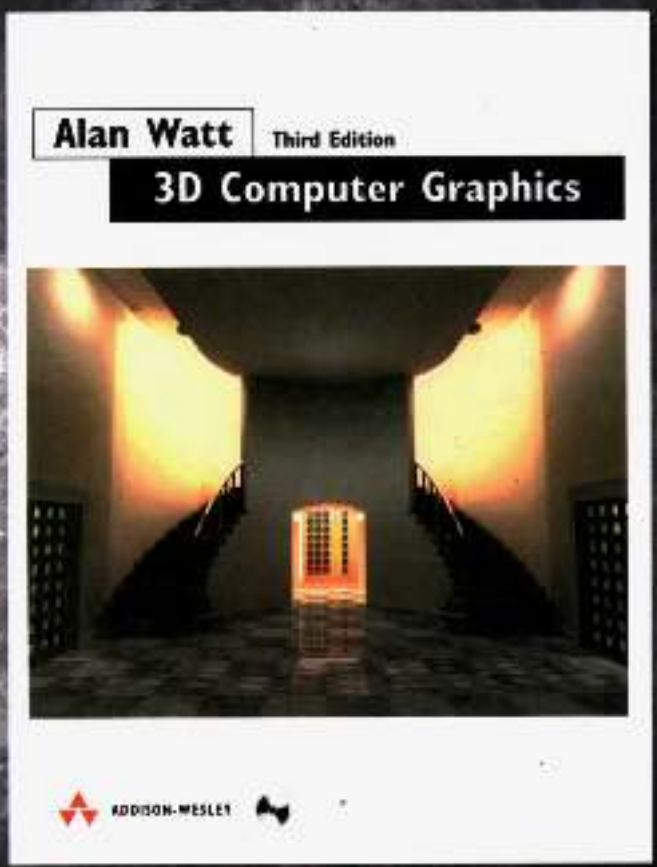
计 算 机 科 学 丛 书

原书第3版

3D计算机图形学

(英) Alan Watt 著 包宏 译



3D Computer Graphics

Third Edition

机械工业出版社
China Machine Press

本书讨论将物体的数学或几何描述转换成可视产品时所涉及的过程——数学或几何描述是一种计算机图形学模型，而可视产品是模拟真实物体外观的二维投影。此外，本书还涵盖了计算机图形学的新进展，主要包括：

- 高级辐射度方法。
- 动画。
- 预计算技术。
- 具有高复杂性的实时应用，例如渐进式网格优化、BSP树、照片建模技术等。

随书光盘中含有400幅图形和几个计算机图形学程序。

作者简介

Alan Watt 英国谢菲尔德大学计算机科学系高级讲师和计算机图形学研究室主任，曾编写过多本优秀教材，除本书外，他还与人合著有《3D游戏》（已由机械工业出版社引进出版）、《The Computer Image》等。



ISBN 7-111-16513-6

9 787111 165132

HZ BOOKS

华章图书

上架指导：计算机，图形学

华章网站 http://www.hzbook.com

网上购书：www.china-pub.com

投稿热线：(010) 88379604

购书热线：(010) 68995259, 68995264

读者信箱：hzjsj@hzbook.com

ISBN 7-111-16513-6/TP・4288

定价：69.00 元（附光盘）



图1-5 多边形网格模型上的全局变换——一个经扭曲和变细的波纹环面柱体（经Steve Maddock许可）



a)



b)



c)

图3-42 将FFD应用于一个多边形网格物体。a）物体的线框，b）用非常小的曲面片网格（显示为半透明灰色盒子）绘制的物体，c）在曲面片中移动控制点引起物体模型以一定的形式变形



一个物体（基于Salvador Dali的一个著名绘画）



点产生器——每个球的半径都受到产生器的影响





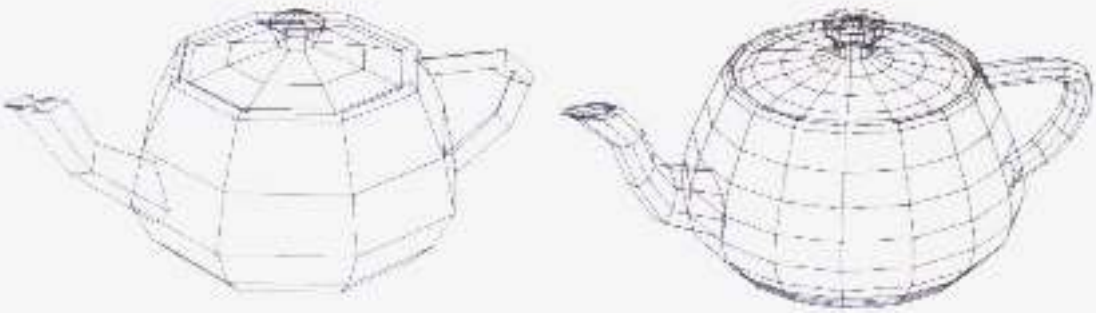
若两产生器的移动产生不希望的结合





随着产生器的移动产生不希望的分离

图2.20　一个隐函数造模系统的例子（经Agata Opalach许可）



分别由128个和512个多边形绘制的图像



图4-9 以八次均匀细分水平绘制的参数曲面片（128、512、2048和8192个多边形）（经Steve Maddock许可）



图6-12 演示Phong明暗处理的“传统”方法。k_s和k_d保持为常数，k从左到右是增加的，而指数从上到下是增加的。模型试图通过增加指数来增加闪光效果，这使得镜面高光的延伸区域变小，可以将其解释为一个变动大小的光源的反射（光线是一个点光源）

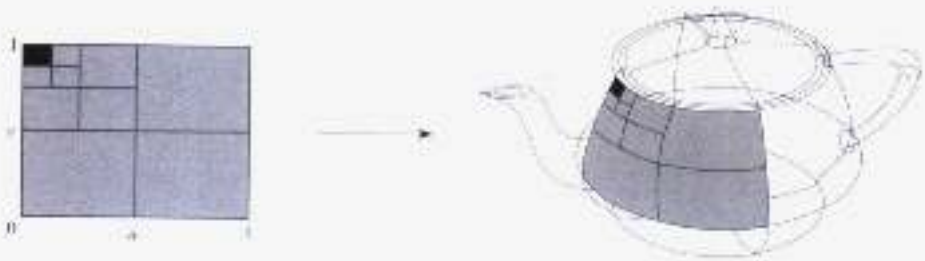


图7-8 旧7.6节中介绍的模型模拟的一些材料。某些材料（例如，抛光的铜和金）之间的差别可能很难通过对Phong明暗处理模型进行细微的参数调节来获得。在这些图像中，把反射模型用作光线跟踪程序的局部分量



a) b) c)

图8-7 带有旋转实体的两段纹理映射的例子。中间表面为：a）一个平面（或者没有表面）；b）一个圆柱体；c）一个



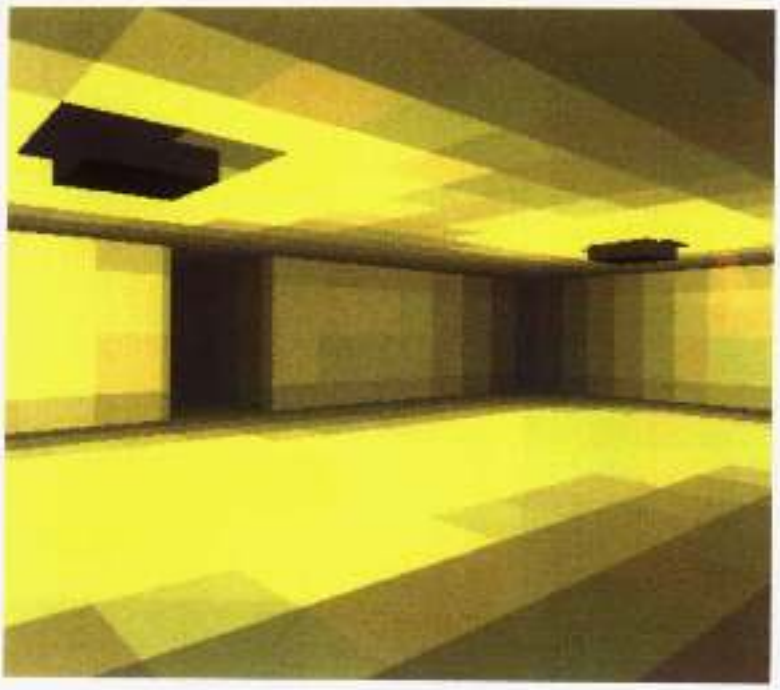
纹理图 一个Bézier曲面片



图8-8 （左）纹理图，（右）物体上的一个Bézier曲面片，（下）渲染的茶壶（经Steve Maddock许可）



图8-10 凹凸映射。a) 一个凹凸映射的物体和一个凹凸映射图；b) 由程序化产生的高度场建立的凹凸映射的物体；c) 将凹凸映射和颜色映射相结合；d) 用于c的凹凸图和颜色图



a)

b)

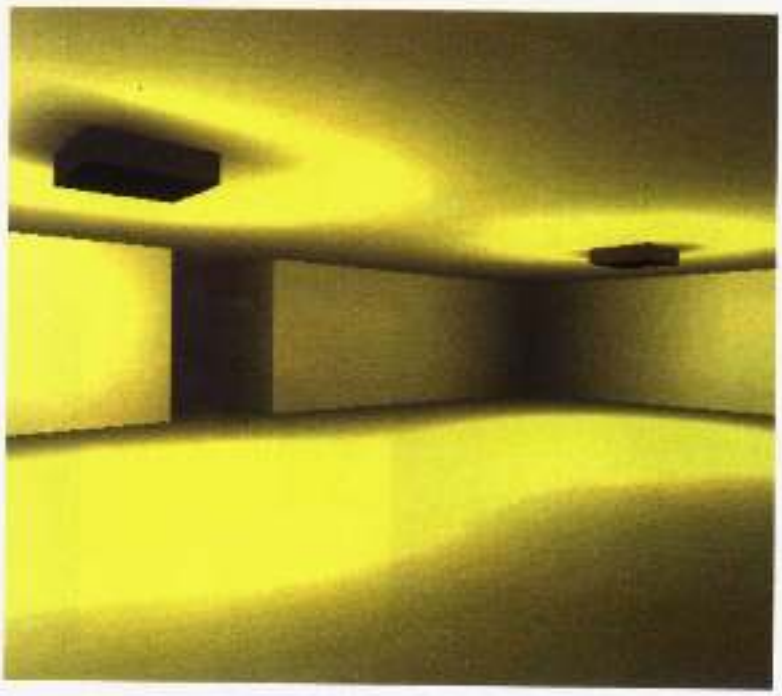


图8-14 用光线图来表现的一个简单场景。a) 在这个图像中，显示了场景中照明器的尺寸，b) 在这个图像中使用了一种称为纹理插值的双线性插值的技术来消除a中照明点的可见性

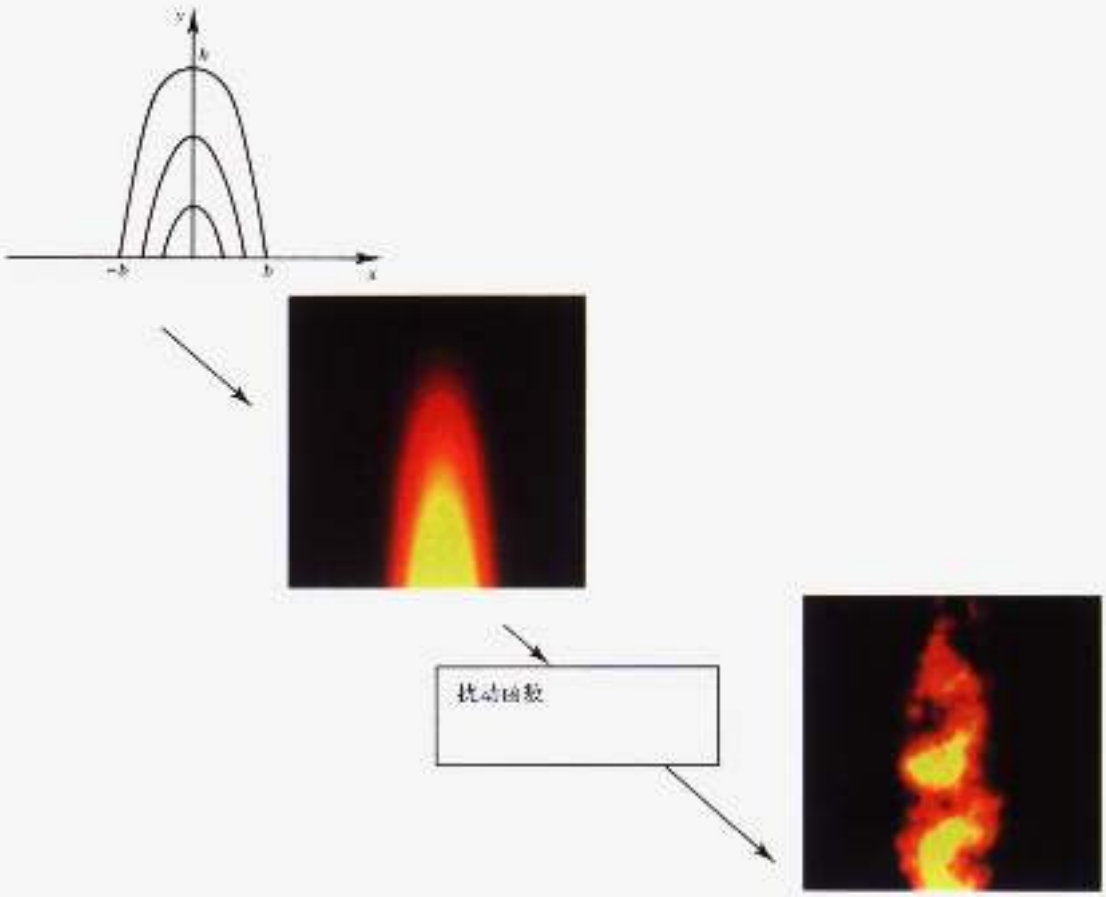


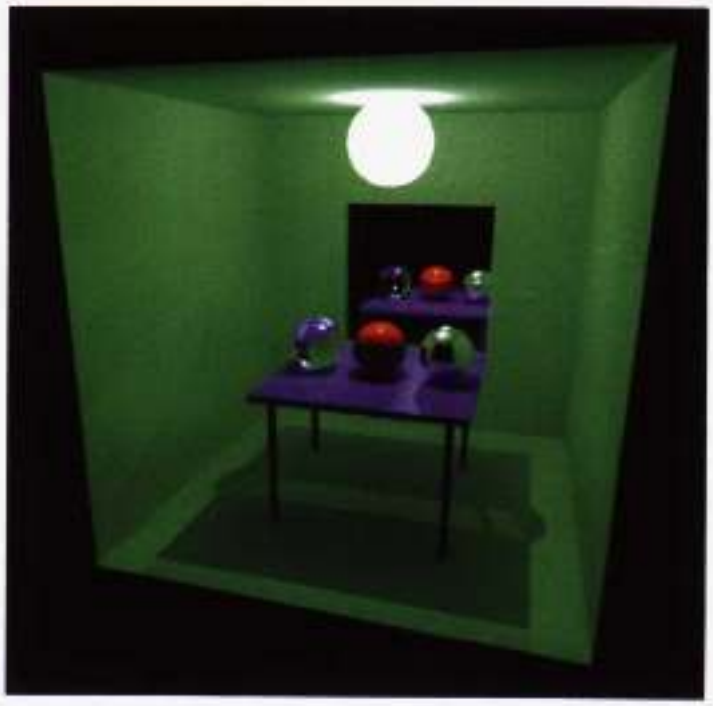
图8-22 用一个扰动函数模拟火焰。（上）未受扰动的火焰；（右）受扰动的火焰



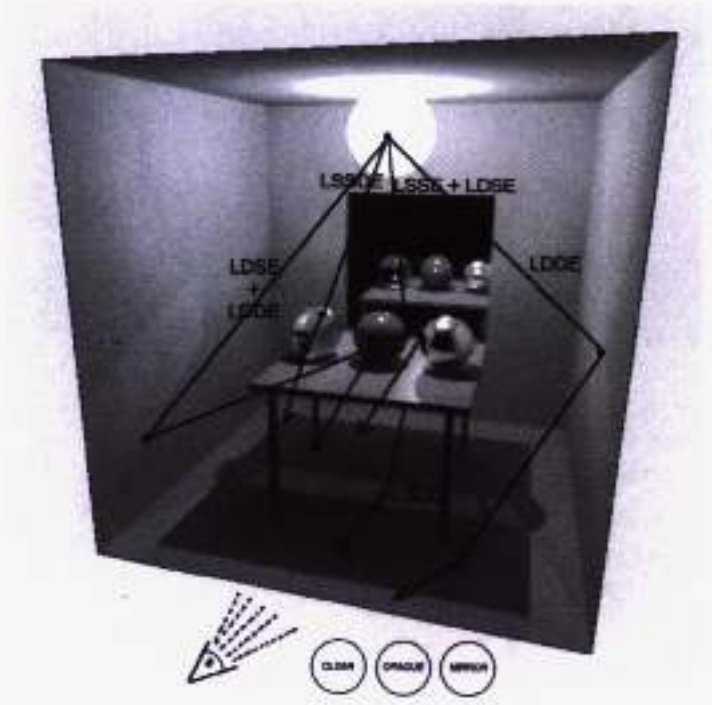
图8-21 模拟大理石——一维程序式纹理的经典例子



图8-26 图8-8中使用的unip块别图



a)



b)

图10-4 a) 一个用RADIANCE产生的图像；b) 一组在图a中采用的全局光照路径

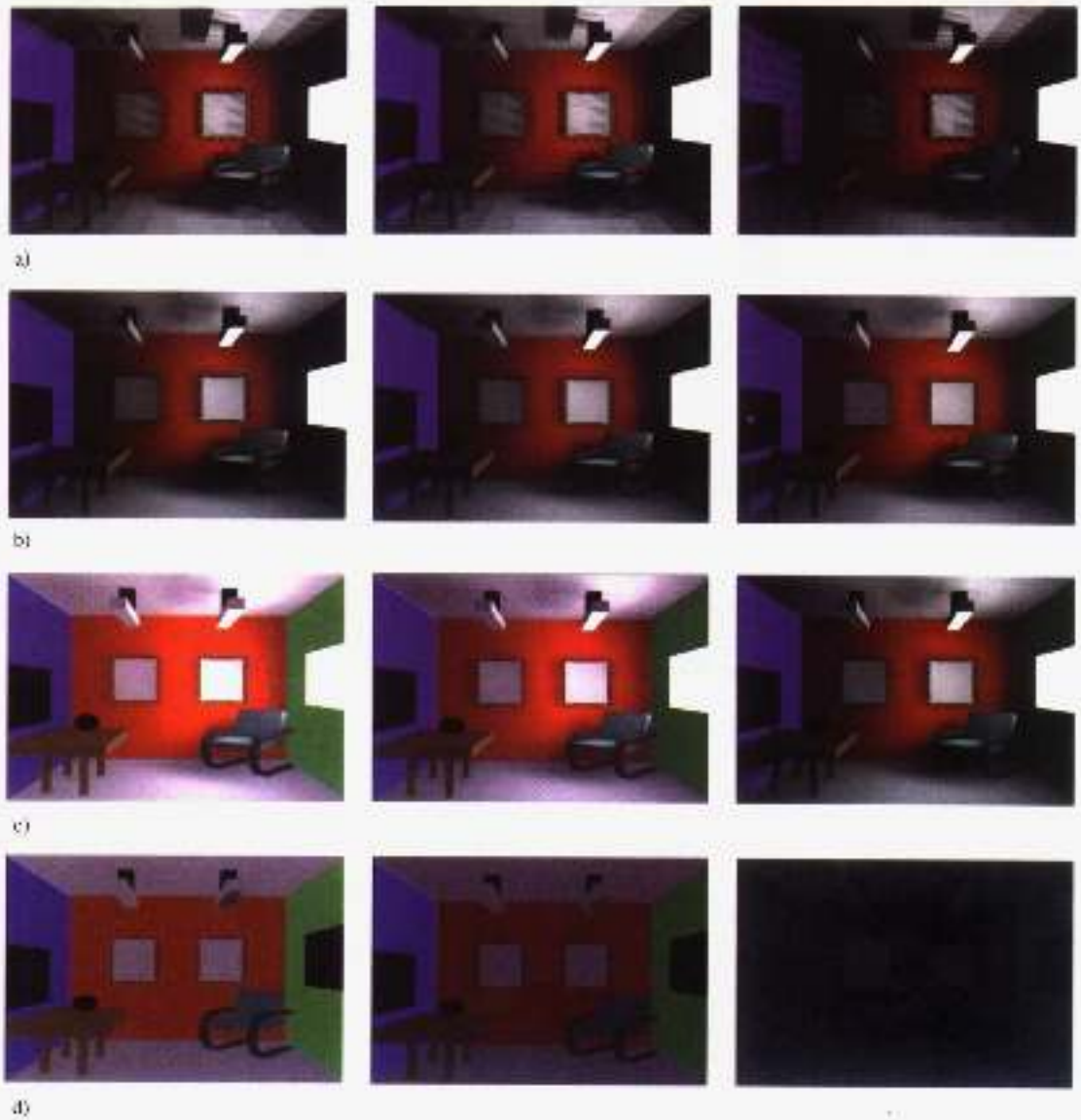


图10-7 渐近式细化方法进行20、250和5000次迭代之后的辐射度图像。从上到下的每一列分别为：a) 辐射度的解作为迭代过程的输出。每一个曲面片都分配了一个恒定的辐射度；b) 对前面的结果进行了插值处理；c) 同样的解，但加入了环境项；d) 前两个图像之差。这给出了必须加入未发射能量部分的辐射度的指示性信息



图10-14　用一个分布式光线跟踪程序绘制的场深度的效果

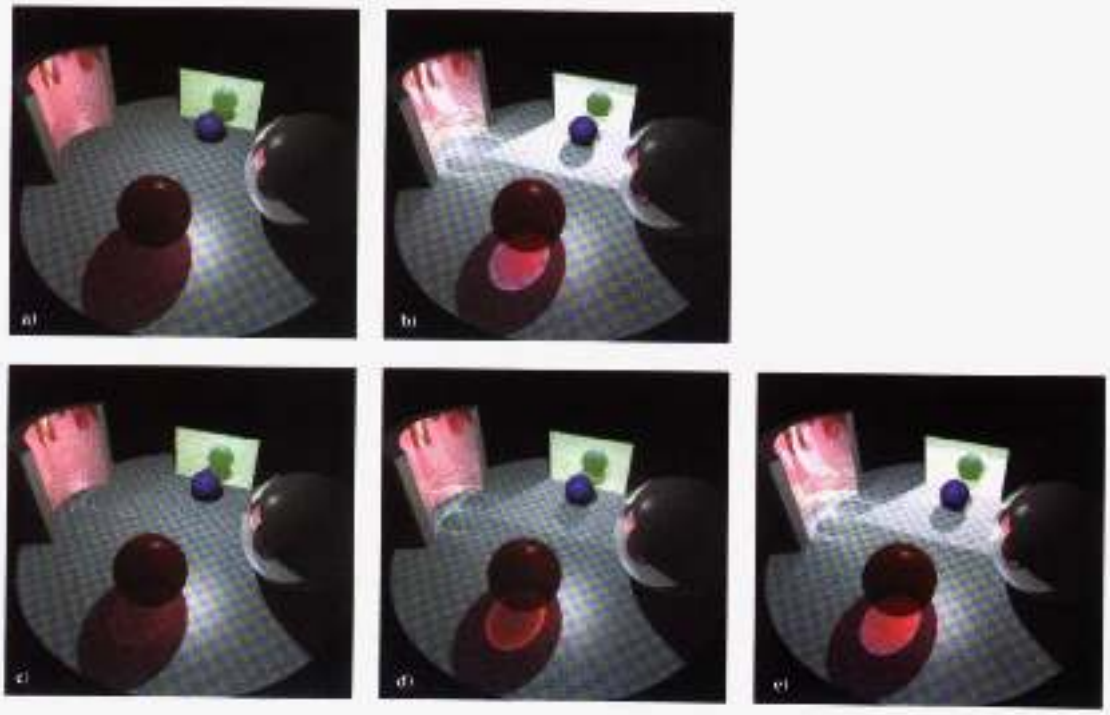


图10-20　二路光线跟踪的例子。a和b分别显示了用Whirted光线跟踪程序和二路光线跟踪程序绘制的场景

注：在这个场景中，有三个LSD路径：

- 两个来自红球的焦散——一个直接来自光线，另一个来自从弯曲的镜子反射出的光线。
- 一个来自圆柱镜子的反射的焦散面。
- 来自平面镜的二次照明（没有焦散的LSDE路径）。

c到e由增加发射出的光线数产生，并且表现出光线在漫反射表面散布的效果，随着光线路径上光线数量的增加，这些光线可以最终被合并，形成图像中的LSD路径。光线路径上发出光线的数量分别是200，400和800。

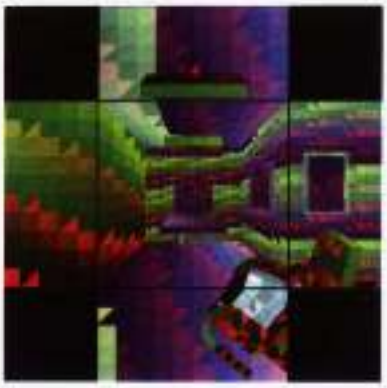


图11-5 显示一个放于窗户上的半立方体在将场景中所有其他曲面片都投影到其上之后的状态。一种颜色标识场景中可以被这个半立方体看到的一个曲面片（以及每一个部分的曲面片）。然后，算法简单地把与每一个曲面片有关的所有半立方体元素的形状因子加起来（这个图的场景示于图10-7中）



Whitted场景简单递归光线跟踪

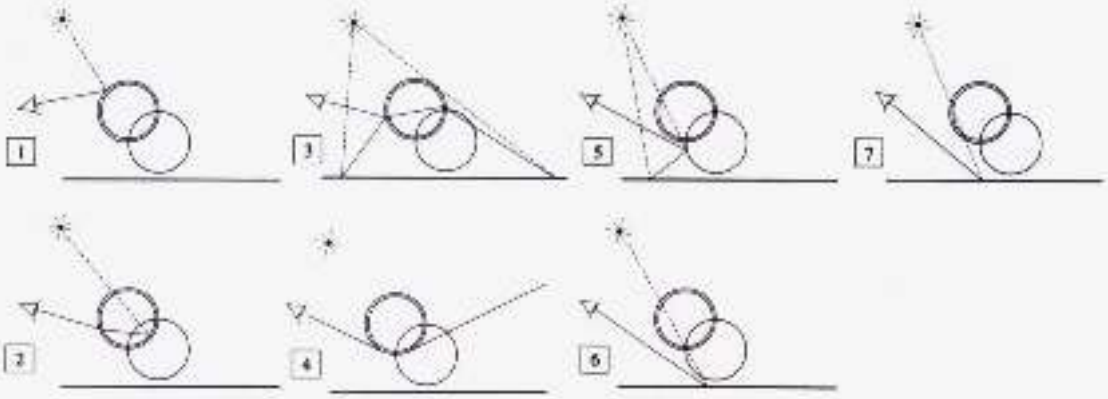
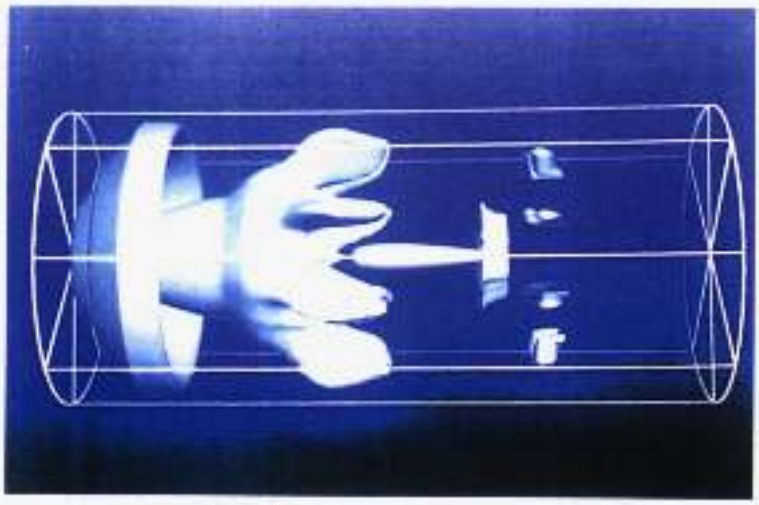


图12-4 Whitted场景



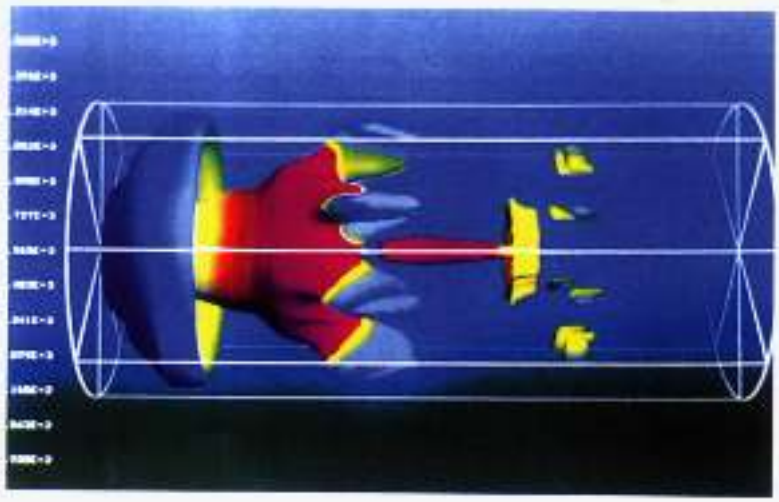


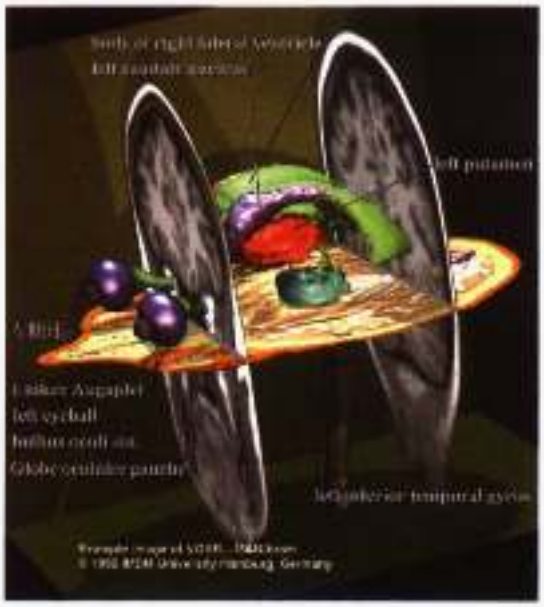
图13-1 步进式立方体和CFD数据。（上）一个反向流动管状燃烧器的Navier-Stokes CFD模拟。流动从左向右，热从右向左。在这些流体的界面上定义一个零速度的等值面。用步进式立方体算法抽取这个表面，然后对其进行绘制。（下）一个经纹理映射的零速度表面。伪彩色标量用于代表场强度，并与上面的示意图中明暗处理所用的颜色相结合（经Mark Fuller许可）



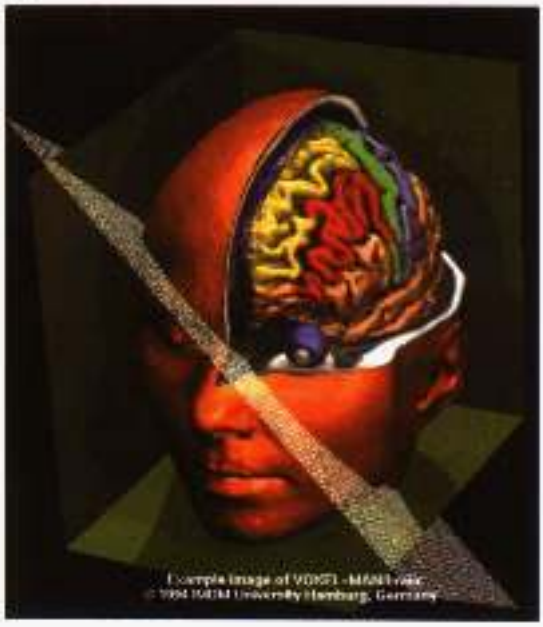
a)



b)



c)



d)

图13-3 a和b显示抽取嵌入在透明的颅骨包围物中的物体。抽取出来的结构已经转换成了计算机图形学的物体并被按正常情况进行了绘制。然后又将它们有效地嵌入到了三维数据体中，这个数据体与其周围被设为半透明的体素一起显示出来。c和d是切割已经绘制的表皮，以一种定位于三维模型中的截面显示其内部的组织。在这里，赋予该组织适当的伪颜色以便简单地表现出它的形状



图13-10 步进式立方体算法应用于X射线CT数据（经Klaus de Geuss许可）



图13-11 用体绘制的相同数据，骨骼的体素设为单位不透明度，其他彩数设为零

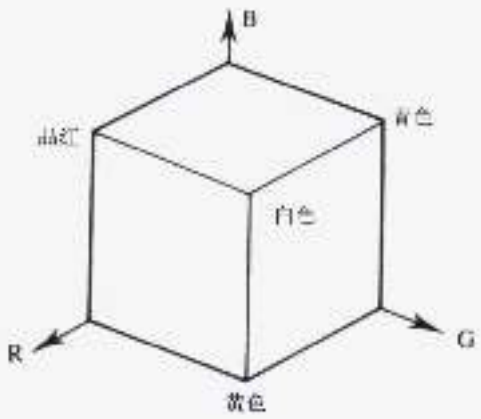




图15-3 RGB立方体



图15-5 HSV颜色模型，在值坐标上每一个小区域的间隔为20

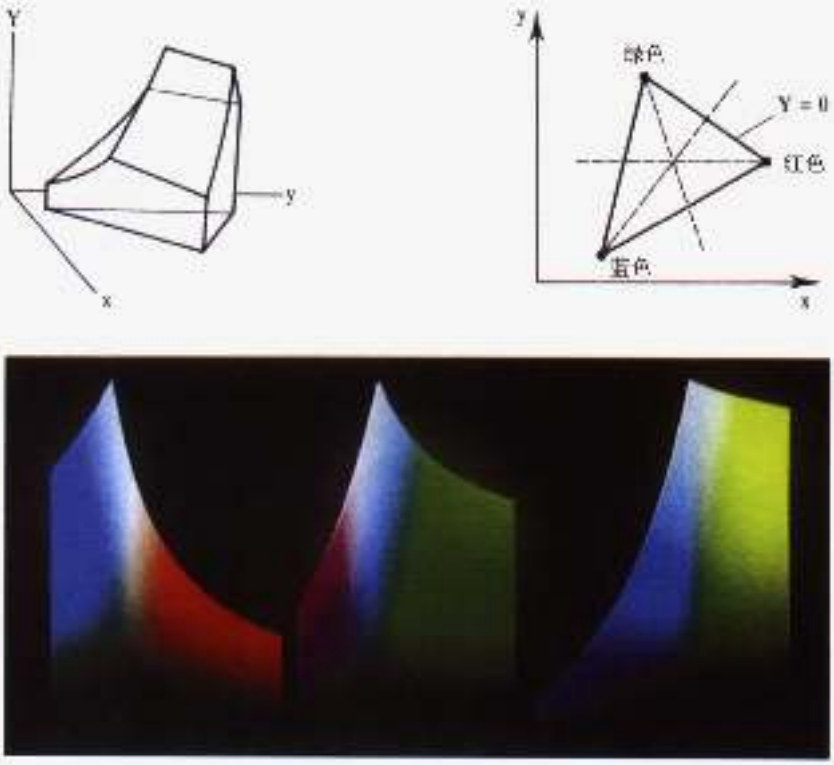


图15-10 （左上）监视器在CIE xyY空间的整个实体范围；（上）在实体CIE xyY空间中的一个截面；（左）在平面Y = 0处的截面的位置



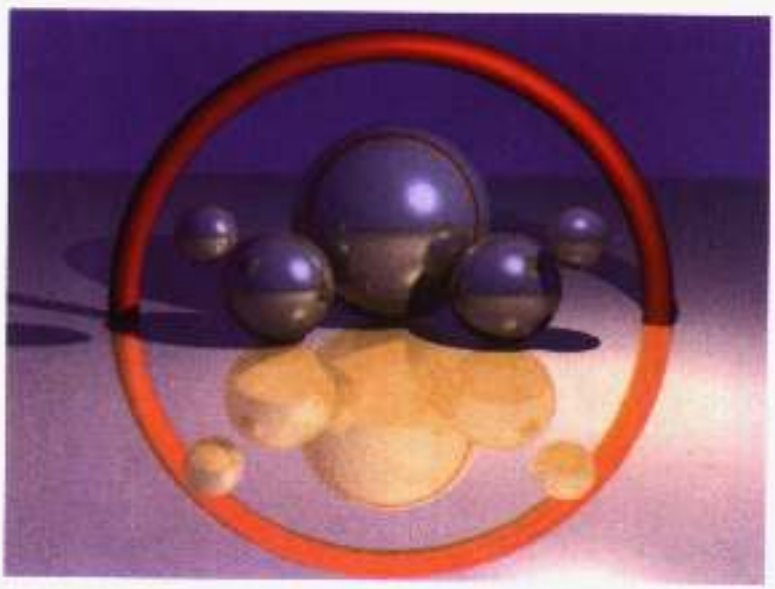


图15-12 对于光线跟踪的图像在镜面空间和RGB空间中绘制的比较

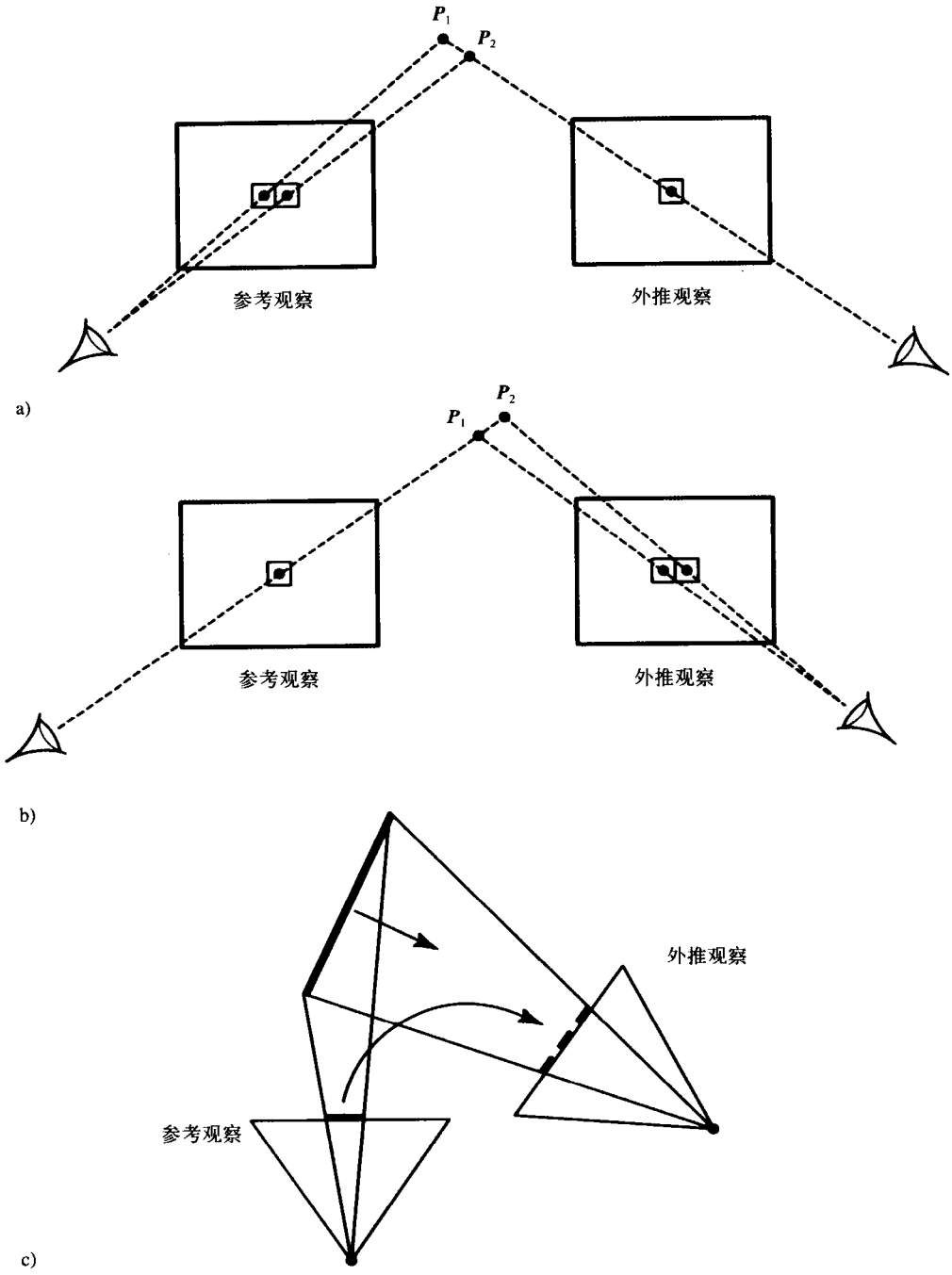
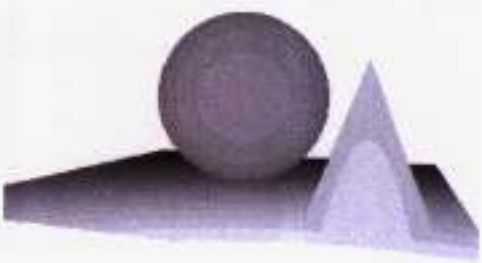


图16-8 图像扭曲中出现的问题。a) 图像打摺：参考观察中多个像素映射到外推观察中的一个像素上；b) 孔洞：在外推观察中需要有关在参考观察中被遮挡了的信息；c) 孔洞：由于其朝观察方向的旋转而使得一个表面的投影区域在外推观察中增加了



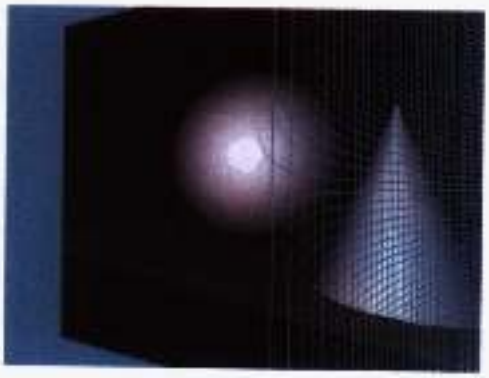
d1)



d2)



d3)



d4)



d5)

图16.8 （续）

d1和d2：一个简单的场景以及相应的Z缓冲器图像

d3：由于这个例子中的平移（只有平移）而产生的人工痕迹是由丢失信息和图像打搭而引起的孔洞

d4：由于旋转（只有旋转）而产生的人工痕迹是由增加了表面的投影区域而产生的孔洞。注意这些孔洞是如何形成连续模式的

d5：由旋转和平移引起的人工痕迹

1. 走一个旋转的摄像机形成重叠区域



2. “连接”到一个圆柱类的全景图像中



3. 一个被扭曲到平面多边形的断面

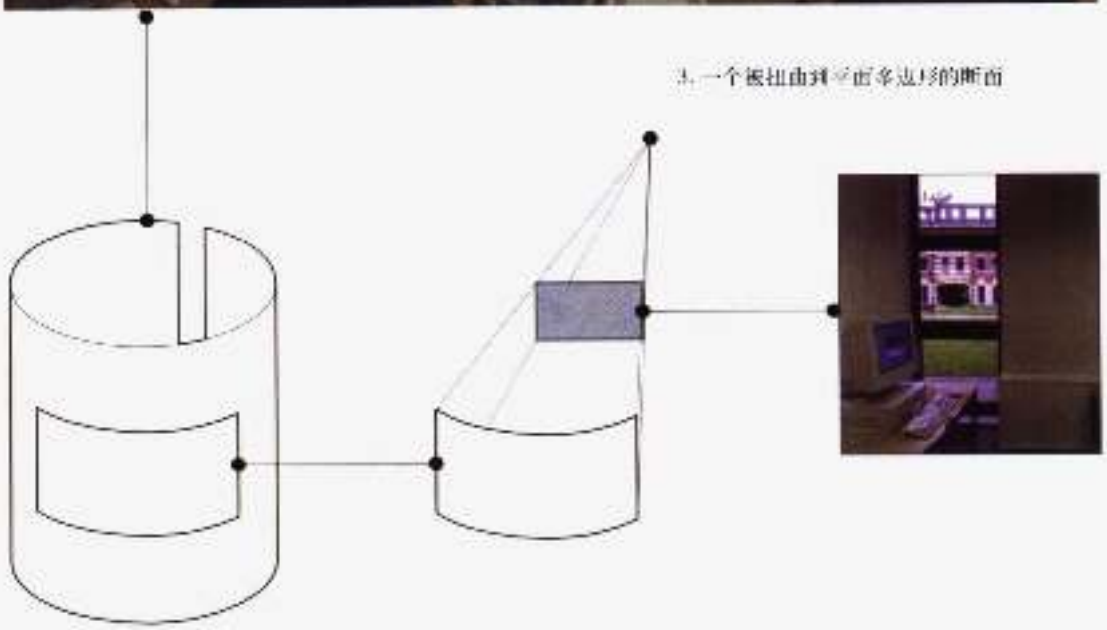


图16-19 QuickTime虚拟现实系统（经Guy Brown许可）

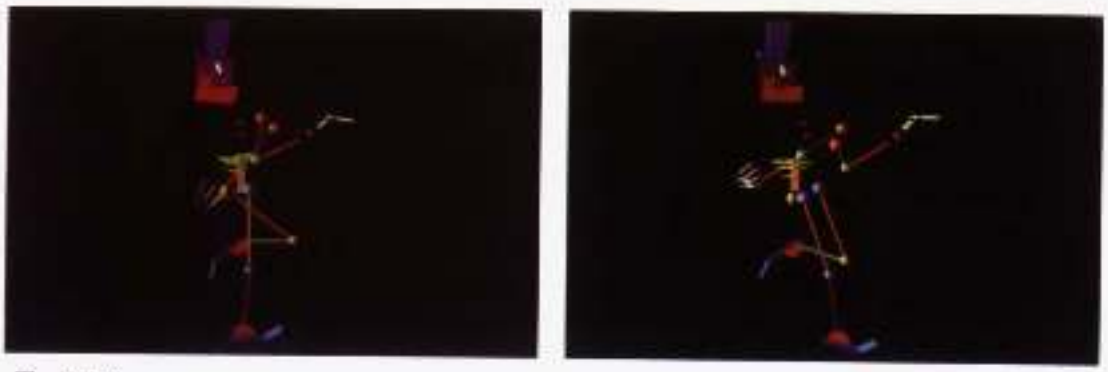


图 17-15

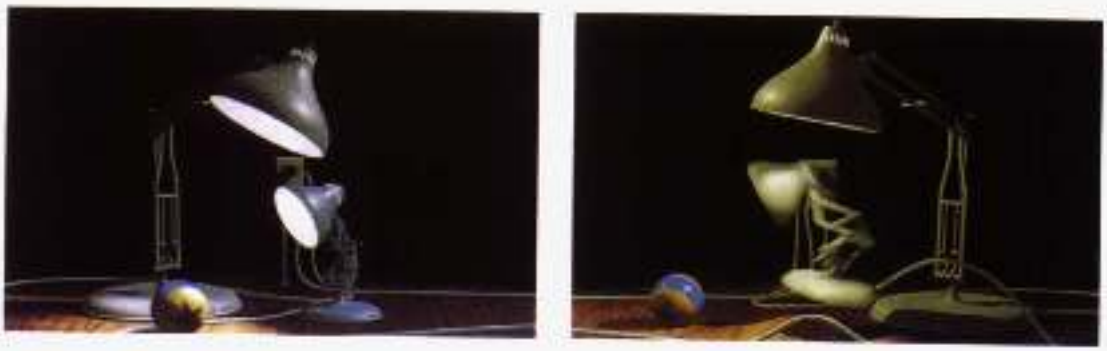


图17-20 （左）一个由John Lasseter、Bell Reeves、Eben Ostby以及Sam Leffler于1986年在Pixar公司生产的《Luxo Jr.》中的镜。Luxo是Jak Jacobsen工业公司的商标。这个短片由一个关键帧动画系统通过程序式的动画辅助进行动画，用多个光源和程序式的纹理技术对帧进行了绘制。（右）这一《Luxo Jr.》中的帧表现出了第10章中所介绍的运动模糊



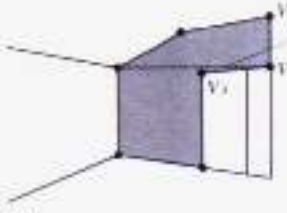
图18-1 一个办公室场景与线框可视图，只用恒定的环境项进行了明暗处理



图18-2 采用平面明暗处理的相同场景。平面明暗处理由于光强度的不连续性而表现出表面的多边形性质



a)



这里的不连续性是由于扫描线从边V_1V_2变到了边V_1V_3

b)



c)

图18-3 Gouraud插值的主要缺陷。a) 彩色图像，在这个图像中有两个缺陷（在正文中已进行了详细的描述）：马赫带（在重现时可能是不可见的）以及曲面上的插值缺陷；b) 点划线表示不连续性出现的位置；c) 需要有新的线框三角形划分来消除插值的缺陷

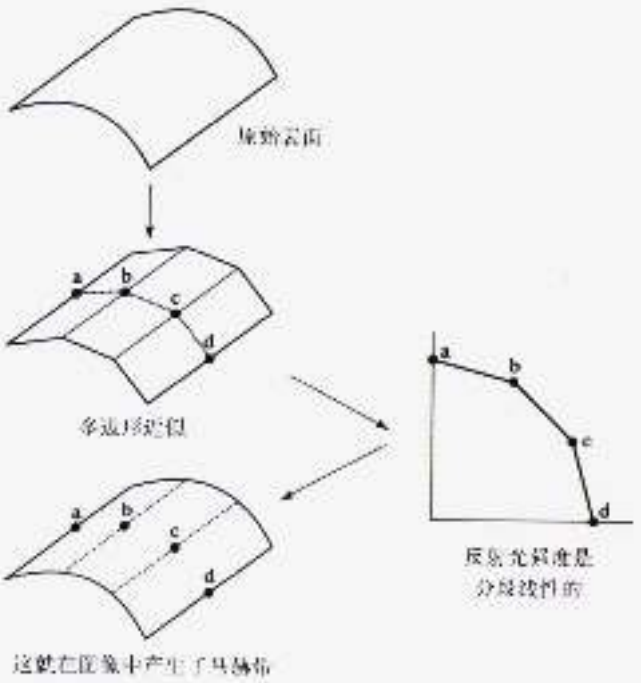


图18-4 Gouraud明暗处理中的马赫带



图18-5 用Phong明暗处理的相同场景。在这个图中表现了Phong插值中的闪烁缺陷。这里由壁灯反射的光以及光的图像由于插值的性质而被分离了

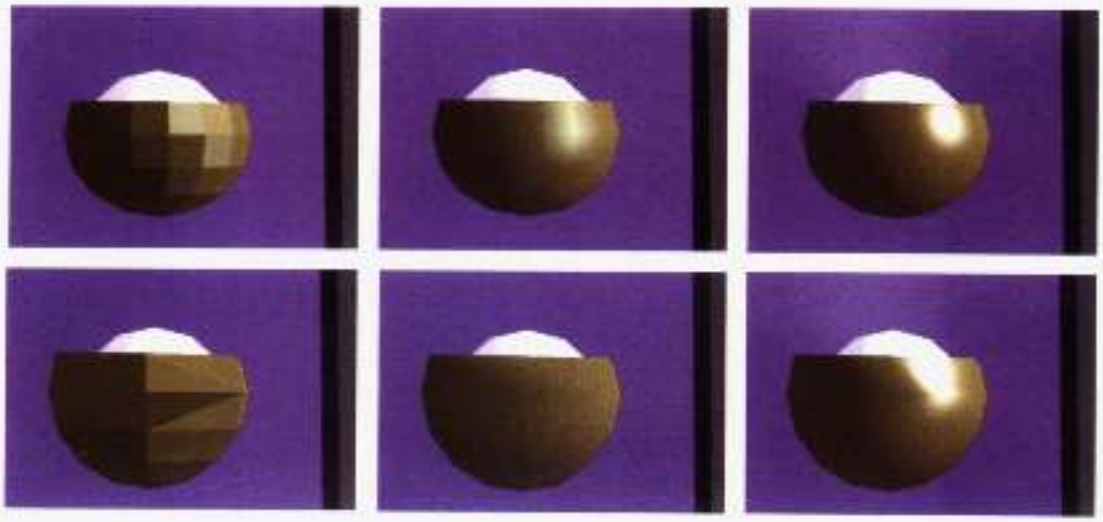


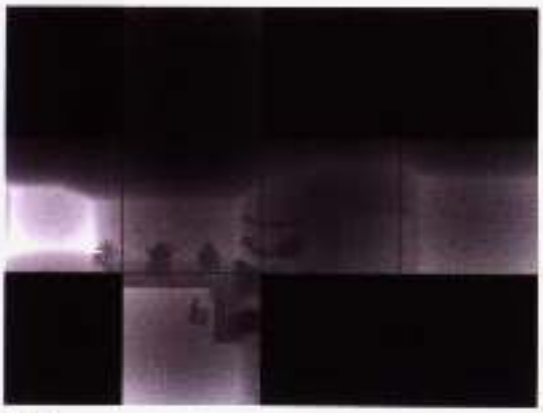
图18-6 一个球体的放大。上面一排比较了Gouraud明暗处理和Phong明暗处理。如果多边形网格明暗处理足够充分，则Gouraud明暗处理和Phong明暗处理之间的差别相当小。下面一排的多边形分辨率减小了，这时Gouraud明暗处理的高光消失了



图18-7 带“传统”的二维纹理的办公室场景



图18-8 增加了环境映射（镜面）以及阴影的相同场景



阴影图



环境图





图18-9 这是用环境映射（左边）和光线跟踪（右边）产生反射的图之间的比较



图18.10 用Whitted型光线跟踪程序所跟踪的场景

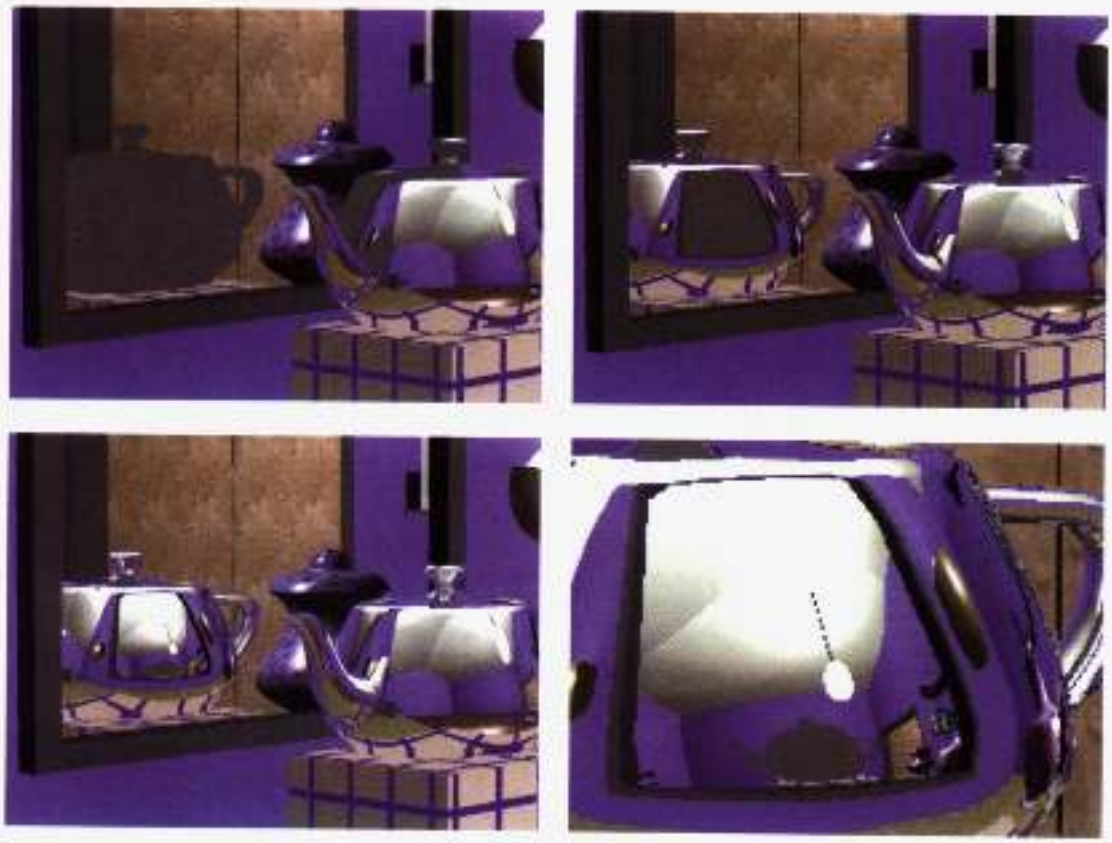


图18-11 一个递归深度的演示。跟踪分别在深度为2、3、4和5（放大的图像）处终止。"未就绪的"像素以灰色显示。作为递归深度的一个函数，走样是很明显的



a)



b)



c)

图18-12 这些图像显示了标准的光线跟踪方法和辐射度方法从它们各自所注重的全局相互作用的性质而言的同等的独特性。a是一个其主要光源被关闭的场景的光线跟踪图像，预设光线跟踪忽略除LDE和LDS*E之外的所有光线路径。b是前一个场景加上了"环境照明"，环境分量的值与绘制于光源打开的场景（见图18-10）时所用的值是相同的。并且，假设该场是当考虑由漫反射到漫反射的相互作用时可能会出现的照明的一种替代。c是用辐射度方法绘制的图像，用光线跟踪方法处理镜面物体，主光源被关闭（包括了光线跟踪的分量以使与下一个图比较）。由于辐射度方法考虑到了漫反射的内部反射，所以房间中的其余部分是可见的



反走样



(3D) 超采样

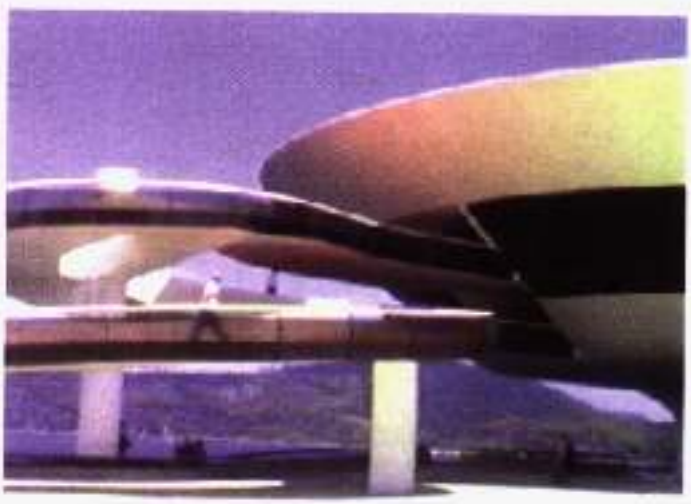


非均匀采样

图18-13 这些示意图表明相比于上下文敏感的反走样而言，独立于上下文的反走样的有效性。在各种反走样图像之间除了成本上有很大的差别之外其效果仅有很小的不同



图18-14 这个图显示了采用辐射度方法的场景



a)



b)

图18-15 a) 里约热内卢的现代艺术博物馆（由Oscar Niemeyer设计）的一张照片，取自明媚的阳光下。可以很清楚地看出颜色泄漏，并固定在照片中。你是否在现实中也有过类似的经历呢？b) 在一个辐射度的解中扩大了颜色泄漏



图18-16 这个图像是用一个“最小”定义计算的，是图18-3中表示的一个三角形划分的版本。它具有明显的阴影泄漏和光线泄露

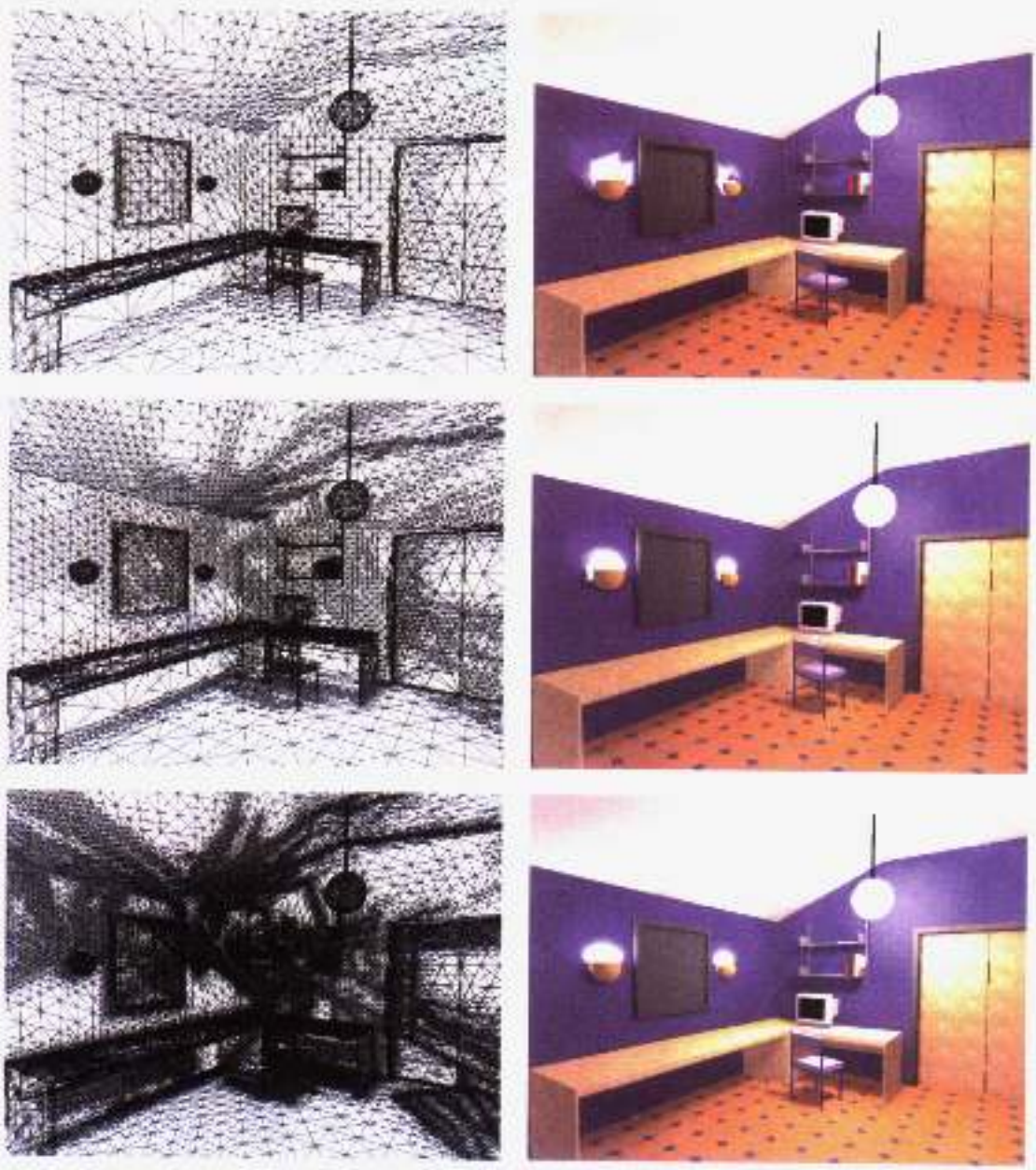


图18-17 这个图中的三对图像表明了在一个渐进式细化框架下的运算中细分策略的效果。增加细分程度和图像质量之间的关系是很明显的

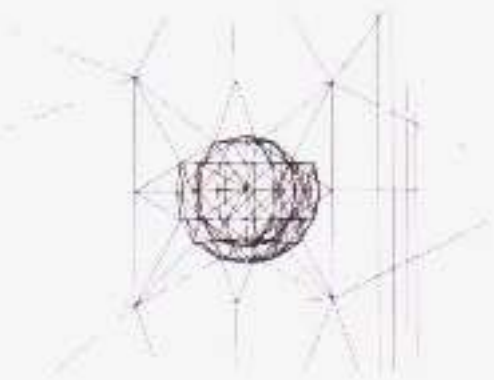


图18-18 在考虑了相互渗透的几何问题之后对壁灯周围的区域进行网格化的结果。这时，填塞的曲面片边界与光线的曲面片边界完全共线，这种网格化的结果完全消除了壁灯周围的泄漏





图18-19 用RADIANCE绘制程序绘制的场景

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭橥了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短、从业人员较少的现状下，美国等发达国家在其计算机科学发展的几十年间积淀的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及庋藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都步入一个新的阶段。为此，华章公司将加大引进教材的力度，在“华章教育”的总规划之下出版三个系列的计算机教材：除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”；同时，引进全美通行的教学辅导书“Schaum's Outlines”系列组成“全美经典学习指导系列”。为了保证这三套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这三套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业

的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证，但我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。教材的出版只是我们的后续服务的起点。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下:

电子邮件: hzjsj@hzbook.com
联系电话: (010) 68995264
联系地址: 北京市西城区百万庄南街1号
邮政编码: 100037

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周克定	周傲英	孟小峰	岳丽华	范　明
郑国梁	施伯乐	钟玉琢	唐世渭	袁崇义
高传善	梅　宏	程　旭	程时端	谢希仁
裘宗燕	戴　葵			

译 者 序

近年来，随着计算机的计算速度与精度不断提高、功能不断增强，硬件、软件进一步发展完善，计算机图形学中遇到的各种问题逐步得到解决。其结果是计算机显示的图形越来越生动和丰富多彩，主要表现在:

1. 由二维图形向三维图形转变

由于消隐和图形之间的布尔运算等方法，以及绘制三维图形的绘图软件的出现，可以由二维图形直接生成三维图形。

2. 由静态向动画转变

无论二维图形还是三维图形都只能反映事物静止的状态，人们更希望它能显示动态变化过程。动画软件的开发成功可以更形象地表现事物的变化过程，揭示其内在规律，使得原来必须要做实验才能解决的问题可以通过动画来完成，提高了效率。

3. 由线框图向真实感图转变

用线条绘制的三维形体真实感差。利用光栅扫描显示器以及各种算法（如填充、裁剪、消隐、光照等），三维物体除显示不同颜色外，还可以显示不同的材质、纹理、光照、视点等，从而使三维物体更接近真实。

以上的改变与3D计算机图形学的不断发展是密不可分的。

本书从3D计算机图形学的数学基础开始，详细论述了建立三维图形所需的各种方法，从建立线框图的多边形表示、Bézier曲线以及多边形网格，到绘制真实感图形采用的隐藏面消除、纹理、映射、阴影算法以及全局照明模型、辐射度理论等。此外，本书还介绍了高级辐射度方法、动画、预计算技术等。更可贵的是，书中对所论述的各种方法的应用范围及其局限性进行了详细的分析。对于那些对3D计算机图形学感兴趣的读者来说，本书是一本详尽的参考书。

由于译者水平的限制，难免在译文中出现错误，敬请广大读者批评指正。

包 宏

2005年 4月

前　言

这本书是第3版，讨论将物体的数学或几何描述转换成可视产品时所涉及的过程——数学或几何描述是一种计算机图形学模型，而可视产品是模拟真实物体外观的二维投影。经常采用一种合成摄像机的类比。假如我们牢记在计算机图形学中摄像机通常不适用的一些重要的限制（场深度和运动模糊是其中的两个例子），而且有些计算机图形学中的功能在摄像机中也不会出现（比如远、近裁剪平面），则合成摄像机是一个好的提法。

计算机图形学中的算法大多数都适用于三维空间，这一空间中的物体在整个建立图形过程的稍后阶段会被映射到一个二维显示器或称图像平面中。一般情况下，计算机图形学按下面的过程产生图像，即从一个非常详尽的几何描述开始，再对其进行一系列的变换以便反映三维空间中观察者和物体的位置关系，然后，通过执行一种称为绘制的过程使物体看起来有立体感并真实来模拟真实性。在20世纪80年代早期，研究工作有结合到一起的倾向（在20世纪70年代进行的反射模型、隐藏面消除等研究中实施），这种倾向导致了一种事实上将立体物体的图像合成的方法的出现。当前，对移动的计算机图像和虚拟现实的需求使以上的工作成果显得不够了。目前，关于如何对复杂物体进行建模使其性质和形状动态地变化，以及如何获取现实世界的丰富信息而不必对其进行非常详细的描述等方面，正在进行大量的研究工作。这些努力正在产生大量的合成方法和建模方法，但是目前还没有出现一种新的产生图像的技术可以与自20世纪70年代中期建立的伪标准建模和绘制实体物体的方法相媲美。

这一切是如何开始的呢？如我们所知，计算机图形学中的大多数进展都是以硬件的革命和可用的新设备的出现为动力的。很多软件迅速被开发出来以利用这些产生图像的硬件。从这个角度考虑，最重要的进展是所谓的光栅显示器，这是一种在PC的开发之后不久就大量充斥市场的设备。在这种设备中，完整的图像被存放于存储器中，该存储器称为帧存储器、屏幕缓存或刷新存储器。离散的计算机图像通过视频控制器连续不断地被转换成一组水平扫描线（光栅），然后再被馈入到TV监视器中。通过一个应用程序来产生图像，该应用程序通常访问一个物体或多个物体的模型或几何描述。图1中展示了这样一个系统中的几个主要组成部分。虚线右侧的显示硬件可以与处理器分离，但是近来在增强型PC或图形工作站中，它们通常是集成在一起的。光栅图形设备发展的重要性超过了所有其他硬件，它使得经明暗处理的三维物体的显示成为可能，这是一个非常重要的理论发展。三维物体与光源之间的相互作用可以被计算并且其效果可以被投影到二维空间并由显示设备显示出来。这种经明暗处理的图像是现代计算机图形学的基础。

两个使得经明暗处理的图像成为可能的早期里程碑式的成就是1971年由Gourand开发和1975年由Phong开发的算法，这些算法使得在对物体进行明暗处理时，对像素点亮度的计算变得容易且迅速。Phong技术目前仍是主流，并且无疑会对大多数计算机图形学中的明暗处理图像起作用。

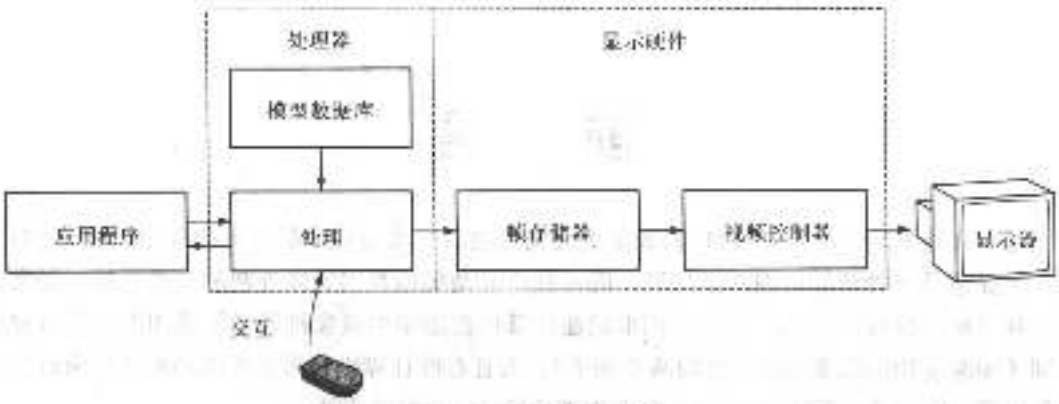


图1 图形系统的主要组成部分

明暗处理图像简史

当从专业人员的角度来看待计算机图形学时，可以看到自20世纪70年代中期以来，开发的动机一直是追求图像的真实性，即追求那种使物体或场景的图像与电视图像或照片相近的技术。这种技术应用的一个更新的趋势是在科学和工程（如医药）中显示信息。

图像真实性的基础是光与物体间交互的计算，这种计算可分成两个方面，即局部反射模型的建立及全局反射模型的建立。局部或直接反射模型只考虑物体与光源间的相互作用，就好像物体和光线在一个黑暗的空间漂浮着。也就是说，只考虑来自物体的第一次反射。全局反射模型考虑光线是如何从一个物体反射并传到另一个物体上的。换句话说，打到表面上一个点上的光或者来自一个光源（直接光），或者来自先打到了另一个物体上再反射出来的间接光。全局交互在大多数情况下还是一个未解的问题，当前已有两种部分解（即光线跟踪和辐射度）被广泛使用。

计算机图形学的研究一直朝着更科学的研究方向前进——早期取得了主要进展，并被合并到一种实用的技术中。后来的显著进展似乎更难于获得。可以说大多数图像使用Phong的局部反射模型（1975年首次报道）产生，很少使用光线跟踪模型（1980年第一次公开），用辐射度方法（1984年首次报道）的也很少，尽管在光-场景交互方法学方面还在进行着大量的研究。但计算机图形学领域的许多当前研究更关注应用，例如，关注动画、可视化和虚拟现实这样的一般应用。在最重要的计算机图形学刊物上（SIGGRAPH会议年度论文集），1985年总共有22篇文章讨论了图像的产生（绘制、建模和硬件），与之相比仅有13篇与应用有一些关系。10年之后在1995年，有37篇关于应用的文章，19篇关于图像生成技术的文章。

用局部交互建模表面反射

两个紧接着出现的技术进步是隐藏面消除算法和明暗处理图像技术的发展，它们模拟物体与光源的相互作用。大多数隐藏面消除研究是在20世纪70年代进行的，对于普通用途的使用，当前最通用的算法是Z缓冲器算法，这是一种非常容易实现并容易与明暗处理或绘制算法结合的算法。

在明暗处理图像技术中，主要技术支撑是Phong反射模型。这是一个灵巧的但完全经验性

的模型，它常常在运行结束时使物体反射的光比它接收到的光还要多。其参数是基于来自一个表面的光反射的最粗略的估计。尽管如此，这仍是在计算机图形学中最广泛采用的模型，它产生了大量的图像。为什么会如此呢？这是因为用户发现该模型是合适的并易于实现。

以理论为基础的反射模型试图更准确地建模反射，并且其参数具有物理意义，即这些参数对于一个真实的表面是可以测量的。例如，光在一个各向同性的表面（如塑料）上的反射与其在一个各向异性的表面（如抛光的铬）上的反射是不同的，这种影响可以通过明确地建模表面特性来模拟。这类模型试图在毫米级模拟光的行为（这时，粗糙程度和表面几何因素比光的波长要大许多）。这类模型的目的是模拟材料特性——不同的材料为何看起来有区别。反过来，一个模型的参数可以从一个真实的表面测得，并用于模拟。到目前为止，在绘制系统领域内实际采用的方法中，更完善的或理论性更高的局部反射模型似乎还没有得到广泛的采纳甚至实现。这可能是由于事实上用户并不认为值得为明暗处理物体的外表看起来稍微改进而付出更多的处理成本。

所有这些试图要精确建模来自表面的光的模型都是局部模型，也就是说，它们只考虑了光与物体的相互作用，就像物体漂浮在一个自由空间中，而没有考虑物体与物体之间的相互作用。因此，由此而产生的主要问题之一就是明暗处理没有被纳入到模型中（明暗处理是一种由于全局交互而产生的现象），因此必须用一个单独的算法进行计算。

Phong反射模型的建立加强了对添加明暗处理算法和纹理映射的研究，而这两项研究又增强了明暗处理物体的外观效果，并且减弱了基本Phong模型的那种看起来像漂浮在自由空间中的塑料的感觉。

建模全局交互

20世纪80年代出现了两个具有重大意义的全局模型，即试图评估物体间相互作用的光反射模型。全局交互产生的现象包括确定一个明暗处理区内的光强度、物体之间相互产生的反射（光学相互作用）以及被称为颜色泄漏的细微影响，这种颜色泄漏使得从一个漫反射表面发出的颜色渗入到另一个邻近的物体表面（漫反射相互作用）。明暗处理区域内的光强度只能通过全局交互计算确定。根据定义，明暗处理区域不能直接从光源得到光，而只能从其他物体反射的光中间接得到光。当我们在一个场景中看一个闪光的物体时，你可能希望从其上面看到其他物体的反射。像镀铬板这样非常光滑的表面的特性更像一面镜子，它记录下其周围所有物体表面的细节，并根据其自身的表面曲率对所有这一切进行几何变形。

成功的全局模型有光线跟踪和辐射度。然而，在其基本实现中，这两个模型都只是照顾到了全局照明的一个方面。光线跟踪方法实现了完美的光学反射——非常光滑的物体互相反射，而辐射度建模漫反射相互作用，即光从粗糙表面反射出来，照明其他物体。漫反射相互作用在室内人造材料中是普遍存在的，如将地毯铺到地板上或墙上的粗糙修饰。在房间中，不能看到光源的区域是由漫反射相互作用来照明的。由于这两种模型所模拟的现象互不相交，所以由两个模型产生的图像也具有标识性的特性。光线跟踪的图像非常适合于完美的递归反射以及锐角折射的情况，而辐射度模型的图像通常适用于柔和光照室内物体并且不包含闪光的物体。

计算机图形学并不是一种纯粹的科学，在计算机图形学中有关光与表面相互作用的许多研究工作都是采用现有的物理模型，然后再用一个计算机图形学算法加以模拟，这就可能会

对原数学模型引入许多简化，以便使其可以作为计算机图形学的算法来执行。光线跟踪和辐射度是这种趋势的典型例子。由于实际的原因，计算机图形学中进行了简化，在数学家看起来这可能过于粗糙，而这种过程仍然能成功的原因是当我们观察一个合成的场景时，我们并没有感觉到在数学上进行了简化，除非简化到出现了走样的程度。然而，大多数人可以很容易地将一幅计算机图像与一幅照片区分开。这样一来，计算机图像就有了一个属于其自己的“真实性”。该真实性是模型的函数，一个计算机图形与一幅真实场景的照片之间的相近程度因采用方法的不同而有很大差别。在计算机图形学中“照片真实性”是指图像看起来真实，而不是按像素对像素的方式逼近一幅照片。这种对计算机图形学产生的图像的主观判断稍稍降低了广泛采用的“照片真实性”的含义，但情况确实如此。在人类对于计算机图形学图像与等价于真实场景的图像（如电视图像）的感知进行比较方面仅有过很少量的研究。

目　录

出版者的话
专家指导委员会
译者序
前言
第1章　计算机图形学的数学基础……1
1.1　处理三维结构 ……1
1.1.1　计算机图形学中的三维仿射变换 ……1
1.1.2　改变坐标系的变换 ……6
1.2　结构变形变换 ……7
1.3　向量和计算机图形学 ……9
1.3.1　向量的加法 ……9
1.3.2　向量的长度 ……10
1.3.3　法向量和叉积 ……10
1.3.4　法向量和点积 ……11
1.3.5　与法向量反射相关的向量 ……12
1.4　光线和计算机图形学 ……13
1.4.1　光线几何——相交 ……14
1.4.2　相交——光线与球 ……14
1.4.3　相交——光线与凸多边形 ……15
1.4.4　相交——光线与包围盒 ……16
1.4.5　相交——光线与二次形 ……18
1.4.6 光线跟踪几何——反射和折射 ……18
1.5　图像平面中的插值性质 ……20
第2章　三维物体的表示和建模（1）……21
引言 ……21
2.1　三维物体的多边形表示 ……25
2.1.1　创建多边形物体 ……28
2.1.2　多边形物体的手工建模 ……29
2.1.3　多边形物体的自动产生 ……29
2.1.4　多边形物体的数学产生 ……30
2.1.5　程序化的多边形网格物体——分形物体 ……34
2.2　物体的构造实体几何表示 ……36
2.3　物体表示的空间细分技术 ……38
2.3.1　八叉树和多边形 ……40
2.3.2　BSP树 ……41
2.3.3　创建体素实体 ……42
2.4　用隐函数表示物体 ……43
2.5　场景管理和物体表示 ……44
2.6　总结 ……48
第3章　三维物体的表示和建模（2）……51
引言 ……51
3.1　Bézier曲线 ……53
3.1.1　连接Bézier曲线段 ……58
3.1.2　Bézier曲线性质总结 ……59
3.2　B样条表示 ……60
3.2.1　B样条曲线 ……60
3.2.2　均匀B样条 ……61
3.2.3　非均匀B样条 ……64
3.2.4　B样条曲线性质总结 ……70
3.3　有理曲线 ……70
3.3.1　有理Bézier曲线 ……70
3.3.2　NURBS ……72
3.4　从曲线到表面 ……73
3.4.1　连续性和Bézier曲面片 ……76
3.4.2　一个Bézier曲面片物体——Utah茶壶 ……78
3.5　B样条表面的曲面片 ……79
3.6　建立曲面片表面 ……82
3.6.1　截面或线性轴设计实例 ……84
3.6.2　控制多面体设计——基本技术 ……87
3.6.3　用表面拟合来创建曲面片物体 ……90
3.7　从曲面片到物体 ……94
第4章　表示和绘制 ……97
引言 ……97
4.1　绘制多边形网格——简单综述 ……97
4.2　绘制参数化表面 ……98
4.2.1　直接由曲面片描述进行绘制 ……99

4.2.2 曲面片向多边形转换 ……100
4.2.3 物体空间细分 ……101
4.2.4 图像空间细分 ……106
4.3 绘制构造实体几何表示 ……108
4.4 绘制体素表示 ……110
4.5 绘制隐函数 ……110
第5章 绘图流程（1）：几何操作……113
引言 ……113
5.1 绘图流程中的坐标空间 ……113
5.1.1 局部坐标系或建模坐标系 ……113
5.1.2 世界坐标系 ……114
5.1.3 摄像机/眼睛/观察坐标系 ……114
5.2 在观察空间中进行的操作 ……116
5.2.1 消隐或背面清除 ……116
5.2.2 视见体 ……116
5.2.3 三维屏幕空间 ……118
5.2.4 视见体和深度 ……120
5.3 先进的观察系统（PHIGS和GKS）……123
5.3.1 PHIGS观察系统概述 ……124
5.3.2 观察方向参数 ……125
5.3.3 观察映射参数 ……126
5.3.4 观察平面的更详细讨论 ……127
5.3.5 实现一个PHIGS型观察系统 ……128
第6章 绘图流程（2）：绘制或算法过程 …131
引言 ……131
6.1 在视见体上裁剪多边形 ……131
6.2 对像素明暗处理 ……134
6.2.1 局部反射模型 ……135
6.2.2 局部反射模型——实际问题 ……139
6.2.3 局部反射模型——关于光源的考虑…140
6.3 插值明暗处理技术 ……140
6.3.1 插值明暗处理技术——Gouraud明暗处理 ……140
6.3.2 插值明暗处理技术——Phong明暗处理 ……141
6.3.3 绘制程序的明暗处理选项 ……142
6.3.4 Gouraud明暗处理和Phong明暗处理的比较 ……143
6.4 光栅化 ……143
6.4.1 光栅化边 ……143
6.4.2 光栅化多边形 ……145
6.5 绘制的顺序 ……146
6.6 隐藏面消除 ……147
6.6.1 Z缓冲器算法 ……148
6.6.2 Z缓冲器和CSG表示 ……148
6.6.3 Z缓冲器与合成 ……149
6.6.4 Z缓冲器和绘制 ……150
6.6.5 扫描线Z缓冲器 ……151
6.6.6 跨跃式隐藏面消除 ……151
6.6.7 一个跨跃式扫描线算法 ……151
6.6.8 Z缓冲器和复杂场景 ……153
6.6.9 Z缓冲器总结 ……154
6.6.10 BSP树和隐藏面消除 ……155
6.7 多路绘制和累加缓冲器 ……157
第7章 模拟光线——物体相交：局部反射模型 ……161
引言 ……161
7.1 来自完全表面的反射 ……162
7.2 来自不完全表面的反射 ……163
7.3 双向反射分布函数 ……163
7.4 漫反射分量和镜面反射分量 ……165
7.5 完全漫反射——经验型散布镜面反射 …166
7.6 基于物理的镜面反射 ……166
7.6.1 建模表面的微观几何 ……167
7.6.2 阴影和屏蔽效果 ……167
7.6.3 观察几何学 ……169
7.6.4 Fresnel项 ……169
7.7 预计算BRDF ……171
7.8 基于物理的漫反射分量 ……172
第8章 映射技术 ……175
引言 ……175
8.1 二维纹理映射到多边形网格物体 ……179
8.1.1 用双线性插值进行反向映射 ……179
8.1.2 用中间表面进行反向映射 ……180
8.2 二维纹理域到双三次参数曲面片物体 ……182
8.3 广告牌 ……184
8.4 凹凸映射 ……184

8.4.1 用于凹凸映射的多路技术 ……186
8.4.2 用于凹凸映射的预计算技术 ……187
8.5 光线图 ……187
8.6 环境映射或反射映射 ……189
8.6.1 立方体映射 ……190
8.6.2 球映射 ……192
8.6.3 环境映射：比较点 ……193
8.6.4 表面性质和环境映射 ……193
8.7 三维纹理域技术 ……194
8.7.1 三维噪声 ……195
8.7.2 模拟扰动 ……196
8.7.3 三维纹理和动画 ……197
8.7.4 三维光线图 ……198
8.8 反走样和纹理映射 ……199
8.9 纹理映射中的交互式技术 ……201
第9章 几何阴影 ……205
引言 ……205
9.1 计算机图形学中阴影的性质 ……205
9.2 地平面上的简单阴影 ……207
9.3 阴影算法 ……207
9.3.1 阴影算法：投影多边形/扫描线 …207
9.3.2 阴影算法：阴影体 ……209
9.3.3 阴影算法：从光源变换导出阴影多边形 ……211
9.3.4 阴影算法：阴影Z缓冲器 ……211
第10章 全局照明 ……215
引言 ……215
10.1 全局照明模型 ……216
10.1.1 绘制方程 ……216
10.1.2 辐射光亮度、辐照度和辐射光亮度方程 ……217
10.1.3 路径的标注 ……219
10.2 全局照明算法的发展 ……221
10.3 已建立的算法——光线跟踪和辐射度 ……221
10.3.1 Whitted 光线跟踪方法 ……221
10.3.2 辐射度方法 ……223
10.4 全局照明中的蒙特卡罗技术 ……225
10.5 路径跟踪 ……228
10.6 分布式光线跟踪 ……229
10.7 二路光线跟踪 ……232
10.8 依赖于观察/独立于观察和多路方法 …234
10.9 存储照明 ……236
10.10 光线体 ……238
10.11 粒子跟踪和密度估计 ……238
第11章 辐射度方法 ……241
引言 ……241
11.1 辐射度理论 ……242
11.2 形状因子的确定 ……243
11.3 Gauss-Seidel方法 ……247
11.4 观察部分解——渐进式细化 ……248
11.5 辐射度方法存在的问题 ……250
11.6 辐射度图像中的人工痕迹 ……251
11.6.1 半立方体人工痕迹 ……251
11.6.2 重建人工痕迹 ……253
11.6.3 网格化人工痕迹 ……254
11.7 网格化策略 ……255
11.7.1 自适应或后网格化 ……256
11.7.2 预网格化 ……260
第12章 光线跟踪策略 ……271
引言——Whitted光线跟踪 ……271
12.1 基本算法 ……272
12.1.1 跟踪光线——初始的考虑 ……272
12.1.2 照明模型分量 ……272
12.1.3 阴影 ……273
12.1.4 隐藏面消除 ……274
12.2 用递归方法实现光线跟踪 ……275
12.3 七条光线的旅程——一个光线跟踪研究 ……277
12.4 光线跟踪多边形物体——多边形交点处的法向插值 ……279
12.5 光线跟踪方法的效率问题 ……280
12.5.1 自适应深度控制 ……280
12.5.2 第一次碰撞加速 ……281
12.5.3 具有简单形状的限定物体 ……281
12.5.4 二次数据结构 ……283
12.5.5 光线空间细分 ……287
12.6 利用光线的连贯性 ……288
12.7 一个历史话题——彩虹的光学问题 …291
第13章 体绘制 ……293
引言 ……293

13.1 体绘制和体数据的可视化 ……295
13.2 “半透明胶质”选项 ……298
13.2.1 体素分类 ……299
13.2.2 变换到观察方向 ……299
13.2.3 沿着光线合成像素 ……300
13.3 半透明胶质和表面 ……301
13.4 体绘制算法中关于结构的考虑 ……304
13.4.1 光线投射（未经变换的数据）……305
13.4.2 光线投射（变换后的数据）……306
13.4.3 体素投影方法 ……307
13.5 体绘制过程中的透视投影 ……309
13.6 三维纹理和体绘制 ……309
第14章 反走样理论及实践 ……311
引言 ……311
14.1 走样和采样 ……311
14.2 锯齿形边 ……315
14.3 计算机图形学中的采样与真实采样之间的比较 ……316
14.4 采样和重建 ……317
14.5 一个简单的比较 ……318
14.6 预过滤方法 ……319
14.7 超采样或后过滤 ……320
14.8 非均匀采样——一些理论概念 ……322
14.9 图像的傅里叶变换 ……326
第15章 颜色和计算机图形学 ……333
引言 ……333
15.1 计算机成像中的颜色集 ……334
15.2 颜色和三维空间 ……335
15.2.1 RGB空间 ……337
15.2.2 HSV单六面体模型 ……338
15.2.3 YIQ空间 ……340
15.3 颜色、信息和感知空间 ……340
15.3.1 CIE XYZ空间 ……341
15.3.2 CIE xyY空间 ……344
15.4 绘制和颜色空间 ……346
15.5 关于监视器的考虑 ……347
15.5.1 $RGB_{监视器}$和其他监视器的考虑 …347
15.5.2 关于监视器的考虑——不同的监视器和相同的颜色 ……347
15.5.3 关于监视器的考虑——颜色显示范围的映射 ……349
15.5.4 关于监视器的考虑——γ校正 ……350
第16章 基于图像的绘制和照片建模 ……353
引言 ……353
16.1 以前绘制的图像的复用——二维技术 …353
16.1.1 平面“冒名顶替者”或小画面 …354
16.1.2 计算平面小画面的有效性 ……355
16.2 改变绘制的资源 ……356
16.2.1 优先绘制 ……356
16.2.2 图像分层 ……357
16.3 运用深度信息 ……359
16.3.1 三维扭曲 ……359
16.3.2 分层深度图像 ……363
16.4 观察插值 ……365
16.5 四维技术——照明绘图或光线场绘制方法 ……368
16.6 照片建模和IBR ……370
16.6.1 利用照片全景图进行基于图像的绘制 ……373
16.6.2 合成全景图 ……373
16.6.3 基于图像绘制的照片建模 ……374
第17章 计算机动画 ……377
引言 ……377
17.1 计算机动画技术的分类和描述 ……379
17.2 刚体动画 ……380
17.2.1 插值或关键帧 ……380
17.2.2 明确的脚本 ……382
17.2.3 旋转的插值 ……384
17.2.4 用四元方法表示旋转 ……386
17.2.5 对四元式插值 ……388
17.2.6 作为动画物体的摄像机 ……392
17.3 连接结构和层次化运动 ……392
17.4 计算机动画中的动力学 ……400
17.4.1 刚体的基本理论——粒子 ……401
17.4.2 力的性质 ……402
17.4.3 刚体——有翼展的物质 ……403
17.4.4 在计算机动画中运用动力学 ……405
17.4.5 模拟成块物质的动力学 ……406

17.4.6 空间-时间限制 ……409
17.5 碰撞检测 ……411
17.5.1 非限制性阶段/限制性阶段算法 …411
17.5.2 用OBB进行非限制性阶段的碰撞检测 ……412
17.5.3 限制性阶段：凸多面体对——准确的碰撞检测 ……414
17.5.4 单阶段算法——物体的层次结构 …415
17.6 碰撞响应 ……417
17.7 粒子动画 ……419
17.8 行为动画 ……421
17.9 总结 ……423
第18章 比较图像研究 ……425
引言 ……425
18.1 局部反射模型 ……425
18.2 纹理映射和阴影映射 ……426
18.3 Whitted 光线跟踪 ……427
18.4 辐射度方法 ……428
18.5 RADIANCE ……429
18.6 总结 ……430
参考文献 ……431
索引 ……439

第1章　计算机图形学的数学基础

1.1　处理三维结构
1.2　结构变形变换
1.3　向量和计算机图形学
1.4　光线和计算机图形学
1.5　图像平面中的插值性质

1.1　处理三维结构

在产生三维场景时变换是重要的工具。变换用于在环境中移动物体，还用于构建用来显示表面环境的二维视图。本章讨论基本的三维变换，并介绍一些有用的变形变换和基本的三维几何，这些内容都将在后面章节中用到。

在计算机图形学中表示一个物体的最普通的方法是多边形网格模型。在第2章中将对这种表示方法进行完整地描述。我们将一个物体的表面表述为一组相关联的平面多边形，每一个多边形为一个（相连的）点的列表。这种表示形式根据物体的性质不同既可以是准确的，又可以是近似的。例如，一个立方体可以用六个平面来准确地表示，而一个圆柱则只能用多边形近似地表示，比如用六个矩形表示曲面以及用两个六边形来近似端面。近似图形中所用多边形的个数决定了所表示物体的准确程度，而这个数量对于建模的成本、存储空间和绘制成本及质量都有影响，由于其内在的简单性和建立这种模型所用的经济的明暗处理算法的发展无疑使多边形网格建模技术得到了普及。

多边形网格模型由一个顶点结构组成，每一个顶点是一个在所谓世界坐标空间中的三维的点。稍后，我们将讨论这些顶点是如何连接起来以形成多边形的，以及这些多边形如何连接构成完整的物体的。但是作为开端，我们将只把物体作为一组三维顶点集来考虑，并考察如何用线性变换在三维空间中对其进行变换。

1.1.1　计算机图形学中的三维仿射变换

在这一节中，我们考察三维仿射变换。这些变换影响到旋转、缩放、剪切和平移变换，仿射变换可以用一个矩阵来表示。可以将一组仿射变换结合成一个总的仿射变换。从技术上，我们说一个仿射变换由线性变换（旋转、缩放、剪切）及相跟的平移变换的任意组合构成。

在世界坐标系中定义物体，这个坐标系通常是右手系。右手系和左手系的三维坐标系统如图1-1所示。右手系是标准的数学规范，而左手系则已经和正在被用于计算机图形学中的特定的视图系统中。这两种系统的区别在于z轴的不同，如图1-1中所示。根据所用系统的不同，当你将手指从x轴的正向到y轴的正向绕z轴旋转时，你的拇指将给出不同的z方向。



a) 右手坐标系　　b) 左手坐标系

图 1-1

在物体自身所处的局部坐标系中定义物体有时是很方便的，当一个三维物体被建模后，以物体中的某些参考点建立顶点是有用的。事实上，一个复杂的物体可能有多个局部坐标系。每一个局部坐标系用于其某个部位。这样，同一物体可能会在一个场景中出现多次，而用一个局部的原点进行定义是唯一合理的解决方法。于是，通过用一组平移、旋转和缩放变换的组合来描述物体，就可以看做是对每一个物体的局部坐标系向世界坐标系的变换。最后，当对物体进行旋转时，如果相对于一个局部参考点（如一个对称轴）来定义旋转会更容易些。

属于一个物体的顶点集或三维点集可以通过一个线性变换来变换到另一个点集。这两个点集处于同一个坐标系，计算机图形学中的矩阵表示法用来描述变换，在计算机图形学中的常规做法是将点或向量用列矩阵表示，并放在变换矩阵T的后面。

使用矩阵表示法，一个点V经平移、缩放和旋转变换为:

$$V' = V + D$$
$$V' = SV$$
$$V' = RV$$

其中，D是平移向量，S和R分别为缩放矩阵和旋转矩阵。

这三个操作在计算机图形学中是最常使用的变换。在动画中，一个刚体只能被平移和旋转，而缩放用于物体建模。为了将上述变换用同样的方式进行处理和组合，我们采用一种称为齐次坐标的坐标系，这个坐标系增加了空间的维数。在计算机图形学中，使用这种坐标系的原因是能够把平移按矩阵乘（而不是加）来进行操作，这样对于线性变换就有一个统一的形式。在齐次坐标系中，顶点

$$V(x,y,z)$$

被表示为

$$V(w\cdot X, w\cdot Y, w\cdot Z, w)$$

对于任何缩放因子$w \neq 0$，三维笛卡儿坐标表示为

$$x = X/w$$
$$y = Y/w$$
$$z = Z/w$$

如果设w为1，则一个点的矩阵表示为

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

3

这时平移就可以像其他两种变换那样按矩阵乘来进行:

$$\boldsymbol{V}' = \boldsymbol{TV}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

这种定义暗示，可通过对每一个定义物体的顶点应用一个位移T_x, T_y, T_z来使物体在三个坐标上得到平移。把变换写成三个方程的方程组是一种表示矩阵的方便且精巧的方法:

$$x' = x + T_x$$
$$y' = y + T_y$$
$$z' = z + T_z$$

加入缩放和旋转，变换才是完整的。首先是缩放:

$$\boldsymbol{V}' = \boldsymbol{SV}$$

$$S = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

此处，S_x, S_y, S_z为缩放因子。对于均一缩放，$S_x = S_y = S_z$。否则，对于非均一缩放，按不同缩放因子缩放。同样，通过三个方程应用于物体中的每个顶点来表示这个过程:

$$x' = x \cdot S_x$$
$$y' = y \cdot S_y$$
$$z' = z \cdot S_z$$

要在三维空间中旋转物体，需要定义一个旋转轴，这个轴在三维空间中可以有任意的方向。但是最容易的方法是考虑平行于一个坐标轴方向的旋转。关于x, y, z轴的逆时针方向旋转的变换矩阵（沿着轴朝向原点）分别为:

$$\boldsymbol{R}_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4

$$\boldsymbol{R}_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{R}_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

z轴的矩阵定义与下列三个方程的方程组等价:

$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$
$$z' = z$$

图1-2为这些变换的例子。

经常需要用到这些变换的逆变换，$\boldsymbol{T}^{-1}$通过对T_x, T_y, T_z求反来得到。用S_x, S_y, S_z的逆来代替S_x, S_y, S_z得到$\boldsymbol{S}^{-1}$，而对旋转的角度求反得到$\boldsymbol{R}^{-1}$。

任意旋转、缩放、平移的集合都可以叠加并连接到一起，得到一个净变换矩阵。例如，如果

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \boldsymbol{M}_1 \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

并且

$$\begin{bmatrix} x'' \\ y'' \\ z'' \\ 1 \end{bmatrix} = \boldsymbol{M}_2 \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}$$

则变换矩阵可以被连接起来:

$$\boldsymbol{M}_3 = \boldsymbol{M}_2\boldsymbol{M}_1$$

及

$$\begin{bmatrix} x'' \\ y'' \\ z'' \\ 1 \end{bmatrix} = \boldsymbol{M}_3 \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

注意顺序: 在乘积$\boldsymbol{M}_2\boldsymbol{M}_1$中，所应用的第一个变换是$\boldsymbol{M}_1$，尽管平移是可交换的，但旋转不可交换. 并且

$$\boldsymbol{R}_1\boldsymbol{R}_2 \neq \boldsymbol{R}_2\boldsymbol{R}_1$$

5 这种情况在图1-2e和图1-2f中给出。

通用的变换矩阵有下面的形式:

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & T_x \\ A_{21} & A_{22} & A_{23} & T_y \\ A_{31} & A_{32} & A_{33} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

其中，左上角的3 × 3子矩阵A代表净的旋转和缩放而T给出净平移。

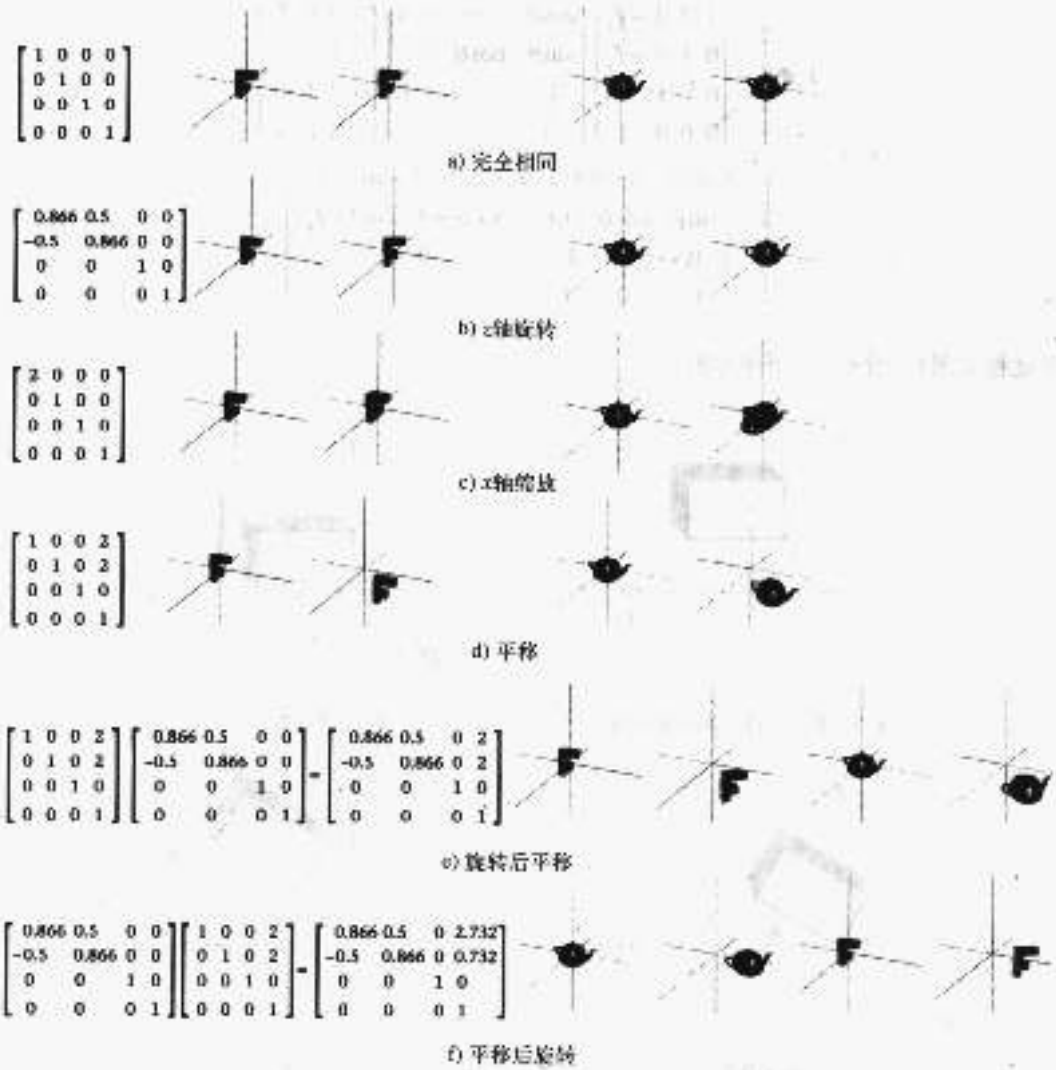


图1-2 线性变换的例子

将变换连接起来形成一个净变换矩阵的能力是有用的，因为这样可以对任意的线性变换给出单个矩阵的定义。例如，考虑绕一条平行于z轴的线旋转一个物体，而这条线通过点（T_x, T_y, 0），也通过物体的一个顶点。此处，我们意指物体不处于原点，希望进行绕物体本身的一个参考点进行旋转。换句话说，我们希望对一个物体进行相对于其自身坐标系（称为局部坐标系）进行旋转（又见1.1.2节）。此时我们不能简单地应用旋转矩阵，因为这种旋转矩阵是相应于原点定义的。而对于不在原点的物体，需先进行旋转再进行平移，这并不是通常的情况。取而代之的方法是我们必须导出一个如下的净变换矩阵：

1) 把物体平移到原点。

2) 进行希望的旋转。

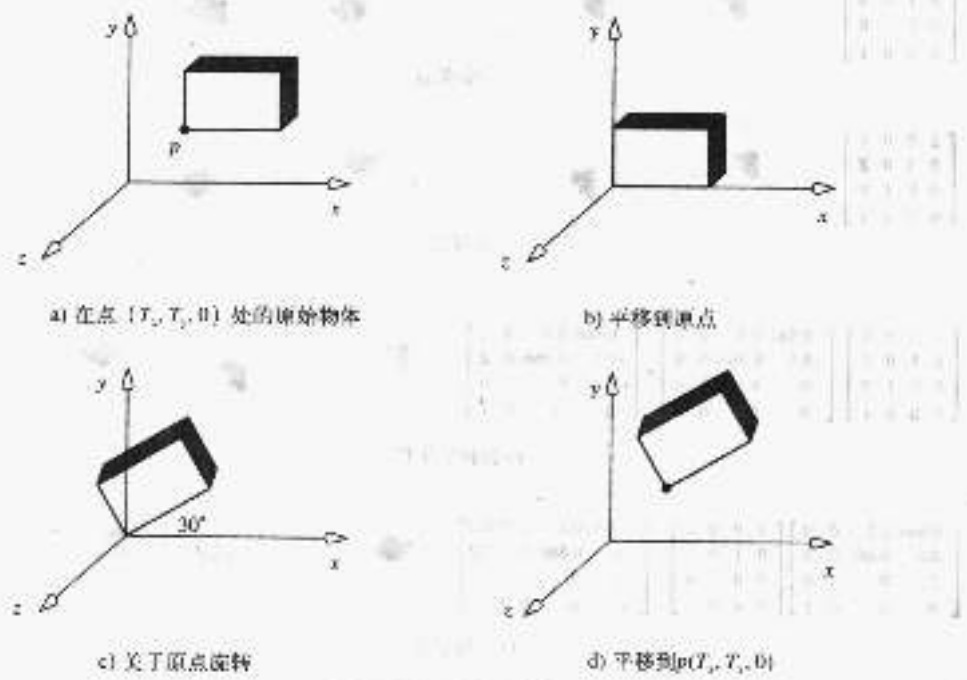
3) 将物体平移回其原来的位置。

净变换矩阵为：

$$\boldsymbol{T}_2\boldsymbol{R}\boldsymbol{T}_1 = \begin{bmatrix} 1 & 0 & 0 & -T_x \\ 0 & 1 & 0 & -T_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & 0 & (-T_x\cos\theta + T_y\sin\theta + T_x) \\ \sin\theta & \cos\theta & 0 & (-T_x\sin\theta - T_y\cos\theta + T_y) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

这个过程如图1-3所示，其中θ为30°。



a) 在点$(T_x, T_y, 0)$处的原始物体　b) 平移到原点

c) 关于顶点旋转　d) 平移到$p(T_x, T_y, 0)$

图1-3　一个物体关于其自身顶点旋转中的两个阶段。绕在点$(T_x, T_y, 0)$处平行于z轴的轴旋转。为清楚起见显示了一个二维投影（z轴指向纸外）

6
7

1.1.2 改变坐标系的变换

到目前为止，我们已经讨论了对于点的变换，所有这些点都是相对于一个特定的坐标系来表示的，这个坐标系称为世界坐标系。而在计算机图形学中的许多场合，需要导出从一个坐标系到另一个坐标系的变换。最普遍的情况是，我们有一些物体，而每一个物体都用物体自身的坐标系中的一组顶点来定义。这种坐标系称为局部坐标系，每一个物体都有一个适宜的局部坐标系。例如，一个由基本圆柱体组成的复杂物体有一个与圆柱体的长轴相重合的坐标轴。如果希望把一些这样的物体放在一起并将其放在一个场景中，则场景将采用世界坐标

系，我们将对这些物体进行平移、旋转、缩放变换，以将其放置于场景中。这样，我们可以考虑关于物体的变换操作，或等价地，考虑物体的局部坐标系上的变换操作。将一个带有局部坐标系的物体放到世界坐标系中一个位置的变换称为模型变换。

另一个涉及坐标系的改变的重要情况是从世界坐标系向观察坐标系的变换——一种观察变换。这里，引入了一种新的坐标系，如果希望的话，对一个相对于世界坐标系的物体，我们必须将世界坐标系中的顶点转换到这个新的坐标系中。

考虑两个轴平行的坐标系，即坐标系只是平移了。如果我们希望将坐标系1中表示的点变换到坐标系2中，可以用变换的逆。将坐标系1中的原点引入坐标系2中，即坐标系1中的一个点 $(x, y, z, 1)$ 用下式变换到点 $(x', y', z', 1)$:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -T_x \\ 0 & 1 & 0 & -T_y \\ 0 & 0 & 1 & -T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$= \boldsymbol{T}_{12} = (\boldsymbol{T}_{21})^{-1}$$

这是将坐标系1中的原点平移到坐标系2中的点的变换（这时此点依然相对于坐标系1表示）。另一种叙述的方法是，通常需要的变换是在当前坐标系中将旧轴引入到新轴中的变换的逆。

这是一个重要的结果，因为我们通常通过对原点和坐标轴的变换来实现坐标系之间的变换。在观察坐标系中，坐标系中的一个改变既包括旋转也包括平移，我们通过将旋转和平移进行某些结合来求出所需的变换。

8

1.2 结构变形变换

以上的线性变换或移动物体（旋转和平移）或是缩放物体。均一缩放保持原形。对S_x, S_y, S_z用不同的值，则物体在特定的坐标轴方向被伸长或缩短。在这一节中，我们介绍一组对物体变形的变换，这些变换在Barr（1984）的文章中有完整的描述，在那里这些变换被称为全局变形。在这篇文章中详细描述的特殊变形有变细、扭曲、弯曲。

Barr用一个公式来定义变换:

$$X = F_x(x)$$
$$Y = F_y(y)$$
$$Z = F_z(z)$$

其中，(x, y, z) 为在一个未变形的实体中的顶点，(X, Y, Z) 是变形后的顶点。采用这种表示法，上面提到的缩放变换为:

$$X = S_x(x)$$
$$Y = S_y(y)$$
$$Z = S_z(z)$$

可以很容易地通过缩放进行变细。选择一个变细的轴，通过沿着这个轴建立一个变细函数使其他两个分量具有不同的缩放。于是沿着z轴使物体变细，则

$$X = rx$$
$$Y = ry$$
$$Z = z$$

其中，$r = f(z)$是一个线性的或非线性的变细轮廓（profile）或函数，于是变换就成为一个r的函数。即根据它所应用的空间而改变变换，实际上是缩放一个缩放变换。

全局轴扭曲变换可以按一种微分旋转来实现，正如变细是一种微分缩放一样。要沿z轴扭曲一个物体，可以用:

$$X = x\cos\theta - y\sin\theta$$
$$Y = x\sin\theta + y\cos\theta$$
$$Z = z$$

其中，$\theta = f'(z)$，$f'(z)$定义了沿z轴每单位长度上的扭曲率。

沿着一个轴进行全局线性弯曲是结合两个区域的一种合成变换，这两个区域包括一个弯曲区域和一个弯曲区域之外的区域，其中变形为旋转变换和平移变换。

9

Barr将一个沿y轴的弯曲区域定义为:

$$y_{\min} \leqslant y \leqslant y_{\max}$$

弯曲的曲率半径为$1/k$，弯曲的中心位于$y = y_0$，弯曲的角度为:

$$\theta = k\,(y' - y_0)$$

其中

$$y' = \begin{cases} y_{\min} & y \leqslant y_{\min} \\ y & y_{\min} < y < y_{\max} \\ y_{\max} & y \geqslant y_{\min} \end{cases}$$

变形变换由下式给出:

$$X = x$$

$$Y = \begin{cases} -\sin\theta\left(z - \dfrac{1}{k}\right) + y_0 & y_{\min} \leqslant y \leqslant y_{\max} \\ -\sin\theta\left(z - \dfrac{1}{k}\right) + y_0 + \cos\theta(y - y_{min}) & y < y_{\min} \\ -\sin\theta\left(z - \dfrac{1}{k}\right) + y_0 + \cos\theta(y - y_{max}) & y > y_{\max} \end{cases}$$

$$Z = \begin{cases} -\cos\theta\left(z - \dfrac{1}{k}\right) + y_0 & y_{\min} \leqslant y \leqslant y_{\max} \\ -\cos\theta\left(z - \dfrac{1}{k}\right) + \dfrac{1}{k} + \sin\theta(y - y_{min}) & y < y_{\min} \\ -\cos\theta\left(z - \dfrac{1}{k}\right) + \dfrac{1}{k} + \sin\theta(y - y_{max}) & y > y_{\max} \end{cases}$$

图1-4显示了各种变换的一个例子。对立方体的变形是效果的直观显示，而对Utah茶壶也做了相同的变形。图1-5（彩色插图）展示了一个经扭曲和变细后的多边形网格物体（一个波纹状圆柱体）的渲染图。

一般来讲，不能将非线性、非限定性的变形应用到多边形网格中。所遇到的一个问题是顶点之间的连接限定条件。例如，我们不能不对其进行限定就扭曲一个以六个表面表示的立方体，并保持一个适合于渲染的结构。另一个问题是，对于那些使顶点分离的变形，会产生

降低原始模型的多边形分辨率的效果，这使得轮廓边缘走样（在第4章中详细介绍）。因此，物体模型的多边形性质限制了变形的性质，这个问题可以通过将原网格细分以作为变形的刚度函数的方法来克服。

10



图1-4 结构变形变换

1.3 向量和计算机图形学

在计算机图形学中的许多地方都用到向量，向量是一种实体，它具有大小和方向。向量的一个最普通的例子是粒子在空间运动的速度。速度既有大小也有方向，这与标量是有区别的，标量只有大小。标量的一个例子是空间中一点的温度。三维的向量由三元组表示：

$$V = (v_1, v_2, v_3)$$

其中每一个分量v_i均为一个标量。

11

1.3.1 向量的加法

向量V和W的加法定义为：

$$\begin{aligned} X &= V + W \\ &= (x_1, x_2, x_3) \\ &= (v_1 + w_1, v_2 + w_2, v_3 + w_3) \end{aligned}$$

从几何上来看，这可以解释如下。W的“尾部”被放置在V的“头部”，X是将V的尾部与W的头部连起来形成的向量，图1-6表示一对二维向量的加法及其解释。

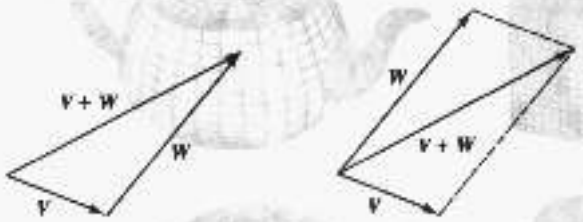


图1-6 两个向量求和的两种几何表示

1.3.2 向量的长度

向量的大小或长度定义为：

$$|V| = (v_1^2 + v_2^2 + v_3^2)^{1/2}$$

从几何上将其解释为从尾部到头部的距离。

我们规格化向量来产生单位向量（即长度等于1的向量）V的规格化向量为：

$$U = \frac{V}{|V|}$$

这是一个单位长度的向量，与U同向。现在可以把U作为参考方向。注意，我们可以写出：

$$V = |V|U$$

该式说明，任意向量都由其长度与其方向的乘积得出，在计算机图形学中经常用到规格化。这是因为我们所感兴趣的是计算并表示物体的方向，而对方向的比较需要规格化的向量。

1.3.3 法向量和叉积

在计算机图形学中，相当大量的处理计算都是用表面法向量来进行的。例如，在多边形网格模型中（见第2章），在将表面与光线的方向进行比较时，用法向量来代表表面的方向。在反射模型中，用这样的比较来计算从表面反射出来的光的强度。光线方向向量与表面法向量之间的夹角越小，则从表面反射出来的反射光光强越大（见第7章）。



图1-7 计算多边形的法向量

一个多边形的法向量是通过三个（非共线的）多边形顶点计算出来的。三个顶点定义了两个向量V_1和V_2（见图1-7），通过取这两个向量的叉积求出此多边形的法向量。

$$N_p = V_1 \times V_2$$

向量V和W的叉积为向量X，定义为：

$$\begin{aligned} X &= V \times W \\ &= (v_2w_3 - v_3w_2)\vec{i} + (v_3w_1 - v_1w_3)\vec{j} + (v_1w_2 - v_2w_1)\vec{k} \end{aligned}$$

其中，i、j和k为标准单位向量：

$$i=(1,0,0)$$
$$j=(0,1,0)$$
$$k=(0,0,1)$$

也就是说，向量的方向是沿着坐标系轴的方向，该坐标系定义向量所在的空间。

从几何意义上来讲，正如我们所暗示的，叉积是向量，其方向为包含形成叉积的两个向量的平面的法向。当要确定多边形的表面法向时，叉积必须指向相应的物体。在右手坐标系中，由右手规则来给出叉积向量的方向。如果你的右手的前两个手指指向V和W的方向，则X的方向由你的拇指给出。

如果表面是一个双三次参数表面（见第3章），则其法向量的方向随表面连续地变化。通过叉积，可以计算表面上的任意一点（u, v）处的法向量。通过首先计算两个参数方向上的切向量可以达到这一目的（在这里出于完整性考虑我们给出这一过程的概述，在第3章中将给出完整的描述）。对于一个定义为$Q(u,v)$的表面，有：

$$\frac{\partial}{\partial u}Q(u,v)\text{和}\frac{\partial}{\partial v}Q(u,v)$$

然后我们定义：

$$N_s=\frac{\partial Q}{\partial u}\times\frac{\partial Q}{\partial v}$$

这一过程的略图见图1-8。

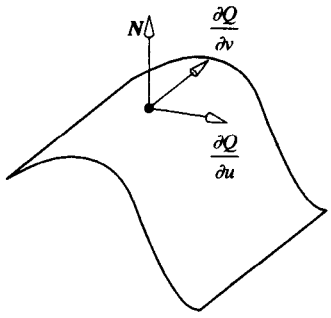


图1-8 参数表面$Q(u,v)$上一个点的法向N

13

1.3.4 法向量和点积

在计算机图形学中，点积最通常的用法是提供对于两个向量之间的夹角的测量。两个向量之一是一个或一组表面的法向量。点积常用于明暗处理计算（即光线的方向向量和表面法向之间的夹角）以及可见性测试（即观察向量和表面法向之间的夹角）。

向量V和W的点积是一个标量X，定义为：

$$X=V\cdot W$$
$$=v_1w_1+v_2w_2+v_3w_3$$

图1-9a为两个向量，运用余弦规则，我们得到：

$$|V-W|^2=|V|^2+|W|^2-2|V||W|\cos\theta$$

其中，θ是向量之间的夹角。从图中还可以看出：

$$|V-W|^2=|V|^2-2V\cdot W+|W|^2$$

于是：

$$V\cdot W=|V||W|\cos\theta$$

得出：

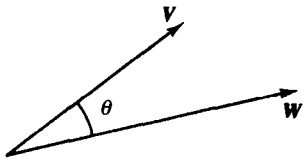
$$\cos\theta=\frac{V\cdot W}{|V||W|}$$

14 或者，两个向量之间的夹角是其规格化向量的点积。

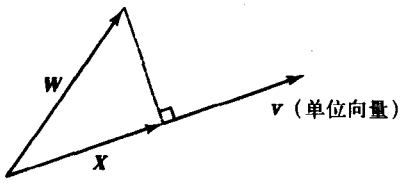
我们可以用点积使一个向量投影到另一个向量上。考虑一个单位向量V，如果我们把任意向量W投影到V上（见图1-9b），并把这个投影称为X，则有：

$$\begin{aligned} |X| &= |W|\cos\theta \\ &= |W|\frac{V\cdot W}{|V||W|} \\ &= V\cdot W \end{aligned} \tag{1-1}$$

因为V是一个单位向量，于是，V和W的点积即为W投影到V上的长度。



a) 两个向量的点积与它们之间的夹角的余弦有关：$\cos\theta = \dfrac{V\cdot W}{|V||W|}$



b) $|X| = V \cdot W$是W投影到V上的长度

图 1-9

计算机图形学中所利用的点积的性质是它的符号。由于它与$\cos\theta$之间的关系，V和W（其中V和W为任意长度）的点积的符号为：

$$\begin{aligned} V\cdot W &> 0，\text{若 } \theta < 90° \\ V\cdot W &= 0，\text{若 } \theta = 90° \\ V\cdot W &< 0，\text{若 } \theta > 90° \end{aligned}$$

1.3.5 与法向量反射相关的向量

有三个与表面法向相关的重要向量，它们是光线方向向量L、反射向量或镜向向量R以及观察向量V。光线方向向量L的方向由从表面法向的尾部到光源之间的连线的方向给出；这个方向在简单明暗处理算法中定义为当前表面上的一个点。这个向量如图1-10a所示。反射向量R的方向由沿着方向L射入进来再从表面反射出去的光的方向确定，有时这个方向称为镜面方
15 向。几何光学告诉我们，反射角等于入射角，如图1-10b所示。

考虑图1-11中的构成，有：

$$\begin{aligned} R &= R_1 + R_2 \\ R_1 &= -L + R_2 \end{aligned}$$

于是：

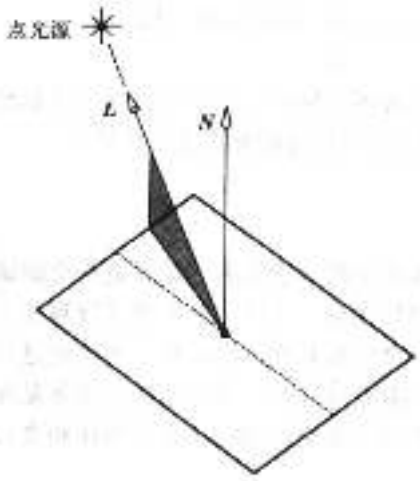
$$R = 2R_2 - L$$

从方程（1-1）：

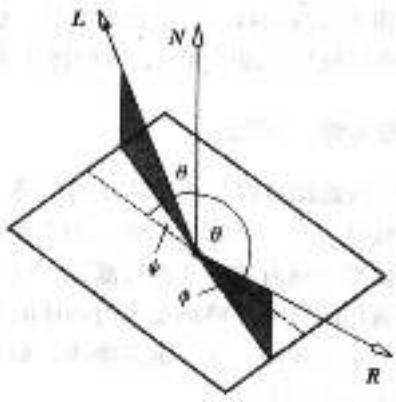
$$R_2 = (N\cdot L)N$$

和

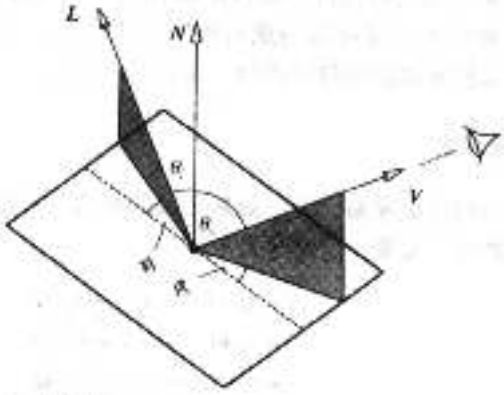
$$R = 2(N\cdot L)N - L \tag{1-2}$$



a) L，光线方向向量



b) R，反射向量



c) V，观察向量，它是一个任意方向的向量

图1-10 与法向量相关的向量

图1-10c为一个观察向量V，注意到该向量具有任意的方向，一般情况下，我们只对在方向L上的入射光分量感兴趣，而L方向是沿着V反射的方向。一般这个值依赖于ϕ_i和θ_i。我们还注意到，反射光的光强依赖于入射角ϕ_i和θ_i，这通常被描述为双向依赖性，因为涉及到三维空间中的两个角（ϕ_r, θ_r）和（ϕ_i, θ_i）。

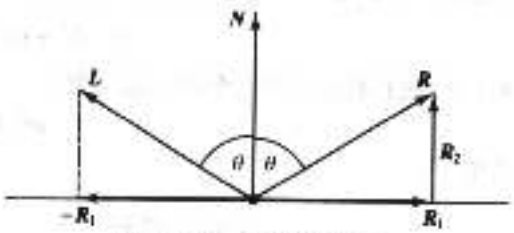


图1-11 反射向量R的构成

1.4 光线和计算机图形学

在计算机图形学中我们关注一种称为光线（ray）的实体，数学上称为有向线段，它具有位置，大小和方向。大多数情况下我们用其把光描述为一种无限细的束——一束光线。如果我们想象一束光线是三维空间中的一条物理的线，则其位置是这条线的尾部的位置，其长度是这条线头尾之间的长度，其方向是这条线的方向。一条光线可以由两点或一点加一个向量

来定义。如果光线的端点分别为 (x_1, y_1, z_1) 和 (x_2, y_2, z_2)，则向量由下式给出:

$$V = (x_2 - x_1, y_2 - y_1, z_2 - z_1)$$

光线不仅用于光线跟踪，而且在体绘制、绘制构造实体几何体以及在辐射度中计算结构因子时也可以找到使用光线的例子。我们接下来将看到与光线相关的更重要的计算。

1.4.1 光线几何——相交

因为光线跟踪模拟光通过环境的路线，与光线相关的最常见的计算就是相交测试，即观察一条光线是否与一个物体相交，如果是的话在何处相交。这里，我们为了计算相交而将一条光线与场景中所有的物体进行测试。这可能是一个代价非常高的计算，为了使这种计算更有效，最通用的技术是将场景中的物体限制在一个限定体之内，最合适的限定体是球。首先测试光线与球的相交，球包围着物体，如果光线与球不相交则它也不会与物体相交。另一种常见的限定体是包围盒。

16
~
17

球和包围盒也被用于计算机动画中的碰撞检测试验（见第17章）。物体对之间只有在限定体相交时才可能相交。此处的动机与光线跟踪相同，在进行逐个的多边形级别的详细的相交测试之前，首先将那些不可能相碰的成对物体挑选出来。检查球与球之间的相交是繁琐的，而对于包围盒，如果包围盒是轴向对齐的，则只需限定在x, y, z方向上进行检查。

1.4.2 相交——光线与球

光线和球之间的相交很容易计算。如果光线的两端分别是 (x_1, y_1, z_1) 和 (x_2, y_2, z_2)，则第一步是将光线参数化（见图1-12):

$$\begin{aligned} x &= x_1 + (x_2 - x_1)t = x_1 + it \\ y &= y_1 + (y_2 - y_1)t = y_1 + jt \\ z &= z_1 + (z_2 - z_1)t = z_1 + kt \end{aligned} \tag{1-3}$$

其中

$$0 \leqslant t \leqslant 1$$

中心在 (l, m, n) 半径为r的球由下式给出:

$$(x-l)^2 + (y-m)^2 + (z-n)^2 = r^2$$

对x, y, z进行代换得到一个t的二次方程:

$$at^2 + bt + c = 0$$

其中

$$\begin{aligned} a &= i^2 + j^2 + k^2 \\ b &= 2i(x_1 - l) + 2j(y_1 - m) + 2k(z_1 - n) \\ c &= l^2 + m^2 + n^2 + x_1^2 + y_1^2 + z_1^2 + 2(-lx_1 - my_1 - nz_1) - r^2 \end{aligned}$$

如果这个二次方程的判定值小于零则光线不与球相交。如果判定值等于零则光线擦过或与球相切，二次方程的两个实数根给出相交的前后位置。用t值代入原始方程中可以得到这些点。图1-12表明t的值还给出相对于 (x_1, y_1, z_1) 和 (x_2, y_2, z_2) 的相交点的位置。只有t的正值是有关的，而t的最小值对应于最靠近光线的起始位置的相交。

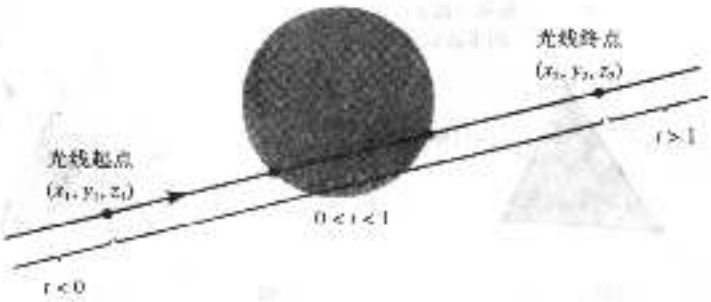


图1-12 在一条光线上参数t的值

另一个通过相交获得的信息是表面法向量（以便计算反射光线和折射光线），尽管若使用球作为限定体，则只需得到是否出现了相交的事实。

如果相交点是 (x_i, y_i, z_i)，球的中心为 (l, m, n)，则相交点处的法向量为：

$$N = \left(\frac{x_i - l}{r}, \frac{y_i - m}{r}, \frac{z_i - n}{r} \right)$$

1.4.3 相交——光线与凸多边形

如果物体由一组多边形表示并且是凸多边形，则直接的方法是逐个地对每一个多边形与光线的关系进行测试。按如下步骤进行：

1) 得到一个包含多边形的平面方程。

2) 检查此平面与光线的相交。

3) 检查此相交是否包含在多边形中。

这个操作的一个更通用的应用是在视见体上裁剪一个多边形（见第5章）。这时“光线”是多边形的一条边，我们需要求出多边形的边和视见体平面的交，以便分离这个多边形使得在视见体外面的那部分多边形可以被裁去。

例如，如果含有多边形的平面为：

$$Ax + By + Cz + D = 0$$

直线如前述以参数化方法定义，则相交由下式给出：

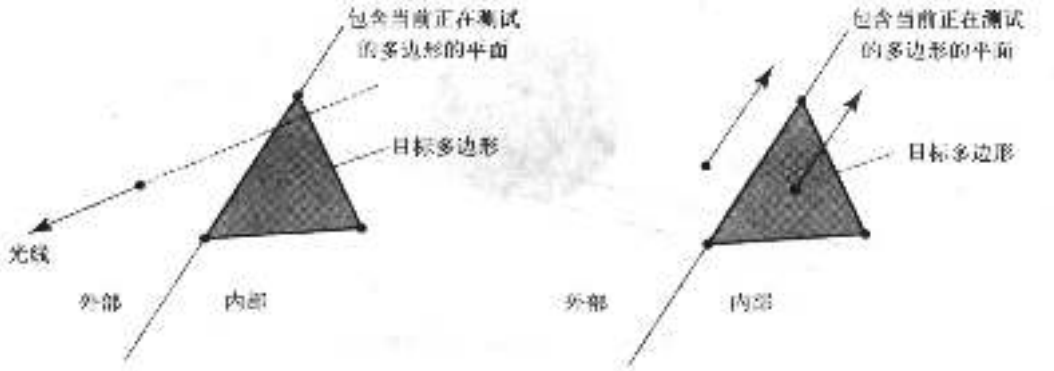
$$t = \frac{-(Ax_1 + By_1 + Cz_1 + D)}{(Ai + Bj + Ck)} \tag{1-4}$$

当 $t < 0$ 时可以退出测试。这意味着光线处于半空间中，它由不含多边形的平面定义（见图1-13a）。如果分母为零也可以退出，因为这意味着光线和平面平行。在这种情况下，光线的初始点或者在多边形之内或者在其外面。我们可以通过检查分子的符号来检测这种情况。如果分子为正则光线位于由物体之外的平面所定义的半空间中，这时不必再进行进一步的测试（见图1-13b）。

测试一个点是否包含于多边形中的直接方法简单但费时。如果点在多边形内则从点到每一个多边形顶点之间的夹角之和为360°。但是如果点在多边形之外则不是如此。

这种直接的方法存在三个缺点。除非同时测试多边形相对于光线的方向是前向还是后向，否则，当第一次相交出现时不能停止测试；包含测试尤其费时；当一条光线和一个多边形的边重合时还可能出现错误。

18
19



a) 在不含有物体（$t<0$）的半空间中的一条光线　　b) 一种可能的退出条件。光线平行于包含当前正在测试的多边形的平面中，它在物体内或在物体之外

图 1-13

所有这些缺点都可以通过一个由Haines（1991）开发的方法来克服。我们使用包含多边形的平面的概念，并用这个概念定义一个半空间。这个平面的一侧的所有点都在多边形之外，而在其另一侧的点则包含在多面体中。所有半空间内的逻辑交都处于被多面体包围的空间中。与平面相交的光线产生一条有向线段（在光线的方向上是无界的），线段由相交点及光线的方向确定。容易看出，所有有向线段的逻辑交给出了通过此多面体的线段。如此继续，当一条平行光线出现一个“外部”原点时结束测试。否则此算法将对每一个多边形进行测试并估算有向线段的逻辑交。考虑图1-14中的例子。对于每一个平面，我们根据相对于光线的方向将其分类为前向的或后向的。这一分类由方程（1-4）中的分母的符号给出（正值为后向，负值为前向）。形成有向线段逻辑交的条件包含于算法中，算法如下：

{将t_{in}初始化为一个大的负值
t_{out}初始化为一个大的正值}
if（平面为后向的）and（$t<t_{in}$）
then $t_{in}=t$
if（平面为前向的）and（$t>t_{out}$）
then $t_{out}=t$
if（$t_{in}>t_{out}$）then{退出——光线消失}

20

1.4.4 相交——光线与包围盒

光线与包围盒的相交是很重要的，因为包围盒在限定体积时比球更为有用，尤其是对于层次结构的情况。而且，通用的包围盒还可以用作有效的限定体。

通用的包围盒由一些相互平行的平面对构成，但平面对之间可以是任意夹角。本小节中我们考虑形成矩形实体的特殊的盒，其每一对平面的法向与光线跟踪轴的方向或物体空间轴的方向相同。

测试一条光线是否与这样一个包围盒相交是很直接的。我们依次测试每一对平行平面，计算沿着光线到第一个平面的距离（$t_{近}$）和到第二个平面的距离（$t_{远}$）。比较过程中保留$t_{近}$中的较大值和$t_{远}$中的较小值。如果$t_{近}$的较大值比$t_{远}$的较小值大，则光线不能与包围盒相交。作为例子，在xy平面中出现这种情况，如图1-15所示。如果出现一个相交则$t_{近}$给出相交点。

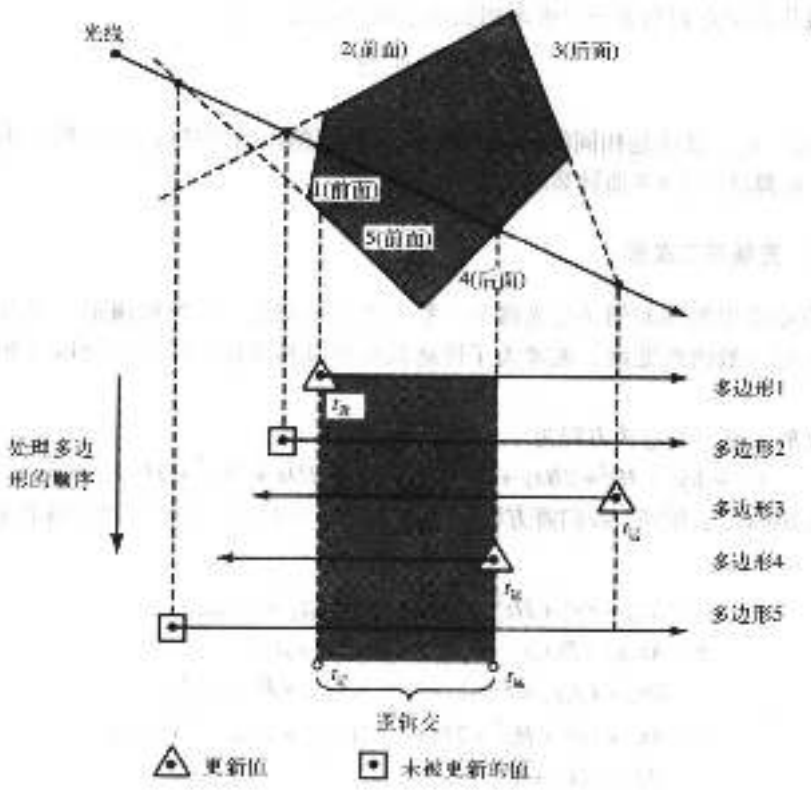


图1-14 光线与凸多面体相交测试（Haines（1991）的方法）

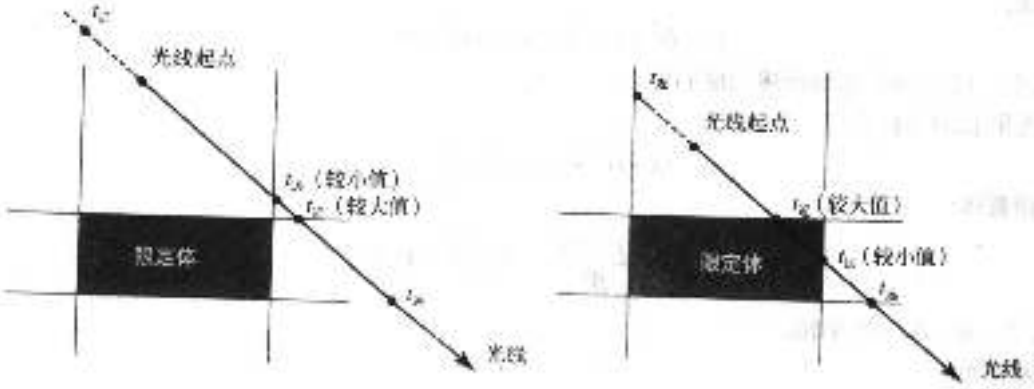


图1-15 光线与包围盒相交

比这种算法更简洁的阐述是把一对平行平面的相交点之间的距离看成区间，如果区间之间有交点，则光线接触到视见体。如果区间不相交则光线离开视见体。

因为凸多边形被削减为一个矩形实体，所以可以以包围盒的尺寸来定义所需的距离。如果包围盒的尺寸为 (x_{b1}, y_{b1}, z_{b1}) 和 (x_{b2}, y_{b2}, z_{b2})，则对于x平面对在光线方向上的距离由下式给出：

$$t_{1x} = \frac{x_{b1} - x_1}{x_2 - x_1}$$

即为沿着光线从其起点到与第一个平面相交点之间的距离，且

$$t_{2x} = \frac{x_{b2} - x_1}{x_2 - x_1}$$

21
~
22

对于t_{1y}，t_{2y}和t_{1z}，t_{2z}，算法是相同的。t_1集合中的最大值给出所需的$t_{近}$，而t_2集合中的最小值给出所需的$t_{远}$。此算法可在y平面计算时结束。

1.4.5 相交——光线与二次形

在1.4.2节中给出的球的例子是光线与一般二次形相交的一个特殊情形。可以将光线与二次形相交看作是一般情形处理，或者为了提高效率可以将其按“特殊”物体（如圆柱体）来处理。

对于二次形，通用的隐式方程为:

$$Ax^2 + Ey^2 + Hz^2 + 2Bxy + 2Fyz + 2Cxz + 2Dx + 2Gy + 2Iz + J = 0$$

与处理球时采用的方法相同，我们将方程1-1代入到上述方程中，对于二次形得到系数a，b，c，如下:

$$\begin{aligned} a &= Ax_d^2 + Ey_d^2 + Hz_d^2 + 2Bx_dy_d + 2Cx_dz_d + 2Fy_dz_d \\ b &= Ax_1x_d + B(x_1y_d + x_dy_1) + C(x_1z_d + x_dz_1) + \\ &\quad Dx_d + Ey_1y_d + F(y_1z_d + y_dz_1) + Gy_d + Hz_1z_d + Iz_d \\ c &= Ax_1^2 + Ey^2 + Hz_1^2 + 2Bx_1y_1 + 2Cx_1z_1 + 2Dx_1 + 2Fy_1z_1 + \\ &\quad 2Gy_1 + 2Iz_1 + J \end{aligned}$$

对于二次形，方程分别为:

- 球:

$$(x-l)^2 + (y-m)^2 + (z-n)^2 = r^2$$

其中，(l, m, n) 如前所述为球的中心。

- 无限长的圆柱体:

$$(x-l)^2 + (y-m)^2 = r^2$$

- 椭圆体:

$$\frac{(x-l)^2}{\alpha^2} + \frac{(y-m)^2}{\beta^2} + \frac{(z-n)^2}{\gamma^2} - 1 = 0$$

其中，α，β，γ为半轴。

- 抛物面:

$$\frac{(x-l)^2}{\alpha^2} + \frac{(y-m)^2}{\beta^2} - z + n = 0$$

- 双曲面:

$$\frac{(x-l)^2}{\alpha^2} + \frac{(y-m)^2}{\beta^2} + \frac{(z-n)^2}{\gamma^2} - 1 = 0$$

1.4.6 光线跟踪几何——反射和折射

在这一小节中给出的公式是标准公式，是适合嵌入到简单的光线跟踪程序中的形式。此

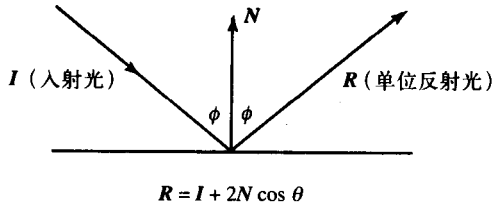
公式来源于7.1节中给出的Fresnel定律。

每当一条光线与一个表面相交时，一般情况下，它都会产生一个折射和一个反射。反射的方向以一个单位向量表示，由下式给出（如1.3.2节中所见）：

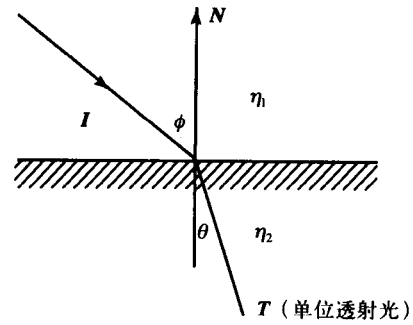
23

$$\begin{aligned} \boldsymbol{R} &= 2\boldsymbol{N}\cos\phi - \boldsymbol{L} \\ &= 2(\boldsymbol{N}\cdot\boldsymbol{L})\boldsymbol{N} - \boldsymbol{L} \end{aligned}$$

其中，$\boldsymbol{L}$和$\boldsymbol{N}$为代表入射光线方向的单位向量，分别与光线方向向量和表面法向量相同。$\boldsymbol{L}$、$\boldsymbol{R}$、$\boldsymbol{N}$同面。这些向量如图1-16所示，其中$\boldsymbol{I} = -\boldsymbol{L}$。



$\boldsymbol{R} = \boldsymbol{I} + 2\boldsymbol{N}\cos\theta$



$\boldsymbol{T} = \mu\boldsymbol{I} - (\cos\theta - \mu\cos\phi)\boldsymbol{N}$

图1-16 反射和折射几何

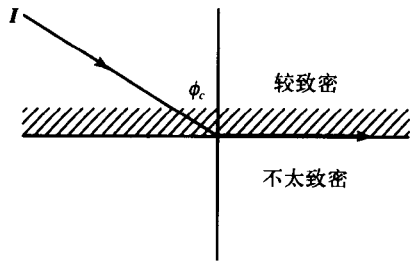
击中一个部分或完全透明物体的光线会由于光在不同介质中的速度变化而产生折射。入射角和折射角用Snell定律进行关联。该定律有如下形式：

$$\frac{\sin\phi}{\sin\theta} = \frac{\mu_2}{\mu_1}$$

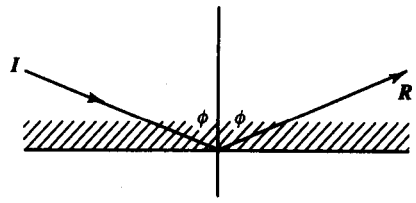
其中，入射光线和透射光线与$\boldsymbol{N}$同面。透射光由$\boldsymbol{T}$表示，公式如下并如图1-16所示：

$$\begin{aligned} \boldsymbol{T} &= \mu\boldsymbol{I} - (\cos\theta + \mu\cos\phi)\boldsymbol{N} \\ \mu &= \mu_1 / \mu_2 \\ \cos\theta &= \frac{1}{\mu^2}(1 - \mu^2(1 - \cos^2\phi))^{\frac{1}{2}} \end{aligned}$$

如果一束光线从一个较致密的介质进入一个不太致密的介质中，则折射光有可能会平行于表面（见图1-17）。ϕ_c称为临界角。如果ϕ增大则可能出现完全内部反射。



a) ϕ_c = 临界角



b) $\phi > \phi_c$

图1-17 物体中的内部反射

1.5 图像平面中的插值性质

在主流的绘制技术（即绘制多边形技术）中，内部像素所需的各种性质都是通过对多边形顶点（即顶点所投影到的像素）的性质的内插获得的。这种插值称为双线性插值，它是这类明暗处理方法效率的基础。

参考图1-18，通过在代表多边形的像素集中从上向下移动一条扫描线，并且同时在适当的顶点性质对之间进行插值得到一条扫描线的起点和终点来得到插值。于是，沿着一条扫描线的插值产生了每一个像素的性质值。插值方程为（对于示意图中所示的特定的边对）：

$$p_a = \frac{1}{y_1 - y_2}[p_1(y_s - y_2) + p_2(y_1 - y_s)]$$

$$p_b = \frac{1}{y_1 - y_4}[p_1(y_s - y_4) + p_4(y_1 - y_s)]$$

$$p_s = \frac{1}{x_b - x_a}[p_a(x_b - x_s) + p_b(x_s - x_a)]$$

这些方程在正常情况下以增量形式进行计算。例如，最终的方程成为：

$$p_s := p_s + \Delta p$$

对每一条扫描线，计算一次常数Δp。

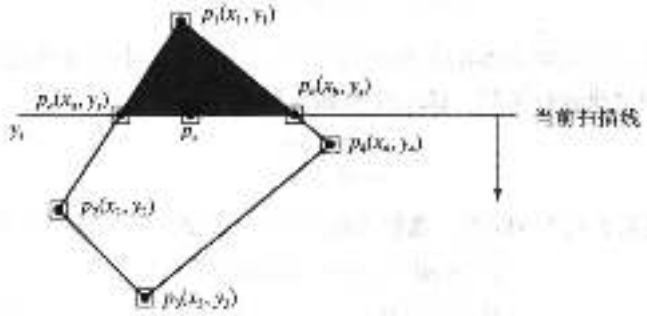


图1-18 从顶点像素的值对某一像素的性质进行插值

24
2
26

第2章　三维物体的表示和建模（1）

2.1　三维物体的多边形表示
2.2　物体的构造实体几何表示
2.3　物体表示的空间细分技术
2.4　用隐函数表示物体
2.5　场景管理和物体表示
2.6　总结

引言

三维计算机图形学的主要目的就是从物体的描述或模型产生物体或场景的二维图像。物体可以是一个实际的物体也可以只是一种计算机描述。一个不太普遍但非常重要的用途是在物体模型的动作的创建以及可视化上的应用。这种应用在交互式CAD应用程序中出现，其中设计者运用可视化技术来辅助创建物体的动作。大多数物体的描述都是近似的，它们仅仅精确到将物体的几何或形状描述输入到绘制程序中，以产生一个质量可接受的图像的程度。然而，在许多CAD应用程序中描述必须精确的，因为描述要被用于驱动一个制造过程。其最终的产品并不是二维图像而是真实的三维物体。

建模和表示是一个通用词汇，可以应用到物体的下列方面:

- 建立三维计算机图形学表示。
- 用于表示物体的技术、方法或数据结构。
- 表示的处理——特别是改变一个现有模型的形状。

建立计算机图形学物体的方法几乎像物体自身那样多样并且可变化。例如，可以通过CAD界面构建一个结构化的物体。我们可以直接从设备（如激光测距仪或三维数字化仪）来获取数据，还可以使用某些基于扫描技术的接口程序。这里，通过沿着一条中心曲线来扫描一个断面，以创建一种所谓的中空实体。到目前为止，创建的方法倾向于手动或半手动，需要由设计者通过一个接口来参与工作。由于像虚拟现实（VR）这样的应用对于高度复杂场景描述的要求增加，所以正在开发一些自动化方法。对于现存的真实物体应用虚拟现实技术来讲，通过照片或者视频建立计算机图形学表示是一个有吸引力的发展方向。

在计算机图形学中，物体的表示在很大程度上还是未解的问题。我们可以把机器或者绘制程序所需要的表示和用户或用户界面所需要的表示区分开来。用一些小的多边形来表示物体（即多边形网格表示）是最流行的机器表示。然而，对于用户或物体的创建者来讲它是一个不太方便的表示方法。尽管如此，这种方法还是被用户或者机器表示方法所采用。其他方法把用户和机器表示分开来处理。例如，构造用户或接口表示的双三次参数曲面片和CSG方法，可以将表示转换为绘制所用的多边形网格。

多边形网格表示在物体很复杂和需要详尽描述的时候有很多缺点。在主流计算机图形学

中，表示一个物体的多边形的个数可以从几十个到成百上千个。网格个数的多少根据在绘制时间和创建物体的成本以及将这样的物体用于动画或者虚拟现实环境中的可行性方面有非常严格的区分。在动画中还会产生其他问题，这时模型必须表示物体的形状同时还要受动画系统所控制。这个系统可能要求计算碰撞，或者是物体按时间函数来改变其形状。尽管如此，这种多边形网格仍然是主流计算机图形学中最好的。造成这种情况的部分原因是有效算法的建立以及绘制这种描述的硬件的发展。从绘制的角度来看，这已经导致了一种奇怪的现象，即用许多简单的元素（多边形）表示形状，而不是用少得多的但是更复杂、更精确的元素（比如双三次参数曲面片）（见3.4.2节）来表示形状。

处理一个存在的物体形状的能力强烈地依赖于表示。多边形网格不能进行简单形状的处理。移动网格的顶点立刻就会破坏多边形表示结果，因为这种表示结果已经按照一定的精确度转换成了多边形的坐标，而这个精确度与被表示表面的局部曲率有关。例如，想象一下扭
28 曲一个用六个正方形表示的立方体，扭曲的物体不能由现有的六个多边形来表示。另一个形状处理的问题是大小。有时我们需要改变一个物体的大部分，这可能要同时移动很多元素而另一些时候我们可能只需要一种细微的变化。

不同的表示方法各有优缺点，但是对于很多现存的问题没有一个通用的解决方案。而通常的情况是特定的建模方法用于特定的目的。这种趋势的一个比较好的例子是构造实体几何（Constructive Solid Geometry，CSG）方法的建立和这种方法在交互式CAD中的普遍采用，因为这个方法提供了一种直接的接口用来交互式地设计复杂的工业物体和表示这些物体。CSG是一种限定性表示方法，我们只能用它来建模那些由基本形状进行有限组合构成的形状，或者系统里所固有的元素。

那么，如何选择表示方法呢？答案是这依赖于物体的性质，以及我们使用的特定的计算机图形学技术，这些技术将物体引入到生活和应用中。所有这些因素都是相互作用的。可以严格地用数学公式来表示某些三维物体，例如圆柱体或球，而对其他情况，我们使用近似的表示方法。对于那些不能严格地用数学方法表示的物体，在表示的精确度和所用的大量信息之间有一个折中方案。图2-1中描述了一个多边形网格结构。可以通过增大多边形的分辨率来提高表示的真实性，但这意味着在绘制时间上要花费得更多。

这种外延的高度不可行性导致了对于非常复杂的或者独特的物体，例如人类的头部，必须建立一种混合方法。例如，在表示一个特定的人头时，可以使用多边形网格模型和图像纹理映射相结合的方法。人头的实体形状由通用的多边形网格来表示，这些多边形网格被延展以匹配要建模的人头的实际尺寸。而详细的外表则通过把图像纹理映射到网格上来获得。其出发点是几何上的细微变化是由纹理映射而不是通过几何上的细微偏移来实现的。当然这样做并不是完美的，因为照片中的细节还要依赖于光照条件和实际的几何细节。但是这种方法的应用却正在增加。究竟将纹理映射看作是表示的一部分还是看作是绘制过程的一部分，这是一种观点问题。但是在这种情况下我们肯定会使用图像纹理映射以便用少量的多边形再加上一幅照片来表示像人头这样复杂的物体。

多边形分辨率和图像纹理映射之间的折中可以达到极至。在计算机游戏界，绘制到屏幕上的多边形的数目必须限定在一定数量之内，例如，在PC上每秒15帧。最近有一个由游戏者
29 组成的足球游戏，他们的头都是用立方体来建模的，立方体上映射了照片纹理。

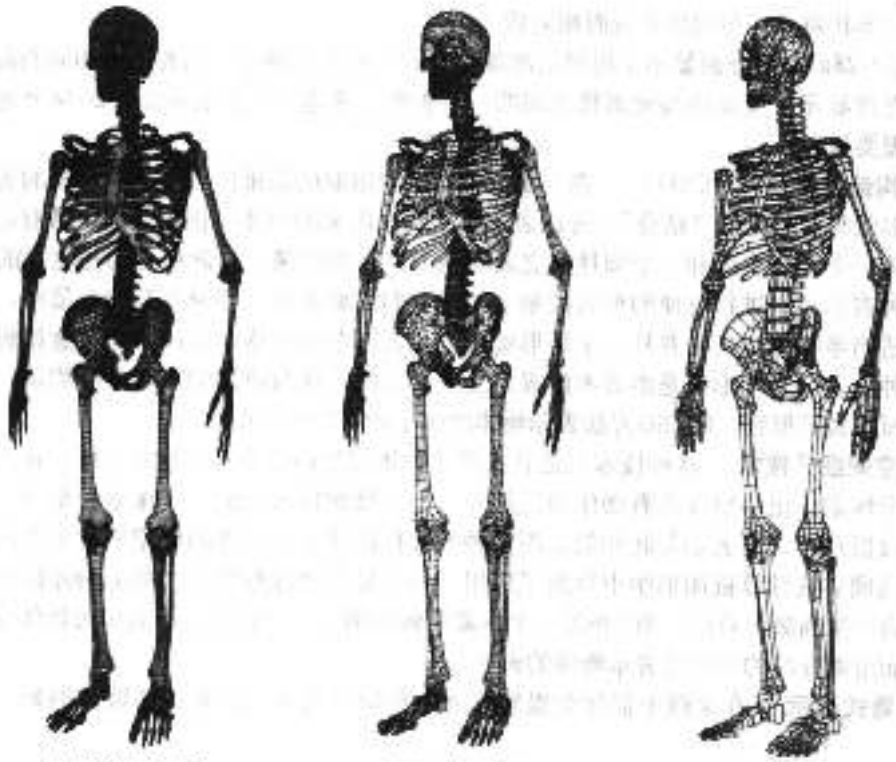


图2-1　线框的艺术　—来自Viewpoint Digital的一个插图

资料来源："3D models by Viewpoint Digital, Inc." *Anatomy*, Viewpoint's 3D Dataset™ Catalog, 2nd edn.

现在我们按照使用频率的大概顺序列出计算机图形学中使用的主流模型。

1）**多边形**　物体用多边形小平面组成的网或网格来近似表示。用这种形式我们可以按照所选的精确度来表示任何形状的物体。然而，在这里精确度有些随意。再来看图2-1，它实际上需要有142 000个多边形，也许我们可以减少多边形的个数而不降低绘制图像的质量，但需要减多少呢？人们设计了明暗处理算法来可视化地转换小平面的描述，使得一个个的小平面在明暗处理图中是不可见的（侧面边界线除外）。与多边形的分辨率相关的是屏幕上物体的最终投影尺寸。当用几千个多边形表示一个复杂的物体并将其投影到仅有几个像素组成的屏幕区域时会产生浪费。

2）**双三次参数曲面片**（见第3章）　这些曲面片是"弯曲的四面体"。一般来讲，可以说这种表示方法与多边形网格相似，只是这时各个多边形的表面变成了弯曲的。每一个曲面片都用一个数学公式来定义，该公式给出曲面片在三维空间中的位置及其形状。这个公式使我们能够产生该曲面片表面上的每一个点。我们可以通过改变数学定义来改变曲面片的形状或曲率，这导致了很强的交互能力。然而问题也是很明显的，要绘制或可视化曲面片是非常昂贵的。当改变曲面片网中的一个曲面片的形状时，出现了如何保持曲面片之间"光滑性"的问题。双三次参数曲面片可以是精确的表示也可以是近似的表示。它们只能是其自身的精确表示，这意味着，一个物体，比如说一辆汽车车身，如果其形状刚好是曲面片的形状，则它只能被精确表示。这个令人烦恼的说明是必要的，因为当将表示用于真实的或现存的物体时，

30

模拟出的形状将不必与物体的表面相对应。

在图3-28a和c中分别显示了用双三次参数曲面片和多边形小平面表示的相同的物体。这个图很清楚地显示了复杂性与元素数之间的一种折中，多边形网格表示需要2048个元素而曲面片表示需要32个。

3) **构造实体几何**（CSG） 在一定的刚性形状限制的范围内这是一种精确的表示。通过将基元形状与几何基元“结合”，可以表示很多构造出来的物体。例如，一块带有孔的金属可以定义为一个矩形实体和一个圆柱体之间相减而产生的结果。与此相连的是这样的事实，即这样一种表示可以进行方便的形状控制。如果我们需要定义一个带有孔的金属板，可以定义一个有适当半径的圆柱体并从一个矩形实体上减去这个圆柱体，以此来代表金属板。CSG方法是一种体表示，即形状是由基本体或基元来表示的。这与前面的两种方法相反，前两种方法用平面来表示形状。用CSG方法表示物体的例子如图2-14所示。

4) **空间细分技术** 这种技术只是意味着把物体空间细分为基本的立方体，称为体素，再把每一个体素标记为空或含有物体的某部分。把二维物体表示成一些像素的集合，这些像素是物体投影点，三维表示与此相似。用这种方式标记所有的三维物体空间肯定是昂贵的，但这种方法确实在计算机图形学中得到了应用。特别是在光线跟踪中，如果物体以这种方式表示，则会产生高效的算法。图2-16为一个体素物体的例子。这种方法是表示被物体占据的三维空间，而前面介绍的方法是表示物体的表面。

5) **隐式表示** 在文献中偶尔会提到作为一种物体表示形式的隐函数。例如，一个隐函
31 数为:

$$x^2 + y^2 + z^2 = r^2$$

这是一个球的定义。就其自身而言，这种隐函数在计算机图形学中用途有限，因为只有有限数量的物体可以用这种方式来表示。而且，从绘制的角度考虑这是一种不方便的表示方法。然而，我们必须注意到，这种表示确实经常在三维计算机图形学中出现，尤其是在光线跟踪中，光线跟踪方法中经常用到球，它或者是作为物体本身出现，或者是用于其他多边形网格表示的限定物体。

可将隐式表示扩展到隐函数，从较宽松的定义来看，可以认为隐函数是一种物体，这种物体是由数学上定义的表面来定义的，这个表面的形状受到函数所隐含着的基元（比如球）的集合的影响。隐函数主要用在形状变化动画中，在表示真实物体时用途有限。

以上把分类按照普及的程度进行安排。另一种有用的比较是：对于体素和多边形网格来讲，每个物体的表示元素的数量可能会很多（如果要获得精确描述），但是表示的复杂性低。这与双三次参数曲面片相反，用这种方法在大多数情况下基元的数量似乎少了很多，但是，表示的复杂性高了。

我们不应该从上述的分类中得出结论说表示方法的选择是自由的。表示形式是由绘制技术和应用共同决定的。例如，考虑连续或离散的表示方法间的区别。离散的表示（如多边形网格法）用于表示现实世界中任意形状的物体，很难看出除此之外我们该如何处理这样的物体。在医学图像中，初始的描述是离散的（体素），因为这是图像技术所产生的。而在CAD图像中，需要一种连续的表示方法，因为我们最终会用内部描述生产诸如机器部件这类产品。因此，表示必须是精确的。

CSG表示方法不太容易进行比较。它既是连续的表示方法，也是离散的表示方法。它是相互作用的基元的离散的结合，其中有些基元可以用连续函数来描述。

另一个重要的区别是表面与体积的表示法。多边形网格是物体表面的一种近似表示，绘制引擎为该表面提供可视化。对于Gouraud模型，其算法只关心使用与表面表示相关的几何性质。在光线跟踪算法中，由于在跟踪光线通过空间和求光线与物体的相交上需要付出很大的代价，所以表面表示方法就意味着高的绘制成本。当根据物体占据的空间对物体进行标记时，采用体积表示法将会大大降低绘制的成本。

绘制方法与表示之间的关系在辐射度方法中非常重要，在该方法中，为了防止最终图像中出现重要的缺陷，在表示和算法的执行之间必须有某种交互。随着算法的进行，表示方法必须与之相适应，以使得那些需要更多关注的地方获得更精确的计算。换句话说，由于方法的复杂性，很难事先（priori）决定表示的详细程度。随着绘制算法的进行，对于场景给出一种表示的概念的困难程度在于辐射度方法从根本上是困难的，还在于它目前还难于作为一种主流工具。

2.1 三维物体的多边形表示

这是一种三维计算机图形学中的经典表示形式。在这种表示法中，物体用多边形小平面组成的网格来表示。在一般情况下，物体只有曲面，而这些小平面是对这样的表面的近似（见图2-2）。多边形也可以包含一个顶点计数，该计数来自创建模型所采用的技术。或许我们可以将所有的多边形限制为三角形，例如，为了使专用的硬件或图形加速卡取得最优的性能，可能有必要这样做。



图2-2 用多边形小平面近似一个曲面

多边形表示方法在计算机图形学中是普遍存在的。有两个原因造成这种情况。创建多边形物体是很容易的，尽管对于复杂的物体这个过程是费时且昂贵的。而且，很明显存在对用这种方法所表示的物体进行明暗处理的有效方法，正如我们已经阐明的那样，多边形网格严格意义上是一种机器表示法，而不是一种方便的用户表示法，并且这种网格还经常被用于不能直接进行绘制的其他表示法中，于是，双三次参数曲面片、CSG、体素表示等方法都经常在绘制之前被转换为多边形网格。

采用多边形网格表示法有一些实际的困难。到目前为止，最大的困难是精确度。模型的精确度，即小平面表示与物体的曲面之间的差别通常是任意的。对于我们所考虑的最终图像质量而言，各个多边形的尺寸理想情况下应该依局部的空间曲率而定。在那些曲率变化迅速的地方，对每单位曲面需要较多的多边形。其换算系数倾向于与创建多边形所用的方法有关。例如，如果由存在的物体创建网格，用三维数字化仪来确定多边形顶点的空间坐标，则数字化仪的操作员可以基于经验来决定每一个多边形应该多大。有时，多边形是通过算法获得的（例如，创建一个旋转的实体，或在一个双三次曲面片细分算法中），并且可能对单位表面上

多边形的个数倾向于更精确的估计。

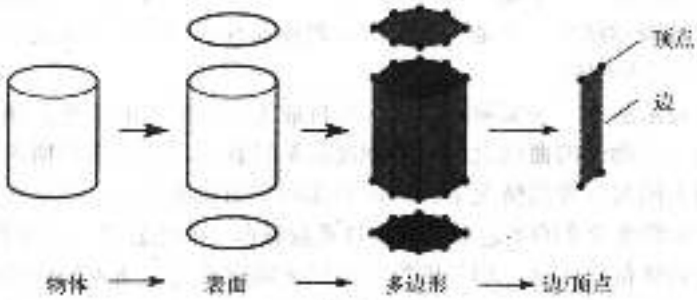
三维图形学中最突出的进步出现在20世纪70年代，即出现了明暗处理算法。这种算法有效地处理多边形物体，同时通过一种插值技术消除了在表示中出现的小平面与小平面之间连线可见的情况。这一因素再加上在固有的程序化绘制硬件中的新的发展确保了多边形网格结构的地位。

最简单的情况下，多边形网格是一种由相链接的坐标 (x, y, z) 的列表来表示的多边形组成的结构，这些相链接的坐标值是多边形的顶点（边可以明确表示也可以隐含表示，正如我们将要看到的那样）。因此存储描述物体的信息最终是一个点的列表或者是顶点的列表。作为物体描述的部分，还要存储其他用于接下来的处理的几何信息。这些信息通常是多边形法向和顶点法向。这些信息只计算一次，把这些信息放在物体的数据结构中，并且对它们进行任何应用于物体的线性变换是很方便的。

将多边形排列成简单层次结构是很方便的。图2-3a显示了一种我们称其为概念层次的分解。其原因是这样分解看起来更清晰，多边形被按照表面分成组，而表面被组合成物体。例如，一个圆柱体具有三个表面，即一个顶平面、一个底平面再加上一个曲面。这样分组的原因是，我们必须区分那些作为近似的表面的边和真实存在的边。例如，对圆柱体近似时，在曲面上相邻矩形之间的边。这些划分出来的组在接下来由绘制过程处理时的处理方法是不同的，真实的边必须保持可见，而那些对曲面近似的边则必须不可见。图2-3b为图2-3a的拓扑结构的一种更正规的表示。

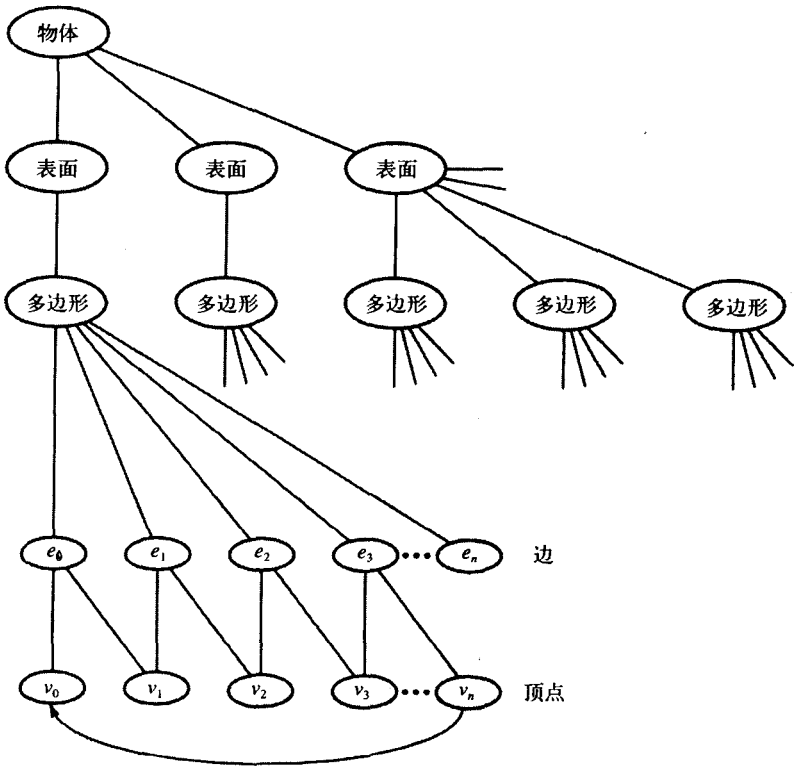
实现这种关系的实际数据结构的一个例子如图2-3c所示。这个图包含了水平和垂直的层次链接，当程序员以水平的顺序访问相邻的实体时需要用到这种链接。此数据结构还包含一个顶点引用表，实际的顶点只存储一次（每一个含有这些顶点的多边形都引用该表）。实际结构和拓扑图之间的另一个区别是允许直接对低层实体的访问。对物体的线框外形可视化的方
34 法被广泛地采用。要产生线框图像，需要按层次直接地访问边。在边的层次和多边形的层次之间的垂直链接可以用前向指针也可以用后向指针，这要依访问此结构的绘制程序的类型而定。在一个扫描线绘制程序中，边是最顶层的实体，而在Z缓冲器绘制程序中多边形是最顶层的实体。Z缓冲器绘制程序把多边形作为独立的实体，每次绘制一个多边形。而扫描线绘制程序对所有跨跃被绘制的扫描线的多边形进行绘制。

上面所描述的方法更倾向于是基于顶点的边界模型。有时，需要采用基于边的边界模型，这种模型的最好表现形式是一种翼边数据结构（Mantyla 1988）。基于边的模型根据边的封闭序列表示表面。

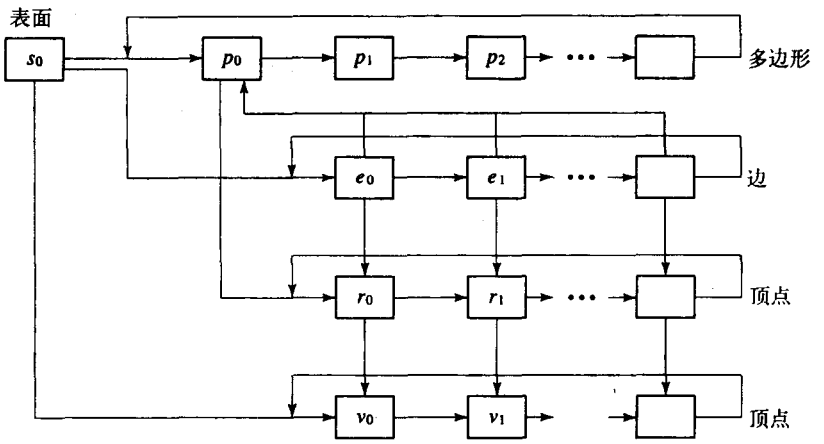


a) 概念层次

图2-3 一个物体的多边形网格表示



b) 拓扑结构表示



c) 一个实际的数据结构

图2-3 （续）

上面描述的数据结构把与物体的多边形小平面相关的基本几何信息进行了封装。应用程序和绘制程序所需的信息通常都包含在场景/物体数据库中。下面的列表详细给出了多边形网格结构中最常见的属性。它们是数据结构指针、实数或者二进制标志。在实际应用中这些信息并不会全部出现，但在大多数物体表示中会出现其中的一部分。

• 多边形属性:

1) 三角形或不是。

2) 面积。

3) 包含多边形的平面的法向。

4) 包含多边形的平面的系数 (A, B, C, D)，其中$Ax + By + Cz + D = 0$。

5) 是否为凸多边形。

6) 平面上是否有洞。

• 边属性:

1) 长度。

2) 边是位于两个多边形之间还是位于两个表面之间。

3) 该边的每一侧的多边形。

35 ~ 36

• 顶点属性:

1) 包含该顶点的多边形。

2) 明暗处理或顶点法向——包含该顶点的多边形法矢量的平均值。

3) 指定映射到二维纹理图像的纹理坐标 (u, v)。

所有这些都是绝对性质，当物体产生时就存在。随着绘图流程的进行，多边形可以获取这些属性。例如，如果一个边处于两个多边形之间，并且这两个多边形的法向分别面向观察者和背对观察者，则可以把该边标记为轮廓边。

在计算机图形学中，以很多不同的方式出现的一个很明显的问题是比例问题。对于多边形表示方法而言，在许多应用中这意味着如果观察距离和多边形的分辨率达到使得许多多边形投影到一个像素上这样一种程度，那么就不能在一个模型中绘制所有的多边形。这个问题困扰了飞行模拟器（对于计算机游戏也是如此）以及虚拟现实应用程序。一个很明显的解决方案是使模型有层次，并对所投影的屏幕区域使用一个适当的模型。这样做有两个问题，第一个问题是在动画过程中（在动画应用程序中这个问题最关键），模型之间的切换可能会在动画序列中引起视觉上的混乱。这时用户可能会看到从一个分辨率水平切换到另一个分辨率水平。另一个问题是如何产生层次以及应该有多少层。很明显，我们可以从最高分辨率模型开始，然后再细分。但是，这样并不太直观。我们将在2.5节中更详细地讨论这个问题。

2.1.1 创建多边形物体

尽管多边形网格是计算机图形学中最常见的表示形式，但是多边形网格法建模虽然直观却很繁琐。这种表示方法的普及是源于建模容易以及处理多边形物体的绘制策略（包括硬件的和软件的）的出现，更重要的是，对于被建模的物体没有形状及复杂性的限制。

通过用一种三维定位器设备，可以通过推动顶点来进行交互式的建模。但实际上这并不是一个非常有用的方法。对于这种模型，除了改变形状之外很难做其他事。一旦创建了一个物体，如果不改变相邻多边形的话，很难单独改变一个多边形。所以，大多数创建方法只使用设备或只使用程序。唯一允许用户进行交互的方法是下面第四项。

四个常用的多边形建模方法为:

1) 使用三维数字化仪或者采用等价的手工策略。

2) 使用像激光测距仪这样的自动化设备。

3) 用数学描述产生一个物体。

4) 通过扫掠（sweeping）产生一个物体。

37

前两种建模方法将真实的物体转换为多边形网格，而后面两种方法通过定义产生模型。我们将用数学公式产生的模型与那些通过用数学定义产生的曲线进行交互而产生的模型区别对待。

2.1.2 多边形物体的手工建模

建模物体的最容易的方法是用三维数字化仪手动进行的，操作人员根据经验判断在物体上放置那些将要成为多边形顶点的点，然后再把这些顶点的三维坐标通过一个三维数字化仪输入到系统中。顶点与多边形之间的关联是很明显的，保证有一个合适的表示的普适策略是在物体的表面画一个网，就像把一张真实的网放到物体上那样，在那些弯曲的网线交叉点处定义多边形顶点的位置。这个过程的一个历史图片如图2-4所示。这张图片显示了1974年学生们正在创建一辆汽车的多边形网格模型。该图片来自早期计算机图形学的杰出先驱Sutherland等（1974）的经典论文。

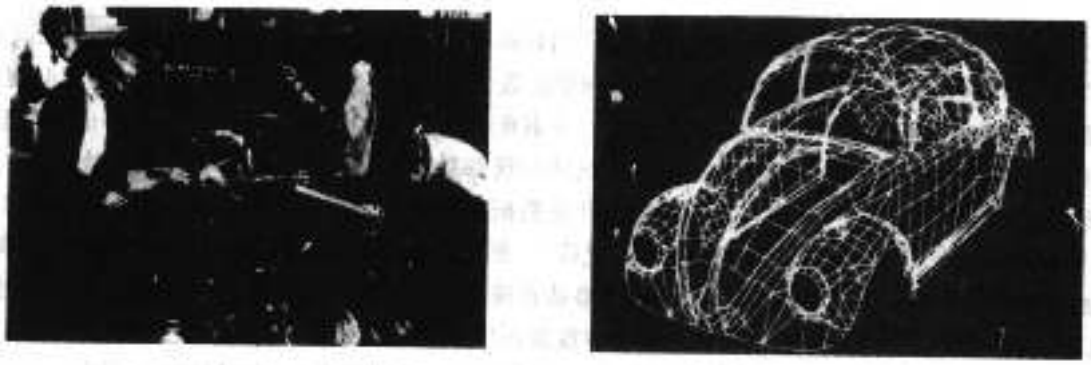
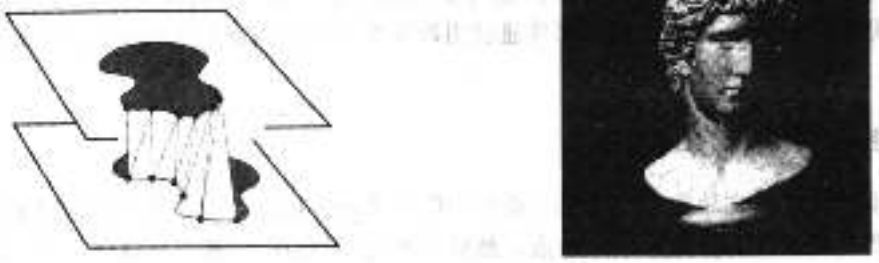


图2-4 Utah Beetle——手工建模的一个早期的例子

资料来源：Beatty and Booth, *Tutorial: Computer Graphics*, 2nd edn, The Institute of Electrical and Electronics Engineers, Inc.:New York. © 1982 IEEE.

2.1.3 多边形物体的自动产生

能够从真实物体创建非常精确或具有高分辨率的多边形网格物体的设备是激光测距仪。在一种设备中，物体被放置在一个旋转的桌子上并置于光束的路径中。桌子还可以垂直地上下运动。激光测距仪通过测量与物体表面之间的距离返回一组轮廓，即物体的截面以及一组紧挨着的平行面。这是一种"蒙皮"（skinning）算法，即对轮廓对进行操作，将边界数据转换为非常大量的三角形（见图2-5a）。图2-5b是用这种方法多边形化物体的绘制结果。对于这个物体，蒙皮算法产生了超过400 000个三角形。已知这些三角形中只有大约一半在屏幕上是可见的，而且这个物体只投影到半个屏幕表面，这意味着平均每个三角形投影到一个像素点 38
上。这很清楚地表明了前面所提到的观点，即用多边形分辨率来绘制是极端浪费的。这时，一个多边形投影到的平均屏幕面积趋近于一个像素。对于模型创建来讲，激光测距仪有严重的缺陷，即在所描述的框架中（完全自动化的旋转桌面设备），测距仪只能精确地模拟凸状物体，带有凹陷的物体将会存在不能被入射光照到的表面。



a) 蒙皮算法连接顺序出现的截面上的点构成二维多边形

b) 由蒙皮算法产生的具有400 000个多边形的物体

图2-5 由一个激光测距仪进行扫描，再用一种简单的蒙皮算法多边形化后绘制的多边形物体

2.1.4 多边形物体的数学产生

很多多边形物体都是通过界面产生的，用户向这个界面中放入以一组曲线形式表示的数学描述，这组曲线是二维空间或两参数空间的函数。在CAD应用程序中尤其如此，在CAD中最流行的例子是以各种不同的方式来扫掠一个截面。这种方法有两个优点，一是它非常显而易见，用户与一些形状的概念打交道，而这些形状都是在由单个多边形小平面构造物体时的低层活动中删除掉的。取而代之的是，形状是根据概念来定义的，这些概念与物体的形状相关——Snyder（1992）将其称为“形状的逻辑”。然后由一个程序取得用户的描述并把它转换成多边形。将用户的描述转换成多边形网格是直接进行的。这种方法的第二个优点是，它可以与将多边形作为基元的方法或者双三次参数曲面片方法结合使用（见3.6节）。

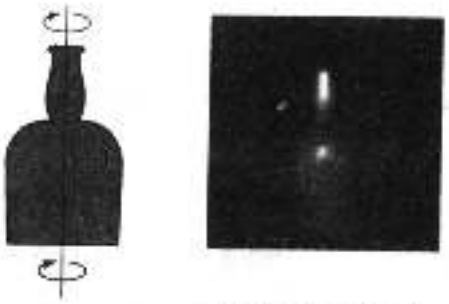
这一方法最熟悉的例证是旋转一个实体，比如说一个垂直的截面被扫掠180°，产生一个具有圆形水平截面的实体（见图2-6a）。实体旋转的明显限制是其只能表现那些具有旋转对称性的物体。

一个更强大的产生模型是这样得到的，即扫掠一个其半径受轮廓线（profile）控制的圆，直到一条直的垂直脊线（spine）为止，可以产生相同的实体（见图2-6b）。当此轮廓线为一常数时，就是我们熟悉的挤压（extrusion）的概念。而去掉对圆形截面的限制，就可以得到任意形状的截面（见图2-6c）。

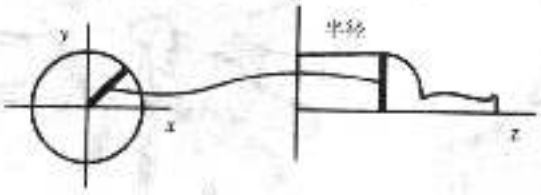
39

现在来考虑控制脊线的形状。可以将一个弯曲的脊线的概念引入到设计中，来产生其截面形状、轮廓线以及脊线受控制的物体，如图2-9所示。

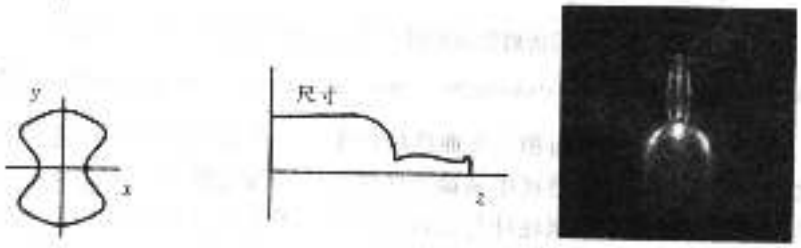
另外，还出现了其他的可能性。图2-7为Snyder称为提手产品表面的例子。在这里，公文包的提手是通过扫掠沿着一条由两个提手曲线的中点确定的路径移动的截面而得到的。类似椭圆的截面的长轴范围由两条相同的曲线控制。一个更复杂的例子是如图2-8所示的涡轮叶片，Snyder把它称为仿射变换表面，因为其脊线现在被仿射变换所取代，此变换由用户定义的曲线来控制。通过沿着z轴挤压一个矩形截面来产生每一个叶片。截面由矩形、以z为函数的三条形状控制曲线来定义，这三条曲线提供了截面在挤压过程中进行变换所需的值。对于z的每一步，截面分别在x轴和y轴方向按比例缩放，在x轴方向平移、旋转、再反向平移，并沿着z轴挤压。



a) 通过扫掠垂直截面产生的旋转物体

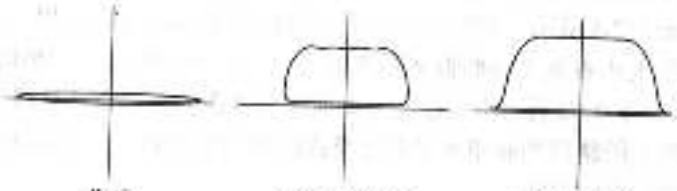


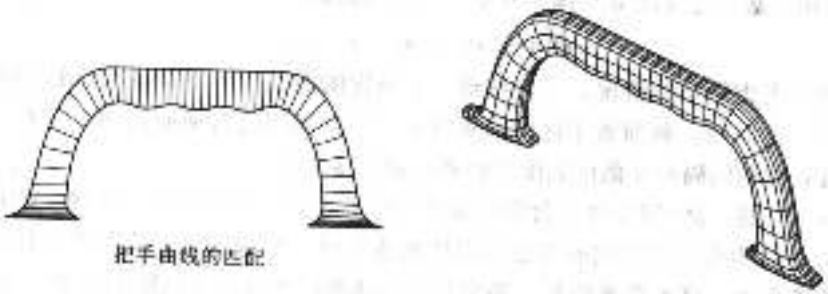
b) 通过扫掠其半径受样条线控制的圆，直到一条直的垂直脊线为止，可以产生相同的实体



c) 非圆形截面

图2-6　直脊线的物体——旋转的实体与截面扫掠





把手曲线的匹配

图2-7　Snyder的把手曲线产品表面

资料来源：J. M. Snyder, *Generative Modelling for Computer Graphics and CAD*, Academic Press, 1992.

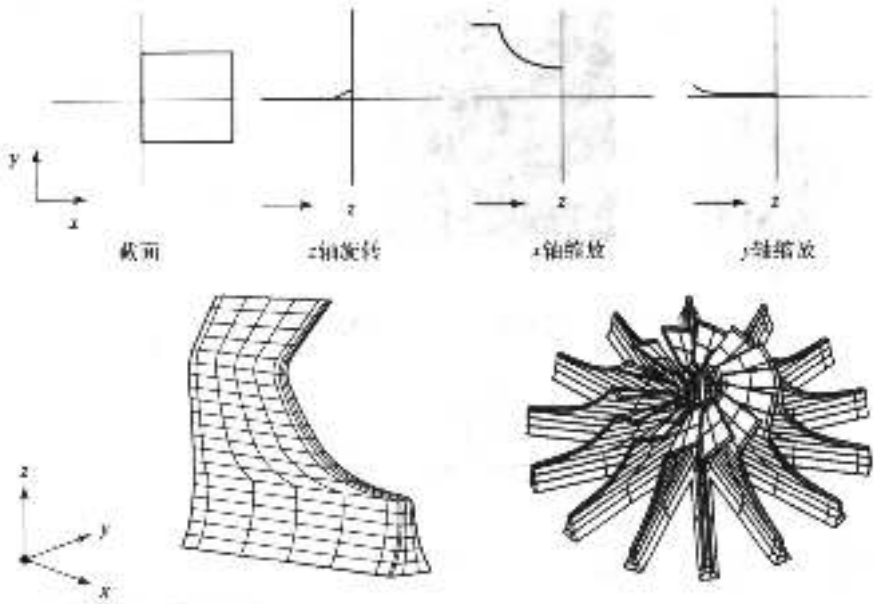


图2-8 Snyder的仿射变换表面，显示了单个涡轮叶片的产生曲线

资料来源：J. M. Snyder, *Generative Modeling for Computer Graphics and CAD*. Academic Press, 1992.

