级上对所有多边形测试, 就不能减少排序步骤, 对于多面图元来说, 被一个多边形覆盖的图素的平均数目虽然很小, 但总开销时间仍然与多边形的数目成正比。在我们例子中, 多边形的数目由于对多边形进行了划分面变多, 因此, 预处理后的大部份时间化在多边形扫描转换以及图象测试上。在今后的研究中, 我们将探讨针对单独的图形处理中对这些步骤编制程序的可能性。

与扫描线算法相比较, 表优先级算法因作了面向三维单元处理, 而更适合于驻留在辅助存储器中的大多边形模型。

6、结 论

我们已对利用 CSG 相关性的表结构作了描述。这种方法可用于所有 CSG 隐面算法, 我们使用了 CSG 表优先算法的方法, 预观察 CSG 模型, 在做完最初的预处理后使用不同的视图参数, 横截面以及 CSG 定义产生出若干个图象。

本算法能有效把存贮在表结构和项缓冲器中信息的 CSG 射线跟踪算法结合起来使用。

参 考 文 献

- [1] Atherton, P. R., A scan—line hidden surface removal procedure for constructive solid geometry, Computer Graphics 17(1983)3, 73—82, Siggraph 83.
- [2] Wijk, J. J. van, Jansen, F. W., Bronsvoot, W. F., Some issues in designing user interfaces to 3D raster graphics, Computer Graphics Forum 4 (1985) 1, 5—10.
- [3] Jansen F. W., Wijk, J. J. van, Fast previewing techniques in raster graphics, in: P. J. W. ten Hagen (ed), Proc. Eurographics' 83, 195—202.
- [4] Requicha, A. A. G., Representations for rigid solids, theory, methods and systems, ACM Computing Surveys 12 (1980) 4, 437—464.
- [5] Sutherland, I. E, Sproull, R. F., Schumacker, R. A., A characterization of ten hidden-surface algorithms, ACM Computing Surevys 6 (1974)

- 1, . 1—55.
- [6] Newell, M. E, Newell, R. G., Sancha, T. L., A new approach to the shaded picture problem, Proceedings ACM National Conference, 1972, 443—450.
 - [7] Schumacker, R. A., Brand, B., Gilliland, M., Sharp, W., study for applying computergenerated images to visual simulation, AFHRL-TR-69-14, US Airforce Human Resources Laboratory, 1969.
 - [8] Roth, S. D., Ray casting for modeling solids Computer Graphics and Image Processing 18 (1982) 2, 109—144.
 - [9] Crocker, G. A., Invisibility coherence for faster scan-line hidden Surface algorithms, Computer Graphics 18 (1984) 3, 95—102, Siggraph84.
 - [10] Chung, W. L., A new method of view synthesis for solid modelling, CAD84 conference proceedings, 470—480.
 - [11] Atherton, P. R., A method of interactive visualization of CAD surface models on a color video display, Computer Graphics 15 (1981) 3, 279—287, Siggraph81.
 - [12] Okino, N., Kakazu, Y., Morimoto, M., Extended depth-buffer algorithms for hidden surface visualization, IEEE Computer Graphics and Applications 4 (1984) 5, 79—88.
 - [13] woodwark, J. R., Quinlan, K. M., Reducing the effect of complexity on volume model evaluation, CAD14 (1982) 2, 89—95.
 - [14] Tamminen, M., The EXCELL method for efficient geometric access to data, Thesis, Acta polytechnica Scandinavica, Mathematics and Computer Science Series no. 34, Helsinki, 1981.
 - [15] Mäntylä, Tamminen, M., Localized set operations for solid modeling, Computer Graphics 17 (1983) 3, 279—288, Siggraph83.
 - [16] Tamminen, M., Karonen, O., Mäntylä, M., Ray-casting and block model conversion using a spatial index CAD 16 (1984) 4, 203—208.
 - [17] Glassner, A. S., Space subdivision for fast ray tracing, IEEE Computer Graphics and Applications 4 (1984) 10, 15—22.
 - [18] Dippé, M., Swensen, J., An adaptive subdivision algorithm and parallel architecture for realistic image synthesis, Computer Graphics 18 (1984) 3, 149—158, Siggraph84.
 - [19] Weghorst, H., Hooper, C., Greenberg, D. P., Improved computational methods for ray tracing, ACM Transactions on Graphics 3 (1984) 52—69.

