

# 基于阴影图的阴影生成算法研究现状

过 洁, 徐晓暘, 潘金贵

(南京大学计算机软件新技术国家重点实验室 南京 210093)  
(guojie\_022@163.com)

**摘 要:** 阴影效果在计算机图形的真实感显示中非常重要, 而阴影图算法又是阴影算法的主流. 文中对阴影图算法展开了系统的研究, 特别是对该领域近几年的发展进行综述, 从阴影图数量、参数化和内容三个方面总结归纳了各类阴影图算法的基本思想、特点及其应用环境. 综合考虑以上三个方面后对特定的场景选择合适的阴影图算法, 除了选择场景外, 还包括光源类型、内存消耗、实时性要求、硬件支持及特殊阴影要求等. 最后结合当前的技术发展和应用需求展望了阴影图领域未来的研究方向.

**关键词:** 阴影图; 反走样; 三维图形学; 真实感绘制; GPU  
**中图法分类号:** TP391

## Present Status of Shadow Generation Algorithms Based on Shadow Maps

Guo Jie, Xu Xiaoyang, and Pan Jingui

(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093)

**Abstract:** Shadow generation is essential to the realistic display of computer graphics, and the algorithm of shadow maps is the mainstream for shadow generation. In this paper, the algorithms of shadow maps are systematically studied, focusing mainly on the development in recent years. The paper summarizes the fundamental ideas, characteristics and application environments for various types of shadow map algorithms from three aspects, the number, parameterization and contents of the shadow maps. After considering the three aspects, appropriate shadow map algorithms can be taken based on the scene, in terms of the light type, memory consumption, real-time requirements, hardware support and special shadow requirements. Finally, based on the development of technology and application requirements, some directions to the future investigation in this area are given.

**Key words:** shadow maps; anti-aliasing; three-dimensional graphics; realistic rendering; GPU

从人类对 3D 世界的认知而言, 阴影是非常重要的视觉线索. 在图形学相关领域, 用计算机生成阴影一直是研究的热点, 尤其是在 3D 游戏领域, 更需要实时逼真的阴影效果<sup>[1]</sup>. 阴影增强了图形显示的真实感, 同时也提供了场景中一些隐藏的信息, 如位置信息和物体几何特征. 但是由于阴影计算与各种因素有关, 其昂贵的计算代价(确定某个表面上的一点是否在阴影中需要查询场景中大量的潜在遮挡

物)使其在早年发展比较缓慢, 效果也比较粗糙. 近年来, 伴随着图形硬件的发展, 研究者们提出了很多高效的算法. 同时, 一些复杂场景元素的加入(包括毛发、烟雾、半透明物体等)也对阴影算法提出了新的要求.

目前流行的阴影算法大致可以分成 3 类: 基于物体的算法、基于图像的算法和全局光照算法. 基于物体的算法以 Crow 的阴影体算法<sup>[2]</sup>为代表, 该类

算法可以生成轮廓清晰、真实感强的阴影,但是涉及到场景分析,运算时间与场景的复杂度成正比,因此对于相对复杂的场景而言,该算法的效率很低;基于图像的算法中最具代表性的就是本文将重点阐述的阴影图算法<sup>[3]</sup>,相比于阴影体算法,它有简单高效、独立于场景复杂度等优点;全局光照算法主要有光线跟踪算法、辐射度算法和光子映射算法,这类算法可以生成非常逼真的阴影效果,但是需要对整个场景空间进行分析,计算过程比较复杂,所以效率相对较低。关于阴影的概念及阴影生成的各种经典方法参见文献[1,4-6]。因为最近几年与阴影图相关的算法及各种改进层出不穷,且该类算法在产业方面的应用(包括计算机动画、3D 游戏等)非常广泛,本文对近年来阴影图方面的发展进行综述。

## 1 基本阴影图算法及其优缺点

阴影图算法最早是由 Williams 提出的一种用 Z 缓存生成场景中物体阴影的方法<sup>[3]</sup>。Z 缓存即深度缓存,存放场景中物体的深度值。Z 缓存在对物体进行着色时执行隐藏面消除工作,从而确定物体的可见性。Z 缓存最主要的参数就是其所用的位数,这对阴影图影响很大,目前图形硬件一般都支持 16 bit 的 Z 缓存,更高的可达 32 bit。

传统阴影图算法可以分为 2 个基本绘制过程: 1) 将光源作为视点,对整个场景进行绘制。此时,仅在 Z 缓存中保存从光源到场景的最小深度值,从而得到一张阴影图,阴影图中的每个单元称为一个阴影纹素。在静态场景的情况下,这一步可以作为预计算部分;而在动态场景中,一旦场景中的遮挡关系发生改变,就需要实时地更新阴影图。2) 回到以相机为视点的视图中,按照传统的渲染流水线方式对场景进行绘制。不同的是,在绘制过程中要进行阴影判断,即判断某个像素是否处于场景的阴影中。若像素不在阴影中,则正常绘制;否则绘制成阴影。判断场景中某个点是否在阴影中需要用到第一次绘制的阴影图:对于场景中的某个点  $P$ ,首先将其转换到以光源为视点的 3D 空间,得到  $P$  点在该空间下的深度值;然后与阴影图中  $P$  点对应的纹素所保存的深度值作比较,若  $P$  点深度值大于阴影图中保存的深度值,则说明  $P$  点被遮挡,处于阴影区域,否则说明不在阴影区域。通过这种简单的判断,就可以得到场景中的阴影区域。

阴影图应用如此广泛是因为它具有以下优势:

1) 原理简单,实现方便。阴影图算法的基本原理比较简单,其涉及 2 个基本的绘制过程:阴影图生成和场景绘制。阴影图可以使用 2D 纹理贴图实现;场景绘制过程增加一个深度比较功能,其实现也很方便,3D 图形库(如 OpenGL)一般自动实现深度比较功能。

2) 独立于场景复杂度。阴影图是一种基于图像的阴影生成方法,对场景空间的复杂度依赖性很低,它的复杂度仅取决于光源数和阴影图分辨率。这一点是区别于基于物体的方法的,在阴影体方法中,需要对场景中的物体进行分析,并根据物体几何特征得到其阴影体,所以场景越复杂,需要分析的物体就越多,该方法效率越低。相反,阴影图方法无需分析场景,无论对什么场景,在光源一定的情况下,只要阴影图的分辨率确定,其复杂度就确定<sup>[3-6]</sup>。

3) 适合 GPU 硬件实现。GPU<sup>[7]</sup>是一种专门用于图形处理的硬件,其相对于 CPU 具有强大的浮点数运算能力和并行处理能力,所以对于处理大数据量的运算(特别是图形图像相关的任务)而言,它具有独特的优势。目前的 GPU 都提供了高度的可编程性,所以一般的阴影图算法都可以移植到 GPU 上,以提高整体渲染速率。

但是,阴影图算法也存在缺点:

1) 走样问题。该问题是阴影图算法的最主要问题,原因是在深度比较过程中涉及到对阴影图的采样。如果阴影图的分辨率低于采样需求,就会产生走样,其典型的表现则为锯齿状阴影边界。还有一种称为自阴影走样<sup>[6]</sup>,是受各类数值精度的影响,包括缓存本身精度的限制和各种转换的数值精度影响而产生的;光源采样和屏幕采样的偏差也会产生这种问题,其表现为阴影中一些斑驳的色块,即光渗现象。通过提高 Z 缓存位数或者引入偏移因子可以改善阴影效果,但是偏移因子的确定很困难。事实上,很多阴影图算法的变种就是为了解决走样问题而提出的。关于走样的原理和理论分析详见文献[6]。

2) 处理全向光或半球形光问题。全向光就是向整个空间发射光线的光源(如白炽灯),半球形光是向半个空间发射光线的光源。由于该类光源的视域非常大,所以处理起来比较麻烦,一般一个简单的阴影图无法保存所有的深度值,所以必须使用其他的技术来解决。

通过一些改进方法可以克服上面的缺点,同时不影响阴影图本身的优势。大部分与阴影图有关的算法都或多或少地解决了上述问题,使得阴影图算

法在当今的阴影生成应用中非常流行,成为事实上的标准算法.著名的 3D 电影《玩具总动员》中的阴影就是用阴影图算法实现的.