于是，通过一个普通的截面和三条曲线就产生了一个复杂的形状。这个例子很清楚地显示，需要依靠一个能够在三维空间中将最终所需形状可视化的用户或设计者，以便能够定义适当的形状曲线。尽管对于像涡轮叶片这样的例子，看起来这种要求有些过高。但是我们应该记住，这样复杂的形状描述是属于专业工程师的领域，而他们对于使用这样的产生模型来定义形状不会不熟悉。

当把问题推广到弯曲的脊线时会出现一些实际的问题。在用到弯曲的脊线时会立即出现三个问题，这三个问题如图2-9所示。图2-9a为一个从曲线到多边形变换过程中出现的问题。我们看到，多边形基元的大小根据脊线的偏离程度而定。另一个问题是，如何随着脊线的变化来为截面定向（见图2-9b）？最后一个问题是，如何防止截面的自相交（见图2-9c）？很清楚，当任何截面曲线跟踪点的路径的曲率半径超过脊线的曲率半径时，立刻会出现这种情况。现在我们来考察解决这些问题的方法。

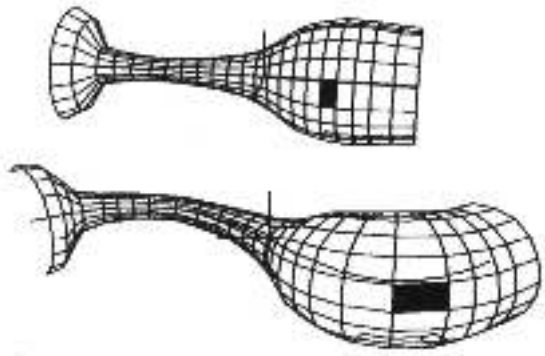
41

考虑一个由参数化定义的立方体，并沿着它扫掠截面。用下面的方程来定义（见3.1节）：

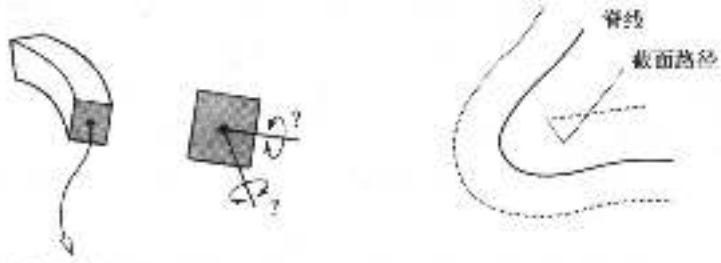
$$Q(u) = au^3 + bu^2 + cu + d$$

现在，如果我们考虑简单的情况，只是移动一个恒定的截面而不是沿着曲线扭曲它，则需要沿着曲线定义一些间隔，截面放于这些间隔点上，且间隔围绕着截面曲线。当有了这些信息后，可以沿着脊线的间隔点和截面周围的间隔点输出多边形。

考虑第一个问题，将u等分将不会给出最好的结果。在特殊情况下，曲线上的点不会以等间隔出现。一个称为弧长参数化的方法可以把曲线等分，但是这种方法并不直接。弧长参数化方法也可能不合适。真正需要的是一种依曲线的曲率把曲线分成间隔的方案。当曲率大时，产生的多边形应增加，以使得当曲线迅速扭曲时可以出现更多的多边形。这样处理的最直接方法是采用曲线细分算法，并继续细分该曲线，直到线性测试值为正时为止（见4.2.3节）。



a) 多边形尺寸的控制可能成为问题



b) 如何随着脊线的变化来为截面定向　　c) 截面路径的自相交

图2-9 在截面扫掠时出现的三个问题

现在来考虑第二个问题。当定义了一组样本点之后，我们需要在每一个点上定义一个参考框架或坐标系。然后，将截面植入到这个坐标系中。通过导入三个相互正交的向量来达到这一目的，这三个向量形成坐标轴。这样做会有许多可能的坐标轴。

通常的情况是采用Frenet框架，该框架由一个原点或样本点P以及三个向量T、N和B组成（见图2-10），T是单位长度的正切向量。

$$T = V/|V|$$

其中，V为曲线的导数。

$$V = 3au^2 + 2bu + c$$

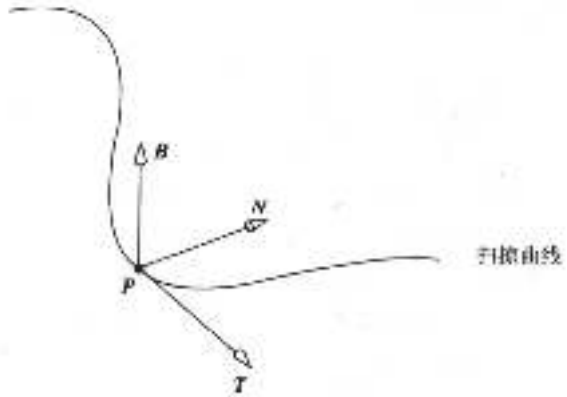


图2-10 在扫掠曲线上样本点P处的Frenet框架

主法向N由下式给出:

$$N = K/|K|$$

其中:

$$K = V \times A \times V/|V|^4$$

A为曲线的二阶导数:

$$A = 6au + 2b$$

最后，B由下式给出:

$$B = T \times N$$

2.1.5 程序化的多边形网格物体——分形物体

在这一小节中，我们将研究一个程序化地产生多边形网格物体的普通的例子。分形几何学是由Benoit Mandelbrot（1977; 1982）创造的一个术语。这个术语用于描述一些自然现象的性质，如海岸。从任一精度水平来观察一条海岸线，例如从显微水平、从可以观察到各个岩石的水平再到“地质”水平，都倾向于显示出相同的弯曲状况。这是一种统计上的自相似。分形几何学对于这种大自然中普遍存在的现象的某些方面及其倾向于自相似的性质提供了一种描述手段。

在三维计算机图形学中，分形技术常用于产生地形图，最容易的技术需要把由三角形或四边形构成的物体的小平面进行细分。应用一种递归划分程序，对每一个小平面划分到所需的深度或所需的精度，这样就会产生出一个有说服力的地形图。在这种情况下，细分意味着取两个顶点之间的边的中点，并沿着此边的法向扰动它。这样做的结果是，将原来的小平面细分为大量的更小的平面。在三维空间中，每一个小平面相对于原始平面来说都有一个随机的方向。物体初始的完整的形状依细分时扰动的大小而保持一定程度的原状，而平面四棱锥将会转换成“Mont Blanc”形的物体。

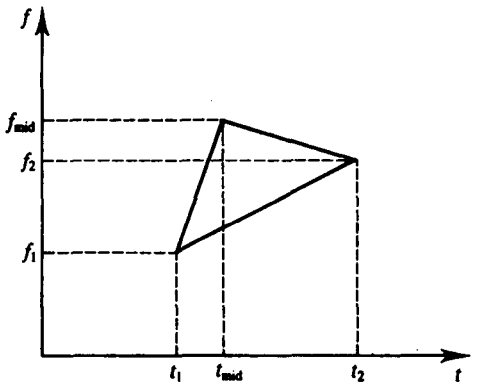
大多数细分算法都基于Fournier等（1982）提出的公式，该公式递归地对一条线段进行划分。这种算法是对另一种数学上更精确但过程更复杂的、由Mandelbrot提出的算法的一种替代。它使用自相似和分形布朗运动的条件期望性质给出随机过程增量的估计。这个过程也是一个高斯过程，描述高斯分布所需的参数是均值（条件期望值）和方差。

42 ~ 44

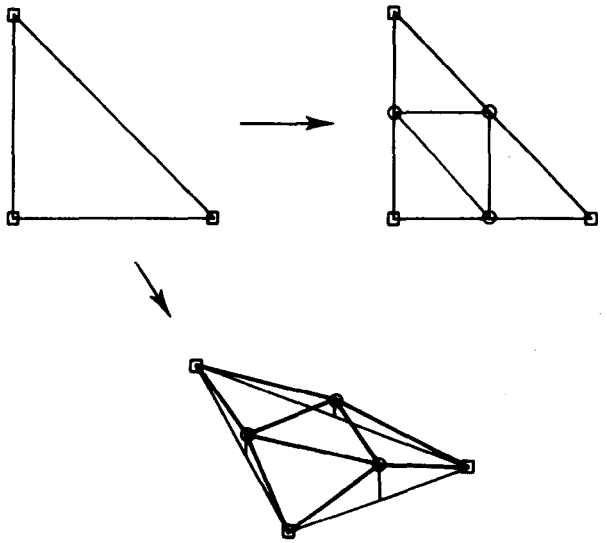
一个过程递归地对一条线段(t_1, f_1)、(t_2, f_2)进行细分，在该线的法向上产生一系列中点的逐级置换（见图2-11a）。

为了将这种过程扩展到三维空间中的三角形或四边形，我们依次处理每一条边，产生对原始小平面的法向（即中点向量）的置换（见图2-11b）。采用这种技术，我们可以先取一个光滑的金字塔，比如说由大的三角形平面构成，再将其转换成一个粗糙的山体。

Fournier对这种方法划分出两类问题——内部一致性和外部一致性。内部一致性需要使所产生的形状保持相同，而不管产生的方向如何，而且边部的细节在进行较大分辨率的重新绘制时要保持相同。为了满足第一个要求，所产生的高斯随机数必须不是点的位置的函数，但对于点本身应是唯一的。每一个点必须有一个不变的点标识符与之相关联。在地形图产生过程中，这个问题可以通过对每一个点给出一个键值以对高斯随机数进行索引的方法来解决。可以采用哈希函数将一条线段上的端点的两个键值映射到中点的键值。内部一致性的级别指在已知的细分水平下，相同的随机数必须总是按照相同的顺序产生。



a) 线段细分



b) 三角形细分

图2-11 多边形网格物体——分形地形图的程序化产生的一个例子

外部一致性较难保持。在三角形网格中，每一个三角形都与其他三角形共享每一条边。这样的话，不同的连接三角形的相应点必须产生相同的随机置换。这个问题已经通过每一个点的键值和哈希函数得到了解决。但是还存在另一个问题，即置换的方向问题。

如果置换是沿着所考虑的多边形表面法向进行，则相邻的具有不同法向的多边形（根据定义，这种情况经常发生）将会把中点放到不同的位置上，这样会产生裂口。一个解决方法是在所有包含它的多边形的平均法向量处进行中点置换，但是，这样的问题在递归的每一个层次都会出现，因此执行起来也是非常复杂的。这种技术还会产生一种不太令人满意的地平线，这是因为没有把置换限定在一个方向。当原始多边形内部的所有点的置换都在原始多边形平面的法向方向进行时可以获得一个较好的地平线。这种廉价的方法解决了不同表面法向和由其产生的裂口的所有问题。现在，不必在所有递归层次上建立表面法向，而且由于这个原因算法简单了很多。

还有两点值得一提，首先注意到在不计算顶点法向时，多边形的明暗处理恒定，多边形之间的不连续也不必进行光滑。其次，考虑颜色。通常的颜色系采用一种基于高度的映射。具体来说，给中点赋的颜色值是其端点的颜色之一。所选择的颜色由布尔随机数决定，而此值由中点的键值来索引。为了保持一致性，必须按这种方式访问，这种方式对于颜色的重要性与对于位置的重要性是一样的。

2.2 物体的构造实体几何表示

我们把前面论述的方法（多边形网格法）归类为机器表示法，它也经常被作为用户表示
45 法。CSG方法很大程度上也是一种用户表示法，它需要特殊的绘制技术，或者需要在表示之
~ 前转换成多边形网格模型。这是一种高层次的表示，它既可以作为一种形状的表示，也可以
46 作为物体如何构造的一种记录。在这种表示中，“形状的逻辑”是最终的形状如何以基元形状的某种结合来表示。设计者通过使用三维构造块（building block）以及在这些构造块可以组合的各种方式中进行某种选择来建立起一个形状。这种表示的高层次特性对于设计者意味着某种程度的负担。尽管在图2-14中部件之间的逻辑事后看来是很明显的，但是在用这种方法进行复杂的机器部件的设计时还需要专业人员来进行。

采用这类表示方法的动机是希望能够实现一种实体建模的交互模式。其论点是，物体通常是由部件组成的，这些部件最终会通过铸造、加工和挤压来制成。在CAD程序中，还可以通过等价的操作把简单的基本物体（称为几何基元）进行结合来构建这类部件。这些基元为球、圆锥体、圆柱体或矩形实体，可以通过（三维）布尔集合操作、线性变换来对这些基元进行组合。物体表示方法被作为特征树（attributed tree）进行存储。叶子上为简单的基元，结点存储操作符或线性变换。表示不仅定义物体的形状，而且给出了其建模的历史，于是，物体的创建及其表示就成为了一件事。通过添加基元并且使它们与现有的基元进行结合构建物体。可以向当前的形状上添加形状，也可以从当前的形状中减去一个形状（产生孔）。例如，将穿过一个矩形实体的孔的直径扩大就意味着一种简单的改变，即定义孔的圆柱体基元的半径只是简单地增加。这与多边形网格表示方法正好相反，在多边形网格表示方法中，这种操作很明显不是简单的操作。尽管在层次化的结构中圆柱体表面的组成多边形很容易访问，但要产生一组新的多边形就意味着要进行产生原始多边形所用到的所有建模步骤。此外，还要考虑这样的事实，即为了保持相同的精确度，就必须使用更多的多边形。

布尔集合操作符既被用作一种表示形式也被用作一种用户界面技术。用户用布尔集合操作符来定义基元实体并进行基元实体的组合。物体的表示是一种用户交互操作的反映或记录。于是可以说，建模信息和表示是不能分离的，就像从输入设备的低层次的信息中导出一个表示的情况那样。在CSG中，低层次的信息已经是以体基元形式存在，而建模活动就成为表示活动。我们将用一个例子来说明这一点。

图2-12给出了实体之间可能的布尔操作。图2-12a表示两个实体的并。如果把物体看作是点集，则并操作包围了在两个原物体之内的所有点。第二个例子（见图2-12b）显示了一个差或减操作的结果。减操作符把第二个物体中被第一个物体所包含的内容去掉。在本例
47 中，定义了一个圆柱体，并将其从由图2-12a产生的物体中减掉。最后一个例子，即图2-12c为一个相交操作。这里，定义了一个实体，它是由一个圆柱体和一个矩形实体进行并操作

得到的（与图2-12a中采用相同的基元，并进行相同的操作）。然后，将这个实体与图2-12b中产生的物体进行相交操作。相交操作产生同时被两个物体所包含的点的集合。从这个例子中看到，这个方法的一个非常显著的特性是基元不仅被用作构建模型而且还被用来取出材料。

图2-13为一个CSG表示，它表示了一个简单物体的构造。在这棵表示树的叶子上出现了三个原始实体：两个长方体和一个圆柱体。用一个并操作将两个长方体合并。通过定义一个圆柱体并将其从两个组装好的长方体中减掉而得一个“钻出的”孔。这样的话，在树叶中必须存储的信息就是基元的名称及其尺寸。一个结点必须含有操作符的名称以及将由操作符结合的子结点之间的空间关系。

在以后的例子中我们会进一步看到布尔操作的威力。在第一个例子中（图2-14a），两个分别建立的部件用并操作符再用减操作符进行了结合，产生了希望的构造。第二个例子（图2-14b）展示了一个只是通过圆柱的并而构建的一个复杂的物体，然后再用它通过减操作产生一个复杂的机罩。

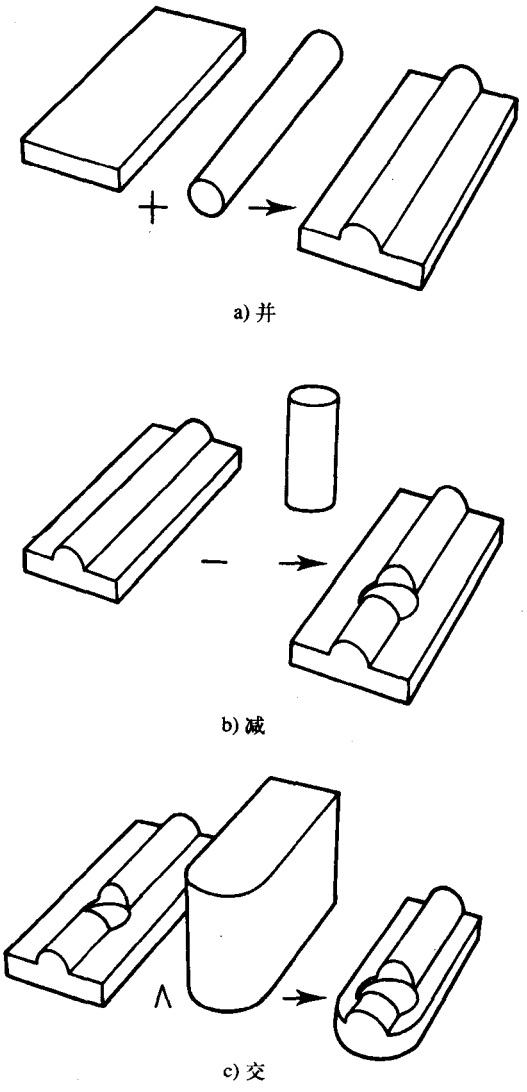


图2-12 在CSG建模中实体之间的布尔操作

尽管CSG表示方法有很多显著的优点，但是确实也有缺点。一个实际的问题是产生模型的绘制图像所需的计算时间。更严重的缺陷是该方法对于创建和修改一个实体可用的操作有很多限制。布尔操作是全局性的，影响到整个实体。而局部操作（比如对于一个复杂物体的一个侧面的细节修改）则不能通过使用集合操作来实现。在很多设计出来的物体当中，所需的一个重要的局部修改是弯曲表面。例如，考虑与一个平的基础底座相连接的一个圆柱体的端面。通常情况下，为了实际制造或者美学原因，在截面处不是用直角而是用圆弧。绕另一条曲线被扫掠的圆弧半径不能用简单的CSG系统表示。这个事实导致了很多实体建模器采用隐含的边界表示方法。顺便指出，布尔操作不能被结合到边界表示系统当中，而且没有原因。例如，很多系统融入了布尔操作，但是会使用边界表示方法来表示物体。在这两种表示之间的一种折中的方法导致了一个大约延续了15年的争论。最后请注意，CSG表示方法是一种体表示方法，所表示的是物体所占据的空间（即体积）而不是物体的表面。

48 ~ 49

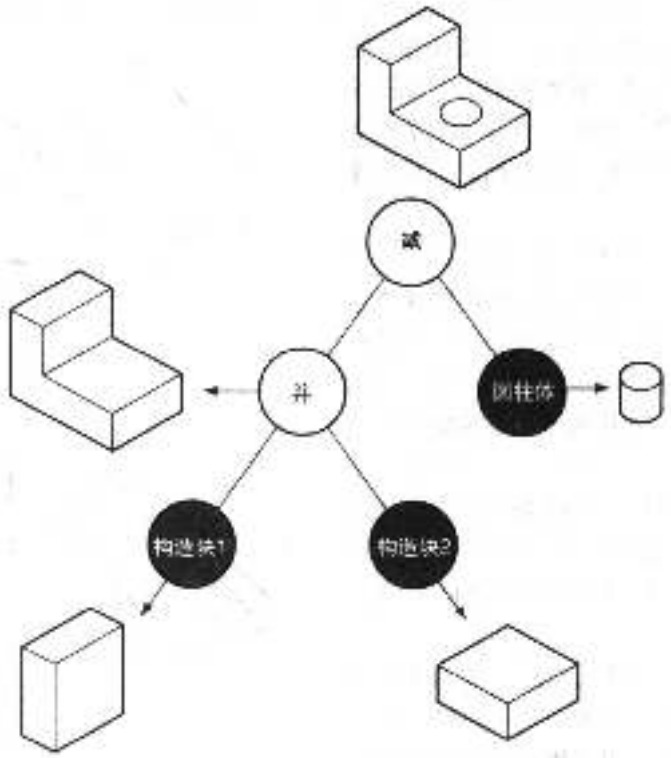


图2-13 一棵反映由三个从元构建的简单物体的CSG树

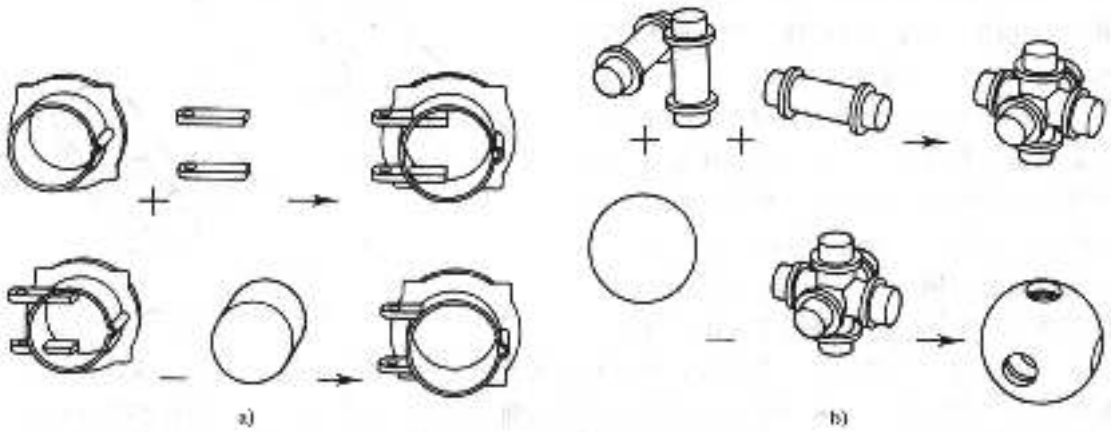


图2-14 由简单物体和布尔操作产生的几何上复杂的物体的例子

50

2.3 物体表示的空间细分技术

空间细分技术考虑整个物体空间，并且根据物体的空间占用程度来标记空间中的每一个点。而CSG方法使用大量的体积元素或几何基元。空间细分技术是基于称为体素的一个立方体元素。体素是立体的元素或称为基元，它是在表示当中所使用的最小的立方体。我们可以

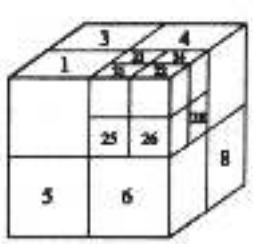
把整个世界的空间细分成规则的或者立体的体素，然后根据它是处于物体当中还是来自空的空间，对每一个体素进行标记。很明显，从存储空间占用的角度来看，这种方法非常昂贵。正因为此，体素表示通常情况下不是令人喜爱的主流的方法，但是人们使用它或者是因为原始的数据可能已经是以这种形式存储的，或者原始数据最容易被转换成这种表示，例如，在医学图像中使用的情况；或者是由于在算法上有这样的需要。例如，在体素空间中进行光线跟踪与传统的光线跟踪方法相比有很明显的优势。这是一个规定物体表示方法性质的一个算法技术的例子。这里不是要提这样的问题："这条光线是否与场景中的任何物体相交?"因为要回答这样的问题需要对每一个物体进行非常复杂的相交测试。而代之以提出这样的问题："当我们跟踪一条通过体素空间的光线时，会碰到什么样的物体?"解这个问题就不需要穷尽地在初始的数据结构当中搜索可能的相交，因此是一个快得多的测试策略。

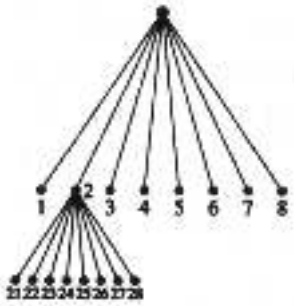
另一个例子是绘制CSG模型（见第4.3节），如果使用传统的技术的话，这种方法不是很直接。一个解决方法是把CSG表示转换成一个中间的含有体素的数据结构，然后从这个数据结构进行绘制。可以将体素看作是一种中间的表示，在医学图像当中这种表示最普遍，这时这种方法把二维原始数据与三维结构可视化地连接起来。另一种选择方案是原始数据本身可以是体素。许多三维物理现象的数学模型（如流体动力学）都是这种情况。

体素标记的主要问题是巨大的存储空间开销和精确度之间的一种折中。例如，考虑用正方形的像素表示一个二维空间中的圆。此像素的尺寸和精确度之比的值是很清楚的。同样的概念可以扩展到用体素来表示一个球，但是这时的开销要根据精确度和立方体的半径而定。因此，这样的方案只能用在那些对优越性的要求远远大于对其开销的考虑的情况。一个减少开销的方法是将结构组织附加到基本的体素标记方案中去。

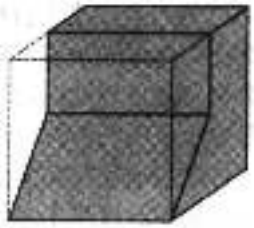
组织体素数据的常见方法是使用八叉树——这是一种层次的数据结构，它描述了场景中的物体是如何在场景所占用的三维空间中分布的。其基本的思想如图2-15所示。在图2-15a中，一个立方空间可以进行递归细分，这种细分使得空间中的任意立方体区域都用一个数字来标记。这样的细分可以进行到任何所希望的精确度水平。图2-15b展示了一个放在这个空间中的物体。而图2-15c为细分过程和与之相关联的八叉树，该八叉树根据空间是否被占据或为空来 51
标记立方体区域。

实际上有两种对于场景的分解的八叉树可以用于表示场景。首先，前面所描述的八叉树可以被它自身用作场景中物体的完整表示。物体所占据的单元的集合构成了该物体的表示。但是对于复杂的场景，需要将其所占据的空间细分成非常大量的小单元以便得到高分辨率的结果，这个技术需要巨大的数据存储空间。一个通用的替换方法是使用一种标准的数据结构进行物体的表示，并使用八叉树作为场景中物体分布的表示。在这种情况下，代表一个被占据区域的树的终端结点将由一个指向任何包含在该区域中的物体（或者物体的一部分）的数据结构的指针来表示。图2-16显示了在二维情况下这种方法的可能性。在这里一旦碰到一个区域只与一个物体相交的情况就停止进行区域的细分。由终端结点表示的区域不一定完全被与这个区域相关联的物体所占据。在区域中物体的形状将由其数据结构表示来描述。在场景的表面模型表示的情况下，"物体"可能是多边形或者是曲面片。一般情况下，由终端结点表示的被占区域可能与几个多边形相交，可以由指向物体数据结构的指针列表来表示。因此，与前面已经描述的其他技术不同，八叉树一般情况下不是自包含的表示方法，而是一个合成策略的组成部分。

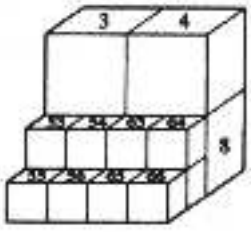


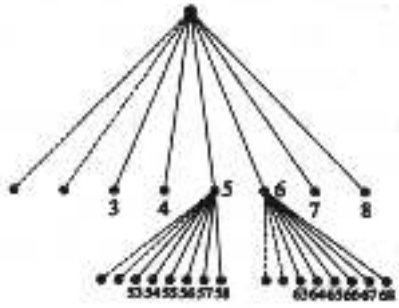


a) 立方体空间和标记策略，对于细分的二层八叉树



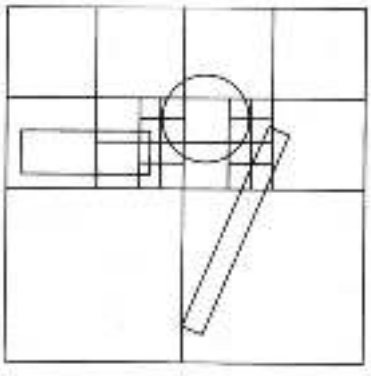
b) 在空间中嵌入的物体





c) 二层细分的物体的表示

图2-15 八叉树表示



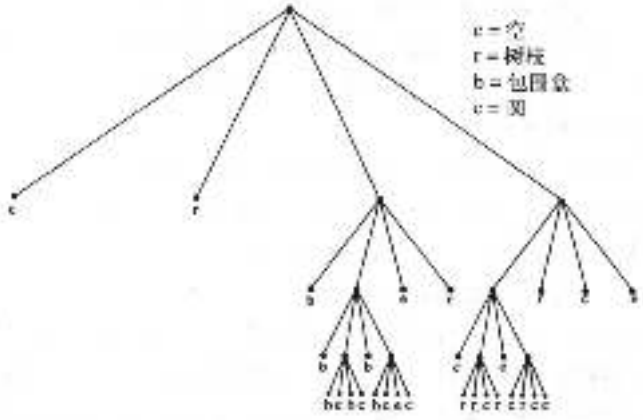


图2-16 一个二维场景的四叉树表示。场景细分到一个单元最多包含一个物体。含有物体的终端结点单元可以用指向代表该物体的数据结构的指针来表示

2.3.1 八叉树和多边形

正如我们已经指出的那样，在计算机图形学中八叉树的最广泛的用途并不是指出一个关于体素数据的数据结构，而是将一个含有很多物体（其中每一个物体都是由很多多边形组成

的）的场景组成一种空间占用的结构。我们并不是用体素来描述物体，而是把多边形占据的矩形空间看成是由体素空间所表示的一些实体。当我们考虑绘制的时候，在图2-16的场景中的矩形区域内按照一定的精细度选择场景的某些部分。例如，可以在一个八叉树的叶结点中包含一组物体、单一物体、物体的某些部分甚至单个多边形。这可以使绘制和很多绘制方法尤其是光线跟踪算法中的很多计算大大加速。

现在我们以光线跟踪作为一个特殊的例子。在原来的光线跟踪算法中，固有的高成本在于对相交的测试。当我们对一条光线穿过场景进行跟踪时，必须求出场景中任意物体与光线的相撞（以及该点所处位置）。当对每一条光线是否与场景中所有的物体相交进行测试时，相当于对物体中的每一个多边形进行测试，对于复杂度合理的场景来说，所需的绘制时间将长得令人无法接受。如果把场景分解成一个八叉树来表示，则跟踪光线意味着使用从一个体素到另一个体素的增量算法进行跟踪。每一个体素都含有指向它所在的多边形的指针，而测试就是求光线与这些多边形的指针的关系。这时可能的相交就从n减少到m，其中：

$n = \sum_{物体}$ 物体的多边形个数；

m是八叉树叶子所包含的候选多边形的数量。

然而把场景分解为八叉树也是很繁琐的操作，因此必须有良好的控制。它包括对每一个多边形寻找其最小最大坐标（其边界包围盒的坐标），并把这一坐标作为分解中的一个实体。可以用以下两种因素来控制分解：

1）每个结点的候选多边形的最小数目。值越小，分解的数目就越大，进入体素的光线的相交测试就越少。对于整个绘制过程，每个体素的相交测试的大概总数由下式给出：

进入体素的光线数 × (0.5 × 体素中多边形的个数)

假设在找到一个相交之前，对于平均50%的候选多边形进行光线测试。

2）八叉树的最大深度。深度越大，分解越多，叶结点上的候选多边形越少。而且，因为在每一个层次上，体素的大小减小到$\frac{1}{8}$，所以对于任意给定的绘制，将会有更少的光线进入体素。

一般情况下分解的程度不应该太高。否则的话，在相交计算时所节省的成本就会被在分解的空间中跟踪体素——光线相交所耗费的高成本所抵消。实验表明，对于均匀地分布在空间中的物体，一般情况下对上面的两个因素取默认值8会给出好的结果。经常会出现的情况是，不满足这种条件时也能绘制一些场景。图2-17显示的一个例子是，具有很多多边形的几个物体分布在一个房间里，房间的体积远远大于被物体所占据的空间。在这种情况下，八叉树细分将会进行到一个很高的细分深度，以划分几乎是空的空间。

52
~
54

2.3.2 BSP树

一个替代八叉树的表示方法是BSP，或称二叉空间分割树（binary space partitioning tree）。在BSP树中，每一个非终端结点都代表一个把空间分成两部分的分割平面。表明此差别的一个二维模拟图如图2-18所示。BSP树并不是直接的物体表示(尽管从某种意义上来讲它确实是)。其实，它是为了特定的目的进行空间分割的一种方法，最通常的情况是用于隐藏面消除。正因为如此，如果不同时讨论HSR（hidden surface removal）的话（见第6章），仅讨论BSP是困难的，甚至没有什么意义。



图2-17 一个由几个具有很多多边形的物体组成的场景，与空内体积相比，物体的体积很小

在计算机图形学的场景中，可以利用的分割平面的性质有：

- 在一个平面的一侧，任一物体都不能截断另一侧上的任何物体。
- 已知场景空间中的一个观察点，与观察者处于同一侧的物体都比另一侧上的任何物体近。

当BSP树用于表示把空间划分成立方体单元时，它没有比八叉树所用的直接数据结构更明显的优点，它只是以不同的方式对相同的信息进行编码。但是，上面并没有提到要细分成立
55 方体单元。事实上，BSP树的思想最早是由Fuchs（1980）引入的，在其提出的方法中，那些用于把空间进行细分的平面可以是任意方向的。我们将在隐藏面消除部分再次讨论BSP树（见第6章）。

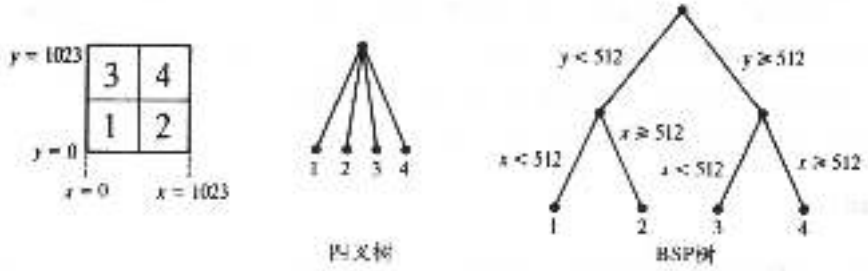


图2-18 四叉树和BSP树表示一个一层细分的二维区域

2.3.3 创建体素实体

体素实体的主要用途之一是在医学图像中的体绘制。在这种应用中，源数据由一组平行平面的密度信息组成，这种信息来自一个物体的某些部分的连续的截面。在这种平面中，比

如说一个像素代表该像素物理上所对应的该物体某部位的X射线的吸收量。问题在于如何将这样大量的平面二维信息转换成所绘制物体的三维信息。解决这个问题的最直接的方法是将一组平面转换成一组体素。在两个连续的平面中相对应的像素被用来形成体素的顶面和底面，并执行某些操作从两个像素值得到一个体素值。体素表示被用作所收集的原始数据（二维的）和三维可视化所需的数据之间的中间产物。从原始数据的选择，到将其转换成体素表示，再到体素数据的绘制，整个过程将在第13章中进行叙述。

通过激光测距仪收集的轮廓线可以转换成体素表示，而不是转换成多边形网格表示。但是，与使用蒙皮算法相比，这可能会使精确度下降。

2.4 用隐函数表示物体

正像我们已经指出的那样，用隐函数公式来表示物体的应用范围被严格地限制在某些像球这样的物体上。然而这种表示也确实在表示称为限定体的那些“算术”物体时起了主要的作用。这些限定体在计算机图形学中的很多不同场合被用作一种复杂性限定的手段。 56

用隐含方法所建立的表示是通过隐含地把物体定义为部件的思想来表示物体的（在这里我们采用了“部件”而不是“基元”这样的术语，这是因为物体本身并不是由一些球简单地接触在一起而构成的，而是从这样一组简单物体导出的表面）。

隐函数是这样一些表面，这些表面是通过一些基元向其邻近的物体施加作用而形成的。例如，考虑一对图2-19中的点热源。可以把它们周围的温度定义为场函数，对于每一个孤立的热源，我们把等温线定义为以每个热源为中心的球面。当把这两个热源靠近到相互产生影响时，我们就定义一个组合的全局的标量场，每一个热源的场与另一个热源的场相结合，形成一个如图2-19所示的合成的等温线。由于一些基元的组合效果而产生的标量场在计算机图形学中用于定义一个建模表面。通常，我们把场中的等值面看作是一个体的边界，该体即我们希望建模的物体。因此，在任何隐函数建模系统中都有下面一些元素：

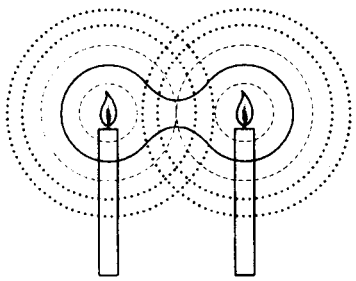


图2-19 两个热源周围的一个等值面（实线）

- 一个产生器或基元，在产生器的方位上对于所有点P可以定义一个距离函数$d(P)$。
- 一个“势”函数$f(d(P))$，它对点P返回一个与产生器之间的距离的标量值。与产生器相关的可以是一个影响区域，在这个区域之外，产生器没有影响。对于点产生器，这个区域通常是一个球。一个势函数的例子为：

$$f(P)=\left(1-\frac{d^2}{R^2}\right)^2 \qquad d<R$$

其中，d为点P到产生器的距离，R为影响半径。

- 一个标量场$F(P)$，它确定各个产生器的势函数的组合效果。这表示存在一种融合的方法，这种方法的最简单情况是“加”。我们通过先求出每一个产生器在点P处的作用，然后将它们的作用合起来来求标量场。 57
- 一个标量场的等值面，该面用于表示所建模的物体的物理表面。

一个例子（见图2-20，彩色插图）阐明了这一点。该图的左侧为Salvador Dali仿制品，它

是一个由放置于空间中的点发生器（如右图所示）形成的等值面。每一个球的半径都正比于每一个发生器的影响半径。紫色的球代表负产生器，用于“分割”出模型中的凹面（尽管可以仅仅用正产生器来形成凹面，但是用负产生器更方便一些，因为这样需要少得多的球）。这个例子表明，这种方法有建模有机物形状的潜力。

可以通过显示或者设计产生物体的点的动作来实现可分解的物体动画。当在动画中采用隐函数表示时，出现的问题是在移动的产生器组和由于这种移动而产生的分解之间没有一个好的直接的联系。当然，这种一般性问题是所有建模技术都会遇到的几何定义和分解方法同为一个或一种表示。

除了这种一般性问题之外，当产生器互相之间移动并保持使用相同的弯曲方法时，可能还会出现不希望的弯曲和不希望的分离。

在动画上下文中，采用隐函数表示的一个很突出的优点是，可以很容易地从一种简单的内部-外部函数来得到碰撞检测。不考虑所建模的表面的复杂性，一个标量值定义等值面，而依$F(P)$是小于还是大于这个值来确定P点是处于物体之内还是在其外部。

2.5 场景管理和物体表示

由于像计算机游戏和虚拟现实等应用对高质量的实时计算机图形学的需求持继增长，所以关于如何有效地管理场景的问题就变得越来越重要了。这意味着表示的形式必须可以扩展到物体的聚集，换句话说，必须把场景本身也看作是一个物体。这一般来讲要求使用层次结构或树状结构，如用BSP树来表示从场景到物体再到子物体的层次结构。由于绘制方法正在逐渐迁移到实时应用中，所以消隐和隐藏面消除的效率就变得与有效地绘制复杂场景同样重要了。随着PC机所用的3D图形加速板的出现，我们正在看到一种趋势，即基本的单个物体的绘制由硬件来处理，而评估哪些物体可见的计算则由软件来完成（我们将在第5章和第6章讨论消隐和隐藏面消除）。下面讨论对于复杂场景中的物体来说一种同等重要的方法，称为细化
58 水平（Level of Detail，LOD）。

多边形网格优化

正如我们已经讨论的那样，多边形网格模型在计算机图形学中已经是一个事实上的标准表示形式。但是，这种方法也有很显著的缺点，最明显的是在对复杂物体进行高质量的绘制时合成这个物体所需的细化程度或多边形的个数是非常大的，如果这个物体以不同的观察距离绘制到屏幕上，则可能会出现这样的情况，即流水线必须处理成千上万个多边形，而它们只能投影到屏幕上的几个像素点上。随着所投影的多边形的尺寸的减小，所需的多边形将变得很多。而在实时应用中，这种情况是不能容忍的。出现单位物体的高多边形计数可能是因为物体复杂，或者是因为建模系统本身的特性。众所周知，激光扫描仪和从像步进立方体算法（这种方法把体素转换成多边形）这样的程序产生的输出会产生非常大的多边形计数。用这样的设备进行绘制时产生的效果几乎总是与其他只用很少的表面进行绘制时的结果有区别。

早在1976年，3D计算机图形学的先驱之一James H. Clark就写道：

> 如果对一个只是在光栅显示设备上占据20光栅单位的物体用500个多边形来描述它是没有意义的。例如，当我们从非常远的距离来观察一个人体时，可能只需用几个点来表示眼睛，或者也许只用一个方块来表示头，而不考虑对眼睛的表示……这

些问题还没有一个统一的结论。

Clark是否认识到在他写了这些话之后没有几年，用500 000个多边形表示的物体也变得很平常了，而且对于复杂场景可能会用成千上万个多边形？

现有的系统倾向于以一种专门的方式来处理这个问题。例如，很多廉价的虚拟现实系统采用一种在表面细节水平上切换的二层或三层表示方法。比如，一部电话上的按钮上的数字，随着观察者靠近它而出现的表示上的变化。这样做会产生随着细节的切换而出现的恼人的视觉上混乱。现在正在提出更成熟的方法。最近，在这一领域中发表的文章也大量增加。

因此，网格优化似乎是必需的，而且问题也不会随着未来的工作站所增加的对多边形的处理能力而消失。目前我们所处的情况是，主要的虚拟现实平台会产生从视觉上来说不适当的结果，这种情况即使在相当简单的场景中也会出现。我们不仅必须看到在处理对这样的场景的图像合成时的不足之处，同时还要能够处理可能需要无数个多边形的真实世界复杂的场景。那些所谓的“浸入式”的虚拟现实的应用不可能达到可接受的程度，除非我们能够克服复杂性的问题。当前的硬件还远远不能将实时的复杂场景处理成单个场景可达到的那种质量水平。 59

对于此问题的一个较明显的解决方案是先产生一个最终精确程度的多边形网格，然后再把这个表示扩展到一组更粗的表示水平。场景在一个经选择的适当的细节水平上被绘制。在计算机图形学中，不断地有人提出采用这种原则的算法。一个可以以任意细节水平实现多边形网格的方法的例子是双三次参数曲面片方法（见4.2.2节）。在这个方法中，我们选择一个曲面片表示，然后把它转换成多边形表示。与此同时，我们还可以很容易地控制每一个曲面片所产生的多边形的个数，并且将其与局部表面的曲率相关联。在曲面片绘制中确实是这样做的，绘制时，使用一个几何判据控制细分的程度，并且会产生一个不带几何走样的图像（即可见的多边形边的轮廓）。我们为这一方法付出的代价是，在一开始就需获得对曲面片的描述所遇到的困难。但是，在任何情况下，我们都可以建立原始的曲面片表示，并在离线时建立一个多边形网格表示的金字塔。

存储一个“细节金字塔”并以适当的细节水平来访问它的思想被应用到很多应用领域。例如，考虑mip映射的例子（见第8章），这时，纹理图以细节层次结构存储，当将纹理图投影到屏幕上时选择精细的细节层次图是很大的。当将此图只投影到一个像素上时，则选择一个单像素纹理贴图，即最详尽纹理图的平均值。在这个方法中，还要小心地处理当投影从一个细节水平向另一个细节水平跳跃时出现的问题，并通过在两个纹理映射之间进行插值来获得对细节的连续水平的近似。

当前这些方法上的差异就意味着这一领域相对较新。由体素集导出的三角形网格方法是一种直接且简单的方法，该方法在1992年由Schroeder等人提出。在这个方法中考察一个表面上的每一个顶点。通过考察那些对顶点有贡献或称分享此顶点的三角形，可以列举出一些判据，用于确定是否可以将这些三角形并入到除了正在讨论的顶点之外的一个顶点中。例如，可以通过检测分享此顶点的三角形的表面法向的变化来提出“减少表面曲率低的三角形的个数”这样的论据。另一种可选的方法是，可以求出从顶点到穿过所有其他分享三角形的顶点的（均值）平面间的距离（见图2-21）。这是一种局部的方法，只考虑那些在几何学上与它紧密相连的环境中的顶点。

一个更新的方法是Hoppe（1996）提出的，下面讨论这个方法。Hoppe对网格优化的问题

和优点进行了非常优秀的分类，列表如下：

- 网格简化。将多边形减少到满足所要求的质量的适当水平（当然，这还依赖于物体在屏幕上的最大投影尺寸）。
- 细节近似水平。适合于观察距离所用的水平。从这个观点来考虑，Hoppe又补充道：由
60 于LOD网格之间的瞬间切换可能导致可察觉的“跳跃”，所以人们希望能够建立不同分辨率下网格之间的平滑的视觉转换即地形。
- 渐进式的变换。这是在因特网上用于变换二维图像常用的渐进式变换的一种三维等价变换，即可以在接收端变换并绘制连续的LOD近似图。
- 网格压缩。与二维图像金字塔类似。我们不仅可以考虑减少多边形的数量，而且还可考虑使任何LOD近似图所占据的空间最少化。正如在二维图像中所做的那样，这样做是很重要的，因为一个LOD层次结构所占据的存储空间比一个以其最高的细节水平存储的模型所占用的空间要大得多。
- 选择性改进。可以以上下文有关的方式使用LOD表示。Hoppe给出了一个用户飞过地形图的例子，其中，地形图网格只需要在靠近观察者的地方完全细化。

在谈到网格压缩时，Hoppe选用了一个“金字塔”方法，并把它与最粗级别的细节近似构造存储在一起，其所需的信息从一个较低的细节水平上升到一个较高的细节水平。为了进行从一个较低的水平向一个较高水平的变换，存储并使用反向的变换，即从最高向最低的构造层次变换。这是以一种顶点分离的形式进行的，顶点分离是一种将一个附加的顶点加入到较低的网格中以获得层次中细节较高的网格的操作。尽管Hoppe原来考虑了三个网格之间的变化，即一个边倒塌、一个边分离及一个边交换，但是他发现边的倒塌对于简化网格就足够了。

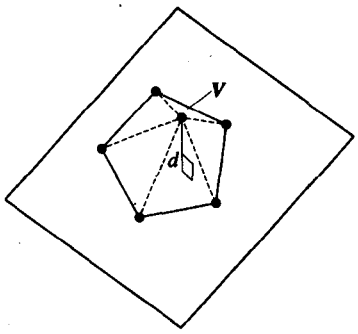


图2-21 一个简单的顶点消除判据。删除V？测量从V到穿过分享V的三角形的（均值）平面的距离d

总的方案如图2-22a所示，该图显示了一个细节金字塔式流程，该金字塔可以由一系列边的倒塌变换离线构建。在这个过程中，通过反复的边倒塌变换，将从原始的网格M_n产生出最终的或最粗的网格M_0。可以将整个金字塔按M_0存储，并同时存储从M_0到任一较精细层次M_i的变换所需的信息，即产生网格压缩所需的信息。这种内部层次的变换是边倒塌的逆，也是顶点分离所需的信息。Hoppe给出了一个例子，该例中含有
61 一个具有13 546个小平面的物体，用6698次边倒塌变换将该物体简化成具有150个小平面的M_0。这样，原始数据按M_0存储，再附加6698次顶点分离的记录。顶点分离记录本身显现出冗余，可以用传统的数据压缩技术对其进行压缩。

图2-22b显示了在两个连续细节水平间一个边的倒塌过程，符号的含义如下：V_{f_1}和V_{f_2}是较精细的网格中的两个顶点，这两个顶点将被倒塌为一个较粗网格中的顶点V_C，其中：

$$V_c \in \left\{ V_{f_1}, V_{f_2}, \frac{V_{f_1} + V_{f_2}}{2} \right\}$$

从图中可以看到，这种操作说明，两个平面f_1和f_2可以倒塌为新的边。

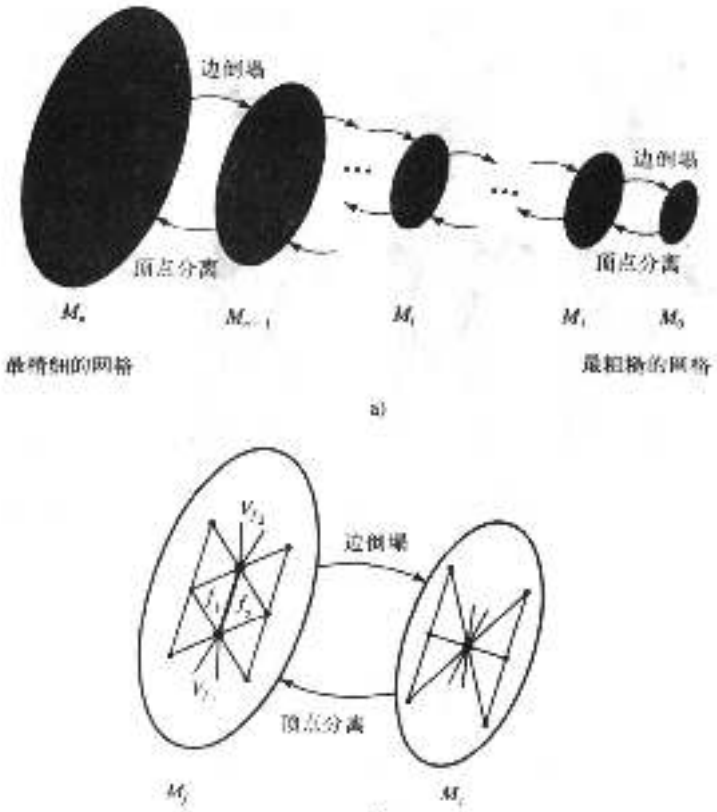


图2-22 Hoppe(1996)的基于边倒塌变换的渐进式网格方案

Hoppe用一个弯曲参数α定义了任意两个细节水平之间的连系数（continuum）。如果定义：

$$d = \frac{V_{s1} + V_{s2}}{2}$$

则可以通过使边闪烁由弯曲参数来控制从而得到两个细节水平之间的地形连系数，表示为：

$$V_{s1} := V_{s1} + \alpha d \text{ 和 } V_{s2} := V_{s2} - \alpha d$$

纹理坐标可以像与某一顶点相关联的标量属性（如颜色）一样进行插值得到。

余下来的问题是：在从M_i到M_{i-1}导出倒塌时如何选择边？这个问题可以通过采用一种简单的启发式方法来解决，也可以通过一种更严密的方法解决，即测量一个特定的近似表示与一个原始网格中的样本之间的差别来获得。一个可以用于对倒塌的边进行排序的简单量度是：

$$\frac{|V_{s1} - V_{s2}|}{|N_{s1} \cdot N_{s2}|}$$

即，边的长度除以顶点法向量的点积。就其本身来说，此量度将很有用。但是，如果连续地使用它的话，则网格可能会突然"倒塌"。一个更合理的选择边的方法是强制性方法。图2-23为一个使用这种技术的例子。

Hoppe把这个例子处理为能量函数最小化的问题，对于网格M关于一个X点集进行最优化，而X点集是网格M_0的顶点以及那些从其表面随机选取的点（这一项是可选的）（尽管这是一个很长的处理过程，但是它当然也只在一个离线的预处理过程中执行一次）。要进行最小化求值

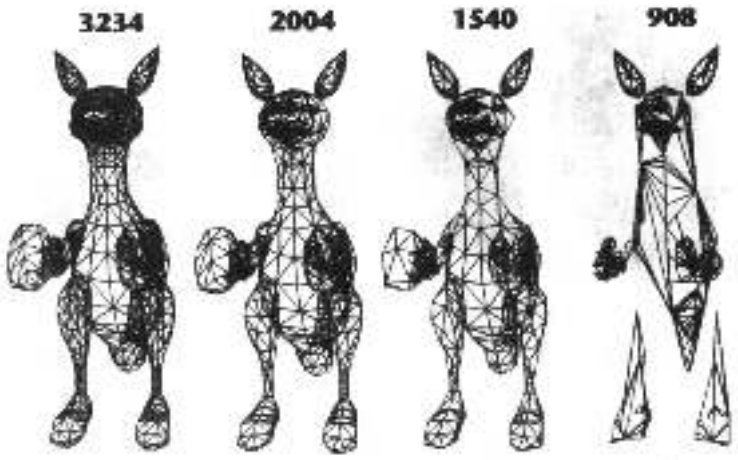


图2-23 应用文中所讲述的简单边消除判据的计算结果——此模型最后断开了

的能量函数为：

$$E(M) = E_{dist}(M) + E_{spring}(M)$$

其中：

63

$$E_{dist} = \sum_i d^2(x_i, M)$$

为从点X到网格的距离的平方和，当去掉一个顶点时该项将会增加。

$$E_{spring}(M) = \sum \kappa \left\| v_i - v_k \right\|^2$$

是一个跳跃（spring）能量项，用于辅助优化计算。这一项等价于在每一条边上放置一个停顿长度为零，跳跃常数为κ的跳跃。

Hoppe通过将所有（合法的）边倒塌变换放在一个优先权队列中来进行优化排序，而每一个变换的优先权是其估计的能量代价ΔE。在每次迭代中，处于队列前面的变换（具有最小ΔE值）被执行，其后，再对这个变换邻近的边的优先权进行重新计算。只有当变换本身不改变网格的拓扑结构时，边倒塌变换才是合法的。例如，如果V_1和V_2是边界顶点，则边(V_1, V_2)必然是一个边界上的边，因为一条内部的边不可能含有两个边界点。

2.6 总结

物体表示方法是在很多因素的影响下发展起来的，这些因素有原始数据的属性，易于绘制、易于进行形状编辑、适合于动画等等。还没有对于所有实际应用均适用的综合的解决方案。而对于我们已经使用了多年的最普及的解决方法——多边形网格法，则一旦离开了离线绘制的静态物体的范畴，这个方法就显示出很明显的缺点。我们以列出界定任何表示方法的属性来结束这一章的内容。这些属性可以对各种表示方法进行一种非常一般性的比较（为了完整性考虑，我们还给出了对双三次参数曲面片方法的评论，这个内容将在下一章中论述）。

- **物体的创建/表示。**这是一个很明显的上下文相关的因素。我们有一些可以由物理数据自动产生表示的方法（例如，通过蒙皮算法而从跨度数据获得多边形网格，通过表面数据的插值获得双三次参数曲面片）。其他方法则直接将输入数据映射到一种体素表示中，

有些方法适合于交互式的创建（CSG方法和双三次参数曲面片方法），而另一些方法则是通过一种基于“数学”的交互设备产生的，如沿着一条脊线扫掠截面（如多边形网格和双三次参数曲面片）。

- **基元的性质。**基元的通常形式或者是表示表面的方法——边界表示（多边形网格和双三次参数曲面片）或者是体积（体素和CSG）。
- **精确度。**表示方法或是精确的，或是近似的。多边形网格表示是近似的方法，但是其精确度可以随着数据的扩展而达到任意程度。以一种智能的方式来提高多边形网格表示的精确度是困难的，较容易但“蛮力”的方法（即在形状中去掉很多的多边形）可能会导致有些区域“过表示”。依应用领域的不同，双三次参数曲面片方法既可以是精确的，也可以是近似的。表面插值可以给出一种近似的表示，而用一个曲面片来设计一辆汽车的门面板则会给出一个精确的表示。CSG表示方法是精确的。但是，这需要有两个先决条件。这种方法只能描述形状的子集，这个子集可以通过所提供的基元的某些结合来得到。CSG表示是精练的，它只是一个合成物体的公式。而要把物体可视化，就必须从公式导出几何表示。 64
- **精确度与数据量。**在精确度和数据量之间总是要有某种折中——至少到目前为止就我们所考虑的绘制的障碍而言是如此。为了提高边界表示方法或体表示方法的精确度，我们必须增加低层元素的数量。尽管球的隐函数完全精确并且很精简，但必须为了绘制而将其进行转换，转换将采用一些会产生低层元素的几何取样过程。
- **数据量与复杂性。**在数据量和表示复杂性之间通常也会有一种折中。而表示的复杂性对于用该表示方法来进行操作会产生实际的后果，最好的例证来自于对多边形网格与采用双三次参数曲面片方法产生的相对应的网格进行比较。
- **编辑和动画的方便性。**这种说法是指对已有模型进行内部编辑或指动画环境中的形状分解技术。对于编辑静态物体形状的最好方法当然是CSG表示方法，这个方法就是为此而设计的。编辑双三次参数曲面片的难易程度要依形状的复杂性以及编辑操作所需的自由度而定。从这个角度来看，编辑一个曲面片容易，而要编辑一个曲面片组成的网格则很困难。以上所进述的方法中没有一个方法是适合于在动画序列中进行形状改变的，尽管人们曾尝试过使用双三次参数曲面片方法和隐函数方法来做这件事。对于精确度的要求和动画形状改变的容易程度似乎是对立的。那些可以具有很高精确度的方法，难于用其进行动画制作，因为它们是由一个可能带有成千上万的低层基元（如叶结点）组成的结构。例如，对一个表示某事物的双三次参数曲面片表示进行控制。比如，控制一个人物的脸，把其组织成一个允许进行局部改变的层次结构（改变是通过对层次降序，并在几个甚至一个曲面片上进行操作而得到的），并且通过在结构的较高层次上进行操作而进行更全局性的变动。由于没有产生很好的结果（比如，在脸部的动画中），所以还没有出现一种可以广泛接受的动画技术。看起来形状改变动画还需要不依赖于物体模型，最成功的技术是把物体模型嵌入到另一种结构中去，而用这种结构来制作形状改变的动画。这样的话，我们就可以通过把一个几何结构贴到一个肌肉控制模型中，或者将一个几何模型嵌入在一个弹性实体的域中，然后再对弹性实体进行动画，从而来控制脸部动画。换句话说，对于目前所使用的表示方法来说，形状的动画似乎不可能直接通过对物体的几何结构进行操作来实现。 65

第3章　三维物体的表示和建模（2）

3.1 Bézier曲线
3.2 B样条表示
3.3 有理曲线
3.4 从曲线到表面
3.5 B样条表面的曲面片
3.6 建立曲面片表面
3.7 从曲面片到物体

引言

在前面的章节中，主要集中讨论了多边形网格表示法。在这种方法中，例如多边形是一个有四个顶点并由四条直线连接起来的平面四边形。在本章中，我们重点关注另一种表示形式，即基元（双三次参数曲面片）是一个曲边的四边形。它有四个角点，这四个点被四条边连接起来，而四条边本身又是三次曲线。曲面片内部是一个弯曲的表面，其表面上的每一个点都是已定义的。这与多边形网格表示法相反，在多边形网格表示法中，物体表面上的点只是在多边形的顶点处有定义。

用双三次参数曲面片来表示物体的表面在计算机图形学中有两个主要的用途:

1) 在CAD中作为交互式设计的基础。在这里，我们可以通过一个交互式的过程（这是一个通过与程序进行交互而建立模型的设计器）来获得模型。在很多CAD应用程序中，表示形式被直接地转换成真实的物体（或者转变成一个按比例缩小的真实物体的模型）。将计算机图形学表示用于控制像数控机床（这种机床用于在某种材料上雕刻）这样的设备。这与“正常”的计算机图形学建模方法学正好相反，不是把真实的物体转换成一种表示，而是用计算机图形学模型来产生真实的物体。

2) 作为对多边形网格的另一种可选的表示形式，这是一种服务于正常的计算机图形学需求的表示，它将真实的物体转换成一种表示形式。在这种用途中，我们通常希望利用精确的参数表示而不用近似性的多边形网格。这里，可以通过一些（表面）插值技术从真实的物体中获得参数表示。

相对于多边形参数表示来讲这种表示的明显优点是:

- 它是一种精确的分析式表示。
- 它具有三维形状编辑的潜力。
- 它是一种更经济的表示方法。

了解了这些优点之后，我们会惊奇地发现这种表示在计算机图形学中并不是主流的表示方法。可以肯定的是，对由一个曲面片的网来表示的物体进行绘制并不困难，所以我们得到的结论是: 这种方法在计算机图形学中（当然它已被用于工业CAD中）没有普及的原因是与之相关的数学公式。

表示因子的精确度需要精心地描述。真实的物体（或真实物体的物理模型）可以由一个曲面片的网格或网来表示（图3-28和图3-43为两个这样的物体）。但是，这种表示不一定完全“精确”。在第一个例子中，由于表示方法的限制，茶壶不可能有一个完全圆的截面，在这种情况下，Bézier形式或Bernstein基函数不可能非常精确地表示一个圆。在第二个例子中，表示表面的那些小曲面片并不是处处与真实物体相一致的。可以通过一个三维数学化仪得到物体表面上的合适的点集，然后可以用同一个点集去建立一个多边形网格模型。接着采用一种称为表面拟合的插值技术来确定一组代表表面的曲面片。但是，曲面片的表面与物体的表面不一定是一致的。拟合的精确程度依赖于插值过程的性质以及物体的表面与双三次曲面片表示方法的形状约束如何相符。但是，我们的确使这个方法在一个具有光滑表面的状态时结束了对物体的表示。与多边形网格表示相比，这种表示肯定具有某些优点，即解决了在绘制多边形网格物体时最突出的视觉上的缺陷——轮廓边的问题。

这种方法也可以用于建模具有精细形状的物体，如用一个曲面片的网表示人类的脸。如果用多边形网格对这样的物体进行适当的表示可能需要非常高的多边形分辨率。除此之外，
67 与双三次参数曲面相联系的是它具有很明显的复杂性。在很多情况下，可以通过采用多边形网格表示方法来避免这一点。当我们对真实的物体进行数字化时，正常情况下所做的应用工作不需要精确的表示。例如，可以为一个动画的TV广告建立一个产品的模型，在这种情况下，一个好的多边形网格模型已经足够了。

事实上，双三次参数曲面片表示的最常见应用并不是建立非常复杂的模型，而是用来表示工业CAD或者CAGD应用中的相当简单的物体。在这里，这种表示的真实价值是：它可以用于将一种在交互式程序上建立起来的抽象设计直接转换成一个物理的真实物体。可以建立一种描述来驱动一个切削设备，例如一个数控机床，在没有人类干预的情况下生产出一个原型物体。正是这个因素使得在CAD中双三次参数曲面片占有很重要的地位。

这些曲面片在CAD中的部分价值在于其具有在保持光滑表面的情况下，改变由曲面片表示的物体的形状的能力。有时，还会提到雕刻加工。可以将此种表示看作是一种“抽象的可塑物”模型，这种可塑物可以拖动，并被分解成任意所需的形状，这给予人们像雕塑家用真实的黏土模型进行创造一样的自由度。在这里，我们应该防止计算机图形学文献中的某些说法，这些说法涉及到用双三次参数曲面片自由地雕刻的经验。我们必须把那些对于所形成的物体的形状复杂性没有约束的、旨在得到一种自由形状的雕刻模型的方法与那些CAD中的更成熟的技术相区分，而后一种技术所表示的物体倾向于相当简单。这一类型技术的一个早期的例子是汽车车身的设计。双三次参曲面片方法被证明在这样应用中是非常成功的。而这种方法作为用于黏土雕刻手段上的成功却引起很多争议。

我们把用单个曲面片表示的物体与那些由于形状（复杂）而需要由一个曲面片的网来表示的物体进行区分。对于一个曲面片进行形状编辑是简单直接的，但是对于那些可以用单个曲面片去设计的物体的形状编辑却受到限制。其中一个问题是，如果我们必须改变网中的一个曲面片的形状的话，就必须保持该曲面片与其所在的整个表面上其他相邻曲面片的光滑的连接关系。另一个问题是尺寸问题的另一种表现形式。假如说我们想要做涉及很多曲面片的一个形状的改变，就必须把这些曲面片放到一起，而且要保持这些曲面片与它们所有相邻曲面片之间的连续性。

尽管有这些困难，但我们必须认识到与多边形网格表示方法相比较，这种表示方法具有很强的形状编辑潜力。这种表示方法已经是一种近似方法，若把顶点向其周围移动以改变所

表示的物体的形状会导致很多困难。一旦顶点移动，则多边形网格表示的精确度也会改变，这将可能导致视觉上的缺陷。几乎可以肯定的是，我们可能总是会移动一组点，而不是在三个空间中移动一个多边形顶点。移动一个顶点也许只是导致一个局部的峰值。 68

在这一章中，我们主要研究单个曲面片，以及由少数几个曲面片构成的网形成简单形状的问题，这个形成过程是采用初步的但威力强大的CAD技术来完成的，比如通过扫掠一条轮廓线360°来产生一个实心物体。

对曲面片的分析表示由于公式的不同而不同，而有些表示方法是以其提出者而命名的。其中一个最著名的公式是Bézier曲面片，这种曲面片是20世纪60年代Pierre Bézier为了设计雷诺汽车而建立的。他的CAD系统称为UNISURF，是最早采用这种表示的系统之一。接下来，我们主要讨论Bézier曲线和B样条公式。

在考虑参数表示时，通常的步骤是从对三维空间曲线的描述开始，然后再泛化到表面或曲面片。三维空间曲线是光滑的曲线，它是一个三个空间变量的函数。跟踪在空间中运动的粒子产生的路径就是一个例子。曲线本身在计算机图形学中也有用途。例如，可以用一条空间曲线来描述三维计算机动画中一个物体的路径。我们还可以通过沿着一条空间曲线扫掠一个截面而建模一个“管道型”的实体。

3.1 Bézier曲线

在这一节中，我们将讨论Bézier的开创性贡献，他是最早在工业设计中建立计算机工具的人。我们将根据Bézier的描述，按其方法本身的进展规律来绘制图形。这样做不仅是因为对其历史发展的兴趣，而且因为可以对形式表示、物理真实性以及要使用该方法的设计者的需求之间的关系有一个透彻的了解。

Bézier的开发工作是20世纪60年代在雷诺汽车公司进行的，他把他的系统称为UNISURF。汽车设计者关心如何设计自由形状的表面，再将其用于加工成主模具，再用这个模具生产出那种可冲压出产品部件的工具。许多其他的工业也使用自由形状的表面。某些像船身、飞机机架、涡轮叶片这样的部件都会受到空气动力学和流体动力学因素的限制，其形状也要经过实验和在风洞中及测试柜中进行测试之后才能确定下来。但是，设计者还是需要有在这种限制之下产生形状的自由。在Bézier这种表示形式开创之前，不能够分析性地表示这种自由形状的表面。而且，曲面一旦建立起来不能够保存供再次生产，也不能通过按坐标进行取样和存储来进行革新。

在Bézier的发明之前，从抽象设计到原型的过程是漫长的，需要有很多人参加，并需要很多道工序。下面摘自Bézier在Piegl的书中（Piegl 1993）的一段描述就是指当时的汽车设计所用的流程。 69

1) 形状设计者用小型的草图和泥塑模拟设计一个总的形状。

2) 通过对模型的测量得到偏离值（在计算机图形学中为世界坐标），设计者跟踪得到汽车车体表面的一个完全尺寸的形状。

3) 抹灰工建起一个完全尺寸的模型，其重量大约有8吨。这个工作从用胶合板来拷贝草图的曲线得到截面开始。接下来由形状设计者和销售经理考察黏土模型，并根据他们的意见进行修改。

4) 当该模型最终被接受时，再次测量其偏离值，从而得到最终的草图。时间可能是一年

或更长。在这一期间，模具和生产专家通常会提出一些修改的建议以防止在生产过程中可能出现的困难和昂贵的操作。

5) 草图最终完成。建立起了一个三维的模具，作为检查压制工具和冲压部件的标准。

6) 主模型的石膏复制品被用来钻孔和在复制机工具上进行铸模。

Bézier的开创性工作完全改变了这些过程中的大部分内容，他的工作允许进行自由形状表面的表示。在此之前，设计者可能使用曲线板（French curve）这样的工具来产生曲线。设计者运用技巧和经验，一步一步地通过运用曲线板上的某些部分的线段来产生一条完整的曲线。这样产生的曲线不能很方便地保存起来，除非作为一组样本进行保存。Bézier的发明是一种定义，它使得这样的曲线可以用四个点以及一个基函数或称弯曲函数的隐含集来进行表示，这四个点称为控制点。当把四个点引入到定义中后，就产生或绘制了曲线。这样做立即会有两个结果。该定义可以直接用于驱动一台数控机床，而部件也可以在没有加入更复杂化的操作和延迟的情况下被生产出来（数控机床自1955年以来就存在了，它还是CAGD发展的另一个动力）。此定义还可以用作CAD程序的基础，在这个程序中，对曲线的修改是在计算机可视化的情况下进行的。

Bézier还阐述了当时他经历的令人关注的困难问题:

> 当人们建议用这些曲线取代扫掠和曲线板时，大多数形状设计者提出了反对意见，他们已经建立起了自己的模板，并且不会改变他们的工作模式。因此，很郑重地保证，将他们保密的曲线转换到保密的列表中，并且使其掩藏在计算机存储器的最保密的地方，除他们自己之外，没有人有打开天花板的钥匙。事实上，标准的曲线已经有足够的灵活性，而这些秘密的曲线也很快被忘记了。设计者和制图员很容易地理解了多边形以及它们与相应的曲线形状之间的关系。

在其他工业中，同时也有很多开发工作。值得一提的是飞机和船泊的制造，而且很多研
70 究工作都是在一些专门的制造商的支持和赞助下完成的，如雷诺公司的Bézier。他们开发了自己的CAD系统，以及适合于他们自己的需求的表面表示方法。这就导致了有许多表面的参数化定义，有兴趣的读者最好参考Piegl的书，在这本书中，每一章都是由这个领域的一个先驱者所写的。

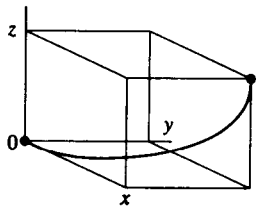
Bézier说道，他的表示方法的最重要的需求之一是这种方法必须是基于几何的，而且所含的数学内容一定要容易理解。他还介绍了空间曲线的概念，这条曲线被包含在一个立方体之中，当立方体变形为一个平行六面体时，曲线也变形了（见图3-1）。该曲线按下列条件被“固定”在平行六面体中:

- 曲线的开始点和终点位于平行六面体的相对的顶点上。
- 在其开始点处，曲线正切于0x。
- 在其终点处，曲线正切于0z。

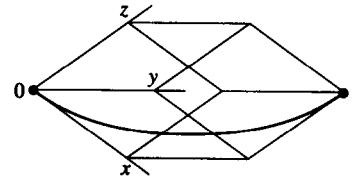
这个几何概念唯一地定义了任一空间曲线（如果理解为该曲线是一定次数的多项式的话），并且这个概念还给出了曲线是如何随着平行六面体的改变而改变其形状的一种直观的看法。现在，此平行六面体以及曲线就可以完全由四个点来定义，这四个点称为控制点，即$\boldsymbol{P}_0$、$\boldsymbol{P}_1$、$\boldsymbol{P}_2$和$\boldsymbol{P}_3$，它们只是图3-1中所示的平行六面体的顶点。已知曲线端点的位置固定，则其在端点处的行为也被确定了，需要对空间中端点之间所得到的轨迹的形状进行定义。选择了一种参数定义，就意味着空间曲线$\boldsymbol{Q}(u)$是以参数$u(0 \leqslant u \leqslant 1)$来定义的。当$u$从0到1变化时，我们通

过按比例调节或弯曲控制点获得曲线$Q(u)$上一个点的位置。这就是说，曲线上的每一个点都是通过一个称为基函数或弯曲函数的三次多项式对每一个控制点进行比例调节来确定的。同时，曲线由下式给出:

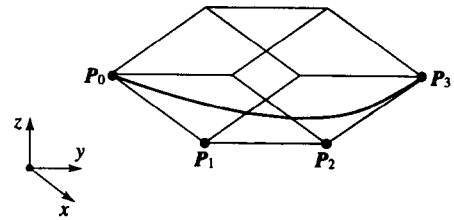
$$Q(u)=\sum_{i=0}^{3}P_iB_i(u) \tag{3-1}$$



包含在一个立方体中的曲线



将立方体拉成平行六面体就改变了曲线的形状



用作控制点的顶点

图3-1 Bézier的曲线表示原理

对于Bézier曲线，其基函数或弯曲函数为Bernstein三次多项式:

$$B_0(u)=(1-u)^3$$
$$B_1(u)=3u(1-u)^2$$
$$B_2(u)=3u^2(1-u)$$
$$B_3(u)=(u)^3$$

图3-2为这种类型的多项式和一条Bézier曲线（投影到图的二维空间中）。

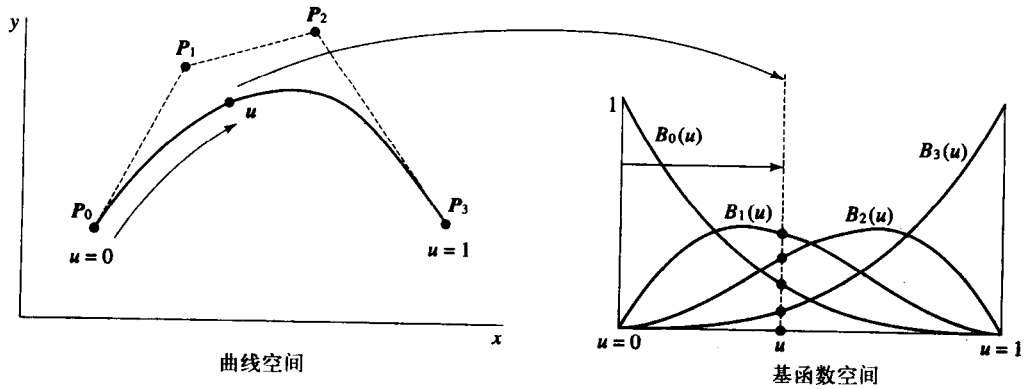


图3-2 通过增加u的值沿着曲线移动等价于移动通过基函数的垂直线。这条线与基函数的截点给出等价于该点的B的值

一个有用的、直观的概念是：从物理的角度看，沿着这条曲线从$u = 0$移动到$u = 1$，我们同时会移动一条基函数空间中的垂直线，函数空间及垂直线定义了基函数的四个值。通过用控制点对每一个基函数求权重然后求和，可以在曲线空间中得到相应的点。注意，对于任意的u值（除了$u = 0$和$u = 1$），所有的函数值都是非零的。这就意味着，所有控制点的位置都对曲线上的每一个点有贡献（端点除外）。在$u = 0$处，只有B_0为非零值。因此：

$$\boldsymbol{Q}(0) = \boldsymbol{P}_0$$

71
~
72

同样地，有

$$\boldsymbol{Q}(1) = \boldsymbol{P}_3$$

我们还注意到：

$$B_0(u) + B_1(u) + B_2(u) + B_3(u) = 1$$

把四个控制点连接起来给出所谓的控制多边形，而移动控制点将产生新的曲线。移动该曲线上的一个控制点，就会以一种直观的方式改变其形状。这种情况如图3-3所示。移动端点的作用是很明显的。当移动内部控制点$\boldsymbol{P}_1$和$\boldsymbol{P}_2$时，曲线在端点处切向量的方向也随之变化，这也是很明显的。不太明显的情况是，$\boldsymbol{P}_1$和$\boldsymbol{P}_2$的位置也可以用来控制切向量的大小，如下式所示：

$$\boldsymbol{Q}_u(0) = 3(\boldsymbol{P}_1 - \boldsymbol{P}_0)$$
$$\boldsymbol{Q}_u(1) = 3(\boldsymbol{P}_2 - \boldsymbol{P}_3)$$

其中$\boldsymbol{Q}_u$是端点处曲线的切向量（一阶导数）。可以看出，曲线随着该值的增加被推向切矢向量，而这个值的大小由控制点的位置来控制。

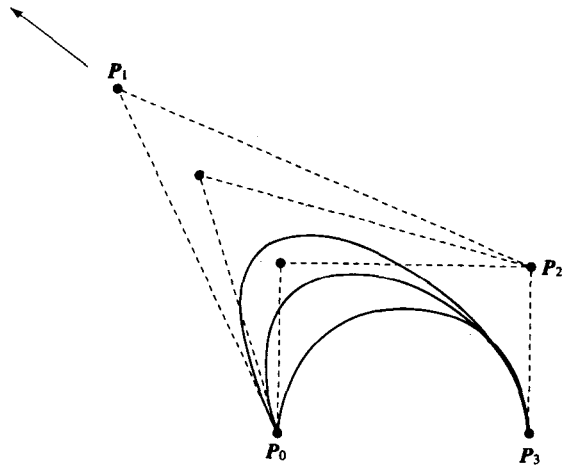


图3-3 移动控制点$\boldsymbol{P}_1$的效果

Bézier曲线不仅仅用在技术含量高的应用中，而且也用在软件中。在现在的字处理器以及DTP应用程序中可以发现，若要绘制一个组件，总是会包含一个基于Bézier曲线的画框的工具。Bézier曲线的另一个非常著名的应用如图3-4所示。在该图中，正在设计一种字体，被填充的字符的轮廓是一组Bézier曲线，设计者可以通过移动控制点对其进行细微的变动，这些控制点定义了描述轮廓的曲线。

Bézier原来的三次的概念（用三个空间变量封装一条曲线）似乎已经被丢掉了，大多数文

献中只处理包含在一个控制多边形中的两个空间变量的曲线。必须设计有三维空间曲线的应用程序，例如，三维计算机动画，这些程序可以在那些使用二维曲线投影的地方加入接口。 73
在17.2.2节中给出了一个这方面的应用（请注意：对于一个三维的曲线，平行六面体确定了一个平面，曲线的切线方向即控制多边形的边在这个平面中被确定）。

在这一点上，考虑用控制点表示一条曲线的所有细节非常有用。到目前为止，对于交互性来说，最重要的性质就是当移动控制点时会给出曲线形状的一种直观的改变。另一种叙述的方法是，曲线模仿了控制多边形的形状。从处理曲线（以及表面）的算法的角度来考虑，一个重要的性质是，曲线总是被包含在由控制多边形所形成的凸包（convex hull）中。一条三维空间曲线的凸包如图3-5所示，这个凸包可以看成是在控制点周围放置一个有弹性的带子所形成的多边形。接下来的事实是，基函数对所有的u求和得到1。

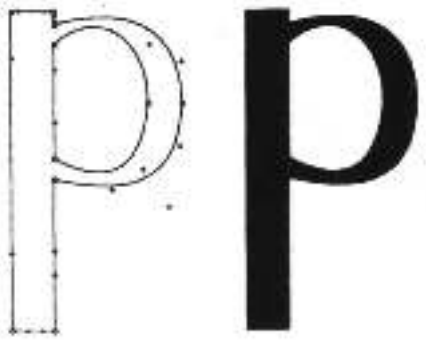


图3-4 在字体设计中使用Bézier曲线，每一个曲线段控制点都由○+●进行标记

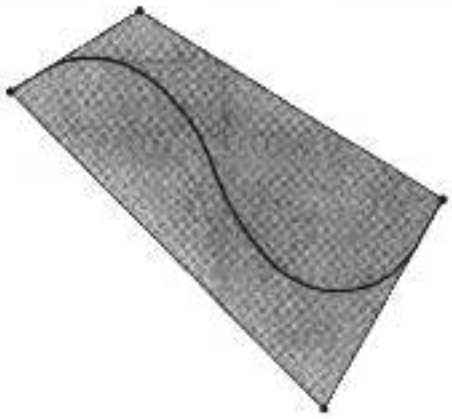


图3-5 三次样条的凸包性质，曲线被包围在由控制点所形成的经明暗处理的区域内

现在考虑变换曲线。由于曲线被定义为控制点的线性组合，通过对控制点集合应用适当的变换就产生了在三维空间中的任意仿射变换（旋转、缩放、平移等），因而变换了曲线。因 74
此，要变换曲线，首先变换控制点，然后计算曲线上的点。在这种情况下，我们注意到，通过计算点来变换曲线并不比变换（像隐含表示方法所做的那样）容易。例如，当放大曲线时，从比例上来讲我们并不清楚为了保证光滑性需要用到多少个点。这里需要注意的是，透视变换是非仿射变换，所以不能将控制点映射到屏幕空间，并在那里计算曲线。然而，如本章之后描述的那样，可以通过使用有理曲线来克服这种显著的缺陷。

最后，下面的计算公式是对于求和形式的一种替代。首先，我们将方程（3-1）进行扩展，得到

$$Q(u) = P_0(1-u)^3 + P_1 3u(1-u)^2 + P_2 3u^2(1-u) + P_3 u^3$$

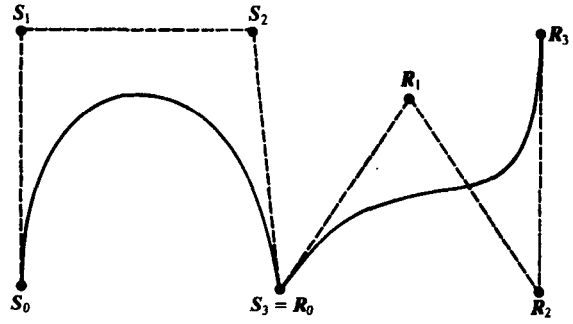
然后可以将其以矩阵形式写出：

$$Q(u) = UB_zP$$

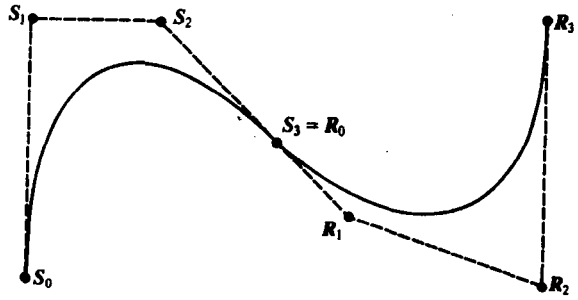
$$= [u^3\ u^2\ u\ 1]\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

3.1.1 连接Bézier曲线段

由一组四个控制点定义的曲线段可以被连接起来，以构成比通过单个曲线段可以得到的更复杂的曲线。这就导致一种所谓的分段多项式曲线。表示更复杂曲线的一种替代方法是增加多项式的阶数，但是这样做在计算中和在有数学上都有缺点，因此一般考虑将其分解成三次片段要容易一些。将曲线段连接起来意味着必须对连接点进行限制。默认的限制是位置连续性，第二个最好的限制是一阶连续性（或称切线连续性）。对于Bézier曲线来讲，位置连续性和一阶连续性之间的差别如图3-6所示。位置连续性意味着第一条线段的终点与第二条线段的起点相连。而一阶连续性则意味着，特征多边形的边是共线的，如图3-6所示。这就是说，一条曲线的末端是与另一条曲线的起点的切矢量相匹配的。在有明暗处理的表面上，仅保持位置上的连续性可能会导致连接点在最终被绘制出来的物体上是可见的。



a) Bézier点之间的位置连续性



b) Bézier点之间的切线连续性

图3-6 Bézier曲线段之间的连续性

如果两条线段的控制点为S_i和R_i，则当下式成立时，将保持一阶连续性:

$$(S_3 - S_2) = k(R_1 - R_0)$$

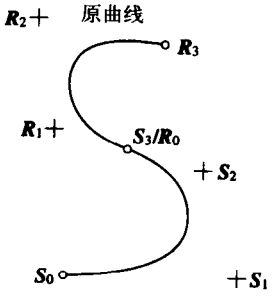
利用这个条件可以很容易地通过一次加入一条线段的方法建立起一条合成的Bézier曲线。然而，能够由片段建立起合成的曲线的优点，在某些程度上被由于连接条件而采用的那些局部控制条件的限制所削弱了。

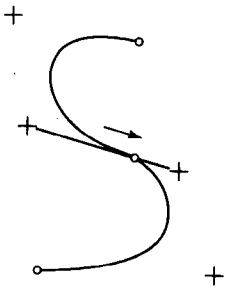
图3-4是一个多段Bézier曲线的例子。在这个例子中，由一些曲线段连接起来表示字符的轮廓，在这些曲线段之间保持了一阶连续性。考虑有一个接口的细节是很有用的，用户可以通过这个接口来编辑多段曲线并保持其连续性。图3-7表明了一些可能的方式。这个图假设用

户已经构造了一条两段曲线，这条曲线的形状将在连接点S_3/R_0的周围被改变。为了保持连续性，我们必须在R_1、R_0/S_3和S_2三个点上同时进行操作。可以按下面的方法来达到这一目的:

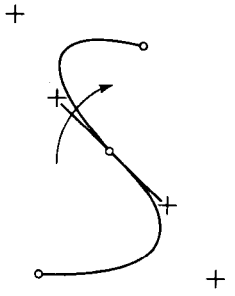
- 保持线段R_1、S_2的方向，对这条线的连接点进行上下移动（见图3-7a)。
- 保持连接点的位置，并绕这一点旋转线段R_1S_2（见图3-7b)。
- 以一种“锁住的”单元的形式移动所有三个点。

这三种编辑的可能性或限定将使得用户能够改变由任意数量的线段组成的曲线的形状，而同时可以保持各曲线段之间的一阶连续性。稍后我们将看到，Bézier曲线的这种复杂性可以通过B样条曲线来克服。

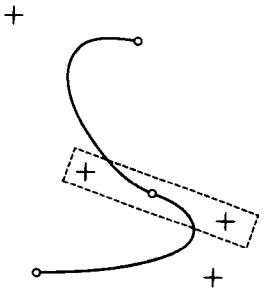




a) 保持R_1S_2的方向，沿着这条线移动三个控制点R_1、S_3/R_0，S_2之一



b) 绕S_3/R_0旋转线段R_1S_2



c) 以一种“锁住的”单元的形式移动三个控制点R_1、S_3/R_0、S_2

图3-7 对一条两段的Bézier曲线进行可能的形状编辑的例子

3.1.2 Bézier曲线性质总结

- Bézier曲线是一个多项式。多项式的阶数总是比控制点的个数少一。在计算机图形学中，我们一般采用三阶。四阶曲线的灵活性不足，而且超过三阶时，复杂性增加，所以选择三阶对于大多数计算机图形学应用来说是最好的折中方案。
- 曲线“遵循”控制点多边形的形状，并且被限制在由控制点形成的凸包内。
- 控制点并不对“局部”进行控制。移动任何控制点都会影响曲线上的所有点，只是其影响作用或大或小。这个作用可以通过考察图3-2看出。在该图中，所有的基函数值除了在$u = 0$和$u = 1$点之外，其他都不为零。
- 第一个和最后一个控制点是曲线段的端点。

- 在端点处曲线的切向量与控制点多边形的第一条和最后一条边相重合。
- 移动控制点会改变切向量的大小和方向，这是Bézier曲线接口的直观感觉的基础。
- 曲线关于任意直线的振荡并不会比关于控制点多边形的振荡多，这种性质被称为变异消失性（variation diminishing property），这个性质具有可以表示的表面本性的含义。
- 通过对其控制点表示应用任意的仿射变换（即任意线性变换的组合），可以变换曲线。在这样的变换下，曲线是不变的（不改变形状）。

75
|
77

3.2 B样条表示

Bézier曲线的简单及其能力无疑是其长期受到欢迎的原因。然而，它也确实有局限性。在这一节中，我们将讨论这些问题，并且考察如何通过用B样条表示来克服这些问题。与以前一样，本节通过给出B样条曲线来介绍B样条。

从历史上来讲，B样条比Bézier曲线出现得早。它们的起源都与工业有关，比如造船厂，在那里要求设计者绘制像船的截面这样的真实尺寸的曲线表示。对于小型的图纸，制图员会使用曲线板，其上是一组小的预先形成的曲线段。他们将由不同曲线板上的不同片段形成的线段放在一起而绘出完整的曲线。对于实物尺寸的图纸，这种方法是完全不可行的。对实物尺寸绘图时，制图员（在造船业，这些人被称为轮廓线放样员）会使用又长又细的金属条。这些金属条被压成所需的形状，用铅块（lead weight）保护起来，铅块和控制点之间的相似关系必须明确。我们可以按系统的要求将样条压成所需的形状并按自己的需求放置任意多的铅块。这就是B样条的物理基础。可以将这一思想与单段Bézier曲线或多段Bézier曲线相比较。如果将其与单段Bézier曲线相比较，可以看到，加入附加的控制点会消除变异消失性，即曲线可以按要求而振荡。当将其与多段Bézier曲线相比较时，可以看到同样的情况，但是这时不必在任何位置很明确地保持连续性。想象一下，轮廓线放样员插入一个附加的铅块——金属样条的物理性质会保证插入了一个新的铅块的点的周围所取的新形状是连续的。金属样条会采取一种使得其内部张力能量最小的形状。然而，由这种真实的工程工作引出的另一个问题是，插入一个铅块的作用是局部的。曲线的形状只是在其邻近得到改变。下面我们以一种正规的方式来讨论这个问题。

3.2.1 B样条曲线

两个Bézier曲线所具有但被B样条曲线克服的缺陷是其非局部性以及曲率与控制点数量之间的关系。第一个性质（非局部性）是指，尽管控制点对最靠近它的那一部分曲线有非常大的影响，但它也会对整条曲线有影响，这种影响可以从图3-2中看出。在整个u的范围之内，所有基函数的值都是非零的。第二个缺点是，在不采用多个曲线段（或通过增加曲线的曲率）
78 的情况下，我们就不能用一个Bézier三次曲线来近似或表示n个点。

与Bézier曲线一样，B样条曲线不通过其控制点。B样条曲线是由任意数量的曲线段组成的完全的分段三次多项式（为了表示上的方便，我们将只讨论三次B样条。但是，B样条可以有任意阶）。系数变化引起某一区间上的三次片段变化，并从一个区间到另一个区间逐个进行。对于只有一个片段的情况，可以用相同的矩阵表示来比较B样条公式和Bézier公式。

B样条公式为:

$$Q_i(u) = UB_sP$$

$$= \begin{bmatrix} u^3 u^2 u 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} P_{i-3} \\ P_{i-2} \\ P_{i-1} \\ P_i \end{bmatrix}$$

其中Q_i为第i个B样条片段，P_i是按控制点顺序排列的一个四个点的集合。另一种表示方法是:

$$Q_i(u) = \sum_{k=0}^{3} P_{i-3+k} B_{i-3+k}(u) \tag{3-2}$$

其中i为片段数，k为局部控制点的下标；也就是片段i的下标。在一个曲线段上u的值为$0 \leqslant u \leqslant 1$。我们可以用这种表示法把$u$描述为一个局部参数，即在参数范围0 ~ 1局部地变化，从而定义一个B样条曲线段。

在这样的表示形式下，可以看到，一条B样条曲线是$m-2$个曲线段的系列。为方便起见，我们将其标记为$Q_3, Q_4, \cdots, Q_m$，这些曲线段由M + 1个控制点$P_0, P_1, \cdots, P_m$来确定或定义，其中$m \geqslant 3$。每一个曲线段都是由四个控制点来定义的，而每一个控制点会影响四个且只四个曲线段。这就是B样条的局部控制特性，同时也是它和Bézier曲线相比的主要优点。

在这里我们必须小心。Barsky（1998）指出，将Bézier曲线与B样条曲线进行比较可能会引起误解，因为这种比较并不是在相同性质上进行的，而是将一个Bézier曲线的片段（它的控制顶点集可以扩展，并且其曲率也可以增大）与一整段B样条曲线或者一个合成的B样条曲线进行比较。一个Bézier曲线段是受全局控制的，因为移动一个控制点会影响整条曲线。而在一条合成的B样条曲线上，移动一个控制点只影响该曲线中的几个片段。比较应该在多段Bézier曲线与B样条曲线之间进行。这里的差别在于，为了保持Bézier曲线片段之间的连续性，对控制点的移动必须满足限制条件，而B样条合成曲线上的控制点可以任意移动。

B样条曲线表现出位置连续性、一阶导数连续性和二阶导数连续性（C^2），能够获得这种性质是因为基函数本身是二阶导数（C^2）分段多项式。这类基函数的一个线性组合也将是C^2连续的。我们把整个曲线段的集合定义为u表示的一个B样条曲线: 79

$$Q(u) = \sum_{i=0}^{m} P_i B_i(u)$$

在这里i为非局部控制点个数，u是一个全局参数，我们将在下一节中对此进行更详细的阐述。

3.2.2 均匀B样条

方程（3-2）表明，B样条曲线中的每一个片段都由四个基函数和四个控制顶点来定义。因此，就有比曲线段多出来的三个基函数和三个控制顶点。在片段之间u值上的连接点被称为结点值（knot value），均匀B样条是指结点值对于参数u是等间距的。图3-8为一条由六个控制顶点或称六个控制点$P_0, P_1, \cdots, P_5$定义的B样条曲线，这个图还显示了改变多项式阶数的作用，图中有阶数分别为2、3、4时的曲线。一般情况下，我们只关心三次曲线，这时，它是一条三段曲线。其左侧的端点为靠近P_0的Q_3，而右侧端点为靠近P_5的Q_5（由此可以看到，均匀B样条一般而言并不会像Bézier曲线那样内插端控制点。而且，Bézier曲线与其控制点多边形更近似。然而，B样条曲线的连续性保持特性比这一缺点更重要）。

这种表示法为我们提供了下列组织方式（每一个曲线段都是交替地以实线/点划线表示

的)：

Q_3由$P_0\ P_1\ P_2\ P_3$定义，由$B_0\ B_1\ B_2\ B_3$来量度。

Q_4由$P_1\ P_2\ P_3\ P_4$定义，由$B_1\ B_2\ B_3\ B_4$来量度。

Q_5由$P_2\ P_3\ P_4\ P_5$定义，由$B_2\ B_3\ B_4\ B_5$来量度。

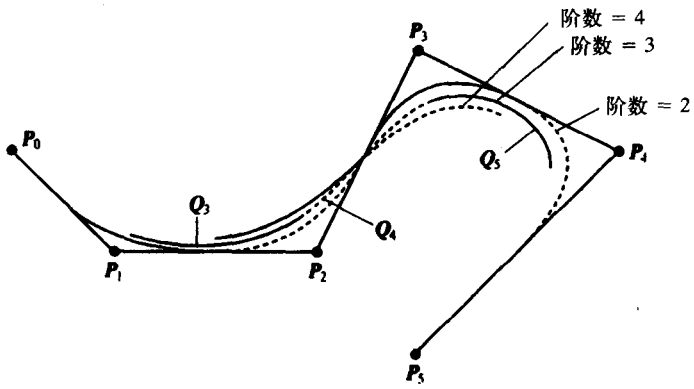


图3-8 由六个控制点定义的三段B样条三次曲线

每一个曲线段都共享控制点这一事实隐含在一个机制中，即在曲线段之间保持C^2连续性。

80 图3-9表示的是改变控制点P_4的位置的效果。这个动作把片段Q_5朝着相应的方向推移，即朝着更靠近片段Q_4的方向，Q_4也是由P_4定义的。但是，它并不会影响到Q_3。这个图表明B样条的重要局部特性。当然，一般情况下，一个控制点会影响四个片段。

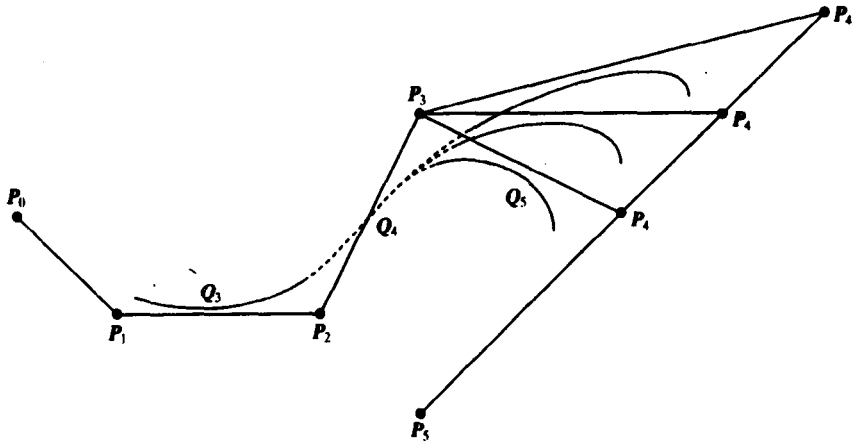


图3-9 展示B样条曲线的局部特性。移动P_4使Q_4和Q_5的距离缩短，但Q_3不变

现在我们来考虑定义曲线的基函数，每一个基函数在u的四个连续区间上都是非零的（见图3-10）。事实上，它是一个由包括其本身在内的四个片段组成的三次函数。B样条在区间u_i，u_{i+1}，…，u_{i+4}上是非零的，并且以u_{i+2}为中心。现在，每一个控制点都由一个基函数来测量，假设结点值是等间距的，则每一个基函数都是一个拷贝或平移。图3-8中曲线所用的基函数集如图3-11所示。

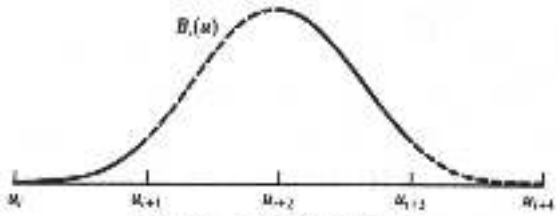


图3-10 三次均匀B样条$B_i(u)$

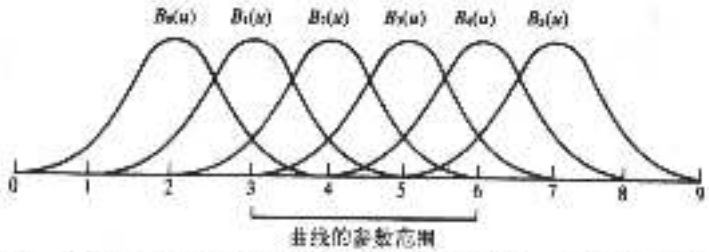


图3-11 在构建图3-8中所示的曲线时所用的六个B样条，它们是互相平移的

在本例中，基函数在区间$u = 3$到$u = 6$的所有值之和为1，$u = 3$到$u = 6$为曲线所定义的参数u的值。接下来的一个情况是，整条曲线均被包含在由其控制点所形成的凸包之中。如果我们只考虑曲线中的一个片段，则即是将参数定义在u_i到u_{i+1}的范围。在第i个参数区间u_i到u_{i+1}为基函数的作用域，基函数为定义一个曲线段的函数，见图3-12中突出显示部分。这个特点给出了对于函数随着u的变化而变化的行为的一种有用解释，一般而言，对于不为结点值的其他u值，四个基函数有效，并且其和为1。当达到结点值$u = u_i$时，一个基函数"关闭"，另一个基函数"开启"，在结点值处，有三个基函数，且其和为1。 81

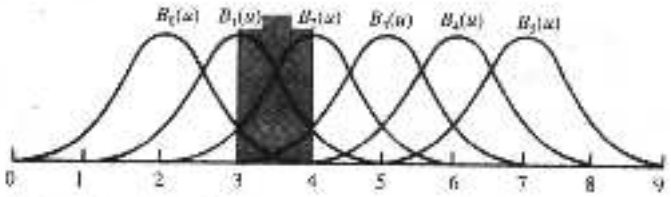


图3-12 对于图3-8中的第一个曲线段，非零或起作用的四个B样条

到了这一阶段，我们可以总结出，B样条曲线是由$m-2$个片段组成的，这些片段是由$m + 5$个结点值上的$m + 1$个基函数的位置定义的。因此，在图3-7中有10个结点值上的三个片段、六个控制点和六个基函数。

现在我们再来看图3-12，参数在$u_i \leqslant u \leqslant u_{i+1}$区间时，我们通过用$0 \leqslant u \leqslant 1$进行替代，求出四个B样条$B_i$、$B_{i-1}$、$B_{i-2}$、$B_{i-3}$：

$$
\begin{aligned}
B_i &= \frac{1}{6}u^3 \\
B_{i-1} &= \frac{1}{6}(-3u^3 + 3u^2 + 3u + 1) \\
B_{i-2} &= \frac{1}{6}(3u^3 - 6u^2 + 4) \\
B_{i-3} &= \frac{1}{6}(1-u)^3
\end{aligned}
\tag{3-3}
$$

重要的是，应指出这种定义只是用来从四个B样条基函数中的每一个基函数在区间$0 \leqslant u \leqslant 1$上定义一个片段。它不是指在区间$0 \leqslant u \leqslant 4$上，由四个片段组成的一个B样条基函数。

现在我们来讨论两端的控制顶点，需要再次注意的是，曲线并不与这些控制点内插。当然，一般而言，B样条曲线不与任何控制点内插。我们可以通过引入多个顶点的方式使得B样条曲线内插控制点。但是，我们将看到，这样做会损失连续性。凭直觉来看，可以考虑通过重复来增加控制点的影响作用。这时曲线被吸引到控制点上。片段由量度控制点的基函数构
82 成。如果将控制点进行复制，则它将在求一个片段中使用多次。例如，考虑图3-13，并把它与图3-8相比较。在图3-8中，最后一个控制点被重复了三次。现在有五个片段，P_5在确定Q_5时使用了一次，而在确定Q_6时使用了两次，在确定Q_7时使用了三次。现在，这条曲线在范围$3 \leqslant u \leqslant 8$上。在$u = 8$处，曲线与$P_5$重合。

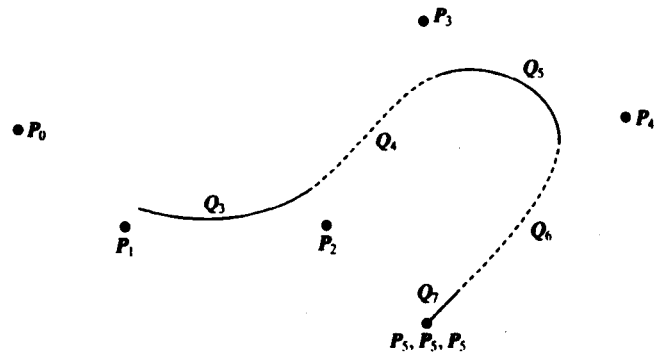
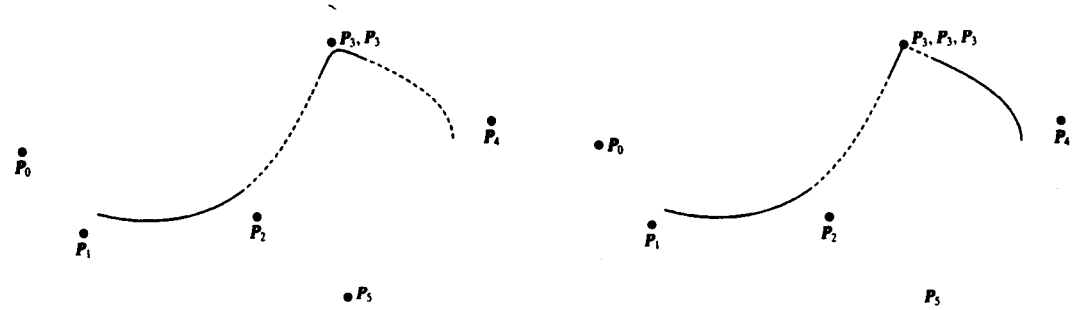


图3-13 多个端控制点作用的示意图。P_5被重复了三次，迫使曲线与其内插

这样的技术可以用于使得曲线在中间控制点和端控制点处内插。图3-14a显示了引入多个中间控制点的效果。在这个图中，P_3被重复。P_3几乎被内插了，而且引入了一个附加的片段。这就引起连续性从C^2G^2到C^2G^1的变化。也就是说，两个片段之间的连接处的连续性减少一阶，尽管这时每一个片段内部的连续性仍然是C^2。图3-14b显示了P_3点成为三重控制点的效果。这时，曲线与控制点内插，而且控制点的两侧曲线变成了直线，这时连续性降低到C^2G^0。



a) P_3被复制　　b) P_3为三重的

图3-14 多个内插控制点作用的示意图

3.2.3 非均匀B样条

在上一小节中，我们考察了一类称为均匀B样条的曲线，因为它们的基函数是可以互相平

移的。本小节我们来讨论非均匀B样条。

非均匀B样条是一种连续结点值之间的区间不一定相等的曲线。这就是说，弯曲函数不再是可以互相平移的，而是区间和区间之间可能不相等。非均匀B样条的最常见形式是通过插入多个结点使得连续的结点值区间中的某些区间减小到零。这种技巧用于内插控制点（包括端点和中间点）。与上一小节中所使用的方法（即插入多个控制点）相比，这种方法有一些优点。特别是，可以内插一个控制点而不产生像多控制顶点所出现的结果，即在该控制点的两边出现直线线段。

考虑图3-8中所产生的曲线。关于这条曲线的结点值分别为u = 3、4、5、6。我们为这条曲线定义一个结点向量为[0, 1, 2, 3, 4, 5, 6, 7]，有用的参数范围是$3 \leqslant u \leqslant 6$（其中，基函数的和为1）。每一个结点值之间的间隔为1。如果采用非均匀结点值，则基函数不再对于每一个参数区间相等，而是在区间u上变化。考虑图3-15，这个图与图3-8使用了同样的控制点，B样条曲线也是由三个片段组成。但是，这条曲线现在与端点内插了，这是因为在结点向量的每一端插入了多个结点。所使用的结点向量为[0, 0, 0, 0, 1, 2, 3, 3, 3, 3]。基函数也显示在图3-15中。现在这条曲线有9个片段，$Q_0 \sim Q_8$。但是Q_0、Q_1、Q_2缩减到一个点上。Q_3、Q_4、Q_5在$0 \leqslant u \leqslant 3$范围内定义，而$Q_6$、$Q_7$、$Q_8$也缩减到一个点上，即$u$ = 3。在实际使用中，经常会使用结点序列[0, 0, 0, 0, 1, 2, ⋯, $n-1$, n, n, n, n]。这就是说，强迫在端点处进行内插而在其他地方采用均匀的结点。在图3-16中给出了第二个显示B样条曲线灵活性的例子。在这个例子中，有9个控制点和13个结点，结点向量为[0, 0, 0, 0, 1, 2, 3, 4, 5, 6, 6, 6, 6] 。

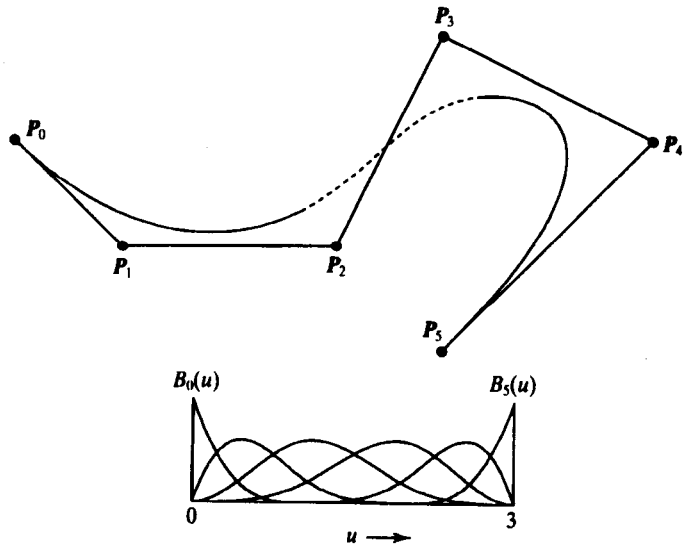


图3-15 用一个结点向量[0, 0, 0, 0, 1, 2, 3, 3, 3, 3]内插端点的一条非均匀B样条

一般情况下，结点向量是其结点值从u_0到u_{m+4}的任意非递减序列。正像我们已经看到的那样，连续的结点值可以相等，相等的值的个数被称为结点的重数。通过采用多个控制顶点来使一条曲线与端点内插并不会产生与多个控制顶点相同的效果。图3-17显示了在标准例子中的最终控制点P_5，用多控制点和用在最终结点值上重数为4的结点向量进行内插的结果。

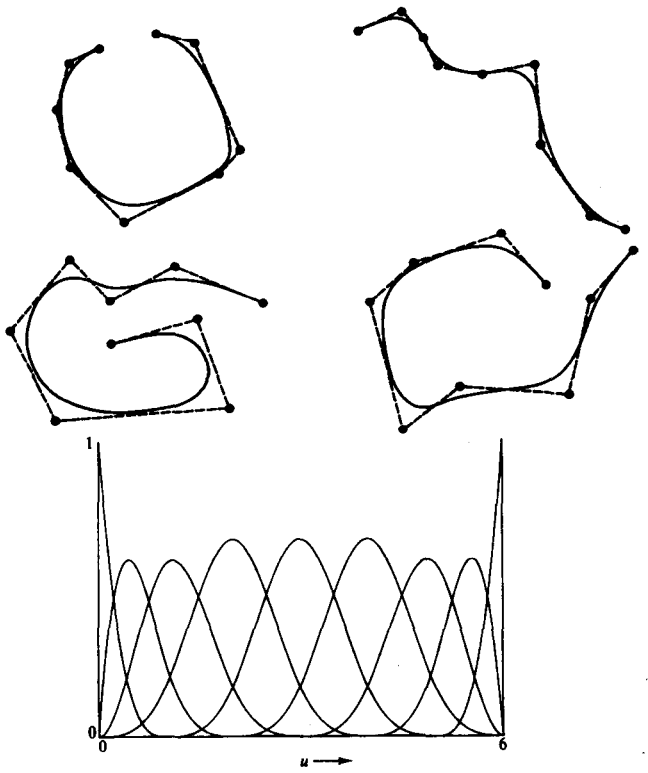


图3-16 显示B样条曲线的灵活性。结点向量为[0, 0, 0, 0, 1, 2, 3, 4, 5, 6, 6, 6, 6]

83
~
85

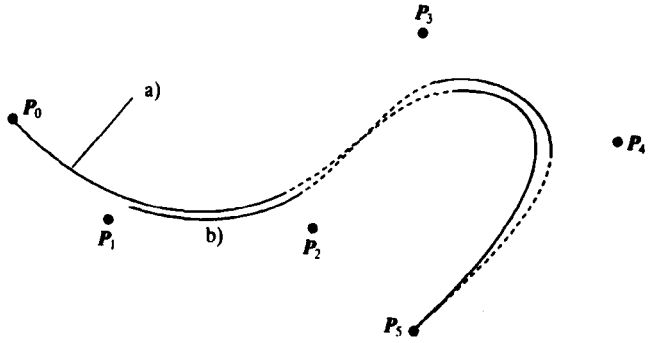


图3-17 多结点和多控制点的比较

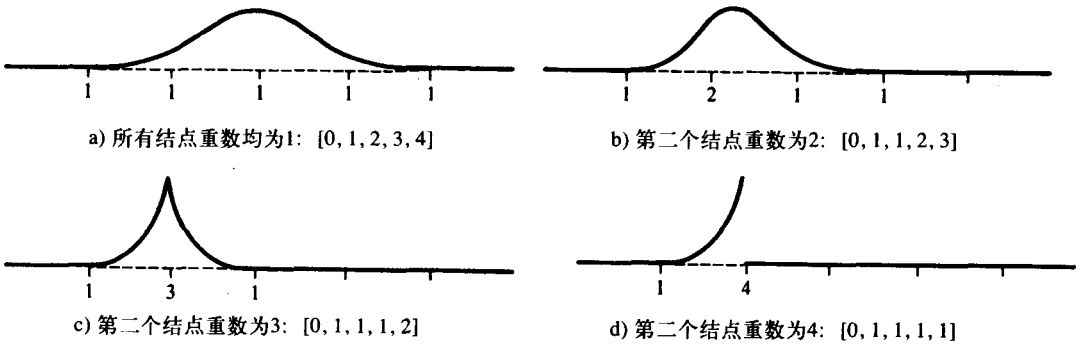
a) 曲线由在起点和终点的值上重数为4的结点向量产生 b) P_5被重复3次

请注意，如果我们采用结点向量[0, 0, 0, 0, 1, 1, 1, 1]，则得到单个的内插P_0和P_3的曲线段。在这种情况下，其基函数为Bézier基函数（见图3-2），产生的结果是一条Bézier曲线。因此，Bézier曲线只是非均匀B样条曲线的一个特例。

可以很容易地看出多结点对基函数的形状的影响。图3-18a为在结点0, 1, 2, 3, 4上定义的均匀B样条基函数。正如在上一小节中解释过的那样，它本身是由在已知范围内定义的四个三次多项式片段组成的。这些片段用方程（3-3），并以u为单位在0, 1, 2, 3, 4处平移每个三次片段来产生。另一种方法是用方程:

$$B_0(u)=\begin{cases} b_{-0}(u)=\dfrac{1}{6}u^3 & 0\leqslant u<1 \\ b_{-1}(u)=-\dfrac{1}{6}(3u^3-12u^2+12u-4) & 1\leqslant u<2 \\ b_{-2}(u)=\dfrac{1}{6}(3u^3-24u^2+60u-44) & 2\leqslant u<3 \\ b_{-3}(u)=-\dfrac{1}{6}(u^3-12u^2+48u-64) & 3\leqslant u<4 \end{cases}$$

将这种表示与方程（3-3）进行比较，并注意它在$0\leqslant u\leqslant 4$范围内定义一个B样条基函数。如果把第二个结点加倍，并使用[0, 1, 1, 2, 3]，$b_{-1}(u)$缩减为零长度，这时函数就成为不对称的，如图3-18b中所示。双重结点消除了二阶导数连续性，但是一阶导数连续性还保持。用结点向量[0, 1, 1, 1, 2]使第二个结点重复三次得到对称函数，见图3-18c。但这个函数现在只有位置连续性。这个结点的四次重复[0, 1, 1, 1, 1]产生图3-18d所示的效果。这时甚至连位置连续性也消失了。



a) 所有结点重数均为1: [0, 1, 2, 3, 4]　b) 第二个结点重数为2: [0, 1, 1, 2, 3]

c) 第二个结点重数为3: [0, 1, 1, 1, 2]　d) 第二个结点重数为4: [0, 1, 1, 1, 1]

图3-18 结点重数对一个三次B样条基函数的影响

如果现在返回去看图3-15，第一个基函数是在[0, 0, 0, 0, 1]上定义的，并且是非对称的，没有位置连续性。第二个基函数是在一个含有三重结点的值的集合[0, 0, 0, 1, 2]上定义的，第三个基函数是在序列[0, 0, 1, 2, 3]上定义的，并且它们均是不对称的。在这个例子中，所有的函数都是非对称的。总结如下:

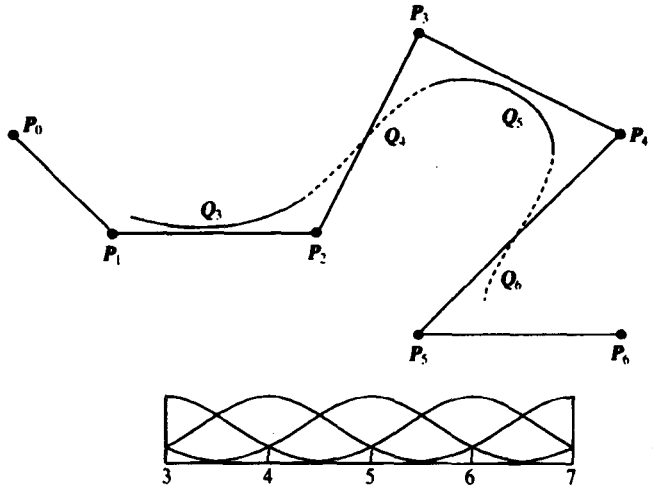
结点向量	基函数
0 0 0 0 1	B_0
0 0 0 1 2	B_1
0 0 1 2 3	B_2
0 1 2 3 3	B_3
1 2 3 3 3	B_4
2 3 3 3 3	B_5

从这组基函数可以进一步得到: 它们在整个u的区间上的和为 1； 在$u=0$和$u=3$处唯有B_0和B_5为非零的基函数，这使得端点分别由Q_3和Q_5内插。

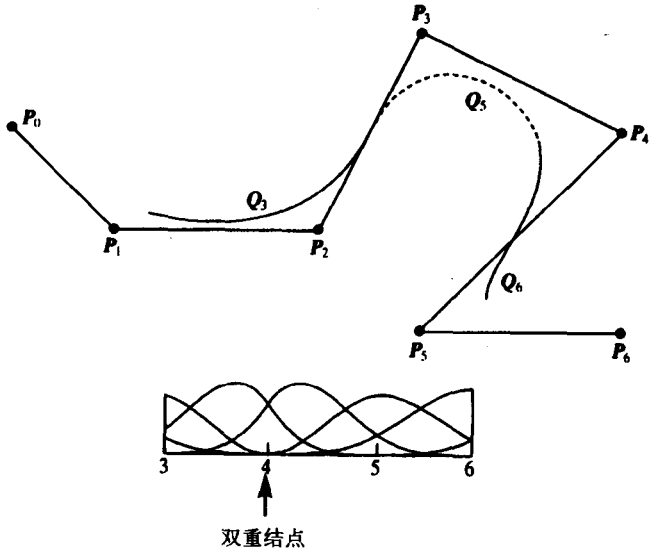
现在我们改变内部结点的结点重数，使连续性的改变问题变得明显。考虑图3-19中给出的例子。这与图3-7中给出的例子情况相同，但加入了一个附加的控制点，给出四个片段曲线。结点向量为 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] ，图3-19a为其曲线，图3-19b为用向量[0, 1, 2, 3, 4, 4,

5, 6, 7, 8, 9]引入一个双重结点的效果，这时片段的个数减少到3。Q_4缩减为零。含有Q_3和Q_5的凸包在边P_2P_3上相遇，在Q_3Q_4和Q_5之间的连接点被压到这条边上。在图3-19c中，引入了一个三重结点[0, 1, 2, 3, 4, 4, 4, 5, 6, 7, 8]，这时曲线减少为两个片段。Q_4和Q_5在P_3处缩减为零。在Q_3和Q_6之间只有位置连续性，但是在控制点P_3两侧的片段是弯曲的。将这个结果与图3-13中用三重控制顶点产生的结果进行比较。在图3-19d中引入了一个四重结点[0, 1, 2, 3, 4, 4, 4, 4, 5, 6, 7]。位置连续性破坏了，曲线缩减到只有一个片段。为了理解这样的结果意味着什么，我们引入了另一个控制点，这样便出现了片段Q_7。这时在Q_3的末端和Q_7的起点之间出现了一个断裂。它们的控制点不同。

86
~
87

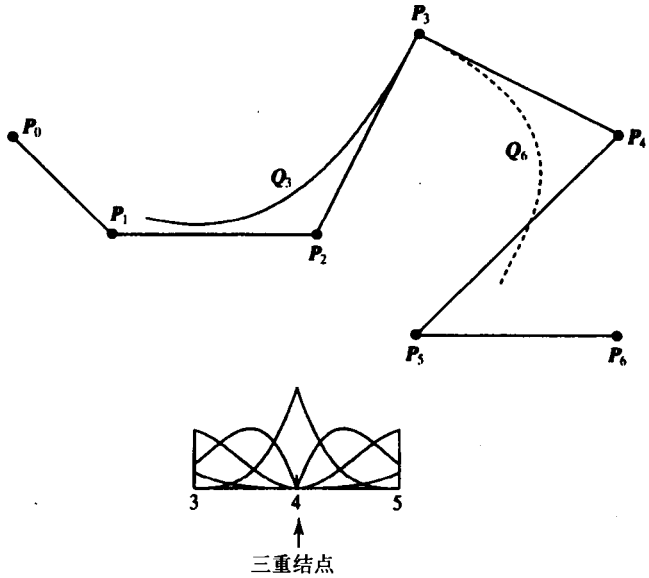


a) 四片段B样条曲线，结点向量为[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]，所有的B样条之间都是互相平移的

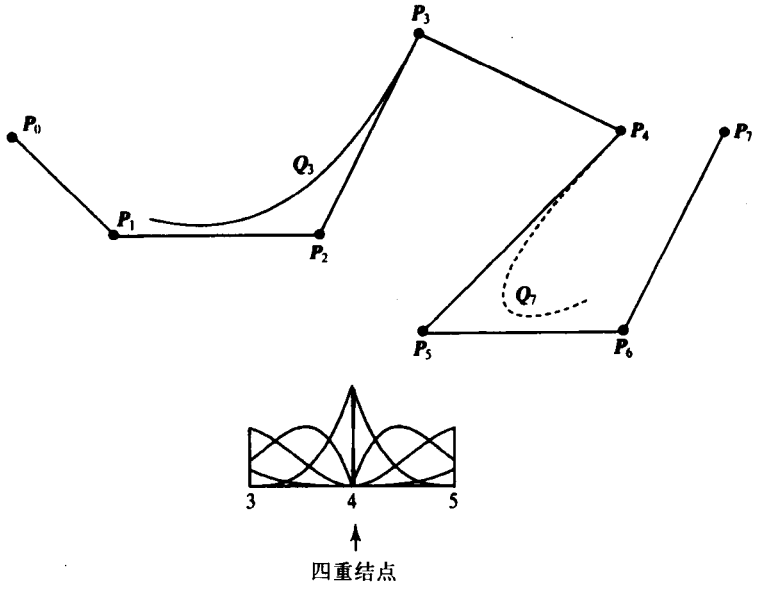


b) 结点向量为[0, 1, 2, 3, 4, 4, 5, 6, 7, 8, 9]，Q_4缩减为零

图3-19 在一条B样条曲线上内部结点重复的影响



c) 结点向量为[0, 1, 2, 3, 4, 4, 4, 5, 6, 7, 8]，Q_4和Q_5缩减为零。在Q_3和Q_6之间位置连续



d) 结点向量为[0, 1, 2, 3, 4, 4, 4, 4, 5, 6, 7]，曲线缩减成只有一个片段Q_3，引入了另一个控制点以表明这时P_3和P_4之间的曲线被分离了

图3-19 （续）

下面我们来考虑一种为非均匀B样条产生基函数或弯曲函数的递归方法。这个方法称为Cox-de Boor算法（Cox 1972; De Boor 1972）。令人惊奇的是该方法能够用一个递归的公式产生任意阶的均匀或非均匀B样条。由于函数之间不再是互相的平移，所以计算就更复杂一些。对于三次（四阶）曲线，可以以其展开形式定义该递归。将B样条表示法扩展，以第二个下角

标的形式引入阶数，我们把加权控制点P_i的基函数定义为$B_{i,j}(u)$，三次B样条的递归关系为:

$$B_{i,1}(u)=\begin{cases}1, & u_i \leqslant u \leqslant u_{i+1}\\ 0, & \text{其他}\end{cases}$$

$$B_{i,2}(u)=\frac{u-u_i}{u_{i+1}-u_i}B_{i,1}(u)+\frac{u_{i+2}-u}{u_{i+2}-u_{i+1}}B_{i+1,1}(u)$$

$$B_{i,3}(u)=\frac{u-u_i}{u_{i+2}-u_i}B_{i,2}(u)+\frac{u_{i+3}-u}{u_{i+3}-u_{i+1}}B_{i+1,2}(u)$$

$$B_{i,4}(u)=\frac{u-u_i}{u_{i+3}-u_i}B_{i,3}(u)+\frac{u_{i+4}-u}{u_{i+4}-u_{i+1}}B_{i+1,2}(u)$$

当结点被重复时，在Cox-de Boor定义中将出现商 0 / 0，而这一项被认为是零。从计算的角度来讲，总是要检查分子是否为零，当为其值为零时，不管分母如何一律将结果设定为零。在使用B样条的商业CAD系统中，对特定结点集的选择通常都是作为系统的预定义部分。

3.2.4 B样条曲线性质总结

前面列出的Bézier曲线的某些性质也适用于B样条曲线。尤其是:

- 曲线“遵循”控制点多边形的形状，并且被限定在由控制点组成的凸包之中。
- 曲线显示了变异消失性。
- 通过对其控制点表示应用任意仿射变换可以变换曲线。

除此之外，B样条还具有下面的性质:

88 ~ 90

- B样条曲线表现出局部控制性——一个控制点与四个片段相连（在三次时），而移动一个控制点可以只影响这几个片段。

3.3 有理曲线

有理曲线是在四维空间定义的曲线，该空间称为投影空间，接着再将其投影到三维空间中。我们首先来考虑有理Bézier曲线，以便与前面的论述相一致，接着讨论非均匀B样条的有理形式（称为NURBS），这是在实际使用中最通用的形式之一。与非有理形式比较有理形式的优点将在下面的论述中阐明。

3.3.1 有理Bézier曲线

我们从讨论将一条三维的Bézier曲线投影到二维空间开始，特别是投影到$z = 1$的平面（见图3-20）。通过除以$z(u)$来定义一个二维的（有理）曲线$\boldsymbol{R}(u)$:

$$\boldsymbol{R}(u)=\left(\frac{x(u)}{z(u)},\frac{y(u)}{z(u)}\right)$$

在三维（投影）空间所定义的曲线为:

$$\boldsymbol{Q}(u)=\sum_{i=0}^{3}\boldsymbol{P}_i B_i(u)$$

其中：

$$P_i=(x_i,y_i,z_i)$$

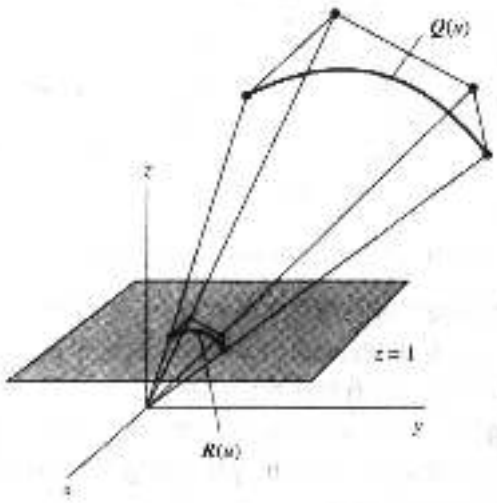


图3-20 一条三维Bézier曲线$Q(u)$的投影，投影平面$z=1$，得到二维曲线$R(u)$ 91

现在使用一种特殊的表示方式来描述二维空间中一条有理曲线的三维控制点，即

$$P_i=(w_ix_i,w_iy_i,w_i)$$

三维曲线表示为：

$$Q(u)=\sum_{i=0}^{3}\begin{bmatrix}w_ix_i\\w_iy_i\\w_i\end{bmatrix}B_i(u)$$

将其投影到二维空间为：

$$R(u)=\left(\frac{\sum w_ix_iB_i(u)}{\sum w_iB_i(u)},\frac{\sum w_iy_iB_i(u)}{\sum w_iB_i(u)}\right)$$

当我们将一条四维空间上的曲线投影到三维空间上时，其表示形式与上面的形式是相同的，这时，对每一个控制点：

$$P_i=(w_ix_i,w_iy_i,w_iz_i,w_i)$$

于是我们有：

$$R(u)=\frac{\sum w_iP_iB_i(u)}{\sum w_iB_i(u)}$$

有理曲线具有非有理曲线的所有性质，而且如果将每一个权重都设为是相同的，则这种表示形式就变为标准的Bézier曲线。下面讨论这种形式在实际应用中的重要性。这种形式允许给控制点设定权重，改变一个与某控制点相对应的权重的值的作用如图3-21所示。顾名思义，增加控制点的权重会使其影响更大。权重对曲线的影响与移动控制点所产生的影响有一些不同，见图3-22。改变控制点的位置，使得曲线上的点朝着与控制点位移所确定的方向平行的方向

移动。然而，改变控制点的权重则使得曲线上的点以图3-22中所示的方式向这一控制点移动。在设计程序中可以利用这些具有很明显的优点的因素。

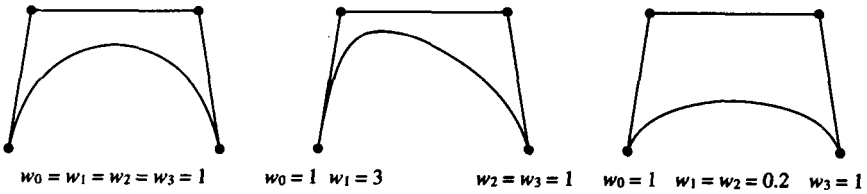
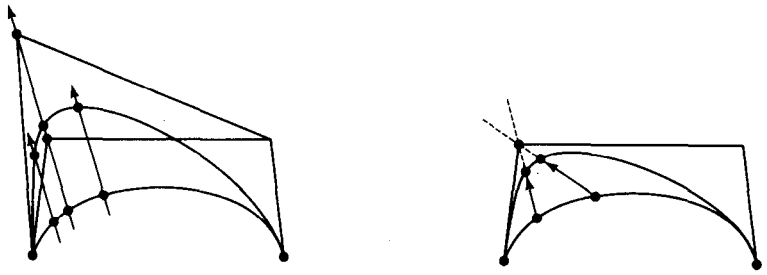


图3-21 改变一条有理Bézier曲线的权重

有理曲线的另一个重要性质是：可以用它来精确地表示二次曲线。圆或者圆上的片段是
92 在CAD中经常用到的曲线。而非有理Bézier曲线不能精确地表示圆。最后，正如我们已经提到的那样，只能以仿射变换的形式对非有理曲线的控制点进行变换。在计算机图形学中所使用的是透视变换而不是仿射变换。其控制点被变换到图像空间后的非有理曲线不能直接在这个空间中产生。然而对于有理曲线来说，由于其透视分割隐含在曲线的构造过程中，这就保证了可以从透视的观点正确地对其进行处理。



a) 控制点的移动引起曲线上的任一点都朝着与该移动平行的方向移动

b) 权重的调节使得曲线上的任一点沿着一条从曲线上的点到控制点之间连成的直线移动

图3-22 有理Bézier曲线：控制点移动和权重调节的不同效果

3.3.2 NURBS

NURBS代表非均匀有理B样条（Non-Uniform Rational B-Spline），这种表示可能在CAD中最通用。在设计应用程序中，可以有以下的可能性：

- 控制点的交互式放置和移动。
- 结点的交互式放置和移动。
- 控制点权重的交互式控制。

将前面小节中讨论的有理曲线的优点与非均匀B样条的性质相结合，可以把NURBS曲线定义为内部结点间跨度不均匀的结点向量上的非均匀B样条曲线。例如，可以有一些重数大于1的内部结点（即结点跨度为零）。一些像圆、圆柱等常见的曲线和表面需要非均匀的结点跨度。而这一选项的使用一般情况下允许对形状有较好的控制，并且可以具有建模较大的形状类的能力。

有理B样条曲线是用一个四维控制点的集合来定义的:

$$\boldsymbol{P}_i^w = (w_i x_i, w_i y_i, w_i z_i, w_i)$$

这样一条曲线在三维空间上的透视图称为有理B样条曲线。 93

$$\boldsymbol{R}(u) = H\left[\sum_{i=0}^{n} \boldsymbol{P}_i^w B_{i,k}(u)\right]$$

$$= \frac{\sum_{i=0}^{n} \boldsymbol{P}_i w_i B_{i,k}(u)}{\sum_{i=0}^{n} w_i B_{i,k}(u)}$$

有理B样条与非有理B样条具有相同的分析和几何特性。如果对于所有的i，$w_i = 1$，则:

$$R_{i,k}(u) = B_{i,k}(u)$$

与每一个控制点相对应的w_i称为权重，可以将其看成是附加的形状参数。可以看到，w_i只是局部地影响曲线。例如，如果对于所有的$j \neq i$，使w_i固定，w_i的改变仅在k个结点的跨度上影响到曲线（正像移动一个控制点只影响k个跨度上的曲线一样）。w_i可以从几何上解释为耦合因素。如果w_i增加，则曲线被推向控制点$\boldsymbol{P}_i$。如果w_i降低，则曲线从控制点移开。

有理B样条的一种特性就是在CAD中很重要的广义的二次曲线段。Faux and Pratt（1979）指出，有理四次形式可以产生一个单参数（w）的二次曲线段族。

3.4 从曲线到表面

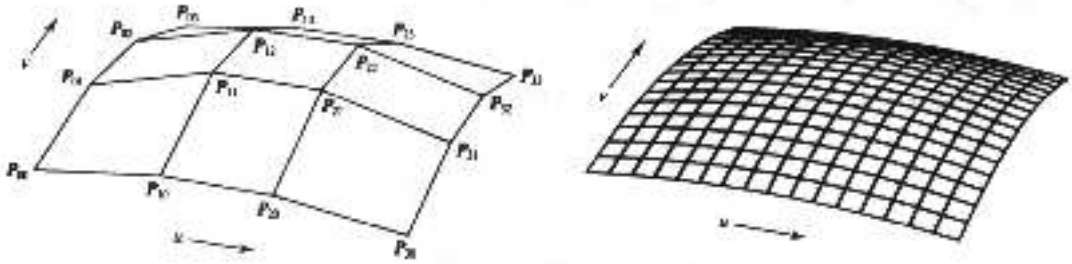
对于前面章节中给出的参数化三次曲线段的讨论可以很容易地推广到双参数三次表面曲面片中。表面曲面片上的点由一个双参数函数给出，对于每一个参数采用了一个弯曲函数或基函数的集合。三次Bézier曲面片定义为:

$$\boldsymbol{Q}(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} \boldsymbol{P}_{ij} B_i(u) B_j(v) \tag{3-4}$$

从数学上讲，三维表面被认为是由两条曲线的笛卡儿积而产生的。图3-23为一个Bézier曲面片及其控制点，图中曲面片用等参数线来表示。16个控制点形成一个控制多面体，这个控制多面体含有它与表面形状的关系的信息。这种关系与特征多边形和一条曲线段之间的关联是一样的。从图3-23a中可以直观地看出，控制点当中的12个与曲面片的边界有关联（而其中的四个点定义端点）。只有角上的顶点是处于表面上的。事实上，如果我们考虑用控制点形成一个4×4点的矩阵，则形成矩阵的边的四组四点的控制点就是曲面片边界曲线的控制点。于是， 94
曲面片的边由4条Bézier曲线组成。现在，我们可以看到，余下的四个控制点必定会用来定义包含在边界之间的表面的形状。

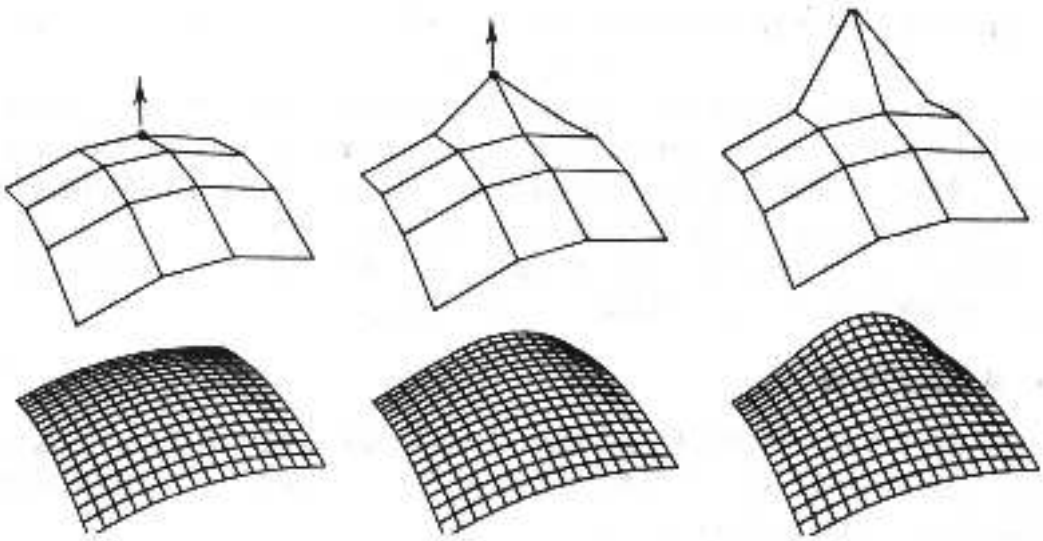
Bézier曲线公式的性质被扩展到表面范畴。图3-24展示了一个曲面片，它是通过“拉伸”一个控制点来定义的。保持了通过这一控制点的表面的直观感觉，同时也保持了确保一阶连续性的能力。通过对每个控制点进行变换也就进行了对表面片的变换。

控制点的“工作”方式可以通过与三次曲线进行对比来了解。其几何学上的解释自然比对于曲线的解释要困难。当然，Bézier公式的目的是为了使设计者免于处理切向量等等，但是为了保持完整性却保留了下来。



a) 一个控制多面体　　b) 所产生的双三次Bézier曲面片

图 3-23



95 图3-24 “举起”Bézier曲面片的一个控制点而产生的效果

方程（3-4）的矩阵表示为：

$$Q(u,v)=[u^3\ u^2\ u\ 1][B_z\ P\ B_z^T]\begin{bmatrix}v^3\\v^2\\v\\1\end{bmatrix}$$

其中：

$$B_z=\begin{bmatrix}-1&3&-3&1\\3&-6&3&0\\-3&3&0&0\\1&0&0&0\end{bmatrix}$$

$$P=\begin{bmatrix}P_{00}&P_{01}&P_{02}&P_{03}\\P_{10}&P_{11}&P_{12}&P_{13}\\P_{20}&P_{21}&P_{22}&P_{23}\\P_{30}&P_{31}&P_{32}&P_{33}\end{bmatrix}$$

考察一个曲面片角上的控制点与导数向量之间的关系是具有启发意义的。例如，考虑角$u = v = 0$，控制点和与顶点P_{00}相对应的向量之间的关系如下:

$$\begin{aligned} Q_u(0,0) &= 3(P_{10} - P_{00}) \\ Q_v(0,0) &= 3(P_{01} - P_{00}) \\ Q_{uv}(0,0) &= 9(P_{00} - P_{01} - P_{10} + P_{11}) \end{aligned} \tag{3-5}$$

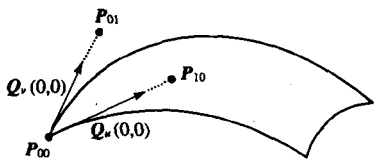
图3-25显示了一个曲面片角上的这些向量。$Q_u(0, 0)$为一个常量乘以在$Q(0, 0)$处u参数方向上的切向量。简单地讲，$Q_v(0, 0)$与v参数方向上的切向量有关。在每一个端点处的交叉点导数，有时称为扭向量，定义了分别对应于u和v的切向量的变化率。这是一个含切向量的平面法向量。

与在Bézier曲线中的控制点类似，曲面片由四个端点、八个切向量（每个角上两个）以及四个扭向量来定义。现在来考虑图3-25b，该图展示了在导数中所涉及的控制多面体的元素。四对点定义每一个角u方向上的切向量（矩阵中的两行），四对点定义v方向上的切向量（矩阵中的两列），所有16个元素定义扭向量。

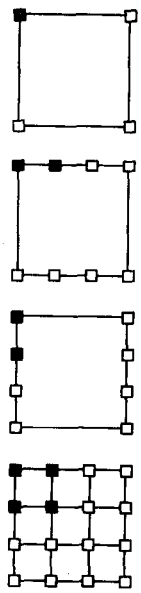
如果设$Q_{uv}(i, j) = 0$，则可以得到一个所谓的零扭表面，或者说是一个具有四个零扭向量的表面。对于这样的表面，可以用三个相邻的边界点导出内部控制点。例如，在角（0，0）处，我们有:

$$0 = 9(P_{00} - P_{01} - P_{10} + P_{11})$$

如果只有边界曲线的信息而希望导出曲面片的16个控制点时，这个性质是重要的。这种情况在进行表面拟合或插值的时候出现（见3.6.3节），这时我们希望通过一组三维空间中的点来拟合一个曲面片。我们首先用曲线插值来定义曲面片的边界曲线，得到一个曲面片的12个控制点，再按一定的方式估计四个内部控制点。而零扭解决方案是一种对四个内部控制点进行评估的非常容易的方法。 96



a) P_{00}处的切向量



角点

在v方向上的切向量:
例如，$Q_v(0, 0) = 3(P_{01} - P_{00})$

在u方向上的切向量:
例如，$Q_u(0, 0) = 3(P_{10} - P_{00})$

扭向量:
例如，$Q_{uv}(0, 0) = 9(P_{00} - P_{01} - P_{10} - P_{11})$

b) P_{00}处向量中所涉及的控制点矩阵中的元素

图3-25 P_{00}处的向量

对于进行明暗处理计算，我们需要一些表面法向的计算。最简单的计算明暗处理的方法之一是将曲面片进行分割，直到分割出来的产品近似为平面（这一技术将在第4章中完整地进行介绍）。然后，可以将曲面片以平面多边形进行处理，再应用Gouraud或Phong的明暗处理模型。由顶点处的两个切向量的叉积来计算顶点法向。例如:

$$\boldsymbol{a}=(\boldsymbol{P}_{01}-\boldsymbol{P}_{00})$$
$$\boldsymbol{b}=(\boldsymbol{P}_{10}-\boldsymbol{P}_{00})$$
$$\boldsymbol{N}=\boldsymbol{a}\times\boldsymbol{b}$$

可以在表面的任意点上通过计算两个偏导数 $\partial Q/\partial u$ 和 $\partial Q/\partial v$ 的叉积来计算法向，但是，通过对由参数描述的内部点的穷举计算来计算曲面片的明暗处理从计算的角度来看代价太大，
97 而且还会遇到其他问题。用参数化的曲面片来描述表面的优点并不能掩盖这样的事实，即准确的世界坐标对于表面上的每一个点都是有效的，而检索这类信息的代价一般来讲太高了。但是，参数化表示有这样的优点，即曲面片表示有利于物体的建模。

3.4.1 连续性和Bézier曲面片

Bézier表示对于单个的曲线段和单个的曲面片表面是很优秀的。当我们希望构造一条更复杂的曲线（或一个表面）时，必须用连续性的约束把Bézier曲线连接起来。在本章的最后，将详细地讨论这样的问题。在这一小节中，我们考察一个对于建模来说很关键的问题，即按什么方式把曲面片连接在一起，并保持曲面上的连续性。像把Bézier曲线段连接起来一样，本小节将建立一个与其相似的变量。

在两个曲面片的交界处保持一阶导数连续性是对于曲线连接约束的一种简单扩展，并且已经被人们从几何上认真研究过。图3-26显示了两个曲面片S和R，它们共用一条边。对于位置连续性或零阶导数连续性，有:

$$\boldsymbol{S}(1,v)=\boldsymbol{R}(0,v)\ ,\ 0<v<1$$

这个条件暗示，两个特征多边形分享一条公共的边界边（见图3-27），并且:

$$\boldsymbol{S}_{33}=\boldsymbol{R}_{03}$$
$$\boldsymbol{S}_{32}=\boldsymbol{R}_{02}$$
$$\boldsymbol{S}_{31}=\boldsymbol{R}_{01}$$
$$\boldsymbol{S}_{30}=\boldsymbol{R}_{00}$$

或者:

$$\boldsymbol{S}_{3i}=\boldsymbol{R}_{0i}\ ,\ i=0,\cdots,3$$

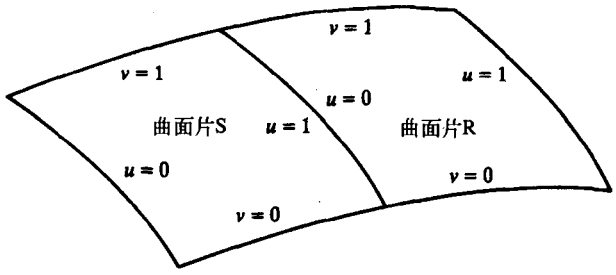
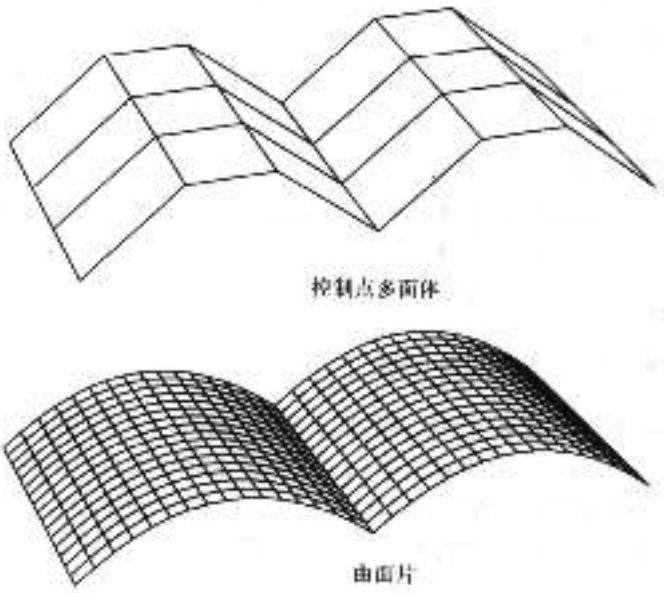


图3-26 连接两个曲面片

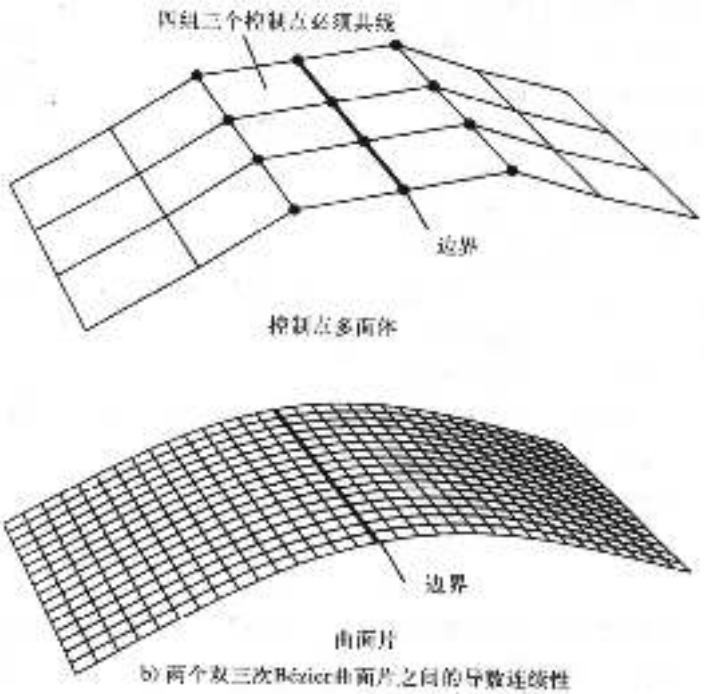
为了满足一阶导数（C^1）连续性，对于第一个曲面片$u = 1$处的切向量必须与第二个曲面片上对于所有的v、在$u = 0$处的切向量相匹配。这就是说，跨跃边界的多面体的四对边中的每一对都必须是共线的。即：

98

$$(\boldsymbol{S}_{3i} - \boldsymbol{S}_{2i}) = k(\boldsymbol{R}_{1i} - \boldsymbol{R}_{0i}),\ i = 0, \cdots, 3$$



a) 两个双二次Bézier曲面片之间的位置连续性



b) 两个双三次Bézier曲面片之间的导数连续性

图 3-27

Faux在1979年指出，在CAD应用中，如果合成表面是由一组Bézier曲面片构成的话，则对构成所需的约束是苛刻的。例如，合成表面可能是由先建立一个曲面片，然后再对其进行加工而得到的。沿着一个公共的边界连接两个曲面片所隐含的条件是第二个曲面片的八个控制点都已经固定，而把一个曲面片与已有的两个曲面片相连接则意味着已固定了12个控制点。

一个稍微放宽一些的连接条件是由Bézier在1972年提出的。在这样连接起来的曲面片中，角具有位置连续性但没有导数连续性。但是，在角上相遇的各个边的切向量必须是共面的。即便具有了在边缘上的较大灵活性，在合成表面的设计中也仍然会有问题。

必须注意到，尽管前面所述涉及的是矩形曲面片，但这样的曲面片不能表示所有的形状。例如，考虑一个具有球面形状的物体。这时在两极的位置必须使矩形退化成三角形。Farin指出，也许在大多数CAD系统中矩形曲面片占有支配地位的主要原因是，在汽车设计中曲面片的第一次使用是对外部车体板的设计。汽车的这些部分几何上具有矩形形状，所以很自然地用更小的矩形将其分割，并且使用了具有矩形形状的曲面片。

3.4.2 一个Bézier曲面片物体——Utah茶壶

也许在计算机图形学中最著名的物体是所谓的Utah茶壶了，这是一个Bézier曲面片网格的早期例子。在这一小节中，我们将对这个人们多次讨论的物体进行描述，并用它来阐明一个有关这种表示形式的重要观点，即它的经济性（与多边形网格模型相比较）。

犹他大学在20世纪70年代早期曾是绘制算法的研究中心。他们手工建立了各种各样的多边形网格模型，包括VW甲壳虫，并在1971年由Ivan Sutherland的计算机图形学班级对其进行了数字化（见图2-4）。

1975年，M. Newell开发了Utah茶壶。这是一个非常著名的物体，后来成为计算机图形学中的一类基准物。在本书中我们也经常提到它。Newell通过先绘制出茶壶的轮廓，对其上的双三次Bézier曲面片估算适当的控制点来完成建模。茶壶的壶盖、边以及壶身作为旋转的实体来处理，而壶嘴和壶把建模为延展的实体。这样做最终产生了32个曲面片。

原始的茶壶现在放在波士顿计算机博物馆中，其旁边展示的是计算机。这个模型的完整描述及其在计算机博物馆中的详细解说可在Crow（1987）中找到。

99
~
100

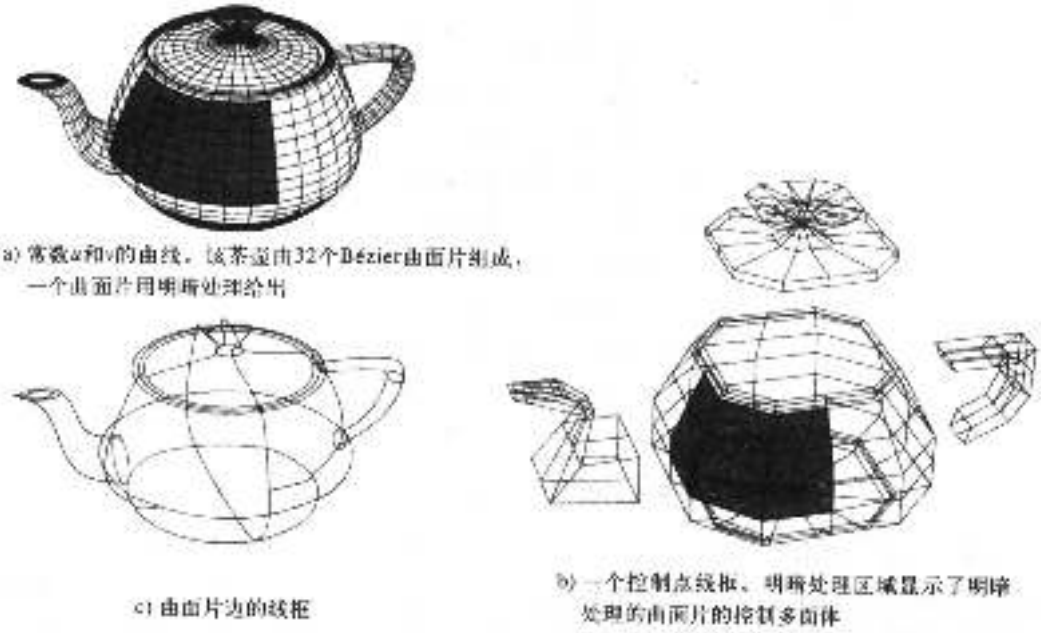
图3-28显示了常数u和v的线框图，物体由32个Bézier曲面片组成。其中一个曲面片以加粗的线来表示（在这个图中还显示了由多条Bézier曲线构成的线框图像，这些Bézier曲线形成曲面片的边。同时显示了一个合成的控制点多面体）。这个表示由下式组成:

32个曲面片 × 16个控制点/曲面片
= 288个顶点（基本上，大多数曲面片间共享12个控制点）
= 288 × 3个实数（假设）

另一方面，一个“合理的”多边形网格表示将需要:

大约2048 × 4个多边形
= 2048 × 3个实数

因此，多边形网格模型（一种不精确的表示）使用2048/32倍于基元表示所用的空间。这是第2章中所述观点的一种较好的演示，也是近二三十年来，三维计算机图形学中一直存在宁愿使用基元而不是更复杂的模型的原因。



a) 常数u和v的曲线。该茶壶由32个Bézier曲面片组成，一个曲面片用明斯处理给出

c) 曲面片边的线框

b) 一个控制点线框。明斯处理区域显示了明斯处理的曲面片的控制多面体

图3-28 Utah茶壶

3.5 B样条表面的曲面片

为了形成双三次B样条表面的曲面片，需要计算：

$$Q(u,v)=\sum_{i=0}^{n}\sum_{j=0}^{m}P_{ij}B_{i,j}(u,v)$$

101

其中，P_{ij}为控制点数组，$B_{i,j}(u,v)$为双变量的基函数。可以从下式产生$B_{i,j}(u,v)$：

$$B_{i,j}(u,v)=B_i(u)B_j(v)$$

其中，$B_i(u)$和$B_j(v)$均是预先定义的单变量三次B样条。于是有：

$$Q(u,v)=\sum_{i=0}^{n}\sum_{j=0}^{m}P_{ij}B_i(u)B_j(v)$$

与B样条曲线相同，我们将B样条曲面片看成是由几个矩形表面的曲面片段组成。现在，我们有u和v方向上的两个结点序列，两者一起形成参数空间中的网格。

下面考虑均匀的B样条曲面片，其中，结点值的网格在u和v参数方向上是等间距的。首先考虑一个曲面片段，我们将用这个词来代表在两参数空间中的实体。此两参数空间实体与单参数空间的曲线段相似。由此，可以说一个一般的B样条表面的曲面片由几个曲面片的片段构成，正像一条B样条曲线是由一些曲线段组成的一样。在一个B样条曲线段的情况下，需要有四个控制点来定义片段。扩展到两参数空间，需要一个具有4×4个控制点的网格P_{ij}来形成一个曲面片。这些控制点与4×4个双变量基函数融合。请回忆一下3.2.2节，在那一节中，一个B样条片段需要一个具有8个结点值$u_0,\cdots,u_7$的向量来表示，因此，一个曲面片段需要一个具有8×8个结点值的网格或结点数组（见图3-29）。双变量基函数在用一个小方块表示的结点值处取得峰值。

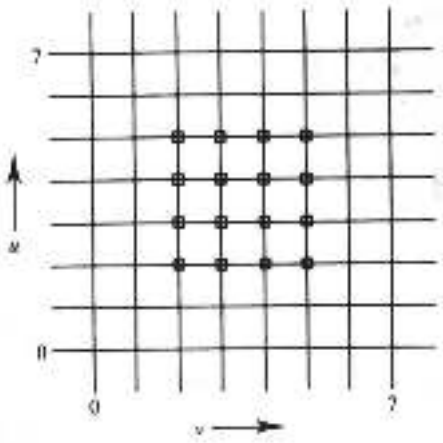


图3-29 16个双变量B样条在参数空间中所示的点处取峰值

考虑一个简单的例子。图3-30显示了一个B样条曲面片，它由16个控制点定义。请注意，这
102 个曲面片被限定在靠近内部中心处的四个控制点的范围之内。正像B样条曲线那样，它不会内插其控制点。B样条曲面片段既不会内插其四个内部控制点，也不会内插其外部的12个控制点。

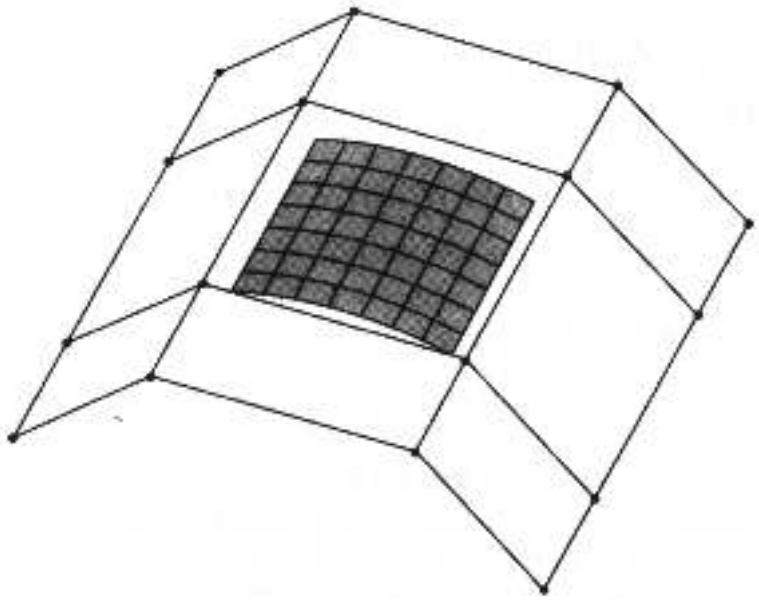


图3-30 一个B样条曲面片段

通过使用多重顶点（正像用多重的端点控制曲线一样）来控制曲面片在控制点多面体边界处的行为。这个任务可以很容易地由一个简单的例子来演示。如图3-31所示，我们对一组边界顶点重复了三次，这些顶点形成一个控制点矩阵，它由24个点组成。这样就形成了一个三个片段的曲面片，也就是说，将曲面片推向了边界顶点。我们注意到，实际上没有哪一个顶点被插值。

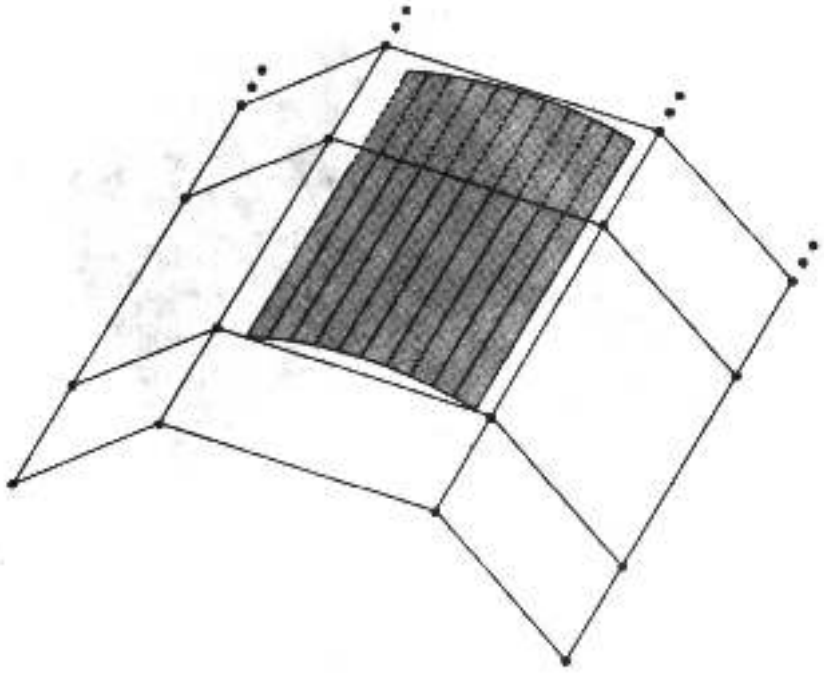


图3-31 对于图3-30中的例子将一行控制点重复二次

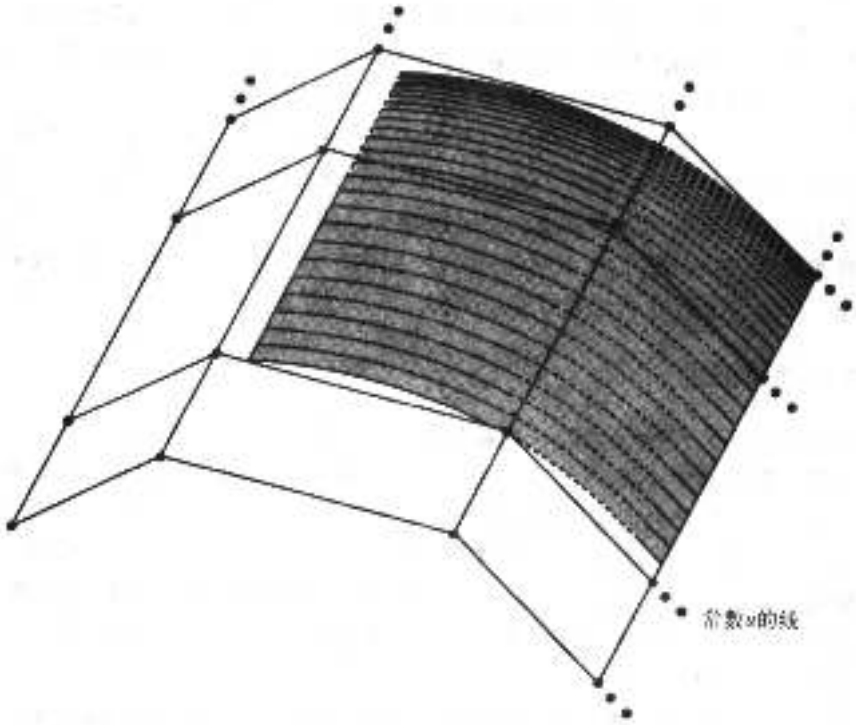


图3-32 通过对一行控制点和一列控制点各重复二次所形成的一个九个片段的B样条曲面片

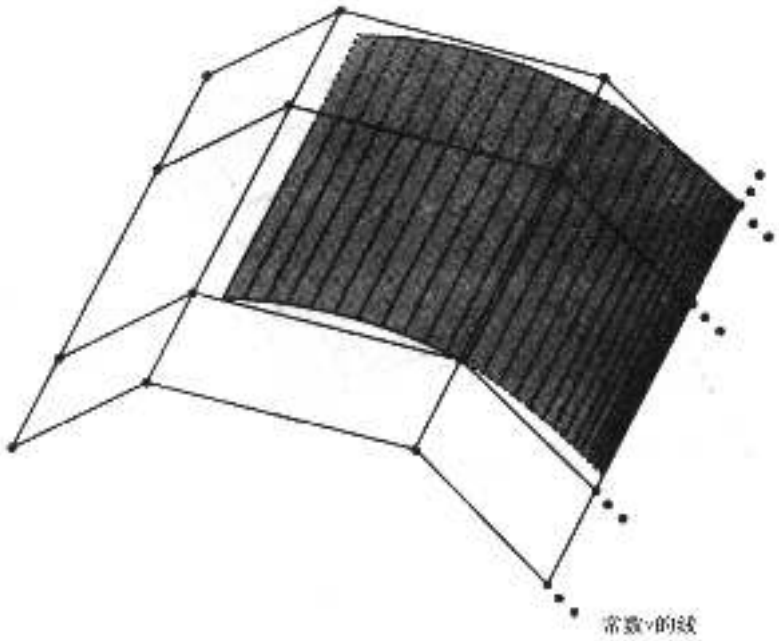


图3-32（续）

在第二个例子中（见图3-32），我们对两组边界顶点重复了二次，其中之一形成一个共线集。这样做形成了一个九个片段的曲面片，这时共线的顶点被插值了。如果将所有的边界顶点都重复二次，则会产生一个25个片段形成的曲面片，一般情况下这只会引起角上的点插值。

双重或三重内部控制点产生的建模效果比Bézier曲面片所产生的效果更好。图3-33是一个例子。控制点多面体如前面的例子一样，但是有一个内部点被向上移动了。一行和一列控制点被重复了二次。对于列，控制顶点中的三个顶点是共线的，这就导致一个带有折缝的表面，这个折缝沿着控制点多面体中相应的边而排列。现在考虑三重的行。在行上，点不是共线的，折缝的效果不是太明显。很清楚，这些影响因素可以结合到一个建模程序中，在这个模型中，行和列都被重复了三次，控制点多面体的整个边被交互式地移动，用于定义或者推拉一个表面上的折缝。含有折缝的表面是常见的，例如汽车车身。

3.6 建立曲面片表面

建模问题包括从头开始建立起一个参数化的网格描述，以及编辑或者改变一个现有描述的形状。采用一种曲面片表示方法的目的之一是获得通过改变其形状就可以“雕刻”一个现有的表面的能力。用一个曲面片来进行建模，比如说用一个Bézier曲面片，是简单直接的。但是，当处理一个曲面片组成的网格时会遇到很严重的困难。这个问题很大程度上仍然在研究之中，正是由于这个原因，大多数文献所涉及的研究内容都不在本文所论述的领域范围之内。然而，这又是一个非常重要的领域，我们将尽可能全面地覆盖这一领域。在这一节中，我们将讨论下面的设计或创建方法：

1）截面设计：在这里，我们将考虑前面章节曾经描述过的简化的扫掠技术，限制物体为线性轴设计。

2) 通过处理控制点多面体来进行交互式设计。

3) 从一组代表真实物体（通常情况下如此）的三维点，创建一个曲面片的网格，这个方法称为表面插值或表面拟合。

103
↓
106

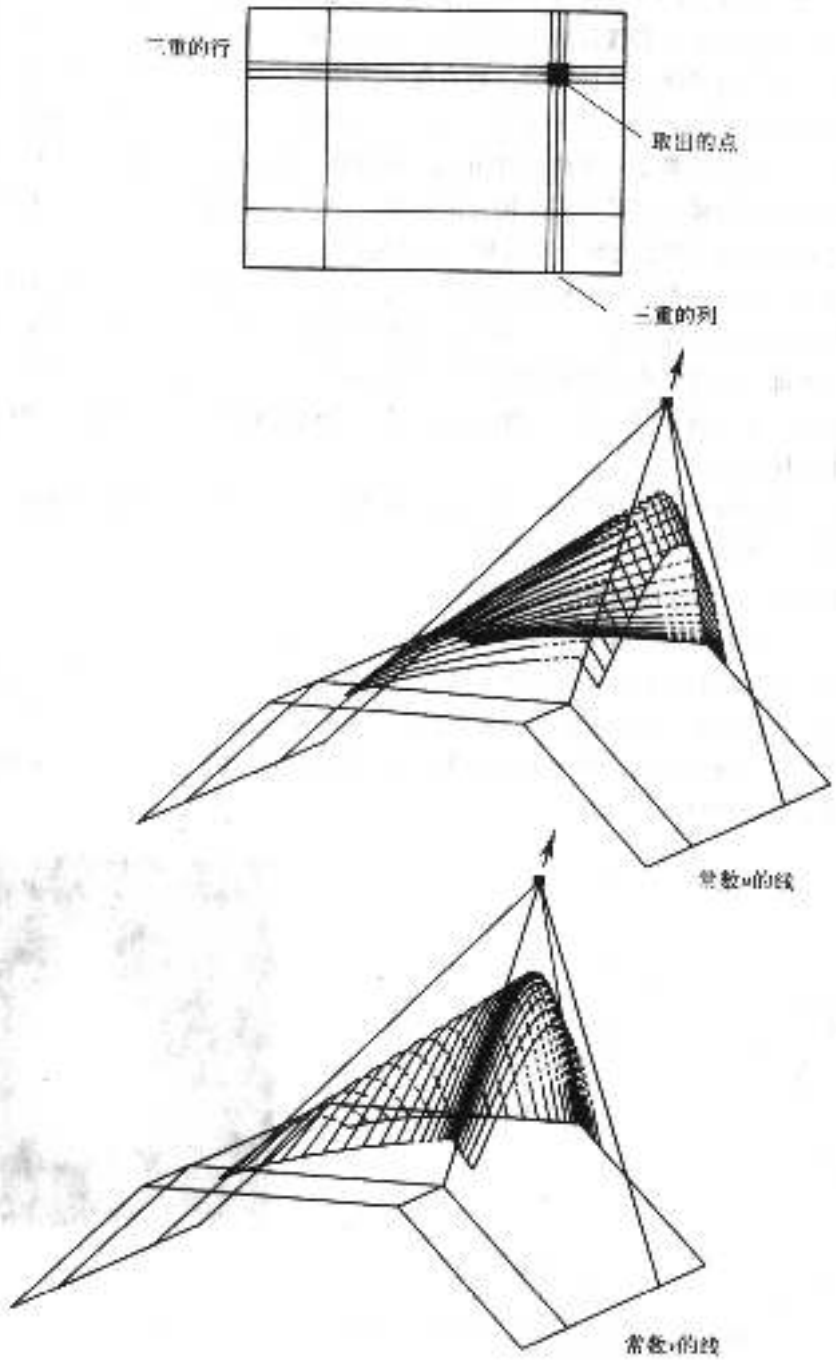


图3-33 三重内部控制点产生一个带有折缝的曲面片

3.6.1 截面或线性轴设计实例

我们将考察一个例子，在这个例子中设计约束使我们能够用曲线导出一个由曲面片的网格构成的物体。

考虑使用八个曲面片的表面来设计容器或碗，容器的截面只有四折对称性。也就是说，最终的物体可以被认为是沿着线性轴或者脊线，扫掠某种截面而形成的。许多工业物体都可以归入这一类。我们将对这一类设计的三种表现形式建立一个层次结构，其中每一层上的截面变化特性的复杂度都会增加。这只是对于上一章中所描述的扫掠技术的一种重复，但是，这一次建模操作产生的结果是一个曲面片组成的网格而不是多边形网格。在这里，我们将讨论局限在线性脊线的范畴，尽管我们所描述的适用于多边形网格的一般扫掠技术也可以扩展到曲面片，但是应用时可能会遇到一些困难，而这超出了本书的范围。

曲面片模型（而不是多边形网格表示方法）在本书中可能是重要的，这是因为我们需要的是进行全局设计形状改变的能力，还因为我们想要得到物体的高质量的可视化。像瓶子这样的物体，其大部分形状是由其轮廓边的投影所确定的，我们并不想进行轮廓边的分解，而用多边形网格模型来可视化时可能会用到分解。为了合理地对形状进行控制，我们决定能够忍受的最少曲面片的个数为8。

现在讨论所需的可能性和交互的协议。图片说明了用八个曲面片构造的模型，所给出的公式在每一种情况下都与一个曲面片有关系。

1. 线性轴设计——按比例缩放的圆截面

在这里，我们只有一个圆截面，可以设计的物体是图3-34中所示的形状（实际上，Bézier曲线只能近似一个完整的圆，但是我们将忽略这个复杂的过程）。为导出本例中的八个曲面片，我们只需与两个片段的Bézier曲线交互。这条曲线被称为轮廓曲线（profile curve），它给出两个曲面片的控制点，而其他六个曲面片的控制点是通过对称性得到的。这类物体作为实体旋转的产物也是众所周知的。

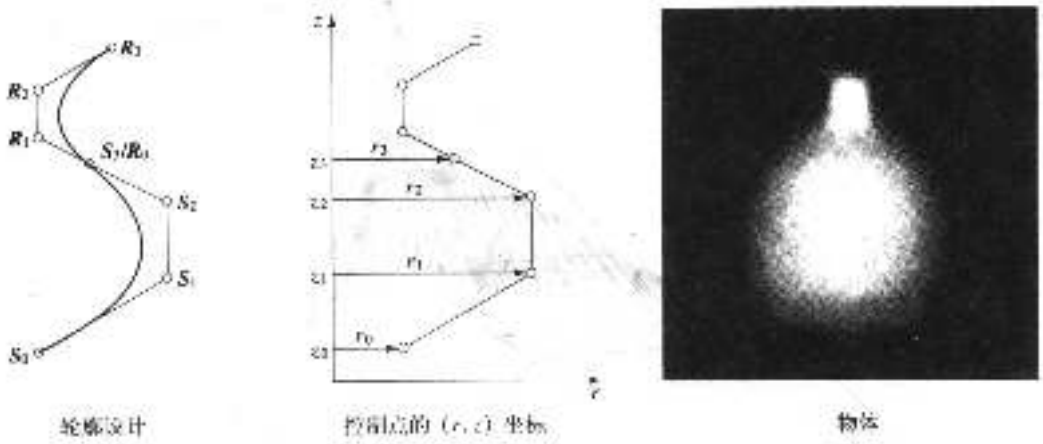


图3-34 线性轴设计——圆形截面，物体由一条轮廓曲线设计

107 如果把z轴作为物体的脊线，则底面曲面片的控制点由下式给出：

$$\begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} = \begin{bmatrix} T_{00} \\ T_{01} \\ T_{02} \\ T_{03} \end{bmatrix} [r_0\ r_1\ r_2\ r_3] \begin{bmatrix} k \\ k \\ k \\ k \end{bmatrix} [z_0\ z_1\ z_2\ z_3] \tag{3-6}$$

其中T_{00}、T_{10}、T_{20}、T_{30}为形成截面圆的四分之一圆弧的控制点。

$$T_{00} = (0,1,0)$$
$$T_{10} = (c,1,0)$$
$$T_{20} = (1,c,0)$$
$$T_{30} = (1,0,0)$$
$$c = 0.552(\text{约数})$$

表面曲面片的控制点在z轴坐标为z_0、z_1、z_2、z_3，半径为r_0、r_1、r_2、r_3的圆上，片段S_0S_1、S_1R_1和R_2R_3扫掠，形成截短的圆锥体。设计的表面与顶部、底部和连接截面处的那些圆锥体部分正切。同样，可以导出顶部的曲面片。

2. 线性轴设计——按比例缩放的非圆截面

现在，我们允许四分之一的截面可以是任意形状。使用前面用到的轮廓曲线来进行设计，并且也使用一条用于四分之一截面的曲线（见图3-35）。截面保持其形状，只是在尺寸上进行改变。

底面曲面片的控制点由方程（3-6）给出，而T_{00}、T_{10}、T_{20}、T_{30}都是从截面设计得到的，而不是预定义的。

108

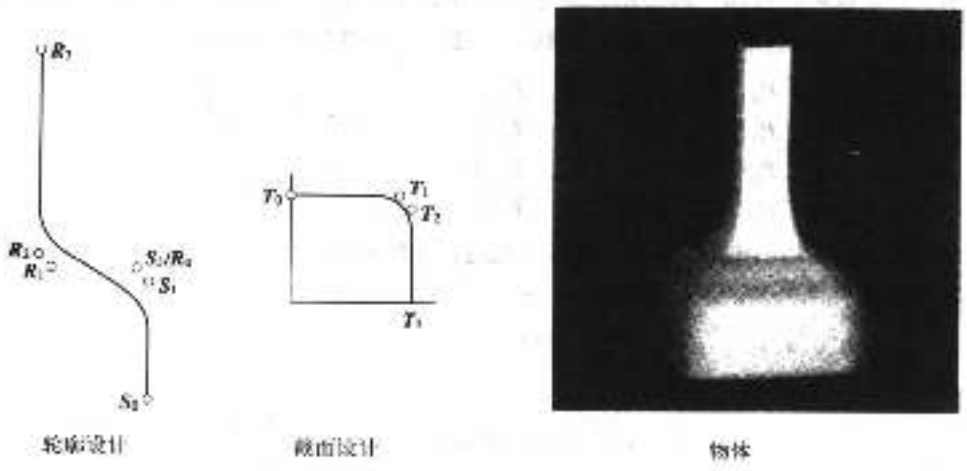


图3-35　线性轴设计——按比例缩放（非圆）截面，物体是由一条轮廓以及一条（1/4）截面曲线来设计的

3. 线性轴设计——非圆变化截面

对于弯曲不同的截面曲线，有很多可选择的方案。一种容易的方法是将对于轮廓曲线的弯曲限制到一个片段上，比如说限制在上部。这样，在本例中上面的四个曲面片将显示出变化的截面，而下面的四个曲面片将有一个与前面例子中一样的恒定的截面。

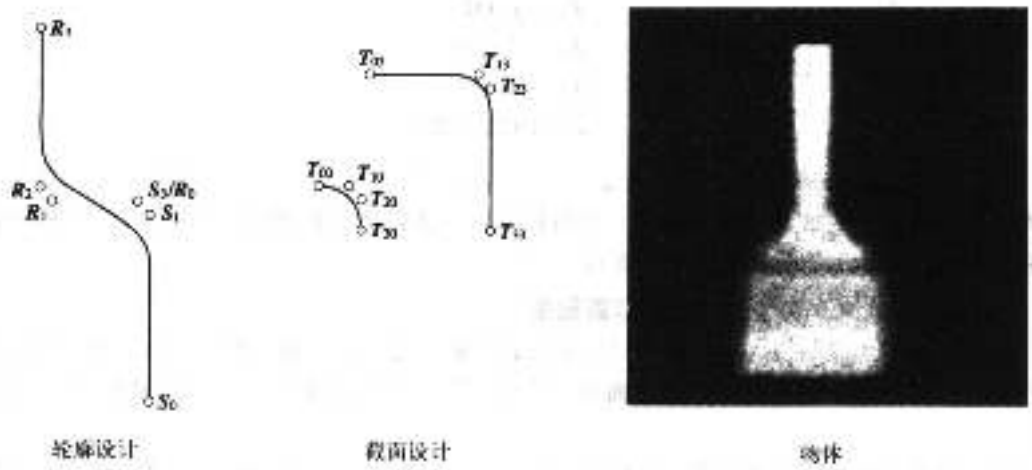
在这里，我们允许截面的形状发生变化。设计三个截面，这三个截面形成顶部曲面片的

顶边，由两个曲面片共享的顶部/底部公共边以及底部曲面片的底边（见图3-36）。现在我们需要定义中间的曲线来形成曲面片的顶部和底部曲线。这可以通过简单地按下式进行定义来完成：

$$Q(u, v) = Q(u, 0)(1 - r(v)) + Q(u, 1)r(v)$$

其中：

$$r(0) = 0 \text{ 和 } r(1) = 1$$



轮廓设计　　截面设计　　物体

109 图3-36 线性轴设计——非圆变化截面，物体是由一条轮廓线和三个截面设计的

图3-37是这种过程的一种表示。从这种表示中可以看出，曲线$Q(u, v)$有一个特征多边形，其控制点位于两个截面的控制点的连接线上。这时，曲面片的控制点由下式给出：

$$\begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} = \begin{bmatrix} T_{00} & T_{01} \\ T_{10} & T_{11} \\ T_{20} & T_{21} \\ T_{30} & T_{31} \end{bmatrix} \begin{bmatrix} 1 & 1-r_1 & 1-r_2 & 0 \\ 0 & r_1 & r_2 & 1 \end{bmatrix} \begin{bmatrix} k \\ k \\ k \\ k \end{bmatrix} [z_0\ z_1\ z_2\ z_3]$$

其中，T_{00}、T_{10}、T_{20}、T_{30}为第一个截面（底边）的控制点，而T_{02}、T_{12}、T_{22}、T_{32}为第二个截面（两个曲面片的公共边）的控制点。

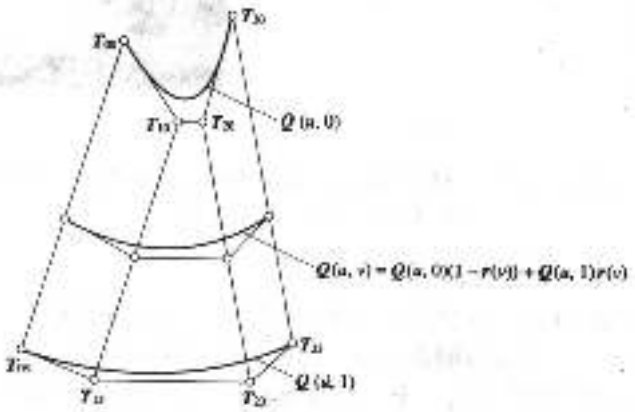


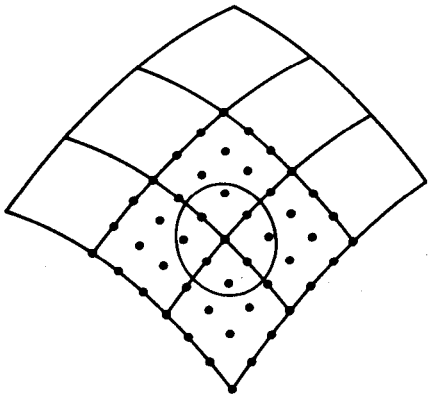
图3-37 用轮廓曲线来弯曲两个不同的截面

3.6.2 控制多面体设计——基本技术

这种用参数化的曲面片物体进行建模的方法在大多数教科书中都有介绍。其主要思想是已知一个现有的曲面片模型，设计者用一个循环（loop）进行交互，这个过程使得控制点被移动，而移动的结果产生一个新的表面。有时，这称为“自由形状雕刻”。设计者可以移动一个或者一些控制点，这样来建模表面就好像用一些有韧性的材料一样，比如用黏土。我们把提供一个三维的编辑系统来实现对控制点的移动这样的实际应用问题先搁置不提，先讨论两个基础性的问题。

首先，当我们正在与一个曲面片网格进行交互而不是与单个曲面片进行交互时，应该怎样做？尽管单个曲面片设计可以覆盖一些实际问题（例如，汽车车身），但我们可能需要考虑曲面片的网格。问题是不能只移动单个曲面片上的控制点而不考虑保持其与周围曲面片之间的连续性。 110

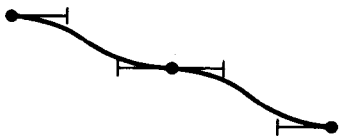
我们针对3.1.1节中讨论的曲线来讨论这个问题。当直接把这个方法扩展到曲面片上时，需要把9个控制点作为一个组来移动，如图3-38a所示。这就自动地保证了曲面片的连续性，但是这样做会向表面引入一些辅助平台。这很容易用曲线看出来。图3-38b显示了一个两段的Bézier曲线。如果移动共线的三组控制点，则会得到分段弯曲的台阶的效果，如图3-38c所示，而我们希望的是图3-38d中所示的形状。



a) 四个相连的Bézier曲面片及其控制点。连续性约束指在不考虑相邻的8个点的情况下不能移动其中心点。9个点可以一起移动而保持连续性



b) 未变形的两段曲线



c) 通过移动在共线组中的控制点而变形的曲线

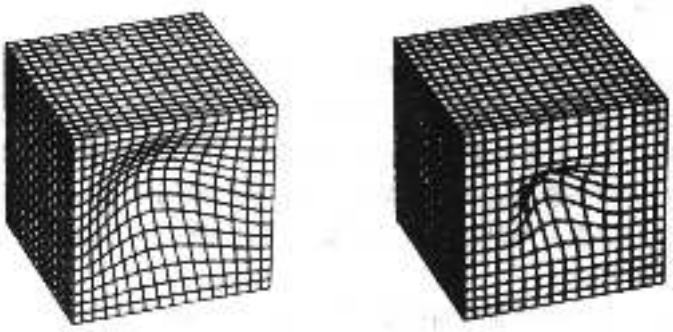


d) 希望的形状

图 3-38

1. 控制多面体设计——精细控制

另一个出现的问题是控制的局部性。我们已经讨论了这个问题对Bézier曲面片与B样条曲面片的不同影响。现在，我们更详细地来讨论B样条的问题。困难集中在对于能够变形的尺度的需求上。尽管移动一个控制点只会改变那些与这个点相连的曲面片，但是在物体空间中变形的尺度却与曲面片的大小有关。这就表明，如果需要精细的变形则可以在变形的区域内局部地细分曲面片来控制变形的尺度。这种方法称为分层的B样条变形（Forsey and Bartels 1988）。图3-39为这种技术的一个简单的例子。在立方体一侧，变形的尺度与构成该面的曲面片的个数有关。



4个曲面片/表面　　16个曲面片/表面

图3-39 把变形的尺度作为表示一个立方体的侧面所使用的曲面片（初始时为平面）的个数的函数

考虑对B样条曲面片的定义：

$$Q(u,v)=\sum_{i=0}^{n}\sum_{j=0}^{m}P_{ij}B_i(u)B_j(v)$$

这个方程可以用将结点插入到一个曲面片（Farin 1990）的方法来重新定义：

$$Q(u,v)=\sum_{i=0}^{N}\sum_{j=0}^{M}R_{ij}B_i(u)B_j(v)$$

111

其中：

$$N>n\text{和}M>m$$

新的控制点的导出方法如Forsey and Bartels（1988）所述。问题是如何应用这个策略来处理我们所感兴趣的表面上的一个区域。Forsey和Bartels通过定义一个最小表面来完成这个工作，该最小表面是可以适用控制点的细化的表面上的最小区域。这个最小表面需满足下面的两个约束：

1) 新控制点的移动产生局限在这个最小表面上的变形。

2) 在最小表面的边界处的导数值保持不变。

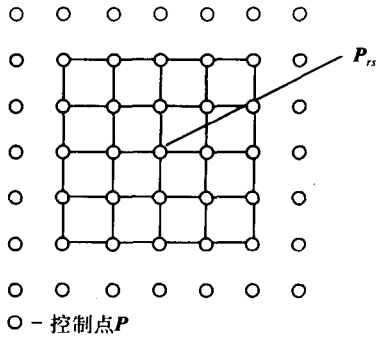
这意味着在细化的表面上的变形将不会影响导出这个变形表面的那个大一些的表面而且各处的连续性也得到保持。提出这种精细的方法的目的是只在需要的地方才影响控制点的细分，而另一种方法则是在整个表面上进行控制点的细分。这个过程可以在细分的表面内重复进行，直到达到一个精细控制的满意水平为止。因此这个方法称为层次化的。

最小表面有16个曲面片，由7×7个控制点矩阵定义（见图3-40a）。如果将中间的四个曲面片细分为16个，所需的控制点如图3-40b所示。在此需注意，原始的3×3个控制点由所细分的曲面片共享。创建了一个动态的控制点的数据结构，在这个结构中原始表面处于控制点分层树的根部。对表面进行编辑引起对树的遍历，这种结构的重点之一是遍历可以在两个方向上进行。在同一表面区域上粗略的细分可以带有以前的精细的细分。这是因为控制点的表示是以相对于局部参考框的偏移来定义的。

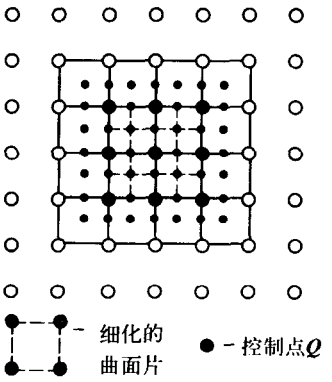
112

2. 控制多面体设计——粗略控制

现在我们考虑与前面的例子相反的情况——粗略控制。假如我们对全局形状变形感兴趣，例如，取一个圆柱或管状的物体，将其弯曲成一个螺旋形的物体。这时，我们可能需要在某种程度上同时对所有的控制点进行操作。



a) 具有49个控制点、16个曲面片的最小表面



b) 中心的4个曲面片细分为16个曲面片

图3-40 动态的控制点（Forsey and Bartels（1988））

考虑应用于一条由四个控制点P_i定义的曲线$Q(t)$的策略。首先，将这条曲线封闭在一个单位正方形中，然后将这个区域分割成规则的点R_{ij}（$i=0,\cdots,3, j=0,\cdots,3$）的网格，如图3-41所示。如果把这个正方形看作是uv空间，则可以写作：

$$(u,v)=\sum_{i=0}^{3}\sum_{j=0}^{3}R_{ij}B_i(u)B_j(v)$$

将这个恒等方程与双三次参数化Bézier曲面片的方程相比较。这个恒等方程遵从多项式$B_i(u)$和$B_j(v)$的线性相关性质。它表示了这样的事实，即一组共面的控制点将定义一个平面曲面片。如果R_{ij}点的网格这时被变形为网格R'_{ij}（见图3-41），则点(u,v)将被映射到点(u',v')。

$$(u',v')=\sum_{i=0}^{3}\sum_{j=0}^{3}R'_{ij}B_i(u)B_j(v)$$

可以用这个方程导出用于新曲线$Q'(t)$的一组新的控制点P'_i。

这是一种全局化地直接改变曲线形状的方法。我们把曲线植入到一个平面曲面片中，通过移动这个曲面片的控制点来使曲面片变形，同时也改变了曲线的形状。

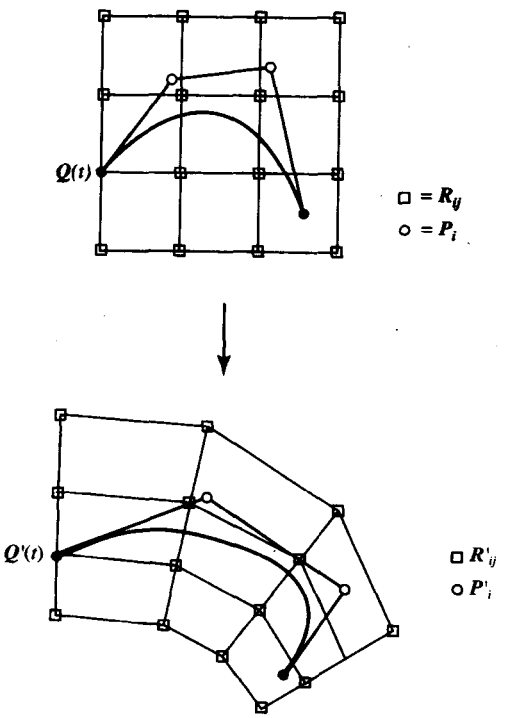


图3-41 一条平面曲线的全局变形（Farin（1990））

现在，为了扩展这个原则，将其应用于全局上对曲面片进行操作，我们把希望变形的曲面片的控制点P_{ij}植入到一个三变量Bézier超曲面片中，这个超曲面片本身是由控制点R_{ijk}的一个三维网格定义的，它形成一个单位立方体。于是，我们有：

$$(u',v',w')=\sum_{i=0}^{3}\sum_{j=0}^{3}\sum_{k=0}^{3}R'_{ijk}B_i(u)B_j(v)B_k(w) \tag{3-7}$$

此单位立方体被全局性地以任何我们希望的方式变形。并计算出新的曲面片控制点P'_{ij}。

这一技术的一个重要方面是，可以将其应用于具有任意参数表示形式的物体上。于是，可以将B样条曲面片植入到三变量Bézier空间中。还可以将多边形网格顶点以相同的方式植入
114 到该空间中。也就是说，将点x植入到一个三变量超曲面片中，然后通过对三变量曲面片的控制网格进行全局变形将其映射到点x'。为了使用方程（3-7）来完成这个工作，需要把x映射到(u, v, w)空间，然后再映射回笛卡儿空间(u', v', w')。其变换方程为：

$$x = x_0 + uu + vv + ww$$

以及

$$u = u \cdot (x - x_0)$$
$$v = v \cdot (x - x_0)$$
$$w = w \cdot (x - x_0)$$

x_0定义(u, v, w)空间的原点，而u, v, w定义该空间。这些值均由设计者来设定，设计者将为单位立方体相应于要进行变形的物体定位。

这一技术开始是由Bézier创立的，但是大多数与图形有关的处理操作都参考Sederburg and Parry（1986）的工作，在他们的文献中这一技术被称为自由形状变形（free-form deformation，FFD）。

图3-42（彩色插图）是一个应用此技术的例子。包围勺子物体几个部分的半透明的矩形实体是三变量曲面片，其中植入了多边形网格的顶点。上面的曲面片由一个$3 \times 3 \times 7$个控制点的网格定义，并注意到在相等的物体空间中有更多的多边形顶点。将网格进行变形就得到了一个勺子的“自然”弯曲，这是在动画中用到的特性。

3.6.3 用表面拟合来创建曲面片物体

在这一小节中，我们考察取一组代表一个物体的三维空间中的点，并通过这些点拟合和插值一个曲面片的表面。我们首先考虑曲线的插值，然后建立一种用于表面拟合的算法。

1. 用B样条插值曲线

通过现有的数据点来拟合B样条曲线（或表面）在计算机图形学中有两个主要的应用。首先是应用于建模：可以利用一个三维数字化设备（如激光测距仪）来产生一组样本数据点。然后的问题是通过这些点来拟合一个表面，使得可以产生一个完整的计算机图像，以便由计算机程序对其进行处理（比如说制作动画或改变形状）。第二个应用是计算机动画。可以用一条参数曲线来表示一个正在三维空间中移动的物体的路径。可以定义物体的特定位置（关键帧的位置），我们需要拟合通过这些点的一条曲线。为了做到这一点，一般采用B样条曲线，
115 这是因为它有C^2连续的性质。在动画中，我们一般关注平滑运动，而用B样条曲线将物体的位置作为时间的函数来表示就可以保证这一点。

我们非正规地论述B样条插值的问题如下：已知一组数据点，我们希望导出一组B样条曲线的控制点，这将定义一条“合法的”代表这些数据点的曲线。我们可以要求该曲线与所有的点是内插的，或者使这些数据点的一个子集与曲线内插而其他的数据点只是靠近它。例如，当已知数据点具有噪声或有些不大可靠时，我们可能不希望对所有的数据点是严格内插的。在Bartels等（1987）的文献中列出了将B样条曲线与一组数据点进行拟合的不同方法。

对于需要使曲线与所有的数据点内插的情况，可以正规地阐述这个问题。如果我们考虑数据点是u中的结点值，则对于立方体有：

$$Q(u_p)=\sum_{i=0}^{m}P_iB_i(u_p)$$
$$=D_p(p=3,\cdots,m+1)$$

其中：

D_p是数据点；

u_p是相应于此数据点的结点值。

现在我们的任务是确定u_p。最容易也是这里使用的方案是设u_p为p。这被称为统一参数（uniform parametrization）。它完全忽略了数据点之间的几何关系，并且被认为这样做是在各个可能性层次中给出的最坏的内插，Farin（1990）对此有详细的描述。另一个最好的方案是弦长参数（chord length parametrization），即把结点的间隔设为正比于数据点之间的距离。然而，统一参数的一个优点是在对数据点的仿射变换下，它是不变的。

这里我们通过数据点来定义B样条曲线。要内插$m-1$个数据点，且曲线由$m+1$个控制点来定义。如果考虑一个单分量，比如说x，则有

$$x(u_p)=\sum_{i=0}^{m}P_{xi}B_i(u_p)$$
$$=D_{xp}$$

其中：

D_{xp}是数据点的x分量。

这样就定义了一个用于解P_{xi}的方程组：

$$\begin{bmatrix}B_0(u_3) & \cdots & B_m(u_3)\\ \vdots & & \vdots\\ B_0(u_{m+1}) & \cdots & B_m(u_{m+1})\end{bmatrix}\begin{bmatrix}P_{x0}\\ \vdots\\ P_{xm}\end{bmatrix}=\begin{bmatrix}D_{x3}\\ \vdots\\ D_{xm+1}\end{bmatrix}$$

116

这一方案产生了$m-1$个方程，$m+1$个未知数。例如，对于$m=5$，要从四个数据点中求出六个控制点。各种可能性都存在。最好的解决方法是选择两个附加点P_2和P_7进行内插。很清楚，这些附加点可以是多出的点，也可以是数据集中已有的点。已知：

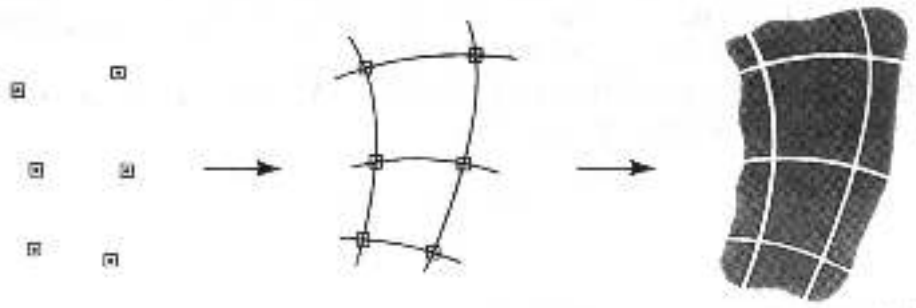
$$\begin{bmatrix}B_0(u_2) & B_m(u_2)\\ B_0(u_3) & B_m(u_3)\\ \vdots & \vdots\\ B_0(u_{m+1}) & B_m(u_{m+1})\\ B_0(u_{m+2}) & B_m(u_{m+2})\end{bmatrix}\begin{bmatrix}P_{x0}\\ \\ \vdots\\ \\ P_{xm}\end{bmatrix}=\begin{bmatrix}D_{x2}\\ D_{x3}\\ \vdots\\ D_{xm+1}\\ D_{xm+2}\end{bmatrix}$$

在这个方程中，矩阵中的项除了沿着主对角线的一个$3(k-1)$宽度的位置之外均为零。而对于均匀三次B样条，其矩阵为：

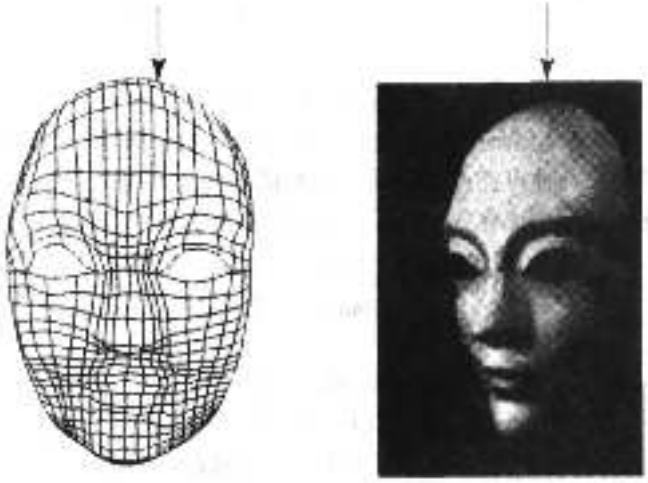
$$\frac{1}{6}\begin{bmatrix}4 & 1 & & & & & \\ 1 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & \ddots & & & & \\ & & & \ddots & & & \\ & & & & 1 & 4 & 1\\ & & & & & 1 & 4\end{bmatrix}$$

2. 插值表面

插值表面指的是将一组三维数据点与一个参数化定义的表面进行插值。我们通过将一个uv曲线的网格与数据点进行拟合来完成这一工作。每一个数据点都被常数u和v的一条曲线内插。这些曲线为用标准B样条曲线插值技术与数据点插值后的B样条。下一步，B样条曲线被转换为Bézier曲线，而这个曲线的网格则被分割成由四个Bézier曲线段组成的各个网格元素(见图3-43)。这些曲线段就是Bézier曲面片的边界，已知这些边后，可以导出曲面片的控制点。这样一来，三维空间中的一组点就被转换成一个Bézier曲面片的网格，而且B样条曲线的网格作为各曲面片的界线。



a) 三维空间上的一组数据点　b) 在两个参数方向拟合通过数据点的曲线　c) 由曲面片的边界得到曲线的网格



d) 通过对数字化的点进行插值得到的一个曲线网格　e) 由d得到的曲面片模型进行的一种绘制

图3-43　表面拟合的一种图示表示

现在让我们考虑第一个阶段——导出一个插值数据点的曲线网格。这里的主要问题是，我们对于点的拓扑结构可能一无所知（在曲线的插值中，我们知道这些数据点是连续的），这是一个只能特定问题特定处理的问题。在Watt and Watt（1992）的文献中给出了一种方法。考虑这样一种情况，用一种手动的数学化仪从真实的物体上获得了数据点，并已知哪些数据点与曲线进行插值。这是一种非常普遍的上下文，这时我们将进入第二个阶段——从一个B样条曲线的网格导出一个Bézier曲面片的网格。

首先，需要将B样条曲线转换成多段的Bézier曲线。如果首先考虑一个B样条，则将其转换成Bézier形式就是简单直接的，这个转换为：

$$[\boldsymbol{P}_0\ \boldsymbol{P}_1\ \boldsymbol{P}_2\ \boldsymbol{P}_3]=\boldsymbol{B}^{-1}\boldsymbol{B}_s[\boldsymbol{Q}_0\ \boldsymbol{Q}_1\ \boldsymbol{Q}_2\ \boldsymbol{Q}_3]$$

$$=\frac{1}{6}\begin{bmatrix}1&4&1&0\\0&4&2&0\\0&2&4&0\\1&4&1&0\end{bmatrix}\begin{bmatrix}\boldsymbol{Q}_0\\\boldsymbol{Q}_1\\\boldsymbol{Q}_2\\\boldsymbol{Q}_3\end{bmatrix}$$

其中：

$\boldsymbol{P}$为Bézier控制点；

$\boldsymbol{Q}$为B样条控制点；

$\boldsymbol{B}$为Bézier矩阵；

$\boldsymbol{B}_s$为B样条矩阵。

（注意：基矩阵的变化只能作用于均匀B样条。如果曲线网格是由非均匀B样条组成的，则必须在转换之后进行结点插入。在Watt and Watt（1992）的文章中对这种更通用的方法进行了阐述。）

对于多段的B样条曲线，我们对适当的控制点反复地应用这个公式。例如，考虑由五个控制点$[\boldsymbol{Q}_0\ \boldsymbol{Q}_1\ \boldsymbol{Q}_2\ \boldsymbol{Q}_3\ \boldsymbol{Q}_4]$定义的一个两段B样条曲线。转换公式对于控制点$[\boldsymbol{Q}_0\ \boldsymbol{Q}_1\ \boldsymbol{Q}_2\ \boldsymbol{Q}_3]$和$[\boldsymbol{Q}_1\ \boldsymbol{Q}_2\ \boldsymbol{Q}_3\ \boldsymbol{Q}_4]$应用了两次。

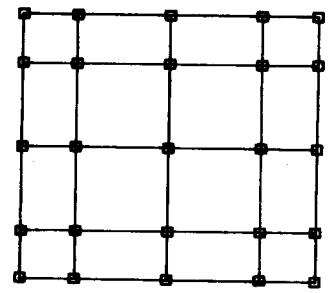
现在考虑一个由5 × 5个两段B样条曲线组成的网格，其控制点如图3-44a所示。将上面的方案按照逐行的方式来应用就产生5个两段的Bézier曲线（见图3-44b）。现在我们来按列地解释这个问题。把一列看做是一条B样条曲线的控制点。每一列都被转换成一个两段的Bézier曲线。这样就会产生一个由7 × 7个两段Bézier曲线组成的网格（见图3-44c）。我们知道，Bézier曲面片的边界上的边是Bézier曲线，在图3-44c中把这些未分段的线解释为四个Bézier曲面片的边界。这样一来，我们就将一个由5 × 5个两段B样条曲线组成的网格转换为2 × 2个Bézier曲面片。然而，我们只定义了每个曲面片的边界，也就是说我们对于每个曲面片只确定了其16个控制点中的12个。

现在出现的问题是：我们将如何来确定其内部的四个控制点呢？首先通过重写方程（3-5）来考察它们的重要性：

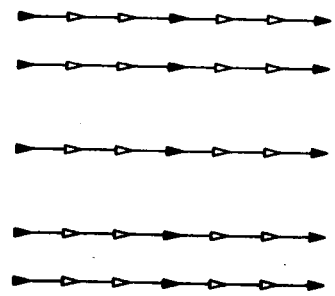
$$\boldsymbol{Q}_{uv}(0,0)=9(\boldsymbol{P}_{00}-\boldsymbol{P}_{01}-\boldsymbol{P}_{10}+\boldsymbol{P}_{11})$$

这个方程将对表面的扭曲定义为混合的偏导数，可以将其解释为一个向量，它是控制多面体角上的四边形与平行四边形的偏离（见图3-45）。当这个扭曲为零时，此向量减为零，四边形角上的点是共面的。非零扭曲的影响的直

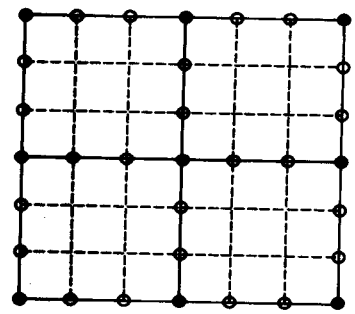
117
2
118



a) 5 × 5个两段B样条曲线组成的网格



b) 曲线网格按行转换为5个两段的Bézier曲线



c) 曲线网格转换成7 × 7个两段Bézier曲线，于是形成了4个Bézier曲面片的边界

图3-44 将B样条曲线网格转换成Bézier曲面片（Farin（1990））

现获得是困难的。而当曲面片在所有四个角上的扭曲为零时，则很容易看出其几何意义。

我们有一个平移了的表面，在这个表面上控制多边形中的每一个四边形都是平行四边形。这些平行四边形之间也是相互平移的。这样的表面称为平移的表面，因为它是由两条曲线$C_1(u)$和$C_2(v)$产生的。u中的任一等参数线都是C_1的平移，而v中的任一等参数线也是C_2的平移。

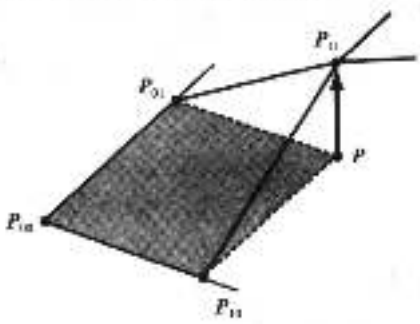


图3-45 扭曲系数正比于控制多边形的子四边形与平行四边形的偏离

干是，估计内部的四个控制点的最容易的方法就是假设曲面片是一个平移的表面，则其内部的点就可以从其边界点导出。这样做的含义是：从表面拟合的观点来看，曲面片的边界曲线应该或多或少相互之间是可平移的。在任何应用情况下，这将依两个因素而定：物体的形状及uv曲线网格的分辨率。

整个过程的可视化描述如图3-43所示。从真实物体获得原始的表面点。在这个图中可以看到，零扭曲假设的有效性在整个物体的表面是如何变化的。这个方法按照全局的C^1连续性给出了一个正确的表面，但是它并不能保证有“好”的形状。确定表面的曲面片的扭曲的另一种方法由Farin（1990）给出。

119
-
120

3.7 从曲面片到物体

我们已经强调过，参数化曲面片表示方法的主要用途之一是对一种数控的切削设备进行编程，而这种设备将使得这个抽象的设计直接转换成物体或物体的一个模型。

在大多数应用中，这涉及到对切削工具进行编程，以便从原料的形状（比如矩形的棱柱体）中去掉一些材料，产生所希望的物体（如图3-46所示）。实际应用的技术依赖于很多因素，比如模型的性质、材料以及切削设备的性能等。例如，对于像金属这样的硬材料来说，不太可能一次就去掉所有的材料，必须对切削设备进行编程，使其可以产生中间的形状，以便一步一步最终产生希望的物体。由于其物理伸展长度的限制，切削设备可以达到的范围有一个最小的曲率半径。当工具有一个半球的尖端时这种局限性就很容易想象。切削机应遵循的实际路径需事先进行确定。所以，我们在这里将只是简单地对其原理进行概述。



图3-46 将计算机图像模型转换成实际物体

资料来源：H.Chiyokura, *Solid Modelling with DESIGNBASE*, Addison-Wesley Longman Singapore Pte Ltd., Singapore, 1988.