附录: CSG测试程序的表格

```

program CSGtest,
{ version without optimizations and without z-buffer test }
type operation = ( union, difference, intersection );
    nodekind = ( composite, leave );
    elementpointer = ^listelement;
    listelement = record
        primnr : integer; { primitive number }
        valid : boolean;
        back, down, forw : elementpointer;
    end;
    nodepointer = ^node;
    node = record
        left, right : nodepointer;
        primnr : integer; { primitive number }
        oper : operation; { +, -, & }
    end;
    nodetype : nodekind
end;
var cbuf : array [ 1.. m, 1.. m ] of elementpointer;
{ c-buffer }
cbufar : array [ 1.. maxprimitive ] of elementpointer;
{ start array of list structure }
function primitivepresent ( pel : elementpointer,
    primitvenr : integer ) : boolean;
    external;
{ primitive is present when uneven number of faces
are encountered }
procedure newelement ( var pel; elementpointer,
    primitivenr : integer,
    backpointer : elementpointer );
{ newelement creates a new list element }
function valid ( pnode : nodepointer ) : boolean ;
{ valid determines if the CSG expression is valid
for a given list of primitive points }
begin { valid }
    with pnode^ do
        if nodetype = composite then

```

```

if valid ( left ) then
  if valid ( right ) then
    valid, = ( oper < > difference)
  else valid, = ( oper < > intersection)
  else { left not valid }
    if ( oper < > union ) then valid, =false
    else valid, =valid ( right )
  else { leave node }
valid, primitivepresent (pel, primnr)
end, { valid }
begin { newelement }
  new ( pel ),
  with pel^ do
    begin
      primnr, =primitivenr,
      back, =backpointer, down, =nil, forw, =nil
    end,
    pel^. valid, =valid ( root )
  end, { newelement }
function validpoint ( var pel , elementpointer,
                      { in C-buffer } prn, integer
                      { primitivenumber } ), boolean,
{ validpoint tests if a pixel is already set or has to be set }
var ploc, elementpointer,
begin { validpoint }
  if pel=nil then { pixel ( x, y ) has not been examined yet }
    begin { first test if primitive has been evaluated }
      if cbufar [ prn ]=nil then newelement ( cbufar [prn], prn, nil ),
      pel, =cbufar [ prn ]
      validpoint, =pel^. valid
    end
  else { pixel has been tested before }
    if pel^. valid then validpoint, =false { already valid point
                                           drawn }
    else { nothing drawn yet, do CSG—list evaluation }
      begin {find next element }
        if pel^. forw=nil then
          begin { add new element to list }

```

```

        newelement ( pel^. forw, prn, pel ),
    , pel, =pel^. forw
    end
else { there is an element }
begin { search in down direction }
    ploc, =pel^. forw,
    while ( ploc^. primnr < > prn ) and ( ploc^. down < > nil ) do
    , ploc, =ploc^. down,
    if ploc^. primnr=prn then pel, =ploc { element found }
    else { primitive number not found }
        begin { append element }
            newelement ( ploc^. down, prn, pel ),
            pel, =ploc^. down,
        end
    end,
    validpoint, =pel^. valid
end { of CSG-list evaluation }
end, { validpoint }
begin { CSGtest }
    { for every polygon do }
    { for every pixel in polygon do }
    if validpoint ( cbuf[ x, y ], primitivenr ) then { setpixel(x, y,
    colour ) },
end,

```

张莉莉译

王世运校

一种射线跟踪的照明模型

法国国家声象通讯研究所 (I.N.A)

Alain Nicolas等

摘 要

本文介绍了一种能够满足射线跟踪程序要求的照明模型。这种模型是James. F. Blinn、Robert L. Cook和Kenneth E. Torrance所描述的关于不光滑金属表面的Torranec-sparrow反射模型的扩展形式。文章着重描述了透明现象的几何造型。在解释了射线跟踪算法提出的约束条件建立的模型间的主要差异之后,又提出了计算每一象素颜色的不同方法。该模型在法国国家声象通讯研究所开发的射线跟踪程序中作为用以产生在P.A.R.I.S(合成图象动画程序)下重新组合的三维图象序列的整体程序的一部分。