## 2 构造阴影图应考虑的问题

通常,使用阴影图算法需要考虑 3 个问题:阴影图的数量、参数化和内容.各种阴影图变种算法及其相关算法都是从这 3 个方面考虑,来解决其中的一个或几个问题,使得产生的阴影更真实,效果更理想.下面将从这 3 个问题出发,讨论各种阴影图算法,并比较它们的优缺点及使用环境.

### 2.1 确定阴影图的数量

一般情况下,针对一个点光源绘制一张阴影图就够了,但是在全向光或多光源情况下需要更多的阴影图.有时,为了对场景中的阴影进行反走样,也需要多张阴影图来柔化阴影边界或显示阴影细节.

#### 2.1.1 全向光(或半球形光)

对于全向光(或半球形光),需要记录其整个 3D 空间  $4\pi$ (或  $2\pi$ ) 空间角内的所有方向的深度值.一种很自然的想法是将阴影图设计为一个球面<sup>[8]</sup>(或半球面),将相对于光源的各个方向上的深度值记录在该球面(或半球面)上,该球面(或半球面)可以展开为一个 2D 平面.每次查询阴影图时,将待查询点的位置向量  $\mathbf{r} = (r_x, r_y, r_z)$  转化为球面坐标( $u = \arccos(-r_z)$ ,  $v = a \tan 2(r_y, r_x)$ )<sup>[6]</sup>.这里使用的是经度-纬度映射,最终得到该点在球面上的经纬度.这种方法比较直观,但是由于每次查询都要进行 2 次三角运算,所以该算法的效率比较低.

立方体阴影图算法(cube maps)<sup>[9]</sup>通过在光源周围放置一个立方体(如果是半球面光源就用半个立方体),光源放在立方体的中心位置,将光源所有方向的深度值分别记录在 6 个正方形中来形成一个阴影图组.如图 1 所示,一个立方体阴影图由 6 张阴

影图组成,每张代表一个方向.这样就不需要对阴影图进行任何的变形,即对每个阴影图用普通的参数化方法即可;而且,由于现在的硬件已经完全支持立方体贴图,这种方法很容易移植到 GPU 上.但是,因为有 6 张阴影图,所以至少要对场景绘制 6 次才能得到整个空间的阴影图,这样的开销有时显得过大.最近,Shader Model 4.0 提供了一种称为几何着色器的新着色器,可以通过一次渲染生成整个立方体贴图,这种技术用在这里可以加速生成立方体阴影图.由于立方体阴影图简单、易实现,同时效果也比较理想,所以它成为解决全向光源下阴影生成的主流算法.

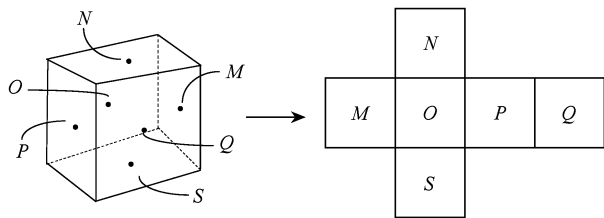


图 1 立方体阴影图的构成

双抛物面阴影图<sup>[10]</sup>使用 2 个抛物曲面来记录  $4\pi$  空间内的深度值(对于半球面就用一个抛物面),每个曲面描述为  $f(x, y) = \frac{1}{2} - \frac{1}{2}(x^2 + y^2)$ ,  $x^2 + y^2 \leq 1$ ;然后通过访问函数<sup>[6]</sup>  $u = \frac{r_x}{2(1+r_z)} + \frac{1}{2}$  和  $v = \frac{r_y}{2(1+r_z)} + \frac{1}{2}$  查询阴影图.

该方法与球面阴影图类似,但是访问函数相对简单,相比立方体阴影图,需要的绘制过程也较少,且可以完全使用可编程的 GPU 实现.但是,由于光栅化过程中线性插值的影响,对于很大的多边形而言该方法可能会失效.另外,该方法生成的阴影图走样问题比立方体阴影图算法严重,但如果采用后面要介绍的一些其他的技术,走样问题可以得到改善.图 2 显示了几种不同的阴影图<sup>[10]</sup>.

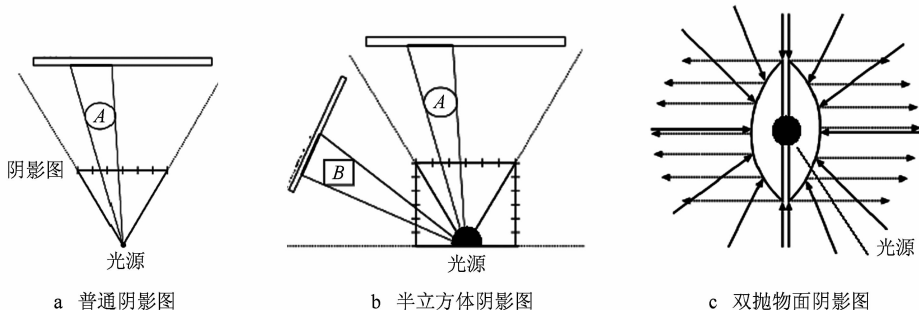


图 2 几种不同的阴影图

2.1.2 多光源或面光源

在一个点光源或者平行光源的情况下,生成的是硬阴影,具有明显阴影的边界.这时,场景中的某个点是一个二值的状态:要么在阴影中,要么不在阴影中.而在多光源下将会生成软阴影,软阴影由图 3 所示的本影和半影组成.一般拥有模糊的阴影边界,其处于本影中的点将被完全遮挡,而处于半影中的点将被部分遮挡.真实世界的阴影都是软阴影,所以合理地生成软阴影会使 3D 图形更具真实感;但是软阴影的生成比硬阴影更复杂,真实地模拟软阴影很困难,所以可以采用一些简化方法.由于光线传播的线性特征,即不同光源的光线传播互不影响,我们只需简单地将各个光源产生的阴影效果进行叠加,这种方法可以使用累积缓存<sup>[11]</sup>实现;或者采用可视百分比(即多少光源可以看到),为每种可视百分比赋予一个权值,最后根据权值在最终的图像上绘制出明暗程度不同的阴影.

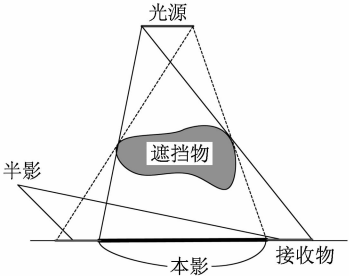


图 3 软阴影示意图

可以对面光源(或者线光源、体光源)采样,将其转化为多点光源<sup>[12]</sup>;再使用多点光源的方式处理.但是为了得到足够精确的软阴影效果,有时需要在面光源上采集相当多的采样点,这样会产生大量的阴影图.事实上,由于很多的阴影图之间有较高的相似性,这样做会极大地浪费时间和空间.Chen 等<sup>[13]</sup>提出一种减少阴影图数量的算法,首先在面光源的几个关键点(如正方形的 4 个顶点)生成关键阴影图,而其他位置的阴影图由一种称为“视图插值”的技术间接生成;再使用传统阴影图的深度比较过程.该算法对于简单面光源效果比较好,对复杂面光源就会涉及到采样点的选择问题.同时,对于每个插值点,为了进行深度值比较,需要将场景投影到该插值点的阴影图空间,但这种做法在实时应用中效果不佳.