在可能出现的最简单的情况中，我们考虑在切削设备中把工具的尖端置于距离切削工具的参考点为01的位置。这是机器中的三维数据点，切削机需要对其进行控制。如果先不考虑 121
实际的路径，这个点必须被移到一个“平行”的表面，称为偏移表面。对于曲面片$\boldsymbol{Q}(u, v)$，一个简单的偏移表面可由下式给出:

$$\boldsymbol{O}(u, v) = \boldsymbol{Q}(u, v) + d\boldsymbol{N}(u, v)$$

这个公式定义了一个沿着法向$\boldsymbol{N}(u, v)$、距离为d处的法向表面$\boldsymbol{O}(u, v)$。导出这个表面之后， 122
就可以从该偏移表面上适当跨度上的等参数曲线中导出切削机的路径。

第4章　表示和绘制

4.1　绘制多边形网格——简单综述
4.2　绘制参数化表面
4.3　绘制构造实体几何表示
4.4　绘制体素表示
4.5　绘制隐函数

引言

在这一章中我们将考察不同的绘制策略，这些策略用于对第2章和第3章中讨论的表示方法进行可视化。绘制方法可以按照对特定的表示方法采取特定策略的方式来讨论，这也是本章的主题；另一种讨论方式是，可以将这些方法作为实现全局照明模型来讨论。在后一种情况下，辐射度和光线跟踪方法是值得一提的。

我们首先从一个简短的综述开始，下面列出了用于第2章和第3章中给出的表示方法的最常见的策略。

1) 多边形网格绘制策略在计算机图形学中是最主要的绘制策略。所采用的方法自1975年以来一直被使用，而且大多数应用程序都使用这种方法。另外，绘制双三次参数曲面片和体素表示的最容易的方法是将其转换成多边形网格，并使用标准的绘制程序。

2) 双三次参数曲面片：最容易的可视化或绘制曲面片的方法是将其转换成多边形。快速的划分算法导致一个多边形网格，而这总体来说就暗示着一个快速的绘制。当然，这也意味着表面的近似，我们将看到，可视化可以以一种看起来并非不精确的方式进行。近似对于从数学描述直接进行绘制是更可取的，其原因将在本章给出。

3) CSG物体可以用一种光线跟踪算法（或称光线投射算法）来绘制；也可以通过先将其转换成体素表示，然后再转换成多边形表示来绘制。在这种表示方法中内在的困难在于操作算子的布尔组合的估算。

4) 体素：这里我们要再一次将其转换成多边形网格表示。通常情况下，这会涉及到对每一个物体上的不规则多边形的计数，而这也是该方法的主要缺陷。

5) 隐式表示：这种表示可以转换成体素表示或者用光线跟踪算法。

由于多边形网格绘制技术的重要性，我们将在第5章和第6章中研究这一问题。这也适用于体素表示，体素表示的问题将在第14章中给出详细的讨论。而在这一章中，我们将讨论用于其他表示方法的绘制技术。

4.1　绘制多边形网格——简单综述

正像我们已经讨论过的那样，在计算机图形学中，多边形物体显然是最常见的表示形式。而固化的硬件程序也应用于许多图形工作站上，图形工作站可以由一个多边形数据库绘制物体。

多边形绘制程序的输入是一个多边形的列表，而其输出是每一个像素的颜色，每个多边形会投影到屏幕上的像素上。多边形绘制程序的主要优点是已经有了把多边形作为一个实体或单元进行计算的算法。多边形成为图形程序设计人员所必须考虑的最底层的元素，这有助于非常快速和简单的处理。然而，我们必须注意到，这些优点随着物体越来越复杂而逐渐被削弱了。物体由成千上万个多边形来描述的情况非常常见。图2-5b就是一个例子。随着多边形的投影区域趋向于单个像素点，插值算法（插值算法是通过对多边形顶点和像素颜色插值来实现的）的优点快速下降。

绘制引擎对于多边形网格主要执行两个任务。第一个任务是将物体的几何学作为各种变换（建模变换、观察变换等）来处理，接着执行估算光线与物体相交的过程，这个过程一般称为明暗处理（shading）。

最常见的多边形网格绘制程序在第二个过程中有两个主要的组件。一个是明暗处理算法，它在多边形的投影中为每一个像素求出其适当的色深（shade）。另一个是隐藏面消除算法，该算法评估一个多边形的某部分是否被靠近观察者的另一个物体所遮挡。

明暗处理算法在多边形的顶点处求一个强度。然后，从这些值中进行插值求出对于多边形像素点的适当光强度。有一个方程用于求顶点处的光强度，方法是将与顶点所处表面相应
124 的法向量的方向与光源的位置进行比较。用内插的方法来求出像素值。将这两个操作以某种方式组合，使得小平面的边界的可见性降低，而且使曲面看起来是弯曲的，这个曲面是已经由平面多边形近似的。

与这个操作相一致的是，用相似的内插方法，根据顶点所处的深度来求出每一个多边形像素的深度（而顶点的深度是根据场景和观察者的几何位置求出的）。这些深度值存储在一个数组中，称为Z缓冲器。其后，将各像素的存储值与当前的深度值按升序进行比较，看看当前的多边形像素与以前绘制的像素中最靠近观察者的那个像素相比较哪一个更靠近观察者。

这个方法使我们能够以任意顺序从数据库中取出多边形，并将其以独立的单元进行绘制。这就意味着，通过求出多边形的色深，使有些多边形最终不被写入屏幕缓冲器就能完成绘制工作（因为它们比之前绘制的多边形更远一些）。这是一个可视化多边形网格物体的简练且直接的方法。数据库中的表示单位（多边形）被绘制程序按单元处理，使得这些单元的近似几何性质几乎是不可见的。

很多人对创建这一方法有贡献，但是最终常被提及的名字是Gouraud和Phong，他们创立了明暗处理算法。该方法的出现可能是三维计算机图形学中最有意义的进步，其在光线跟踪和辐射度算法中的长期流行证明了它的精练和有效。

4.2 绘制参数化表面

绘制那些由双三次参数曲面片表示的表面的算法一般分成两类:

1) 直接由参数描述或用方程来描述曲面片进行绘制的方法。

2) 用多边形网格来近似表面，并使用平面多边形网格绘制程序来绘制这种近似的方法。因此，在这种情况下，绘制参数化表面就变为预处理操作或者转换操作。

第二类方法目前看来最流行。可以肯定的是，这种方法执行起来较容易，而且从计算角度讲较经济。第一类方法的例子可在Blinn（1978）、Whitted（1978）、Schweitzer and Cobb（1982）、Griffiths（1984）等文章中找到。Lane等（1987）对这些方法给出了综合性的描述，

并描述了第二类方法的一个实现。

4.2.1 直接由曲面片描述进行绘制

直接由曲面片描p述进行绘制意味着采用一种扫描线算法。当我们沿着一条扫描线移动时，对于特定的像素，从曲面片方程求出曲面片上相应的点。由此，我们就有这样的问题，即把 125
像素或屏幕空间（x, y）的求值强加在曲面片的数学描述上了，这就是困难的根源。我们可以通过考虑一个弯曲的曲面片与一个平面或平的曲面片（换句话说，就是一个多边形或平行四边形）之间的关系来感觉其困难之处。为了进行简单的扫描转换而构建的多边形的性质（对于曲面片很难获得）有：

1) 最大和最小Y坐标，可很容易由顶点列表中得到。

2) 增量方程，可以用于按Y的函数跟踪每一条多边形的边。

3) 增量方程，可用于按X的函数计算屏幕深度Z。

参数化定义的表面没有这些性质。最大和最小Y坐标不一定在表面的边界或边处，因为曲面片通常是显示出屏幕空间上的一个轮廓边，而这个轮廓由曲面片的内部突出部分产生。对于参数化定义的表面，其边界上的边以及其轮廓边都需要跟踪。更困难的是，一条轮廓边和一条边界面还可能会相交。最后，一般来讲无论是边界边还是轮廓边都不会是在X和Y方向单调的。

首先，参数化定义的表面或者表面的曲面片可以由三个两变量方程来表示。

$$x = X(u, v)$$
$$y = Y(u, v)$$
$$z = Z(u, v)$$

其中，v和u在0～1之间变化。表面曲面片的边界由值$u = 0$、$u = 1$，$v = 0$，$v = 1$来定义。这就产生了一个四边的曲面片。

在双三次参数曲面片上下文中，一种方式是把扫描转换看成是一个算法，该算法求出由XZ扫描平面与该表面相交而形成的曲线（见图4-1）。Blinn指出，这些XZ曲线没有一个明确的公式，求得曲线上点的唯一方法是用数值方法，如牛顿迭代法。对于平面多边形，这条曲线是一条直线，对其只需存储端点，而对于参数化表面，则需要确定相交曲线上所有的点。

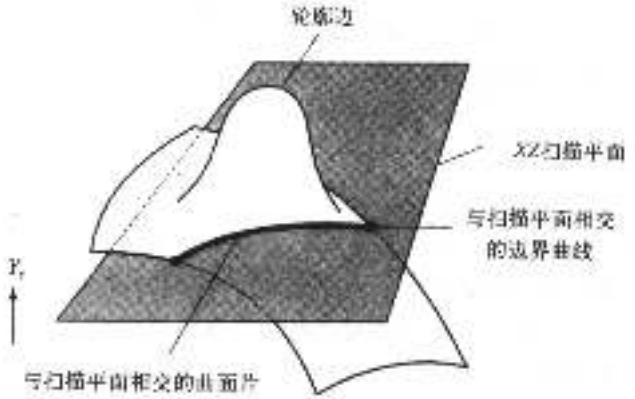


图4-1 一个用于绘制曲面片的扫描线算法。可以按如下方式进行：先求出曲面片相交曲线的结构，再用一种迭代算法来跟踪它

Blinn（1978）提出的用于参数扫描转换的Blinn算法是直接的代数方法，它涉及求解边界线和轮廓线与每一条扫描线相交的方程（采用迭代技术）。也就是说，曲线可以由一些片段组成。例如，从边界到轮廓边的片段、两个轮廓边之间的片段以及从轮廓边到边界的片段等。
126 扫描线Y_s与曲面片的边界之间的相交由下式给出:

$$Y_s = Y(0, v)$$
$$Y_s = Y(1, v)$$
$$Y_s = Y(u, 0)$$
$$Y_s = Y(u, 1)$$

轮廓边相交由下式给出:

$$Y_s = Y(u, v)$$
$$0 = N_z(u, v)$$

其中，N_z是曲面片表面法向的Z分量。轮廓边是由表面上的这些点定义的，而这些点表现了一个Z分量为零的表面法向。局部的最大值和最小值由下式确定:

$$Y_u(u, v) = 0$$
$$Y_v(u, v) = 0$$

其中Y_u和Y_v为相对于u和v的偏微分。

在这两个点之间曲线上的点由下式给出:

$$Y(u, v) - Y_s = 0$$
$$X(u, v) - X_s = 0$$

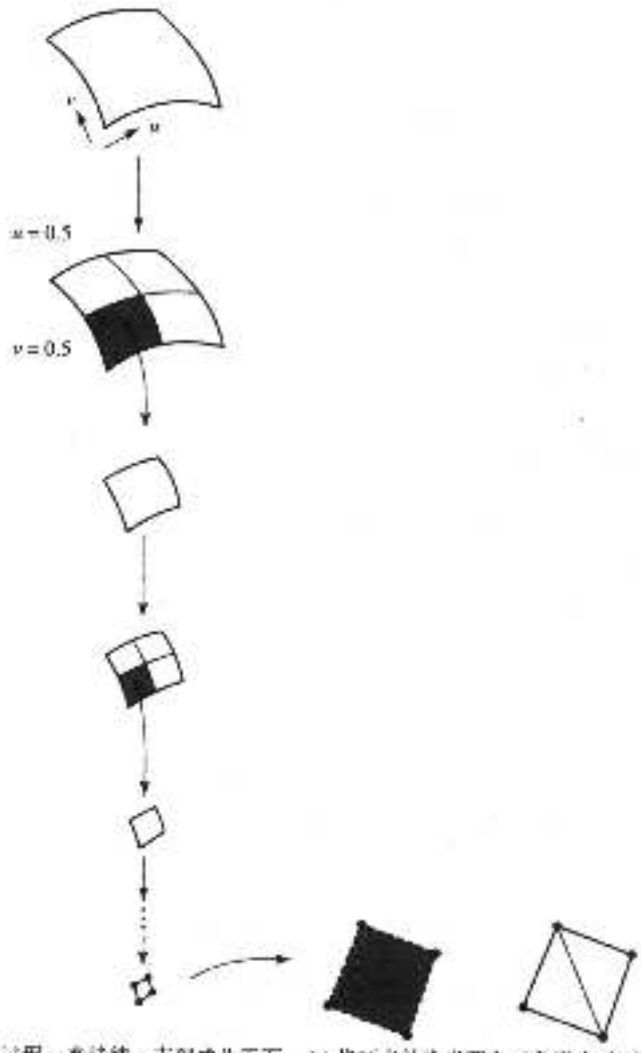
解这组方程得到参数空间上的一组点（u, v），代入X和Z方程，产生在当前扫描平面上曲线的X和Z值。对每一个曲面片或表面执行这一过程，产生代表区域已有明暗处理的连续的边界对。随着扫描平面向屏幕下方的移动，必须跟踪边界和轮廓边，并保持它们的连接。这个处理过程并不容易，Blinn给出了一个利用马鞍点的有启发意义的例子。找到一个局部最大值就意味着将一条新的相交曲线加入到相交曲线列表中。在Y最小值处删除曲线。这个处理过程
127 的结果是，把表面的曲面片划分成一些在Y方向为单调减而在Z轴上为单值的一些区域。Lane等（1980）指出，这个方法有很多的问题。但是，对于大多数形状，这个算法是健壮的。这样的话，对于平面多边形扫描转换来说，外部的Y扫描循环跟踪边界和轮廓边，而内部的X扫描循环向Y扫描平面上的相交曲线中“填入”点。Blinn进一步通过在精确度和速度之间进行某种程度的折中将这个过程进行改进，即将相交曲线用直线段进行逼近。

Whitted方法、Schweitzer方法和Griffith方法均是这种基本方法的变种。

4.2.2 曲面片向多边形转换

从曲面片网格导出多边形网格原则上是简单的。可以简单地将曲面片进行细分，将分割产品的角点作为多边形的顶点。每四个顶点转换成两个三角形小平面（见图4-2）。这样在将曲面片转换成多边形时只要连续地细分，直到我们认为在某种程度上多边形的近似已经足够逼近真实的表面（曲面片）为止。在这里可能有一些自相矛盾的地方。如果我们用多边形小平面来近似的话，那么为什么还要用到曲面片表示呢？其实有很多理由。最一般的解释是应用程序需要这样的表示，尤其在CAD中更是如此。对物体的可视化可能只是一种近似，但是设计者需要精确的表示。我们还必须记住，正如将要看到的那样，表示方法允许我们控制多

边形近似的精确度。



a) 过程一直持续，直到成为平面　b) 将顶点转换成两个三角形小平面

图4-2　曲面片分割过程

这个问题可以依所采用的判据而自然地分成几类。这些判据用于确定细分的深度和将其用在曲面片表面的什么位置上。其主要的区别在于是物体空间还是屏幕空间终止判断的执行。屏幕空间终止判断为调节物体的多边形分辨率以便使其与物体在屏幕上的投影尺寸相匹配提供了潜在的可能。在物体空间，我们用物体空间的量度来近似真实的表面；在屏幕空间用像素单位作为量度。从其依据于观察点和观察距离的角度来看，屏幕空间终止判断的方法是动态的。

4.2.3　物体空间细分

在处理物体空间时，我们首先列出下面的一些简单分类：

1）物体空间均匀细分。这是最简单的情况，用户定义一个所有曲面片均匀细分终止的

水平。

2) 物体空间非均匀细分。这个方法是指当细分出的产品满足曲面片平滑度判据时停止细分过程。

第二类在理论上更合理一些，在这里，根据需要可以“定位”细分的程度。在表面曲率高的地方需要更多的细分。但是，在测试平滑度上要花费很多时间和空间，还可能会因为不必要的细分而超过所需的消耗。

均匀细分过程如下所述。可以用等参数曲线来划分曲面片（在CAD系统中，将曲面片分
128 割成等参数曲线是很常见的显示方法。该方法使表面线框的可视化足可以满足这种系统的需
~ 求）。其结果是在这些曲线之间和边界的交点处产生点的网格。这个点的网格可以用于确定平
129 面多边形网格的顶点，平面多边形网格可以用平面多边形绘制程序进行绘制。这个基本方法有两个问题。可见的边界和轮廓边可能表现出不连续性。一般来说，对有一个好的多边形分辨率以消除可见的曲面片间的线性不连续的需求，超过了对曲面片之内保持光滑的明暗处理的需求。另一个问题是曲面片中的内部轮廓边一般高于三次。如果要对轮廓边给以特别的关注的话，最好是在屏幕空间来进行处理，这将在下一节讲述。

现在考虑非均匀细分。这只是意味着那些平滑的曲面片区域将进行较少的细分，而局部曲率高的区域将进行较多的细分。实际上，曲面片细分的程度依赖于局部曲率。这是在Lane等（1980）中由Lane和Carpenter提出的一种方法。在图4-3中展示了这个方法。

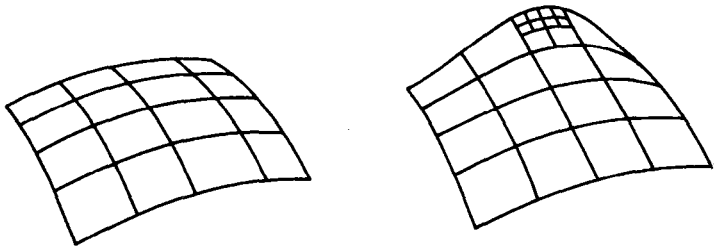


图4-3 一个Bézier曲面片的均匀细分和非均匀细分

对曲面片进行细分，直到细分的产物符合平滑度的判据。这时，将这些满足平滑度判据的曲面片近似地看成是平面多边形，再以曲面片的角点作为多边形网格中矩形的顶点，用一种常规的多边形绘制方法进行扫描转换。可对代表表面的曲面片的集合进行预处理，产生一组多边形，然后再对其按正常情况进行扫描转换。这是Clark（1979）提出的方法。Lane将这种曲面片分离的方法与一种扫描转换方法进行了结合。

该方法与曲面片分离方法相比有两个明显的优点：

1) 它的速度快。

2) 可以根据细分的深度来改变速度。这对于交互式系统是重要的。

非均匀细分方法的一个缺点是由于用一条直线来近似一个曲面片的边界，所以在两个曲面片之间可能会产生孔。图4-4为这种变化过程的一个例子。

对于曲线来说，细分算法是最好的。其后这些曲线可以容易地进行扩展或一般化，以用来处理曲面片。这一方法的核心并不是直接去求曲线上的点，而是通过递归地细分控制点而
130 得到的一些线性的线段来近似曲线。这样会得到一条越来越近似的曲线。细分/递归终止条件是满足一种线性判据。Lane and Reisenfeld（1980）证明，假如进行了足够的细分的话，分段

的线段会最终“塌落”到曲线上。

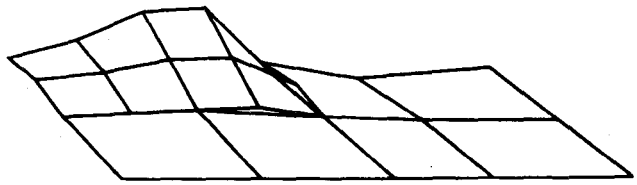


图4-4 由非均匀曲面片细分产生的撕裂

Lane等（1980）给出了对于Bézier基函数（或更一般化的Bernstein基函数）的细分公式，Lane and Riesenfeld（1980）进行了推导。这个过程可以用于任何基函数，方法是首先按Watt and Watt（1992）提出的方法将表示转换成Bézier基函数。将一条Bézier曲线用细分控制点的方法细分成两条曲线，形成两个新的控制点集合R_i和S_i。控制点R_3/S_0是第一条曲线的终点和第二条曲线的起点。这个公式为:

$$
\begin{aligned}
R_0 &= Q_0 & S_0 &= R_3 \\
R_1 &= (Q_0 + Q_1)/2 & S_1 &= (Q_1 + Q_2)/4 + S_2/2 \\
R_2 &= R_1/2 + (Q_1 + Q_2)/4 & S_2 &= (Q_2 + Q_3)/2 \\
R_3 &= (R_2 + S_1)/2 & S_3 &= Q_3
\end{aligned}
$$

从图4-5可以看到，在一次细分之后，连续两个新的控制点集合的分段线性曲线是对原曲线的一个较好的近似。三次细分之后的近似如图4-6所示。

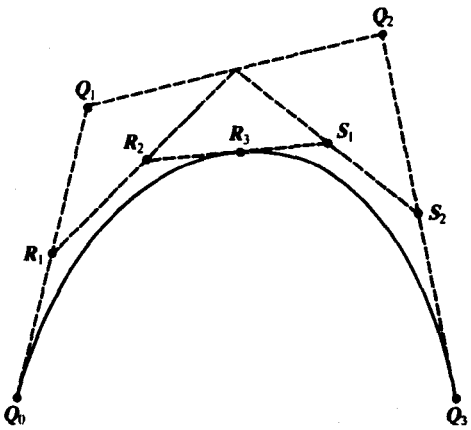


图4-5 分离一条双三次Bézier曲线

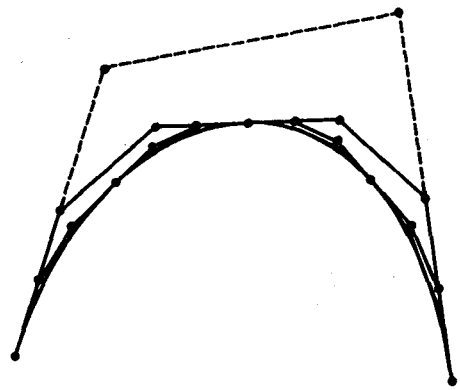


图4-6 每一个细分水平的控制点示意图

曲线的分离过程可以很容易地扩展到曲面片，如图4-7所示。我们把曲面片看成是由4条常数u的曲线和四条常数v的曲线组成的，其控制点是控制点矩阵中的连续的列元素和连续的行元素。我们把曲线细分公式分别地应用于u上的四条曲线上，产生原始曲面片的两个子曲面片。然后再对这两个曲面片重复这一过程，但这一次是在v方向上细分曲线。将两次分割放在一起，产生四个曲面片。 131

这种有效的公式（即只采用加和除2），使细分过程很快进行。细分的深度很容易用线性判据进行控制。Bézier基函数之和恒等于1。

$$\sum_{i=0}^{3} B_i(u) = 1$$

这意味着曲线被限制在由控制点P_i形成的凸包之内。当这些分段的线性细分产物与连接两个端点的连线“合并”时将处于共线位置。达到这一效果的程度，即连接四个控制点的连线的线性程度可以通过测量从两个控制点的中点到连线的端点之间的距离来确定（见图4-8）。

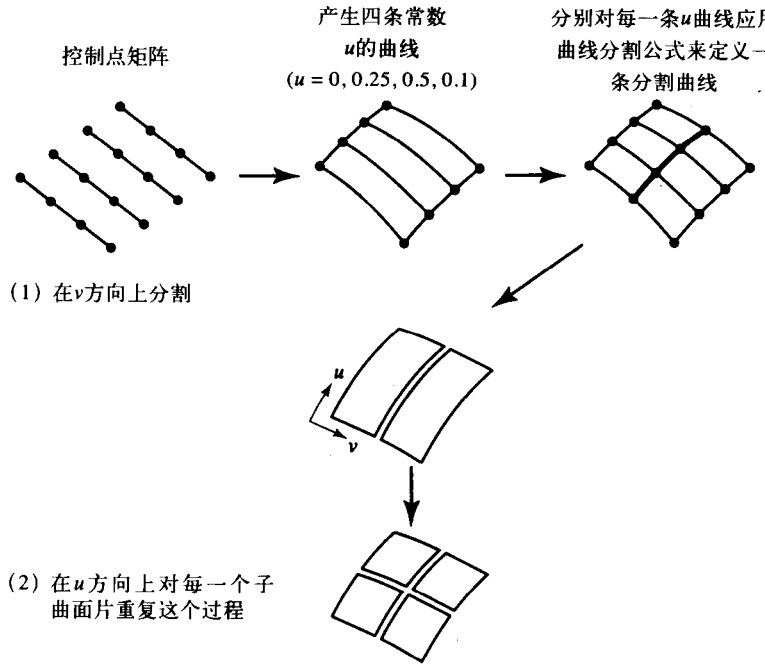


图4-7 用曲线细分方法将一个曲面片细分为四个

这种测试的原则很容易扩展到表面曲面片。一个平面用三个非共线控制点来拟合，然后逐个计算这一平面与其他13个控制点之间的距离。如果这一距离之一超过了预先定义的偏差值，则对曲面片进行进一步的细分。实际上，我们测量的是限定盒（bounding box）的厚度，限定盒是一个矩形的实体，它包围了其厚度由从含有角点的平面到最远控制点的最大距离确定的曲面片。这一计算有时称为凸包平滑度测试。

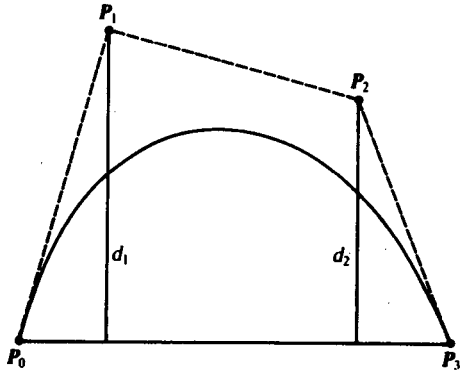


图4-8 具有控制点P_0, P_1, P_2, P_3的一条三次Bézier曲线

当将非均匀细分方法（细分持续进行，直到满足某个平滑度判据为止）与均匀细分到某个预先确定的水平的方法相比较时，出现了一个实际的问题，即平滑度测试的成本。只是简单地采用均匀细分，忽略某些区域将被不必要地细分（因为它们已经是平面）是否是一种较简单的好方法还有待讨论。对于给定的图像质量，非均匀细分测试的成本要大于均匀细分额外绘制的成本。下一个图片展示了这个观点。图4-9（彩色

插图）为对Utah茶壶采用均匀细分方法以1、2和3次细分所得的结果。这一图像的重要性之一是它预示绘制图像的质量可以主要通过轮廓边看出。

如果采用非均匀细分方法，则在绘制双三次曲面片时可以减少对常规Z缓冲器绘制程序的预处理过程。但这个方法需要一个大的数据库来存储细分产物。如果没有足够的存储空间再加上采用扫描线算法的复杂性，则非均匀细分方法不是一个好的选择。 132 ~ 133

如Lane-Carpenter算法等扫描线算法的总体结构如下:

1) 通过对可能的Y值取最大值来对曲面片排序。正如已经讨论过的那样，该值可以是一个角点、一个边界上的点或者一个轮廓边上的点。估计该值的唯一快速方法是取控制点凸包的最大Y值。

2) 对于每一条扫描线，进行活动的和不活动的曲面片列表的更新。对于每一个活动的曲面片进行细分，直到满足平滑度判据或者任意细分产物跨过了扫描线。“平面”多边形被加入到活动的多边形列表中，并按常规方案进行扫描转换。不再跨越扫描线的划分产物可以被加入到不活动的曲面片列表中，于是就有了一个活动的和一个不活动的参数曲面片列表以及一个平面多边形的活动列表。

必须处理的实际问题之一是：撕裂（tear）问题（见图4-4）。撕裂是递归或非均匀细分方法的必然结果。如果表面曲面片的某部分沿着一条与另一个曲面片分享的边界细分，而另一个曲面片不需要进一步细分的话，则在两部分之间自然会产生裂缝。这种“撕裂”是一个未被表示的区域，在最终的绘制图像上以裂缝或孔的形式出现。Lane-Carpenter算法没有处理这个问题。可以通过使得平滑度判据更精确来将裂缝减到最小，但是这又会遇到计算的困难。细分方法的原则是将表面的区域进行相对应于局部曲率的程度的细分。大的、平滑的表面细分次数最少。曲率快速变化的区域将被多次细分，直到分成足够小的多边形。对平滑度判据的进一步严格意味着会产生更多的多边形，而使最终的绘制工作花费更长的时间。

Clark的方法以一种更简练的方式对这一问题进行了处理。他采用的方法是一开始把细分限制在边界曲线上。这一方法分为三步:

1) 对边界$u = 1$和$u = 0$，应用凸包的判据，在v方向上细分曲面片，直到满足该判据。

2) 对边界$v = 0$和$v = 1$应用相同的方法。

3) 最后，对细分产物进行常规的凸包测试，必要时该过程在v或u方向继续进行。

一旦有一条边界满足了凸包判据，则假设它为一条直线。之后，沿着这条边界的任何进一步的细分步骤都不会导致分割。

沿着u或v方向边界上的细分过程的一个可能的优点是，对于某些物体，这种方法可以产生较少的曲面片。例如，考虑细分一个“管状”实体，这种物体的一个通常的例子是圆柱体。沿着平行于其长轴的方向细分此圆柱体将使该算法更迅速地收敛，而且比采用在两个参数方向进行细分的方法产生的曲面片数少。这一方法的一个缺点是，很难将其与扫描线算法相结合。扫描线算法依赖于曲面片与扫描线之间的相互关系来“驱动”或控制细分过程的顺序。 134

另一个需要考虑的方面是表面法向的计算。当然，这一计算在计算明暗处理时也是需要的。法向量可以很容易地从原始参数描述中得到，即对表面上的任意点(u, v)，计算u和v偏导数的叉积。但是，如果将细分方法用于扫描转换，则最终的多边形绘制需要采用Phong插值方法，通过取角点处切向量的叉积可以很容易地计算出顶点法向量。一般来讲，依细分程度的不同计算将给出“平面”多边形的非平行顶点法向量，但是所有分享该顶点的多边形将具有相同的法向量。“平面”多边形将随非平行的顶点法向量被送入一个明暗处理程序，这将导

致两个后果。首先，在较低的细分水平下可能出现错误的明暗处理效果。第二，会出现选择哪一个顶点法向量来进行计算的问题。由于并不是顶点周围的所有多边形都是可利用的，所以会出现问题。因为细分过程在不断地发生着，而平均的顶点法向量不可能像在多边形网格中那样被随时计算出来。在法向量与观察向量的夹角大于90°和该夹角小于90°时很明显会出现这种结果。唯一安全的方法是通过对每一个多边形的顶点法向进行测试来选择一个多边形。如果有任一顶点法向是“可见的”，则不选择该多边形。

一种由Catmull（1974）提出的较早方法是细分曲面片，直到其大致近似于一个像素的大小为止，然后把结果存入Z缓冲器中。这种直接但从计算角度来讲昂贵的方法却引出了另一个问题，即曲面片投影的屏幕尺寸与细分程度之间的关系。很明显，当投影到屏幕空间时，如果曲面片仅仅占据几个像素，则没有必要细分到很细。Clark把细分测试与曲面片控制点的深度坐标之间进行了关联。

最后，这些细分方法所使用的基函数是不同的。Lane-Carpenter算法使用一个三次Bernstein或Bézier基函数。而Clark用泰勒级数展开式，并采用中心差分导出更有效的细分公式，以及利用可直接从细分组分中求出的平滑度测试。

4.2.4 图像空间细分

上一小节中提到的方法对于许多应用是足够的，尤其是对于CAD中的单个物体。但是，当物体在屏幕空间形成的投影尺寸变化很大时，最好考虑用屏幕空间判据来控制细分的深度。
135 我们将这种方法称为依赖于观察的或屏幕空间控制的细分。

有三个简单的方法来进行这一计算。这三个方法均涉及对曲面片上样本点的投影，在当前细分水平，将样本点投影到屏幕空间，并将其与对该曲面片近似的多边形的投影相比较。以像素的长度单位为度量来进行比较。其度量为:

1) 曲面片占据的最小像素面积。

2) 曲面片的屏幕空间平滑度。

3) 轮廓边的屏幕空间平滑度。

1. 曲面片占据的最小像素面积

一旦曲面片投影到足够少的像素面积上，就认为这个曲面片已经被细分得足够细了。极限的情况是可以细分到一个像素的大小，但这样做肯定是昂贵的。

2. 曲面片的屏幕空间平滑度

对于这一测试，可以按照在物体空间中进行测试的方法进行，但采用以像素为基准的度量而不是以物体空间的单位进行度量。换句话说，以像素为单位测量由凸包定义的限定盒的厚度。还可以选择另一种不是特别精确但快一些的测试。这种判据示于图4-10中。图4-10展示了在其当前细分水平下的一个曲面片。其上有三个样本v_0、$v_{0.5}$和v_1，三点均处于常量$u_{0.5}$的曲线上。这条曲线（即下一水平的边界线）处于曲面片的真正表面上，而这三个点是下一个细分水平上多边形的顶点。可以将这些点与由四个顶点形成的双线性曲面片上的点进行比较（双线性曲面片是一个非平面的四边形，其上所有的常数u和常数v的线是线性的）。将该曲线上的这些样本点与二分双线性曲面片的直线相比较得到屏幕空间中曲面片平滑度的估计。此过程可对曲线$v_{0.5}$重复进行。如图4-9（彩色插图）所示，物体的内部不太会受到细分水平的影响，因为明暗处理算法是专门设计用于消除多边形可见边的。这些观察结果使我们得出的结论是，需要将细分集中到靠近轮廓边的位置。

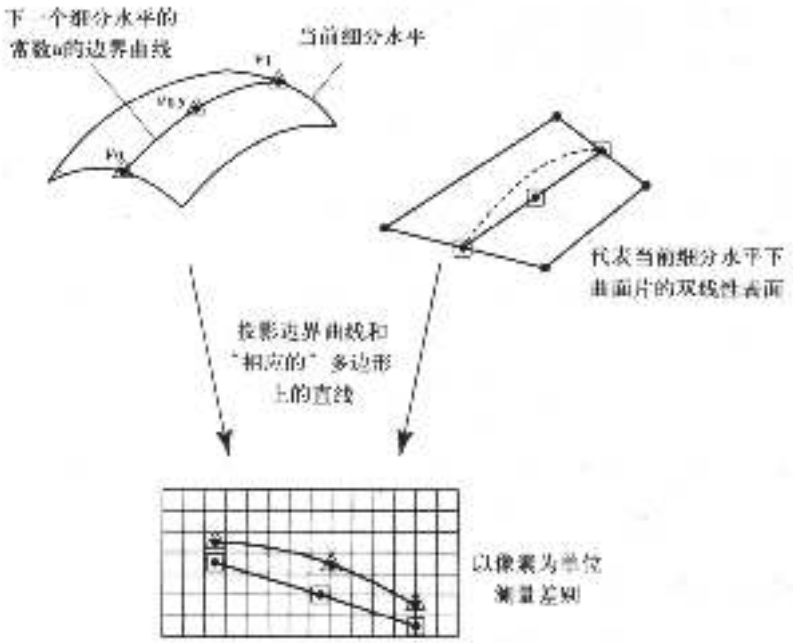


图4-10 屏幕空间细分结束在当前细分水平，提高了屏幕空间中曲面片的平滑度

3. 轮廓边的屏幕空间平滑度

如果是为了使表示物体的多边形个数达到最少，则由于明暗处理算法的效率问题，最好的策略是采用两种多边形分辨率，一个用于内部曲面片，而较精确的一个用于轮廓边。所谓算法效率问题，是指我们可以忍受一个相当粗糙的多边形近似，只有轮廓边处除外。两者之间的折中示于图4-11中。图4-11a为用小的屏幕空间平滑度判据进行细分的一个球。在这个图中，由于这样细分而得到较好的轮廓边。图4-11b通过把细分过程集中在轮廓边处，用远远少于上图的多边形数保持了绘制的质量。然而，这一策略在这种情况下的应用是需要先决条件的。依观察情况而定是指除非采用一些相关的策略，否则当出现任一视点变化时都必须实行新的细分操作。



a) 是可以消除轮廓边处分段线性的细分



b) 用较大的"内部"多边形保持相同的图像质量

图4-11 轮廓边的屏幕空间平滑度

现在来讨论轮廓边检测的问题，或者更确切地说，讨论如何检测那些含有一条轮廓边曲线的曲面片。抛开含有褶皱或尖点的曲面片不谈，观察空间中的一条轮廓边由满足下式条件的曲面片上的点来定义：

$$N \cdot L_{os} = 0$$

其中，N是表面法向量，L_{os}是观测向量的边界线。

另一种表示方法是对于图像平面在$z = 0$的平行投影，有：

$$N_z = 0$$

对于在轮廓边上的点，其表面法向的z分量为零。如果曲面片含有一条轮廓边，则会有这样一个N_z为正的区域以及一个N_z为负的区域。N_z由下式给出：

$$N_z = \frac{\partial x(u,v)}{\partial u}\frac{\partial y(u,v)}{\partial v} - \frac{\partial y(u,v)}{\partial u}\frac{\partial x(u,v)}{\partial v}$$

式中如果第一项总是为正，而第二项总是为负，则N_z处处为正，因此不可能包含一条轮廓边。这可以通过求控制点矩阵的16个偏导数来测试。

4.3 绘制构造实体几何表示

对构造实体几何（CSG）表示的绘制策略是完全不同的。CSG表示的突出特性是，它并不是一种“直接”的几何物体表示，而是一个必须对其进行解释的公式。在CSG表示中，物体数据库是一棵树，它通过布尔运算将一组基元物体关联起来。这种表示方法达成了强大的交互能力。而我们为此付出的代价是费时、复杂的绘制策略。为了展示这一点，我们考虑一个由两个圆柱体相交形成的物体，如图2-14所示。CSG表示只含有两个几何基元及其尺寸、其空间关系以及把它们结合在一起的集合操作符。这些信息并不能为绘制程序提供“作图用的”物体，即含有像相交曲线这样的几何性质的合成物体，这些曲线必须导出，而困难的根源正是这种导出或物体的构建。

在CSG物体绘制中，所涉及的主要问题是如何从CSG数据库导出适合于绘制的物体表示。已经形成了三种技术：

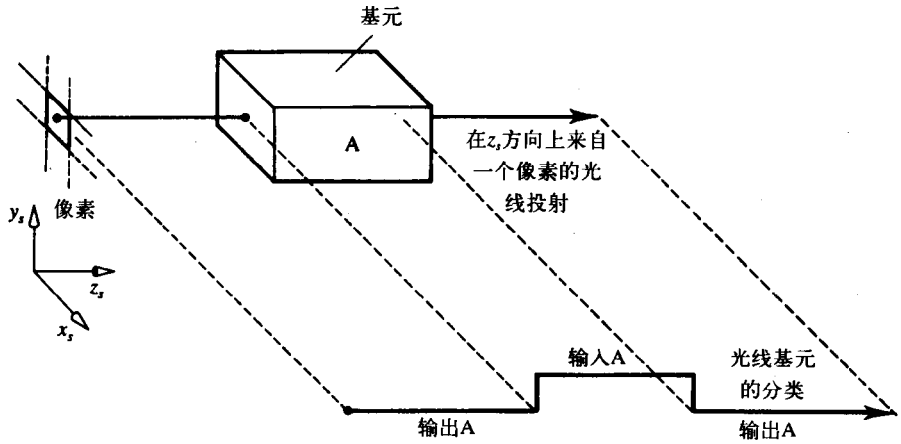
1) CSG光线跟踪。

2) 转换为体素表示，接着进行体绘制（见第14章）。

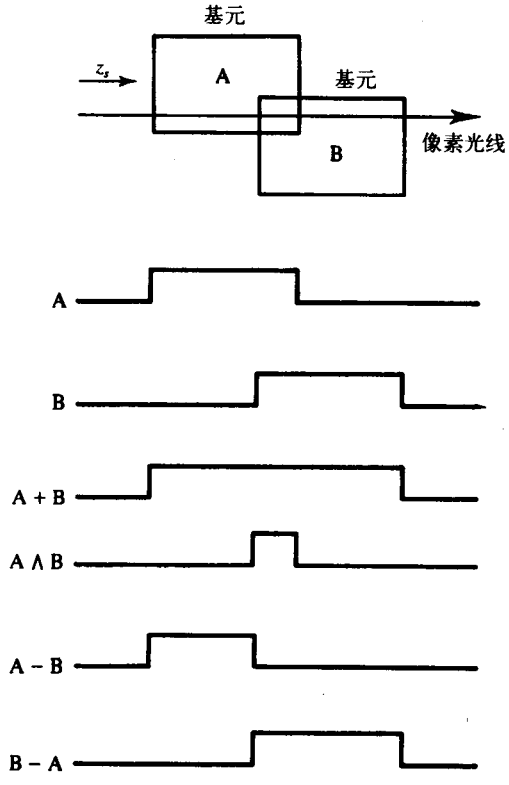
3) 用Z缓冲器算法（见6.6.2节）。

在本节中我们简单地论述其中的第一种方法，这是对第10章中所讲述的标准光线跟踪技
138 术的一种非常直接的改进。

可以通过将问题转化为一维问题求出用CSG表示方法描述的物体。为了做到这一点，观察平面中的每一个像素投射一条光线。在最简单的情况下（平行投影），我们用一组平行光线来探究物体空间。处理过程分两步进行。首先考虑单条光线，将每一个基元实例与这条光线相比较，看光线是否与基元相交。这意味着求解一个如第1章所述的线性二次曲线相交问题。任何相交都以深度z保存。然后，对每一束光线进行光线/基元的分类，并可以检查沿着光束所遇到的前两个基元的布尔和。从图4-12b可以很容易看出，沿着一条光线求基元之间的布尔运算是直截了当的。之后可通过在沿着光线方向上的第一次相交处应用一个简单的反射模型求得明暗处理值。图4-12b提示，这一点是随基元之间的布尔运算而变的。



a) 导出一种光线/基元的分类



b) 沿着一条光线求布尔运算

图 4-12

以这种方式进行的光线跟踪是一种绘制CSG模型的“经典”方法，但需付出很高的代价。最后需要注意的是，此方法被集成到下面的方法之一中：

1) 从CSG表示中求边界表示。

2) 隐藏面消除——只考虑最靠近观察者处的两个基元之间的相交。

3) 明暗处理计算。

需注意的是，所有这些操作都是在屏幕空间进行的，我们应直接从CSG表示的空间映射到屏幕空间。如果采用如上所述的平行投影的方法，即可避免涉及观察空间的复杂性。

4.4 绘制体素表示

选择最终由体素表示的物体的绘制是预先由应用决定的。正如我们已经介绍过的，在光线跟踪算法中可能会选择体素表示法，在这里绘制方法即是光线跟踪算法。这时，体素将表示实体或常规物体，其精确度依赖于体积。

在体视化过程中，体素通常代表一种性质而不是物理上的物体。例如，在一个由X射线CT扫描仪导出的物体中一点的X射线吸收系数。在这里我们不仅必须决定如何绘制，而且必须决定我们希望绘制的是什么。其含义是性质在整个数据空间中是变化的。每一个体素不再只代表物体是否存在。取而代之的是，我们有一个在其所占据的空间中处处有定义的物体，即它不再是一个实体物体的边界。我们所感兴趣的是观察物体内部信息的变化。

在这样的应用中，我们可能只希望绘制边界信息使得体素集看起来像一个实心的物体；或者我们可能希望体绘制，这就意味着使用透明表示。这里遇到可视化物体（通常是身体内部的一个器官）的模拟问题，就好像该器官是由彩色玻璃构成的，而其颜色根据该器官的实际性质如想象的那样变化。如果，在一个头部的CT数据集中，我们希望绘制颅骨，则必须采集那些表示骨头的值的体素，然后绘制这些体素。或者，也许我们希望绘制脑部，但这时也会表示颅骨，只是这时颅骨在图像中是作为一种透明的物体表现出来。

139 ~ 140

对于绘制一个体素集的边界，一种粗略的方法是简单地找出这些体素，对它们进行明暗处理计算，就好像这些体素都是用标准的多边形网格绘制程序绘制的立体物体。很明显，这样做会产生一块一块的外观。更通常的方法是用一种称为步进立方体算法（marching cube algorithm）的简捷算法，用一个具有多边形小平面的表面去拟合体素集。这种算法用于提取图2-5b中绘制的物体的表面。这种算法的明显缺点是将产生非常大量的多边形。步进立方体算法的详细讨论见第14章。

4.5 绘制隐函数

像CSG方法一样，隐函数表示是难于绘制的。原则上来说，它与CSG方法相似，即模型并不是直接表示几何形状，而是一个公式，必须从其中提取几何形状的信息。这个问题的描述是简单的。我们有物体的定义，其表面上的点是由一个标量值定义的。然后，在所有用于建模物体的产生器的影响空间内的无限点集中选择一个子集。有两个常用的方法。

第一个方法是对每一个产生器的无限点集的估算的近似。它把定义映射到一个体素集上，然后用对这种物体的一种绘制策略对体素集进行绘制。这一过程的一种直接但昂贵的方法如下。对于每一个产生器，我们在其影响范围内对每一个体素求一个标量值，并在每一个体素中对所有包围它的产生器求和。然后提取出等值面，并用如步进立方体算法这样的绘制方法进行绘制。

另一种方法是对光线跟踪球状物体算法（见第12章）的一种简单的扩展。对于每一条光线，解一组同步方程，该方程定义光线和函数$F(P)$，得到光线与等值面的交。执行过程是：

1) 找出对物体所在区域有贡献的产生器，方法是找出在光线的路径上与每一个产生器的作用球相交的光线。

2) 计算每一个产生器的光线与球的相交。

3) 找出最近的和最远的相交点，得到与物体的相交点。

图2-20（彩色插图）中展示的物体就是用这种方法绘制的。 141

第5章　绘图流程（1）：几何操作

5.1　绘图流程中的坐标空间
5.2　在观察空间中进行的操作
5.3　先进的观察系统（PHIGS和GKS）

引言

绘图流程的目的是选取一个三维空间中的场景描述，将其映射到观察表面（即监视器屏幕）上的二维投影中。尽管有各种三维显示设备，但大多数计算机绘图的观察表面是二维的，最常见的是类似电视的监视器。在大多数虚拟现实的应用程序中，产生一对投影，并在头盔上安装的小的监视器上显示出来，又称为安装在头上的显示器（head-mounted display，HMD）。在这种情况下，唯一的不同点是有一对二维投影而不是一个投影，而操作是相同的。

接下来我们将考虑多边形网格模型的情况。可以把绘图流程中涉及的各种过程简单地分成两类——几何的（本章）和算法的（第6章）。几何过程涉及对于多边形顶点的操作，即把顶点从一个坐标空间变换到另一个坐标空间或者删除不能从视点看到的那些多边形。绘制过程涉及明暗处理和纹理映射，并且比几何操作的代价更高，大多数几何操作涉及矩阵乘法。

图5-1是一个绘图流程的连续过程。从该图中可以看出，这是一个贯穿各种三维空间的过程，即把物体的表示从一个空间变换到另一个空间。最后一个空间称为屏幕空间，在这个空间中进行绘制操作。我们很快会发现这个空间也是三维的。

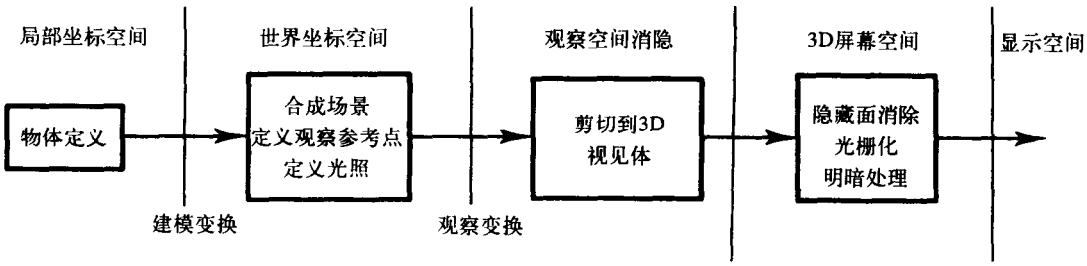


图5-1　一个三维绘图流程

5.1　绘图流程中的坐标空间

5.1.1　局部坐标系或建模坐标系

为了建模的方便，将多边形网格物体的顶点与某些位于物体或靠近物体的点联系起来存储是有意义的。例如，我们几乎总是希望将立方体的原点置于立方体的某一个顶点上。或者，我们可能希望将一个由某实体旋转而产生的物体的对称轴作为z轴。而在将多边形的顶点保存在物体所处的坐标系中的同时，我们可能还希望存储多边形的法线和顶点的法线。当将局部

变换应用于物体的顶点时，对其相关的法线也要实施相应的变换。

5.1.2 世界坐标系

一旦对物体进行了建模，下一步就是将其放置于我们希望绘制的场景中。所有构成场景的物体都有其自己的局部坐标系。场景的全局坐标系称为“世界坐标系”。为了定义所有物体之间的空间关系，必须将这些物体变换到这个公共空间中。将一个物体放置于场景中的动作定义了把物体从局部空间放到世界空间所需的变换。如果要使物体成为动画，则动画系统提供一种时变变换，该变换把物体放在一个帧的世界空间中。

场景在世界空间中被照亮。光源是具体指定的，如果绘制程序函数中的明暗处理器处于世界空间中，则物体的法向量必须经历的变换是最终的变换。物体的表面属性（纹理、颜色等）均在这个空间定义并建立联系。

5.1.3 摄像机/眼睛/观察坐标系

142 ~ 143

眼睛、摄像机或观察坐标系是用于建立观察参数（观察点、观察方向）以及视见体的空间。经常将虚拟的摄像机用作观察系统中的一个类比。但是，如果建立了这样的类比关系，我们必须小心地区分外部摄像机参数和内部摄像机参数，外部摄像机参数指它的位置以及方向，这个方向是指向内的。同时还要区分那些影响胶片平面上的图像的性质和尺寸的因素。大多数绘制系统都模仿摄像机，该摄像机在实际使用中可以是一个完善的针孔（或镜头）设备，胶片平面可以放在相对于针孔的任意距离。但是，在计算机图形学中有另一些设备，而这些设备是不能用摄像机来模仿的。正因为如此，类比的作用是有限的。

下面详细讨论一个基本的观察坐标系以及从世界空间到观察坐标空间的变换。这个空间的存在原因是有一些操作（和定义）在观察空间中执行最方便，最终我们会直接从世界空间转到屏幕空间。标准的观察坐标系正像PHIGS图形标准中所定义的，允许用户定义很多的设备，从这一角度考虑该标准很复杂。我们将在5.3节中讨论这一标准。

我们把观察系统定义为对观察坐标与对某些工具（如视见体）的定义的一种结合。最简单成最小化的系统可由下列元素组成:

- 一个观察点，它在世界空间中建立观察者的位置。这个点可以是观察坐标系的原点，或者是投影中心与观察方向N的组合。
- 一个关于观察点的观察坐标系。
- 一个观察平面，场景的二维图像投影到该平面上。
- 一个观察平截体或观察体，它定义观察的范围。

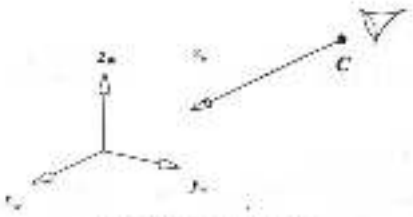
这些实体示于图5-2中。观察坐标系UVN的N与观察方向共线，V和U位于平行于观察平面的平面中。可以认为坐标系的原点是观察点C。含有V和U的观察平面是无限扩展的。我们定义一个观察平截体或观察体，它在观察平面定义了一个窗口。这个窗口中的内容是视见体中所包含的场景中的那一部分的投影，该投影最终会出现在屏幕上。

这样的话，利用一个虚拟的摄像机，我们就有了一个可以放置在世界坐标空间中任意位置、指向任意方向并关于观察方向N旋转的摄像机。

为了对世界坐标空间中的点进行变换，我们引入了一种对坐标系变换的改变，其过程有两个组成部分：一个变换部分和一个旋转部分（见第1章）。于是:

$$\begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \boldsymbol{T}_{view} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

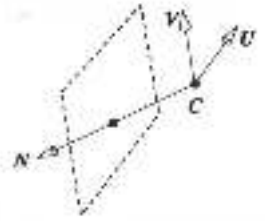
144



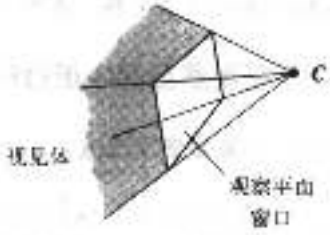
a) 观察点C和观察方向N



b) 一个与观察方向N垂直，并位于距离C点d个单位的观察平面



c) 一个原点为C、UV轴嵌在与观察平面平行的平面中的观察坐标系



d) 一个视见体，通过由C和观察平面窗口形成的平截体定义

图5-2 实际的观察系统中所需要的最小实体

其中：

$$\boldsymbol{T}_{view} = \boldsymbol{RT}$$

并且：

$$\boldsymbol{T} = \begin{bmatrix} 1 & 0 & 0 & -C_x \\ 0 & 1 & 0 & -C_y \\ 0 & 0 & 1 & -C_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{R} = \begin{bmatrix} U_x & U_y & U_z & 0 \\ V_x & V_y & V_z & 0 \\ N_x & N_y & N_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

现在剩下的唯一问题是为这个系统定义一个用户界面，并把所用到的所有参数通过该用户界面映射到U、V和N上。用户需要指定C、N以及V，C很容易给出，N为观察方向或观察平面的法向，可以输入，例如在一个球坐标系中用两个角度来作为输入值，这似乎很直接。 145

θ：方位角；

ϕ：仰角，

其中：

$$N_x = \sin\phi\cos\theta$$
$$N_y = \sin\phi\sin\theta$$
$$N_z = \cos\phi$$

V的问题最多。例如，用户可能需要指向与在世界坐标系中等同的高度，但不能通过下面

的设定达到这一点：

$$V = (0, 0, 1)$$

因为V必须与N垂直。一种合理的策略是允许用户对V定义一个近似的方向，比如说V'，并使系统计算出V值。图5-3演示了这种做法。V'为用户定义的向上的向量。将这一向量投影到观察平面上：

$$V = V' - (V' \cdot N)N$$

并单位化。U可以定义，也可以不需要用户定义。如果U是未定义的，则可通过下式获得：

$$U = N \times V$$

这导致一个左手坐标系。这虽然有些不协调，但确实与我们对一个实际的观察系统的感觉一致，随着观察方向坐标轴上的值的增加，距离观察点的距离也增加。当用UVN坐标建立了观察变换之后，
146 在接下来的章节中我们将用(x_e, y_e, z_e)来定义观察坐标系中的点。

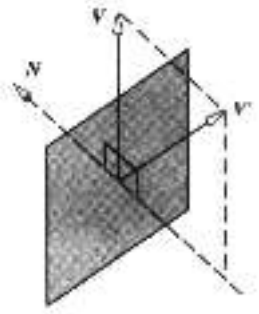
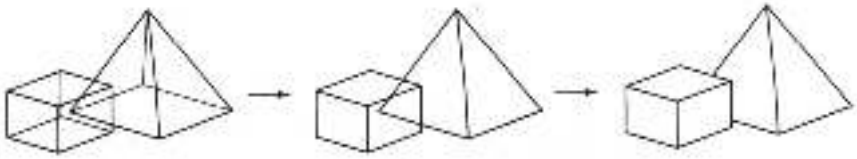


图5-3 向上的向量V可以从V'给出的“指示”求出

5.2 在观察空间中进行的操作

5.2.1 消隐或背面清除

消隐（culling）或背面清除（back-face elimination）将完整的多边形的方向与观察点或投影中心相比较，去掉那些看不到的多边形。如果场景中包含一个凸状物体，则消隐过程就退化为隐藏面消除（hidden surface removal）。当一个多边形将另一个多边形部分地遮挡住的时候，总是需要一个一般化的隐藏面消除算法（见图5-4）。平均来说，多面体中有半数多边形都是背向的，而这个处理过程的优点就是通过简单的测试就可以去掉这些多边形。



a) 消隐去掉不可见的多边形　　b) 隐藏面消除处理一般性的问题：多边形的部分遮挡其他多边形

图5-4 消隐和隐藏面消除

对于可见性的测试是直接的，并且最好在观察空间进行。首先计算多边形的外向法线（见第1.3.3节），然后测试这个法线与来自投影中心的向量的点积的符号（见图5-5）。于是有：

可见性：$= N_p \cdot N > 0$

其中：

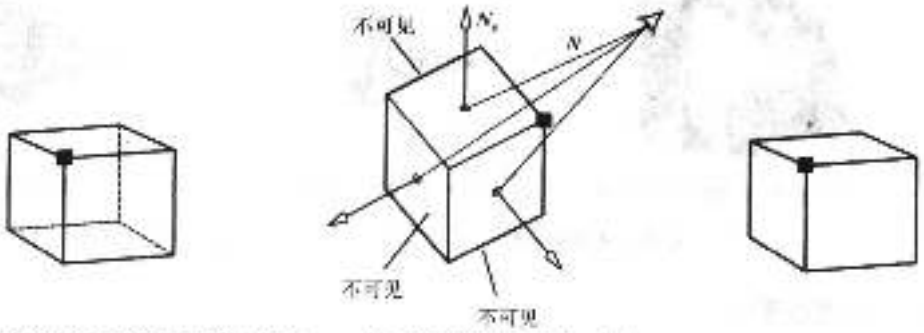
N_p是多边形法线；

N是“视线”向量。

5.2.2 视见体

在图5-2中，引入了一个半无限的金字塔作为视见体。在很多应用中，这个金字塔被进一

多限定为一个一般的视见体，它由一个观察平面窗口、一个近的裁剪平面和一个远的裁剪平面组成。但是为了简化计算，我们将免去近处的裁剪平面，因为这个平面只有很有限的实际利用价值。我们重新考虑一个视见体，它由一个观察平面窗口和一个远的裁剪平面构成（见图5-6）。我们注意到，该远的裁剪平面是一个与观察方向垂直的横断面，任何超过这一平面的物体都是不可见的。在实际过程中，我们把一个半无限的金字塔变成了一个无限体。远的裁剪平面是非常有用的，当对一个非常复杂的场景进行绘制时，需要进行处理的多边形的个数可以通过使用远的裁剪平面而减少。例如，在三维计算机游戏中，当飞过一个环境时我们可以定义一个远的裁剪平面，并使用深度可调节的迷雾方法，在物体突然与远的裁剪平面相交时，消除物体“打开”时出现的扰动。



a) 对物体所希望的观察（背面以点划线示出）　b) 消隐操作的一种几何观察　c) 消隐后的物体

图5-5 消隐或背面清除

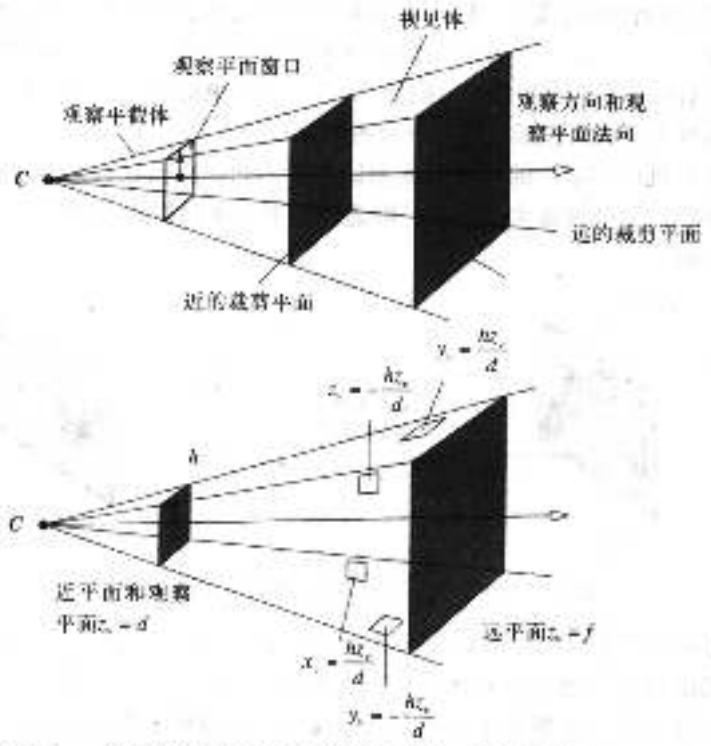


图5-6 一个实际的视见体：近的裁剪平面与观察平面共线

如果进一步从几何上简化，定义一个正方形的观察平面窗口，其大小为2h，关于观察方向对称放置，则定义视见体大小的四个平面由下式给出：

$$x_r = \pm\frac{hz_r}{d}$$

$$y_r = \pm\frac{hz_r}{d}$$

关于视见体的裁剪（见图5-7）现在可以通过1.4.3节中给出的多边形平面相交计算来完成。这个图演示了裁剪的原理。但是我们将看到，所涉及的计算若在三维屏幕空间执行将更有效。



a) 将视见体外部的多边形删掉　　b) 视见体内部的多边形被保留　　c) 与边界相交的多边形被裁剪

图5-7 关于视见体的裁剪——绘图流程中一个程序化的多边形操作

5.2.3 三维屏幕空间

绘图流程中最终的二维空间称为三维屏幕空间。在这个空间中，我们进行（实际的）视见体的裁剪，以及执行我们将在后面章节中阐述的绘制过程。使用三维屏幕空间是因为它简化了裁剪和隐藏面消除的操作，隐藏面消除的操作是通过对投影到相同像素上的不同物体的深度值来实现的。另外，这个空间中的物体最终都要经过一个向二维观察平面坐标变换的过程（这个过程有时称为透视分割）（术语"屏幕"和"观察平面"的含义稍有不同。严格地讲，屏幕坐标系是从视平面坐标系通过基于设备的变换导出的）。

由于在计算机图形学中的观察表面被认为是平的，所以我们只考虑称为平面几何投影的那类投影。现在讨论两种基本的投影，即透视投影和平行投影。图5-8给出了这两种投影以及它们性质的差别。



图5-8 用平行投影和透视投影向一个平面投影的两个点

透视投影在计算机图形学中更普及或是一种普遍的选择，因为它更简练。在透视投影中，并不保存相关的尺寸，远距离的线以比近距离的相同长度的线短的形式显示（见图5-9）。这种效果使人们能够在二维照片中或者在三维真实空间的格式化表示中感觉到深度。透视投影以一个称为投影中心的点为特征。三维空间中的点向观察平面的投影是取从每一点到投影中

心连线的交。这些连线称为投影线。

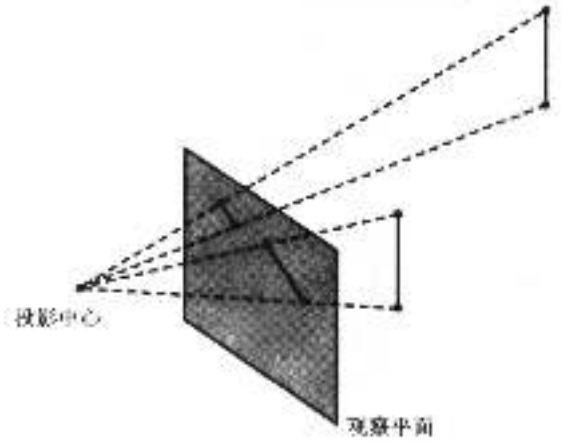


图5-9 在透视投影中，远距离的线段比近距离的相同长度的线段短

图5-10显示了透视投影是如何导出的。点$P(x_e, y_e, z_e)$是观察坐标系中的一个三维空间点。该点被投影到垂直于z_e轴的一个观察平面上，点的位置距坐标系的原点距离为d。点P'是该点在观察平面上的投影。它在观察平面坐标系中有一个二维的坐标(x_s, y_s)，观察平面坐标系的原点位于z_e轴与观察平面的交点处。 150

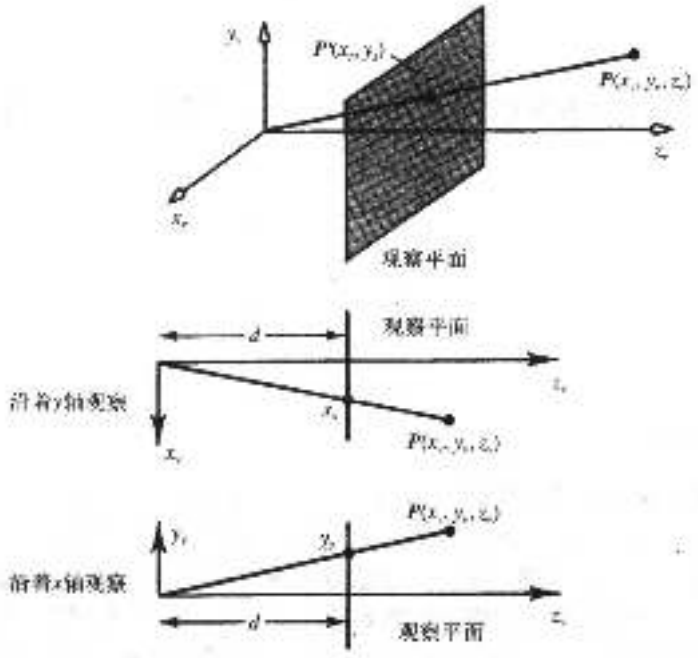


图5-10 导出一个透视变换

由相似三角形得到：

$$\frac{x_s}{d} = \frac{x_e}{z_e} \qquad \frac{y_s}{d} = \frac{y_e}{z_e}$$

为了将这一非线性变换表示成一个4 × 4的矩阵形式，我们将其分成两部分来考虑，即一个线性部分和一个非线性部分。若采用齐次坐标，有：

$$X = x_v$$
$$Y = y_v$$
$$Z = z_v$$
$$w = z_v / d$$

我们可以写：

$$\begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} = \boldsymbol{T}_{\text{pers}} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix}$$

其中：

$$\boldsymbol{T}_{\text{pers}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$$

151

接着进行透视分割，我们有：

$$x_s = X/w$$
$$y_s = Y/w$$
$$z_s = Z/w$$

在平行投影中，如果观察平面与投影方向垂直，则投影是正投影，我们有：

$$x_s = x_v \quad y_s = y_v \quad z_v = 0$$

表示成矩阵形式为：

$$\boldsymbol{T}_{\text{ort}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

5.2.4 视见体和深度

现在考虑把上面这种简单的变换进行扩展，使其包含图5-6中所引入的简化的视见体。我们将详细讨论屏幕空间的第三个分量的变换，即到目前为止一直被忽略的z_s这一分量，因为导出这一分量的变换有一些敏感。现在，在绘制图像时所涉及的大量计算都在屏幕空间发生。在屏幕空间中，多边形是关于扫描线和像素进行裁剪的，而隐藏面的计算也是在这些裁剪过程所得的片段上进行的。为了进行隐藏面的计算（在Z缓冲器算法中），必须对多边形中的任意点产生深度信息。实际应用中，这意味着已知屏幕空间中的一条线和平面，平面和直线可以相交，沿着此直线从线的两个端点的深度对这个交点的深度进行插值。除了从观察空间向屏幕空间移动，将一些线段变换成另一些线段以及从一些平面变换到另一些平面的操作之外，在屏幕空间这是唯一有意义的操作。可以看到（Newman and Sproull 1973），假如对z的变换

取下式的形式，则这些条件是可以满足的：

$$z_s = A + B/z_v$$

其中，A和B为常数。这些常数根据下列限制条件确定：

1) 选择$B<0$，以便使z_v增加时z_s也增加，这样就保持了我们对深度的欧几里德表示法的一致。如果一个点位于另一个点的后边，则它的z_v值较大，如果$B<0$，它也有一个较大的z_s值。

2) 一个重要的实际应用考虑是我们要存储的深度值的精确度。为了使这个精确度尽可能高，我们把z_s的值域范围单位化，使区间$z_v \in [d, f]$映射到区间$z_s \in [0, 1]$。 152

考虑图5-6中的视见体，完全的透视变换由下式给出：

$$x_s = d\frac{x_e}{hz_v}$$

$$y_s = d\frac{y_e}{hz_v}$$

$$z_s = \frac{f(1-d/z_v)}{(f-d)}$$

其中，对x_s和y_s的变换中出现了常量h，这个值保证使x_s和y_s的值在正方形的屏幕上落于范围$[-1, 1]$内。采用与5.2.3节中相似的处理方法，我们有：

$$X = \frac{d}{h}x_v$$

$$Y = \frac{d}{h}y_v$$

$$Z = \frac{fz_v}{f-d} - \frac{df}{f-d}$$

$$w = z_v$$

得到：

$$\begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} = \boldsymbol{T}_{\text{pers}} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix}$$

其中：

$$\boldsymbol{T}_{\text{pers}} = \begin{bmatrix} d/h & 0 & 0 & 0 \\ 0 & d/h & 0 & 0 \\ 0 & 0 & f/(f-d) & -df/(f-d) \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{5-1}$$

现在可以把从世界空间到屏幕空间的整个变换写成一个变换，这个变换通过把观察变换和投影变换连接起来得到：

$$\begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix} = \boldsymbol{T}_{\text{pers}}\boldsymbol{T}_{\text{view}} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

这一公式对于更进一步地考察z_v和z_s之间的关系有指导意义。尽管在构造这一变换的过程中我们已经看到，两种变换都提供了一种对点的深度的量度，但是在观察空间沿着一条线进
153 行插值与在屏幕空间沿着这条线进行插值是不同的。图5-11展示了这一点。在z_v上的相等间隔与z_s中的相应间隔相比，随着z_v向远的裁剪平面移动，z_s以更迅速的方式靠近1。于是屏幕空间中的物体朝着观察平截体的后方推动并扭曲。这一差别导致在插值一些量时出现错误。插值屏幕空间的位置不会出现这种情况。

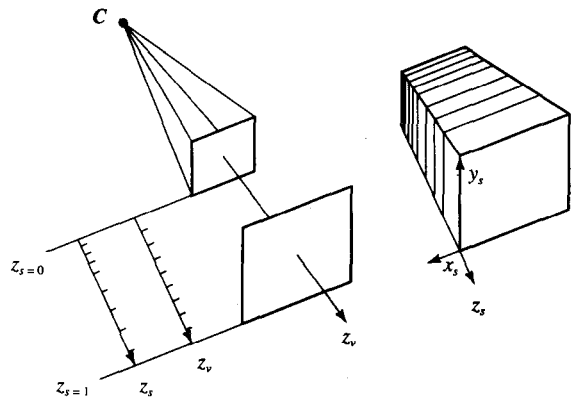


图5-11 由于z_v到z_s的变换而产生的三维空间中的扭曲示意图

除了这个困难之外，由于其非常高的结构性，屏幕空间非常适合于进行隐藏面的计算。这时，由于将投影中心沿着z_s轴移到了负无限远处，所以所有通过观察点的光线都平行于z_s轴。可以通过把$z_v = 0$代入到上面的方程中得到$z_s = -\infty$来看到这一效果。把屏幕空间中进入眼睛的那些光线设为平行意味着隐藏面的计算只需要在那些具有相同坐标（x_s, y_s）的点上进行。而测试只是对z_s值进行比较，以区分某一点是否在另一个点的前方。有一个盒子，其一个边平行于图像平面，对该盒子的变换如图5-12所示。在这里，从盒子的顶点到观察点的射线在三维屏幕空间中是平行的。

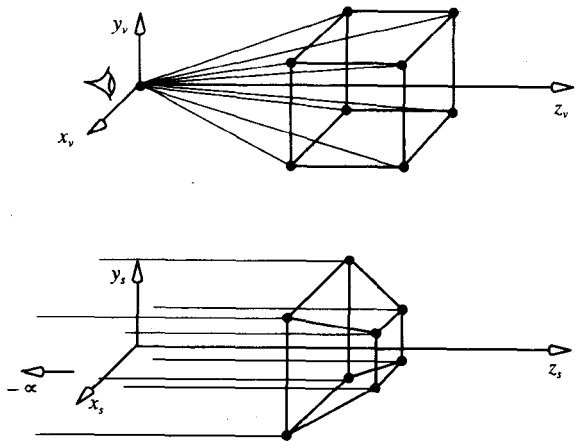


图5-12 将盒和光线从观察空间变换到屏幕空间

屏幕深度所要求的总的精确度是一个场景复杂度的函数。对于大多数场景，8位是不够的，通常16位就足够了。精确度不够的效果是很容易看到的，比如在两个相交的物体的连接处使用Z缓冲器算法时，如果在两物体的相交处物体有一个弯曲，则随着屏幕深度的精确度的降低，产生的走样会严重增加。

现在回到裁剪的问题上来。从图5-12很容易看出，在屏幕空间用齐次坐标表示时，视见体的各个面是平行的。这就意味着裁剪的计算降为对界限的比较，即我们不必再把点代入平面方程。在透视分割之前必须首先对齐次坐标执行裁剪操作，而把对观察平截体的定义变换成齐次坐标表示为我们提供了对裁剪的限制。

$$-w \leqslant x \leqslant w$$
$$-w \leqslant y \leqslant w$$
$$0 \leqslant z \leqslant w$$

把由方程（5-1）表示的从观察空间向眼睛空间的变换分离成一个乘积来考虑也是有指导意义的：

$$\boldsymbol{T}_{pm} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & f/(f-d) & -df/(f-d) \\ 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} d/h & 0 & 0 & 0 \\ 0 & d/h & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \boldsymbol{T}_{pers2}\boldsymbol{T}_{pers1}$$

这种分离对过程的可视化是很有用的。第一个矩阵是对x和y的一种按（d/h）的重新划分。这样做的效果是把一个被截断的、其各个面的侧斜度由h/d来确定的金字塔转换成一个规则的、其各个面的侧斜度均为45°的金字塔（见图5-13）。例如，将点

$(0, h, d, 1)$ 变换到 $(0, d, d, 1)$

和将点

$(0, -h, d, 1)$ 变换到 $(0, -d, d, 1)$

第二个变换将这个规则的金字塔变换成了一个盒子。近处的平面映射到 (x, y) 平面，而远处的平面映射到$z = 1$的平面。例如，点

$(0, d, d, 1)$ 变换到 $(0, d, 0, d)$

这个变换等价于 $(0, 1, 0, 1)$。

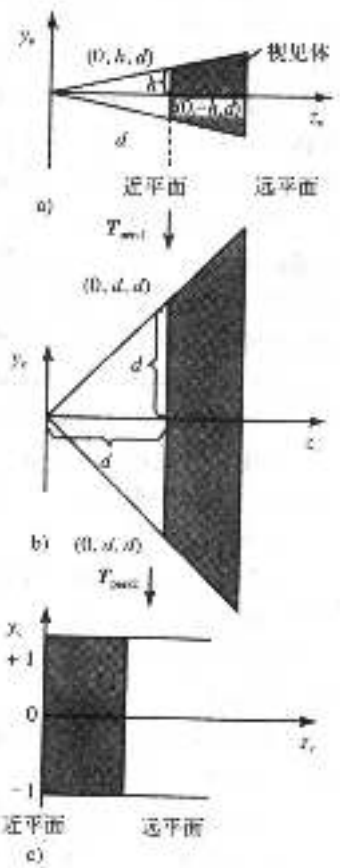


图5-13 用两个矩阵变换将视见体变换成标准型（一个盒子）

5.3 先进的观察系统（PHIGS和GKS）

由图形学标准PHIGS和GKS定义的观察系统比我们已描述的系统更一般化，而且更难于实施和理解。标准的观察系统的一个很不好的方面是，由于其要提供这种一般性，因此也就很繁琐，并且很难将其与接口进行连接，即使采用了其参数的一个子集，也必须对那些未用到的默认值有正确的理解。也许这是不可避免的。从某种意义上来说，一个标准的函数并不能反映其通常的用途，而只是定义了一组完整的工具，在通用的观察系统中用户可能会用到这些工具。但是，其中的某些工具几乎从没有人用过。我们之所以把PHIGS观察系统作为学习的主题，是因为不管你遇到什么样的观察系统，这个系统都无疑是PHIGS的一个子集。

154 / 156

5.3.1 PHIGS观察系统概述

下面综述一下上一节中所描述的最小观察系统的扩展。这些扩展是:

1) 观察点或视点的概念。这一点建立了观察坐标系的原点，这时放弃了投影中心。在PHIGS中等价于观察空间的是观察参考坐标系（reference coordinate system，VRC），它是通过定义一个观察参考点（view reference point，VRP）来定义的。而投影中心是通过定义一个投影参考点（projection reference point，PRP）来单独定义的。

2) 性质1）意味着从投影中心到观察平面窗口的中心之间的连线不必平行于观察平面的法向，即可以有斜投影。这与用虚拟摄像机类比时允许胶片平面相对于摄像机所对的方向有倾斜是等价的。在一些摄像机设计中利用了这一点来校正透视的变形，比如在从地面为一个高的建筑物照相的情况下。

3) 定义了近的裁剪平面、远的裁剪平面以及观察平面。在前面提到的观察系统中，我们使后裁剪平面与观察平面共面。

4) 定义了一个观察窗口，该窗口可以有任意图像纵横比，并可以位于观察平面中的任一位置。在前面提到的观察系统中，我们定义了一个正方形的窗口，它被对称地置于观察平面的中心。

5) 可以定义多个观察参考坐标系。或者，可以建立对一个场景的多个观察。

考虑观察系统中的距离的概念。我们以前使用过观察点的距离的概念，用来反映一种直觉，即观察点距离场景越远，该场景在观察平面上的投影就越小。这时出现的问题是，在任何真实的系统中或者在计算机图形学系统中并没有这样的观察点。一个投影中心、一个观察平面在摄像机或眼睛的模拟物中就够了。在摄像机中，观察平面或者胶片平面在摄像机中。场景的投影由摄像机与物体的距离和镜头的聚焦长度来确定。但是在计算机图形学中，我们可以相对于投影和场景的中心任意地移动观察平面。实际中并没有这样的镜头。那我们如何能对距离进行划分呢？是用观察平面与世界坐标系原点之间的距离，还是用投影中心与世界坐标系原点之间的距离，或是用观察平面与投影中心之间的距离呢？像PHIGS这类通用系统让用户来回答这样的问题。也许正是PHISG观察系统的这一性质使得它看起来很繁琐。

157

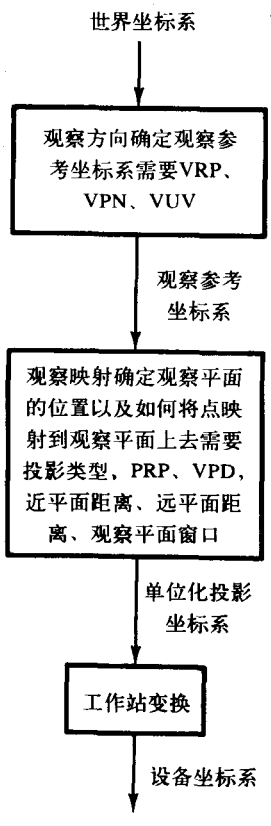


图5-14 建立一个观察系统PHIGS

PHIGS将观察系统分成三个阶段（见图5-14），确定观察平面的位置和方向称为观察定向。需要用户提供如下信息:

1) 观察参考点（VRP）。它是世界坐标空间中的一个点。

2) 观察方向或观察平面法向（view plane normal，VPN）。世界坐标空间中的一个向量。

3) 观察的上方向量（view up vector，VUV）。世界坐标空间中的一个向量。

第二个阶段称为观察映射，并确定点是如何映射到观察平面上的。这需要下列信息:

1) 投影类型（平行的还是透视的）。

2) 投影参考点（PRP）。观察参考坐标系空间中的一个点。

3) 观察平面距离（view plane distance，VPD）。观察参考坐标系空间中的一个实数值。

4) 后平面和前平面的距离。观察参考坐标系空间中的实数值。

5) 观察平面窗口的限定。观察参考坐标系空间中的四个限定值。

6) 投影视口的限定。单位化投影空间（normalized projection space，NPC）中的四个限定值。

这些信息用于将VRC中的信息映射到单位化的投影坐标系（NPC）中。NPC空间是一个立方体，其坐标值在每一个方向都被限定在0～1的范围之内，采用这样一种空间的理由是，当需要对一个场景有多个观察时，允许建立不同的VRC（并接着被映射到屏幕上不同的观察部位上去）。每一个观察都伴随有其自身的VRC，不同的观察均被映射到NPC空间中。

在最后的处理阶段，进行工作站变换，也就是正常的根据设备而定的变换。

下面详细地讨论这些方面。

5.3.2 观察方向参数

在PHIGS中，我们建立起一个观察空间坐标系，其原点放于世界坐标系中的任意VRP位置处。这个坐标系与VPN和VUV一起定义了一个右手坐标系，其轴分别为U、V和N。N为观察方向，UV定义一个平面，该平面或者与观察平面是共面的，或者平行于它（VUV与5.1.3节中的V'具有相同的功能）。VUV和VPN的叉积确定了U，而VPN与U的叉积确定了V：

$$U = (\mathrm{VUV}) \times (\mathrm{VPN})$$

$$N = (\mathrm{VPN}) \times U$$

用5.1.3节中所给出的建议可以很容易地建立起构建VPN所需要的接口（注意：VUV必须是不平行于VPN的）。方向和映射参数之间的几何关系如图5-15所示。于是，在观察方向阶段就建立起了与世界坐标系原点相关的位置和方向，观察平面也相应于VRC进行定义。

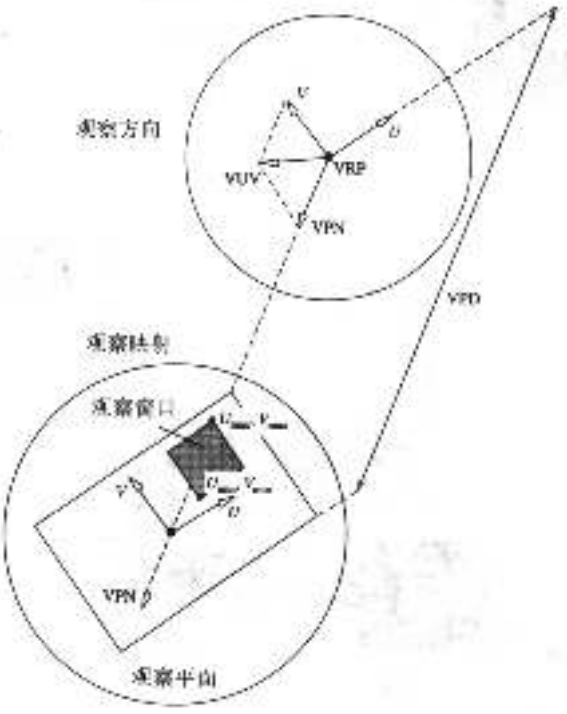


图5-15 PHIGS——观察方向和观察映射参数。注意：这是一个右手坐标系

5.3.3 观察映射参数

观察映射阶段确定如何将场景投影到观察平面上去，以及对场景的哪些部分进行投影操作。在观察方向阶段，参数是在世界坐标系中定义的。而在这一阶段，参数是在观察空间中定义的。

第一步是建立观察平面。这是一个无限延伸的平面（理论上讲），它平行于UV平面，其与VRP的距离由VPD给出。通过定义一个前平面、一个后平面以及一个观察窗口建立起一个视见体。观察窗口是观察平面上的任意矩形，它由二维观察空间中的最小最大坐标值来确定，即将UV坐标沿着VPN变换到观察平面。这些坐标值是(u_{min}, v_{min})和(u_{max}, v_{max})。对于平行投影和透视投影，这些关系如图5-16所示。

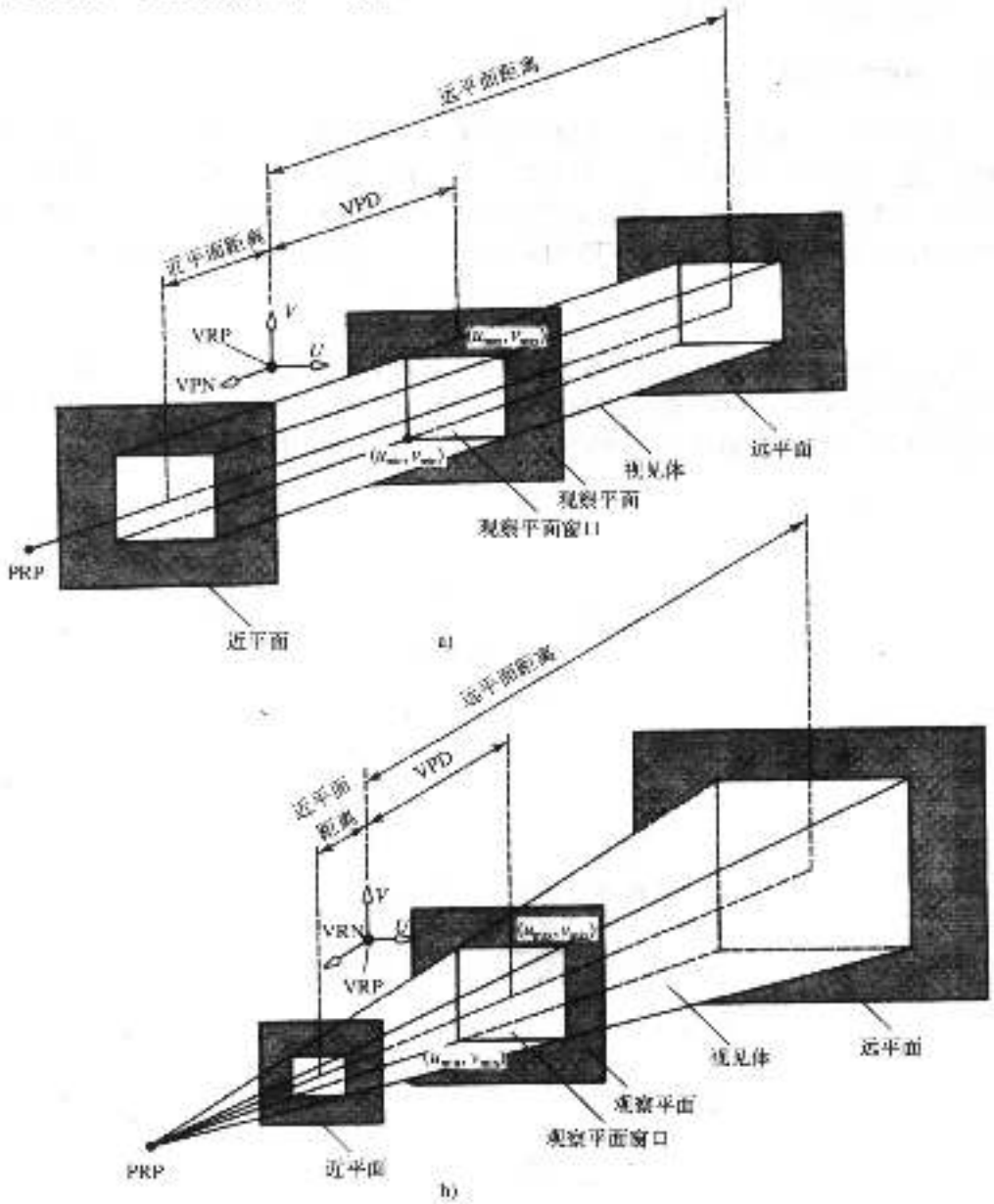


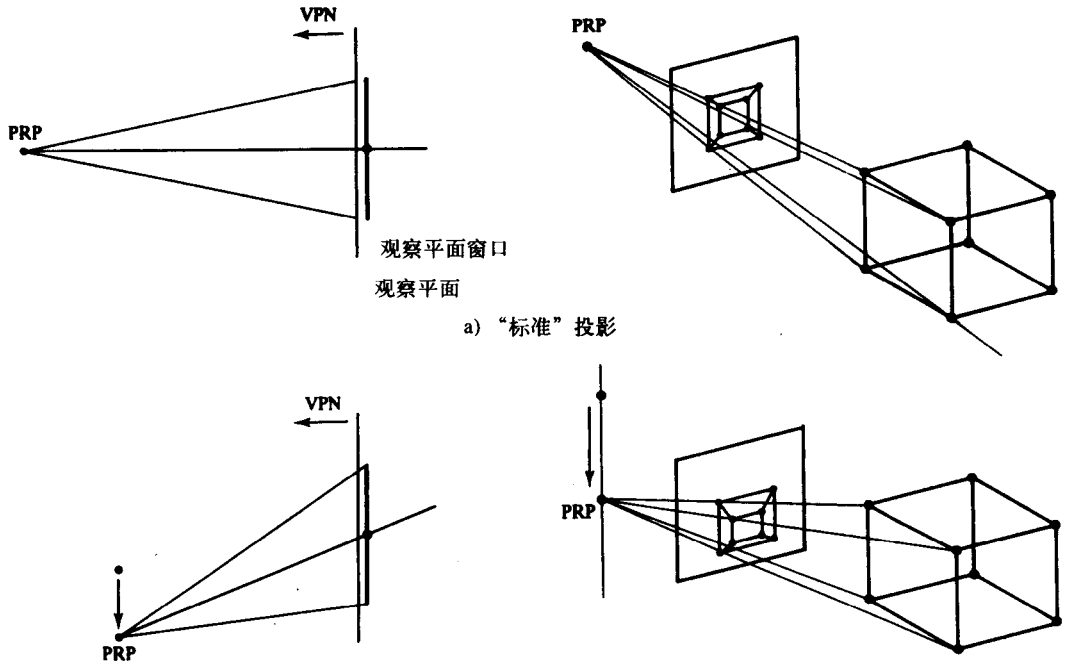
图5-16 平行投影和透视投影视见体的几何学

用投影类型和PRP结束了这个图。正如我们前面提到的，PRP不必一定平行于VPN，并通过观察平面窗口的中心。如果PRP偏离了这条线，则形成一个倾斜的投影。 158 ~ 161

PRP和观察平面窗口之间的两个可能的关系如图5-17所示。这些关系是:

1) 从PRP到观察平面窗口中心的连线平行于VPN。

2) 移动PRP产生一个斜投影。这时（1）中的条件不再满足。



a) “标准”投影

b) 通过在平行观察平面的方向垂直向下移动PRP获得斜投影

图 5-17

5.3.4 观察平面的更详细讨论

观察平面的位置和方向由观察参考点（VRP）、观察平面的距离（VPD）以及观察平面的法向（VPN）来定义（见图5-15）。这时，观察方向就由VPN设定。前面所述的系统都使用摄像机和焦点来建立VPN和产生距离d。与这些系统相比较，现在我们用两个向量，即VRP和VPN（再加上一个距离，即观察平面距离（VPD），用于确定VRP与观察平面的位置关系）。前两个系统也用到VRP，但是是作为投影中心，而现在这个系统对PRP（投影参考点）却必须单独定义。这时VRP只是一个坐标系的参考点，可以将其置于任何用起来方便的地方。例如，可以用它形成观察平面原点，或者把它放置在世界坐标系的原点，也可以放在所感兴趣的物体的中心位置处。将这一点放置于观察平面原点之外的点上有其优点，即这时VPD具有了观察距离的含义。如果将VRP置于观察平面处，则VPD为零，并且作为一个参数它是冗余的。另外，要向物体靠近或远离物体只需要改变VPD即可。 162

观察平面可以放置于世界坐标系的任意位置。可以将其置于物体之前、之后或者穿过物体内部。建立了观察平面的位置和方向之后，我们现在建立观察平面上的uv坐标系，将VRP（或者VRP对于观察平面的投影）作为原点（见图5-18）。这个二维的uv坐标系和VPN形成了

一个二维右手坐标系。这就使得观察平面窗口关于VPN进行了“扭曲”，形成了一个窗口（见图5-19）。这个“扭曲”是通过观察的上方向量（VUV）来建立的。观察的上方向量的一般作用是确定场景是否被直立地观察，或被倒着观察，或者以其他方向观察。v轴的方向通过将VUV以平行于VPN的方向到观察平面上的投影来确定。

随着建立了观察平面中的二维坐标系，可以建立起一个二维的窗口（见图5-20）。这个窗口用来将世界坐标系的未受限制的或面向应用的顶点值映射到观察平面上的相应的值。所有其他的事情都是一样的。这个窗口的建立确定了观察表面上物体的大小。

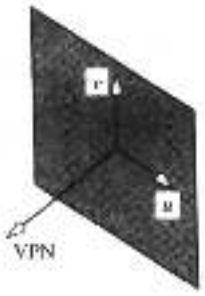


图5-18 在观察平面建立了一个uv坐标系，形成了一个带有VPN的二维左手系（对于GKS-3D和PHIGS来讲是右手系）

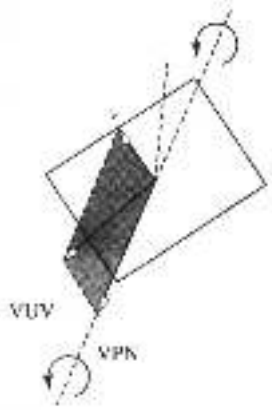


图5-19 VUV确定了v轴的方向使得观察平面关于VPN“扭曲”

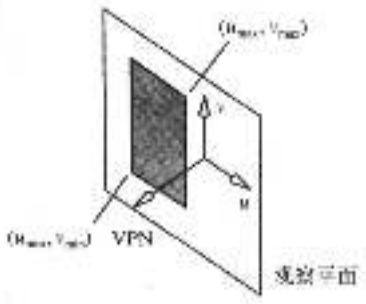


图5-20 在观察平面建立一个二维的窗口

163

5.3.5 实现一个PHIGS型观察系统

对于一个简化的观察系统建立了变换之后，可以通过对其进行扩展来处理一个PHIGS型的系统。首先，考虑PRP。在PHIGS型系统中，PRP被定义为VRC空间中的一个点，也就是说这是一个与VRC的原点相关的点。可以通过将PRP设定为原点来处理PRP与VRC之间空间上的不同。观察平面和裁剪平面的参数均为距离，这些距离必须以与投影中心相关联的形式来表示，可以通过从PRP对其进行转化来实现这种表示。我们可以简化PHIGS型观察系统的参数，使其位置的分配如图5-21所示，可以将这个图与图5-13a相比较。PHIGS型系统的另一个性质是，现在有了一个观察平面、一个近平面、一个远平面，其视见体的各个面具有不同的斜率，这是因为我们去掉了对简化系统的观察平面窗口的限制。我们所采用的规范见表5-1。

表 5-1

窗口值	从PRP变换到VRC原点之后
VPD	d
远平面距离	f
近平面距离	n
u_{min}, u_{max}	x_{min}, x_{max}
v_{min}, v_{max}	y_{min}, y_{max}

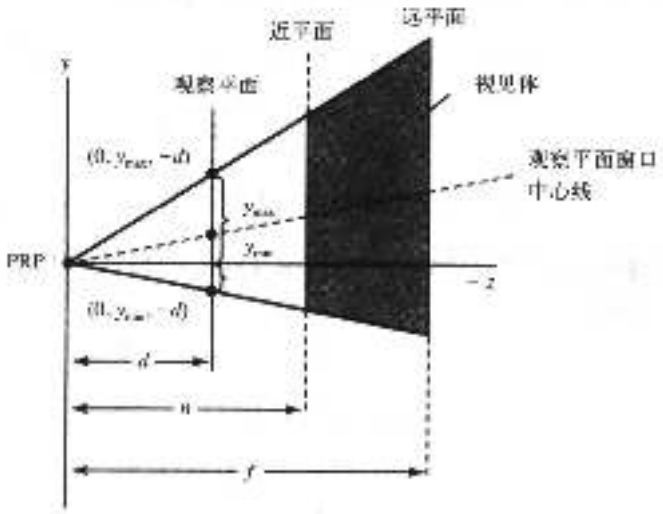
（请注意：尽管这里的f和d与简化系统中的f和d具有相同的解释，但是，它们的值不同。

在这里我们保持符号相同，目的是要压缩标记的扩展。）

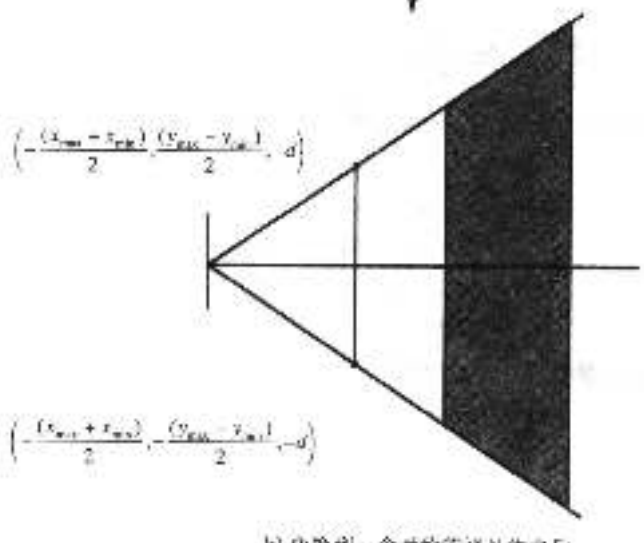
这时T_{pers1}被分解为两个分量。我们有：

$$T_{pers} = T_{pers2}T_{pers1b}T_{pers1a}$$

其中，T_{pers1b}和T_{pers2}与以前的观察系统中的T_{pers1}和T_{pers2}具有相同的作用。这些值可通过对T_{pers1}和T_{pers2}进行修正，使其包含观察平面窗口参数，并且将近平面与观察平面进行分离来获得。



a) 以PRP为原点之后的情况



b) 变换到一个对称的视见体之后

图 5-21

164
2
165

首先，我们需要视见体的中心线与z轴重叠。这就是说要以正比于z的量来调节x和y的值。

很容易看出:

$$\boldsymbol{T}_{\text{pers1a}} = \begin{bmatrix} 1 & 0 & \dfrac{x_{\max}+x_{\min}}{2d} & 0 \\ 0 & 1 & \dfrac{y_{\max}+y_{\min}}{2d} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

例如，上部和下部的观察平面窗口的边按下式变换:

$$(0, y_{\max}, -d, 1) \text{ 变换为 } \left(-\frac{x_{\max}+x_{\min}}{2}, \frac{y_{\max}-y_{\min}}{2}, -d, 1\right)$$

和

$$(0, y_{\min}, -d, 1) \text{ 变换为 } \left(-\frac{x_{\max}+x_{\min}}{2}, -\frac{y_{\max}-y_{\min}}{2}, -d, 1\right)$$

将原来的视见体变换为一个对称的视见体（见图5-21b)。

第二个变换是:

$$\boldsymbol{T}_{\text{pers1b}} = \begin{bmatrix} \dfrac{2d}{x_{\max}-x_{\min}} & 0 & 0 & 0 \\ 0 & \dfrac{2d}{y_{\max}-y_{\min}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

这一计算与简单的观察系统中将对称的视见体变换为45°的视见体时的$\boldsymbol{T}_{\text{pers1}}$具有相同的作用。例如，考虑在穿过观察平面窗口中心的线上$\boldsymbol{T}_{\text{pers1}} = \boldsymbol{T}_{\text{pers1b}}\boldsymbol{T}_{\text{pers1a}}$的作用。按这一等式对点进行变换:

$$\left(\frac{x_{\max}+x_{\min}}{2}, \frac{y_{\max}+y_{\min}}{2}, -d, 1\right) \text{ 变换到 } (0, 0, -d, 1)$$

这个变换使得从原点到该点的线段与$-z$轴共线，这正是我们希望的结果。

最后，我们有

$$\boldsymbol{T}_{\text{pers2}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & f/(f-n) & -fn/(f-n) \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

166 该式将常规的金字塔形状变换成盒子的形状。

第6章　绘图流程（2）：绘制或算法过程

6.1 在视见体上裁剪多边形
6.2 对像素明暗处理
6.3 插值明暗处理技术
6.4 光栅化
6.5 绘制的顺序
6.6 隐藏面消除
6.7 多路绘制和累加缓冲器

引言

在这一章中我们将讨论绘制多边形网格物体所需的算术操作。我们将讨论一个特殊但通用的方法，这个方法采用一种称为Z缓冲器算法的隐藏面算法，并且利用某些插值明暗处理的形式。这种策略的优点是它使我们能够从物体的数据库中以任意的顺序获得多边形。它还意味着这个方法从多边形计数的角度来考虑，对于场景的复杂度的要求没有绝对的上限。这个方法的持续成功不仅是由于这些因素，而且还由于插值明暗处理技术可视化上的成功，该技术使得物体的分段线性性质几乎是不可见的。这个方法的缺点是其天生的效率不高，而这个因素并不是很重要。可以将多边形绘制到屏幕上，然后在接下来的绘制中再用更靠近观察者的多边形对其进行覆盖。

在这个绘制程序中，我们需要执行的过程是光栅化或求出多边形投影到的像素集、隐藏面消除以及明暗处理。完成这些工作之后，我们再根据视见体进行裁剪。关于裁剪我们在上一章作为一个纯的几何操作进行了简单的讨论。在这一章中，我们将建立一个包含了几何操作的算法结构。

正如我们在上一章中所指出的，这些过程几乎都是在三维屏幕空间中进行的，而算法中最内层的循环均是“对每一个”像素的结构。换句话说，这些算法均称为屏幕空间算法。但是，肯定地说情况并不总是如此。用光线跟踪方法进行绘制几乎就是一个观察空间的算法，而用辐射度方法进行绘制就是一个世界空间的算法。

6.1 在视见体上裁剪多边形

在上一章中我们已经讨论了裁剪的原理，现在我们来讨论用于这一操作的一个有效的结构。在前面的分析中，我们看到了如何确定一个点是否处于视见体之内。这是一种效率不高的算法。我们需要一种快速的算法来估计一个物体是完全处于视见体的外部还是完全在其内部，或者是跨在视见体上。随着多边形计数的增长以及对于实时绘制的需求的增加，裁剪成为一种非常重要的操作。从原理上来讲，我们希望在绘图流程的早期阶段尽可能多地去掉一些多边形。用于防止详尽的底层测试的两个通常的方法是场景管理技术和限定体（限定体本身可以被看作是一种场景管理的形式）。我们将考察如何使用一个简单的限定体。

可以执行一种简单的测试，以抛弃完全处于视见体之外的物体，而接受完全处于视见体之内的物体。这一任务可以通过对物体计算其限定球，并将这个限定球与视见体进行测试来完成。物体的限定球的半径是一个固定的值，可以预先计算出这个固定值，并作为数据库的组成部分将其存储起来。当用三维观察流水线来处理物体时，物体的局部坐标系原点也由流水线进行变换（假设限定球以原点为圆心）。

如果物体完全处于视见体之外，则可以将其删除；如果它完全处于视见体之内，则不必对其进行裁剪。如果不属于这两种情况，则可能需要对其进行裁剪。但是，我们还不能肯定，这是因为尽管限定球也许会与视见体相交，但是它所包围的物体却不一定与视见体相交。这个与限定物体有关的问题在整个计算机图形学应用中都有影响，在第12章中将对此进行进一步的讨论。

对于球，可以很容易地看出其限定条件（见图6-1）应是：

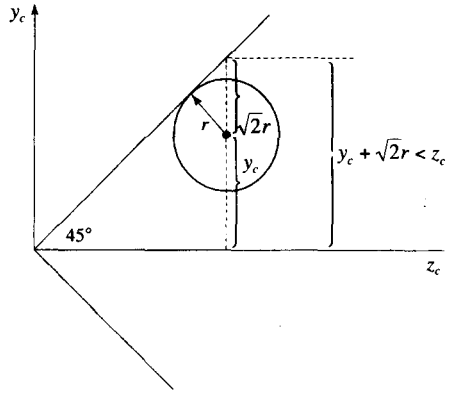


图6-1 对于完全处于视见体之内的限定球的一个限定条件

• 如果满足下列所有条件，则完全在视见体内：

$$z_c > x_c + \sqrt{2r}$$
$$z_c > -x_c + \sqrt{2r}$$
$$z_c > y_c + \sqrt{2r}$$
$$z_c > -y_c + \sqrt{2r}$$
$$z_c > r + n$$
$$z_c > -r + f$$

• 如果满足下列条件之一，则完全在视见体之外：

$$z_c > x_c - \sqrt{2r}$$
$$z_c > -x_c - \sqrt{2r}$$
$$z_c > y_c - \sqrt{2r}$$
$$z_c > -y_c - \sqrt{2r}$$
$$z_c > -r + n$$
$$z_c > r + f$$

其中：

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} d/h & 0 & 0 & 0 \\ 0 & d/h & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix}$$

换句话说，这一操作是在裁剪空间进行的，如图5-13所示。

需要被裁剪的物体可以用Sutherland-Hodgman的凹多边形裁剪程序（Sutherland and Hodgman 1974）来处理。可以将这种处理很方便地扩展到三维空间。多边形与裁剪边界的关系可以通过将多边形的每一条边与一个无限的裁剪边界进行测试来完成。这一工作流程的结构如图6-2所示。

现在考虑算法最内部的循环，这时将一个边与一个裁剪边界进行测试。在这一步中，处

理过程会输出零个、一个或两个顶点，将这些点加入到定义被裁剪的多边形的顶点列表中。图6-3显示了四种可能的情况，边由顶点S和F定义。在第一种情况下，边位于裁剪边界之内，将现在的顶点F添加到输出列表中。在第二种情况下，边跨过裁剪边界，这时计算一个新的顶点I，并输出。第三种情况为一条边完全位于裁剪边界之外，这时不产生输出（在前一次迭代中计算与从内向外伸出的边的相交，在下一次迭代中计算与从外向内伸进来的边的相交）。最后一种情况又会产生一个新的顶点，该顶点被加入到输出列表中。

168
169

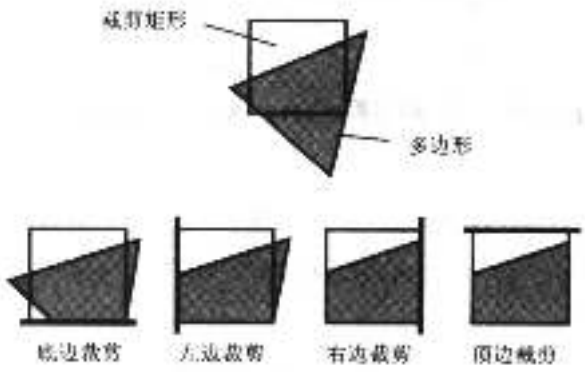


图6-2 用Sutherland-Hodgman裁剪程序对每一个裁剪矩形裁剪每一条多边形的边

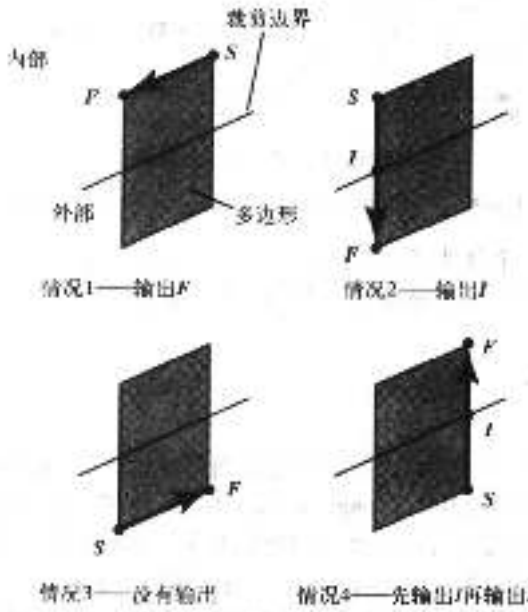


图6-3 Sutherland-Hodgman裁剪程序——在多边形的循环中，对一个多边形的每一条边都与每一个裁剪边界进行比较测试

为了计算一个点或顶点是位于裁剪边界之内，之外还是位于裁剪边界线上，我们采用点积测试。图6-4所示为一条裁剪边界C（其外向法线为N_c）和一条端点为S和F的线段。用参数化方法将线段表示为：

$$P(t) = S + (F - S)t \tag{6-1}$$

其中:

$$0 \leqslant t \leqslant 1$$

在裁剪边界上任意定义一点X，并定义一条从X指向线上任一点的向量。利用该向量与法线的点积可以区分线上的一点是在裁剪边界的内部、外部还是在其上。在图6-4中所示的情况下:

$$N_c \cdot (S - X) > 0 \Rightarrow S \text{位于裁剪区域之外}$$
$$N_c \cdot (F - X) < 0 \Rightarrow F \text{位于裁剪区域之内}$$

并且:

$$N_c \cdot (P(t) - X) = 0$$

定义线段与裁剪边界相交的交点。解方程（6-1）求t，可计算出相交的顶点，并将其存储到输出列表中。

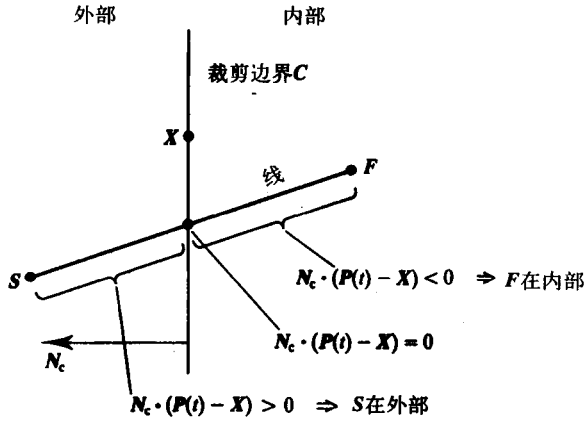


图6-4 点积测试，确定一条线是位于裁剪边界之内还是之外

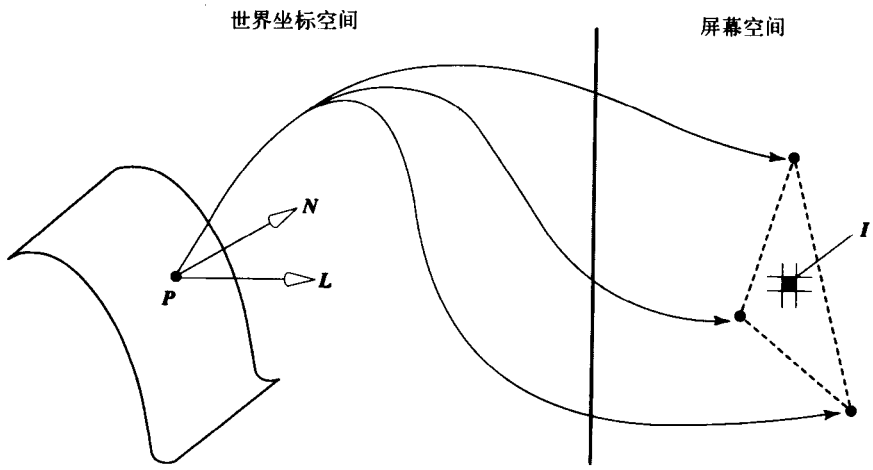
在实际应用程序中，这个算法是一个递归过程。一旦输出了一个顶点，则过程会用该顶点来调用自身，并且对于这个部分被裁剪的多边形不需要有中间存储。这种结构使得该算法非常适合硬件实现。

6.2 对像素明暗处理

170 ~ 171

在计算机图形学中，第一个高质量的明暗处理图像是由H. Gouraud在1971年（Gouraud 1971）建立的。1975年，Phong Bui-Tuong（Phong 1975）对Gouraud的模型进行了改进。众所周知，Phong明暗处理模型变成了主流的三维图形学中的事实标准。尽管后来“全局”技术（如光线跟踪方法和辐射度方法）发展起来，但是Phong明暗处理技术一直保持着强劲的势头。这是因为用这种方法可以用合理的成本将现实（物体）模拟到一个可接受的水平。

有两个关于对多边形投影到的像素点进行明暗处理的考虑。第一个考虑是，如何计算物体表面上任意位置的反射光。我们已经知道一个能够完成这一工作的理论框架，可以用它来计算多边形投影到的像素点的光强度。对于第一个考虑，我们称其为“局部反射模型”，而将第二个考虑称为“明暗处理算法”。两者在概念上的差别见图6-5。例如，最容易的明暗处理方法之一（Gouraud明暗处理方法）对每一个顶点应用局部反射模型来计算顶点的光强度。然后，用前面章节中介绍的计算深度值的插值方程导出每一个像素点的光强度。



a) 局部反射模型用于计算物体表面上任意点处的光强度

b) 明暗处理算法用于从多边形顶点处计算出的光强度插值得到像素值

图6-5 局部反射模型和明暗处理算法之间的差别示意图

从原理上来讲，这里有一个矛盾。我们只是希望计算对每一个多边形投影到的像素的明暗处理。但是，多边形表面的每一个像素点上的反射光的强度都是用世界空间的计算来定义的。我们的计算是基于表面相对于光源的方向进行的，而两者都是在世界空间定义的。于是，我们用多边形的二维投影作为插值算法的基础，而这个算法控制对强度的世界空间的计算，但这是不正确的。用等间距进行线性插值，这在屏幕空间中并不能与世界空间中的反射相对应，在世界空间中反射光的强度应该在多边形的表面上发生变化。出现这种情况的原因之一是我们在进入到屏幕空间时已经进行了一次（非仿射）透视变换。像三维计算机图形学中的许多算法一样，透视变换产生了一个可以接受的可视化结果，甚至用不正确的数学计算也可以产生这种结果。但是，这种算法在某些情况下并不能导出可见的作品。在第18章的比较研究中，给出了一个由这种错误产生的作品。 172

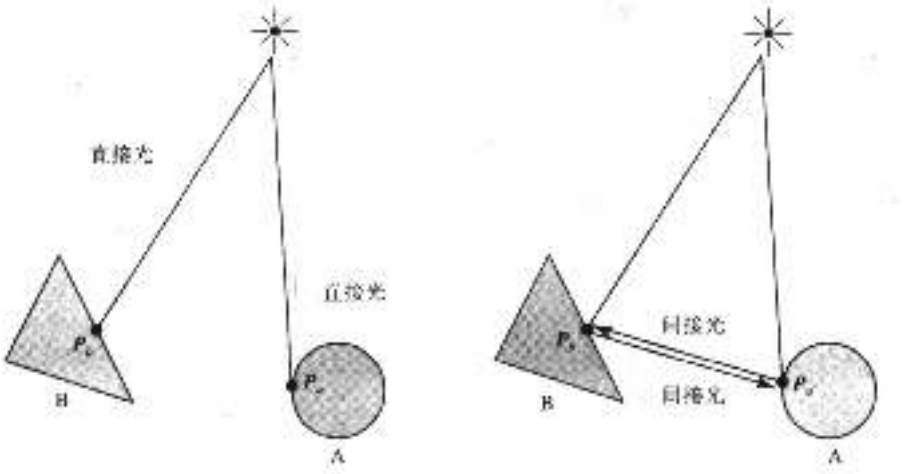
6.2.1 局部反射模型

局部反射模型可以用于计算物体表面上的一点的反射光的强度。第7章将讨论各种各样的局部反射模型，这里，我们只是从实际的观点讨论最通用的模型，并考察如何使其与绘制程序相匹配。

这个模型在1975年被提出，它将反射光的强度作为一个函数来考虑，即所讨论的点处表面相对于点光源的位置的方向以及表面性质的函数。之所以称这样的模型为局部反射模型，是因为它只考虑直接照明。就好像所考虑的物体是漂浮在自由空间中的一个孤立的物体一样。局部反射模型没有考虑可以导致阴影的与其他物体的相互作用和物体内部的反射。图6-6中强调了这一点。在第10章中我们将详细地讨论全局照明。

模型所模拟的物理反射现象为:

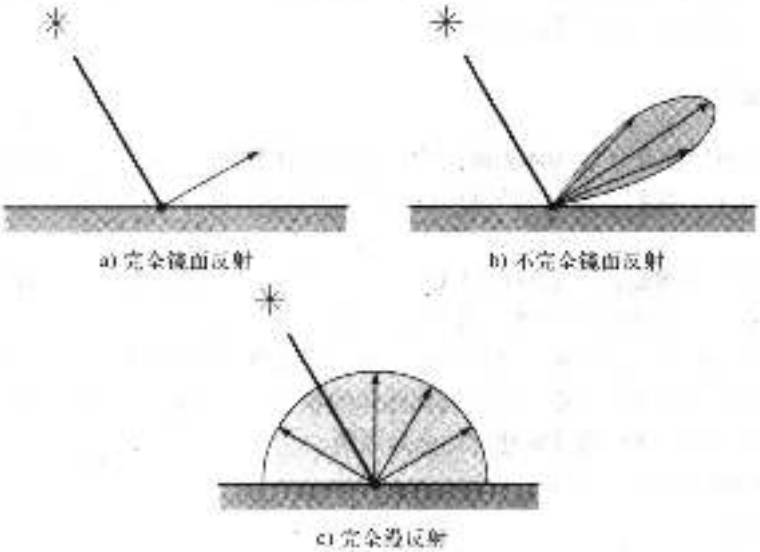
- 完全镜面反射。
- 不完全镜面反射。
- 完全漫反射。



a) 局部反射模型用于计算P_1和P_2处的光强度，只考虑直接照明　　b) 任何从A到B或从B到A的间接反射光都不考虑

图 6-6

对于点光源发送一束无限细的光束到表面上的一个点上，上述反射现象如图6-7所示。当入射光线在“镜面”方向被反射而没有散射时出现完全镜面反射。当一束很细的光束射到一个不光滑的镜面上时出现不完全镜面反射，所谓不光滑镜面，是指其表面的反射性质只有在微观水平才是一个完全的镜面。因此，这种表面从物理上来讲是粗糙的。不光滑镜面上的任
173 一区域单元都可以看成是由成百上千个小的完全镜面组成的，而其中的每个小镜面都有不同的方向。完全镜面反射在实际中并不会出现，我们在光线跟踪模型（见第12章）中用到它只是因为要计算相交，因为计算不完全镜面反射代价太大。一个完全的漫反射表面朝所有的方向反射光线，这样的表面通常称为铜镜（matte）。



a) 完全镜面反射　　b) 不完全镜面反射

c) 完全漫反射

图6-7 计算机图形学中的三种反射现象

Phong反射模型认为来自一个表面的反射由下面三个线性叠加的分量组成：

反射光 = 环境光 + 漫反射光 + 镜面反射光

其中，环境光一项是一个常数，并且是全局照明或间接照明的一种模拟。这一项是必需的，因为一个表面的各个部分不可能“看到”光源，但是观察者可以看到光源，而且需要光源来照明。否则的话，物体将被绘制成黑色的。在现实中这样的光照来自全局照明或间接照明。只要简单地增加一个常量就绕过了对间接照明或全局照明的计算。

考虑一下这种模型模拟的是什么类型的表面是有用的。漫反射光和镜面反射光的一种线性叠加出现在抛光表面的情况，比如漆面的木材，镜面反射由透明层产生，而漫反射来自其下面的表面（见图6-8）。可以用这个模型模拟很多物理类型，尽管并不是在物理上与漆面的木材相同。这一现象的真实性可以通过观察真实的漆面木材、闪光的塑料和有光涂料的样本得到演示。如果把所有上下文的线索都去掉，并且从每一个样本上反射出的光都表现出相同的镜面扩散，观察者可能会发现很难对样本进行区分。 174

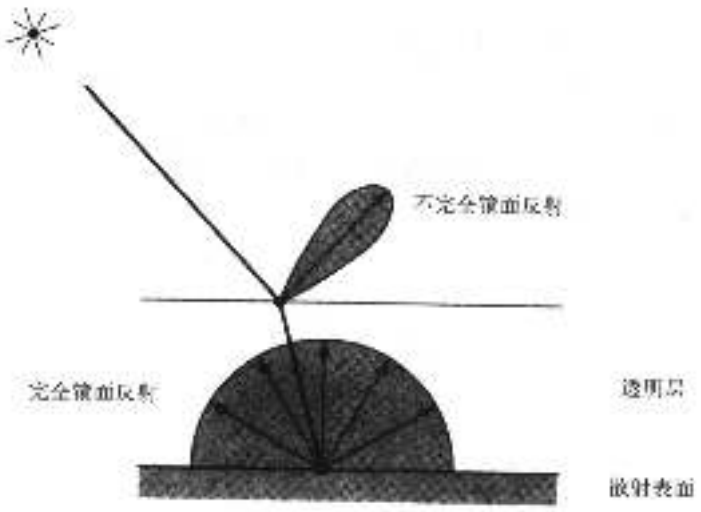


图6-8 “计算机图形学”表面

除了作为局部模型所具有的局限性之外，Phong反射模型还是完全经验的或模拟的模型。其主要的缺点之一是经模型所计算出来的反射光强度的值只是一个观察方向以及相对于光源来说表面的方向的函数。在实际中，反射光强度表现出双向性，它还依赖于入射光的方向。这一缺陷导致了对基于物理原理的反射模型的大量研究。人们企图通过模拟真实表面性质来建模反射光。但是，使用这类模型所可能产生的细微改进，比如使表面看起来像金属的能力，并没有导致Phong反射模型的消失，也没有导致人们相信当前的有关克服“局部”限制的绘制方法的研究为主。像辐射度这样的全局方法导致在对场景的外观真实性的显示上有了非常显著的进步。

我们先抛开关于颜色这个话题不谈，表面的物理性质是通过控制散射和镜面反射之间的比例来模拟的。我们有反射光：

$$I = k_a I_a + k_d I_d + k_s I_s$$

其中，三个分量（环境、漫反射和镜面反射）的比例由三个常数控制，其中：

$$k_a + k_d + k_s = 1$$

I_d由下式估算:

175

$$I_d = I_i \cos\theta$$

其中:

I_i是入射光的强度;

θ是所讨论的点处的表面法向和光源方向之间的夹角.

用向量形式标记为:

$$I_d = I_i(\boldsymbol{L}\cdot\boldsymbol{N})$$

其几何解释如图6-9所示.

从物理上来讲,镜面反射由一个光源的图像"涂抹"在表面的一个区域上,产生一种称为高光(highlight)的现象.只有当观察者的观察方向靠近镜面的方向时,观察者才能够看到高光.因此,用下式来模拟镜面反射:

$$I_r = I_i \cos^n\Omega$$

其中:

Ω是观察方向和镜面方向$\boldsymbol{R}$之间的夹角;

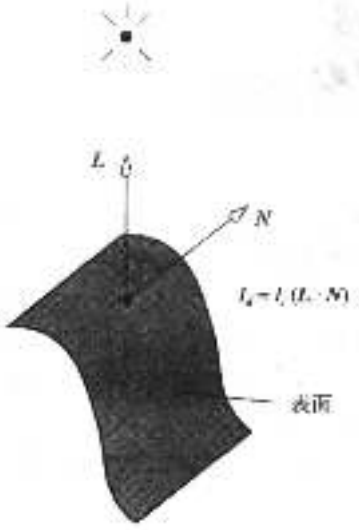
n是模拟表面不完全程度的一个系数.

当$n=\infty$时,表面是一个完全的镜面.也就是说,所有反射光都归入到镜面的方向.对于n的其他值,即用来模拟一种不完全的镜面反射器(见图6-7b).这个作用的几何解释如图6-10所示.当用向量表示时,我们有:

$$I_r = I_i(\boldsymbol{R}\cdot\boldsymbol{V})^n$$

将这些项合并到一起,有:

$$I = k_a I_a + I_i(k_d(\boldsymbol{L}\cdot\boldsymbol{N}) + k_r(\boldsymbol{R}\cdot\boldsymbol{V})^n)$$



176 图6-9 Phong模型的漫反射分量

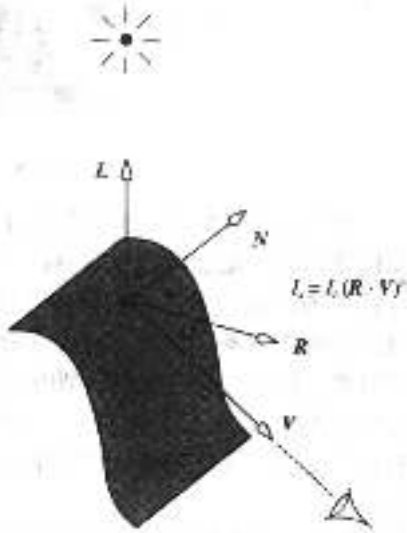


图6-10 Phong模型的镜面反射分量

这个方程的行为如图6-11和图6-12所示(彩色插图).图6-11显示了P点上的光强度,该值是观察向量$\boldsymbol{V}$的方向的函数.半圆是常数环境项与漫反射项的和,对于一个特定的$\boldsymbol{N}$值,它是

一个常数。加上镜面反射项之后得到图中所示的轮廓线。随着n的值增加，镜面部分的波束变窄。图6-12（彩色插图）显示的是应用于同样物体的方程，但采用不同的K_s、K_d和n值。

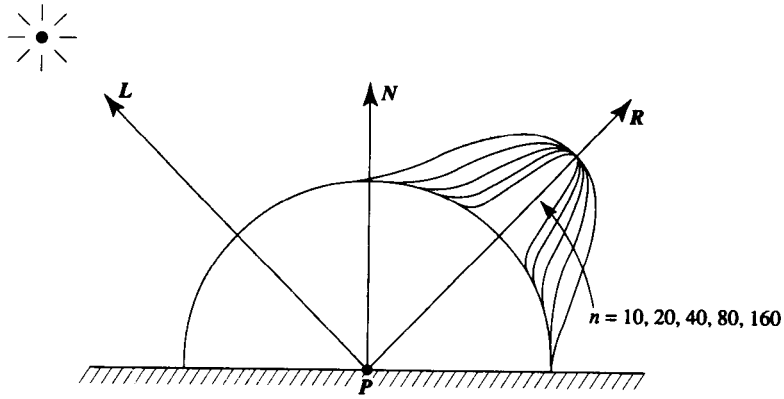


图6-11 点P处的光强度是观察方向V的函数

6.2.2 局部反射模型——实际问题

现在解释一些关于颜色以及几何学上的简化等实际问题。 177

上面给出的明暗处理方程执行时付出的代价为，对每一个像素都要应用几次该方程。可以通过进行减少计算时间的几何简化来降低这种代价，但前提是这种简化不会影响明暗处理的效果。首先，如果将光源看作是点光源，并位于无限远处，则L在场景域内即可被认为是常数。其次，还可以把观察点置于无限远处，使V为常数。当然，对于观察变换和透视变换来讲，观察点必须固定在世界空间中，所以我们在进行几何变换时采用一个有限远的观察点，而对于明暗处理方程使用一个无限远的观察点。

接下来，R的计算也很繁琐，定义一个向量H(中间) 可使事情变得容易些。H是一个假设表面的单位法向，该表面的方向位于光线方向向量L和观察向量V的中分位置处（见图6-13）。很容易看出:

$$H = (L + V)/2$$

如果一个表面要在V方向上反射光最大，则这个表面的方向需要朝向H。这时明暗处理方程变为:

$$I = I_a k_a + I_i(k_d(L \cdot N) + (N \cdot H)^n)$$

这是由于$(N \cdot H)$的变化方式与$(R \cdot V)$的变化方式相同。这一简化意味着I这时只是N的一个函数。

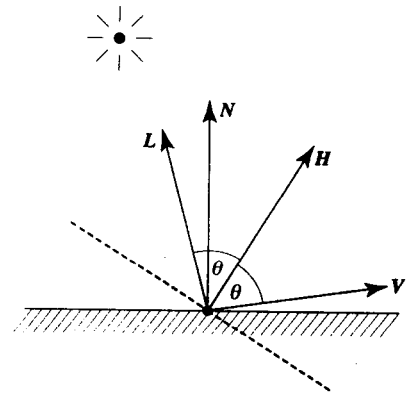


图6-13 H是朝向在V方向反射所有光的表面方向的法向

对于那些彩色的物体，我们产生三个强度分量I_r、I_g和I_b，并通过适当地设定漫反射系数k_r、k_b和k_g来控制物体的颜色。实际上，镜面高光正是物体表面上光源的反射，我们通过设定k_s的比例来匹配光的颜色。对于白色光，在所有的三个方程中k_s都是相等的。于是我们有:

$$I_r = I_a k_{ar} + I_l((k_d(L\cdot N)+k_s(N\cdot H)^n)$$
$$I_g = I_a k_{ag} + I_l((k_d(L\cdot N)+k_s(N\cdot H)^n)$$
$$I_b = I_a k_{ab} + I_l((k_d(L\cdot N)+k_s(N\cdot H)^n)$$

178 在计算机图形学中对颜色的更细致的处理在第15章中给出。

6.2.3 局部反射模型——关于光源的考虑

上面的模型中，最有限制性的近似之一就是设光源位于无限远处的一个点上。我们还可以从图6-12（彩色插图）中看出，在对参数n的解释上有一种令人不满意的混淆，并给人一种"咬文嚼字"的感觉。在实际中，这样做看起来好像在改变光源的尺寸。

对一种简单的带方向性的（非点的）光源很容易进行建模，Warn（1983）给出了下面的建议。在他提出的方法中，以和镜面反射表面相同的方法建模了带方向的光源。从光源射出的光以余弦函数的一个幂的形式给出。这里，我们假设对于带方向的光源，其在特定方向上的光强度由角度ϕ给出：

$$I_l\cos^n\phi$$

这时，ϕ是$-L$与L_l之间的夹角。而$-L$为所讨论的表面上的点的方向，L_l为光源的方向（见图6-14），在明暗处理方程中，我们采用的I_l的值由下式给出：

$$I_l = I_l(-L\cdot L_l)^n$$

注意，我们不再将向量L看作是在整个场景中为常数。

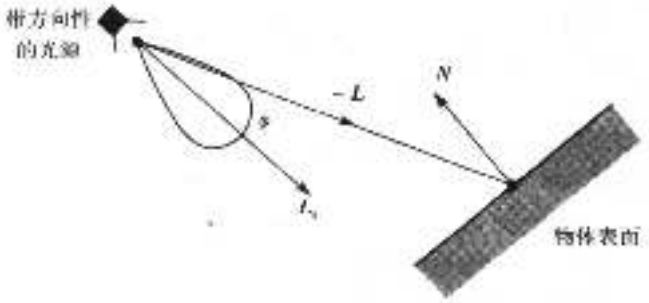


图6-14 表示为镜面反射表面的光源

6.3 插值明暗处理技术

在讨论了计算一个点上的光强度问题之后，我们现在来考虑如何把这样的模型应用到多边形中，计算多边形表面上的光强度。有两个经典的技术——Gouraud明暗处理和Phong明暗处理。这两个技术之间质量上的差别在第18章的比较性实例研究中给出，并进行了讨论。现在分别对它们进行讨论。像我们将要看到的那样，Phong算法给出更精确的高光，一般而言，人们更喜欢用这个模型。另一方面，Gouraud明暗处理方法要经济得多。两种技术都是基于在
179 多边形表面上高效地插值信息数据，并且在最终的图像上消除了多边形边的可见性。信息是由多边形顶点上的值进行插值的，这种计算与深度插值时的计算是相同的。

6.3.1 插值明暗处理技术——Gouraud明暗处理

在Gouraud明暗处理技术中，我们采用上一节中介绍的局部反射模型先计算多边形顶点处

的光强度。然后，在这些光强度之间进行插值，求出投影像素上的值。为了完成这一工作，我们采用1.5节中给出的双线性插值方程，性质p为顶点光强度I。顶点处特定的表面法向是特殊的法向，称为顶点法向。如果将多边形看作是孤立的多边形，则它的顶点法向当然都是平行的。但是，在Gouraud明暗处理算法中，我们采用了称为顶点法向的特殊法向，而正是因为这一点降低了多边形边的可见性。考虑图6-15。在这里，顶点法向N_A是N_1，N_2，N_3，N_4的平均值。

$$N_A = N_1 + N_2 + N_3 + N_4$$

然后，将N_A用于计算顶点A处的光强度。对于共享顶点A的所有多边形，它是一个公共值。

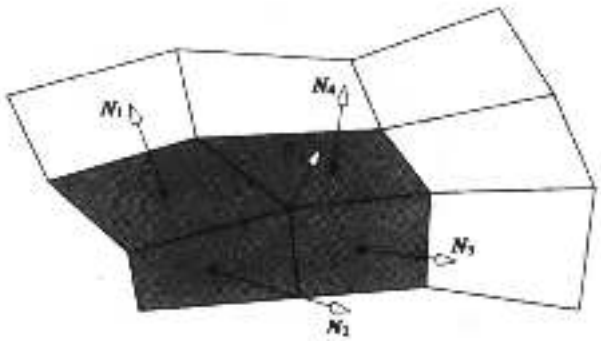


图6-15 顶点法向N_A是N_1，N_2，N_3，N_4的均值，是在顶点处相遇的多边形的法向

从计算效率的角度来考虑，插值方程按照增量计算的方法来执行。这种处理在第二个方程的计算中尤为重要，因为这个方程对于每一个像素都要被用到。如果将Δx定义为扫描线上距离的增量，则从一个像素到另一个像素光强度的改变ΔI为：

$$\Delta I_s = \frac{\Delta x}{x_b - x_a}(I_b - I_a)$$

$$I_{s,x} = I_{s,x-1} + \Delta I_s$$

由于只在顶点处计算光强度，所以这个方法不能很好地处理高光问题，而这正是该方法的主要缺点。引起这一问题的原因可以通过考察图6-16a来进行理解。我们必须记住，多边形网格是对曲面的一种近似。对于特定的观察方向和光源方向来说，可以在A和B处有一个漫反射分量，而在A、B之间的某个区域有一个镜面高光。很明显，如果我们由A点和B点的信息导出P点的光强度，那就不会计算A、B之间的高光。这种情况可通过插值顶点法向而不是插值光强度得到间接的处理，如图6-16b所示。这个方法称为Phong明暗处理方法。 180

6.3.2 插值明暗处理技术——Phong明暗处理

在这里，我们在多边形的内部插值顶点法向，并对每一个多边形像素的投影计算其插值的法向。然后将插值的法向用到明暗处理方程中，明暗处理方程被应用于每一个像素的投影中。这种处理的几何效果（见图6-16）是把某些曲率“保存”到了多边形化的表面上了。

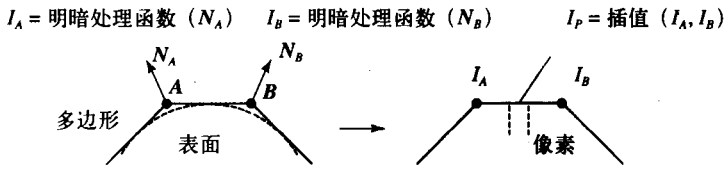
对于模型的这种改进所付出的代价是效率。不仅向量插值是光强度插值计算量的三倍，而且对于每一个像素投影都必须计算其明暗处理方程，每一个向量都必须单位化。

可以像对光强度插值那样采用增量计算，插值应按以下方式进行：

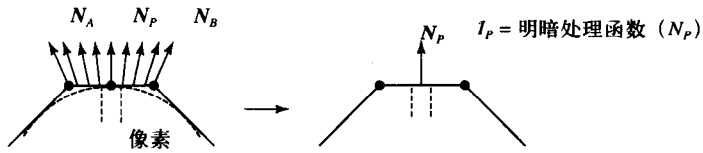
$$N_{sx,n} = N_{sx,n-1} + \Delta N_{sx}$$
$$N_{sy,n} = N_{sy,n-1} + \Delta N_{sy}$$
$$N_{sz,n} = N_{sz,n-1} + \Delta N_{sz}$$

其中，N_{sx}, N_{sy}, N_{sz}为通用扫描线法向量N_s的分量，表示为:

$$\Delta N_{sx} = \frac{\Delta x}{x_b - x_a}(N_{bx} - N_{ax})$$
$$\Delta N_{sy} = \frac{\Delta x}{x_b - x_a}(N_{by} - N_{ay})$$
$$\Delta N_{sz} = \frac{\Delta x}{x_b - x_a}(N_{bz} - N_{az})$$



a) Gouraud明暗处理



b) Phong明暗处理

图6-16 Gouraud明暗处理和Phong明暗处理方法的差别

6.3.3 绘制程序的明暗处理选项

大多数绘制程序都有一个明暗处理选项的层次结构，可以在等待时间和明暗处理后图像的质量之间做一个选择。当然，这一步也适用于添加阴影和纹理。一般的层次结构是:

- **线框** 根本没有绘制和明暗处理。可以将线框显示用于在场景中放置物体，方法是与观察参数进行交互。它还被普遍用于动画系统中，即动画程序员用于交互地在场景中建立物体的移动动画。他可以对动画的各个方面进行调节，并以线框显示模式产生一个实时的动画序列。在这两个应用中，很显然并不需要产生一个完全明暗处理过的图像。一个实际的问题是，像进行明暗处理那样对线框绘制运用相同的整套绘制策略（即单独地绘制每一个多边形）会导致对每一条边绘制两次，因此使得对物体的绘制时间加倍。
- **平面明暗处理多边形** 再次选择快速（明暗处理），采用单个的“真实”多边形法向，用Gouraud方程对每一个多边形进行一次明暗处理计算，省去明暗处理插值过程。
- **Gouraud明暗处理** 基本明暗处理选项，它在多边形的面上产生一些变化。由于不能很好地处理镜面高光问题，通常选择用Gouraud明暗处理选项计算漫反射。
- **Phong明暗处理** “标准”质量的明暗处理方法。由于向量插值和在每一个像素点上的明暗处理方程计算，所以该方法比Gouraud明暗处理方法要慢4到5倍的时间。
- **Phong明暗处理和Gouraud明暗处理的混合使用** 考虑一个漫反射物体。尽管对这个物

体运用Gouraud明暗处理与用Phong明暗处理方法产生稍微不同的效果，但是，如果将镜面反射系数设为零，则这种差别在视觉上并不是很重要。这就是说，在对一个包含有镜面和漫反射物体的场景进行处理时，对于漫反射物体使用Gouraud明暗处理方法，而对镜面反射物体只用Phong明暗处理方法。这时，Gouraud-Phong选项就成为物体性质数据的组成部分。

这些选项在第18章的比较实例研究中以较详细的方式进行了比较分析。

181
182

6.3.4 Gouraud明暗处理和Phong明暗处理的比较

Gouraud明暗处理对于以漫反射形式反射光线的表面是高效的。也可以用Gouraud明暗处理方法进模镜面反射，但是，所产生的镜面高光的形状依赖于其所在多边形之间的相对位置。Gouraud明暗处理方法的优点是，在这两个模型中，从计算角度来讲它是比较好的模型。它只需要计算多边形顶点处的光强度，然后，对于每一个像素计算这些值的双线性插值。

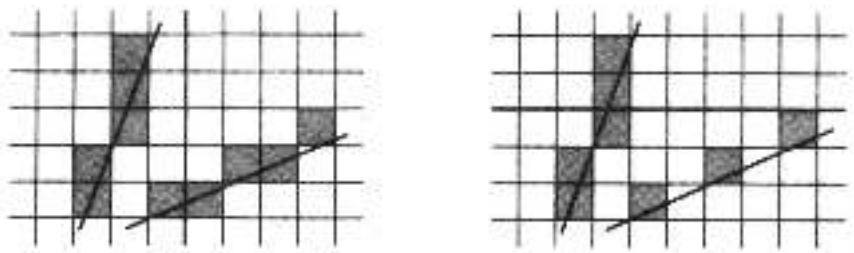
Phong明暗处理方法产生高光，而且高光很少依赖于其所处的多边形，但是，需要进行更多的计算，包括表面法向插值和对每一个像素点计算光强度函数值。这些因素使人们很容易想到通过将Gouraud和Phong方法结合起来加速Phong明暗处理方法。

6.4 光栅化

在讨论了如何在多边形中的一般点上分配光强度值（这些值是通过顶点的值来确定的）之后，现在让我们来了解如何确定实际的像素，并为其找到所需的光强度值。这个过程称为光栅化（rasterization），或者扫描转换（scan conversion）。我们把这个有些难对付的问题分成两部分。首先，应该如何确定那些跨在多边形边上的像素？其次，应该如何安排这些信息以确定内部点？

6.4.1 光栅化边

对边的光栅化有两种不同的方法，这取决于是采用划线的方式还是采用填充小块区域的方式。本书中不讨论划线的方式，这是因为我们对实体填充的物体感兴趣。但是，划线算法（例如，Bresenham算法（Bresenham 1965））的主要性质是必须产生一个没有缝隙的像素的线性序列（见图6-17）。



a) 划线方法所需的像素序列　　b) 多边形填充方法所需的像素序列

图 6-17

183

对于实体区域填充，只要有不太严密的方法就足够了。可以用水平的线段来填充多边形，

可以把这种填充看作是多边形与特定的扫描线相交而成的。于是，对于任意已知的扫描线，所需要的是线段的左侧和右侧的边界，也就是扫描线与多边形左侧和右侧的两个边的交点。这就意味着对于每个边，需要产生相应于边与扫描线相交的一个像素序列（见图6-17b）。当将这个序列解释为一条直线时，它可能会有缝隙，如图中右侧的边所示。

计算这些像素坐标的传统方法是采用一种称为“数字差分分析器”（digital differential analyzer）的方法，或简称DDA方法。这个算法中所包含的实际意义是求出每条扫描线的x坐标的增量值，然后再重复地对这个增量求加。

设（x_s, y_s）和（x_e, y_e）为边的起点和终点（假设$y_e > y_s$）。对于多边形边进行光栅化所用的最简单的算法是：

```
x:= x_s
m:= (x_e − x_s)/(y_e − y_s)
for y:= y_s to y_e do
  output(round(x), y)
  x:= x + m
```

这一方法的主要缺点是m和x的值都必须以浮点数表示，在每一次循环中都必须进行浮点加法和实数到整数的转换。Swanson和Thayer(Swanson and Thayer 1986)提出了这一算法的整数版。可以将上面的算法分成两个逻辑阶段。首先，将x和m分成整数和分数部分。然后，在程序的每一次循环中，分别对这两个部分求加，当分数部分溢出时再向整数部分加1。另外，将x的分数部分的初始值设为−0.5，这是为了便于取整计算，并且可以简化溢出的条件限制。其伪码如下：

```
xi:= x_s
xf:= −0.5
mi:= (x_e−x_s)div(y_e − y_s)
mf:= (x_e−x_s)/(y_e − y_s)−mi
for y:=y_s to y_e do
    output(xi, y)
    xi:= xi + mi
    xf: = xf + mf
    if xf >0.0 then {xi:= xi + 1;xf:= xf−1.0}
```

由于这时分数部分独立于整数部分，所以就可以在整个区间对其除以2（$y_e - y_s$），这样处
184 理的效果是将每一个计算都转换成了整数计算。

```
xi:= x_s
xf:= −(y_e − y_s)
mi:= (x_e − x_s)div(y_e − y_s)
mf:= 2*[(x_e − x_s)mod(y_e − y_s)]
for y:= y_s to y_e do
    output(xi,y)
    xi:=xi + mi
    xf:= xf + mf
    if xf >0 then {xi := xi + 1; xf:= xf−2(y_e − y_s)}
```

尽管这看起来好像进行了两次除法，而不是一次，但这两次除法都是整型的而不是浮点除法。而且，在给出适当的硬件的前提下，可以用相同的除法来计算这两者，这是由于第二个除法（mod）是第一个除法（div）产生的余数。最后，唯一需要指出的是，在循环中2

(y_2-y_1)是一个常数，所以在实际程序中应放在循环之外，并只进行一次计算。

6.4.2 光栅化多边形

现在，我们已经了解了如何沿着多边形的边求出像素点，应该将注意力转移到填充多边形上来。由于我们考虑的是明暗处理，所以“填充多边形”就意味着求出内部点的像素坐标值，并用6.3节中所给出的增量明暗处理策略之一为这些像素点赋计算值。我们需要产生片段端点对，然后水平地在端点之间进行填充。这项工作通常通过对每一个多边形建立一个“边列表”来完成。

从原则上来讲，可以用一个链表的数组来进行，这时每一条扫描线为一个元素。开始时，将数组中的所有元素都设为NIL。然后，依次对多边形的每一条边进行光栅化。这时，所产生的每一个像素(x, y)的x坐标就被插入到相应于y值的链表中。然后，将每一个链表按x的升序进行存储。结果如图6-18所示。于是，多边形的填充就通过对每一条扫描线，依次取x的值对，并在两者之间进行填充（因为多边形必须是封闭的，所以在链表中也总是会有偶数个元素）而实现。请注意，这个方法的功能强大，足可以处理带有孔的凹多边形。

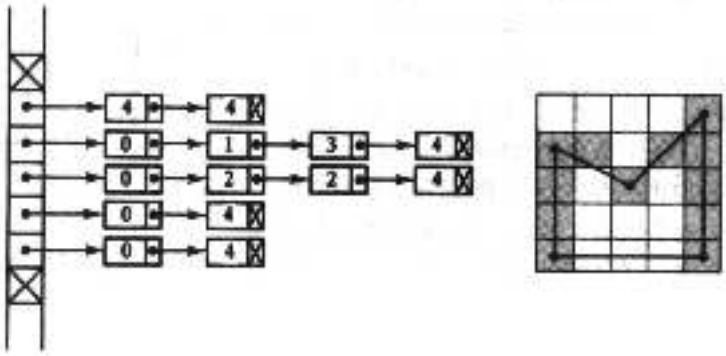


图6-18 多边形光栅化中的一个链表的例子

在实际应用中，链表的排序可以通过只是在适当的初始位置处插入值来完成，而不是最后进行大排序，而且，在对每一个边上的像素计算x值并保存时，对相应的明暗处理值需要同时进行计算并保存起来（例如，Gourmand明暗处理的光强度值；Phong明暗处理法所插值出来的法向量的x、y、z分量）。

如果物体只有凸状的多边形，则x链表仅含有两个x坐标，这就简化了边列表的数据结构，也不需要进行排序。在实际的计算机图形学应用中，将物体限定为凸多边形并不是一个很大的限制。

到目前为止，有一件事一直被稍微曲解了，即对多边形的边界到底严格地处于何处的考虑。其实，这一点可以通过该多边形与相邻的多边形的边之间的连接方式来判断，比如两者之间出现缝隙或者两者重叠起来。例如，在图6-19中，多边形的宽度是3个单位，所以它的面积应为9个单位，而用16个单位的面积对它进行了绘制。对于这个问题的传统的解决方案，而且教科书中通常提倡的方法是，把像素的样本点看作是位于它的中心，即在$(x + 0.5, y + 0.5)$处

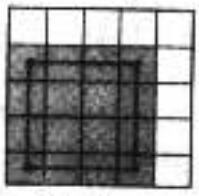


图6-19 多边形边界的问题。9个像素的多边形填充16个像素的空间

（可以将一个像素看作是一个矩形的有限区域，其大小为1.0 × 1.0，其样本点是像素区域中的点，为了确定像素的值在场景的这个位置取样）。因此，例如，在计算一条边与一条扫描线相交时是取$y + 0.5$进行计算，而不是像前面假设的y。这是混乱的，因此也就排除了使用整数算术的可能性。一个较简单的解决方法是假设样本点位于像素的四个角点之一的位置，我们选择像素的右上角作为这一位置。其结果是，整个图像被放在了向左下方移动半个像素的位置，而这一结果在实际使用中并不明显。这样做的结果是它提供了下面的简单的光栅化规则：

1) 去掉水平的边。

2) 从扫描线$y_{底}$到$y_{顶}$的一个边应该对扫描线从$y_{底}$处一直到$y_{顶}-1$处产生一些x值（也就是说，没有顶部的扫描线）。如果$y_{底} = y_{顶}$则不产生x值。

185 ~ 186

3) 同样地，水平的线段应该从$x_{左}$到$x_{右}-1$被填充（如果$x_{左} = x_{右}$，则不产生像素）。

顺便说明一下，在规则2和规则3中，第一个元素和最后一个元素可以任意忽略，可基于编程的习惯进行选择。这两条规则的四种可能的变换是将样本点定义为像素的四个角点之一。这些规则的作用示于图6-20中。在这个图中，有三个相邻的多边形A、B和C，有四条边a，b，c，d。对于所示扫描线由这些边产生的取整的x值分别为2，4，4，7。接着，由规则3给出了对多边形A的像素点2和3，多边形B没有值，而多边形C为4～6。这样，从整体来看没有缝隙，也没有重叠。将水平的边去掉的原因是与这些边相邻的边已经对x值作出了贡献，并构成了线段（例如，图6-18中多边形的底边，还需注意的是，为了简化起见，这个多边形的扫描转换并不是严格地按照上述的光栅化规则进行的）。

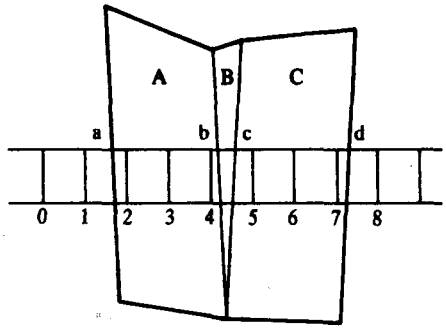


图6-20 三个多边形与一条扫描线相交

6.5 绘制的顺序

为场景的绘制进行排序的基本方法有两个。这两个方法是：按多边形排序，依次绘制每一个多边形，而其余所有的多边形处于隔离状态；按扫描线排序，场景中所有穿过一条已知扫描线的多边形上的片段，在移到下一条扫描线之前均被绘制。在某些教科书中，这种分类方法的表现是，没有办法将其与隐藏面消除算法的分类进行区分。事实上，场景绘制的顺序对可以采用的隐藏面消除算法有严格的限制，但是这个顺序本身并不会依赖于隐藏面消除所选的方法。与这两种排序方法相匹配的通用的隐藏面消除算法为：

- 按多边形：Z缓冲器。
- 按扫描线：Z缓冲器、扫描线Z缓冲器、跨跃的扫描线算法。

187

按多边形进行绘制有一些优点。它执行简单，一次动作所需要的数据少。正因为这一点，它对于场景的复杂度没有上限的要求。这与扫描线绘制方法不同。扫描线绘制算法需要在内存中同时保存所有多边形跨跃一条特定扫描线的光栅化、明暗处理信息，也许还需要保存纹理信息。按多边行进行绘制的主要缺点是它不能利用可能提高效率的因素，例如共享多边形之间的信息（例如，在一个场景中，大多数边都是两个多边形之间的共享边）。这个方法只能用于Z缓冲器隐藏面算法。而我们将会看到，这种算法从存储空间占用的角度来讲代价是相当

高的。而且，基于扫描线的算法所具有的性质是完整图像按扫描线的顺序产生，这种算法对于硬件实现和反走样计算都有优势。

两种绘制方法的一个重要差别是在构建边列表时的不同。这一点已在讨论按多边形进行绘制的方法时进行了阐述。然而，如果按扫描线的顺序进行绘制，则会出现两个问题，一个问题是若先对所有多边形的所有边进行光栅化可能会消耗大量的内存空间，这为场景的最大复杂度设定了一个更低的限制。而且，通常还需要保持一个“活动边”的列表。当开始一条新的扫描线时，从这条扫描线开始的所有边都被加入到列表中。同时，那些在该扫描线上结束的边被删掉。对于活动列表中的每一条边，存储其当前的x值、明暗处理信息等，以及这些值的增量。每次加入一条新边时，均对这些值先初始化，然后对于每一条新的扫描线加一个增量。

另一个问题出现在确定线段时，这是因为这时在一条已知的扫描线上有多个活动的多边形。一般来讲，对于每一条处于活动边列表中的边需要存储一些多余的信息以表明这条边属于哪一个多边形。这个处理的详细过程非常依赖于所使用的隐藏面消除算法。通常，保留一个活动多边形列表，该活动多边形列表指出那些与当前扫描线相交的多边形，因此，可以由这些多边形产生活动边。对于每一条扫描线有一个更新的列表，加入新的多边形，并将“不活动”的多边形删除。

按多边形的绘制程序的框架程序为：

```
for每一个多边形do
    由多边形的边构造一个边列表
    for y: = y_min to y_max do
        for 在边列表EdgeList[y]中每一对（x_i , x_i+1）do
            对从（x_i, y）到（x_i+1, y）的水平片段明暗处理
```

而按扫描线的绘制程序的框架程序为：

```
清除活动边列表
for 每一条扫描线do
    for 从该扫描线开始的每一条边do
        把该边加入到活动边列表中
        对明暗处理、光栅化值和它们的增量赋初值
        去掉在该扫描线结束的边
        分析活动边列表，得到并绘制片段
        把增量加入到所有的活动边
```

188

最后，有必要指出的是，有可能对这两种方法进行融合。通常可以将一个场景分离成一些相互没有联系的物体。如果场景是基于物体绘制的，则在每一个物体内部采用扫描线排序。这时，在每一个物体内信息共享的优点得到实现。这样做对于场景的复杂度没有上限的要求，而只对各个物体的复杂度有限制。

6.6 隐藏面消除

在大多数计算机图形学教科书中都有关于主要的隐藏面消除算法的论述，并将这些算法按照Sutherland等（1974）提出的“十种隐藏面算法”进行分类。在他的文章中，算法按照操作是在物体空间还是在图像（屏幕）空间进行的，以及不同的算法“连贯性”来分类。所谓

连贯性（coherence），是用来描述那些特定算法的一个术语，这些隐藏面消除算法以几何单位（比如说区域或者扫描线片段）而不是按单个的点来进行运算的。

有两个流行的隐藏面消除算法，它们是基于扫描线的系统和基于Z缓冲器的系统。其他隐藏面消除算法，如区域细分（Warnock 1969）、深度列表策略（Newell等1972）都不是很普及，是在像飞行模拟这样的特定应用中使用的。

6.6.1 Z缓冲器算法

Z缓冲器算法由Catmull（1975）提出，这一方法像计算机图形学中的Phong反射模型和插值算法一样普及。将这些表示方法进行结合是最流行的绘制程序的选择。如果用Sutherland提出的分类策略（Sutherland等1974），则这种方法是在图像层次即屏幕空间的运算。

采用一种增量明暗处理的方案，对多边形内部的像素进行明暗处理。在实施了观察变换之后，根据多边形顶点的z值经插值得到它们的深度。用1.5节中给出的方程插值深度值。

Z缓冲器算法等价于对每一个点(x_s, y_s)搜索其相关的每个内部多边形点的z值，以求出具有最小z值的点。为了方便起见，搜索是用Z缓冲器来完成的，即对于当前点(x, y)保持到目前为止遇到的最小z值。在处理多边形时，是否将点(x, y)的光强度写入帧缓冲器要依赖于当前点的深度z是否小于Z缓冲器中所记录的到目前为止的深度值。

189 Z缓冲器算法的主要优点之一是它不依赖于物体的表示形式。尽管在多边形网格表示方法中最常看到这种方法的使用，但实际上它可以用于任意的表示方法。所需要的只是计算物体表面上每一个点的z值。可以将其应用于CSG物体，利用对每一个物体的Z缓冲器中的信息可以将各个分别绘制的物体合并到一个多物体的场景中。

Z缓冲器算法的最重要优点是其实现的简单性。其主要的缺点是Z缓冲器所需的存储器容量。Z缓冲器的大小依赖于存储每个点(x, y)的深度值的精确度，而这个值是场景复杂度的函数。通常认为20到32位字长就足够了，必须将场景规范到这个z值的范围之内，以使得场景内的精确度达到最大。回顾在上一章中，我们讨论了z_s值的压缩问题。这意味着一对完全不同的具有不同z_v值的点可以被映射成具有相同的z_s值。我们注意到，对于每个像素少于24位的帧缓冲器，Z缓冲器事实上将大于帧缓冲器。过去，Z缓冲器倾向于成为主机的主存储器的组成部分。但是，现在有了图形终端，这种终端有专用的Z缓冲器，而这是最好的解决方法。

存储器问题可以通过将Z缓冲器在屏幕空间中划分成条或分区来得到缓解。这样做付出的代价是多遍进行绘制程序的几何部分的计算。从数据库中取出多边形，如果它们的投影落在屏幕空间中Z缓冲器的分区中则对其进行绘制。

Z缓冲器的一个有趣的用途是由Foley等（1989）提出的。这个用途是对所选的物体进行绘制，但这些物体并不对Z缓冲器中的内容进行修改。这个想法可以用于交互，这时，一个三维的光标物体可以在场景中到处移动。光标就是所选择的物体，它在当前的位置上被绘制了，但并没有写入到Z缓冲器中。毫无疑问，Z缓冲器用于执行对物体的隐藏面消除，而物体在场景中移动挡住一些物体，或者被其他的物体遮挡住。

6.6.2 Z缓冲器和CSG表示

可以用Z缓冲器算法来帮助CSG物体的绘制。回忆4.3节中所述，在那一节中描述了一种绘制这类物体的光线跟踪算法。绘制过程包括计算复杂物体的边界表示，复杂物体由一些用

布尔运算符结合起来的基元物体组成，用一个结构树来描述或表示。

光线跟踪方法的问题是费时。常规的递归光线跟踪程序是这样的，即求出一条任意方向上的光线与场景中的物体之间的交。这种模型递归进行，可到任意深度，以此来求镜面相互作用。但是，对于CSG表示的物体来讲，因为所有的光线都是平行的，所以我们只关心第一次相交的情况。因此，从这个侧面来考虑，用光线跟踪是不合适的，而Z缓冲器算法执行起来要容易一些，而且代价较小（Rossignac and Requicha 1986）。Z缓冲器算法是从物体的表面 190
导出的，而不是按像素计算光线。考虑两种算法的总体结构：

- 光线跟踪

```
for每一个像素do
    产生一束光线并求出与该光线相交的所有物体表面
    计算CSG树，确定沿着光线方向的第一个表面的边界
    应用Z缓冲器算法，进行明暗处理（或不进行明暗处理）
```

- Z缓冲器

```
for每一个基元物体do
    for基元物体的表面F do
        forF上的一个足够致密的网格中的每一个点F do
            对P投影并应用Z缓冲器算法
            if当前为可见的then
                if通过估算CSG树，确定P是表面的边界then
                    绘制到帧缓冲器中
```

必须对这两种算法按CSG树的降序进行测试，用布尔集合运算求出物体的边界。但是，Z缓冲器算法避免了与每一条像素光线相关的相交测试。

6.6.3 Z缓冲器与合成

基于Z缓冲器的算法的一个重要的优点是，与每一个像素相关的z值都被保留下来，并用于使分别产生的场景元素进行合成或合并。

三维图像经常是由分离的子图像组成的。Porter and Duff（1984）和Duff（1985）提出了一种按像素把场景中分离的元素合成起来的系统。这个简单的系统是基于一种RGBαZ表示的，该表示用于子图像中的像素。其中的α参数使得子图像能够分别地建立并合并起来，方法是保留子像素信息，这些信息是在每一个子图像的绘制中已经计算过的。

通过二元操作符结合两个子图像来建立合成图像：

$$c = f \ \mathbf{op} \ b$$

例如，考虑操作符$Z_{\min}$。我们可以有两个子图像，例如两个单个的物体，这两个物体已经被分别绘制好了，而在最终的绘制中每一个像素的Z值都包含在Z通道中。在这种情况下，合成意味着实现了物体之间隐藏面的消除。对每一个像素，定义为：

$$RGB_c = (\mathbf{if}\ Z_f < Z_b\ \mathbf{then}\ RGB_f\ \mathbf{else}\ RGB_b)$$

$$Z_c = \min(Z_f, Z_b)$$ 191

参数α（$0 \leqslant \alpha \leqslant 1$）是物体所覆盖的像素区域的分数，是控制两个图像的颜色混合的因子。α通道的使用有效地将区域反走样扩展到了图像的合成。当然，这个参数通常不是由基本的Z缓冲器绘制程序来计算。正因为如此，该方法只适用于在A缓冲器隐藏面消除算法

(Carpenter 1984) 中使用，14.6节中介绍了一种对Z缓冲器的反走样扩展。

操作符over定义为:

$$RGB_c = RGB_f + (1-\alpha_f)RGB_b$$
$$\alpha_c = \alpha_f + (1-\alpha_f)\alpha_b$$

这意味着，随着α_f的减少，在像素上表现出的RGB_b更多。

合成操作符comp对上面的操作符进行结合。当像素的角上的Z值在RGB_f和RGB_b上有差别时，这个运算求出像素的结果值。在四个角上对Z_f和Z_b进行比较。如果Z值不同，则可能有16种输出。这时，将像素称为混乱的。沿着边进行线性插值，并计算一个分数β（即在f位于b之前的位置上像素的区域）。于是，我们有了comp操作符:

$$RGB_c = \beta(f \textbf{ over } b) + (1-\beta)(b \textbf{ over } f)$$

三维图像合成中的另一个例子是Nakamae 等 (1986) 给出的。这是一种蒙太奇方法。也就是说，用这种方法产生的点用于合成三维的计算机生成的图像，比如一个新的建筑，并且有真实的场景照片。这种方法的成功是由于所产生物体的照明是根据背景照片计算出来的，并且把环境效果合成到了完成的图像中。

随着存储（设备）成本的降低以及OpenGL的流行，累加缓冲器的使用也变得普及了。累加缓冲器的一种强大且简单的使用就是用于支持多路绘制技术。这个方法在6.7节中介绍。

6.6.4 Z缓冲器和绘制

Z缓冲器对于数据库的组织没有限制（对明暗处理插值所需的限制除外），其最简化的形式可以按任何方便的顺序用被表示的多边形按多边形导出。

从原理上，我们对每一个多边形计算下列值:

1) 内部像素点的 (x, y) 值。

2) 每一点 (x, y) 的z深度值。

192 3) 每一点 (x, y) 的光强度I。

这样的话，我们同时进行三个双线性插值，即有一个三重的嵌套循环。在每一个顶点处可以得到z值和强度值I，对z和I的插值在算法的两个内部循环中出现。

Z缓冲器隐藏面消除的按多边形的一个扩展版本如下:

```
for所有的x, y do
  Z_Buffer[x, y]:= maximum_depth
for每一个多边形do
  由多边形的边构造一个边列表（即，对每一条边，通过插值为每一条扫描线计算其x、z和I值，并将这些值存
  储到边列表中）
for y:= y_min to y_max do
  for EdgeList[y]中的每一个片段do
    取X_左,X_右,Z_左,Z_右,I_左,I_右的值
    for x:= X_左 to X_右do
      分别在Z_左,Z_右,I_左,I_右之间线性插值z和I
      if z < Z_Buffer[x, y] then
        Z_Buffer[x, y]:= z
        frame_buffer[x, y]:= I
```

算法的结构显示了该方法的主要缺点，该方法的明暗处理计算是在隐藏的像素上进行的，

这些隐藏的像素在其后或者被忽略或者被覆盖。

如果采用Phong的插值算法，则最后的反射模型计算也会出现在最内部循环中，最后的反射模型计算是插值的法向量的一个函数。最内部的循环是插值N而不是插值I，用下面的语句替代以上算法的最后一句:

frame_buffer[x, y]:= *Shading Function*(N)

6.6.5 扫描线Z缓冲器

对于基于扫描线的绘制程序所使用的Z缓冲器算法有一个变体，被称为扫描线Z缓冲器方法（这样的称呼并不令人吃惊）。这只是一个简单的Z缓冲器，只有一个像素高，用于一条已知扫描线上的隐藏面消除问题。对于每一条新的扫描线Z缓冲器均被重新初始化。相对于完全开放的Z缓冲器来说，它的主要优点在于所需要的存储空间很少。通常可以看到一个基于扫描线Z缓冲器的程序运行在没有足够的存储空间来支持完全Z缓冲器的系统上。

6.6.6 跨跃式隐藏面消除

跨跃式隐藏面消除算法试图对每一条扫描线求出可以进行明暗处理的“跨度”。于是，隐藏面消除的问题就通过将扫描线分成一些段加以解决，这些段是由单个表面所占据的。这意味着，对每一个像素只进行一次明暗处理计算，这样就排除了Z缓冲器方法所固有的效率不高的缺点。与之相比的问题是，跨度段有时并不与多边形片段相对应，这使得进行增量式的明暗处理计算更困难了（对多边形片段上的任意一点必须计算其初始值，而不是在左手的边界处设定这些值）。另一个主要的缺陷是，随着场景复杂度的增加，算法本身的复杂性也增加了。我们将在后面看到这种情况。 193

除了针对非常多的多边形之外，一般宣称跨跃式算法比基于Z缓冲器的算法更有效（Foley等 1989; Sutherland等 1974）。然而，现在非常复杂的场景越来越多，显然Z缓冲器算法更有效，除非使用非常复杂的明暗处理函数。

6.6.7 一个跨跃式扫描线算法

正如我们已经提到的，基本的思想是宁愿采用z的增量计算，以像素为基础来解决隐藏面问题。而跨跃式扫描线算法在扫描线上采用跨度，在这个跨度上没有深度的冲突。隐藏面消除过程在x方向利用连贯性，以很多像素为单位进行处理。这种处理的含义是，对于每一条扫描线需要进行x方向上的排序，并且必须求跨度。

要考察扫描线算法如何工作的最容易的方法是，考虑三维屏幕空间（x_s, y_s, z_s）中的情况。扫描线算法高效地移动一个扫描线平面，这个平面与（x_s, z_s）平面平行，移动方向是沿着y_s轴向下。这个平面与场景中的物体相交，从而将隐藏面的问题降到二维空间（x_s, z_s）中。在这个空间中，扫描线平面与物体多边形的相交成为一条直线（见图6-21）。然后将这些线段进行比较，通过“跨度”来解决隐藏面问题。跨度是一条线段的一部分，它被包含在所有活动多边形边的相交点之间。可以将跨度看成是一个连续的单位。在这个连续的单位内部，隐藏面消除的问题是常数，可以在这个跨度上的任一侧通过比较深度来解决这个问题。注意，如果允许有穿过的多边形，则必须采用一种更复杂的算法。

从这一段几何上的概述中可以看到，跨跃式扫描线算法中的第一阶段是按y_s顶点值来对多边

形的边排序。这个排序导致生成一个活动边列表。随着扫描线沿着y轴向下移动，这个列表被不断更新。如果不允许有穿过的多边形，则每一条边与当前扫描线的交点定义了扫描线上的一个点，在这个点上“发生”了一些变化。所以，这些点的集合就定义了所有的跨度边界点。

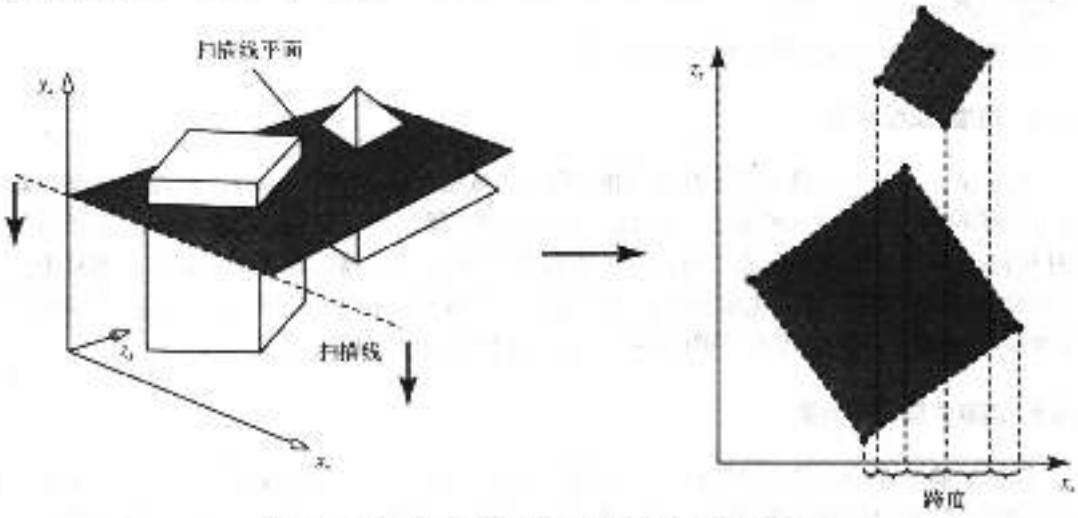


图6-21 扫描线平面在场景中向下移动产生线段和跨度

通过按顺序遍历活动边列表，有可能产生一个线段的集合，其中的每一条线段都代表了扫描线平面与一个多边形的相交。然后，将这些线段按x的升序进行排列。

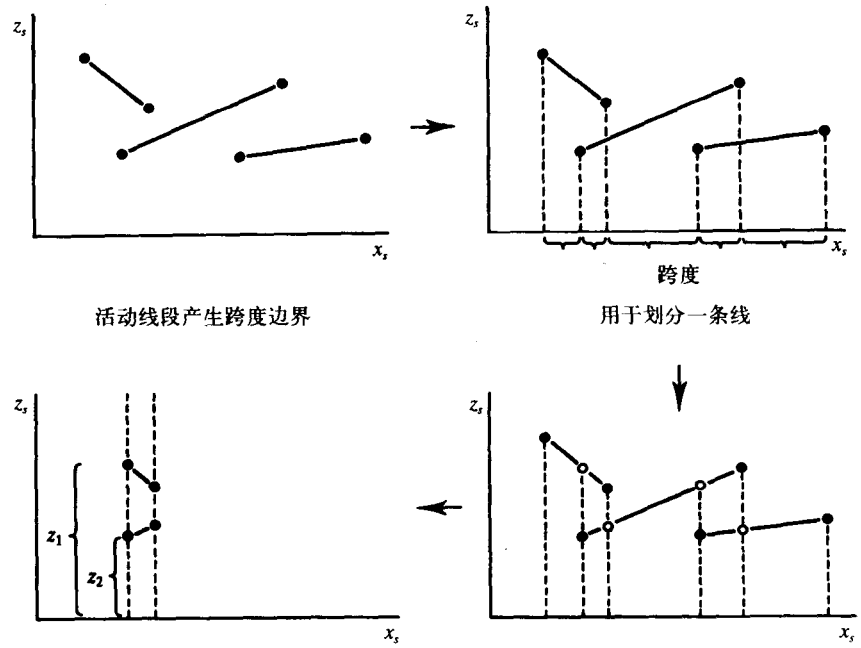
然后，最内部的循环处理当前扫描线上的每一个跨度。在跨度的边界对活动的线段进行裁剪，这样就使这些线段被边界所划分了。然后，在每一个跨度边界上求每一个被分线段的深度，并通过在跨度中查找最靠近的被分线段来实现隐藏面的消除。这个过程如图6-22所示。

算法的伪码如下：

```
for每一个多边形do
  产生并按y,初排序多边形边的信息
for每一条扫描线do
  for每一个活动多边形do
    确定扫描线平面和多边形的片段或相交点
    按x,对这些活动片段排序
    更新每一条扫描线的明暗处理参数变化速率
    产生跨度边界
      for每一个跨度do
      对跨度边界裁剪活动片段
      在一个跨度边界上求所有被裁剪线段的深度
      对具有最小z的跨度求解隐藏面问题
      对可见的被裁剪线段进行明暗处理
      按每个象素的变换速度乘以跨度的宽度更新所有其他线段的明暗处理参数
```

请注意，对明暗处理信息的集成要比Z缓冲器方法繁琐得多。对被裁剪的线段端点的值的记录必须保存起来，并更新。这在过程对效率和存储空间的需求之外又增加了另一个场景的复杂性（除了多边形的绝对个数之外）。

194
195



活动线段产生跨度边界　　用于划分一条线

在跨度中，隐藏面问题通过在一个跨度边界上考察深度值来解决

图6-22 处理跨度

6.6.8 Z缓冲器和复杂场景

20世纪70年代，在计算机图形学中关于隐藏面消除的研究占了很重要的地位。但是随后由于工业界接受了Z缓冲器算法，隐藏面消除的问题就被认为解决了。到了20世纪80年代，尽管对于改进Z缓冲器算法的研究还在继续——主要研究走样和图像合成的方法，但是绘制方面有关光线传输模型的研究已经成为主流。Z缓冲器方法具有众所周知的效率问题，也许对于虚拟现实的图像生成的需求会使人们重新强调高效的隐藏面消除算法的重要性。

Z缓冲器算法的缺点是它将绘制看不到的多边形。如果我们要保留传统Z缓冲器方法的优点，则避免复杂的场景是一个主要的高代价的因素。Z缓冲器算法的主要优点是它的简单性，这个优点是对于每一个多边形来说计算成本较低。它利用了图像的空间连贯性，用一个简单的增量计算处理了多边形投影中的连续的像素。但是，随着所绘制的场景越来越复杂，多边形越来越小，这个优点很快就消失了。在多边形顶点处的预置计算要比按像素计算优越。 196

场景的深度复杂性是场景中的物体个数的一个函数，正像物体的复杂性是一个函数一样。我们希望在虚拟现实应用中绘制的场景都既有深度也有物体复杂性。

1993年Greene等人提出了一个方法，该方法采用了基本的Z缓冲器算法，并将其发展成为一个适合于复杂场景的方法。与标准的Z缓冲器方法相比，Greene报告说，其算法缩减了绘制时间，对于一个含有53 000 000个多边形的场景，算法所需时间从原来的超过1小时缩减到6.45秒（尽管这里有一个重要的实际因素起了作用，即能够促使这种减少是因为大场景是由只有15 000个多边形的小场景重复出现来构建的）。该算法的另一个重要贡献是，只需进行很少的设计变动就可以在现有的硬件上实现。

Z缓冲器算法效率不高的原因在于它是一种图像空间的算法。它不能利用物体空间连贯性。而一个利用了物体空间连贯性的算法是光线跟踪算法中的隐藏面处理部分，光线跟踪算法在光线跟踪时采用一种空间划分策略（例如，八叉树）。对于每条光线投射，光线所击中的第一个表面就是可见表面，在算法中不考虑这个表面的背后出现的相交（另一方面，光线跟踪算法没有利用图像的空间连贯性，每一个像素计算之间都是互相独立的）。Greene的改进注意到了这一点，他在传统的Z缓冲器算法中用了空间划分技术来加入物体空间连贯性的因素。他还采用了一种所谓的Z金字塔来进一步加速对传统的Z缓冲器图像空间连贯性的处理。

物体空间连贯性是通过为场景构建一个常规的八叉树来建立的，并利用这个八叉树来指导绘制策略。如果与结点相对应的那个立方体的所有面对于Z缓冲器都是隐藏的，则八叉树上的这个结点是隐结点，当然，如果是这种情况的话，该立方体所包含的所有多边形或完整的物体也都是隐藏的。这一事实导致了显而易见的绘制策略，也就是从树的根结点开始，通过绘制立方体的每一个侧面来“绘制”八叉树，从而确定整个立方体是否是隐藏的。如果立方体不是隐藏的，我们就继续处理立方体内部。这样一来，大量的隐藏多边形在绘制立方体表面时被删掉了，所以我们就在Z缓冲器中对常规的随意的多边形顺序安排了一个绘制的顺序。Greene指出，如果将被绘制的立方体投影到大量的像素上的话，这一方法本身的处理是昂贵的。这样的考虑利用了常规的Z缓冲器关于图像空间连贯性的优点。