一、导 言

目前使用的大多数照明模型都适合于扫描线算法。使用这种算法,对每个象素曲面中的每一可见物体的照度只需计算一次。采用这些照明模型,能够计算出光源发出的光反射到观察者的亮度。这样,就必须引入一个背景的扰动函数或反射到前景中可见物体颜色中物体的扰动函数,以此模拟透明度或镜象效应。

此外,这些算法把象素最后颜色当作以下三种不同的组成光的和来考虑,即:

外来光;

漫射光;

镜面反射光。

事实上,所有这些照明模型都是由下列基本公式推导出来的,即对作为照明函数 $I(L)$ 和反射函数 $R(I, V)$ 之积的照明范围 H 积分,求出一曲面反射光强度。公式为:

$$I = \int_H I(L) \cdot R(I, V) \cdot dL$$

式中, L 是表示照明方向的变量, V 是表示观察方向的变量。

注:图1给出了各符的含义。

根据以下假定可获得上述三种组成光:

若 K 是总的施照光源的数目,那么,照明函数 $I(L)$ 可分成如下 $K+1$ 个函数之和:

— 一个与景的每一施照光源亮度相对应的 K 个Dirac函数 $I_k(L_k)$ 组成的函数群。

—环境亮度 I_e ，它表示不是由任何一个光源发出的光照度。假设外来光同时以相同的照度从各个不同方向照射到景中的所有物体。

——反射函数 $R(L, V)$ 可分解成以下两个函数的和。

— $R_i(L, V)$ 表示从材料表面的直接反射，通常是以类似镜面反射方向 $\vec{L}$ 为中心的一种辐射。 $\vec{L}$ 以曲面的单位法矢量 $\vec{N}$ 为对称轴与照射方向上单位矢量 $\vec{L}$ 对称（参见图2）。

— $R_d(L)$ ，表示与材料相互作用后的反射（内散射，材料不光滑表面的多重反射），与观察方向无关。

如果考虑到入射光中经镜面反射的那部分 S 不能与材料相互作用，反射函数 $R(L, V)$ 为：

$$R(L, V) = S \cdot R_i(L, V) + d \cdot R_d(L), \text{ 其中 } S + d = 1$$

这些简化了的函数可用来阐述三种常用的组成光：

外来光亮度为环境照度与反射函数照明范围积分 R_i 的乘积，表示成：

$$I_A = R_i \cdot I_e = I_e \cdot \int_{H_i} R(L, V) \cdot dL$$

除前面提到的假设条件外，还可用Lambert定律，即入射光通量等于反射光通量，可获得相应的漫射光和镜面反射光的表达式，即：

$$I_D = \sum_{K=0}^{K-1} d \cdot I_K \max(0, \vec{N} \cdot \vec{L}_K) \cdot R_d(L_K)$$

及

$$I_i = \sum_{K=0}^{K-1} S \cdot I_K \max(0, \vec{N} \cdot \vec{L}_K) \cdot R_i(I_K, V)$$

因此，反射模型的实现，将在很大程度上取决于用来计算反射函数中三个组成色亮度的方法，尤其是取决于这些方法接近于现实情况的程度。

Bui-Tuong Phong描述的模型是一种简单的广为人们运用的模型。这种模型把 R_i 和 R_d 的值当作算法的输入常数来考虑，而把 R_i 分布函数比拟为一个简单的余弦。这种模型的主要不足之处在于所有物体看上去是由同一类材料（类似于塑料的材料）构成的。为了克服这一缺点，我们采用了J.Blinn描述的模型。这种模型是根据K.Torrance和E.Sparrow对不光滑表面的光反射所作的物理实验得出的结果而提出来的，它更接近于物理现实，为镜面反射计算引入了两个重要的元素：

1. 菲涅尔定律（折射率、斜角……）
2. 通过分布若干个类似于镜面的微小平面来造型的材料表面状态。

R_i , Cook和K.Torrance提出了一种计算入射光能与反射光能间光谱移动的方法，并把菲涅尔定律推广运用到 R_i 和 R_d 的计算，从而使这种模型变得更为完善。

二、射线跟踪技术的特点

射线跟踪程序的原理是允许计算透明度以及真实的镜象效应。这是射线跟踪的第一个特点，L.Whitted的一篇以Phong的浓淡型为基础改进面向射线跟踪的照明模型的文章中早已