Agrawala 等<sup>[14]</sup>提出了分层衰减图的方法,它是一般多面光源阴影图算法的扩展,具有 2 个基本阶段.在预计算阶段,首先对光源进行采样,然后根

据这些采样点为整个场景生成分层衰减图.具体过程如下:对每个光源采样点,将视点转换到该点,并渲染场景到类似阴影图的衰减图中(只需深度值);然后将这些图像都变形转换到以光源中心采样点为参考点的图像中,即将所有图像集中到一张分层衰减图中.分层衰减图中每一点存储的是一个列表,列表的每一项包括深度值和从该点可以看到的光源采样点数(也可以是可视百分比),不同的深度值形成不同的层,这张图就是累计所有采样点的衰减图得到的最终分层衰减图.在渲染阶段按传统的方式渲染场景,并将每个像素点投影到以光源中心采样点为视点的空间中,查找分层衰减图确认在其中是否可以找到该点.若找到,则根据该点对应的可视百分比修正颜色;否则,说明该点完全处于阴影中.该方法的最大的优势就是可以用一张图实现,其效率和真实感取决于采样点的数目.

针对线性光源,Heidrich 等<sup>[15]</sup>提出了一种快速生成软阴影的方法,其最大的优势是只需采用较少采样点.对于线性光源,通常 2 个顶点就已足够,该方法使用一种双通道的阴影图,一条通道即传统的阴影图通道,保存从光源看到的最近点的深度值;另一条称为“可视通道”,保存深度通道中对应点的可视百分比.当渲染某点  $P$  时,对 2 个阴影图分别处理:对每个阴影图,根据比较的深度值大小决定是否要查询可视通道,若点  $P$  的深度值小于阴影图中的深度值,说明针对该光源采样点,点  $P$  可见,需要查询该光源采样点的可视通道,并根据查询到的可视百分比调整局部光照;如果点  $P$  在阴影中,则局部光照为 0.最后累加各个采样点的局部光照,得到最终的光照效果.这种算法的最大问题是,如果采样点选择不充分,可能导致错误的阴影效果.Ying 等<sup>[16]</sup>扩展了这种方法,使之适用于多边形面光源,其主要思路是对多边形的每个顶点生成阴影图,对多边形的每条边生成可视通道.

针对面光源,除了上述多采样算法之外,还有一些高效算法. Atty 等<sup>[17]</sup>和 Guennebaud 等<sup>[18]</sup>同时提出了使用软阴影图的面光源软阴影生成方法,对每个面光源使用一张阴影图即可生成近似的软阴影效果.它们将阴影图用作场景的一种简化的离散化表示,阴影图的每个纹素代表场景中的潜在遮挡物,并将其后投影到光源,以计算半影区的遮挡.不同之处是,Atty 等是在阴影图空间计算软阴影,而 Guennebaud 等是在屏幕空间计算软阴影.由于软阴影图算法对场景进行了离散化表示,所以容易出现光渗现象和过

阴影现象. 光渗现象的出现是因为阴影图的不同纹素后投影到光源时出现缺口, 导致在阴影区中出现亮带, 为此文献[18]中提供了一种缺口填充方法. 过阴影的出现是因为阴影图的不同纹素后投影到光源时出现重叠, 这样阴影区域就被过量估计了, 使阴影区过暗. 为了解决这些问题, Guennebaud 等<sup>[19]</sup>对文献[18]的算法进行了改进, 改进后的算法不是将所有的潜在遮挡纹素都后投影到光源, 而是首先对这些潜在遮挡纹素构造遮挡轮廓边, 并将遮挡轮廓边后投影到光源; 然后对投影后的轮廓边进行径向面积分, 以计算光源可见性. 该方法可以解决走样问题并提高阴影生成效率, 但是涉及复杂的轮廓边提取和径向面积分过程. Schwarz 等<sup>[20]</sup>对光源引入位掩膜以解决过阴影问题, 其核心是对光源进行采样, 并对每个光源采样点用一个比特位表示其可见性, 以避免重叠区域的多次计算. Bavoil 等<sup>[21]</sup>结合深度剔除(depth peeling)提出了一种新的软阴影方法, 其使用多层阴影图(即从光源方向提取多层深度图)以减少光渗现象, 并使用中点阴影图处理自阴影效果. 为了进行深度剔除, 生成多层阴影图, 需要在光视图空间对场景多遍绘制, 所以比较耗时.

还有其他一些使用单个阴影图生成软阴影的算法, 这些算法主要基于对阴影图的图像分析处理, 包括对阴影边界的模糊处理及滤波处理等, 所以由这些方法产生的阴影大多是假的软阴影, 但是其生成速度快, 适合交互式应用.

### 2.1.3 多阴影图反走样

阴影图算法的最主要问题是走样, 其中大多是由阴影图的低分辨率引起的. 解决走样问题的方法很多, 下面介绍使用多个阴影图的反走样技术.

采用多阴影图反走样的基本思想是对场景进行划分, 包括深度划分(也称  $z$  划分)和自适应划分, 然后对每个子场景生成一个阴影图, 以满足不同深度或不同区域场景的分辨率需求. 图4所示为2种场景划分方法: 图4a所示为场景的深度划分, 将场景

划分为多个平行于视平面的子场景; 图4b所示为场景的自适应划分, 在场景的不同位置使用不同分辨率的阴影图.

对场景分割的一个主要原因是场景太大, 一张阴影图无法同时满足近平面和远平面的采样率要求. 最早由 Tadamura 等<sup>[22]</sup>提出一种分割场景并使用多个阴影图来处理太阳光阴影的思想, 以达到减少走样的目的. Lloyd 等<sup>[23]</sup>对其进行了深入研究, 并总结了2种常见的划分方式:  $z$  划分和面划分.  $z$  划分即深度划分; 面划分主要应用在光源方向与观察方向近似一致的情况下, 这时可以根据视锥体的各个侧面对场景进行划分, 这种划分方式也可以用在全向光源的情况下.

Zhang 等<sup>[24]</sup>在此基础上提出了一种对大规模虚拟场景的深度划分算法: 平行分割阴影图(parallel split shadow maps, PSSM). PSSM 用平行于视平面的平面将视锥体划分为多个子区域, 每个子区域使用不同的分辨率的阴影图产生阴影. 在该算法中提出了一种基于走样误差分析的快速划分方法, 使得最终生成的各个阴影图的走样误差相对较小, 并在整个场景上分布适中. 同时, 对划分后的子场景使用紧密的包围盒来提高阴影图的分辨率. 文献[24]中比较了3种沿深度方向的划分模式, 这也是一般深度划分常考虑的模式: 一种是对数划分模式  $C_i^{\log}$ , 其各分割面的计算方法为  $C_i^{\log} = n \left( \frac{f}{n} \right)^{\frac{i}{m}}$ ; 另一种是规则划分模式  $C_i^{\text{uniform}}$ , 其计算为

$$C_i^{\text{uniform}} = n + (f - n) \frac{i}{m}.$$

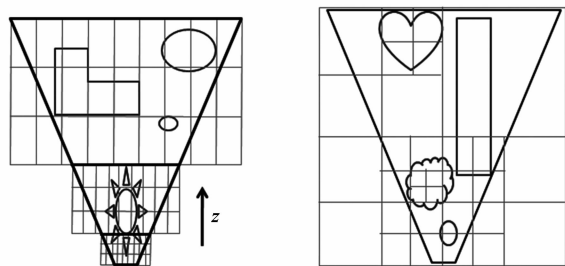
其中,  $m$  表示子场景个数,  $i$  表示第  $i$  个子场景, 其近平面为  $C_{i-1}$ , 远平面为  $C_i$ ;  $n$  和  $f$  分别表示原视锥体的近平面和远平面.

由于前2种划分无法同时为近平面层和远平面层子场景提供合适的采样密度, 所以文献[24]提出了第3种模式用于调和远近之间的走样. 第3种模式是前2种的混合, 称为实用划分模式  $C_i$ , 其计算方法为

$$C_i = (C_i^{\log} + C_i^{\text{uniform}}) / 2.$$

文献[25]中, Zhang 等又提出了一种将 PSSM 移植到 GPU 上的方法. 在 DX10 级别的硬件下, 由于 DX10 支持几何着色器和绘制目标阵列, 这样可以一遍绘制出所有的阴影图, 所以只需绘制 1+1 次(即一次生成所有的阴影图)就可以得到最终的图像.