Z金字塔是一种策略，它试图确定完整的多边形的可见性，而不进行像素的逐个详细比较。Z金字塔是一个详尽的层次结构，原来的Z缓冲器处于最底层。在每一个层次上都有一个半分辨率的Z缓冲器，其中一个单元的z值是通过取其下一个层次上四个单元中的最大z值获得的。对Z金字塔的维护包括沿着较高分辨率的方向向上层跟踪，直到遇到与当前深度值相同的深度为止。用Z金字塔来测试立方体表面的可见性包括求最细的细节层次，其在屏幕空间上的相应
197 投影刚好覆盖了该表面的投影。接下来的工作就是一个简单的比较，也就是把该表面的最近处的顶点深度与Z金字塔中的值相比较。用Z金字塔来测试一个完整的多边形的可见性是同样的，只是这时采用了多边形的屏幕空间限定体。

这种技术就是用这样的方法来利用物体和图像的空间连贯性。采用空间细分的方法来加速隐藏面消除是一种古老的想法，这个想法由Schumaker等（1969）首次提出。它的应用是飞行模拟，而在当时对于实时的限制是一个可怕的问题。

通过保存前一帧的可见立方体的方法利用了暂时连贯性。对于当前帧，首先绘制位于这些立方体中的多边形，而立方体就这样被标记。算法按正常方式继续。这种策略之所以可行是因为通常情况下，前一帧中的大多数立方体仍然可见。当前帧中有几个立方体变为不可见，而前一帧中几个不可见的立方体变成了可见立方体。

6.6.9 Z缓冲器总结

从易于实现的角度来考虑，Z缓冲器是最好的算法。它对存储有很大的需求，尤其是对高分辨率的帧缓冲器，但是，它对于场景的复杂性没有上限的要求，这是一个当前越来越重要的优点。对场景它一次绘制一个物体，而对每一个物体一次绘制一个多边形。到目前为止，从数据库的角度来考虑这是一种自然而且使用方便的顺序。

对于用Z缓冲器可以绘制的物体形状的一个重要限制是，不经过高代价的修改，这个方法不能处理透明的物体。一个部分透明的物体可以：

1) 被一个不透明的较近处的多边形完全覆盖，在这种情况下不会出现问题。

2) 是最近处的多边形，在这种情况下必须保留其后面的所有多边形的一个列表，以便对透明多边形和下一个最近的多边形之间的某种适当结合进行计算（下一个最近的多边形直到所有的多边形均被处理过之前当然是未知的）。

与扫描线算法相比，反走样算法尤其是它的硬件实现更困难。

Cook，Carpenter and Catmull（1987）指出，Z缓冲器有一个非常重要的“系统”优点。它提供了一个“后门”，使得其点样本可以与一些具有其他功能的算法（比如光线跟踪或辐射度方法）得到的点样本相结合。

如果对存储空间的需求太大，则第二个最好的解决方案是扫描线Z缓冲器算法。除非绘制程序是在简单场景上进行高效的工作，否则，对于是否值得考虑采用复杂度大幅增加的跨跃式扫描线算法是令人怀疑的。

198

从历史的角度来看，一直有一个从隐藏面消除问题向现实图像合成迁移的问题。易于获得高的空间分辨率和颜色分辨率的终端一直是这种迁移的动因。所有“经典”的隐藏面消除算法都先于对复杂物体明暗处理的方法被开发出来，看起来Z缓冲器算法将成为传统绘制方法中最流行的“幸存者”。

6.6.10 BSP树和隐藏面消除

在第2章中介绍了BSP树的思想，现在我们对它进行更详细的研究，尤其是考察它如何用于执行隐藏面消除的计算。

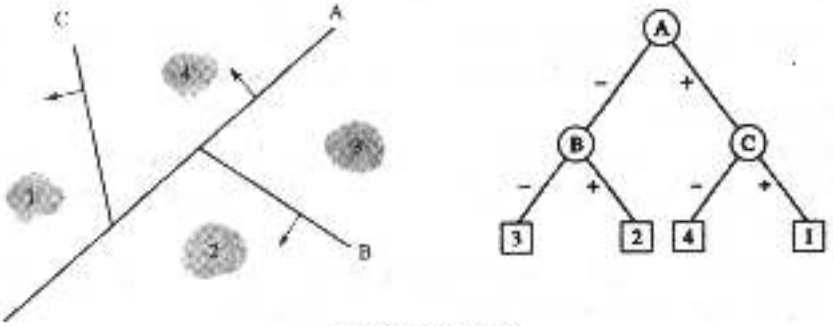
在将BSP树引入到计算机图形学应用程序之后将近20年，这一方法似乎一直主要用在飞行模拟上。随着三维视频游戏和其他动画应用程序在PC上的出现，使我们看到BSP树的一种全新应用，即用其进行可视化计算。现在我们将较详细地讨论这一方法。

原始的BSP隐藏面消除的思想是基于有一个静态的场景和一个变动的观察点。这是典型的飞行模拟和计算机游戏应用。这个方法分两个阶段进行。在第一阶段，建立场景BSP树（只建立一次，这个过程在实际应用中是离线完成的），在第二阶段，将观察点与这个结构相比较以确定可见性。这一方法在实时绘图中之所以有吸引力，是因为很多可见性的处理工作都是按预处理来进行的。首先，我们将了解较简单的情况，即确定在物体中的可见性。接下来分析这些原则是如何被扩展到确定物体中的多边形可见性的。

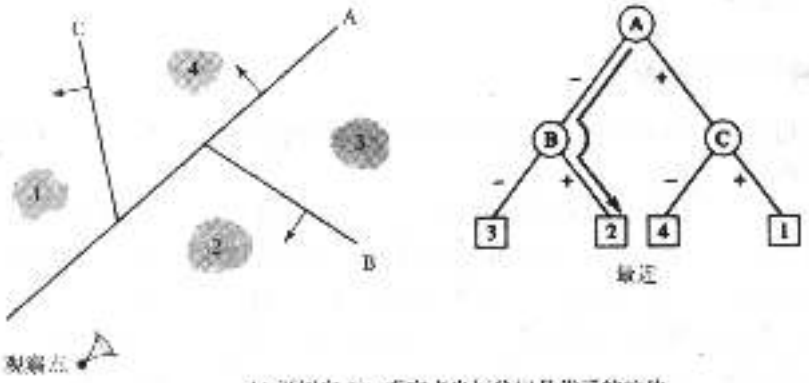
如果场景是由一些可以被凸状区域分隔开的凸状物体组成的，而这些凸状区域又构成平面，则可以采用递归的分治策略来划分空间。假设有一个放置平面的适当策略，而且当所有的区域中都只含有一个物体时树是完整的。这一思想如图6-23所示。树上每一个叶子都是标识物体的一个标签，每一个结点都是一个分离的平面。构建了一棵树之后（见图6-23a），我们就可以确定可见性的顺序，使一个观察点从根结点降序向下移动，直到观察点的坐标给出最靠近观察点的物体为止（见图6-23b）。从根结点开始，我们从最靠近观察点的A平面一侧的子树向下（在本例情况下是反面）到达与平面B相对应的结点，此后到达物体2。图6-23c表明了用于确定可见性顺序的路线。物体3是下一个最靠近观察点的物体，返回到根结点，再向下，余下的顺序是物体1，接着是物体4。这个结果是一个从近到远的顺序，对于这个场景来说，此顺序为物体2, 3, 1, 4。也可以很容易地产生从远到近的排序。

在实际应用中，这个方案还不是很有用，因为大多数计算机图形学应用程序都是由场景

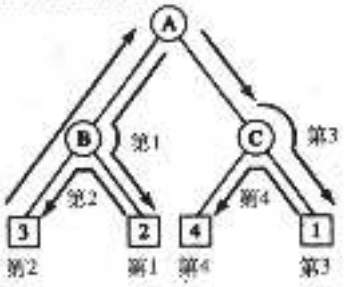
199 组成的。而这些场景中的物体复杂性（每个物体的多边形个数）远远大于场景的复杂性（每个场景中物体的个数）。为了使算法有用，我们必须处理物体中的多边形而不是只处理物体。还有一个为平面定位的问题，这个问题本身并不简单。如果物体的个数小于我们可以承受的，则每一对物体有一个分离平面，对于具有n个物体的场景，这个总数为n^2。



a) 构造一棵BSP树



b) 沿树向下，观察点坐标给出最靠近的物体



c) 求所有物体的可见性顺序

图6-23 对于有四个物体的场景的BSP运算

对于多边形可视性排序来说，我们可以选择那些含有平表面的多边形的平面，选择一个多边形，并将其作为根结点。所有其他的多边形都与含有这个多边形的平面进行测试，并将其置于下面分支的一个适当位置。任何与根多边形平面相交的多边形都被分离成两个部分。这个过程递归进行，直到一个平面包含了所有的多边形。很明显，这个过程产生的多边形比场景中原有的多边形要多。但是实践表明，这个值不会大于系数2。

对于一个简单的例子运用这一过程如图6-24所示。所选择的第一个平面是平面A，它含有一个物体1上的多边形，并将物体3分离成两部分。像前面那样建立树。现在我们用常规的IN/OUT来表示一个实体落在分区的哪一边，因为这一表示具有与多边形物体相关的含义。

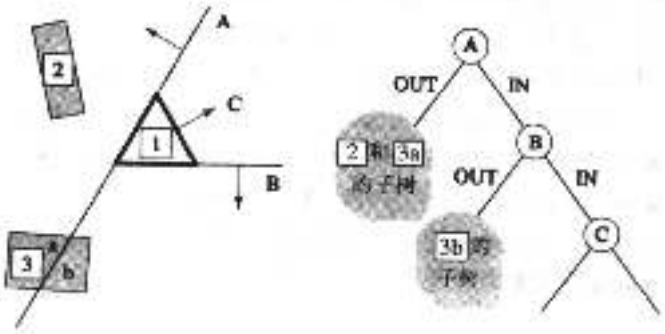


图6-24 一棵用于多边形的BSP树

BSP树原来的组织方式是从远到近的。以这个顺序将多边形绘制到帧缓冲器中导致了所谓的画家算法，也就是将近处的多边形写在远一些的多边形的“上方”。也可以采用从近到远的顺序。但是，在这种情况下，我们必须以某种方式标记已经访问过的像素。如果采用某些策略来防止对完全被遮挡的表面的绘制，那么对于非常复杂的场景来说，宜采用由近到远的排序。例如，通过把它们的图像平面的延伸与（已经绘制的）较近的表面的投影相比较。

于是，为了产生一个场景的可见性排序，我们进行如下工作：

- 按观察点坐标使树降序。
- 在每一个结点上，确定观察点位于结点平面的前方还是位于其后方。
- 先在较远一侧的子树上由上向下运行，并输出多边形。
- 在较近一侧的子树上由上向下运行，并输出多边形。

从多边形相对于当前观察点的位置来看，这导致了一种由后向前的排序，而多边形就以这种顺序被绘制到了帧缓冲器中。如果运用了这样的过程，则这个算法与Z缓冲器算法具有同样的缺点，即被绘制的多边形可能在接下来的运算中被遮住。然而，Z缓冲器算法的一个缺点被立即克服了。多边形排序无需任何附加工作就可以允许无限制地使用透明性，透明的多边形只是根据它们的透明值进行合成。

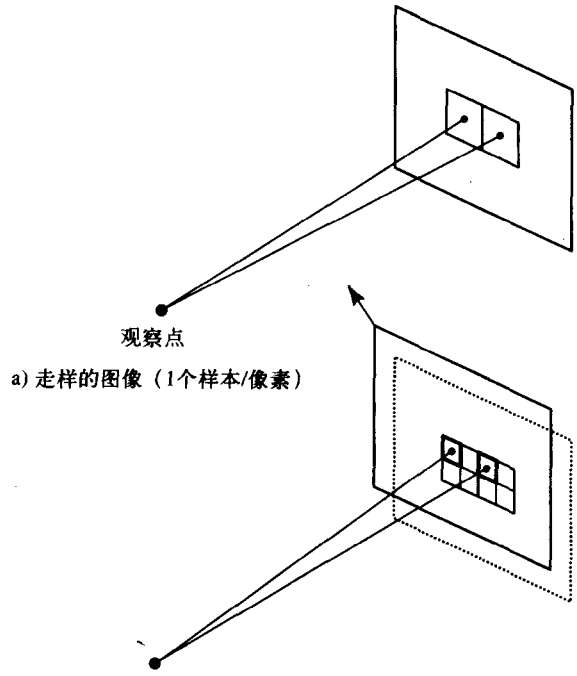
200
|
201

6.7 多路绘制和累加缓冲器

我们所概述的绘制策略都是单路的方法，所绘制的图像是由绘图流程中的单路形成的。在6.6.3节中，我们考察了一种功能，通过加入αZ分量使得分别绘制图像的某些操作成为可能。在这一节中，我们将讨论多路绘制，即通过对场景中图像的一种结合来合成场景中的一个图像，该场景通过在绘图流程中以不同的绘制参数传递进行绘制。这种方法是可行的，这是因为硬件的不断扩展以及专用于绘制的存储器的出现。纹理映射硬件就是证明，这种硬件大大地增加了PC机上产生实时图像可视的复杂性。还可以使用多个屏幕分辨率缓冲器，例如模板缓冲器（stencil buffer）和累加缓冲器（以及帧缓冲器和Z缓冲器）。累加缓冲器（accumulation buffer）是A缓冲器的一个简化。利用这类设备的可能性导致了采用多路技术的算法的发展。

顾名思义，累加缓冲器累积所绘制的图像，其标准的运算为加法和结合到“带权重的加”操作中的乘法。实际应用中，累加缓冲器可以比屏幕缓冲器具有较高的精确度以便消除取整误差的作用。累加缓冲器的使用使得特定的单路算法的作用可以通过多路来获得。多路上的运算完成之后，将累加缓冲器中的最终结果传送到屏幕缓冲器中。

最简单的例子是常见的反走样算法（对这个方法的详细论述见14.7节），该算法产生一个$n \times$最终图像分辨率大小的虚拟图像，然后再用一个过滤器将这个图像还原成最终图像。通过抖动观察部位，产生n个图像，再把这些图像以适当的权重累加起来，也可以获得相同的效果。其中，权重值是抖动值的函数。在图6-25中，为了产生四个这样的图像，需要对每一个像素采样四次，把观察窗口以1/2像素沿水平方向和垂直方向移动。为了求出这个偏移量，只需要以像素为单位计算观察部分的大小（注意，用第5章中给出的简单观察系统是不能完成这一任务的，因为这个系统假设观察窗口总是在穿过观察点的线的中心位置上）。



a) 走样的图像（1个样本/像素）

b) 反走样图像（4个样本/像素或四路）的一个分量/路。对于这一路，观察点向左上方移动了1/2个像素

图6-25 多路超采样

在这种情况下，我们只有保存在存储器中。然而，在很多情况下，按多路绘制方式实现的算法复杂度都比单路的同样算法的复杂度要低。在Haeberli and Akeley（1990）的论文中给出了运动模糊、软阴影和场深度的一些附加的例子。可以通过分布式的光线跟踪来达到这些效果，分布式光线跟踪在第10章中讨论，两种方法的复杂性差别也是很显然的。

为了产生一个运动模糊的图像，只需要在移动场景中的物体时不断改变场景的位置，并把这样绘制出来的一系列图像累加起来。与反走样的例子完全相似，现在我们是在时间域里进行反走样。对于运动模糊有两种方法。一种方法是通过将累加缓冲器中建立起来的n个图像取平均值并以单个图像的形式显示它。另一种方法是通过在一个窗口中按时间平分n个帧来显

示每一个计算的图像。为了做到这一点，开始要累加n个图像。下一步，将$n-1$帧之前累加起来的帧再次绘制，并将其从累加缓冲器中减掉。然后，显示累加缓冲器中的内容。这样，在产生了初始序列之后，每次显示一帧，绘制两帧，即第$n-1$帧和当前帧。

通过按我们在反走样中所做的那样抖动观察窗口以及观察点可以得到对场深度（depth of field）的模拟。场深度是在照片上看到的效果。在照片上根据镜头和光圈的设定值的不同，距摄像机一定距离的物体是聚焦的，而其他较近或较远的物体都是不聚焦和模糊的。抖动观察窗口使得所有的物体都是不聚焦的，而再同时抖动观察点就保证了在等价的聚焦平面中的物体仍保持聚焦。这个思想如图6-26所示。确定一个完全聚焦平面，选择观察点抖动值和观察点扰动以便在完全聚焦平面上保持一个公共的矩形区。对于观察平截体应用一个总的变换，包括裁剪和平移。同样，用5.2节中所介绍的简单的观察平截体是不能完成这一功能的。因为这种简单系统不允许裁剪投影。 202 ~ 203

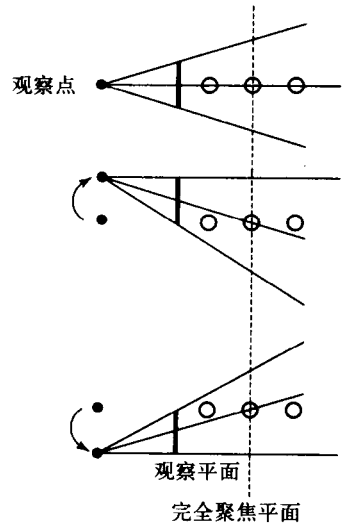


图6-26 通过裁剪观察平截体和变换观察点对场深度进行模拟

通过累加n路，并改变多路之间点光源的位置来模拟一个区域的采样可以很容易地建立软阴影。很明显，用这种方法也能够使来自不同的光源的阴影得到绘制。 204

第7章 模拟光线——物体相交：局部反射模型

7.1 来自完全表面的反射
7.2 来自不完全表面的反射
7.3 双向反射分布函数
7.4 漫反射分量和镜面反射分量
7.5 完全漫反射——经验型散布镜面反射
7.6 基于物理的镜面反射
7.7 预计算BRDF
7.8 基于物理的漫反射分量

引言

自20世纪70年代中期以来，局部反射模型，尤其是Phong模型（在第5章中介绍）一直是主流的绘制程序的组成部分。当与多边形插值明暗处理方法相结合后，局部反射模型就被纳入到了几乎所有的传统绘制程序中。其局部性的明显限制成为这种模型的最大缺点。但是，尽管可以利用光线跟踪程序和辐射度绘制程序，主流的绘制方法却仍然是我们前面描述的策略的某些变体。换句话说，局部反射模型是这些过程的中心。然而，近年来也很难找到没有附加像纹理映射和阴影计算这样的附件的绘制程序了（见第8章和第9章）。纹理映射增加了趣味和变化，而几何阴影计算克服了局部模型的最显著的缺陷。

尽管强调全局模型的发展是可以理解的，但还是有相当大量的研究工作致力于改进局部反射模型。然而，人们并没有对这些加以更多的注意，大多数绘制程序还是采用Phong模型，一方面，人们致力于这种技术的效率和简化。另一方面，人们却忽略了在这一领域已经取得的实质性的进步。

有关局部模型的一个重点是，将它们用在了某些全局方案中。正如我们将在第12章中讨论的那样，最简单的光线跟踪程序是模型的混合，它把局部反射模型与全局光线跟踪模型相结合。在每一点上用局部模型求出该点上可以看到的所有方向上的照明的一个贡献值，再加上一个考虑非直接照明的（光线跟踪的）分量（事实上，这是相互矛盾的，因为对于局部和全局的贡献使用了不同的参数。但是，在实际应用中这种方法被广泛接受）。

在这一章中，我们考察一个有代表性的局部模型，以便探究如下的问题：我们如何来模拟光线反射现象的差别？比如，具有相同颜色的闪光的塑料和金属。通常，人们可以感觉到真实物体的这种细微的差别，而我们也应该能够在计算机图形学中模拟这种现象。

大多数局部反射模型的基础是一个经验的方法。在这个方法中，我们设计了一个简易的评估函数来模拟来自一个表面的光的反射，或者建立一个对完全表面的反射理论加上对一个不完全表面的模拟。

7.1 来自完全表面的反射

我们从考察一个光学上光滑的表面（一个完全的镜面）上光的入射现象开始。入射行为由Fresnel公式来确定，这个公式由Maxwell波动方程导出。它还是1.4.6节中给出的光线跟踪公式的来源。公式本身是计算一个系数，这个系数把反射能量和透射能量关联为一个入射方向、极化作用和材料性质的函数。为了简单起见，假设光是非极化的（这一方法在计算机图形学中通常被采用），光线在空气中传播（近似为真空），并假设称为消光系数（extinction coefficient）(7-1）的因数（见7.6.4节）为零。于是，我们有：

$$F = \frac{1}{2}\left\{\frac{\sin^2(\phi-\theta)}{\sin^2(\phi+\theta)} + \frac{\tan^2(\phi-\theta)}{\tan^2(\phi+\theta)}\right\} \tag{7-1}$$

其中：

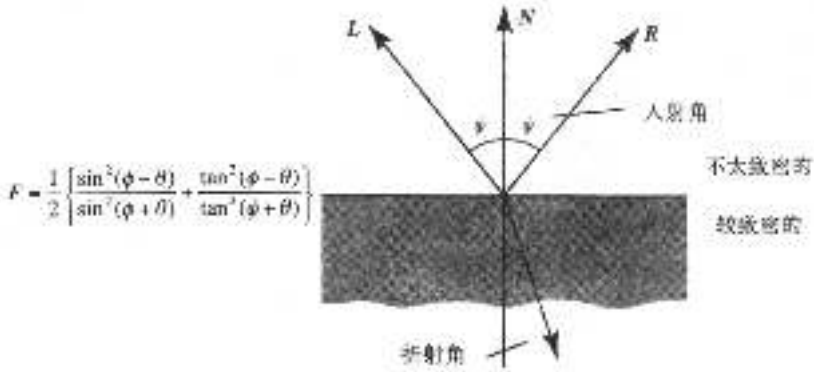
ϕ是入射角；

θ是折射角；

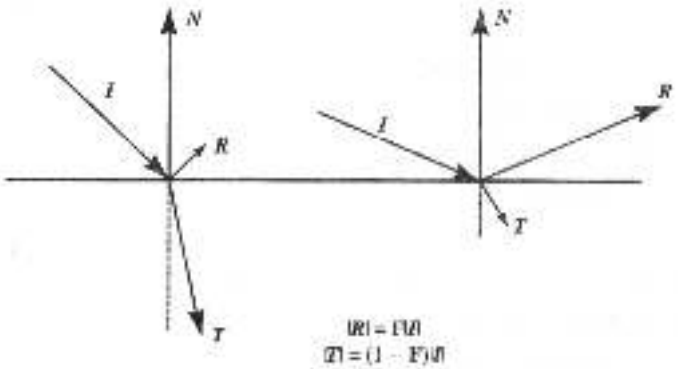
$\sin\theta = \sin\phi/\mu$(其中$\mu$是材料的折射指数)。

这些角度的含义如图7-1所示。当$\phi = 0$或者法向入射时，F最小，即大多数光线被吸收了。

2016 对于$\phi = \pi/2$，F等于1，表面不吸收光线。F的波长依赖性质是因为μ是波长的一个函数。



a) Fresnel 方程中的角

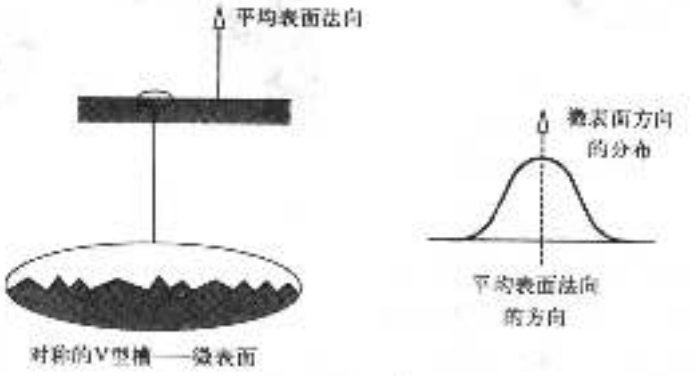


b) 显示方程行为的两个例子

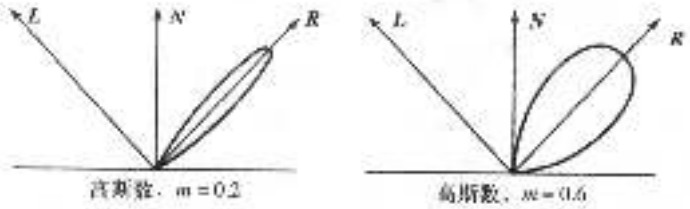
图7-1 Fresnel方程

7.2 来自不完全表面的反射

在实际应用中，表面并不是光学上完美的。除了玻璃或水之外，所有的表面都表现出一种微观几何性质。然而，如果把Fresnel方程与模拟微观几何的模型结合起来的话，Fresnel方程还是有用的。图7-2是这种结合的一种形式，即将表面看成是很多微表面的集合，并且为了简单起见，把这些微表面看成是对称的V形槽。在一个小的范围内，我们可以把入射光的反射描述在这种沟槽的一个代表性的区域上，以便形成一个可以对其参数化的波束（lobe）。



a) 将一个表面按全部为V型槽进行建模



b) 在高斯分布中对于不同的m值的反射波束

图7-2 用一个微表面的集合模拟一个粗糙表面，每一个小平面均被看成是完全镜面

当然，表面微观几何在现实中并不仅仅是不完全的。例如，闪光的金属表面的老化并形成一个脏的表面以及划痕这样的大的不完整。这类“真实”表面更难于建模，必须强调的是，按照我们所描述的方法对微观几何进行了建模的表面仍然假设成是非常干净的，这是实际中不存在的现象。

7.3 双向反射分布函数

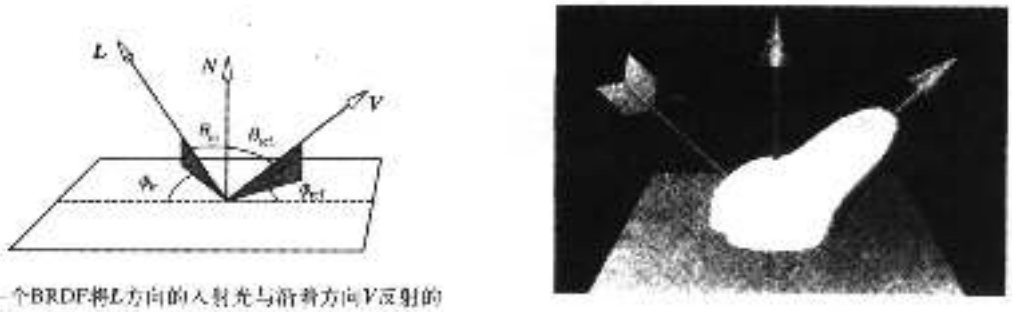
一般而言，物体表面上一点的反射光线可以用双向反射分布函数（bi-directional reflection distribution function, BRDF）来分类，这个术语强调，任何方向上的反射光都不仅仅是这个方向的函数，而且是入射光方向的函数。在计算机图形学中，我们主要对观察方向V上的反射光感兴趣，BRDF可以写成：

$$\mathrm{BRDF} = f(\theta_{in}, \phi_{in}, \theta_{ref}, \phi_{ref}) = f(\boldsymbol{L}, \boldsymbol{V})$$

在计算机图形学中所采用的许多模型都是不同的，主要根据对那些依赖因素进行模拟而加以区分。图7-3为这些角以及对于一组特定的角计算的BRDF。绘制的BRDF表示了在所示方

207
↓
208

向上一束无限细的入射光的反射光的大小（以任意的输出方向）。在实际过程中，光线可能从多个方向向表面入射。因此，应该通过对每一束入射光分别求BRDF并且求和来得到反射光。



a）一个BRDF将L方向的入射光与沿着方向V反射的光相关联，构成一个角度θ_i、ϕ_i、θ_r、ϕ_r的函数

b）一个BRDF的例子

图7-3 双向反射函数

很多年以来，计算机图形学一直用简单的高度受限的BRDF，如图7-3所示。图7-4给出了一种思想，说明了这样的计算机图形学模型与实际中真正发生的情况之间的差别。这个示意图是一个在含有L和R的平面中的BRDF的截面，其中，L和R为不同角度θ的镜面方向，而角度θ为入射角（和反射角）。尤其应注意，反射波束作为入射光的波长、入射角以及材料的函数具有很大的变化。对于铝质材料，根据入射光的波长的不同它可以表现得像一个镜面，或者像一个带方向的漫反射表面。我们还应该考虑到，在实际中，入射光从来都不是单色的（于是，就需要对于每一种波长有一个BRDF）。我们看到，反射光的行为远比用简单的近似方法进行建模的情况要复杂得多，而像Phong模型只近似地建模三种波长。

各向同性的表面和各向异性的表面之间必须有一种重要的区别。各向同性的表面表现为一种BRDF，其形状不依赖于入射的方位角ϕ（见图7-3）。而各向异性的表面是比如擦过的铝表面或者仍然保持着来自磨床的相关特性的表面。对于擦过的表面，其镜面波束的大小依入射光线是否与表面的特性相一致而定。

另一个现实中出现的复杂情况是大气的性质。在局部反射模型中采用的大多数BRDF都限定到应用来自真空中的不透明物体的反射光。大部分情况下，我们不考虑反射光在大气中的散射（我们同样也不考虑光线到达物体之前的散射）。当然，这样做的原因是为了简化计算，因此也减少了对光强度的计算量，使其只是简单地对按表面形状、光线方向和观察方向进行分类的那些向量进行比较。

可以想象，如果我们对一种材料有了一个BRDF，则光线与物体相交的问题就解决了。但是，尽管已经进行了四分之一世纪的研究，现在仍然还是有很多问题没有解决，其中包括：

- 从何处获得BRDF（尤其是对于那些实际的与完美材料不同的材料）？在治金类刊物上可以获得某些材料的数据，但这是远远不够的。
- 以什么样的比例来表示BRDF？接收入射光的区域是多大？这个区域是否应该足够大，以使得统计模型在表面上一致？这个表面是否应该足够大，以便包含一些表面的不完美之处，比如划痕？这些都是远远没有解决的问题。
- 应如何表示BRDF？最后这一点考虑了模型之间的不同，尤其是，经验模型和基于物理的模型经常有差别。经验模型是用来模拟光线与物体的交互的，例如，在Phong模型中

用了一个简单的数学函数来表示镜面波束。而在Cook和Torrance模型中，用一个统计分布来表示表面的几何形状。这个模型称为基于物理的模型（Cook and Torrance 1982）。有趣的是，对于经验模型和基于物理的模型可见的效率问题并没有达成共识。通常，通过小心地调节经验模型的参数可以比用基于物理的模型得到更好的结果。

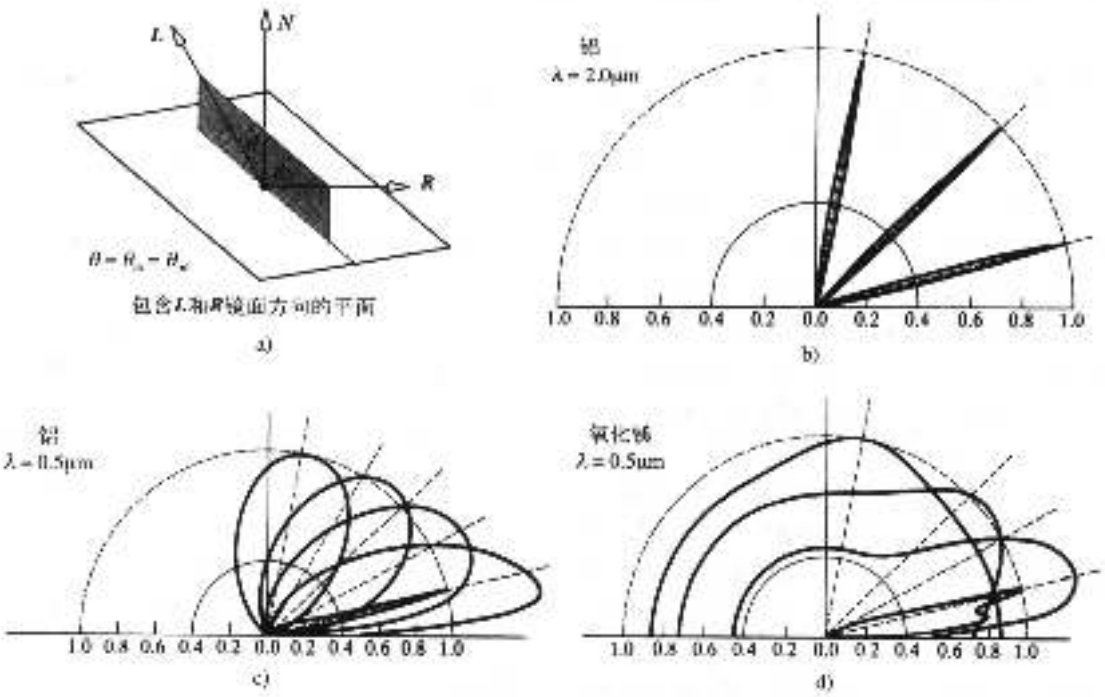


图7-4 不同材质和波长的BRDF截面（He等（1991））

接下来是对早期的局部反射模型的一个综述，以及对近期的进步的简短评述。我们从考察Phong模型固有的缺陷以及如何克服这些缺陷开始。这并不意味着做出了一个综合性的总结，只是希望对Phong模型的某些变化有一个表述，因为光线"画"到物体上的方式提供了很多细节。

7.4 漫反射分量和镜面反射分量

在计算机图形学中采用的局部反射模型通常被认为是一个漫反射分量和一个镜面反射分量的合成。这种合成对于很多情况都可以工作得很好。但是，它只是一种简化形式。镜面反射的更简化的模型把一些不完全的现象看成是对于完全镜面反射的一种修正。当光线射到一个完全的镜面上，并且一束入射光在这样的表面上按照众所周知的定律（即反射角等于入射角）反射时，出现完全镜面反射。当入射光从一个完全不光滑的镜面朝所有方向等量散射时出现完全漫反射。这种表面在实际中可以是一个非常薄的粒子的膜。将分别计算的镜面反射分量和漫反射分量相结合可以模仿真实表面的行为。并且，这也是许多计算机图形学模型中所做的假设。通过把各种效果加入到镜面反射分量中已经取得了大部分对真实表面之间细微的视觉差别的模拟。我们通过研究这类模型中的一些模型将考察这一点，这些模型是：

209 / 211

1) Phong模型——完全漫反射与经验型散布镜面反射相结合（Phong 1975）。

2) 由Blinn（1977）和Cook and Torrance（1982）提出的基于物理的镜面反射模型。

3) Cabral等（1987）提出的，在绘制过程中被索引的预计算BRDF。

4) 由Hanrahan and Kreuger（1993）提出的基于物理的漫反射模型。

这些模型既是历史样例，同时又说明了研究者对局部反射模型所采用的不同方法。

7.5 完全漫反射——经验型散布镜面反射

事实上，这就是Phong反射模型。我们已经讨论了这个模型的实用性，尤其是论述了如何将其集成到绘制系统中去。在这里，我们将从更理论性的观点来考察这个模型，以便能够将它与其他直接反射模型相比较。

Phong反射模型用Lambert的余弦定理来计算漫反射。在这个定律中，反射光的强度是表面法向和入射光方向间余弦的一个函数。

Phong用了一个经验型散布镜面项。其思想是，一个实际的表面，比如闪光的金属表面，反射出的围绕完全镜面反射的方向形成一个波束区域，这是因为可以将其看作是由一些小镜面（所有这些小镜面的方向都有微小的差别）组成。而不是由一个完全光滑的以物体为形状的镜面组成。于是，该（闪光）表面的粗糙度就可以用指数n来模拟，n的值越大，波束越紧，表面越光滑。用这个模型模拟的所有表面都有一种像塑料一样的外观。

从几何角度来看，在三维空间中，该模型产生一个以$\boldsymbol{R}$为中心的光线的圆锥体。随着光线与$\boldsymbol{R}$之间的夹角的增加其密度按指数降低。

现实行为的一个更细微的方面，即说明了塑料和闪光金属之间差别的方面，则完全没有在模型中出现。这个方面是镜面反射的光线的量依赖于入射光的仰角θ_i（见图7-3）。在阳光下开汽车，可以看到从路面发出炫目的强光——在中午暗表面只有很少或没有镜面反射分量。为了计入这一现象，也就是对于任何物体，把高光的细微变化看作是入射光方向的一个函数，出现了早期的基于对表面的物理微表面模拟的局部反射模型。

尽管在Phong模型中，镜面反射的方向依赖于入射光的方向，但是，我们可以说，镜面反
212 射在镜面方向周围是对称分布的，其大小是不变的，Phong模型实现了将BRDF降为:

$$\text{BRDF} = f(\theta_{\text{ref}}, \phi_{\text{ref}})$$

图7-3中所示的BRDF是用Phong反射模型计算的。

7.6 基于物理的镜面反射

在Phong于1975年提出其模型之后两年，Blinn（1977）发表了他的文章，阐述了如何在计算机图形学中使用基于物理的镜面反射分量。1982年，Cook和Torrance将这一模型扩展到高光的光谱合成，即高光与材料类型及光线的入射角之间的依赖关系。与用Phong模型所获得的产品相比较，这些模型在高光的大小和颜色上产生了细微的作用。模型仍然把反射光分成漫反射成分和镜面反射成分。新的模型把注意力完全集中到镜面反射部分，而对于漫反射部分的计算与以前的方法相同。这个模型最适合绘制闪光的金属表面，且通过对镜面高光的颜色变化能够对相似颜色的金属进行不同的绘制。

高光的形状问题是相当难以捉摸的，高光只是一个光源的图像，或者是物体中反射出的光源的形状。除非物体表面是平的，否则图像会扭曲，而且随着入射光线方向的变化，它会落在物体的不同部位上，并且其形状也会改变。因此，我们所获得的高光图像的形状依赖于入射光投射到的物体表面上那一块区域的曲率以及观察方向，观察方向确定了从观察方向来

看有多少高光是可见的。这些都是我们确定在物体表面上可以看到的闪光的曲面片形状的基本因素，用Phong模型很容易计算这些值。

决定高光图像的另一个因素是光强度和颜色对入射光与所关心的表面上该点的切平面之间的夹角的依赖程度。这个依赖程度表明了材料的性质，使我们能够将金属物质和非金属物质区分开来。

令人惊奇的是，尽管产生出了更精确的高光，但是这些模型并没有被图形学界采纳，廉价且简单的Phong模型仍较流行，而且近年来它也确实得到普及。造成这种情况的可能原因是，由更精细的模型所产生的差别是微小的。由Phong模型所绘制出来的物体尽管不精确，而且在高光的绘制上也是不正确的，但是它产生的物体看起来真实。不论以前还是现在，在大多数计算机图形学的应用程序中，这正是人们所希望的。正如我们多次提到的，模型的三维计算机图形的照片真实性除了局部反射模型之外，还依赖于很多因素。为了使物体看起来更真实，仅仅从这么窄的角度来考虑似乎不值得付出这么多。 213

光线反射的物理模拟所意味的是，我们力图建模引起光线反射的表面微观几何，而不是像Phong模型那样简单地用经验项来模拟其表现行为。

这种对镜面高光的早期模拟有四个分量，并且是基于物理的微表面模型，该模型模拟在一个平均的表面上出现的对称的V型槽（见图7-2）。现在我们依次来讨论这四个分量。

7.6.1 建模表面的微观几何

对于微观表面的方向建立一个统计的分布，这个分布在特定的方向（观察方向）上出现的光以项D来给出。可以用一个简单的高斯数:

$$D = k\exp[-(\alpha/m)^2]$$

其中α是微平面与表面（平均）法向之间的夹角，也就是N和H之间的夹角，m是分布的标准偏差。在这一角度上求分布只得到这一方向上的微观表面的个数，也就是对观察方向上出现的光线做出贡献的微表面的个数。图7-2b为两个（$m = 0.2$和$m = 0.6$）反射波束。

采用微表面来模拟光线反射对表面粗糙度的依赖给出了两个假设:

1) 尽管微表面从物理的层面上很小，但仍假设它相对于光线的波长来说还是足够大的。

2) 入射光束足够大，以致与一些微表面相交，这对于产生代表反射光的行为是足够的。

在BRDF项中，这个因素控制镜面作用的扩展。

7.6.2 阴影和屏蔽效果

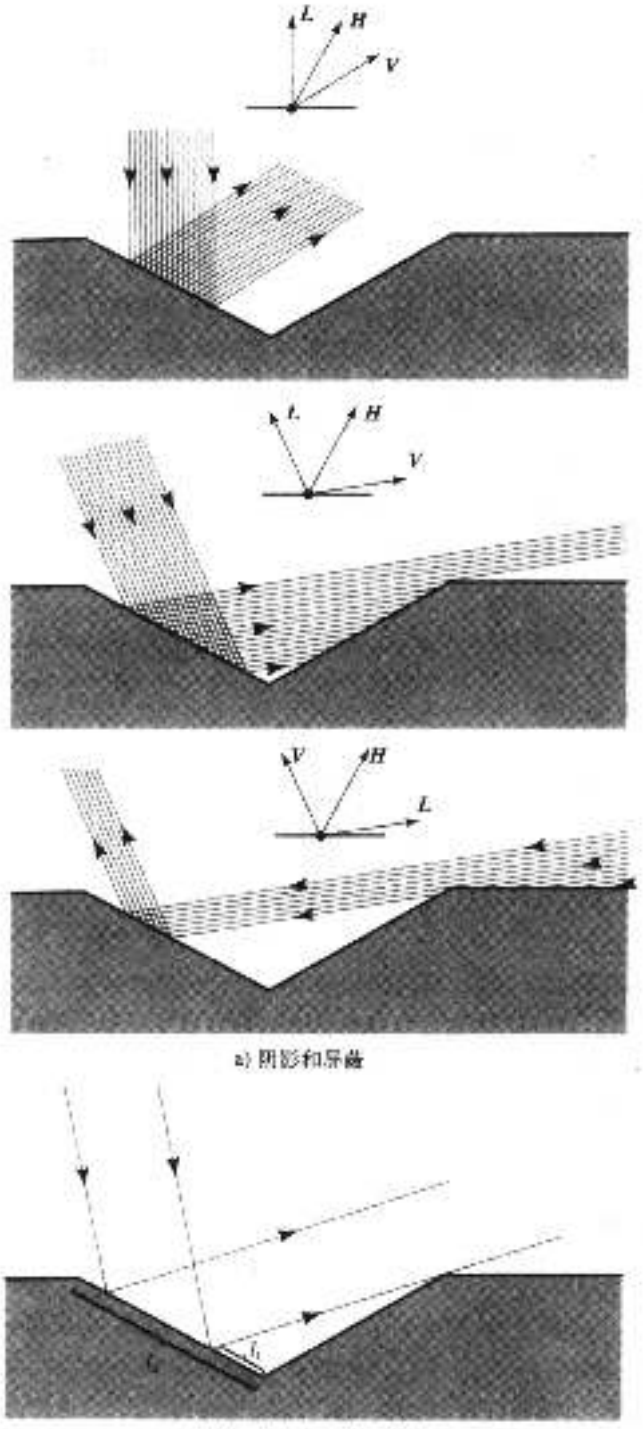
在观察向量或者光线方向向量开始向平均表面移动时，出现干涉效果。这种效果被称为阴影（shadowing）和屏蔽（masking）。当一些反射光被遮挡时出现屏蔽，而当入射光被拦截时出现阴影。如图7-5所示。

屏蔽和阴影的程度依l_1/l_2的比例而定（见图7-5b），该比例描述对反射光有贡献的小表面的比例数量，由下式给出: 214

$$G = 1 - l_1/l_2$$

当l_1减小到零时，所有的反射光都逃逸掉了，于是:

$$G = 1$$



a) 阴影和屏蔽

b) 第1 l_1段逃逸的光线的量

图7-5 光线与微反射表面的相互作用

Blinn（1977）给出了对l_1/l_2依赖于L、V和H的详细推导。对于屏蔽：

$$G_m = 2(N \cdot H)(N \cdot V)/V \cdot H$$

对于阴影，从几何上来讲，情况是相同的，只是L和V进行了互换。于是，对阴影有：

$$G_s = 2(N \cdot H)(N \cdot L)/V \cdot H$$

必须用到的G值是G_s与G_m中的最小值。即：

$$G = \min\{1, G_s, G_m\}$$

7.6.3 观察几何学

采用了另一个纯几何项来处理7.5节中提到的闪烁效果。随着观察向量与平均表面法向之间的夹角越来越接近90°，观察者看到越来越多的微表面，由一个项来计算：

$$1/N \cdot V$$

也就是说，观察者所看到的微表面数量的增加反比于观察方向与表面法向之间的夹角。如果入射光的角度小，则光线朝观察者方向的反射就多于当入射光靠近表面法向时观察者得到的光。这一效果通过阴影效果来计算，而这一效果也是随着观察方向向平均表面方向靠近而开始起作用的。

7.6.4 Fresnel项

下一个要考虑的项是Fresnel项，F（见7.1节）。这一项考虑反射光而不是被吸收的光的量，这是一个依赖于作为完全镜面表面的材料类型的因素，而反射光的量是每一个微表面都有的。换句话说，我们现在将以前所建模的完全平的表面看作是各个都表现出完全镜面的微表面的一个集合。这个因素决定了反射波束的强度作为入射角和波长的函数。而波长一项包含了在镜面高光中的细微的颜色作用。

对于任一入射角，计算F所需的系数通常是未知的。Cook and Torrance（1982）提出了一个实际的解决方案，即使用已知的F_0值（经过测量的）来计算μ的值，这是在法向入射时的F值。然后，再对任一入射角用方程（7-1）求F值。在法向入射时，方程（7-1）化简为：

215
~
216

$$F_0 = \frac{(\mu-1)^2}{(\mu+1)^2}$$

（μ事实上复杂的，它含有一个想象的成分，即消光系数。对于像塑料这样的绝缘体，这一项为零。而对于像金属这样的导体，在法向入射时这一项也是零。因此在法向入射时可以将这两类材料的这一项忽略不计。）

另一种从F_0计算任一入射角的F值的方法是Schick（1993）提出的，其公式为：

$$F_\phi = F_0 + (1-\cos\phi)^5(1-F_0)$$

这一项的实际作用是把镜面高光的颜色上的细微变化看作是入射角的一个函数。对于任何材料，当光线以几乎平行于表面的角度入射时，高光的颜色将趋近于光源的颜色。对于其他入射角，这一颜色依赖于入射角和材质。这种依赖性的一个例子是抛光铜，如图7-6所示。

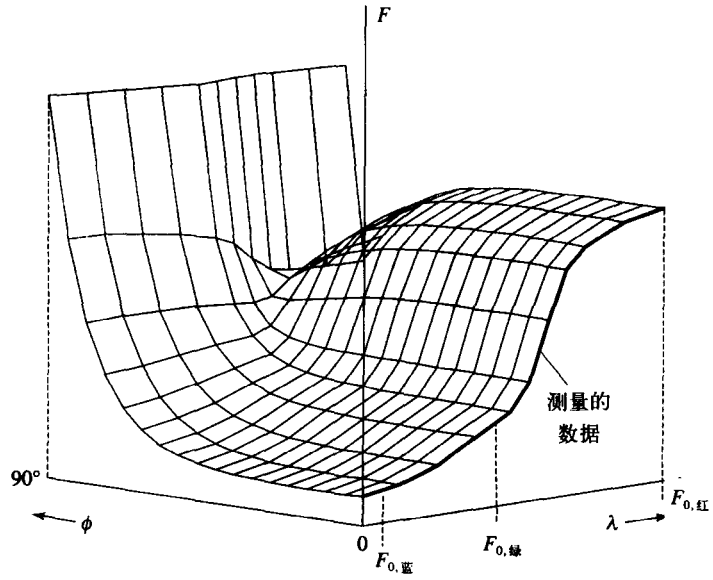
这一项的作用是使反射的光强度随着入射角的增加而增加（就像前一项$1/N \cdot V$一样）。也就是说，材料吸收了更少的光而反射了更多的光（更细微的作用是，随着入射角的增加，镜面波束的峰值也移动并离开了完全镜面反射的方向，见图7-7）。

把这几项放到一起来考虑，则镜面项就成为：

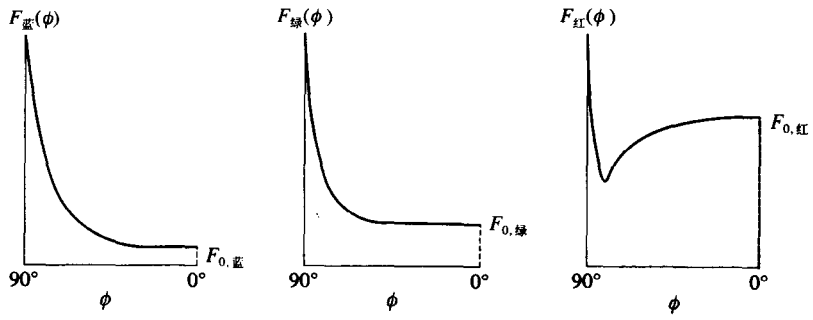
$$\text{镜面分量} = \text{DGF}/(N \cdot V)$$

其中：

D为微观几何项;
G为阴影/屏蔽项;
F为Fresnel项;
$(N \cdot V)$为闪烁效果项。



a) 反射系数F作为波长（λ）和入射角（抛光的铜）的函数



b) F对红、绿和蓝波长中ϕ的依赖性

图7-6 Fresnel方程和抛光的铜

总结如下:

1) 一个因素将反射光的光强度建模为表面物理性质的函数，并简化成对几何模拟的近似。

2) 两个交互因素模拟“闪烁”效果的行为，闪烁效果在光线以很大的入射角入射时出现(相对于表面法向N而言)。

3) 一个因素将每一个微表面（完全镜面）上的反射光光强度与材料的光电性质进行关联。这是一个入射光的方向的函数，它控制了细微的处于第二位的关于高光的颜色和形状的效果。在试图模拟闪光的塑料和金属之间的差别时，这一效果是重要的。例如，当用白光照明时，金属表现出黄色的高光，只有当光线擦过表面时，高光才趋于白色。

镜面项是分别计算的，并且与一个均匀漫反射项相结合:

$$\text{BRDF} = sR_s + dR_d \text{（其中} s + d = 1\text{）}$$

例如，通常用$d = 0$和$s = 1$来模拟金属，而用$d = 0.9$和$s = 0.1$来模拟闪光的塑料。注意，如果将d设为零，则对于金属来讲，镜面项就控制了物体整个表面上的颜色。将这一表示与Phong反射模型相比较，Phong模型中物体的颜色总是由漫反射分量来控制。正因为这一点，Phong模型不能产生看起来像金属的表面，而用Phong模型绘制出的所有表面都明显看起来像塑料。 217 / 218

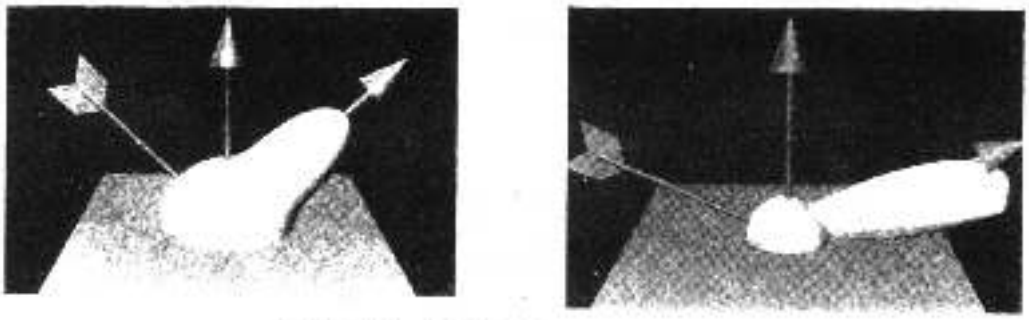


图7-7 在Blinn反射模型中不同入射角的BRDF

在这个模型中，反射光的光强度依赖于入射光的仰角，但这个模型不依赖于入射光的方位角。不管入射光的方位角方向如何，它都有一个长的、平行的、对称V型槽式的统计分布（在实际应用中这是一种几乎不可能出现的情况）。于是：

$$\text{BRDF} = f(\theta_{in}, \theta_{out}, \phi_{in})$$

分别对应于较小和较大入射角的BRDF如图7-7所示。这个图表明，随着入射角的增加，镜面波束的值也增加（并且也会偏离镜面反射方向）。图7-8（彩色插图）给出了用这一模型可以获得的物体外观上的变化。

7.7 预计算BRDF

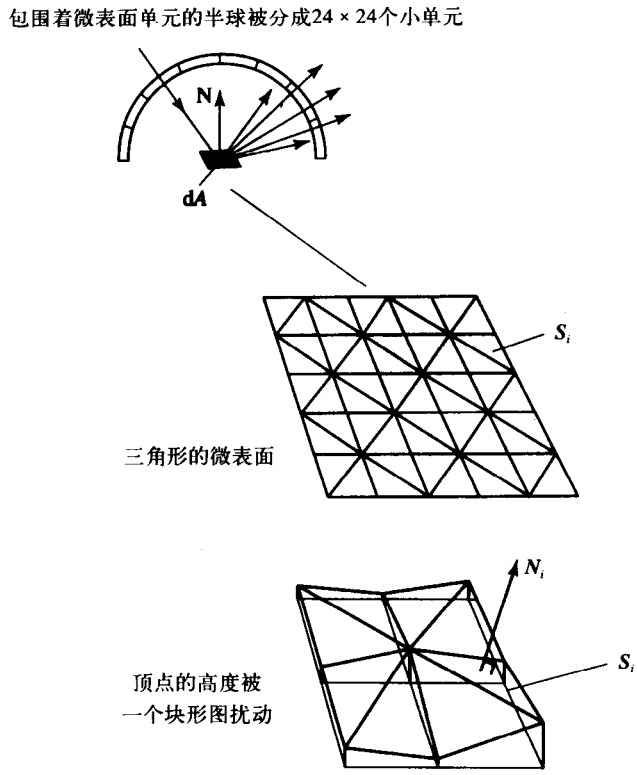
前面介绍的方法的主要不足之一是，它不能用于各向异性的表面。而许多表面都表现出各向异性的特性。布和"装饰"工程应用中所用的抛光的金属，如汽车的轮子，就是两个例子。例如，我们考虑布的情况，它表现出各向异性是因为它是用带有圆截面的平行的线做成的。当入射光线处于一个平行于线的平面上时，每一条线都会有少量的散射光线。而当入射平面平行于线的圆截面时散射面更宽一些。把BRDF中这种各向异性行为结合起来的两个流行方法一直是建立特殊表面模型（通常是基于圆柱体的）以及预计算。

1987年，Cabral等（1987）提出了一个模型，这个模型可以处理入射光线的方位角的依赖性问题。它对每一个L预计算了一个BRDF，L是通过把一个半球分成以V为索引的小容器（bin）的方式来表示的。BRDF通过跟踪每一个入射方向的光线来计算，入射光线是一束平行的随机放置的光线，当其射到表面上时进行反射并射到周围的半球上。BRDF的角度的依赖性为： 219

$$\text{BRDF} = f(\theta_{in}, \phi_{in}, \theta_{out}, \phi_{out})$$

BRDF是通过向包围了微表面的足够大的区域的一个表面单元发射光线或光束产生的。这个表面单元用三角形微表面的一个数组或网格来建模。那些击中未被遮挡的单元并未被屏蔽地射出的光线构成了对BRDF的贡献，完整的函数是所有这些贡献的总和。通过将一个半球划

分成一些小的单元来构建这一信息。这个过程的一种表示如图7-9所示。在图中，表面上的微表面用块形图由平均平面扰动产生。注意，这个过程的一个优点是对于小型平面的几何形状没有限制，例如，微表面不必形成高斯分布。



220

图7-9 用高度场被扰动的三角形微表面建模表面

于是，这个BRDF就是一个连续的BRDF经粗略采样后的版本。预计算的建立是通过把每一个单元看作是一个入射光的光源，并计算所导致的向半球中的反射来完成的。当要对表面进行绘制时，选择最靠近入射角的那个半球。这时，这一分布的北极必须与该点的表面法向相同，而预计算的BRDF给出观察方向上的反射强度。

7.8 基于物理的漫反射分量

直到最近，在计算机图形学中局部反射模型几乎一直把注意力全部放在反射光的镜面反射分量上。正如我们已经看到的那样，这些分量是基于物理的微表面建模的。

漫反射光通常都用Lanbert的余弦定理来建模，这个定理假设反射光是各向同性的，其强度正比于入射角的余弦。对漫反射光的表面模拟是不可能的，因为漫反射光实际上可能来自材料的内部。这一分量在材料内部被吸收并散射。依赖于波长的吸收与材料的颜色有关。实际上，入射的白色光被材料进行了过滤。引起出射光（emerging light）（近乎）各向同性的原因是在材料内部的多次散射。因此，对漫反射的物理模拟必须基于表面的散射。

我们可能会提出这样的问题，即采用Lambert定理时出了什么错？这一问题的答案可能与对建立基于物理的镜面模型的动机提出的问题的答案相同。即，由漫反射光产生细微的效果，

而这些效果对区分一些材料的外观是重要的。Hanrahan and Kreuger（1993）近期的工作是建立了一个用于漫反射的基于物理的模型，作者称这个模型尤其适合那些本质上看起来有层次的材料，例如生物组织（皮、叶子等）以及雪和沙等无机物。当然，这个模型的输出是各向异性的。它反映了这样的事实，即很少有现实的材料表现出各向同性的漫反射。

Hanrahan和Kreuger将表面上一个点的反射光定义为:

$$L_r = L_{rs} + L_{rv}$$

其中L_{rs}是由于表面散射而造成的反射光，是一种不完全镜面反射；L_{rv}是由于子表面散射而造成的反射光。确定子表面散射的算法是基于一个一维传递模型，用蒙特卡罗算法来求解。其细节已经超出了本书的范围，对于我们来说，更重要的是理解这些研究人员所做出的成果的概念及其可视化衍生物。

这两个分量的结合产生各向异性的特性，这是因为一些我们将要讨论的因素造成的。首先，考虑光线的入射角。对于平的表面，进入表面的光线的量由Fresnel定律给出，进入表面 221
的光线越多，子表面对总的反射光L_r的贡献或影响就越大，所以L_{rv}的影响依赖于入射角。子表面的散射依赖于材料的物理性质。通过悬浮一个散射点或粒子，并用吸收和散射截面进行参数化来建模材料。这些因素都表示了每单位路径上散射或吸收的出现概率。这些参数的相对大小决定了散射是前向的、后向的，还是各向同性的。

对于简单的情况，这两个因素的作用如图7-10所示。第一行表示的是高/低镜面反射为入射角的函数。反射光的行为在入射角较大时由表面散射或镜面反射所决定，当入射角较小时由子表面散射所决定。正如我们已经了解到的那样，由7.6节中介绍的Cook Torrance模型对这种行为进行了一定程度的建模。第二行展示的是由于子表面的散射而产生的反射波束。可以看出，材料可以表现出后向的、各向同性的或者前向的散射特性（当然底部的波束对L_r没有贡献，但是，当考虑的材料由很多层组成或者是薄的黑色半透明材料时这一因素也是重要的）。第三行展示的是L_{rs}和L_{rv}的结合一般来讲将导致非各向同性现象，它表明了下列一般的性质:

- 由于增加了子表面的散射，所以随着材料厚度的增加反射增加。
- 子表面散射可以是后向的、各向同性的或者前向的。
- 与Lambert定理的半球（理想化的）相比，子表面散射造成的反射趋向于使得波束的顶部变平缓。

这些因素导致了此模型和Lambert定理之间的细微差别。

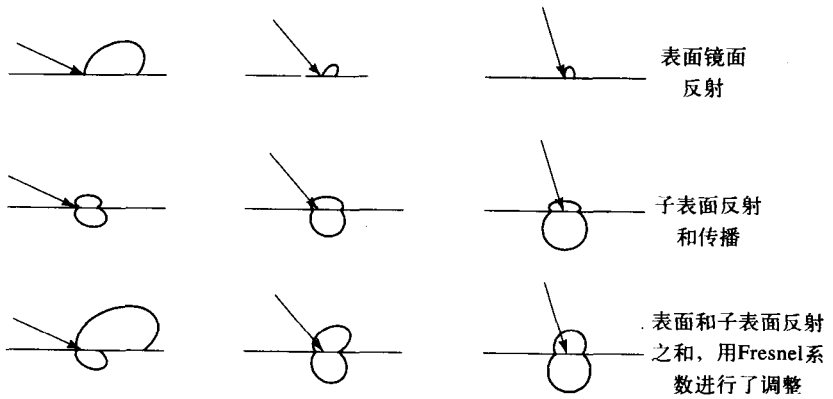


图7-10 Hanrahan 和Kreuger模型产生的反射特性 222

第8章 映射技术

8.1 二维纹理映射到多边形网格物体
8.2 二维纹理域到双三次参数曲面片物体
8.3 广告牌
8.4 凹凸映射
8.5 光线图
8.6 环境映射或反射映射
8.7 三维纹理域技术
8.8 反走样和纹理映射
8.9 纹理映射中的交互技术

引言

在这一章中，我们将考察通常在二维域中存储信息并在绘制时用这种信息模拟纹理的技术。其主要的应用是纹理映射，但同时介绍了许多其他的应用，比如用反射映射来模拟光线跟踪。随着纹理映射硬件的发展，利用这些设备实现实时的绘制促进了光线映射技术的发展。这些技术利用纹理功能可以进行（不依赖于观察的）光照的预计算，于是，这种预计算就将绘制“降”为纹理映射运算了。

在20世纪80年代，纹理映射就成为了一种高度发展的工具。并且，这种技术被用于增强经Phong明暗处理过的场景，使这些场景在视觉上更吸引人，看起来更真实或更神秘。仅仅用Phong明暗处理绘制的物体看起来像塑料，而纹理映射明显是增加效果且不会付出很大代价的方法。

纹理映射技术的进步与关于全局照明算法（即光线跟踪和辐射度方法，见第10、11和12章）的研究是并行进行的。纹理映射是一种可以用于增强场景的视觉效果的方法，而不是具有像照片一样的真实性，其主要的吸引人之处是它的廉价。可以将其移植到标准的绘制方法中而不需要增加太多的处理成本。这与全局照明方法刚好相反，该方法采用了完全不同的算法，它比直接反射模型要花费更大的代价。

在20世纪80年代，纹理映射的另一个无所不在的应用是，通过把周围的环境反射到物体上来将虚拟的现实添加到闪光的动画物体上。这样一来，翻滚着落下来的徽标和标题就被镀了铬而反射到其上的纹理也随着物体的移动而动了起来。这一技术称为环境映射。还可以将这一技术用于现实的拍照环境，有助于将计算机动画的物体合并到一个真实的环境中。环境映射并不能实现光线跟踪算法不能完成的任何事，但是它更高效。环境映射技术最近的一个新用途是在基于图像的绘制上，这一内容将在第16章中介绍。

在计算机图形学中使用时，“纹理”是一个很易混淆的术语。一般来说，它并不意味着对一个计算机图形学物体的表面的小尺度几何的控制，而这却是该术语的常规含义。通过控制

三个漫反射系数，很容易对由Phong明暗处理模型绘制的物体的颜色进行调整，而这些也就成为了由纹理映射控制的最普通的物体参数（当然，在物理世界中颜色的变化一般与纹理没有关系）。于是，随着绘制程序一个一个像素的处理过程的进行，我们为Phong漫反射系数选取值，而明暗处理的漫反射（颜色）分量也作为纹理映射的函数发生变化。一个较好的术语是颜色映射，这一术语现在已经很通用了。

这种简单的像素级的运算隐藏着很多困难，纹理映射的几何意义也不是很直观。我们像往常一样对其进行简化，使其能产生视觉上可接受的结果。困难的原因有三:

1) 我们最希望采用的纹理映射是在计算机图形学中最流行的表示——多边形网格表示。正如我们所知，这是一种几何表示，其物体的表面是近似的，而且这种近似只是在顶点上定义。在某种意义上，我们没有表面，而只有对场景的一种近似。所以，如果表面并不存在的话，我们如何物理地导出表面上一点的纹理值呢?

2) 我们主要希望采用二维的纹理图，因为可以通过对现实世界进行帧获取通过二维绘图软件或者通过程序化地产生纹理来取得几乎是无尽的纹理资源。于是，我们的主要需求就是把一个二维的纹理映射到一个由多边形网格近似的表面上。随着廉价的纹理映射硬件设备的发展，这种情况得到了加强。

3) 在纹理映射中，通常都会出现走样问题。根据定义，纹理通常都有某种程度的连贯性
224 或周期性。走样打破了这种规律，而它产生的混乱也是高度可见的。随着纹理的周期性逐渐趋近于像素水平就会出现这种情况。

现在，我们列出一些可能的方法。通过利用这些方法，一个计算机图形学模型的某些性质的变化可以通过控制纹理图来得到调整。下面按照这些方法的流行程度的大致顺序列出它们（这个顺序也倾向于与其易用性或实现有关）。这些方法是:

1) **颜色** 正如我们已经指出的，这是目前为止由纹理图控制的最常见的物体性质。我们只是简单地根据相应的纹理图的颜色调整Phong反射模型中的漫反射系数（还可以将在物体表面上的镜面系数作为纹理图的函数进行改变，使它看起来有闪光或者像铜镜。但是，这种做法不是太常见。因为在使用基本的Phong反射模型时，根据所产生的闪光部位的高光可能在所绘制的物体上会感觉出这个作用)。

2) **镜面“颜色”** 这个技术称为环境映射（environment mapping）或镀铬映射。这是光线跟踪的一种特殊情况，在光线跟踪算法中，用纹理图技术来避免完全光线跟踪的浪费。图是经过设计的，以便使其看起来像（镜面的）物体正在反射它所处的空间中的环境或背景。

3) **法向量扰动** 这一简明的技术根据图中相应的值向表面法向施加一个扰动。该技术称为凹凸映射（bump mapping），是由著名的三维计算机图形学技术的先驱J. Blinn提出的。这一方法之所以可行，是因为如果进行适当的简化的话，由Phong明暗处理方程所返回的强度可以简化为当前被明暗处理的点上表面法向的一个函数。如果对表面法向进行扰动，则明暗处理发生变化，被绘制的表面看起来就像是有了纹理。因此，我们就可以对数据库中以多边形网格结构表示的表面的纹理使用一个全局的或通用的定义。

4) **置换映射** 与前面给出的技术相关联，这种映射方法使用一个高度场，沿着其表面法向的方向扰动一个表面点。这是不容易实施的技术，因为映射必须对模型从几何上进行扰动，而不是调整明暗处理方程中的参数。

5) **透明** 用图来控制透明物体的不透明性。经过蚀刻的玻璃就是一个好例子，这时对闪光的玻璃表面用一些装饰的图案进行打磨（为了引起不透明）。

有很多纹理映射方法，其选择主要依赖于时间限制以及所需图像的质量。开始时，我们将把讨论限制在二维纹理图及多边形网格物体上。二维纹理图是最流行也是最通用的形式 225
（本节中的许多深入的探讨基于Heckbert（1986）的描述，该书定义了这一领域的工作）。

把二维纹理图映射到物体的表面，然后，再将物体投影到屏幕空间，这是一个二维到二维的变换。因此，可以把这一过程看成是一种图像的扭曲运算。最通常的做法是反转图，即对每一个像素求出其在纹理空间中相应的“预置图像”（见图8-1b）然而，对这个整体的变换的定义并不直观，原因我们将很快了解到。开始时，把纹理映射看成是一个两个阶段的过程，从而使我们能够从二维的纹理空间到物体的三维空间，然后，再通过投影变换到二维的屏幕空间（见图8-1a）。第一个变换称为参数化，第二阶段是常规的计算机图形学的投影变换。参数化将纹理空间中的所有点与物体表面上的点相关联。

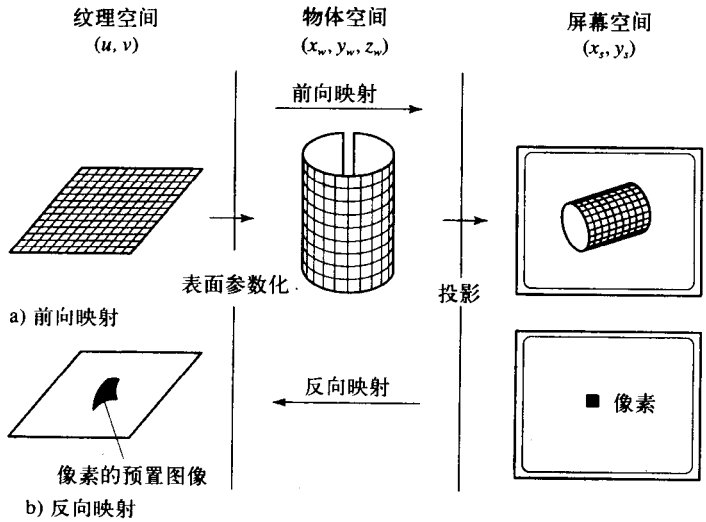


图8-1 两种观察二维纹理映射过程的方法

对反走样方法的使用对于纹理映射是强制性的。从下面讨论的情形可以很容易地看出这一点，即考虑使一个物体撤离观察者，以便使得物体在屏幕空间上的投影占据越来越少的像素。随着物体尺寸的减少，在纹理空间中一个像素的预置图像将增加，并覆盖更大的区域。如果我们简单地在像素的中心点处采样，并取纹理空间中的相应的点的$T(u, v)$值，则将会产生不正确的结果（见图8-2a、b和c）。这一效果的一个例子如图8-3所示。该图中，随着有方格的图案退到一定的距离，它就开始以一定的扰动方式分离。当生成动画时这些问题就高度可见了。考虑图8-2b和c。例如，我们将物体投影到一个像素上，并且以预置图像穿过$T(u, v)$移动的方式运动。随着物体的移动，它的颜色将从黑色变为白色。

在这种情况下，反走样就意味着对像素预置图像上的信息进行积分，并对于当前的像素采用这个值进行明暗处理计算（见图8-2d）。我们最好能估计出这个积分值，因为只有四边形四个顶点的值，而没有关于四边形形状的信息。

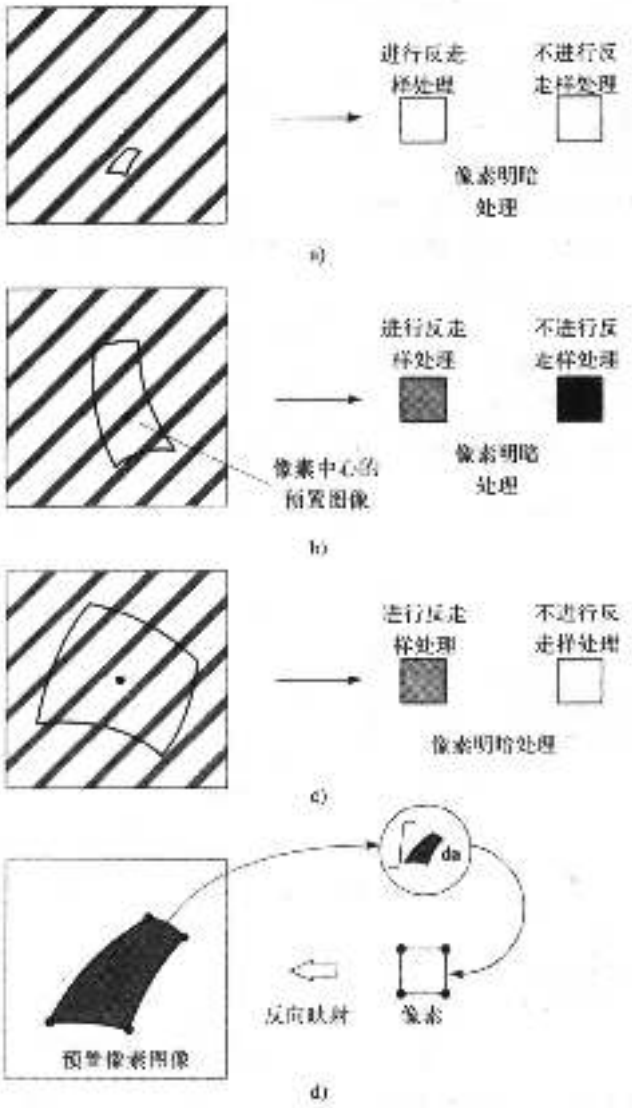


图8-2 $T(u,v)$空间中的像素和预置图像

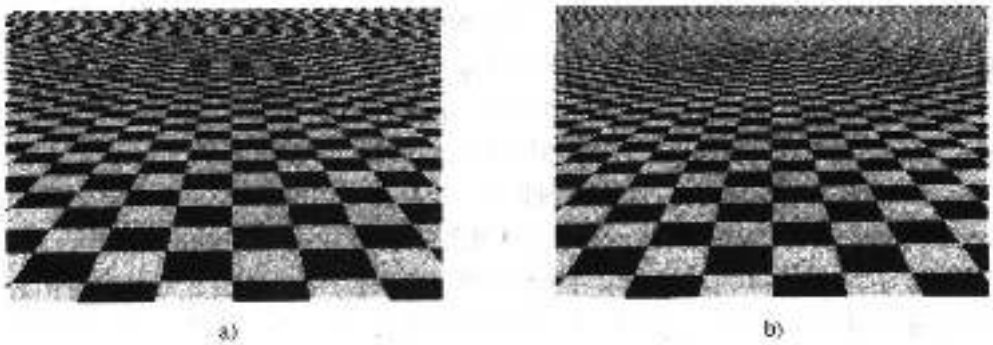


图8-3 纹理映射中的走样。b中的图案是a中图案的一个超采样，仍然有走样，但是走样的空间频率要高一些

8.1 二维纹理映射到多边形网格物体

纹理映射最流行的实现策略是在建模阶段将纹理空间坐标 (u, v) 与多边形的顶点相关联。这时，绘制引擎的任务就是对每一个多边形内的像素求出其相应的坐标 (u, v)。由于只对多边形的顶点进行了几何定义，换句话说，没有解析参数化的可能，所以出现了一个主要的问题（如果物体有一个解析的定义，例如一个圆柱体，则我们有一个参数化，这时纹理向物体表面的映射就很容易了）。

在纹理映射中有两个主要的可行的算法结构，即反向映射（更通用）和前向映射(Heckbert将这两种结构分别称为屏幕顺序算法和纹理顺序算法)。反向映射（见图8-1b）是由屏幕空间驱动的，对于每一个像素点，通过在纹理空间反向映射“预置图像”求得。对于每一个像素点，求出与之相应的 (u, v) 坐标。用一个过滤运算将预置图像中包含的信息进行积分。并把求出的颜色结果赋值给像素。如果将纹理映射与Z缓冲器算法相结合，则这种算法是有优势的。在Z缓冲器算法中将多边形光栅化，并以一条扫描线为基准插值深度和光照。正方形的像素产生一个作为预置图像的弯曲的四边形。

在前向映射中，算法是由纹理空间驱动的。这时，纹理空间中的正方形纹理在屏幕空间产生一个弯曲的四边形。由于映射到屏幕空间时在纹理图像上出现孔和重叠的现象，这就出现了一种潜在的问题。前向映射似乎把纹理图看成是一块橡胶，这块橡胶被拉开撑大（由参数化决定），使得它被贴在了物体的表面上，这样就形成了从常规的物体空间向屏幕空间的变换。

226 ~ 228

8.1.1 用双线性插值进行反向映射

尽管前向映射易于理解，但是在实际的算法中还是愿意采用反向映射。因此，从现在开始，我们只考虑将这种策略用于多边形网格物体。对于反向映射来说，考虑一个（合成的）从二维屏幕空间 (x, y) 到二维纹理空间 (u, v) 的变换是方便的。这只是一种图像扭曲变换，可以将其建模为一种有理线性投影变换:

$$x = \frac{au + bv + c}{gu + hu + i} \quad y = \frac{du + ev + f}{gu + hv + i} \tag{8-1}$$

当然，正像我们所预料的那样，这是一个非线性变换。可以将这一变换换一种形式，写成齐次坐标系中的变换:

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} u' \\ v' \\ q \end{bmatrix}$$

其中:

$$(x, y) = (x'/w, y'/w) \text{ 和 } (u, v) = (u'/q, v'/q)$$

这一变换称为有理线性变换。在实际应用中，我们所感兴趣的逆变换由下式给出:

$$\begin{bmatrix} u' \\ v' \\ q \end{bmatrix} = \begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} \begin{bmatrix} x' \\ y' \\ w \end{bmatrix}$$

$$= \begin{bmatrix} ei - fh & ch - bi & bf - ce \\ fg - di & ai - cg & cd - af \\ dh - eg & bg - ah & ae - bd \end{bmatrix} \begin{bmatrix} x' \\ y' \\ w \end{bmatrix}$$

现在我们回忆一下，在大多数纹理映射应用程序中，在建模阶段都会建立多边形网格的顶点与纹理图坐标之间的关联。所以，比如说，如果我们有了对于一个四边形的四个顶点的关联，就可以求出9个系数 $(a, b, c, d, e, f, g, h, i)$。这样，对任意的多边形内的点，求出所需的逆变换。可以按下面的方法来完成这一工作。回到方程（8-1）的前半部分，即对于x的方程。请注意，由一个任意的非零标量常数，不需要改变y的值就可以将上部和下部乘起来。实际上，我们只有5个自由度而不是6个，正因为这一点，可以不失一般性地设$i = 1$。于是，在总的变换中，只需要确定8个系数，而其余未知数可以通过任意的解线性方程组的标准算法来求解，例如，用高斯消去法。在Heckbert（1986）的文章中对这一过程有
229 非常详尽的论述。

另一种较好的选择方案也取得了相同的效果，即在屏幕空间的双线性插值。所以，这时我们在对光照和深度进行插值的同时也对纹理坐标进行了插值。然而，前面已经提到，必须进行的插值是在齐次坐标系下进行的 $(u', v' q')$，因为u，v并不随x，y进行线性改变。

对于所有多边形的每一个顶点，假设其顶点坐标/纹理坐标的齐次纹理坐标为:

$$(u', v', q)$$

其中:

$$u = u'/q$$
$$v = v'/q$$
$$q = 1/z$$

我们用常规的双线性插值方法在多边形中（见1.5节），将这些齐次坐标作为顶点进行插值，给出每一个像素的 $(u', v' q')$。于是，所需的纹理坐标由下式给出:

$$(u, v) = u'/q, v'/q$$

请注意，这一计算方法对每个像素要进行两次除法。对于这种插值过程用标准的增量方法来实现的话，需要对每一条边有三个梯度（在当前的边对上），以及对当前扫描线需要有三个梯度。

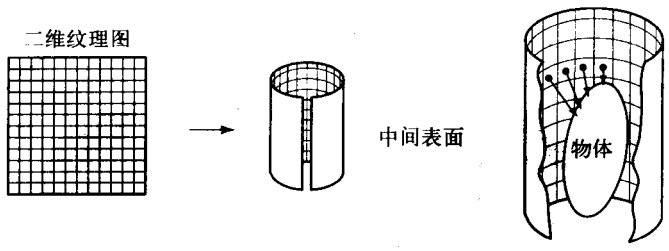
8.1.2 用中间表面进行反向映射

对于映射二维纹理来说，前面提到的方法无疑是当前最流行的方法。而我们现在要讨论的方法可以用在没有纹理坐标和顶点坐标相关性的情况。另一种用途是先用这一方法进行预处理，以便确定这一相关性。然后，在绘制时采用第一种方法。

两段纹理映射是另一种技术，它克服了多边形网格物体上的表面参数化问题，方法是采用一种易操作的中间表面，纹理初始时在中间表面上进行投影。该方法由Bier and Sloan（1986）提出，可以用于进行环境映射。因此，这种方法也对纹理映射和环境映射进行了统一。

这个过程之所以称为两段映射，是因为在纹理被映射到物体上之前先要被映射到中间表面上。一般来说，中间表面是非平面的，但是，它具有一个解析的映射函数，二维纹理可以毫无困难地映射到这种表面上。于是，求出物体点与纹理点之间的相关性就变成了三维到三维的映射。

230 这一方法的基础可以很容易地表示为两段前向映射过程（见图8-4）。



a) S映射　　b) O映射

图8-4 作为前向过程的两段映射

1) 第一阶段是从二维纹理空间到一个简单的三维中间表面（如圆柱体）的映射。

$$T(u, v) \rightarrow T'(x_i, y_i, z_i)$$

这一阶段称为S映射。

2) 第二阶段是把三维纹理图案映射到物体表面。

$$T'(x_i, y_i, z_i) \rightarrow O(x_w, y_w, z_w)$$

这一阶段称为O映射。

这种结合的运算可以将纹理图案以“自然”的方式扭曲到物体的表面上。例如，这个方法的一个变种是“收缩”映射，这时，平整的纹理图案以一种人为规定的方式收缩到物体上。

对于S映射，Bier描述了四种中间表面；一个任意方向上的平面、一个圆柱体的曲面、一个立方体的表面和一个球面。尽管从数学角度来讲这没有什么不同，但是，考虑将$T(u, v)$映射到这些物体的内表面是有用的。例如，考虑圆柱体的情况。将圆柱体的曲面的参数定义为点(θ, h)的一个集合，把点(u, v)变换到圆柱体上。我们有:

$$S_{\text{cylinder}} : (\theta, h) \rightarrow (u, v)$$

$$= \left(\frac{r}{c}(\theta - \theta_0), \frac{1}{d}(h - h_0) \right)$$

其中，c和d是比例因子，θ_0和h_0将纹理定位在半径为r的圆柱体上。

对于O映射，可以出现各种可能性。这时，对于$O(x_w, y_w, z_w)$，纹理值可从$T'(x_i, y_i, z_i)$获得，而这些值最好从光线跟踪的角度进行考虑。4个O映射如图8-5所示。它们是:

1) 反射的观察光线与中间表面的交T'。事实上，这个值与8.6节中所描述的环境映射是相同的。利用这个O映射和环境映射在整个过程中的唯一区别是，映射到中间表面上的纹理图案是周围的环境，像一个房间的内部。

2) 在(x_w, y_w, z_w)处表面法向与T'的交点。

3) 一条穿过(x_w, y_w, z_w)与物体的质心的连线与T'的交点。

4) 从(x_w, y_w, z_w)到T'的连线的交点，连线的方向由点(x_w, y_w, z_w)处的表面法向确定。如果中间表面只是一个平面，则它等同于把纹理图看成是幻灯片投影机中的一张幻灯片。来自幻灯片投影机的一束平行的光线打到物体的表面上。另一种情况是它也等价于高度场是由二维纹理图沿着一条垂直于图案平面的轴“伸展”来定义的三维纹理映射（见8.7节）。

现在，把这个过程作为收缩（shrinkwrap）情况的反向映射过程来考虑。可以把这个过程分为三段（见图8-6）。

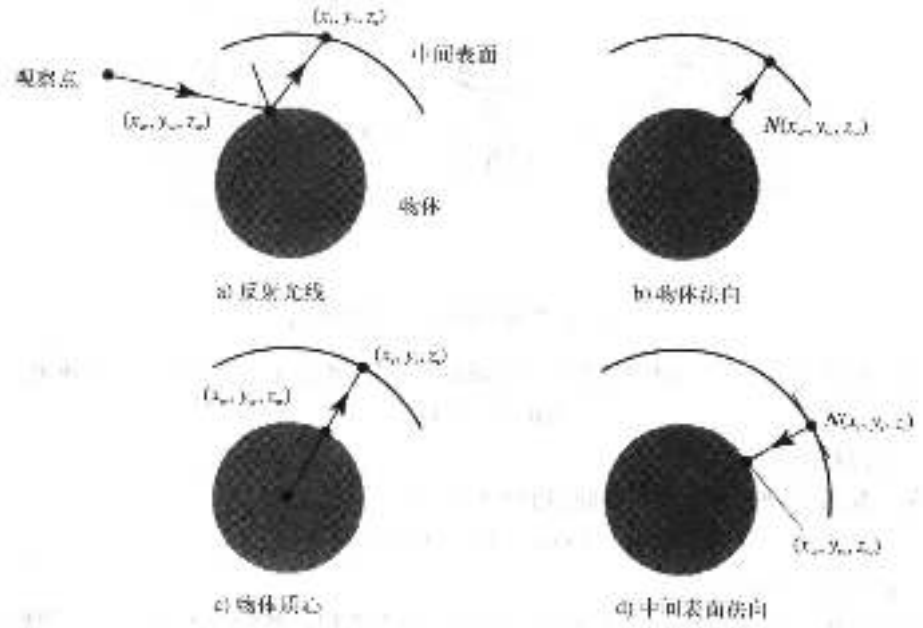


图8-5 将中间表面纹理T'映射到物体上的四种可能的O映射

1) 反向映射四个像素点到物体表面的四个点 (x_w, y_w, z_w) 上。

2) 应用O映射，求出圆柱体表面上的点 (θ, h)。在收缩的情况下，我们仅仅是把物体的点与圆柱体的中心相连接。这条线与圆柱体表面的交给出 (x_w, y_w, z_w)。

$$x_w, y_w, z_w \rightarrow (\theta, h)$$
$$= (\tan^{-1}(y_w / x_w), z_w)$$

231
↓
232

3) 应用S映射，求出相应于 (θ, h) 的点 (u, v)。

图8-7（彩色插图）展示的例子是把同样的纹理用不同的中间表面映射到一个物体上。中间物体是一个平面（等同于没有中间表面，这时纹理图是一个平面）、一个圆柱体和一个球。选择了一种简单形状的花瓶来展示每一种中间物体所产生的不同的扭曲。从这些图片中可以看出两点。第一，可以选择一种适合于物体形状的中间映射。例如，一个实心的旋转体最适合于圆柱体。第二，尽管这个方法本身并没有对物体的形状施加任何限制，但最终的可视效果可能并不令人满意。通常意义上的纹理并不涉及可能会产生很大几何扭曲的纹理图案。正是由于这个原因，许多实际的方法都是交互式的，并包括一些策略来处理这一问题。比如，先在二维空间对纹理图进行预扭曲，直到当其贴到物体上时能产生好的结果为止。

8.2 二维纹理域到双三次参数曲面片物体

如果物体是二次的或三次的，则对其进行表面参数化就是很直接的方法。前一节中，我们采用二次形状作为中间表面也正是由于这个原因。如果物体是一个双三次参数曲面片，则纹理映射是简单的。这是因为根据定义，参数曲面片已经在其表面的各处都具有了 (u, v) 值。

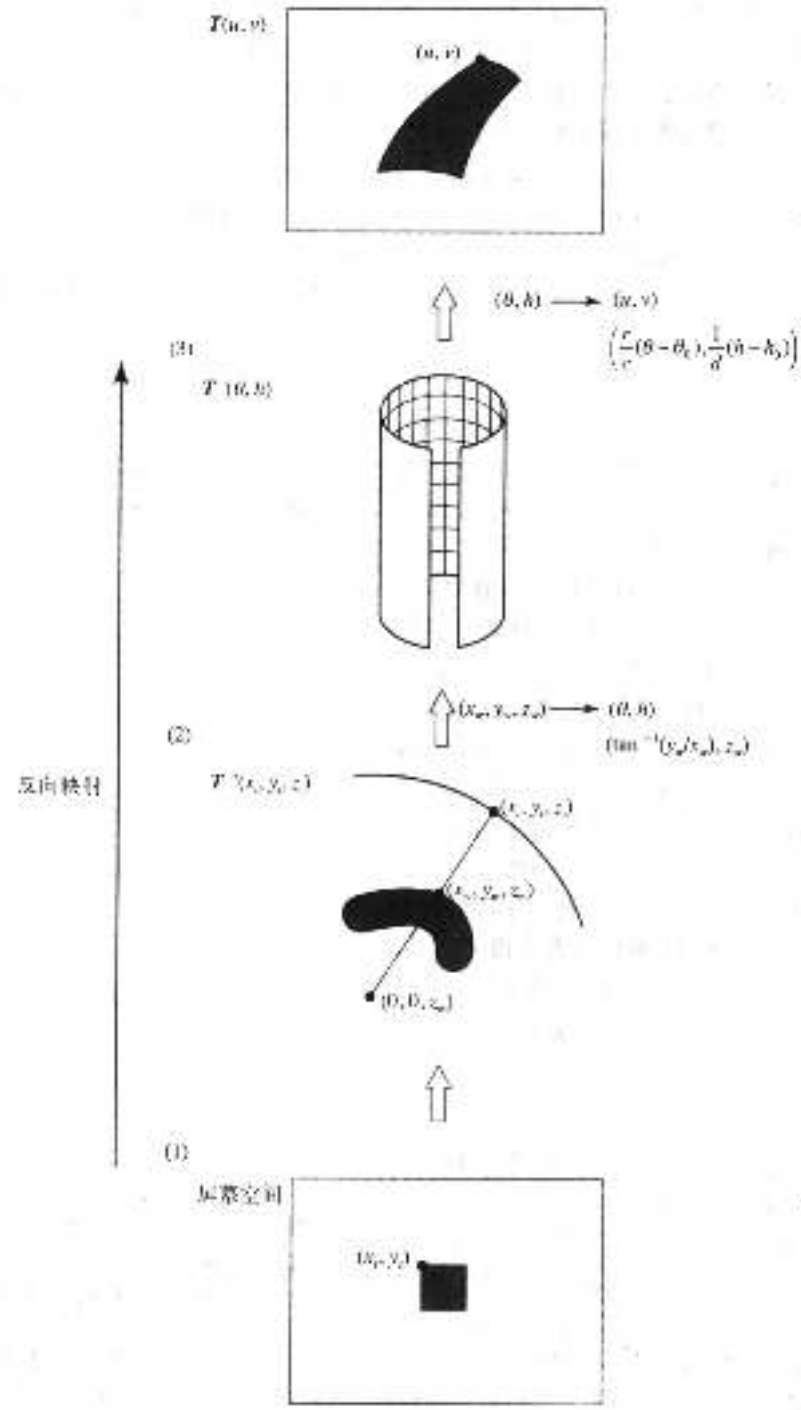


图8-6 用收缩方法的反向映射

在计算机图形学中首次采用纹理的是一种由Catmull提出的方法。这一技术应用于双三次参数曲面片模型，把物体空间的一个表面片进行划分，并同时在纹理空间执行相应的划分。其出发点是，对曲面片的划分一直持续到它仅覆盖一个像素点为止（标准的参数曲面片的绘

制方法已在第4章中论述）。当曲面片划分过程终止时，对于像素所需的纹理值可以从纹理域中当前划分水平下所封闭的区域上获得。这是一种直观的技术，并且可以作为对双三次曲面片绘制程序的扩展方法的一个变种，它把物体的表面划分成“微多边形”，并用来自纹理空间相应的细分水平的值为小平面进行明暗处理。

图8-8所示（彩色插图）是这一技术的一个例子。在这里，茶壶上的每一个曲面片都会引起对一个纹理图的划分，而每一个纹理图本身也是一个绘制的茶壶。对于每一个曲面片，来自参数空间划分的u，v值被用来索引纹理图，而纹理的u、v值也在0和1之间变化。可以将这一策略很方便地改造成把四个曲面片映射到整个纹理域，方法是在u，v映射时采用比例因子2。

233
|
234

8.3 广告牌

广告牌（bill board）是给一种技术起的名字。在这一技术中，把纹理图看成是三维的实体并置于场景中，而不是将其看成是在物体表面上控制颜色的图。这是一种简单的技术，它通过旋转图像的平面，使得它垂直于观察方向（观察方向是指从观察点到其所在位置的连线），而在三维空间中利用二维空间中的图像。这一技术的思想如图8-9所示。可能这一技术最通常的例子是一棵树的图像，这是一棵近似于圆柱体对称的树。这样的物体不可能实时绘制，而这一技术的可视化效果也是相当令人满意的，只是它的观察向量靠近了场景空间中的水平面。在二维投影中，物体原有的三维性质几乎是不可察觉的，这可能是因为我们并没有树的投影到底应该是什么样子的精确的内部概念。广告牌事实上是一个二维的物体，并按垂直于观察方向的角度绕y轴旋转，再被平移到场景中适当的位置。在广告牌中背景的纹理块设为透明。

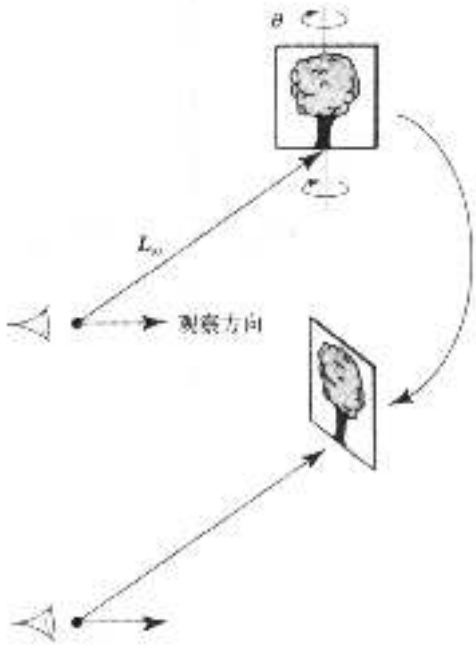


图8-9 假设观察方向近似地平行于地面，像树这类物体就可以被表示为一个广告牌，并绕y轴旋转，使其方向垂直于L_{oc}向量

对于广告牌的建模旋转由下式给出：

$$\theta = \pi - \cos^{-1}(\boldsymbol{L}_{oc} \cdot \boldsymbol{B}_o)$$

其中：

$\boldsymbol{B}_o$为广告牌的法向量，比如（0,0,1）；

$\boldsymbol{L}_{oc}$为从观察点到所需的广告牌在世界坐标系中的位置构成的观察方向向量。

已知θ以及所需的平移，我们可以构造一个对广告牌几何表示的建模变换，并对其实施这个变换。当然，这个简单的例子只有当观察方向平行于或近似地平行于地面的时候才有效。当这一条件不成立时，广告牌的二维性质就显露出来了。

广告牌是一种百充的特例，它基本上是把预先计算的纹理图用于在观察点只有微小的变化时绕过常规的绘制。这些细节将在第14章中进行阐述。

8.4 凹凸映射

凹凸映射是由Blinn（1978）提出的一种技术。它很简明，使得表面看起来像起皱的波纹，

而不必从几何上来对这些压力建模。实际上，是根据二维凹凸图中已知的信息对表面的法向在角度上进行了扰动，而这个扰动好像是一个局部反射模型，其中光强度主要是表面法向的一个函数。扰动（表面上看来）产生了光滑表面上的局部几何变化。凹凸映射的唯一问题是由于这些坑或压力在模型中并不存在，所以表面上看起来通过压力带的轮廓边不会产生所期望的截面。换句话说，轮廓边将按照模型原始的几何形状存在。

这是一种重要的技术，因为它看起来是把纹理涂到了表面上，而不是调整一个平整表面的颜色。图8-10（彩色插图）所示为采用这一技术的例子。

在绘制阶段使表面具有纹理，不进行几何的扰动，可以绕过可能会出现的严重建模问题。如果物体是多边形的，则网格必须足够小，以便接收来自纹理图的扰动，这对于原始建模阶段是一种非常过分的要求，尤其对于纹理是一种选项的情况。这样一来，该技术将称为凹凸图的二维高度场$B(u, v)$转换成对局部表面法向的适当扰动，而这个二维高度场代表某些希望的表面位移。当将这个表面法向用于明暗处理方程时，反射光计算发生了变化，就好像表面已经发生了位移。

考虑在一个（参数化的）表面上，相应于$B(u, v)$的一个点$\boldsymbol{P}(u, v)$，我们将点上的表面法向定义为: 235 ~ 236

$$\boldsymbol{N} = \frac{\partial \boldsymbol{P}}{\partial u} \times \frac{\partial \boldsymbol{P}}{\partial v}$$
$$= \boldsymbol{P}_u \times \boldsymbol{P}_v$$

其中，$\boldsymbol{P}_u$和$\boldsymbol{P}_v$为点$\boldsymbol{P}$处表面的切平面上的偏导数。我们想要做的是取得把点$\boldsymbol{P}$朝表面法向的方向移动到由$B(u, v)$所指定的一个点上，$B(u, v)$是一个一维的模拟量，如图8-11所示。也就是说:

$$\boldsymbol{P}'(u, v) = \boldsymbol{P}(u, v) + B(u, v)\boldsymbol{N}$$

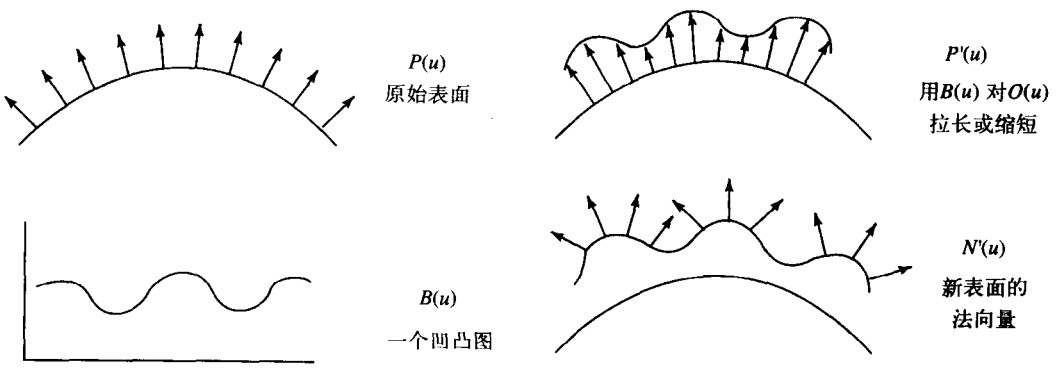


图8-11 在凹凸映射中涉及的各阶段的一个一维的例子（Blinn（1978））

从局部上来看，因为这个位移使得这时的表面不像之前那么光滑了，所以通过对这个方程进行微分得出“新”表面的法向量$\boldsymbol{N}'$:

$$\boldsymbol{N}' = \boldsymbol{P}'_u + \boldsymbol{P}'_v$$
$$\boldsymbol{P}'_u = \boldsymbol{P}_u + B_u\boldsymbol{N} + B(u, v)\boldsymbol{N}_u$$
$$\boldsymbol{P}'_v = \boldsymbol{P}_v + B_v\boldsymbol{N} + B(u, v)\boldsymbol{N}_v$$

237

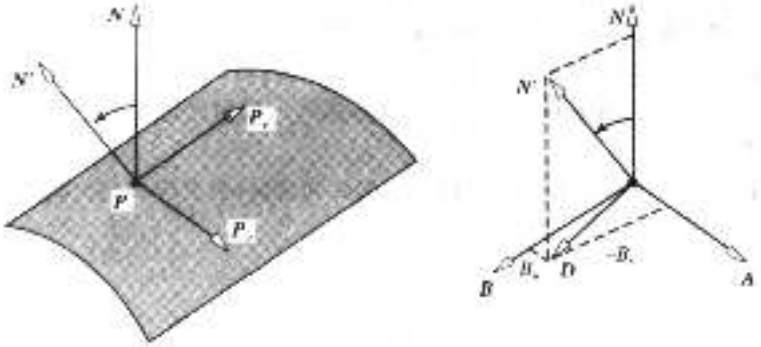
如果B很小，则可以忽略各个方程中的最后一项，于是：

$$N' = N + B_u N \times P_v + B_v P_u \times N$$

或者

$$\begin{aligned} N' &= N + B_u N \times P_v - B_v N \times P_u \\ &= N + (B_u A - B_v B) \\ &= N + D \end{aligned}$$

这时，D是一个位于切平面的向量，它把N拉向希望的方向，并可以由凹凸图的偏导数及切平面中的两个向量求出（见图8-12）。



P点的表面法向：
$N = P_u \times P_v$

由下式给出D：
$D = B_u A - B_v B$

图8-12 凹凸映射的几何解释

8.4.1 用于凹凸映射的多路技术

McReynolds and Blythe（1997）对于多边形网格物体定义了一种多路技术，这种技术可以利用标准的纹理映射硬件设备。为了达到这一目的，他们把计算分成如下的两个部分，最终的光强度值正比于$N' \cdot L$：

$$N' \cdot L = N \cdot L + D \cdot L$$

第一部分是常规的Gouraud分量，第二部分由两个图像投影的微分系数求出，可通过将带有高度场的表面作为常规的纹理图来绘制而形成这个投影。为了做到这一点，有必要把光线 238 向量变换到多边形的每一个顶点处的正切空间。这个空间由N、B和T来定义，其中：

N为顶点法向，

T为物体空间坐标系中u（或者v）增加的方向；

$B = N \times T$。

这些向量的单位化分量定义了把点变换到正切空间的矩阵：

$$L_{TV} = \begin{bmatrix} T_X & T_Y & T_Z & 0 \\ B_X & B_Y & B_Z & 0 \\ N_X & N_Y & N_Z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

算法如下：

1）用一个常规的带有纹理映射功能的绘制程序绘制物体。所用的纹理图是凹凸图或高度场。

2) 在每个顶点处求T和B，并把光线向量变换到正切空间。

3) 按相同的方法产生第二个图像，但是，这时在L_{TS}的X、Y分量方向上纹理/顶点的对应关系有一个小的移动。我们这时有两个图像投影，在这个投影上已经将高度场或凹凸图映射到了物体上，并相应于表面做了移动。如果减掉这两个图像，则得到$D \cdot L$项所需的微分系数（通过减法来求一个图像的微分系数是一种标准的图像处理技术，见Watt and Policarpo(1998)）。

4) 以常规的方式对物体进行绘制，这时不用任何纹理，并将这一成分加入到第3步计算的减数中求出最终的凹凸映射的图像。

这样一来，把直接的凹凸映射计算用两个纹理映射绘制来替代，即一个图像减法和一个Gouraud明暗处理，然后加入一个图像来获得最终的结果。

8.4.2 用于凹凸映射的预计算技术

也可以利用正切空间来实现预计算技术，Peercy等（1997）提出这一方法。它依赖于这样的事实，即在正切空间中被扰动的法向N'_{TS}只是表面和凹凸图本身的一个函数。Peercy等以三个预计算的系数来定义每个顶点上的法向。

可以看出，在正切空间被扰动的法向量由下式给出:

$$N'_{TS} = \frac{a,b,c}{(a^2+b^2+c^2)^{1/2}}$$

239

其中:

$$a = -B_u(B \cdot P_v)$$
$$b = -(B_v|P_u| - B_u(T \cdot P_v))$$
$$c = |P_u \times P_v|$$

对于凹凸图中的每一个点，都可以对这些点进行预计算。在绘制期间，存储了一个被干扰法向的图，而不是凹凸图。

8.5 光线图

光线图是对纹理图的一个很明显的扩展。它使得光照能够预先计算，并且可以作为二维的纹理图进行存储。我们在表面上对反射光采样，然后把所采样的内容存储在一个二维的图中。于是，明暗处理计算就降为对一个光线图的索引，或是对一个用光照调整过的纹理图的索引。这一技术的一个优点是对于预计算中所使用的绘制方法的复杂性没有限制，例如，可以使用辐射度方法或者任何独立于观察的全局照明方法来产生光线图。

从原理上来讲，光线图与环境图（见8.6节）是相似的。在环境映射中，我们在一个二维的图中将所有入射的照明高速缓存在场景中的一个点上。而用光线图将从场景中每一个表面反射的光高速缓存在一个二维图的集合中。

如果使用了一种精确的预计算方法，则可以期望这种技术能产生质量较高的明暗处理，而且这种技术比Gouraud插值方法要快。这意味着，可以把阴影结合到最后的绘制中。这一技术的明显缺陷是，对于移动的物体我们只能调用一种非常简单的光照环境，用无限远处的光源进行漫反射明暗处理。一种折中的方法是对移动的物体进行动态的明暗处理。并且假设，就明暗处理而言这些物体并不与用光线图明暗处理过的静态物体进行交互。

光线图既可以与纹理图分开存储，也可以用光线图对物体的纹理图进行预先的调整。如果将光线图作为独立的实体来保存的话，则可以把它以低于纹理图的分辨率来存储。因为除了在阴影的边缘之外，独立于观察的照明的变化比纹理的细节变化要慢。它还可以是高通滤波的，这将有助于改进效果，例如最终的图像中的光带。而且这还有助于使阴影的边界变得模糊（用于有界线分明的阴影产生的过程）。

如果物体要接受一种纹理，则可以在建模阶段调整纹理的亮度，以便使得它所具有的效
240 果就好像（未经调整的）纹理颜色被注入到了Phong明暗处理方程中。这称为表面存储（surface caching），这是因为它存储像素所需的最终的值，表面上的点将会投影到这个像素之上；还因为使用了纹理存储硬件来实施它。如果采用了这一策略，则纹理映射变换和将物体表面上的光线样本映射到一个光线图的变换应该是相同的。

光线图首先被用于二路光线跟踪中（见10.7节），它还被用于Ward（1994）RADIANCE绘制程序中。在这些应用中，其动机是把漫反射照明存储起来，并使得全局照明的执行时间处于合理的范围之内。这些方法最近在游戏引擎中的运用当然是为了促进实时明暗处理。

光线图遇到的第一个问题是如何在一个二维的数组中取样和存储计算出来的反射光，这些光来自三维空间中一个多边形的平面。实际上，这与纹理映射相反。对于纹理映射，我们需要的是一种从二维空间向三维物体空间的映射。另一个问题与经济有关。很明显，对于具有任意复杂性的场景，为每个多边形建立一个光线图是不经济的。相反，我们希望几个多边形共享一个光线图。

Zhukov等（1998）探讨了三维采样问题，即通过把多边形组织成称为polypack（聚合包装）的结构来解决问题。将多边形投影到世界坐标系平面，如果它们与坐标系平面的夹角不超过某一阈值则将其收集到polypack中（以便对多边形选择一个最大的投影平面）。而如果其夹角超出了这一阈值，则在投影中不重叠。将世界空间坐标平面划分成正方形的小单元，并反投到多边形上。多边形上的一个正方形小单元的图像是一个平行四边形（其较大的角小于等于102°）。这些平行四边形称为曲面片，是细分的多边形元素，对这些元素计算反射光。因此，这一方法对每一个多边形取样，而几乎是正方形的元素进行存储导致了光线图（见图8-13）。

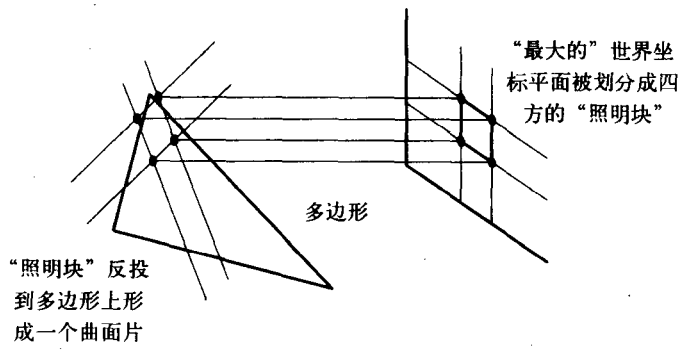


图8-13 在“最大的”世界坐标平面形成一个光线图

这些曲面片形成了对足可以产生光线图的场景的划分，可以采用任何应用所需要的算法（例如，采用Phong明暗处理方法或辐射度方法）来计算每一个曲面片的单个的光强度。在这
241 一阶段完成之后，对每一个多边形都有了一组（平行四边形状的）样本。然后，必须用每一个坐标平面中的最少的二维光线图来填充它们。Zhukov等（1998）用一种“首次击中渐减”

的算法来完成这一工作，这个算法首先用纹理数对每一个多边形组排序。

作者提到的另一个问题是相应于一个多边形的样本组必须被一些“沙状”纹理块（texel）所包围。而这些“沙状”纹理块是辅助性的，并不属于一个平面组。但是，当进行关于一个光线图的双线性插值时被用到，以便防止在多边形的边界处出现可见的人工的光照效果。这样一来，每一个纹理图都是由光线纹理块、沙状纹理块和未被占据的纹理块混合组成的。Zhukov等（1998）报告说，对于一个由24 000个三角形组成的场景（700个曲面片）产生256 × 256纹理块的14个光线图，具体划分是75%的光线纹理块、15%的沙状纹理块和10%的未被占据的纹理块。

对于那些由三角形构成的场景，而且我们已经对这些场景进行了顶点/纹理坐标的关联的情况，计算光线图的一种直接的方法是利用关联导出纹理空间向物体空间的仿射变换，然后用这个变换在一个三角形的平面上对光线采样。对纹理图空间采用此算法（通过扫描转换纹理空间的多边形投影），对每一个纹理块求出其在物体空间相应的点或投影，算法为:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

其中，(x, y, z) 是相应于物体上纹理块 (u, v) 的点。可将这一变换看成是三维空间中的一个线性变换，纹理图置于$z = 1$的平面中。通过按下式对U矩阵求逆:

$$\begin{bmatrix} x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \\ z_0 & z_1 & z_2 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} u_0 & u_1 & u_2 \\ v_0 & v_1 & v_2 \\ 1 & 1 & 1 \end{bmatrix}$$

从顶点/纹理坐标对应关系求出系数。将其写为:

$$X = AU$$

我们有:

$$A = XU^{-1}$$

假设三个点是非共线的，则逆矩阵U^{-1}保证存在。请注意，根据8.1节中的术语，这是一个从纹理空间到物体空间的前向映射。用这一技术照亮一个场景的例子如图8-14所示（彩色插图）。 242

8.6 环境映射或反射映射

环境映射一开始称为反射映射，由Blinn and Newell（1976）提出，Greene（1986）在一篇重要的文章中将其合并到主流的绘制技术中。环境图对于绘制那些反射其所处空间中的环境的闪光的物体是一种捷径。这一方法对于镜面反射几乎可以达到光线跟踪程序的质量，通过将由反射观察向量引起的问题降为索引一个二维图的问题来达到这一目的，而后一种做法与传统的纹理图没有什么区别。于是在光线跟踪程序中可能引起的处理成本问题就被调整为（离线地）构建图的问题了。从这一角度来看，它是典型的部分离线的或预计算的技术，正像在隐藏面消除算法中的预排序那样。如图18-8所示为一个场景及其相应的环境图的一个例子。

环境映射的缺点为:

- 只有当物体相对于所包含的环境来说较小时，这一方法（在几何学上）才是正确的。当我们没有在一个闪光的物体的弯曲表面上用“错误”的反射来扰动时，这个作用通常是不被注意的。如图18-9所示为此问题的一种扩展，该图展示了用光线跟踪和用环境映射的同一物体。

- 一个物体只能反射环境而不是反射其自身，所以，这一技术对于凹的物体是“错误的”。在图18-9中也可以看到这一点，图中在光线跟踪的图像中出现对壶嘴的反射。
- 对于要进行环境映射的场景中的每一个物体需要单独有一个图。
- 在一个通常的环境映射（球映射）中，无论何时需要进行观察点的改变时都需有一个新的图。

在这一节中，我们将考察三种环境映射方法，根据将三维环境信息映射到二维的方法对它们进行分类。这三种方法为立方体方法，纬度-经度方法和球映射方法（纬度-经度方法也是一种球映射方法，但是“球映射”术语现在已被应用于最新的技术）。一般原理如图8-15所示。图8-15a为传统的光线跟踪方法，用图8-15b的方法来代替传统方法，这一方法包括把反射的观察向量映射到一个二维的环境图中。我们按下式计算反射的观察向量（见1.3.5节）。

$$R_r = 2(N \cdot V)N - V \qquad (8\text{-}2)$$

图8-15c表明在实际应用中，对于一个像素我们应该考虑反射光束，而不是只考虑一个向量，而图中光束对着的区域需针对像素的值进行过滤。一个反射光束或者源自像素的四个角（当我们对每个像素索引图时），或者源自多边形的顶点（当我们采用一种快速（近似）的方法时）。需要指出的一个重点是，在环境图中相交的区域是在物体表面上投影像素区域曲率的一个函数。然而，因为我们现在使用的是纹理映射，所以可以采用预过滤的反走样方法（见8.8节）。

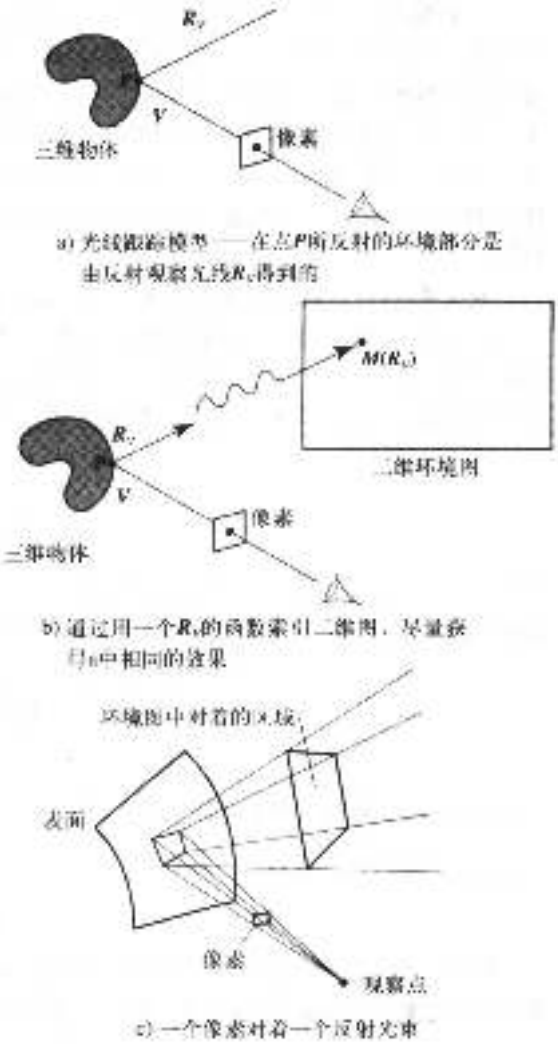


图8-15 环境映射

243
244

在实时多边形网格绘制中，可以仅计算顶点处反射的观察矢量并运用线性插值，就像传统的纹理映射中一样。因为我们期望看到结果图像中的精美细节，所以这一方法的质量与多边形的大小密切相关。

实际上，环境图存储环境中一个点上来自所有方向上的入射照明光，但是从场景中去掉将要接收此映射的物体。通过应用前面提到的几何近似方法，从这个入射照明光计算一个物体表面上的反射照明光。几何近似法指物体本身的尺寸可以认为趋近于该点，简单的BRDF是一个完全镜面项，即反射的观察向量。这就是一个独立于观察的预计算技术。

8.6.1 立方体映射

正如我们已经指出的，环境映射是一个两阶段的过程。作为预处理，它包含图的构造阶段，

立方体映射是很流行的，因为采用传统的绘制系统就可以容易地建立映射图。在实际应用中，环境图是6张图，它形成了一个立方体的表面（见图8-16）。一个环境图的例子如图18-8所示。观察点固定在物体的中心来接受环境图，绘制了六个视图。考虑将观察点固定在房间的中心。如果我们把房间看做是空的，则这些视图将包括四面墙、地板和天花板。立方体映射的问题之一是如果我们把反射光束看做是由像素的角点形成的，或者等价地看做是由多边形顶点处反射的观察向量形成的，则光束可以索引多个图（见图18-16）。在这种情况下，可将多边形细分，使得每一个小块都被限定到一个映射图中。

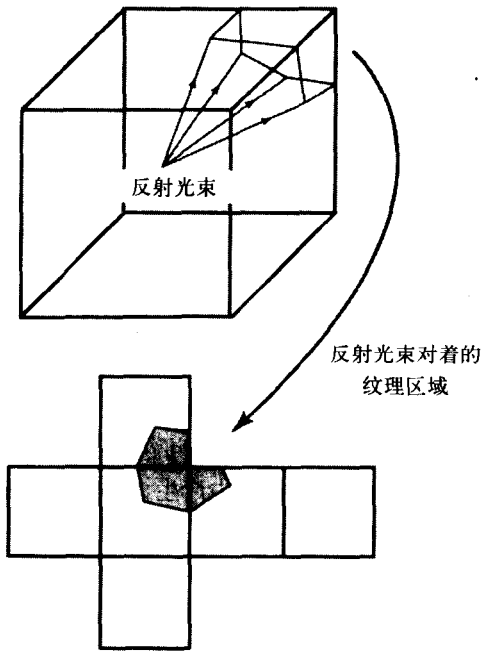


图8-16 立方体环境映射：反射光束可以覆盖多个映射图

对于立方体映射图，我们需要一种算法来确定从三维的观察向量向一个或更多的二维映射图的映射（而对于下一节中讨论的技术，这一映射算法被一种简单的计算所替代）。如果我们考虑反射的观察向量与环境图立方体处于相同的坐标系框架中（如果观察是通过把虚拟的或真实的摄像机沿着世界坐标轴在两个方向放置来建立的），则映射如下：

对于单个的反射向量：

1) 求出它相交的小平面——映射因数。这是一个将单位化的反射观察向量与以原点为中心的（单位）立方体的大小进行简单的比较。

2) 把这些分量映射到（u, v）坐标上。例如，与负的z轴的表面法向相交的点（x, y, z）由下式给出：

$$u = x + 0.5$$
$$v = -z + 0.5$$

规范如图8-17所示。

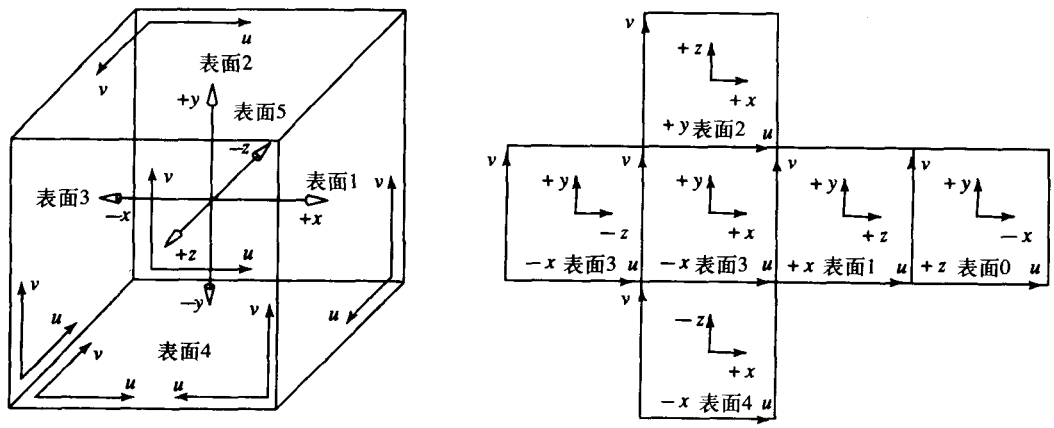


图8-17 立方体环境映射图规范

立方体环境映射方法（或者实际上对任何的环境映射方法）的应用之一是把一个动画了的计算机图形学物体与一个真实的环境相“混合”，这种应用在20世纪80年代很流行。在这种情况下，用一个真实环境的照片构建环境图，而（镜面的）计算机图形学物体可以被混合到场景中去，随着它的移动和反射周围的物体，看起来就好像它是场景中的一部分。

8.6.2 球映射

环境映射的第一次使用是由Blinn and Newell（1976）完成的，他们所采用的是球而不是立方体。环境图由一个纬度-经度投影构成，而反射观察向量R_v被映射到(u,v)坐标系：

$$u=\frac{1}{2}\left(1+\frac{1}{\pi}\tan^{-1}\left(\frac{R_{vy}}{R_{vx}}\right)\right) \quad -\pi<\tan^{-1}<\pi$$

$$v=\frac{R_{vz}+1}{2}$$

这一简单技术的主要问题是在极点处的奇点。在极点区域，对反射向量方向的很小变化就会在(u,v)坐标系中产生大的变化。随着$R_{vz}\to\pm1$，R_{vx}和$R_{vy}\to0$而R_{vy}/R_{vx}变为未知的。同样，随着$v\to0$或1，u的行为开始失去控制，引起表面上视觉的混乱。可以通过用$\sin\theta$（θ是极坐标中的仰角）调节映射图的水平分辨率来修正这种情况。

另一种球映射形式（Haeberli and Segal 1993; Miller等 1998）由一个圆的映射图构成，它是环境反射的正投影，正像在一个完全镜面的球的表面上所看到的那样（见图8-18）。很清楚，这样一种图可以由对观察平面的光线跟踪产生（另一种方法是取自一个闪光的球上的照片）。尽管映射图通过正投影存储了在参考点处的入射照明，它仍可以在过程的精确度范围之内产生一个常规的透视投影。

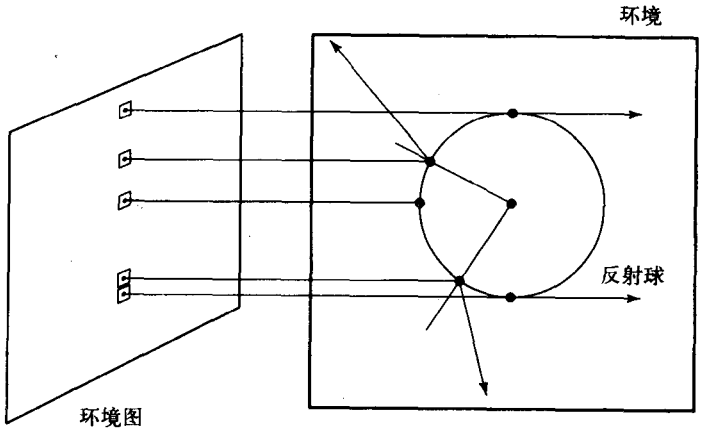


图8-18 通过从图上的纹理块到反射球跟踪光线构建一个球映射图

245 ~ 247

为了产生这个映射图，我们按下列过程进行。跟踪一束平行的光束——每一个纹理块(u,v)一个光束，并从球上反射每一束光线。在球上由来自(u,v)的光线击中的点是$\boldsymbol{P}$，其中：

$$\boldsymbol{P}_x=u \qquad \boldsymbol{P}_y=v$$

$$\boldsymbol{P}_z=(1.0-\boldsymbol{P}_x^2-\boldsymbol{P}_y^2)^{1/2}$$

这也是在击中点处球的法向，我们可以用方程（8-2）计算反射向量。

为了对映射图进行索引，我们从物体反射观察向量（对每一个像素或者对每一个多边形顶点）并计算映射图的坐标:

$$u = \frac{R_x}{m} + \frac{1}{2}$$

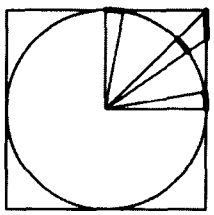
$$v = \frac{R_y}{m} + \frac{1}{2}$$

其中:

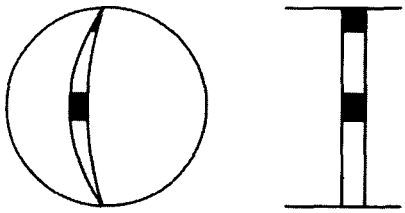
$$m = 2(R_x^2 + R_y^2 + (R_z + 1)^2)^{1/2}$$

8.6.3 环境映射：比较点

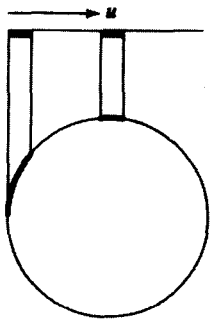
球映射克服了立方体映射的主要局限性，一般来说，立方体映射需要获取一些小平面映射图。当速度是重要因素时，球映射更受欢迎。但是，两类球映射都有比立方体映射更不均匀的采样问题。参考图8-19，这个图试图说明这一点。在所有这三种情况下，我们均认为环境映射图正在对单位球的表面上的入射光进行采样。图8-19显示了由环境映射图中的一个纹理块所采样的球表面区域的差别。只有当在绘制阶段的观察方向与所计算的映射图的观察方向相一致时才可以进行均匀的采样。由于这个原因，这类球映射被认为是依赖于观察的，当观察方向发生变化时，必须计算一个新的映射图。



a) 立方体透视图：角上的点与图中心（赤道和子午线）处的采样比较



b) 墨卡托投影或纬度－经度投影：在v方向（极点）上图的边缘处严重超采样



c) 正投影：在u方向（赤道处）图的边缘处严重欠采样

图8-19 在球的表面上采样

8.6.4 表面性质和环境映射

到目前为止，我们的讨论都限制在几何学上，并假设被进行环境映射处理的物体都具有完全镜面表面，而映射图也是以单个的反射光线来索引的。但是如果我们希望使用一种具有反射性质而不是具有完全镜面的物体的话，将会如何呢？在利用常规的Phong局部反射模型时，我们可以考虑两个分量，即漫反射分量和镜面反射分量，并构造两个映射图。漫反射图由所关心的点处的表面法向来索引，而镜面图则由反射观察向量索引。每一个图的相对分布由漫反射系数和镜面反射系数来确定，就像在标准的Phong明暗处理方法中那样处理。这使我们能绘制物体，就好像它们已经被Phong模型着了色。但是，加上了反射的环境细节，而这些细节也可以被模糊，以便模拟非光滑的表面（请注意：这一过程近似了一种效果，否则的话，要产生这种效果必须用分布式的光线跟踪方法进行绘制）。

这一技术是由Miller and Hoffman（1984）首次报道的（这一引用出自SIGGRAPH的课堂笔记。这些技术一般是无法利用的，尤其是其中较早期的技术。我们只对到目前为止就我们所知没有出现在其他刊物中的那些部分加以引用）。我们在这里给出的也是他们的规范。由处理环境映射图产生漫反射图和镜面反射图。这样一来，我们可以把过程看做是两个阶段的过程，第一阶段——环境映射图，由于场景中移去了物体，对一点进行光照编码。第二阶段对图进行过滤，以便把关于物体表面的信息进行编码。

Miller and Hoffman（1984）由下列的定义中产生了漫反射图:

$$D(N) = \frac{\sum_{L} I(L) \times Area(L) \times f_d(N \cdot L)}{4\pi}$$

其中:

N为所关心的点处的表面法向;

$I(L)$为以一个L的函数表示的环境映射图，L为相对应于图中I项的入射方向;

$Area$为与L相关的单位球表面上的区域;

f_d为漫反射卷曲函数:

$$f_d(x) = k_d\, x, x > 0 \quad f_d(x) = 0, x \leqslant 0$$

k_d为漫反射系数，为在对漫反射和镜面反射的贡献求和时$D(N)$的贡献的权重。

于是，对于N的每一个值，我们在所有的L值上对区域权重的点积或Lambert项求和。

镜面映射图定义为:

$$S(R) = \frac{\sum_{L} I(L) \times \text{Area}(L) \times f_s(R \cdot L)}{4\pi}$$

其中:

R为反射的观察向量;

f_s为镜面卷曲函数:

$$f_s(\text{x}) = k_s\, x^n \text{ for } x > 0 \quad f_s(x) = 0 \text{ for } x \leqslant 0$$

k_s为镜面反射系数。

（注意，如果取f_s为单位值，则表面是一个完全镜面，环境图不变。）

248 ≀ 250 于是，在一个表面点处的反射强度为:

$$D(N) + S(R)$$

8.7 三维纹理域技术

我们在前面的章节中已经看到，在将一个二维的纹理映射到一个三维物体的表面上时有很多困难。其原因是:

1) 基于一个表面坐标系的二维纹理映射在反映一个相应的表面上曲率的变化的压缩时可以产生很大的变化。

2) 对于向一个具有非平凡拓扑结构的物体表面上连续地进行纹理映射可能很快地变得非常难以操作。表面的元素可以是不同类型的，而且也可以以任何特定的方式连接在一起，因此，在表面上就很难保持纹理的连续性。

三维纹理映射巧妙地绕过了这些问题，因为需要为点赋的纹理值的唯一信息是该点在空间中的位置。为物体分配纹理只涉及计算一个在物体表面点上的三维纹理函数。这一技术的一个相当明显的需求是要程序化地产生三维纹理域。否则的话，对内存的需求会很大，尤其是采用三维mip映射时。而且，当我们只需要物体表面的纹理值时，构建一个完整的纹理的立方体域本来就是低效率的。

已知物体表面上的一个点 (x, y, z)，颜色定义为 $T(x, y, z)$，其中 T 是纹理域的值。也就是说，我们只是利用了恒等映射（可能与一个比例相关）：

$$u = x \quad v = y \quad w = z$$

其中：

(u, v, w) 是纹理域中的一个坐标。

可以将这看做是与实际上用一块材料雕刻出一个物体相类似。物体的颜色由其表面与纹理域的交来决定。这一方法由Perlin（1985）和Peachey（1985）同时提出，并同时杜撰出“实体纹理”这一术语。

这一技术的缺点是，尽管消除了映射的问题，但是纹理图案本身就被限制到你能够想象的程度。这与二维纹理映射相反，在二维纹理映射中，可以通过利用比如从电视摄影机上捕捉的帧图像建立起任何纹理。

8.7.1 三维噪声

一类流行的程序化的纹理技术的共同点是它们事实上都将一个三维的或空间的噪声函数作为建模的基元。这些技术可以产生对其现实的、自然的纹理效果的令人惊奇的变化，其中最值得一提的是对扰动的模拟。在这一小节中，我们将集中讨论基本基元——实体噪声的算法产生中涉及的问题。 251

Perlin（1985）首先建议应用噪声，它定义了一个噪声函数 *noise*()，这个函数将一个三维的位置作为其输入，并返回一个标量值。这被称为模型导向的合成，我们只在所关心的点处求噪声函数。理想情况下，此函数应具有下列三个性质：

1) 在旋转时统计不变性。

2) 在平移时统计不变性。

3) 频率限定在一个窄的带宽上。

前两个条件保证了噪声函数是可控的，也就是说无论在空间中如何移动和给噪声函数定向，都可以使它的总体外观保持相同。第三个条件使我们能够在不产生走样的情况下采样噪声函数。未充分采样的噪声函数可能不会在静态图像中产生显而易见的缺陷，但是如果用于动画应用程序中的话，不正确采样的噪声将会产生一种闪烁或起泡的效果。

Perlin产生噪声的方法是定义一个整数的格子，或者空间中的一组点，置于位置 (i, j, k) 处，其中 i，j 和 k 都是整数。格子中的每一个点都关联着一个随机数。可以通过一个简单的查询表的方法或者按Perlin（1985）所建议的那样，为了节省空间采用一个散列函数来完成这一工作。在与格子上的一个点共线处的空间中的点上，噪声函数的值只是一个随机数。而对于不在格子上的空间中的点 (u, v, w)，可以由从其相邻的格子上的点进行线性插值获得其噪声值。采用这种方法时，如果我们产生了一个实体的噪声函数 $T(u, v, w)$，则它将倾向于表现出方向（与轴对齐的）相关性。可以通过用三次插值调整这一现象，但是这样做代价太大，而

且相关性也还是可见的。另一种可以消除这一现象的噪声产生方法可以在Lewis（1989）中找到。然而，值得记住的是整个实体噪声函数是由表面采样的，而且要经历一个变换（例如，被调整后用于模拟扰动），这本身也足以消除相关性了。

8.7.2 模拟扰动

一个噪声可以被用于模拟大量的效果。到目前为止，其应用中最富于变化的形式是将其应用于所谓的扰动函数，如Perlin所定义的那样，它取一个位置x，返回一个扰动的标量值。
252 它是以级数形式书写的，其一维的形式可定义为：

$$turbulence(x) = \sum_{i=0}^{k} \text{abs}\left(\frac{noise(2^i x)}{2^i}\right)$$

求和在k处被截断，k值是满足下式的最小整数：

$$\frac{1}{2^{k+1}} < \text{一个像素的大小}$$

截断的范围限制了函数，保证了适当的反走样。考虑级数中前两项之间的差别，即$noise(x)$和$noise(2x)/2$。后面项中的噪声函数将比第一项的变化快两倍，即它有两倍的频率，而且它还具有是第一项大小的一半这样的性质。此外，其对扰动的最终值的贡献也是其一半。在每一个细节等级中加入到级数中噪声的量都正比于噪声的细节等级，反比于噪声的频率。这是自相似的，并且与通过不规则曲面片的细分而获得的自相似性相类似。只是这时细分并不是驱动位移，而是驱动噪声的八音变化，产生一种在一定等级范围内表现出相同的噪声行为的函数。关于为什么这个函数被证明是如此有用，我们最好从对信号分析的角度来看这个问题，这个分析告诉我们*turbulence*()的能谱遵循$1/f$次幂定律。因此，也不严格地近似于$1/f^2$布朗运动幂定律。

孤立的扰动函数只表现出一半的作用。然而，对扰动函数进行绘制直接导致一种不能以自然的形式来描述的一种齐次图案。这是由于这样的事实，即大多数以自然形态出现的纹理都含有一些非齐次的结构特征，所以不能单用扰动来模拟。例如，以大理石为例，它具有很容易区分的一些彩色岩脉穿过其中，这些彩色岩脉是于早期的地质时代在大理石固化之前由扰动形成的。根据这一事实，我们可以在模拟扰动的过程中确定两个不同的阶段，即：

1) 通过某些基本的函数形式表示纹理的基本的、一阶的结构特性。一般情况下，函数是连续的并在一阶导数中包含重要的变化。

2) 通过利用扰动扰动函数的参数加入二阶和更高阶的细节。

首先被Perlin描述的典型例子是一个正弦波的扰动，以给出大理石的外观。并没有经过扰动，大理石中的彩色岩脉是由一个正弦波通过一个彩色的图形成的。对于一个沿着x轴运动的正弦波，我们有：

$$marble(x) = marble\_colour\ (\sin(x))$$

彩色图*marble_colour*()把一个标量输入映射到光强度上，可视化这一表示式，图8-20a是
253 一个二维的大理石片段，它是用图8-20b中给出的彩色曲线绘制的。下一步，我们加入扰动：

$$marble(x) = marble\_colour(\sin(x) + turbulence(x))$$

以便得到图8-20c，这是大理石纹理的一种有说服力的模拟方法。图8-21（彩色插图）为三维效果图。



a) 未被扰动的图片，用图b中所示的曲线获得　b) 用于产生a的颜色曲线　c) 通过扰动图a中的图片获得的大理石片段

图8-20 模拟大理石

当然，扰动函数的运用不必严格地限制在只是对物体的颜色进行调节。任何影响物体外观的参数都可以被扰动。Oppenheimer（1986）扰动了一个锯齿函数，对树上的树皮进行凹凸映射，扰动还可以驱动像云这样的物体的透明性。云可以用一个不透明的图纹理映射到以地球为中心的一个球上来进行模拟。不透明的图可以用一种画笔程序来产生。云可以表示成带有模糊边界的白色团状物，这个物质慢慢消退成完全透明的。在经过纹理坐标的扰动之后，这些边界就变成了扰动的形式。Perlin（1989）对其早期的工作进行了扩展。他对空间的立体绘制区域进行扰动而不只是计算一个物体表面的纹理。实体纹理用于调节物体的几何形状及其外观。对定义物体的软区域的密度调节函数进行扰动，并用一个光线行进算法进行绘制。他介绍了很多的应用，其中包括腐蚀、火以及毛皮。

8.7.3 三维纹理和动画

可以简单地通过增加维数来代表时间，使扰动函数能够定义在一个时间和空间范围内，以便对整数的格子产生噪声。所以，格子上的点现在将由 (i, j, k, l) 索引，以便使我们能够将参数列表扩展到噪声 (x, t)，同样，对于扰动 (x, t) 也是如此。在这些过程的内部，对时间轴的处理与对三维的空间轴的处理并没有什么不同。

例如，如果我们想要模拟火，第一件事就是试着用函数来表示其基本形态，也就是说，要找一个“火焰的形状”。在这里，这种函数化的雕刻的完全特殊的性质是明显的。所决定的 254
最后的形状只是那些经过试验验证可以给出最好结果的形状。由于在最后一节的结尾处要讨论到三维立方体方法的扩展，所以我们将在两种空间中进行讨论。

一个火焰区域由矩形定义在xy平面。其最小最大坐标分别为 $(-b, 0)$，(b, h)。在这个区域中，火焰的颜色由下式给出：

$$flame(x) = (1 - y/h)\, flame\_colour\,(\mathrm{abs}(x/b))$$

其流程如图8-22（彩色插图）所示。*flame_colour*(x) 由三个分离的颜色曲线分量组成。它把一个标量值x映射到一个颜色向量上。每一条R、G、B曲线在$x = 0$处都有一个最大的强度值，这个值相应于火焰的中心，在$x = 1$处强度值消退为零。绿色曲线和蓝色曲线比红色曲线消退的快一些。*flame_colour*() 返回的颜色根据其距离火焰底部的高度加权得到沿着y的适当的变化。通过将*flame*() 应用于对一个覆盖了火焰定义区域的矩形区域进行涂颜色来绘制这个火焰。还通过利用一个相似的函数结构对多边形的不透明性加上了纹理。图8-22还表明通过引入扰

动函数获得了扰动的副本。于是：

$$flame(x,t) = (1-y/h)\,flame\_colour(\mathrm{abs}(x/b) + turbulence(x,t))$$

为了使火焰动起来，我们只是简单地绘制噪声的连续图片，这些图片垂直于时间轴而且用相应于火焰间隔的一个量来等间距地排放，就好像我们沿着时间轴在平移多边形。然而，只是按时间来平移还是不够的，火焰中可以察觉的细节虽然其形状是随着时间而变化的，但在空间中它还是令人惊奇地保持静态。这是因为与火焰相伴的是有一个综合性的方向，对流传热使这些细节成为向上的。通过在y轴上向下移动多边形，并按时间变化来模拟这一点，而且立即得到了较好的结果。如图8-23所示。最终的结构是：

$$flame(x,t) = (1-y/h)\,falme\_colour(\mathrm{abs}(x/b) + turbulence\ (x + (0, t\Delta y, 0), t)$$

其中，Δy是在y方向上每单位时间多边形相对噪声移动的距离。

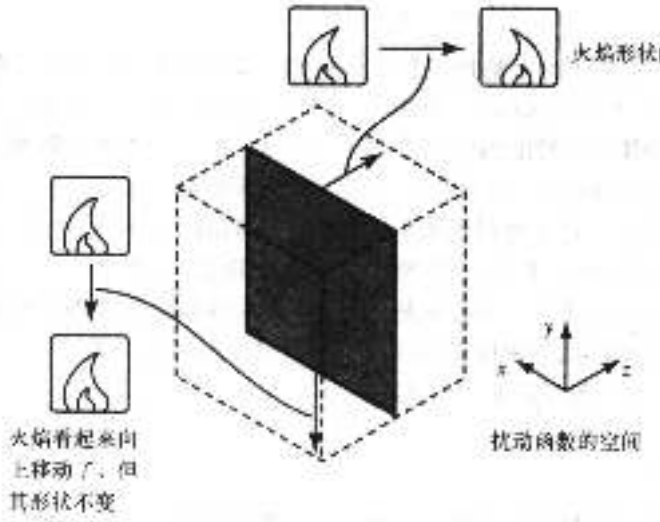


图8-23 对一个二维物体的动画扰动

8.7.4 三维光线图

从原理上来讲，我们没有原因不能有三维的光线图，而实际的限制是它可能需要非常巨大的存储空间。有时候，我们可以有一种保存场景中每一个点上的反射光的方法，我们采用任何独立于观察的绘制方法把计算出来的物体空间中的点(x,y,z)处的光强度分配给$T(x,y,z)$。这时对我们的预计算映射方法进行比较是有趣的。

对于环境映射，我们把物体空间的一个点上所有的入射光存储在二维映射图中，这个图用该点上入射光的方向进行标记。然后，用一个反射观察向量来查询指向用户的反射光。这些信息一般用于完全镜面的表面，并给出快速的依赖于观察的效果。

对于二维光线图，我们把场景中的每一个表面的反射光存储在一组二维图中，在绘制阶段依所采用的方法来索引这些图，以便对三维物体空间进行采样。我们利用这些样本去存储独立于观察的非动态光照。

对于三维光线图，我们把一个点上的反射光保存在一个代表物体空间的三维结构中。三维光线图是光线场的一个子集（见第16章）。

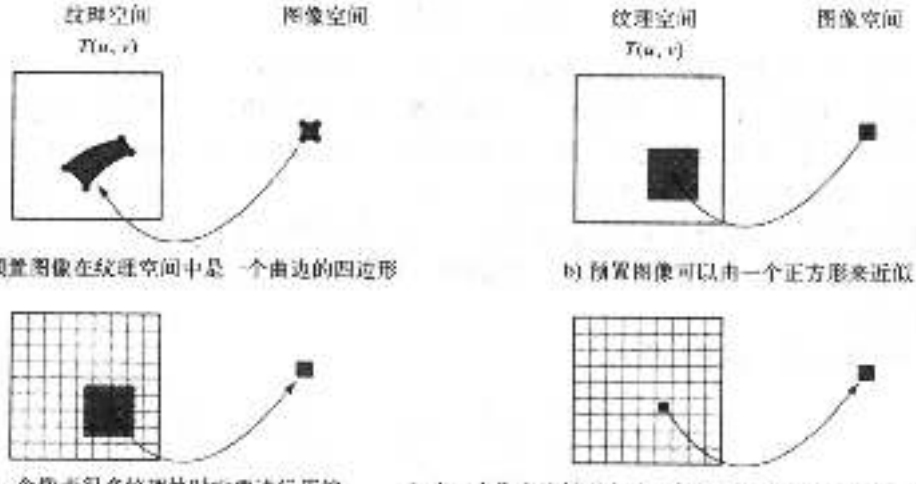
8.8 反走样和纹理映射

正如本章的引言中所讨论的那样，在纹理映射中，人工痕迹是非常严重的问题，而除非方法中集成了一个反走样的过程，否则大多数纹理过程都会产生可见的人工痕迹，尤其是在那些表现出连贯性或周期性的纹理中，一旦在纹理图案中主要的空间频率趋向于一个像素的大小，其缺陷是非常惹人注意的（这一效果的一个典型例子如图8-3所示）。由纹理映射产生的人工痕迹用一般的反走样方法（如超采样方法）不能很好地解决问题，正因为如此，标准的二维纹理映射过程一般都会结合一种专门的反走样技术。

纹理映射中的反走样是困难的，这是因为，为了很好地反走样，我们需要求出一个像素的预置图像，并对加权值$T(u, v)$求和，该加权值处于预置图像范围之内，以便得到对该像素的单个的纹理强度。但是，预置图像的形状随着像素的不同而变化，而这一过滤处理因此而变得非常昂贵，再一次来看图8-2，这个图表明，当我们考虑一个像素时，一般来讲其在纹理空间中的预置图像是一个曲边的四边形，这是由于纹理映射和透视投影都是非线性的变换产生的净效果，这个图还显示，对于对角线一带，除非运算进行了预先的定制或逼近，否则其纹理将出现错误的结果。尤其是，如果纹理图只是在像素中心的逆映射上采样的话，则当像素的逆图像尺寸足够小时采样的密度是正确的，但是总体来看还是错误的。

255 / 256

在图8-24a的情况下，反走样意味着对图中所示的集合的逼近。虽然从视觉上看是成功的，但是一个近似的方法忽略了形状而不忽略预置图像的大小或范围，并且预先计算所有所需的过滤操作。这就是由Williams（1983）创立的mip映射方法，这可能是一种专门用于纹理映射的最通用的反走样方法。他的方法是基于预计算和一个假设，即逆的像素图像合理地近似于一个正方形。图8-24b中展示了由一个正方形近似的像素的预置图像，正是由于有了这一近似，才使得反走样或者过滤运算可以预先进行计算。事实上还存在有两个问题，第一个问题更普遍一些，称为压缩或削减。当物体在屏幕空间变得很小，而像素在纹理空间却有一个大的预置图像时会出现这个问题。图8-24c即为这种情况，很多纹理元素（有时称为"纹理块"）需要映射到一个像素中。另一个问题称为放大。这时，物体非常近地靠近观察者，而只有物体的某个部分能够占据屏幕空间，导致像素的预置图像只占据小于一个纹理块的区域（见图8-24d）。mip映射方法可处理压缩问题，而对于放大问题，通常需要对mip映射方法更进一步精心地运用。



a) 像素的预置图像在纹理空间中是一个曲边的四边形　b) 预置图像可以由一个正方形来近似

c) 当映射到一个像素很多纹理块时需要进行压缩　d) 当一个像素映射到小于一个纹理块上时需要进行放大

图8-24 mip映射近似

257

在mip映射方法中，代之以纹理域包含一个图像的是，Williams用了很多图像，所有这些图像都是对原始图像均匀地、连续地降低分辨率得到的。换句话说，它们形成了一组预过滤的纹理图。在序列中，每一个图像都是它前一个图像分辨率的一半。对于这一思想的一种近似描述如图8-25所示。一个靠近观察者的物体和一个在屏幕空间中大的物体就从一个高分辨率的图中选择一个纹理块。用参数D选择一个近似的图。图8-26（彩色插图）为图8-8中所用的mip映射方法。

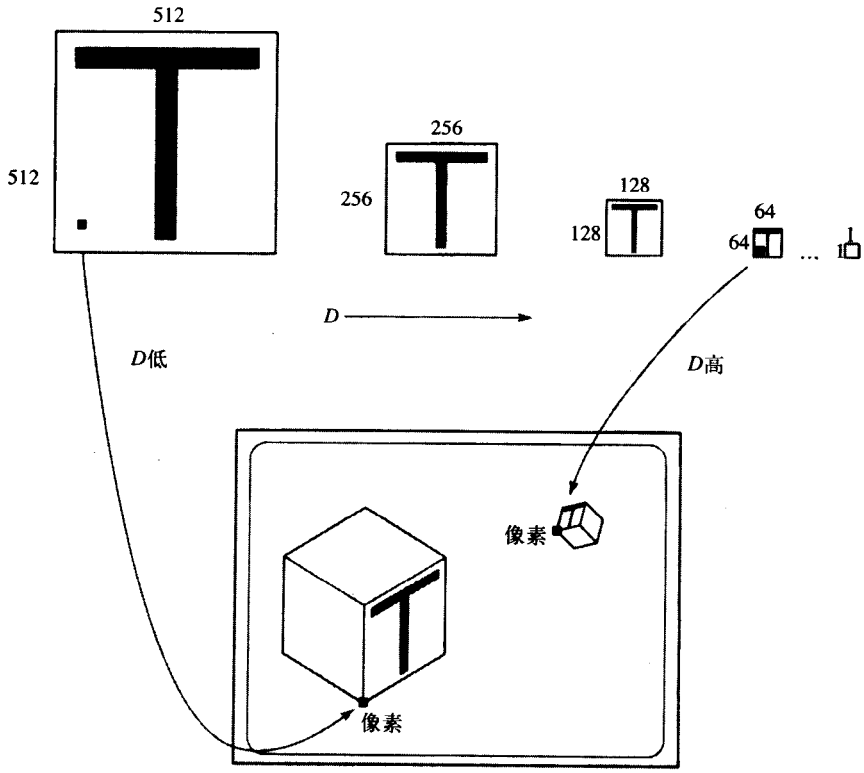


图8-25 mip映射方法的原理

在低分辨率的图像中，每一个纹理块都代表前面图像中一些纹理块的平均值。通过适当地选择D，就选择了一个具有适当分辨率的图像，而过滤的成本保持相同，这时就避免了很多纹理块映射到一个像素的成本问题。像素的中心被映射到由D确定的映射图中，并且使用这个单个值。原始的纹理就用这种方法得到了过滤，为了防止在分辨率发生变化的图像之间的不连续性的出现，还混合了不同的分辨率水平。在不同分辨率水平之间的混合，是在选择D时出现的。图像在分辨率上是不连续的，但是D是一个连续的参数。对两个最靠近的水平之间进行
258 线性插值。

Williams 由下式来选择D:

$$D = \max\_of\left(\left(\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2\right)^{1/2}, \left(\left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2\right)^{1/2}\right)$$

其中：∂u和∂v为纹理空间中预置图像原来的维，对于一个正方形的像素，$\partial x = \partial y = 1$。

对于D进行正确的或精确的估计是很重要的。如果D值太大，则图像看起来是模糊的。而D值太小，则仍然可以看到走样的人工痕迹。在Watt（1992）的文章中对根据映射的情况来确定D的详细的实际方法进行了论述。

从理论的角度来看，放大的问题是不存在的。在理想情况下，我们希望mip映射方法可以在任意细节的水平上使用。但是在实际应用中，存储的限制把最高的分辨率限定在如512 × 512个纹理块的范围。这个问题似乎并没有在文献中被提到过。Silicon Graphics为其图形工作站系列提供了下面的两个方法。他们给出了两个解答，第一个是对mip映射方法向高分辨率方向的简单的延伸，而另一个更简捷的过程是分别把纹理信息向低频部分和高频部分进行延伸。

外延由下式定义：

$$LOD(+1) = LOD(0) + (LOD(0) - LOD(-1))$$

其中，LOD（细节水平）代表的mip映射图如下：

$LOD(+1)$ 是外延的mip映射图。

$LOD(0)$ 存储最高分辨率的mip映射图。

$LOD(-1)$ 存储次高分辨率的mip映射图。

这一操作导出一个4 × 4像素块的外延的mip映射图，在这个区域之内该图没有变化。然而，放大处理保留了边界。

当将高频信息与低频的结构信息相关联时，也就是说，当高频信息代表纹理中的边时外延进展顺利。例如，考虑纹理图案是由字符块组成的情况，这时外延会模糊/放大字符的内部，而保持边界清晰。

当高频的信息不与低频的信息进行关联时，外延会引起模糊。纹理在整个范围内均匀变化的情况下（比如木头等）会出现这种情况。Silicon Graphics建议把低频信息和高频信息分离，并把高分辨率（无法存储的，比如2K × 2K）图转换成一个512 × 512的图，用其存储低分辨率或结构信息，而用256 × 256的图来存储高分辨率的细节。这种分离可以用经典的过滤技术精确获得。另一种空间域的过程如下：

1) 通过对原始的2K × 2K的图重新采样构建一个512 × 512的低频图。 259

2) 按如下方式构建256 × 256的细节面：

   i) 从含有代表性的高频纹理图的原始图中选择一个256 × 256的窗口。

   ii) 对这个窗口重新采样为64 × 64的窗口，并将其重新放大成256 × 256的窗口，产生原来的256 × 256图的一个模糊化的版本。

   iii) 从原图中减掉模糊的图，并加一个偏移，使减数图像为无符号的。这就导致了一个256 × 256的高频图。

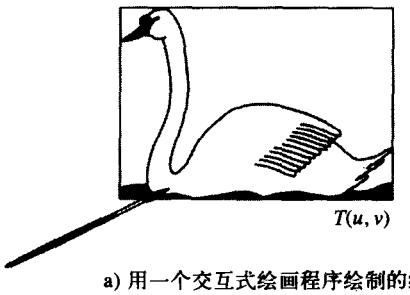
这时，当需要放大时，就使用512 × 512的低分辨率纹理和高分辨率的细节图的一种混合。

8.9 纹理映射中的交互式技术

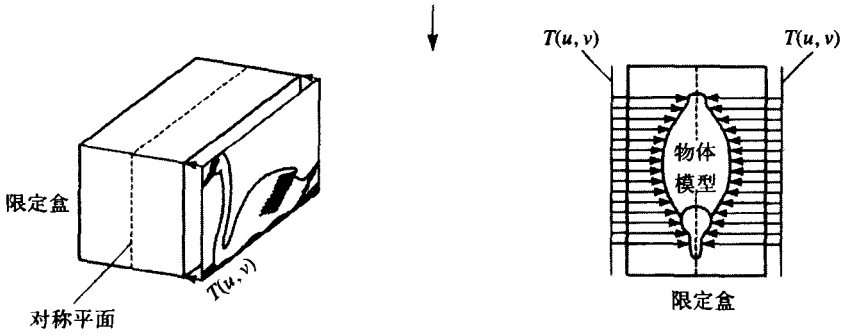
设计一个传统的二维纹理图的主要问题之一是，在所绘制物体上的结果的可视化。假如，一个艺术家或一个设计者正在二维的uv图像空间直接通过绘画来创建纹理图。我们知道当一张图被“贴”到物体上时，图的扭曲是物体的形状以及所采用的映射方法的函数。为了交互式地设计一个纹理图，艺术家需要看到最终绘制出来的物体，还需要有对映射机制的某些感知，以便他能够预测进行纹理映射时的改变效果。

现在，我们来介绍两个交互式技术。在第一种技术中，设计者在uv或纹理空间进行绘画。第二种技术试图让设计者认为他正直接在三维世界空间中向物体上绘画。

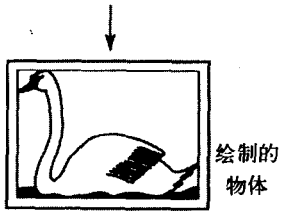
第一种技术非常简单，它被进一步发展成对于表现出一个对称平面的动物/物体加纹理。它只是两阶段纹理映射的一个交互式版本，即在两阶段纹理映射中以一个平面作为中间物体（见8.1.2节）。其总的思想如图8-27所示。将动物的模型包含在一个限定盒内。然后，将纹理图$T(u, v)$用盒子的最小最大坐标“贴”到限定盒的两个表面上，并用一个平行投影在投影机与对称平面垂直的方向把$T(u, v)$中的点投影到物体上。



a) 用一个交互式绘画程序绘制的纹理



b) 利用物体的限定盒，纹理图中的点被投影到物体上。所有的投影机都是互相平行的，并且与限定盒的表面垂直



c) 物体被绘制，可以看到“扭曲”，如果有必要，艺术家可以重复这一循环

图8-27 交互式纹理映射——在$T(u, v)$空间绘画

第二种技术允许艺术家直接与屏幕上的绘制结果进行交互。艺术家用一个模拟刷子的交互设备施加纹理（效果），屏幕上的效果就好像是画家正在直接对三维物体进行绘画。首先来看一看它与常规的二维绘图程序如何不同，就很容易看出这种技术的优点了，常规的二维绘图程序基本上是使用户能在屏幕上对所选择的点进行明暗处理。

例如，我们有一个球（在屏幕空间中是一个圆）。若用常规的绘画程序，比如说，如果我们选择了绿色，并用该色绘制了该球，则除非我们显式地改变了颜色，否则球的投影将被所

选的均匀的绿色填满。然而，在物体空间采用一种交互绘画的思想是这样的，当你应用绿色来绘画时，其颜色是根据对Phong明暗处理方程的应用而变化的。如果所绘的是发光的颜色，还会出现镜面高光。将这一思想扩展到纹理映射中去就意味着艺术家可以直接在物体上画上纹理，而这个程序还可以从物体导出纹理图，即常规的纹理映射过程的反向操作。一旦程序执行完成，还可以按正常的方式对物体的新的视图进行绘制和纹理映射。

这一过程需要一种技术，以便从所指向的那个屏幕像素标识出物体空间中相应的点。在Hanrahan and Haeberli（1990）所描述的方法中，采用了一个辅助的帧缓冲器，称为项缓冲器。用屏幕上光标的坐标来访问这个缓冲器就给出了指向物体表面上位置的指针，以及其纹理图的相应坐标值（u, v）。很明显，这时我们需要一种物体表示，使得表面上的每一处都是参数化的。Hanrahan and Haeberli（1990）将物体的表面分成了大量的微多边形。其总的思想如图8-28所示。

260
~
261

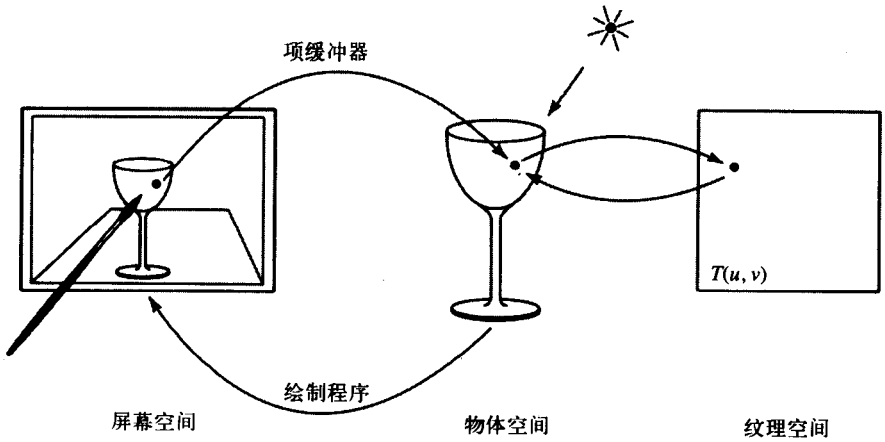


图8-28 交互式纹理映射——在物体空间绘画

262

第9章 几何阴影

9.1 计算机图形学中阴影的性质
9.2 地平面上的简单阴影
9.3 阴影算法

引言

这一章讨论“几何”阴影的问题，或是讨论算法来计算阴影区域的形状，但只是对其反射光的强度进行猜测。长期以来，在主流的绘制程序中都容忍有这种限制，其根本原因大概是有一个带有猜测出来的光强度的阴影总比根本没有阴影要好。

像纹理一样，阴影通常都是采用一种经验性的添加算法来处理的。它们也像纹理图一样被贴到场景中去。另一个与纹理图并行的地方是用来计算场景中每一个光源的图的最容易的算法称为阴影图。在绘制期间，访问阴影图正像纹理图被引用一样，以求出一个像素是否处于阴影中。像用于隐藏面消除的Z缓冲器算法一样，这个算法很容易执行，也已经成为一种伪标准。另外，它也像Z缓冲器算法一样以简单性来换取高的存储成本。

在场景中，阴影是重要的。一个没有阴影的场景看起来像人造的。阴影给出关于场景的线索，加强了物体之间的空间关系，并且给出有关光源位置的信息。为了完整地计算阴影，我们需要有关于形状以及内部光强度的信息。在阴影中一个场景的区域并不是完全没有光线。它只是没有直接的照明，而是从另一些邻近的物体接收间接的照明。因此，阴影的强度不可能把这些也计算进来，而这就意味着需要用像幅射度算法这样的全局照明模型。在这个算法中（见第11章），阴影区域的处理与场景中其他区域的处理没有什么两样，阴影的强度是一种光强度，像任何其他光强一样，它是从一个表面反射出来的。

阴影是一个光照环境的函数。它们的边界可以是明显划分的，也可以是模糊的，并且包含一个本影区和一个边缘区域。本影与边缘区的相对大小是光源的大小和形状以及它与物体之间距离的一个函数（见图9-1）。本影是阴影中完全与光源隔断的阴影部分，而边缘就是从光源接收一些光线的部分。边缘部分包围本影部分，而且在光强度上总有一个从边缘到本影的逐步的变化。如果在计算机图形学中不是对照明光源进行建模的话，则我们通常是考虑很远距离的点光源，并且假设在最简单的情况下物体产生带显明边界的本影。这也还只是一种近似。尽管如此，来自远距离的光线也几乎产生平行的光线，在物体的背后也还是有由于衍射而产生的光，而阴影也是逐渐消失的。这种效果还会在阴影所投射到的距离上产生变化。这种确定阴影质量的效果使我们能够推断出关于光源性质的信息，而且这些效果对于人类感觉三维环境也肯定是重要的。例如，我们在室外看到的阴影依赖于时间，以及天空是否下雨。

9.1 计算机图形学中阴影的性质

阴影的许多方面被用在一些现象的计算机生成中。它们是:

- 由于点光源的作用使来自多边形A的阴影落在多边形B上。这个阴影可以通过把多边形A投影到包含多边形B的平面上来计算。点光源的位置作为投影的中心。
- 如果观察点与（单个的）光源共线，则不能看到阴影。这种情况的一个等价的形式是可将阴影看成是被光源隐藏的区域。这就暗示着经修正的隐藏面消除算法可以用来解决阴影问题。
- 如果一个或多个光源是点源的话，则不必计算边缘区域，阴影具有硬边界。
- 对于静态的场景，阴影是固定的，并且不会随着观察点的改变而改变。如果物体和光源之间的相对位置发生变化，则必须重新计算阴影。这对于三维动画来讲是一个很沉重的负担，因为在三维动画中对于深度和运动的感知来说阴影是很重要的。

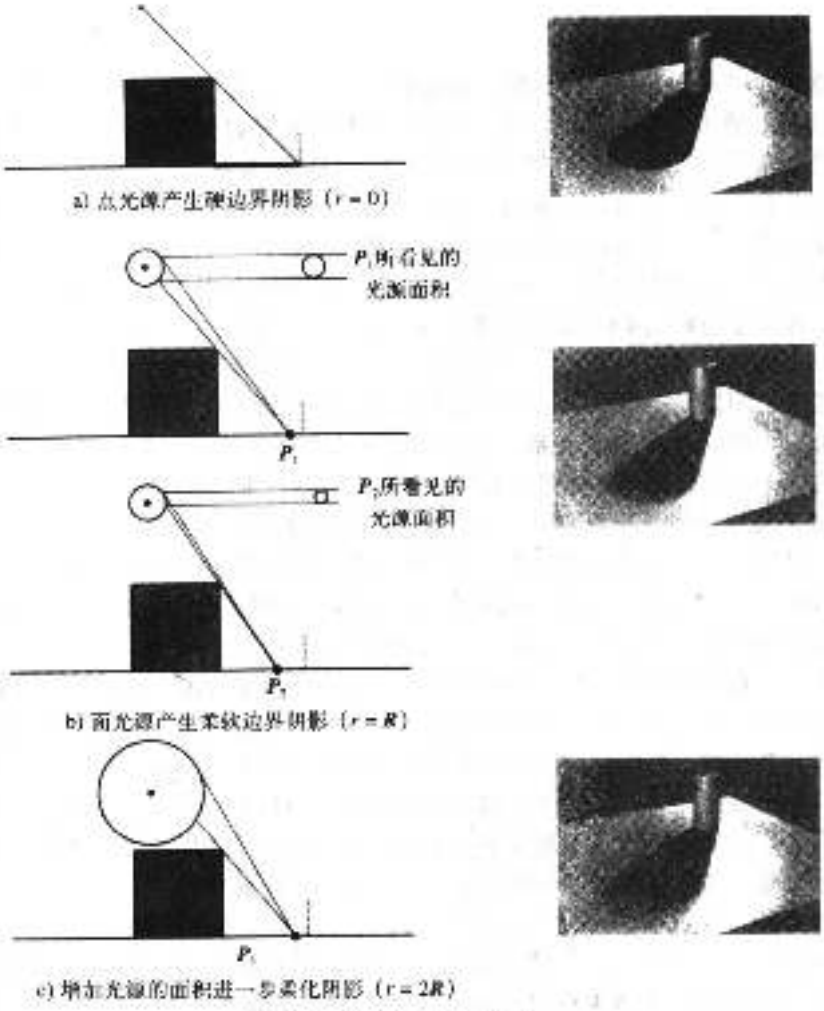


图9-1 球光源投射的阴影

由于有很沉重的计算负担，所以阴影一直被作为一种修饰与纹理映射一样加以考虑。它们被作为一种必需品来进行观察，并且与明暗处理算法进行比较，因为明暗处理算法很少考虑阴影的质量。大多数阴影生成算法都产生硬边界点光源阴影，而且大多数算法都只处理多边形网格模型。

9.2 地平面上的简单阴影

产生阴影的一种非常简单的方法是Blinn（1988）报告的。它用于满足在一个平整的地平面上单个物体的场景投射出阴影的情况。这个方法只是画出物体在地平面上的投影。因此，这一方法限定于单个物体的场景，或者用于虽然场景中有多个物体，但物体之间应足够孤立，以便互相之间不会投射阴影的场合。地平面投影可以很容易地通过一个线性变换获得，投影出来的多边形可以作为初始化过程中的一部分以适当的（黑色）光强度扫描到一个Z缓冲器中。

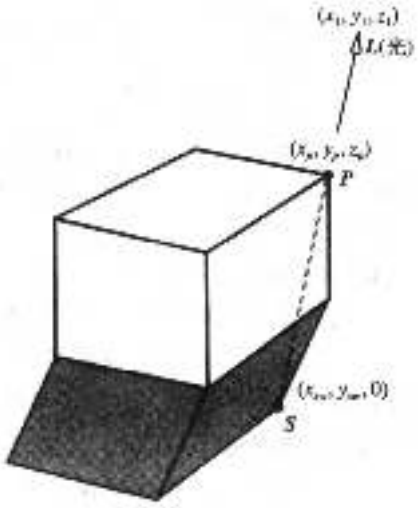


图9-2 对于单个物体的地平面阴影

如果进行了通常的对照明的近似，即单个点光源和光源距离无限远，则我们在$L = (x_l\ y_l\ z_l)$方向上有平行的光线，如图9-2所示。物体上的任何一点$P = (x_p, y_p, z_p)$都会在$S = (x_s, y_s, 0)$处投影。考虑图中的几何情况，有：

$$S = P - \alpha L$$

已知$z_s = 0$，则我们有：

$$0 = z_p - \alpha z_l$$

$$\alpha = z_p / z_l$$

以及：

$$x_s = x_p - (z_p/z_l)\, x_l$$

$$y_s = y_p - (z_p/z_l)\, y_l$$

以齐次方程表示为：

$$\begin{bmatrix} x_s \\ y_s \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -x_l/z_l & 0 \\ 0 & 1 & -y_l/z_l & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

注意到在这一公式中很容易在一个垂直的背面或侧面上产生阴影。Blinn还指出了如何把这一思想扩展到处理光源与物体之间的距离有限远时的情况。

这一类（在一个平整的地平面上）近似的阴影是传统的动画制作者所喜爱的，而且它的运用肯定也增强了三维计算机动画中的运动效果。

264 - 266

9.3 阴影算法

阴影算法与隐藏面消除算法不同，在隐藏面消除这一类算法中，有一两个算法目前占主导地位，而其他算法只用于一些特殊的场合。而在阴影算法中，目前还没有哪些算法流行起来作为顶级的阴影算法。事实上，阴影的计算在计算机图形学中是一个被忽略的领域。下面简短地介绍四种主要的方法。光线跟踪中阴影的产生在第12章中单独介绍。

9.3.1 阴影算法：投影多边形/扫描线

这个方法由Appel（1986）和Bouknight and kelley（1970）提出。将阴影加入到扫描线算

法中需要有一个预处理阶段，在这一阶段建立起一个辅助的数据结构，该结构把可能对给定多边形投射阴影的所有多边形联系起来。一个多边形加上它可能投射阴影的那个多边形构成阴影对，可以通过将所有的多边形阴影投影到一个以光源为中心的球上来检测。检测出没有相互关联的阴影对并将其删除，这是很重要的一个步骤，因为对于一个含有n个多边形的场景来说，可能的投影阴影的个数为$n(n-1)$。

这一算法把辅助数据结构与一个常规的扫描转换过程一起处理，以便决定对于当时产生可见的扫描线的多边形上是否有任何阴影。如果没有阴影多边形，则扫描线算法按正常情况继续。对于当前的多边形：如果有一个阴影多边形存在，则将光源作为一个投影中心，通过向含有当前多边形的平面投影产生阴影，然后，与一个确定当前像素是否处于阴影中的处理过程同时处理常规的扫描线转换。这时出现三种可能的情况：

1) 阴影多边形不覆盖所产生的扫描线片段，这种情况与没有阴影的算法是相同的。

2) 阴影多边形完全覆盖可见的扫描线片段，这时扫描转换继续进行，但是对像素的光强度进行一定量的调整，这个调整的量依赖于覆盖着该片段的阴影数量。对于单个光源的情况，该片段或者是处于阴影中，或者不是。

3) 阴影多边形部分地覆盖可见的扫描线片段。在这种情况下，将片段分离，处理过程迭代地进行，直到得到一个结果为止。

这三种可能的情况如图9-3所示。沿扫描线的顺序，它们是：

a) 多边形A是可见的，因此对它进行绘制。

267

b) 多边形B是可见的，并进行绘制。

c) 多边形B被多边形A遮挡，并按适当的光强度被绘制。

d) 多边形B是可见的，并被绘制。

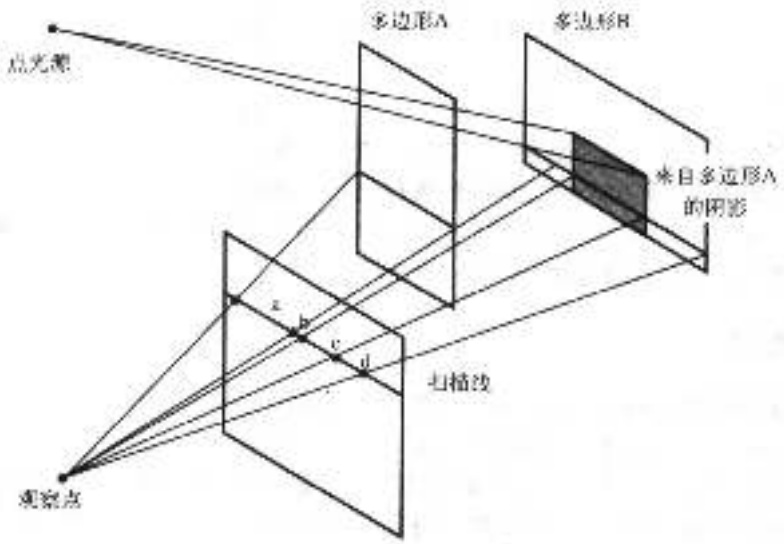
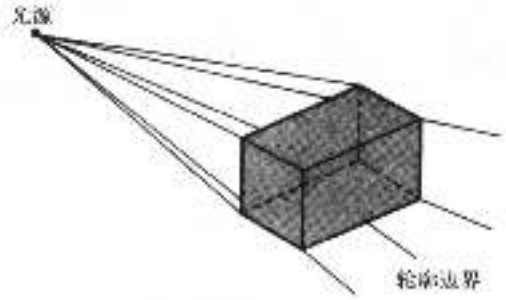


图9-3 从另一个多边形处接收阴影的多边形在辅助数据结构中被连接起来。这时的扫描线片段由观察点投影边界和阴影边界刻画

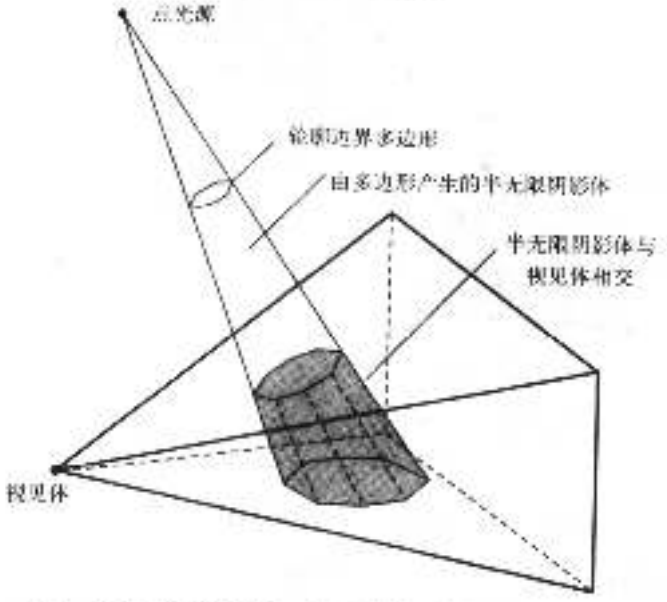
9.3.2 阴影算法：阴影体

阴影体算法开始是由Crow（1977）提出来的，后来又被其他人进行了扩充。尤其是Brotman and Badler（1984）将这种思想用作产生“柔软”阴影的基础。也就是说，阴影是由一个分布式的光源产生的。

阴影体是一个其空间是不可见的体积，由物体的阴影所扫掠。它是由从点光源穿过物体上的顶点所发射出的线定义的无限的体积。图9-4表示了阴影体的概念。有限的阴影体是通过把无限的阴影体与视见体相交而得到的。通过首先计算物体的边界或轮廓边界，就如从光源处看到的那样，来得到阴影体。图9-4a是一个简单物体的轮廓边界。物体的轮廓边界是由属于物体的一个或多个相连的多边形形成的边界。轮廓边界把可以从光源接收光线的那些多边形与不能从光源接收光线的多边形分离开来。



a）一个物体的轮廓边界



b）由一个轮廓边界多边形、一个点光源和一个视见体所定义的有限阴影体

图9-4 形成阴影体的示意图

由光源和轮廓边界所定义的多边形定义了阴影体的边界表面，如图9-4b所示。于是，每一个与点光源相连的物体产生一个阴影体的物体。这个物体由一组阴影多边形组成。请注意：

这些阴影多边形是“不可见的”，也不应与下一小节中我们将介绍的可见的阴影多边形相混淆。这些阴影多边形本身用于确定阴影，但是它们并不被绘制。

这一方法可以结合到一些隐藏面消除的算法中，而且定义阴影体的多边形也可以像对物体多边形一样进行处理，只是要将它们看成是不可见的。对“前向的”多边形和“背向的”多边形之间有一个区分，以这种方式标识的阴影多边形和物体多边形之间的关系被测试。如果物体上的一个点位于前向阴影多边形之后并位于背向多边形之前，则它在阴影中消失。也就是说，如果它被包含在一个阴影体之内的话，它就会消失，这样一来，前向的阴影多边形把任何在它后面的事物放在阴影中，而背向的阴影多边形删掉了前向多边形的影响。

268 ~ 269

正如它所存在的那样，这一算法最容易被集成到深度优先的隐藏面消除算法中。现在考虑对一个特定像素的算法的操作。考虑一个向量或者光线从观察点穿过像素，并在这个向量方向上观察真实多边形与阴影多边形之间的关系。对于像素来讲，保持对像素的计数，如果观察点已经处于阴影中了，则计数为1，否则计数为0。如果我们在按深度排序的一个多边形列表中进行降序处理，则当通过一个前向的多边形时，此计数增加，而当通过一个背向的多边形时，此计数减少。当碰到一个真实的多边形时，这个计数告诉我们是否处于阴影体内部。这一程序如图9-5所示。

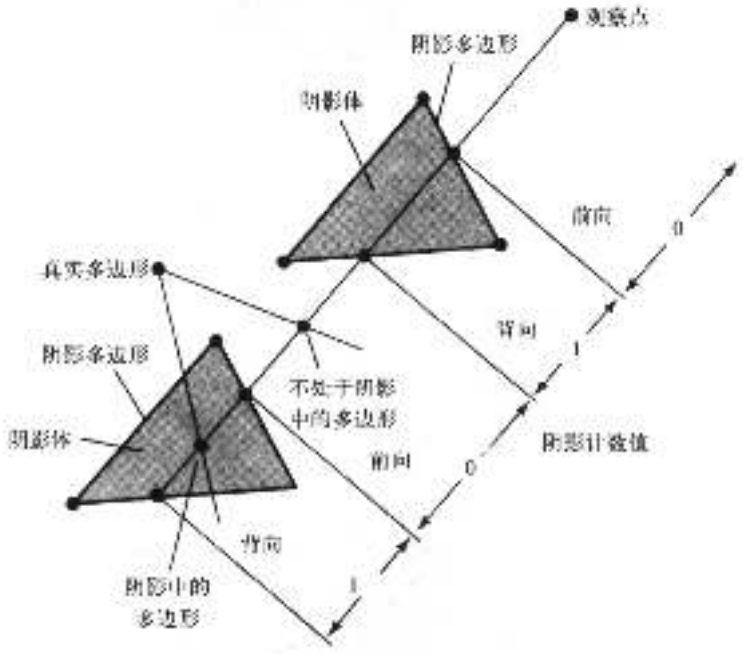


图9-5 前向和背向阴影多边形以及阴影计数值

Brotman and Badler（1984）采用了一种增强的Z缓冲器算法。他们的算法具有以下两个明显的优点：

1）保留了Z缓冲器绘制方法的优势。

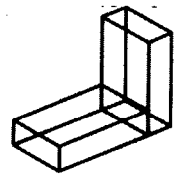
2）能够计算柔软的阴影或者可以产生本影/边缘区的效果。

把阴影体方法与Z缓冲器方法一起使用所付出的代价是存储成本的增加。必须对Z缓冲器进行扩展，使每一个像素的位置都是一个5个属性域的记录。随着对阴影多边形的“绘制”，

会对像素记录中的计数进行修改，这样就可以确定一点是否是处于阴影中了。

软阴影是通过将分布的光源模拟成点光源阵列，并对每一个点光源进行线性加的计算来实现的。

最初的阴影体算法对数据库的环境有很严格的限定。其中，最严格的限制是物体必须是凸多边形的。Bergeron（1986）开发了一个Crow算法的通用版，这一版本克服了这些限制，允许物体为凹状多边形或穿透式多边形。

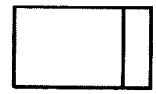


a) 在建模坐标系中的简单多边形物体

270

9.3.3 阴影算法：从光源变换导出阴影多边形

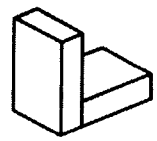
这一方法是Atherton等（1978）提出的，它基于这样一种事实，即把隐藏面消除算法应用于光源进行观察会产生处于阴影内部的多边形或多边形的某些部分。这一方法还依赖于由Weiler and Atherton（1977）提出的物体空间多边形的裁剪算法（以产生是现有多边形的组成部分的阴影多边形）。



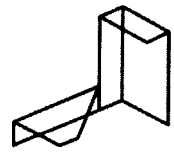
b) 平面观察，显示光源位置

这一方法宣称的优点是它是在物体空间操作的。这意味着它可以从算法中抽取出大量关于阴影的信息。例如，这一优点可以在架构CAD中找到应用。

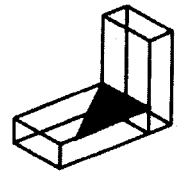
这一算法把物体的数据结构用阴影多边形进行增强，以产生一个完整的阴影数据文件。然后，可将这个文件用于产生任何对具有阴影的物体的观察。因此，它在产生动画序列时是一种好的方法，因为在动画序列中，虚拟摄像机的位置发生变化时，物体和光源的相对位置也同时发生变化。对于一个简单的例子运行这一算法的过程如图9-6所示。为了清楚起见，这里采用了一个阴影多边形。在图9-6中，算法的第一步是应用一个变换，使得物体或场景被从光源的位置进行观察。然后，利用隐藏面消除算法产生可见的多边形，也就是说，这些多边形对于光源来说是可见的，因此也就不在阴影中。正如示意图中所暗示的，这些多边形可以是完整的，也可以是经过裁剪的。然后，可以把这个多边形的集合与原物体的多边形结合。假设两个数据集合都处于同一个坐标系的话。对这些数据集合进行结合的过程导致一个完整的阴影数据文件，即对特定的点光源根据阴影多边形加强的原始多边形集合。将该数据库变换到所需的观察点上并应用隐藏面消除算法，将导致一个具有阴影的图像。这一算法利用了这样的事实，即阴影多边形是与观察点相独立的。实际上，该场景对于隐藏面消除被处理了两次。一次是将光源作为观察点，产生阴影多边形；另一次是用常规的隐藏面消除（从任一观察点）。



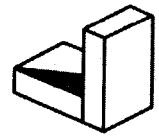
c) 把光源作为观察点进行隐藏面消除



d) 由c变换回建模坐标系后的可见多边形



e) a和d合并产生含有阴影多边形的数据库



f) e部分可以产生对带有阴影物体的任意观察

图9-6 由变换导出阴影多边形

9.3.4 阴影算法：阴影Z缓冲器

对于阴影计算，最简单的方法并且最容易集成到基于Z缓冲器

的绘制程序中去的方法可能就是阴影Z缓冲器算法了，这一方法由Williams（1978）提出。这一技术对于每一个光源需要一个单独的阴影Z缓冲器，它的基本形式只适用于由单个光源照明的场景。另一种选择是可以将单个的阴影Z缓冲器用于多个光源，对每一个光源执行一次算法，但这样一来有些效率不高而且慢。

该算法是一个两阶段的过程。绘制一个场景，并且以光源作为观察点在阴影Z缓冲器中存储深度信息。不计算光强度。这一方法计算对于光源可见的那些多边形的深度图像。

第二步是用Z缓冲器算法绘制场景。这一过程可按下列方法进行增强：如果点是可见的，则用一个坐标变换，把三维屏幕空间中的坐标（x, y, z）（即从观察点）映射至屏幕空间中的点的坐标（x', y', z'），即作为坐标原点的光源点。把（x', y'）用于索引阴影Z缓冲器，而相应的深度值与z'进行比较。如果z'大于该点在阴影Z缓冲器中所存储的值，则有一个表面比当前所考虑的点光源更近，于是该点处于阴影中。这样就利用了阴影的“强度”，否则的话，该点就按正常情况进行绘制。阴影图的一个例子如图18-8所示。注意，在这个特定的例子中，我们产生了六个阴影图。这使我们能够从场景中任何合适位置上的观察点来观察一个房间。

除了针对Z缓冲器隐藏面消除算法对存储的高需求进行了扩展之外，该算法还针对其效率不高进行了扩展。对于那些可能在接下来的计算中被“覆盖”的表面执行的阴影计算（进行了改进）正像对明暗处理计算那样。

反走样和阴影Z缓冲器

与Z缓冲器算法相同，阴影Z缓冲器算法由于进行点的采样而可能受到走样现象的影响。有两种出现走样的可能性。第一种是，在阴影Z缓冲器的建立阶段，由于直接的点采样而产生人造痕迹。这些人造痕迹在阴影的边界处是可见的，这里指的是通过一个点光源而产生的有鲜明边界的阴影投影。第二种走样的问题是在访问阴影Z缓冲器时产生的。这与纹理映射时产生的采样问题有些相似。这个问题是因为我们在高效地将一个像素投影到阴影Z缓冲器上而造成的。这一过程的流程如图9-7所示。如果我们考虑的是在阴影Z缓冲器图中的一个正方形像素的所谓预置图像，则一般来讲，这将是一个四边形，它包围一些阴影Z缓冲器中的像素。我们必须处理的正是这个从多个图中的像素到一个屏幕像素问题。它意味着，一个像素可能是部分位于阴影中，另一部分不在阴影中。如果我们要做一个二元决定的话，则会出现走样。因此，我们通过由阴影Z缓冲器计算这一点来考虑处于阴影内部的那一部分像素。这个分数可以通过在阴影Z缓冲器像素的集合上对Z值的比较来计算，这些像素是屏幕像素会投影到的像素。然后，再利用这一分数给出一个适当的阴影强度。过程总结如下：

1) 对于每一个像素计算相应于四个角点的（x', y'）值。这一计算定义了在阴影Z缓冲器空间中的一个四边形。

271 ~ 273

2) 通过比较屏幕像素的z值与在阴影Z缓冲器四边形中的每一个z'值，对这个四边形中的信息进行集成。这一比较给出一个反映处于阴影中的像素区域的一个分数。

3) 利用这个分数给出一个适当的逐渐变小的强度。这一动作的可视效果就是对于那些跨在阴影边界上的像素，其阴影的鲜明边界将被柔化。

在Reeves等（1987）的文章中，对这一方法有非常详尽的介绍。对于反走样所付出的代价是在处理时间上有很大的增加。不能使用预过滤技术（见第14章），上述文章中提出了在阴影Z缓冲器图的像素预置图像中集成一个随机采样的方法。

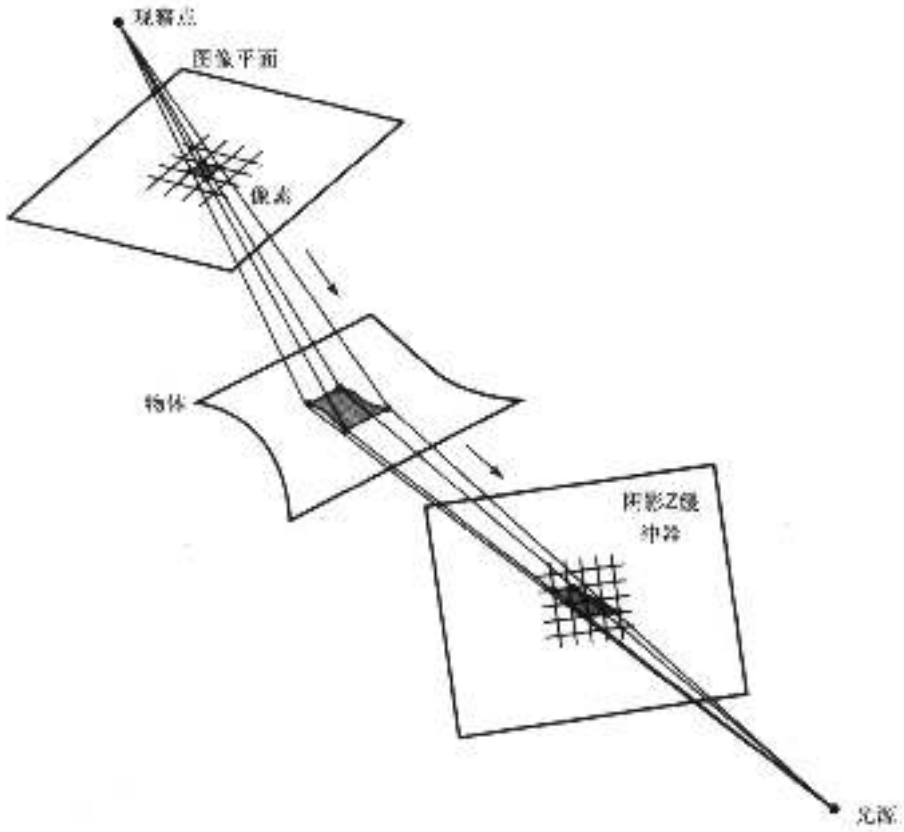


图9-7 在阴影Z缓冲器中一个像素的预置图像

274

第10章 全局照明

10.1 全局照明模型
10.2 全局照明算法的发展
10.3 已建立的算法——光线跟踪和辐射度
10.4 全局照明中的蒙特卡罗技术
10.5 路径跟踪
10.6 分布式光线跟踪
10.7 二路光线跟踪
10.8 依赖于观察/独立于观察和多路方法
10.9 存储照明
10.10 光线体
10.11 粒子跟踪和密度估计

引言

在计算机图形学中，全局照明是用来指一类模型，这类模型通过估算从点x处反射的光线，并考虑所有到达该点的照明而绘制对一个场景的观察。也就是说，我们不仅要考虑直接从光源到达该点的照明，还要考虑可能穿过其他物体的来自光源的间接照明。

在一般的追求照片真实性的情况下，大多数的研究都致力于解决全局照明的问题。尽管如我们在第7章中已经了解到的那样，有大量关于局部反射模型的研究工作也在平行地进行着，人们一直致力于解决模拟光线与整个环境的相互作用的困难问题。必须在环境中从发射器到感应器之间跟踪光线，而不是只跟踪从发射器到一个表面然后直接到达感应器或者到达眼睛。因此，这样一种方法并不需要关于阴影的附加算法，因为阴影只是由于一个附近的物体的靠近减少了照明的水平而产生的一个区域。其他的全局照明效果，比如说物体之间的相互反射以及透明的效果也可以被正确地建模。

现在，关于全局照明对于照片真实性是如何重要并不是很清楚。但可以肯定的是，我们现在已经熟悉了“封闭的”人造环境的情况，在这种环境中有很多全局交互。但是，为了获得大多数计算机图形学应用程序可以接受的真实程度，必须对这种交互性进行模拟，而这却还是一个未解的问题。相反，这个问题一直被作为一种纯的研究性问题进行严格的分析研究，并假设可以在全局交互性的精确度上得到提高。

现在出现了两个已建立的（部分的）全局算法，即光线跟踪算法和辐射度方法。很明显，这两种方法最通常的实现形式只模拟全局交互的一个子集：光线跟踪关注于（完全的）镜面交互，而辐射度方法关注于（完全的）漫反射交互。换句话说，当前对于这个问题的解决方案，是通过集中地讨论一种特定的全局交互来处理其内在的难题，而忽略其余的问题，并且认为交互是完美的。在镜面交互的情况下，所谓“完美”意味着无限细的光束击中表面，并在没有散射的情况下反射回来，即假设表面是完美的。而对于完全漫反射交互的情况，则假

设入射的光束在所有方向进行等量的反射形成一个以反射点为中心的半球。

我们简单地说明对于全局交互的一个解决方案，而不考虑关于计算资源的问题。讨论从光源开始，并跟随每一条光的路径（或者光线）穿过环境，到光线击中视点时停止。或者随着光线碰到物体，其能量被逐渐吸收，直到能量减到低于一定的最小值时停止。再或者光线离开了环境进入空间中时停止跟踪。为了了解全局照明算法的相关性，我们需要确定阐述问题的方法，也就是说，需要一些模型来描述光线在环境中的行为的精髓。在这一章中，我们将介绍两个全局照明模型，并给出对很多全局照明方法及其变种的综述。对于光线跟踪和辐射度方法的实施细节将在另一章介绍。