自适应阴影图(adaptive shadow maps, ASM)<sup>[26]</sup>



a 深度划分

b 自适应划分

图4 场景的2种划分方式(梯形框内是场景)

是采用自适应划分方法的典型阴影图算法. ASM 通过逐层细分一张普通的阴影图,并在场景细节丰富的区域,特别是阴影边界生成新的、具有更高分辨率的子阴影图,来提高阴影质量.一般将 ASM 组织为树状结构,如图 4 b 所示,其组织成二叉树,树的每个结点为一个固定分辨率的阴影图,对应场景中的一块区域(根结点对应整个场景).当某个区域的阴影图无法达到误差要求时,继续细分场景,直至一个约定的状态. ASM 可以使用 mip-mapping 技术实现,它有 3 个主要的特点:1)ASM 由视点驱动,意指这种树状结构是随着视点变化不断更新的;2)ASM 的内存消耗可以有效控制;3)ASM 是一种增量式的算法,可以动态地增加树的层数以增强阴影效果,直到一定的存储限制.这种方法基于硬件的实现比较复杂,直到 2005 年 Lefohn 等<sup>[27]</sup>才提出基于 GPU 的实现方法,而且为了确定是否要继续划分,CPU 需要回读最终图像以进行误差判断,而这种回读是相当耗时的,所以性能不佳.

针对回读的问题,Giegl 等提出了 queried virtual shadow maps<sup>[28]</sup>,其主要思想就是使用硬件提供的遮挡查询功能确定是否更新子阴影图.为了确定是否要对某个子阴影图继续划分,使用遮挡查询统计划分前后有多少像素值发生了改变,直到低于某个阈值.

RMSM(resolution-matched shadow maps)<sup>[29]</sup>是另一种改进的 ASM 算法,它同样使用二叉树存储阴影图,并在 GPU 上实现. RMSM 基于图像空间和阴影图空间的连续一致性,即图像空间中连续可见表面在自适应阴影图中也具有连续一致性,以降低对阴影图页面的冗余请求,从而减少迭代边界查找开销. RMSM 使用连通分支分析在 GPU 端完成场景分析,大大减少了 GPU 到 CPU 的数据传输.

另一种自适应划分方法称为 tiled shadow maps<sup>[30]</sup>,其将传统的阴影图划分为不同尺寸的瓦片,每个瓦片分配不同的权值以自适应调整不同瓦片中阴影图采样率.为了得到每个瓦片的权值,该方法需要增加一个过程:光源视图分析,其基于深度值不连续性和视点到表面的距离以及其他的影响因子调整瓦片的大小,以增大阴影边界和靠近相机的物体位置处权值,提高采样率.

2.1.4 其他多阴影图技术

多阴影图技术也可以配合全局光照算法使用,以提高该类算法的效率.如 Ritschel 等提出的 coherent

shadow maps<sup>[31]</sup>和 coherent surface shadow maps<sup>[32]</sup>,它们在场地的不同位置和方向为场景中的物体生成多个阴影图,以记录其几何细节;同时,采用一种无损压缩方式对这些阴影图进行压缩,以减少存储消耗.文献[31-32]算法的区别在于前者是在物体外不同位置生成的多个阴影图,而后者是在物体上的采样点生成的多个立方体阴影图.它们都可以在预处理阶段生成,以减少渲染阶段的可视性查询,提高全局光照算法的渲染效率.

2.2 阴影图的参数化

阴影图的参数化是指对每张阴影图,如何在场景中采样深度值并将其存储在阴影图中,其在文献[23]中也称为阴影图的变形.以最简单的与场景水平面正交的平行光源为例,最普通的参数化方式为一致参数化,即对场景等间距采样,使得生成的阴影图中的相邻纹素之间的距离是相等的.显然,这种采样方式无法同时满足视点的近平面和远平面的分辨率要求,会造成严重的走样.若满足远平面的采样率,则近平面区域分辨率达不到要求;相反,若满足近平面采样率,为了同时满足远平面,生成的阴影图过大,消耗大量内存,甚至可能无法覆盖整个场景.所以为了反走样并控制阴影图的存储消耗,研究人员提出了各种参数化方法.

阴影图的参数化主要解决透视走样<sup>[33]</sup>问题.透视走样是阴影图中的一种常见走样误差,它是由阴影图采样和屏幕图像采样不匹配引起的.对于 3D 图形的透视投影,在近平面容易出现一个阴影纹素对应多个屏幕像素的情况,使得近平面处阴影的边界有严重的锯齿现象.图 5 显示了阴影图走样<sup>[34]</sup>的原因:如果阴影图纹素  $ds$  所对应的视平面区域  $dp$  包含多个像素点,那么这些像素点都会取相同的值,这样就产生了锯齿状阴影边界.

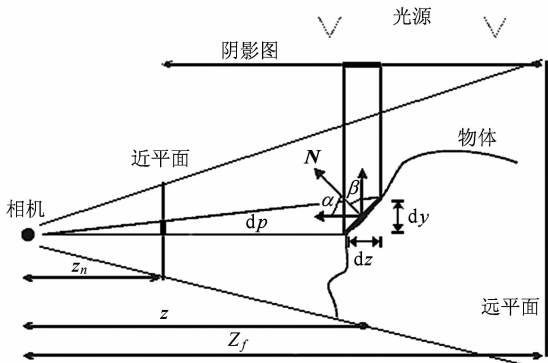


图 5 阴影图走样示意图

图5中的走样程度可以用  $\frac{dp}{ds} = \frac{z_n}{z} \frac{dz}{ds} \frac{\cos \alpha}{\cos \beta}$  度量<sup>[34]</sup>,其中,  $\frac{z_n}{z} \frac{dz}{ds}$  称为透视走样,  $\frac{\cos \alpha}{\cos \beta}$  称为投影走样. 若  $\frac{dp}{ds} = \frac{resolution_{shadowmap}}{resolution_{screen}}$ , 表示阴影图没有走样; 若  $\frac{dp}{ds} > \frac{resolution_{shadowmap}}{resolution_{screen}}$ , 则存在走样; 若  $\frac{dp}{ds} < \frac{resolution_{shadowmap}}{resolution_{screen}}$ , 则没有走样, 但是阴影图过采样.

对于一般的阴影图,  $dz/ds$  是个常数, 误差与  $z$  成反比. 所以当  $z$  变小时误差变大, 这从理论上解释了为什么近平面容易出现走样. 由于一般透视走样与场景中的具体物体无关, 所以比较容易处理; 而投影走样涉及到面的法向量, 与物体形状有关, 处理起来比较困难. 而且一旦处理了透视走样, 阴影图的整体误差也会相应减少, 因此一般的阴影图只处理透视走样.

最早用于处理透视走样的是 Stamminger 等提出的透视阴影图 (perspective shadow maps, PSM)<sup>[33]</sup>, 该方法的主要思想是在透视投影后的空间中生成阴影图, 从而减少走样. 传统的阴影图算法是在生成了阴影图之后再进行透视变换的, 而 PSM 则先对场景在照相机视图空间进行透视变换, 将场景转化为一个规则的正方体, 然后在透视变换后的场景上生成阴影图, 其他部分与传统的阴影图算法一致. 现在阴影图面对的是完成了透视变换后的场景, 此时近平面的场景相对于远平面的场景变大, 相当于增加了近平面的采样率. 这里要注意的是由于光源也要进行相应的变换, 因此有可能出现点光源变成平行光源, 平行光源变成点光源的情况. 该方法的最大问题是会出现“奇点”<sup>[35]</sup>. 通常投影变换会将照相机背面

的物体投影到场景的无穷远处, 但是有时光源在相机背后, 而一些不在视锥体中的物体可能是遮挡物, 如图6a所示. 一旦将物体投影到另一面就会出错, 如图6c所示. Stamminger 等<sup>[33]</sup>提出一种虚拟相机的方法, 用一个扩大的相机空间将所有物体装进去, 如图6b, 6d所示. 但是这样会降低阴影质量, 因为视点的移动降低了阴影图的分辨率. PSM 的另一个问题是阴影图的质量依赖于光源的位置.

为了解决 PSM 的诸多问题, Wimmer 等<sup>[34]</sup>提出了光源空间的透视阴影图 (light space perspective shadow maps, LisPSM), 该方法与 PSM 的最大不同在于其透视变换是在以光源为视点、光源方向为观察方向的空间进行的. 这种变换的最大优势是不改变光源的方向, 也不会产生奇点, 因为光源方向是和以光源为视点的光空间的视平面平行的. 该方法最重要的过程就是构造光空间的视锥体, 其中近平面  $n$  的确定尤其重要.  $n$  是唯一的自由参数, 最终阴影图的好坏直接取决于  $n$ . 文献<sup>[34]</sup>中指出, 当相机空间视平面和光线方向正交时, 取  $n = z_n + \sqrt{z_f z_n}$  (其中  $z_n$  和  $z_f$  分别是相机空间的近平面距离和远平面距离) 时阴影图参数化最优.

事实上, 任何空间的透视投影都可以用来对阴影图重参数化, 而不一定是相机空间的或者光源空间的, 它们本质上的差别在于使用的  $4 \times 4$  投影矩阵不同. 文献<sup>[34]</sup>中提到存在一种最优参数化的方法称为对数阴影图, 发生在  $dp/ds = 1$  的情况下, 此时若视平面与光线方向正交, 则

$$ds = \frac{dz}{z},$$

$$s = \int_0^s ds = \int_{z_n}^z \frac{dz}{z} = \ln \frac{z}{z_n}.$$

Lloyd 等<sup>[36]</sup>继承并发展了对数阴影图思想, 提出了对数透视阴影图 (logarithmic perspective shadow maps, LogPSM). 这是一种将对数化和透视投影相结合的方法, 图7所示为其基本操作过程. LogPSM 首先对场景进行普通的透视投影, 然后对透视投影后的场景沿阴影图的  $Y$  方向进行对数变换, 使得阴影图沿  $Y$  方向的采样间距呈线性变化. 其变换公式为  $F(y) = c_0 \log(c_1 y + 1)$ , 其中  $c_0 = \frac{-1}{\log(f/n)}$ ,  $c_1 = \frac{1 - (f/n)}{(f/n)}$ .

该算法简单地在阴影图上施加一个修正, 称为分辨率再分配, 使得最终的阴影图在2个方向具有近似相等的分辨率.

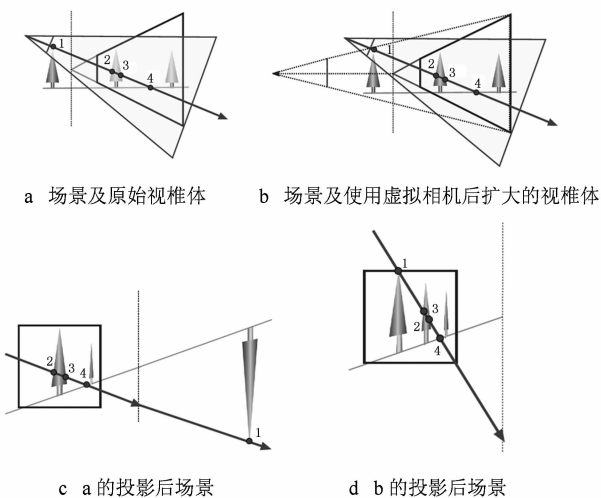


图6 PSM的奇点问题<sup>[33]</sup>

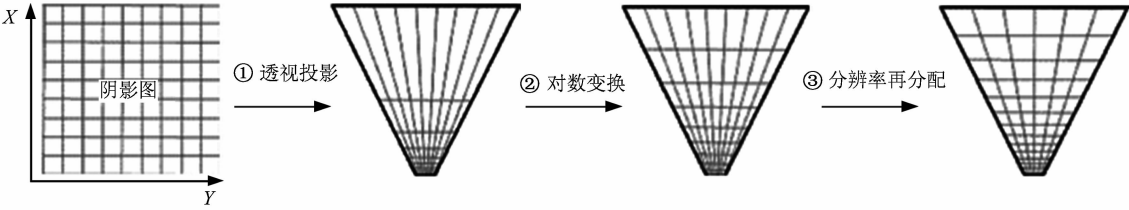


图 7 LogPSM 操作过程

Lloyd 等同时对 3 种阴影图为反走样所需要的存储消耗(即阴影图的大小),其中,普通的阴影图存储消耗为  $O((f/n)^2)$ ,而 PSM 为  $O(f/n)$ ,当使用了 LogPSM 后降为  $O(\log(f/n))$ . 可见,使用了 LogPSM 后可以在不增加存储开销的前提下,大大减少走样误差.

还有其他一些阴影图参数化方法,如 TSM (trapezoidal shadow maps)<sup>[37]</sup>. 该方法的特别之处是在以光源为视点的投影后空间中,为以相机为视点的视锥体构造一个最小的梯形包围盒;然后通过一个梯形变换(文献[37]中称之为  $N_T$  变换)将该梯形包围盒映射到阴影图中. 图 8 a 所示为光源视图下的相机视锥体;图 8 b 所示为该视锥体构造的最小梯形包围盒;图 8 c 所示为使用  $N_T$  变换映射到一个新的规范化空间. 使用这种方法可以最大化地利用阴影图,减少损耗(即阴影图中落在视锥体外的区域).

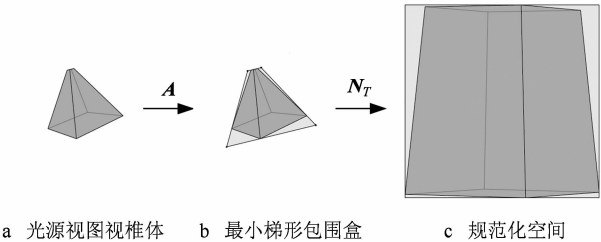


图 8 TSM 操作过程<sup>[37]</sup>

以上这些方法都可用可编程 GPU 实现,因为大部分只需增加一个利用  $4 \times 4$  矩阵的变换过程. 这些都属于规则采样,还有一类非规则采样,即采样点的分布在场景中是不规则的,它可以自适应地在阴影边界或细节丰富的区域增加采样点.

AFSM(alias-free shadow maps)<sup>[38]</sup>是一种重要的非规则采样方法,其采用逆向反走样方式提高阴影图采样率. 该方法首先在相机视图下渲染场景获得可视采样点,然后将可视采样点变换到光源视图下,并将其记录到阴影图中. 由于可视采样点在场景中的不规则性,其在阴影图上的投影分布也是不规

则的. 通过这种方法匹配了光源视图采样和相机视图采样,以减少甚至消除了走样问题,但是由于当前的硬件不支持这种非规则采样,其硬件实现比较困难. Johnson 等<sup>[39]</sup>设计了一种不规则 Z 缓存,采用一种新的硬件架构,并将其应用到不规则阴影图中. Arvo<sup>[40]</sup>使用深度剔除技术在当前硬件系统上实现了 AFSM,但其需要多次绘制,并生成多层的深度图. Sintorn 等<sup>[41]</sup>在此基础上提出了一种一遍绘制的实现方法,但是过程比较复杂,存储需求较大.

2.3 阴影图的存储内容及其滤波处理

根据阴影图算法,阴影图中应该存储光源到场景中物体的最近深度值. 在当前的图形硬件系统中,阴影图一般是一张特殊的纹理图,其存储精度有限,所以经常会出现走样. 有些阴影图算法从另一个角度考虑阴影图的构造,即在阴影图中存放其他值,而不仅仅是深度值,或者对阴影图进行一些处理,主要是对阴影图进行滤波处理,以达到减少锯齿现象、柔化阴影边界的目的. 同时,有些半透明的物体或者像毛发、火焰等物体的阴影,仅用单个深度值是无法有效生成的,这就需要存储其他的内容,以形成合适的阴影效果. 下面介绍几种可以解决以上问题的阴影图.