需要注意的是，因为大多数全局照明算法都采用了多种技术的某种结合，所以很难对这些方法进行分类。例如，是将二路光线跟踪算法看成是全局照明方法，还是将其看成是光线跟踪算法的一种扩展？因此，在这一章中对技术的划分不可避免地会使一些算法跨在各种算法之间，对于这些技术的排序也只是作者的一种安排。

10.1 全局照明模型

我们从介绍两个全局照明问题开始这一章。第一个问题是数学公式问题，第二个问题是
276 对当光线从一个表面运动到另一个表面时出现的交互的类型的性质进行分类。这类模型的作用是，可以用其对多个全局照明算法进行比较。大多数算法都解决整个问题当中的一部分问题。从本质上来说，这些算法含有大量的启发式细节信息，而全局照明算法模型可以实现对于模型是在哪方面进行了计算的一个比较。

10.1.1 绘制方程

我们要考察的第一个模型是1986年由Kajiya（Kajiya 1986）在计算机图形学期刊上提出的，称之为绘制方程。它通过描述表面上的点x发生的情况来概括全局照明。这是一种关于问题的完全数学性的说明，全局照明算法可以根据这个方程进行分类。事实上，Kajiya陈述这一方程的目的是：

提供对于观察它们（绘制算法）的一个统一的上下文，看这些算法以多大程度近似于一个方程的答案。

在Kajiya原来的理论中，积分由下式给出：

$$I(x,x') = g(x,x')[\varepsilon(x,x') + \int_S \rho(x,x',x'')I(x',x'')dx'']$$

其中：

$I(x, x')$ 为传递强度，或者通过点x'到点x的光线强度。Kajiya将这一值称为未被阻塞的两点间的传递强度。

$g(x, x')$ 为x和x'之间的可见性函数，如果x和x'互相是不可见的，则这个值为0。如果它们之间是可见的，则g随着两者之间距离的平方的倒数而变化。

$\varepsilon(x, x')$为从x到x'的传递发射率，它与在x的方向上x'点自发射的任何光线的强度有关。

$\rho(x, x', x'')$ 为散射项，它对应于x''和x'的方向，是来自点x''或来自x''方向到达一个表面上的点x'再向x散射出的能量强度。Kajiya将这项称为未经阻塞的三点传递反射。它与BRDF（见第7章）关联如下：

$$\rho(x, x', x'') = \rho(\theta'_{in}, \phi'_{in}, \theta'_{ref}, \phi'_{ref}) \cos\theta \cos\theta'_{ref} .$$

其中，θ'和ϕ'为与点x'相关的方位角和仰角（见7.3节）。θ为点x处的表面法向与线段$x'x$之间的夹角。

积分在s上进行，即对场景中所有表面上的所有点求积分，或者等价地是在点x'所处的半球上的所有点上积分。该方程表明，从点x'到x的传递强度等于从点x'向x的所有发射的光加上场景中所有其他表面到x'再散射到x的光，也就是说源自x''的方向的光。 277

以上述各项的形式表示的绘制方程表明，我们必须有：

- 一个由表面$\varepsilon()$发射出的光的模型。
- 对于每一个表面，一个BRDF $\rho()$的表示。
- 估算可见性函数的一种方法。

我们已经遇到过这些因素。在这里，这个公式将这些因素都结合到了一个方程中。从对这个绘制方程的考察中我们得出一些重要的概念，包括：

1) 积分的复杂性意味着，它不能用分析的方法来估算。大多数实际的算法都以某种方式降低算法的复杂性。这个方程的直接计算可以用蒙特卡罗方法来进行，而很多算法也遵循这一方法。

2) 这个方程是对问题的一种独立于观察的说明。其中的点x'是场景中的任意点。全局照明算法可以是独立于观察的，一个通常的例子就是辐射度算法；也可以是基于观察的，这时只有对那些从观察位置可见的点x'才进行估算。依赖于观察可以看成是降低了绘制方程的内部复杂性的一种方法（对于有关独立于观察/依赖于观察的方法的更详细的讨论见10.8节）。

3) 这是一个迭代方程。即，为了计算$I(x, x')$需要计算$I(x', x'')$，而这个值自身也将用相同的方程进行计算。这为解决问题提供了一个最流行的实际算法，即从图像平面以光线传播的相反方向，沿着从物体到物体的反射路径跟踪光线。采用了这一方法的算法包括：路径跟踪、光线跟踪和分布式光线跟踪，后面将对所有这些方法进行介绍。

10.1.2 辐射光亮度、辐照度和辐射光亮度方程

绘制方程的原始形式在全局照明方法中并不特别有用，在这一节中，将介绍能以不同的形式写出的有关定义，我们称其为辐射光亮度方程。

辐射光亮度L是基本的辐射度学的量，对于三维空间中的一个点，它是以$W/(sr-m^2)$来量度的光的能量密度。一个点处的辐射光亮度是方向的一个函数，我们可以为一个点定义一个辐射光亮度分布函数。作为图10-1中的二维的例子，这个函数一般来讲是不连续的。这样一个分布函数存在于三维空间中的所有点上，因此，辐射光亮度就是一个五维的量。辐照度是所有方向上入射辐射光亮度的积分：

$$E = \int_{\Omega} L_{in} \cos\theta \, d\omega$$

278

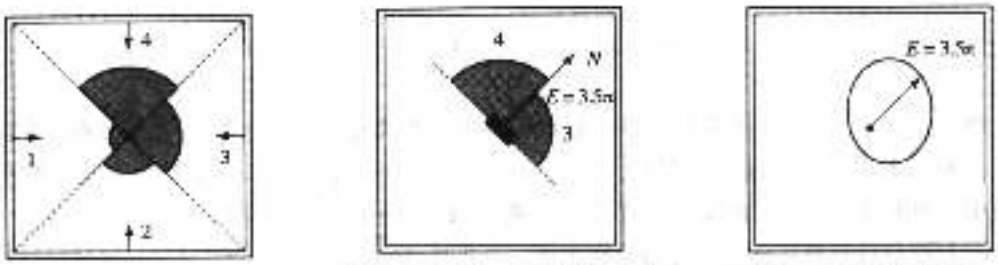
其中：

L_{in}为来自方向ω的入射或场辐射光亮度；

θ为表面法向与ω之间的夹角；

如果L_{in}为常数，则对于漫反射表面我们有：

$$L_{diffuse} = \rho E/\pi$$



a) 房间中心的一个点的二维辐射亮度分布，每一面墙表现出不同的辐射光亮度

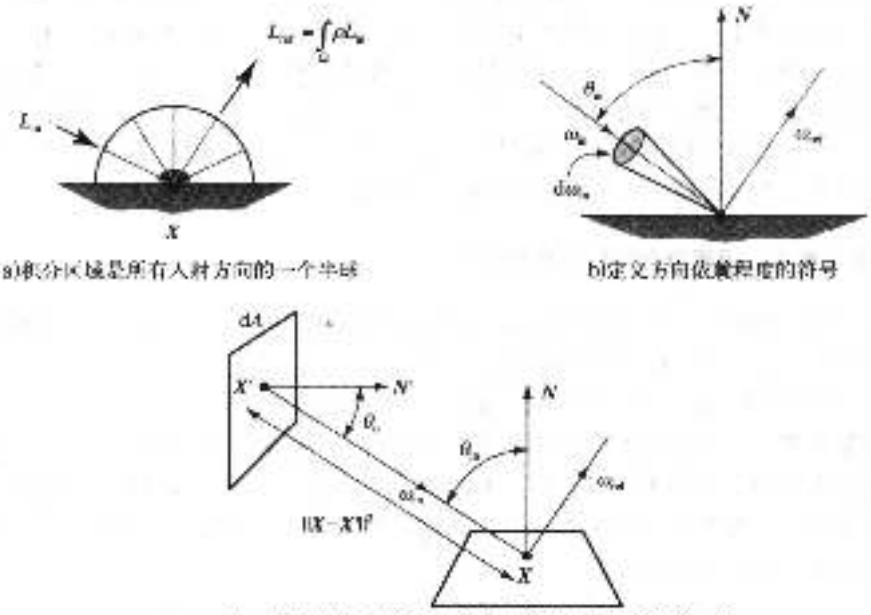
b) 表面元素上一个点的场辐射光亮度，辐照度E是辐射光亮度的余弦加权平均值，此处为3.5π

c) 如果表面元素经过了旋转，就定义了一个辐照度分布函数

图10-1 辐射光亮度、辐照度和辐射光亮度分布函数（Greger等（1998））

这两个量之间的区别在全局照明算法中是重要的，因为算法的形式可以被分类成“发射”和“收集”。发射意味着在一个表面上分布辐射光亮度，而收集意味着对辐照度的聚集，或者指在表面上光流量的累积（辐射度B与辐照度紧密相关，其单位为W/m^2）。

关于辐射光亮度和辐照度分布函数的一个重要的实际问题是前者一般来讲是不连续的，而后者一般是连续的，阴影的边界处除外。如图10-1所示，在这个简单的例子中，由于积分
279 的平均作用使辐照度分布函数变得连续了。



a)积分区域是所有入射方向的一个半球

b)定义方向依赖程度的符号

c) $\cos\theta_x dA/\|X-X'\|^2$是$\omega_x$方向上可见区域dA的投影立体角

图10-2 辐射光亮度方程

这时，绘制方程可以改造为辐射光亮度方程，其最简单的形式为：

$$L_{out} = \int \rho L_{in}$$

把方向因素加入之后，可以写成：

$$L_{ref}(X,\omega_{ref}) = L_e(X,\omega_{ref}) + \int_{\Omega} \rho(X,\omega_{in} \rightarrow \omega_{out}) L_{in}(X,\omega_{in}) \cos\theta_{in} \mathrm{d}\omega_{in}$$

其中的符号在图10-2b中加以定义。这个方程可以进行修改，使得积分在所有表面上进行（这是在实际的算法中更方便的方法），而不是使其在所有的入射角上进行。这样一来，以辐射光亮度表示的绘制方程就成为:

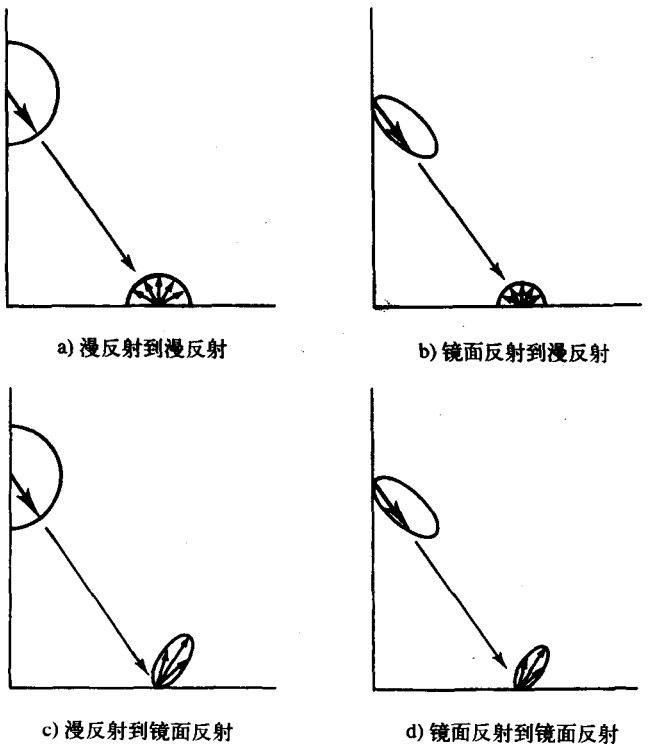
$$L_{ref}(X,\omega_{ref}) = L_e(X,\omega_{ref}) + \int_{S} \rho(X,\omega_{in} \rightarrow \omega_{out}) L_{in}(X',\omega_{in}) g(X,X') \cos\theta_{in} \frac{\cos\theta_0 \mathrm{d}A}{\| X - X' \|^2}$$

这时，方程中包含了可见性函数。这是通过按照在ω_{in}的方向上微分可见表面区域的投影区域表示实体角$\mathrm{d}\omega_{in}$来实现的（见图10-2c）：

$$\mathrm{d}\omega_{in} = \frac{\cos\theta_0 \mathrm{d}A}{\| X - X' \|^2}$$

10.1.3 路径的标注

另一个对全局照明算法的行为分类的方法是详细探讨算法所实施或模拟的是哪一种表面到表面的相互作用。这是一种简单得多的方法，而且是非数学的分类，它使我们能够容易地对常见的算法进行比较和分类。我们把某对相互作用的表面之间的交互看成是光线从光源到感应器的传播。于是，在一个点上入射光可能散射、漫反射或镜面反射，而这些光线自身也可能是其路径上前面表面的镜面反射或漫反射。接下来，我们就可以说，对于光线路径上的一对连续表面有（见图10-3）:



a) 漫反射到漫反射　　b) 镜面反射到漫反射

c) 漫反射到镜面反射　　d) 镜面反射到镜面反射

图10-3 四种光线传播的机制

- 漫反射到漫反射的传播。
- 镜面反射到漫反射的传播。
- 漫反射到镜面反射的传播。
- 镜面反射到镜面反射的传播。

280 281

在一个只有漫反射表面存在的环境中，只能有漫反射到漫反射的传播，而这一类场景用辐射度方法来求解。同样，如果环境只有镜面反射表面，则只能表现出镜面交互，而Whitted光线跟踪方法可以解决这个问题。基本的辐射度方法除了漫反射到漫反射的传播之外并不允许有其他类型的传播机制。而且，它也排除了很重要的镜面反射到镜面反射的传播。另一方面，光线跟踪方法只能处理镜面反射到镜面反射的传播。较新的算法，比如“反向”光线跟踪以及增强的用于镜面反射的辐射度方法，是另一类包含了在光线从光源到感应器的旅程上的所有交互的方法。这导致Heckbert提出了字符串标记的方法（Heckbert 1990），以便列举出光线从光源（L）到眼睛（E）的途中所发生的所有交互。这里，从光源到第一次碰撞的光线路径被标识为L。接下来在一个表面点上涉及各种传播机制的路径分类为DD、SD、DS或SS。图10-4（及彩色插图）显示了一个简单场景以及各种路径的例子。最终终止于眼睛处的路径称为E。例子中的这些路径为：

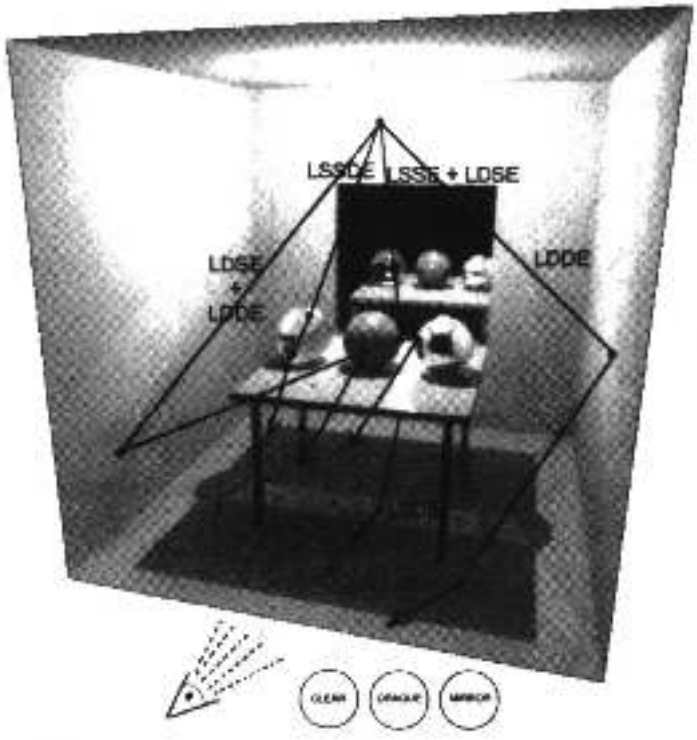


图10-4 一个简单环境中的一组全局照明路径（又见这一图形的彩色插图版）

1) LDDE。对于这个路径，观察者看到了由桌子投出的阴影。光线从右侧的墙壁向地板漫反射。注意，任何从一个阴影区域反射的光线在L和E之间的两个交互中都必须有一个最小值。

282

2) LDDE+LDSE。在这里，用户看到了球的黑暗的一侧，在这一侧不能接收任何直接的光。光线被调节为一个点光源，所以“赤道”线以下的任何区域都将处在阴影中。从墙上反射出来的漫反射照明被导向眼睛，而且因为球是“闪光”的，所以向眼睛的反射既有镜面反射也有漫反射。

3) LSSE+LDSE。光线是从完全镜面表面向眼睛反射的，观察者看到了镜面表面中不透明的或有颜色的球的反射。

4) LSDE。这时，观察者看到了一个阴影区域，这个区域比主要的桌子阴影区域要亮一些。这是由于从镜面反射了额外的光，并且被导入到了桌子的下方。

5) LSSDE。这个路径在L和E之间有三种交互，用户在桌子的上面看到了一处焦散，桌子的上表面是一个漫反射表面。第一个镜面交互发生在球的上表面，来自点光源的光线在整个球上被反射出来。当光线从球上反射出来并碰到漫反射的桌子表面时，出现第二个镜面交互。反射的效果是把穿过球的光线集中到桌面上方的小区域上，如果没有透明球的话，这些光线将占据这一区域。于是用户在漫反射表面上看到了一块亮的区域。

一个完整的全局照明算法将必须包含任何光线的路径，这些路径可以写成L(DIS)*E，其中“|”指“或”，而“*”指可重复出现。对于局部反射模型的使用意味着使用LDIS类的光线路径（每一种路径的光强度都被单独计算，然后再像用Phong反射模型那样将这些值结合起来），而附加一个隐藏面消除算法就意味着对LDISE类光线路径的模拟。于是，局部反射模型只模拟单位长度的字符串（在L和E之间），而观察阴影中的一个点就意味着一个长度至少为2的字符串。

10.2 全局照明算法的发展

现在让我们来考察流行的或已建立起来的全局照明算法的发展，以此作为前述概念的基础。所讨论的算法的安排顺序有一些随意，但主要是按照从不完全的解（光线跟踪和辐射度方法）到一般性解的顺序。这一节主要是按照全局交互的思想来讨论算法。

返回到对此问题的蛮力解答上来。那时，我们考虑的概念是从一个光源开始，追寻在场景中每一条被发射出来的光线。也就是说，从计算的角度来看，这是一个很棘手的问题。对于这个问题解的一种近似是以某种形式从讨论光线与物体相互作用开始，或者只考虑一个光 283
线的子集，即只考虑那些从光源开始到碰到场景的光线。导致出现光线跟踪方法和辐射度方法的主要近似是把场景限制在分别只含有镜面反射或只含有（完全）漫反射的情况。

接下来，我们给出对于光线跟踪方法和辐射度方法的一个综述，这已足够与我们讨论的其他方法相比较。我们把这些重要方法的实施细节留到其他章节再讨论。

10.3 已建立的算法——光线跟踪和辐射度

10.3.1 Whitted 光线跟踪方法

Whitted光线跟踪方法（可视性跟踪和眼睛跟踪）从眼睛以与光线传播方向相反的方向跟踪到场景再朝着光源跟踪光线。为了用光线跟踪方法产生一个场景的二维图像的平面投影，我们只关注那些在感应器或眼睛处终止的光线。因此，从眼睛开始跟踪光线并到达场景是有意义的。因此，这是一种依赖于观察的方法。对于这一算法的一个简单表示如图10-5所示。

这个过程通常被可视化成一棵树，树上每一个结点都有一个表面碰撞点。在每一个结点上，我们产生一条光线跟踪以及一条反射光线或者一条折射光线，或者两者都有。

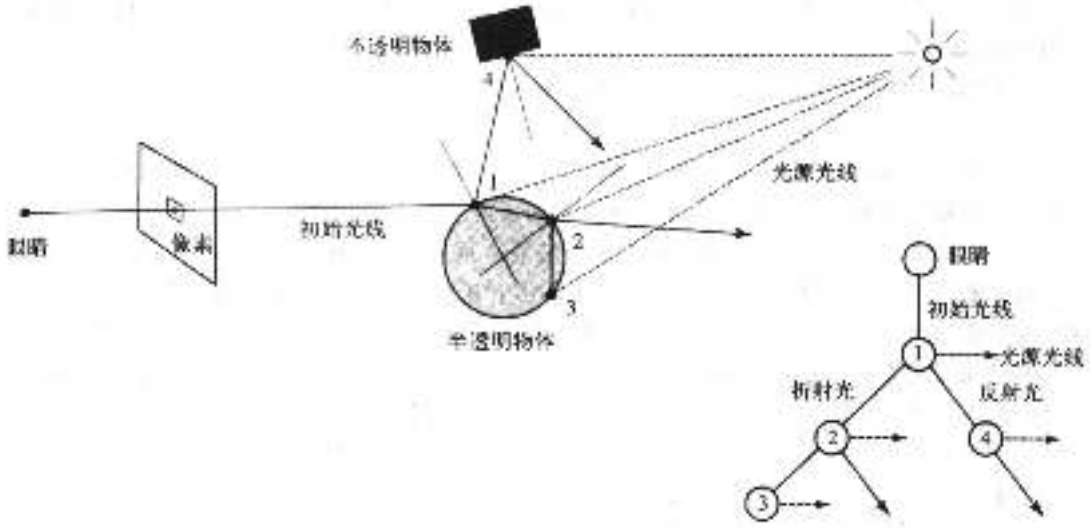


图10-5 Whitted光线跟踪方法

Whitted光线跟踪是一种混合物，即全局照明模型又加入了局部模型。现在让我们来考虑全局交互，典型的算法只包括了完全镜面交互。光线射入场景中，当光线碰到一个表面在相互作用点上发出了反射光（和折射光），然而这些光本身也被递归地跟踪。当光线的能量低于
[284] 一个预先定义的最小值，或者光线离开了场景进入到空的空间，或者光线碰到了一个完全漫反射表面时算法终止。这样一来，光线跟踪的全局部分就只计算纯的镜面反射到镜面反射的相互作用。从理论上来讲，没有什么理由能阻止我们计算漫反射的全局交互。只是这时在每一个碰撞点上，入射光线会向以该点为中心的一个半球表面的所有方向发射反射光。

对于全局镜面分量，我们加入了一个直接的贡献，即由该点发射一条光线到光源的计算结果，光源在这个模型中总是设为点光源。由光源到点的可见性及其方向可以用于计算一个局部的或者直接漫反射分量，在局部反射模型中，光线只是L。这样一来就考虑到了（直接的）漫反射（但是没有考虑由漫反射到漫反射的情况）相互作用。这一部分有时被称为阴影光线或阴影试探器（shadow feeler），因为如果它碰到了两点之间的任何物体，则我们就可以判断该点是处于阴影中的，但是，一个更好的术语是光源光线（light ray），它强调的是用于对一种直接的贡献（用局部反射模型）进行计算，而这个贡献将在树上被传递。Whitted光线跟踪算法的主要问题是它局限于镜面的相互作用，而大多数实际的场景都含有大量的漫反射表面。

考虑图10-4中的路径LSSE + LDSE，重画于图10-6中，并加入了光线树。从眼睛发出的初始光线碰到了完全镜面球。对于这个球来讲，局部的漫反射模型对其没有贡献。在下一个相交处，击到了一个不透明球，并跟踪一个全局的镜面分量进入到天花板，天花板是一个完全的漫反射表面，这时递归终止。在该点处又有一个对于球的来自局部反射模型的贡献，观察者在与该光线相关的像素上看到了镜面球中不透明球反射图像的颜色。

可能人们会产生一些想法，即Whitted光线跟踪可以模拟的路径只能局限于LS*E和LSD*E。因此，被光线跟踪的图像就表现出在邻近物体中的闪光物体表面的反射。如果该物体是透明

的，则任何在该透明物体后面的、观察者可以看到的物体都被折射了。而且，我们还将在第12章中看到，阴影作为模型的一部分被计算出来，但这些阴影是“完全”的或者是硬边界的阴影。

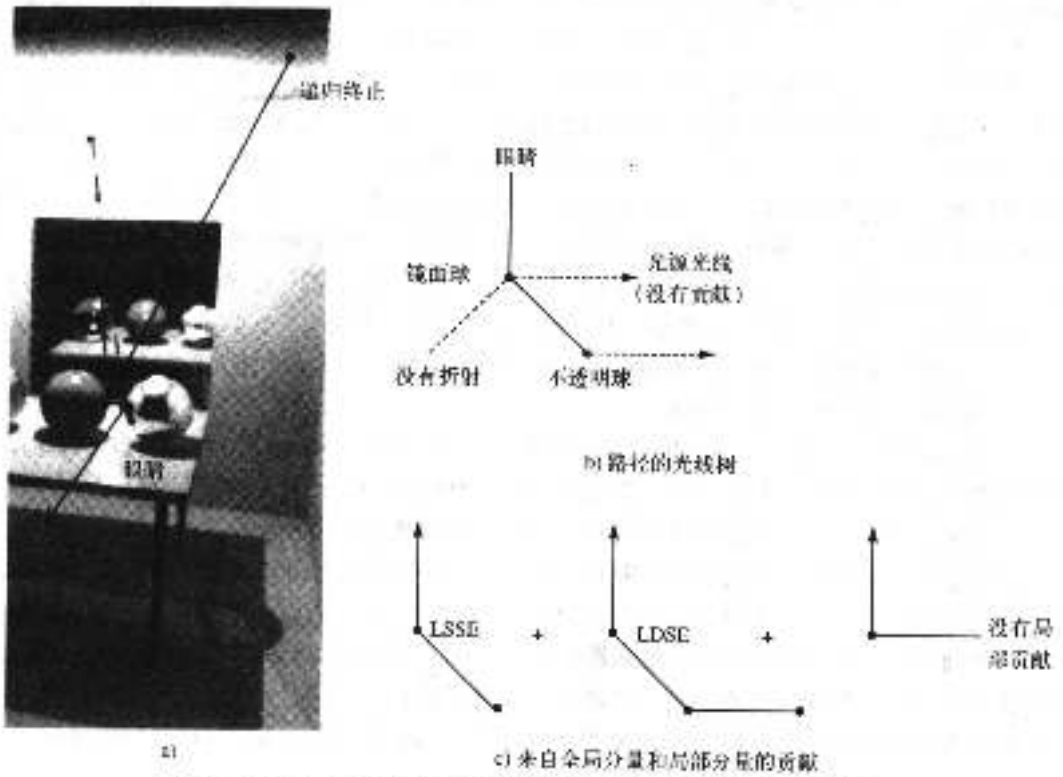


图10-6 Whitted 光线跟踪：对于图10-4中所示的一种情况的光线路径与局部贡献和全局贡献之间的关系

按照绘制方程的思想来考虑Whitted光线跟踪算法，则下面的情况成立，散射项ρ被简化成了完全反射（和完全折射）定律，因此，在所有的S上的积分，即整个场景就简化成（对于反射）只计算一条向外的光线再加上光源光线，光源光线给出漫反射分量，然后，再把这两者加起来。这时，绘制方程的递归结构就在算法中完全反映出来，但是积分操作就简化成了三个计算分析分量的一个求和，即反射、折射和光源光线三个贡献的和。

10.3.2 辐射度方法

经典的辐射度方法实现对漫反射到漫反射相互作用的计算。考虑了场景中曲面片（或多边形）之间各条光线的“相互作用”。解法是独立于观察的，并由场景中每一个曲面片的恒定辐射度组成。独立于观察意味着问题的解是对场景中的所有点进行计算，而不是只对那些眼睛看得见的点进行计算。这就暗示着辐射度方法的求解必须再跟着另一个过程来计算投影。但是，大多数工作都在辐射度过程中完成了。一个问题或者与经典的辐射度方法相矛盾的地方是在处理过程开始之前必须对场景进行初始离散化，但是，进行这一工作的最好方法是依赖于问题的解的。换句话说，在对问题有了解或部分解之前，我们并不知道离散场景的最好

方法。这是辐射度方法所遇到的一个很突出的问题，也是采用辐射度方法时遇到的困难。

可视化辐射度处理过程的一种方法是从考虑把光源作为发射曲面片（数组）开始。我们从光源向场景发射光线，然后考虑发出光线的曲面片与所有该曲面片可以看到的那些接收光线的曲面片之间的从漫反射到漫反射的相互作用，发出光线的曲面片为第一次击中的曲面片。
285 ~ 286 在这些曲面片上存放了一些光线，然后，根据传送到曲面片上的能量和已经被发送回场景中去的能量的大小对这些曲面片排序。具有最大的未发送能量的曲面片被选择出来，并被看成是下一个将要发射能量的曲面片。过程迭代地进行，直到初始光线的能量中的一个（高的）百分比被分布到了场景中为止。在这个过程中的任一阶段都会有一些分布出来的能量返回到那些曾经被考虑过的曲面片上，这也是过程迭代进行的原因。由于从定义来看，每一个曲面片的反射系数都小于1，而在迭代过程中的每一个阶段将会有越来越多的初始光线被吸收，所以这个过程最终将收敛。图10-7（彩色插图）为采用这一算法的一个正在处理的求解过程。所示的阶段是在第20次迭代之后的解。其中的四个示意图分别是:

1) 辐射度的解作为迭代过程的输出。每一个曲面片都分配了一个恒定的辐射度。

2) 对前面的解进行了插值处理。

3) 同样的解，但加入了环境项。环境照明在场景中的所有曲面片上均匀分布，以给出一个早期的良好照明的解（在第11章将详细讨论这一增强作用）。

4) 前两个图像之差。这给出了必须加入未发射能量部分的辐射度的指示性信息。

在任意两个曲面片之间的从漫反射到漫反射交互的光线传播通过考虑曲面片之间的几何关系（表示为形状因子）来计算。与光线跟踪算法相比，我们从光源穿过场景，以一种从曲面片到曲面片的漫反射相互作用方式来跟踪光线，而不是跟踪每一束光线，两个曲面片之间的形状因子对把各个曲面片连接起来的曲面片的效果进行了平均。事实上，以这样的方式来考虑辐射度方法是以一个算法结构来实施的。它被称为渐进式细化方法（progressive refinement method）。

必须对这种简单的原理进行修正。在一般情况下，因为某些中间曲面片的存在而使一个曲面片对于另一个曲面面来讲可能是部分可见的。可以用计入了这一现象的可见性过程来修正算法（不要与后面的投影计算相混淆，因为投影计算正常情况下包括隐藏面消除）。最终的结果是为场景中的每一个曲面片分配一个恒定的辐射度值，这是一个不依赖于观察的解。然后再把这一个解注入到Gouraud型绘制程序中，产生一个投影。按照曲面片分类的思想来看，传统的辐射度方法是LD*E。

辐射度方法的明显问题是，尽管人造的场景通常情况下大部分是由漫反射表面组成的，但是，镜面物体也是很常见的，而这些镜面物体却不能用辐射度绘制程序来处理。还有一个更细节性的问题是在使用辐射度方法进行计算之前，必须将场景离散成曲面片或三角形，如果这些多边形太粗糙的话，就会出现困难。

现在让我们从绘制方程的角度来考虑辐射度方法，辐射度指的是每单位时间在每单位面
287 积上的能量，由于我们只考虑漫反射照明，所以把绘制方程重新改写为:

$$B(x') = \varepsilon(x') + \rho(x')\int_S B(x)F(x,x')\mathrm{d}x$$

这时，唯一对方向的依赖被结合到了形状因子F中。这时的方程表明，表面元素x的辐射度等于发射项加上由场景中所有其他元素向x上所辐射的辐射度。形状因子F是一个系数，

它只是x和x'之间空间关系的函数。这一关系决定了到达x的$B(x')$的分数。F还包括了可见性计算。

10.4 全局照明中的蒙特卡罗技术

在这一节中，我们将对蒙特卡罗技术进行介绍。这一技术在数学上的详细介绍已经超出了本书的范围（有关对于蒙特卡罗理论的综合性论述以及该方法在全局照明中的应用请见Glassner（1995)),具体的情况是，运用蒙特卡罗技术的方法可以用算法的术语来解释。然而，关于其中所隐含着的因素的某些直观的理解是必须的，以便对我们将在下面讨论的例子中使用的特定策略有所理解。没有这种直观的感觉就难于理解Whitted光线跟踪方法和Kajiya的蒙特卡罗方法之间的差别，后一种方法又被Kajiya称为路径跟踪（见10.5节）。

蒙特卡罗技术用于解像绘制方程这样的积分问题，这种方程没有分析解或数值解。蒙特卡罗技术的做法是计算被积函数随机样本的平均值，然后把这些值加在一起，再取平均值。在最终绘制的图像中，这一过程的可视化效果是噪声。蒙特卡罗技术的引人之处是其易于实施，因为算法从原理上来讲是简单的。一个同等重要的优点是其一般性。不需要对其进行预先的简化（比如像Whitted光线跟踪方法中的完全镜面反射以及辐射度方法中的完全漫反射)。由于这一方法在场景的几何采样和在表面光学性质的几何采样上都是采用点采样，所以出现了这种情况。蒙特卡罗方法问题在设计那些可以迅速获得对于积分的精确的或低变化率的估计值的技术时出现。

对于估计积分的蒙特卡罗方法所隐含的思想可以用一个一维的简单例子来说明，考虑估算下面的积分:

$$I=\int_0^1 f(x)\mathrm{d}x$$

可以通过取一个随机数$\xi \in (0, 1)$，并计算$f(\xi)$来估算I。这被称为初始估计数。我们可以把估算值的变化定义为:

$$\sigma^2_{\text{prim}}=\int_0^1 f^2(x)\mathrm{d}x-f^2(\xi)$$

288

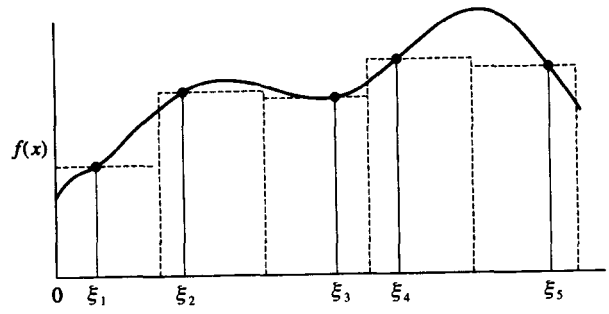
对于单个样本，这一值可以期望为很高。在实际应用中，我们取N个样本，给出一个所谓的二次估值，很容易看出:

$$\sigma^2_{\text{sec}}=\frac{\sigma^2_{\text{prim}}}{N}$$

这一观察在实际应用中非常重要，估算的误差反比于样本数的平方根。例如，如果要将误差减半，则必须选取四倍的样本。同样地，我们可以说，每一个增加的样本都对结果有越来越少的影响。这一说法必须满足这样的事实，即对于每一个样本来说，计算机图形学都倾向于付出相等的一般来说较高的代价。于是，蒙特卡罗技术的主要目的是在已知样本个数N下得到可能的较好结果。这意味着，所导致的策略的方差减少。选择样本的两个策略是分层采样和重要性采样。

分层采样的最简单形式是把积分的区域分成相等的层次，用一个或多个随机样本估算每一个偏积分（见图10-8)。用这种方法对每一个子区域分配相同数量的样本，于是有:

$$I=\int_0^1 f(x)\mathrm{d}x$$
$$=\sum_{i=1}^{N}\int_{S_i} f(x)\mathrm{d}x$$
$$=\frac{1}{N}\sum_{i=1}^{N} f(\xi)$$



289

图10-8 分层采样$f(x)$

这种估算导致一个方差，该方差保证低于在整个积分区域分布随机样本所获得的估算值。在计算机图形学中最为人所熟知的分层采样的例子就是像素采样中的抖动。在这个例子里，一个像素代表积分的区域，该区域被细分成相等的层次，通过对每一个层次的中心点抖动产生一个样本点（见图10-9）。

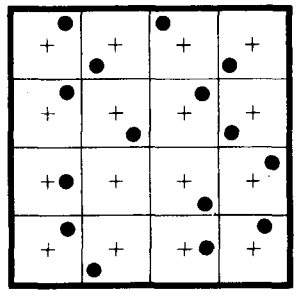


图10-9 计算机图形学中的分层采样：把一个像素分成16个子像素，通过对每一个子像素或层的中心点进行抖动产生16个样本点

顾名思义，重要性采样倾向于在被积函数的重要区域进行采样。重要性采样暗示着要对我们将要估算出的积分函数有事先的了解，对于这一点，初看起来似乎是矛盾的。但是，大多数绘制问题都涉及一个被积函数，而这个函数又是两个函数的乘积，其中的一个函数在绘制方程中是先验的。例如，在蒙特卡罗方法中，对一个镜面的表面进行光线跟踪，我们可能选择这样的反射光，即它们倾向于在镜面反射方向聚集，这样就可以在那些可能会返回高值的区域对（已知的）BRDF进行采样。一般来说，我们把样本进行分布，使得在函数值大的区域或函数值变化很明显、很迅速的区域的密度最高。再一次考虑简单的一维例子，可以写出：

$$I=\int_0^1 p(x)\frac{f(x)}{p(x)}\mathrm{d}x$$

其中，第一项$p(x)$是一个重要性权重函数。而这个函数本身又是样本的概率密度函数（PDF）。也就是说，样本的选择需要满足$p(x)$。为了达到这一目的，我们将$p(x)$定义成PDF的累积函数：

$$p(x)=\int_0^x p(t)\mathrm{d}t$$

选一个均匀随机样本τ，估算$\xi=P^{-1}(\tau)$。用这一方法使方差成为下面的形式:

$$\sigma^2_{\mathrm{imp}}=\int_0^1\left[\frac{f(x)}{p(x)}\right]^2 p(x)\mathrm{d}x-I^2$$
$$=\int_0^1\frac{f^2(x)}{p(x)}\mathrm{d}x-I^2$$

问题是如何选择$p(x)$。这可以是一个满足下列条件的函数: 290

$$p(x)>0$$
$$\int p(x)\mathrm{d}x=1$$
$$P^{-1}(x)\text{ 是可计算的}$$

例如，我们可以选择使$p(x)$为$f(x)$的单位化的绝对值，或者选$f(x)$的一个近似(见图10-10)。任何满足上述条件的$f(x)$并不是充分必要条件。如果我们所选的$f(x)$离理想情况太远，则这一技术的效率就会低于采用随机抽样的基本方法。重要性采样在全局照明算法中占据非常重要的地位，其出于简单和显而易见的原因而使用蒙特卡罗方法。即，尽管绘制方程在场景中的每一个点处描述了全局照明，但是，我们并不需要一个处处精确的解。例如，我们需要在具有很亮的镜面表面上的解比在有较暗照明的漫反射墙壁上的解更精确。重要性采样使我们能够建立起运算成本可以根据最终精确度进行分布的各种算法，而最终精确度是一个光照水平和表面类型的函数。

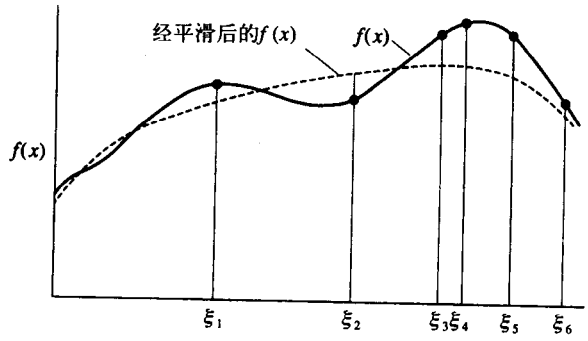


图10-10 重要性采样思想的示意图

蒙特卡罗算法在计算机图形学中的一个重要的隐忧是它会产生统计性的噪声。例如，我们来考虑Whitted光线跟踪方法，并将蒙特卡罗技术用于光线跟踪。在Whitted光线跟踪方法中，完全镜面反射的方向总是朝着所选择的方向，在某种意义上，积分就简化为一个产生无噪声图像的确定性算法。而一个基本的蒙特卡罗算法若模仿Whitted光线跟踪方法的话，会产生一个最终像素的估值图像，而这个图像一般来讲与Whitted光线跟踪方法的解会有一些不同。这些不同表明它们是一种可察觉的噪声。还应注意到，在Whitted光线跟踪方法中，如果我们忽略潜在的走样问题的话，就不必要在一开始为每一个像素发射一束光线。而用蒙特卡罗方法，

我们是用绘制方程的样本去计算一个像素的光强度估值，这样我们就需要很多在场景中来回跳跃的光线/像素对。在Kajiya的创始性算法中（Kajiya 1986），对于每一个像素总共使用了40个光束。Kajiya的算法我们将在下一节论述。

使用蒙特卡罗方法的全局照明算法都是基于这些简单的想法。这些想法的内部复杂性源
291 自这样的事实，即这时的积分是多维的。

10.5 路径跟踪

Kajiya（1986）在其介绍绘制方程的经典文章中首先意识到Whitted光线跟踪算法对于绘制方程来讲有确定解。在同一篇文章中，他还提出了Whitted光线跟踪方法的一个非确定变种——一种他称为路径跟踪的蒙特卡罗方法。

Kajiya给出了绘制方程与路径跟踪算法之间的一种直接的连接，他把方程重新写为:

$$I = g\varepsilon + gMI$$

其中，M为绘制方程中的积分给出的一个线性算子。然后，可将这个方程改写成称为Neuman级数的一个无限级数:

$$I = g\varepsilon + gMg\varepsilon + g(Mg)^2\varepsilon + g(Mg)^3\varepsilon + \cdots$$

其中，I为方向项、一次散射项、二次散射项…的和。这就直接导出了路径跟踪，理论上称为随机行走。光源的光束被反向跟踪，正如在Whitted光线跟踪中那样，从像素开始，在场景中跳跃，从第一个碰撞点到第二个碰撞点再到第三个碰撞点等。随机行走在一定的步数之后必须停止下来，也就是说当我们确信不会再有显著的贡献时在某些点上对上述级数进行截断。

像Whitted光线跟踪方法那样，路径跟踪是一种依赖于观察的解。我们在前面已经讲过，对于将光线跟踪方法扩展到处理所有光线-表面间的相互作用，包括漫反射和从一个碰撞点发出的光线是没有理论上的限制的，要实现这种扩展只是有碍于计算上的不可行性。路径跟踪对漫反射相互作用的实现是通过在每一个像素上发出大量的光线（Whitted光线跟踪方法只发出一条光线），然后，对每一束光线跟踪一个路径，而不是跟踪一个光束并在每一次碰撞之后跟踪大量的子反射路径。其主要思想如图10-11所示，可以将其与图10-5相比较。所有的表面，不管是漫反射表面还是镜面反射表面，都可以发射出一个反射/折射光线，这与Whitted光线跟踪方法正好相反，因为Whitted光线跟踪方法在遇到一个漫反射表面时就终止了递归。两种方法之间的另一个重要差别是一些光线（在原例中为40）是对于每一个光素所发出的，以便对BRDF进行采样。这样一来，该方法就模拟了L(D|S)*E交互。

对于从光源到终止用一条路径的基本路径跟踪算法是高代价的。如果随机行走在光源处不终止，则它将返回，对于最终的估值变为零贡献，除非光源足够大，这时，路径将趋向于在它们到达光源之前终止。Kajiya通过引入一束光线或阴影光束解决这个问题，该光束射向在随机行走中每一个碰撞点发出的光源（区域）上的点，并且在路径中对每一个点上的这一贡献进行累加（如果来自相同点上的反射光线直接击中了光源，则忽略其直接的贡献）。

Kajiya指出，Whitted光线跟踪方法在某种意义上是浪费的，而且随着算法深入到树中，它会进行越来越多的工作。而同时，树叶深处的光线反射和折射对于像素光强度的贡献却越来越少。在Kajiya的方法中，树有一个分数，在每一个碰撞点上，有一个来自基于镜面反射和漫反射BRDF的分布的随机变量用于发射一束光线。Kajiya指出，在这个过程中，对于每一个

像素反射、折射和阴影光束必须保持一个正确的比例。

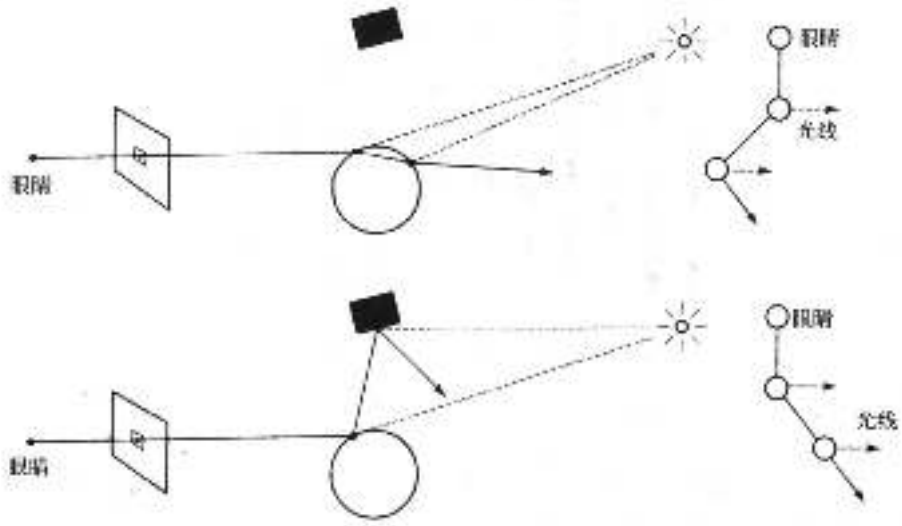


图10-11 路径跟踪中的两种光线（从同一个像素点发出）

根据蒙特卡罗理论，原来的算法降低了直接照明的方差。但是，间接照明的方差还是保持为高，这对于LS*DS*E路径尤其是如此（关于这类路径的更深入的考虑见10.7节）。在这类路径上，漫反射表面从一个发射程序通过一些镜面反射路径接收光线。于是，算法需要用很长的时间来产生一幅具有良好质量的图像（Kajiya的记录是对于一个512×512像素的图像，每个像素40个路径则需花费20小时）。

重要性采样可以引入到路径/光线跟踪算法中，引入方法是使其基于BRDF，并且保证在将要返回大的贡献值的方向上发送更多的光束。但这也可以通过近似的方法来实现，因为相伴的PDF是不能被积分也不能反转的。另一个问题是，BRDF只是局限于当前表面点上的被积分函数的一个分量，我们对于在这个点上来自半球空间的所有方向上的入射光并没有相关的知识，而这一值是一个域辐射光亮度（除了由于直接照明而产生的光线之外）。在传统的路径/光线跟踪方法中，对所有的光线都是互相独立地进行跟踪，并且累加到一个像素上再进行平均，没有利用在过程中所累积并获得的相关信息。这一重要的观察导致一些存储光线跟踪中所获得信息的策略，10.9节将讨论这些策略中最熟悉的那些策略。

292
293

10.6 分布式光线跟踪

像路径跟踪一样，可以将分布式光线跟踪看成是Whitted光线跟踪方法的一种扩展，或者看成是蒙特卡罗方法的一种策略。分布式光线跟踪（分布光线跟踪、统计光线跟踪）是由Cook在1986年提出的，其可能的动机是Whitted光线跟踪方法只能考虑完全镜面交互，而这种情况只有当场景是由完全镜面表面或者完全折射表面的物体组成的情况下才能发生。用Whitted光线跟踪程序对于完全的实体玻璃所产生的效果尤其使人困惑或者是不真实的。例如，考虑一个玻璃球。观察者看到一个圆圈，在其内部发生了完全的清晰的折射（见图10-12），而并没有我们常遇到的由于不完全反射出现时发出散射而造成的那种球。

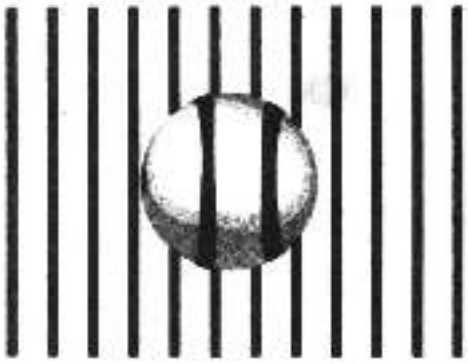


图10-12 通过一个实体玻璃球的完全折射与纹理图是无法区分的

根据我们对光线交互的考虑，这时的分布式光线跟踪也是只考虑镜面相互作用，但是，这时也模拟了不完全的镜面反射相互作用。其方法是利用光线跟踪方法，并在每一个碰撞点处建立一个反射波束，波束的形状依赖于材料的表面性质。不是在交点处扩展一束反射或折射的光线，而是扩展了一组光线，这组光线是反射波束的样本。这样就产生了更具真实性的光线跟踪场景。在邻近物体的表面上反射的物体的形状看起来模糊不清，因为可以模拟散射的不完全性，所以透明的效果也更真实了。可以将区域光源包含到场景中，以产生阴影。考虑图10-13，如果像现实中那样，球的镜面表面实际上是不完全的，则我们可能会期望看到一种镜面球中的不透明球的模糊反射效果。

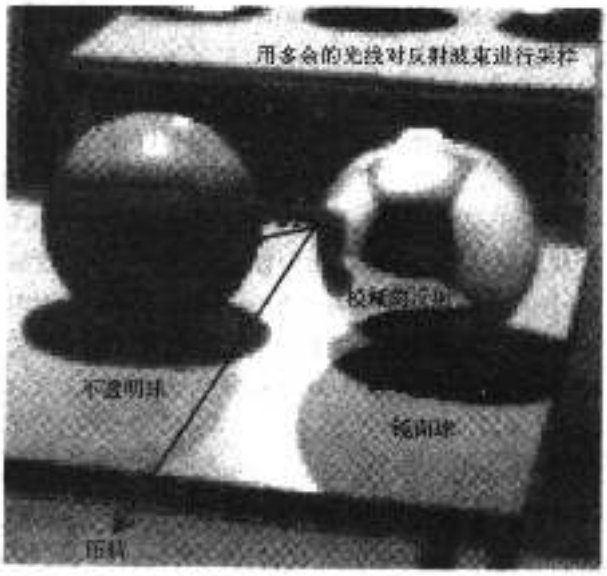


图10-13 对于反射的分布式光线跟踪（对于这种情况的完整几何描述见图10-4）

因此，路径的分类方法又一次为LDS*E或者LS*E，但是，这时所有的路径都被计算到了（或者更准确地说，通过明智地选择采样，使得计算了所有路径上的反射和折射效果的估计值）。在图10-13中，通过一束目光光线可以发现有三种LDSE路径。由这些光线所击中的墙上的点

被结合到了一条光线中（并最终进入到一个像素）。

除了上面提到的效果外，Cook等（1984）的方法考虑了一个有限的孔阻照相机模型，它产生的图像表现出场深度以及由于移动物体而产生的运动模糊和有效的反走样（关于这一算法所暗示的反走样见第14章）。图10-14（彩色插图）是一个用分布式光线跟踪程序绘制的图像，它表现出了场深度的现象。这一工作在理论上的重要性是实现了将所有这些现象结合到一个多维的积分中去，然后再用蒙特卡罗技术对其进行求解。在这一算法中，光线路径与Kajiya方法中的光线路径相似，只是增加了摄像机的镜头。算法采用了层次采样和重要性采样的一种结合。一个像素被分成16个子像素，用未经校正的抖动在一个子像素中从一点发出一条光线。镜头也被分层了，像素上的一个层次与镜头上的一个层次相对应（见图10-15）。对反射波束和折射波束进行重要性采样，采样点和抖动点相同。Cook等（1984）对这些样本进行了预计算，并将其存储在一个查询表中，表中还关联有表面类型的信息。每一条光线导出一个像素中其位置函数的索引，初始的光线和其所有的后裔具有相同的索引。这意味着，从第一次碰撞发出的沿着与R相关的方向的一条光线将与对于每一个物体上的同样的与R相关的方向中所有其他碰撞产生的光线进行合并（见图10-16）。这就保证了，每一个像素的光强度都是基于分布式采样的，每个像素的光强度最终要由16个样本来确定，采样的分布是根据重要性采样判据在每一个物体都有的镜面反射函数的全部范围之内进行的。注意，没有什么因素可以影响将查询表定位为二维的，并且用入射角进行索引。这使得依赖于入射角的镜面反射函数能够被执行。最后，请注意，折射的计算与镜面传递函数对于折射方向的方法完全相同。

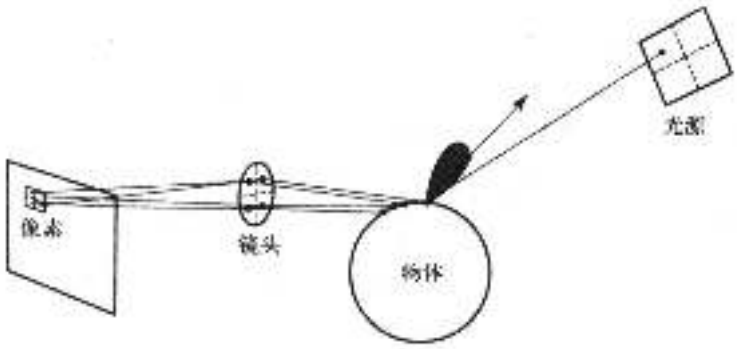


图10-15 分布式光线跟踪：每个像素4条光线。对像素、镜头和光源进行分层采样，对反射波束进行重要性采样

总之，我们有：

1) 分布光线的过程意味着统计型反走样成为方法的一个积分部分（第14章）。

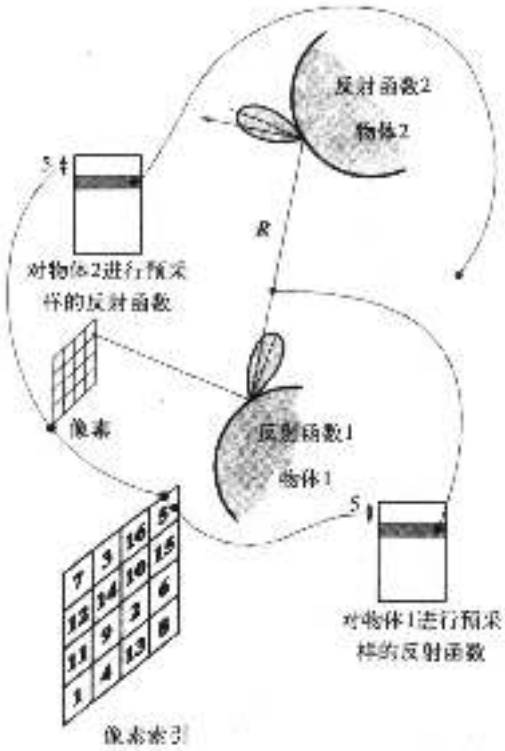
2) 分布反射光线产生模糊的反射。

3) 分布折射光线产生有说服力的半透明。

4) 分布阴影光线导致半影。

5) 在摄像机的镜头区域上分布光源产生场深度。

6) 按时间分布光线产生运动模糊（暂时的反走样）。



294 - 296

图10-16 分布式光线跟踪和反射的光线

10.7 二路光线跟踪

二路光线跟踪（或称双向光线跟踪）最初是为了将镜面反射到漫反射的传播机制与通用的光线跟踪模型相结合而建立的。这就考虑到了焦散的情况，这一现象是由通过像玻璃或水这样的介质反射的光线在漫反射表面形成的一种散射类型。人们可能会经常在游泳池的底部和两侧看到美丽的椭圆形的亮光，这是由于太阳光在被风扰动后的水面上的折射造成的，它使得光线的能量在游泳池各侧的漫反射表面上发生了变化。图10-17展示的是图10-4中的场景发出的一条光线，它从光源发出，在球中折射，并在（漫反射的）桌子表面形成焦散。这是一种LSSDE路径。

二路光线跟踪首先是由Arvo（1986）提出的。在Arvo的方法中，来自光源的光线在透明的物体中被追踪，来自镜面物体的光线也同样被跟踪。这样一种策略的工作中心是在第一条路径上导出的信息如何传递给第二条路径。Arvo建议用光线图或照明图来达到这一目的，光线图或照明图是由数据点的一个网格组成的，它被以与传统的纹理映射方法非常相似的方法赋予场景中的每一个物体。

总之，二路光线跟踪模拟LS*DS*E型的光线路径。这个方法依赖于这样一种假设，即光源和眼睛只能遇到一次D相互作用。第一条路径由从光源发射出的光线组成，并跟随这些光线进行镜面相互作用，直到其击中了一个漫反射表面为止（见图10-17）。来自每一条光线的光能在漫反射表面被存储，这个表面已经按某些方式细分成了小单元或小容器。实际上，第一条路径是对漫反射表面施加了一个纹理图或照明图，即焦散的亮度变化。照明图的分辨率是至关重要的。对于一个固定数量的发射光线，映射图若太细可能导致映射图元素没有光线，

但如果太粗，则可能导致模糊。

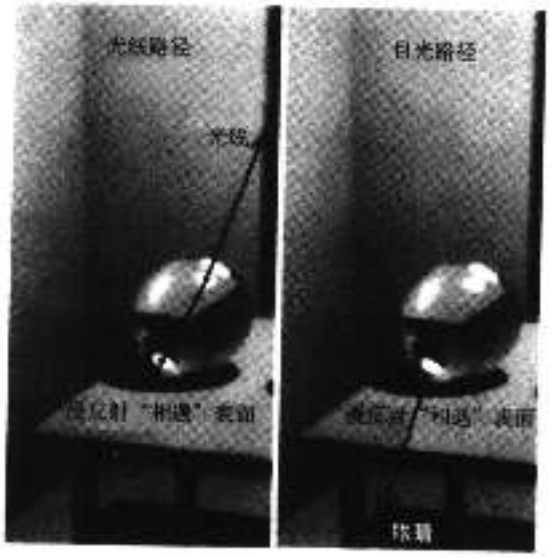


图 10-17

第二条路径是目光的跟踪路线，也就是传统的Whitted光线跟踪。它在漫反射表面上终止，并在照明图中使用存储的能量作为对光线能量的一种近似，这种光线能量在从碰撞点发出的任何可能的方向上有漫反射时可以获得。在所显示的例子中，第二条路径模拟DE路径（或者对于跟踪方向来说是ED路径）。在漫反射表面上来自第一条路径上被跟踪光线的照明的扩展依赖于这样一个事实，即表面上漫反射照明的变化速率是很慢的。指出下面的情况是重要的，也就是在任一路径上都可能只有一个漫反射表面。目光的跟踪和光线的跟踪都在漫反射表面上结束，这一表面是两种跟踪的会合点。

很容易看出，我们不能仅用目光跟踪来模拟LS*D路径。目光不一定击中该光线，我们也不能得出由于镜面反射到漫反射的作用而使表面接收到了多余的照明的结论。对于最简单的LSDE路径，这一问题的示意图如图10-18所示。

详细的过程示意图如图10-19所示。光线在折射之后在P点击中了一个表面。将其用一个标准的纹理映射函数T索引到与物体有关的光线图中。在第二条路径中，目光击中P，用同样的映射函数收集任何对P点的照明，并用这个贡献对于该点所计算的局部光强度进行

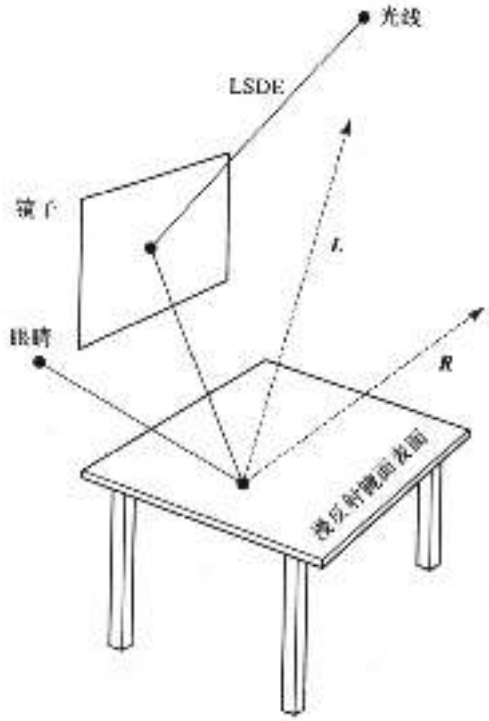
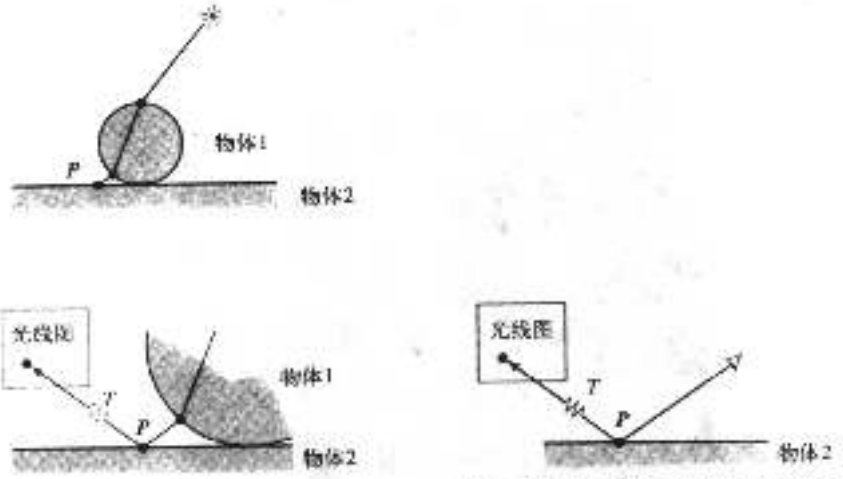


图10-18 一个LSDE路径的例子（对于SDE路径的例子又见图10-4和图10-17）。目光可以"发现"光线L和反射的光线R，但是不能发现LSDE路径

加权。



a) 第一条路径：用标准的纹理映射T把光线存储在光线图中

b) 第二条路径：当按惯例对物体2进行日光跟踪时，通过对光线图用T进行索引来获得P点的多合照明

图10-19 二路光线跟踪和光线图

在这里，重要的一点是第一条路径是独立于观察的，我们为每一个物体建立一个光线图，在这个场景中，光线图与纹理图相似，它成为该物体表面性质的组成部分。在完成了光线图的构建之后，我们可以从任意观察点来使用它。对于每一个场景，光线图只需要计算一次。

图10-20（彩色插图）a和b分别显示了用Whitted光线跟踪程序和二路光线跟踪程序绘制的场景。在这个场景中，有三个LSD路径：

1) 两个来自红球的焦散——一个直接来自光线，另一个来自从弯曲的镜子反射出的光线。

2) 一个来自圆柱镜子的反射的焦散。

3) 来自平面镜的二次照明（没有焦散的LSDE路径）。

图10-20c－e由增加发射出的光线数产生，并且表现出光线在漫反射表面散布的效果。随着光线路径上光线数量的增加，这些光线可以最终被合并，形成图像中的LSD路径。光线路径发出光线的数量分别为200、400和800。

正如Arvo（1986）所介绍的那样，二路光线跟踪方法很显然是第一个使用存储照明这一想法的算法。这一方法后来被Ward在他的RADIANCE绘制程序中所采用，它也是大多数现代的全局照明方法的基础（见10.9节）。我们已经第8章中介绍了光线图的概念。本章中提到的光线图与第8章中的光线图之间的差别在于其应用。在全局照明中，光线图被用作绘制过程的组成部分，使得解更有效、更可行。光线图在第8章中的应用是在实时绘制中绕过光线计算，其作用是作为把预计算的绘制操作带入到实时应用程序中的一种工具。

297 ↓ 299

10.8 依赖于观察/独立于观察和多路方法

在10.5节中，我们把路径跟踪作为一种实现完全L(DIS)*E相互作用的方法进行了介绍。但是，我们也指出，对于解全局照明问题来讲，这是一种高代价的方法。在本节中，我们将考察一些结合了现有的部分解的方法，如结合光线跟踪方法和辐射度方法，这些方法被称为多路方法。

在全局照明方法中，多路方法通常情况下是指一种依赖于观察的方法（辐射度方法）与一种独立于观察的方法（光线跟踪方法）的结合（尽管我们将二路光线跟踪分类为多路方法，但是，我们还是选择了将其作为光线跟踪方法的一种扩展的说法）。首先，让我们来考虑依赖于观察的方法与独立于观察的方法之间的差别意味着什么。依赖于观察的解正常情况下只代表依赖于观察的相互作用（纯的从漫反射到漫反射相互作用），因为这些解大部分都是将场景中每一个点上的光线级别写入到一个三维场景的数据结构中。然而，我们需要记住，从原理上来看，并没有什么理由阻止我们计算一个存储镜面相互作用的独立于观察的解。我们可能需要增加解的维数，以计算/存储表面上光线的方向以及光线的强度。我们将在10.11节回到这个话题。

纯的独立于观察的算法只计算足够的全局照明以便确定最后的图像，如果需要进行不同的观察，则算法会重新开始。这是很明显的。一个较细微的问题是，一般来说，依赖于观察的算法为每一个像素求一个独立的解。这对于漫反射相互作用的情况是一种浪费，因为在大的漫反射表面上照明变化很慢，正是基于这一观察导致了存储照明的想法。

另一方面，独立于观察的算法一般来讲代价更高一些，而且，从计算和存储的角度来看，在没有大量消耗的情况下，也不能处理以高频率变化的相互作用，比如镜面相互作用。多路算法通过对两种算法的结合而采纳了它们各自的优点。

一种常见的方法是用光线跟踪路径对辐射度算法的解进行后处理。于是，加入了镜面反射细节的独立于观察的图像就可以由此而获得。但是，这一图像并没有把所有的路径考虑进来。通过将辐射度方法与二路光线跟踪方法相结合，使得路径的类型由LS*DS*E扩展到了LS*(D*)S*E型，即包含了辐射度之后，把D分量扩展为D*。这就暗示着，它遵循对扩展的辐射度算法的排序。光源的光线跟踪第一次被采用，光线被从光源跟踪到经过所有的镜面传播一直到达一个漫反射表面为止，光线的能量被存储起来。这样就计入了LS*路径。然后，用这 300
些值作为发射曲面片启动一个辐射度的解过程，所存储的能量被通过D*链分布出去。最后启动目光的路径，这样就提供了最后的投影和ES*或ES*D路径。

将字符串LS*(D*)S*E与完整的全局解相比较，我们看到，中间的D*路径可以扩展成(D*S*D*) *,形成了LS*(D*S*D*)*S*E路径，而这一路径等价于完全的全局解L(S|D)*E。传统的或经典的辐射度方法并不包括从漫反射到漫反射的传播，这种传播是通过一个中间的镜面反射发生的。换句话说，一旦我们启动了辐射度算法，则需要包含一个通过中间镜面路径DSD传播的可能性。

包含一个镜面传播到辐射度解的第一种也可能是最简单的方法是基于对经典的辐射度算法的修正，使其可应用于平板的镜面表面（例如镜面），这种方法称为虚拟窗口方法。其思想如图10-21所示。对于传统的辐射度方法，光源、地板以及LDE路径之间的几何关系是通过两个表面之间的漫反射到漫反射相互作用来计算的（请注意，由于光源本身是一个发光的漫反射曲面片，所以我们把这个路径称为LDE或者DDE）。还没有考虑到的问题是来自LSD或DSD路径上的光线能量的贡献，这一贡献将会在地板上照出一片明亮的区域。从光源经过镜面到地板的DSD路径可以通过构建一个虚拟环境来计算，该虚拟环境是通过作为一个窗口的镜面被“看到”的。这时，虚拟光源的行为就好像它是从镜面反射出的真实的光源。但是，我们还需要考虑LSE路径，这是在镜子中形成的详细的反射图像。它是依赖于观察的，需要在第二个路径光线跟踪阶段确定。事实是，这一算法只处理一种特殊的情况，这也表明把辐射度

方法扩散到包含其他传播机制时其内在的困难。

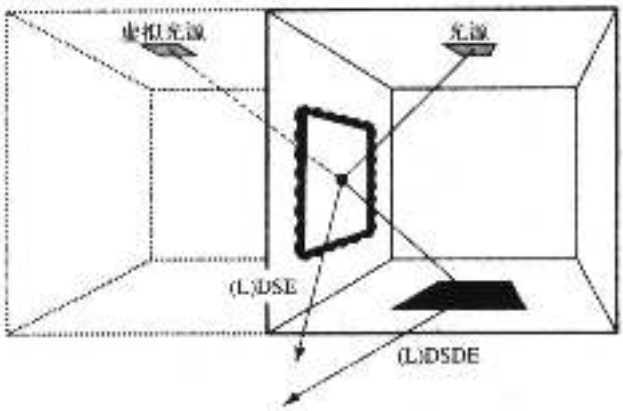


图10-21 在辐射度算法中结合DSD路径的虚拟环境方法

10.9 存储照明

存储照明是指我们随着解过程的进行将照明的三维或五维的值存入到与场景相关的数据
301 结构中去的一种方法。这样一种方法通常与依赖于观察的算法有关。换句话说，所获取的值用于加速或增加解的精确度，它们本身并不会构成依赖于观察的解。我们可以将这一算法与独立于观察的算法（比如辐射度方法）相比较，辐射度方法自身就高效地获取了离散化表面的最终的照明值。这样的算法与将在本节中介绍的存储算法之间的差别是存储方法是独立于表面的。这意味着避免了表面离散化方法（第11章）中所固有的网格化问题。表面上的照明值被存储在像八叉树这样的数据结构中，八叉树代表场景的整个三维空间上的分布。

再次考虑辐射光亮度方程的简化形式：

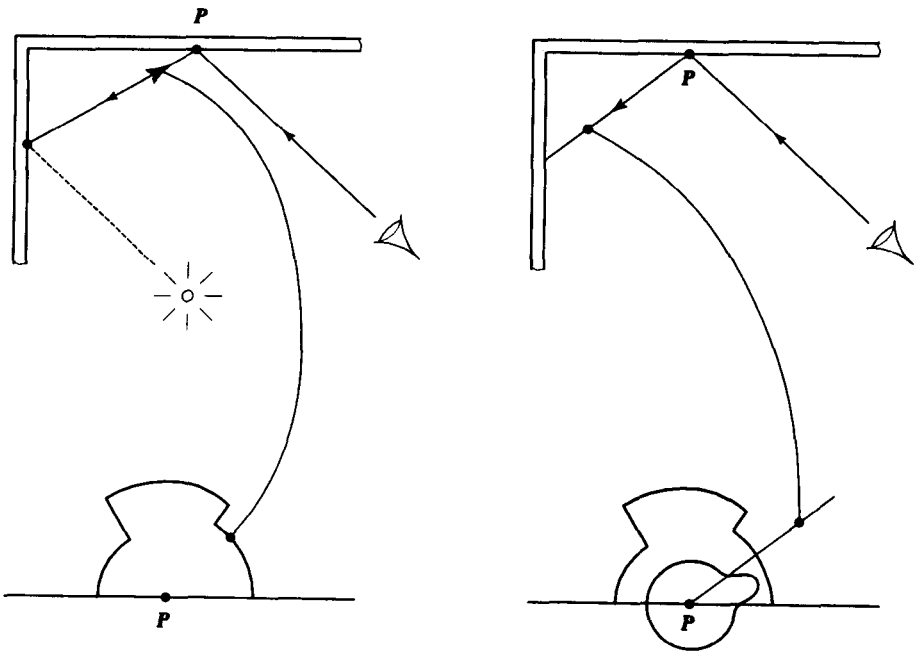
$$L_{surface} = \int \rho L_{in}$$

BRDF是已知的，L_{in}未知。正如我们第10.4节中所指出的，正是这一点限制了重要性采样的有效性。随着解的进行，可以获得对L_{in}的一个估计，这就需要将该值存储起来。该估计值可以用来促进重要性采样。这正是Lafortune and Williams（1995）在他们称其为自适应重要性采样的技术中所采用的方法。他们的方法是一种高效的路径跟踪算法，该方法利用以前计算的辐射光亮度值来引导当前的路径。其思路如图10-22所示，从图中可以看到，在路径跟踪中根据某点的BRDF和该点的场辐射光亮度分布函数的当前值选择了一个反射方向，而这个当前值是根据前面的值建立起来的。Lafortune and Williams（1995）将辐射光亮度的值存储到一棵五维树中，这是一个（三维）八叉树的二维扩展。

RADIANCE绘制程序可能是最众所周知的全局照明绘制程序。它由Ward（1994）用了9年的时间建立起来，该程序是基于路径跟踪的一种策略，在大多数条件下可解绘制方程的一种版本。这一工作特别强调在各种各样的光照条件（这些光照条件包括从光线到复杂的人工照明布局）下结构模拟所需的精度。该算法是确定性方法和随机方法的一种有效的结合。Ward（1994）描述了他建立这种方法的动机：

快速收敛的关键是确定采用哪一类样本，方法是：去掉那些可以用确定性方法

计算的积分部分，对其他部分的重要性进行调整，使我们的光线计算获得最大的收益。



a) 在点*P*处入射的辐射光亮度被存储到五维树中，并建立到一个分布函数中

b) 基于BRDF和场辐射光亮度分布函数选择了从*P*点的一个进一步反射的方向

图10-22 路径跟踪中的自适应重要性采样（Lafortune and Williams（1995））

镜面计算部分被分离出来，单独进行，核心的算法只处理间接的漫反射相互作用。由完全的漫反射相互作用产生的值被存储到一个（三维）八叉树中，这些所存储的值用于在一个当前的碰撞点足够靠近存储点时插值一个新的值。通过对当前被计算的区域确定“辐照度梯度”的方法精心地执行这一基本的算法，将导致在插值时使用较高阶（三次）的插值过程。RADIANCE绘制程序是一个路径跟踪算法，如果所存储的值足够接近，则该计算早早就结束了。

最后，Ward对于辐射度方法的实际效率给出了他的观点。在计算机图形学的文章中出现这样的批评意见是不寻常的，Ward一般性地认为，辐射度方法并不是从研究室引入进来的。他说到：

> 例如，大多数辐射度方法都不是很自动的，也不允许采用通用的反射模型或曲面……对于基于物理原理的绘制过程要接纳该方法的话就应该对其进行改进，但是，研究人员应该首先展示他们的技术的真实的应用。在应用辐射度方法以满足实际设计者的需求方面确实有一些成功的例子。有很多的研究工作都是基于改进基本的辐射度方法的效率，而与更真实、更复杂的几何学相关的问题只在最近才得到了其应有的注意。不管出于什么原因，辐射度方法已经实现了其承诺，现在是重新从现实应用的角度来考察这一技术以及其他解绘制方程的其他方法的时候了。

使用RADIANCE绘制程序的一个例子在第18章关于比较图像研究中给出（图18-19）。

10.10 光线体

302 ~ 303

光线体是指通过存储整个空间（包括空白的空间）的样本点处的辐射光亮度或辐照度的值来获取独立于观察的全局照明的方法。这样看来，它们与前面提到的方法有所不同，前面的方法只存储表面上点的值。在第16章中我们还会遇到光线体（那里称其为光线场）。光线场与光线体是相同的，其差别只是应用的不同。它们都是用于高效地存储对场景的独立于观察的绘制的预计算值。但是，在全局照明方法中，它们用于以某种方式得到一个解。

Greger等（1998）描述了光线体应用的一个例子。其思想是在“半动态”的环境中使用全局照明的解。这样一种环境被定义为与静态的物体相比移动的物体小，这就意味着移动的物体在场景中的位置不会对全局照明的解有很大的影响。在光线体中建立一个全局解，而这个解又用于确定移动的物体所接收到的全局照明。事实上，移动的物体在静态的光线体中移动，并接收照明，但是对于预处理的解并没有贡献。

10.11 粒子跟踪和密度估计

在本章的最后，我们将考察一个最新的方法（Walter等 1997），该方法的神奇之处就在于意识到将全局问题分解成光线传输和光线表示的计算的优点。在这一方法中，并不是存储照明，而只是存储粒子的历史。这样处理的原因是光线的传输，即光线在表面之间的流动，具有高的全局或界面间的复杂性。另一方面，对于物体表面上光线的表示又有高的局部或界面内的复杂性。表面可以表现出阴影、镜面高光焦散等现象（这些现象的一个好例子是辐射度算法，在这个算法中，传输和表示被合并成了一个过程。而困难在于要对输入到算法中的所需的网格的预测，例如，要定义阴影的边界。也就是说，我们必须在有光线传输的解之前确定网格）。

将这个过程分成三个连续的阶段：

- **粒子跟踪**。在某种意义上，这是路径跟踪的一种独立于观察的形式。携带光线的粒子被从每一个光源发射出去，并在环境中旅行。当每一次碰撞到表面时，就记录下这一碰撞（表面标识符、碰撞点、粒子的波长），并根据表面的BRDF计算其反射/折射的方向。每一个粒子都对它碰撞到的每一个表面产生一个信息列表，直到相互作用最小粒子过程终止或它被吸收为止。这样，粒子的路径就产生一个相互作用的历史，而不是返回一个像素的光强度。
- **密度估计**。在粒子跟踪过程完成之后，每一个表面都具有一个粒子的列表，所存储的碰撞点用来按照碰撞点的空间密度建立起表面上的照明。这一过程的结果是一个Gouraud明暗处理的三角形网格。

304

- **网格优化**。结果是独立于观察的。第三个阶段优化或大大地减少网格，需根据删除的网格所导致的变化不会低于阈值之下（从感观的基础上）的原则来删除网格。这一阶段的输出是一个不规则的网格，其细节与光线在表面上的变化有关。

Walter等（1997）指出，这一技术的一个很重要的优点是其模型化使得对于不同目标的优化成为可能。例如，光线传输阶段可以根据BRDF所需的精确度进行优化。密度阶段可以根据

感觉的精确程度来改变其判据，而毁灭（decimation）阶段可以在保持感觉质量的情况下获得很高的压缩比。

这一方法目前的缺点是它是一个三维的独立于观察的解，而这就意味着它只能显示从漫反射到漫反射的相互作用。然而，Walter也指出这一限制源自密度估计阶段，粒子跟踪模块可以处理任何类型的BRDF。 305

第11章　辐射度方法

11.1　辐射度理论
11.2　形状因子的确定
11.3　Gauss-Seidel方法
11.4　观察部分解——渐进式细化
11.5　辐射度方法存在的问题
11.6　辐射度图像中的人工痕迹
11.7　网格化策略

引言

光线跟踪方法，包含全局交互的第一个计算机图形学模型，至少就其一个侧面而言，遇到了一个标识可见信号的问题：你可以用光线跟踪方法辨别出图像是否被合成了。它只能建模光线相互作用中的一类，也就是完全镜面反射和折射。而漫反射表面间的相互作用是用环境常数来建模的（即在模型的局部反射分量中），而这种相互作用却可能是内部的主要光线传播机制。例如，考虑一个有墙和天花板的房子，用一种混合材料和地毯装饰。如果在房间里没有镜面反射的物体，则那些不能看到光源的房间中的部位只能靠漫反射光束照明。这样的房间倾向于在其表面上有一种缓慢的、细微的变化。

在1984年，康奈尔大学的研究人员采用了一种方法，即辐射度方法（Goral等 1984），这一方法的理论基础是辐射热传递的原理。这一方法现在被称为经典的辐射度方法，它模拟LD*E路径，也就是说，只能以不扩展的形式将其用于全部由完全漫反射表面组成的场景的绘制中。

为了实现这一目的，将场景分解成称为曲面片的元素，并基于光能的守恒原理建立起一组方程。在这样的环境中，一个曲面片反射从环境中每一个其他曲面片接收的光线。如果它是一个光源的话，它也发射光线，光源被处理成除具有非零的自发射的曲面片之外任何类型的曲面片。曲面片之间的相互作用依赖于几何关系。也就是依赖于距离和相对方向。两个平行的且相距很近的曲面片，相互作用较大。对于环境中的每一个曲面片，如果我们计算它与环境中的其他每一个曲面片之间的相互作用，则可能得到一个平衡解。

康奈尔小组的一个主要贡献是找到了一种有效的途径来计算曲面片对之间的几何关系——半立方体算法。事实上，在20世纪80年代，关于辐射度方法的大多数革新都是来自这一工作小组。

算法的代价是$O(N^2)$，其中N为曲面片数，环境被分成N个曲面片。为了使处理过程的成本降低，曲面片分得很大，而在一个曲面片上，假定光强度是一个常数。这样处理马上就出现了质量问题，即如果照明的不连续性与曲面片边界不一致，则会出现人工的痕迹。这种尺寸的限制是算法只能够计算漫反射相互作用的实际原因，因为漫反射的本质是在表面上变化缓慢。把镜面相互作用加入到辐射度方法中的代价是很大的，因此，这仍是一个正在研究的问题。于是，我们就处在一种很奇怪的状况中，即两个全局相互作用方法（光线跟踪方法和辐

射度方法）都是专门用于它们计算中所关心现象的。光线跟踪方法不能计算漫反射相互作用，而辐射度方法不能结合到镜面相互作用中去。除了这一点之外，辐射度方法产生了一些到目前为止在计算机图形学中最具真实性的图像。

在不进行进一步增强的情况下，辐射度方法就可以处理阴影。正如我们已经讨论过的，阴影的几何形状计算起来多少也是直接的，可以成为光线跟踪算法的一个组成部分，或者可以作为一种算法加入到局部反射模型绘制程序中去。但是，在阴影中光强度是漫反射相互作用的一个适当的部分，不可能由其他算法随意地近似。辐射度方法在其计算中考虑了阴影。该方法像其他方法一样按光强度输出其解。唯一的问题是，可能必须减小曲面片的尺寸，以便以所希望的精确度水平来刻画阴影的边界。阴影的边界是一些区域，在这个区域中漫反射光强度变化的速率快，而常规的曲面片尺寸可能在阴影的边界处引起可见的走样。

辐射度方法是一种物体空间的算法，它用来解离散的点或者环境中的表面曲面片的光强度问题，而不是用于解图像平面投影中的像素问题。因此，解是独立于观察者的位置的。然后，将这个完整的解注入到一个绘制程序中去，该绘制程序通过隐藏面消除和形成一个投影来计算一个特定的观察。该方法中的这一阶段并不需要很多的计算（已经计算了光强度），而
307 且从通用的解中很容易获得不同的观察。

11.1 辐射度理论

在本书的其他章节中，我们已经尝试着保持实现方法的算法与算法本身所隐含的数学之间的分离。但是，在这种情况下，对于辐射度方法来说，算法本身与其隐含的数学之间互相缠绕，以至于很难对其以一种分离的方式进行处理。该理论本身只包含了一些定义，而并没有涉及处理。希望对于其理论有进一步了解的读者可以参考Siegel and Howell（1984）的书。

辐射度方法是一种能量守恒或者叫做能量平衡的方法，它为一个封闭的空间中所有表面的辐射度提供一个解。对于系统的输入，能量来自那些作为发射器的表面。事实上，对光源的处理与算法中处理的其他表面是相同的，只是它具有一个初始的（非零的）辐射度。该方法基于这样的假设，即所有的表面都是完全漫反射的或为理想的Lambert表面。

辐射度B定义为单位时间单位面积上离开表面曲面片的能量，它是发射和反射的能量之和：

$$B_i \mathrm{d}A_i = E_i \mathrm{d}A_i + R_i \int_j B_j F_{ji} \mathrm{d}A_j$$

将这一方程用文字来叙述，则对于一个曲面片，我们有：

辐射度 × 区域 = 发射的能量+反射的能量

E_i是从曲面片发射出的能量。反射的能量由入射的能量R_i与曲面片的反射性相乘的乘积来表示。入射的能量是在环境中从所有其他曲面片到达曲面片i的能量。也就是说，对于所有的$j(j \neq i)$在环境中求积分，积分项为$B_j F_{ji} \mathrm{d}A_j$。这也是离开每一个曲面片j到达曲面片i的能量。F_{ji}是个常数，称为形状因子，它是曲面片j和i之间关系的参数化。

我们可以用一个相互作用关系给出：

$$F_{ij} A_i = F_{ji} A_j$$

通过除以$\mathrm{d}A_i$，得到：

$$B_i = E_i + R_i \int_j B_j F_{ij}$$

对于离散的环境，其积分由一个求和来替代，在一些小的离散曲面片上，假设辐射度为常数。于是有:

$$B_i = E_i + R_i \sum_{j=1}^{n} B_j F_{ij}$$

对于包围环境中的每一个曲面片都有这样一个方程，而完整的环境产生一组如下形式的n个同时存在的方程:

308

$$\begin{bmatrix} 1-R_1F_{11} & -R_1F_{12} & \cdots & -R_1F_{1n} \\ -R_2F_{21} & 1-R_2F_{22} & \cdots & -R_2F_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ R_nF_{n1} & -R_nF_{n2} & \cdots & 1-R_nF_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix} \tag{11-1}$$

对这个方程求解就是辐射度方法。这个解中的B_i是每一个曲面片的辐射度。但是，还有两个问题没有解决。一是需要一种计算形状因子的方法。二是需要计算一个观察，以显示曲面片。为了做到这一点，我们需要一个线性插值方法，就像Gouraud明暗处理方法那样。否则的话，细分出的形状即曲面片本身将是可见的。

只有在提供照明的那些表面上E_i才是非零的，而这些非零项代表向系统输入的照明。R_i是已知的，而F_{ij}是环境几何的函数。反射性是依赖于波长的项，上面的方程应看成是一个单色的解，也可以考虑通过解很多的颜色带来得到一个完整的解。到这时我们可能已经注意到，对于一个平板的或一个凹的表面，$F_{ij} = 0$，即没有离开表面的辐射会击中其自身。另外，从形状因子的定义出发，对任何一行的形状因子求和得到单位值。

因为形状因子只是系统几何学的一个函数，所以对其只计算一次。方法受到计算形状因子所需花费的时间的限制，形状因子表示出两个表面曲面片A_i和A_j之间辐射度的交换。这依赖于它们相对的方向及其之间的距离，由下式给出:

$$F_{ij} = \frac{\text{离开表面}A_i\text{直接击中}A_j\text{的辐射能}}{\text{在环绕}A_i\text{的半球空间所有方向上离开表面}A_i\text{的辐射能}}$$

可以看出，这就给出:

$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos\phi_i \cos\phi_j}{\pi r^2} \, dA_j dA_i$$

其中，几何规范示意图如图11-1所示。在很多的实际环境中，A_j可能从A_i处是全部或者部分可见的，积分值需要与一个阻断因子相乘，这个因子是一个二元函数，它依赖于微分区域dA_i是否可以看到dA_j。这个双重积分除了对特殊形状之外都是很难解的。

11.2 形状因子的确定

估算形状因子的一个简明的数值方法是在1985年提出的，这个方法称为半立方体方法。这是一个高效地确定形状因子的方法，而同时也为插入式曲面片的问题提供了解决方法。

从曲面片到曲面片之间的形状因子可以通过微分区域近似成有限区域方程:

$$F_{dA_i A_j} = \int_{A_j} \frac{\cos\phi_i \cos\phi_j}{\pi r^2} \, dA_j$$

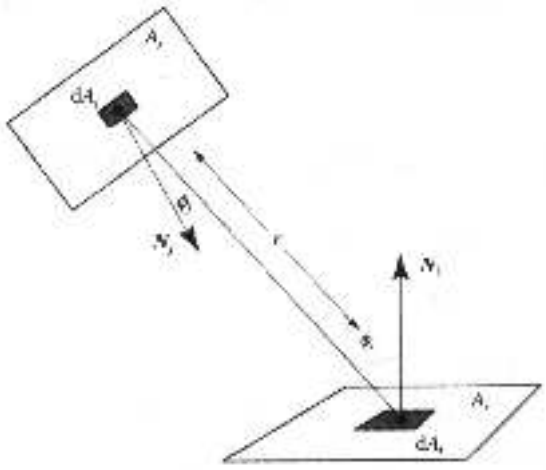


图11-1 对于两个曲面片i和j的形状因子几何（Goral等（1984））

这时我们考虑的是单元区域dA_i与有限区域A_j之间的形状因子。dA_i被定位在曲面片i的中心点处。这个近似值的变化依赖于两个曲面片的面积与两者之间距离r之比，如果r大，则在外部积分的范围之内，内部的积分变化不大，外部积分的作用只是乘以一个单位值。

一种称为Nusselt相似性的理论告诉我们，可以考虑一个曲面片j向一个围绕单元曲面片dA_i的半球的表面上的投影，这样做的效果与考虑曲面片本身是等价的。而且，在半球上产生相同投影的曲面片具有相同的形状因子，这是对于半立方体方法的证明，其示意图如图11-2所示。曲面片A、B和C都具有相同的形状因子，我们可以通过考虑其在半立方体表面上的投影而不是曲面片自身来估算任意曲面片j的形状因子。

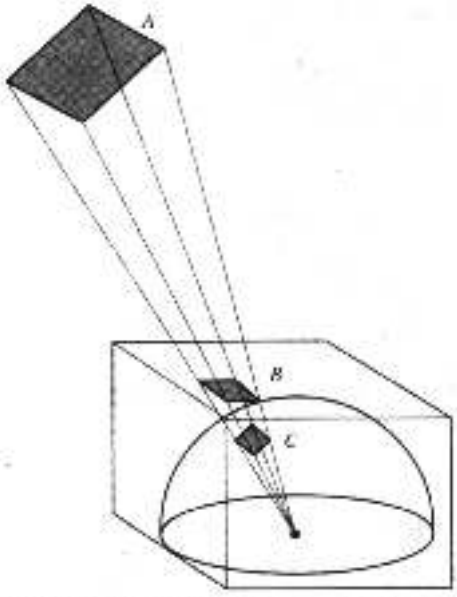


图11-2 采用半立方体的理由。曲面片A、B和C具有相同的形状因子

之所以使用半立方体来估计半球，是因为平板的投影平面计算上代价较小。在每一个曲面片的中心的周围建立起半立方体，并使半立方体的Z轴和曲面片的法线同线（见图11-3）。将半立方体的各个表面划分成像素，这个术语可能与我们前面用到的曲面片有些混淆，因为我们是在物体空间进行处理。环境中的任何其他曲面片都投影到整个半立方体上。两个投影到相同像素上的曲面片可进行深度比较，若再有更多的曲面片将被拒绝，因为这些曲面片从接收的曲面片上是看不到的。这个方法与Z缓冲器方法是相似的，只是在这一阶段没有光强度的相交。半立方体算法只是促进了形状因子的计算，形状因子接着将用于计算漫反射光强度，保留了一个"标签缓冲器"以便对最靠近半立方体像素的曲面片进行索引。

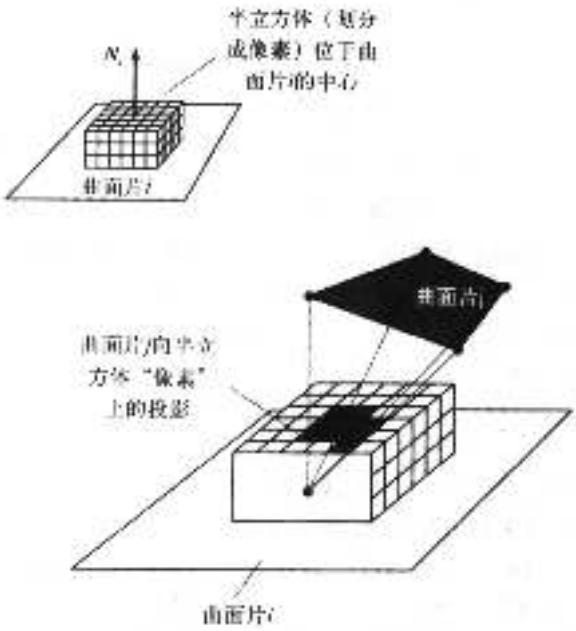


图11-3 通过把曲面片j投影到以曲面片i为中心的半立方体上来估算形状因子F_{ij}

309 / 311

半立方体上的每一个像素都可以看成是一个小曲面片。对于每一个像素定义一个对有限面积的形状因子的微分，称为Δ形状因子。像素的形状因子是对于曲面片的有限面积的形状因子微分的一个分数部分，可以定义为：

$$\Delta F_{dA_i,A_j} = \frac{\cos\phi_i \cos\phi_j}{\pi r^2}\Delta A = \Delta F_q$$

其中，ΔA是该像素的面积。

这些形状因子都被预先进行了计算，并存储在查询表中。这也正是半立方体方法效率高的基础。再一次利用这样的事实，即对于在以曲面片A_i为中心的接收表面具有同样的投影的面积都具有相同的形状因子，我们可以得出结论，对于任何曲面片来讲，其F_{ij}都可以通过对曲面片A_j投影到的所有像素的形状因子求和来获得（见图11-4）。

这样一来，形状因子的计算就降阶为对同一定向的平面的投影以及一个求和运算。

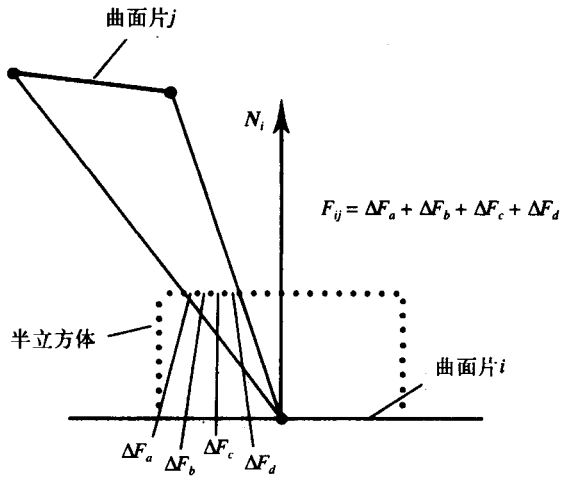


图11-4 F_{ij}通过对曲面片i投影到其上的像素的形状因子求和得到

图11-5（彩色插图）是一个有趣的图像，它显示了当场景中所有其他曲面片都投影到其上之后的一个放在窗户上的半立方体（图10-7）的状态。在可以被这个半立方体所看到的场景中，每一种颜色标识一个曲面片（以及每一个部分的曲面片）。然后，这个算法只是简单地将与每一个曲面片相关的所有半立方体元素的形状因子求和。

方法可以总结成如下的几个阶段：

1) 形状因子F_{ij}的计算。每一个半立方体的位置计算（$n-1$）个形状因子或者方程中的一行。

2) 解辐射度矩阵方程。

3) 通过将阶段2所得的结果注入到双线性插值算法中进行绘制。

4) 对于所感兴趣的颜色带重复步骤2和3。

312

这个过程如图11-6所示。形状因子只是环境的一个函数，并且只计算一次。对于不同的反射和光源值可以在步骤2重新使用。这样，对于相同的环境而出现如某些光源关闭时的情况也可以获得一个解。由第2步产生的解是一个独立于观察的解，如果需要有不同的观察，则只需要重复计算步骤3。这一方法可以用于如要在建筑物的内部产生动画的遍历时。动画中的每一帧都通过改变观察点来进行计算，并且从未经变化的辐射度的解来计算一个新的观察。只有当我们要改变场景的几何形状时，才有必要重新计算形状因子。如果改变了光照条件，而几何形状没有改变，则只需要重新解方程，而不必要重新计算形状因子。

步骤2暗示着对于辐射度方程需要计算一个独立于观察的解，这个解对于环境中的每一个曲面片给出一个值，即辐射度。根据这些值可以计算顶点的辐射度，在双线性插值算法中利用这些顶点的辐射度来提供最终的图像。在这一阶段使用了一种深度缓冲器算法来估算在屏幕上每个像素上的每一个曲面片的可见性（不应将这一阶段与半立方体的运算相混淆，因为半立方体运算必须在计算形状因子期间估算曲面片内的可见性）。

完成形状因子计算所花的时间依赖于曲面片个数的平方。在每一个曲面片上执行了一次半立方体计算（所有其他的曲面片都向其上投影）。于是，总的计算时间就依赖于环境的复杂性以及解的精确度，这一精确度由半立方体的分辨率来确定。尽管漫反射照明只在表面上缓慢地变化，但是，当半立方体的分辨率太低和阴影边界上需要的精确度高时也会引起走样（见11.7节）。存储的需求也是所需曲面片个数的一个函数。所有这些因素意味着，对于用辐射度方法可以处理的场景的复杂性有一个上限。

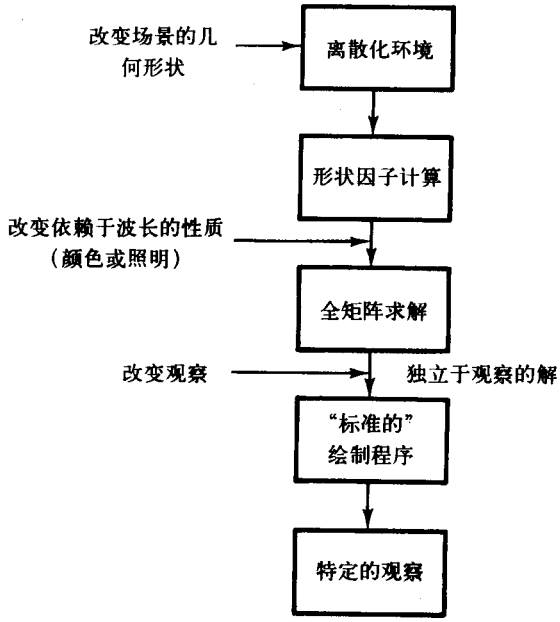


图11-6 一个完整的辐射度解中的各个阶段，所示的是那些可以对图像进行各种修改的过程中的点

11.3 Gauss-Seidel方法

Cohen and Greenberg（1985）指出，Gauss-Seidel方法可以保证对于像方程（11-1）这样的方程组有一个快速的收敛。任何一行形状因子的和从定义上来看都应该小于1，而每一个形状因子都与一个小于1的反射性相乘。于是，在方程（11-1）的行中各项的和便小于1（不包括主对角线上的项）。平均对角线项总是1（对于所有的i，$F_{ii}=0$），这些条件保证了快速收敛。Gauss-Seidel方法是下列迭代方法的一种扩展。已知一个线性方程组：

$$\boldsymbol{A}\boldsymbol{x}=\boldsymbol{E}$$

像方程（11-1）一样，我们可以对$x_1, x_2, \cdots, x_i$重写方程为如下的形式：

$$x_1=\frac{E_1-a_{12}x_2-a_{13}x_3-\cdots-a_{1n}x_n}{a_{11}}$$

这就导致了迭代：

$$x_1^{(k+1)}=\frac{E_1-a_{12}x_2^{(k)}-a_{13}x_3-\cdots-a_{1n}x_n^{(k)}}{a_{11}}$$

一般形式为：

$$x_i^{(k+1)}=\frac{E_i-a_{i1}x_1^{(k)}-\cdots-a_{ii-1}x_{i-1}^{(k)}-a_{ii+1}x_{i+1}^{(k)}-\cdots-a_{in}x_n^{(k)}}{a_{ii}} \tag{11-2}$$

可以将这一公式用于迭代过程中：

1）选择一个初始的近似值，比如：

$$x_i^{(0)}=\frac{E_i}{a_{ii}}$$

对于$i=1,2,\cdots,n$，其中E_i只对于发射光线的表面或者光源为非零值。

2) 用方程（11-2）确定下一个迭代:

从 $x_i^{(k)}$ 确定 $x_i^{(k+1)}$

313
~
314

3) 如果 $\left|x_i^{(k+1)}-x_i^{(k)}\right|<a$ 阈值，对$i=1,2,\cdots,n$，则停止迭代，否则返回步骤2。

这个过程称为Jacobi迭代。Gauss-Seidel方法通过对方程（11-2）进行调整使其利用最新的有用信息而改进了Jacobi方法的收敛性。当计算新的迭代 $x_i^{(k+1)}$ 时，其新值:

$$x_1^{(k+1)},x_2^{(k+1)},\cdots,x_{i-1}^{(k+1)}$$

已经被计算出来了。方程（11-2）被修改为:

$$x_i^{(k+1)}=\frac{E_i-a_{i1}x_1^{(k+1)}-\cdots-a_{i,i-1}x_{i-1}^{(k+1)}-a_{i,i+1}x_{i+1}^{(k)}-\cdots-a_{in}x_n^{(k)}}{a_{ii}} \tag{11-3}$$

注意，当$i=1$时，方程的右边只含有上角标为k的项，方程（11-3）就降为方程（11-2）。当$i=n$时，方程右边只含有上角标为（$k+1$）的项。

Gauss-Seidel方法的收敛性可以通过下面的方法得到改善。在产生了一个新值 $x_i^{(k+1)}$ 之后，由旧值和新值的一种加权平均可以给出一个较好的值:

$$x_i^{(k+1)}=rx_i'^{(k+1)}+(1-r)x_i^{(k)}$$

其中，r (大于0)是一个不依赖于k和i的参数。Cohen等（1988）指出，松弛系数1.1可以用于大多数环境。

11.4 观察部分解——渐进式细化

在实际过程中运用辐射度方法，比如说在对建筑物内部进行设计时，设计者必须等待很长时间才能看到一个完整的结果。这是一个缺点，因为采用基于计算机的设计的理由之一就是允许用户自由、快速地选用各种设计参数。而长的反馈时间阻止了这种实验，也妨碍了设计的进程。

1988年康奈尔工作组建立了一种方法，称为“渐进式细化”方法，它允许设计者看到一个早期的（但是近似的）解。在这一阶段，可以看出主要的设计错误，并对其进行改正，最后得出另一个解。随着解变得越来越精确，设计者可能看到更多必须进行改动的地方。我们已经在上一章中介绍了这一方法，现在对其进行详细的讨论。

渐进式或自适应细化方法的总体目标可以通过采用任何慢速图像合成技术来达成，该技术是希望找到在对交互性的图像质量的需求竞争之间达到的一种折中。一种提供自适应细化的合成方法可能会为用户提供一个迅速绘制的图像。然后，对这个图像以一种“优雅”的方式进行渐进式细化。这一过程被定义成一种朝着较高质量、较好的真实性等方向的进步，它是以一种自动的、连续的、不使用户分散注意力的方法进行的。较早地提供一种近似图的能力大大地辅助了技术的进步和图像的改进，通过近似也减少了循环反馈，这也是对于辐射度
315 方法的必要的减负。

在辐射度方法中，两个主要的高代价因素是存储成本以及对形状因子的计算。对于一个具有50×10^3个曲面片的环境，尽管所产生的形状因子的矩阵有90%是稀疏的（许多曲面片互相之间看不到），但也还是需要10^9字节的存储空间（每四个字节一个形状因子）。

对于渐进式细化以及消除形状因子的预计算和存储的需求可以通过对基本的辐射度方法的一种巧妙的重新布置而得到满足。在渐进式细化方法中，各个阶段是通过作为随着迭代的解进行的结果的显示而获得的。对解进行重新构建，对形状因子的估算顺序进行优化，以便使收敛过程可以“可视地优雅前行”。这种重新构建使得解中的每一步都对所有曲面片的辐射度进行更新，而不是对一个单一的曲面片提供解。解的步骤之间的最大的可见差别可以根据其对环境的能量贡献来处理曲面片得到。辐射度方法特别适合于渐进式的细化方法，是因为它计算的是一种独立于观察的解，观察这个解（通过由特定的观察点进行绘制）的过程可以随着辐射度解的过程而独立地进行。

在计算辐射度矩阵的传统过程中（例如，用Gauss-Seidel方法），对于一行的解，为单个曲面片i提供辐射度：

$$B_i = E_i + R_i \sum_{j=1}^{n} B_j F_{ij}$$

这是基于当前所有其他曲面片的估计值对于曲面片i的辐射度的估计值。对于曲面片i的辐射度的估计，称为“收集”。该方程意味着（从算法本身来讲），对于曲面片i，我们访问到了场景中的每一个其他曲面片，并根据形状因子从每一个曲面片j向曲面片i传送适量的光线。算法逐行进行，在对矩阵的一次全部计算之后更新一个完整的解（尽管Gauss-Seidel方法只要计算出值，这个值就被采用）。如果动态地观察这个过程，就会看到，随着解过程的进行，每一个曲面片的光强度都根据辐射度矩阵中行的位置进行更新。从场景中的每一个其他曲面片上收集光线，用于对当前处理的曲面片进行更新。

渐进式细化方法的思想是，在每一次迭代中对所有曲面片的整个图像进行更新。这个动作称为“发射”，这时来自每一个曲面片i的贡献被分布到所有其他曲面片上。这两个动作之间的差别示于图11-7a和b中。这个算法的重新排序是按照下列的方法完成的。

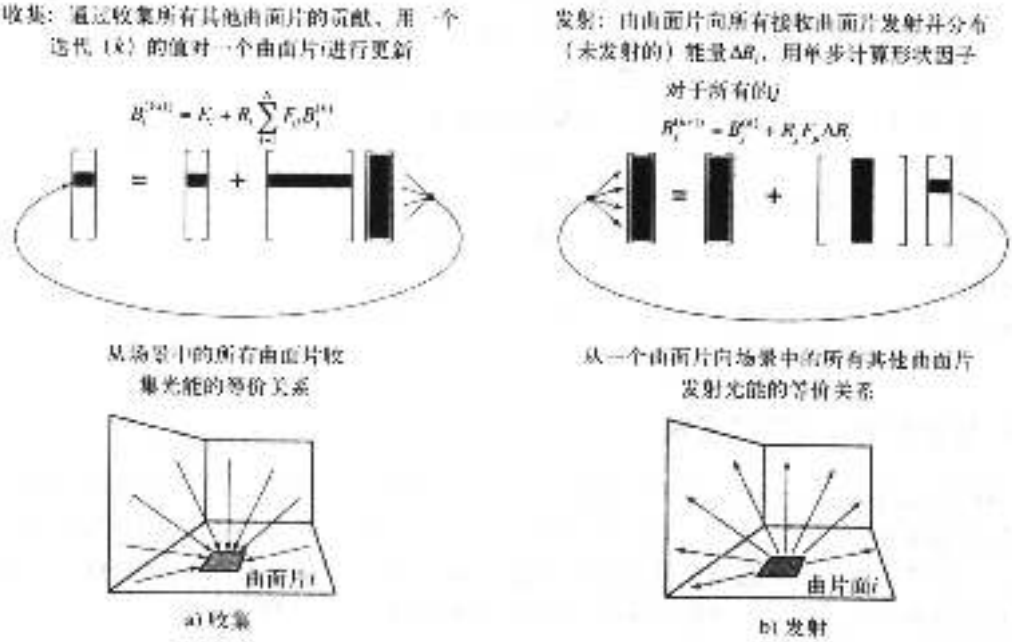


图11-7 辐射度解策略中的收集和发射

一个项确定由于曲面片i而对曲面片j所做的贡献:

$$B_j \text{ 由于 } B_i = R_j B_i F_{ji}$$

这一公式可以用互逆性关系求反关系:

$$B_j \text{ 由于 } B_i = R_j B_i F_{ij} A_i / A_j$$

这一关系对于所有的曲面片j都成立。可以将这个关系用于确定环境中由单个曲面片i对每一个曲面片j的贡献。一个辐射度（曲面片i）将光线发射到环境中，同时，对所有曲面片j的辐射度进行了更新。第一个完整的更新是由形状因子计算获得的（对于环境中所有的辐射度）。于是，对于完整场景的初始近似就可以在只计算了第一行的形状因子时就出现。这就削减了启动或预计算的高成本。

这一过程一直重复进行，直到收敛为止。所有辐射度值的初值或者设为零，或者设为其发射值。随着这一过程对于每个曲面片i的重复进行，解就显示出来。在每一步上，每个曲面片j的辐射度都被更新。随着解的进行，对于曲面片i的辐射度的估计越来越精确。对于迭代，环境中已经含有了以前对B_j估计值的贡献，而所谓的“未发射”辐射度，即当前估计值和以前估计值的差别就是被注入到环境中去的所有内容。

如果算法的输出在不经过进一步精细处理的情况下被显示出来，则开始时的黑暗场景会随着辐射度的增量被逐渐加入到每个曲面片上而逐渐明亮起来。这一过程的“可视化收敛”可以通过对曲面片处理的顺序根据它们将要辐射的能量进行排序的办法进行优化。这意味着，比如像发射光线的曲面片或者光源可以首先得到处理。这就给出了一个早期的、具有良好照明的解。下一个将要处理的曲面片是那些从光源中接收最多能量的曲面片，如此继续。通过采用这种排序方法，解过程的进展就以一种光能在环境中增值的近似的方式进行。尽管这样做之后会产生比未经排序的过程要好的可视化序列，但是解仍然是从一个黑暗的场景到一个完全照明的场景的渐进过程。为了克服这一作用，对于中间辐射加入了一个任意的环境光项。这一项仅用于增强显示，它并不是解的组成部分。环境项的值是基于环境中所有曲面片辐射度的当前估计值的。随着解的进行，光照条件变好，这一环境项的贡献就降低了。

316
~
317

在算法中，每一次迭代由四个主要的阶段来完成。这些阶段是:

1) 求出具有最大（未发射）辐射度或发射能的曲面片。

2) 计算列形状因子，也就是说，从这个曲面片到环境中所有其他曲面片的形状因子。

3) 更新每一个接收曲面片的辐射度。

4) 降低临时环境项，它是以步骤3中计算的当前值与以前值之间差值之和的一个函数的形式进行的。

在执行期间，渐进式细分方法的一个例子如图10-7所示，在10.3.2节中对这个图有详尽的说明。

11.5 辐射度方法存在的问题

与辐射度绘制方法相关的主要问题有三个。这些问题是图像中出现的算法的人工痕迹、不能处理镜面相互作用以及在绘制一个中等复杂度场景时所需花费的不适当的时间。很奇怪的是，几乎没有什么研究致力于解决时间问题，而这也可能正是辐射度方法没有被广泛地移植到应用程序中去的原因。这种情况与在20世纪80年代的光线跟踪方法正好相反，因为那时很快就出现了第一个用光线跟踪方法给出的图像。当时，有很多、很强的研究动力来使该算

法变得更快。在这一章的后面几节中，我们将主要关注图像的质量，而不再关注执行时间。质量可以通过更精确地定义场景来得到改善，这在主流的方法中意味着程序中允许有更多次的迭代。

除了前面章节中所描述的技术之外，辐射度方法的发展大部分都是由一组较大的曲面片来表示场景而产生了缺陷或人工痕迹所激发。尽管也有其他因素，像考虑在大气中的扩散，以及将镜面反射结合进来等也是重要的，但是，由于建立网格而产生的可视化的缺陷被大多 318
数的研究工作所强调，正是由于这一点，我们将只讨论这种缺陷。

11.6 辐射度图像中的人工痕迹

用前一章中所述的经典方法所产生的辐射度图像中的常见人工痕迹是由于:

1) 为了确定形状因子而在半立方体方法中所做的近似。

2) 用双线性插值作为由恒定的辐射度解对辐射度函数的重新构建。

3) 采用了对场景的网格化或划分，它独立于辐射度函数的变化性质的划分。

当然，可见性和可见性的重要性依赖于场景的性质，但是，通常第三类缺陷更引人注意也是最难处理的。在实际过程中，不能独立地处理这些人工痕迹，很少有人认为需要建立一种强有力的网格化策略而不同时处理由于双线性插值而造成的人工痕迹。现在，我们将来考察这些图像的人工痕迹，对两种原因都加以讨论，并寻找可能的解决方法。

11.6.1 半立方体人工痕迹

半立方体方法的严重问题是由将半立方体规则地划分成均匀的像素而造成的走样。误差是作为一个半立方体像素尺寸的函数形式出现的，这是因为假设曲面片将正好投影到整数个像素上，而一般来讲，这种情况当然是不会发生的。这与光线跟踪时的走样相似。我们试图通过向一个固定数量的方向进行观察来获取三维环境的信息。在光线跟踪中，这些方向初始时由均匀放置的目光到像素的光线给出。在辐射度方法中，通过将曲面片投影到半立方体上，高效地从半立方体的原点处对投影光线进行采样。图11-8显示的是一个该问题的二维模拟，图中一些相同的多边形投影到一个或两个像素上，投影到的像素数依投影光线和多边形网格之间相互遮挡而定。多边形的大小是相等的，且相对于曲面片i来讲具有相同的方向。它们的形状因子应该是不同的，因为，对于每一个多边形，每一个多边形投影的像素数是不同的。但是，正如从例子中看到的，应该具有相同形状因子的相邻多边形将以2：1的比例产生值。

任何实际场景的几何形状都可能在利用半立方体方法时引起问题，其精确度依赖于计算所涉及的曲面片之间的距离。当距离变小时，方法就会失败。例如，当将一个物体放在一个支撑表面时，实践中就会出现这种情况。形状因子的误差会准确地在一些特定的区域出现，在这些特定的区域我们可能期望辐射度技术能够更优越，并产生像颜色泄漏和软阴影等细微的现象。Baum等（1989）对确定靠近中心的表面的形状因子时所涉及的误差进行了量化，并表明半立方体方法只有在曲面片之间的距离至少五倍于曲面片直径的情况下才是精确的。

然而，在处理光源时出现另一个半立方体方法的问题。在使用了辐射度方法进行绘制的场景中，我们通常考虑的是像荧光光源这样的区域光源，像对环境中的任何其他表面一样，我们把光源划分成曲面片，而因此就埋下了问题的种子。对于标准的解，环境将被离散化成曲面片，对于曲面片的细分程度依赖于表面的区域（以及所需的解的精确度）。然而，对于光源，所需的半立方体的个数或所需的曲面片的个数依赖于与其最靠近的照明表面的距离。半

立方体的运算有效地将发射曲面片降为点光源。如果光源不能充分地细分，则作为光线的孤立区域，封闭的表面将出现误差。对于条状光源，由于其长宽比较大，所以不充分的细分可以引起带状的或走样的人工痕迹，这种痕迹平行于光源的长轴。如图11-14所示为不充分的光源细分的一个例子。

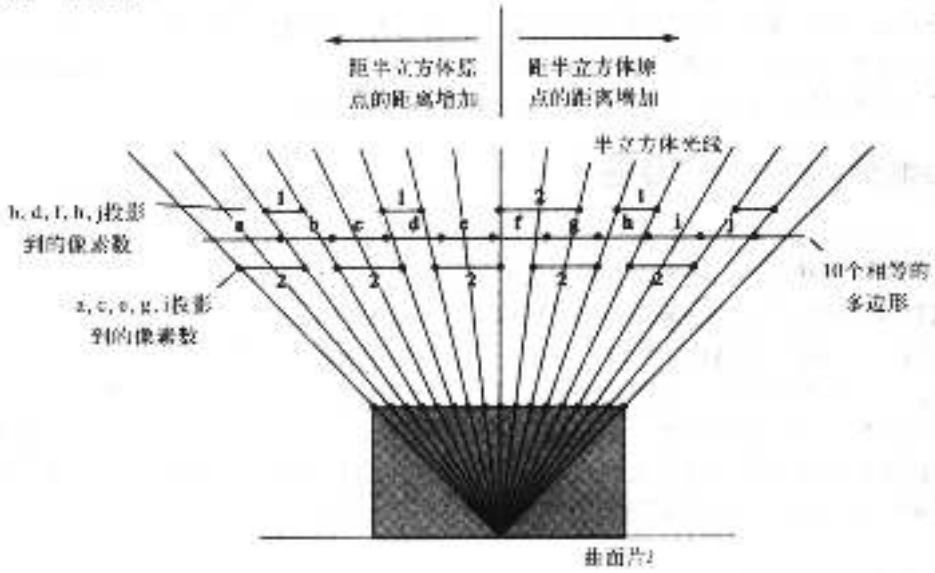


图11-8 半立方体采样和一组相等的多边形之间的相互干扰（Wallace等（1989））

当然，半立方体走样可通过增加半立方体的分解来得到改善。但是，这是不够的。增加对场景中所有元素所需的计算并没有涉及到对它们的变形是否是由半立方体造成的分析。对于传统的（上下文无关的）反走样（见第14章）也出现同样情况。

319
/
320

问题来自于近似（见前面的章节）：

$$F_{ij} = F_{dA_i A_j}$$

半立方体由一个微分面积（实际是一个点）到一个有限的面积计算形状因子。这样做有两个结果，图11-9示意了对于互相穿透的曲面片可能产生的问题。在这里，计算了从曲面片i到曲面片j的形状因子，就好像互相穿透的曲面片不存在一样。这是因为从半立方体的原点处可以完全看到曲面片j。

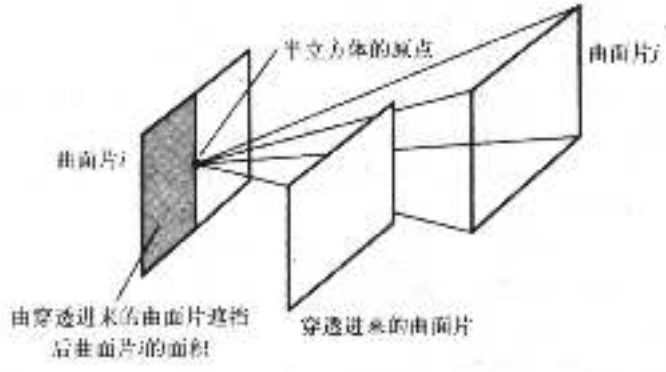


图11-9 从半立方体的原点处可以看到所有的曲面片j，$F_{dA_i A_j}$近似失败

最后，考虑半立方体采样的有效性。在法线方向从半立方体可以“看见”的曲面片比水平方向的曲面片更重要（它们投影到那些具有较高形状因子的增量的半立方体单元上）。如果我们考虑把计算以重要性采样为基础均匀地分布，则那些越靠近水平线的单元重要性就越小。在Max and Troutman（1993）中报告了一个例子，导出了最优的分解、形状以及网格单元的分布。在这一工作中，他们建议顶面的分解比侧边的分解高出40%，而且边的高度是宽度的70%。注意，这样做也会导致由均匀的半立方体单元引起的走样的减少。

11.6.2 重建人工痕迹

正如标题所言，重建人工痕迹是由于绘制或近似的方法的性质所决定的，也就是从常量的辐射度解对连续辐射度函数重建近似的方法的性质。回忆一下，辐射度方法只能在常数辐射度的假设条件下才能工作，也就是说，我们把环境划分成曲面片，以每一个曲面片上的辐射度都是常数为基础来解方程组。

在图11-10中对最常见的方法——双线性插值方法进行了总结。其中我们假设如图11-10a所示的曲面将表现出其辐射度的值沿着点划线而连续变化。在辐射度方法中，第一步是计算一个常数的辐射度解，这将导致对于连续函数的一种台阶式近似。通过对共享一个顶点的曲面片的辐射度求平均值来计算该顶点处的辐射度值（见图11-10b）。然后，再把这些值注入到双线性插值算法中，表面再用Gouraud明暗处理方法进行明暗处理，导致分段线性的近似（见图11-10c）。

321

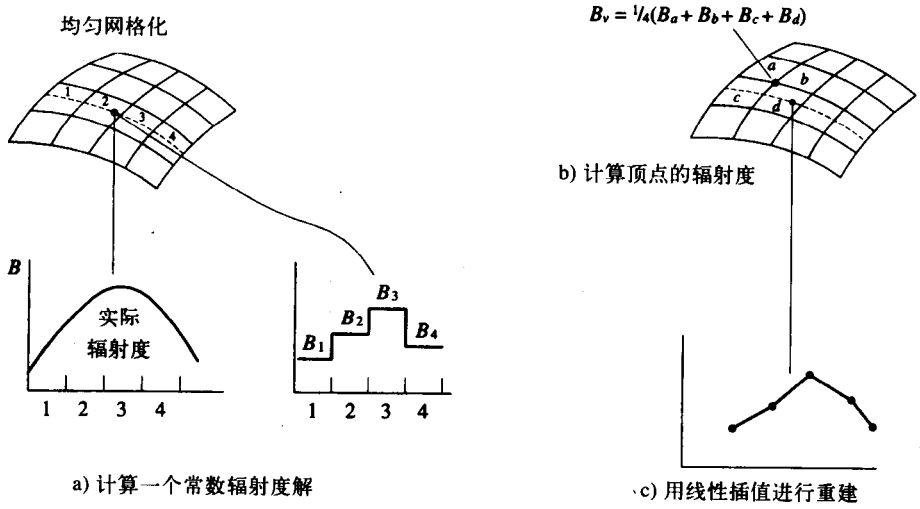


图11-10 辐射度方法中所用的常规重建方法

从这个方法产生的最引人注目的缺陷就是马赫带，当然，这种情况在一般的Gourand明暗处理中已经遇到过了，一般的Gouraud明暗处理也使用相同的插值方法。这种“可见的重要性”可以通过采用纹理映射来减低，但是，在辐射度应用程序中这可能会成为一个问题，因为很多有这种表现的区域都是没有纹理的表面，比如建筑物的内墙。对网格细分的策略也会减少马赫带的可见性，因为该方法降低了元素的尺寸，这也就减小了顶点辐射度之间的差别。

322

更先进的策略包括表面插值方法（见第3章）。在这里，辐射度的值被处理为连续辐射度

函数的样本，并将二次或三次Bézier/B样条曲面片网格配合到这些样本中。这一方法的明显困难（即其与生俱来的高成本）以及防止所需要的不连续性被平滑掉等意味着大多数最流行的重建方法仍然是线性插值。

11.6.3 网格化人工痕迹

辐射度方法最困难的问题之一，因此也是目前的主要研究领域，是关于网格化的问题。在上面的讨论中，我们已经简单地讨论了作为实体的曲面片问题，场景被划分成曲面片，其限制条件是曲面片应大到能够找到一个不是令人望而却步的高代价的解。然而，我们进行这一工作的方式还要顾及到最终图像的质量问题。应该如何完成这一工作以使得人工痕迹达到最小呢？这个问题之所以困难，是因为只有当有了一个解的时候才能进行这一工作，以便使我们能够看到什么地方出现了问题。另一种选择是我们必须能够预测出什么地方可能会出现问题，然后根据预测结果进行划分。我们先从考察网格化人工痕迹的性质及其起源开始。

首先给出一些术语：

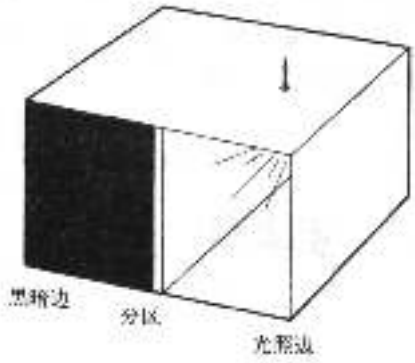
- **网格化** 这是一个通用的术语，在辐射度方法中用于描述初始场景细分，或者描述可能在程序的执行过程中发生的进一步细分的动作。“初始场景细分”的结果可能是一个通用的场景数据库，该数据库是辐射度绘制程序的输入，它并不是必须创建的。然而，我们马上就会看到，它更像是一个这种数据库的预处理的版本，或者是一个专门为辐射度的解而建立的场景。
- **曲面片** 曲面片是场景的初始表示中的实体。在标准的辐射度解中，解的过程会出现细分，而曲面片形成对程序的输入。
- **元素** 元素是由曲面片的分解而产生的。

网格化人工痕迹的最简单类型是一种不连续性，即所谓的D^0不连续性。它是指辐射度函数的值的不连续性，这种不连续性通常的起源是由一个点光源和互相接触的物体形成的阴影边界。在前一种情况下，随着我们移过一个表面，光源突然变得可见了，重建和网格化的“扩散”使阴影边界向网格边界扩展。于是，阴影边界就倾向于取网格边界的形状，使其有了一个台阶式的外观。但是，由于我们在辐射度应用程序中倾向于使用面光源，所以所出现的
323 不连续性要高于D^0。不管怎样，这些都将引起可见的人工痕迹。在辐射度函数的导出过程中，不连续性出现在由面光源所照明的场景阴影中的半影和本影的边界处。这时，再一次存在边界和网格之间的互相干扰，并给出阴影边界的特殊的“台阶状”的外观，这些现象比D^0不连续性更难于处理。

当物体之间互相接触时，除非相交边界与网格的边界共线，否则将会出现阴影泄漏或光线泄漏。对于简单的场景，这一思想如图11-11所示。图中房间由一个从地板到天花板的分区进行了划分，划分的界线并不与地板上的曲面片边界共线。房间的一半含有一个光源，另一半完全是黑的。根据曲面片边界的位置不同，重建将产生向黑暗区域的光线泄漏，或者向光照区域的阴影泄漏和光线泄漏。图18-16为一个更复杂场景的阴影泄漏和光线泄漏的效果。尽管对于传统的计算机图形学绘制过程（Gouraud明暗处理），在表示中使用了比所需多得多的曲面片，但其质量仍然低得令人无法接受。

很明显，对场景的进一步细分并不能完全消除阴影泄漏和光线泄漏，细分只能将这种泄

漏减到一种可接受的水平。但是，通过强迫地使网格沿着相互接触的物体之间的相交曲线划分则可以完全消除这种现象，图18-18所示即为在考虑了灯和墙壁相交的情况下，在壁灯区域范围的网格化结果。这时，墙壁的曲面片边界与灯的曲面片边界共线，从而消除了之前进行网格化时出现的泄漏。



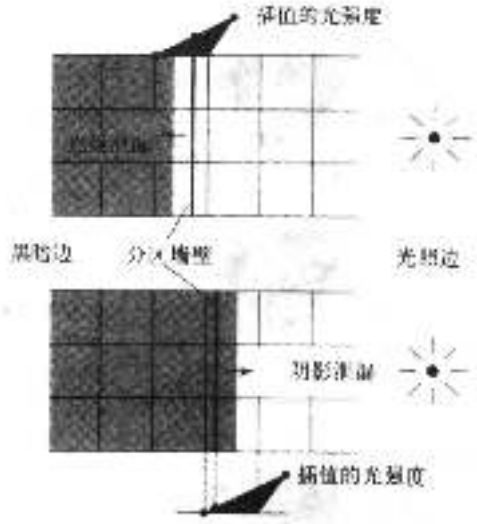


图11-11 阴影泄漏和光线泄漏

11.7 网格化策略

试图要消除这些缺陷的网格化策略，可以按很多方法进行分类。按照细分发生的时间来分类是一种重要的分类方法。

1) 预（priori）细分。在引出辐射度方法的解之前完成划分，也就是说，我们要预测什么地方会出现不连续，并相应地进行网格化。这种划分也称为不连续网格化。

2) 后（posteriori）细分。用一个“启动”网格先开始解过程，随着过程的继续这个网格被逐渐细化。这种划分也称为自适应网格化。

正如我们已经看到的那样，当两个物体相互接触时，我们可以通过保证使得每一个物体

的网格元素的边界互相共线来消除光线泄漏或阴影泄漏。因此，这是一种预网格化。

还可以依网格化的几何性质进行另一种分类。例如，我们可以简单地划分成正方形曲面片（非均匀地划分），使误差减到一个可接受的水平。至今为止，最通常的方法是Cohen and Wallace（1993）所称的h细化方法。我们还可以采用一种方法，它在表面上跟踪辐射度函数的不连续性，并把网格的边界沿着不连续边界放置。这种方法的一种形式被Cohen and Wallace（1993）称为r细化。这种方法的初始网格结点以某种方式移动，即在共享结点的元素之间均分误差。这些方法的概念性示意如图11-12所示。

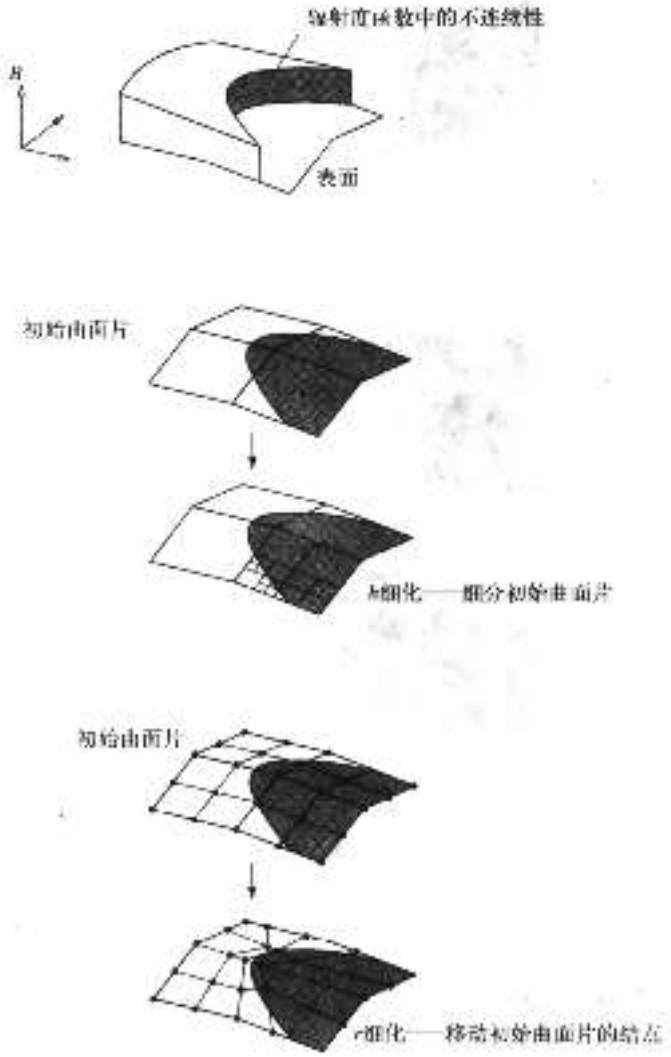


图11-12 细化策略的例子（后细分）

11.7.1 自适应或后网格化

Cohen等（1986）描述了称为子结构的经典的自适应算法。它是在渐进式细化算法的发展

过程中报告的，这一方法开始是与一个全矩阵的解相结合的。通过考虑结点处或元素的顶点处的辐射度变量来使细分过程自适应地进行，如果误差超过某一阈值则进行细分。

其思想是由通过对初始的“粗糙”曲面片进行计算而得到“全局”的辐射度对一个点的辐射度值产生一个精确的解。曲面片被细分成元素。计算元素到曲面片的形状因子，元素到曲面片和曲面片到曲面片的形状因子之间的关系由下式给出:

$$F_{ij} = \frac{1}{A_i}\sum_{q=1}^{R} F_{(iq)j}A_{(iq)}$$

324
~
326

其中:

F_{ij}为从曲面片i到曲面片j的形状因子;

$F_{(iq)j}$为从曲面片i的第q个元素到曲面片j的形状因子;

$A_{(iq)}$为曲面片i的第q个元素的细分区域;

R为曲面片i中的元素个数。

以这种方式得到的曲面片形状因子被用在标准的辐射度解中。

这就使曲面片形状因子的个数由$N \times N$增加到$M \times N$，其中M为所建立的元素的总数，这自然也就增加了形状因子计算所占用的时间。需要被细分成元素的曲面片是通过检查粗糙曲面片解的分度等级而显露出来的。保留前次计算的（粗糙）曲面片解，然后再由这个解用下式获得细分元素的辐射度:

$$B_{iq} = E_q + R_q\sum_{j=1}^{N} B_j F_{(iq)j} \tag{11-4}$$

其中:

B_{iq}为元素q的辐射度;

B_j为曲面片j的辐射度;

$F_{(iq)j}$为元素q到曲面片j的形状因子。

换句话说，就我们所考虑的辐射度解而言，被细分的曲面片的元素的累积效应与未被细分的曲面片是相等的。或者说将一个曲面片细分成元素并没有影响被该曲面片所反射出的光线的量。所以，在确定了曲面片的解之后，曲面片之内的辐射度就可以按曲面片之间各自独立的方式来解。在进行这一工作的过程中，方程（11-4）假设，只对有问题的那个曲面片细分成元素，而其他的曲面片保持不变。这一过程迭代地进行，直到获得所需的精确度。在这一迭代过程中任何一步都可以有三个阶段:

1) 将所选择的曲面片细分成元素，并计算元素到曲面片的形状因子。

2) 用曲面片到曲面片形状因子计算辐射度解。

3) 由曲面片的辐射度确定元素的辐射度。

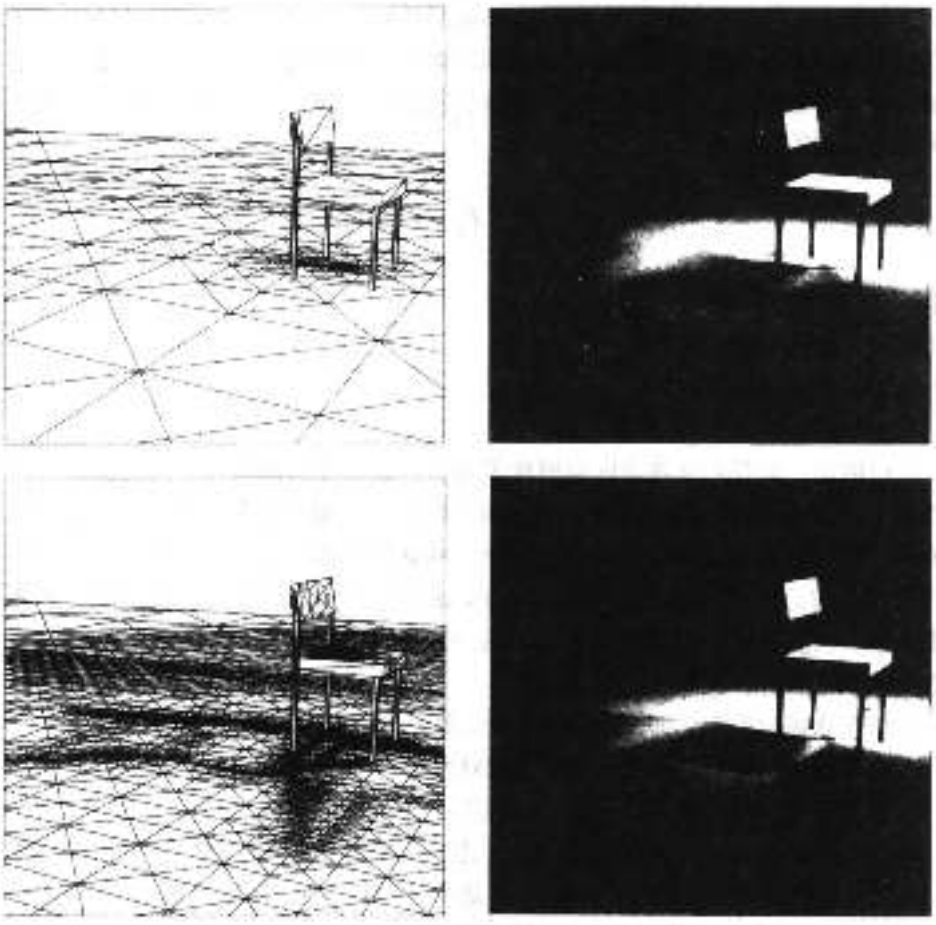
其中的第二步只有在第一次迭代时才进行，粗糙曲面片的辐射度只计算一次。这一方法与只是简单地把环境细分成较小的曲面片是有区别的。这一策略可能导致$M \times M$个新的形状因子（而不是$M \times N$个）以及一个$M \times M$的方程组。

327

对将曲面片细分成元素的细分过程自适应地进行。需要细分的区域在得到解之前是未知的。这些区域从一个初始解获得，然后被用于形状因子的细分。之前所得到的形状因子矩阵

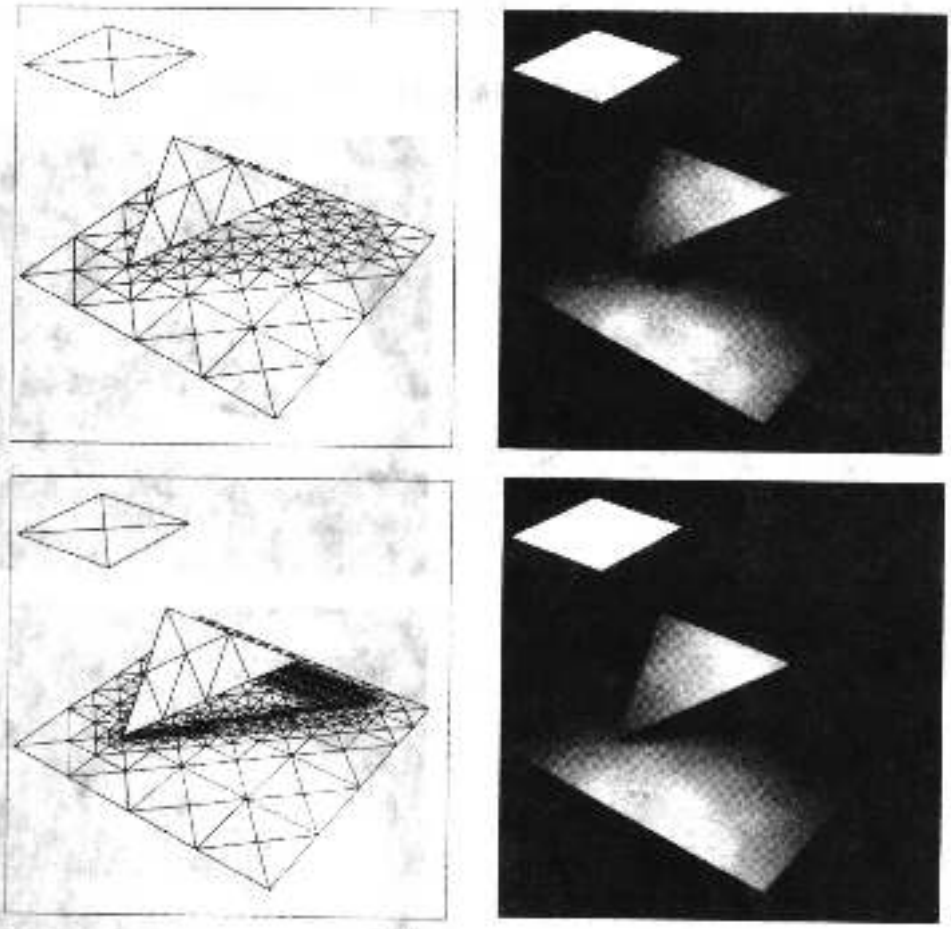
仍然有效，并不需要重新计算辐射度的解。

只有形状因子确定中的部分内容需要进一步离散化，然后将其用于第三个阶段（由粗糙曲面片的解确定元素的辐射度）。这个过程重复进行，直到其收敛于所希望的精确度。于是，在那些需要精确处理的区域上，图像质量得到了改善。这个过程的一个例子如图11-13所示。请注意阴影边界质量的效果。图11-14是相同的设置，但这时光源被细分成了一个较低分辨率和一个较高分辨率的情况。尽管在这种情况下对于发光的曲面片和不发光的曲面片的不充分细分从视觉上看是相似的，但是产生这些差异的原因是不同的。对于不发光的曲面片，我们已经改变了那些与曲面片边界不共线的反射光的光强度，增加了曲面片的个数以获取不连续性。而对于发光的曲面片，问题出在对每个光源所分配的半立方体的个数上。在这里，我们增加了代表发射器的曲面片的个数，因为每一个半立方体的位置都降为一个点光源，而我们希望足够密的一组这种光源来代表发射器的空间宽度。在这种情况下，我们把曲面片（发射器）在整个表面上进行细分，这时的光强度将被认为是均匀的。



a）形状和阴影面积与遮挡物的形状不相对应

图11-13 自适应细分和阴影



b) 阴影的边界正逐渐变得尖锐

图11-13（续）

可以将自适应细分结合到渐进式细化方法中。一种很自然的方法就是计算辐射度的梯度，基于当前发射曲面片的贡献进行细分。但是，这种方法可能导致不必要的细分。图11-15表示了随着每一次迭代之后在壁灯周围细分过程的进行所遇到的困难。一开始，有两个大的曲面片放置于离墙一定距离的位置上，提供对物体的照明。因为程序检测到了属于相同曲面片的顶点之间的很大差别，所以立即引起在灯与墙之间边界线周围的细分。这些曲面片所具有的顶点既有处于光线中心的也有处于墙上的。但是，这个细分还不够好。当我们从光源本身发射能量时，出现了光线泄漏。光源曲面片按照数据库中所存储的模型中的顺序继续发射能量，我们就上升到球上，并向其内部发射能量。这就引起越来越多的光线泄漏。最终，光线被合并到了墙上，照亮了相应的曲面片。随着光的上方光线扇区的旋转，出现越来越多的不适当的细分。这是因为细分是基于当前的光强度梯度的，而随着更多的曲面片被照到，这一值向上增。请注意最后一帧图片，这种细分导致了高光饱和区域中的大量的细分。这些冗余的曲面片越来越降低求解的速度，而从计算时间来考虑，我们正在使情况变得更糟。

可能的替代策略是：

1) 通过仅在每n个曲面片之后才启动细分过程，而不是在每一个曲面片被射到之后就细分来限制细分过程。

2) 通过等待，直到照明代表了所希望的最终分布时再细分来限制对细分的启动。

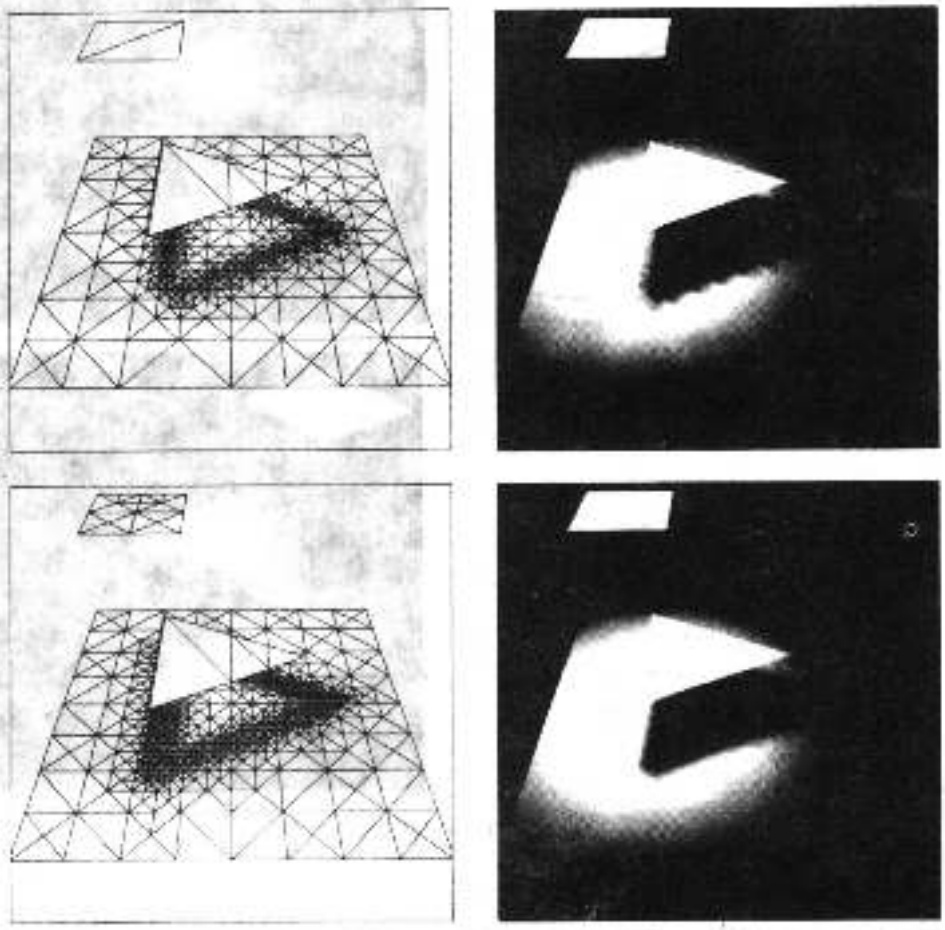


图11-14 发射器不充分细分的效果

11.7.2 预网格化

现在，我们来考虑两个预网格化的策略，即在场景被用于辐射度的解之前对其进行处理。

1. 层次化的辐射度

自适应细分是当前称为层次化的辐射度方法的一种特殊情况。层次化的辐射度是对自适应细分的一种泛化，它是一种两层的层次结构，应用层次的闭联集进行细分。表面之间的相互作用用一个与两个表面之间的几何关系相适应的形状因子进行计算。换句话说，层次化的方法试图通过将计算的精确度限制到由曲面片间的距离确定的量的方法来限制形状因子的计算。

层次化的辐射度方法可以与渐进式细化过程相结合，形成一种后处理的算法；另一种途径是，也可以按预处理的形式来进行层次化的细分，然后再解方程组。下面我们将讨论预处理形式的结构。

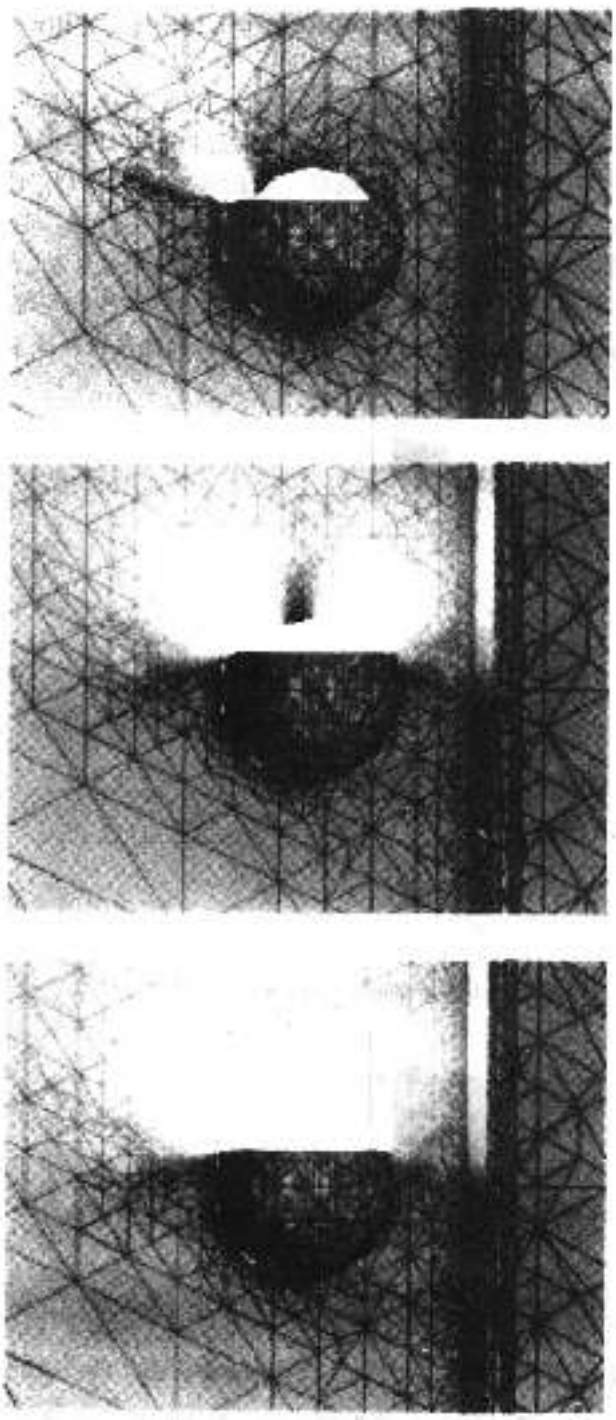
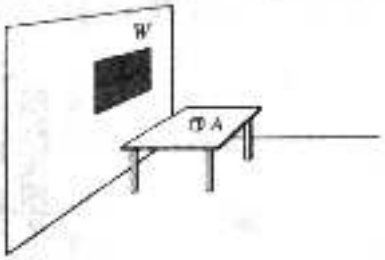
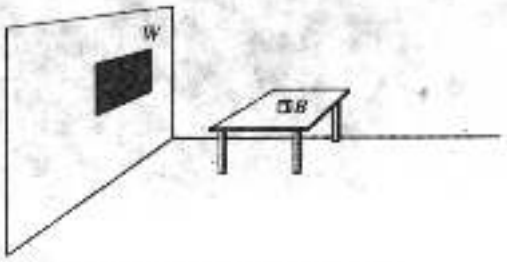


图11-15 这个图像序列表现了随着细分过程的进行所出现的困难，细分在每一次迭代后在聚灯的周围进行。随着光线扇区在光的上方旋转，出现越来越多的不适当的细分，这是因为细分是基于当前光强度梯度进行的，而随着更多的曲面片被照到，这一值向上增。注意最后的一帧图片，这种细分导致了高光饱和区域中的大量的细分。这些冗余的曲面片越来越降低求解的速度，而从计算所需时间来考虑，我们正在使情况变得更糟

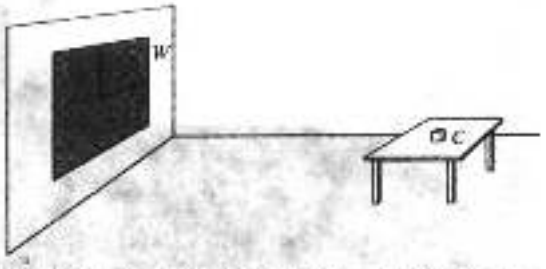
其原理可以很方便地表示出来。图11-16中是一个墙壁曲面片W和三个小的物体A、B、C，三个物体分别放在距墙不同的距离处。从W到A的距离可以与A的大小相比较，我们假设必须对W进行细分，以便计算A的附近由于墙壁所发射或反射的光线而引起的照明的变化。而对于物体B，我们假设可以使用整个的曲面片W。在B的附近由于曲面片W而产生的辐射度详细变化不会受到对B的细分的过分影响。对C的距离，我们假定它足够大，使W和C之间的形状因子相对来说较小。在这种情况下，我们考虑了一个墙上的较大区域，它使W合并成一个是其面积四倍的区域。于是，对于这三种相互作用，当考虑墙壁和物体A、B、C之间的相互作用时，我们选用一个经细分的W、整个W或者将W作为一个大的实体的一部分。请注意，这意味着不仅要有曲面片的细分，而且还要有相反的过程，即将曲面片合并成组。



a) 对于WA的相互作用，将曲面片W细分成元素



b) 初始的曲面片W对于WB的相互作用



c) 对于WC的相互作用，将曲面片W合并到了一个较大的曲面片中

图11-16 层次化的辐射度：场景的细分由能量的互换来决定

这一思想是由Hanrahan等（1991）首先提出的，其中假设，如果当前处理的两个曲面片之间的形状因子超过了阈值，则以当前的尺寸来使用这些曲面片会向解中引入一个不可接受的误差，因此必须对曲面片进行细分。将这一方法与上一节中所介绍的策略相比较，我们现在正在将微分阈值在整个处理过程中倒退一个阶段来使用。不是对相邻曲面片所计算的辐射度之间的差别进行比较，而是在必要时进行细分和重新计算形状因子。这时我们直接地观察

形状因子本身，并进行细分，直到形状因子降到阈值以下。对于两个分享公共边界的曲面片的简单情况很容易展示这种思想。图11-17表示的是基于一个形状因子阈值的细分的几何效果。初始的形状因子的估计值是大的，将曲面片细分成四个元素。在下一个细分水平上，16个形状因子估计值中只有两个超出了阈值。将这两个值进行细分。很容易从示意图中看出，在这个例子中细分的模式被归并到了公共边界。

328 ~ 332

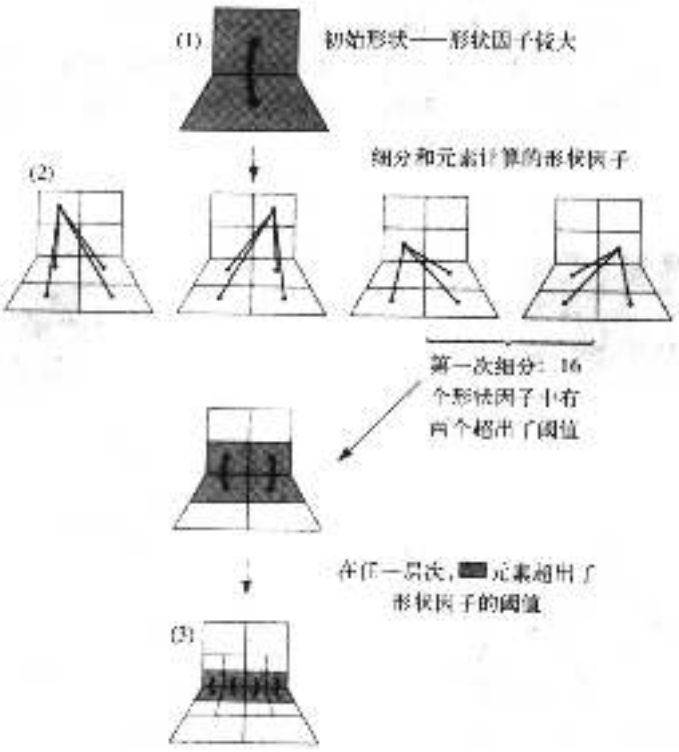


图11-17 层次化的辐射度：两个垂直的曲面片细分的几何效果（Hanrahan等（1991））

层次化的细分策略从将一个大的（初始）的曲面片细分为n个曲面片开始。这导致了对$n(n-1)/2$个形状因子的计算。在这个初始阶段不能用的曲面片对再被按照前一个图中的建议进行细分。这一过程递归地进行。这样一来，每一个初始的曲面片都用一个层次结构和连接来表示。这个结构包含几何细分和一个元素与场景中的其他元素的连接。在这个层次结构中，一个结点代表一组元素，而一个叶结点代表一个元素。为了使这个过程尽可能地快，可以利用对形状因子的一个粗略估计。例如，在形状因子积分定义中的表示式：

$$\frac{\cos\phi_i \cos\phi_j}{\pi r^2}$$

可以利用这一表示式。但是，请注意这个式子并没有对任何遮挡的曲面片进行考虑。

因此，在建立层次结构的过程中所经历的各个阶段是：

1）从初始曲面片的细分开始。这在正常情况下可能会采用比传统的解中所需的曲面片要多的曲面片。

2) 递归地应用下列的过程：

a) 对相连表面对之间的形状因子采用一种快速的估计。

b) 如果这个估计值降到阈值之下或者达到了细分的限制，则记录下在该层次上它们之间的相互作用。

c) 细分表面。

重要的是，要意识到在任一各自所在的层次中的任何结点对之间，两个曲面片都表现出相互作用。图11-18所示的是一个连接，即曲面片A中的一个叶结点和曲面片C中的一个中间结点之间的连接。在图中A的树代表A与曲面片B的相互作用所必需的细分，而曲面片C的树代表它与其他某些曲面片X的相互作用。这意味着从曲面片A到曲面片C的中间结点传输的能量被C中目标结点之下的所有子结点继承了下来。

333
↓
334

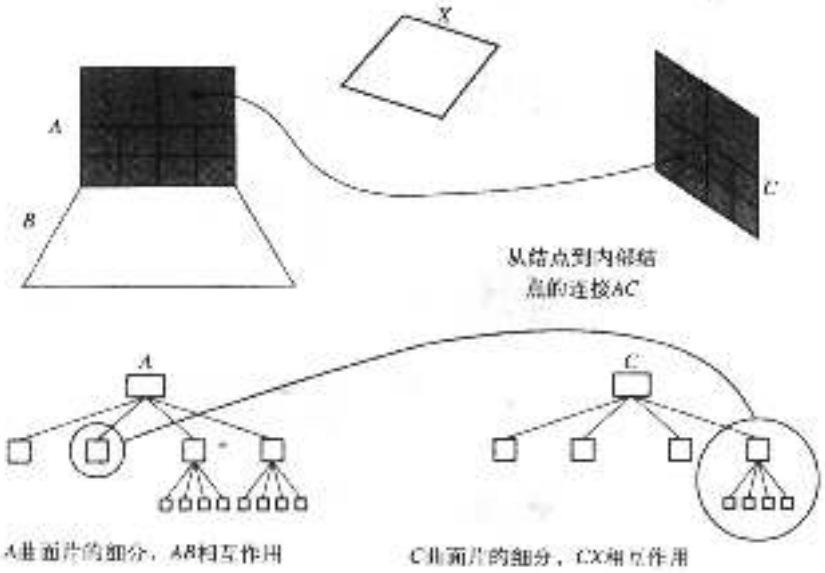


图11-18 两个曲面片之间的相互作用可以由曲面片所处层次上的任何结点对之间的能量互换构成

将这个公式与经典的全矩阵解相比较，我们现在用一种层次式的表示来代替形状因子矩阵。这就暗示着从每一个结点“收集”解，方法是通过沿着与该结点有连接的所有连接线前进，并将连接线端点处所求得的辐射度与相关联的形状因子相乘。因为一般来讲，已经在各个层次上建立起了连接，所以需要从结点向两个方向跟踪层次连线。

通过在每一个根结点上执行两步直到解收敛时为止来进行迭代式求解过程。第一步是在每一个进入的连接线上收集能量。第二步称为“推拉”过程，将一个结点反射的辐射度沿着树向下推，并再将其拉回来。推的过程只是简单地把每一个子结点上的辐射度加起来（注意，由于辐射度的单位是能量/单位面积，所以随着细分过程的进行这个值仍不会消除）。一个元素所接收到的总能量是其直接接收到的能量加上其父辈接收能量的总和。当到达叶结点时，过程反方向进行，能量被沿着树向上拉，在每一个结点上当前存储的能量通过对子结点贡献求平均值来计算。

事实上，这正是11.3节中所介绍的Gauss-Seidel松弛算法的一种精细的过程。对于一个特定的曲面片，我们从场景中所有其他曲面片收集其贡献，以得到当前曲面片的一个新的估计

值。现在的差别是，收集过程涉及到跟踪当前层次结构上的所有连接线，必须以双向遍历或推拉过程对层次结构进行正确的更新。

上面对算法的描述暗示着四叉树数据结构中的每一个结点可以获得下列的信息: 335

- 收集辐射度和发射辐射度（B_g和B_s）。
- 发射值（E）。
- 面积（A）。
- 反射性（ρ）。
- 对四个孩子结点的指针。
- 对（收集）连接线（L）列表的指针。

算法本身有一个简单优雅的结构，由Cohen and Wallace（1993）给出的非常优秀的处理方法可以用下列伪码来表示:

```
while not收敛
 for所有的表面（每一个根结点p）
     GatherRad(p)
 for所有的表面
     PushPull(p,0.0)
```

顶层的过程是一个直接的迭代过程，这一过程收集来自进来的连接线的所有能量。并将其沿着结构向下推，然后再把辐射度值沿层次结构向上拉。

GatherRad(p) 计算结点p处所吸收到的并被反射的辐射度，过程如下:

```
GatherRad(p)
  p.B_g:=0
  for每一个从L到p的连接
  p.B_g:=p.B_g + p.ρ(L.F_pq*L.q.B_s)
  for p的每一个子结点r
     GatherRad(γ)
```

这里，q是由L连接到p的元素，F_{pq}是这一连接的形状因子，ρ是元素p的反射性。连接线的目的/反射器端的辐射度是B_s。通过将它与形状因子F_{pq}和反射性ρ相乘而将B_s转换成在连接线的源/收集器端的反射辐射度B_g。

PushPullRad(p, B)可以看成是一个过程，它在整个层次性的平衡树上正确地分布能量。其过程如下:

```
PushPullRad(p, B_down)
  if p是一个叶结点 then B_up:= p.E + p.B_g + B_down
  else B_up:= 0; for p的每一个子结点r
              B_up := B_up + (r.A/p.A)*PushPullRad(r, p.B_g + B_down)
  p.B_s := B_up
  return B_up
```

这个过程在层次的顶部被首先调用，在该层有收集到的辐射度。递归过程的效果是将这个辐射度传递或推到子结点上。在每一个中间结点处所收集到的能量与沿着向下的路径累积 336
起来并继承的能量相加。当达到一个叶结点时，任何发射都被加入到为该点所收集的辐射度中，其结果被分配给该结点的发射辐射度。然后，递归被展开，把叶结点的辐射度沿着树向

上拉，并在每一个结点处对区域进行加权。

尽管层次化的辐射度方法是一种有效的方法，而且可能是一种很好控制的方法（解的精确度依赖于形状因子的容差以及最小的细分面积），但是它也还是存在阴影泄漏和锯齿状阴影边界问题，因为它很规则地对环境进行划分（虽然是非均匀的划分）而不考虑阴影边界的位置。降低控制参数的值来获得一个更精确的解也仍然需要非常高的代价。这也正是我们将在下一小节介绍的方法的产生动机。

最后，我们摘录一段原文，在这段原文中作者给出了其对这种方法的看法：

> 在本文中所提出的层次化的细分算法受到了最近发展起来的解决N主体问题的方法的启发。在N主体问题中，n个粒子中的每一个粒子都会向所有其他n-1个粒子施加一个力，也就是暗示会有$n(n-1)/2$对相互作用。此快速算法计算出一个粒子上的所有力所用的时间小于时间的平方，这一方法的两个关键思想是：
>
> 1) 数值计算可能会出现错误。因此，一个粒子上的作用力只需要计算到一定的精确度。
>
> 2) 一簇粒子在某个距离上产生的力可以在已知的精确度范围之内进行近似，只要用一个项，而缩减了相互作用计算的总数。

2. 不连续性网格化

预网格化的最普通、最简单的形式就是像我们在这一节的开头所提议的那样处理插值几何（D^0）的特殊情况。当场景被构建并出现阴影泄漏和光线泄漏时，即出现最易看到的人工痕迹时，这一处理大多数情况下是半自动进行的。更通用的方法试图处理较高阶的不连续性问题。当一个物体与一个区域光源相互作用时出现D^1和D^2不连续性，如第9章中所述的阴影区域中的特征性半影-本影的转变。

正如我们已经看到的，常见的后处理方法一般通过对辐射度函数不连续的区域进行细分来解决问题，这种处理只能通过采用越来越高的网格密度来消除误差。而在不连续性网格化背后所隐含的思想是预测在什么位置会出现不连续，并且调整网格边界的方向，使其正好处于不连续的路径上。从定义上来讲，这种方法是一种预处理方法。我们预测到什么地方会出现不连续性，并在启动解过程之前进行网格化，这就使得在解过程进行时不会由于没有对不
337 连续性和网格边界进行校正而出现人工痕迹。

为了预测不连续性的位置，采用了阴影检测算法，而从可视化情况和临界表面两方面来说通常都会出现问题。可以考虑两类可视化事件，即VE和EEE。当一个光源的顶点“穿过”一个在这种情况下被称为接收器的遮挡多边形的边时，顶点-边或VE事件发生。图11-19所示的是三角形光源的一个顶点与长方形遮挡物的一条边之间的相互作用。边和顶点一起构成了一个临界表面，这个表面与一个接收表面相交形成了外部半影边界的组成部分。对于遮挡物的每一条边，都可以相应于光源中的每一个顶点定义一个临界表面。我们还可以定义EV事件，这种事件在一个光源的边界与一个接收多边形相互作用时发生。

如图11-20和图11-21所示，VE事件可能引起D^1和D^2不连续性。图11-20表现的是D^1不连续性的情况。这时，有一个巧合事件发生，即遮挡物和光源的边界是平行的。两个顶点V_1和V_2对半影有贡献。当我们从本影沿着路径xy方向外移动时，光源的可见区域线性地增加，辐射光亮度表现出分段线性或D^1不连续性。图11-21所示的是由VE事件引起的D^2不连续性。在这种情况下，光源的一个顶点被包含到路径xy中。当我们从本影向外移动时，光源的可见性按平方增长，而辐射度呈现D^2不连续性。

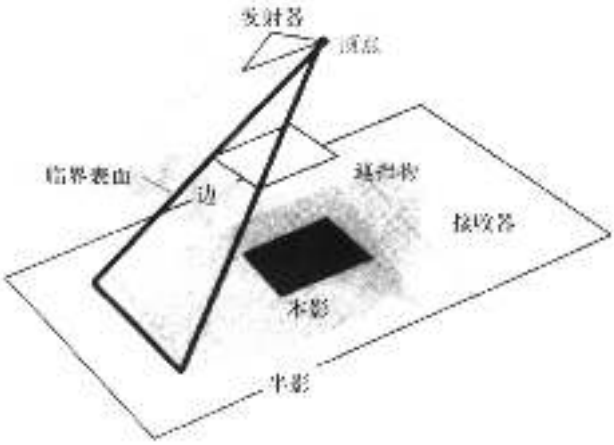


图11-19 VE事件：遮挡物的边和发射器的一个顶点形成了一个临界表面，该表面与接收器的相交线形成了半影的外部边界（Nishita and Nakamae（1985））

当有多个遮挡物时就出现EEE或边-边-边事件，其重要的区别就在于半影的边界（临界曲线）不再像前面的VE例子中那样是一条直线了，而是一条二次曲线，其在曲线上的辐射光亮度函数的相应不连续性为D^2。尽管这时的临界表面不再是一块平面，但是，却是一个（古纹的）二次曲面。

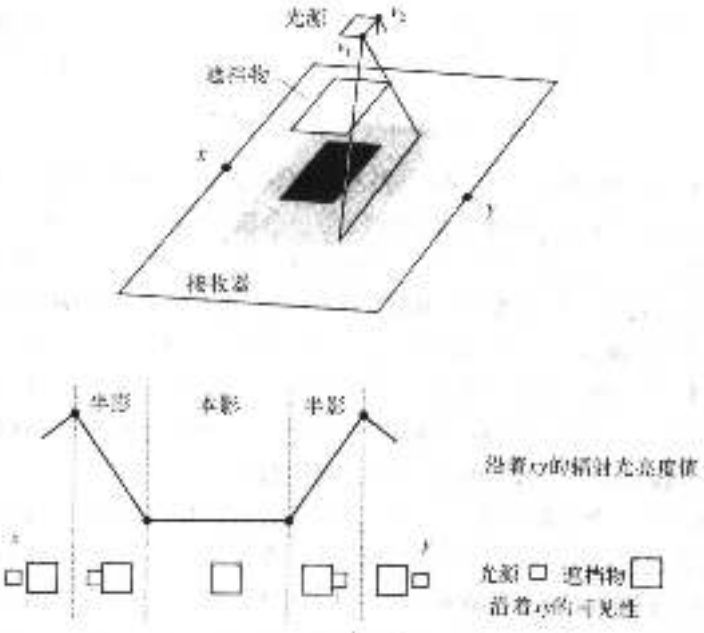
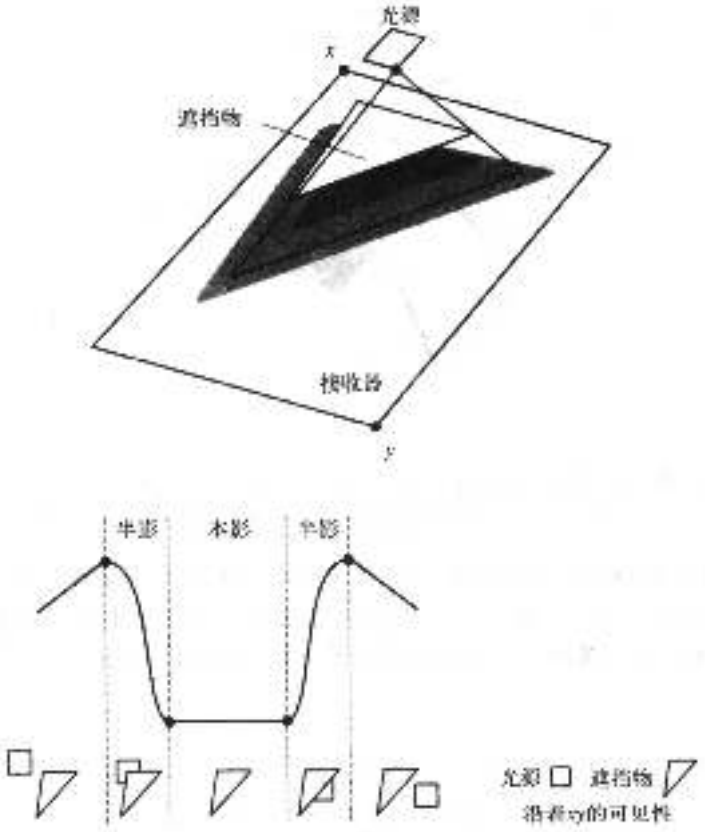


图11-20 引起D^2不连续性的VE事件

对于场景中的任一物体的顶点和任一边都可能出现可视化事件。对于一个具有n个物体的场景，就可能有$O(n^2)$个VE临界表面和$O(n^3)$个EEE临界表面。由于EEE事件具有较高的复杂性和运算成本，所以许多方法都将注意力集中到检测VE事件上了。



338 / 339

图11-21 引起D^2不连续性的VE事件

一种由Nishita and Nakamae（1985）提出的直接方法通过利用阴影体而清晰地确定了半影和本影的边界。对于每一个物体由光源中的每一个顶点构建了一个阴影体。于是，对于每一个光源顶点就有一个与之相关的体积，就好像有一个点光源。所有接收表面上的体积相交形成本影，而半影的边界由包含着阴影体的凸包给出。其做法的一个例子如图11-22所示。

现在，我们将详细地讨论一个由Lischinski等（1992）提出的较新的和更精细的不连续网格化的方法。这种方法把不连续性网格化集成到了一个经修正的渐进式细化结构中，它只处理VE（和EV）事件。这种特定的算法是有代表性的，它处理在实际的不连续性网格化方法中所必须处理的大多数情况，包括处理多个光源和重建问题。

Lischinski等为每一个光源建立了一个单独的不连续网格，并把各自的结果累加成一个最终的解。将场景多边形以BSP树的形式存储。这意味着可以从一个光源的顶点处以一种由前向后的顺序对其进行存取。对于光源来说，由于单个VE事件而产生的不连续性位于如下位置。图11-23所示的是产生一个由顶点确定的楔形的单个VE事件，以及通过边E的端点的投影。该事件是按照A、B和C的顺序获取多边形来处理的。A比E更靠近光源，因此没有受到事件影响。如果表面（B和C）面向光源，则该楔形与表面的交就对该表面增加了一个不连续性。将这个不连续性“插入”到表面的网格中。随着对每一个表面的处理过程的继续，它将把楔形的某部分裁剪下来，算法向下进行，只对未裁剪过的楔形部分进行处理。当楔形被完全裁剪好之

后，对于该VE事件的处理工作也就完成了。

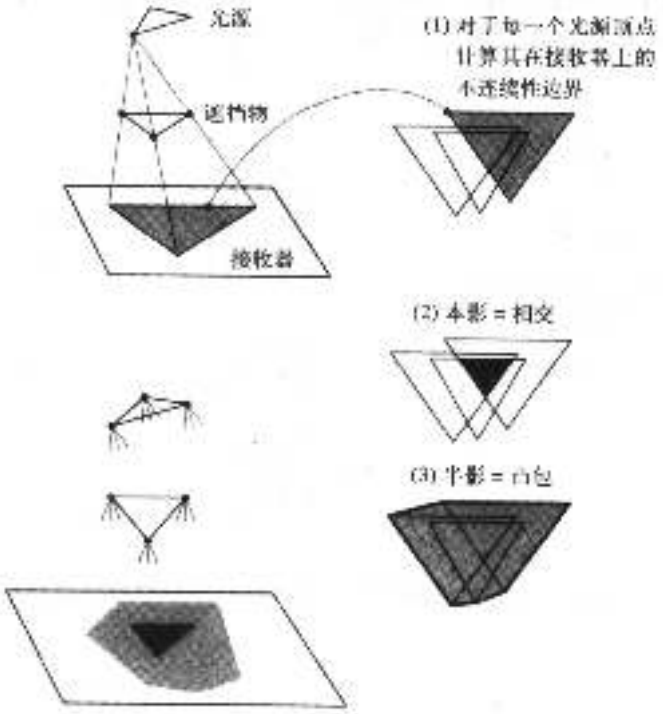


图11-22 来自由VE事件形成的阴影体的本影和半影

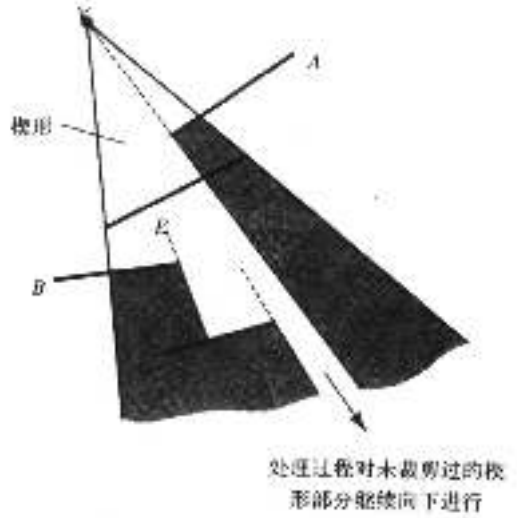
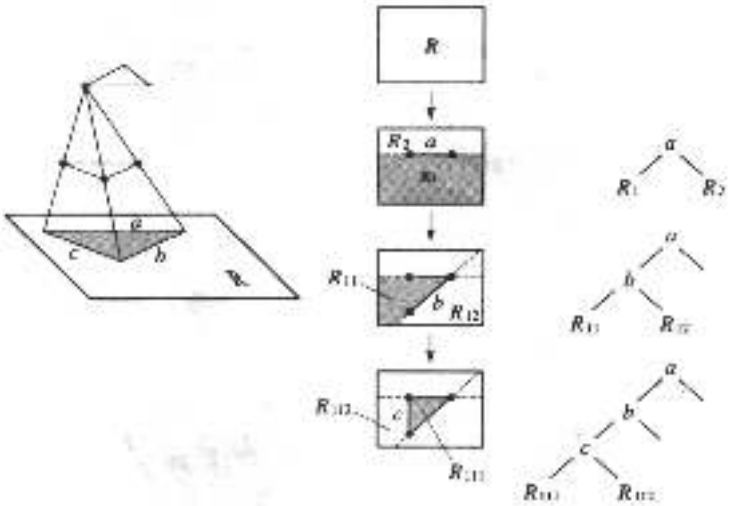


图11-23 处理一个VE楔形（Lischinski等（1992））

将不连续性插入到代表该表面的网格中是通过使用一种DM树来实现的，该树本身由两部分组成，即一个二维的BSP树并将其与一个翼边数据结构相连。翼边结构（Mantyla 1988）代表的是表面结点之间的相互连接关系。这种表示形式的工作方法如图11-24所示，这是一个产

生二个VE事件的顶点的例子，在接收表面上产生了三条不连续性/临界曲线。如果处理的顺序是a，b，c，则对于a的线性方程，表现为根结点，并将其分成所示的两个区域。下一个要处理的楔形b用区域R_1进行检查，将R_1分裂成R_{11}和R_{12}等等。



340
:
341

图11-24 为一个VE事件建立一棵DM树（Lischinski等（1992））

第12章　光线跟踪策略

12.1　基本算法
12.2　用递归方法实现光线跟踪
12.3　七条光线的旅程——一个光线跟踪研究
12.4　光线跟踪多边形物体——多边形交点处的法向插值
12.5　光线跟踪方法的效率问题
12.6　利用光线的连贯性
12.7　一个历史话题——彩虹的光学问题

引言——Whitted光线跟踪

在第10章中，我们曾给出了对Whitted光线跟踪算法的一个简单综述。现在，我们将详细地讨论这个流行的算法。尽管这个方法是由Appel（1968）首次提出来的，但是，光线跟踪算法一般都与Whitted的经典论文（1980）相关联。我们采用术语“Whitted光线跟踪”是为了避免混淆，而这种混淆由于有像“眼睛”、“前向”、“后向”等越来越多的形容词来描述光线跟踪算法已经出现了。

Whittted光线跟踪是一个优雅的部分全局照明算法，它将下列内容结合到了一个模型中：

- 隐藏面消除。
- 对于直接照明的明暗处理。
- 全局镜面相互作用效果，比如物体之间的相互反射和光线穿过透明物体时的折射。
- 阴影计算（但是，只计算硬边界的阴影的几何形状）。

它通常“包含”一个局部反射模型，比如说Phong反射模型。这时就会提出问题：为什么不将光线跟踪算法作为绘制的标准方法，而是用一个Phong方法再加上其他的隐藏面消除、阴影计算和透明计算？对这个问题的直接回答就是运算成本。光线跟踪是高代价的，尤其是对于多边形物体，因为事实上这时需要将每一个多边形看成是要被光线跟踪的物体。这就是光线跟踪方法的困境。如果场景是由“简便的”物体组成的，该方法只能在合理的时间内起作用。二次物体（如球状物）是简便的物体，如果物体是多边形物体，则其表面数必须少，以便光线跟踪程序能在合理的时间之内完成。如果场景复杂（比如有很多物体，而且每一个物体都有很多多边形），则应对基本算法进行有效的规划，规划本身的成本应是场景复杂度的一个函数。在20世纪80年代，很多对光线跟踪算法的研究都集中在有关效率的问题上。但是，当时正好处于硬件发展时期，这时的光线跟踪算法刚好是一个对实际绘制程序的切实可行的选择方案，算法本身的很明显的优势就开始压过了对于成本的考虑。在本章中，我们将建立一段程序来对球进行光线跟踪。然后，再对程序进行扩展，使其能够处理多边形物体。

12.1 基本算法

12.1.1 跟踪光线——初始的考虑

我们已经知道，如何在场景中跟踪无限细的光线，沿着每一条光线去发现完全镜面相互作用。跟踪意味着检测光线与场景中的物体，即相交测试，以求出光线是否碰到其中的任何一个物体。当然，这正是光线跟踪方法高代价的根源，在原始的算法中，我们必须为每一条光线相对于场景中的所有物体（以及每一个物体上的所有多边形）进行测试。在空气和物体之间（或者是在物体和空气之间）的每一条边界处，一条光线“产生”出两条光线。例如，一条光线一开始击到一个部分透明的球，将对物体产生至少四条光线，即两条显现出来的光线和两条内部光线（见图10-5）。我们适当地将传输的光线束缚起来意味着已经考虑了由于折射而产生的几何变形。也就是说，当形成一个投影图像时，透明物体背后的物体被适当地扭曲了。如果球是空心的，则情况会更复杂，这时，一条穿过物体移动的光线会遇到四个交点。

为了进行这种跟踪，我们朝着光线传播的反方向来跟踪光束，即从眼睛开始跟踪光束。进行目光跟踪是因为从光源处开始的光线跟踪毫无疑问是高代价的，我们只是对穿过图像平面窗口光线的那一个小的子集感兴趣。

在每一个碰撞点上，必须进行相同的计算，而这又意味着实现简单的光线跟踪程序的最方便方法是使用递归过程。此递归可以根据一些判据终止，这些判据是：

343

- 总是在一束光线与一个漫反射表面相交时终止。
- 当光线跟踪已经达到了一个预设的深度时终止。
- 当光线的能量低于阈值时终止。

这种计算方法的行为如图18-11（彩色插图）所示。图中，跟踪在递归深度分别为2、3和4时终止，未指派的像素（即那些落在一个纯的镜面表面而没有漫反射贡献的光线对应的像素）以灰色表示。可以看到，灰色的区域作为一个递归深度的函数而“缩减”。

12.1.2 照明模型分量

每一条光线碰撞到物体上的点P，一般情况下会产生一个反射光和一个透射光。而且，通过向光源发射一束光线来计算该点的L，进而求一个局部反射模型，一般我们把光源设为点光源。于是，在每一个点上，光强度由三个分量组成：

- 局部分量。
- 所跟踪的全局反射光线的贡献。
- 所跟踪的全局透射光线的贡献。

我们将这三个分量线性地组合或加到一起，产生点P的光强度。有必要包含一个局部模型，因为在碰撞点上可能会有直接的照明。但是，这也导致了混淆。局部反射模型的采用暗示着经验性的模糊反射（宽的高光），然而，在该点的全局反射光线并不是模糊的，而是沿着一条非常细的路径继续传播直到发现有与任何物体的相交为止。这是因为我们不能提供模糊的全局反射光线，只能跟踪“集中”的光线。这就在光线跟踪的图像中产生一种视觉上的矛盾，即物体中光源的反射（镜面的高光）被模糊了，但是其他物体的图像是完全的。产生这种情况的原因是我们想要通过使物体表现出镜面高光的方法而使物体看起来是闪光的，并且还包含了其他物体的图像。因此，大多数算法都采用了一个局部的镜面分量和一个全局的镜面分量。

对于局部漫反射的考虑也是有必要的，否则的话，我们可能不会有彩色的物体。在光线跟踪方法中，我们不能像辐射度方法中那样处理漫反射相互作用，这可能意味着对于每一个碰撞点都产生一组漫反射光线，这组光线是半球的漫反射光束上的采样，当物体表面的碰撞点上发生漫反射时会出现漫反射半球。这组光线中的每一条光线都必须被跟踪，它可能在一个漫反射表面上终止，有可能会因此而产生组合爆炸。这个问题正是我们在第10章中介绍的诸如路径跟踪方法之类的蒙特卡罗方法发展的动机所在。

如果一条光线碰到一个纯的漫反射表面，则跟踪停止。这样就会出现一种情况，即在每一个碰撞点上局部模型计算的结果与镜面交互作用一起被沿着树向上传递。 344

12.1.3 阴影

可以方便地将阴影加入到基本的光线跟踪算法中。计算光线方向向量L，并将其插入到算法的相交测试部分中去。也就是说，将L看成是与任何其他光线一样的光线。如果L与任何物体相交，则发射出L的那个点就处于阴影中，在该点处直接照明的光强度就减少（见图12-1）。这就产生了硬边界的阴影，其光强度是任意的。这一方法还会导致很高的计算代价。如果有n个光源，则我们必须产生n次相交测试。我们已经在每一个碰撞点上产生了两个光束，再加上一个阴影光束，对于n个光源，就变成了$(n+2)$个光束，可以看出，随着光源数量的增加，阴影计算迅速占据了主导地位，因为在每个碰撞点上的主要计算成本是相交测试的成本。

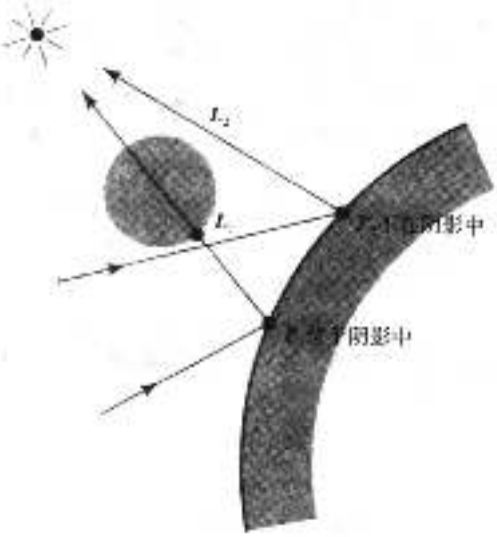


图12-1 通过计算L并将其插入到相交测试器中计算阴影的形状

在Haines and Greenberg（1986）提出的一种计算方法中，采用了一种"光线缓冲器"作为阴影测试的加速器。采用了这样一种过程之后，阴影测试时间减少到1/4 - 1/30。这个方法对于每一个光源预计算一个光线缓冲器，该缓冲器存储的是一组单元或记录，从几何上来看，它是在一个点光源周围形成的立方体的六个表面上放置了一个二维的数组（见图12-2）。为了建立起这个数据结构，所有场景中的多边形都被投影到该立方体的每一个表面上，即将其作为投影中心，也是光源所在位置。因此，在光线缓冲器中，每个单元含有一个多边形的列表，可以从光源看到这些多边形，每一个多边形的深度都是在基于光源的一个局部坐标系中计算

的，记录按照深度的升序进行排序。这意味着对于一束来自眼睛的特定光束，立即就可以有一个可能会遮挡所讨论的相交点的那些物体表面的列表。

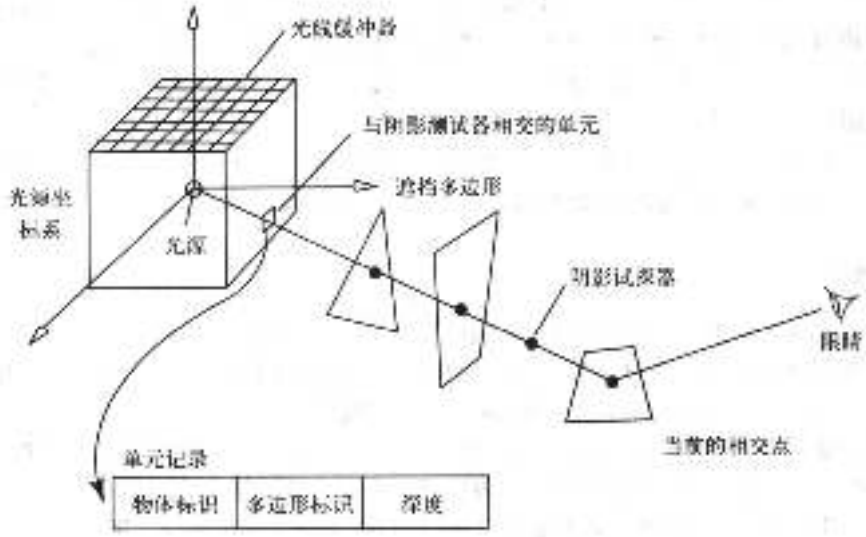


图12-2 Haines and Greenberg（1986）的阴影测试加速器

阴影的测试就降为求那些阴影试探器的光线所通过的单元，即访问经排序的多边形的列表，测试列表中的多边形，直到找到遮挡多边形位置或者可能的遮挡多边形的深度大于相交点的深度为止（这意味着不存在遮挡，因为多边形是按照深度排序的）。存储需求是巨大的，并且依赖于光源的个数以及光线缓冲器的分辨率（请注意光线缓冲器与第11章中所介绍的辐射度半立方体的相似性）。

除了对效率的考虑之外，在Whitted光线跟踪算法中阴影的主要问题是，由于进行了点光源的假设，并且对阴影区域内的光强度必须进行假设，所以产生的阴影是硬边界的。当然，这与完全镜面反射相互作用并非不一致，即与由单个的无限细的光束从每个碰撞点对每一种相互作用进行跟踪而产生的结果不相矛盾。正像分布式光线跟踪（见10.6节）通过对每一个相互作用都考虑多束光线来处理"混淆"的相互作用那样，该方法也通过采用多个光束射向一个（非点的）光源来实现软阴影。

12.1.4 隐藏面消除

隐藏面消除是"自动地"包含进基本的光线跟踪算法中的。我们对场景中的所有物体与每一条光线测试相交。一般来讲，这将给出一个与光线相交的物体的列表。通常情况下，相交测试展现了从碰撞点到相交点之间的距离，而隐藏面消除只简单地从所有的相交点中找到最近的碰撞点，从而求出光线起始处可见的表面。这个模型会出现某些细微的差别，即从标准的绘制或隐藏面消除算法的角度来看，某些隐藏的表面在光线跟踪方法中可能是可见的。这一点如图12-3所示，该图中有这样一个表面，当从目光方向观察时是隐藏表面，而在用入射

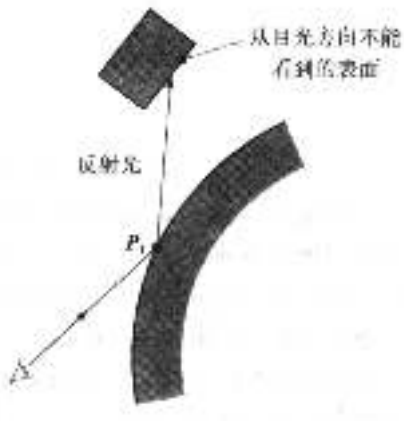


图12-3 一个被反射的"隐藏"面

光线碰撞到的物体中可以被反射到。

345
|
346

12.2 用递归方法实现光线跟踪

现在，让我们用一个特殊的例子考察光线跟踪算法的工作过程。这个例子基于Turner Whitted在1980年产生的一个著名的图像，它通常被认为是计算机图形学中的第一个光线跟踪图像。图12-4是该图像的一个模仿物（这里是一个复制的单色图像，其彩色图像位于彩色插图部分）。

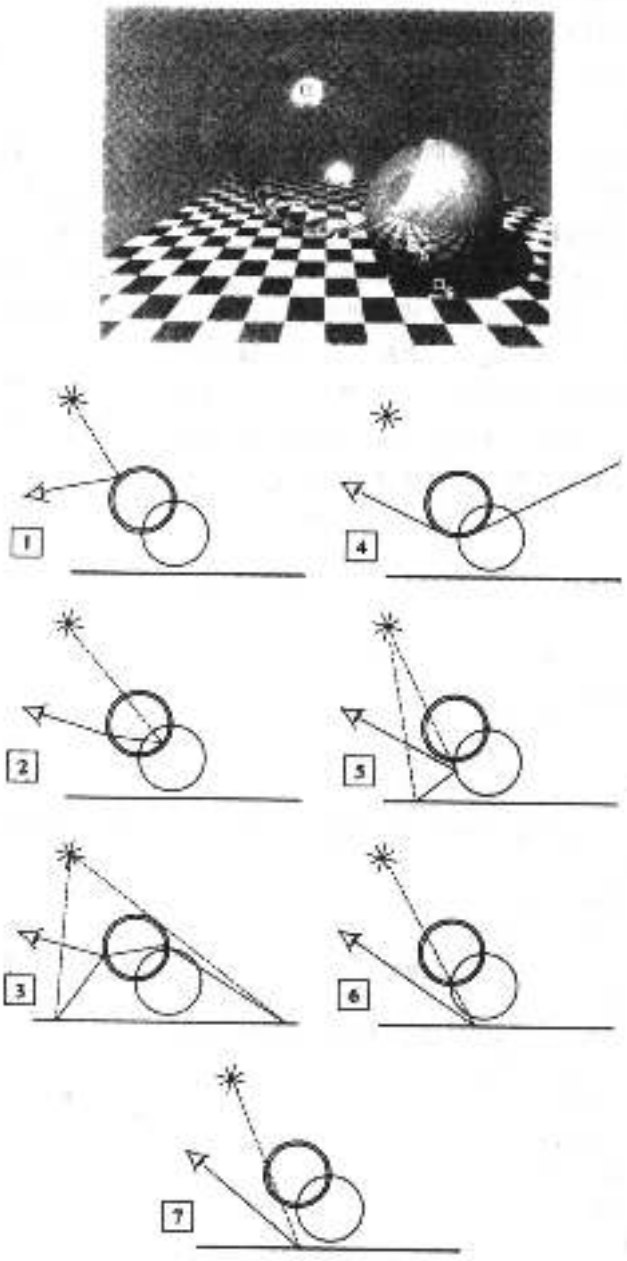


图12-4 Whitted场景（又见彩色插图）

首先，给出符号的定义。在每一个用一条光束碰撞到的点P上，考虑两个主要的分量，即一个局部分量和一个全局分量：

$$I(P) = I_{local}(P) + I_{global}(P)$$
$$= I_{local}(P) + k_{rg}I(P_r) + k_{tg}I(P_t)$$

其中：

P为碰撞点；

P_r为由P点跟踪反射光线求出的碰撞点；

P_t为由P点跟踪折射光线求出的碰撞点；

k_{rg}为全局反射系数；

k_{tg}为全局折射系数。

这个递归方程强调一个点上的照明是由三个分量组成的，即：一个局部分量，该分量通常用Phong局部反射模型计算；一个全局分量，这一分量的计算是通过先求出P_r和P_t，然后再递归地在这些点上应用方程来求出的。整个过程有时用一棵树来表示，如图10-5所示。

执行光线跟踪的过程是很容易写出来的，其代码的复杂性也低。顶层过程调用其自身，以计算反射光线和折射光线。反射光线和折射光线的方向的几何计算在第1章中已经给出，在该章中也可以找到关于一束光线与一个球进行相交测试的细节。

347 ~ 348

光线跟踪程序的基本控制过程由一个简单的递归过程组成，该过程反映了一个结点处的动作，一般来讲，会产生两束光线。因此，过程将包含两个对自身的调用，一个用于折射光线，另一个用于反射光线。我们可以将这一动作总结如下：

```
ShootRay（光线结构）
    相交测试
    if光线与一个物体相交
        求相交点的法向
        计算局部光强度（I_local）
        缩减跟踪的当前深度
        if跟踪的深度大于0
            计算并发射反射光线
            计算并发射折射光线
```

过程中的最后两行意味着对ShootRay()的递归调用。这就是基本的控制过程，在进行递归调用时，还有一些细节，即：

```
计算并发射反射光线的细节为：
if物体是一个反射的物体
    计算反射向量，并将其包含到光线数据结构中
    光线源：＝相交点
    使光线变细（将当前的k_rg与前次调用时使用的值相乘）
    ShootRay（反射的光线数据结构）
    if反射的光线与一个物体相交
        将颜色（k_rgI）与I_local相结合
计算并发射折射光线的细节为：
if物体是一个折射的物体
    if光线进入物体
        累加折射指数
        增加当前光线位于其内的物体的个数
```

```
            计算折射向量，并将其包含在折射光线数据结构中
    else
            减掉累加的折射指数
            降低当前光线位于其内的物体的个数
            计算折射向量，并将其包含在折射光线数据结构中
        光线源：= 相交点
        使光线变细（k_rg）
        if折射光线与一个物体相交
            将颜色（k_rg I）与I_local相结合
```

光线的数据结构中至少要包含下列信息：

- 光线的源。
- 其方向。
- 其相交点。
- 其相交点处的当前颜色。
- 其当前的大小（强度）。
- 相交点与光源之间的距离。
- 光线当前所经的折射指数。
- 当前的跟踪深度。
- 当前包含在内的物体的个数。

349

于是，总的结构是对于反射光线和折射光线调用自身两次的一个过程。过程的第一部分求出距光线起点最近的物体。然后，求其法线并应用局部明暗处理模型，如果在相交点*P*和物体之间有任何物体的话，就减小光强度。然后再分别对反射光和折射光递归地调用过程。

ShootRay() 过程递归调用的次数由跟踪参数的深度来控制。如果这个值为单位值，则场景只能用局部反射模型绘制。为了发现点*P*处另一个物体的任何反射，我们需要深度至少为2。为了处理透明的物体，我们需要深度至少为3（初始的光线、穿过物体的光线以及我们必须跟踪的那束光线。被跟踪的光线从任何它所碰到的物体上返回一个光强度）。

12.3 七条光线的旅程——一个光线跟踪研究

返回到图12-4。我们考虑在7个像素点的情况下表现出高光时光线跟踪方法的工作方式。场景自身由一个薄壁的或中空的球组成。它几乎是完全透明的，再加上一个半透明的白色球组成。两个球都是在一个布满场景的红黄相间棋盘上漂浮着的。除了物体空间之外的其他地方都是蓝色的背景。物体的性质总结于表12-1中。注意，这个模型允许我们将k_s根据k_{rg}设为四种不同的值，k_{rg}是12.1.2节中介绍的矛盾根源；反射光线要根据所考虑的是哪一个分量（局部的还是全局的）而分别进行处理。

表 12-1

非常透明的中空球				
k_d(局部的)	0.1	0.1	0.1	(低)
k_s(局部的)	0.8	0.8	0.8	(高)

（续）

非常透明的中空球						
k_{rg}	0.1	0.1	0.1	（低）		
k_{tg}	0.9	0.9	0.9	（高）		
不透明的（白色）球						
k_d（局部的）	0.2	0.2	0.2	（白色）		
k_s（局部的）	0.8	0.8	0.8	（白色）		
k_{rg}	0.4	0.4	0.4	（白色）		
k_{tg}	0.0	0.0	0.0			
棋盘						
k_d（局部的）	1.0	0.0	0.0/1.0	1.0	0.0	（高亮红或黄）
k_s（局部的）	0.2	0.2	0.2			
k_{rg}	0					
k_{tg}	0					
蓝色背景						
k_d（局部的）	0.1	0.1	1.0	（高亮蓝色）		
环境光	0.3	0.3	0.3			
光源	0.7	0.7	0.7			

考虑如图10-4所示的与像素相关的光线。

1. 光线1

这束光线所朝的方向是在高度透明的球上看到镜面高光的方向。由于该光线靠近$\boldsymbol{L}$的镜面方向，所以在$I_{\text{local}}(\boldsymbol{P})$中镜面成分的贡献就高，而$k_{rg}I(\boldsymbol{P})$的贡献就低。对于这个物体的$k_d$，局部反射系数是低的（它需要乘以透明值1）。而$k_s$相对于$k_{rg}$来讲是高的。但是请注意，内部的贡献只是在物体表面上的一个非常小的区域占主导地位。还需注意到，正如我们已经提到的那样，高光是不能扩散的。但是，如果我们让高光只占据一个像素的位置，那么它也是不可见的。

2. 光线2

几乎与光线1相同，只是镜面高光出现在中空球的内壁上。这束特殊的光线表示了光线跟踪算法中已有的另一种错误。实际上，光线从光源出发穿过球而没有发生折射（也就是说，我们仅仅将$\boldsymbol{L}$值与$\boldsymbol{N}$的局部进行了比较，而忽略了光线是处于球的内部这样的事实）。这意味着镜面高光位于错误的位置。但是，我们还是接受了这个错误，因为除此之外也没有关于正确位置的直观的推测。只是认为它是正确的。

3. 光线3

350
~
351

光线3也碰到了薄壁的球。在所有与中空球碰撞的点上的局部贡献为零，占主导地位的贡献来自棋盘。这会由于球壁的折射作用而变得有一些扭曲。红（或黄）颜色来自$I_{\text{local}}(\boldsymbol{P})$中的高$k_d$部分，$\boldsymbol{P}$是棋盘上的一个点。对于这个表面$k_{rg}$和$k_{tg}$为零。但是需注意我们混合了两个棋盘。其中的一个正如我们已介绍的，另一个是在球的外表面上反射上去的。

4. 光线4

这束光线再次碰到了薄壁球。但是，这一次碰撞的方向是穿过距玻璃有一定距离的地方

（也就是说它只是穿过玻璃，但是却没有进入空气），这引起了一个高的折射效果，使光线在蓝色的背景中终止。

5. 光线5

这束光线碰撞到了不透明的球，由于有一个白色的k_d（局部的）而从局部分量返回一个明显的贡献。在第一次碰撞时，全局反射的光线碰撞到棋盘，因此存在一个混合：

- 白色（来自球面的漫反射分量）。
- 红/黄色（由棋盘反射的）。

6. 光线6

这束光线一开始碰到了棋盘，其颜色完全来自该表面的局部分量。但是，该点是位于阴影中的，是由光线L与不透明的球相交求出的。

7. 光线7

这一束光线的情况与光线6的情况完全相同，只是这一次与L相交的是薄壁球。因此，阴影区域的光强度并没有像前一种情况降低那么多。我们也没有把L可能会经历的那种递归效果考虑进去。所以，事实上阴影的位置是错误的。

12.4 光线跟踪多边形物体——多边形交点处的法向插值

把建模基元限定为一个球，或者最好限定为一个二次实体对于实际应用来说太严格了。在这一节中，我们将讨论光线跟踪多边形物体。把上面给出的程序进行扩展，使其覆盖一般的多边形物体需要对多边形物体进行相交测试（见1.4.3节），以及一个计算或插值碰撞点P处法线的方法。需要引起注意的是，多边形的小平面只是对曲面的某种近似，正像在Phong明暗处理方法中那样，我们需要由顶点的法向进行插值，对于“真实”表面的表面法向的近似即 352
为小平面近似。对于局部照明分量需要有这个实体，对于反射和折射的计算也需要有这个实体。回忆在Phong插值方法中（见6.3.2节），我们采用了屏幕空间的二维分量来逐像素地、逐扫描线地对投影到多边形上的每一个像素进行法向插值。我们将二维屏幕空间作为插值的基础对三个顶点的法向进行插值。在光线跟踪算法中如何由顶点法向进行插值呢？请记住我们是在世界空间中进行运算的。一种简易的方法是存储每一个多边形的多边形法线以及其顶点的法线。我们求出其三个分量x_w、y_w和z_w中的最大值。最大的分量标识此多边形在方向上与这三个世界坐标平面中的哪一个靠得最近，我们可以利用这个平面进行插值，插值所用的方法与在Phong插值方法中所用的相同（见1.5节）。这个平面与在Phong插值方法中所用的屏幕平面是等价的。此方法的思想如图12-5所示。该平面被用作按如下方法插值。我们考虑多边形是在一个坐标系中表示的，在这个坐标系中碰撞点P为原点。然后，我们必须搜索多边形的顶点，以求出与“中间”轴交叉的那些边。这使我们能够对适当的顶点法向进行插值以求出N_a和N_b，再由N_a和N_b求出所需的法线N_p（见图12-6）。求出插值的法线之后，就可以计算局部照明分量以及反射和折射光线了。请注意，因为我们进行的是“随机的”插值，所以也就失去了Phong插值算法的高效性的优点，Phong算法是逐像素、逐扫描线地按增量进行的。

结论是，在光线跟踪算法中，对多边形物体要承担两种显著的成本。首先，占主导地位的成本是在对物体的每一个多边形进行相交测试时的成本。其次，还必须承担计算所需要的求插值法线所花费的成本。

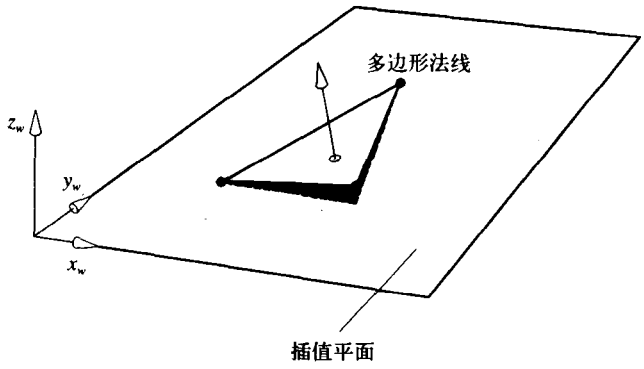


图12-5 位于$x_w y_w$平面中的一个多边形将有一个高分量z_w，选择这个平面进行顶点法向的插值

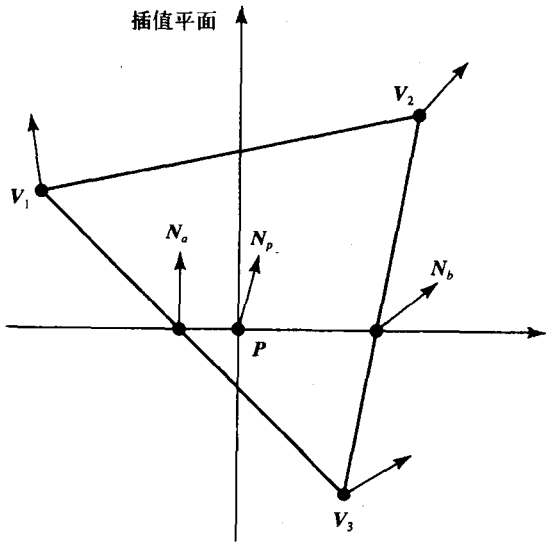


图12-6 在碰撞点P上求一个插值的法线

12.5 光线跟踪方法的效率问题

12.5.1 自适应深度控制

在光线跟踪程序中所需的跟踪深度依赖于场景的性质。一个含有高度反射表面和透明物体的场景比一个由不太具有反射能力的表面以及不透明的物体组成的场景需要较高的最大深度值（注意，如果将深度值设为单位值，则光线跟踪程序的作用就像传统的绘制程序一样了，传统的绘制程序只进行了隐藏面消除并应用一个局部反射模型）。

在Hall and Greenberg（1983）中指出，一般来讲，场景中高度透明和反射的表面所占的百分比是小的。因此，对每一束光线都跟踪到最大深度是不经济的。Hall 和Greenberg提议用一种自适应的深度控制，即依赖于光线与之相互作用的那些材料的性质来进行深度控制。这时被跟踪的光线就决定了终止的深度，这个深度值可以是从1到最大预置深度值之间的任意值。

随着光线穿过一个场景，它的变细方式是多种多样的。当一束光线在一个表面上被反射时，其变细是用该表面的全局镜面反射系数来实现的。当一束光线在一个表面上被折射时，其变细是用该表面的全局折射系数来实现的。在目前这一阶段，我们只考虑表面相互作用处的这种变化。被作为一个穿过几层的相交之后反向跟踪得到的结果而被检查的光线将对由几个这些系数变细的顶层光线做出贡献。从深度为n的那束光线对于顶层的颜色所做出的任何贡献都会被每个结点上所遇到的全局系数的积所削弱:

$$k_1k_2\cdots k_{n-1}$$

如果这个值低于某个阈值，则没有什么理由继续对其进行跟踪。

当然，总的来讲，对于每一束光线都会有三种颜色（RGB）的贡献，并且对于每一次系数的削弱也会有三个分量。于是，当启动递归过程时，程序被赋予一个累积的权重参数，该参数表示最终的权重，它将在顶层赋给由该过程启动所代表的光线所返回的颜色。对于一个新的过程的启动，正确的权重很容易通过对当前被跟踪的光线累积权重，并将该值与表面相交处的反射或折射系数相乘而得到，表面相交处将产生新的光线。 353 ~ 354

另一种可以使光线变弱的方法是在一个不透明的材料中穿行一段距离。可以通过将一个折射系数与构成物体的材料相关联来达到这一目的。这样一来，颜色值就被这个系数以及一束光线在该材料中穿过的距离所确定的量而减小。对于光线跟踪过程中的相交计算的简单加可以允许将这个性质结合进来。

对于自适应深度控制的使用，将防止例如一束光线碰撞到了一个几乎不透明的物体，并产生一束折射光，然后这束光线再被跟踪穿过物体并进入场景的情况发生。从场景中返回的光强度就可以这样被初始物体所削弱，使得这一计算被消除。因此，根据所预设的阈值的不同，在这种情况下光线将在第一次碰撞时被终止。

对于一个具有最大树深15的高反射性的场景，在Hall and Greenberg（1983）中指出，这种方法给出的平均深度为1.71，使得图像的生成时间大大减少。实际所获得的节省将依整个物体的性质及其在场景中的分布而定。

12.5.2 第一次碰撞加速

在前一小节中已经指出，即使对于非常高反射性的场景，光线被跟踪的平均深度也在1～2之间。这个事实使得Weghorst等（1984）提出建议采用一种混合的光线跟踪程序，在这种程序中初始光线的相交测试是由隐藏面消除算法在预处理阶段估算出来的。这里暗示着，隐藏面消除算法比用于第一次碰撞的一般性光线跟踪程序更有效。Weghorst等（1984）建议执行一种经修正的Z缓冲器算法，但使用的是相同的观察参数。对于Z缓冲器进行简单的修改，使它对图像平面中的每一个像素产生一个指向在该像素处可见的物体的指针。再从这一点处继续执行结合了自适应深度控制的光线跟踪。这样，与第一次碰撞相关的代价高昂的相交测试就消除了。

12.5.3 具有简单形状的限定物体

已知光线跟踪方法的高成本是由其中的相交测试造成的，我们可以通过使这一部分的计算尽可能高效而大大地提高递归光线跟踪算法的效率。一种明显的和大量采用的方法是把物

体限定在一个“简单”形状的体积中，该体积称为限定体。一开始，我们测试光线与限定体相交的情况，只有当光线进入了这个体积中时，才测试光线与物体的相交。注意，我们在挑选视见体的运算（见第6章）和碰撞测试（见第17章）中也采用了这种方法。

355

限定体需要有两个性质。首先，它应该有一个简单的相交测试，而球就是一种很明显的候选物体。其次，它必须能有效地包围物体。从这一角度来说，球又是不充分的。如果物体又长又薄，则球中会含有很大的无效体积空间，很多光线都会进行相交测试但却并不与物体相交。相对尺寸可以调节的长方形实体可能是最好的简单限定体（在第1章中给出了球和盒的相交测试的细节）。

限定体的困难是其复杂程度不能太高，因为这违背了其自身的初衷。通常情况下，对任一场景，限定体计算的成本将与其包围物体的效率有关。这一点从概念上很容易了解。图12-7所示为一个二维的场景，它含有两个长条和一个圆代表复杂的多边形物体。图12-7a所示为一个圆（球）作为限定体，其对长条的包含效率低，不仅球的效率是低的，而且它们自身还相互相交，空间被其他物体所占据。用包围盒与场景的轴和标定（使轴与限定盒并线，AABB）是较好的选择（见图12-7b）。但是，这时包围倾斜的长条的包围体是不高效的。对于这个场景，最好的限定体是任意方向性的盒（见图12-7c），测试限定体的成本从球到任意方向的盒是增加的，这些盒称为OBB。



a）圆（球）作为限定体　　b）长方形（盒）作为限定体　　c）任意方向上的长方形（盒）

图12-7　三个不同的限定体，从a到c。限定体的复杂性成本与其封闭的效率一起增加

Weghorst等（1984）定义了一个限定体的“空的区域”，它是物体和限定体在一个垂直于光线并通过光线的原点的平面上正交投影之间的差（见图12-8）。这些表明“空的”区域是物体，限定体和光线方向的一个函数，它定义了相交测试的一个成本函数：

$$T = b*B + i*I$$

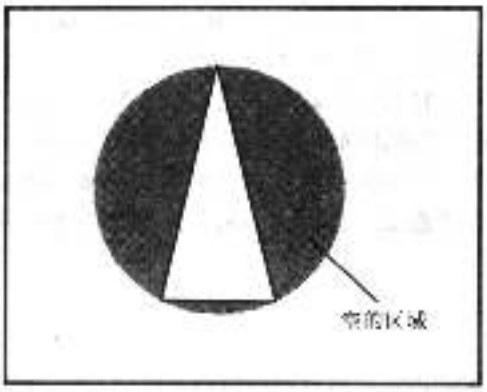


图12-8　一个限定球的空区域

其中:

T为总的成本函数;

b为限定体用于相交测试的次数;

B为对限定体进行相交测试的成本;

i为某项用于相交测试的次数（$i<b$）;

I为测试某项相交的成本。

作者指出，这两个乘积总体来看是相互依赖的。例如，通过降低限定体的复杂性来降低B将几乎可以肯定地增加i。给出了一个定量的方法去选择最优的球、平行四边形以及圆柱体作为限定体。

12.5.4 二次数据结构

另一种使相交测试提高效率的常见方法是建立一个二次数据结构来控制相交测试。二次数据结构用作一种指南，初始数据结构（即物体数据库）在最适当的点上被插入。

1. 限定体的层次结构

首先由Rubin and Whitted（1980）提出并由Weghorst等（1984）进行讨论的对限定体的一种通用的扩展是，试图在场景上加上这种限定体的一种层次化的结构。如果可能的话，在空间上最靠近的物体允许形成簇，而这些簇自身被封闭在限定体中。例如，如图12-9所示是一个容器a和一个大的物体b以及其中的四个小物体（c_1, c_2, c_3, c_4）。树代表了七个边界之间的层次关系:一个圆柱体封闭了所有的物体，一个圆柱体包围了b，一个圆柱体包围（c_1, c_2, c_3, c_4）以及这些物体中的每一个的限定圆柱体。一束光线对于限定体的跟踪意味着从最顶层遍历这样一棵树。在这个例子中，一束光线若与c_1相交就应该与包含c_1, c_2, c_3, c_4的限定体进行测试，这只是因为它与代表该簇的限定体相交。这个例子还表明，根据场景的性质应该能够选择出相邻物体的合理的簇，如果能够比非层次化的限定方法获得更大的节省的话。现在，相交测试是作为一种递归过程执行的，执行过程通过层次结构向下进行，并且只穿过那些有相交出现的结点。于是，场景就被尽可能地组成物体的簇，每一个簇都可能包含其他在空间上组成簇的物体组。在理想情况下，高层的簇被封装在含有低层簇和限定体的限定体内。只有在物体之间靠得足够近时才可以创建簇。若创建稀疏分散的物体的簇将妨碍处理过程。潜在的集聚和层次的深度将取决于场景的性质: 层次越深，潜在的节约就越多。这种方法的缺点是它严重依赖场景的性质。此外，要建立一个合适的层次需要相当多的用户投资。

356 ↓ 357

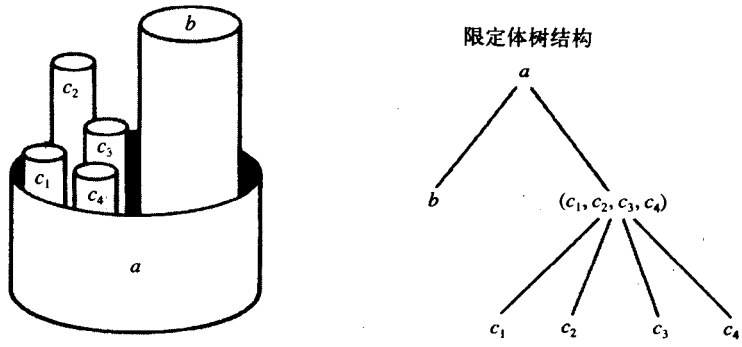


图12-9 一个简单场景和与其相联系的限定圆柱体的树结构

在碰撞检测过程中使用的限定体在第17章讨论。虽然原则上相同，但碰撞检测要求适合在限定体对之间相交的有效率的测试，而不是光线/体积测试。OBB 层次已经被证明在这里以及17.5.2节描述的情形下是有用的。

2. 使用空间连贯性

目前，空间连贯性看来是使光线跟踪成为程序化的图像合成中的一种实际建议的唯一方法。 由于这个原因，它被进行了详细的讨论。在光线跟踪中物体的连贯性一般被忽视了。原因很明显。从本质上来看，光线跟踪算法在场景内的任意位置大量生产任意方向的光线。使用这样“随机的”光线访问物体的数据结构和在这样的光线路径上有效地抽取出物体是很困难的。 与图像空间扫描转换算法不同（在那种方法中，例如，可以列出活跃的多边形），关于将由一束初始光线或者观察光线大量生产的光线序列没有先验的信息。原始的光线跟踪算法在每次碰撞之后对全部对象执行彻底的搜寻，或许可以用一个像限定体那样的修改来限制搜寻。

空间连贯性方法的基本原理是简单的。场景占据的空间被再分成区域。 现在，不是用全
358 部物体或者限定物体的集合来检查一束光线，而是我们试图回答这样的问题：光线目前正在穿行的区域已经被物体占用了吗？在这个区域或者没有什么，或者该区域包含了物体的一个小的子集。然后测试这组物体是否与该光线相交。子集的大小和确定物体的空间占用的准确度取决于细分空间的方法以及物体的数量和性质。

这种方法有时称为空间连贯性，一些工作者独立地建立起了空间细分或空间跟踪的方法，值得提及的是Glassner（1984）、Kaplan（1985）和Fujimoto等（1986）。所有这些方法都涉及到对空间的预处理，以便建立含有物体空间占用信息的辅助数据结构。然后使用这种辅助数据结构跟踪光线进入目标数据结构。注意到这一基本原理（预处理物体的环境以降低计算一个观察所需的计算工作量）首先被Schumaker等（1969）用于为飞行模拟器所建立的隐藏面消除算法中（参阅6.6.10节）。在这个算法中，场景内的物体通过用平面细分空间而集合成组。空间细分由二叉树表示。 任何观察点都位于由树叶所表示的一个区域中。对于一个特定的观察点，在线树遍历会迅速产生组群的深度优先序列。这个算法的要点是空间细分是离线计算的，辅助结构（即用二叉树表示细分）用来确定物体簇的初始优先序列。这一工作的动机是加速在线隐藏面消除处理过程，使图像生成能够以实时的方式进行。

为了降低光线与物体相交的测试次数而对限定体或者区域方法的不满是促进空间连贯性方法（Kaplan 1985）发展的部分原因。已经指出了对限定体的主要反对意见之一。该方法的“效率”依赖于物体与限定体的空间匹配的好坏。一个更基本的异议是这样的方法可能会增加光线与物体相交搜索的效率，但是它对于降低对场景内的物体数量的依赖却没有任何贡献。每一束光线仍然必须依照每个物体的限定范围进行测试，搜索时间成为一个场景复杂性的函数。 此外， 虽然通过使用一个限定体的层次结构可以达到主要的节省，但是，需要有相当多的投资来建立一个合适的层次，并且取决于场景内的物体性质的不同，有些分层的描述可能是困难的或者是不可能的。这一小节里所描述的方法的主要革新是使得绘制时间成为常数（对于一个特定的图像空间分辨率来说），并消除了它对场景复杂性的依赖。

各种利用空间连贯性的方法之间的差别主要在于所采用的辅助数据结构类型不同。Kaplan（1985）列出了六个性质，如果要在程序化的绘制应用中使用这一技术的话，一个实
359 际的光线跟踪算法应该表现出这些性质。Kaplan的需求是：

1) 计算时间应该相对独立于场景的复杂性（环境内物体的数量，或者单个物体的复杂性），以便可以绘制有真实复杂性的场景。

2) 每一次的光线时间应该相对恒定，而不依赖于光线的起源或者方向。这个性质保证了对于明暗处理的图像的总的计算时间将只依赖于总的图像分辨率（第一层被跟踪光线的数量）以及明暗处理的效果（第二层或更高层被跟踪光线的数量）。对于一个已知的图像分辨率以及真实性水平，这就保证了可以预测的性能。

3) 计算时间应该“合理”并且在可承受的处理器系统上“交互式”（在几分钟内）地进行。

4) 算法不应该要求用户提供分层的物体描述或者物体聚簇的信息。 用户应该能够将在不同的时间和以不同的方法产生的数据结合进单个的场景中。

5) 算法应该能够处理各种不同的基元几何类型，并且应该能够方便地扩展到新的类型。

6) 算法对连贯性的使用不应该把其应用性降低到并行处理或者其他高级的架构中。相反，它应该适合于在这样的架构上实施。

Kaplan总结了这些需求：“为了真的可用，应该能够在一个复杂的环境内，以合理的、可预测的时间和合理的费用跟踪大量的光线”。

出现了两种与辅助数据结构相关的方法，包括八叉树表示（Fujimoto等 1986; Glassner 1984）和一种叫BSP（二叉空间分割）的数据结构。BSP 树最初是由Fuchs（1980）提出的并且被Kaplan（1985）使用。

3. 在光线跟踪中使用八叉树

八叉树（参见第2章）是在允许我们利用空间连贯性的场景内的物体的表示——空间中彼此靠近的物体在八叉树中由互相靠近的结点来表示。

当跟踪一束光线时，不是对光线与场景中的每一个物体进行相交计算，而是在所占用的空间的细分中从子区域到子区域地跟踪光线。 对每个光线穿过的子区域来说它可能只与少量物体（通常为一两个）相交。假若可以在八叉树中求出对应于一束光线正在通过的一个子区域的结点，我们就能迅速找到处于或者靠近光线路径的那些物体。只需对这些物体进行相交计算。如果空间已经被细分到了某种水平，使每个子区域只包含一两个物体，则对于一个区域所需的相交测试的次数将是少的，而且将不会随着场景复杂度的增加而增加。 360

用八叉树跟踪一束光线

为了用空间细分方法确定哪个物体接近于一束光线，我们必须确定光线通过了空间的哪个子区域。这包含跟踪光线沿途进出每个子区域。在这个过程期间要求的主要操作是在八叉树中找到相应于一个点 (x, y, z) 的结点，以及因此找到的其在空间内的区域。

总的跟踪过程通过检测相应于光线的起始点的区域开始。光线被与位于这个区域内的任何物体进行相交测试，如果有任何相交，则遇到的第一个是所需要的那束光线。如果在最初区域没有相交， 那么光线必须被跟踪到它通过的下一个区域。方法是通过计算光线与区域的边界之间的相交，因此，也计算光线离开该区域的点。然后用光线在下一区域内一定距离上的一个点来求出相应于下一个区域的八叉树上的结点。然后，对这个区域里的任一物体进行与光线的相交测试。这一过程随着光线的踪迹从一个区域到一个区域重复进行，直到找到了与一个物体的相交，或者光线离开了所占据的空间。

求相应于一个点 (x, y, z) 的八叉树中结点的最简单方法是使用代表八叉树的数据结构引

导结点的搜寻。从树的顶部开始， 对坐标的一种简单的比较将确定哪个子结点代表包含点 (x, y, z) 的子区域。相应于该子结点的子区域自身可以被再分，另一个坐标比较将确定其哪个子结点代表包含点 (x, y, z) 的子区域。搜寻沿着树进行直到达到一个终端结点。在这个搜寻期间穿过的最大的结点数将等于树的最大深度。即使对一个所占据的空间的相当精细的细分，搜寻长度也将是短的。例如，如果空间以1024 × 1024 × 1024的分辨率进行细分，则八叉树将有深度10(= $\log_8$ (1024 × 1024 × 1024))。

迄今为止，我们描述了一个简单的方法来使用对空间占用的八叉树表示，以加速跟踪一束光线的过程。Glassner（1984）和Fujimoto等（1986）描述了对这种基本方法的两个变种。Glassner描述了适合发现八叉树内相应于点 (x, y, z) 的结点的一种替代方法。实际上，它不显式地存储八叉树的结构，而是通过包含每个体素一项的一张散列表来访问关于体素的信息。散列表是用从一个点的 (x, y, z) 坐标计算的一个代码数进行访问的。 整个的光线跟踪过程如我们在基本的方法中所述的那样进行。

Fujimoto等（1986）描述了另一种穿过八叉树里的体素跟踪光线的方法。这种方法消除了
361 浮点的乘和除。为了理解这种方法，先从忽略八叉树表示开始。 我们首先描述一个空间细分的简单数据结构表示，叫做SEADS（Spatially Enumerated Auxiliary Data Structure，空间枚举的辅助数据结构）。这就涉及把所占据的所有空间都细分成相等尺寸的体素，而不考虑是否被物体所占据。以这种方法所获得的三维网格与将一个二维图形屏幕细分成像素所获得的网格是相似的。由于区域是在不考虑物体占据位置的情况下细分的，所以，SEADS方法细分所产生的体素比前面所介绍的八叉树方法要多得多。这样就涉及到对存储空间的“不必要”需求。但是，利用SEADS方法可以非常快速地从一个区域到一个区域进行光线跟踪。所用的跟踪算法是二维图形学中所用的DDA方法的一种扩展。DDA方法用于在二维图形学中选择代表两个端点之间的一条直线的像素序列。在二维图形学中所用的DDA方法是选择由一条直线穿过的像素的子集，但是，该算法也可以很容易地修改为求接触到该直线的所有像素点。Fujimoto等（1986）描述了如何将这种算法扩展到三维空间，并用于在一个SEADS三维网格中跟踪一束光线。3D-DDA 的优势是它不涉及浮点乘和除。所涉及的操作只是加、减和比较，主要的操作是在体素坐标系中的整数加。

完整的SEADS的沉重的空间负担可以通过返回一个空间细分的八叉树表示来避免。可以对3D-DDA算法进行修改，以便穿过八叉树在体素中间跟踪一束光线。在八叉树中，带有公共父结点的八个结点的集合代表一块八个相连的立方体区域，形成一个2 × 2 × 2的网格。当在这个集合中从一个区域到另一个区域对光线进行跟踪时，可以在不进行修改的情况下使用3D-DDA算法。如果一束光线进入了一个没有被树中的端结点表示的区域，但这个区域会被进一步细分，则可以通过沿着树向下移动而找到它所进入的子区域。可以通过在其上一层调节DDA算法的控制变量来发现下方的每一层所需的子结点。如果3D-DDA算法在跟踪一束光线的过程中超出了当前所行进的2 × 2 × 2的区域，则必须向上跟踪代表此完整区域的父结点。然后，3D-DDA算法在这一层次上继续进行，在含有父区域的八个区域的集合中跟踪光线。在树中，向上和向下的穿越涉及到对DDA控制变量的乘2和除2，但这是一个代价很小的操作。

最后，我们通过列出其最重要的高效特性来总结和比较这三种空间连贯性方法:

- 八叉树: 适用于空间密度变化很大的场景。因为低密度的区域将被稀疏地细分，而高密度的区域将被精细地细分。但是，在大的区域中也可能有小的物体。从一个区域向另一

个区域的移动比起其他两个方法来说要慢，因为树可能是不平衡的。

- SEADS：速度比八叉树要快，但是二次数据结构所需的存储成本很高。
- BSP：对于大多数场景，树的深度小于八叉树，因为树是平衡的。八叉树的分支可以是短的，对于具有高的空间占用率的区域也可能非常长。存储空间的成本一般来讲比八叉树要低。空的区域相对较小。

12.5.5 光线空间细分

这个奇妙的方法由Arvo and Kirk（1987）提出，它不是根据空间占用来划分物体空间，而是将光线空间划分成五维的超立方体区域。在五维空间中，每一个超立方体都与用于相交测试的候选物体列表相关联。这种方法消除了物体空间细分方法中为了要在物体空间中跟踪光线而涉及的三维空间计算。求出包含着光线的超立方体，就产生了可以与光线相交的所有物体的完整列表。这时，相交测试的成本换成了较高的场景预处理的复杂性。

可以把一束光线看成是五维空间中的一个点，它实际上是三维空间中的一条线和可以由单位球中的两个角定义的一个方向。代之以使用一个球来对方向进行分类，Arvo and Kirk（1987）采用了一个“方向立方体”（这与Haines and Greenberg（1986）所用的光线缓冲器刚好是相同的工具，见12.1.3节）。这样一来，一束光线就由五元组(x, y, z, u, v)再加上光线所穿过的方向立方体侧面的标签来定义，其中x, y, z是光线的原点，u, v是方向坐标。于是，就由一个五维超立方体的六个拷贝（每个方向立方体的侧面一个）定义一组具有相同原点和相同方向的光线。

这个空间根据物体的占用情况进行细分，对于所分出的区域建立起候选物体列表，用于细分的是一个“超八叉树”——一个五维的八叉树。

随着五维空间的细分，为了建立候选物体列表，必须在三维空间用到超立方体的三维空间的等价物。这是一个“束”或者是一个不受约束的三维体积，可以把这个体积看成是光线原点的体积和由一个光线原点以及与其相关的方向单元形成的方向金字塔的一种联合（见图12-10）。注意，三维空间中的束将处处互相相交，而它们在五维空间中的超立方体等价物并不相交。这正是方法的关键所在，可以对五维空间进行细分，用二元分区即可以完成细分。但是，这时的候选物体列表比物体空间细分方法要更困难了。束必须与物体的限定体相交。Arvo and Kirk（1987）报道说，测试多面体的相交代价更大，他们建议当束用圆锥来表示或限定并与作为物体限定体的球相交时进行近似。

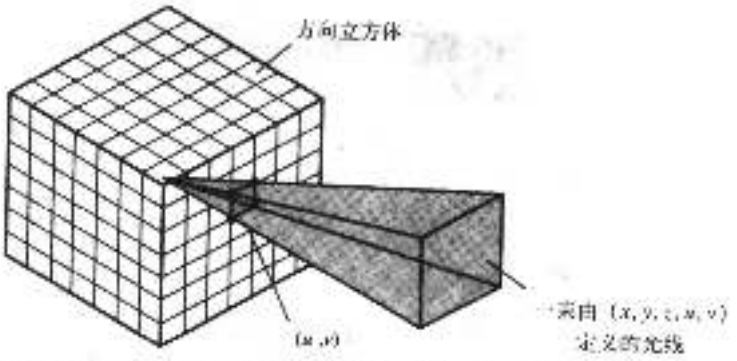


图12-10 在(x, y, z, u, v)空间中作为一点的一条（或一束）光线

12.6 利用光线的连贯性

到目前为止，我们考虑的是一条无限细的光线，并研究了尝试加速基本算法的效率问题。很容易看出，直到现在还没有论及的效率低的一个主要来源是缺乏对光线连贯性的使用。这仅仅表示，如果这种光线跟踪算法为每个像素产生一条光线，并分别跟踪每一条这样的光线，则我们并没有把初始光线是否倾向于沿着相同的路径运行这样的事实考虑在内。我们现在将考虑把一条光线"扩大"成一个几何实体的方式。

Heckbert and Hunraban（1984）利用了连贯性。通过观察可以了解到，对于任何场景来说，一条特定的光线有很多相邻的光束，每一条都倾向于沿着相同的路径前进。代之以跟踪单条光线，为什么不跟踪一组平行的光线，在一束光线之间分享相交计算？通过递归调用一种Weiler-Atherton隐藏面消除算法（Weiler and Atherton 1977）可以完成这一任务，Weiler-Atherton算法是一个投射空间细分算法，它包括一个初始的多边形深度的排序，以及一个由经排序的多边形之间相互裁剪而产生的片段的排序。最后，递归的细分用来整理出任何剩下的含糊不清的地方。这种方法限制物体是多边形的。如此一来就破坏了光线跟踪程序的重要的优势之一，即由于光线跟踪程序与相交测试的分离，不同的物体定义容易被合并。

最初的光束是观察半截体，这个光束在环境中被跟踪，并且用来建造一棵相交树，与单个的光线树不同的是，光束可能与很多表面相交而不是一个。树中的每个结点现在包含与光束相交的一个表面的列表。

程序在一个称为光束坐标系的变换坐标系中执行，最初这是观察或者眼睛坐标系。光束
364 是作为xy平面中的一个二维多边形沿着z轴平移而扫掠的体积。

反射（以及折射）通过递归地调用光束跟踪程序来建模。对每一个光束与物体相交都产生一个新的光束。任何反射光束的截面都是由入射光束和一个虚拟的目光点裁剪出的多边形区域所定义的（见图12-11）。

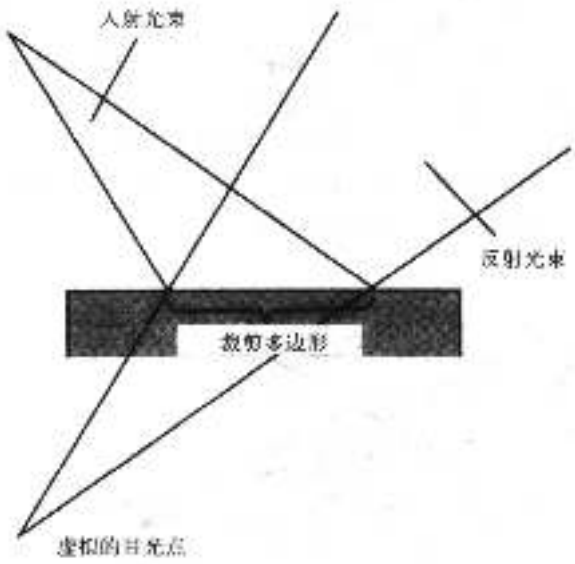


图12-11 光束跟踪中的反射

除对多边形物体的限制之外，这种方法还有其他的缺点。部分与物体相交的光束变成了

具有复杂截面的一束光束。截面可能是不连续的或者可能包含一个洞（见图12-12）。另一个缺点是折射是一种非线性现象，并且折射的光束的几何学信息将不被保存。因此折射必须使用线性变换对其进行近似。

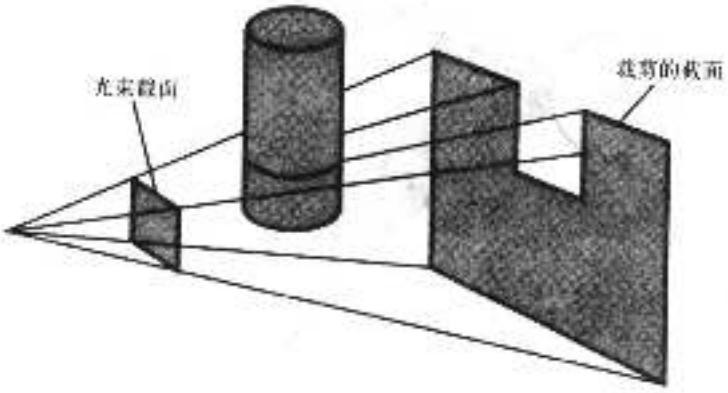


图12-12 部分与物体相交的光束产生一个碎的截面

另一种光束跟踪方法是Shinya等（1987）提出的铅笔技术。在这种方法中，由称为"近轴光线"的光线组成一支铅笔。这些光线是与一束叫做轴向光线的参考光线靠近的光线。用与轴向光线相关的一个坐标系中的四维向量描述一束近轴光线。在光学设计和电磁分析过程中众所周知的近轴近似理论用于在环境中跟踪近轴光线。这表明对于任何靠近近轴光线的光线，铅笔变换是线性的，并且是4×4矩阵。在近轴理论里的误差分析提供估计误差的函数，并且为铅笔的扩展角提供限制条件。

通过跟踪轴向光线确定4×4系统矩阵。铅笔里的全部近轴光线可以使用这些矩阵进行跟踪。近轴近似理论依赖于被平滑的表面，使得近轴光线不会因为遇到表面的不连续性而突然出现分裂。这是该方法的主要缺点。

Speer等（1986）提出了一种利用连续的光线产生的相交树之间的相似性处理光线连贯性的方法。这是一种直接进行光束跟踪的方法，其优点是它利用了光线的连贯性而不引入一种新几何实体来代替光线。这里的想法是试图使用以前的光线产生的路径（或者相交树）来建立当前的光线的树（见图12-13）。随着当前树的建立过程的进行，来自以前的树的相应分支上的信息可以用来预测被当前的光线撞击到的下一个物体。这表明任何"新"介入的物体必须被如图12-14中所示的那样进行检测。为了处理这种情况，在光线集里的每束光线周围建造圆柱体的安全地带。对于光线$_{i,j}$，其安全地带如图12-15所示。现在，如果当前的光线不刺透相应于以前的光线的圆柱体，并且这条光线与相同的物体相交，则它不能与任何新介入的物体相交。如果一条光线不刺透一个圆柱体，则像标准光线跟踪方法所需的那样，要进行新的相交测试，并建立一棵不同于以前的新树。

实际上，Speer等（1986）指出，这种方法受到通常的计算费用之苦。作为一个场景复杂性的函数，利用光线的连贯性特性所必需增加的复杂性比标准光线跟踪花费的要多，尽管光线的2/3表现出连贯性。对于这种情况，产生的原因是对安全圆柱体的保持并且进行刺透检测的费用，安全圆柱体的平均半径和长度随着一个场景复杂性的函数而减少。

365

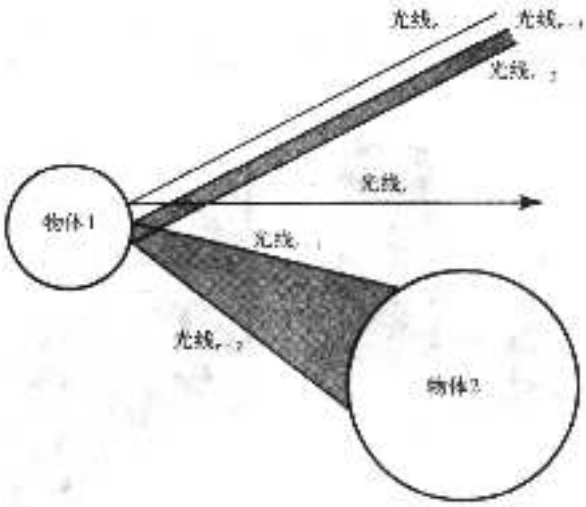


图12-13 光线连贯性：以前光线的路径可以用来预测当前光线的相交

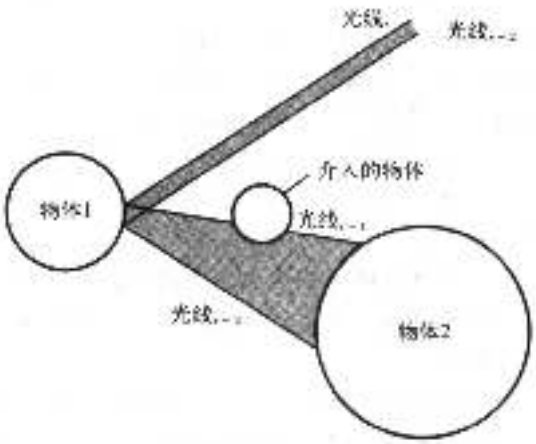


图12-14 在光线r的路径上“介入”的物体

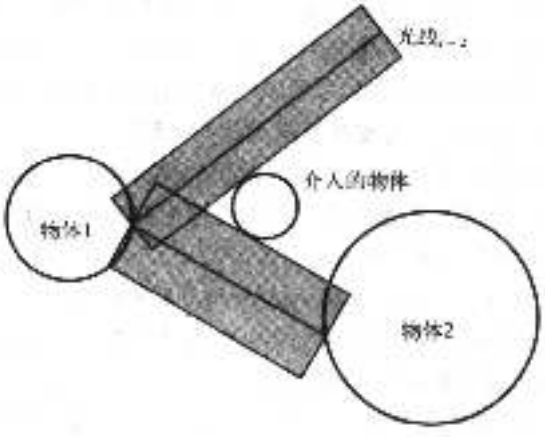


图12-15 圆柱体的安全地带

12.7 一个历史话题——彩虹的光学问题

很多人把术语“光线跟踪”与一种新奇的技术相联系，但是，实际上，它总是几何光学的一部分。例如，在René Descartes（笛卡儿）1637年出版的论文里发现有几何光学方面的早期使用，用来解释彩虹的形状，从与一只充满水的球形的玻璃细颈瓶有关的实验报告中。笛卡儿以光线跟踪作为一种理论的框架来解释这一现象。笛卡儿采用了已知的反射定律和折射定律，通过一滴球形的水滴跟踪光线。

进入一个球形水滴的光线在第一个空气与水界面处被折射。在内部的水与空气的界面处被反射，最后，当它们由水滴中出来时被折射。如图12-16所示，水平光线在水平直径上方进入水滴，并以相应于入射光线的角度不断增加的形式出来。出射光线的角度是入射光线在水平直径上方的高度的一个函数。该值向上到达某一最大值。当该行为反转，入射光线和出射光线之间的角度减少时，这个趋势继续，直到某一光线，这束光线称为笛卡儿光线，并且在这一点上入射光线和出射光线之间的角度是42°。靠近笛卡儿光线的入射光线也靠近于它出射，图12-16表现了在出射的笛卡儿光线周围光线的集中。正是这种光线的集中使彩虹可见。

366
-
367

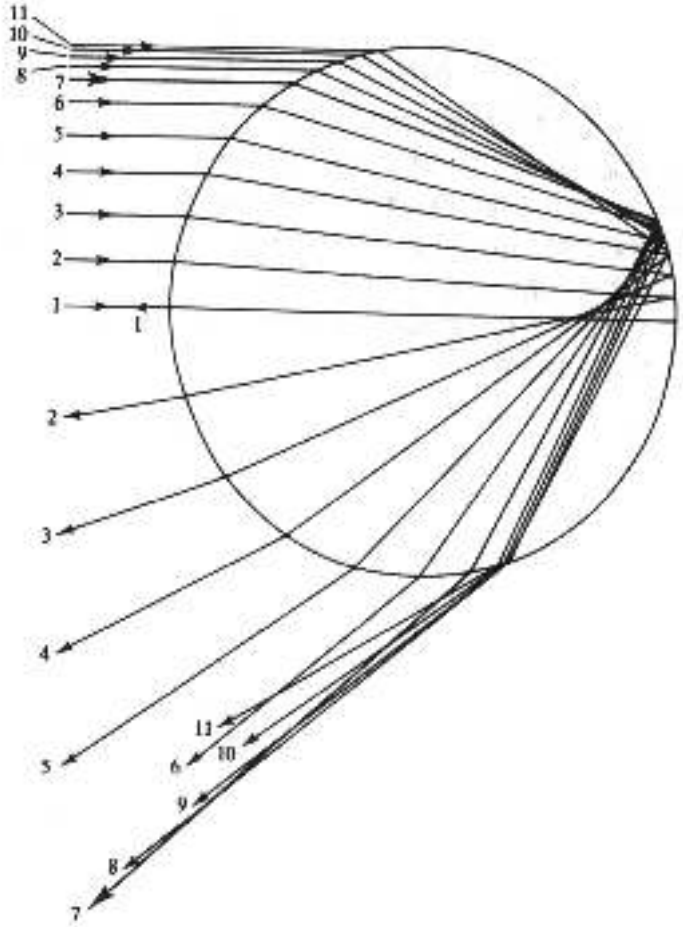


图12-16 跟踪光线通过一个球形的水滴（光线7是笛卡儿光线）

图12-17表明了彩虹的形成。一个把脸转过去的观察者从太阳看见由来自太阳的42°光线形成的彩虹。这种光线的路径形成了一个42°的以观察者的眼睛为中心的"半圆锥"（这个模型
368 的一个有趣的结果是每个观察者有他自己的个人彩虹）。

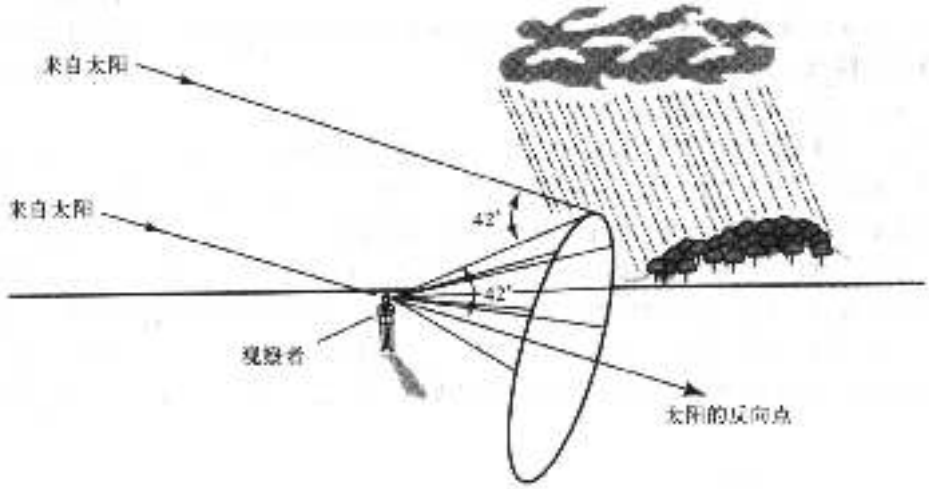


图12-17 一道彩虹的形成

不过，早期这种对光线跟踪的精巧使用并没有解释彩虹的不可思议的属性——颜色。30年之后，牛顿发现在白光中包含了全部波长的光。利用了任何材料的折射指数对于不同波长的光是变化的这一事实，笛卡儿的初始模型很容易被扩展。大约42°是红色光所取的最大角度，而紫色光线在被反射和折射之后在40°出现。此模型可以被看成是一组同心的半圆锥体，每一种波长一个，以观察者的眼睛为中心。

这个简单的模型也用来解释更微弱的二次彩虹。这个二次彩虹在51°发生，并且是在水滴
369 里面由于两次内部反射而形成的。

第13章 体 绘 制

13.1 体绘制和体数据的可视化
13.2 “半透明胶质”选项
13.3 半透明胶质和表面
13.4 体绘制算法中关于结构的考虑
13.5 体绘制过程中的透视投影
13.6 三维纹理和体绘制

引言

体绘制表明基于体素的数据绘制或可视化。在第2章中，我们介绍的数据表示技术是基于标明在一个被物体占用的物体空间中的全部体素的技术。我们看到在有些应用中大量的同类物体可以占用成百上千的体素，我们可以在数据上加一个层次结构，例如八叉树。另一方面，在像医学图像数据结构这样的应用程序中，数据结构可能是来自扫描仪的一个巨大的三维体素数组。在本章里，我们考虑可视化这样大型的非结构化的体素。

在过去大约10年的时间里，出现了一种新的学科，即科学计算过程中的可视化（ViSC）。主要的应用领域之一是三维空间变量的标量函数的可视化。这种数据在成为硬件和软件绘制时可用的数据之前，使用“传统”的技术（例如作为截面平面内的等轮廓线）进行了可视化。体绘制的出现意味着可以把数据看成是计算机图形学的对象，并且全部用三维显示。三维空间变量的标量函数在科学和工程中大量存在。工程师主要关心设计三维物体，并且分析它们的潜在行为。计算可以产生例如与温度和压力有关系的预示。

一个体素的体积可以由数学模型产生，例如像计算流体动力学中那样，或者从医学图像等真实世界中收集体素。可视化软件以同样的方式处理这两种类型。在不同的数据源之间，主要的实际区别是体元素的形状。在医学图像中，体素是矩形或者立方的。在其他应用中可能不是这种情况。在图13-1（彩色插图）所示的例子里体元素是楔形的，即一个圆柱体以“蛋糕的薄片”形式划分。

370

医学成像已经成为体绘制最常见的应用之一。它使得作为一组平行的平面从一个X线照相系统收集到的数据以一种三维的计算机图形学物体的形式进行观察成为可能。本章里的材料主要基于这种特定的应用。虽然某些依赖于上下文的考虑是必要的，但是医学成像问题是相当普遍化的，任何对于这种情况发展出的策略都很容易适用于其他应用程序。

在医学成像过程中，三维的数据可以从许多平行的CT（计算的X线体层照相术）数据栈中获得。这些系统根据一些特定的性质重建或者收集数据，原先的形态是平面中的每个点都是X射线吸收系数。基本的医疗系统使临床医师能够在每个平面里查看信息。用可视化方法，所有的平面数据被看成体数据，并被相应地绘制。

一个X线体层照相图像系统的非常简化的示意图如图13-2所示。从这张图中可以看到，信息是在很多零厚度的二维平面上采集的。从这些数据中导出体素数据值。这些数据表明了这

样一种性质，即一个平面内的分辨率（通常为512×512）比两个平面之间的分辨率要大很多。扫描一般只是以0.5cm的距离为间距的。然后，将这些数据解释为一组体素，每一个体素代表一个X射线吸收数据系数。体绘制正是基于这些数据的。当前，重建X射线体层照相的系统与可视化系统是独立的，前一个系统一般只是用来做诊断的，下一节我们将继续讨论这个问题。

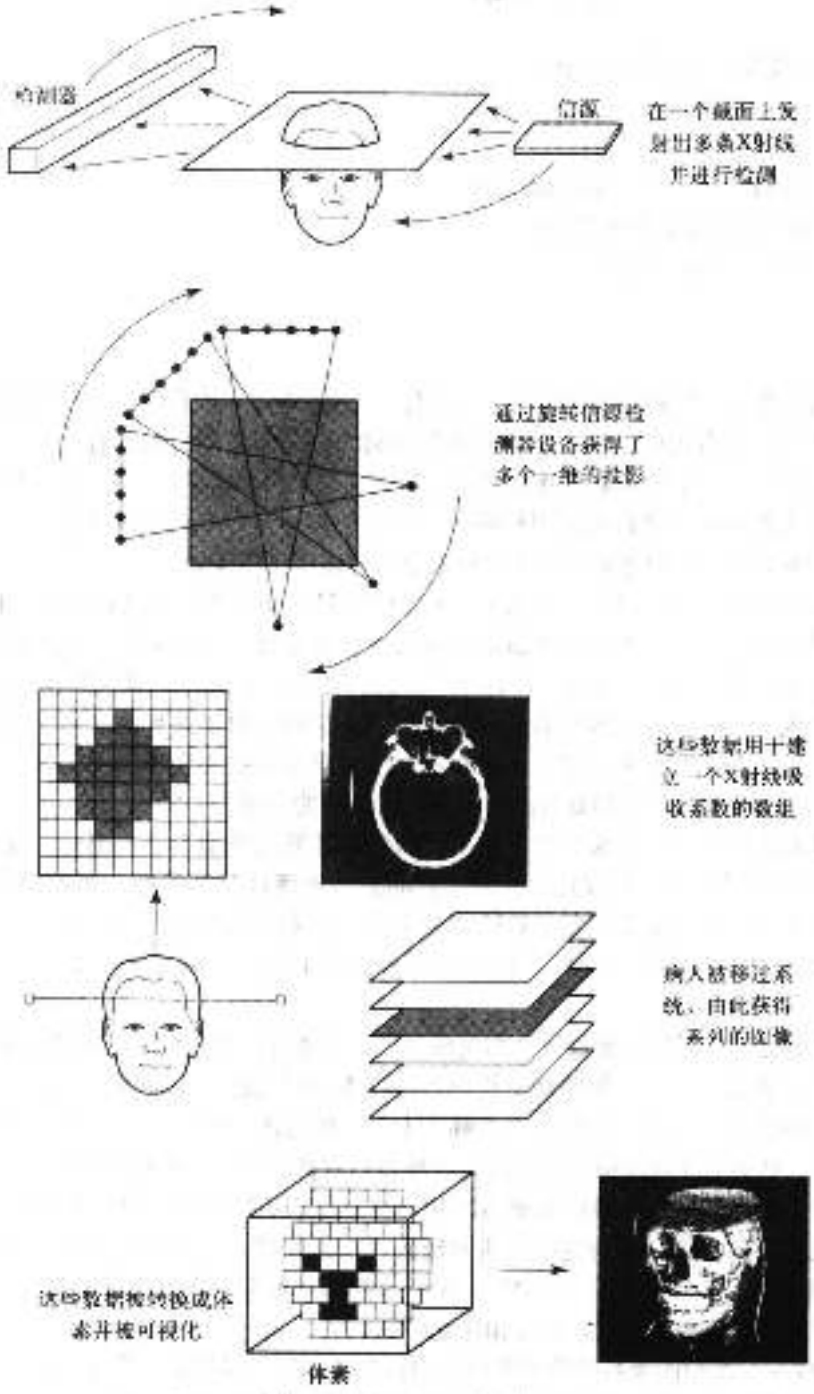


图13-2 X射线计算机体层照相和体绘制

在医药界最令人注目的项目之一是人体可视化项目（1998）。这是由MRI和X射线CT以及由解剖尸体获得的解剖学图像组成的一个15GB（男人）和40GB（女人）的体素数据集。可视化人体项目的初始目的是获得以平均1mm为间距的一个代表性男人和女人的解剖学人体的截面CT、MRI以及低温断面图像。在这三种形式的断面中的相应断面都互相进行了记录。

可视化男人体数据集包含了头部和颈部的轴向MRI图像，身体其他部分的纵向断面是以4mm为间距获得的。MRI图像的分辨率是256 × 256像素 × 12位。CT数据是以512 × 512像素 × 12倍为分辨率，按1mm的间隔对整个人体进行轴向CT扫描得到。轴向的解剖学图像为2048 × 1216像素 × 24位。这些数据也是以1mm为间隔的，并且与CT轴向图像共线。对于每一个轴模、CT和解剖学都有1871个截面。

可视化女人体数据集与男人解剖体有相同的特性，只有一个例外。轴向解剖数据是以0.33mm为间隔获得的，而不是1mm的间隔，以便构造立方体体素。这导致了超过5000幅解剖学图像的产生。 371

这一项目的长期目标如下:

> 可视化人体项目的数据集被设计成一种用于研究人体解剖学的公共参考点、一组用于测试医学成像算法的公共领域数据、一种试验床，以及作为构建可以通过网络访问的图像库的模型。这些数据集应用于各种领域，包括教育、诊断、治疗的规划、虚拟现实、艺术、数学以及工业等方面。在41个国家中发放了超过1000个许可证。但是，在发展将这样的图像数据与基于文本的数据相关联的方法时还是遇到了一些关键性的问题。对于这种关联的标准当前并不存在。对基于图像的结构的表示和描述以及把基于图像的结构-解剖数据与基于文本的功能-生理学数据进行关联方面还需要有一些基础的研究。可视化人体项目的较大的、长期的目标是：透明地将功能-生理学知识的书面数据库与结构-解剖知识的图像数据库连接起来构成有关健康信息的统一资源。

13.1 体绘制和体数据的可视化

体绘制的基本思想是观察者应该能够从观察平面上绘制的投影感觉到数据的体积。在医学成像中，我们可能希望观察一个表面或者一个体积，也可能希望观察体积上的一个部分。

因此，作为体绘制问题的一部分，我们是观察存在于数据中的“硬”表面的一种抽取和显示。在很多情况下，我们可能有一个体数据集，必须从这个数据集中抽取并显示出在该体积中处处存在的那些表面。并不是要限定物体的表面，我们可以处理那些具有很多“网状”表面的物体，比如洋葱皮。如果可以通过某些唯一的性质来提取这样的表面，那么我们就可以通过使它们完全不透明而把它们绘制成可见的，而这时该体积中的所有其他数据都是完全透明的。

现在，我们将把医学的例子进行扩展，考虑作为三维体数据的CT胶片的堆栈的可视化问题。在这一章中所介绍的所有技术都可以应用于大多数体数据，这些数据都或多或少是完全通用的。很容易考虑在特定应用领域中各种不同的可能性。

正如我们曾经提到的，所重建的CT数据由一些无限薄的胶片或二维数组构成。在实际应用中，胶片之间的距离比胶片中一个像素的尺寸要大。为了将这一堆栈转换成一个立方体体

素的规则的三维数组，我们必须引入某种形式的插值。然后，可以考虑各种可能性，或者考虑这个体数据的各种显示模式。

372～373 对于任何应用，由于我们处理的是体数据，可用的选择就比对一个物体表面的绘制多得多，显示的特定模式将依赖于应用。这些需求的性质以及数据的性质确定了所用的算法。

对于在二维的显示表面上显示体数据，三个可用的基本选择是:

1) 用截面平面将数据集切成片。很明显这是最容易的选择，但如果这个平面平行于体数据集的坐标平面的话，它也是繁琐的方法。由于它只能有效地显示数据中的两维，所以它也是最不需付出很大努力的方法。

2) 把已知存在于数据集中的物体抽取出来并以常规的方法进行绘制。于是，物体的内部器官可以以孤立的形式显示出来，就好像被解剖出来了一样。这首先意味着物体可以从剩余的数据中分割出来。其次，这种分割出来的形式可以转换成一种计算机图形学的表示。

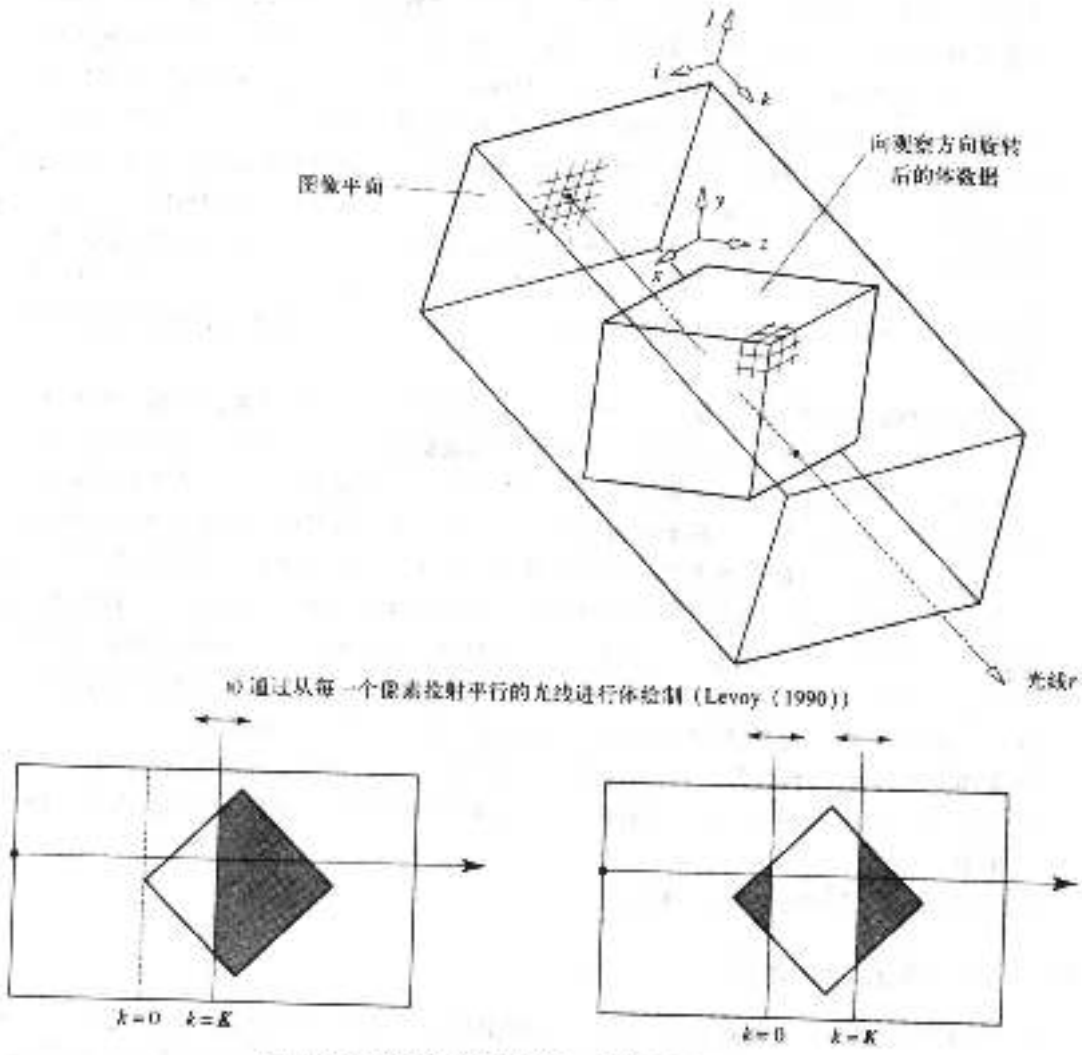
3) 给物体中的体素赋透明度和颜色，然后就可以从任何角度观察整个数据集。这一过程通常称为体绘制。在医学应用中，它有时又被称为计算的X射线，因为它与传统的X射线相似。换句话说，有可能从任何观察角度产生计算的X射线，甚至包括从物理上来讲用传统的X射线设备不可能出现的那些角度。由于有选择任意观察角度的自由，我们还可以以任意所希望的方式改变物体的不透明性。

当前，体数据可视化主要应用在医学的示意图方面，以交互式图谱集的形式用于医学教育、医学研究、治疗方案以及计算机图形学的研究。对于诊断方面的应用，医师似乎更喜欢一面一面地研究原始的解剖图片。这部分是由于我们将要了解到的，在处理过程中涉及了插值算法，而这样一来就可能会对原始数据的完整性有所干扰。例如，在第二个方法（从数据中抽取物体）中，可能会在表面上填上小洞。

很多质量问题都可以在原始数据的收集阶段（对物体的扫描阶段）得到解决，这一事实更强化了体数据应用与医学示意图之间的联系。但是，与此相关的限制也是存在的。在X射线CT扫描的情况下，病人接收到的X射线剂量是分辨率的一个函数，包括重建平面中的空间分辨率和所收集的平面的数量。增加分辨率意味着向病人投射更高的X射线剂量，而这就会比传统的X射线的剂量要高。这就是高质量的图像一直是由解剖尸体获得数据的原因。

由体数据集建立医学图片集导致了上述三种显示方法创造性的结合的大量出现。普通结合的例子如图13-3（彩色插图）所示。前两个例子显示的是把抽取出来的物体安放在一个透明的骨架中。抽取出来的结构被转换成了计算机图形学物体，并正常地进行了绘制。然后，
374 再将其有效地植入到三维数据体中，三维数据体是通过设为半透明值的周围的体素来显示的。半透明的体素可以设为灰度级，以对X射线进行模拟，或者也可以将其设为任何希望的颜色。可以采用方法3来实现这一点。这时，将物体的体素设为不透明的。但是，正常情况下，通过对所感兴趣的物体进行传统的绘制至少在医学示意图的层面上可以获得较好的结果。产生这类示意图的动机是明显的，即它可以增亮物体，并相应于身体或皮肤对其进行定向。另一个流行的结合（第二个例子）是把皮肤的绘制结果切掉，把内部的组织以定位在三维模型中的截面的形式显示出来。在这里，通过增亮器官的形状而赋予其一种适当的“伪颜色”。这种颜色可以是纯粹的假色，以指出或标识所感兴趣的结构。可以以某些人可理解的形式将颜色与相关的体素的值相联系。一个标准的色度集可以反映比如说吸收系数的值。

体绘制的总的思想如图13-4所示，这是一个光线投射算法，它表示的是将一个以立方体



a) 通过从每一个像素投射平行的光线进行体绘制（Levoy（1990））

b) 用平行于观察平面的平面建立一个数据集的视见体

图 13-4

表示的体积数据集旋转到希望的观察方向，并与一束平行的光束相交。每一个像素一束光线（术语“光线投射”用于与光线跟踪算法中的光线相区别。在本例的情况下，光线是以一个平行束的形式前进的，并穿过体积，而不是在碰撞之后发散）。这种方法是一种有用的概念的起始点。在实际应用中，执行这一方法有很多不同的方式。现在，我们将讨论下面的一般选项，以及关于这种算法的考虑。

375

- **在图像平面中我们想要看到的是数据的什么性质？** 我们可能希望看到明暗处理过的物体的外部边界表面。在医学成像中，这可能是皮肤表面。而这就是说，我们必须“发现”这种表面，并对其进行明暗处理。在光线投射的情况下，可能只需要在光线碰到了第一个非零的体素时终止光线，也就是要估算体素的表面法向，并应用一个局部明暗处理模型。另一种情况是，我们可能把一个内部的物体进行可视化，并对其明暗处理。在医学

成像中，我们还可能希望看到皮肤/肌肉层之下的骨骼结构。而这就意味着我们在绘制计算机图形学物体之前必须从数据集中抽取出这样的表面。我们还可能希望在数据中移动一个横断切平面，如图13-4b所示，并观察随着切平面的移动，切平面与数据之间相交得到的内容。也可能希望看到骨骼（例如肋骨）以及其中所含的器官。这些既可以通过把骨骼绘制为不透明的，以便使观察者透过骨头之间的缝隙看到器官，也可以通过把骨骼绘制为半透明的来实现。很容易想象出其他可能的结合方式。我们可以对于每一个像素组织一个投影。它是光线路径上所遇到的最大数据值，一个不太明显的方法是显示每一条光线路径上的总和。这时，方法会给出与传统X射线相似的图像，这给了我们一种能产生（虚拟的）可能用传统的方法不可能实现的某个观察角度上的X射线型图像的手段。

- **真实性和数据之间是什么关系？**一般情况下，体数据将由点的三维数组构成，它代表了对真实世界的三维采样。这可能是一个非常大的数据集，比如说512^3。我们把单个样本与整个的体素体积相关联，正像在二维图像处理中，一个样本与一个正方形的像素面积相关联一样，但是，这一个样本代表了什么呢？在这里，较简单的情况是把样本看成是二进制的空间占位。假设体素的分辨率足够高，使任何体素只含有一个材质或者什么也不含。另一种选择是，考虑一个体素含有各种材质的混合。在医学成像中，可能的情况是一个体素的物理范围相应于一个区域，这个区域可能会跨越骨头和体组织两个区域。我们是否要把体素的值看成在其整个的范围内是常数？或者考虑该值在范围内发生变化？如果是后者，我们要用什么模型对相邻体素之间的变化进行插值呢？
- **体素的大小蕴涵了什么呢？**与传统的表面绘制方法不同，在这个方法中我们对每一个像素都定义了与一个物体的关联。就好像一个体素的整个范围向图像平面的投影将占据很多像素。

现在让我们来较详细地讨论这些问题。

13.2 “半透明胶质”选项

最一般的观察选项是给观察者能看到所有数据的能力。没有任何体素被认为是完全不透明的，因此可以看到所有的数据。其物理的相似物是由不同颜色的透明胶质物构成的物体。每一个体素都被赋予了一种颜色C、一个透明度α。与材料类型相关联的颜色可以选择为“艺术的”。在CT的例子中，可能对骨骼选择白色，将透明度设为正比于密度，以便使骨骼能够变成完全不透明的。

然后，我们从每一个像素投射出一束光线到数据体，这个数据体已经被旋转到了希望的观察方向，并执行一个合成的运算。这就为该像素累积了一个最终的颜色和不透明度。这个过程就好像把体看成是由具有不同颜色和透明度的半透明胶质做成的。就好像在体的背面有漫反射的白光，而我们正在从前面向内看。这个过程与在观察方向对体做一个传统的X射线相似。但是，现在我们是在体中传播平行的光束，并显示出结果，体的不透明性与组织的密度有关。

在Jahn Hopkins医学研究所的临床应用中，Ney等（1990）说到：

> 用这种不经明暗处理的绘制过程产生的图像使人联想到一种传统的放射线图像。这些图像对于检查骨骼中的不正常现象尤其有用。骨骼是半透明的，因此，其内部

的细节是可见的，表面的细节也是可见的。可是，这种不经明暗处理的技术不太适合软组织成像。骨骼密度的高度可变性使得不经明暗处理的算法能产生可感知的细节。软组织变薄的值被限制在一个非常窄小的光谱范围内，使其很难将比如血管或结节与相邻的肌肉相分离。

376 ≀ 377

于是我们看到，这个可视化涉及到一些步骤：

1) 对原始数据中的每一个体素进行分类，并对其赋予希望的颜色和不透明度值。

2) 把（已经分类后的）体数据变换到观察方向。

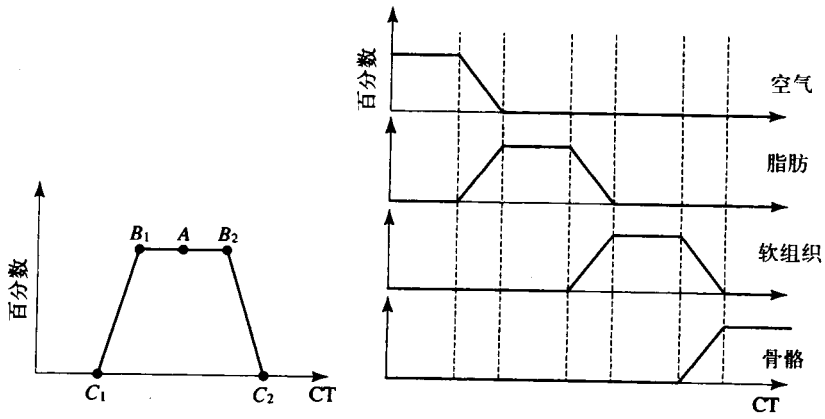
3) 对于每一个像素投射一束光线，并通过沿着光线合成求出该像素的颜色。

下面分别对这些步骤进行描述。

13.2.1 体素分类

考虑一个含有多个组织类型的体素的更一般的情况。Drebin等（1988）介绍了一种典型的分类方法（对于X射线CT数据的特殊情况）。在这种方法中，根据X射线吸收系数的值将体素分成四类。这些类是：空气、脂肪、软组织和骨骼。这种方法称为“概率分类”。它假设在一个体素中存在两种但不多于两种的材质。于是体素由七种类型组成：空气、空气和脂肪、脂肪、脂肪和软组织、软组织、软组织和骨骼、骨骼。只有吸收系数等级上的相邻材质之间才可能进行混合，例如空气决不会与骨骼相邻。

这种分类方法采用一种分段线性的“概率”函数（见图13-5）。考虑分配了这样一个函数的一种特定的材质。将存在一个最能代表这种材质的特定的CT数（图13-5a中的点A）。点B_1和B_2代表仍然被认为是这种材质的情况下CT数的最大偏移。任何小于B_1或大于B_2，以及位于由C_1和C_2限定的界限之内的CT数都被认为是相邻材质的一种混合。完整的方法如图13-5b所示。体素根据某些方法被赋予（R, G, B, α）值，如果在一个体素中存在着两种材质的一种混合，则按照材质的比例对两种颜色进行混合。



a) 对于一种材质的梯形分类函数　　b) 分类函数

图13-5 CT数据中的材质分类（Drebin等（1988））

13.2.2 变换到观察方向

从理论上讲，对于一个简单的过程这一步会有困难。图13-6所示为一个观察过程的简单

示意图。一般来讲，数据体可以朝任何希望的方向旋转，当像素光束投射到旋转体中时，这就会涉及到重新采样的操作，同时也必须考虑到走样的问题。在体绘制算法的整个建立过程中，主要的选项之一是这种转换执行的方式，以及在13.2节中所述的二个步骤的顺序中它所处的位置。

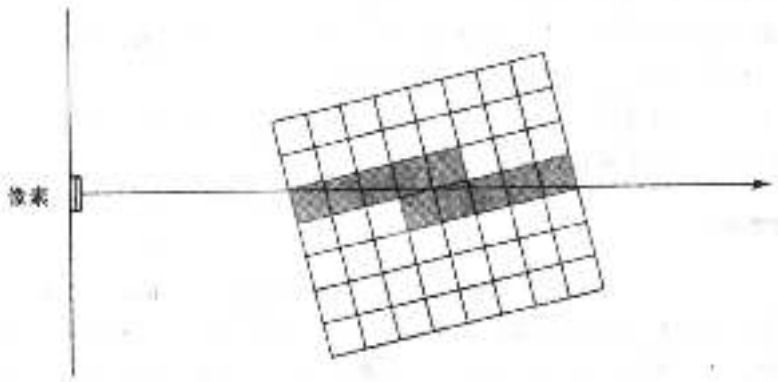


图13-6 光线投射暗示了对数据重新采样。一般来讲，一束光线将不会与体素的中心相切

在CT的例子中，只有关于z轴的旋转（脊柱旋转）和关于x轴的旋转（醒筋斗旋转）才是有用的。这意味着体的旋转可以通过对垂直于这些轴的二维平面进行旋转来完成。

13.2.3 沿着光线合成像素

最简单的合成操作（见图13-7）递归地应用下面的公式：

$$C_{out} = C_{in}(1-\alpha) + C\alpha$$

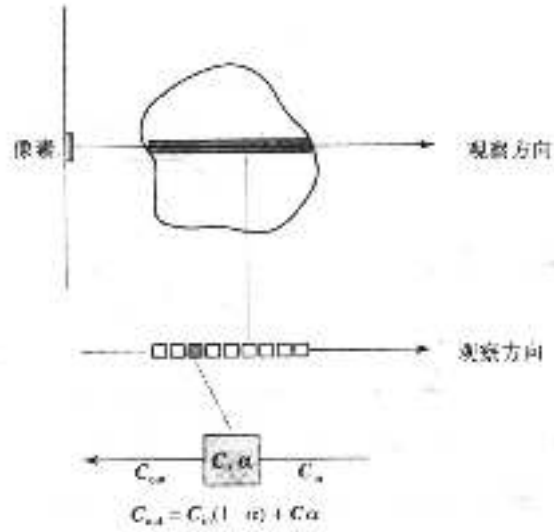


图13-7 光线合成操作

其中：

C_{out}是从一个体素出来的累积色彩;

C_{in}是进入该体素的累积色彩;

α是当前体素的不透明度;

C是当前体素的颜色。

注意，这种形式只是6.6.3节中合成两个图像所用的操作的一种扩展。C_{out}和C_{in}所确定的方向是相应于观察平面从后向前的方向。也就是说，操作从相距观察平面最远的体素开始。

在这个模型中光线来自何方无关紧要。我们只要注意到任何沿着观察方向从我们所关注的体素出去的光线都具有该体素本身的颜色再加上入射光与（$1-\alpha$）的乘积即可。对这个简单的模型可以进行精心的计划。例如，由于α将根据体素颜色的R、G、B分量的不同而不同，所以在真实情况下α应是一个向量。这个操作的效果是先使具有高α值的体素占主导地位，挡住那些其后面的体素，并使其前面的体素是可见的。

13.3 半透明胶质和表面

如果我们假设不透明的表面在数据体中是存在的，则应该对前面的方法补充一个明暗处理方案，根据13.1节中所述的各种选项作为显示的一部分表示这些表面。假设一个体素可以包含一个表面的某部分，我们可以估算一个法向，以这个法向和照明光源的方向的一个函数的形式计算出明暗处理分量。这个明暗处理分量可以在合成操作中代替C。

这时，随着光照模型在表面上对细节的增强，在法向方向感觉到表面的形状。现在有各种选项，我们可以比如说只显示那些含有骨骼的体素加上其表面形状的细节，使其通过软组织的一块模糊云隐约可见。然后，还可以使得骨骼是完全不透明的，或者仍然给它一个不透明度以便骨骼后面的细节可见。

378 ~ 380

通过用梯度的方法求一个法向来检测表面。这个法向的分量为:

$$N_x = R(x+1,y,z) - R(x-1,y,z)$$

$$N_y = R(x,y+1,z) - R(x,y-1,z)$$

$$N_z = R(x,y,z+1) - R(x,y,z-1)$$

其中对于每一个体素，都由对体素中每种材质的百分数与为其所赋的密度值的乘积进行求和来计算R。如果材质是均匀的，则这些差值的计算值为零，所考虑的体素就被认为不含表面的片段。这种方法的示意图见图13-8。

一个表面的存在通过表面法向的大小进行量化，这个值越大则表面越有可能存在。表面大小或“强度”（$|N|$）可以用于对明暗处理分量的贡献进行权重计算。对于表面的存在与否不进行二元决策。计算出来表面法向的单位化向量，可以用于像Phong反射模型这样的明暗处理方程中。我们必须牢记，这个技术完全是用于可视化的，它与真实物理世界绝对没有关系。假设每一个体素都有光源对其进行未被干扰的观察，即使它隐藏在体的中间也是如此。

这种运算的局部性意味着它对于噪声是敏感的。可以通过降低其局部性来消除这一作用。在上面的公式中，梯度是考虑了六个相邻的体素后估算的。我们可以将其扩展到18个甚至24个体素。

我们已经通过计算含有表面的体素的法向与光源之间的相互作用对表面进行了明暗处理。这时，表面的形状细节就变得可见了。我们还可以将明暗处理的表面与半透明胶质模型相结合，也可以使表面不透明，并去掉所有不含表面的体素。这就使得光线首次击中的表面成为

观察者可以看见的表面。这些选项示于图13-8中。

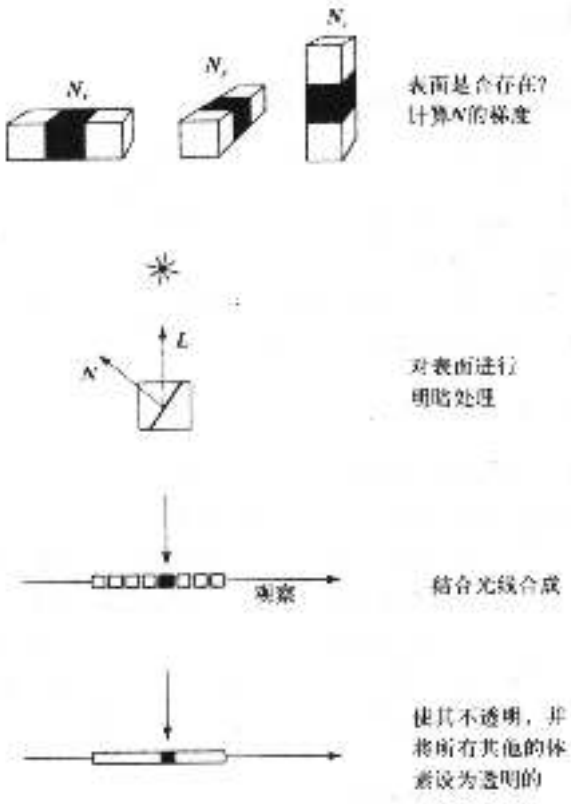


图13-8 表面检测和明暗处理

重要的是要认识到表面检测是局部的，是对于单个体素计算的。如果将明暗处理分量结合到半透明的胶质模型中，则不必确定表面存在与否。这在医学应用中是重要的，因为医师们对于那些对表面进行有关是否存在的二元决策的方法持怀疑态度。然而，有一些应用对于这种明确的抽取（假设的）连续表面的方法是需要的，我们将在下一节对此进行介绍。

等值表面的明确抽取

如果体数据中被认为包含有连续的等值表面，则可以将这些表面明确地抽取出来，将其转换成多边形网格的结构，并用常规的方法对其进行绘制。这样一种方法对于每一个体素求出一个或多个适当的多边形，并且从含有表面的体素集产生这种多边形的连续集。

所以，当我们可以用密度梯度的方法在体中找出并且明暗处理表面时，为什么还要费力去找出多边形网格的表面？其中的一个动机是如果表面是用传统的图形学基元表示的则可以使用传统的绘制技术，这时的体绘制就降为表面抽取的预处理操作。

所采用的技术称为步进式立方体算法，该方法由Lorenson and Cline（1987）提出。一个实际的表面是通过在那些可能含有表面的体素中拟合一个或多个多边形来建立的。一个体素具有八个顶点，如果我们假设一开始体素可以跨在表面上，则可以为这个体素赋一个多边形，

赋值的方式依赖于顶点处值的构成。这样的处理意味着在立方体的八个顶点上表面内外的顶点的分布。如果进行了某些假设，则可能有256个选择。从对称性的角度来考虑，可将这些可能的选择降低到15个，这15种情况如图13-9所示。每一种体素类型内部的每一个多边形的最终位置和方向由顶点处场值的强度确定。由一个正常的多边形网格构成了一个表面。在绘制这样一种表面与由体绘制中赋予适当的不透明度而抽取出来的效果表面之间，质量上的差别是由于体绘制方法分辨率不够而产生的。在体绘制方法中，一个表面可以存在于一个体素中的某一部位上。将这样一种体素的不透明度赋值为1，则关于这个表面片段的位置和方向的信息化简为该表面的法向。在步进式立方体算法中，表面片段在体素中至少在插值算法所用的限定范围内被精确地定位和定向。但是，明确的表面抽取方法有时会由于假设一个表面在相邻的体素中存在而产生错误，可能会出现对实际上是相邻表面的片段进行拟合的情况。换句话说，该方法进行的是二元决策，这种决策可能是错误的。步进式立方体算法的另一个问题是，可以产生的基元的透明体，当有很多基元投影到相同的像素时可能会产生成千上万这种体。

381
382

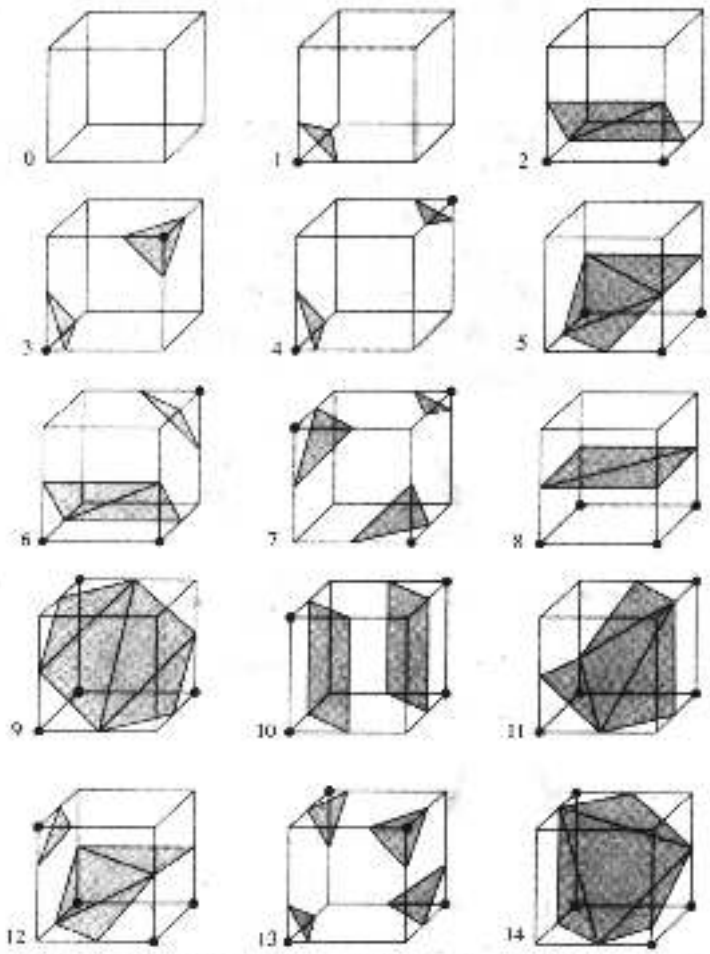


图13-9 在步进式立方体算法中的15种可能性。图中所用的点（•）代表在表面内部的一个顶点

图13-10（彩色插图）和图13-11（彩色插图）将绘制一个感兴趣的物体的两种方法进行了

比较。原始的数据是23个X射线CT数据的平面，每个平面的分辨率为512 × 512。图13-10所示为一个用步进式立方体算法绘制的颅骨。第二个图（图13-11）采用的是完全相同的数据，但是是用体绘制算法绘制的，将骨骼的不透明度设为单位值。在再现上尽管不是很明显，但是步进式立方体算法看起来质量或分辨率更高一些，这是算法的一种虚假的结果。所访问的是相同的数据，但对每个体素建立起了一个或多个多边形的明确的计算机图形学模型。体绘制算法只是基于局部信息将法向赋予每一个体素。

13.4 体绘制算法中关于结构的考虑

在建立体绘制算法时可以有许多选项。正如我们已经看到的那样，观察一个体数据的过程从概念上看是简单的，它涉及把体积朝观察方向进行旋转，然后向体积内投射光线或者进行等价的操作，以便为每一个像素找到一个适当的值。体绘制方法上的主要研究工作来自对硬件实施的效率的重要性。在大多数应用领域中交互性和动画是重要的，这是因为它们对数据解释的贡献。因为我们一般会处理非常大量的数据集，一般为512^3，所以若有对满足交互/动画的需求，算法的设计和可用的硬件（比如并行处理器）之间的关系就变得至关重要了。

在体绘制中描述算法的选项所用的术语有些混淆。混淆似乎来自为主要的分类所起的名字上。有两个主要的类别:

1) 光线投射方法（有两个变量）。也称为图像（或像素）空间遍历或后向投影。

2) 体素投影方法（有两个变量）。也称为物体（或体素）空间遍历或前向投影。

这些选项示于图13-12中。在光线投射中，我们可以对体数据进行变换并重新采样，以便使其与平行于图像平面的坐标轴定向。或者，可以保持体数据而不对其进行变换。如果数据在光线投射之前进行了变换，则会产生一组平行于变换后数据的行（或列）的光线。对于未经变换的数据，光线集应对观察变换求反。光线投射方法还可以被分类为图像空间方法，因为这类方法中算法的最外层循环是在图像空间中进行的。

383 ~ 384

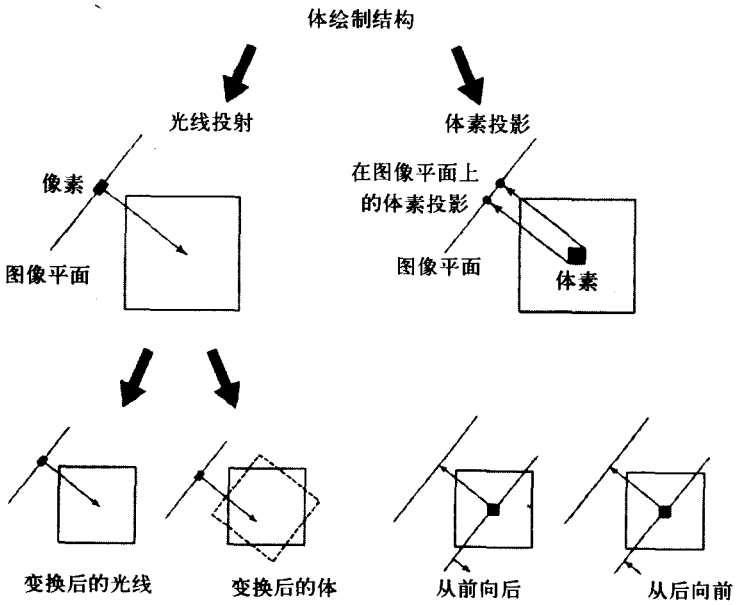


图13-12 体绘制结构的分类

尽管乍一看好像光线投射方法可以并行地实施，会产生存储的瓶颈问题。但如果允许有任意的观察方向的话，就不可能在存储器中出现对体素的分布，以保证不出现竞争。

前向投影方法的一个潜在的问题是在图像平面上可能会出现孔洞。对于体素投影方法，我们必须记住在大多数的应用中，单个体素将在图像平面上形成分布在许多像素上的一个投影（这种情况被称为脚印）。如果忽略透视投影，则这个脚印对于所有的体素（对于一个已知的观察来说）都是一样的。可以利用这种连贯性，这有利于快速实现和高效反走样。现在，我们将更详细地考虑这些选项，方法之间重要的差别在并行执行的适用性以及如何实现再采样方面得到证明。

13.4.1 光线投射（未经变换的数据）

在光线投射方法中，我们游历于图像空间，从每一个像素投射出一束光线，并通过执行前面介绍的合成运算求出该像素的一种颜色（这种方法与光线跟踪方法几乎没有关系，光线跟踪方法是依据所击中的物体的集合形状和属性在场景中的任意方向上跟踪一束像素光线。 385
在体绘制方法中，我们投射一组平行的像素光线，使这组光线总是朝着相同的方向前进）。为了完成这一任务，必须执行两个非平凡的任务。首先，必须求出那些光线所穿过的体素。其次，必须由分类的数据集为每个体素求出一个值。

考虑第一个问题。这个问题本身可以分解成两个部分。求光线所穿过的体素是一个已经完成了的任务，我们只需要使用3DDA方法（三维微分分析器），这是多年来在处理二维的直线/像素问题中得到的知识在三维空间中的扩展。但是，当我们求出了这些体素之后，该如何来处理它们的值呢？如何获得插入到合成方法中的值呢？如果采用所遇到的每一个体素的基本值就错了。这其中的一个原因是显而易见的。穿过每一个体素的路径长度是变化的，从一个非常短的距离（即光线仅仅切入到一个体素的角落）到一个很长的距离（即光线靠近对角线穿过）。我们正在沿着光束进行有效的观察，穿过体素的长路径所产生的对合成的贡献要比短路径高。这当然是用一个无限细的光束对实际的体数据进行采样的结果之一，或者是用更精确的重新采样的结果。这是在图像处理中等价于重新采样处理的一个三维问题。我们从已采样的数据开始处理，将其旋转到一个新的方向，并对其重新采样。当进行重新采样时，为了防止走样我们必须进行过滤。体绘制方法的复杂之处在于数据是三维的，而重新采样也是在三维空间进行。因此，继续进行下去的一个适当的方法就是沿着光线测量相等的点，并通过在一个三维的区域上进行过滤来求出这些点上的重新采样值，这是用等空间距离的光线采样作为三维过滤器内核的一个中心。

这一算法有时被描述为一个图像空间的遍历算法，其最外层的循环通常被定义为“为每一个像素投射一束光线”。然而，我们必须认识到，不可能比向体投射一组穿过数据中的每一个体素的平行光束更好。得到这种效果的一个简单方法如图13-13中的截面所示。光线的集合是由穿过数据集的前表面中每个体素的中心点的光线组成的。

Yagel等（1992）采用了相同的概念，他采用了一种光线“模板”的思想。光线模板方法采纳了沿着一条称为基平面的线一次使光线移动一个体素的简单方法。因此，光线或光线模板只计算一次，并被保存在一个数据结构中。然后，通过从这些信息获得适当的位移来跟踪所有的光束。图13-13中经明暗处理的体素形成一个光线模板。事实上，这个方法利用的是光线之间的连贯性。

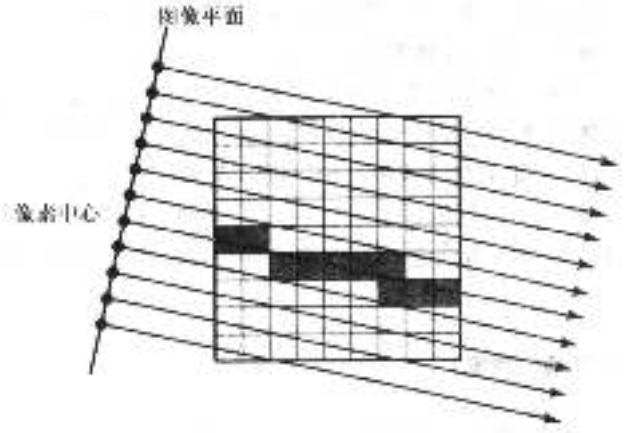


图13-13 光线投射算法中的一个适当的光线集合

现在，我们来考虑重新采样的问题。如果体没有动过，则绘制（或合成）过程和重新采样过程就合并成了一个操作。在相同的采样点上沿着光线前进，并且对于每一个采样点估算一个C_i以便在合成时使用。我们只需用一个C值，该值是含有采样点的体素的值。但是，正常情况下，采用了更精确的三次线性插值过程。图13-14所示为这个过程的截面。图中，此过程变成了在二维空间中的双线性插值。为了计算C_3值，从周围的网格点进行插值。先估算光线与体素的网格线的水平和垂直的相交点，然后，就可以求出C_3值。这个过程是在多边形明暗处理方法中所用的对于多边形为正方形的情况下的双线性插值方法的一种简化（见第1章）。

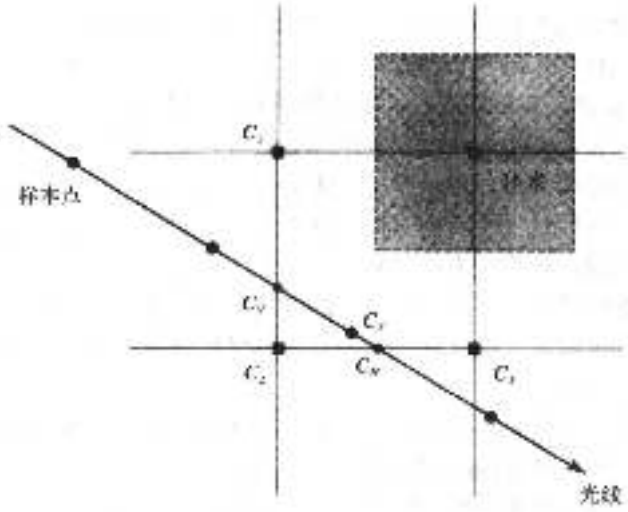


图13-14 C_5（光线上的一个样本点的值）由双线性插值进行计算。C_3是由C_1和C_2产生的估值。C_4是由C_2和C_3产生的估值。C_5是由C_3和C_4产生的估值

13.4.2 光线投射（变换后的数据）

光线投射方法的第二个变种涉及到将数据预先变换到希望的方向，然后，实际光线投射

的几何就是简单的（或被消除了），因为这时只是沿着数据的行或列进行合成。

为了转换数据，可以采用Wolberg（1990）提出的一种三路（都是切向的）分解。这时，一个观察变换就变成了一系列的纯的切变换，每个轴三个变换。所以，总的变换就是九个切变换的一个集合。只有切变换的过程的重要性在于在专用硬件中的执行。尤其是，它具有在一次切变换中每一个体素都只移动一个常量这样一种性质。 386 ~ 387

这两类光线投射方法的最显著的区别是在重新采样上。现在，在每一次切变换中间都必须进行重新采样，而且重新采样的过程在合成之前进行。在切变换的过程中重新采样涉及简单的线性插值（见图13-15）。

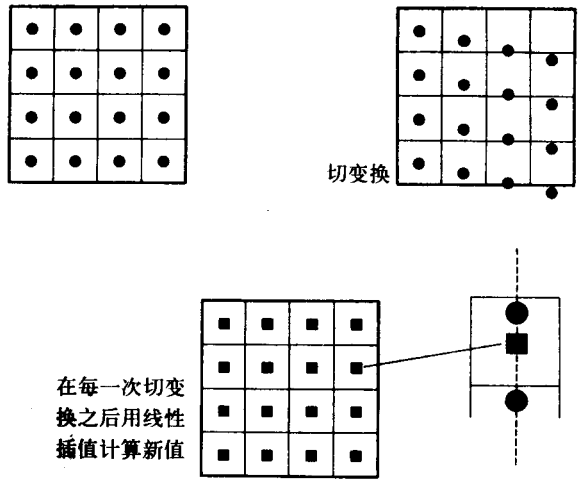


图13-15 在每一次切变换期间进行重新采样

在光线投射方法中，对于效率的一种重要的增强方法是忽略体数据中空的空间。一条投射光线一直在空的空间中前进，直到碰到一个物体时为止。它穿透物体，直到累积了足够的不透明度为止。对于高不透明度的情况，与在空的空间中遍历相比，这可能是一个很短的距离。空的空间对于最终的图像没有贡献，由于体素的数量很大，所以重要的是执行某些空间跳跃的过程。这个过程可以基于一个限定体，正像传统的光线跟踪中的加速算法那样，数据集上的遍历从限定体的表面开始。

13.4.3 体素投影方法

体绘制方法这类变种的可能性涉及到遍历数据集和把每一个体素投影到图像平面上，如图13-12所示。如果我们像图中所示在数据中移动一个平面，则采用帧缓冲器作为加速器，对所有的像素都同时更新，一直进行到所有的数据都被遍历，且像素都具有了其终值时为止。

我们既可以从前向后遍历数据，也可以从后向前进行。这两种方法的显著差别在于从后向前遍历仅仅需要累积颜色，而从前向后遍历既需要累积颜色也需要累积透明度（这正好等价于说对于从前向后遍历我们还需要有一个Z缓冲器）。

体素投影算法是重要的，因为这种算法更容易并行化。也就是说，在过程中对每一个体素上的每一个点，我们只需要关于其一个小领域的知识。这与将光线投射到未经变换的数据 388

上的情况正好相反，因为对于后一种情况我们一般在投射一束光线时需要整个数据集。

可能最著名的体素投影算法是Westover（1990）提出的算法，被称为泼溅（splatting）算法。这个奇怪的名字用于描述一个体素在图像平面中所起的作用。事实上，这个算法考虑的是一个体素的贡献如何在图像平面中分散或泼溅。考虑图13-16，在一个特定的体素中心，数据中的一个点被投影到一个像素上。为了确定像素的值应是什么，可以通过在采样的体素周围的三维区域上进行过滤来计算一个贡献值，另一种方法是可以取样本体素值，并将够个值分布到图像平面中的一些像素上。两种方法是等价的。如果我们把过滤函数看成是一个三维的高斯函数，则投影到图像平面上是一个圆环函数，因此，可以通过取体素值，并通过将值与过滤器权重相乘再累积这些值的方法将其泼溅到图像平面中，从而达到投影并过滤数据的目的。这个数据集被称为体素的脚印，对于平行投影所有脚印的权重都是相同的，并且可以被保存在一个查找表中。

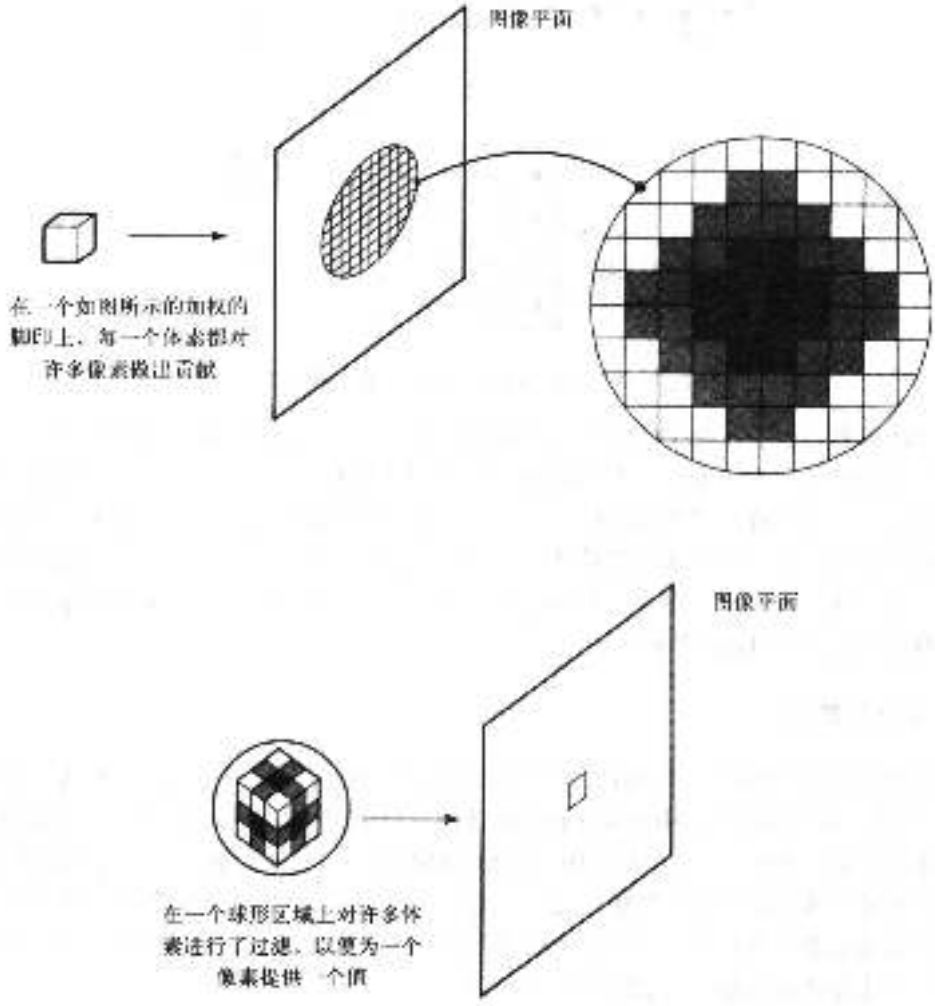


图13-16 体素投影中的过滤

13.5 体绘制过程中的透视投影

到目前为止，我们还没有讨论关于透视投影的问题。在医学成像中，可能并不需要进行透视投影。在医学应用中，体数据通常都被限制在几厘米的空间范围之内，我们不可能在这个距离之内感觉到明显的透视的线索。尽管通常它并不是对于观察者来说很重要的那种总结构的形状，而只是某些像骨折或肿块等细节及其与周围组织结构之间的关系。在医学中，某些特殊的应用确实需要有透视投影。一个例子是放射线治疗计划中的“光束的目视”结构。治疗的光束发散，所以需要有一个透视投影。

在体绘制程序中建立透视投影会出现一些很明显的困难。最严重的困难来自于从投影中心向外发散的光束（见图13-17）。如果光束的密度是这样产生的，即在采样时选取体数据中最靠近的平面，并且每个体素有一束光线，则在所示的例子中，该值很快会降为每两个体素一束光线，小的细节可以被忽略。另一个问题是重新采样期间的反走样。如果考虑沿着穿过像素以及投影中心的四个角上的四条光线遍历，在必须过滤的相邻的体素中心处，体的几何形状就不再是一个立方体的体素，而是一个经裁剪的金字塔形状。

执行透视投影最容易的方法之一是扩大体素投影或脚印算法。Westover（1990）对这一方法进行了详细的介绍。

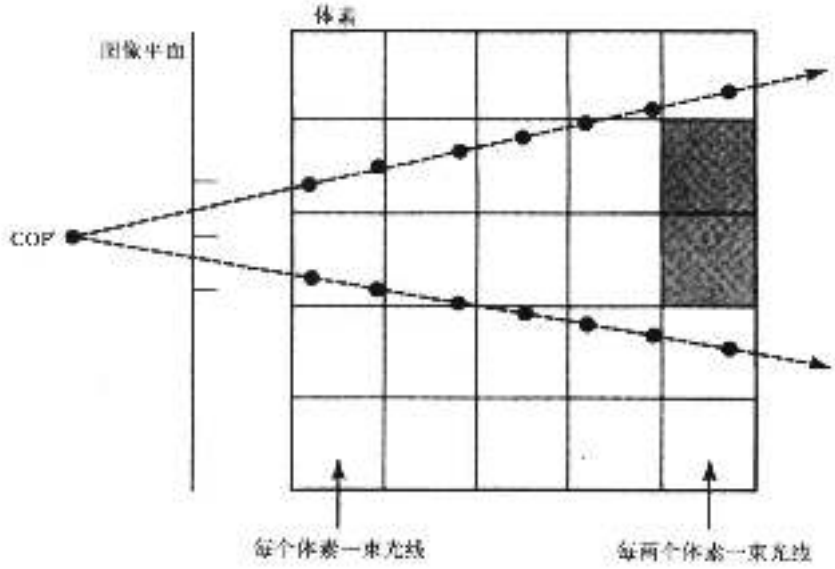


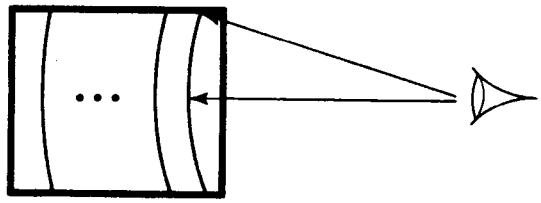
图13-17 光线密度和透视投影（Novins等（1990））一束光线/像素导致采样速率的降低

389
390

13.6 三维纹理和体绘制

由于可以将体数据集看成是一个三维的纹理，所以体绘制可以通过三维纹理映射功能来进行（见8.7节）。这一算法（Hacherli and Segal 1993）首先考虑的是对垂直于观察方向的平行多边形集合的计算。这就需要求出一组平行平面与数据体的限定平面之间的交。然后，将

多边形的顶点进行纹理映射，整个多边形集合按照从后向前的顺序进行合成。由于我们这时是对具有平行平面的数据进行采样，而不是沿着逐个的光束以等距离前进，所以对于透视投影，对从观察点发射出的光束来说，平面将产生不相等的采样间隔。在这种情况下，我们就需要用一个球形的片段来进行采样，而不是用平面进行采样，如图13-18所示。



391

图13-18 用一个以目光点为中心的球的片段采样的体数据

第14章　反走样理论及实践

14.1　走样和采样
14.2　锯齿形边
14.3　计算机图形学中的采样与真实采样之间的比较
14.4　采样和重建
14.5　一个简单的比较
14.6　预过滤方法
14.7　超采样或后过滤
14.8　非均匀采样——一些理论概念
14.9　图像的傅里叶变换

注意　这一章讨论经典的反走样方法，需要一些对傅里叶理论的了解。在14.9节有关于这一理论的简单介绍，这些介绍对于理解该方法已经足够了。

引言

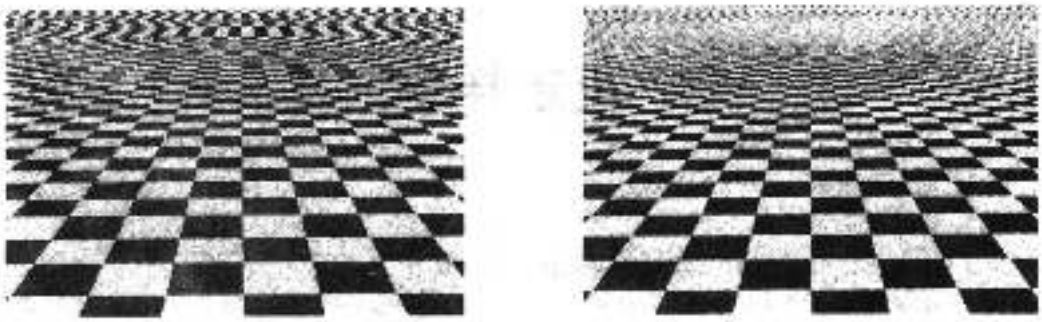
计算机图形学所生成图像的最终质量依赖于许多变化因素。在用于产生图像的特定的绘制方法中，由于运算而造成的其他影响因素和建模都会产生人工痕迹。例如，考虑多边形网格场景中的许多图像缺陷。建模中的人工痕迹通常称为走样，即在多边形网格物体轮廓边界上可见的分段线性现象。明暗处理算法也产生某些人工痕迹，比如马赫带和由于插值方法而产生的不适当的情况（有关这些缺陷的讨论见第18章）。在辐射度方法中，独立于观察的阶段 392
产生了困难的质量问题，这种问题用第11章中介绍的一般的反走样方法是不能处理的。

反走样是对于处理由于欠采样导致的差异的方法的总称。在这一章中我们将讨论这种差异问题。这样的方法用于传统的绘制方法中，比如第6章中讨论的多边形网格物体，第12章中介绍的光线跟踪以及第10章中介绍的蒙特卡罗技术。纹理映射中使用的反走样是在第8章中介绍的，其原因是尽管它是一种经典的方法，但是其特殊的实现方式（mip映射）是纹理映射所单独采用的。

14.1　走样和采样

我们首先考虑术语“走样”。理论上来讲，这一术语涉及的是一种特定的图像人工痕迹，即当纹理的变化周期趋向于一个像素大小时这些缺陷在纹理图中大部分是可见的。这种缺陷很容易展示，图14-1a是这类效果的经典例子，这是一个无限大的棋盘。在这个图的上方，四方形先是缩小，然后又很明显地增大，这就引起了很刺眼的可见的扰动。这是由于欠采样造成的。在计算机图形学中，采样的概念来自于我们是在为每一个像素计算单个的颜色或值。我们是在解空间的离散点上对解进行采样。这个解空间中，可以在图像平面上的任意位置或每一个位置上计算样本，因为计算机图形学的图像是通过抽象产生的。从这个意义上来讲，

这个解空间可能是连续的。

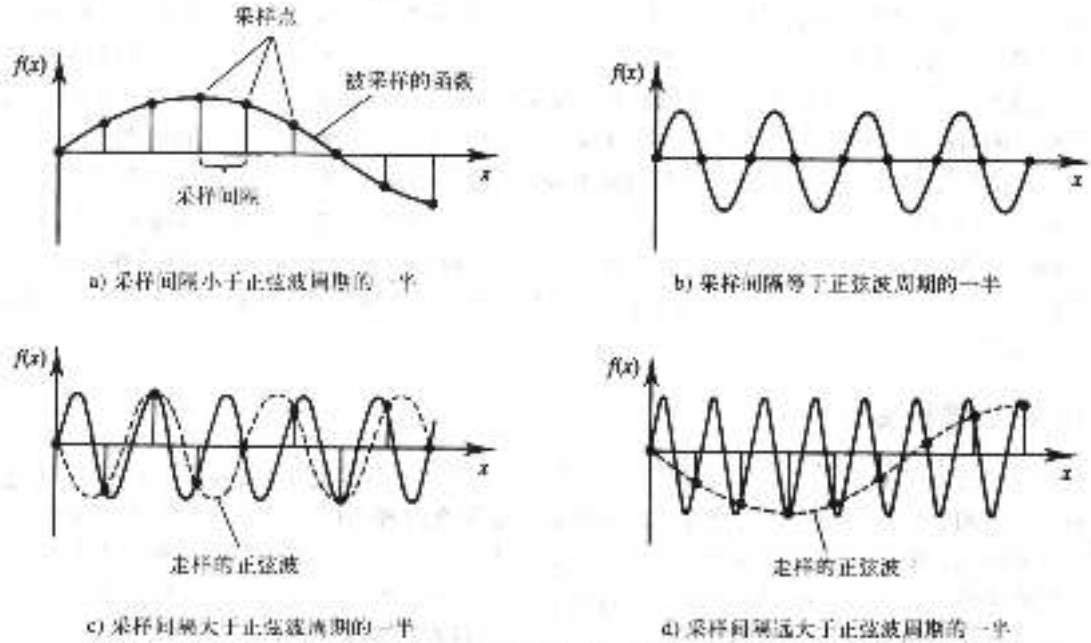


a) b)

图14-1 b中的模式是a的一个超采样，还是有些走样，但是其在较高空间频率出现

现在让我们来考虑一个简单的一维的例子，这个例子将把欠采样、走样和空间频率的概念关联起来。考虑用一个正弦波代表一个信息信号（尽管正弦波本身并不含任何信息，但这对于讨论没有影响）。图14-2是以不同速率（相对于正弦波的频率）采样一个正弦波。对此正弦波进行欠采样，并由这些样本重新构建连续的信号（图中的点划线），产生了对原始信号的"走样"。另一个正弦波的频率比被采样的正弦波要低。我们可以说出这种情况是因为采样模式的连贯性或规律性干扰了信息的规律性。为了防止产生走样的人工痕迹，我们必须相对于图像信息或信号以适当的频率进行采样。正常情况下，我们把图像平面的离散点上计算一个图像函数的过程看成与采样是等价的。

393



a) 采样间隔小于正弦波周期的一半 b) 采样间隔等于正弦波周期的一半

c) 采样间隔大于正弦波周期的一半 d) 采样间隔远大于正弦波周期的一半

图14-2 一个正弦波采样的空间域表示

由于不充分的计算或采样而产生的计算机图形学中的缺陷，容易用一种图像平面的采样

模型进行建模，这些缺陷是对连贯性模式的分解，如我们已经讨论过的那种情况。之所以丢失小碎片，是因为这些碎片位于两个采样点之间。

再来讨论棋盘的情况。模式的单位大小非常迅速地趋向于像素的尺寸，于是模式就被打碎了。高的空间频率也像低空间频率那样产生走样，并形成新的对连贯性模式的扰动。现在考虑图14-1b，这里，我们在观察平面上绘制了一个相同的图像，但较之前一幅图像的分辨率加倍。还是出现了走样的人工痕迹，但是其空间频率高了一些。从理论上来看，我们已经增加了采样的频率，但是走样的效果仍然保留着，只是发生在较高的空间频率上。这说明了两个重要的事实。计算机图形学图像中的空间频率是无限的，因为这个频率来自于一个数学的定义。不可能通过增加像素的分辨率来消除走样。人工痕迹只是在一种较高的空间频率下出 394
现。当然，这时的人工痕迹不太引人注意。

现在，可以通过对于一个含有不是纯正弦波的信息的$f(x)$在频率域中考虑这些情况来对图14-2中的例子进行泛化。我们有一个$f(x)$，其中x是任意的变量，比如，它可以代表一段扫描线上光强度的变化。$f(x)$的频谱将表现出某些“包迹线”（envelope）（见图14-3a），其极限是$f(x)$的最高频率的分量f_{max}。一个样本函数的频谱（见图14-3b）是一系列的线段。理论上这些线可以扩展到无限远，并且其间隔为f_s（采样频率）。空间域中的采样涉及到采样函数与$f(x)$的乘积。频率域中的等价过程是求卷积，采样函数的频谱与$f(x)$进行卷积运算，产生如图14-3c所示的频谱，即经采样的$f(x)$的频谱。然后，将这个采样后的函数与一个重建过滤器相乘，重新产生原函数。这个过程在时间域上的一个好例子是现代的电话网络。在最简单的形式下，这个过程包含对音频的波形进行采样编码，在通信信道上传输每一个样本的数字版，然后用重建过滤器对样本解码并重建原始信号。

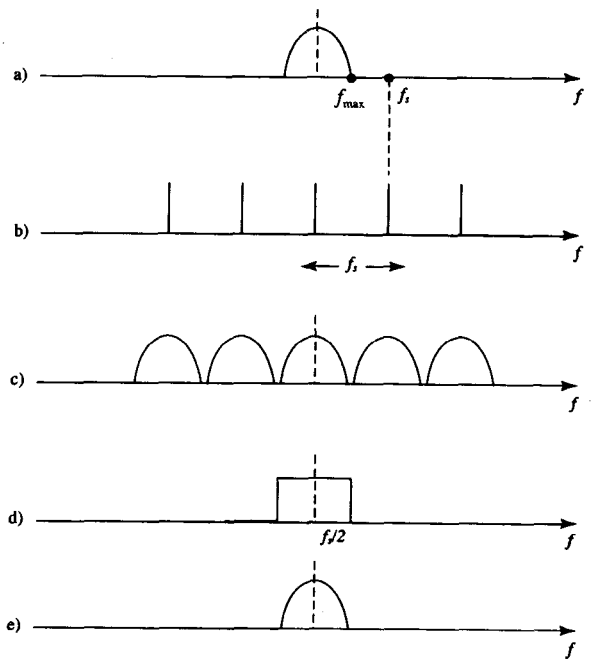


图14-3 当$f_s > 2f_{max}$时采样过程的频率域表示

a)$f(x)$的频谱 b)采样函数的频谱（a和b的卷积）
c)采样后函数的频谱 d)理想的重建过滤器 e)$f(x)$的重建

注意，频率域为乘法的重建过程在空间域是卷积。总之，空间域中的过程是原函数与采样函数之积，再对函数的采样与重建过滤器进行卷积。

现在，上面例子中的条件

$$f_s > 2f_{\max}$$

为真。在第二个例子中（见图14-4），我们看到的是相同的两个乘积和卷积过程，只是这时:

$$f_s < 2f_{\max}$$

另外，$f_s/2$称为Nyquist限制。这个位置上代表$f(x)$中信息的包迹线重迭了。就好像在由Nyquist限制所定义的线上光谱被“折叠”了（见图14-4e）。这个折叠是一个信息的消失过程，高频部分（图像中的细节）被丢失了，表现出低频区域上的干扰（走样）。图14-1中对这个效果有精确的展示，图中在高频区域出现了低空间频率的结构。

将采样定理扩展到二维频率或空间频率上。在连续产生域中，图形学图像的二维频谱理论上是无限的。在计算机图形学中，采样和重建是对一个像素的中心处的值的计算过程，并且将这个值分配给该像素的整个空间范围。

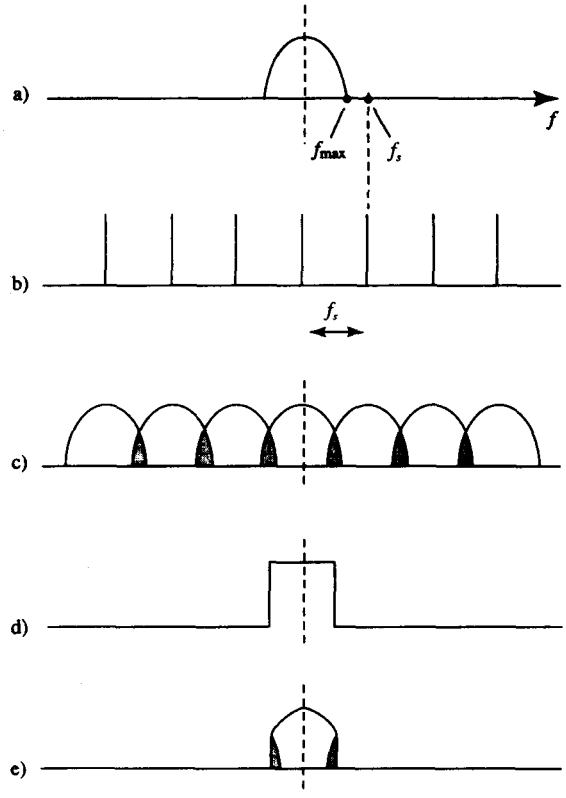


图14-4 当$f_s < 2f_{max}$时采样过程的频率域表示

a) $f(x)$的频谱 b) 采样函数的频谱 c) 采样后函数的频谱
d) 理想的重建过滤器 e) 被扭曲的$f(x)$

在计算机图形学中，走样的人工痕迹可以通过增加采样网格的频率来降低（也就是增加像素数组的空间分辨率）。这个方法有两个缺点：明显的缺点是增加显示的空间分辨率存在经济

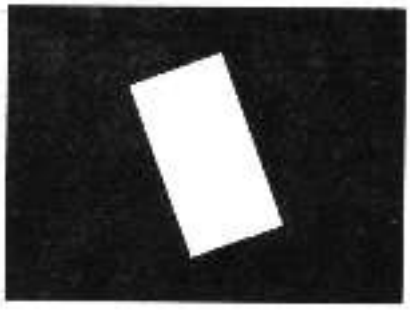
上和技术上的限制（更不必说图像产生过程中对于计算成本的限制），由于计算机图形学图像的频谱可以无限扩展，而增加采样的频率也不一定能够解决问题。例如，当在透视投影中对连续的纹理采用增加分辨率的方法时，我们仅仅是把这个效果在空间频谱中向上进行了移动。

14.2 锯齿形边

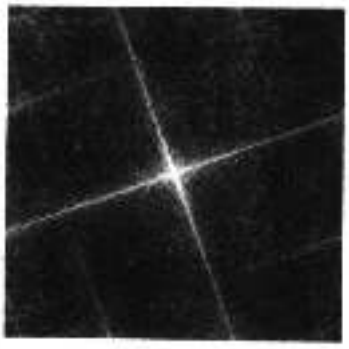
在计算机图形学中，最熟悉的缺陷就是所谓的锯齿。当在图像中出现高对比度的边时，（通常使用的）正方形像素的有限尺寸会产生这种锯齿。在动画图像中这些锯齿形边尤其会引起麻烦，这些边的移动使其看起来像小型的动画物体，并且这些物体忽隐忽现。这种缺陷很容易克服，因为它们从本质上来看并不是由于算法造成的，它们只是图像平面分辨率的产物。

锯齿形的边任何人都可以识别出来，这种现象在所有的计算机图形学的教科书中都有介绍，但是，它们并不是在经典的空间频率情况下的走样缺陷，其中高的空间频率是随着一个由较低频率而产生的裂纹出现的，它们是由显示设备最终的限制作用产生的缺陷。例如，我们肯定可以通过以一种比像素的分辨率要高的分辨率来计算一个图像而消除锯齿的影响。换句话说，增加采样的频率既处理了走样问题，也处理了锯齿形边的问题。当有锯齿时，边的信息被“强迫”为像素的水平或垂直的边。考虑图14-5，图中是一个完整的矩形和一个像素化的版本。对于完够矩形的傅里叶变换将边的信息沿着相应于图像的边的方向映射到了高能分量上。像素图的傅里叶版也含有这个信息，并附加沿着轴的相应于（逻辑）假或像素边的高能分量。由于将高的空间频率走样作为低的走样来处理，所以不会出现锯齿形边。

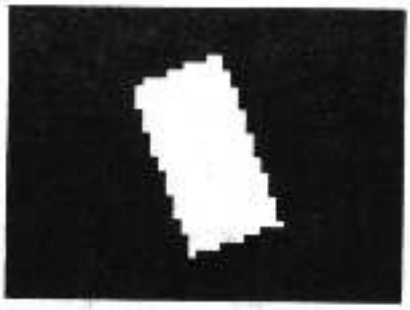
395
|
397



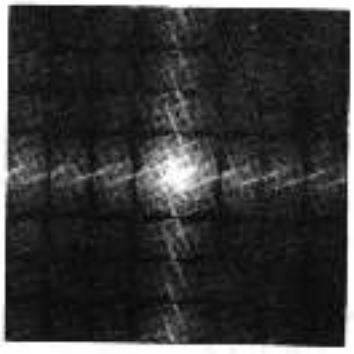
a）一个完整的线的模拟



b）a的傅里叶变换



c）一个锯齿形线段的模拟



d）c的傅里叶变换

图14-5 锯齿形的作用是把高能分量向傅里叶域中的水平或垂直轴旋转

14.3 计算机图形学中的采样与真实采样之间的比较

现在让我们返回到图像平面中的采样概念进行更详细的分析。在图像合成时，我们所做的是对于每一个像素执行一些操作（这些操作有时是非常复杂的），这些操作最终会为该像素计算一个常量值，并在像素的整个范围内“分布”该值。

我们假设从原理上这种处理与用一个观察平面上的连续图像以一个样本点的离散的二维数组进行采样之间没有区别（每个像素一个样本点）。之所以说这个假设是有效的，是因为我们对图像的处理可以通过不断增加样本的分辨率并在图像平面中以越来越多的点为图像计算其值来完成。但是，重要的是需要记住，在计算机图形学中，我们没有办法访问一个连续的图像，这就限制了我们所采用的反走样的方法。

事实上，“采样”和“重建”（“重建”是另一个从数字信号处理领域借用的术语）这两个
398 术语都是不加选择地采用的。我们感觉在计算机图形学中它们有一些混淆，现在我们将强调如下两种处理系统的差别，即图像处理系统的完全适当的使用以及在计算机图形学中的有些人工痕迹的使用。

考虑图14-6，这是一个图像处理器的流程图和一个计算机图形学系统。在图像处理器上，一个采样器将一个二维的连续图像转换成一个样本数组。然后，对数字图像执行了某些操作，接着一个重建过滤器把经处理后的样本再转换成模拟信号。

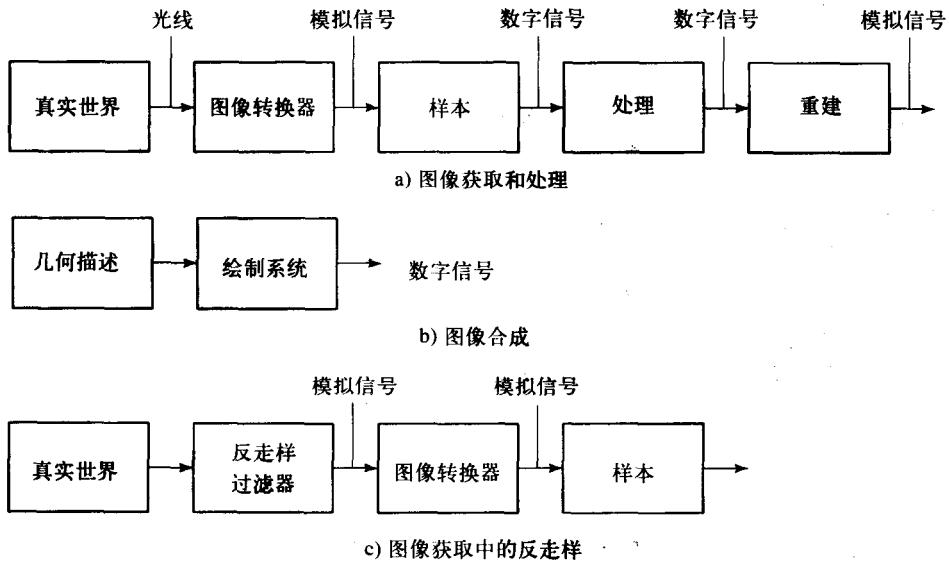


图14-6 图像处理和图像合成中的采样、重建和反走样

图像合成中的情况并非如此。这时候的采样并不具有相同的意义，当为一个像素赋一个值时所涉及到的操作依赖于所采用的绘制算法。我们只能计算这些点上的一个图像函数的值。

图像合成中的重建并不意味着由一个数字的图像产生一个连续的图像，但是它可能意味着比如说从一个以较高分辨率存储的图像（不可显示的）产生一个低分辨率的（像素的）图像。我们并不是重建一个图像，因为一个连续的图像一开始并不存在。对这些差别的一种正确的评价将可以防止混淆（实际上，我们确实在计算机图形显示器上重建了一个用于显示的连续的图像。但是，这一工作是由固定的电子设备对由图形程序以帧存储的形式产生的图像

进行操作来完成的。一个反走样的综合性的方法可能需要把电子信号的转换特性一并考虑，但是，本书中将不对这一部分进行考虑）。

返回到关于走样人工痕迹的问题上来，傅里叶理论告诉我们，走样的出现是因为我们对一个连续的图像进行采样（或者是计算机图形学中的等价操作），而我们在足够高的分辨率下获取高的空间频率或者图像中的细节时并不这样做。采样理论告诉我们，如果希望对一个图像函数进行采样而不损失信息的话，则（二维）采样频率必须至少是图像中最高频率分量的2倍。 399

所以，在实际的计算机图形学中这意味着什么呢？它只意味着：如果我们考虑在一个具有正方形像素的网格的观察平面上对一个连续的图像进行采样的话，则在一条扫描线上可以出现的最高频率为：

$$f = 1/2d$$

其中d是像素中心点之间的距离。

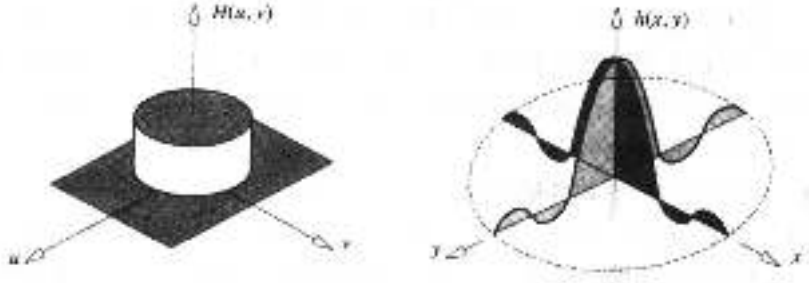
把这个概念搞清之后，就容易理解为什么反走样在计算机图形学中是如此困难了。问题出自两个令人吃惊的事实。在计算机图形学中对于高频的值是没有限制的，我们已经用无限棋盘的例子对这点进行了讨论。对于这些空间频率也没有直接的方法来加以限制（其技术术语是“带宽限制”）。

通过将图像合成与通过一个类似电视摄像机（见图14-6c）这样的设备捕捉到的图像进行比较就很容易看出这一点。在对一个连续的图像进行采样之前，可以使其通过一个带宽限制过滤器（或者反走样过滤器）。显示不出来的较高的频率在其被采样之前从图像中删掉了。我们将这种处理称为预过滤。在这种系统中，走样问题仅仅是处理成不让其出现。

在图像合成中，场景数据库作为一种数学描述或者作为一种由边连接起来的点的集合。关于采样的概念无法避免地与绘制紧密结合在一起。通过估算场景在离散点上的投影进行采样。我们不可能对图像进行带宽限制，因为根本就没有图像存在，只是在选择点上定义了它的存在。

14.4 采样和重建

在图14-3中，我们看到假如遵循了采样理论，则可以通过用一个盒形的重建过滤器获得由样本得到的重建信息。但是，这是一种傅里叶域的表示，在计算机图形学领域中，所有运算都必须在空间域中执行。因此，重建过程就是在空间或图像域的卷积。在计算机图形学中，这（通常）意味着以某种方式过滤一幅绘制的图像。如果所绘制的图像是连续的，则重建过滤器应由一个正弦函数$h(x, y)$组成，这个函数是傅里叶域的变换，等价于一个圆的步进函数（见图14-7）。



a）一个理想的低通（乘法的）过滤器$H(u, v)$　　b）等价的（卷积）过滤器$h(x, y)$

图14-7 在傅里叶和空间域中的理想的过滤器 400

但是，与此相关的也会存在一些困难。过滤器不可能进行无限的延伸，它必须在某点上被截断，而截断的方式是过滤器的设计中重要的考虑因素。

14.5 一个简单的比较

现在，我们简单地以概念比较的形式来讨论计算机图形学中的反走样选项。图14-8所示为四种主要的方法。

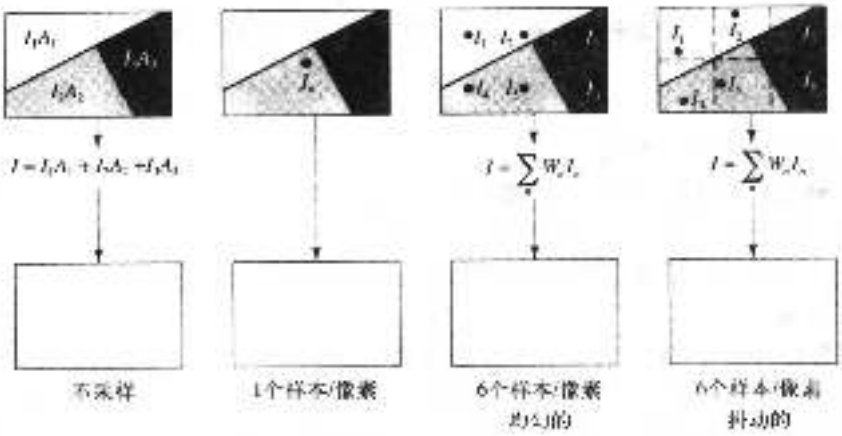


图14-8 为一个像素计算一个值的四种方法的比较

1. 预过滤——每个像素点“无限”采样

这时是计算所投影的物体结构片段出现在像素上时的精确的贡献，将这个贡献值作为像素的颜色，这一方法的实际效果仅仅是将“无限”的分辨率降低为像素显示的有限分辨率。如果一个像素的物理延伸范围小则这种方法是一种高质量的方法，但也是一种完全不切实际的方法。然而，我们注意到，尽管这种方法假设有精确的几何学形状，但我们还是假设光强度在任何片段上都是常数，实际上，我们用这种方法所做的就是预过滤，也就是说用一个盒子形的过滤器在采样之前进行过滤。

这是如图14-6c所示的采用反走样过滤器的方法，它实际上是消除了那些证明是子像素细节的高频率，但是，因为计算是连续的，所以它是在采样之前完成这一工作的。

2. 不过滤——每个像素一个样本

在第二种情况下，我们仅仅考虑每个像素有一个采样，这就等价于第一种情况，当且仅当投影是使一个像素仅含有一个几何结构，在投影中所有的几何结构的边界都与像素的边界共线，这是在实际过程中不可能实现的限制条件。这种“什么也不做”的方法在实时动画中尤其普遍，这种方法也被用作在离线产品上的预览，这时只有当用一个创建程序对预览满意时才产生一个最终的反走样图像。

401

3. 后过滤——每个像素n个均匀的样本

这是最普遍采用的反走样方法，它涉及以n倍于最终的屏幕图像分辨率绘制一个虚拟的图像。这是对于连续图像概念的一种近似。接着，对虚拟的图像采样，并通过一个卷积运算对其进行重建产生最终图像，将这两个运算结合到一个运算中。这种方法的效率依赖于超采样的数量以及一个像素中图像的结构与采样的网格点之间的关系。注意，尽管我们可以将这种

方法视为是对第一种方法的近似，但这时对于相同的片段上的样本可能有不同的强度值。

4. 后过滤——随机采样

这个方法可以看成是对前一种方法的简单的改变——代之以均匀地在一个像素中采样，这时我们会根据某些策略抖动地采样。这种方法已经在第10章中（见图10-9）作为蒙特卡罗方法的一个组成部分进行了讨论。在这一章中，我们将主要讨论为什么这种方法被作为一种“纯的”反走样方法。

14.6 预过滤方法

这一技术的创始人是Catmull（1978）。尽管Catmull原来的算法的高代价使人望而却步，但是它却已经产生出了一些更有实用价值的后继者。

这个算法本质上是在连续的图像产生域中执行子像素的几何运算，并为每一个像素返回一个光强度值，这个值是由光强度总和值中以可见的子像素片段的面积作为权重计算得出的。这就等价于把图像与一个盒状过滤器进行卷积，并用卷积在一个点上的积分值作为最终的像素值。（注意，过滤器的宽度小于理想值，用来自相邻区域的信息得到的一个较宽的过滤器将得出一个较低的截断频率。）考察这个方法的另一种方式是说它是一个区域采样方法。

我们可以提出这样的问题：在实际的计算机图形学中执行“子像素几何”意味着什么？为了回答这个问题，我们必然要采用一种可实际使用的近似方法（重申早先的观点，即我们没有处理连续图像的方法。在计算机图形学中，我们只能在某些点上定义一个图像）。这意味着采样技术与超采样之间的区别有些是人为的。其实，A缓冲器技术（已经简短地进行了介绍）也可以看成是超采样技术，而通常将它分类为区域采样方法。

Catmull的方法被结合到一个扫描线绘制程序中。它是通过把连续的图像产生域分割成正方形的像素范围来进行处理的。根据正方形的像素边界对多边形进行裁剪来计算每一个正方形的光强度。如果在一个正方形中有一些多边形片段重叠，则将它们在Z缓冲器中进行排序，并互相之间进行裁剪，以产生可见的片段。将一个多边形的明暗处理与其可见片段的面积相 402
乘并求和来计算其最终的光强度。

这个方法中内在的严重的计算负荷的根源是很明显的。原始的方法代价如此之高，以至于它只用于涉及极少量的多边形的二维动画应用程序中。在这种情况下，大多数像素都完全被一个多边形所覆盖，并不会进入递归的多边形片段与多边形片段之间的裁剪过程。

最近的发展涉及到将子像素片段与位掩码的近似（Carpenter 1984; Fiume等 1983）。Carpenter（1984）以一个Z缓冲器来使用这种方法，产生了一种称为A缓冲器的技术（反走样的、区域平均的累加缓冲器）。这个方法的显著进步是避免了浮点的几何运算。通过在位模式或代表多边形片段的掩码之间按位的逻辑运算符实现覆盖和区域加权。这是一种高效的区域采样技术，其对每一个像素正方形的处理将依赖于可见片段的个数。

区域采样的另一个高效的方法是Abram等（1985）提出的。该方法预先计算对卷积积分的贡献，并将这些贡献存储在以多边形片段为索引的查找表中。该方法所基于的事实是一个多边形覆盖一个像素的方式可以由有限的情形来近似。这个算法被植入到了一个扫描线绘制程序中。卷积并没有局限在一个像素的范围，而是更正确地扩展到比如3 × 3的面积上。一个像素作为一个累加器，当所有可以影响到最终值的片段都被计入之后，累加器的最终值是正确的。

考虑一个3 × 3的像素区域和一个3 × 3的过滤器内核（见图14-9）。当过滤器位于9个正方形中的每一个的中心时，在中心像素上一个可见的片段将对卷积作出贡献。这样的片段构成的9个贡献可以预先计算，并存储在查询表中。过程中的两个主要阶段为:

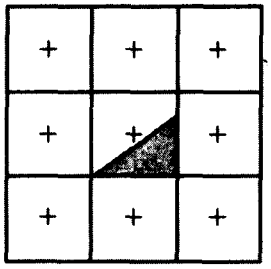


图14-9 在中心像素上的一个片段引起在9个正方形中的每一个上对过滤的贡献

1) 求出可见的片段并标识或对其形状分类。

2) 对预计算的查询表进行索引，这个表中给出每一种形状的9个贡献。对片段的光强度与预计算的贡献
403 加权值的乘积给出所希望的结果。

Abram假设片段的形状有以下七类:

1) 在像素中没有片段。

2) 片段完全覆盖了像素。

3) 片段是不规则四边形，沿着对边把像素进行了分割。

4) 片段是三角形，将像素沿着邻边进行分割。

5) 对4) 的补充（五边形的片段）。

6) 片段是一种临时的形状，可以由两个或多个前述的形状之差来描述。

7) 不能很容易地用这些简单形状类型进行定义的片段。

14.7 超采样或后过滤

在反走样中超采样是最常见的采样方法，是多边形网格绘制通常采用的方法。它涉及以一种高于像素分辨率的空间分辨率计算一个虚拟的图像，并把高分辨率的图像“平均”到一个较低的（像素）分辨率。从广义来看，考虑前面对术语“采样”的利用的保留条款，我们现在是增加采样的频率。此方法的优点是平凡的执行，但却是以高成本以及对Z缓冲器存储需求的增加为代价。以傅里叶理论的观点来看，我们可以:

1) 以某一分辨率（比像素的分辨率高）产生一个$I(x, y)$样本的集合。

2) 对这个图像进行低通滤波，我们将这一操作看成是对一个连续图像的近似。

3) 重新以像素的分辨率对图像采样。

第2步和第3步（经常被混淆为重建步骤）是同步执行的，即将一个过滤器与虚拟的图像进行卷积，并将其作为像素宽度的卷积间隔的步长。也就是说，对于3 × 3的虚拟图像，过滤器以三个超像素的步长位于虚拟图像的（超）像素上。图14-10表示了此方法的工作过程和两个以加权列表的形式给出的过滤器的例子（请注意，这些权重不是归一化的，过滤器的权重之和必须为1）。对于一个（奇数的）比例因子S和k维的过滤器h:

$$I'(i,j) = \sum_{p=Si-k}^{Si+k} \sum_{q=Sj-k}^{Sj+k} I(p,q)h(Si-p, Sj-q)$$

这一方法适用于大多数计算机图形学的图像，并且易于被结合到Z缓冲器算法中。但它不适用于图像的光谱能量不随频率的增加而增加的情况（正如我们已经提到的，一般来说，超
404 采样从理论上来讲并不是正确的反走样方法）。

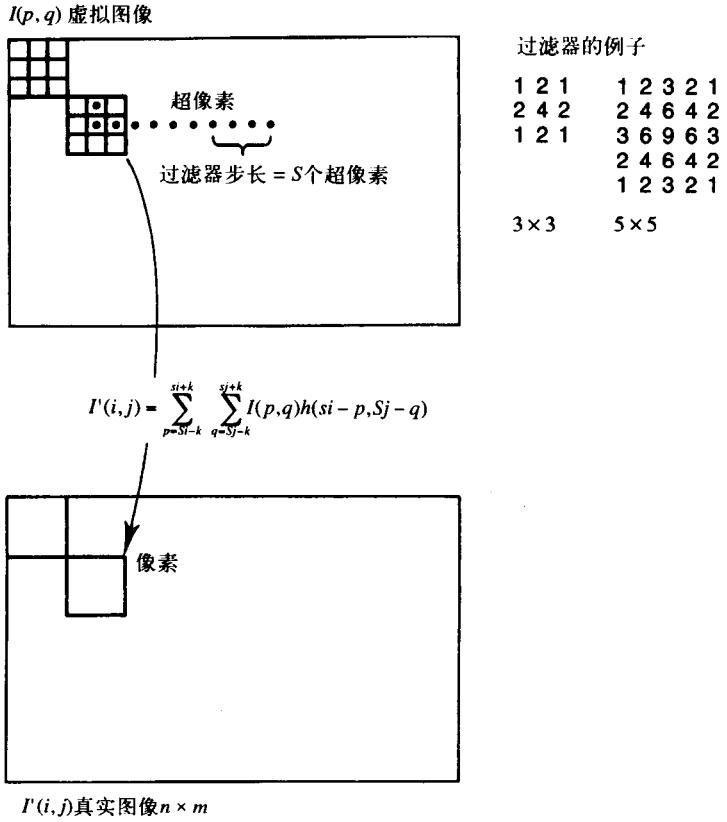


图14-10 通过卷积“降阶”一个虚拟图像

超采样方法在n的取值和所用过滤器的形状上稍微不同。例如，对于一个中等分辨率的512 × 512的图像，通常认为适合以2048 × 2048（$n = 4$）的分辨率进行超采样。通过进行平均，可以将高分辨率的图像降为最终的512 × 512的形式，这等价于用一个盒式的过滤器进行卷积运算。用有形状的过滤器可以获得较好的结果，有形状的过滤器是指其值随内核发生变化。关于过滤器的最佳形状相对于其所要处理的信息之间的对应关系有相当多的知识可以参考（例如，参见Oppenheim and Shafer(1975)）。这一类工作大部分都是在数字信号处理领域，并且其执行是以一个单变量函数$f(t)$进行的。计算机图形学的问题有特殊性，常规的信号处理技术并没有涉及这类问题。例如，在纹理映射中需要有空间变形的过滤器。这时过滤器内核的权重及其形状都必须改变。

返回到超采样和不变的过滤器，Crow（1981）采用了一种Bartlett窗口，表14-1列出了其中的三个窗口。

数字卷积是很容易理解和实施的，但是从计算上来说代价高昂。在一个超采样的中心安放一个窗口，将每一个超采样与相应过滤器的权重相乘得到乘积的加权和。可以对权重进行 405
调节以实现不同的过滤器内核。通过在n个超采样上移动窗口并计算下一个乘积的加权和来进行数字卷积。采用一个3 × 3的窗口意味着在最终的像素计算中要涉及到9个超采样。另一方面，用一个7 × 7的窗口就意味着要进行49次整数乘法的计算。计算开销的含义是明显的。例如，把一个2048 × 2048超采样的图像降为512 × 512的图像，若用一个7 × 7的过滤器内核则需进行

512 × 512 × 49次乘法和加法。

表14-1 用于后过滤一个超采样的图像的Bartlett窗口

3×3	5×5	7×7
1 2 1	1 2 3 2 1	1 2 3 4 3 2 1
2 4 2	2 4 6 4 2	2 4 6 8 6 4 2
1 2 1	3 6 9 6 3	3 6 9 12 9 6 3
	2 4 6 4 2	4 8 12 16 12 8 4
	1 2 3 2 1	3 6 9 12 9 6 3
		2 4 6 8 6 4 2
		1 2 3 4 3 2 1

过滤的一个不可避免的副作用就是模糊。事实上，我们可以说，是用走样的人工痕迹与模糊进行交换。出现这种情况是因为信息是由相邻像素集成的。这意味着对于过滤器的空间范围的选择是一种折中。一个宽的过滤器具有低的截断频率，对于降低走样的人工痕迹有好处。但是，它会使图像的模糊程度比一个窄的具有高的截断频率的过滤器要高。

最后，需要指出此技术的缺点。超采样对于处理非常小的物体不是一种合适的方法。尽管它是一种“全局”的方法，但是计算并不是上下文相关的。一个表现为几个大面积多边形的场景的计算开销与具有大量小面积多边形的场景的计算开销相当。当方法必须采用一个Z缓冲器时图像对存储的需求是大的。在应用过滤过程之前必须建立并存储图像的超采样版。这就使Z缓冲器的存储需求增加了n^2，使其成为一种虚拟内存技术。

14.8 非均匀采样——一些理论概念

非均匀采样在计算机图形学中引起人们很大的兴趣是因为它解决了传统的反走样技术遇到的高成本问题。它摒弃了均匀采样的思想，允许我们处理上下文敏感的反走样问题。或者，把计算的资源集中到图像中那些需要引起注意的部分上。恒定的完成这一功能的方法意味着
406 我们所研究的算法在“绘制”部分和反走样部分之间是不分离的。我们不能像上面的超采样方法那样在不采用反走样方法的情况下进行绘制。

考虑非均匀采样的另一个好处是它使得算法在可能将走样转换成噪声的地方也可以运行。也就是说，可以对于一个已知像素分辨率的图像以这样的方式设计算法，即使算法在那些用传统的算法可能产生走样的地方产生噪声。进行这一工作的方法称为随机采样方法，这种方法通过使样本之间的均匀间隔变得不规则来实现非均匀采样。

理想情况下，我们希望产生一个图像时，在“繁忙”的区域倾注最大的精力，而在光照变化缓慢的地方花较小的精力。在图像合成中问题的核心是：在产生出图像之前，我们如何知道对哪一个区域给予更多的注意呢？关于这一问题的考虑很自然地使我们采用最常见的策略，即产生一个低分辨率的图像，对其进行检查，并在低分辨率图像中那些看起来需要进一步分析的区域产生一个较高分辨率的图像。我们可以递归地重复这一过程，直到达到了预先定义的某种限制条件为止。这种方法称为自适应细化方法（这一技术的一个例子如图18-13所示）。

一个简单的但绝不完整的非均匀采样的分类可能是两个非均匀主类，即细分和随机采样。有很多种细分法，即影响随机采样的不同方法，以及把这两种方法结合成一个采样策略的结合方式。例如，可以按不同的尺度（即每单位区域上样本的个数）产生一个随机的采样模式，

以便使其可以被结合到一个自适应的细化策略中。

这些方法示于图14-11中。这两种方法都是在已经对图像平面进行了初始采样之后应用的。在计算机图形学中最通常的情况是均匀采样，通常是在像素层次上进行，但不是必需的。然后，这一方法就成了非均匀的超采样，因为其非均匀策略是在子像素层次上执行的。



图14-11 两种主要的非均匀采样技术

非均匀细分是一种通用的策略。计算机科学的很多算法中都采用这一策略。自然，它也是适合图像合成的自适应细化方法。这个方法先把图像平面划分成初始（即正方形的）采样盒的网格，然后，递归地将这些网格划分成正方形，直到达到一定的分辨率限制。这样的方法存在着另一个细微的问题，即从运用这类策略的算法中产生的输出将这些样本转换成一个非均匀的 407
样本集合。在显示之前，这些样本必须转换成一个均匀的（像素）样本集合。换句话说，我们必须对非均匀的样本进行重建，然后再以一种均匀的速率重新采样。对于由非均匀的样本进行重建没有现成的理论可循，并且存在着很多专门的技术。图14-12为一个简单的方法。

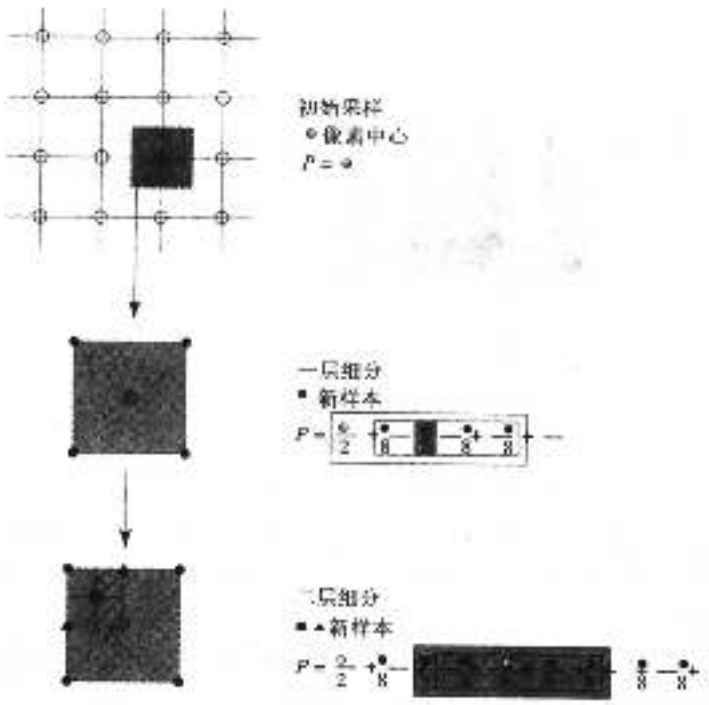


图14-12 对于非均匀细分的简单重建

乍看起来随机采样似乎是一种奇怪的思想，但是对其高效性的直观的解释却是直接的。一个图像中走样的出现是作为以采样模式的规则性对抗图像中的规则性或连贯性的一个直接的后果。如果我们使样本不规则，则图像中的较高频率的连贯性作为噪声而不是走样出现。这种规则采样的扰动以及合乎逻辑的走样与噪声之间的折中就是随机采样。

对这种折中最方便的演示就是回到我们所列举的正弦的例子。图14-13所示为一个正弦波，并且用规则的采样模式进行了采样。现在，我们通过将每一个样本关于规则采样的瞬间以某个随机量进行“抖动”来进行随机采样，考虑对于一个其频率低于Nyquist限制的正弦波进行
408 这种抖动的效果（见图14-13a）。在这里，我们的过程将对正弦波不精确地采样，并引入较大的扰动或噪声，扰动的大小依采样的瞬间的抖动程度而定。对于一个其频率远远高于Nyquist限制的正弦波（见图14-13b），采样的抖动范围包含了许多个周期，连续地对这些波形包进行采样的结果将产生一组随机数。于是，可能由于规则的采样间隔而产生的走样的正弦波就换成了噪声。

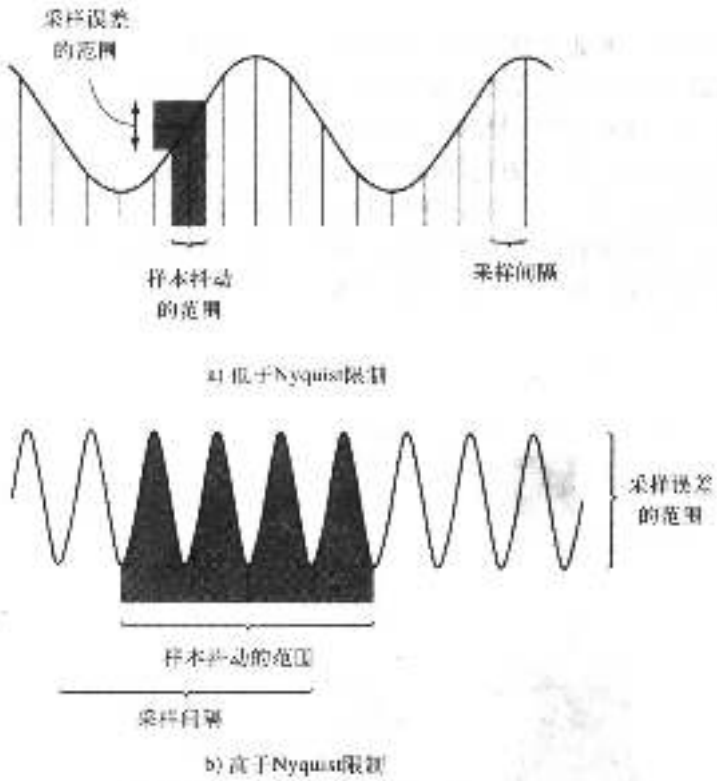


图14-13 对正弦波采样（Cook）

在一个二维的区域上很容易进行抖动，比如对一个像素，从一个均匀的网格开始启动，并分别在x和y方向上应用两个抖动分量，该方法既经济又方便，正因为如此，它可能是计算机图形学中最常见的策略。

随机采样有一个有趣的背景。在1982年，Yellot（Yellot 1982）指出，人类的眼睛中含有一个非均匀分布的感光器的阵列，他认为这就是人类的眼睛本身不产生走样人工痕迹的原因。视网膜中央凹处的感觉单元紧密地排列在一起，眼球的晶状体作为一种反走样过滤器，但是，

视网膜中央凹区域之外的区域感觉单元的空间密度就小得多了，正因为这个原因，这些感觉单元是非均匀分布的。

通过考虑一个以这种方法采样的正弦波，并且关于Nyquist限制变化频率，可以方便地在频率域中表现出这些影响因素来（见图14-14）。随着采样频率相对于正弦波的降低，正弦波峰的幅度减小，噪声的幅度增加。最后，正弦波峰消失。这个示意图的重点是没有出现噪声 409
的波峰。由正弦波所表示的信息最终消失了，但是，没有了走样而代之以噪声。扰动的范围可以在最小为半个周期的范围上变化（正弦波的频率受Nyquist限制）。一般来讲，变化范围可以达到几个完整的周期。如果这个范围包含了几个完整的周期，则对于白噪声的抖动，对正弦波的每一部分的采样的概率趋向于相等，样本中的能量就作为白噪声出现。由于白噪声抖动和高斯抖动而减少的数学处理方法由Balakrishnan（1962）给出。

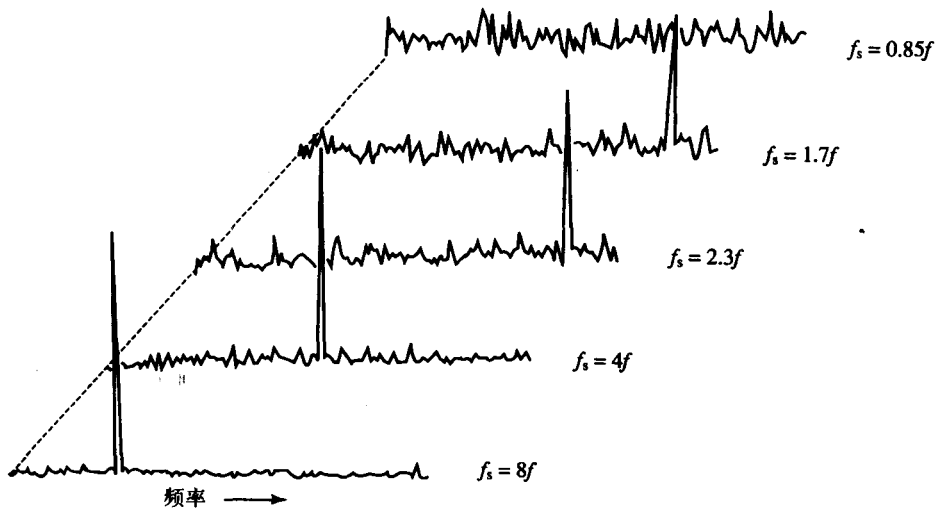


图14-14 相对于一个扰动的采样频率（f_s）改变正弦波的频率（f）

这种方法的问题之一是它只能与那些为每个样本进行独立计算的方法相结合。光线跟踪肯定符合这种情况，在这种方法中光线是朝着连续的物体空间域扩散的，而事实上也是在这个空间上进行的采样，样本可以方便地被抖动。在“标准的”图像合成方法中，比如说采用带Z缓冲器的插值明暗处理或者扫描线算法，若引入抖动会造成更多的困难。这种方法基于在屏幕空间的均匀增量的方法，可能会需要对其进行相当多的改进才能具有二维空间采样扰动的效果。尽管这种方法等价于在一个连续的域中产生图像，再进行二维的采样，但是在实际过程中采样和产生图像两个阶段要想不结合在一起是不容易的。

然而，一个称为REYES（Cook等 1987）的主要的绘制系统确实将基于Z缓冲器的方法与随机采样结合了起来。它是通过把初始的基元（比如双三次参数曲面片）划分成（平的）微多边形（划分到屏幕空间中的约半个像素大小的尺寸）来达到这一目的的。所有的明暗处理和可见性的计算都是在微多边形上进行的。在可见性计算之前进行明暗处理，而明暗处理在 410
微多边形上是一个常数。然后，再从屏幕空间随机地对微多边形进行采样，每一个样本点的Z值由插值计算，可见的样本碰撞经过滤的样本产生像素的光强度（见图14-15）。于是，明暗处理就在微多边形层次上进行，而可见性的计算是在随机采样层次上进行的。

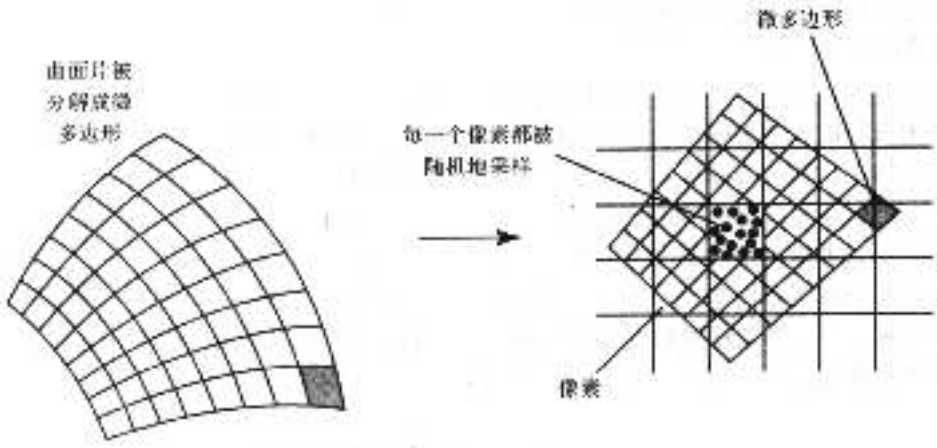


图14-15 图形基元被划分成微多边形，通过在屏幕空间随机地采样对这些微多边形进行明暗处理和可见性计算（Cook等（1987））

这个方法由于把物体分解成了微多边形而废弃了“经典”绘制方法所具有的连贯性。它更适合于含有双三次参数曲面片的物体，因为这些物体容易划分。

14.9 图像的傅里叶变换

傅里叶理论并不用于计算机图形学的任意场合，它只用于一些像用傅里叶合成产生地势的高度场这样的特殊应用中。然而，对于它的一种直观的理解对于理解由于欠采样而造成的图像缺陷的作用以及修正方法是非常重要的。

傅里叶变换是现代科学和工程的基本工具之一，它被应用在模拟电子和数字电子领域中，在这些领域中信息是以一个时间的连续函数的形式表示的（通常情况下如此）。还可以在一些与计算机成像相关的工作中找到其应用，在这些应用中图像$I(x, y)$是以两个空间变量的密度函数的形式表示的。

计算一个图像的傅里叶变换意味着图像是以空间频率的一个加权的集合（或者按正弦曲线加权的波浪形的表面）来表示的，这样的处理至少对于一些我们所关心的操作是尤其有利的。各个空间频率称为基本函数。

任一采用傅里叶域的过程通常都由三个主要阶段构成。即把图像转换到傅里叶域、在图像的这种表示上执行某些操作，然后再将其转换回正常的表示形式（称为空间域）。变换被称为前向变换和逆向变换。傅里叶变换是重要的。在图像处理计算机中执行这种变换的算法是在硬件中进行的这一事实就反映出这一点。

411

在将一个图像变换到傅里叶域中时没有信息的损失，图像中的可见信息只是以不同的方式进行了表示。对于一个不懂数学的人来讲，第一眼看到这个方法会认为它是一个很奇怪的难以控制的事物。在傅里叶域中表示的图像中的一个点含有关于整个图像的信息。点的值告诉我们图像中的空间频率有多大。

将一个图像$I(x, y)$的傅里叶变换定义为：

$$F(u,v)=\frac{1}{2\pi}\iint I(x,y)e^{-j(ux+vy)}dxdy$$

并定义其逆变换为:

$$I(x,y)=\frac{1}{2\pi}\iint F(u,v)e^{j(ux+vy)}dudv$$

傅里叶变换是一个复杂的量，可以将其用一个实数和一个虚数部分来表示:

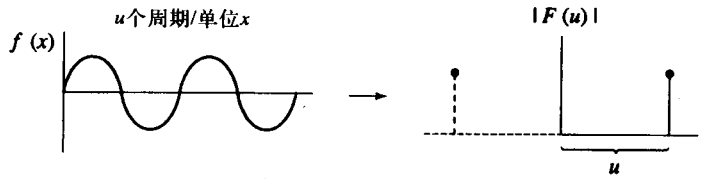
$$F(u,v)=\text{Real}(u,v)+\text{j Imag}(u,v)$$

可以把$F(u,v)$表示为两个函数，分别称为振幅和相能谱。

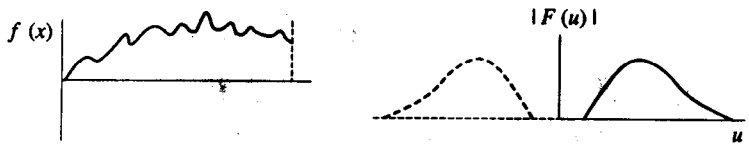
$$|F(u,v)|=(\text{Real}^2(u,v)+\text{Imag}^2(u,v))^{1/2}$$

$$\varphi(u,v)=\tan^{-1}(\text{Imag}(u,v)/\text{Real}(u,v))$$

现在，重要的是要对变换的性质有一种直观的了解，尤其是对一个空间频率的物理意义有正确的理解。首先，让我们来考虑一个单个变量的函数$I(x)$的较简单情况。如果我们把这个函数转换到傅里叶域中，则有变换$F(u)$。振幅部分$|F(u)|$定义了一组正弦曲线，当把这些正弦曲线加起来后，就产生原函数$I(x)$。相能谱部分定义了每个正弦曲线的相关系（在$x=0$处正弦曲线的值）。也就是说，在$|F(u)|$中的每一个点都定义振幅和单个正弦波分量的频率。另一种解释是任意函数$I(x)$分解成正弦波系数的一个集合。这种情况如图14-16所示。图的第一部分是一个正弦波曲线中的振幅谱，它在傅里叶域中只是一个点（实际上是关于原点对称的一对点）。第二个例子是一个含有信息的函数，它可以是一个语音信号。这个图中表现的是在傅里叶域中占据范围的一个图谱。从最小频率到最大频率的区间称为带宽。



a) 将一个正弦波映射到一个点上



b) 一个信息波的“窗口”映射成一个频谱

图14-16 一维傅里叶变换

一个二维函数$I(x,y)$，即一个图像函数被分解成一组空间的频率$|F(u,v)|$。一个空间频率是一个表面，即一个正弦波曲线的“波纹”，其频率或波浪起伏的速率由从原点到点(u,v)的距离给出:

$$\sqrt{u^2+v^2}$$

其方向，即波峰和峰底与x轴之间的夹角由下式给出:

$$\tan^{-1}(u/v)$$

一个点$F(u,v)$告诉我们，图像中包含有多少该空间的频率。图14-17是图14-16的一个二维的模拟。图中，一个正弦波具有空间的扩展，并被映射到傅里叶域中的一个点上（实际上是一对点）。如果现在让我们考虑一个图像$I(x,y)$，将这个图像映射到一个二维的频谱中，这

个频谱是两个变量u和v的一个函数。不同类型的图像会表现出不同类型的傅里叶变换，我们将用例子简单地说明这种现象。然而，大多数图像的傅里叶表示在(0, 0)处都有振幅的特性峰值，并且这个值随着空间频率的增加而降低。自然场景的图像展示出一种没有连贯性结构的傅里叶表示。一般来说，人造场景的图像在傅里叶域中表现出连贯性，它反映在原来的场景中出现了连贯性的结构（道路，建筑物等）。计算机图形学的图像通常在高的空间频率分量部分具有高能量，反映出在图像中出现了详细的纹理。

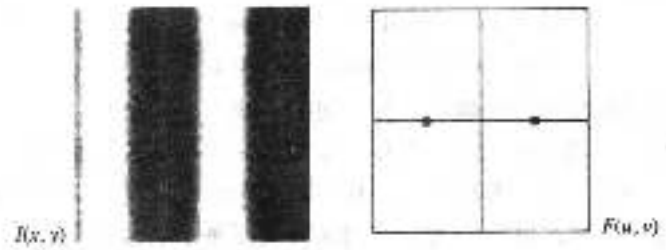


图14-17 一个由单个空间频率构成的图像及其傅里叶变换

412
↓
413

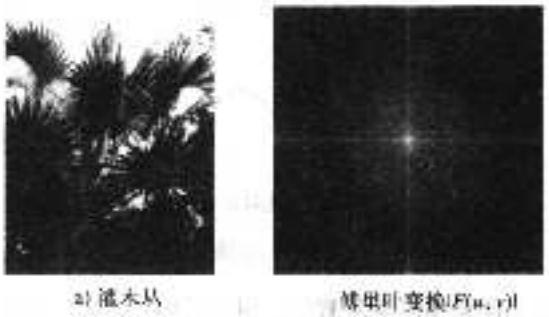
在图像处理中，一个重要的傅里叶表示的性质是在以原点为中心的一个圆的周边，它定义了一组相同波浪速率的空间频率：

$$r = \sqrt{u^2 + v^2}$$

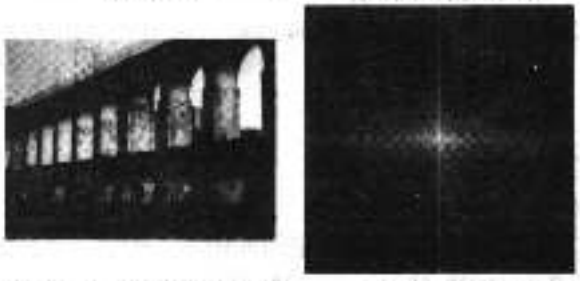
它具有各种可能的方向。

现在，通过考察三个不同的振幅谱的例子定性地研究变换的性质。

图14-18a是一个来自自然界的例子。它产生一个实质上没有连贯性的傅里叶变换。尽管在叶子的边界处有很多线性的结构，但是这些线有任何可能的方向，所以在傅里叶域中没有可见的连贯性。



a) 雨木从　　傅里叶变换|F(u, v)|



b) Arcos da Lapa（里约热内卢）　　傅里叶变换|F(u, v)|

图14-18 自然场景和人造场景的傅里叶变换

图14-18b是一个人造场景的图像，在傅里叶域中有明显的与场景相关联的线性结构。首先，有源自有轨电车路线的不连续性的线性结构（拱门的上方）。其次有上下拱门之间的不连续性，这在傅里叶域中给出另一条线。在v轴附近有连贯性，这是由于结构的水平边界产生的。由于这些线的方向因为摄像机的透视作用而绕着垂直线变化，所以它们在傅里叶域中被映射为非垂直的线。在傅里叶域中有一个垂直的连贯性，它与数据采集设备中的扫描线有关，这个连贯性也是由于长的阴影所导致的水平不连续性造成的。傅里叶域中的其他贡献来自图像中的自然成分，如拱墙上的纹理。

图14-19a和b是两个人造的纹理，应该清楚地理解纹理的连贯性和傅里叶域中的结构之间的关系。在两种情况下，纹理都用一个树叶遮挡着，这在傅里叶域中显示为一个模糊的“偏离垂直的”线。

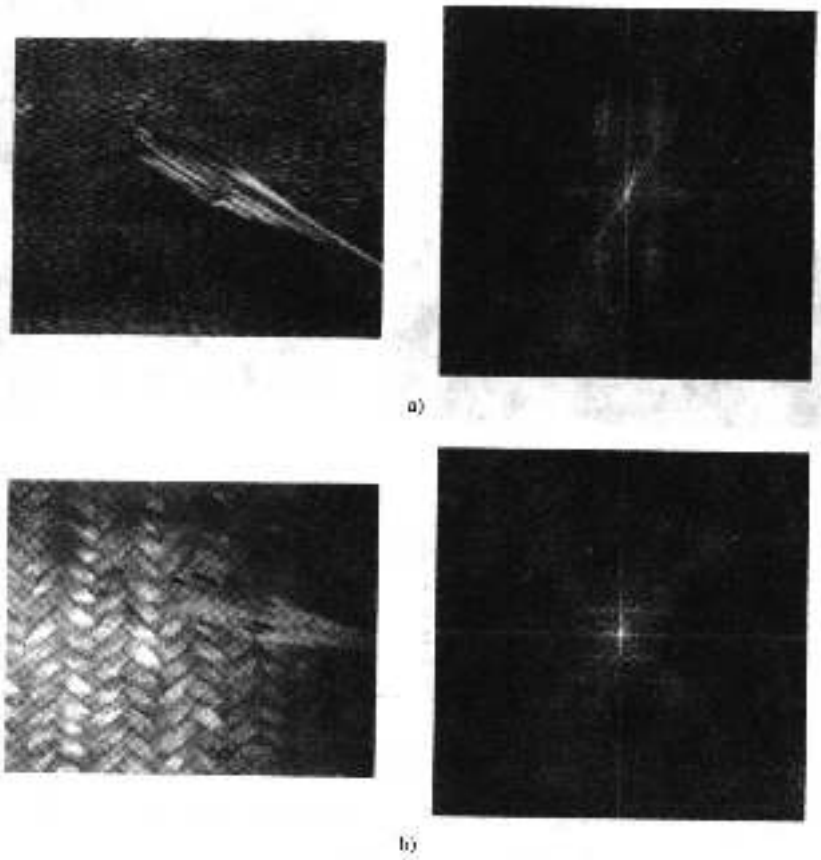


图14-19 纹理的傅里叶变换

从这些例子中我们能得出什么结论呢？一个非常重要的观察是，空间域中追布的信息在傅里叶域中被分开了。尤其是我们看到，在第二个例子中，图像结构中的连贯性在傅里叶域中反射成了线段或者穿过原点的轮辐。在第三个例子中，纹理产生的成分严格地位于傅里叶域中其占主导地位的空间频率中。傅里叶域的这个性质可能多数用于处理空间过滤，在这种情况下，我们可能希望增强某些空间频率并消除其他的频率以对图像产生特定的变化。它还

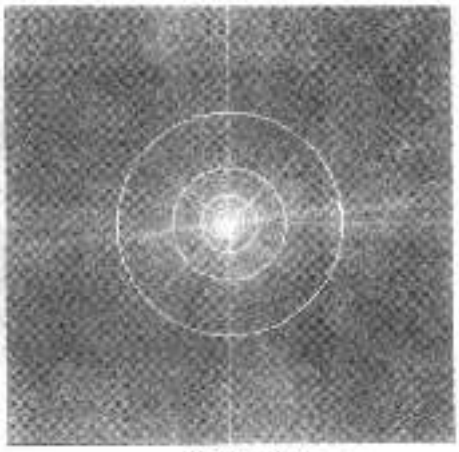
414 / 415

用于图像的压缩中，这时我们对图像的变换进行编码或量化，而不是对图像本身进行处理。这使我们有机会对于那些已知不太“重要”的变换部分用较少的信息进行编码。这是一个功能强大的方法，可以用少得多的信息对变换的某些部分进行编码而不会明显地降低图像的质量。图像中的原始信息在变换中被重新排了序，以使我们能够对图像域中关于其相对重要性进行方便的判断。

傅里叶域的一个非常重要的性质如图14-20所示，该图显示，大多数图像能量集中在低频部分。图中表示，在图中所示图像的傅里叶变换上圆以不同的半径叠加了起来。如果我们在每个圆内所包含的整个域上计算$|F(u,v)|^2$的总和的比例值，则可以求出如图14-20b所示的关系。



Pão de Açúcar（里约热内卢）



傅里叶变换$|F(u,v)|$

a)

半径（像素）	包含图像能量
8	95
16	97
32	98
64	99.4
128	99.8

b)

图14-20 在递增半径的同心圆中所包含的图像能量的百分比

416 在图像处理中作为基础的傅里叶变换对的一个性质称为卷积理论。这个理论可以写为：

$$f(x,y)*h(x,y)=\Im^{-1}[F(u,v)H(u,v)]$$

其中：

*指卷积。

用文字来表示则为：在空间域中图像函数$f(x,y)$与$h(x,y)$的卷积等价于（对于其逆变换的）在傅里叶域中$F(u,v)$和$H(u,v)$的乘积。其中：

$$f(x,y)=\Im^{-1}[F(u,v)]$$

以及：

$$h(x, y) = \Im^{-1}(H(u, v))$$

相似地，我们有：

$$I(x, y)h(x, y) = \Im^{-1}(F(u, v)*H(u, v))$$

这两个结果称为卷积理论。卷积及其特殊情形（交叉关联）是我们将一个超采样的图像过滤到屏幕分辨率时在计算机图形学图像上执行的操作。 417

第15章　颜色和计算机图形学

15.1　计算机成像中的颜色集
15.2　颜色和三维空间
15.3　颜色、信息和感知空间
15.4　绘制和颜色空间
15.5　关于监视器的考虑

引言

这一章讨论关于颜色的定量问题。在实际的计算机图形学中对颜色的大多数处理都是定性的。当建立一个场景数据库时，我们对于物体颜色的选择多多少少是任意的。然而，在计算机图形学中出现了某些应用，需要从颜色的角度对光线-物体相互作用有一个精确的模拟。而且，在可视化领域，用颜色来表示数值信息，必须结合人类颜色视觉系统中神经物理机制的知识来采用适当的数值信息进行颜色映射。

令人奇怪的是，一个将主要研究的努力都倾注于照片真实性的行业竟然差一点儿忽略了关于颜色的精确方法。毕竟，帧存储多年来已经成为一种司空见惯的方法，而每一帧中的像素能够显示16 000 000种颜色中的任何一种颜色。我们猜想其原因有三:

1) RGB或三种方程方式在绘制方法（如Phong明暗处理、光线跟踪和辐射度方法）中占主导地位，而且当在多于三种波长时运算这些模型的成本过高。

2) 绘制模型本身有明显的缺陷，这些缺陷在视觉上远远比颜色的细微处理严重得多（空间域的走样是可见的，而颜色域的走样一般是不可见的）。

3) 实践中缺乏对于精确的颜色处理的需求。

也有一些例外，例如，Hall and Greenberg（1983）和Hall（1989）对带有颜色处理的绘制方法进行了少量的研究。然而，也有越来越多的应用可能从精确的颜色模拟中得到好处。 418
有一种绘制方法（辐射度方法）在其对光线-物体相互作用的处理上足够精细，使其能够从这样的方法中获益。很清楚，这是CAAD（计算机辅助建筑设计）将来的主要发展方向之一。一个建筑设计的计算机图形学的可视化，不管是内部的还是外部的通常其自身都是可识别的。我们知道，图像并不是照片。这看起来主要是因为缺乏精细的几何细节。建模的成本是很高的，需进行近似。在辐射度方法中，强制性地进行一种粗略的细节模拟。所以，我们首先注意到了几何上的不恰当。然而，“二次”的努力无疑是重要的，而像不真实的阴影以及“看起来不是很正确的”光线这类情况都是计算机图形学图像的可见的符号。

另一个颜色尤为重要的领域是ViSC中的体绘制（见第13章）。在这一领域颜色被用来使观察者能够感觉到三维空间中数据值的变化，而这个变化可能非常细小。在这种情况下，重要的是所选用的颜色要按光学的方式传达信息。这一问题要依赖于感知颜色模型。

如果我们认为精确的颜色模拟是重要的，则将三个波长扩展到n个波长时除了成本上的考虑之外还会产生其他的问题:

1) 应采用什么颜色表示系统或模型来对所使用的颜色进行分类？很清楚，对于物体的反射性质和一个光源的强度我们可以采用波长的采样函数。尽管这样做对于计算可能是方便的（而且是必要的），但是这对于建筑师可能没有什么用处，比如说建筑师可能希望在标准的系统中用一个颜色标签或一个三元组来定义一个画笔的颜色。图像的存储和交流应采用什么颜色空间？存储n个波长计算的结果可能是极其不可行的。

2) 在选用精确颜色时的主要问题是再现和观察。在标准系统中定义的两种颜色对于观察者来说看起来应该是相同的。但是，这只有用精心校准的计算机图形学监视器，在相同的条件下观看时进行重现才能实现。尽管用一个色度仪可以对局部进行精确的颜色测量，但是视觉上的颜色偏差还是会出现的，比如说由于与周围颜色的对比作用。这类实际的问题不太容易克服，除非对于精确的颜色模拟的要求有一些松动。

15.1 计算机成像中的颜色集

为了在计算机成像过程中处理颜色，需要对颜色进行某种程度的量化。这使我们有了颜
419 色空间或颜色域的概念。这是一个三维的空间，在这个空间中有我们所关心的所有颜色。首先，需要定义一个颜色集的层次结构，以便参考。这些层次是:

1) 对于具有正常的颜色视觉的人类可感知的所有颜色的集合。

2) 可以由监视器的屏幕显示或者通过输入设备获取的颜色的集合。这是（1）的子集，其原因将在本章的讲解过程中得到澄清。

3) 可以由图形学程序计算并在帧存储器中存储的颜色集。对于一个24位的系统（16 000 000种颜色），一般来说这个集合是（1）的子集，但是是（2）的超集。也就是说，除非我们提前进行特殊的预告，否则，将可能产生超出监视器范围的颜色。

这个层次结构的示意图（见图15-1）是以一个三维颜色空间的截面的形式出现的，对于这个三维的颜色空间我们将在后面进行解释。

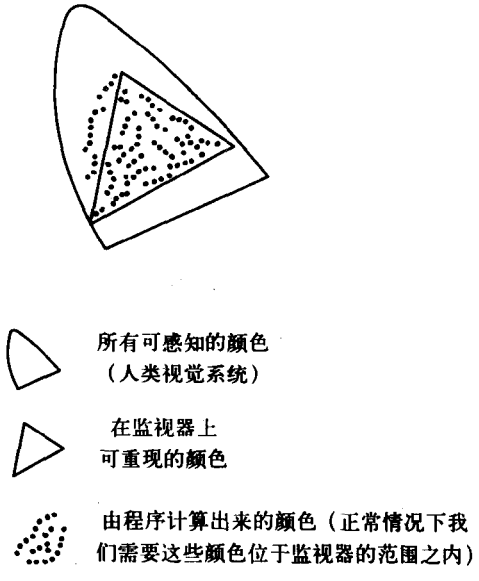


图15-1 与一个计算机成像相关的颜色集的层次结构。在这个空间中一个颜色是一个二维的点

15.2 颜色和三维空间

为什么颜色是一个三维的向量呢？我们必须谨记颜色是一种人类的感觉。传统上我们用文字来描述颜色，通常对于普通的物体间接的描述为“苹果绿”或“血红”等。油漆和燃料工业中是通过样本颜色图的形式来更精确地进行颜色的交流的。对于颜色的数值化的定义有 420
着悠久的历史，从牛顿时期就开始了。但是，只有到了20世纪数值系统在工业中才变得重要起来。

“为什么颜色是由三个数值标记定义的”这个问题的答案是，在我们的视网膜中有三个不同的锥面，它们对于不同的波长具有不同的敏感性（见图15-7a）。可以从物理上把光线定义为一个光谱的能量分布（SPD），把光能客观地度量为一个光波长的函数，可以用三个权重对任何SPD作用于人类观察者上的作用进行分类，即三个不同类型的锥面的相对反应。因此，就出现了可以通过添加三种颜色的混合光线来匹配视觉上的样本颜色的情况。例如，可以通过控制红光、绿光和蓝光三种光的光强度来匹配一个样本或目标颜色。但是，在与原始的红、绿、蓝色进行匹配时，重点是我们所基于的不是在锥面光谱敏感度曲线上SPD的标记，而是用人类的视觉系统去与原始颜色的一个混合物进行颜色的匹配。逐个波长地对所有的颜色进行这种匹配就导致出现光谱敏感度曲线，如果视网膜上的锥面最大程度地响应这些颜色的话，则我们的视网膜也会有这样的曲线。采用这种稍微有些绕弯儿的方法的原因是可以方便地从颜色的匹配试验出导出这些函数。而对于视网膜的实际光谱敏感度曲线的精确知识则较难得到。

于是，颜色的数值定义就是一个颜色基元的三元组。大多数（但不是全部）可感知的颜色都可以通过三种基色（例如，红色、绿色和蓝色）的适量混合来产生。如果用**C**来代表一个颜色，则:

$$\mathbf{C} = r\mathbf{R} + g\mathbf{G} + b\mathbf{B}$$

其中，r、g和b是为了要匹配指定的颜色所需的每一种基色的相对权重。这里的重点是，尽管这个系统没有直接地把定义的信息与颜色的SPD相关联，但是它却表明颜色**C**可以由一个数值三元组来定义。由于进行了匹配试验，观察者就可以选择分量r、g、b来匹配或模拟颜色**C**。

在计算机图形监视器上，颜色是通过激励由红、绿、蓝磷光体构成的相邻点的三元组而产生的。这些点很小，眼睛将这个三元组感觉成一个颜色的点。于是，我们就用三个基色来定义或标记真实的颜色，监视器上产生出来的颜色也以相似的方式进行定义。但是，请注意其重要的差别，即监视器上的颜色并不是由三个光源发出的光线的混合造成的，而是将光源相互靠近造成的。