为解决自阴影走样, Woo<sup>[42]</sup>提出使用 2 个深度值的中间值作为最终深度值的方法. 该方法需要保存从光源出发的 2 个最近的深度值,而不仅是一个最小值;然后对这 2 个值取平均,存放到最终阴影图中,以解决阴影图的偏置问题(即上面提到的偏移因子问题). 但是该方法无法解决物体边界的自阴影问题,例如在凸面体的角上,由于光线与其只有一个交点,使 2 个最近深度值相等,偏置因子降为 0. 另一方面,若仅仅使用一个常数的偏置项,如果该偏置项设置过大会导致错误的阴影区域. 为了解决这些问题, Weiskopf 等<sup>[43]</sup>综合了中值偏置项和常数偏置项,以取长补短,即

$$z_{\text{bias}}(z_1, z_2) = \min\left(\frac{z_2 - z_1}{2}, z_{\text{offset}}\right);$$



其中,  $z_1$  和  $z_2$  为最近的 2 个深度值,  $z_{\text{offset}}$  是一个常数偏置项, 最终的偏置项取  $(z_2 - z_1)/2$  和  $z_{\text{offset}}$  的最小值. 这样既可以根据具体的物体特征设置偏置项, 又可以保证偏置项不会太大, 产生不正确的阴影区域. 在该方法中还采用背面剔除算法来解决物体边界的自走样: 采用背面剔除后, 第二深度值就不会是同一物体的背面, 而是另一物体的正面, 避免 2 个深度值相同. 文献[34]中同时提供了该方法对应的硬件实现. 其他解决自阴影走样的方法还有 Hourcade 等[44]的优先缓冲器法, 其在阴影图中保存物体的唯一 ID 值而不是深度值, 该方法也可以解决偏置问题. 但由于其需要按照严格顺序对物体绘制, 所以性能不佳[6].

RSM(reflective shadow maps)[45]是传统阴影图的一种扩充, 其在阴影图中不仅保存场景深度值, 而且保存该深度值对应场景中点的法向值、世界坐标值和该点的反射光通量值. 这样, RSM 中每个纹素就可以看作是一个间接光源, 通过这些间接光源渲染场景可以产生阴影等光照效果. 该方法可以用硬件实现, 使用屏幕空间插值之后对复杂场景进行交互式渲染, 但是其存储消耗较大, 产生的全局光照效果不太精确.

Lokovic 等[46]提出深度阴影图(deep shadow maps, DSM)算法用于生成毛发、烟雾、透明物体等复杂物体的阴影. 它不像传统的阴影图算法只存储场景深度值, 而是存储各采样点随深度分布的可视性函数, 即一种表示光线随空间距离衰减的函数. 每个可视性函数都从 1 出发, 最终衰减为 0. 如果某个采样光线与场景中的一个不透明物体相交, 则此采样光线上该交点后的物体的可视性立刻衰减为 0, 即不可见, 处于阴影中; 如果采样光线与某个透明物体相交, 则可视性衰减为某个权值  $w(0 < w < 1)$ , 以此递减; 如果光线与烟雾相交, 则沿一个连续递减函数下降. DSM 的运算代价非常高, 所以目前无法用于实时应用. 同时, DSM 无法在硬件上实现, 除非找到一个有效的对可视性函数编码存储的方法. MTSM(multilayer transparent shadow maps)[47]用于在面光源下生成毛发、透明物体的高质量软阴影. MTSM 是 DSM 和分层衰减图的一种混合扩展方法, 每个纹素保存多层的深度值和不透明或半透明值. 类似于分层衰减图, MTSM 需要在每个纹素保存不同深度层的列表, 每层包括该层深度值、累计可

视性和层与层之间可见的光源采样数, 最终通过平均各采样光线的衰减率得到最终的衰减率. 用于生成毛发等物体的自阴影效果的方法还有 opacity shadow maps[48]及其扩充版本 deep opacity maps[49]等.

为了反走样, 解决锯齿形边界问题, 高质量的纹理滤波在当代已经很流行, 但是由于阴影图中存储的是深度值, 阴影测试必须在滤波之前进行, 所以无法用类似的方法进行滤波. 针对这一问题, Reeves 等[50]提出了一种阴影图滤波方式, 称为百分比邻近滤波(percentage closer filtering, PCF), 其中每个像素点的强度由其周围若干点的加权和决定. PCF 不直接对存储深度值的阴影图进行滤波, 而是对完成了深度比较并存储了比较结果的布尔值阵列进行滤波. 每个像素点的深度测试结果由其在阴影图中对应的 4 个邻近纹素点作双线性插值而得. 该方法可以生成光滑的阴影边界. Brabec 等[51]提供了 PCF 的一种硬件实现, 采用传统的 PCF 技术可以柔化阴影边界, 并产生软阴影效果, 但是其光顺性还是较差, 特别是对直线型的边界. 为了改善 PCF 产生的阴影边界的光顺性, 赵乃良等[52]提出了一种数据修正的 PCF 算法, 其主要思想是在双线性插值前对插值的数据进行必要的修正. 该方法对阴影边界为直线的改进效果最为明显, 但是当阴影边界是曲率较大的曲线时, 由于只对阴影边界做了小范围的局部修正, 所以效果不是很理想.

方差阴影图(variance shadow maps, VSM)[53]通过计算阴影图中每个点的深度期望值和方差来模拟深度值的分布, 以减少误差. 期望值和方差可以由一阶矩和二阶矩计算而得, 一阶矩和二阶矩分别为  $M_1 = E(x) = \int_{-\infty}^{\infty} xp(x)dx$  和  $M_2 = E(x^2) = \int_{-\infty}^{\infty} x^2 p(x)dx$ . 利用一阶矩和二阶矩可以计算期望值和方差  $\mu = E(x) = M_1$  和  $\sigma^2 = E(x^2) - E(x)^2 = M_2 - M_1^2$ .

为了得到期望值和方差, 必须同时保存深度值和深度值的平方. 最后在屏幕渲染阶段, 利用切比雪夫不等式计算每个像素的值: 深度值小于期望值的, 没被遮挡, 而深度值大于期望值的就是阴影区域. VSM 本质上是一种阴影图滤波方法, 其保证直接对阴影图滤波就可以达到对最终图像反走样的目的. 但是由于它只保存 2 个矩, 所以对于复杂的可视函数不能有效地模拟, 会出现光渗现象. 解决这个问题

的一个显而易见的想法是增加多个矩(三阶矩、四阶矩……),但是高阶矩由于其不稳定性,很难使用<sup>[53]</sup>. 另一种解决办法称为分层 VSM<sup>[54]</sup>,其主要思想是将阴影图所在的光源视椎体沿深度方向划分为多个层,每个层对应一个深度区间. 由于每个层都降低了深度范围,所以相对提高了精度,可以有效地抑制光渗现象. 该方法的性能开销和存储开销与层数呈线性变化,因此可以合理地调整层数以适应不同的应用需求. 文献<sup>[55]</sup>提出另一种消除光渗的方法,它通过在对 VSM 进行单独滤波处理的过程中同时生成一个最小-最大阴影图来增加绘制时所需的额外深度信息,从而减少光渗现象.

另一种对阴影图进行的滤波处理方法称为卷积阴影图(convolution shadow maps, CSM)<sup>[56]</sup>,它将阴影图看作一幅普通的二值图像,可以对该图像进行各种数字图像处理. 在该方法中,首先对阴影图进行傅里叶变换,然后对进行了傅里叶变换后的傅里叶基函数作卷积处理(卷积处理可以平滑阴影边界),最后在渲染阶段使用变换后的基函数重构阴影图. CSM 可以移植到当前的可编程硬件上,只需对标准的阴影图渲染管线进行修改:在完成了第一阶段的阴影图生成后,对保存深度值的阴影图进行傅里叶变换,得到一组基函数,将其存储到一组纹理中;然后对这些纹理进行滤波处理,最后在渲染阶段重构阴影图,并绘制最终的图像. 这种方法也可以解决光渗问题,但是占用内存比较多.