可是，在计算机图形学中对颜色的三种分量的定义以及为监视器产生一个三分量RGB信号的需求导致了人们普遍假设光线-物体相互作用只需要在光谱的三个点上进行估算。这是“标准”的RGB范式，将在Phong明暗处理、光线跟踪和辐射度方法中采用。如果试图精确地模拟光线与物体在场景中的相互作用，则需要采用比三种波长要多的波长对这种相互作用进 421
行计算。否则的话，由于对光线分布和物体反射函数的欠采样，将会在颜色域中出现走样。当然，颜色域中的走样只是由于颜色偏离了所希望的效果造成的，而在这种情况下它是不可见的（这与空间域的走样正好相反，因为空间域的走样产生令人烦恼的视觉上的人工痕迹）。在大多数计算机图形学的应用中颜色在很大程度上是任意的，而一般来讲，在颜色域中，由

于不精确的模拟而导致的偏移也是不重要的。只有在应用中那些把颜色作为模拟的一个精细的组成部分的地方，比如说在内部设计中，才必须把这种作用考虑进来。

我们已经用数值标记表示或描述了对颜色的感觉，并且到目前为止对于我们所关心的颜色匹配试验来说，所面对的问题是：要选用哪一个数值？这就预示着要有不同的颜色空间或颜色域的概念。

如前面一节中所建议的那样，可能计算或绘制域是一个波长或光谱的空间。然而，最终我们需要在$RGB_{监视器}$空间产生一个图像来驱动特定的监视器。图像的存储和交流又是怎样的呢？在这里我们需要一个通用的标准。正如我们将要看到的，$RGB_{监视器}$空间是专门用于设备的。这些设备有不同的颜色范围，而所有这些范围又都是可感知颜色集中的子集。一个通用的空间将是独立于设备的，并且将涵盖所有可感知的颜色。这样一种空间是存在的，称为CIE XYZ标准。CIE三元组是一个与任意可感知的颜色相关的唯一的数值标记。

在计算机图形学中，另一个需求是一种允许对颜色进行管理和设计的功能。人们一般都认为允许用户对基色进行混合的界面是非直观的，在这种情况下像色调、饱和度和亮度这类感知的敏感性是人们所倾向于使用的空间。

下面列出计算机成像过程中所采用的主要的颜色空间。

1) CIE XYZ空间：这是用于颜色定义的主要的国际标准。颜色是用一组三个激励值或者人工的基色XYZ来定义的。

2) 对于不同的情况多年来建立起来的对CIE XYZ空间的变化或变换（比如说，CIE xyZ空间）：这些CIE XYZ的变换较好地反映了颜色感知中的一些细节，例如感知的直线性。

3) 光谱空间：在图像合成中，光源是在这个空间中以一个密度分布的n个波长样本进行定义的。物体的反射性也是按相似的方法定义的。逐个波长地定义颜色的方式与用像分光光度计这类设备测量颜色的方式是一样的。正如我们已经指出的那样，这与把SPD感知成一种颜色或者另一种颜色并不一定有关系。我们在n个波长上合成一个图像，然后还需要把它“降”
422 成三个分量以便显示。

4) RGB空间：这是Phong明暗处理的“标准”的计算机图形学范式。这只是光谱空间的一个三样本版本。光源和物体的反射性都是以三个波长定义的：红色、绿色和蓝色。我们知道，基色R、G和B是纯的、饱和的颜色。

5) $RGB_{监视器}$空间：在这个空间中，一个三元组在特定的显示器上产生出一个特定的颜色。换句话说，它是一个显示空间。相同的三元组在不同的显示器上不一定会有相同的颜色敏感性，因为监视器并不是在一个标准下标定的。监视器的RGB并不是纯的或饱和的基色，因为一个来自受激励的磷光体的光线的发射在频带上表现出一种光谱的能量分布。如果在绘制中使用了通常的三样本方法，则通常情况下，不管在RGB空间中计算出什么值都假设其是$RGB_{监视器}$空间中的权重。如果执行的是n个样本的计算，则用一个依赖于设备的变换产生$RGB_{监视器}$空间的一个点。

6) HSV空间：一个对RGB空间的非线性变换使得颜色能够按色调、饱和度和亮度来定义。

7) YIQ空间：在模拟电视中采用的对RGB空间的一种非线性变换。

现在，我们将讨论有关这些颜色空间的问题，我们将从RGB空间开始论述，因为它是最常见的，同时也是最易使用的。然后，我们将考察一些有关CIE空间的问题。

15.2.1 RGB空间

已知上述空间中（4）和（5）之间的细微差别，现在我们把RGB空间描述为一个通用的概念。这个模型是计算机成像中定义颜色的传统形式。例如，它使得明暗处理方程中的漫反射系数以一个三元组（R, G, B）的值的形式给出。在这个系统中，(0, 0, 0) 是黑色，(1, 1, 1) 是白色。在一个加性系统中，用红、绿、蓝三个基色将颜色标记为三个基色的相对权重。在这个系统中，可用的所有颜色是用RGB立方体表示的（见图15-2和图15-3（彩色插图））。关于RGB空间，要点是：

1）从感觉上看它是非线性的。在这个空间上相等的距离一般来说并不对应相同的感觉。在该空间中，一个区域中两点之间的距离可能不会产生感觉上的差别，而在另一个区域上相等的增量可能会产生可以觉察的颜色变化。换句话说，由多个RGB三元组可能产生相同的颜色。例如，如果每一个RGB都可以在0－255之间变化，则可以有16 000 000个不同的RGB编码。

2）由于RGB值以及在每一个磷光点上所产生的光强度之间的非线性关系（见15.5节），低RGB值在屏幕上产生小的变化。在低光强度时大约需要有20步才能产生“刚刚可以察觉的差别”，而在高光强处只要一步就可以产生一个可以感觉的差别。 423

3）在计算机图形监视器上产生的所有颜色的集合，即RGB空间，总是人类可以感觉出的颜色的一个子集。这对于RGB空间并不奇怪。任意三个可见基色的集合都只能通过对可感知颜色集合的一个子集的添加、混合来获得。

4）它不是一个好的颜色描述系统。如果没有足够的经验，用户就会发现很难以颜色标记给出RGB值，“什么时中棕色的RGB值呢？”一旦选择了一个，对这种颜色的性质进行细微的改变并不明显。例如，改变一种选定颜色的鲜艳性将需要对RGB分量进行不同的改变。

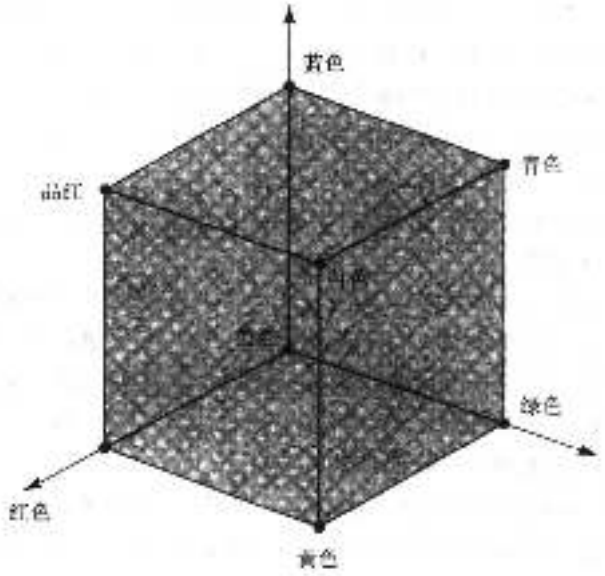


图15-2 RGB颜色实体，又见图15-3（彩色插图）

15.2.2 HSV单六面体模型

色饱和值（HSV）或者单六面体模型是由A. R. Smith在1978年（Smith 1978）提出来的。它的目的是实现一种比使用三基色进行颜色选择更直观的用户界面。颜色空间的形状是一个六边形的锥体或六面体。HSV锥面是RGB立方体的一个非线性变换，尽管人们倾向于将其作为一个感知模型，但是它还只是一种在监视器显示空间标记颜色的方法。在这种情况下，感知是指用于表示颜色的属性，它更接近于我们人类观察颜色的方式。但感知并不意味着这个空间在感觉上是线性的。对于RGB空间在感觉上的非线性被带到了HSV空间，尤其是感觉上
424 色调的变化是以角度来非线性地区分的。

可以将其用到任何用户希望以审美的或相似的观点选择或控制一种或一些颜色的上下文中。也可以基于感知地用色调、饱和度和亮度值来控制RGB监视器的显示范围。这意味着可以在那些容易感知出改变三个量之一的效果的地方建立用户界面。当使用了可感知的变量时，像使颜色更亮、更淡或更黄这类任务比必须确定需要对RGB变化采取哪些组合更容易一些。

HSV模型是基于极坐标的，而不是基于笛卡儿坐标的。H是按度定义的，范围从0 ~ 360。第一类基于极坐标和感知参数的颜色系统之一是由Munsell提出的。他的颜色概念系统是在1905年首次提出的，目前这种系统仍然在使用。Munsell把其感知变量称为色调、饱和度和亮度，我们不可能再有比这些更好的定义了。色饱和度与饱和度有关，而饱和度似乎在计算机图形学中更受欢迎。

Munsell的定义为：

- 色调：它是一种量，可以用来区分不同的色系，比如区分红色与黄色、绿色与蓝色等。
- 饱和度：它也是一种量，可以用来区分颜色的强弱，区分对一种颜色的感觉与对白色或灰色的感觉偏离的程度，区分一种有特色的色调的光强度，区分颜色的强度。
- 亮度：它是一种使我们能够把亮的颜色与暗的颜色相区分的量。

通过引用一组样本使用Munsell系统，即Munsell颜色手册。这些样本在颜色空间是以“刚好可辨别的”步长出现的。

HSV模型与艺术家混合颜色的方法有关。提到了为产生一种颜色需要从内心想象R、G、B的相对量的困难，Smith说到：

> 试着用这一混合技术从内心思考改变RGB的值以获得粉红色或棕色。有困难是很正常的……以下的[HSV]模型模拟了艺术家在其调色板上混合涂料的方式：他选择了一个纯的色调或者颜料，并且通过添加白色把这个色调调成淡色，又或者通过添加黑色把这个色调调成深颜色，又或者一般情况下是通过添加白色和黑色或灰色的某种混合物来获得该色调的一种调和色。

在HSV模型中，改变H就相当于选择一种颜色。降低S（对颜色去饱和）就相当于添加白色。而降低V（减少亮度）就相当于增加黑色。RGB和HSV空间之间的变换的推导通过考虑一个对六面体的几何解释就很容易理解。如果将RGB立方体沿着其主对角线投影到一个垂直于该对角线的平面上，则产生一个六角盘。

425 于是在六个RGB顶点与HSV模型中的六面体的六个点之间就建立起下列对应关系：

RGB	HSV	
(100)	红色	(0, 1, 1)
(110)	黄色	(60, 1, 1)
(010)	绿色	(120, 1, 1)
(011)	青色	(180, 1, 1)
(001)	蓝色	(240, 1, 1)
(101)	品红色	(300, 1, 1)

其中，H是以度来衡量的。这个六角盘是在六面体的模型中含有V = 1的平面。在RGB立方体中沿着主对角线上的每一个值（沿着增加黑度的方向）都定义一个其所包含的子立方体，每一个子立方体都定义一个六角盘。所有的六角盘堆积起来就构成了HSV的颜色实体。

图15-4是HSV的一个六面体，图15-5（彩色插图）是对它的进一步解释，显示出穿过白色轴的各个薄片。每一个薄片的右边一半都是恒定H的平面，而左边一半为H+180的平面。

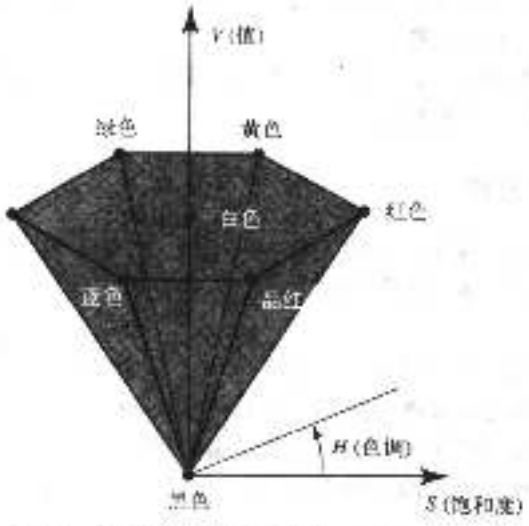


图15-4 HSV单六面体颜色实体，又见图15-5（彩色插图） 126

除了感觉上的非线性之外，HSV系统中的另一个细微的问题是从感觉上来看各个属性并不是独立的。这也就是说当实际上改变的是亮度值时，所检测到的是色调上的明显变化。

最后，也许与感觉的真实性出现最严重的偏移的是模型的几何形状。颜色空间标识了所有在计算机图形监视器上可重现的颜色，这意味着在常数V的平面上的所有颜色都是等亮度的。但是情况并非如此。例如，最大光强度的蓝色所感觉出的亮度要比最大光强度的黄色的亮度低。由此我们得出结论，在最大V值处，不同的色调感觉表现出不同的值。用任何“常规的”几何实体（比如立方体或者六面体）代表一个监视器的显示范围只是对颜色敏感性的一种近似，这一事实说明，必须考虑我们在感觉上所基于的颜色空间。

表示这种事实的一种简单的方法是重申颜色是一种感觉的信号，不可能通过将监视器的电压水平进行划分再对这种度量进行颜色的标识方法来精确地标识它。最重要的是，我们在用RGB和HSV模型做什么，而文字“感觉上的”与HSV模型之间的关系却是混乱的。

15.2.3 YIQ空间

YIQ空间是RGB空间的一种线性变换，它是模拟电视的基础。其目的是解决带宽使用的效率问题（与RGB形式相比较），并保持与黑白电视的兼容性（黑白接收器所需的所有信息都包含在Y分量中）。

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.144 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

注意，常数矩阵的系数意味着变换假设RGB分量自身是相应于一个标准来定义的（在本例中这是一个NTSC定义）。Y分量与CIE的Y基色是相同的（见15.3.1节），称为亮度。颜色信息被隔离在I和Q分量中（相等的RGB分量将导致I和Q值为零）。这就出现了带宽优化问题，因为人类在这种情况下对于亮度变化的敏感程度比对颜色的敏感程度要高。我们可以在灰度等级的变化方向进行比颜色更加详尽的空间细化。因此，对于I和Q分量可以允许有较低的带宽，这就导致采用RGB分量时带宽上的节省。

在颜色和亮度信息分离的情况下，当我们希望对图像的结构进行操作而不影响图像的颜色时，颜色的表示在图像处理中是重要的。

15.3 颜色、信息和感知空间

现在让我们来考虑计算机成像中对于感知空间的运用。在实际应用中，我们将考察CIE XYZ空间——这是一个国际的、基于数字的颜色标记系统，于1931年首次提出，是由颜色匹配试验导出的。

为了处理颜色真实性问题，需要控制与感知的经验有某些关系的空间中的颜色。我们已经提到有关考虑这类问题很重要的那些专门的应用程序。例如，用于内部设计的DAAD，对于在像昂贵的耐用消费品（比如汽车等）的纤维或终端的设计中，计算机图形学就有必要从
427 随意的RGB域中移到一个能精确模拟颜色的空间中来。当然，在计算机图形学模拟中试图变换真实性的一种幻象时，还涉及到表面纹理、颜色的微观几何学（例如，金属漆或常规的彩色漆）以及几何上的精确度，但是这时，计算机图形学中的情况是RGB三元组已经是绘制的事实上标准了。

颜色多数用于在可视化的应用中传递数值的信息。这有一个很长的故事。可能最熟悉的证明就是一个彩色的地质图。在这个图中选择颜色来代表高度。传统上，所选的颜色中以绿色代表低的高度。从0 ~ 100m范围上的高度可以通过改变由绿到黄的亮颜色的深度方法来标识。暗棕色的颜色可以代表1000 ~ 3000m的范围。3000m以上通常都是粉红色的两种深度。白色代表6000m及以上的高度。

这种技术一直用在图像处理和所谓的伪彩色增强的计算机图形学中。最常见的情况是将其用在显示二维空间中的一个双变量函数$f(x, y)$上，在用等值曲线显示这样一个函数之前采用。在伪颜色增强中选择了一个经仔细限制的颜色列表（比如，10个颜色的列表），将f的值映射到最靠近的颜色上。这种图看起来像是一个地质图，一种颜色的岛以另一种颜色作为背景。

在计算机图形学和图像处理中，最流行的将$f(x, y)$映射成颜色的是彩虹颜色的某些变化使红色用于代表高处或热，而蓝色用于代表低强度或冷。换句话说，这是在HSV空间外部边界

周围的一条路径。这种映射的问题之一是根据所采用的颜色等级数的不同，不同颜色之间的变换看起来好像是错误的轮廓线。在函数f连续的地方出现了严重的颜色不连续性。这里存在一个矛盾：我们需要有这种不连续性来突显函数的形状，但是它们可能很容易被解释为在没有函数变换的地方出现了变换。尤其在非数学图像中更是如此，这种图像可能不会是处处连续的。不管怎么说自然的不连续是存在的，比如说由一个设备对不同组织的响应而生成的医学图像。在这种图中出现的错误的轮廓线可能是不希望的。

于是，从f的性质和形状的感知上是否进行轮廓线的加减最后还要依赖于图像的内容。错误的轮廓线的效果很容易通过向映射添加更多的颜色来消除，但是，这样做的后果是使得函数更难于解释了。

在数值信息上下文中，采用感知颜色空间是非常重要的。如果需要在颜色和数值信息之间有一个精确的关联，则应该采用一种感觉上是线性的颜色等级。我们在15.2.1节中讨论了RGB空间的感觉的非线性。很明显，除非要处理这类因素，否则，它将防碍一种颜色与一个数值的关联。除了文化上的关系，比如像地图绘制中的地质图像编码的例子，没有理由将色 428
调圆作为伪颜色的刻度。

在二维空间使用伪颜色来显示两个空间变量的函数的方法已经出现很多年了。在过去的10年中，三维计算机图形学技术在科学结果和模拟的可视化（这是一个被称为ViSC的领域）中的应用增加了。所采用的图形学技术主要是动画、体绘制（本书对这两方面都有论述）以及三维空间中伪颜色的使用，现在我们来讨论后者。

图13-1（彩色插图）示意了一种应用。它表现的是从对一个逆向流管道燃烧器的Navier-Stokes模拟中抽提出来的等值表面。在这个模拟中，基本的气流是从左向右的。空气从左侧在压力下被挤入燃烧室，并用两个风扇进行驱散。八个燃料喷嘴放射性地分布在燃烧器的中段。喷嘴朝着将燃料的混合物在一个通向燃烧室前方的螺旋状路径中输送燃料的方向。可燃物的混合在中心区域进行，在右侧的废气出口处建立起推动力。所示的等值表面把所有的点连接起来，沿着长轴的净流量为零，即一个零速度的表面。

这种等值表面可以用传统的三维绘制技术来显示。如示意图所示。在第二个示意图中，我们试着附加了一个伪颜色来代表温度。使用了在HSV锥体的周围从蓝到品红的一个光谱颜色路径。

这样一来，在相同的三维图像上，我们试图同时表现的是两个函数。第一个函数是一个等值表面的形状，第二个是等值表面的每个点上的温度。在这种情况下出现了感知问题，因为我们用颜色标识形状和温度，而正常情况下颜色是与一种现象相关联的。例如，在这种表示中要在一个色调或温度迅速变化的区域上解释等值表面的形状是困难的。但不管怎样，这类表示方法在可视化技术中越来越常见。它们代表了对复杂数据的一种总结。在采用三维计算机绘图之前，只能每次考察一部分信息。例如，示意图中的有关模拟可能已经用一个旋转截面进行了研究。这就把建立数据的三维图像这样的困难任务留给了观察者。

15.3.1 CIE XYZ空间

我们在前面的章节里已经论述了需要有一个光谱空间来模拟真实性。这指的是需要有一种方法来为监视器的显示“减少”或转换光谱空间的计算。而且，我们已经看到，还需要有感知的颜色空间来为伪颜色增强选择映射。在计算机图形学中，感知颜色空间存在的另一个理由是 429
计算机图形学领域中文件的存储和交流，以及计算机绘图者与使用颜色的工业设备之间的通信。

CIE标准允许以一个数值的三元组（X, Y, Z）的形式定义一种颜色。CIE XYZ空间包含了人类可以感觉出的所有颜色，并且是基于由试验确定的颜色匹配函数。因此，与前面所述的三种颜色空间不同，它并不是一个监视器的显示空间。

这个在1931年正式通过的标准的基础是颜色匹配试验，这个试验中用户控制三种基本光源或对其给出权重来与目标单色光源相匹配。所用的光源大多数情况下是单色的，其中**R** = 700nm，**G** = 546.1nm，**R** = 435.8nm。换句话说，下式中的权重是由试验确定的:

$$\mathbf{C} = r\mathbf{R} + g\mathbf{G} + b\mathbf{B}$$

这种试验的结果可以由颜色匹配函数来总结。这些匹配函数如图15-6b所示。它表明红、绿、蓝光混合的量将在一个标准的观察器上产生一个单色光，该光的波长由λ给出。即:

$$\mathbf{C}_\lambda = r(\lambda) + g(\lambda) + b(\lambda)$$

对于任何表现出能量为SPD $P(\lambda)$ 的颜色敏感度**C**，其r、g、b由下面的式子给出:

$$r = k\int_\lambda P(\lambda)r(\lambda)\mathrm{d}(\lambda)$$

$$g = k\int_\lambda P(\lambda)g(\lambda)\mathrm{d}(\lambda)$$

$$b = k\int_\lambda P(\lambda)b(\lambda)\mathrm{d}(\lambda)$$

于是，我们看到，颜色匹配函数还原一个颜色**C**，即将任何形状的光谱能量分布到一个三元组rgb上。到了这个阶段，我们可以得出一个非常重要的结论，即三元组rgb与前面提到的（计算机图形学）系统中的三元组RGB没有任何关系。正如15.2节中所述，计算机图形学把三元组RGB理解为一个照明的SPD中的三个样本，或者是物体的反射函数的三个样本，在绘制模型中这些样本线性地结合在一起产生一个反射光线的经计算的RGB值。换句话说，可以用三个样本来进行绘制，或者也可以将这个方法扩展成n个样本。而与此相反，三元组rgb并不是SPD中的三个样本，而是通过对SPD与每一个匹配函数的乘积进行积分得到的值。也就是说，它是SPD的一个定义，就像人看到的那样（从颜色匹配的角度来说），而不是如一个分光光度计所见的那样。

然而，在用一个附加基色的系统代表颜色时有一个问题，即若用正的权重只有可感知颜色中的一个子集可以用权重（r, g, b）进行描述。问题出自这样的事实，即当混合两种颜色时，
430 其结果是一种欠饱和的颜色。通过添加颜色不可能形成一种高度饱和的颜色。三个基色的任意集合形成一个受限制的空间，在这个空间之外还存在着一些可以感知的高度饱和的颜色。对于这样的颜色需要负的权重值。

为了避免负权重的出现，CIE建议采用三个超饱和的（或非真实的）基色X、Y和Z的一种标准。使用这些颜色，当加入混合物时用正的权重就可以产生所有可感知的颜色。图15-6c中所示的三个相应的匹配函数$x(\lambda)$，$y(\lambda)$ 和$z(\lambda)$ 总是正的，于是我们有:

$$X = k\int_\lambda P(\lambda)x(\lambda)\mathrm{d}(\lambda)$$

$$Y = k\int_\lambda P(\lambda)y(\lambda)\mathrm{d}(\lambda)$$

$$Z = k\int_\lambda P(\lambda)z(\lambda)\mathrm{d}(\lambda)$$

其中：

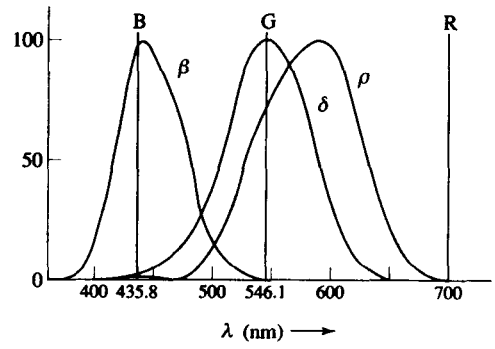
$k = 680$，对于自照明的物体。

在视网膜中ρ、δ和β锥面的光谱敏感度曲线，以及其与如下单色光的关系：

红色 = 700nm

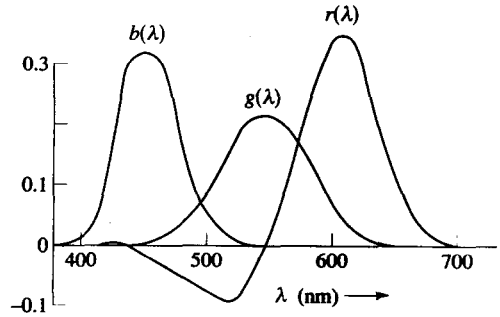
绿色 = 546.1nm

蓝色 = 435.8nm



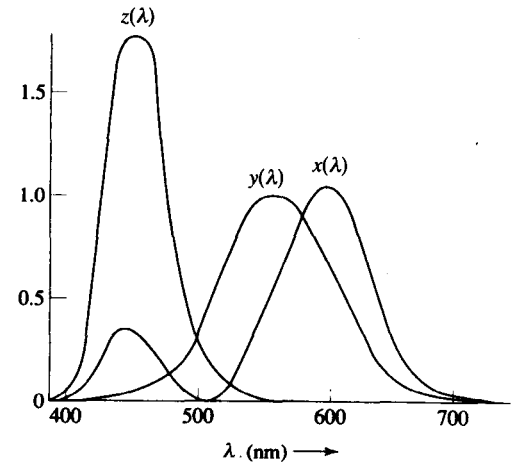
a)

对于CIE 1931标准彩色观察器的RGB颜色匹配函数



b)

对于CIE 1931标准彩色观察器的CIE匹配函数

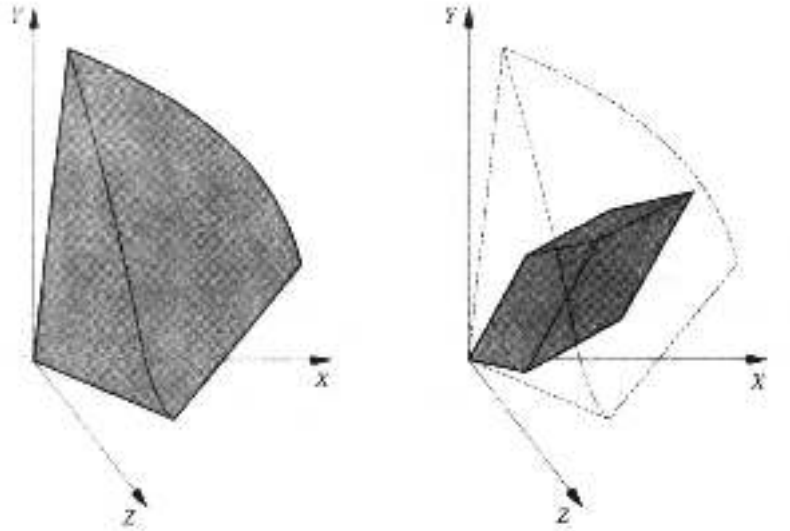


c)

图15-6 CIE颜色匹配函数的发展

由XYZ的值对于所有可感知的颜色所形成的空间是CIE XYZ空间。匹配函数是试验结果的变换。此外，定义了匹配函数$y(\lambda)$，以便有一个相应于人眼的发光效率特性的颜色匹配函数，这是一个峰值在550nm（黄-绿）的函数。

CIE XYZ颜色实体的形状基本上是圆锥形的，其锥面的顶点在原点处（见图15-7）。在这个图中还示意了一个监视器的显示范围，它像一个平行六面体。如果用这个空间与HSV空间相比较，可以看到的是变形的HSV空间，原点处是黑点，HSV空间被变形以便包围所有的颜色。并且顾及到这样的事实，即这个空间是基于感知测量的，例如，如果我们考虑变形锥体的外表面，那么这个表面是由光束形成的，这些光束从原点发射出来，在圆锥体的边界处终止，在任一光束上都是一组相同色度的颜色（见下一小节）。如果有一束光线朝着白点进入，并落在变形的圆锥体的基座上，则得到由此光线定义的去饱和颜色集合。在这个空间中，监视器的显示范围是一个变形的（切变并且缩放的）立方体，形成可感知颜色体的一个子集。



a) CIE XYZ实体　　b) 在CIE XYZ空间中一个典型的监视器显示范围

图 15-7

15.3.2 CIE xyY空间

定义（X, Y, Z）三元组的另一种方法是（x, y, Y），其中（x, y）称为色度坐标：

$$x = \frac{X}{X+Y+Z}$$

$$y = \frac{Y}{X+Y+Z}$$

对于所有的可见颜色给x，y产生一个二维的（x, y）空间，称为CIE色度图。

翼形的CIE色度图（见图15-8）在颜色学中被广泛采用，通过忽略亮度Y，在二维空间中包含了所有可感知的颜色。纯的饱和颜色或光谱颜色的轨迹由曲线形成，从蓝色（400nm）到红色（700nm），端点之间的直线称为粉红色或品红色线。沿着这条线上的点是粉红色或品红。这些颜色的感知是不能通过单色的激励来产生的，也不能从日光中分离。

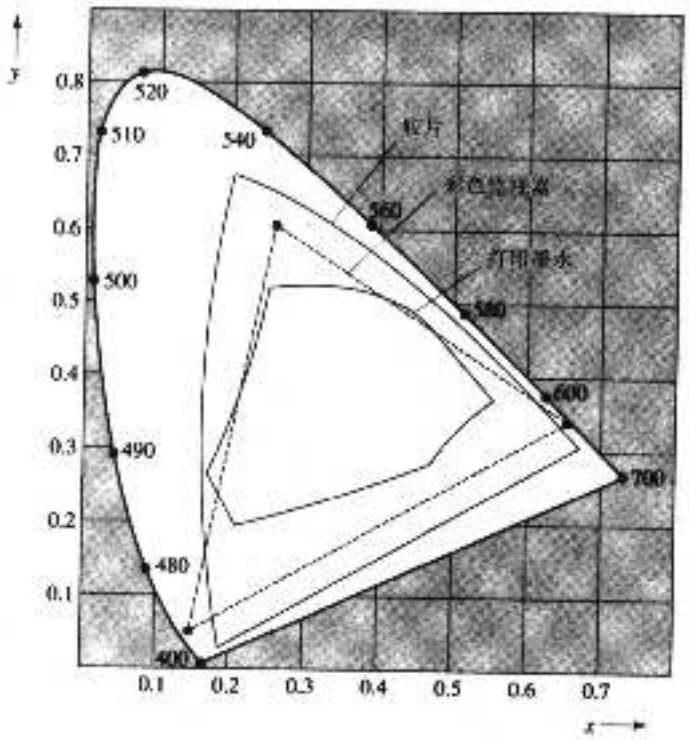


图15-8 显示彩色胶片、彩色监视器和打印墨水的典型显示范围的CIE色度图

图15-8还表明，这个颜色的作用范围在计算机图形监视器上可由三个荧光体再现。监视器的显示范围是一个三角形，是通过在三个RGB点上连接直线形成的。RGB点被包围在单色或饱和颜色的轮廓曲线之内。对于荧光体的发射特性的检查将展现出一个在主要波长周围的分布，这意味着颜色中含有白色，并且是不饱和的。例如，当完全激发蓝色和绿色的荧光体时，它们的发射性质加到一起进入一个较宽的带，这意味着所产生的颜色将不像蓝色或绿色那么饱和。

在CIE xyY空间中，三角形的监视器显示范围在大多数计算机图形学的教科书中有关颜色学的章节都可以找到，但是却有一些误导。这个三角形实际上是监视器显示范围的CIE xyY空间的投影。如图15-9是CIE xyY空间中监视器显示范围的一般形状，图15-10（彩色插图）显示了这个空间中的三个截面。对XYZ空间中的缩放并且切变的立方体进行几何或形状的变换使其成为xyY空间中的具有六个面的曲线组成的实体是难于解释的。例如，将立方体的一个边映射到一个点上。

CIE色度图有一些重要的用途，这里给出一个重要的例子。它可以用于比较各种显示设备的显示范围。在计算机图形学中，当一个图像最终要在不同的设备上重现时这种比较是重要的。如图15-8所示是一个CIE色度图，并含有一个典型计算机图形监视器的显示范围以及现代的打印墨水的显示范围。打印墨水的显示范围包含在监视器的显示范围之内，而后者本身又被彩色胶片的显示范围所包围。这就意味着某些在胶片上可以获得的颜色在计算机图形学监视器上是不能再现的，而某些在监视器上可以出现的颜色打印时也不能再现。显示设备和再现技术的显示范围总是包含在可感知颜色的范围之内，饱和颜色或光谱颜色是最难重现的。但是，一般来说这并不成为问题，因为自然现象中一般并不会出现光谱或接近光谱的颜色。

真正重要的是设备显示范围的相对覆盖范围，而不是任何相对于可视的显示范围的大小。

15.4 绘制和颜色空间

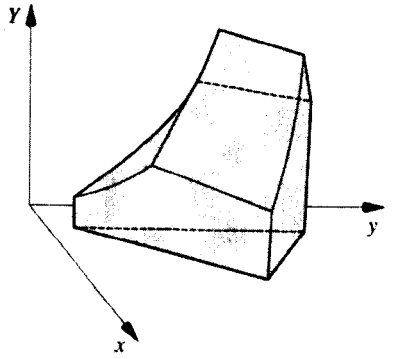


图15-9 在CIE xyY空间中的监视器显示范围（又见图15-10（彩色插图））

我们已经讨论了计算机图形学中缺乏精确颜色的原因，现在更仔细地研究这些原因当中的一个——颜色的走样是不可见的。

物理学告诉我们，由表面反射出的光线是波长的一个函数，它是依赖于表面反射函数的波长与光源的光谱能量分配函数的乘积。如果我们仅仅在三个波长上估算这个乘积（如第6章中所讨论的RGB Phong明暗处理模型），则很明显，由于总的欠采样，我们将不会产生一个模拟真实特性的结果。所发生的情况是三样本的方法将产生一种偏离真实颜色的颜色。但是，这种偏离在大多数情况下是完全不可见的，因为事实上我们对于从计算机图形学模型应该产生出什么特定的颜色并没有什么期待。错误的颜色看起来也并不一定是错的。

为了数值地模拟真实的颜色相互作用，我们可以把三样本的方法扩展到n个样本，并且在光谱空间进行运算，对光源分布函数和物体的反射性以适当的波长间隔进行采样。

下面讨论图15-11中总结的三种方法。第一种实际上是事实上的标准绘制方法，它在大多数近似方式中都没有考虑颜色的例外情况。照明的SPD是按三个波长进行采样的，或更通常的情况是将白色随意地定义为1, 1, 1。同样地，物体的反射也是在每一个R、G、B波长上定义的。应用了三个绘制方程/模型，并将所计算出的RGB强度直接反馈到监视器上而没有进行进一步的修正。这种方法产生输入值随意的作品，用户可能希望绘制一个暗红色的物体，但是他可能并没有考虑要以任意的精确度定义物体的颜色以及照明。仅仅利用了三个绘制方程。

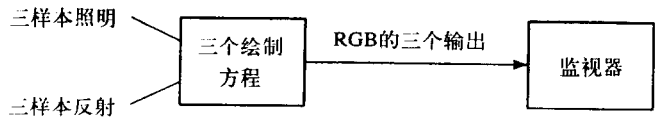
第二种方法对于一组波长（n = 9可能是一种好的折中方案）在光谱空间应用绘制方程。在这里，绘制的成本至少比“随意的”颜色方法多3倍。绘制程序的输出是一个经采样的强度函数，这个函数必须转换到用于显示的（三样本）$RGB_{监视器}$空间。这就是说，如果我们对于n个波长的绘制出现了问题，则希望显示尽可能精确的结果。我们还需要某些参数以导出从光谱空间到$RGB_{监视器}$空间的变换（见15.5.2节）。

431 ~ 435

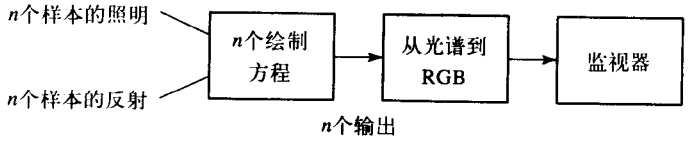
在最后一种方法中，我们是在CIE空间绘制的。这就暗示要用匹配函数将SPD照明定义为CIE XYZ值。然而，我们遇到了表面反射性的问题。对此我们应该选用什么值呢？这是一个细致的问题，读者可以参考Borges（1991）的文章，该文章严谨地讨论了这个问题。我们在这里可以指出，可以简单地把反射函数表示为一个CIE XYZ的三元组，并在一个三方程的绘制方法中使用这个三元组。绘制程序的输出是一个CIE XYZ的三元组。接着需要有一个从CIE到$RGB_{监视器}$的变换来显示结果。

对于光线跟踪程序，其由光谱绘制方法和RGB绘制方法所产生的图像之间的差别见图15-12（彩色插图）。

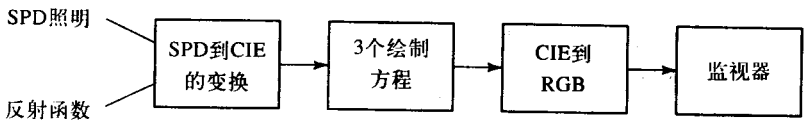
必须记住，现在我们只关心对其真实性模拟的一个方面，即防止由于在光谱空间的欠采样而导致的错误的颜色偏移。颜色也可以由局部反射模型自身来确定，但仍然存在着反射模型模拟真实性时精确度上的缺陷。不能简单地通过增加光谱空间的样本数来解决这个问题。



a) 对于“随意”颜色应用的“标准”绘制



b) 对于颜色敏感应用的光谱空间绘制



c) 对于颜色敏感应用的CIE空间绘制

图15-11 绘制策略和颜色

15.5 关于监视器的考虑

15.5.1 $RGB_{监视器}$和其他监视器的考虑

在计算机成像时严谨地利用颜色需要小心地关注监视器的某些特性。计算机图形监视器并不是标准化的，将相同的RGB三元组应用到不同的监视器上将在屏幕上产生不同的颜色。 436
最重要的因素是:

1) 监视器上的颜色并不是通过对三种颜色重叠混合产生的，而是依赖于眼睛在空间上对由三个荧光点产生的很细的光线进行混合而产生的。关于这一点我们不可能进行任何变动。其后果之一就是饱和颜色没有按照满亮度进行显示。一个纯红色的区域只是1/3的红色和2/3的黑色。例如，这意味着即使是人类也似乎要对这个效果采取补偿措施，因为直接从屏幕上获取照片会产生不好的效果。

2) 不同的监视器所带的荧光体具有不同的光谱能量分布。例如，用不同的荧光体来获得不同的持久性（一个荧光体被激励之后发光时间的长度）。这可以通过一个线性变换来进行校正，我们将在下一小节进行介绍。

3) 应用于监视器上的RGB值与屏幕上产生的光强度之间的关系是非线性的。对于这种情况的解决方法即是γ校正，将在下一小节进行介绍。这种校正是一个非线性变换。

4) 在图像合成时，明暗处理方程可以产生处于监视器显示范围之外的颜色——不可显示的颜色。因此，我们必须对这些颜色进行裁剪，使其能够回到监视器的显示范围之内。这也是一种非线性的运算。

15.5.2 关于监视器的考虑——不同的监视器和相同的颜色

在计算机成像时产生真实颜色的上下文例如在光谱空间进行绘制并用感知空间进行映射。

对于光谱空间，我们可以从最终的结果集中产生一个CIE XYZ的三元组。将CIE XYZ空间用作一个最终的标准，我们需要有一个针对设备的变换以便从CIE XYZ空间变换到特定的$RGB_{监视器}$空间。

可以有:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{bmatrix} \begin{bmatrix} \mathrm{R}_m \\ \mathrm{G}_m \\ \mathrm{B}_m \end{bmatrix} = \boldsymbol{T} \begin{bmatrix} \mathrm{R}_m \\ \mathrm{G}_m \\ \mathrm{B}_m \end{bmatrix}$$

其中，$\boldsymbol{T}$是专用于监视器的值，假设荧光体的输出与RGB值之间是一种线性关系。如果$\boldsymbol{T}_1$为监
437 视器1的变换，$\boldsymbol{T}_2$是监视器2的变换，则$\boldsymbol{T}_2^{-1}\boldsymbol{T}_1$将监视器1的RGB值变换成监视器2的RGB值。$\boldsymbol{T}$可以按下式计算。我们定义:

$$D_r = X_r + Y_r + Z_r$$
$$D_g = X_g + Y_g + Z_g$$
$$D_b = X_b + Y_b + Z_b$$

得到

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} D_r x_r & D_g x_g & D_b x_b \\ D_r y_r & D_g y_g & D_b y_b \\ D_r z_r & D_g z_g & D_b z_b \end{bmatrix} \begin{bmatrix} \mathrm{R}_m \\ \mathrm{G}_m \\ \mathrm{B}_m \end{bmatrix}$$

其中:

$$x_r = X_r/D_r \quad Y_r = Y_r/D_r \quad z_r = Z_r/D_r \text{ 等}$$

把系数写成两个矩阵的乘积，有

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} x_r & x_g & x_b \\ y_r & y_g & y_b \\ z_r & z_g & z_b \end{bmatrix} \begin{bmatrix} D_r & 0 & 0 \\ 0 & D_g & 0 \\ 0 & 0 & D_b \end{bmatrix} \begin{bmatrix} \mathrm{R}_m \\ \mathrm{G}_m \\ \mathrm{B}_m \end{bmatrix}$$

其中，第一个矩阵是监视器荧光体的色度坐标系。现在我们定义，相等的RGB电压（1, 1, 1）产生标定的白色:

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} = \begin{bmatrix} x_r & x_g & x_b \\ y_r & y_g & y_b \\ z_r & z_g & z_b \end{bmatrix} \begin{bmatrix} D_r \\ D_g \\ D_b \end{bmatrix}$$

例如，对于标准的白色D_{65}，我们有:

$$x_w = 0.313 \quad y_w = 0.329 \quad z_w = 0.358$$

将白色点进行缩放给出单位亮度，产生:

$$X_w = 0.951 \quad Y_w = 1.0 \quad Z_w = 1.089$$

对于一个交织的监视器（长的荧光持久性），示例的色度坐标为:

	x	y
红色	0.620	0.330
绿色	0.210	0.685
蓝色	0.150	0.063

用这些坐标我们得到:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.584 & 0.188 & 0.179 \\ 0.311 & 0.614 & 0.075 \\ 0.047 & 0.103 & 0.939 \end{bmatrix} \begin{bmatrix} R_m \\ G_m \\ B_m \end{bmatrix}$$

438

对系数矩阵求逆得到:

$$\begin{bmatrix} R_m \\ G_m \\ B_m \end{bmatrix} = \begin{bmatrix} 2.043 & -0.568 & -0.344 \\ -1.036 & 1.939 & 0.043 \\ 0.011 & -0.184 & 1.078 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

负数分量的重要意义在于，RGB是XYZ空间的一个子集，位于监视器显示范围之外的XYZ颜色将产生负的RGB值。

15.5.3 关于监视器的考虑——颜色显示范围的映射

监视器的显示范围一般都是相互覆盖的，在一个监视器上可以显示出的颜色可能在另一个监视器上不能重现。在应用了变换$T_2^{-1}T_1$之后RGB的值小于零或大于1就证明了这一点。这个问题在绘制时也会发生。在进行精确的颜色模拟时，若采用真实的颜色值，计算产生的颜色三元组可能会位于监视器的显示范围之外。换句话说，一般而言，图像的显示范围要比监视器的显示范围大，这个问题在打印机这类硬拷贝设备上更突出，因为打印机显示范围比监视器的显示范围要小。

处理的目的就是压缩图像的显示范围，使其在保持图像质量的前提下刚好与设备的显示范围相匹配。一般来讲这将依赖于图像的内容，这个问题目前仍然还是一个研究课题。然而，还有一些简单的策略是我们可以接收的。从一个处于监视器的显示范围之外的图像产生一个可显示的颜色的过程称为“颜色裁剪”。

很明显，我们可以采用一种简单的夹紧方法，限制范围之外的颜色。Hall（1989）提出了一种较好的策略。不可显示的颜色可分为如下两类:

1) 具有在监视器显示范围之外的色度的颜色（负的RGB值）。

2) 具有可显示的色度，但是其强度超出了监视器范围的颜色（RGB值大于1）。

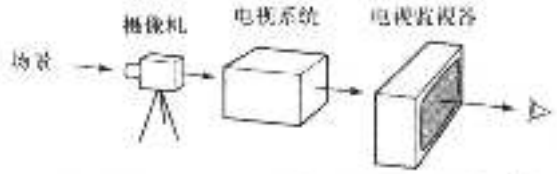
任何修正都会导致对所计算的颜色的偏移或改变，我们可以根据希望对色调、饱和度和亮度中的哪一个容许偏移来选择一种方法。

对于第一类情况，最好的办法是向颜色中加入白色，或者对该颜色去饱和直到它成为可显示的。这样就以损失饱和度为代价保持了色调或主要的波长以及亮度。在第二类情况下，

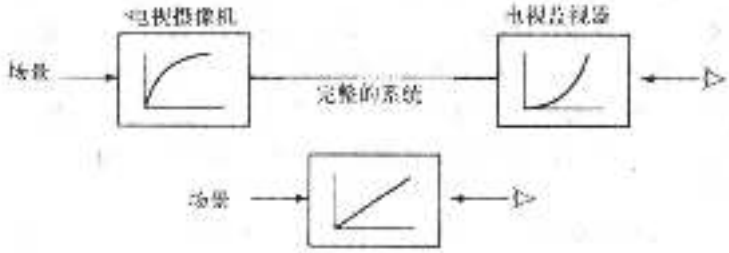
有一些可选方案。可以对整个图像进行缩放直到其最高的光强度位于范围之内为止，这样处理的效果与减小摄像机的孔径相似。另一种选择方案是保持色度，对光强度进行缩放，最后，
439 可以保持主要的色调和光强度，而通过加入白色降低饱和度。

15.5.4 关于监视器的考虑——γ校正

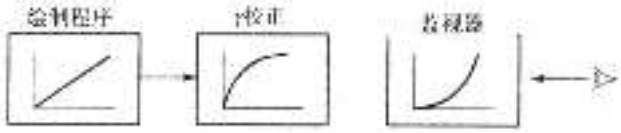
前面的所有讨论都隐含着如下的假设，即输入到监视器中的实际的RGB值与屏幕上所产生的光强度之间有一种线性的关系。但是情况并非如此。我们需要保持线性是由于这样的事实，即需要尽最大的可能使观察者在监视器上看一个场景的电视图像所感觉出的颜色关系与他从场景中所感觉到的一样。这就意味着电视系统端到端的响应应该是线性的（见图15-13a）。在电视系统中，摄像机处应用了γ校正，以对监视器的非线性进行预补偿（应用γ校正也是与最优化地对信号进行编码防止噪声有关）。如图15-13b所示，该图表明在摄像机处引入了γ校正以补偿监视器上的非线性。计算机图形系统（见图15-13c）与带有线性光强度特性的电视摄像机相似，因为绘制过程的计算是线性的。计算之后需要进行γ校正，而这通常是以查询表的形式实现的。



a) 理想情况下，一个观察者在电视显示器上看到的颜色应该与
他亲临场景所看到色彩相同



b) 在电视摄像机上应用了γ校正



c) 计算机图形学系统

图15-13 γ校正

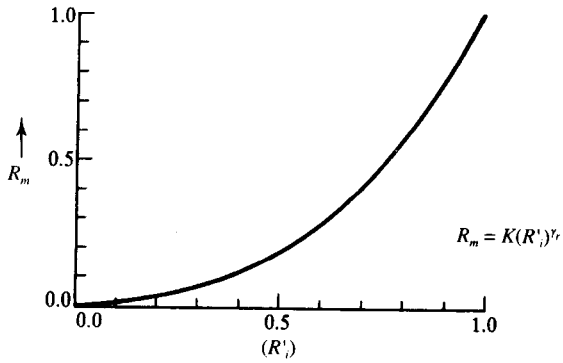
现在，让我们详细地讨论这个问题。例如，通过输入一个R_i值在监视器屏幕上产生的红色的光强度为：

440 $$R_s = K(R_i')^{\gamma}$$

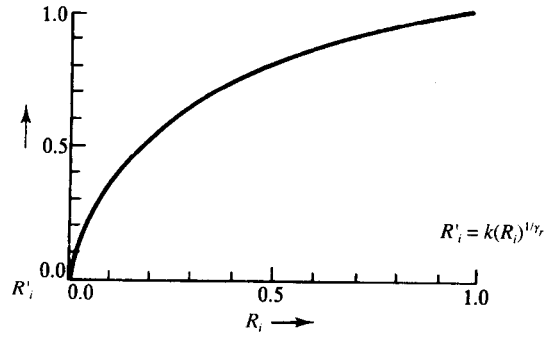
其中，γ_r情况的取值范围是2.3～2.8。这个过程的目的是使由程序产生的RGB值之间的关系线性化，如果γ_r、γ_g和γ_b已知，则可以将所谓的γ校正应用于将程序值R_i转换成一种值，当将该值插入到上述方程中时将导致一种线性关系。即：

$$R_i' = k(R_i)^{1/\gamma_r}$$

确定γ的一种代价不高的方法是由Cowan（1983）在一篇文章中提出的。这两种关系如图15-14所示。第二个图可以方便地结合到一个视频查询表中。注意，γ校正所需的成本在动态范围是缩减的。例如，如果所选择的k值使0映射为0、255映射为255，则可将256种光强度减少到167。这可能会引起带状，最好能以浮点进行校正运算然后再四舍五入。



a) 光强度为一个所用电压值的函数



b) 校正后的值是一个所用电压的函数

图15-14 γ校正

若采用了不正确的γ，监视器会导致光强度和色度都偏离由程序计算出的颜色值。例如，考虑三元组（0，255，127）。如果这个值不是经γ校正的，则显示将会减少蓝色的成分，而红色和绿色成分不变。

γ校正使零和最大的光强度保持不变，而改变两者中间的光强度值。如果因为没有采用校正或者在校正中使用了不正确的γ值，则相应于所计算的颜色来讲，错误的γ总是导致错误的图像。

441
~
442

第16章　基于图像的绘制和照片建模

16.1　以前绘制的图像的复用——二维技术
16.2　改变绘制的资源
16.3　运用深度信息
16.4　观察插值
16.5　四维技术——照明绘图或光线场绘制方法
16.6　照片建模和IBR

引言

作为一个新的有很多形形色色的方法的领域，基于图像的绘制（image-based rendering，IBR）是难于分类的，这个名字的起因是大多数技术都是基于二维图像的，但是情况并不总是如此，利用成像技术的方式也随方法的不同有很大的变化。一个更精确的、贯穿在所有方法中的主要思路是预计算。所有的方法在预先计算场景的表示时都会增加成本，运行时会从这些预先计算的表示中导出图像。对基于图像的绘制（IBR）的主要研究都是对于静态场景的普遍情况进行的，但是也建立起了对于动态场景的应用。

然而，关于IBR的目标并没有什么争议，这个目标就是使绘制所用的时间与场景的复杂性相分离，使得对于一个已知的帧时间限制，在像计算机游戏和虚拟现实这样的应用中图像的质量可以比用传统方法绘制的场景有所改进。因为在传统的绘制方法中无论任何时候改变观察点都必须向绘图流程中重新插入所有的几何运算。这种技术是与LOD方法（见第2章）以及场景的管理技术同时出现的，它被作为一种消除绘制时间对场景复杂性的依赖性的有效工具。

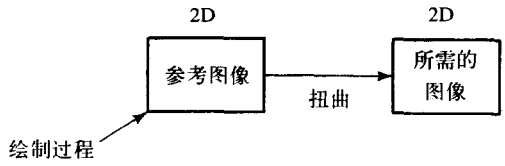
在这一章中我们还将讨论照片建模的问题。这与基于图像的绘制是有关联的，因为很多基于图像的绘制方法都是设计成处理照片建模的。照片建模的思想是捕捉真实世界的复杂性而同时又保留三维图形学灵活性的优点。

16.1　以前绘制的图像的复用——二维技术

我们从考虑依赖帧连贯性概念的方法以及以这种方式复用已经绘制的图像开始讨论。而且，正如本章标题所指出的那样，我们所考虑的技术基本上是二维的。当然，尽管基于图像的绘制的一般议题本身所指的是二维的技术，但是我们将在本章后面看到，还是有对于与图像相关的深度信息方面的利用的。其区别是对于所分类的二维技术，我们并不会对详细的深度值进行操作，例如每个像素一个值。我们可能仅仅有一个由在图像层上的可见性排序所指的与图像实体相关的深度值（见16.2.2节）。

基于图像的绘制程序的一个有用的模型是考虑所需的图像是由一个源或者一个参考图像通过在图像空间对参考图像进行扭曲产生的——参考图像是以正常方式绘制的（见图16-1）。在这一节中，我们将考虑简单的基于纹理映射的技术，这种纹理映射技术可以利用当前的3D图形卡的硬件功能。这里，新的方法是我们把在场景中绘制的物体看成纹理图，把一个纹理图

看成一个三维的实体，并将这个实体在绘图流程中传输。这类技术常应用于观察者在一个静态的环境中移动的系统中。



作为一个过程的基于图像的绘制，该过程
通过扭曲参考图像来产生一个图像

444

图16-1 平面“冒名顶替者”和图像扭曲

与用传统技术计算的投影相比较，这一类技术都在不同程度上涉及到某些近似。这种方法的一个重要部分是当它可以用来复用先前生成的图像以及当必须产生新图像时，它是确定的。

16.1.1 平面“冒名顶替者”或小画面

“冒名顶替者”是以纹理图的形式被利用的一个物体的图像，这是一个在第8章中被称为广告牌的实体。在第8章中，广告牌本身是一个物体，它是一个插入到场景中的二维实体。“冒名顶替者”是这一思想的概括。该思想是这样的，由于在一个移动的观察点序列中连续的帧的内在连贯性，在一定数量的帧上可以重复使用相同的小画面直到误差的量度超过了某个阈值为止。这类小画面有时用程序动画来限定，以便与那些不被更新的预先计算出的物体图像相区别。在一个常规的绘制引擎中，将一个平面的小画面用作纹理图。我们用程序化的平面来表明没有与小画面相关联的深度信息（尽管我们保留了含有小画面的长方形顶点处的深度信息）。在绘制程序中，法向（透视）纹理映射解决了随着观察点的改变而发生的小画面的扭曲。

将小画面结合到一个绘制的序列当中去有很多种可能性。Schaufler的方法（Schaufler and Sturzlinger 1996）便是典型的方法。为了从一个物体模型中产生一个小画面，该方法按如下步骤进行。将物体封闭在一个限定盒里，将这个限定盒投影到图像平面便确定了对于该特定观察来讲的屏幕空间中物体的矩形范围。所选择的小画面的平面垂直于观察平面的法向，并穿过限定盒的中心。在屏幕空间中的矩形范围被初始化为透明的，物体在其上进行绘制。然后，再将其作为纹理图处理，并放在纹理存储器中。当场景被绘制时，将物体作为一个透明的多边形进行处理，并进行纹理映射。请注意，纹理映射考虑到了当前的观察变化，因此小画面在帧与帧之间有轻微的扭曲，由透明的像素覆盖着的那些像素的值或其z深度不受影响。对于不透明的像素，将小画面处理为一个正常的多边形，并随着深度的变化对Z缓冲器更新。

在Maciel and Shirley（1995）的文章中指出，“依赖于观察的小画面”是经过预计算的，对于物体限定盒的每一个面计算一个小画面。然后物体周围的空间由平截体划分成观察点区域。这个平截体是由限定盒的表面及其中心形成的。如果选择了一个小画面作为一种适当的表示，则不管当前的观察点处于哪一个区域，这个区域即确定了所采用的小画面。

16.1.2 计算平面小画面的有效性

正如我们已经暗示的那样，小画面的使用需要一个计算误差的度量来对小画面的有效性进行定量。由于我们没有使用深度信息因而使小画面失效。从产生小画面的观点之外的其他观点来看，小画面正是我们所认为的那样——一个平面的图像被嵌入到三维空间中，错觉被破坏了。 445

误差的大小依赖于由小画面所表示的场景的区域中深度的变化、该区域与观察点之间的距离以及观察点从绘制小画面时的参考位置的移动（距离因素可以通过对远处的物体采用低分辨率小画面以及由多个物体组成簇等方法来使用）。对于改变观察点的应用，必须对有效性进行动态的估算并在需要时产生新的小画面。

Shade等（1996）采用了一种基于角度差的简单度量。图16-2所示为一个物体限定盒的二维观察，其中小画面平面以粗线表示。v_0是小画面绘制时的观察点，v_1是当前的观察点，x是小画面视图上的一个点或者是与x'共线的物体的顶点。只要观察点离开v_0，x和x'便形成一个夹角θ，Shade等计算了一个误差的度量，它是在所有x点上的最大角。

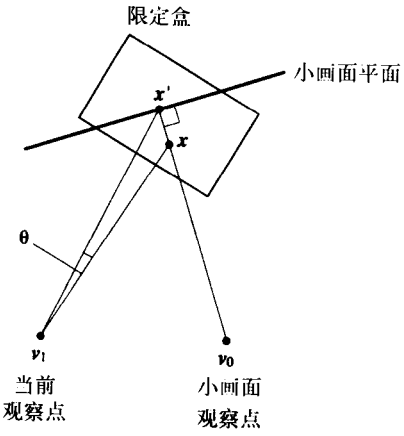


图16-2 一个小画面图像的角度差

Schaufler and Sturzlinger（1996）的误差度量是基于与像素尺寸相关的角度差以及对两种最坏情况的考虑。第一种情况，考虑由于观察点平行于小画面平面的平移而产生的角度差（见图16-3a）。当观察点与包围限定体的一个立方体中某个对角线相垂直，而这个限定体的小平面又与其他对角线共线时，这个角度差最大。当观察点向v_1移动时，点x'、x_1和x_2应被看成独立的点。由于观察点的运动这个分量的作用而导致的角度差则由向量v_1x_1和向量v_1x_2之间的夹角θ_{trans}给出。只要这个值小于观察点处一个像素所包含的角度则这个误差就是可以容忍的。对于一个向物体移动的观察点，我们可以考虑图16-3b中的结构。在这里，最坏的情况是在立方体的前表面的角落处。当观察点移到v_1时，x_1和x_2点将可以被分别看到，其角度差由θ_{size}给出。于是可以将一个小画面用作:

$$\text{use\_imposter} := (\theta_{trans} < \theta_{screen}) \text{ 或 } (\theta_{size} < \theta_{screen})$$

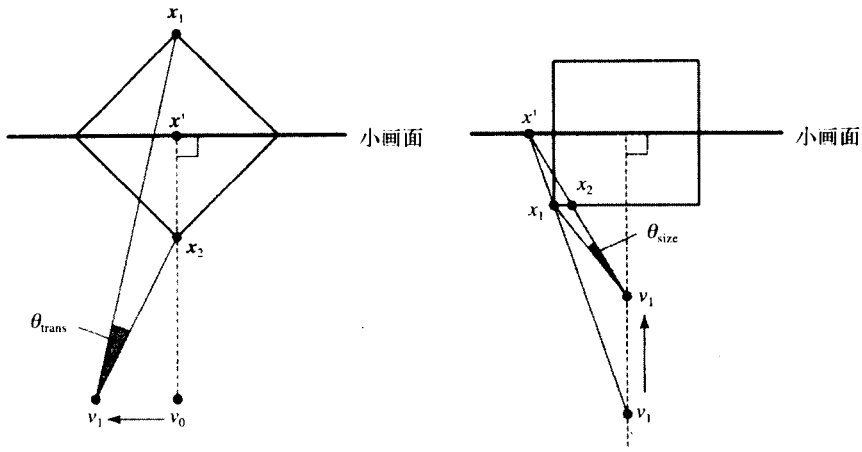
其中

$$\theta_{screen} = \frac{\text{观察域}}{\text{屏幕分辨率}}$$

使用小画面的最简单方法是将其作为纹理图结合到在正常的利用纹理映射硬件的绘制方案中。

到目前为止并没有提及由什么构成一个小画面，我们一直假设由一个物体模型产生一个图像。Shade等（1996）在称为层次化的图像存储方法中对这一概念进行了泛化，由场景的BSP树中结点的全部内容产生小画面，并将这种功能强大的场景划分方法的优点与预先绘制的图像的运用相结合。于是，对于不需要经常更新的远距离的物体可以组合成簇，并为这个簇产生一个小画面。因此，这个算法得到应用并且利用场景表示的层次结构。物体可以被分

解到不同的叶结点上，这就造成了一个物体具有多个小画面的情况，于是引起视觉上的缺陷。Shade等（1996）通过确保BSP划分策略尽可能少地分解物体以及在叶结点区域对几何形状进行稍微的“膨胀”来使这种缺陷最小化，从而使得小画面互相重叠以消除在其他情况下可能出现的最终图像上的缝隙。



a) 平行于一个小画面的观察点的平移　　b) 观察点向一个小画面平面的平移

图16-3 Schaufler的最坏情况下角度差的度量（Schaufler（1996））

16.2 改变绘制的资源

16.2.1 优先绘制

446
~
447

与2D成像技术相联系的一直被采用的一个重要技术是对于图像的不同部位分配不同的绘制资源。一个有影响的（硬件）方法是由Regan and Pose（1994）提出来的。他们把不同的帧速率作为物体与观察点间距离的一个函数分配给场景中的物体，这称为优先绘制，因为它把环境映射方法与以不同的速率对场景进行更新结合了起来。他们使用一个有六个观察面的立方体的环境图作为基本的预计算的解。此外，对于图像合成采用了一个多显示存储器，而对于场景的改变则与预先绘制的图像相结合。

这个方法是基于图像的方法与传统的绘图流程方法的一种混合物。它的运行依赖于把场景划分成一个优先性的层次结构。根据与当前观察者的靠近程度以及它们对绘制资源的分配，分配给物体优先权，并相应地确定更新时间。场景是作为环境图预先绘制好的，如果观察者保持其位置不变，则不必对环境图进行改动。随着观察者改变其位置，则根据优先权方案从新的观察点上绘制新的环境图。

Regan and Pose（1994）采用了多显示存储器来实现优先绘制，根据它所包含的信息以不同的速率对每一个显示存储器进行更新。如果一个存储器含有用户正在处理的那个场景的部分，则必须对其进行更新，而一个含有远离当前用户位置信息的存储器则可以保持不变。因此，场景中各个部分都以不同的速率进行更新，所以将其称为优先绘制。Regan and Pose（1994）使用存储器以60帧/秒、30帧/秒、15帧/秒、7.5帧/秒以及3.75帧/秒的速率进行运算。绘制的重点被导向那些最需要它的场景的部分。在任意时刻，场景中的物体都将根据其当前

与用户的距离被组织到显示存储器中。为了简化起见，对存储器的占用可以按从用户当前位置发散出去的同心圆来安排。动态地对每一个物体分配一个适当的显示存储器涉及到一种计算，该计算是相应于限定球进行的。最后，这个因素还必须对场景的复杂度有一个上限的限制。Regan and Pose（1994）报告了一个只有1000个物体的测试场景的测试实验。另一种选择是物体必须分层地进行分组，并通过一个二次数据结构来进行处理，就像在某些加速策略中对于传统的光线跟踪方法所做的那样。

16.2.2 图像分层

Lengyel and Snyder（1997）对小画面的概念以及绘制资源的各种应用进行了归纳，他们把这个技术称为“连续的图像层”。在这里，其思想是把绘制资源根据以不同的空间和/或临时的采样速率表示的需求分配到图像的不同部位。这个技术也可处理互相之间运动的物体，它是通过把图像划分成层次来进行的（当然，这是一个旧的想法，早在20世纪30年代卡通片的生产就一直通过把图像划分成层次来进行优化。这些层次独立地工作，并被合成到最终的胶片上）。因而快速移动的前景物体可以比慢速移动的背景物体分配到更多的资源。 448

Lengyel和Snyder的方法的另一个关键思想是任何一个层次其本身也可以被分解成一些分量。分层的方法被结合到自身的明暗处理方法中，不同的资源分配给了明暗处理中的不同分量。一个移动的物体由一个散射层、一个高亮层以及一个阴影层组成。每一个分量都产生一个图像流和一个代表其变换以及其在图像空间中的扭曲的二维变换流。小画面可以按不同的分辨率来表示，并可以以不同的速率进行更新。于是，小画面在空间和时间上都有了不同的分辨率。

在这种情况下，小画面现在是一个“独立的”实体，而不是一个由法向顶点/纹理坐标关联联系到物体上的纹理图。它还是一个纯二维物体，而不是一个三维物体的二维部分（一个纹理图）。因而随着一个小画面的移动，必须计算适当的扭曲。

事实上，传统的绘制流程被分解成了“平行”的片段，每一个片段代表图像的不同部分（见图16-4）。对于每一个层次可以设定不同的质量，每一层次都以不同的帧速率和不同的分辨率出现。然后将这些层次按照深度的顺序以透明度或α值在合成器中合成。

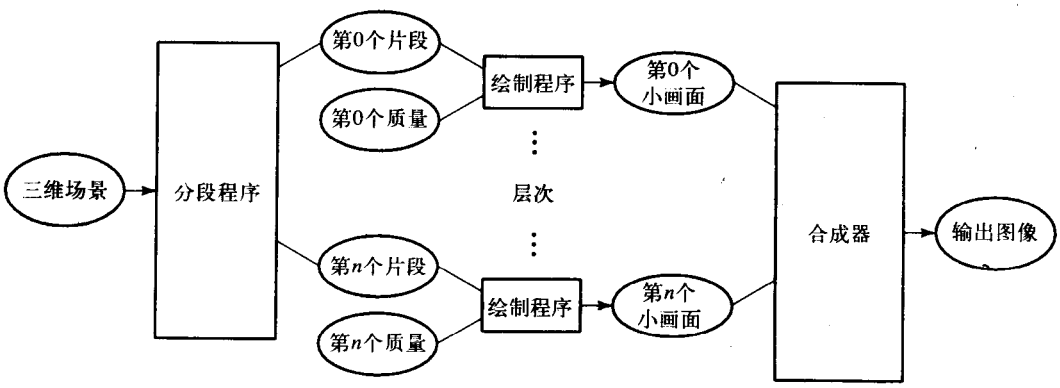


图16-4 Lengyel和Snyder的层次化方法（Lengyel and Snyder(1997)）绘制资源被分配给感觉上重要的那些场景中的部分（层次）。以较低的帧速率和较低的分辨率对缓慢改变的层次进行更新

通过建立一个绘制变换A将一个小画面创建成一个矩形的实体，使得在小画面域中物体的投影与一个限定盒紧密配合。这样做的目的是要使小画面中的点不会对非物体空间进行采样。变换A是一个仿射变换，它把小画面映射到屏幕上。该变换按如下方式定义。如果我们考虑屏幕空间的一个点p_s，则有：

$$p_s = Tp$$

这里p为世界空间中相等的物体点，T是此建模、观察以及投影变换之间的连接关系。

我们需要一个A，使得（见图16-5）：

449

$$p_s = A^{-1}ATp = Aq$$

其中q是小画面坐标系中的一个点：

$$A = \begin{bmatrix} a & b & t_x \\ c & d & t_y \end{bmatrix}$$

于是，用一个仿射变换得到一个与由一般的变换T而产生的扭曲等价的扭曲。

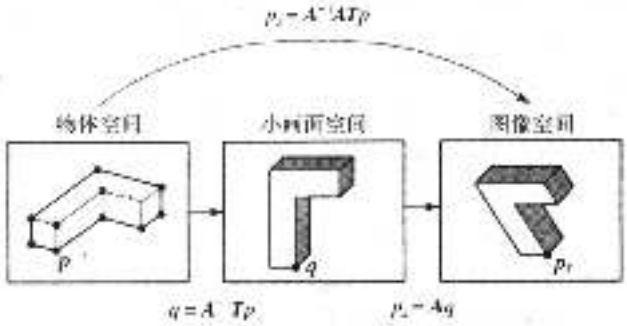


图16-5 小画面绘制变换A

变换A是一个2×3的矩阵，这个矩阵随着物体的刚性移动而被更新，并且提供由于物体的移动而在屏幕空间改变小画面形状所需的扭曲。这通过对两个连续的时间间隔T_{t-1}和T_t，将代表物体的一个特征多面体（见图16-6）上的点变换到屏幕空间，并求出A的六个未知的系数来实现。这一过程的详细介绍在Lengyel and Snyder（1997）中给出。

1. 计算层次的有效性

像任何序列过程的进行一样，需要监视层次的可复用性。在16.1.2节中我们描述了对于小画面有效性的一个简单的几何度量。利用图像的层次，Lengyel and Snyder（1997）建立了基于几何学、光度学和采样理论的更详尽的判据。几何学和光度学实验测量由于分层或小画面以及如果按传统方法绘制而产生的图像之间的差别。

几何误差的度量由下式计算（Lengyel和Snyder将其称为度量基准）：

$$F_{Geometric} = \max_i \| p_i - Ap'_i \|$$

其中Ap'_i是当前帧中某层上的特征点，当前帧是由前一帧扭曲形成的，p_i是点实际所在的位置（这些特征点总是由T以及建模、观察和透视变换进行变换的，目的是要计算扭曲。这看起来好像是一个循环的问题，但是求A（前一节）包含一个最佳匹配过程，请记住，扭曲是用来对变

换T进行近似的)。因此，可以设置一个阈值，如果超出阈值的话，就要再次绘制所考虑的层。

450

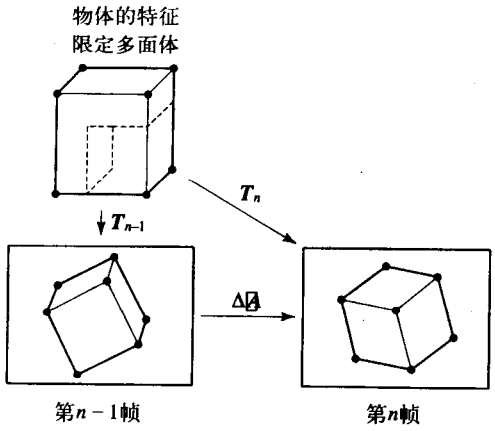


图16-6 场景空间中限定多面体内点的刚性移动所产生的效果被表示成仿射变换A中的变化（Lengyel and Snyder(1997)）

对于由于光源与由层次所表示的物体之间的相对运动而产生的变化，以L表示的角度的变化以及物体的光线方向都是可以计算的。

最后，必须计算一个与层次的放大/缩小相关联的度量。如果观察者和物体之间的相对运动是层次的样本被膨胀或压缩，则该层次应该进行重新绘制。这一操作类似于确定映射中的深度参数。在这种情况下，可以由仿射变换中2×2的子矩阵进行计算。

在完成了一个帧之后，用调节器对下一帧进行资源分配。这可以基于一种“填单”的形式来进行，这时场景的质量被最大化，或者也可以基于一个阈值。这时误差的阈值被设定为用户可以容忍的最高等级（释放绘制资源以作他用）。这种分配是通过估算误差判据并以每一个特定层上所消耗的绘制成本分率为基础估算每一层的绘制成本。然后，可以将层次按照效益/成本的顺序进行排序，并由调节器进行重新绘制或扭曲。

2. 按深度对层次排序

到目前为止，还没有论及层次的深度问题。合成器需要深度信息以便能够从各个分离的层次产生一个最终的图像。由于方法是被设计为处理移动物体的，所以层次的深度顺序可以改变，并且这个方法就是要保持被动态更新的层次的一个经排序的列表。绘制程序在一个层次内部产生隐藏面已经消除了的图像，一个特殊的算法将层次的相对可见性处理成不可见的实体。使用了一种Kd树连同凸的限定层次的几何形状的多面体，为了在不进行分离的情况下处理碰撞，采用了一种增量算法（Snyder（1998）对此有完整的描述）。

451

16.3 运用深度信息

16.3.1 三维扭曲

正如前面已经提到的，平面小画面的主要缺点是它不能产生运动视差。它所产生的是一种由一个阈值所限制的扭曲，超过这个阈值其平面的性质就可以被感觉出来。

现在来考虑对深度信息的利用。当然，这些信息是合成图像中容易得到的信息。尽管现

在许多技术都开始使用三维信息，但是仍然把它们看作是基于图像的技术。尽管增加了深度信息我们仍然要利用已经绘制的图像作为源或参考。

首先，考虑要选用何种深度信息。按照所需存储的顺序排列，三种最常见的形式为：使用带有深度信息的层次或小画面（前一节）的形式、用一个完整的（未经分段的）带有关联的Z缓冲器的图像（换句话说，是每个像素一个深度值）以及一个分层的深度图像（LDI）。LDI是场景的一个观察，它在每一条视线上有多个像素。LDI所需的存储量是投影到一个像素上平均数量表面的深度复杂度的一个函数。

我们从考虑带有每个像素的深度信息的图像开始——这是按常规方法合成的图像的正常状态。从直觉上来看，很显然，应该能够在一个新的观察点上从参考图像产生或外推出一个图像，假如新的观察点靠近参考观察点的话。可以把图像空间中像素的运动定义为扭曲：

$$I(x,y) \rightarrow I'(x',y')$$

也就是说，一个参考像素将移动到一个新的目的地（这是对于问题的一种简单说明，它忽略了我们将在后面讨论的重要的实际问题）。如果假设观察点的变化由旋转$\boldsymbol{R} = [r_{ij}]$接着进行观察坐标系的变换$\boldsymbol{T} = (\Delta x, \Delta y, \Delta z)^{\mathrm{T}}$（在世界坐标空间）来定义，观察系统/摄像机的内部参数不变，将焦点的长度设为1，于是，扭曲由下式定义：

$$x' = \frac{(r_{11}x + r_{12}y + r_{13})Z(x,y) + \Delta x}{(r_{31}x + r_{32}y + r_{33})Z(x,y) + \Delta z}$$
$$y' = \frac{(r_{21}x + r_{22}y + r_{23})Z(x,y) + \Delta y}{(r_{31}x + r_{32}y + r_{33})Z(x,y) + \Delta z} \tag{16-1}$$

其中：

452

$Z(x,y)$是点$\boldsymbol{P}$的深度，点$\boldsymbol{P}$的坐标(x,y)是投影。

这从以下公式得出：

$$x' = \frac{x_{v'}}{z_{v'}} \quad y' = \frac{y_{y'}}{z_{v'}}$$

其中，$(x_{v'}, y_{v'}, z_{v'})$为$\boldsymbol{P}$点在新的观察系中的坐标。这个过程如图16-7所示。

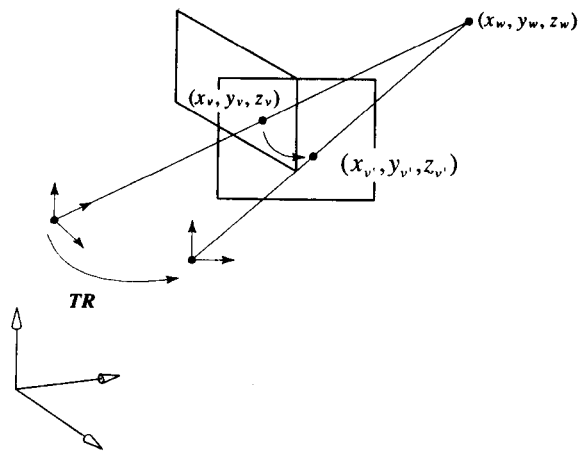


图16-7 通过将旋转$\boldsymbol{R}$和变换$\boldsymbol{T}$应用于观察坐标系计算一个三维的扭曲

现在来考虑该过程将出现的问题。第一个问题被称为图像打摺或者拓扑褶皱，当将参考图像中的多个像素映射到外推图像中的位置（x', y'）时出现这种情况（见图16-8a）。解决这个问题的直接方法是由$Z(x, y)$ 计算$Z(x', y')$。但这需要一个附加的有理表达式，并需要另外一个Z缓冲器来存储结果。

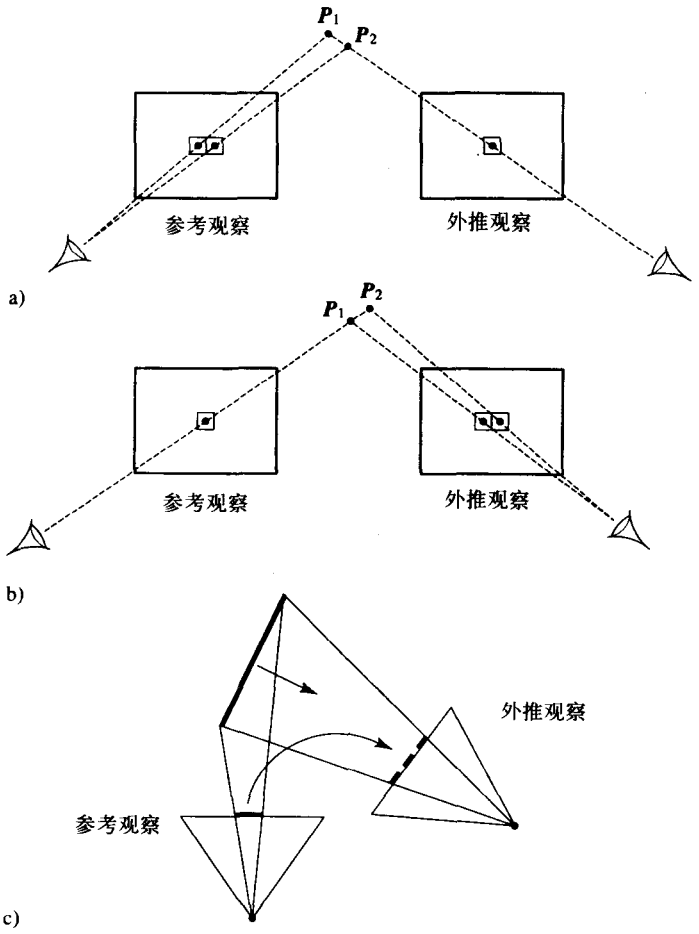


图16-8 图像扭曲中出现的问题

a) 图像打摺：在参考观察中多个像素映射到外推观察中的一个像素上
b) 孔洞：在参考观察中被挡住的信息在外推观察中是需要的
c) 孔洞：因为其朝观察方向的正常旋转而使得一个表面的投影区域在外推观察中增加了

McMillan（1995）建立了一种算法，为计算扭曲函数定义了一个唯一的估算顺序，使得表面以一种从后向前的顺序进行绘制。这就使得用一个简单的画家算法就能解这类可视性问题。对于这种算法直觉的判断可以通过参考如图16-9所示的一种简单的特殊情况得到。在这种情况下，观察点被向左移动了，以便使其在参考观察坐标系中图像平面中的投影位于范围之外，并一直延伸到参考观察窗口的左侧。可见，在参考系中需要访问的像素的顺序是从右向左的。这样，就解决了在参考图像中最左边的像素覆盖经扭曲的图像中的右侧像素的问题。McMillan表明，参考图像的访问或枚举顺序可以依据参考坐标系中新的观察点投影的位置减

为九种情况。这些情况如图16-10所示。一般情况下，新的观察点处于参考观察窗口之内，把图像分成了四份。于是，采用这种方法解决多对一情况的深度问题的一种算法结构为:

1) 计算参考坐标系中新观察点的投影。

2) 根据投影点确定枚举的顺序（九种情况中的一种如图16-10所示）。

3) 应用方程（16-1）扭曲参考图像，并把结果写入到帧缓冲器中。

当参考图像中被挡住的区域“需要”在外推图像中可见时会出现图像扭曲产生的第二个问题（见图16-8b），这将在外推图像中产生孔洞。正如图中所示，孔洞和褶皱之间是相互反向的。但是，对于褶皱有确定性的解决方法而对于孔洞却没有理论的答案，所以需要采纳一些经验知识。乍看起来我们不能恢复本来就不存在的信息。然而，要检测孔洞出现在哪里是容易的，它们只是在外推图像中未赋值的像素，这使得问题被局部化，其最常见的解决方法是利用其邻近像素的颜色来填充它们。孔洞问题的范围依赖于参考观察点与外推观察点之间的差别。可以通过考虑一个以上的参考图像、对每一个图像计算一个外推的图像然后将结果合成使得这种现象得到改善。显然，如果使用了足够数量的参考图像，则可以消除孔洞问题，而且不需要得到一个局部解，因为它可能会插入错误的信息。

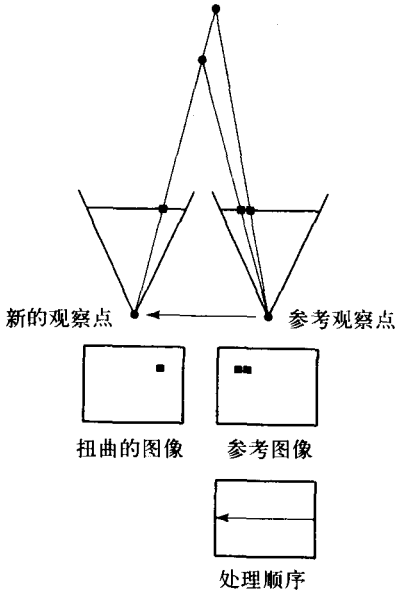


图16-9 将观察点变换到左侧，以便使参考观察坐标系的图像平面中新观察点的投影在参考观察窗口的左侧。参考像素正确的处理顺序是从右向左

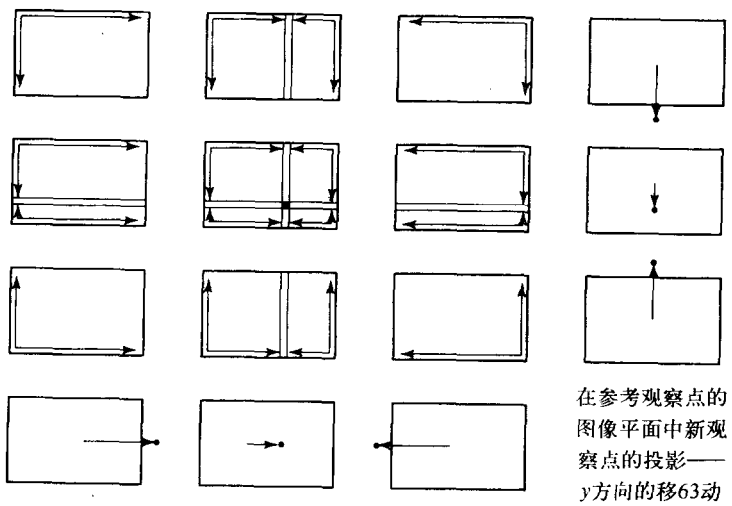


图16-10 McMillan的优先算法的一种可视化，表明正确的处理顺序是对于九种情况的观察点运动的一个函数

在外推图像中出现未被赋值的像素的一个更微妙的原因是很明显的，如果考虑那些朝向新观察系统的观察方向正交旋转的表面的话（见图16-8c）。外推图像平面这样一个表面的投影区域将比其在参考图像平面上的投影要大，而且对于一个一对一的前向映射将产生孔洞。这就提醒我们要在插值图像上采取精确的方法进行重建。Mark等（1997）提议为每一个参考像素计算重建内核的合适尺寸，并将其作为一个观察点运动的函数。但是他们也指出，这样做也导致每个像素的显示成本比扭曲的成本要大。这个度量通常称为溅泼尺寸（见第13章），该值的计算对于带有仅用于可见像素的Z深度的参考图像来说并不是直截了当的。一种对一个像素存储多个深度值的方法将在下一节中进行介绍。

图像上这些问题的效果如图16-8d（彩色插图）所示。前两个图展示了一个简单的场景及其相应的Z缓冲器图像。下一个图像显示由于平移而产生的缺陷（仅仅如此）。在这种情况下，这些缺陷是由丢失信息而产生的孔洞以及图像的褶皱。接下来的图像显示由于旋转而产生的缺陷（仅仅如此）——这些孔洞是由于增加表面的投影区域而引起的。注意它们是如何形成连贯的模式的。最后一个图像显示由旋转和平移而产生的缺陷。

最后应提起注意的是，依赖于观察的照明效果用这种简单的方法一般来讲是不能被正确处理的。这在基于图像的建模方法中仍然是一个非常重要的问题（第16.6节）。正如在图像扭曲中已经指出的那样，我们必须有其观察点靠近所需观察点的参考图像。

16.3.2 分层深度图像

在前一节中遇到的许多问题在源图像是以LDI形式出现时就消失了（Shade等 1998）。尤其是我们可以解决孔洞问题，这时需要有关在源或参考图像中被挡住区域上的外推图像的信息。LDI是一个二维的数据结构，该结构与特定的观察点有关，而且它对于每个像素在所有的表面上采样，其深度值是与穿过该像素的光束相交的值（见图16-11）（在实际应用中，需要一些LDI来代表一个场景，所以可以把一个场景的表示看成是四维的——或者与在第16.5节中的光线场具有相同的维数）。因而，每一个像素都与一个信息矩阵相联系。该信息矩阵中的元素或层次是由一些表面相交而确定的。每个元素含有表面的颜色、表面法向和深度等信息，很明显，这种表示所需的存储空间比一个图像加上Z缓冲器需要的存储空间要多得多，但该需求随着深度复杂性的增加只是线性地增长。

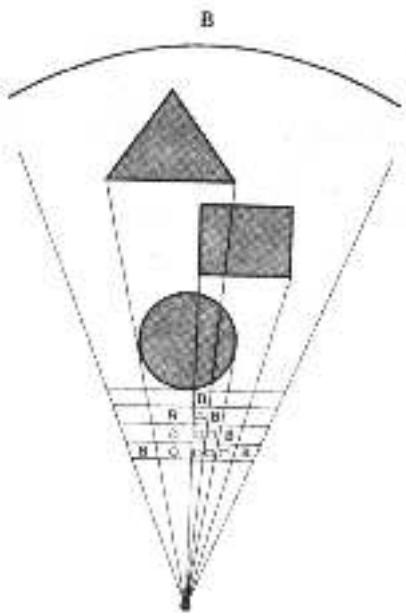


图16-11 分层深度图像（LDI）的一种表示

在Shade等（1998）中提出了两个为合成图像预计算LDI的方法。第一种方法，他们建议把n个由不同观察点绘制的图像扭曲到一个观察点上。在扭曲过程中，如果有多个像素映射到一个LDI像素上，则对与每个源观察相关联的深度值进行比较，使得层次可以按照深度的顺序进行排序。

另一种便于更严格地对场景进行采样的方法是使用一种改进的光线跟踪程序。其简要的过程是从LDI观察点为每一个像素启动一条光线，并允许该光线穿透物体（而不是被反射或折射）。每一次碰撞都作为LDI中的一个新的深度像素存储下来。所有场景都可以通过预计算六个LDI进行考虑，而每一个LDI都由一个以参考观察点为中心的90°的平截体构成。Shade等（1998）指出这种采样方法对于以观察点为中心的方向半球来讲并不是均匀的。相邻像素的光线以图像平面法向与光线方向之间的夹角的一个函数为依据将一个小的区域投影到图像平面上，并以该角度的余弦值作为光线方向的权重。因此，每一束光线都有四个坐标：两个像素坐标和两个光线方向的角度。计算LDI的算法结构如下：

1) 对于每一个像素，修改方向并向场景中投射光线。

2) 对于每一个碰撞，如果相交的物体位于LDI平截体之内，则它通过LDI观察点被重新投影。

3) 如果新的碰撞处于一个现有的深度像素的允许范围之内，则新样本的颜色与现有样本的颜色进行平均，否则建立一个新的深度像素。

在绘制阶段，以从后向前的顺序对每一层次应用一个增量扭曲，图像被全部融合到帧缓
457 冲器中而不必进行Z排序。McMillan的算法（见第16.3.1节）用于确保根据LDI系统中输出摄像机的投影以正确的顺序选择用于扭曲的像素。

为了进行溅泼尺寸计算，Shade等（1998）采用了下面的公式（见图16-12）：

$$\text{size} = \frac{d_1^2 \cos\theta_2 \ \text{res}_2 \tan\dfrac{\text{fov}_1}{2}}{d_2^2 \cos\theta_1 \ \text{res}_1 \tan\dfrac{\text{fov}_2}{2}}$$

其中：

size为一个正方形内核的尺寸（实际应用中，这个值取整为1，3，5或7）；

角度θ近似为角度ϕ，ϕ是表面法线与摄像机系统的z轴之间的夹角；

fov为一个摄像机观察的场；

res = $w*h$（LDI的宽度和高度）。

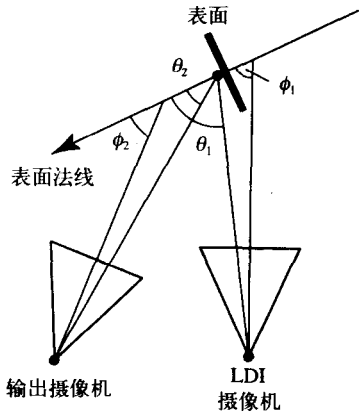


图16-12 溅泼尺寸计算中所使用的参数（Shade等（1998））

16.4 观察插值

观察插值技术可以被认为是3D扭曲方法的一个子集。代之以从一个参考图像外推一个图像，它们对一对参考图像进行插值。但是，为了进行这种插值需要进行三维计算。从前面对二维/三维分类的观点来看，它们可以被认为是二维的技术。但是，我们在这里强调其插值特性，并把它们另归一类。

Williams and Chen（1993）首先为一个预排演的应用实施了观察插值，它是通过预先计算一组代表一个内部的参考图像来实现的。该例中是一个虚拟的博物馆。在预排演中所需的帧在运行时从这些参考帧中进行插值。通过存储一个定义参考帧之间的像素移动的“扭曲脚本”来实现插值。这是一个运动向量的密集集合，它将源图像中的像素与目的图像中的像 458
素相关联。一个运动场的最简单的例子是一个摄像机平行地平移到它的图像平面而产生的场。在这个例子中，运动场是一组平行的向量，每一个向量代表一个像素，向量的方向与摄像机运动的方向相反，其大小正比于像素的深度。可以为每一对图像确定这种逐个像素间的对应关系，因为每一个像素的三维（图像空间）坐标是已知的，正如摄像机或观察点的运动是已知的一样。扭曲脚本的确定是一个预处理的步骤，其内部最终由一组参考图像和一个与每一个相邻对相关联的扭曲脚本来表示。对于一个需要一些变化的预排演的大场景，其总的存储需求可能非常大，然而，任何导出的或插值的观察仅仅需要一对适当的参考图像和扭曲脚本。

在运行时，两个参考图像之间的一个观察或者一组观察变为线性插值。在源图像和目的图像中的每一个像素通过线性地插值图像坐标所给出的数量沿着其运动向量的方向进行移动（见图16-13），这样就给出了一对插值图像。可以对这些插值图像进行合成，而按照这种方法利用一对图像就可以减少孔洞问题。Chen and Williams（1993）用一种过程填充了其余的孔洞，该过程利用了局限于孔洞的颜色。通过采用一个Z缓冲器来确定最近的表面解决了重叠的问题，其z值被沿着(x, y)坐标进行了线性插值。最后，注意到运动向量的线性插值产生了

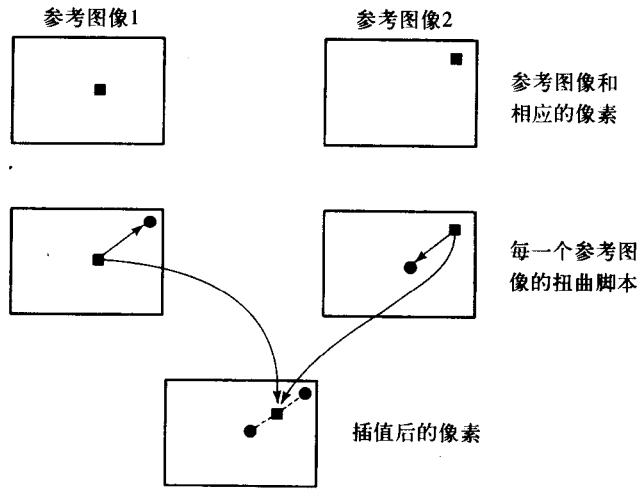


图16-13 简单的观察插值：一对对应的像素定义了图像空间的一个路径，可以从这个路径构建一个插值的观察

一种扭曲，它可能与将摄像机移动到希望的位置所产生的扭曲不完全相同。这个方法只有在将摄像机平行地平移到它的图像平面这种特殊的情况下才产生完全相同的结果。Willams and
459 Chen（1993）指出，可以由图像平面中的二次或三次插值获得一种较好的近似。

观察变形

到现在为止，我们考察了那些处理一个运动的观察点和静态场景的技术。在一种被称为观察变形的开发中，Seitz and Dyer（1996）提出了产生没有出现刚性变形的中间图像的问题。他们通过强调前面章节中提到的近似的含义并对“有效的”和“无效的”中间观察进行区分来实现这一过程。

将一个参考图像扭曲成一个外推图像的观察插值是在二维的图像平面空间进行的。扭曲操作也就是指它改变了物体的二维投影的形状。很明显，应该继续进行插值，以便使得参考投影中物体的投影形状与它们实际的三维形状相符合。换句话说，插值出来的观察应该等价于以正常方式（也可以用一个摄像机或者用一个传统的绘图流程）通过将观察点从参考观察点改变到被插值的观察点而产生的观察。一个“无效的”观察是指所插值的观察没有保持物体的形状。如果没有保持这种状况，则插值得到的观察将对应于一个在实际的三维空间中其形状已经扭曲了的物体。在两种形状之间传统的图像变形正是这种情况。“不可能的”、不存在的或者是随意的形状以中间图像的形式出现，是因为这里的运动就是将物体变成一个完全不同的物体。有效的与无效的观察插值之间的区别如图16-14所示。

有一个对图像进行线性插值产生有效的插值观察的例子。该例子中图像平面保持平行（见图16-15）。实际上，如果允许一个摄像机平行于其图像平面进行移动（并有选择地放大和缩小）则会出现这种情况。如果令对于两个参考图像的组合观察及透视变换（参见第5章）为$\boldsymbol{V}_0$和$\boldsymbol{V}_1$，则对于一个中间图像的变换可以通过线性插值来获得:

$$\boldsymbol{V}_i = (1-s)\boldsymbol{V}_0 + s\boldsymbol{V}_1$$

如果考虑参考图像中的一对对应的点$\boldsymbol{P}_0$和$\boldsymbol{P}_1$（它们是世界坐标空间中点$\boldsymbol{P}$的投影），则容易看出（见Seitz and Dyer（1996））点$\boldsymbol{P}$从中间（插值的）观察点的投影是由线性插值给出的:

$$\begin{aligned}\boldsymbol{P}_i &= \boldsymbol{P}_0(1-s) + \boldsymbol{P}_1 s \\ &= \boldsymbol{V}_i \boldsymbol{P}\end{aligned}$$

换句话说，假如保持了平行观察的话，在一条由两个参考图像中对应像素确定的路径上像素的线性插值正好等价于投影场景中的点，使这些像素经过一个由中间摄像机的位置确定
460 的观察和投影变换。也就是说，如果$\boldsymbol{V}_0$和$\boldsymbol{V}_1$是线性插值的，则当可以获得时采用变换$\boldsymbol{V}_i$。还应注意到我们正在插值的观察可能会对应于以一条从$\boldsymbol{C}_0$到$\boldsymbol{C}_1$的直线移动摄像机所获得的那些观察。换言之，插值的观察对应于摄像机的位置:

$$\boldsymbol{C}_i = (sC_x, sC_y, 0)$$

如果参考观察不是以这种方式相关联的，则插值之前（后）必须进行附加的变换。当参考观察的图像平面以及所需的或插值的观察没有平行关系时，通常有这种情况出现。Seitz和Dyer将其称为“预扭曲”的第一个变换对参考图像进行扭曲，以使其看起来像是由一个摄像机在一个平行于图像平面的平面中移动时所获得的。像素的插值或变形也可以像前面所

述的那样进行，其结果被后扭曲以形成最终的插值观察。这就是从虚拟摄像机的位置所需的观察。

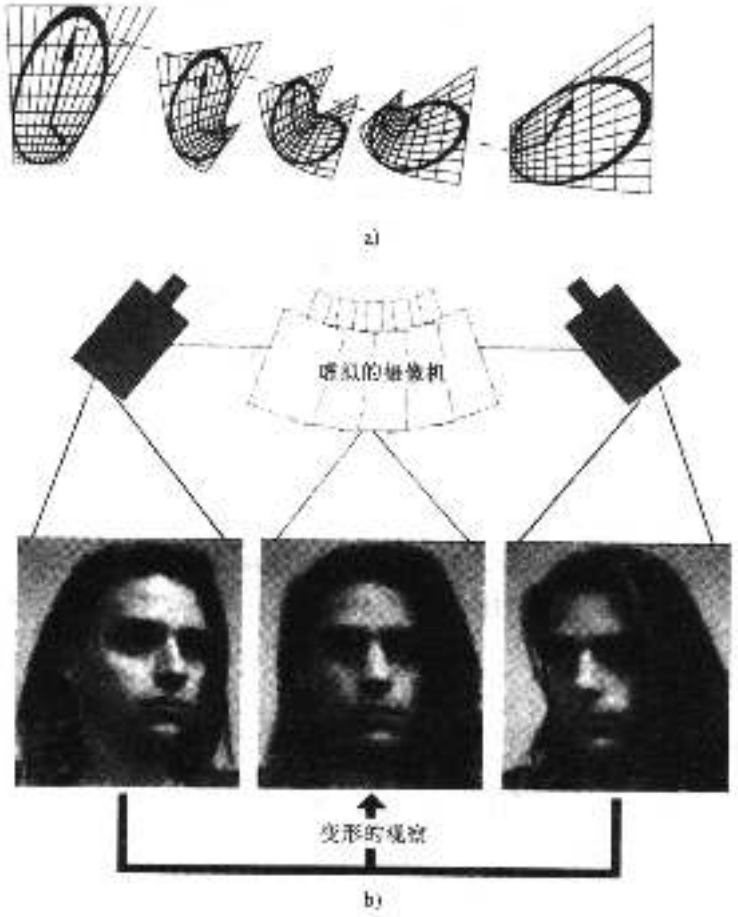


图16-14 有效的和无效的观察插值之间的区别。在a中，用一个线性插值的标准（变形）方法产生粗略的形状变形（如果在两个不同的物体之间进行变形则这种情况无关紧要——它成为效果的组成部分）。在b中，插值的（或变形的）观察与物体的形状相符（经Steven Seitz许可）

这个过程的一个简单的几何示意图如图16-16所示。图中$\boldsymbol{R}_0$和$\boldsymbol{R}_1$为参考图像。分别将它们预扭曲成$\boldsymbol{R}_0'$和$\boldsymbol{R}_1'$指的是此时可以对这些校正的图像线性插值成$\boldsymbol{R}_t'$。然后对这个图像进行后扭曲产生所需的$\boldsymbol{R}_t$。这个方法的一个重要结果是尽管扭曲操作是基于图像的，但是我们还是需要有关影响预扭曲和后扭曲变换的观察点的知识。这样就再次出现对于不同方法的使用在内容上的分歧。这意味着在照片成像的情况下必须记录或恢复摄像机的观察点。

预扭曲和后扭曲变换的导出如下。首先，可以看出，任何两个共享相同投影中心的透视观察都由一个平面投影变换进行关联，这是一个从组合的观察透视变换V得到的3×3的矩阵。于是，$\boldsymbol{R}_0$和$\boldsymbol{R}_1$就由两个这样的矩阵$\boldsymbol{T}_0$和$\boldsymbol{T}_1$关联成$\boldsymbol{R}_0$和$\boldsymbol{R}_1'$。其过程如下：

1) 用$\boldsymbol{T}_0^{-1}$和$\boldsymbol{T}_1^{-1}$预扭曲$\boldsymbol{R}_0$和$\boldsymbol{R}_1$产生$\boldsymbol{R}_0'$和$\boldsymbol{R}_1'$。

2) 插值计算R_i'、C_i和T_i。

3) 将T_i用于R_i'得到图像R_i。

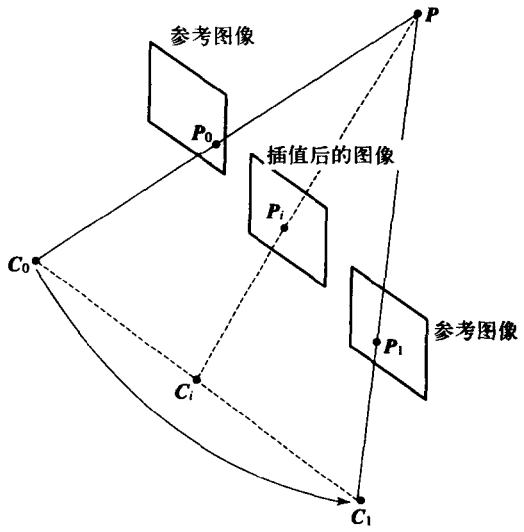


图16-15 从C_0到C_1移动摄像机（并缩放）意味着图像平面保持平行，可以由P_0和P_1线性地插值得到P_i（Seitz and Dyer（1996））

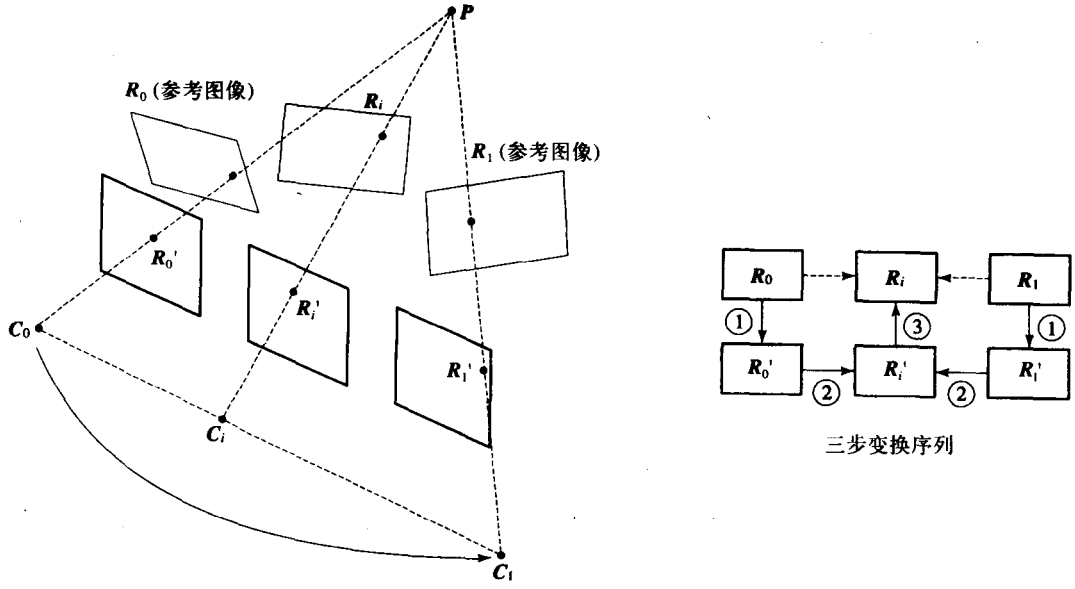


图16-16 观察插值中的预扭曲参考图像、插值和后扭曲（Seitz and Dyer(1996)）

16.5 四维技术——照明绘图或光线场绘制方法

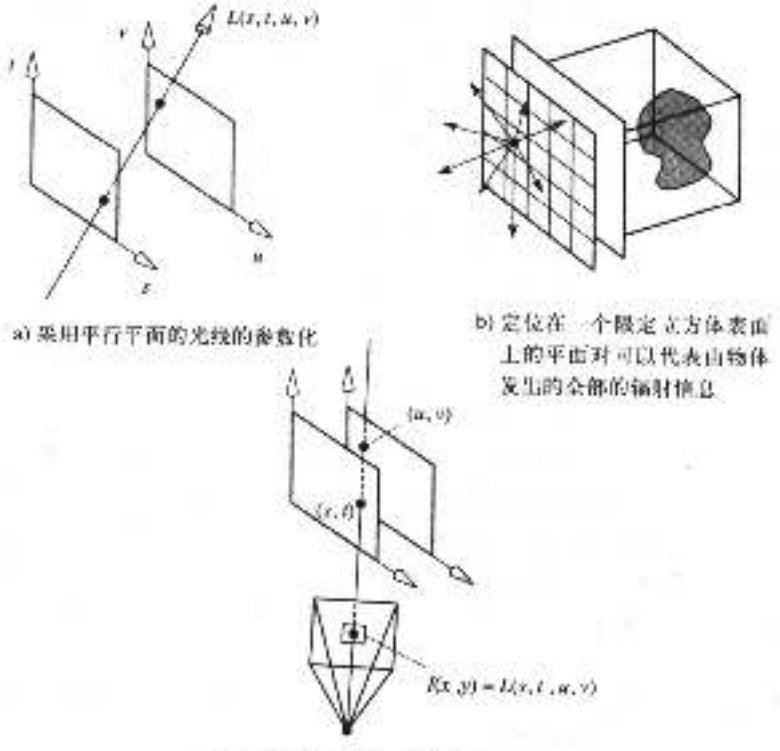
至今为止所考虑的系统使用的是单个的图像，或者是少量可从中产生所需图像的参考图像。我们已经考察了二维技术和使用深度信息的方法——三维扭曲。其中的某些方法涉及到

对已绘制的图像（LDI）的一个特殊形式的预计算，还有一些方法对用传统方法绘制的图像进行后处理。现在我们来讨论一种几乎完全都是预计算技术的方法，它是一种与环境映射有在某些关系的方法。一个环境图捕捉到到达场景中的一个点上的所有光线，该点是环境图的源或参考点。将一个物体放在该点上就可以大致地通过对图的索引来确定达到物体表面的光线。可以将这种方法进行扩展，使得我们对于场景中的每一个采样点都有效地存储一个环境图。也就是说，对于场景中的每一个点，我们熟悉到达该点的所有光线。因此可以把一个物体放在场景中的任何点上，并计算反射光。这一方法的优点是以存储大量的数据为代价将大多数三维扭曲中的问题最小化了。

光线场是一种相似的方法。对于希望重建观察的场景中的一个区域中的每一个点和所有的点，我们对其进行预计算并存储所有方向上的辐射。这种表示称为光线场或照明绘图（Levoy and Hanrahan 1996, Gortler等 1996）。我们用这种方法对自由空间中的一个区域进行构建是指构建一个没有遮挡物的区域。自由空间的重要性在于它将光线场从五维函数降阶为四维函数。一般来说，对于场景空间中的每一个点（x, y, z）都有在所有方向上传播的光线（参数化为两个角度），它们给出一个五维的函数。而在没有遮挡物的空间，可以假设（除非存在大气相互作用）沿着一条光线的辐射是常数。我们所感兴趣的两个“自由空间场景”为：从其凸包之外的任何位置上观察物体以及从其（空的）内部的某个位置观察类似于房间这样的环境。

461 ~ 463

空间中任何区域中的光线集合可以用它们与平行平面的相交来参数化，这是对于光线场的最方便的表示（见图16-17a）。这些平面可以放置在任何位置上，例如可以将一对平面平行于



a) 采用平行平面的光线的参数化

b) 定位在一个假定立方体表面上的平面对可以代表由物体发生的全部的辐射信息

c) 一个象素点$I(x, y)$的重建

图16-17 用光线的平行平面表示的光线场的绘制

包围物体的立方体的每一个表面进行放置，并存储由物体发出的所有的辐射信息(见图16-17b)。于是，对于物体的任何观察的重构由观察平面中的每一个像素组成，该观察平面投射一束光线穿过平面对，并把$L(s, t, u, v)$分配给该像素（见图16-17c)。重建实质是一个再采样过程，与前面所介绍的方法不同，它是一个线性运算。

由绘制的图像可以方便地构建光线场。对于放在靠近物体的平行平面对的一个光线场可以通过在(s, t)平面以相等的增量移动摄像机产生一系列切变的透视投影来建立。然后，每一个摄像点(s, t)定义一个光线束，这个光束来自于由(u, v)的范围所限定的一个平截体中的所有方向。可能会有人置疑，我们做的只是预计算在运行时所需要的物体的每一个观察。然而，有两个因素缓解了这个强制的方法。第一个因素，(s, t)平面上的分辨率可能大大地低于(u, v)平面中的分辨率。如果考虑与(u, v)平面共面的物体表面上的一个点，则(s, t)平面包含所有方向上的反射光线（由(s, t)平面的范围限定)。根据定义，物体表面的一个点上的辐射随着方向的变化是缓慢的。(s, t)平面上的一个低的采样频率就可以捕捉到这种变化。对于计算作为物体表面上位置的函数的变化需要有较高的采样频率。第二个因素，一个光线场表现出相当大的连贯性。据Levoy and Hanrahan（1996）报告，对于一个402Mb的光线场，其压缩比为118∶1，并得出结论，在给定的这一压缩比下，简单的（线性）再采样方法加上它比其他IBR方法简单的优点使得光线场方法成为一种切实可行的方法。

16.6 照片建模和IBR

IBR方法中另一个区分的因素是这些方法是否仅仅用于计算机图形学的成像（当有深度信息时)，或者它们是否将照片作为其源图像。摄影有解决由场景复杂性带来的其他主要问题——建模成本的潜力。其实世界的细节的丰富性和复杂性难倒了甚至最精巧的照片真实性的绘制程序，这些细节很容易用传统的照相方法来获得。其思想是用IBR技术处理照片，以使得这些照片可以被用于从不同于摄像机的观察点的一个观察点产生一个图像。

照片总是被用于纹理映射，但在那些需要有真实的印象而用传统的建模技术除非以很高的代价不能获得的地方，这种经典的工具又找到了它新的用途。一个好的例子是面部动画。在这个动画中将一个面部照片变形到一个计算机图形模型或结构上。照片映射保证了细节的精细程度，这对于使人信服和表情的真实是必要的。使用底层的三维模型作为控制动画的一个基础。

在从照片建立几何表示时，所遇到的许多问题从传统上来看都是属于计算机视觉领域，但是其目的是不同的。在计算机视觉中从一个场景恢复出来的几何信息通常都只有一个目的，比如在机器人行走中的防止碰撞或者物体的识别，通常考虑的是减少受低水平传感器影响的信息的方法。我们一般所感兴趣的是复现一个物体的形状而不考虑像纹理这一类不相关的信息。尽管可以将这类信息用做提取所需几何形状的一个手段，但是我们对其本身并不感兴趣。当对一个场景进行详细地建模时，像纹理这类细节才是我们所感兴趣的，另外还有纯的几何信息。

464
~
465

首先考虑用摄影来对建模进行辅助的手段。当前，可以利用的商用照片建模软件注重于较高地采用手工介入的方式抽取出纯的几何信息。一般的方法是采用一个预先标定的摄像机、每一次拍摄摄像机位置的知识以及足够数量的拍摄次数来获取比如说要建模的建筑物的结构。从对建筑物的拍摄中抽取边界可以建立起一个框架结构。这个工作通常是半自动地进行的，

操作员在不同的投影中对相应的边界进行匹配。它与使用特性的对应性由空间信息抽取形状的问题是一样的，只是现在是用人而不是采用建立对应关系的算法。我们可以结束在投影图上大量的手工工作了，因为许多工作可以通过使用一个CAD软件包来完成。其明显的优点是照片建模提供了自动抽取场景丰富的可见细节以及几何信息的可能性。

有趣的是，在从照片方法进行的建模中，计算机图形学工作者们已经绕过了最困难的问题。这些问题在计算机视觉领域中通过引入大量的手工干预得到研究。例如，从不同观察点投影的图像之间对应关系的经典问题通过由一个操作员手动地对帧之间的对应性建立一个对应度得到了解决，这样可以使得详细地建立逐个像素的对应度的算法能够成功。在计算机视觉中，似乎并没有考虑到这样的方法。也许这是因为在计算机视觉中已有的传统态度以及应用的限制造成的，因为传统的态度倾向于把对人类能力的模仿看成是终极的目标。

用照片建模获取细节有一些问题。一个问题是所获得的信息可能包含光源以及依赖于观察的现象，比如阴影和镜面反射。这些现象都必须在可以将图像用于从任意的观察点产生模拟的环境之前消除。另一个明显的问题是需要对照片中的细节进行扭曲以便与几何模型相匹配。这可能会涉及到对一个非常小的图像区域进行膨胀。例如，考虑一个取自地面的高大建筑物的带有详细外观的照片。靠近建筑物顶端的重要的详细信息可能会由于投影的变形而映射到一个很小的区域上。事实上，这个问题与观察插值是相同的。

现在来考虑使用照片建模而不去抽取其几何信息。我们只是简单地把所收集到的图像作为二维的投影，并用这些投影去计算新的二维投影。我们并不会企图去恢复场景的三维几何形状（尽管在考虑投影的三维信息时这一点是必需的）。这是基于图像的一种形状，它已经有一段历史了。

考虑一个艺术画廊或博物馆的虚拟的预排演，其对质量的需求是明显的。用户需要感受为了最好地观察展览而设计的精巧的照明条件。这些必须被重现，而且画卷中必须可以看到足够的细节。一个标准的计算机绘图方法可能会导致利用一个（独立于观察的）用于绘制的辐射度解，以及用于绘画的（照片的）纹理图。在进行昂贵的绘制计算时辐射度方法执行一 466
次只给出一个独立于观察的解，它对于虚拟现实中的许多情况已经足够了。但是它对于含有复杂的几何细节的场景并不是一个通用的解。正如我们所知，对一个按辐射度方法绘制的场景必须划分成尽可能多的元素以便于得到一个解，对于详细的场景的几何形状总是需要有很高的代价。

这一类应用（建筑物中虚拟的游历以及类似的应用）已经由于激光视盘和光盘提供的大的存储自由度而出现。大多数方法的内在缺陷是它们并不会提供连续的运动或预排演，而只是一些由用户（交互地）在建筑物之间“行走”时其位置选择的离散观察。它们与交互式的目录相似，需要用户“以离散的脚步”从一个位置到另一个位置进行行走，这些位置是由采用的照片图像的点来确定的。用户从观察点到观察点“跳跃前进”。

视频光盘实现的一个早期的例子是在1980年（Lippman 1980）建立的“电影图”。在这个早期的例子中Aspen街道以100英尺[⊖]的间隔被拍成了电影胶片。为了启动一个预排演，观察者从两个视频光盘机上检索出所选择的观察。为了记录环境，在每一个观察点上用四台摄像机，这样就使观察者能够从左到右进行摇摄。这个例子显示了这个方法中所隐含的交换，因为所有重建的观察都是预先存储的，记录只限于离散的观察点。

⊖ 1英尺 = 0.3048米。——编辑注

一个明显的计算机图形学方法是使用环境图。环境图是在绘制时建立的以便使周围的环境能够在一个闪光的物体上反射出来（见第8章）。在基于图像的绘制中我们只是将闪光的物体用一个虚拟的观察者来替代。考虑用户位于一个已经建立起六个观察（立方体）环境图的点（以照相的形式或合成的方式）。如果做一种近似，使得用户的眼睛始终刚好定位于环境图中的观察点，就可以合成任何依赖于观察方向的投影，该投影只要通过用户对适当的环境图进行采样来改变其凝视的方向即可。这个想法以流程图的形式示于图16-18。于是，对于一个静态的观察者，重叠地与环境图观察点定位在一起就达到了建立一个独立于观察的解的目的。我们已经将观察方向与绘制流程进行了分离。这时，合成一个新的观察包括对环境图的采样，而场景的复杂性问题已经被预计算的或照片图的分辨率给限定了。

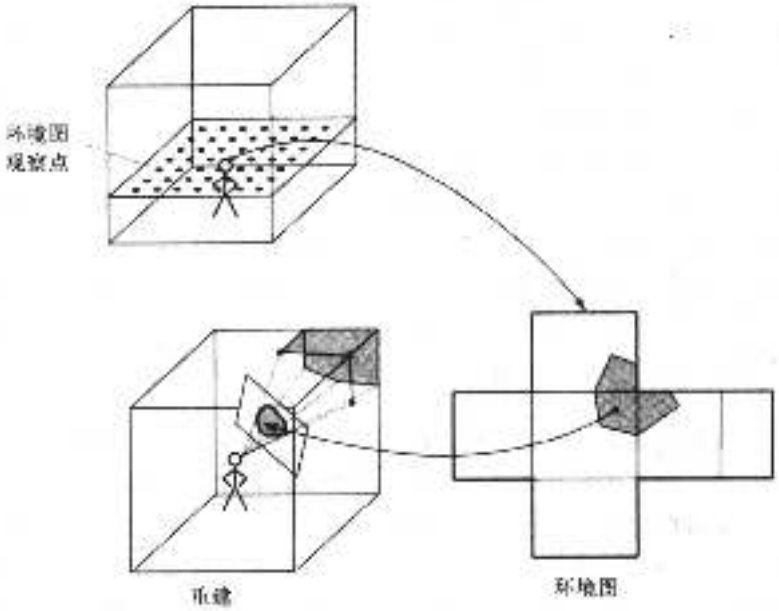


图16-18 从环境图合成一个用户投影

在浸入式的虚拟现实中，对一个所采用的图像产生器的最大需求来自于头部的移动（需要每秒钟计算60帧以防止头部的迟滞效果）。如果可以设计一种方法使得绘制成本几乎是独立于头部运动的，这将是一个很大的进步。然而，环境图只是对于静态观察者有效。我们可能需要观察者可能处于的每一个位置的一组图，是否可以将环境图方法扩展成可以涵盖完整的预排演呢？利用在预排演当中采用的限制条件即用户的眼睛总是在一个恒定的高度上，我们可以建立一些环境图，其观察点位于平面中一个粗糙的网格中的结点上。该平面平行于地平面并位于眼睛的高度。对于任何用户位置，能够通过利用四个相邻的结点上的环境图上的信息合成一个具有某种精确程度的用户投影吗？最终的投影质量将会依赖于图的分辨率和在一个房间中所拍取的图的数量——即眼睛平面网格的分辨率。图的分辨率将确定投影的细节质量，而图的数量确定了几何的精确度。

为了能够仿真使用一个传统的绘图流程方程的灵活性，通过使用照片（或预先绘制的环境图），我们必须使用一个强制的方法，并收集足够的与预排演所需的“分辨率”相匹配的观察，或者必须试图从现有的观察中得到新的观察。

当前从圆柱形的全景图进行的观察正被作为一个基于PC设备的流行工具而建立（见16.5.1节）。这包括通过以一种半受限的方式，即在一个水平的平面中旋转地移动摄像机来收集组分图像。计算机只是用来将组分图像“连接”成一个连续的全景图，而不恢复深度信息。

这个系统可以看成是发展的开始，最终能够通过用一个视频摄像机各处遍历来获取场景中的所有信息，从而产生场景的三维照片。我们可以将这样的发展看成是建模和绘制的一个分阶段的出现，现在它们之间还没有什么区别。虚拟的观察者可以浸入到一个照片质量的环境中去，并可以在其中自由地移动而不必将运动局限于摄像机的移动行程。 467 ~ 468

16.6.1 利用照片全景图进行基于图像的绘制

1994年建立起来的苹果公司的QuickTime VR是用照片全景图作为一个预存储的虚拟环境的经典例子。对于这样一个系统选用了一个圆柱形的全景图，这是因为这种全景图除了标准的摄像机和一个带有某些附件的三脚架之外不需要任何特殊的设备。至于说到投影，一个圆柱形的图具有只朝一个方向弯曲的优点，这使得为产生所希望的平面投影所必须做的扭曲运算更快。圆柱形图的基本缺点即限定了观察的垂直范围的缺点可以通过采用另一种立方体或球形图来克服，但是这两种方法都会涉及更困难的照片收集过程，而球形更难于扭曲。圆柱内在的观察弱点依赖于应用。例如，在建筑可视化中它可能是一种严重的缺陷。

图16-19（彩色插图）是这一系统的一个示意图。用户用一个摄像机在三角架上旋转来获取一系列正常的照片，然后将这些照片“拼接”在一起形成一个圆柱形的全景图像。观察者将自己定位在观察点上并注视此圆柱体表面的一部分。从圆柱体上所选择的部分再投影到一个（平面的）观察表面涉及到一个简单的图像扭曲操作，这个操作与其他加速策略结合可在一台标准的PC机上实时地运行。观察者可以连续地在水平方向和垂直方向对观察限制之内的垂直区域进行摇摄。

当前还是受到单色图像的限制。很有趣的是虚拟现实——即三维化和浸入式的最值得称赞的方面在当前被忽略了。可能在不久的将来，单色的非浸入式的图像在虚拟现实设备的普及中将占据统治地位，因为它不需要昂贵的立体观察设备而只需集中于重现一个视觉上的复杂环境。

16.6.2 合成全景图

用合成图像来合成环境图是直截了当的。例如，为了建立一个圆柱形的全景图，我们把观察空间的坐标 (x, y, z) 映射到一个圆柱形的观察表面 (θ, h) 上，表示为:

$$\theta = \tan^{-1}(x/z) \qquad h = y/(x^2 + z^2)^{1/2}$$

由照片建立一个圆柱形的全景图包括一些实际应用的点，由照片取代三维坐标，上面的方程还可以代替z的镜头的聚焦长度，并由照片平面的坐标计算x和y的值以及镜头的参数。这与将场景看成是其自身的一幅画是一样的。场景中的所有物体都被认为具有相同的深度。 469

圆柱形全景图的另一个内在的优点是在重叠之后，平面照片被映射到圆柱坐标上（就好像在摄像机中有一个圆柱形的胶片），对完整的全景图的构建只要通过平移就可以完成。这意味着要进行过程的自动化是直截了当的。使分离的图像一个接着一个地移动起来，直到达到某一种匹配，这是一个有时被称为“缝合”的过程。和平移组分图像一样，必须对照片进行处理以校正曝光的误差，否则的话会在全景图的垂直边界上留下可见的边界。

这时，可以把总的过程看成是一个将场景向观察表面的扭曲，再加上逆扭曲，以重新从全景图得到一个平面的照片。从用户的观察点来看，圆柱体使得能够有一个方便的图像采集模型，并有一个在场景中的用于观察的自然模型，也就是从一个固定的高度（目光）正常地观察环境，左右上下观看。

16.6.3 基于图像绘制的照片建模

在基于图像绘制的照片建模的最初的综合性研究之一中，Debevec等（1996）描述了一种具有一些有趣的和潜在的重要性质的方法。他们的基本方法是从对一个场景的稀疏的观察中导出足够的信息以便于进行基于图像的绘制（尽管所导出的模型也可以用于传统的绘制系统中）。他们的工作重点是建筑场景，主要基于三种创新：

1) **摄影学的建模**：在这里，他们基于一种简单的体基元和从一组稀疏观察得到的摄像机的观察点恢复出了一个建筑物的三维几何模型。

2) **依赖于观察的纹理映射**：用于绘制恢复出来的模型。

3) **基于模型的立体化**：用于解决一致性问题（并且因此可以进行观察插值），以及对1) 中没有建模的细节进行恢复。

Debevec等（1996）指出，他们的方法之所以成功，是因为：

> 把建模的任务从图像中分离成可以容易地由一个人完成的任务（但是并不是由一个计算机的算法来完成）和可以由一个计算机算法（而不是由一个人）很方便执行的任务。

470

摄影学建模过程包括用户观察一个建筑物的一组图像以及将一组体基元与照片观察相关联来定义一个近似的几何模型。其方法是启动模型的一个组分，例如一个矩形实体，并交互式地将模型中的边与场景中的边相关联。比如说一个盒子可以用这种方法半自动地与一个或一些以一个结构元素包含盒子的观察相匹配。这种手动的介入使得可以从照片中导出一个完整的几何模型，尽管在场景中只有模型的某些部分是可见的。几何模型的精确度，也就是模型和实物之间的差别依赖于用户启用了多少细节，即体基元的性质和场景的性质。其思想是获得一个表现该建筑物的几何模型，这个模型在接下来的处理中被用于得到摄像机的位置并实现一个一致性算法。因此，一个现代的塔式建筑就可以用一个盒来表示，而在一个表面上由于包含在一个平行于墙平面的平面中的窗口而产生的深度变化在这一阶段被忽略掉了。

一旦定义了一个完整的几何模型，即可对于每一个照片的观察都启动一个重建算法。这个过程的目的是恢复摄像机的观察点（这对于观察插值是必须的）以及恢复模型的边界坐标，而这一点在此模型要用于一个传统的绘制系统时是必须的。这一任务是通过采用一个假设的观察点将几何模型投影到照片观察上，并且比较图像边界的位置与所投影的模型边界的位置来完成的。这一算法通过使目标函数最小化来进行，目标函数作用于所观察到的图像边界与投影的模型边界之间的误差。模型边界和图像边界之间的对应关系已经建立之后，算法必须进行求解，而不应被一个局部最小值所阻断。

摄影学建模和重建这两个过程抽取足够的信息可以实现一个传统的绘制过程，Debevec将其称为“依赖于观察的纹理映射”。在这里，建筑物的一个新的观察是通过从所希望的观察点投影几何模型、把参考观察作为纹理图来处理并将从新的观察点重新投影到几何模型上产生的。其隐含意义在于建筑物是“过采样”的，其中的任何一点都可以出现在两个或更多的照

片观察中。因此，当产生一个新的、虚拟的观察时，对于新的观察中的每一个像素都将有一个纹理图的选择问题，因为这些纹理图（可能）由于空间的和未被建模的几何细节的原因而在建筑物上对相同的点（可能）有不同的值。这个问题可通过混合一些贡献得到解决，这些贡献反比于新观察与参考观察方向之间的角度，如图16-20所示。因此，有了术语“依赖于观察的纹理映射”，根据虚拟观察点相应于参考观察的位置对贡献进行选择和混合。

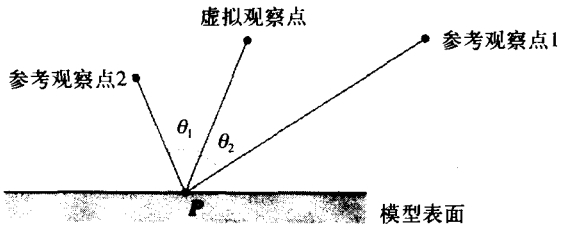


图16-20 在虚拟观察中相应于点*P*的像素接收到参考图像中相应像素的加权平均值。权重反比于θ_1和θ_2

这种绘制方法的精确度受限于由几何模型所捕捉到的细节。在真实的几何和模型几何之间有一个差，其范围依赖于用户在交互式建模阶段所付出的劳动并且假设几何模型将要丢失像窗口的凹陷处等细节。例如，一个以平面进行建模的表面可能接收到一个含有像明暗处理差别这样的深度信息的纹理，这可能会使得图像看起来不正确。该差别的大小依赖于所需的观察角度与选择纹理图时的观察角度之间的差。Debevec等（1996）继续扩展了他们的模型，用几何模型实现一种可以计算深度图的匹配算法，而使得从原始模型中丢失的几何细节可以被抽取出来。建立匹配关系也可以用于进行观察的插值。

这个过程称为“基于模型的”立体化，它将几何模型用作先验的信息，使得算法能够处理已经从相对较远的位置获取的那些观察。这一工作的实际动机之一就是对一组稀疏的观察进行操作（传统的空间匹配算法的主要问题是试图在没有有关场景结构知识的前提下进行工作。在这里，匹配问题的范围主要依赖于两个观察之间相互靠近的程度）。

471
~
472

第17章　计算机动画

17.1　计算机动画技术的分类和描述
17.2　刚体动画
17.3　连接结构和层次化运动
17.4　计算机动画中的动力学
17.5　碰撞检测
17.6　碰撞响应
17.7　粒子动画
17.8　行为动画
17.9　总结

计算机动画是一个很大的科目，完全可以用一整本教材来进行论述。本章集中讨论该领域中已经建立起来的一些基础问题，并且只作为对这一领域的一个介绍，而不是一个全面的综述。其主要目的在于为读者在概念上打下一个良好的基础，因为大多数现行的系统都是基于这些概念的。

引言

先不谈19世纪出现的某些玩具，从胶片的发明到现在，我们具有创建和传播移动图像的能力只有很短的一段时间。在这段时间内，动画好像并没有发展成为主流的艺术形式。在迪士尼世界及其模仿者之外，很少有动画片能够进入到普通观众的视野。令人感到惊奇的是，我们好像并不对那些表现运动的艺术感兴趣。而且，大部分情况下，都把动画归类于儿童娱乐的世界。也许可以认为是迪士尼把动画片上升到了一种艺术形式，但与此同时我们会抱怨它是一个聪明动物的奇怪世界，是人们把一些人类的情感赋予了它们。

随着计算机动画的出现，这种情况是否会有所改变呢？也许这样说还太早。计算机动画长期以来被归入到它的“艺术”领域，其最常见的证明是用于电视的标题序列和电视广告。这些产品被戏称为“飞行的标识图案”，它们在一个三维空间中移动刚体，并且以一种主流的计算机动画形式存在了20多年了。长久以来，这些产品已经不再新鲜，产品表现出一种奇怪的矛盾状态：其特性必须保持其传统功能，而同时又要有一种“人性的动画”。计算机动画正在被越来越多地在电影院中使用。但只是作为一种特殊效果的工具，而不是作为一种媒介（而且，近来的产品所采用的最无处不在的工具之一，即通过计算机变换图像，严格地讲并不是一种计算机动画，而只是一种二维的逐个像素进行的胶片图像的后加工）。在20世纪90年代后期，出现了全长的计算机动画产品，但是，要判断这种媒介是否将继续发展和延续还为时过早。

起初，计算机动画界很乐观。在1971年的经典著作《The Techniques of Film Animation》（胶片动画技术）（Hallas and Manvell 1971）中，作者对早期的科学计算机动画现状作了如下的评论：

> 目前的状态是，科学家和动画制作者可以创建在三维或四维中移动、在空间中

旋转的图画以及涉及到具有很高的数学精度的、代表复杂数学因素或科学原理的图画。处理过程对于传统的卡通片产生只需要很短的时间，这对于动画制作者来说是从电影的发明以来早就盼望的一种情况。现在所需要的就是像Klee这样聪明的艺术家，可以发明新的用以创建形状和形式的规范。工具已经有了，下一个十年可以完全用来发展令人兴奋的可视化的探索。

而事实上，在接下来的20年里，我们没有看到多少超出文献记录之外的计算机动画的发展。但是，也许在20世纪90年代，开始看到了这个早期预见的征兆。

那么，计算机动画能够为动画艺术家提供什么呢？有两个主要的工具是肯定的。首先，它可以大大地缩短传统动画片制作的工作流程。其次，提供一种构造三维动画的能力，这意味着可以将运动与三维物体的交互“变成胶片”。胶片动画一直被固定在二维空间，大多数的努力都集中在运动上，而对于三维的情况，比如说明暗处理和阴影只是偶尔加以考虑。看起来，动画制作者还是主要使用手工技术，而在20世纪90年代也确实有一些最流行的商业作品采用了由粘土建模生成的停止-运动的动画角色。

抛开有关艺术的话题不谈，对于所有的计算机动画来说，其核心的实际问题在于对运动的定义或控制。除了在构成一个计算机动画的物体或角色的复杂模型的建立过程中所涉及的
474 人力之外（这也是静态绘制所面临的相同问题），还要有对有真实感的运动的脚本和控制，这些毕竟是动画艺术的基础。随着模型越来越复杂，这一工作也变得越来越困难。一个具有单个参考点的单个刚体的动画相对比较直观，而对于一个复杂物体的动画，比如说一个有很多部分移动的动物，其组成部分之间相互关联，尽管其关联的方式是受限制的，要实现动画则是非常困难的。当然现在好莱坞正在生产最复杂的计算机动画。为了强调运动控制的困难，我们通过对一个当代的例子进行分析来开始这一章的论述。

史蒂文·斯皮尔伯格（Steven Speilberg）的电影《侏罗纪公园》被看作是到目前为止最逼真的计算机动画。它的来历很有趣。由于它是在现实动画当中所获得成绩的一个顶峰，所以我们将简单地了解生产这一影片所用到的技术（在17.3节中）。这种情况下计算机动画的角色是产生不能被胶片记录的生命，其目的就是“现实性”。然而，这并不是计算机在影片中被使用的唯一方式。在迪士尼的产品《狮子王》（1994）中用计算机动画来模仿迪士尼造型的动物，从而使其与传统的动画有相同的外观和感觉，以便使其与传统的动画无缝地融合。在这个产品中，采用了与17.7节和17.8节中所阐述的技术相似的方法产生了一群动物惊慌乱跑的序列。从所使用的动物演员的个数以及它们之间相互作用的角度来看，它比迄今为止用人工生成的动画要复杂得多，而这正是使用计算机技术的动机。

对于《侏罗纪公园》，斯皮尔伯格一开始雇用了一个停止——运动（木偶或模型）动画的专家，用这个高度发展的艺术形式使物体动起来。计算机唯一所做的事是停止——运动动画的后期处理（运动模糊）以使得镜头之间更平滑和真实性更强。这项工作由Industrial Light and Magic（ILM）来完成，这是一家在使用“传统的”特殊效果和使用像变形这样的数字技术方面都非常有经验的公司。然而，LIM公司同时又开发了一个只用计算机动画技术的Tyannosaurus Rex的测试序列。当斯皮尔伯格看到了这个序列之后，就有了下面的故事。他立即决定，所有的动画都应该由LIM公司的计算机来完成。在电影业中，《侏罗纪公园》被看作是一个转折点。许多人把这部电影看作是最终将创立计算机图形作为特技效果业中优先选用的工具，并看到好莱坞将用

这个使《侏罗纪公园》取得巨大商业成就的技术在未来产生更多的好产品。

从这个动画当中出现的真实性方面的进步使角色的运动看起来更令人信服。尽管在对建模和像皮肤纹理这样的细节上倾注了很大的热情，但是最后还是强调了运动。运动的真实性几乎可以肯定是由于对模型运动的脚本化的唯一系统产生的。尽管计算机技术对于停止-运动的木偶动画给了很大的自由度，这里模型的全局运动是受到连接到一个支撑装置上的机械性能的限制的，但它是一个有效脚本化与模型的可视化实现的一种紧密结合。也许在不久的将 475
来，我们会像《狮子王》那样看到由此模型产生的电影。

17.1 计算机动画技术的分类和描述

计算机动画技术可以按照某种不适当的混合方式进行分类，包括将要被进行动画的物体类型和性质以及用于获得动画的编程技术。我们选择介绍下列几类计算机动画类型:

- 刚体动画。
- 有关节结构的动画。
- 动态模拟。
- 粒子动画。
- 行为动画。

这个分类并不意味着是计算机动画技术的一个完整集合。例如，我们排除了软性物体或可变形物体动画的深入研究的领域。已经被采用的技术遍布在动画产品的各个角落。我们特别选出了这5个技术是因为它们在相对很短的计算机动画技术的历史中似乎已经成为相当完善的技术。当然有些动画也可能是用上述这些技术的某种混合产生的。刚体动画是自解释的，也是最容易和最普及的形式。其最简单的形式是使用一种标准的绘制程序并在一定的范围之内移动物体和/或者观察点。

有关节的结构是一些计算机图形模型，它模拟四足动物和二足动物。这样的模型可以有很多，从单节棍图到模拟完全的带有皮肤和/或衣服的表面表示的动物及人类。脚本化有关节结构的运动的困难程度是物体复杂度和所需运动的复杂度的一个函数。通常，我们对非常复杂的有关节的结构、人类或动物感兴趣，而正如我们将要看到的那样，这就意味着对运动的控制是困难的。

动态模拟意味着用物理学规律来模拟运动，它的动机是希望这些规律能够产生比手工制作更加真实的动作。动态模拟的缺点是它去掉了动画制作者的艺术性的控制。

粒子动画意味着逐个地使大量的粒子产生动画来模拟某些现象，看起来好像粒子“云”在进行整体的运动，比如一团烟火。顾名思义，粒子是很小的物体，每一个粒子一般都有其自己的动画脚本。 476

行为动画意味着对物体的行为进行建模。在这里，我们所说的“行为”是指比基本运动更复杂的运动。该运动可能依赖于某些行为规则，而这些规则是物体属性的一个函数，并且会演变成物体与其邻近物体之间的空间关系。行为动画与粒子动画很相似，只是它有一个重要的扩展，即粒子脚本不是独立的。行为动画中的一组实体根据一群粒子当中其邻近粒子的行为而展开动作。例如，在《狮子王》中一群逃窜动物单个地运动，也会根据逃窜的动物群中其自身的位置而运动。另一个例子是，鸟在一群鸟中的运动和鱼在一群鱼中的运动。每一个单个的实体既有自主运动，又有受到与场景中其他实体空间关系的连续变化的影响而产生

的运动。在这种情况下，行为规范的目的是对群体作为一个实体进行令人信服的刻画。

17.2 刚体动画

刚体动画是计算机动画中最古老和最熟悉的形式，其最常见的表现形式是在电视屏幕上无所不在的“飞行的标识图案”，而且它看起来好像成为了电视节目的开头标题所必须遵循的一种技术。刚体动画可以被描述为基本的动画需求，并且似乎以某种形式被所有其他类型的计算机动画技术所采用。这是一种执行起来最简单的计算机动画技术而且是应用最为广泛的技术，它主要由那些没有正式的计算机或者编程基础的人来使用，因此，接口问题就变得至关重要了。这类动画很明显是对三维场景进行绘制的一种扩展。将一个物体放在不同的位置，或者移动观察点（虚拟摄像机），并且在磁带上或胶卷上记录下所产生的单个的帧，以这样的方式绘制一个个的场景来产生一个动画序列。

问题在于，如何在场景中控制和定义物体的运动呢？物体是可以运动的，虚拟摄像机是可以运动的，还可以是两者同时运动。我们将描述如何移动一个单个的物体，但是很明显，这不是我们所需要的。

有两种现成的方法来建立“过程式的”刚体动画，即关键帧或插值系统以及明确的脚本系统。

17.2.1 插值或关键帧

关键帧系统是基于电影或动画界所熟知的生产技术。为了应付生产任何长度的动画短片
477 所需的庞大的工作负担，动画生产公司建立了一种层次化的系统，聪明的动画片制作者在这个系统中通过以一定间隔绘制帧定义了一个序列。这些帧被传送给“中间人”，由其来制作中间帧，然后对这些中间帧用“上墨辊”进行上色（当这一队伍当中的成员接受奖赏时，这个层次结构也被反映出来。在迪士尼的《白雪公主和七个小矮人》剧组中，四个主要的动画制作者每周获酬金100美元，中间人35美元，上色人20美元）。

将这个过程扩展到三维计算机动画中是很自然的事情，场景中物体的空间并置可以由关键帧来定义，计算机可以对中间帧进行插值。但是出现了很多问题，这些问题主要是由于简单插值策略不能代替人类中间人的智慧。一般来讲，我们在计算机动画系统中需要定义的关键帧比传统动画制作中所需要的要多。

考虑一个跳动的球的简单问题。如果我们用三个关键帧，即初始位置、结束位置和顶点位置，再加上线性插值，则所产生的弹道将是不真实的（见图17-1）。线性插值在大多数情况下都是不适当的。

通过允许动画制作者对于关键帧之间的动作特性定义更多的信息，可以使情况得到改善。例如，可以定义一个弯曲的路径。但是，这样做对于沿路径速度如何变化不能提供任何有用的信息。比如，一个球以均匀的速度沿着一条抛物线移动也是不真实的。因此，当移动物体时，要正确地控制运动就必须明确地定义作为时间函数的位置变化和沿特定路径的动态行为。

可以以很多方式来给出这样的信息。例如，可以用一组点，即关键帧的点来定义在某个特定时刻物体所处的位置，并且通过这些点放置一个物体，比如说一个立方体。

如果用B样条来进行插值（如3.6.3节所述），则关键位置就成为了结点。总的来讲，需要一条C^2连续曲线来模拟一个刚体的运动。如果我们严格地把自己限定在关于位置的讨论上，

则可以用一个4 × 4的建模变换M在场景中以一个时间的函数来放置物体。变换如下:

$$M(t)=\begin{bmatrix}0 & 0 & 0 & t_x(t)\\ 0 & 0 & 0 & t_y(t)\\ 0 & 0 & 0 & t_z(t)\\ 0 & 0 & 0 & 1\end{bmatrix}$$

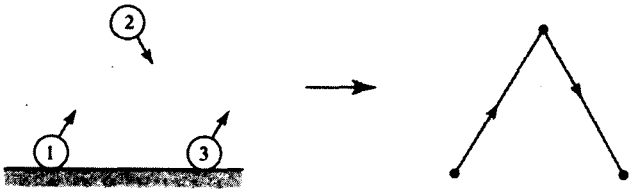


图17-1 对于定义在三个关键位置的跳动的球，线性插值将产生一条不真实的抛物线 478

牛顿定律给出:

$$\frac{F}{m}=a=\frac{\mathrm{d}^2M(t)}{\mathrm{d}t^2}$$

于是，为了要模拟空间中一个移动物体的效果，也就是说要使运动看起来更“自然”，变换矩阵中的元素必须具有连续的二阶导数。尽管看起来关键帧的插值是很直接的，而且对于简单的情况也确实如此，但还是有一些问题。通常，希望动画中的一个物体在沿着一条路径移动时有一些转动，因此，不可能向下式中应用相同的插值策略:

$$M(t)=\begin{bmatrix}a_{11}(t) & a_{12}(t) & a_{13}(t) & t_x(t)\\ a_{21}(t) & a_{22}(t) & a_{23}(t) & t_y(t)\\ a_{31}(t) & a_{32}(t) & a_{33}(t) & t_z(t)\\ 0 & 0 & 0 & 1\end{bmatrix}$$

这是因为矩阵元素$a_{11},\cdots,a_{33}$不是独立的。我们并不希望物体改变形状，所以，子矩阵A必须在所有时刻都保持是标准正交的，即列向量必须是单位向量，并形成一个正交三角形。这样一来，位置元素就可以被独立地插值，而旋转元素则不能。如果希望在九对元素$a_{11},\cdots,a_{33}$之间进行线性插值，则中间矩阵A_i将不是标准正交的，而且物体将改变形状（关于旋转插值的问题将在17.2.3节中单独论述）。

另一个问题是由于物体运动的动力学（速度和加速度）和路径的几何学是由相同的实体即变换矩阵$M(t)$来定义的。一般来讲，一个动画制作者需要进行控制，以便可以修改沿着路径的物体的动力学。

然而，由插值策略的定义产生了另一个问题，也就是关键帧之间路径的性质可能不是动画制作者需要的，尤其是根据所定义的关键帧的不同个数，可能还会出现不必要的偏移。此外，还有在B样条曲线中作为结点的那些关键帧对位置的影响问题。有可能动画制作者需要改变的路径不能通过改变一个帧的位置来进行，而是需要插入新的关键帧。这些缺点使人们建议采用另一种办法，即动画制作者明确地定义路径曲线和沿着一条路径的运动，而不是将一组关键帧送到一个插值策略中，这个策略的行为是“神秘的”。

17.2.2 明确的脚本

由此得出了明确的脚本的思想以及某些使人们能够书写这些脚本的界面的思想。最好的
479 方法是使用一个图形界面。这将遇到通常的问题，即试图从二维的投影看出三维的场景或者场景表示。但是，如果可以产生实时的序列或者最终序列的一个线框版，则问题就可以得到改善。

一个很明显的想法就是将三次参数曲线作为一种脚本形式（见第3章）。这种曲线可以被用作路径，参考点或者物体的原点将在其上运动。这些曲线可以很容易地进行编辑和存储，以便将来进一步使用。最好的方法称为双插值方法，它使用两条曲线，一条作为物体在空间中运动的路径，另一条用于其在路径上的运动特性。于是，一个开发者可以单独地改变一条曲线的性质。

一种可能的界面如图17-2所示。路径特性在三个窗口中被可视化和改变，窗口是曲线在xy、yz和xz平面上的投影。路径本身是嵌入在场景中的，其三维解释的线索来自场景中其他物体的位置，垂直线从曲线画到xy平面。动画制作者建立起路径曲线$\boldsymbol{Q}(u)$，应用一条速度曲线$\boldsymbol{V}(u)$并观察所得到的动画，如果需要的话，对其中的一个或两个特性进行编辑。

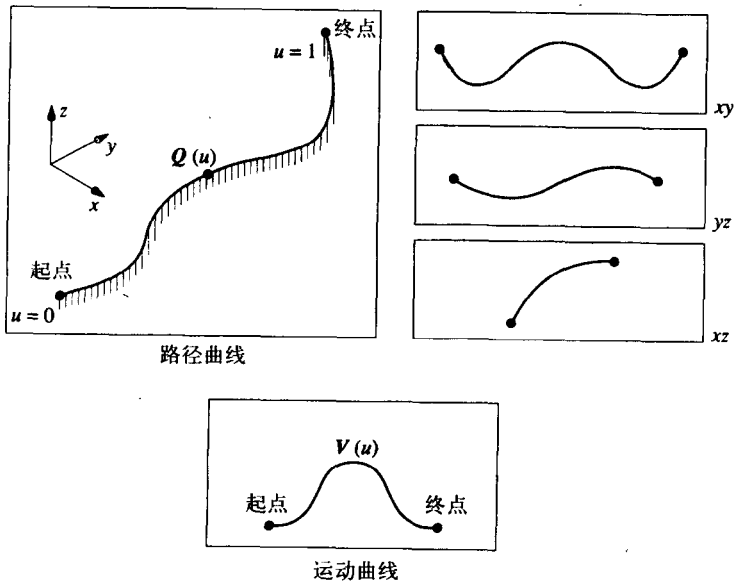


图17-2 刚体动画的运动定义——一个界面定义

从这些特性中生成动画意味着沿着路径特性曲线以等间距的时间间隔导出物体的位置，其原理如图17-3所示。步骤是:

1) 从$\boldsymbol{V}(u)$中为时刻t的一个帧求出相应于帧时刻t的距离s。

480 2) 沿着路径的特性曲线$\boldsymbol{Q}(u)$测量s单位，以求出u相应的值。

3) 把u代入到$\boldsymbol{Q}(u)$的方程中去，求出物体的位置(x,y,z)。

4) 在这个位置上绘制物体。

在这个简单的过程中隐藏了一个称为重新参数化的附加问题。$\boldsymbol{V}$根据u被参数化了。也就是说:

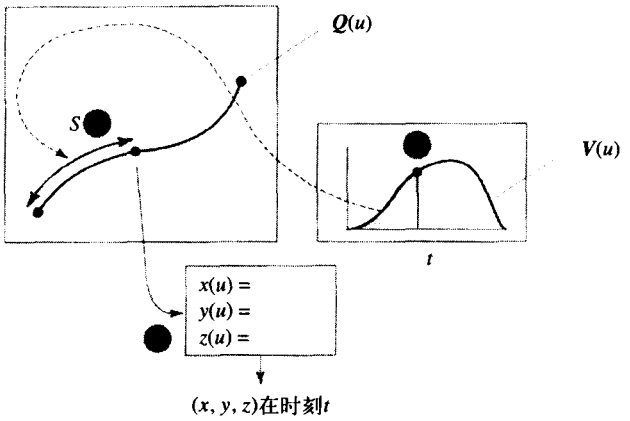


图17-3 求物体在时刻t的位置(x, y, z)

$$V(u) = (t, s),\ 其中 t = T(u),\ s = S(u)$$

已知帧时刻t_f，必须求出使$t_f = T(u)$的u值。然后把这个u值代入到$s = S(u)$中去，并且把这个距离在路径特性曲线$Q(u)$上“画出”。这里出现了同样的问题，路径参数按照u而不是按照s被参数化了。这个问题的意义以图形的方式示于图17-4中，该图中对等长（弧长）间距与曲线参数中的等间距情况进行了比较。

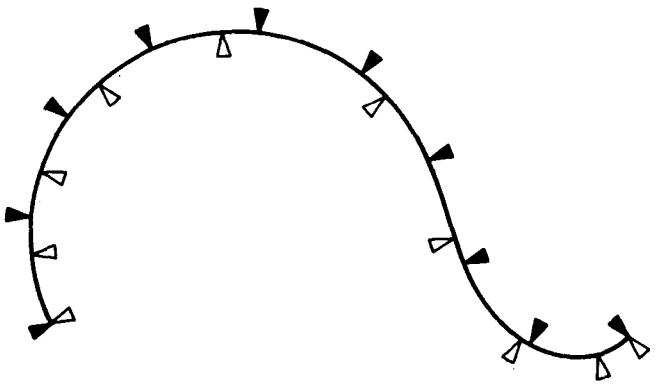


图17-4 等参数长度（空心的箭头）的间隔与弧长上的等间隔（实心的箭头）不相对应

481

在两种情况下，重新参数化的一般问题涉及到对两个方程的求逆，即:

$$u = T^{-1}(t) \text{ 和 } u = Q^{-1}(s)$$

已知t或s，求u的一个靠近值的近似方法是累积弦长。其原理如图17-5所示，算法为:

1) 通过对u取一些小的间隔，计算距离$l_1, l_2, l_3, \cdots$并且在表中插入$l_1, (l_1 + l_2), (l_1 + l_2 + l_3), \cdots$一直到累积的长度来构建一个累积弧长的表。

2) 为了求出相应于s的u值，即在该方法所允许的精确度范围之内的u值，我们取表中与s最靠近的输入。

这个简单的方法对实际系统并没有很多需求，但它是一个好的基本方法，在这个方法的基础上可以产生基于内容的增强功能，尤其是它可以形成一个脚本系统和一个交互性界面的基础。我们简要地对一些更重要的省略当中的某些内容进行介绍。

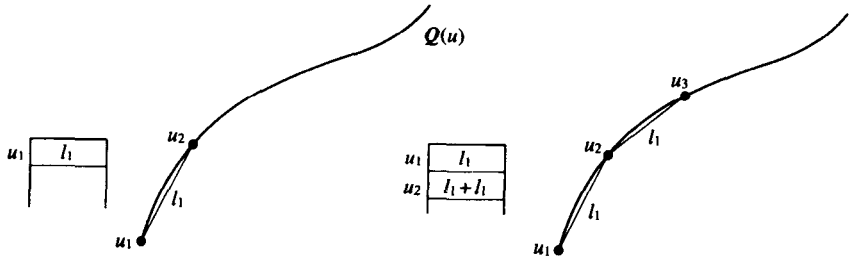


图17-5 累积的弧长近似

第一个重要的省略是，如果自由地改变$V(u)$，则物体沿着曲线运动所花的总时间一般将会变化。例如，可以使物体从静止状态更快地加速以缩短沿着路径行进所需的时间。然而，很多动画都必须恰好被限定在一个时间段内，更一般的情况是在行进时间保持恒定的限定条件下改变$V(u)$。

另一个更明显的问题是：一个物体在沿着一条路径行进时它应该采取什么姿态呢？方法本身只适合于单个粒子，或者等价地，只适合于一个物体上的单个参考点，而这个点只能沿着路径“垂直”地平移。通常，我们希望物体随着平移而发生旋转。最简单的情况下，可以引入另外三条脚本曲线以代表物体在沿着路径移动时的姿态。要达到这一目的最容易的办法是用三个角度对旋转进行参数化以定义绕着紧紧贴在物体上的三个坐标轴当中的每一个轴的旋转。这些角度称为欧拉角，其各自的名称分别为翻滚角、倾斜角和摇摆角。

如果要产生一个场景中有很多物体在运动的动画，并且如果这些物体独立地一次一个地动起来，则对物体之间的碰撞该如何来处理呢？如果使用了一种标准的绘制流程的话（具有一个Z缓冲器），则碰撞的物体将会只是简单地互相穿过，除非我们非常明确地检测到了这种现象，并通过设计界面对其发出信号。碰撞检测是一个与众不同的、并非不重要的问题。通常想要在计算机动画中处理的物体可能是非常复杂的，它们的空间延伸用几何描述来定义，而这个定义在大多数情况下并不适合于碰撞检测。考虑两个多边形网格物体，每一个都含有大量的多边形。如何检测一个物体上的一个顶点移动到了另一个物体的空间当中的情况并不是很明显。一个直接的方法涉及到将一个物体当中的每一个顶点与另一个物体中的每一个多边形进行比较，这是一个非常耗时的问题。而对于碰撞的检测只是问题的一部分；如何对物体的响应以及在碰撞之后的变形和运动进行建模？（碰撞检测在17.5节中有更为详细的介绍。）

17.2.3 旋转的插值

在刚体动画中，通常希望能够处理平移和旋转。一个物体在空间运动，并且在运动过程中改变方向。为了达到这一目的，需要对旋转进行参数化（把旋转和方向这样进行区分：方向由物体中的一个法向量来定义；旋转由一个轴和一个角度来定义）。传统的方法是使用欧拉角，这时使用相对于三个互相垂直的坐标轴的角度来代表旋转。在很多工程应用中，例如在航空学中，这些角度分别称为翻滚角、倾斜角和摇摆角。于是可以把一个旋转写为：

$$R(\theta_1, \theta_2, \theta_3)$$

欧拉角是用一个变换矩阵来实现的，如第1章中所介绍的那样，每一个欧拉角都有一个矩阵。于是一个总的旋转就由三个矩阵的乘积来表示。正如第1章中所述，为了实施一个旋转，我们定义三个旋转矩阵。旋转矩阵是不可交换的，旋转的性质依赖于所用的三个矩阵的顺序。

但是，现在先不考虑这个问题，要看到如果按这种方法来对旋转参数化，则动画制作者会遇到更大的困难。

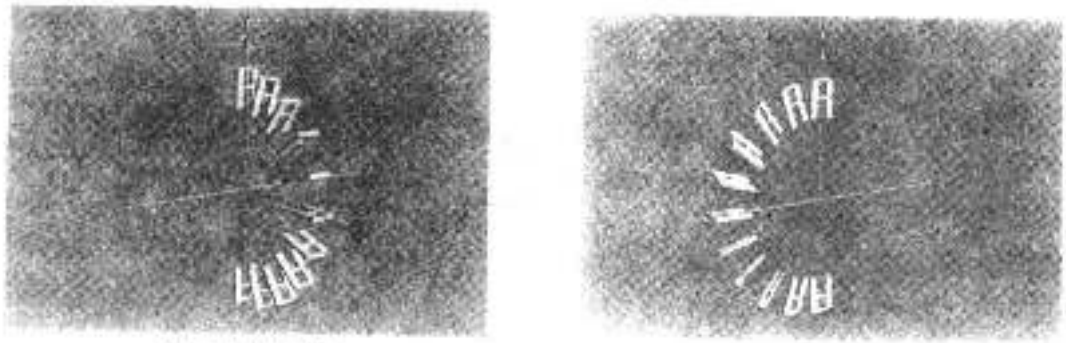
现在来考虑一个简单的例子，图17-6显示了一个字符R，它从一个初始点向一个终止点移动。在两种情况下，初始点和终止点的位置相同，但是两者的路径完全不同。在第一种情况下，进行了一个关于x轴的180°旋转；在第二种情况下，有两个180°旋转，同时在y轴和z轴上进行。单个旋转导致字符在一个二维平面上的移动而不发生"扭曲"，而在后一种情况下，字符在空间中走了一个完全不同的路径，并且随着其平移绕着穿过字符的一个轴扭曲。我们可以从中得出什么结论呢？这里有两个重要的结论。如果动画制作者希望有一条从一个位置和方向到另一个位置和方向的路径，则一般来讲，所有这三个欧拉角都必须以某种方式得到控制，使其可给出希望的效果。反回来看图17-6，这里的例子是由下列两个序列产生的。对于第一个路线：

482 ~ 483

$$R(0,0,0),\cdots,R(\pi t,0,0),\cdots,R(\pi,0,0)\quad t\in[0,1]$$

对于第二个路线：

$$R(0,0,0),\cdots,R(0,\pi t,\pi t),\cdots,R(0,\pi,\pi)$$



a) π的一个绕x轴翻滚　　b) π的一个绕y轴翻滚，接着π的一个绕z轴翻滚

图17-6 欧拉角参数化

特别考虑第二种情况可以得出结论，希望动画制作者把一个物体扭曲着穿过空间的想法转换成一种由欧拉角定义的特定的运动是不可行的。

对于插值有同样的考虑：如果动画制作者定义关键帧，则如何进行插值呢？事实上，在欧拉角的参数空间上存在着无限多种从一个关键帧到另一个关键帧的方法，很清楚，有一种对于能被理解的从一个关键帧到另一个关键帧的旋转的需求，这个旋转可能不是动画制作者所希望的，但是这比没有旋转的情况要好一些。

欧拉的理论告诉我们，通过一个固定的旋转从一个方向旋转到另一个方向是可能的。尤其是，他还提到，对于两个方向O和O'，存在一个轴l和一个角度θ，当绕着l旋转角度θ时，O就被旋转到了O'。可以用这个方法来理解图17-6。第一个例子是一个单轴旋转，它使物体从起点移动到终点。但那只是一种特殊的情况，而且易于可视化。通常对于两个方向O和O'，将如何求出并定义这个运动呢？这个问题用四元法得到了解决。

在上述欧拉角的插值策略中，还有另一个潜在的重要考虑。在离散表示的动画方法中，我们把运动和路径分离了。因为一般来讲，需要将物体在一条路径上表现出的运动的控制与

空间中对路径的定义相分离。同样的考虑也适用于对旋转的定义。可能出现的情况是由线性插值欧拉角产生的运动（角速度）并不是动画制作者所希望的。

17.2.4 用四元方法表示旋转

关于四元方法的一个有用的介绍性的概念是，可以把它们看成是像矩阵这样的运算符，
484 它把一个向量转换成另一个向量，但是去掉了对矩阵元素的无限制的选择。代替定义一个旋转矩阵的九个元素的是，我们定义了四个实数。从观察向量的角位移开始，把一个向量关于一个轴n旋转一个角度θ。

我们把旋转定义为一个角度位移，由(θ, n)给出，其中θ是关于轴n的旋转角。也就是说，代之以用$R(\theta_1, \theta_2, \theta_3)$定义一个旋转，将旋转写为$R(\theta, n)$。考虑在一个向量$r$上的角度位移，使其置于$Rr$的位置，如图17-7所示。

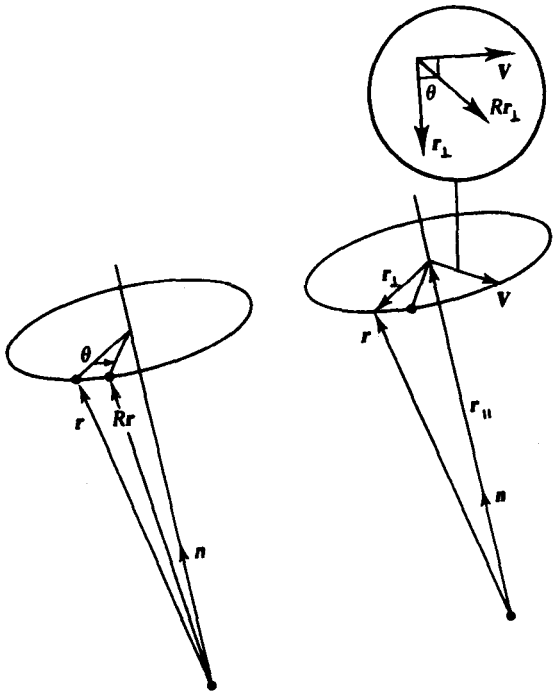


图17-7 r的角度位移(θ, n)

可以通过把r分解成平行于n的分量$r_{\parallel}$来分解问题，而$r_{\parallel}$从定义上来看，在旋转之后是不变的，并且垂直于n，而$r_{\perp}$位于穿过r和Rr的平面上。

$$r_{\parallel} = (n \cdot r)n$$

$$r_{\perp} = r - (n \cdot r)n$$

$r_{\perp}$被旋转到位置$R\ r_{\perp}$。我们建立一个垂直于$r_{\perp}$的向量，并且它位于正交于n的平面中。为了计算这个旋转，我们有:

$$V = n \times r_{\perp} = n \times r$$

其中×定义叉积，所以:

485

$$Rr_{\perp} = (\cos\theta)r_{\perp} + (\sin\theta)V$$

因此，

$$\begin{aligned} \boldsymbol{Rr} &= \boldsymbol{Rr}_{\parallel} + \boldsymbol{Rr}_{\perp} \\ &= \mathrm{R}\boldsymbol{r}_{\parallel} + (\cos\theta)\,\boldsymbol{r}_{\perp} + (\sin\theta)\boldsymbol{V} \\ &= (\boldsymbol{n}\cdot\boldsymbol{r})\boldsymbol{n} + \cos\theta(\boldsymbol{r} - (\boldsymbol{n}\cdot\boldsymbol{r})\boldsymbol{n}) + (\sin\theta)\boldsymbol{n}\times\boldsymbol{r} \\ &= (\cos\theta)\boldsymbol{r} + (1-\cos\theta)\boldsymbol{n}(\boldsymbol{n}\cdot\boldsymbol{r}) + (\sin\theta)\boldsymbol{n}\times\boldsymbol{r} \end{aligned} \tag{17-1}$$

现在我们将表明，通过角度位移来旋转向量$\boldsymbol{r}$可以通过一个四元变换来实现。也就是说，应用一个像矩阵一样的四元式来改变向量。

为了实现这样一个操作，只需要四个实数（与一个矩阵中的九个元素相比）。我们需要：

- 向量的长度变化。
- 旋转平面（可以用两个轴的两个角度来定义）。
- 旋转的角度。

换句话说，需要一种根据欧拉理论所要求的只具有四个自由度的一种表述。为此我们将使用单位四元式。顾名思义，四元式是“四个向量”，可以将其看成是复数的泛化，其中，s为实数或者标量，而x, y, z为虚数部分：

$$\begin{aligned} q &= s + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} \\ &= (s, \boldsymbol{v}) \end{aligned}$$

这里我们可以注意到它们与二维复数的相似性。二维复数可用于定义二维空间中的一个点或者一个向量。一个四元式可以定义四维空间中的一个点，如果$s = 0$，则定义三维空间中的一个点或者一个向量。在本书中，它们被用来表示一个向量加上一个旋转。$\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$为单位四元式，与向量系统中的单位向量等价。但是，它们服从不同的结合律：

$$\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1,\quad \boldsymbol{ij} = \boldsymbol{k},\quad \boldsymbol{ji} = -\boldsymbol{k}$$

用这些定律，可以导出加法定律和乘法定律，每一个定律都导出一个四元式：

$$\text{加法：}\quad q + q' = (s + s', \boldsymbol{v} + \boldsymbol{v}')$$

$$\text{乘法：}\quad qq' = (ss' - \boldsymbol{v}\cdot\boldsymbol{v}', \boldsymbol{v}\times\boldsymbol{v}' + s\boldsymbol{v}' + s'\boldsymbol{v})$$

四元式

$$q = (s, \boldsymbol{v})$$

的共轭为：

$$\bar{q} = (s, -\boldsymbol{v})$$

四元式和其共轭的乘积定义其大小：

$$q\bar{q} = s^2 + |\boldsymbol{v}^2| = q^2$$

486

如果

$$|q| = 1$$

则q称为一个单位四元式。所有单位四元式的集合在四元空间形成一个单位球，单位四元式在定义总旋转时起着非常重要的作用。

可以看出，如果：

$$q = (s, \boldsymbol{v})$$

则存在一个$\boldsymbol{v}'$和一个$\theta \in [-\pi, \pi]$，使得：

$$q = (\cos\theta, \boldsymbol{v}'\sin\theta)$$

如果q是一个单位四元式，则：

$$q = (\cos\theta, \sin\theta\, \boldsymbol{n}) \qquad \text{（命题17-1）}$$

其中$|n| = 1$。

现在，用四元式来考虑对图17-7中一个向量$\boldsymbol{r}$的操作。$\boldsymbol{r}$定义为四元式$p = (0, \boldsymbol{r})$，并将操作定义如下:

$$R_p(p) = qpq^{-1}$$

也就是说，建议通过把它表示为左乘一个q并且右乘一个q^{-1}的四元式来旋转向量$\boldsymbol{r}$。这就保证了产生的结果将是以（0, v）形式出现的四元式。换句话说，q被定义为单位四元式（s, v）。很容易看出:

$$R_p(p) = (0, (s^2 - v \cdot v)\boldsymbol{r} + 2v(v \cdot \boldsymbol{r}) + 2s(v \times \boldsymbol{r}))$$

用命题（17-1）并代入，得出:

$$Rq(p) = (0,(\cos^2\theta - \sin^2\theta)\boldsymbol{r} + 2\sin^2\theta\, \boldsymbol{n}(\boldsymbol{n}\cdot\boldsymbol{r}) + 2\cos\theta\sin\theta(\boldsymbol{n}\times\boldsymbol{r}))$$
$$= (0,\boldsymbol{r}\cos 2\theta + (1-\cos 2\theta)\boldsymbol{n}(\boldsymbol{n}\cdot\boldsymbol{r}) + \sin 2\theta(\boldsymbol{n}\times\boldsymbol{r}))$$

现在，将上式与式（17-1）相比较，你会注意到，在系数2的边上出现的角度形式上是相同的。可以从中得出什么结论呢？由一个角度位移（θ, $\boldsymbol{n}$）所做的向量旋转动作与把这个角度进行偏转，并“合并”到四元式空间相同。将其表示为单位四元式:

$$(\cos(\theta/2), \sin(\theta/2)\boldsymbol{n})$$

对于四元式（0, $\boldsymbol{r}$）进行运算$q()q^{-1}$。因此，可以以四个参数的形式对方向进行参数化。

$$\cos(\theta/2), \quad \sin(\theta/2)\boldsymbol{n}_x, \quad \sin(\theta/2)\boldsymbol{n}_y, \quad \sin(\theta/2)\boldsymbol{n}_z$$

用四元式的代数方法来处理各个分量。

现在让我们回到图17-6中的例子来看一看这种运算是如何进行的。第一个π的单个x翻滚由四元式表示为:

487

$$(\cos(\pi/2), \sin(\pi/2)\ (1, 0, 0)) = (0, (1, 0, 0))$$

同样，π的一个y翻滚和π的一个z翻滚分别由（0, (0, 1, 0)）和（0, (0, 0, 1)）给出。现在，对π的y翻滚接着再进行π的一个z翻滚的结果可以表示为单个的四元式，以这两个四元式相乘的形式给出:

$$(0,(0,1,0))(0,(0,0,1)) = (0,(0,1,0)\times(0,0,1))$$
$$= (0,(1,0,0))$$

这与π的一个x翻滚是相同的。

对这一节的总结是，使用了四元式来表示旋转，它们可以用于表示平移，但是，把平移和旋转结合到一个与齐次坐标系相似的方案中并不是很直接。

17.2.5 对四元式插值

已知四元式的参数化比欧拉角度参数化的优越性，这一节讨论了在四元空间中对旋转进行插值的问题。设想动画制作者正坐在一个工作站的旁边，交互性地建立一个关键帧方向的序列，而所用的方法是恰当的。这一工作通常是用基本的旋转操作来完成的，但是现在，当使用欧拉角度时，对于动画制作者所加的限制可以被排除掉了。也就是说对于每一个关键帧，以一个固定的顺序使用一个固定数量的基本旋转来完成。总的来讲，每一个关键帧都将被表示成为一个旋转矩阵。这个矩阵的序列再被转换成一个四元式的序列。在关键四元式之间的插值被执行并产生一个中间四元式的序列，然后，再把这些序列转换回旋转矩阵。然后把这

个矩阵应用到物体上。所用到的四元式插值对于动画制作者是透明的。

1. 进出四元式空间

这样一个方案的实施需要我们不断进出四元式空间。也就是说，从一个总的旋转矩阵进入到一个四元式空间，再从四元式空间返回到旋转矩阵。为了用四元式q旋转一个向量$\boldsymbol{p}$，采用如下的运算:

$$q(0, \boldsymbol{P})q^{-1}$$

其中q是四元式:

$$(\cos(\theta/2), \sin(\theta/2)\boldsymbol{n}) = (s, (x, y, z))$$

可以看出，这相当于将下列的旋转矩阵应用于该向量: 488

$$\boldsymbol{M} = \begin{bmatrix} 1-2(y^2+z^2) & 2xy-2sz & 2sy+2xz & 0 \\ 2xy+2sz & 1-2(x^2+z^2) & -2sx+2yz & 0 \\ -2sy+2xz & 2sx+2yz & 1-2(x^2+y^2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

这就意味着，我们可以从四元式空间移动到旋转矩阵。

从一个旋转矩阵到一个四元式的反向映射如下。所有需要的条件是转换一个总的旋转矩阵:

$$\begin{bmatrix} M_{00} & M_{01} & M_{02} & M_{03} \\ M_{10} & M_{11} & M_{12} & M_{13} \\ M_{20} & M_{21} & M_{22} & M_{23} \\ M_{30} & M_{31} & M_{32} & M_{33} \end{bmatrix}$$

其中: $M_{03} = M_{13} = M_{23} = M_{30} = M_{31} = M_{32} = 0$和$M_{33} = 1$直接转换成上面的矩阵形式。已知一个总的旋转矩阵，要做的第一件事情是检查其对角线分量M_{ii}的和。该值为:

$$4 - 4(x^2 + y^2 + z^2)$$

由于相应于旋转矩阵的四元式是单位大小的，所以有:

$$s^2 + x^2 + y^2 + z^2 = 1$$

和

$$4 - 4(x^2 + y^2 + z^2) = 4 - 4(1 - s^2) = 4s^2$$

于是，对于一个4 × 4的齐次坐标矩阵，有:

$$s = \pm\frac{1}{2}\sqrt{M_{00} + M_{11} + M_{22} + M_{33}}$$

和

$$x = \frac{M_{21} - M_{12}}{4s}$$

$$y = \frac{M_{02} - M_{20}}{4s}$$

$$z = \frac{M_{10} - M_{01}}{4s}$$

2. 球形线性插值（slerp）

在对我们的方案进行了概述之后，现在来讨论如何在一个四元式空间进行插值。由于一

个旋转映射到一个单位大小的四元式中，所以整个旋转集合也会映射到四元式空间中四维单位超球面的表面上。因此，通过关键帧定向插值的曲线就应该位于这个球的表面上。考虑最
489 简单的两个关键四元式之间的插值问题。一个自然的、直接的两个关键帧之间的线性插值导致一个中间加速的运动。这种情况在一个二维平面的相似状况如图17-8所示。该图表示，在由线性插值产生的球的表面上的路径出现了不相等的角度并引起了角速度的加速。

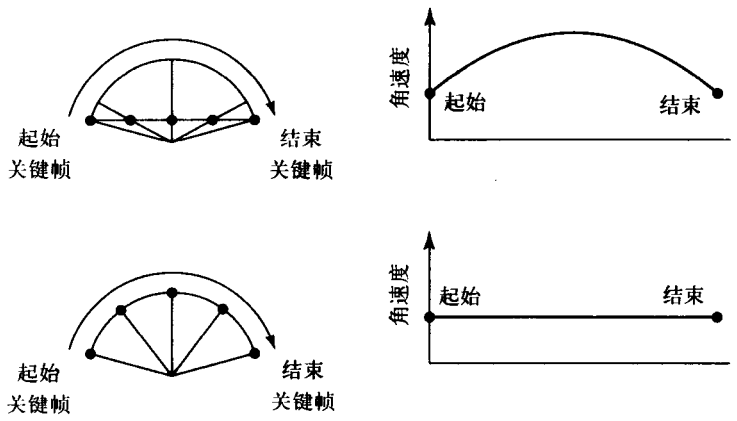


图17-8 一个二维的类比，表明简单的线性插值和简单的球面线性插值之间的差别

这是因为我们并没有沿着超球面的表面移动，而是穿过它。为了保证一个稳定的旋转，必须采用球面线性插值（slerp）。这时，我们沿着穿过两个关键帧的两点间的最短弧线移动。

用于球面线性插值的公式从几何上很容易导出。考虑二维的情况，有两个向量A和B，它们由角Ω分开，向量P与A之间的夹角为θ，如图17-9所示。P由A和B之间的球面插值导出。我们有：

$$P = \alpha A + \beta B$$

繁琐一点可以求出α和β。已知：

$$|P| = 1$$
$$A \cdot B = \cos\Omega$$
$$A \cdot P = \cos\theta$$

可以给出：

$$P = A\frac{\sin(\Omega-\theta)}{\sin\Omega} + B\frac{\sin\theta}{\sin\Omega}$$

在两个单位四元式q_1和q_2之间的球面线性插值由将上面的式子向四维扩展并用Ωu代替θ得到，其中，$q_1 \cdot q_2 = \cos\Omega u \in [0,1]$。我们有：

$$\text{slerp}(q_1,q_2,u) = q_1\frac{\sin(1-u)\Omega}{\sin\Omega} + q_2\frac{\sin\Omega u}{\sin\Omega}$$

现在，给出任意的两个关键四元式p和q，有两个可以沿着其运动的弧，这相当于在不同起始方向

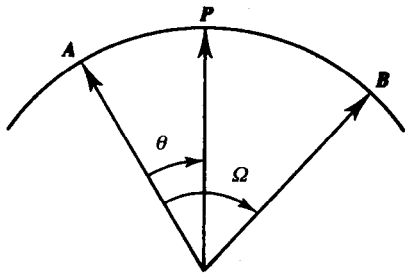


图17-9 球面线性插值

上的最短连接弧。其中之一转了一个圈，而这是我们希望避免的。很自然，人们可能会假设，这种情况可以化简成p和q之间用角度Ω进行的球面插值，其中：

$$p \cdot q = \cos\Omega$$

或者用角度$2\pi-\Omega$从相反的方向进行插值。但是，这将不会取得希望的效果。其原因是，方向的超球面拓扑并不是三维欧几里得球的直接扩展。为了正确地评价这一问题，考虑下面的事实就足够了，即每一个旋转在四元式空间都有两种表示，也就是q和$-q$，这就意味着q和$-q$的作用是相同的。产生这种情况的原因是，从代数上来看，对于运算$q()q^{-1}$与运算$(-q)()(-q)^{-1}$的效果是相同的。因而正好相对的点代表相同的旋转。正是由于这一点，在确定最短弧时，必须小心处理拓扑学上的奇异问题。一种可以处理这一问题的策略是选择在四元式对p和q或者p和$-q$之间进行插值。已知两个关键帧的方向p和q，求其差的大小，也就是$(p-q)\cdot(p-q)$，当取消了第二个关键帧时，将这一值与差值的大小相比较，即$(p+q)\cdot(p+q)$。如果前者较小，则我们已经沿着最小弧移动，这时不必再做其他动作。否则，如果第二个值最小的话，则用$-q$代替q，并继续。这些考虑以示意图的形式示于图17-10中。

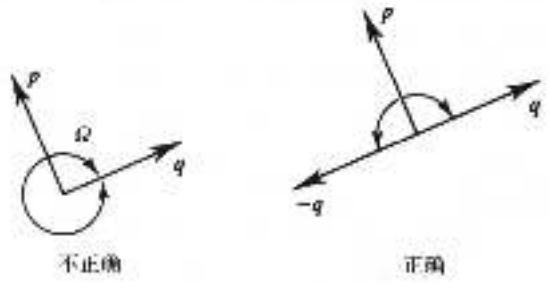


图17-10 四元式超球面上的最短弧确定

到目前为止，我们已经讨论了两个关键帧方向之间的线性插值的球面等价性，而正像对于线性插值那样，在多于两个关键帧方向之间的球面线性插值将产生不平稳的、在关键帧之间突然改变的运动。这种情况以一种三维模拟的形式总结于图17-11中，它表明在关键帧之间球表面上的曲线是不连续的。在这个图中还可以看出，角速度并不是常数，而且在关键帧上并不连续。可以使角速度在所有的帧上为常数，方法是为关键帧之间的每一个间隔都赋以正比于间隔大小的一定数量的帧。也就是说，用下式计算关键帧对q_i和q_{i+1}之间夹角θ的大小。

490
|
491

$$\cos\theta = q_i \cdot q_{i+1}$$

其中，两个四元式$q=(s,v)$和$q'=(s',v')$的内积定义为：

$$q \cdot q' = ss' + v \cdot v'$$

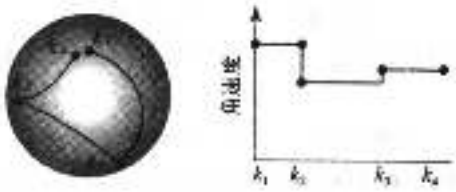


图17-11 使用slerp对四个关键帧进行插值的三维模拟

消除路径连续性是更困难的。较高阶的连续性所需要的是三次样条的球面等价性。可是，由于现在是在一个四维的超球面的表面上进行运算，所以问题比在三维欧几里得空间建立样条要困难得多。Duff（1986）和Shoemake（1985）都遇到了这个问题。

最后，还要提一下应用四元式时的一个潜在问题。四元式插值是不加区别的，即它并不会偏向于一个方向而不倾向于另一个方向。两个关键帧之间的插值产生一个移动，该移动依赖于关键帧的方向，而且仅此而已。当设计虚拟摄像机时这是不方便的。正常情况下，当移动摄像机时，总是需要使胶卷的平面向上，这通常由一个“向上”的向量来定义。因为它非常自然，所以，一个首选方向的概念不能被容易地建立到四元式表示中。如果在这种情况下使用它，则必须重新设置摄像机向上的向量或者用到其他的方位（当然，在某些场合下使用翻滚摄像机）。

17.2.6 作为动画物体的摄像机

任意的或者所有的外部摄像机参数都可以制作动画。但是，大多数普通类型的摄像机动画都肯定应用了第一人称计算机游戏和相似的程序，就是说一个摄像机在用户的控制下飞过大部分是静态的环境，即所谓的“遍历”或“飞过”。这时，用户在控制着观察点，并且通常有一个（两个自由度的）观察方向。在形成关键帧的连续界面采样之间通常需要插值，而最
492 重要的限制是保持摄像机向上的向量向上。一般情况下不允许对观察方向矢量进行定向（只有在摄像机绕着观察方向翻滚时才可以，在第一人称游戏中就是在你死时）。

另一个常见的应用是在观察点处于用户的控制之下，但是摄像机总是指向一个感兴趣的物体，并且该物体可以是静态的也可以是移动的。这种应用的一个常见的例子就是第三人称计算机游戏，这时游戏中摄像机被通过一个“不可见的”硬连接固定在角色的头上。代替通过角色的眼睛看到环境的是用户越过角色的肩膀来看。在这种情况下，观察方向向量是由角色导出的。观察点高效地在以角色为中心的一个球的表面上的一部分中运动。如果在这个应用程序中运用四元式插值则在一次插值之后必须对向上的向量重新进行设置。

17.3 连接结构和层次化运动

在计算机动画中对四足或二足模型的运动的描述一段时间以来一直是人们努力追求的研究主题。这种计算机模型被称为有关节的或连接的结构。在计算机动画中用来进行运动控制的大多数方法都是试图对工业机器人领域中的技术进行扩展。正像插值是第一个被用于刚体动画中的思想一样，使用机器人方法学在连接结构中对连接或肢体的运动进行参数化似乎是继续进行的方向。尽管这可能是一种很显然的方法，但是这种方法目前还不是很成功。一个问题是机器人的控制本身也是一个研究领域，在该领域中并不是所有问题都得到了解决。可能的一个重要原因是对一个工业机器人精确的机械运动的控制所需的技术并不一定提供一种舒适的有创造性的环境，可以使得动画制作者能够对自由的人或事物以及人类或者动物更复杂、更细微的运动进行脚本化。

另一个原因是动物的结构不是刚性的，而且连接本身也会变形，如图17-12所示。事实上，至今为止最成功的连接结构的动画《侏罗纪公园》使用的是一种特别的技术来表示或者导出复杂（恐龙）模型中连接的运动。首先简单地介绍一下这些技术。这将使我们能够正确地评价在《侏罗纪公园》中动画制作者所面临的问题的困难程度，以及其解的高效性。

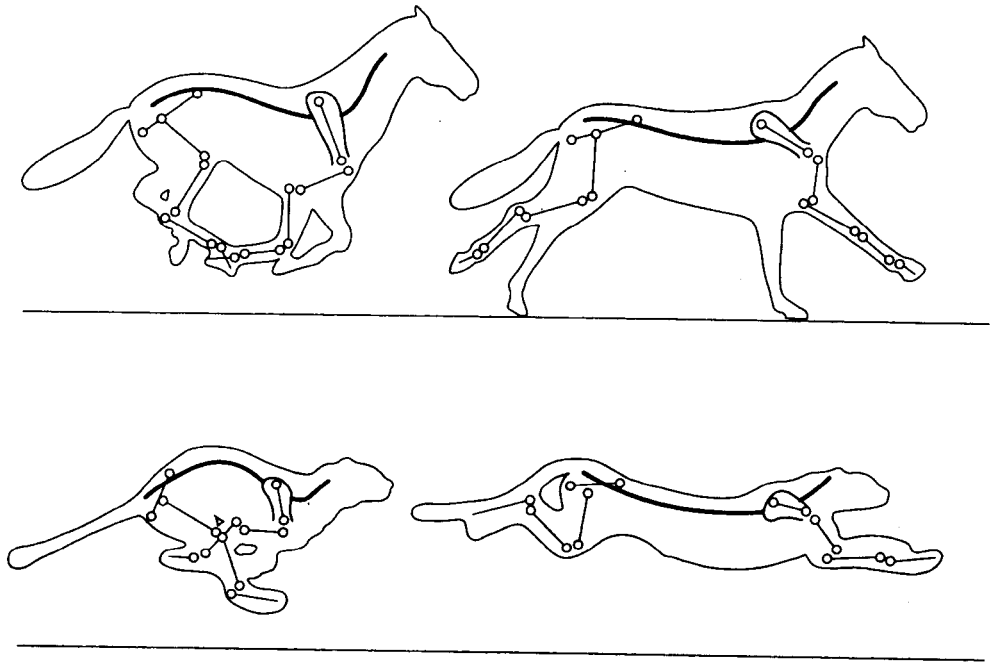


图17-12 马和猎豹中的样条弯曲（Gray (1968)）

首先，什么是带关节的结构？它只是一组刚体或者连接部件，由关节互相连接在一起，这些关节使得结构的各个部分能够以一种互相之间有关联的方式运动。对于动物动画，连接部件形成一种简化的骨架、一种棍图，只用于对结构的控制。它们是一种并不被绘制的抽象概念。连接部件被动物物体的外表面所“覆盖”并绘制（原则上来讲，从一个由多边形网格表示的刚体进行绘制是没有差别的。在这种情况下可以有效地控制代表物体的一个向量的位置和方向）。

考虑一个简单的例子——一个人的腿部。可以按如图17-13a所示的模型来建模，用两个连杆连接三个关节——髋关节、膝关节和踝关节。为了简化起见，可以把运动限制在包含关节的平面中，允许髋关节和膝关节之间的连杆绕着髋关节在一定的限定范围之内旋转，并允许踝关节和膝关节之间的连杆绕着膝关节旋转（当然，这种连杆只能朝一个方向旋转）。脚关于踝关节的旋转更复杂，因为脚本身也是一个关节结构。已知这样一种结构，如何开始来定义一个动作的脚本（比如说，腿的结构方法）使其执行一个行走的动作？非常明显，结构的运动要受到整体连接性的限制。结构由一些连杆链组成，一个连杆引起其周围的连杆运动，连杆本身具有限制，类似人类、动物骨架关节的旋转限制。这种结构的实际效果是，由于这些限制必须在所有的插值位置上有效，所以不能轻易地使用关键帧系统。于是就接受了这样一个系统，它的连杆具有其自身的相对运动的定义。也就是说，连杆i的运动是相对于与之相连接的连杆j的运动而定义的。于是，这种系统的动画是通过分别对每一个连杆的动画来实现的。从定义上来讲，它们还必须具有一个层次结构，即除非其本身是顶端连杆，否则每一个连杆的上方必须有一个连杆；除非其本身是底部连杆或者是端效应器，否则每一个连杆的下方必须有一个连杆。一个连杆继承其上方所有连杆的传输动作。

493
~
494

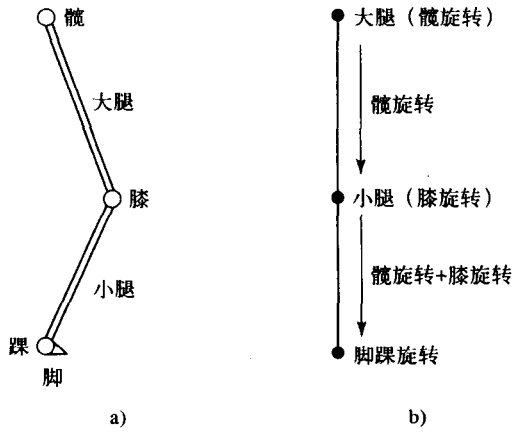


图17-13 一个简单的关节结构及其层次化表示

对于这个问题有两个方法，即前向动力学和反向动力学，两者均来自机器人学。

前向动力学是一种稍微繁琐的低层次的方法，动画制作者必须清晰地定义关节结构中的每一个部件的所有运动。与所有低层次的方法一样，动画制作者必须完成的工作量是结构复杂程度的一个函数。关节结构被看作是一种结点的层次结构（见图17-13b），它具有相关的移动以某种形式连接到结点上的连杆的变换能力。每一个结点代表一个物体的部位，如上臂和小臂。可以通过使用明确的脚本曲线以一种时间的函数来定义变换值的方法来使这样一个结构产生动画。代替只有一个路径属性移动刚体的一个参考点的情况，现在有很多属性，每一个属性都可以移动结构中的一部分。考虑这个结构中的继承性，本例中的髋部旋转引起小腿以及大腿的旋转。显然有如下的考虑:

- **髋关节** 这是该结构中的“顶部”连接，需要赋予全局的运动。在一次简单的行走中，这只是在平行于地面的一个平面中的平移。在一种更真实的模拟中，可能必须考虑在行走周期中由于脚的提升动作，使得髋部上升和下降。
- **髋-膝连杆绕髋关节的旋转** 可以把旋转定义为一个时间的角度函数。我们把这个连杆下方的所有部件都固定，则会得到一种直腿的行走(礼貌地讲，是圆规式的步态，但我们更熟悉的名字是“正步”)。
- **膝-踝连杆绕膝关节的旋转** 为了将正步松弛成自然的步伐而定义膝关节的旋转。

等等。

为了获得希望的运动，动画制作者从层次结构的最顶层开始，并向下工作，清晰地应用每一点的脚本。一个脚本的演变过程如图17-14所示。应用顶层脚本导致一个正步。第二个脚本（膝旋转）允许小腿弯曲。同时应用这两个脚本可能导致一种使脚总是与小腿形成直角的行走。这就产生了踝的脚本。

495

即使在这个简单的例子中也出现了问题。不难看出，当开始脚本化脚的时候，是不能允许髋关节以平行于地面的直线移动的，这可能引起脚穿透地板。必须对髋应用某些垂直的位移作为一个时间函数等等。现在只是在考虑一个非常简单的例子，即一个行走的动作。如何把这种技术进行扩展，使其能应用于一个复杂的关节结构来执行一个打斗的序列，而不是只进行一种重复性的行走循环呢?