一种与 CSM 类似的方法称为指数阴影图(exponential shadow maps, ESM)<sup>[57]</sup>,其思想也是使用某种展开项近似阴影测试. 该方法与 CSM 最大的不同是其使用的是一个指数基函数,而不像 CSM 使用一组正弦函数,这样可以大大地减少存储开销,同时提高渲染速度.

3 综合讨论

当考虑完上面的 3 个问题之后,就可以对特定的场景选择最合适的阴影图. 选择依据除了场景外,还包括光源类型、内存消耗、实时性要求、硬件支持、特殊阴影要求等因素.

1) 场景. 是室内场景还是室外场景,场景范围有多大,根据场景大小判断是否需要分割场景,以及如何分割. 一般在室外场景下需要对其进行分割处理.

2) 光源类型. 场景中是否包含全向光、多光源、面光源或者其他复杂光源.

3) 内存消耗. 一些生成高质量阴影的算法需要消耗大量的内存,如果系统中有内存方面的限制,则存储消耗大的算法无法正常使用.

4) 实时性要求. 如果在实时应用中使用阴影图,则需要考虑其实时性,有些算法虽然可以生成高真实感的阴影,但是实时性差,有时为了满足实时性要求需要牺牲一些物理真实性.

5) 硬件支持. 根据系统使用的硬件选择合适的算法. 大部分可以移植到硬件上的阴影图算法需要可编程 GPU 的支持,若系统不支持可编程,则无法使用该类算法. 有些阴影图算法需要比较新的图形硬件的支持,如几何渲染器等.

6) 特殊阴影要求. 场景中如果包含半透明物体,或者毛发、烟雾等,则需要使用特殊的阴影图产生特殊的阴影.

为了针对特殊场景,或者为了生成高质量的阴影,可以综合使用以上多项算法,但是算法越多就越复杂,导致性能下降. 表 1 所示为几种常见场景下的阴影图算法选择,其中 standard 表示使用传统阴影图算法的默认处理方式.

表 1 不同场景阴影图选择建议

应用场景	构造阴影图		
	数量	参数化	存储内容及滤波
室内场景,含有多光源,视点变化的静态场景,要求交互式响应,如虚拟博物馆漫游系统	ASM	TSM	standard
大规模室外场景,如飞行仿真系统	PSSM	PSM	PCF
含有全向光源的场景,需要较高的真实感	cube maps	LogPSM	ESM
对于含有毛发或烟雾的场景产生质量较高的阴影效果	standard	LisPSM	PCF+DSM
存在全向光源的大规模场景,需要实时响应	cube maps+PSSM	PSM	standard

## 4 进一步研究方向

阴影在3D图形显示中起着非常重要的作用,它不仅增强图形的真实感,还可以提供场景的各种空间信息。本文主要介绍各种当前流行的基于阴影图的阴影生成技术,并对该领域近年来的研究从阴影图的数量、参数化方式和存储内容及滤波处理3个方面做了个总结,指出了各种阴影图算法的使用环境和优缺点。

对阴影图算法的进一步研究可以从以下几个方面进行:

1) 与其他阴影生成技术的结合。近年来,出现了一些阴影图算法和其他阴影生成算法综合的混合算法,如文中介绍的阴影图使用到全局光照算法中的2种技术。最常见的还有阴影图和阴影体算法的混合<sup>[58-59]</sup>,以同时提高阴影绘制效率和阴影质量。阴影图技术可以混合其他一些绘制技术以促进自身的优化,从而增强阴影的物理正确性和绘制实时性。

2) 利用可编程GPU进一步提高阴影图算法效率。目前的大多数阴影算法都可以移植到可编程的GPU上计算,随着GPU技术的发展,一些新的功能出现,阴影图算法可以有效地利用这些新功能来提升自身性能。

3) 处理复杂光源。更复杂的光源包括任意形状的面光源、体光源和多面光源、多体光源,以及复杂的环境光。这些光源类型虽然可以生成复杂的软阴影,增强场景的真实感,但是却增加了使用阴影图生成阴影的难度。目前的大多数阴影图算法在处理这类光源方面有一定的局限性,如何降低存储消耗,并提高绘制效率是进一步研究的重点之一。

4) 处理复杂场景环境或动态场景。复杂的场景包括一些特殊场景,如水下环境<sup>[60]</sup>、浓雾环境、雨景、雪景下的阴影绘制等,这类场景下的阴影涉及到一些物理模拟过程,生成高真实感的阴影比较复杂。动态场景<sup>[61]</sup>的阴影实时绘制是该领域另外一个研究热点,动态场景下需要考虑真实感和实时性的平衡。

## 参考文献(References):

[1] Watt A, Policarpo F. 3D Games, real-time rendering and software technology [M]. Beijing: China Machine Press, 2005 (in Chinese)

([英]Watt A, Policarpo F. 沈一帆,等译. 3D游戏,实时渲染与软件技术[M]. 北京:机械工业出版社,2005)

[2] Crow F C. Shadow algorithms for computer graphics [C] // Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, San Jose, 1977: 242-248

[3] Williams L. Casting curved shadows on curved surface [C] // Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Atlanta, 1978: 270-274

[4] Woo A, Poulin P, Fournier A. A survey of shadow algorithms [J]. IEEE Computer Graphics and Applications, 1990, 10(6): 13-32

[5] Hasenfratz J M, Lapierre M, Holzschuch N, *et al.* A survey of real-time soft shadows algorithms [J]. Computer Graphics Forum, 2003, 22(4): 753-774

[6] Akenine-Möller T, Haines E. Real-time rendering [M]. 2nd ed. Beijing: Peking University Press, 2004 (in Chinese) (Akenine-Möller T, Haines E. 普建涛,译. 实时计算机图形学[M]. 2版. 北京:北京大学出版社,2004)

[7] Wu Enhua, Liu Youquan. General purpose computation on GPU [J]. Journal of Computer-Aided Design & Computer Graphics, 2004, 16(5): 601-612 (in Chinese) (吴恩华,柳有权. 基于图形处理器(GPU)的通用计算[J]. 计算机辅助设计与图形学学报,2004,16(5): 601-612)

[8] Blinn J F, Newell M E. Texture and reflection in computer generated images [J]. Communications of the ACM, 1976, 19(10): 542-547

[9] Greene N. Environment mapping and other applications of world projections [J]. IEEE Computer Graphics and Applications, 1986, 6(11): 21-29

[10] Brabec S, Annen T, Seidel H P. Shadow mapping for hemispherical and omnidirectional light sources [C] // Proceedings of Computer Graphics International, Bradford, 2002: 397-408

[11] Haeblerli P, Akeley K. The accumulation buffer: hardware support for high-quality rendering [J]. Computer Graphics, 1990, 24(4): 309-318

[12] Heckbert P S, Herf M. Simulating soft shadows with graphics hardware [R]. Pittsburgh: Carnegie Mellon University, 1997

[13] Chen S E, Williams L. View interpolation for image synthesis [C] // Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Anaheim, 1993: 279-288

[14] Agrawala M, Ramamoorthi R, Heinrich A, *et al.* Efficient image-based methods for rendering soft shadows [C] // Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New Orleans, 2000: 375-384

[15] Heidrich W, Brabec S, Seidel H P. Soft shadow maps for linear lights [C] // Proceedings of the Eurographics Workshop on Rendering Techniques, London, 2000: 269-280

[16] Ying Z M, Tang M, Dong J X. Soft shadow maps for area light by area approximation [C] // Proceedings of 10th Pacific Conference on Computer Graphics and Applications, Beijing, 2002: 442-443

- [17] Atty L, Holzschuch N, Lapierre M, *et al.* Soft shadow maps: efficient sampling of light source visibility [J]. Computer Graphics Forum, 2006, 25(4): 725-741
- [18] Guennebaud G, Barthe L, Paulin M. Real-time soft shadow mapping by backprojection [C] //Proceedings of Eurographics Symposium on Rendering, Nicosia, 2006: 227-234
- [19] Guennebaud G, Barthe L, Paulin M. High-quality adaptive soft shadow mapping [J]. Computer Graphics Forum, 2007, 26(3): 525-534
- [20] Schwarz M, Stamminger M. Bitmask soft shadows [J]. Computer Graphics Forum, 2007, 26(3): 515-524
- [21] Bavoil L, Callahan S P, Silva C T. Robust soft shadow mapping with backprojection and depth peeling [J]. Journal of Graphics Tools, 2008, 13(1): 19-29
- [22] Tadamura K, Qin X G, Jiao G, *et al.* Rendering optimal solar shadows with plural sunlight depth buffers [J]. The Visual Computer, 2001, 17(2): 76-90
- [23] Lloyd D B, Tuft D, Yoon S E, *et al.* Warping and partitioning for low error shadow maps [C] //Proceedings of Eurographics Symposium on Rendering, Nicosia, 2006: 215-226
- [24] Zhang F, Sun H Q, Xu L L, *et al.* Parallel-split shadow maps for large-scale virtual environments [C] //Proceedings of ACM International Conference on Virtual Reality Continuum and its Applications, Hong Kong, 2006: 311-318
- [25] Zhang F, Sun H Q, Nyman O. Parallel-split shadow maps on programmable GPUs [M]. Nguyen H. GPU Gems 3. Boston: Addison-Wesley, 2007: 203-238
- [26] Fernando R, Fernandez S, Bala K, *et al.* Adaptive shadow maps [C] //Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Los Angeles, 2001: 387-390
- [27] Lefohn A, Sengupta S, Kniss J, *et al.* Dynamic adaptive shadow maps on graphics hardware [C] //Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Los Angeles, 2005; Disc 2
- [28] Giegl M, Wimmer M. Queried virtual shadow maps [C] // Proceedings of Symposium on Interactive 3D Graphics and Games, Seattle, 2007: 65-72
- [29] Lefohn A E. Resolution-matched shadow maps [J]. ACM Transactions on Graphics, 2007, 26(4): 20:1-20:17
- [30] Arvo J. Tiled shadow maps [C] //Proceedings of the Computer Graphics International, Washington D C, 2004: 240-247
- [31] Ritschel T, Grosch T, Kautz J, *et al.* Interactive illumination with coherent shadow maps [C] //Proceedings of Eurographics Symposium on Rendering, Grenoble, 2007: 61-72
- [32] Ritschel T, Grosch T, Kautz J, *et al.* Interactive global illumination based on coherent surface shadow maps [C] // Proceedings of Graphics Interface, Ontario, 2008: 185-192
- [33] Stamminger M, Drettakis G. Perspective shadow maps [C] // Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, San Antonio, 2002: 557-562
- [34] Wimmer M, Scherzer D, Purgathofer W. Light space perspective shadow maps [C] //Proceedings of Eurographics Symposium on Rendering, Norrköping, 2004: 143-151
- [35] Kozlov S. Perspective shadow maps: care and feeding [M] // GPU Gems. Boston: Addison-Wesley, 2004: 217-244
- [36] Lloyd D B, Govindaraju N K, Quammen C, *et al.* Logarithmic perspective shadow maps [J]. ACM Transactions on Graphics, 2008, 27(4): 1-39
- [37] Martin T, Tan T S. Anti-aliasing and continuity with trapezoidal shadow maps [C] //Proceedings of Eurographics Symposium on Rendering, Norrköping, 2004: 153-160
- [38] Aila T, Laine S. Alias-free shadow maps [C] //Proceedings of Eurographics Symposium on Rendering, Norrköping, 2004: 161-166
- [39] Johnson G, Mark W R, Burns C. The irregular Z-buffer and its application to shadow mapping [D]. Austin: The University of Texas at Austin, 2004
- [40] Arvo J. Alias-free shadow maps using graphics hardware [J]. Journal of Graphics, GPU, & Game Tools, 2007, 12(1): 47-59
- [41] Sintorn E, Eisemann E, Assarsson U. Sample-based visibility for soft shadows using alias-free shadow maps [J]. Computer Graphics Forum, 2008, 27(4): 1285-1292
- [42] Woo A. The shadow depth map revisited [M] //Graphics Gems 3. Boston: Academic Press, 1992: 338-342
- [43] Weiskopf D, Ertl T. Shadow mapping based on dual depth layers [C] //Proceedings of Eurographics, Granada, 2003: 53-60
- [44] Hourcade J C, Nicolas A. Algorithms for antialiased cast shadows [J]. Computers & Graphics, 1985, 9(3): 259-265
- [45] Dachsbacher C, Stamminger M. Reflective shadow maps [C] //Proceedings of Symposium on Interactive 3D Graphics and Games, New York, 2005: 203-231
- [46] Lokovic T, Veach E. Deep shadow maps [C] //Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New Orleans, 2000: 385-392
- [47] Xie F, Tabellion E, Pearce A. Soft shadows by ray tracing multilayer transparent shadow maps [C] //Proceedings of the Eurographics Symposium on Rendering, Grenoble, 2007: 265-276
- [48] Kim T Y, Neumann U. Opacity shadow maps [C] // Proceedings of the 12th Eurographics Workshop on Rendering Techniques, London, 2001: 177-182
- [49] Yuksel C, Keyser J. Deep opacity maps [J]. Computer Graphics Forum, 2008, 27(2): 675-680
- [50] Reeves W T, Salesin D H, Cook R L. Rendering antialiased shadows with depth maps [J]. Computer Graphics, 1987, 21(4): 283-291
- [51] Brabec S, Seidel H P. Hardware-accelerated rendering of antialiased shadows with shadow maps [C] //Proceedings of the International Conference on Computer Graphics, Washington D C, 2001: 209-214

- [52] Zhao Nailiang, Chen Yanjun, Pan Zhigeng. Anti-aliasing shadow maps based on data shift [J]. Journal of Computer-Aided Design & Computer Graphics, 2006, 18(8): 1130-1135 (in Chinese)  
(赵乃良, 陈艳军, 潘志庚. 基于数据修正的实时阴影反走样算法[J]. 计算机辅助设计与图形学学报, 2006, 18(8): 1130-1135)
- [53] Donnelly W, Lauritzen A. Variance shadow maps [C] // Proceedings of Symposium on Interactive 3D Graphics and Games, Redwood Shores, 2006: 161-165
- [54] Lauritzen A, McCool M. Layered variance shadow maps [C] // Proceedings of Graphics Interface, Windsor, 2008: 139-146
- [55] Xue Gaichao, Lu Weiwei, Liu Xuehui. Removal of light bleeding in variance shadow maps [J]. Journal of Computer-Aided Design & Computer Graphics, 2009, 21(2): 165-171 (in Chinese)  
(薛盖超, 吕伟伟, 刘学慧. 方差阴影图中的光渗现象消除算法[J]. 计算机辅助设计与图形学学报, 2009, 21(2): 165-171)
- [56] Annen T, Mertens T, Bekaert P, *et al.* Convolution shadow maps [C] // Proceedings of Eurographics Symposium on Rendering, Grenoble, 2007: 51-60
- [57] Annen T, Mertens T, Seidel H P, *et al.* Exponential shadow maps [C] // Proceedings of Graphics Interface, Windsor, 2008: 155-161
- [58] Chan E, Durand F. An efficient hybrid shadow rendering algorithm [C] // Proceedings of Eurographics Symposium on Rendering, Norrköping, 2004: 185-195
- [59] Aila T, Akenine-Möller T. A hierarchical shadow volume algorithm [C] // Proceedings of the ACM SIGGRAPH/Eurographics Conference on Graphics Hardware, Grenoble, 2004: 15-23
- [60] Liu Li, Ma Lizhuang. Under water shadow generation based on GPU [J]. Journal of Computer-Aided Design & Computer Graphics, 2008, 20(4): 494-498 (in Chinese)  
(刘 力, 马利庄. 基于 GPU 的水底阴影[J]. 计算机辅助设计与图形学学报, 2008, 20(4): 494-498)
- [61] Annen T, Zhao D, Mertens T, *et al.* Real-time all-frequency shadows in dynamic scenes [C] // Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Los Angeles, 2008: 34:1-34:87