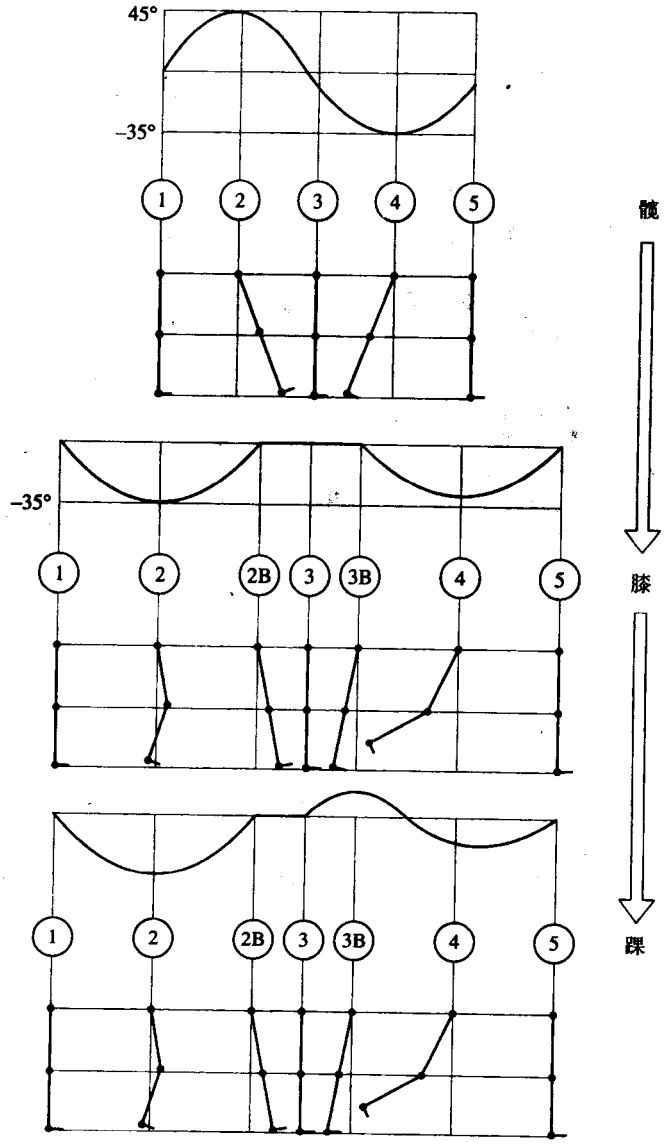


图17-14 一个腿的脚本的演变

另一方面，反向动力学是一个更高层次的方法。在这种技术中，动画制作者可能需要进行某些如下的规定：慢慢地从点A走到点B。反向动力学技术给出对该结构所有部件的精确脚本，以便使整个物体按所希望的动作执行。更准确地说，反向动力学意味着只定义所希望的结构端点的位置。动画制作者并不指示关节结构中的每一个分离的部件是如何移动的，而只是指出它以所需的方式移动的端点。这一思想来自机器人学，在机器人学中大多数情况下希望一个机器人手臂的端效应器取精确的位置，并执行某些动作。因此，反向动力学得出这样的看法，即在结构中的所有其他关节都必须牢固，以便使端效应器按要求定位。但是，问题正出在这里。随着关节结构变得越来越复杂，反向动力学的解也就越来越难于求出。而且一般来讲，反向动力学也没有给动画制作者很多空间来设计“角色”的运动，因为这毕竟是动

画的艺术。反向动力学是以一个黑箱的形式工作的，向其中输入的是所希望的结构端点的运动，而反向动力学方法控制整个结构的详细运动。动画制作者使角色运动，而前向动力学从这个角度来讲就更灵活。但是，如果所处理的是一个复杂模型的话，则花费的代价会很大。图17-15（又见彩色插图）所示是一个简单的角色，它正在执行某种“华丽的”步态，该步态是用前向动力学进行动画的。

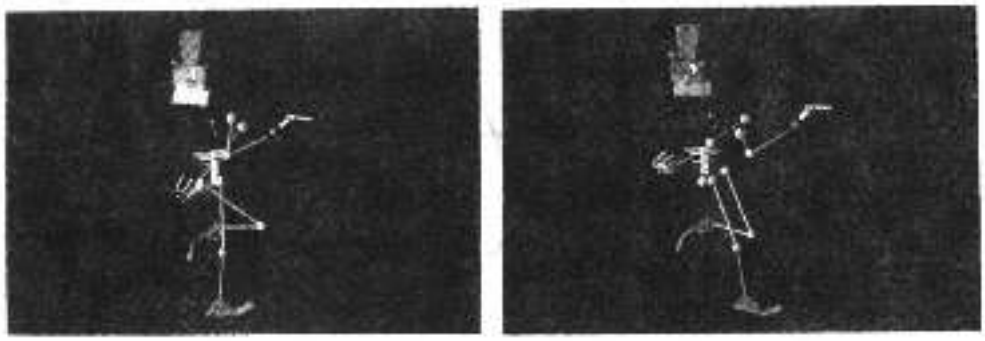


图17-15 使用关节结构的简单特性——用前向动力学对华丽步态进行动画

于是，我们有了两种“正规的”方法来脚本化一个关节结构。反向动力学允许通过列出层次结构中端点的连续位置来给出一个脚本，即把手或脚的位置作为一个时间的函数。但是，完整结构的行为方式是用于解反向动力学方程的方法的一个函数，动画制作者对于该结构的“全局”行为没有控制权。换句话说，如果结构是复杂的，则不可能得到一个反向动力学的解。另一方面，前向动力学可以对复杂的结构进行明确的脚本，但是需要以大量的人力为代价，除非是非常简单的结构。必须通过从层次结构的顶部开始向下工作来进行所有的细化工作。

496
?
497

现在用一个例子对前向动力学和反向动力学之间的区别进行更正式的描述。这是一个可能有的最简单的关节结构——一个两连杆的机器，其中一个连杆被固定，每一个连杆都在纸平面中移动（见图17-16）。在前向动力学中明确地定义了所有关节的运动，所有的关节都是互连的，端效应器（在一个动物图中为手和脚）的运动由传达到端效应器的所有变换的累积来确定。

$$X = f(\Theta)$$

其中X是端效应器的运动，Θ是一个定义位置、方向以及系统中所有关节旋转的状态向量。在简单的两连杆机制的情况下，有：

$$X = (l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2), l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)) \tag{17-2}$$

但是，这个表达式不适用于对像这样一种手臂控制和动画的场景，而用前向动力学，可以简单地定义：

$$\Theta = (\theta_1, \theta_2)$$

可以对其应用此模型，两个角度将导致移动X。

在反向动力学中，可以定义端效应器的位置，此算法在已知X的情况下必须估算所需

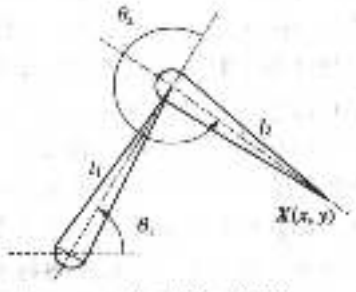


图17-16 一个两连杆的结构

的Θ。我们有:

$$\Theta = f^{-1}(X)$$

在我们的简单例子中，可以从三角函数学中得到:

$$\theta_2 = \frac{\cos^{-1}(x^2 + y^2 - l_1^2 - l_2^2)}{2l_1 l_2}$$

$$\theta_1 = \tan^{-1}\left(\frac{-(l_2 \sin\theta_2)x + (l_1 + l_2\cos\theta_2)y}{(l_2 \sin\theta_2)y + (l_1 + l_2\cos\theta_2)x}\right)$$

现在，随着结构复杂性的增加，反向动力学的解就变得越来越困难。很快，就发展成为有很多构造都满足所需的端效应器的运动。例如，在简单的两连杆机制中，很容易看出，有两个连杆的构造对于每一个位置X都是可能的，一个连杆的内关节在端效应器之上，另一个在 498
其下。这种机制的状态由两个角度来定义（自由度），我们可以很容易地从这里预见到，随着结构越来越复杂，要导出$\Theta = f^{-1}(X)$形式的表示，其难度相应增加。因此，对于用前向动力学的动画制作者来说，必须要处理越来越多的变换，而在反向动力学中，除非是对于相对简单的机制，否则，要得到解也是不可能的。一个人体具有200多个自由度。而对于这样一种情况用反向动力学几乎是不可能的，而前向动力学的脚本又非常复杂。一种解决方法是建立一些模型，对于像行走、跑、抓等普通的姿态预先写出前向动力学的脚本，然后由动画制作者通过将预先写出的序列放在一起来建立一个脚本。

在《侏罗纪公园》中对恐龙的动画中，并没有采用这两种技术。在高效新技术的以时间为上的传统方法中，出现了比关节计算机图形学动画的文献中所记录的简单得多的答案。他们的方法是用一个低层次的前向动力学脚本驱动模型，但通过半自动化地创建一个脚本而绕过了脚本复杂性的问题。他们有效地使得停止-运动动画制作者能够把其专家经验直接输入计算机中。停止-运动动画制作者以正常的方式移动他们的模型，计算机对运动采样，产生一个计算机模型的脚本。ILM用下面的方式来描述他们的技术:

> 该系统是精确的、快速的、紧凑的，并且易于使用。它让传统的停止-运动动画制作者在计算机上产生动画，但他们不必去学习复杂的软件。其工作环境与传统的工作环境非常相似，但是没有灯光、摄像机和易碎的泡沫塑料的皮肤等这些恼人的东西。所产生的动画缺乏停止-运动动画的人工痕迹，但是仍然保持了一些故意保留的细节和计算机动画中经常缺乏的硬停止。

其总的思想并不是原创的。很多年以来就可以通过由操作员握住机器人的手，并拉着它进行各种动作来训练工业机器人，使机器人最终能够代替人类的操作员来执行各种动作。汽车工业中的点焊和喷漆是这个技术应用的一个好例子。然后，机器人的关节结构中的所有运动通过传感器读入，从这些读数产生控制机器人的一个脚本。这样一来，一个任务当中的动作序列就可以无休止地并完美地重复。事实上，机器人将继续完美地重复产生这个序列，即使其他地方出现问题，或者在没有汽车存在的情况下也是如此。

在《侏罗纪公园》中，机器人已经是可以利用的了，因为停止-运动动画制作者已经在由停止-运动动画制作技术产生的预先生产的胶片中建立起了“动画”的模型。然后，由停止-运动动画制作者以相反的顺序使用这些模型，来产生对计算机模型的一个脚本。如图17-17所示为一个停止-运动动画制作者正在创作恐龙拧汽车场景的运动。这时的模型没有服装，也不 499
是一次一帧地制作胶片，而是被转入到一个输入设备中，并从中导出一个脚本。

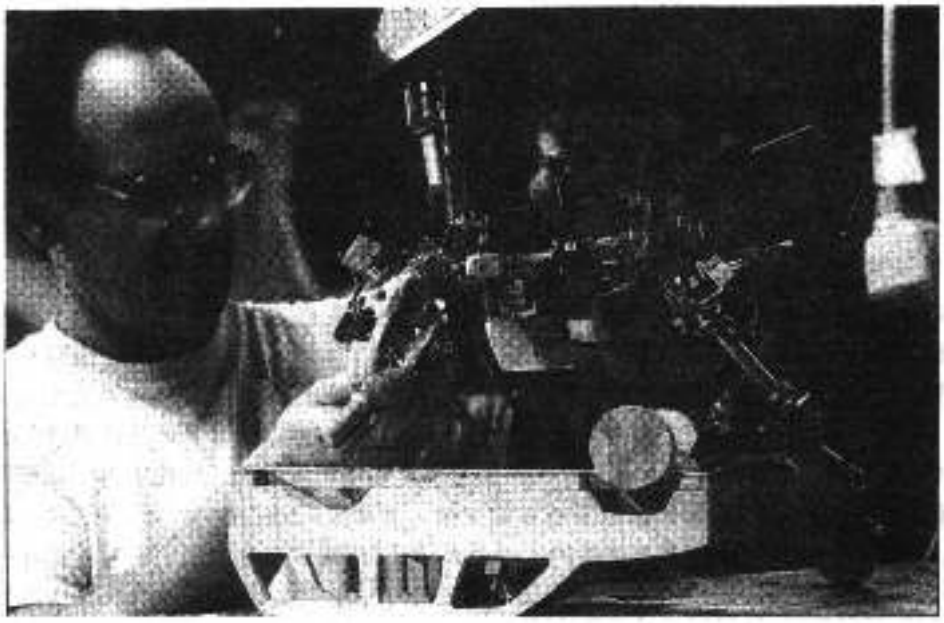


图17-17 停止-运动动画制作者正在用一个（真实的）模型固定在变换器上，并用它导出一个（虚拟）计算机模型的脚本

资料来源：Magid, R. "After Jurassic Park", *American Cinematographer* December 1993.

一种相似的方法称为"运动获取"，采用了人类的演员，以此导出对计算机模型的一个运动脚本。该方法涉及将运动跟踪设备固定到演员身体的适当部位，以这种方法由演员的真实运动导出一个动力学脚本。这种方法在视频游戏行业当中尤其流行。视频游戏行业已经完成了由二维动画向三维动画的转变。在这类交互性的计算机动画中，预先记录下的运动序列根据用户的交互事件而重新播放。在这种情况下，采用运动获取是自然的和经济的，它可以记录计算机模型的原始运动脚本，尽管这种方法的隐含缺陷是用户可以看到的动画只能是预先计算好的序列的某种结合。

因此，我们从例子中看到，对于定义复杂的关节结构的运动，这种困难的问题我们才刚刚起步，问题的很多解答都超出了计算机的范畴，并且涉及到导出一种源自真实世界的脚本（这使人想起早期迪士尼动画工作者的某些照片，可以看到他们以其自身在镜子中的图像为指南）。

解决反向动力学问题

在这一节中，我们来了解构成反向动力学基础的一个重要概念。我们将充分研究这一话题以给出所涉及困难的正确评价。在Watt and Watt（1992）中给出了一个反向动力学引擎的完整处理过程。对于这个问题的大多数方法都包括向希望目标的迭代，也就是说，在连接的角度上计算一个小的变化$\delta\theta$，这将引起端效应器向目标的移动。这个过程由下式给出：

500

$$\mathrm{d}x/\mathrm{d}\Theta = J(\Theta)$$

J就是所谓的雅可比矩阵，即对于单变量微分的一个多维扩展。在这种情况下，它把Θ的微分变化与端效应器的位置x的微分变化相关联。注意，J是结构Θ当前状态的一个函数。在反向动力学系统中遇到的普遍问题源自这样的事实，即在方程：

$$\Theta = f^{-1}(x)$$

中，函数$f()$是非线性的，并且随着连杆数的增加变得越来越复杂。这个函数的逆很快就变得不可能存在。可以通过对雅可比矩阵求逆并将结构的行为局部化为关于当前操作点的小范围运动的办法使问题线性化。

$$d\Theta = J^{-1}(\Theta)(dx)$$

目标是已知的，所以迭代过程就是由减去当前位置和目标来计算dx，代入到上述方程得到$d\Theta$并继续。过程如下:

repeat

dx:= x方向的小运动

$d\Theta$: = $J^{-1}(\Theta)(dx)$

x: = $f(\Theta + d\Theta)$

J: = $dx/d\Theta$

反向J

x:= $x + dx$

until 到达目标

对于有三个连杆的手臂的一个迭代如图17-18所示。

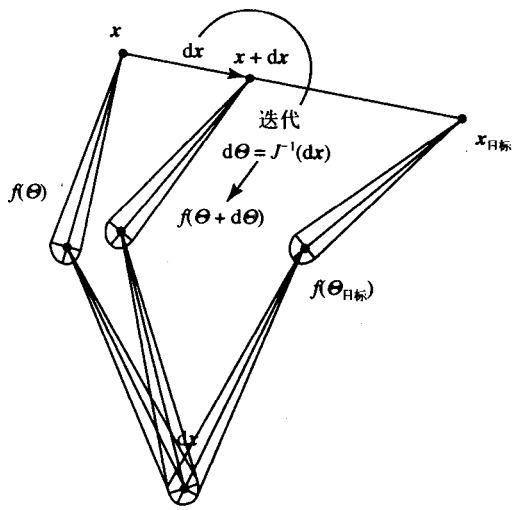


图17-18 朝向目标的一个迭代步骤

将连锁规则用于微分方程（17-2），对于两连杆的手臂的雅可比矩阵由下式给出:

$$J = \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

现在讨论这个方法所造成的问题。首先，对于x的表示的复杂性使得微分过程非常难于进行，希望有一种几何的方法来确定雅可比矩阵（Watt and Watt（1992））。其次，除非雅可比矩阵是一个方矩阵，否则它是不可逆的。而大多数希望在计算机图形学的应用中使用的（骨架）结构却不是这种情况。对于这个问题的近似的解在迭代方面带来了困难。尤其是，跟踪误差即x中希望发生的改变与实际的改变是有差别的。跟踪误差由下式给出: 501

$$\|J(d\Theta) - dx\|$$

必须采用一种比上面伪码程序所给出的更智能化的迭代方法。这包括从dx开始，估计跟踪误差并减去dx，直到误差降到阈值以下为止。然而存在于任何系统中的奇异点产生另一个问题。在两个连杆的手臂的例子中，当两个连杆排成行时（$\theta_2 = 0$），θ_1和θ_2上的改变会在相同的方向上产生端效应器的运动——即垂直于（公共）连接轴的方向上的运动。这时，不可能有指向基础的运动，于是损失了一个自由度。在这种情况下，当外部的连杆摺回到与内部的连杆并排时，即$\theta_2 = \pi$，出现了另一个奇异点。这些奇异点在机器人学中称为工作空间边界奇异点，因为这些点正是在边界上出现的。两连杆手臂的工作空间，即端效应器可以到达的区域是一个中空的圆盘（$l_1 \neq l_2$），并且其内部和外部边界的周长在二维空间中形成所有点的轨迹，在这些位置出现奇异点。

当然，我们所讨论的是一个非常简单的机械结构。正如我们所知，动物的骨架要复杂得多，尤其是它们含有分支连杆。图17-19所示为一个在典型人体动画中使用的简单结构。在这个例子中结构的根本是位于两个髋关节之间的关节（它有6个自由度）。该图还显示了一种反向动力学求解的观点所产生的对结点的分类。只有一个根结点，其余结点都是根结点的子结点。底层结点和端结点定义了一个链，可以为这个链调用一个反向动力学的解。可以在骨架
502 之中的任意两个结点之间应用反向动力学，需要遵循的唯一规则是端结点位于比底层结点位置低的链的下方。反向动力学的解定义端结点和底层结点之间的所有结点的位置和方向，底层结点又称为空结点。

也可以有其他的布置。例如，Phillips and Badler（1991）将根结点定位在一只脚上，让另一只脚作为端结点，以便对当重心从一只脚移到另一只脚上并旋转时站立的动作进行动画。

然而，在机器人系统和涉及到约束的动画之间还存在着另一个主要的差别。在机器人学中，主要的约束是由自由度以及确定机器的工作空间的连接角的约束确定的。一个简单的例子是人体手指图中的连杆，因为穿过手指的腱一般不会独立地移动。另一种考虑是能量的约束是否应该考虑，以便使得反向动力学方法能够对结构的运动产生直观的令人信服的解。这意味着，几何的约束和肌肉的约束都应得到满足。可以假设动物通过使改变所需的能量最小化来对结构的几何状态产生一个变化。只满足几何约束条件可能会产生
503 一个“看起来不正确”的动画。

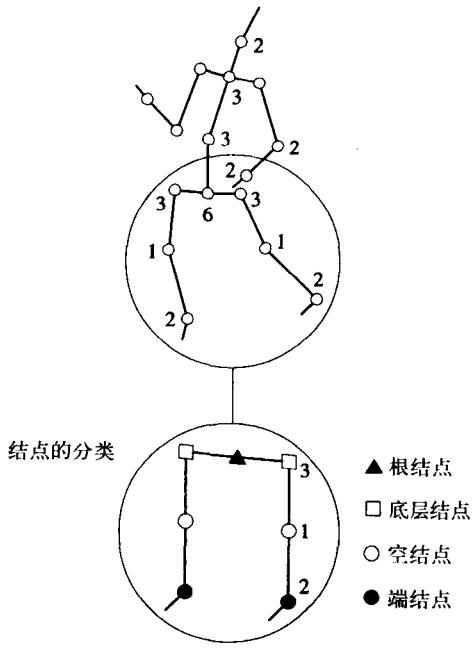


图17-19 对于人形关节结构的“最小”连杆图

17.4 计算机动画中的动力学

迄今为止所描述的计算机动画的方法一直是动力学，也就是它们涉及到对运动的定义而不是考虑我们正试图模拟的物理环境中所涉及到的质量和力。在这一节中，我们将考虑如何大体上写出模拟场景中的力的程序，并通过牛顿力学“自动”地产生出所希望的运动。这种

方法称为动力学模拟或者基于物理的动画。

《Luxo Jr.》（图17-20，彩色插图）是1987年由Pixar公司的John Lasseter制作的一个动画短片，它可能是第一个让人能感觉出其运动和感染力在质量上可以与由传统的动画制作者制作的产品相比较的计算机图形学动画。John Lasseter利用其技巧渲染了一个台灯，使它具有类似迪士尼卡通式的拟人行为。《Luxo Jr.》中的运动是由关键帧产生的，动画制作者在关键帧上以一种关键点的形式定义关节结构的状态，并把结构的全局运动定义成一个样条曲线。尽管这并不是100%的明确的控制，使动画制作者能够对每个帧的全部状态进行定义，但是动画制作者还是有很高的控制度。事实上，Lasseter在SIGGRAPH' 87会议上的论文“将传统的动画应用于3D计算机动画的原理”就强化了这个观点。

1988年Witkin和Kass发表了一篇文章（Witkin and Kass 1988），在这篇文章中他们展示了一种基于动力学模拟的高级运动控制技术，可以用于《Luxo Jr.》的动画，他们对其工作的动机做了如下的评论：

> 尽管《Luxo Jr.》向我们表明动画制作者的队伍、关键帧系统以及绘制程序可以是一个强大的整体，但是定义运动的责任仍然几乎完全要归于动画制作者。动画中的某些方面，比如说个性和感染力可能要靠动画制作者的艺术修养和技巧来处理，而这需要很长时间来获得。然而，动画的很多原理都是关于使角色的动作从基本的机械层次上看起来是真实的。此外，对于运动目标或者物理模型的简单改变为基本的运动带来一些有趣的变化。例如，对《Luxo Jr.》的质量进行加倍（或者四倍）产生了有趣的夸张的动作，这时的基座看起来很重。

换句话说，他们认为这一类的计算机动画可以受惠于高级的运动控制。除了对于中间帧执行基本的插值之外，还可以将一个程序设计成解释像“从A跳到B”的脚本。动力学模拟将产生精确的因此是真实的运动。因此我们看到，在计算机动画当中运用动力学的动机是在某种上下文中写出控制运动的微分方程比直接地定义运动或由关键帧定义运动要容易。我们还假设，如果能正确地建立起物理模拟，则随后的运动要比由一个运动学系统所产生的运动更“自然”。把这些明显的优点与采用动力学的缺点相比较，动力学系统的环境易于建立（粒子系统），但这个环境对于大多数我们所感兴趣的动画环境来说太简单了，而复杂的交互式的环境就更难于定义了。另一个问题是，对于这类系统的解从计算上来看是集约的。 504

动力学模拟不可能对很多动画应用提供一个完整的解，还存在总体的艺术控制的问题。将动力学模拟与模仿传统的动画技术的计算机方法相比较，Cohen（1992）是这样认为的：

> 传统的动画方法为艺术家提供了很大的控制权，但是没有提供任何工具自动地创建真实的运动。而另一方面，动力学模拟可产生物理上正确的运动（在限制范围之内），但是它并不为艺术家或科学家建立所需的运动提供足够的控制手段。

17.4.1 刚体的基本理论——粒子

我们所熟悉的基本的运动定律——牛顿第二定律为：

$$\boldsymbol{F} = m\,\boldsymbol{a}$$

在一个粒子或一个质点的上下文下，这个公式是最容易理解的。$\boldsymbol{F}$和$\boldsymbol{a}$一样是一个三维的向量，$\boldsymbol{a}$是该点的加速度。质点是一个简单的抽象，可以用于模拟简单的行为。可以假设一个具有大小的刚体的行为像一个粒子，因为我们认为其质量以一个点为中心——即质心。质点在力的作用下只能进行平移。

牛顿第二定律还可以写为:

$$F = m\frac{dv}{dt} = m\frac{d^2x}{dt^2}$$

其中v为速度，x为粒子的位置。这导致一个在已知t时刻的位置时，通过积分求粒子在时刻$t + dt$的位置的方法，如下所示:

$$v(t + dt) = v(t) + \frac{F}{m}dt$$

$$x(t + dt) = x(t) + v(t)dt + \frac{1}{2}\frac{F}{m}dt^2$$

F本身是一个时间的函数，可能有多个力作用在物体上。在这种情况下，用向量相加的方法即可计算净作用力。如果物体的质量在其运动时改变，例如，一个燃烧燃料的机动车的情况，则第二定律表示为:

$$F = \frac{d(m \cdot v)}{dt}$$

505

作为一个简单的例子，考虑由一个大炮炮口出来的被发射的炮弹。这种情况可以用上面的方程进行建模。炮弹上作用的是两个力，由于重力的作用而产生的恒定的加速度以及一个与速度反向的空气阻力。这个空气阻力是速度及横截面积的平方的一个（二次）函数。可以用初始（炮口）速度和弹道斜率进行模拟。牛顿第二定律用于计算炮弹的弧状轨迹。这里我们所得到的是一个模拟，在每一个时间步长上，该程序以一个时间函数计算炮弹连续的行为。

这个基本理论只能直接应用于初始值问题，模拟的过程完全由初始条件来确定。我们可以从大炮上发射一发炮弹，然后，炮弹的弹道轨迹就完全由炮弹的初始速度、质量以及重力来确定了。然而，动画制作者可能更需要这样一个系统，由它来定义大炮于点A发射一发炮弹，该炮弹会击中位于点B处的城堡。

在初始值类型的模拟中，动画制作者在初始条件确定之后就没有了控制权。换句话说，需要对问题提供约束。正是通过这些约束使动画制作者能够设计一个希望的动作。任何有潜在用途的系统都必须具有有效的可以提供真实运动的物理模型，并且同时有约束条件，使动画制作者能够获得所希望的总体运动。这些约束被称为空间-时间约束，与这些约束同样重要的是如碰撞响应这一类的问题，其包含了比物理定律的应用更难的动力学模拟。我们将在后面再返回来讨论这一问题。

17.4.2 力的性质

只有在非常简单的情况下，才能把一个物体看成是一个质点，或者是一个块状物质，该物体在一个力的作用下加速。在建模的环境中一个物体的移动方式依赖于模型本身、其约束条件以及力的性质。在基于物理的动画中所用的不同类型的力的普通例子为:

- 重力（我们已经讨论过这个力）加速度: 是一个作用在物体上的恒定向下的力，正比于物体的质量，并作用于质心。
- 阻尼力: 这是一个反向的力，正比于物体的速度，阻止物体的运动。阻尼力从物体上消除能量并以热的形式耗散。粘性阻尼力线性正比于速度，一个二次力正比于速度的平方。

空气阻力近似为二次的，如果忽略掉由于空气的扰动而产生的作用的话。

- 弹性弹簧：这些弹簧可以连接两个物体，其弹力正比于弹簧长度的位移（Hooke定律）。 506
- 物体可以具有几何限制，这些限制可以被认为是一个（限制）力。一个简单的钟摆表现出一种“固定点”的限制。不论施加于物体上的力如何这些力都会保持几何限制。不论施加的保持钟摆运动的力在何处，钟摆上的点都只能沿着一个圆的圆周移动。

17.4.3 刚体——有翼展的物质

当一个动力应用于具有大小的物体时，所引入的运动可以由平移和旋转组成。很显然，一个不受限的又长又薄的圆柱将在有一个力施加于除了其质心的其他位置时在空间中倒下。我们把这样的力描述为力矩τ。现在，把任何动因都看成是一个力F（或者一些力，其和产生F），其作用于质心并导致一个运动，就好像它是一个粒子，而力矩τ导致了关于一个特定轴的旋转。物体在空间中平移和旋转。

考虑对于一个对称的物体在其质心两侧相等的距离处施加两个相等的力。在没有几何约束的情况下物体进行平移。如果这时移动施加力的点，则物体既有平移也有旋转。从这个例子可见，尽管在两种情况下对物体施加了相同的净力，但在旋转的情况下，物体在时刻$t + \mathrm{d}t$获得了较高的动能，因为它既获得了线速度也获得了角速度。

把这种直观的运动结合到物理定律中，首先需要定义一个物体的质量是如何围绕着质心进行分布的。对于对称的物体，是通过三个转动惯量来定义的，每一个轴有一个转动惯量。将刚体看成是一个无限小的粒子，其转动惯量刚好是每一个粒子的质量之和，其权重为对于每一个轴的垂直距离的平方。

$$I_x = \int (y^2 + z^2)\mathrm{d}m$$

$$I_y = \int (x^2 + z^2)\mathrm{d}m$$

$$I_z = \int (x^2 + y^2)\mathrm{d}m$$

对于均匀的对称刚体，转动惯量的例子为：

- 半径为R，质量为m的一个球：

$$I_x = I_y = I_z = \frac{2}{5}mR^2$$

- 半径为R，高度为h的一个圆柱体，其z轴与圆柱体的长轴相重合：

$$I_x = I_y = \frac{1}{4}m(R^2 + \frac{1}{3}h^2)$$ 507

$$I_z = \frac{1}{2}mR^2$$

- 边长分别为a, b, c的矩形的盒子，其x, y, z轴分别沿着a, b, c的侧面：

$$I_x = \frac{1}{12}m(a^2 + b^2)$$

$$I_y = \frac{1}{12}m(a^2 + c^2)$$

$$I_z = \frac{1}{12}m(b^2 + c^2)$$

有关盒子的公式是有用的，因为对于物体，其通常的近似是限定盒。质量m由某个默认的密度乘以体积给出，质心是盒的中心。如果用限定体来计算一个多面体的转动惯量，则其近似程度依赖于这个限定体与物体匹配的紧密程度。

如果物体不是对称的，则对于一个动态模型，为了对其质量分布有一个完整的描述还需要惯性的乘积，即:

$$I_{xy} = \int xy \, \mathrm{d}m$$

$$I_{xz} = \int xz \, \mathrm{d}m$$

$$I_{yz} = \int yz \, \mathrm{d}m$$

所有这些值都被排列表示成3 × 3的矩阵，称其为惯性张量:

$$\boldsymbol{I} = \begin{bmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{xy} & I_y & -I_{yz} \\ -I_{xz} & -I_{yz} & I_z \end{bmatrix}$$

如果积分是在一个严格贴附于物体并随着其移动的“物体”坐标系中进行的，则$\boldsymbol{I}$是常数。这使得$\boldsymbol{I}$是一个对物体的不变的描述，只被计算一次。对于任意形状的物体，有一个使$\boldsymbol{I}$为对角化矩阵的特殊坐标系（即惯性的乘积为零）。这个坐标系称为主轴，并把$\boldsymbol{I}$的对角线元素称为主转动惯量。因而选择这个坐标系作为物体的坐标系很有意义。为了求出主轴，把惯性张量旋转变换到一个使其为对角化的框架中。通过求特征向量$\boldsymbol{E}$和相关联的特征值来达到这一目的。

刚体的运动可以由两个运动的和来表示。回忆一下，我们考虑了两个坐标系：一个固定的坐标系和一个移动的坐标系，后者嵌入物体中并参与其运动。这个坐标系习惯上被定位在物体的质心。下面关于汽车的例子是对这种需求的一个好的展示。大多数的力，比如偏转，
508 即由于旋转运动的角加速度而产生的力是在一个固定于汽车上的坐标系中被处理的。同时，还需要一个地面坐标系，汽车在这个坐标系中移动，可以用来处理诸如驾驶的运动学问题。

积分模型中运用的一个无限小的偏移量可以被描述为物体从其在时刻t的位置到其在时刻$t + \mathrm{d}t$的位置的一个平移再加上一个绕着其质心的旋转，这个旋转对物体关于其质心的最终位置进行定向。这就使得刚体成为一个具有六个自由度的系统。

可以采用欧拉方法以六个方程来描述运动及其结果。其中的三个方程是平移分量的运动平移方程:

$$\boldsymbol{F} = m\boldsymbol{a}$$

另三个方程为运动旋转方程，它把角加速度与质量扭矩相关联（上述方程的旋转的类似）。具有一个对角的惯性张量的物体的旋转动力学由下式给出:

$$\boldsymbol{\tau} = \frac{\mathrm{d}\boldsymbol{H}}{\mathrm{d}t}$$

$$\boldsymbol{H} = \boldsymbol{I}\boldsymbol{\omega}$$

其中$\boldsymbol{H}$是物体的角动量，τ是所应用的扭矩，ω是角速度。这两个方程导致了一组简化的欧拉方程，假设惯性的乘积为零。

$$\tau_x = I_x \frac{d\omega_x}{dt} + (I_z - I_y)\omega_z\omega_y$$

$$\tau_y = I_y \frac{d\omega_y}{dt} + (I_x - I_z)\omega_x\omega_z$$

$$\tau_z = I_z \frac{d\omega_z}{dt} + (I_y - I_x)\omega_y\omega_x$$

其中ω为角速度，它相应于参考的局部帧。

这些方程也可以如前面那样进行数值积分。

现在，可以建立起一个完整的模拟，这个模拟由计算物体的不变的物理特性——质心、惯性张量以及主轴组成。将所施加的扭矩和力求和。这些力是由一个三维向量以及一个作用力的点的位置向量定义的。净力由对力的向量求和得到（不考虑力的作用点）。净扭矩由对力的作用点处力的分量产生的扭矩求和来求出。这样就给出了六个方程（三个扭矩方程，三个力的方程）。一个物体的动力学状态（线性和角速度以及在时刻$t + dt$的位置和方向）从其在时刻t的状态开始计算，在大多数应用中，dt将等于帧间隔。如果在场景中有多个物体，则可能会出现碰撞——这个话题在第17.5节中讨论。 509

17.4.4 在计算机动画中运用动力学

前面两节所述的内容在物理学或动力学的教科书中都可以找到，它构成了任何计算机模拟的绝对基础。那么，在实际情况下应如何应用动力学呢？扩展上述内容最简便的方法之一就是建立由（比如）成千上万个粒子组成的系统，并且用这些粒子来模仿像喷泉中的流水或者烟火的行为这样的现象。尽管每一个粒子都会遵循上面所述的简单的运动定律，但众多粒子的总的效果是模仿“流动”物体的类型。这种模拟方法称为粒子系统，在第17.7节中介绍。

更一般地，可以很有效地应用动力学模拟的动画环境具有更复杂的结构。可以把基本的模拟看成是将一个刚体的运动合并为一个黑箱。对于这个引擎的输入是驱动运动的力和扭矩。但是它们应如何计算或导出呢？上面引入的简单的初值问题对于大多数实际的动画一般来说是不够的。动画制作者不仅仅关心初始点，而且关心物体结束其运动的位置以及它到达这个位置所经过的路径。这个问题可以叙述为：求出引起物体从一个初始位置沿着某一个路径到达一个终止位置所必需的力和扭矩，而这个问题还可能遵从于其他的限制条件，比如，在执行运动时用最小的能量或者到达目的地所需的时间。这个问题对于有任意复杂度的物体是困难的，也是特别需要深思熟虑的。例如，动物或人类的运动是一个流行的应用，在这种情况下，物体结构的复杂的几何限制和肌肉的动力学限制都必须得到满足。此外还存在着能量的限制，因为身体姿态上的改变通常都是按照使改变所需的能量最小的方式来进行的（这与这一类结构的运动学刚好相反，而运动学本身也是一个困难的问题）。

需要确定适当路径的一个简单例子是足球游戏。一个较高水平的处理将决定下一次射门是在球的轨迹的最高点处击中球门的顶部——横梁。这就是说，击球的初始速度和角度都必须确定下来，以使得所希望的结果出现。可以通过采用反向动力学而不是前向动力学来达到这一目的。在前向动力学中，我们向模拟器中输入力和扭矩，由模拟器输出位置。而采用反向动力学时，输入和输出是反向的，我们输入的是位置，而输出是力和扭矩。

所有这一切都意味着这些方法极为多样化，而且大多数方法在数学上也是需要的。它们的变化方式在于定义限制条件以及解的计算。许多我们希望模拟的系统都是多物体的系统，

尽管上面简述的牛顿动力学涵盖了所有的行为，但是动画制作者的困难在于确定运动学和所感兴趣系统的动力学方程。现代工程系统的多样性及其复杂性使得整个教科书可能只能专注
510 于一个系统。例如，考虑对机动车的模拟。汽车的模拟需要与飞机的模拟相差很大的一组方程。

因此，我们将论述限制在两个例子上，这两个例子表明了所涉及到的困难问题的某些原理和性质。这些方面都只讨论到足够的深度以对问题给出正确的评价。

尽管技术本身依应用及模拟性质的不同变化很大，但在离线动画和交互式实时动画之间仍然有较大的差别。前者是作为电影院中放映的全长的产品，而后者的主要应用是计算机游戏。对于计算机游戏来讲，对每一帧所分配的大多数处理资源是用于产生图像中可见的复杂情景，这就暗示着动力学的模拟必须非常简单。

17.4.5 模拟成块物质的动力学

我们从考察一个用于机动车模拟器或赛车游戏的简单方法开始，将讨论这类流行动画的三个方面：一个总体的概述，说明动力学模拟是如何植入到计算机程序中的，一个基本的模拟和一个可能需要使其更精确和更真实的扩展。这个例子将表明另一个重点，即模拟的粒度强烈地依赖于应用。一个用于汽车设计所需的机动车模拟器将比计算机游戏中看到的娱乐性的驱动模拟器所需的机动车模拟器要详细得多。而在本书的处理中，一开始将机动车考虑成一个块状的物质，它具有一个重心。然后，把整个汽车作为一个刚体来处理。

这一形式的交互式计算机动画允许有一个简单的分支结构，对于每一帧都对输入设备进行采样，根据当前状态的参数和新的输入，程序可能会进入一个新的状态。状态变量控制着绘图，并且与所建模的汽车的所有部件相关联。状态变量只含有模拟中所涉及到的所有参数的当前值。对于一个粒子来说，这些值为质量、加速度和位置。也可以引入简单的灾难性的结局。例如，如果汽车在弯道上跑得太快，那么它将翻车。一个总的结构如图17-21所示。控制模块将对输入设备进行采样，并提供一个当前的加速度或者发动机扭矩作为动力学模拟器的输入。一种简单但有效的策略是比例控制：

$$a_c = k_1(v_t - v_i)$$

其中：

v_t是由加速器踏板的位置所给出的目标速度，或者，如果用了刹车则设为零；

v_i是当前速度；

511 k_1是用于加速和刹车的依赖于机动车的常量（或变量）。

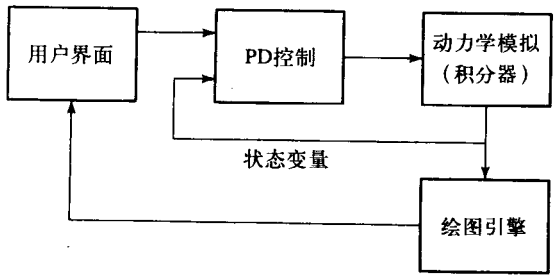


图17-21 交互式的动力学模拟中的功能成分

这个基本的控制必须用一个限制系数进行增强，因为作为一个一阶近似，加速度变量是引擎HP的一个函数，它反比于v_i，即:

$$0 < a_c < \frac{k_2}{v_i} \quad k_2 = \frac{550HP}{m}$$

其中HP为引擎的马力，m为机动车的质量。可以对驾驶建立相似的控制。

例如，程序的状态为给出当前道路条件（平路、斜坡等）的当前位置、汽车的当前姿态，即以直线行驶、转弯或（灾难性地）翻转。

一个近似的动力学模拟可能非常简单，它由一个具有与汽车车身紧密相连的坐标轴的机动车构成，换句话说，没有悬置物。这样给出的是只有三个方程的模拟，这三个方程控制了汽车的行为，并且不管它是否会翻转。这三个方程是:

$$\sum F_f = ma_c$$
$$\sum F_p = 0$$
$$\sum M = 0$$

其中:

F_f是在一个汽车所行驶的表面上施加的牵引力，行驶方向为线性加速的方向;

F_p为垂直于牵引力的力;

M代表关于物体质心的所有力的动量。

请注意，汽车是向前运动的，或者由于牵引力的作用慢下来，这个牵引力的作用为马路表面和轮胎之间的摩擦力。引擎或刹车仅仅对车轮施加动力。对于在平整的表面上以直线形式的汽车加速有（图17-22b):

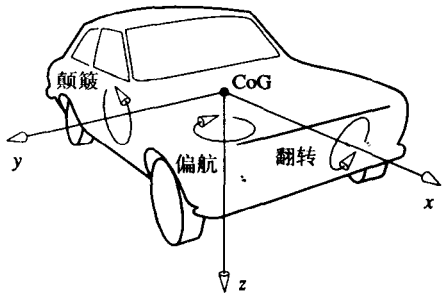
$$F_t = ma_c$$
$$mg = R_r + R_f$$
$$F_t h + R_f l_1 = R_r l_2$$

最后一个方程隐含的意思是，一个逆时针的动量把汽车的前轮拉升使其离开地面。对于一般的汽车，假设其重量使其足够保持在道路上行驶（尽管对于实际的汽车若有悬置力的话它的前端将升起来）。然而在一个拉力赛车上，由于其重心的位置关系而很容易使其前轮离开地面（$R_f = 0$）。 512

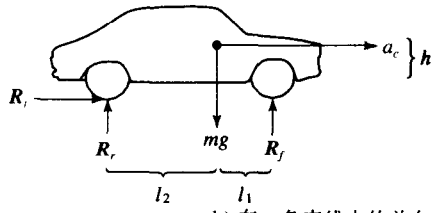
在这里需要提出的一个重要观点是不管牵引力的大小如何，我们在这个平凡的模拟中隐含地实施了一个限制，即不实施任何轮子的滑动。假设轮胎和地表面之间的摩擦系数使滑动不会出现。

现在考虑汽车行驶在表面上的一个弯道时的力（见图17-22c)。在稳态的情况下，使机动车转弯将服从于指向一个圆心的向心加速度v^2/r，该圆的周长形成弯道。这用轮胎和道路之间的横向摩擦力F_s来表征，并用前面提到的三个方程表示:

$$F_s = m\frac{v^2}{r}$$
$$mg = R_o + R_i$$
$$F_s h + R_i \frac{w}{2} = R_o \frac{w}{2}$$

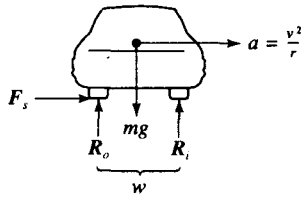


a) 机动车坐标系（美国汽车工程师协会）



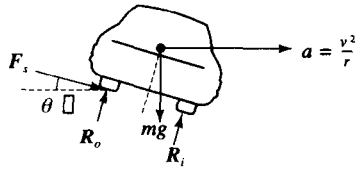
$$\sum M = 0$$
$$F_i h + R_f l_1 = R_r l_2$$

b) 在一条直线上的前向运动



$$\sum M = 0$$
$$F_s h + R_i \frac{w}{2} = R_o \frac{w}{2}$$

c) 稳态转弯



与c比较
F_s减小
R_i和R_o增加

d) 稳态的堤岸式的转弯，与c比较F_s降低，R_i和R_o增加

图17-22 一个机动车的准静态计算的力

这就是说R_o大于R_i，如果v和r能使得R_i趋向于零，则汽车翻转。

当然，倾斜的道路将使情况得到改善。确实，这也是其存在的理由，现在我们有（见图17-22d）：

$$F_s \cos\theta = m\frac{v^2}{r}$$
$$mg + F_s \sin\theta = (R_o + R_i)\cos\theta$$
$$F_s h + R_i \frac{w}{2} = R_o \frac{w}{2}$$

斜坡的作用是减小F_s并增加两个车轮的作用力，于是也就减小了侧翻的倾向。这时的作用力有一个水平的分量，它辅助产生所需的减小摩擦力的加速度。

现在返回到前面的问题，即粒度问题，以及依赖于粒度应用的模拟的精确性问题。在这

一情况下考察上面的模拟的缺陷是有指导意义的。

考虑翻转的计算。在实际中，汽车将有一种悬挂力，在转弯时，这个悬挂力造成了悬浮质量重心的横向偏移。在这种情况下，悬挂特性和转动速率（关于x轴的角速度）必须加以考虑。此外，即使考虑到了这些因素，计算也是“准静态”的：假设汽车是稳态地转动，没有关于穿过重心的垂直的z轴上的角加速度。例如，准静态计算不能建模内侧轮子离开地面以及由于悬挂力分量之间的相互作用而产生的折回的瞬间行为。对于当前的条件，这些只是确定了翻转的阈值。于是，在这样的模拟中，驾驶员一旦达到了这个阈值，就没有恢复的可能性了，此时汽车只有翻车。实际过程中，这一类的翻车对于家用汽车而言几乎不会出现，因为驾驶员不会使汽车进入一种会引起汽车翻车的稳态转弯状态。驾驶员在行驶中总是进行瞬间调节，当汽车滑行或撞上一个障碍物时家用轿车通常会出现翻车的情况，这是一种所谓的被绊倒的翻车。这种情况非常难于建模。 513 ~ 514

除了不执行悬挂之外，另一个关键的因素是轮胎的物理学，以及轮胎-道路表面的相互作用。汽车的整体表现由四个轮胎与道路表面之间的接触力来确定。如果机动车不在直线上行驶，则所建立起来的侧向力将引起轮胎滑动或打滑。轮胎的性质、温度、齿轮以及轮胎的充气压力将随着汽车的行驶而变化，当然这些对于F1汽车拉力赛是至关重要的。

空气动力学的作用作为一个正比于速度平方的简单的刹车函数尽管在低速下是易于建模的，但是这个作用在高速下就变得越来越复杂，因为这时需要对汽车的形状加以考虑。

因此，这个简单的例子表明，精确的动力学模拟可能是非常复杂的，它实际上是机械工程设计的一部分，而不是计算机图形学的任务。未来的动画制作者通常根据建模状况和建模的精确度来采用一个简化的模拟。

17.4.6 空间-时间限制

在前面一节中，我们给出了一个动力学模拟的简单例子。这一类模拟称为初始值问题，或者前向模拟。用户输入初始值（虽然在计算机游戏中是连续地输入），然后，其运动由用于执行模拟的方程完全确定。这对于上面给出的例子是合适的，但是，可以肯定地说，对于大多数的离线动画，还需要给动画制作者更大的控制权。动画制作者更像是为一个动力学模拟规定动作，这使得其变成一个两点边界值问题。这样的问题非常难以解决，在这样的动画中，动画制作者希望规定游戏者踢到了球、在战斗中基于物理原理进行运动以及在某一特定点上着陆。由于计算上的需求，这种模拟当前仅限于离线的。

早期经典计算机动画的成功，比如前面提到的*Luxo Jr.*，是由于动画制作者的艺术才能而创造的，他们规定并调整运动。这种产品中计算机的角色是插值和绘制。而动画制作者则保持对动作的完全控制。在这一节中，我们将描述一些重要发展的基础。这些发展的目的是使得动画制作者能够应用现实的运动，这些运动由动力学模拟导出，而不损失对系统所含初始值的使用的控制。

1988年Witkin和Kass（Witkin and Kass 1988）介绍了空间-时间限制的概念，他们用的是*Luxo Jr.*的一个（平面）模型的例子，用四个连接角和平移进行了参数化（见图17-23）。而且这些参数可以由动画制作者利用一个传统的动画系统进行控制。研究的目的是能够进行基于物理的动画，并且由所允许的动画制作者进行高水平的控制来规定： 515

- 角色必须做什么。例如“从这里跳向那里”。

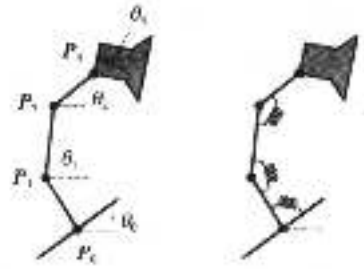
- 如何实现一个移动。例如，“不要消耗能量”或者“使劲跳下来离开你落下位置上的任何东西”。
- 角色的物理结构是什么，即各个片是什么形状，它们的重量是多少，它们是如何连接在一起的等。
- 角色要完成所希望的动作可以利用什么物理资源，例如，角色的肌肉，……要推动的一块地板等。

在对他们的方法的成功进行评论时，Withkin and Kass（1988）指出：

> 使一个Luxo灯执行一个令人信服的跳跃时只要告诉它起点和终点即可。结果表明课预期、顺势的动作、压挤、伸展以及时间安排等性质都是来自于对运动目的的描述及其发生的物理上下文。

这种运动的细节在20世纪30年代由迪士尼动画制作者作为其艺术的组成部分建立起来，它们具有John Lasseter 以手工建立的原始动画的精确效果。这种方法的潜在用途不仅可以使动画制作者能对基于物理的动画进行控制，而且可造成一种更高层次的运动控制而让物理模型专注于其自身的运动细节。

这类问题的解是五个运动曲线，一个用于平移，四个用于连接角。这些曲线是时间和由动画制作者给出的位置限制的一个函数，或者由某些较高层次的脚本控制，因此将其称为“时间-空间限制”。



a) 一个由刚性连接和无摩擦连接构成的平面模型



b) 一个跳跃的模拟

516 图17-23 《Luxo Jr.》的Wirkin和Kass模拟（基于Witkin and Kass（1988）的一个示意图）

通过引入一个待最小化的目标函数及限定条件可以得到一个解。该限定条件在这种情况下由初始条件，最终姿态以及位置组成。于是解算法的框架为求出符合限定条件的使目标函数最小化的连接角度运动曲线的集合。系统中的力为“肌肉”力以及地板与基础和重力之间的接触力。肌肉由每个点上连接连杆的三个角弹簧模拟，其中的弹簧力由下式给出：

$$F_i = k_i(\varphi_i - \rho_i)$$

其中：

k_i为弹簧的硬度；

φ_i是连接角;

ρ_i是支撑角。

允许硬度和支撑角发生变化。用弹簧来建立目标函数，该函数通过确保在每一个时间步长上肌肉所消耗的能量最小而优化运动的机械效率，而肌肉的能量是肌肉力与连接的角速度的乘积。

这个例子表明了这个方法的潜力和它的局限性。从本质上来看这是一个简单的模型。问题复杂性的增长对于我们想要进行动画的“生物”来说是严重的问题，近期的关于空间-时间限制的研究已经集中于解决这个问题了（例如，见Liu et al.（1994））。

17.5 碰撞检测

任何允许有一个以上的运动物体存在的动画系统的命令部分都是碰撞检测。对碰撞检测已经进行了许多研究，其代价是在场景中移动物体的个数及其复杂性（每个物体平均的表面个数）的一个二次函数。标准的碰撞检测方法是一个“非限制性阶段”接着一个“限制性阶段”。在非限制性阶段求出可能发生碰撞的物体对，而在限制性阶段确定准确的碰撞点（如果有的话）。一个基本的非限制性阶段算法的复杂性为$O(n^2)$。在这个过程中，对于多面体的一次准确的碰撞计算在典型情况下其复杂度为$O(m^2)$，其中m为每个物体的表面个数。

可以把这个算法与一个光线跟踪相交测试进行比较，后者是一个简单得多的碰撞检测问题。在这里，等价的非限制性阶段检查一个已知物体，即当前光线与物体的限定体之间的相交，其复杂性为$O(n)$。在光线跟踪的限制性阶段，即求光线与多边形之间的相交，其复杂性为$O(m)$。碰撞检测与光线跟踪之间的另一个关联是对于限定体的大量使用。当检查一对限定的物体之间的碰撞时，首先启动一个（快速的）物体的限定体之间的相交测试。如果限定体 517
之间不碰撞，则物体也不可能碰撞。限定体检查可能是两阶段算法（将在下一节中描述）中的非限制性阶段中最流行的算法。

计算机上消耗的另一个来源是Hubbard（1993）所谓的“固定时间步长的缺陷”，再加上上面提到的“所有的配对的缺陷”。固定时间步长的缺陷只是所有配对的算法在等时间间距情况下的应用。这是一种浪费的方法，除非环境在每一个时间间隔上都产生一个碰撞。我们所需要的是反比于碰撞或然率的采样间隔。

除了浪费之外，对于时间采样来说还存在与碰撞检测相关的其他困难。如果所模拟的物体速度相对于碰撞检测的采样速率要高的话，则物体可以互相穿越而不会检测到碰撞。一般而言，一个碰撞在发生瞬时冲击力时是不能被检测到的，但是在这之后的某个时间，当碰撞的物体互相移入到对方的空间时就可以检测到了。于是，确定冲击点的某些策略就是必要的，比如说将其中的一个物体沿着其路径往回移动。

17.5.1 非限制性阶段/限制性阶段算法

这是在现阶段最常见的算法。这种结构试图消除不可能碰撞的移动物体对，把真正的碰撞检测应用于消除过程中保留下来的那些物体对。这一策略的一个重要优点是对于每个阶段的算法选择可以独立进行。在这一节中，将简要地介绍非限制性阶段的策略。

有很多非限制性阶段的策略存在，比如利用时间连贯性、空间连贯性以及限定体，最可能的是限定体或者限定体的层次结构。

利用时间连贯性的一个直接方法是由Hubbard（1993）给出的，在他的方法中与一个物体相关联的是四维的空间-时间限定体。Hubbard表明，四维物体是由一个三维的移动物体延伸出来的，它以任何动作从一个起点开始运动，是一个“抛物线的角”。为了简化相交测试，他将这个运动用一个四维的不规则四边形来限定。非限制性阶段是基于计算最早的可能在任何物体对之间出现的碰撞时间、在达到该时间之前不做任何进一步的碰撞检测（这一工作的动机是对于必须在实时情况下起作用的VR的碰撞检测）。如果有两个物体要在未来的某一时刻t发生碰撞，则它们的空间-时间限定必须在某一时刻$t' \leq t$时相交，因为在四维空间中空间-时
518 间的限定是守恒的，正像三维空间中的限定体那样。检测算法计算对于所有物体对的最早的t'。从定义上来看，这一方法消除了恒定的时间步长的缺陷。Hubbard还强调了相交测试中的所有物体对缺陷。如果物体的路径是已知的，则可以用一个较简单的限定体在一个时间间隔中限定物体所占据的空间。关于物体路径的预备知识并不是交互式计算机动画的性质，这个方法只适合离线的动画。

空间连贯性是最易于利用的，这通过把场景空间分割成单位单元来实现。对于使运动的物体保持在地面上运动的交互式计算机动画，单元就成了一个二维的网格。通过检查一个单元，看其是否含有一个以上的物体来检查碰撞。这种简单方法的问题是对于一个单元尺寸的优化选择，而且这种算法只适用于那种所有的物体都或多或少尺寸相同的环境。

对于均匀的空间细分的显著增强是使用一种分叉树分区（第1章和第12章）。为了检查潜在的碰撞物体对，树是降序的，而且只有那些含有多个物体的区域才被检查。事实上，八叉树消除了对于那些互相之间离开有一定距离的物体对的测试。然而，与第12章中的应用不同，由于物体是移动的，八叉树必须在每一个时间步长上进行更新。这可能会导致大量超额的计算。

另一种可能的方法是为每一个物体保留一个基于某种距离阈值的最近邻物体的列表。这种策略还可以允许有一种自适应的时间步长。Cohen等（1995）通过对与限定盒标定的轴（AABB）进行排序来保留潜在的非常接近的碰撞物体对。

在非限定性阶段碰撞检测中的限定盒的利用非常普遍，主要使用三种体，即球、AABB（与坐标轴对准的限定盒）以及OBB（其方向与它们所限定的物体最合适的限定盒）。通过考虑把顶点作为一组点，应用一种称为主元素分析（其详细的细节由Gottschalk等（1996）给出）的多变量统计技术可以对一个特定的物体建立OBB。

17.5.2 用OBB进行非限制性阶段的碰撞检测

尽管用AABB进行重叠检查涉及直接的一维限定检查，但据报道用OBB进行重叠检查更快一些（Gottschalk等（1996）），现在将介绍它是如何工作的。

对于两个OBB之间互相干扰的检查的一个基本算法应该需要一个边-面测试，每一个边-面测试导致$12 \times 6 \times 2 = 144$次的边-面测试。Gottschalk等建立了下面的策略。首先，考虑图17-24，它表明两个OBB投影到一个（任意的）轴上。从这个例子中可以清楚地看到，如果投影顶点的间隔不重叠，则盒就不会重叠。这样这个轴就是一个分离的轴，因为它的方向是为了从不相交的OBB产生不相交的间隔而设定的。如果这些间隔是重叠的，则OBB可能是也可
519 能不是不相交的，这就需要进行进一步的测试。

现在考虑如果OBB是不相交的，则它们可以（可能）由一个含有它们的表面之一的平面进行分离。不太明显的是，如果它们是不相交的，但是又不能用一个平面将其分开，则它们

可以用平行于来自每个盒的一个边的平面分开。这意味着如果存在着一个分离轴的话，则OBB是不相交的。这个分离轴正交于OBB的某个面或正交于每个面的一个边。因此，为了进行分离测试只要测试15个潜在的轴就足够了（即每个盒的三个面的方向加上九个对边）。因此正干扰需要15次测试，如果OBB不相交，则确定这个不相交将取平均7.5次测试。

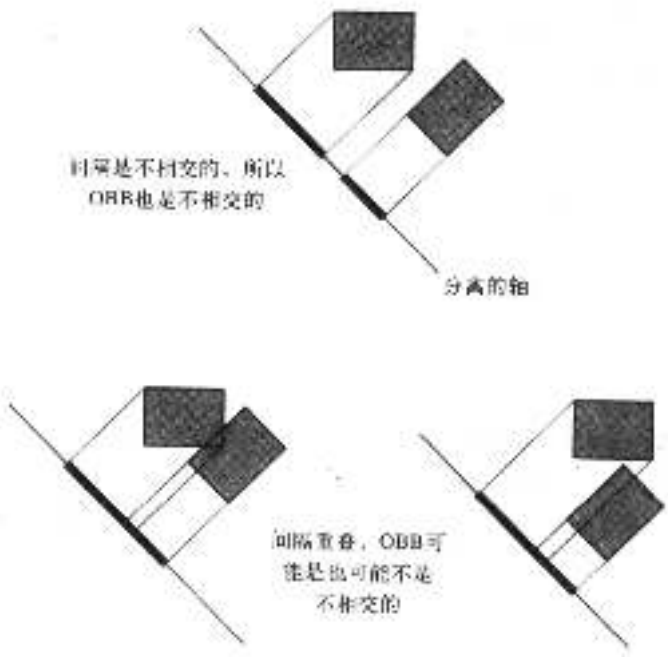


图17-24 将OBB投影到轴上。如果间隔是不相交的，则轴是一个分离的轴。检查OBB的不相交涉及到搜索一个分离的轴

测试分离轴的存在按如下方式进行。参考图17-25，注意轴的放置（而不是方向）是不占空间的，假设轴穿过盒A的中心，尽管出于清晰的考虑它被示于盒的外部。根据如我们所解释的在对比中所涉及到的盒子的几何性质，选择了至多15个L。如果盒A的轴为单位向量A_1、A_2和A_3，A的尺寸的一半为a_1、a_2和a_3，则盒的"半径"的投影长度为：

$$r_A = a_1A_1 \cdot L + a_2A_2 \cdot L + a_3A_3 \cdot L$$
$$= \sum_{i=1}^{3} |a_iA_i \cdot L|$$

对于盒B，r_B有一个类似的表示。如果D是B相对于A的平移，若下式成立，则区间是不相交的：

$$D \cdot L > \sum_{i=1}^{3} |a_iA_i \cdot L| + \sum_{i=1}^{3} |b_iB_i \cdot L|$$

或者：

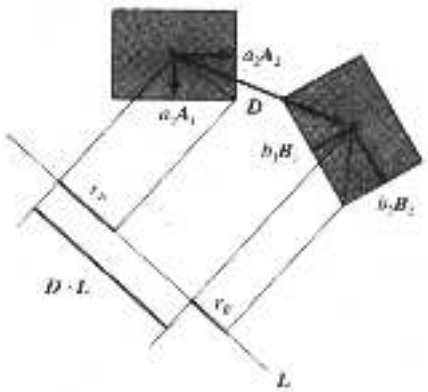


图17-25 L是一个被测试的轴，看其是否是一个分离轴。每一个OBB的"半径"被投影到L上。对于分离：$D \cdot L > r_A + r_B$

$$D-L>r_A+r_B$$

由于15个潜在的分离轴关于物体之一被进行了定义，所以这些轴随着物体移动，其自身的间隔也有一个与其相伴的速度。如果为了简化起见假设物体以恒定的速度在帧之间做直线运动，相应于物体的间隔的速度易于计算。则对于任何帧之间的时间可以求出运动的物体是否将要与一个静态的物体碰撞，该静态物体是当前进行比较的物体。

Gottschalk等（1996）通过使用一个由Weghorst等（1984）首先提出的用于分析光线跟踪中半球方法的效率的成本函数来衡量OBB的效率。即：

$$T=N_cC_c+N_pC_p$$

其中：

T为对两个由OBB树表示的大的物体互相干扰的测试总成本；

N_c为限定体对的重叠测试次数；

C_c为限定体的测试成本；

N_p为基元对的测试次数；

C_p为基元对的测试成本。

紧密限定体的思想是尽可能地降低N_p，但是这通常是以C_c为代价的。球和AABB的C_c比OBB的要快得多。碰撞检测与光线跟踪方法也是有区别的，因为一般而言我们是比较两个复杂的物体（在光线跟踪中一个物体总是一束光线）。Gottschalk等（1996）指出，总成本依赖于模型的相对位置，当它们相距较远时，球树和AABB树比OBB树成本低。他们得到的结论是，在相距很近时对于大的模型：

520
|
521

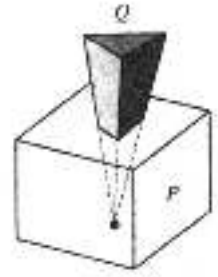
- C_c比球树和AABB树的要高出一阶。
- OBB树的N_c和N_p渐进地比球树和OBB树的两个值要低。

17.5.3 限制性阶段：凸多面体对——准确的碰撞检测

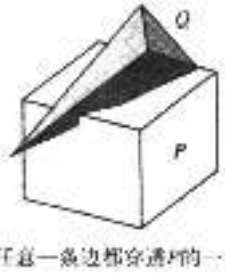
在这一节中，将介绍常见的采用物体必须是凸多面体这种约束的“直接的”精确碰撞检测算法（从原理上来看，可以将凹多面体分解成凸多面体的集合）。这个算法是由Moore and Wilhelms（1988）提出的。

应用了三种测试，它们当中的每一个成功都暗示着已经出现了一次碰撞。考虑两个多面体P和Q。首先，Q的所有顶点都被检测，看其是否被P所包含，反之亦然（见图17-26a）。其次，测试Q的边，看其是否穿透P的表面，反之亦然（见图17-26b）。最后，对不是经常出现的两个（相同的）多面体互相穿过、其表面也完全吻合的情况进行测试。这种测试通过考虑Q的每一个表面的质心，并且采用对点包含所做的同样的测试来完成。

考虑第一个测试：必须对Q的每一个顶点与P的每一个面进行检查，如果有任何顶点位于P的所有面的内侧，则检测到了一个碰撞。因此，对于Q的每一个顶点v，以及P的每一个面f，计算点积：



a) Q的任何一个顶点都包含在P中



b) Q的任意一条边都穿透P的一个表面

图17-26 对于凸多面体的碰撞检测

$$(v_i - u_i) \cdot n_i$$

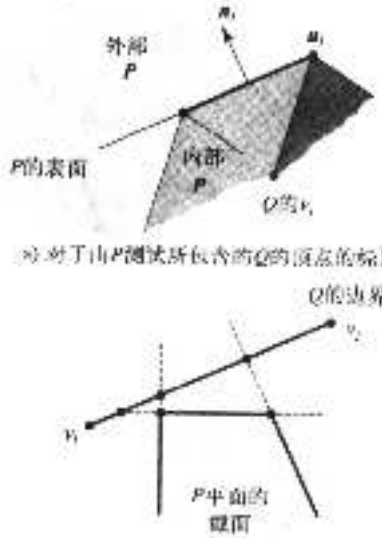
其中u_i是表面i的任一顶点，n_i是其（外向的）法线（见图17-2a）。如果这个点积是负值，则顶点v_i位于表面i的内侧。

第二个测试首先计算一个Q的边(v_i, v_j)，它是边与（无限薄的）含有P的表面的平面的相交。对于P的任何k平面，如果每一个顶点到该平面的垂直距离改变符号，则有一条边与它相交（见第1章）。相交点x可以按下式计算：

$$d_i = (v_i - u_k) \cdot n_k$$
$$d_j = (v_j - u_k) \cdot n_k$$
$$t = \frac{|d_i|}{|d_i| + |d_j|}$$
$$x = v_i + t(v_j - v_i)$$

一般而言，这组方程给出一些沿着该边界的相交点。那些$t \in [0, 1]$的点被排除，其余的点按其t值的顺序排序。这些点形成了一个从一个顶点到另一个点间潜在的相交点的序列（见图17-27b），每一个点对都是由含有相邻表面的平面形成的。最后，为了检查相交，Moore and Wilhelms（1988）用每一个对的中点来代替第一个测试中的这个值。

Cohen等（1995）介绍了一种结构更精巧的基于求出并跟踪两个多面体之间最靠近点的算法。这个算法有趣的性质是它采用了对物体的预处理，实现了执行时间不依赖于物体的复杂性。换句话说，它在精确的碰撞检测阶段而不是在不精确的预处理阶段采用了离线计算。



a) 对于由P测试所包含的Q的顶点的标记

b) 对于裁剪P的一个表面的Q的一个边的标记

图17-27 对于凸多面体碰撞测试的标记

17.5.4 单阶段算法——物体的层次结构

层次结构在碰撞检测算法中以两种不同的方式得到使用。可以将其用于表示环境的整个空间，例如用八叉树；也可以使每一个物体都关联上一个层次结构。物体层次结构的潜在优点是它可以实现一种统一化的方法，即可以建立一个单阶段的算法，它检测碰撞的精确程度依赖于我们从层次结构向下走多远。这就使得Hubbard（1996）所谓的时间关键性的碰撞检测成为可能。在这种检测中，交互式的计算机动画中的精确度是由可用的计算时间所确定的。当然，这一物体表面层次化的表示方法与第2章中所介绍的细化方法是相同的。然而，在这种情况下，一个由多边形构成的LoD层次结构对于碰撞检测是不方便的。Hubbard（1996）使用了一种球树进行快速的相交判断。一个球状层次结构的最高层是物体的限定球。当我们下行时，越来越小的球"收缩"到表面上，如图17-28所示。Hubbard（1996）指出，球树的两个重要需求为（离线）建立过程应该是自动的，而且层次结构中的每一层都必须尽可能紧密地与物体相配合。

（注意过程的需求有些不同于光线跟踪算法中使用层次化的限定体。根据定义，光线跟踪是一个两阶段的过程，我们必须有一个限制性阶段，在这个阶段准确地计算一束光线和一个多边形之间的相交。在光线跟踪中，建立只包围物体的某些部分的层次结构可能是足够的。例如，一个桌子可以只用五个限定体来表示。而在碰撞检测中，需要在层次结构中的任意水平上对球进行相交比较，而且球必须相交，并且范围要包含一个物体的表面，图17-28清楚地表明了这一点。）

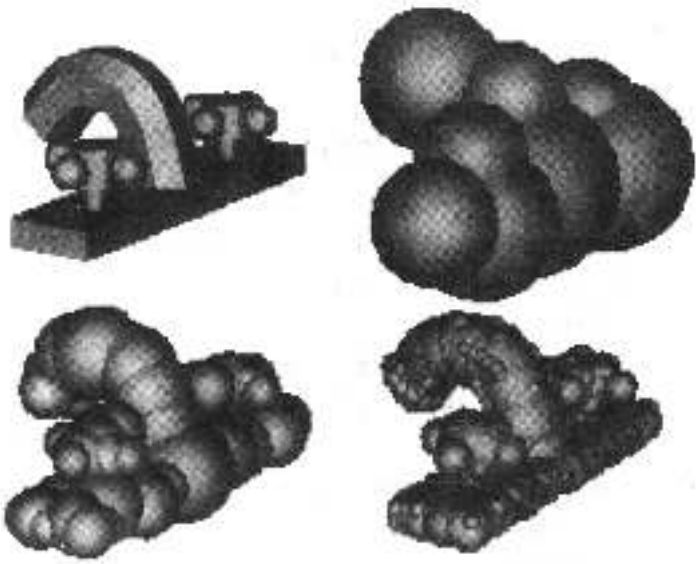


图17-28 球树的建立过程中的一个三层的例子

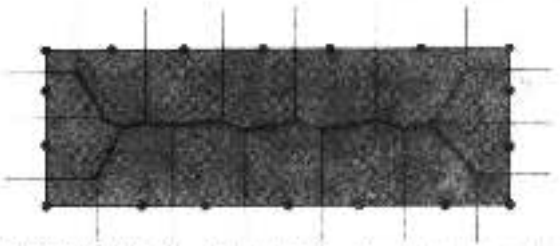
资料来源：Hubbard, P. M., *ACM Transactions on Graphics*, 15：3, July 1996, reprinted with permission from ACM Publications.

获得这样的树的方法并不是直截了当的，我们将用一个二维的模拟来简要综述这个过程。这个算法的全部细节由Hubbard（1996）给出。Hubbard首先从物体导出了一个中间轴表面，然后再用这个表面去放置球。一个二维物体的中间轴如图17-29a所示。中间轴有时称为骨架，它是物体两边的等距离点的中心。当有一个物体时，它可以是一个表面。中间轴是由为一组分配给物体表面的点P建立Voronoi图来近似的。图17-29b展示了形状上的一组点，以及与这些点相关联的Voronoi区域。点的Voronoi区域定义为到该点比到任何其他的点都要靠近的空间区域。所以，对于物体表面上的一组点，Voronoi单元必须有近似地位于中间轴的表面。更特殊的情况是，物体内部的Voronoi单元的顶点位于中间轴上，如图17-29c所示，根据定义，每一个这样的顶点都是一个圆的圆心（在三维空间中是一个球），该圆穿过三个点（在三维空间中为四个点）。这些圆（球）是建立层次结构的基础。

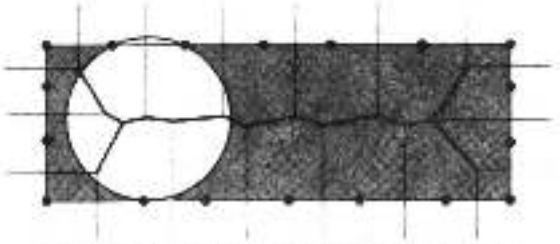
这种球的完整集合紧紧地包围物体的表面，形成层次结构的叶子，而这个层次结构通过用一种合并操作的方法减少占据每个层次的球的个数来构建。这个过程以一个二维模拟再次示于图17-29d中，将两个球s_1和s_2合并到一个球s_{12}中，这是由使用一种保证s_{12}通过或包含有s_1和s_2的形成点的算法来实现的。Hubbard（1996）将这个操作看成是一个最小化问题，方法是选择最小的成本合并。即候选的物体对的合并最大限度地保持了该层次的紧密度。



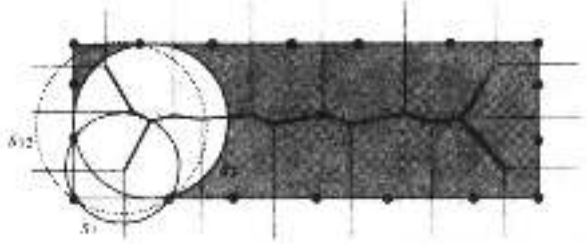
a) 一个多边形的中间轴（粗实线）



b) 由用点"增强"边界的方法，导出中间轴的一个近似，求出相应的Voronoi区域



c) 每一个Voronoi顶点是一个穿过三个点的圆的圆心



d) 合并两个圆c_1和c_2

图17-29 建立球树的Hubbard机制的一个二维模拟（基于Hubbard（1996）的示意图）

17.6 碰撞响应

碰撞响应是严格地依赖于应用的。例如，在路径规划中，应用不允许有碰撞，一个物体必须将一个源移动到一个目的地而不出现碰撞。当允许物体碰撞时，其反应依赖于物体的性质，所涉及到的计算也是动力学模拟的组成部分。

较清晰的区分是，弹性碰撞时的物体不会永久性地变形，也没有动能的损失，而非弹性碰撞则反之。非弹性碰撞和由此引起的变形更难处理，而且在这种情况下能量在碰撞中被耗散。经典的动力学通过考虑一个大的脉冲力或以一个非常短的时间周期即碰撞时间进行的反应来处理问题。考虑图17-30，图中显示有两个球沿着连接其中心的一条线移动。我们有：

522 ~ 526

$$m_1(\boldsymbol{v}_1^{after}-\boldsymbol{v}_1)=-\boldsymbol{P}$$
$$m_2(\boldsymbol{v}_2^{after}-\boldsymbol{v}_2)=\boldsymbol{P}$$

其中：

$\boldsymbol{P}$为脉冲力；

$\boldsymbol{v}_1$和$\boldsymbol{v}_2$为冲击之前球的速度；

$\boldsymbol{v}_1^{after}$和$\boldsymbol{v}_2^{after}$是冲击之后球的速度。

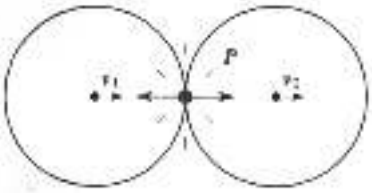


图17-30 两个球碰撞

于是有：

$$m_1\boldsymbol{v}_1^{after}+m_2\boldsymbol{v}_2^{after}=m_1\boldsymbol{v}_1+m_2\boldsymbol{v}_2$$

这个式子我们随时可以导出，因为在系统上没有内力的作用。

我们定义：

$$\boldsymbol{v}_{rel}^{before}=\boldsymbol{v}_1-\boldsymbol{v}_2$$

作为方法的速度，并且：

$$\boldsymbol{v}_{rel}^{after}=\boldsymbol{v}_1^{after}-\boldsymbol{v}_2^{after}$$

作为分离的速度，假设这些值由一个复原系数e连接起来：

$$\boldsymbol{v}_{rel}^{after}=-e\boldsymbol{v}_{rel}^{before}$$

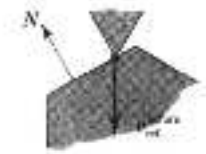
其中：

如果$e=0$，则$\boldsymbol{v}_1^{after}=\boldsymbol{v}_2^{after}$，没有弹回——这是一个非弹性的碰撞。

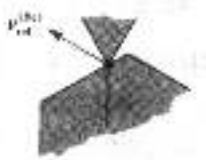
如果$e=1$，则$\boldsymbol{v}_1^{after}=\boldsymbol{v}_2,\boldsymbol{v}_2^{after}=\boldsymbol{v}_1$，球之间交换速度——这是一个完全的弹性碰撞。

因此，复原系数确定了在碰撞中动能损失有多大。当然在实际中，这个能量的损失可能会作为一个物体或两个物体的变形显示出来，从定义上来看这是质量的重新分布，这种变形使得分析失效，所以我们假设任何变形都不会导致重心的改变。

现在考虑多面体碰撞。为了简化起见，只考虑一种模式的接触——一个顶点与一个表面的碰撞（在实际应用中，当然可以有两个表面或两个顶点的碰撞）。在这种情况下，接触的表面是碰撞物体之一的表面，可以定义该接触表面的一个法向$\boldsymbol{N}$。然后，考虑在方向$\boldsymbol{N}$上相关速度的分量发生了什么变化。如果$e=1$（见图17-31b），则方向$\boldsymbol{N}$上的分量被反向，而垂直于$\boldsymbol{N}$的相对速度保持不变。当$0\leqslant e\leqslant 1$时，$\boldsymbol{N}$方向上的相对速度被降低。



a) 一个顶点表面接触



b) $e=1$，方向$\boldsymbol{N}$上的相对速度被反向，而垂直于$\boldsymbol{N}$的相对速度保持不变

图17-31 多面体的顶点表面碰撞

碰撞检测过程用于确定出现了一个碰撞以及冲击点。常规的假设是没有相互穿透。如果一个碰撞检测程序检测到了相互的穿透，则它“会将模拟移回”，直到确定了接触点为止。

一般而言，我们必须考虑处于运动中的刚体在平移和旋转。可以在碰撞发生之后马上定义$\boldsymbol{Q}$的接触点的速度为：

$$\boldsymbol{v}_A^{after}=\boldsymbol{v}_A^{after}+(\boldsymbol{\omega}_A^{after}\times\boldsymbol{r}_A)$$

其中$\boldsymbol{\omega}_A^{after}$是冲击之后物体$A$的角速度，$\boldsymbol{r}_A$是接触点到$A$的重心之间的距离。

现在回来更详细地考虑脉冲力P的性质。正如已经提到的，P是一个在非常短的时间间隔之内作用的实体：

$$P = \int_{\Delta t} F \mathrm{d}t$$

P的作用是产生速度的瞬间变化，它取动量的单位。如果令P的大小为p，则可以写为：

$$v_A^{\text{after}} = v_A^{\text{before}} + \frac{pN}{m_A}$$

其中m_A为物体A的质量，并且：

$$\omega_A^{\text{after}} = \omega_A^{\text{before}} + I_A^{-1}(r_A \times pN)$$

对于物体B可以得到相似的表示。导出以p的函数表示的相对速度，对于A和B的速度和角速度的方程，用前面的方程产生：

$$p = \frac{-(1-e)V_{\text{rel}}^{\text{before}}}{\frac{1}{m_A} + \frac{1}{m_B} + N \cdot (I_A^{-1}(r_A \times N) \times r_A + N \cdot (I_B^{-1}(r_B \times N)) \times r_B}$$

527
≀
528

如果一个移动物体与一个大的静态物体碰撞，可以假设它根本就不动，则可以将$1/m$设为零，I为一个零矩阵。

17.7 粒子动画

粒子动画是在十多年前出现的经典技术。它现在被广泛接受并且目前仍然是一种很流行的工具。其基本的思想是，某些（自然）现象可以通过脚本并绘制大量单个粒子的运动来模拟。一个粒子通常是一个基元，其几何尺寸很小或为零——也就是说，很多粒子都可以投影到一个像素域上。但是，它具有某些像颜色这样的特定性质。对每一个粒子都进行脚本化，其思想是一帧一帧地绘制成群的粒子来产生一类云状的物体，这类物体可以长大、收缩、运动、改变形状等。一个动画可以包含成千上万的粒子，对每一个粒子提供单独的脚本已经超出了本文的范畴。取而代之的是为每一个具有内置随机特性的粒子提供一个通用的脚本，由于粒子的位置随着时间变化，这种随机特性为每个粒子产生必需的差别。通过采用通用的粒子脚本并改变粒子的特性，例如颜色，来对不同的现象进行建模。例如，在模拟烟火时，基本的粒子脚本可以是一条抛物线。每一个粒子的参数是可变的，这些参数包括抛物线的起点、形状参数、作为沿着其抛物线路径上的位置函数的粒子的颜色以及沿着其路径的生命周期。

因而，作为一个时间函数的粒子的动力学行为和它们的外观就可以被合并到同一个脚本中。可以用随机过程来控制粒子行为的这些方面。总的结果是像云彩这样的动画物体由于成千上万构成其总的形状的粒子都遵从其脚本，所以其形状也随脚本的改变而改变。这个领域的先驱者是Reeves，他在1983年发表的文章中采用了粒子集来模仿像火、云这样的“模糊”物体。其他人运用了他的思想来模仿水的行为，比如说喷泉、瀑布和打碎的浪花中的水。

Reeves（1983）把动画序列中火焰的产生描述为以下一个五步的过程：

1) 产生新的粒子并注入到当前系统中。

2) 为每一个新的粒子赋予其独立的属性。

3) 消除任一超过了其生命周期的粒子。

4) 根据其脚本移动当前的粒子。

5) 绘制当前的粒子。

由一个依赖于应用的随机过程来脚本化或控制一个粒子云的瞬间粒子数量。例如，特定
529 时刻t产生的粒子数量可以由下式导出:

$$N(t) = M(t) + \text{rand}(r)V(t)$$

其中$M(t)$为由变量V的一个随机变化扰动而产生的粒子的平均数。这个方程的时间依赖性可以用于控制总的云的尺寸的生长（或缩小）。

Reeves（1983）在其所给出的例子中使用了一个具有恒定变量的线性时间依赖性。但是他也指出，控制可以合并二次的、三次的甚至于随机变量。粒子的个数也可以与物体的屏幕尺寸相关联——这是一种允许所承受的计算量与物体的最终尺寸相关联的机制。

尽管这一机制很明显会对云的形状估算有一些贡献，但是这也是由对单个粒子的脚本决定的。这两种脚本机制的结合用于动画现象，例如在运动的图画《Star Trek II: The Wrath of Khan》中所采用的火墙的膨胀，并且一直用于模拟多彩的烟火。单个粒子脚本是基于下列属性的:

1) 初始位置。

2) 初始速度和方向。

3) 初始尺寸。

4) 初始透明度。

5) 形状。

6) 生命周期。

速度和生命周期的脚本可以基于动力学的约束。例如，膨胀可以引起一个粒子被向上弹起，然后在重力的作用下被拉下来。与属性脚本和总体粒子的脚本相关联的是一个“产生形状”，这是一个围绕着粒子云中心的几何区域，“新生”的粒子被放在这里。例如，爆炸的烟火可能有一个空间的产生形状。图17-32所示为用这些技术产生的动画序列中某部分的一个例子。

尽管Reeves（1983）描述的应用一般是增长的现象，粒子云总体倾向于增加。但是这个方法已足够通用，它可以模仿比如说粒子数总体保持恒定而云的形状被扰动，或者总数减少或内爆的现象。如我们已经指出的那样，最终物体的外观由单独对所有的粒子进行绘制的净效果来确定。通过简单地把每一个粒子作为一个单个的光源并且采用外观参数的最终值进行绘制。

在Reeves and Blan（1985）后面的文章中进一步发展了粒子系统。离开用粒子来建模无定形物和连续地改变形状的轨道，他们将粒子用作“体积填充”的基元以产生实体的形状，
530 该形状的样式通常持续地保持，但是还有能力改变其形状，比如说在风中草叶摆动的情况。在电影《The Adventures of André and Wally B》中采用了这些技术来产生三维的森林和草的背景图像。

在这种情况下，基本的粒子系统的重要意义不在于它能够模拟形状改变的物体，而在于其“数据库放大”的性质，也就是用一个简单的数据库来描述物体总体性质的能力，这个物体可以由此被模拟成所需的细节水平。物体以一个最终的复杂性进行建模，这个复杂性远远比用传统的技术所能获得的复杂性高。例如，Reeves（1983）论述到，在一个树林场景中，一般情况下将从基本的树的描述中产生成百万的粒子。

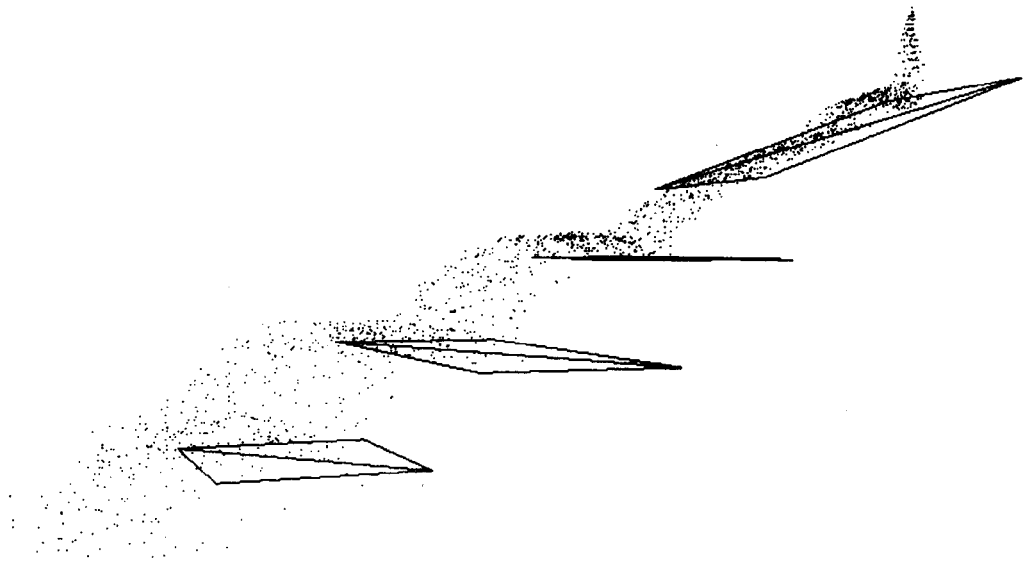


图17-32 一个物理地基于粒子动画的例子。在空间的顶部释放出一个粒子流，在重力作用下降落，并在每一个阶梯上弹起

17.8 行为动画

行为动画从许多现象来看都是粒子动画的详尽细节。通常，我们建立的是涉及到实体的总体行为的一种基本的“社会”模型（或者一个单个的实体）。行为动画和粒子动画之间的显著差别在于，在行为动画中每一个实体都被分配了一组规则，按其与相邻实体之间的关系（通常是空间的关系）的一个函数控制其行为。

Reynolds（1987）开发了一个早期的有影响力的行为模拟的例子，用来模拟鸟和鱼成群的现象。在这个例子中，每一个鸟或鱼都具有一组控制其相应于该组相邻成员的行为规则，而组是应用全局方向向量进行控制的。这个基本思想被用于迪士尼的产品《狮子王》中并产 531
生很大影响，在这个片子中兽群蜂拥前进的序列是以这种方式控制的。

Reynolds（1987）指出，兽群的行为由两个相反的因素组成，一个因素希望呆在兽群中，另一个是希望避免在兽群中发生碰撞。他将这种行为模拟成三个规则，按降序顺序这些规则为:

1) 避免碰撞: 避免与邻近的同伴碰撞。

2) 速度匹配: 试图与邻近的兽群同伴的速度进行匹配。

3) 兽群集中: 试图与邻近的兽群同伴靠近。

模型的行为由Reynolds（1987）总结如下:

> 所描述的兽群模型使鸟有一种渴望加入到一个可接受的类似兽群的运动近似中去。分散的鸟开始集合在一起，跳跃着拥挤着找到自己的位置。鸟们互相之间站得很近（距兽群的中心），但是却总是小心地与它们的邻居分离开（避免碰撞）。并且，兽群会很快“极化”，即它的成员以近乎相同的方向和近乎相同的速度前进（速度匹配）。当改变方向时，它们是同步改变的。孤单的鸟和较小的兽群加入进来成为更大的兽群，当有外部障碍物存在时，较大兽群可以分开成较小的兽群。

从Reynolds开创性的工作开始，让我们考虑一个新的行为动画系统。这个工作由Tu and Terzopoulos（1994）完成，它模拟鱼的行为，是一个半动画的系统，其中，模型带有一个基本的画面，用基于物理的与液体动力发生反应的运动能力来模拟游泳行为和一组行为规则。这似乎是第一个在计算机动画中将所有方面集成到一个系统中的尝试。

构成每一条鱼的独立模型是由不同层次的抽象组成的。在鱼的身体内部是一个“动画的弹簧-质量系统”。这是一个23个结点的集合，这些结点由91个弹簧相互连接。有些弹簧也作为可伸缩的肌肉。结点的位置控制着鱼的形状。鱼通过伸缩其肌肉即减小一个肌肉弹簧的静止长度便可以像真正的鱼那样游泳。例如，尾巴摇动的特性是通过伸缩身体一侧的肌肉而同时放松身体另一侧的肌肉来建立的。为每一个结点建立一个方程以关联结点的质量、加速度以及所施加的力。这个力来自所有其他结点及外部液体动力，通过弹簧施加到结点上。这些方程在每一个时间步长上被求解一次以求出结点的总体运动。因此，模型这一部分的基础是一团在水中向前运动并同时互相之间运动的结点。运动是通过运动控制器启动的。运动控制器把像向前游或翻转这样的动作转化为详细的肌肉的动作。

为了使这些工作得以进行，结点将模拟鱼的表皮的一个参数表面的控制点进行耦合，这
532 导致了一个可以变形的物体，其变形由其隐含的物理模型来控制。此外，控制鱼鳍的方向得到对倾斜、摇摆和翻转这些基本运动的控制。这个模型以一种高保真的水平成功地模仿了鱼的运动，对其增加了一个行为系统。该系统的输入信息来自基本的可视感知以及温度传感器。可视传感器抽取出像颜色、尺寸、距离和进入其观察域的物体标识等信息。

行为是由一组产生控制肌肉的适当动作的程序实现的。这些程序由一个意向产生器进行选择，而意向产生器选择基于传感器的信息、鱼当前的精神状态及其习性的行为。习性用参数表示，精神状态由变量表示，像饿这一类状态是随着时间增长的，而当鱼吃了食物粒子时该状态便减少。行为程序将这样的行为模拟为躲避静态障碍物、躲避鱼、吃食物、配对、逃跑、集群等。鱼模型的一个信息流表示如图17-33所示。

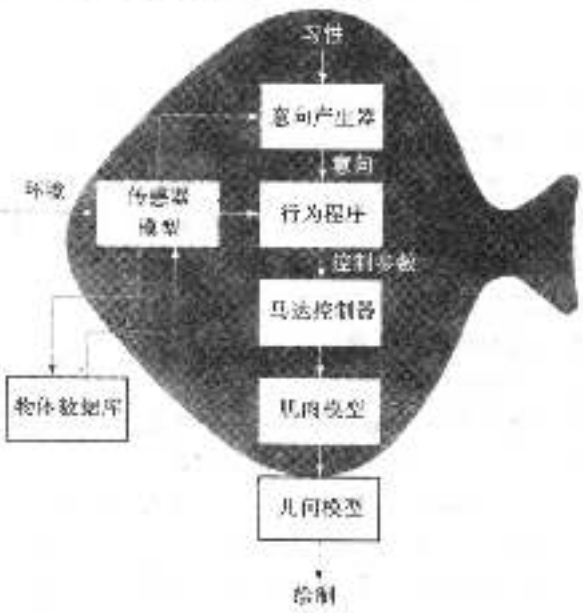


图17-33 在人工鱼上的信息流（Tu and Terzopoulos (1994)）

开发者声称，他们的系统产生“某些令人惊奇的行为”，而且，他们的产品“去钓鱼”确实给人以非常深刻的印象。然而，他们的动画的可视化成功引出了一些问题。我们再一次看到了关于艺术家的自由度的问题。如果所有的动画来自模型，那么动画制作者们如何应用工具呢？完全的自主动画的潜在应用是什么呢？作为一个实验上的试验床让行为科学家去测试或模拟他们的理论？作为一个虚拟的环境？作者在这里宣称：“我们可能已经达到能够模仿雌性和雄性产卵行为的计算模型的程度，因此，通过模拟繁殖可以发展出新的各种种类的人工鱼”。

17.9 总结

对动画方法进行比较的一个好方法是通过对系统中实现的运动控制的层次进行比较，换句话说，即对高层次与低层次运动控制的至关重要问题进行比较。这是一个至关重要的问题，因为动画从程序员手中传递到艺术团体需要高层次的脚本工具的发展。对于高级语言和API的发展有一个类比。初始阶段，使用计算机的专有权属于那些机器代码程序。高级编程语言的发展使得计算机得到大量应用。当今，对计算机的全球化的使用主要是通过运行API程序来实现的。可以这么说，当前使用计算机动画的方便程度差不多相当于在20世纪60年代末期和20世纪70年代早期由于高级语言的出现使人们进行编程的方便程度。

那么，如何来称呼计算机动画的高级和低级呢？考虑采用明确的脚本的刚体动画。在这个方法中，动画制作者对于动画序列中的每一个点的运动都有完全的控制权。这是一种低级的运动定义。对于所考虑的动画来说，对运动有完全的艺术上的控制，即动画制作者完全有自由来选择什么样的运动适合于应用。对于这个自由度所付出的代价是在定义中所涉及到的人力，尽管作为一种脚本形式对曲线的使用将工作负荷压到了一个可以接收的水平。

当有带关节的结构时，有一个选择至少在理论上是如此。如果采用的是前向动力学的话，则这是一个低级的运动定义——必须对每一个点单独脚本化。工作负荷依赖于系统的复杂性，但这时动画制作者有完全的自由度。然而，即使对于每一个连接都采用曲线脚本化其工作的负荷也是巨大的。一般而言，将有关节的结构按层次进行排序。例如，对于一条腿，可以将其结构表示成一棵树，臀部的关节作为其最顶部的结点，膝关节作为其下一个结点，踝关节作为膝关节之下的结点等等。动画制作者脚本化这样一个模型是从最顶部的结点开始的，并向下进行。每一个结点都继承了其上面结点的变化等等。如果端效应器的运动是错误的将会出现问题。动画制作者必须在最上面的结点处重新开始并向下进行。可选择的方法是采用反向动力学系统，这是一个高级系统。在这里，动画制作者可以只定义一个角色从点A走到点B。对于行走的性质没有控制，这个性质是由反向动力学算法计算出来的。工作减少了，但是却没有了对细节进行艺术上的控制的可能性。

533 ~ 534

大多数的表现形式中，动力学模拟是一个高级的定义。对于参数的变化可以有某种程度的控制。参数是已经建立在模型中的，但是通常只是简单地进行模拟，动画是在没有任何进一步的干扰的情况下被计算的。动画的效率依赖于过程对运动的模拟程度的好坏。

如我们所见，行为动画可以是完全自动的。系统在建立之后是自动画的。所有的设计自由度在设计系统时都结合了进来。如果对于系统的设计没有进行改动，则最后它只有产生大量动画的能力。在对社会群落以及成员的设计中考虑艺术家的自由度，尽管动画制作者可以通过一个界面访问系统，并且比如说改变行为的常规。我们所讨论的系统在它所产生的运动现实方面给人很深的印象。但是，这一技术的缺陷是它涉及非常详细的模拟策略，最后限制

到一个物种，在本例中为鱼。这个定义的问题在图像合成中经常出现。它意味着我们设计了一个模拟策略，尽管它尤其可以很好地适合物理真实性，但是它并没有一个通用的应用。这里的经典例子是用分形策略模拟山区的地形图，在模拟星球时使用L系统。

所以可以得出结论，总之，当前的计算机动画中的技术进行了平衡，这导致了一个矛盾。低级的运动定义可能是不寻常的繁琐，尤其是如果动画的物体有很多自由度时，但是，它为动画制作者留下了一个机会，即在进行大量工作的前提下加入其自己的动作。高级的定义对于一个艺术家来说是易于使用的，但是一般来讲导致有较少的艺术自由度。也许正是由于这一矛盾的存在阻止了计算机动画传播到艺术界中去，以及使动画继续作为一种手动的艺术形
535 式而存在。

第18章　比较图像研究

18.1　局部反射模型
18.2　纹理映射和阴影映射
18.3　Whitted 光线跟踪
18.4　辐射度方法
18.5　RADIANCE
18.6　总结

引言

本节采用比较图像研究的形式，阐述本书中已经描述的主流绘制程序之间可视性的差别。图像应该在监视器上观看，而不是以彩色图片的形式进行观看。由绘制程序所产生的某些缺陷是很难重现的，但是在屏幕上是可见的。本研究由光盘上的400个图像研究中的一部分组成。

处理是基于一个旧的想法，即尽可能地用相同的场景进行不同绘制程序的输入，这样可以对不同的方法进行良好的比较。当观察者对于特定的绘制程序进行调整后产生的场景进行观察时，视觉上的差别并不是太明显。所用的图像阐明了不同方法的优点和缺点。使用的原始场景含有1万个多边形，这对于朴素的相交测试光线跟踪程序来说太多了，对于辐射度绘制程序也太多了，这种方法中一个多边形就等于一个曲面片。光线跟踪问题可以用空间划分的方法来解决。在辐射度方法中表示和绘制方法之间的关联是非常困难的，我们将用一个例子来展示一些可能性。

场景中的细节根据所演示的内容和所采用的方法在例子之间有些微的不同。对于辐射度方法，小的镜面物体被去掉了。

本章涉及下列议题:

- 局部反射模型和插值明暗处理。
- 纹理映射和阴影映射。
- Whitted（光线后向或眼睛）光线跟踪。
- 采用场景细分的辐射度方法。

18.1　局部反射模型

本节阐述主流多边形网格绘制中使用的标准成本-图像质量。低质量的图像用于预观察和调节工作。

图18-1（彩色插图）

一个办公室场景与线框可视图，只用恒定的环境项进行了明暗处理。

图18-2（彩色插图）

采用平面明暗处理的相同场景。平面明暗处理由于光强度的不连续性而表现出表面的多边形性质。

图18-3（彩色插图）

用Gouraud模型进行了明暗处理的办公室场景。

在使用插值技术明暗处理过的图像中，有三种高度可见的缺陷。第一种缺陷经常在Gouraud明暗处理图像中出现，这种现象称为马赫带。这种现象在文献中几乎是不可能重现的（但是，通常在监视器上是可见的），我们在这里仅限于描述。如果我们考虑在一个多边形网格物体的表面的光强度的轮廓，则这个轮廓将如图18-4所示是分段线性的（彩色部分）。当在屏幕上观察这个表面时，人类的视觉系统会在相应于多边形边的表面上看到这个带或线。这个带或线实际上并不存在，但却是视网膜处理低层次信息时对光强度的分段线性的变化的反应。这些现象在暗处出现，但是仍然是可见的带，它表现出比其周围的表面有较低的光强度（当用Gouraud型插值计算最终的投影图像时）。

在Gouraud明暗处理中出现的插值缺陷表现了在一个表面上光强度的不希望的变化。这个问题在图18-3的紧靠着门的墙上可以很清晰地看到。在这个地方由于所使用的边界位于插值的转换点外，所以在扫描线上出现了不连续（用于产生这一示意图的插值方程见1.5节）。这个问
537 题可以通过把大的多边形细分成三角形加以解决，换句话说，我们必须更精确地对几何形状进行采样；我们不能仅仅将大的平面区域映射到单个多边形上而不考虑绘制程序中的结果。

图18-5（彩色插图）

这是用Phong明暗处理模型建立的相同场景。

在这个图中，显示了一个Phong插值中的闪烁缺陷。在这个图中，由于插值的性质而使得来自壁灯的反射光与光线的图像分离了。在这种情况下，为了计算反射，我们必须在扫描线上按等步长进行插值。但是，因为这是用一个透视投影从世界空间进行的映射，所以光线的图像出现在不同的位置上。换句话说，屏幕空间的投影在墙壁上用相同的像素单位，但是却与世界空间的等长单位不相适应。这个问题以及当我们试图把光源作为场景的一部分包含进来的时候所遇到的问题是将光线排除在场景之外的动机。考虑球形光线。将其作为一个明暗处理过的物体进行处理则无法工作。在这种情况下，点光源处于包围球之内，从定义上不能对其进行照明。光线物体只能从壁灯处得到照明，我们可以找到点光线的环境分量以使得它看起来像一个光源。

图18-6（彩色插图）

这是壁灯上的一个区域。上面一行对Gouraud明暗处理和Phong明暗处理进行比较。如果多边形网格的分辨率足够大，则Gouraud明暗处理和Phong明暗处理之间的差别相当小。随着多边形的尺寸趋向于一个像素，Gouraud明暗处理趋向于Phong明暗处理。在下面一行，降低了多边形的分辨率。这时，由于没有任何网格的顶点靠近Gouraud高光所以它消失了。

18.2 纹理映射和阴影映射

我们沿着成本-图像质量层次结构进入到标准的纹理映射和阴影映射的“附加”作用。

图18-7（彩色插图）

这是一个带有传统的二维纹理映射的办公室场景。向一个Phong明暗处理过的场景附加简单纹理的外观复杂性/真实性/可视的趣味性的原因是由于不断的普及。如果有什么不同的话，
538 那是因为添加纹理后，场景增加了其很明显的计算机签名，看起来更缺少了真实性。

图18-8（彩色插图）

同样的场景，但加入了阴影和环境影射（茶壶）。阴影图上的分辨率是256 × 256，这引起了几何走样，如左面墙上的阴影，它实际上已经分裂开了。请注意，在阴影映射和环境映射之间有一个“相互作用”，操作的顺序是很重要的。在这个图像中，环境映射在阴影映射之前计算。因此，茶壶的阴影没有出现在茶壶的反射中。

另一种方法是，可以产生包括茶壶的整个场景的阴影映射，然后再移动它，并计算环境映射。然后，再把茶壶重新插入到场景中以便绘制，这时，环境映射将包括它的阴影。

还应注意阴影边界的定义。其不足之处是由于阴影图的空间分辨率造成的，在这里，每一个分量为256 × 256。

图18-9（彩色插图）

这是对用环境映射和光线跟踪产生反射的比较。图中显示出由环境映射所引入的几何变形的扩展。其不正确性是环境中物体尺寸的一个函数，随着物体趋向于一个点，反射变得正确了。环境映射的另一个缺点是它并不进行自我反射，茶壶并没有包含其壶嘴的反射。当产生了环境映射时，物体被从场景中去掉了。

不值一提的是，在光线跟踪中，很容易给物体的全局镜面反射系数赋值，物体的个数可按我们所需。对于环境映射，我们必须为每一个物体建交一个独立的图。

18.3 Whitted 光线跟踪

下面的三个图展示了主流光线跟踪算法的某些方面，Whitted或递归光线跟踪用无限细的光束穿过经过细分的物体空间进行跟踪。

图18-10（彩色插图）

用Whitted光线跟踪程序对场景进行光线跟踪。在这个场景中，有大约10 000个多边形，采用一种朴素或强力的相交测试，其中每一条产生的光线与场景中的每一个多边形进行测试，这将导致非常长的执行时间。采用八叉树的表示对场景进行光线跟踪。尽管与Phong明暗处理方法之间的差别是很明显的。但是，有几个问题还是值得考察的。在跟踪光线中标准的选项时，阴影是边界分明的（在每一个相交点上，有一条光线射向点光源）。这些边界看起来是错误的，与把这样的功能加入到阴影映射程序中相比，在光线跟踪算法中，实现或者近似软阴影更加困难（见前面的图）。

539

另一个看起来“错误”的效果是门边的蓝色玻璃花瓶。在这里，墙上阴影的界线分明的反射使得花瓶看起来好像具有一条黑色的带子，而不是一个反射的细节。

在光线跟踪算法中，我们产生一组内部的光线作为场景几何的样本。在简单的正交投影的情况下，这个带子可能在图像平面平行线的位置处出现。在第一次碰撞之后，所有这些平行线束都将扩散，其扩散的程度依光线初次碰到的表面的性质而定。总的来说，这将意味着所有第二次对表面的碰撞都将以较低密度的光线被采样，并且在整个递归中一直进行下去。这意味着反射的图像作为一个递归深度的函数会变得越来越差（见下一个图）。而同时，随着反射之中的反射变得越来越小，而使得缺陷变得不太引人注意了。

图18-11（彩色插图）

一个递归深度的演示。所示的是茶壶-镜面相互作用，深度分别为2、3、4和5。物体均是完全镜面反射器，并且没有局部分量。所有的颜色必须来自全局镜面分量。终止递归导致未

知的分量，并以灰色表示。一个灰色的茶壶的“阴影”递归地进入到镜子中。有趣的是，在放大的图像中显示了光线带作为递归深度的函数中断了。

图18-12（彩色插图）

这三个图像都采用标准的光线跟踪和辐射度算法，到目前为止两者都希望模拟全局相互作用的性质。

图18-12a是一个光线跟踪的场景的图像，其主要的光线被慢慢减弱，以强调光线的跟踪忽略了除LDE和LDS*E之外的所有光线路径。这个场景中的内部并不是非典型的，在这个场景中，大多数全局相互作用都是从漫反射到漫反射的（也就是说，来自墙上面向光线的间接照明），因此，光线跟踪是完全不恰当的。图18-12b是前一个场景加上了“环境照明”。环境分量的值与绘制只有主要光线的场景时所用的值是相同的（见图18-10）。并且，假设该值是与考虑从漫反射到漫反射的相互作用时可能会出现的照明的一种替代。图18-12c是一个用辐射度方法绘制的图像。用光线跟踪方法处理镜面物体，主光源被关闭（包括了光线跟踪的分量以与下一个图比较）。由于辐射度方法考虑到了漫反射光的内部反射，所以房间中的其余部分是可见的。

图18-13（彩色插图）

光线跟踪算法尤其适合于与一种非均匀采样的方法相结合（图18-13彩色插图），因为它
540 允许我们在图像平面的任何地方产生初始光线（这与多边形网格绘制方法相反。多边形网格绘制方法中最常采用的绘制策略是计算间隔在图像空间是等间距划分的）。

非均匀采样出现在Whitted的原稿中，在文章中他使用了一个经典的自适应细化方案。初始光线以每个像素一条光线的速度产生，这是标准的一阶近似。对四个像素角点处所计算的强度值进行比较，如果它们的差大于预先指定的阈值，则这个像素被递归地细分直到达到阈值，或者分辨率的界线。为了重建一个像素的最终值，对每一个子像素区域的贡献按区域进行加权。这个过程如图14-12所示，在该图中讨论了在对一个像素进行细分时，以数值方法对样本进行加权。在这种算法中嵌入的是一个细化测试以及一种把从新的光线返回的光强度值结合（或者重建和过滤）成一个值的方法。细化测试使用基于对区域加权的光强度值。这些都是特定的解决方案，自从Whitted发表了其经典文章之后，又有很多人发表了很多其他关于细化策略以及重建问题的文章。

这些图表明了上下文无关与上下文敏感的反走样效果。这些图像分别代表没有经过反走样、以3倍屏幕分辨率超采样和刚刚讨论的上下文敏感的方法。请注意，在两个反走样的版本之间没有明显的差别。正常的方法包括9倍于反走样版本的工作量，而上下文敏感的方法只是1.5倍。

18.4 辐射度方法

辐射度方法的图像是用基于渐进式细化半立方体的方法产生的。

图18-14（彩色插图）

这个图显示了场景的辐射度方法版。这个图像与前面图像之间的引人注意的差别之一是所谓的彩色渗出效果——使图像的颜色有所不同。由于有从漫反射到漫反射的相互作用，来自一个漫反射表面的颜色被传递到相邻曲面片。这一现象在天花板上尤其明显。现实生活中的颜色渗透现象是有争议的。换句话说，可能因为颜色恒久不变的感觉占统治地位，我们对于颜色渗透现象并没有多少感觉，但这种现象可以通过仪器来测量。这就意味着，作为人类，

我们倾向于把物体的颜色看成是物体的恒定属性，而不考虑照在其上的光线的颜色。于是，问题就出现了，我们是否要把这个效果包括在图像合成中呢？可以肯定的是，颜色渗透现象是“固定”在照片中的。而且，这种现象在一个场景的照片中比从摄像机镜头中所实际观察到的要明显。图18-15a（彩色插图）是一个我们在里约热内卢中午耀眼的阳光下拍的照片。甚至从远距离都可以生动地看出颜色渗透现象。 541

图18-15b是一个辐射度方法的图像，它显示了从彩色的光源到墙壁上的颜色渗透现象，以及由窗户对一个彩色物体的光线反射所导致的颜色渗透现象。这些图像取自附近的会聚点。而其效果尽管到目前为止从算法上来看是正确的，但是看起来却有一些错了。这可能部分是由于我们在真实环境中的颜色恒定性与观察照片的方法有所不同。圆柱体的反射性被设为0.85。也可能是由于只用三个波长来计算而造成的。

图18-16（彩色插图）

在辐射度方法中，最严重的缺陷是由于不适当的网格化。尽管辐射度方法也会产生其他可见的缺陷——主要是由于半立方体的大小有限以及由于插值造成的马赫带，但是那些由不适当的网格化或者不适当的细分所产生的缺陷通常看得最清楚。不适当的网格化所产生的最常见的结果是块状的阴影、阴影泄漏或者光线泄漏。除非场景非常简单，或者把场景构建成把这些人工痕迹都包括进来，否则这些缺陷将总是出现。

正如我们所知，辐射度方法从场景数据库计算出一个独立于观察的结果，而数据库中的内容已经被细分成了称为曲面片的单元。计算结果把一个恒定的辐射度分配给每一个曲面片，然后采用Gouraud插值给出曲面片上光强度的线性变化。可以通过将曲面片分得足够大使辐射度算法在合理的时间内运行，而这意味着采用随意的细分策略的一种基本的辐射度算法将总是产生图像缺陷。

这个图像是用一个“最小表示”进行计算的。线框图是一个图18-1中所示表示的一个三角形划分的版本，这是我们可以试着用辐射度方法求解问题的最基本的细分方法。尽管有了比原始图像多得多的多边形，但是图像的质量还是由于阴影泄漏和光线泄漏令人不可接受。

图18-17（彩色插图）

在这个图像中，三对图像显示了细分策略的效果（如11.7.1节中所示），即在一个渐进式的细分框架中运算。增加细分程度和图像质量之间的关系是很明显的。对于最终的图像对，网格和图像的检查表明了一个与细分有关系的问题。即网格显现出了一些看起来不需要的细分区域。认我们来观察靠近门处的“条纹”，请记住，每一个曲面片只有在被照射到时再细分，而且由此而细分出的区域接下来可能还会被来自其他曲面片的能量照射到。 542

图18-18（彩色插图）

尽管在图18-17（彩色插图）中的最后一对图像中表现出了高度的细分，但是在壁灯周围还是出现了光线泄露。通过在考虑了相互渗透的几何学之后对壁灯周围的区域进行网格化可以完全消除这种现象，如这个示意图所示。这时，墙壁的曲面片边界与光线的曲面片边界共线。

18.5 RADIANCE

RADIANCE绘制程序是一个全局照明绘制程序，它处理含有镜面反射和漫反射物体的场景。

图18-19（彩色插图）

这个场景是用RADIANCE绘制程序绘制的。与其他图像相比，这个图有轻微的闪烁现象，这是由于对漫反射相互作用的采样方式不同造成的（见10.9节）。

18.6 总结

比较各种不同的图像，它们明显地表现出很多差别。可能这些差别中更引人注意的是有关光线的水平和颜色的偏移问题。除了明显的算法属性（例如硬边界和软边界阴影）之外，很少有人试图对每一个接口上的物体性质以及光线水平进行匹配。要找出由于不同的模型和算法的方法论的不同而产生的差别，以及哪种差别是由于不适当地在绘制程序界面之间的切
543 换造成的是很困难的。这是一个在使用绘制程序时遇到的长期存在的实际问题。

参考文献

Abram A. G., Westover L. and Whitted T. (1985). Efficient alias-free rendering using bit-masks and lookup tables. *Computer Graphics*, **19** (3), 53–9

Appel A. (1968). Some techniques for machine rendering of solids. *AFIPS Conference Proc*, **32**, 37–45

Arvo J. (1986), Backwards ray tracing, developments in ray tracing. *SIGGRAPH Course Notes*, **12**

Arvo J. and Kirk D. (1987). Fast ray tracing by ray classification. *Computer Graphics*, **21**(4), 55–64. (*Proc. SIGGRAPH '87*)

Atherton P., Weiler K. and Greenberg D. (1978). Polygon shadow generation. *Computer Graphics*, **12** (3), 275–81

Balakrishnan A. V. (1962). On the problem of time jitter in sampling. *IRE Trans. on Information Theory* (April 1962), pp. 226–36

Barr A. H. (1984). Global and local deformations of solid primitives. *Computer Graphics*, **18** (3), 21–30

Bartels R., Beatty J. and Barsky B. (1987). *An Introduction to Splines for use in Computer Graphics and Geometric Modelling*. Los Altos CA: Morgan Kaufmann

Bartels R. H., Beatty J. C. and Barsky B. A. (1988). *Splines for Use in Computer Graphics and Geometric Modeling*. Los Altos CA: Morgan Kaufmann

Baum D. R., Rushmeier H. E. and Winget J. M. (1989). Improving radiosity solutions through the use of analytically determined form factors. *Proc. SIGGRAPH '89*, pp. 325–34

Bergeron P. (1986). Une version générale de l'algorithme des ombres projetées de Crow basée sur le concept de volumes d'ombre. MSc Thesis. University of Montreal

Bier E. A. and Sloan K. R. (1986). Two-part texture mapping. *IEEE Computer Graphics and Applications*, **6** (9), 40–53

Blinn J. F. (1977). Models of light reflection for computer synthesized pictures. *Computer Graphics*, **11** (2), 192–8

Blinn J. F. (1978). Simulation of wrinkled surfaces. *Computer Graphics*, **12** (3), 286–92

Blinn J. F. (1988). Me and my (fake) shadow. *IEEE Computer Graphics and Applications*, **8** (1), 82–6

Blinn J. F. and Newell M. E. (1976). Texture and reflection in computer generated images. *Comm. ACM*, **19** (10), 362–7

Borges C. F. (1991). Trichromatic approximation for computer graphics illumination models. *Computer Graphics*, **25** (4), 101–4 (*Proc. SIGGRAPH '91*)

Bouknight W. J. and Kelly K. (1970) An algorithm for producing half-tone computer graphics presentations with shadows and moveable light sources. *Proc. AFIPS, Spring Joint Computer Conf.*, **36**, 1–10

Bresenham J. E. (1965). Algorithm for computer control of a digital plotter. *IBM Systems J.*, January, 25–30

Brotman L. S. and Badler N. I. (1984). Generating soft shadows with a depth

buffer algorithm. *IEEE Computer Graphics and Applications*, **4** (10), 5–12

Cabral B., Max N. and Springmeyer R. (1987). Bidirectional reflection functions from surface bump maps. *Computer Graphics*, **21** (4), 273–81. (*Proc. SIGGRAPH '87*)

Carpenter L. C. (1984). The A-buffer, an anti-aliased hidden surface method. *Computer Graphics*, **18** (3), 103–8

Catmull E. (1974). Subdivision algorithm for the display of curved surfaces. PhD Thesis, University of Utah

Catmull E. (1975). Computer display of curved surfaces. In *Proc. IEEE Conf. on Computer Graphics, Pattern Recognition and Data Structures*, May 1975 (Reprinted in Freeman H. (ed.) (1980). *Tutorial and Selected Readings in Interactive Computer Graphics*. New York (IEEE), pp. 309–15)

Catmull E. (1978). A hidden surface algorithm with anti-aliasing. *Computer Graphics*, **12** (3), 6–10

Clark J. H. (1979). A fast scan line algorithm for rendering parametric surfaces. *Computer Graphics*, **13** (2), 289–99

Cohen J. D., Lin M. C., Manocha D. and Pongami M. K. (1995). I-COLLIDE: An interactive and exact collision detection system for large scale environments. *Proc. 1995 Symp. on Interactive 3D Graphics* (Monterey CA) pp. 291–302

Cohen M. F. (1992). Interactive spacetime control for animation. *Proc. SIGGRAPH '92*, pp. 293–302

Cohen M. F. and Greenberg D. P. (1985). A radiosity solution for complex environments. *Computer Graphics*, **19** (3), 31–40

Cohen M. F. and Wallace J. R. (1993). *Radiosity and Realistic Image Synthesis*. Boston MA: Academic Press Professional, Harcourt Brace and Co.

Cohen M. F., Greenberg D. P. and Immel D. S. (1986). An efficient radiosity approach for realistic image synthesis. *IEEE Computer Graphics and Applications*, **6** (2), 26–35

Cohen M. F., Chen S. E., Wallace J. R. and Greenberg D. P. (1988). A progressive refinement approach to fast radiosity image generation. *Computer Graphics*, **22** (4), 75–84

Cook R. L. (1986) Stochastic sampling in computer graphics. *ACM Trans. on Computer Graphics*, **5** (1), 51–72

Cook R. L. and Torrance K. E. (1982). A reflectance model for computer graphics. *Computer Graphics*, **15** (3), 307–16

Cook R. L., Porter T. and Carpenter L. (1984). Distributed ray tracing. *Computer Graphics*, **18** (3), 137–45

Cook R. L., Carpenter L. and Catmull E. (1987). The REYES images rendering architecture. *Computer Graphics*, **21** (4), 95–102

Cowan W. B. (1983). An inexpensive scheme for the calibration of a colour monitor in terms of CIE standard coordinates. *Computer Graphics*, **17**, 315–21

Cox M. G. (1972). The numerical evalutation of B-splines. *J. Inst. Maths. Applics.*, **10**, 134–49

Crow F. C. (1977). Shadow algorithms for computer graphics. *Computer Graphics*, **13** (2), 242–8

Crow F. C. (1981). A comparison of anti-aliasing techniques. *IEEE Computer Graphics and Applications*, **1** (1), 40–8

Crow F. C. (1987). The origins of the teapot. *IEEE Computer Graphics and*

Applications, **7** (1), 8–19

Debevec P. E., Taylor C. J. and Malik J. (1996). Modelling and rendering architecture from photographs: a hybrid geometry and image based approach. *Proc. SIGGRAPH '96*, pp. 11–20

De Boor C. (1972). On calculating with B-splines. *J. Approx. Th.*, **6**, 50–62

Drebin R. A., Carpenter L. and Hanrahan P. (1988). Volume rendering. *Computer Graphics*, **22** (4), 65–74

Duff T. (1985). Compositing 3-D rendered images. *Computer Graphics*, **19** (3), 41–4

Duff T. (1986). Splines in animation and modelling. *SIGGRAPH Course Notes*, **15**

Farin G. (1990). *Curves and Surfaces for Computer Aided Design*. 2nd edn. Boston: Academic Press

Faux I. D. and Pratt M. J. (1979). *Computational Geometry for Design and Manufacture*. Chichester: Ellis Horwood

Fiume E., Fournier A. and Rudolph L. (1983). A parallel scan conversion algorithm with anti-aliasing for a general-purpose ultracomputer. *Computer Graphics*, **17** (3), 141–50

Foley J. D., Van Dam A., Feiner S. K. and Hughes J. F. (1989). *Computer Graphics: Principles and Practice*. Reading MA: Addison-Wesley

Forsey D. R. and Bartels R. H. (1988). Hierarchical B-spline refinement. *Computer Graphics*, **22** (4), 205–12

Fournier A., Fussell D. and Carpenter L. (1982). Computer rendering of stochastic models. *Comm. ACM*, **25** (6), 371–84

Fuchs H. (1980). On visible surface generation by *a priori* tree structures. *Computer Graphics*, **14**, 124–33

Fujimoto A., Tanaka T. and Iwata K. (1986). ARTS: Accelerated ray tracing system. *IEEE Computer Graphics and Applications*, **6** (4), 16–26

Glassner A. S. (1984). Space subdivision for fast ray tracing. *IEEE Computer Graphics and Applications*, **4** (10), 15–22

Glassner A. S. (1995). *Principles of Digital Image Synthesis*. San Francisco CA: Morgan Kaufmann Pubs Inc.

Goral C., Torrance K. E., Greenberg D. P. and Battaile B. (1984). Modelling the interaction of light between diffuse surfaces. *Computer Graphics*, **18** (3), 212–22

Gortler S., Grzeszczuk R., Szeliski R. and Cohen M. F. (1996). The lumigraph. *Proc. SIGGRAPH '96*, pp. 43–52

Gottschalk S., Lin M. C. and Manocha D. (1996). OBB trees: A hierarchical structure for rapid interference detection. *Proc. SIGGRAPH '96*, pp. 171–80

Gouraud H. (1971). Illumination for computer generated pictures. *Comm. ACM*, **18** (60), 311–17

Gray J. (1968). *Animal Locomotion*. London: Weidenfeld and Nicolson

Greene N. (1986). Environment Mapping and Other Applications of World Projections. *IEEE Computer Graphics and Applications*, **6** (11), 21–9

Greene N., Kass M. and Miller G. (1993). Hierarchical Z-buffer visibility. *Proc. SIGGRAPH '93*, 231–8

Greger G., Shirley P., Hubbard P. M. and Greenberg D. (1998). The irradiance volume. *IEEE Computer Graphics and Applications*, March/April, pp. 32–43

Griffiths J. G. (1984). A depth-coherence scan line algorithm for displaying curved surfaces. *Computer Aided Design*, **16** (2), 91–101

Haeberli P. E. and Akeley K. (1990). The accumulation buffer: hardware support

for high-quality rendering. *Computer Graphics (Proc. SIGGRAPH '90)* **24**, August, pp. 309–318

Haeberli P. and Segal M. (1993). Texture Mapping as a Fundamental Drawing Primitive. From: http://www.sgi.com/grafica/texmap/index.html

Haines E. (1991). Essential ray–convex polyhedron intersection. In *Graphics Gems II*, Arvo. J. (ed.) pp. 247–50. Boston: Harcourt Brace Jovanovich

Haines E. A. and Greenberg D. P. (1986). The light buffer: a shadow-testing accelerator. *IEEE Computer Graphics and Applications*, **6** (9), 6–16

Hall R. A. (1989). *Illumination and Color in Computer Generated Imagery*. New York: Springer-Verlag

Hall R. A. and Greenberg D. P. (1983). A testbed for realistic image synthesis. *IEEE Computer Graphics and Applications*, **3** (8), 10–19

Hallas J. and Manvell R. (1971). *The Techniques of Film Animation*. 3rd edn. London: Focal Press

Hanrahan P. and Haeberli P. (1990). Direct WYSIWYG painting and texturing on 3D shapes. *Proc. SIGGRAPH '90*, pp. 287–96

Hanrahan P. and Kreuger W. (1993). Reflection from layered surfaces due to subsurface scattering. *Proc. SIGGRAPH '93*, pp. 165–74

Hanrahan P., Salzman D. and Aupeperle L. (1991). A rapid hierarchical radiosity algorithm. *Proc. SIGGRAPH '91*, pp. 197–206

He X. D., Torrance K. E., Sillion F. X. and Greenberg D. P. (1991). A comprehensive physical model for light reflection. *Computer Graphics*, **25** (4), 175–86

Heckbert P. S. (1986). Survey of texture mapping. *IEEE Computer Graphics and Applications*, **6** (11), 56–67

Heckbert P. S. (1990) Adaptive radiosity textures for bi-directional ray tracing. *Proc. SIGGRAPH '90*, pp. 145–54

Heckbert P. S. and Hanrahan P. (1984). Beam tracing polygonal objects. *Computer Graphics*, **18** (3), 119–27

Hoppe H. (1996). Progressive meshes. *Proc. SIGGRAPH '96*, pp. 99–108

Hubbard P. M. (1993). Interactive collision detection. *Proc. IEEE Symp. on Res. Frontiers in Virtual Reality*, pp. 24–31

Hubbard P. M. (1996). *ACM Transactions on Graphics*, **15** (3), July, 179–210

Kajiya J. T. (1986). The rendering equation. *Proc. SIGGRAPH '80*, pp. 143–50

Kaplan M. R. (1985). Space Tracing, a constant time ray tracer. *SIGGRAPH '85 Course Notes*, San Francisco CA, July

Lafortune E. P. and Williams Y. D. (1995). A 5D tree to reduce the variance of Monte Carlo ray tracing. *Proc. 6th Eurographics Workshop on Rendering*, Dublin, Ireland, June, pp. 11–20

Lane J. M. and Riesenfeld R. F. (1980). A theoretical development for the computer generation and display of piecewise polynomial surfaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **2** (1), 35–46

Lane J. M., Carpenter L. C., Whitted T. and Blinn J. T. (1980). Scan line methods for displaying parametrically defined surfaces. *Comm. ACM*, **23** (1), 23–34

Lengyel J. and Snyder J. (1997). Rendering with coherent layers. *Proc. SIGGRAPH '97*, pp. 233–42

Levoy M. (1990). Efficient ray tracing of volume data. *ACM Trans. on Graphics*, **9** (3), 245–61

Levoy M. and Hanrahan P. (1996). Light field rendering. *Proc. SIGGRAPH '96*, pp. 31–42

Lewis J. P. (1989). Algorithms for solid noise synthesis. *Computer Graphics*, **23** (3),

263–70

Lippman A. (1980). Movie maps: an application of the optical videodisc to computer graphics. *Proc. SIGGRAPH '80*, pp. 32–43

Lischinski D., Tampieri F. and Greenberg D. (1992). Discontinuity meshing for accurate radiosity. *IEEE Computer Graphics and Applications*, November, pp. 25–39

Liu Z., Gortler S. and Cohen M. (1994). Hierarchical spacetime control. *Proc. SIGGRAPH '94*, pp. 35–42

Lorensen W. E. and Cline H. E. (1987), Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics*, **21** (4), 163–9

Maciel P. W. and Shirley P. (1995). Visual navigation of a large environment using textured clusters. *Symposium on Interactive 3D Graphics*, April, pp. 95–102

Mandelbrot B. (1977). *Fractals: Form, Chance and Dimension*. San Francisco CA: Freeman

Mandelbrot B. (1982). *The Fractal Geometry of Nature*. San Francisco CA: Freeman

Mantyla M. (1988). *Introduction to Solid Modelling*. Rockville MD: Computer Science Press

Mark W. R., McMillan L. and Bishop G. (1997). Post-rendering 3D warping. *Proc. 1997 Symposium on Interactive 3D Graphics*, pp. 7–16, Providence RI, April

Max N. and Troutman R. (1993). Optimal hemi-cube sampling. *4th Eurographics Workshop on Rendering* (June), pp. 185–200

McMillan L. (1995). *A List-priority Rendering Algorithm for Redisplaying Projected Surfaces*. UNC Technical Report 95-005, University of North Carolina

McReynolds T. and Blythe D. (1997). Programming with OpenGL: Advanced Rendering. From: http://www.sgi.com/software/opengl/advanced97/notes/

Miller G., Halstead M. and Clifton M. (1998). On-the-fly texture computation for real-time surface shading. *IEEE Computer Graphics and Applications*, March/April, pp. 44–58

Miller G. S. and Hoffman C. R. (1984). Illumination and reflection maps: simulated objects in simulated and real environments. *SIGGRAPH '84 Course Notes*, July

Moore M. and Wilhelms J. (1988). Collision detection and response for computer animation. *Computer Graphics*, **22** (4), 289–98

Munsell A. H. (1946). *A Color Notation*. Baltimore MD: Munsell Color Co.

Nakamae E., Harada K., Ishizaki T. and Nishita T. (1986). A montage method: the overlaying of computer generated images onto a background photograph. *Computer Graphics*, **20** (4), 207–14

Newell M. E., Newell R. G. and Sancha T. L. (1972). A new approach to the shaded picture problem. *Proc. ACM National Conf.*, pp. 443–50

Newman W. and Sproull R. (1973). *Principles of Interactive Computer Graphics*. New York: McGraw-Hill

Ney D. N., Fishman E. K., Magid D. and Drebin R. A. (1990). Volumetric rendering of computed tomography data: principles and techniques. *IEEE Computer Graphics and Applications*, **10** (2) pp. 33–40

Nishita T. and Nakamae E. (1985). Continuous tone representation of three-dimensional objects taking account of shadows and interreflection. *Computer Graphics*, **19** (3), 23–30

Novins K. L., Sillion F. X. and Greenberg D. P. (1990). An efficient method for volume rendering using perspective projection. *Computer Graphics*, **24** (5),

95–102

Oppenheim A. V. and Shafer R. W. (1975). *Digital Signal Processing*. Englewood Cliffs NJ: Prentice-Hall

Oppenheimer P. E. (1986). Real-time design and animation of plants and trees. *Proc. SIGGRAPH '86*, pp. 55–64

Peachey D. R. (1985). Solid texturing of complex surfaces. *Computer Graphics*, **19** (3), 279–86

Peercy M., Airey J. and Cabral B. (1997). Efficient bump mapping hardware. *Proc. SIGGRAPH '97*

Perlin K. (1985). An image synthesizer. *Computer Graphics*, **19** (3), 287–96

Perlin K. (1989). Hypertexture. *Computer Graphics*, **23** (3), 253–62. (*Proc. SIGGRAPH '98*)

Philips C. B. and Badler N. I. (1991). Interactive behaviour for bipedal articulated figures. *Computer Graphics*, **25** (4), 359–62. (*Proc. SIGGRAPH '91*)

Phong B. (1975). Illumination for computer-generated pictures. *Comm. ACM*, **18** (6), 311–17

Piegl L. (1993). *Fundamental Developments of Computer-Aided Geometric Modelling*. New York: Academic Press

Porter T. and Duff T. (1984). Composing digital images. *Computer Graphics*, **18** (3), 253–9

Reeves W. T. (1983). Particle systems – a technique for modelling a class of fuzzy objects. *Computer Graphics*, **17** (3), 359–76

Reeves W. T. and Blau R. (1985). Approximate and probabilistic algorithms for shading and rendering structured particle systems. *Computer Graphics*, **19** (3), 313–22

Reeves, W., Salesin, D. and Cook, R. (1987). Rendering antialiased shadows with depth maps. *Computer Graphics*, **21** (4), 283–91. (*Proc. SIGGRAPH '87*)

Regan M. and Pose R. (1994). Priority rendering with a virtual reality address recalculation engine. *Proc. SIGGRAPH '94*, pp. 155–62

Reynolds C. W. (1987). Flocks, herds, and schools: a distributed behavioural model. *Computer Graphics*, **21** (4), 25–34

Rossignac J. R. and Requicha A. A. G. (1986). Depth buffering display techniques for constructive solid geometry. *IEEE Computer Graphics and Applications*, **6** (9), 29–39

Rubin S. M. and Whitted T. (1980). A three-dimensional representation for fast rendering of complex schemes. *Computer Graphics*, **14**, 110–16

Schaufler G. and Sturzlinger W. (1996). A 3D image cache for virtual reality. *Proc. Eurographics '96*, August, pp. 227–36

Schlick C. (1993). A customizable reflectance model for everyday rendering. *4th Eurographics Workshop on Rendering*, Puech C. and Sillion F. (eds), Amsterdam: Elsevier, pp. 73–89

Schroeder W. J., Zarge J. A. and Lorenson W. E. (1992). Decimation of triangular meshes. *Proc. SIGGRAPH '92*, 65–70

Schumaker R. A., Brand B., Guilliland M. and Sharp W. (1969). *Applying Computer Generated Images to Visual Simulation*. Technical Report AFHRL-Tr-69, US Airforce Human Resources Lab.

Schweitzer D. and Cobb E. S. (1982). Scan line rendering of parametric surfaces. *Computer Graphics*, **16** (3), 265–71

Sederburg T. W. and Parry S. R. (1986). Free-form deformation of solid geometric models. *Computer Graphics*, **20** (4), 151–60

Seitz S. M. and Dyer C. R. (1996). View morphing. *Proc. SIGGRAPH '96*, pp. 21–30

Shade J., Gortler S., He L. and Szeliski R. (1998). Layered depth images. *Proc. SIGGRAPH '98*

Shade J., Lischinski D., Salesin D., DeRose T. and Snyder J. (1996). Hierarchical image caching for accelerated walkthroughs of complex environments. *Proc. SIGGRAPH '96*, pp. 75–82

Shinya M., Takahashi T. and Naito S. (1987). Principles and applications of pencil tracing. *Computer Graphics*, **21** (4), 45–54. (*Proc. SIGGRAPH '87*).

Shoemake K. (1985). Animating rotation with quaternion curves. *Computer Graphics*, **19** (3), 245–54. (*Proc. SIGGRAPH '85*)

Shoemake K. (1987). Quarternion Calculus and Fast Animation. *SIGGRAPH Course Notes*, **10**, 101–21

Siegel R. and Howell J. R. (1984). *Thermal Radiation Heat Transfer*. Washington DC: Hemisphere Publishing

Smith A. R. (1978). Color gamut transformation pairs. *Computer Graphics*, **12**, 12–19

Speer L. R., DeRose T. D. and Barsky B. A. (1986). A theoretical and empirical analysis of coherent ray tracing. *Computer Generated Images*, **27** (31), 11–25

Snyder J. M. (1992). *Generative Modelling for Computer Graphics*. New York: Academic Press

Snyder J. M. (1998). Visibility sorting and compositing without splitting for image layer decomposition. *Proc. SIGGRAPH '98*, pp. 219–30

Sutherland I. E. and Hodgman G. W. (1974). Reentrant polygon clipping. *Comm. ACM*, **17** (1), 32–42

Sutherland I. E., Sproull R. F. and Schumacker R. (1974). A characterization of ten hidden-surface algorithms. *Computer Surveys*, **6** (1), 1–55

Swanson R. W. and Thayer L. J. (1986). A fast shaded-polygon renderer. *Computer Graphics*, **20** (4), 107–16

Tu S. and Terzopoulos D. (1994). Perceptual modelling for the behavioural animation of fishes. *Proc. Second Pacific Conf. on Computer Graphics* (PG '94), Beijing, China

The Visible Human Project, National Library of Medicine (1998). From: http://www.nlm.nih.gov/pubs/factsheets/visible_human.htn

Wallace J. R., Cohen M. F. and Greenberg D. P. (1987). A two-pass solution to the rendering equation: a synthesis of ray tracing and radiosity methods. *Computer Graphics*, **21** (4), 311–20

Wallace J. R., Kells A. E. and Haines E. (1989). A ray tracing algorithm for progressive radiosity. *Computer Graphics*, **23** (3), 315–24. (*Proc. SIGGRAPH '89*)

Walter B., Hubbard P. M., Shirley P. and Greenberg D. P. (1997). Global Illumination using local linear density estimation. *ACM Trans. on Graphics*, **16** (3) July, pp. 217–59

Ward G. J. (1994). The RADIANCE lighting simulation and rendering system. *Proc. SIGGRAPH '94*, pp. 459–71

Warn D. R. (1983). Lighting controls for synthetic images. *Computer Graphics*, **17** (3), 13–21

Warnock J. (1969). *A Hidden-Surface Algorithm for Computer Generated Half-Tone Pictures*. Technical Report 4–15; NTIS AD-753 671, University of Utah Computer Science Department

Watt A. and Policarpo F. (1998). *The Computer Image*. Harlow: Addison-Wesley

Watt A. and Watt M. (1992). *Advanced Animation and Rendering Techniques.* Wokingham, England: Addison-Wesley

Weghorst H., Hooper G. and Greenberg D. P. (1984). Improved computational methods for ray tracing. *ACM Trans. on Graphics*, **3** (1), 52–69

Weiler K. and Atherton P. (1977). Hidden surface removal using polygon area sorting. *Computer Graphics*, **11** (2), 214–22

Westover L. (1990). Footprint evaluation for volume rendering. *Computer Graphics*, **24** (4), 367–76. (*Proc. SIGGRAPH '90*)

Whitted J. T. (1978). A scan line algorithm for the computer display of curved surfaces. *Proc. 5th Conf. on Computer Graphics and Interactive Techniques*, Atlanta GA, p. 26

Whitted J. T. (1980). An improved illumination model for shaded display. *Comm. ACM*, **23** (6), 342–9

Williams L. (1978). Casting curved shadows on curved surfaces. *Computer Graphics*, **12** (3), 270–4

Williams L. (1983). Pyramidal parametrics. *Computer Graphics*, **17** (3), 1–11

Williams L. and Chen S. E. (1993). View interpolatrioin for image synthgesis. *Proc. SIGGRAPH '93*, pp. 279–88

Witkin A. and Kass M. (1988). Spacetime constraints. *Proc. SIGGRAPH '88*, pp. 159–68

Wolberg G. (1990). *Digital Image Warping.* Los Alamitos CA: IEEE Computer Society Press

Yagel R., Cohen D. and Kaufman A. (1992). Discrete ray tracing. *IEEE Computer Graphics and Applications*, **12** (5), 19–28

Yellott I. (1982). Spectral analysis of spatial sampling by photoreceptors: topological disorder prevents aliasing. *Vision Research*, **22**, 1205–10

Zhukov S., Iones A. and Kronin G. (1998). Using light maps to create realistic lighting in real-time applications. *Proc. WSCG '98.* (Central European Conf. on Computer Graphics and Visualisation 1998)

索　引

索引中的页码为英文原书的页码，与书中边栏的页码一致。

A

a posteriori meshing in radiosity method（辐射度方法中的后网格化），325-331
a priori meshing in radiosity method（辐射度方法中的预网格化），332-341
Abram, A., 403
A-buffer in anti-aliasing （反走样中的A缓冲器方法），403
acceleration due to gravity（重力加速），506
accumulation buffers in graphics pipeline（绘图流程中的累加缓冲器），202-204
adaptive（a posteriori）meshing in radiosity method（辐射度方法中的自适应（后）网格化），325-331
adaptive depth control in Whitted ray tracing（Whitted光线跟踪中的自适应深度控制），354-355
adaptive importance sampling in global illumination（全局照明中的自适应重要性采样），302
adaptive refinement in radiosity method（辐射度方法中的自适应细化），315-318
The Adventures of André and Wally B（André和Wally B历险记），531
affine transformations（仿射变换），2-7
  surface in polygon meshes（多边形网格中的表面），42
Akeley, K., 203
algorithmic processes in graphics pipeline（绘图流程中的算法过程），167-204
  clipping polygons against view volume（对视见体裁剪多边形），168-171
  hidden surface removal（隐藏面消除），189-202
    and BSP trees（和BSP树），199-202
  interpolative shading techniques（插值明暗处理技术），179-183
    Gouraud shading（Gouraud明暗处理），180-181, 183
    Phong shading（Phong明暗处理），181-182, 183
    shading options（明暗处理选项），182
  multi-pass rendering and accumulation buffers（多路绘制和累加缓冲器），202-204
  rasterization（光栅化），183-187
    edges（边界），183-185
    polygons（多边形），185-187
  rendering, order of（绘制顺序），187-189
  shading pixels（像素明暗处理），171-179
    light source considerations（关于光源的考虑），179
    local reflection models（局部反射模型），173-179
  spanning hidden surface removal（跨越隐藏面消除），193-194
  spanning scan line algorithm（跨越扫描线算法），194-196
  Z-buffer algorithm（Z缓冲器算法），189-190
    complex scenes（复杂场景），196-198
    and compositing（和合成），191-192
    and CSG representation（和CSG表示），190-191
    and rendering（和绘制），192-193
    scan line（扫描线），193
aliases /aliasing（走样），228
  and colour（和颜色），435
  and sampling（和采样），393-397
animation in 3D texture domain techniques, Figure 8.22（Colour Plate）（3D纹理域技术中的动画，图8-22（彩色插图）），254-256
animatronic models（动画模型），499-500
anisotropic surface, BRDF from（BRDF中的各向异性的表面），219-221
anti-aliasing（反走样），226-228, 392-417
  aliases and sampling（走样和采样），393-397
  comparisons of（比较），401-402
  Fourier transform of images（图像的傅里叶变换），411-417
  jagged edges（锯齿形边界），397-398
  non-uniform sampling（非均匀采样），406-411
  pre-filtering methods（预过滤方法），402-404
  sampling and reconstruction（采样和重构），400-401
  and sampling reality（和采样真实性），398-400
  and shadow Z-buffer algorithm（和阴影Z缓冲器算法），273-274
  stochastic sampling（随机采样），407, 408-409
  supersampling（post-filtering）（超采样（后过滤）），404-406

and texture mapping, Figure 8.26 (Colour Plate) (和纹理映射, 图8-26 (彩色插图)), 256-260
Appel, A., 267, 342
artefacts in radiosity method(辐射度方法中的人工痕迹), 319-325
hemicube (半立方体), 319-321
meshing (网格化), 323-325
reconstruction (重构), 321-323
articulated structure animation (带关节结构的动画), 476, 493-503
animatronic models in (电子动物模型), 499-500
described (描述), 493-494
forward kinematics in (前向动力学), 495-497
inverse kinematics in (反向动力学), 497-499
solving problems (解题), 500-503
Arvo, J., 297, 299, 363
Atherton, P., 271, 364
automatic generation of polygonal objects (多边形物体的自动产生), 38-39
axis aligned bounding boxes (AABB) in computer animation (计算机动画中与坐标轴标定的限定盒), 519

B

back-face elimination in graphics pipeline (绘图流程中的背面清除), 147-148
Badler, N. I., 268, 270, 503
Balakrishnan, A., 410
Barsky, B., 79
Bartels, R., 79, 111, 112, 116
Bartlett windows (Bartlett窗口), 405-406
beam tracing (光束跟踪), 365
Beatty, J., 79
behavioural animation (行为动画), 477, 531-534
bending (弯曲), 10-11
global linear (全局线性), 9
Bergeron, P., 270
Bernstein cubic polynomials (Bernstein 三次多项式), 67, 72
Bézier curves (Bézier曲线), 69-77
control points (控制点), 71
control polygon for (控制多边形), 73
joining segments (连接片段), 75-77, 98-100
properties (属性), 77
rational (有理的), 91-93
Bézier patch object (Utah Teapot) (Bézier 曲面片物体 (Utah茶壶)), 100-101
Bézier patches (Bézier曲面片), 98-100
bi-cubic parametric patch objects, Figure 8.8 (Colour Plate) (双三次参数曲面片物体, 图8-8 (彩色插图)), 234
bi-cubic parametric patch representation (双三次参数曲面片表示), 30-31, 66-122
advantages (优点), 67
Bézier curves (Bézier曲线), 69-77
joining segments (连接片段), 75-77
properties (属性), 77
rational (有理的), 91-93
B-spline surface patches (B样条表面曲面片), 101-106
B-splines (B样条), 78-90
curves (曲线), 78-80
non-uniform (非均匀), 84-90
properties (属性), 90
uniform (均匀), 80-84
curves to surfaces (曲线到表面), 94-101
Bézier patch object (Utah Teapot) (Bézier 曲面片物体 (Utah茶壶)), 100-101
Bézier patches (Bézier曲面片), 98-100
continuity patches (连续曲面片), 98-100
patch surfaces (曲面片表面), 106-121
control polyhedron design (控制多面体设计), 110-115
linear axis design (线性轴设计), 107-110
and surface fitting (和表面拟合), 115-121
patches to objects (曲面片到物体), 121-122
rational curves (有理曲线), 90-94
Bézier, 91-93
NURBS, 93-94
rendering (绘制), 125-138
image space subdivision (图像空间细分), 135-138
object space subdivision (物体空间细分), 128-135
patch descriptions (曲面片描述), 125-128
patch to polygon conversion(曲面片到多边形转换), 128
bi-directional reflectance distribution function (BRDF), Figure 7.8 (Colour Plate) (双向反射分布函数, 图7-8 (彩色插图)), 208-211
in global illumination (在全局照明中), 278, 305
in caching illumination (在存储照明中), 302
in path tracing (在路径跟踪中), 293
isotropic and anisotropic surfaces (各向同性和各向异性表面), 209
pre-computing (预计算), 219-221
Bier, E.A., 230
billboards (广告牌), 235-236, 445

binary space partitioning (BSP) trees (二叉空间分割树), 55-56
and hidden surface removal (和隐藏面消除), 199-202
in Whitted ray tracing (在Whitted 光线跟踪中), 363
Blau, R., 530
Blinn, J., 125, 126, 212, 213-219, 225, 236, 243, 247, 266
Blythe, D., 238
Boolean set operators in CSG representation (CSG表示中的布尔集合运算符), 47-50
Bouknight, W.J., 267
bouncing ball problem (弹跳球问题), 478
bounding objects in Whitted ray tracing (Whitted光线跟踪中的限定物体), 355-357
bounding volume hierarchies in Whitted ray tracing (Whitted光线跟踪中的限定体层次结构), 357-358
bounding volumes (限定体), 56
box (盒), 21-22
Bresenham, J.E., 183
Brotman, L.S., 268, 270
B-spline surface patches (B样条表面曲面片), 101-106
B-splines (B样条)
curves (曲线), 78-80
knot values (结点值), 80-82
non-localness of (非局部的), 78-79
non-uniform (非均匀), 84-90
properties (属性), 90
surface fitting of patch objects (曲面片物体的表面拟合), 115-117
uniform (均匀), 80-84
bump mapping, Figure 8.10 (Colour Plate) (凹凸映射, 图8-10 (彩色插图)), 225, 236-240
multi-pass technique (多路技术), 238-239
pre-calculation technique (预计算技术), 239-240

C

Cabral, B., 219, 492
caching illumination (存储照明), 301-303
camera coordinate systems in graphics pipeline (绘图流程中的摄像机坐标系), 143-146
Carpenter, L.C., 130, 134, 192, 198, 403
Catmull, E., 135, 189, 198, 234, 402
Chen, S.E., 458-459
Chiyokura, H., 121
CIE XYZ space, colour in (CIE XYZ 颜色空间), 422, 427, 429-433
evolution of (演变), 431
CIE xyY space, colour in, Figure 15.10 (Colour Plate) (CIE xyY颜色空间, 图15-10 (彩色插图)), 433-435
Clark, J.H., 59, 130, 134, 135
Clifton, M., 247
clip planes (裁剪平面), 147-149, 157
clipping (裁剪)
in graphics pipeline (在绘图流程中), 147-149
against view volume (与视见体), 168-171
in Whitted ray tracing (在Whitted光线跟踪中), 365
Cobb, E.S., 125
Cohen, M.F., 314, 325, 505, 519
collision avoidance in behavioural animation (行为动画中的避免碰撞), 532
collision detection in computer animation (计算机动画中的碰撞检测), 517-526
broad phase, with OBBs (OBB的非限制性阶段), 519-522
broad phase/narrow phase algorithms (非限制性阶段/限制性阶段算法), 518-519
collision response (碰撞响应), 526-529
exact detection (准确检测), 522-524
narrow phase (限制性阶段), 522-524
object hierarchies (物体层次结构), 524-526
collision response in computer animation (计算机动画中的碰撞响应), 526-529
colour (颜色), 418-442
and 3D space (和3D空间), 420-427
HSV single hexcone model, Figure 15.5 (Colour Plate) (HSV单六角锥体模型, 图15-5 (彩色插图)), 424-427
RGB space, Figure 15.3 (Colour Plate) (RGB空间, 图15-3 (彩色插图)), 423-424
YIQ space (YIQ空间), 427
hierarchy of sets (集合的层次), 420
information and perceptual spaces (信息和感知空间), 427-435
CIE XYZ space (CIE XYZ空间), 429-433
CIE xyY space, Figure 15.10 (Colour Plate) (CIE xyY 空间, 图15-10 (彩色插图)), 433-435
in local reflection models (在局部反射模型中), 177
modulation of (调节), 225
monitor considerations (关于监视器的考虑), 436-442
colour gamut mapping (颜色显示范围映射), 439-440
different monitor, same colour (不同的监视器, 相同的颜色), 437-439
gamma correction (gamma校正), 440-442

RGB$_{monitor}$ space (RGB$_{监视器}$空间), 436-437
and rendering, Figure 15.12 (Colour Plate) (和绘制, 图15-12 (彩色插图)), 435-436
sets in computer graphics (计算机绘图中的集合), 419-420
colour aliasing (颜色走样), 435
colour gamut mapping in monitors (监视器中的颜色全范围映射), 439-440
compositing, Z-buffer algorithm in (Z缓冲器算法中的合成), 191-192
computer animation (计算机动画), 473-535
collision detection (碰撞检测), 517-526
broad phase, with OBBs (OBB中的非限制性阶段), 519-522
broad phase /narrow phase algorithms (非限制性阶段/限制性阶段算法), 518-519
collision response (碰撞响应), 526-529
exact detection (准确检测), 522-524
narrow phase (限制性阶段), 522-524
object hierarchies (物体层次结构), 524-526
techniques (技术), *参见* articulated structure
animation; behavioural (行为动画)
simulation; dynamic (动力学模拟)
animation; particle (粒子动画)
animation; rigid body (刚体动画)
use of (采用), 474
computer monitors and colour (计算机监视器和颜色), 436-442
colour gamut mapping (颜色全范围映射), 439-440
different monitor, same colour (不同的监视器, 相同的颜色), 437-439
gamma correction (gamma校正), 440-442
RGB$_{monitor}$ space (RGB$_{监视器}$空间), 436-437
Computer Aided Design (CAD) (计算机辅助设计), 66-67
Computer Aided Graphics Design (CAGD) (计算机辅助图形设计), 68
constructive solid geometry (CSG) representation (结构实体几何表示), 29, 31, 32, 46-50
Boolean set operators in (布尔集合运算符), 47-50
primitives in (基元), 47
rendering (绘制), 138-140
volume rendering in (体绘制), 373
Z-buffer algorithm in (Z缓冲器算法), 190-191
continuity patches (连续曲面片), 98-100
control polygon for Bézier curves (Bézier曲线的控制多边形), 73
control polyhedron design of patch surfaces (曲面片表面的控制多面体设计), 110-115
coarse control (粗线条控制), 113-115
fine control (精细控制), 111-113
scale (刻度), 111
convex hull flatness test (凸包平整性测试), 133
convex polygon (凸多边形), 19-20
convex polyhedra, collision detection of (碰撞检测的凸多面体), 522-524
convolution theorem, and Fourier transform (卷积理论, 傅里叶变换), 417
Cook, R.L., 198, 212, 213-219, 222, 234, 294-295, 409, 411
coordinate spaces in graphics pipeline (绘图流程中的坐标空间), 143-146
camera (eye, view) coordinate systems (摄像机 (眼睛, 视图) 坐标系), 143-146
local or modelling systems (局部或建模系), 143
world coordinate systems (世界坐标系), 143
corrugated cylinder, twisted and tapered Figure 1.5 (Colour Plate) (起皱的、扭曲的、一头变细的圆柱体, 图1-5 (彩色插图))
Cowan, W.B., 441
Cox, M.G., 90
cross products (叉积), 12-14
cross-sectional sweeping (截面扫掠), 42
Crow, F.C., 100, 268, 405
cubic Bézier patch (三次Bézier曲面片), 94
cubic mapping (三次映射), 245-247
cubic splines (三次样条), 74
culling in graphics pipeline (绘图流程中的挑选), 147-148
curved spines (弯曲的样条), *参见spine curve*

D

damping forces in dynamic simulation (动力学模拟中的阻尼力), 506
De Boor, C., 90
Debevec, P.E., 470-472
density estimation in global illumination (全局照明中的密度估算), 304
Descartes, R., 367
diffuse components, reflection from (反射的漫反射分量), 211-212
diffuse interaction (漫反射相互作用), 306
diffuse to diffuse transfer in global illumination (全局照明中的从漫反射到漫反射传递), 281-283
diffuse to specular transfer in global illumination (全局照

明中的从漫反射到镜面反射传递），281-283
Digital Differential Analyzer in Whitted ray tracing（Whitted 光线跟踪中的数字微分分析器），362
discontinuity meshing in radiosity method（辐射度方法中的不连续网格化），337-341
displacement mapping（移位映射），225
distortion in 3D screen（3D屏幕中的变形），154
distributed ray tracing in global illumination, Figure 10.14（Colour Plate）（全局照明中的分布式光线跟踪，图10-14（彩色插图）），294-296
dot products（点积），14-15
Drebin, R.A., 378
Duff, T., 191, 492
Dyer, C.R., 460-462
dynamic simulation, Figure 17.20（Colour Plate）（动力学模拟，图17-20（彩色插图）），476, 504-517
  forces, nature of（力的性质），506-507
  lumped mass, simulating dynamics（块状物质，模拟动力学），511-515
  particles in basic theory（基本理论中的粒子），505-506
  rigid bodies-extended masses（刚体——延伸的质量），507-510
  space-time constraints（空间-时间约束），515-517
  use in computer animation（在计算机动画中的使用），510-511

E

edge-edge-edge events in radiosity method（辐射度方法中的边-边-边事件），338-341
edges（边）
  attributes（属性），36
  in polygon mesh optimization（在多边形网格优化中）
    collapse（倒塌），61-63
    split（分离），61
    swap（交换），61
  in polygonal mesh representation（在多边形网格表示中），34, 35
  rasterization（光栅化），183-185
elements in radiosity method（辐射度方法中的元素），323
environment mapping（环境映射），243-250
  comparative points（比较点），248-249
  cubic mapping（三次映射），245-247
  sphere mapping（球面映射），247-248
  surface properties and（表面性质和），249-250
exact collision detection in computer animation（计算机动画中的准确碰撞），522-524
explicit scripting in rigid body animation（刚体动画中的明确的脚本），479-483
extended masses in dynamic simulation（动力学模拟中有翼展的物质），507-510
eye coordinate systems in graphics pipeline（绘图流程中的眼睛坐标系），143-146
eye space, transformations to screen space（眼睛空间变换到屏幕空间），154

F

Farin, G., 100, 111, 116, 120
Faux, I.D., 100
field radiance in global illumination（全局照明中的场辐射光亮度），279
filtering, none, in anti-aliasing（过滤，反走样），401
Fiume, E., 403
flat shaded polygons（平面明暗处理的多边形），182
flock centring in behavioural animation（行为动画中的兽群集中），532
flocking behaviour, modelling（兽群行为，建模），531-534
Foley, J.D., 190, 194
forces, nature of in dynamic simulation（动力学模拟中力的性质），506-507
form factor determination in radiosity method（辐射度方法中的形状因子确定），310-314
Forsey, D.R., 111, 112
forward kinematics in articulated structure animation（带关节结构的动画中的前向动力学），495-497
Fourier transform of images（图像的傅里叶变换），411-417
Fournier, A., 44-45
fractal objects（分形物体），44-46
free form deformation（自由形状分解），115
Frenet frame（Frenet 帧），43
Fresnel term in specular reflection（在镜面反射中的 Fresnel 项），216-219
Fuchs, H., 360
Fujimoto, A., 359, 360, 361
fuzzy objects, modelling（模糊物体，建模），529

G

gamma correction in colour monitors（彩色监视器中的γ校正），440-442
gathering in radiostity method（辐射度方法中的收集），316-317
Gauss-Seidel radiosity method（Gauss-Seidel辐射度方

法)，314-315
geometric operations in graphics pipeline（绘图流程中的几何运算)，142-166
advanced viewing systems（先进的观察系统)，参见PHIGS
coordinate spaces（坐标空间)，143-146
camera（eye, view）coordinate systems（摄像机（眼睛，视图）坐标系)，143-146
local or modelling systems（局部或建模系)，143
minimum coordinate system（最小坐标系)，144
world coordinate systems（世界坐标系)，143
operations in view space（观察空间中的运算)，147-156
culling（挑选)，147-148
3D screen space（3D屏幕空间)，149-152
view volume（视见体)，147-149
view volume and depth（视见体和深度)，152-156
glare effect（闪烁效果)，212, 216
Glassner, A.S., 288, 359, 360, 361
global axial twisting, Figure 1.5（Colour Plate)（全局坐标轴扭曲，图1-5（彩色插图))，9
global illumination（全局照明)，275-305
algorithms（算法)
evolution of（的发展)，283-284
radiosity, Figure 10.7（Colour Plate)（辐射度，图10-7（彩色插图))，276, 286-288
Whitted ray tracing（Whitted 光线跟踪)，284-286
caching illumination（存储照明)，301-303
density estimation（密度估算)，304
distributed ray tracing, Figure 10.14（Colour Plate)（分布式光线跟踪，图10-14（彩色插图))，294-296
light volumes（光线体)，303-304
mesh optimization（网格优化)，305
models（模型)，276-283
path notation, Figure 10.2（Colour Plate)（路径标注，图10-2（彩色插图))，281-283
radiance, irradiance and radiance equation（辐射光亮、辐照度和辐射光亮度方程)，278-281
rendering equation（绘制方程)，277-278
Monte Carlo techniques（蒙特卡罗技术)，288-291
multi-pass methods（多路方法)，300-301
particle tracing（粒子跟踪)，304
path tracing（路径跟踪)，288, 292-293
two-pass ray tracing, Figure 10.20（Colour Plate)（二路跟踪，图10-20（彩色插图))，297-300
view dependence/independence（依赖于观察的/独立于观察的)，参见Whitted ray tracing，300-301
global linear bend（全局线性弯曲)，9
Goral, C., 306
Gortler, S., 463
Gottschalk, S., 519, 521-522
Gouraud, H., 171, 182, 238
Gouraud shading（Gouraud明暗处理)，171, 186
interpolative shading techniques（插值明暗处理技术)，180-181, 183
in local reflection models, Figure 18.3（Colour Plate)（局部反射模型中，图18-3（彩色插图))，537
graphics pipeline（绘图流程)，参见algorithmic processes; geometric operations
Greenberg, D.P., 314, 345-346, 354, 363
Greene, N., 197, 243
Greger, G., 279, 304
Griffiths, J., 125, 128
ground plane, shadows on（阴影所在的地平面)，265-266

H

Haeberli, P., . 203, 247, 261, 391
Haines, E.A., 20, 345-346, 363
Hall, R.A., 354, 439
Hallas, J., 474
Halstead, M., 247
Hanrahan, P., 212, 221-222, 261, 332, 334, 364, 463, 465
He, L., 210
Heckbert, P.S., 226, 228, 282, 364
hemicube algorithm in radiosity method, Figure 11.5（Colour Plate)（辐射度方法中的半立方体算法，图11-5（彩色插图))，307, 310-314
hemicube artefacts in radiosity method（辐射度方法的半立方体人工痕迹)，319-321
hidden surface removal（隐藏面消除)
in graphics pipeline（在绘图流程中)，189-202
and BSP trees（和BSP树)，199-202
in Whitted ray tracing（在Whitted 光线跟踪中)，346-347
hierarchical radiosity（层次化的辐射度)，332-337
Hodgman, G.W., 169
Hoffman, C.R., 250
holes in IBR, Figure 16.8（Colour Plate)（IBR中的孔洞，图16-8（彩色插图))，454
homogeneous coordinates（齐次坐标系)，4
Hoppe, H., 61-64
Howell, J.R., 308
HSV single hexcone model, colour in, Figure 15.5（Colour Plate)（在HSV单六角锥体模型中的颜色，图15-5（彩色插图))，423, 424-427
Hubbard, P.M., 518-519, 524-526

I

image comparisons（图像比较），536-543
  local reflection models, Figures 18.1, 18.2, 18.3, 18.5, 18.6（Colour Plates（局部反射模型，图18-1，18-2，18-3，18-5，18-6（彩色插图）），537-538
  RADIANCE, Figure 18.19（Colour Plate）（RADIANCE，图18-19（彩色插图）），543
  radiosity, Figures 18.14, 18.15, 18.16, 18.17, 18.18（Colour Plates）
（辐射度，图18-14，18-15，18-16，18-17，18-18（彩色插图）），541-543
  texture and shadow mapping, Figures 18.7, 18.8, 18.9（Colour Plates）
（纹理和阴影映射，图18-7，18-8，18-9（彩色插图）），538-539
  Whitted ray tracing, Figures 18.10, 18.11, 18.12, 18.13（Colour Plates）（Whitted光线跟踪，图18-10，18-11，18-12，18-13（彩色插图）），539-541
image folding in IBR, Figure 16.8（Colour Plate）（IBR中的图像打褶，图16-8（彩色插图）），448-451
image layering in IBR（IBR中的图像分层），448-451
  ordering in depth（深度排序），451
  validity of（其有效性），450-451
image place, interpolating properties（图像位置，插值性质），25-26
image power, and Fourier transform（图像能量和傅里叶变换），416
image space subdivision, rendering（图像空间细分，绘制），135-138
image-based rendering（IBR）（基于图像的绘制），443-465
  depth information（深度信息），452-458
    3D warping（3D变形），452-456
    layer depth images（分层深度图像），456-458
  light field rendering（光线场绘制），463-465
  rendered imagery, reuse of（绘制后图像的再利用），444-447
    planar imposters（平面烘托物），445
    validity of planar imposters（平面烘托物的有效性），445-447
  varying resources for（改变资源），447-451
    image layering（图像分层），448-451
    layers, validity of（层次的有效性），450-451
    ordering layers in depth（按深度对层次排序），451
    priority rendering（优先绘制），447-448
  view interpolation（观察插值），458-463
    view morphing（观察变形），460-463
参见photo-modelling
imperfect specular reflection（不完全镜面反射），174
imperfect surface, reflection from（反射的不完全表面），206-207
implicit representation, Figure 2.20（Colour Plate）（隐含表示，图2-20（彩色插图）），31-32, 56-58
  rendering functions, Figure 2.20（Colour Plate）（绘制函数，图2-20（彩色插图）），141
importance sampling in global illumination（全局照明中的重要性采样），302
interactive techniques in texture mapping（纹理映射中的交互技术），260-262
interative texture mapping（交互式纹理映射），261
interpolation（keyframing）in rigid body animation（刚体动画中的插值（关键帧）），477-479
interpolative shading techniques in graphics pipeline（绘图流程中的插值明暗处理技术），179-183
  Gouraud shading（Gouraud 明暗处理），180-181, 183
  Phong shading（Phong 明暗处理），181-182, 183
  shading options（明暗处理选项），182
intersections（相交）
  and rays（和光线），17-23
    box（盒），21-22
    convex polygon（凸多边形），19-21
    quadric（二次的），23
    sphere（球），18-19
inverse kinematics in articulated structure animation（带关节结构的动画中的反向动力学），497-499
  solving problems（解题），500-503
inverse mapping（反向映射）
  by bilinear interpolation（通过双线性插值），229-230
  by using an intermediate surface, Figure 8.7（Colour Plate）（通过采用一个中间表面，图8-7（彩色插图）），230-234
irradiance in global illumination（全局照明中的辐照度），278-281
isosurfaces, explicit extraction in volume rendering, Figure 13.10 (Colour Plate), Figure 13.11 (Colour Plate),（体绘制中的等值面，直接抽取，图13-10，13-11（彩色插图）），382-384
isotropic surface, BRDF from（BRDF 的各向同性表面），219-221

J

jagged edges and anti-aliasing（锯齿形边和反走样），397-398
jittering in anti-aliasing（反走样中的颤抖），408-409
Jurassic Park（侏罗纪公园），475, 493, 499

K

Kajiya, J.T, . 277, 288, 291-293
Kaplan, M.R., 359, 360
Kass, M., 504, 516
Kelly, K., 267
keyframing in rigid body animation (刚体动画中的关键帧), 477-479
Kirk, D., 363
knot values in B-splines (B样条中的结点值), 80-82
knot vectors in B-splines (B样条的结点向量), 85
Kreuger, W., 212, 221-222

L

Lane, J.M., 125, 128, 130, 131, 134
Lasseter, J., 504, 516
layered depth images (LDI) in IBR (IBR中深度分层的图像), 456-458
left-handed systems (左手系), 2
Lengyel, J., 448-451
level of detail (LoD) (详细程度), 58, 60-61
Levoy, M., 376, 463, 465
Lewis, J.P., 252
light field rendering in IBR (IBR 中的光线场绘制), 463-465
light leakage in radiosity method (辐射度方法中的光线泄漏), 324
light maps, Figure 8.14 (Colour Plate) (光线图, 图8-14 (彩色插图)), 240-242
  three-dimensional (三维的), 256
light volumes in global illumination (全局照明中的光线体积), 303-304
lighting components in Whitted ray tracing (Whitted 光线跟踪中的光线分量), 344
lighting environment and shadows (光照环境和阴影), 264
linear axis design of patch surfaces (曲面片表面的线性轴设计), 107-110
  non-circular scaled cross-sections (非环状带刻度的截面), 108
  non-circular varying cross-sections (非环状变化的截面), 109-110
  scaled circular cross-sections (带刻度的环状截面), 107-108
linked structures (链接结构), 参见articulated structure animation
The Lion King (狮子王), 475
Lippman, A., 467
Lischinski, D., 339, 340-341
Liu, Z., 517
local reflection models (局部反射模型), 173-179, 205-222
  bi-directional reflectance distribution function, Figure 7.8 (Colour Plate) (双向反射分布函数, 图7-8 (彩色插图)), 208-211
    pre-computing (预计算), 219-221
  comparisons, Figures 18.1, 18.2, 18.3, 18.5, 18.6 (Colour Plates) (比较, 图18-1, 18-2, 18-3, 18-5, 18-6 (彩色插图)), 537-538
  diffuse components (漫反射分量), 211-212
  from imperfect surface (来自不完全表面), 206-207
  light source considerations (关于光源的考虑), 179
  perfect diffuse-empirically spread specular reflection (完全漫反射——经验地分布镜面反射), 212-213
  from perfect surface (来自完全表面), 206-207
  physically based diffuse component (物理地基于漫反射分量), 221-222
  physically based specular reflection (物理地基于镜面反射), 213-219
    Fresnel term (Fresnel 项), 216-219
    masking effects (屏蔽效果), 214-216
    micro-geometry of surface (表面的微观几何), 214
    shadowing effects (阴影效果), 214-216
    viewing geometry (观察几何), 216
  practical points (实际点), 177-179
  specular components (镜面分量), 211-212
local systems in graphics pipeline (绘图流程中的局部系统), 143
Lorenson, W.E., 382
Lumigraph in IBR (IBR中的Lumigraph), 463-465
lumped mass, simulating dynamics (块状物质, 模拟动力学), 511-515
Luxo Jr., 504, 516

M

Maciel, P.W., 445
Mandelbrot, B., 44
Manocha, D., 519, 521-522
Mantyla, M., 36, 341
manual modelling (手工建模), 38
Manvell, R., 474
mapping techniques (映射技术), 223-262
  anti-aliasing, Figure 8.26 (Colour Plate) (反走样, 图8-26 (彩色插图)), 256-260

billboards（广告牌），235-236
bump mapping, Figure 8.10（Colour Plate）（凹凸映射，图8-10（彩色插图）），236-240
multi-pass technique（多路技术），238-239
pre-calculation technique（预计算技术），239-240
cubic mapping（三次映射），245-247
3D texture domain techniques（3D纹理域技术），251-256
and animation, Figure 8.22（Colour Plate）（和动画，图8-22（彩色插图）），254-256
3D noise（3D噪声），251-252
light maps（光线图），256
turbulence, Figure 8.21（Colour Plate）（扰动，图8-21（彩色插图）），252-254
2D texture maps（2D纹理图），228-234
bi-cubic parametric patch objects, Figure 8.8（Colour Plate）（双三次参数曲面片物体，图8-8（彩色插图）），234
inverse mapping, Figure 8.7（Colour Plate）（反向映射，图8-7（彩色插图）），229-234
environment（reflection）mapping（环境（反射）映射），243-250
comparative points（比较点），248-249
surface properties（表面性质），249-250
interactive techniques（交互技术），260-262
light maps, Figure 8.14（Colour Plate）（光线图，图8-14（彩色插图）），240-242
3D, 256
sphere mapping（球映射），247-248
marching cubes algorithm（步进式立方体算法），140-141, 383
Mark, W.R., 456
masking effects in specular reflection（镜面反射中的屏蔽效果），214-216
mathematical fundamentals（数学基础），1-26
mathematical generation of polygonal objects（多边形物体的数学产生），39-44
Max, N., 321
McMillan, L., 453, 457
McReynolds, T., 238
mesh compression（网格压缩），61
mesh optimization in global illumination（全局照明中的网格优化），305
mesh simplification（网格简化），60
meshing in radiosity method（辐射度方法中的网格化），323
artefacts（人工痕迹），323-325
discontinuity（不连续性），337-341
strategies（策略），325-341
adaptive（a posteriori）（自适应（后）），325-331
discontinuity meshing（不连续网格化），337-341
hierarchical radiosity（层次化辐射度），332-337
a priori（预），332-341
microfacet reflecting surfaces（具有微表面的反射表面），215
Miller, G.S., 247, 250
mip-mapping（mip映射），257
model-based stereo（基于模型的立体照片），470
modelling systems in graphics pipeline（绘图流程中的建模系统），143
Monte Carlo techniques in global illumination（全局照明中的蒙特卡罗技术），288-291
Moore, M., 522
morphing（变形），474
multi-pass methods in global illumination（全局照明中的多路绘制），300-301
multi-pass rendering in graphics pipeline（绘图流程中的多路绘制），202-204
multi-pass technique for bump mapping（块映射的多路技术），238-239
Munsell, A.H., 425

N

Nakamae, E., 192, 338, 340
Newell, M., 100, 189, 247
Newell, R.G., 189
Newman, W., 152
Newton, I., 421
Ney, D., 377
Nishita, T., 338, 340
noise in 3D texture domain techniques（3D纹理域技术中的噪声），251-252
non-uniform B-splines（非均匀B样条），84-90
non-uniform rational B-splines(NURBS)（非均匀有理B样条），84-90
non-uniform sampling in anti-aliasing（反走样中的非均匀采样），406-411
non-uniform subdivision of object space（物体空间的非均匀细分），130
testing for, Figure 4.9（Colour Plate）（测试，图4-9（彩色插图）），133
normal vector perturbation（法向量扰动），225
Novins, K. L.，390
Nyquist limit in anti-aliasing（反走样中的Nyquist限制），396, 408, 409

O

object hierarchies in collision detection (碰撞检测中的物体层次), 524-526
object representation and modelling (物体表示和建模), 66-122
  Bézier curves (Bézier 曲线), 69-77
    joining segments (连接片段), 75-77
    properties (性质), 77
    rational (有理的), 91-93
  B-spline surface patches (B样条表面曲面片), 101-106
  B-splines (B样条), 78-90
    curves (曲线), 78-80
    non-uniform (非均匀), 84-90
    properties (性质), 90
    uniform (均匀), 80-84
  curves to surfaces (曲线到表面), 94-101
    Bézier patch object (Utah Teapot) (Bézier 曲面片物体 (Utah茶壶)), 100-101
    Bézier patches (Bézier曲面片), 98-100
    continuity patches (连续曲面片), 98-100
  patch surfaces (曲面片表面), 106-121
    control polyhedron design (控制多面体设计), 110-115
    linear axis design (线性轴设计), 107-110
    and surface fitting (和表面拟合), 115-121
  patches to objects (曲面片到物体), 121-122
  rational curves (有理曲线), 90-94
    Bézier, 91-93
    NURBS, 93-94
object space subdivision, rendering (物体空间细分, 绘制), 128-135
octrees (八叉树), 51-52, 53-55
  in Whitted ray tracing (在Whitted光线跟踪中), 360-361
    attributes (属性), 362
    tracking rays (跟踪光线), 361-363
offset surfaces (偏移表面), 122
Oppenheim, A.V., 405
Oppenheimer, P.E., 254
oriented bounding boxes (OBB) (定向的限定盒)
  broad phase collision detection with (非限制性的相位碰撞检测), 519-522
oriented bounding boxes (OBB) in computer animation (在计算机动画中的定向的限定盒), 519

P

painting in texture mapping (纹理映射中的绘画), 261-262
panoramas in IBR (IBR中的全景图)
  compositing, (合成) 469-470
  photographic, Figure 16.19 (Colour Plate) (摄影的, 图16-19 (彩色插图)), 469
parallel projections (平行投影), 149-150, 161
Parry, S.R., 115
particle animation (粒子动画), 476, 529-531
particle tracing in global illumination (全局照明中的粒子跟踪), 304
particles in dynamic simulation (动力学模拟中的粒子), 505-506
  constraints (约束), 506
patch surfaces (曲面片表面), 106-121
  control polyhedron design (控制多面体设计), 110-115
  linear axis design (线性轴设计), 107-110
    non-circular scaled cross-sections (非环状带刻度的截面), 108
    non-circular varying cross-sections (非环状变化的截面), 109-110
    scaled circular cross-sections (带刻度的环状截面), 107-108
  to objects (对于物体), 121-122
  rendering to polygon (从绘制到多边形), 128
  screen space flatness of (屏幕空间平整度), 136-137
  and surface fitting (和表面拟合), 115-121
patches in radiosity method (在辐射度方法中的曲面片), 323
path notation in global illumination, Figure 10.2 (Colour Plate) (全局照明中的路径标注, 图10-2 (彩色插图)), 281-283
path tracing in global illumination (全局照明中的路径跟踪), 288, 292-293
Peachey, D.R., 251
Peercy, M., 239
penumbra (半影), 264
  in radiosity (在辐照度中), 340
perfect diffuse-empirically spread specular reflection (完全漫反射 - 经验地分布镜面反射), 212-213
perfect diffuse reflection (完全漫反射), 174
perfect specular reflection (完全镜面反射), 174
perfect surface, reflection from (从完全表面反射), 206-207
Perlin, K., 251, 252, 254
perspective projections (透视投影), 149-151, 161
PHIGS viewing system (PHIGS 观察系统), 156-166
  implementing (执行), 164-166
  orientation parameters (方向参数), 159
  overview (概述), 157-159
  view mapping parameters (观察映射参数), 159-162

view plane（观察平面），162-164
Philips, C.B., 503
Phong, B., 171, 182, 212
Phong reflection model, Figure 6.12（Colour Plate）（Phong 反射模型，图6-12（彩色插图）），174, 186
Phong shading（Phong 明暗处理），171
interpolative shading techniques（插值明暗处理技术），181-182, 183
in local reflection models, Figure 18.5（Colour Plate）（在局部反射模型中，图18-5（彩色插图）），538
photogrammetric modelling（摄影学建模），470
photo-modelling（照片建模），465-472
compositing panoramas（合成全景图），469-470
for IBR（为 IBR），470-472
photographic panoramas, Figure 16.19（Colour Plate）（照片全景图，图16-19（彩色插图）），469
physically based specular reflection（物理地基于镜面反射），213-219
Fresnel term（Fresnel 项），216-219
masking effects（遮蔽效果），214-216
micro-geometry of surface（表面的微观几何），214
shadowing effects（阴影效果），214-216
viewing geometry（观察几何），216
Piegel, L., 69, 71
pixels（像素）
compositing along ray in volume rendering（在体绘制中沿着光线合成），379-380
shading in graphics pipeline（绘图流程中的明暗处理），171-179
light source considerations（关于光源的考虑），179
local reflection models（局部反射模型），173-179
planar imposters in IBR（IBR中的平面烘托物），444, 445
validity of（有效性），445-447
planar projections（平面投影），149
polygon objects, ray tracing（多边形物体，光线跟踪），352-354
polygonal mesh representation（多边形网格表示），28, 30, 33-46
attributes（属性），36
automatic generation（自动产生），38-39
creating objects（产生物体），37-38
disadvantages（缺点），28-29
fractal objects（分形物体），44-46
manual modelling（手工建模），38
mathematical generation（数学产生），39-44
optimization（优化），59-64
rendering（绘制），124-125
supersampling in（超采样），404
two-dimensional texture maps to（二维纹理映射），228-234
polygons, rasterization（多边形，光栅化），185-187
Porter, T., 191
Pose, R., 447-448
post-filtering in anti-aliasing（反走样中的后过滤），402, 404-406
pre-calculation technique（预计算技术）
for bump mapping（凹凸映射的），239-240
in texture mapping（在纹理映射中），257
pre-filtering in anti-aliasing（反走样中的预过滤），401, 402-404
priority rendering in IBR（IBR 中的优先绘制），447-448
progressive refinement in radiosity method（辐射度方法中的渐进式细化），315-318
progressive transmission（渐进式传播），61
projection reference point in PHIGS（PHIGS中的投影参考点），157
projective space（投影空间），90

Q

quadric（二次的），22-23
quadtrees（四叉树），54, 56
quaternions in rigid body animation interpolation of（在刚体动画插值中的四元组），488-492
quaternion space, moving in and out of（四元组空间，进出移动），488-489
spherical linear interpolation（球形的线性插值），489-492

R

radiance equation in global illumination（全局照明中的辐射光亮度方程），278, 280
radiance in global illumination（全局照明中的辐射光亮度），278-281
RADIANCE renderer（RADIANCE 绘制程序），241, 299, 303
comparisons, Figure 18.19（Colour Plate）（比较，图18-19（彩色插图）），543
radiosity（辐射度），306-341
artefacts（人工痕迹），319-325
hemicube（半立方体），319-321
meshing（网格化），323-325
reconstruction（重构），321-323
comparisons, Figures 18.14, 18.15, 18.16, 18.17,

18.18 (Colour Plates) (比较，图18-14，18-15，18-16，18-17，18-18 (彩色插图))，541-543
form factor determination, Figure 11.5 (Colour Plate) (形状因子确定，图11-5 (彩色插图))，310-314
Gauss-Seidel method (Gauss-Seidel方法)，314-315
in global illumination, Figure 10.7 (Colour Plate) (在全局照明中，图10-7 (彩色插图))，276, 286-288
meshing strategies (网格化策略)，325-341
adaptive (a posteriori) (自适应 (后))，325-331
discontinuity meshing (不连续网格化)，337-341
hierarchical radiosity (层次化的辐射度)，332-337
a priori (预)，332-341
problems with method (带有方法的问题)，318-319
progressive refinement (渐进式细化)，315-318
theory (理论)，308-310
rail product surface (轨迹产品表面)，40-41
rainbow, optics of (彩虹的光学)，367-369
rasterization (光栅化)
in graphics pipeline (在绘图流程中)，183-187
edges (边界)，183-185
polygons (多边形)，185-187
rational curves (有理曲线)，90-94
Bézier, 91-93
NURBS, 93-94
ray casting in volume rendering (在体绘制中的光线投射)，375, 379, 385-388
transformed data (变换后的数据)，387-388
untransformed data (未经变换的数据)，385-387
ray coherence in Whitted ray tracing (Whitted光线跟踪中的光线连贯性)，364-367
ray compositing in volume rendering (体绘制中的光线合成)，380
ray space subdivision in Whitted ray tracing (Whitted 光线跟踪中的光线空间细分)，363-364
ray tracing (光线跟踪)
geometry (几何)，22-25
in global illumination (在全局照明中)，276
in spatial subdivision, representation (空间细分中的表示)，53-55
see also whitted ray tracing (参见Whitted 光线跟踪)
rays (光线)，17-25
and intersections (和相交)，17-23
box (盒)，21-22
convex polygon (凸多边形)，19-21
quadric (二次的)，23
sphere (球的)，18-19
reconstruction (重建)，398-399
artefacts in radiosity method (辐射度方法中的人工痕迹)，321-323
sampling and (采样和)，400-401
recursion in Whitted ray tracing (Whitted 光线跟踪中的递归)，347-350
re-entrant polygon clipper (凹多边形裁剪器)，169-170
Reeves, W.T., 274, 529-530
reflection (反射)，22-25
phenomena used (所用现象)，174
reflection mapping (反射映射)，243-250
comparative points (比较点)，248-249
cubic mapping (三次映射)，245-247
sphere mapping (球映射)，247-248
surface properties and (表面性质和)，249-250
refraction (折射)，22-25
Regan, M., 447-448
Reisenfeld, R.F., 131
rendering (绘制)
bi-cubic parametric patch (双三次参数曲面片)，125-138
image space subdivision (图像空间细分)，135-138
object space subdivision (物体空间细分)，128-135
from patch descriptions (来自曲面片描述)，125-128
patch to polygon conversion (曲面片到多边形的变换)，128
and colour space (和颜色空间)，435-436
CSG description (CSG描述)，138-140
equation, in global illumination (全局照明中的方程)，277-278
image-based (基于图像)，参见image-based rendering
implicit functions (隐函数)，141
order of in graphics pipeline (绘图流程中的次序)，187-189
polygonal mesh (多边形网格)，124-125
shading options (明暗处理选项)，182
in spatial subdivision representation (在空间细分中的表示)，53-55
voxel description (体素描述)，140-141
Z-buffer algorithm in (Z缓冲器算法)，192-193
REYES rendering system (REYES 绘制系统)，411
Reynolds, C.W., 531-532
RGB space, colour in, Figure 15.3 (Colour Plate) (在RGB空间中的颜色，图15-3 (彩色插图))，423-424
$RGB_{monitor}$ space, colour in ($RGB_{监视器}$中的颜色)，422, 423, 436-437
right-handed systems (右手系)，2
rigid bodies in dynamic simulation (动力学模拟中的刚

体)，507-510
rigid body animation（刚体动画)，476, 477-493
camera as animated object（作为动画物体的摄像机)，492-493
explicit scripting（明确的脚本)，479-483
interpolation（keyframing)（插值（关键帧))，477-479
quaternions, interpolation（四元组，插值)，488-492
space, moving in and out of（空间，进出运动)，488-489
spherical linear interpolation（球状线性插值)，488-492
rotation（旋转)
interpolation of（插值的)，483-484
quaternions in（其中的四元组)，484-488
rotation in rigid body animation（刚体动画中的旋转)
interpolation of（插值的)，483-484
quaternions in（其中的四元组)，484-488
Rubin, S.M., 357

S

sampling（采样)
and aliases（和走样)，393-397
reality, comparisons（真实性，比较)，398-400
and reconstruction（和重构)，400-401
Sancha, T.L., 189
scan line algorithm（扫描线算法)，127
projecting shadows（投影阴影)，267-268
spanning（跨距)，194-196
scene management（场景管理)，58-64
polygon mesh optimization（多边形网格优化)，59-64
Schaufler, G., 445, 446, 447
Schroeder, W.J., 60
Schweitzer, D., 125, 128
screen space, transformations from eye space（屏幕空间，来自目视空间的变换)，154
screen space flatness（屏幕空间平整度)
of patch（曲面片的)，136-137
of silhouette（轮廓的)，137-138
Sederburg, T.W., 115
Segal, M., 247, 391
Seitz, S.M., 460-462
selective refinement（选择性细化)，61
semi-transparent gel option in volume rendering（体绘制中的半透明胶选项)，377-380
compositing pixels along ray（沿着光线合成像素)，379-380
viewing direction, transforming（观察方向，变换)，379
voxel classification（体素分类)，378-379
semi-transparent gel plus surfaces in volume rendering（体绘制中的半透明胶加表面)，380-384
explicit extraction, Figure 13.10（Colour Plate）, 13.11（Colour Plate)（显式抽取，图13-10（彩色插图)，图13-11（彩色插图))，382-384
Shade, J., 446, 447, 457-458
shading algorithms（明暗处理算法)，34
shadow mapping, comparisons, Figures 18.7, 18.8, 18.9（Colour Plates)（阴影映射，比较，图18-7，18-8，18-9（彩色插图))，538-539
shadowing effects in specular reflection（镜面反射中的阴影效果)，214-216
shadows（阴影)，263-274
algorithms（算法)，267-274
anti-aliasing（反走样)，273-274
derivation from light source transformations（由光源变换产生的派生物)，271
projecting polygon/scan line（投影多边形/扫描线)，267-268
volumes（体)，268-270
Z-buffer（Z缓冲器)，271-274
on ground plane（在地平面上)，265-266
properties（性质)，265
and radiosity method（和辐射度方法)，324, 328-331
in Whitted ray tracing（在Whitted 光线跟踪中)，345-346
Shafer, R.W., 405
Shinya, M., 365
Shirley, P., 445
Shoemake, K., 492
shooting in radiosity method（辐射度方法中的发射)，316-317
Siegel, R., 308
silhouette, screen space flatness of（轮廓的屏幕空间平整度)，137-138
Sillion, F.X., 390
skinning algorithm（表面算法)，38
Sloan, K.R., 230
Smith, A.R., 424
Snow White and the seven Dwarfs（白雪公主和七个小矮人)，478
Snyder, J., 39-40, 42, 448-451
solids of revolution（旋转的实体)，39
space domain, and Fourier transform（空间域和傅里叶变换)，412

space tracing in Whitted ray tracing (Whitted 光线跟踪中的空间跟踪), 359
space-time constraints in dynamic simulation (动力学模拟中的空间－时间约束), 515-517
spanning hidden surface removal (跨跃的隐藏面消除), 193-194
spanning scan line algorithm (跨跃的扫描线算法), 194-196
spatial coherence, use of in Whitted ray tracing (Whitted 光线跟踪中采用的空间连贯性), 358-360
spatial subdivision (空间细分)
  representation (表示), 31, 51-56
    BSP trees (BSP树), 55-56
    octrees (八叉树), 51-52, 53-55
    rendering (绘制), 53-55
    voxels (体素), 51
    creating objects (创建物体), 56
  in Whitted ray tracing (在Whitted 光线跟踪中), 359
Spatialy Enumerated Auxilliary Data Structure (SEADS) (空间枚举的辅助数据结构), 362-363
spectral space, colour in (光谱空间中的颜色), 422
specular colour, modulation of (镜面颜色的调整), 225
specular components, reflection from (反射的镜面分量), 211-212
specular interaction in radiosity method (辐射度方法中的镜面交互), 281-283, 318
specular to diffuse transfer in global illumination (全局照明中的镜面反射到漫反射的传递), 281-283
specular to specular transfer in global illumination (全局照明中的镜面反射到镜面反射的传递), 281-283
Speer, L.R., 366
Speilberg, Steven, 475
sphere (球形), 18-19
sphere mapping (球映射), 247-248
spherical linear interpolation (slerp) (球状线性插值(slerp)), 489-492
spine curve objects (轮廓曲线物体), 41
splatting (voxel projection) (溅泼 (体素投影)), 389
sprites in IBR (IBR中的小精灵), 444, 445
Sproull, R., 152, 189, 194
Star Trek II: The Wrath of Khan (星际旅行II: Khan 的愤怒), 530
stochastic ray tracing in global illumination (全局照明中的随机光线跟踪), 294
stochastic sampling in anti-aliasing (反走样中的随机采样), 402, 407, 408-409
straight spine objects (直线轮廓的物体), 40
structure-deforming transformations (结构分解变换), 9-11
Sturzlinger, W., 445, 446
substructuring in radiosity method (辐射度方法中的子结构), 325
supersampling (post-filtering) in anti-aliasing (反走样中的超采样 (后过滤)), 404-406
surface detection and shading in volume rendering (体绘制中的表面检测和明暗处理), 381
surface fitting of patch objects (曲面片物体的表面拟合), 115-121
  interpolating surfaces (插值表面), 117-121
  using B-splines (运用B样条), 115-117
surface micro-geometry and specular reflection (表面微观几何和镜面反射), 214
Sutherland, I., 100, 169, 189, 194
Swanson, R.W., 184

T

tapering, Figure 1.5 (Colour Plate) (变细, 图1-5 (彩色插图)), 9, 11
tears, in non-uniform subdivision of object space (物体空间非均匀细分中的水滴), 131, 134
Terzopoulos, D., 532
texture mapping (纹理映射)
  anti-aliasing, Figure 8.26 (Colour Plate) (反走样, 图8-26 (彩色插图)), 256-260
  comparisons, Figures 18.7, 18.8, 18.9 (Colour Plates) (比较, 图18-7, 18-8, 18-9 (彩色插图)), 538-539
  interactive techniques (交互技术), 260-262
  pre-calculation technique (预计算技术), 257
  view-dependent (依赖于观察的), 470
Thayer, L.J., 184
three dimensional texture (三维纹理)
  domain techniques (域技术), 251-256
    and animation, Figure 8.22 (Colour Plate) (和动画, 图8-22 (彩色插图)), 254-256
    3D noise (3D噪声), 251-252
    light maps (光线图), 256
    turbulence, Figure 8.21 (Colour Plate) (扰动, 图8-21 (彩色插图)), 252-254
  and volume rendering (和体绘制), 391
three-dimensional light maps (三维光线图), 256
three-dimensional screen space in graphics pipeline (绘图流程中的三维屏幕空间), 149-152
three-dimensional space and colour (三维空间和颜色), 420-427
  HSV single hexcone model, Figure 15.5 (Colour Plate) (HSV 单六角锥体模型, 图15-5 (彩色插图)),

424-427
RGB space, Figure 15.3 (Colour Plate) (RGB空间, 图15-3 (彩色插图)), 423-424
YIQ space (YIQ空间), 427
three-dimensional structures (三维结构)
manipulating (处理), 1-8
representation and modelling (表示和建模), 27-65
three-dimensional warping in IBR, Figure 16.8 (Colour Plate) (IBR中的三维变形, 图16-8 (彩色插图)), 452-456
Torrance, K.E., 212, 213-219, 222
transformations (变换)
affine (仿射), 2-7
changing coordinate systems (改变坐标系), 8
structure-deforming (结构变形), 9-11
transparency (透明), 225
trivariate Bézier hyperpatch (三变量Bézier超曲面片), 114
Troutman, R., 321
Tu, S., 532
turbulence in 3D texture domain techniques, Figure 8.21 (Colour Plate) (三维纹理域技术中的扰动, 图8-21 (彩色插图)), 252-254
twisting (扭曲), 9, 11
global axial (全局轴), 9
two-dimensional texture maps (二维纹理图), 228-234
bi-cubic parametric patch objects, Figure 8.8 (Colour Plate) (双三次参数曲面片物体, 图8-8 (彩色插图)), 234
inverse mapping, Figure 8.7 (Colour Plate) (反向映射, 图8-7 (彩色插图)), 229-234
two-pass ray tracing in global illumination, Figure 10.20 (Colour Plate) (全局照明中的二路光线跟踪, 图10-20 (彩色插图)), 297-300

U

umbra (阴影), 264
in radiosity (在辐射度中), 340
uniform B-splines (均匀B样条), 80-84
uniform subdivision of object space (物体空间的均匀细分), 129-130
Utah Teapot (Bézier patch object) (Utah 茶壶 (Bézier 曲面片物体)), 100-101
testing for flatness, Figure 4.9 (Colour Plate) (平整度测试, 图4-9 (彩色插图)), 133

V

Van Dam, A., 194
vectors (向量), 11-17
addition (加), 12
length (长度), 12
normal (法向)
associated vectors (相关的向量), 15-17
and cross products (和叉积), 12-14
and dot products (和点积), 14-15
velocity matching in behavioural animation (行为动画中的速度匹配), 532
vertex split in polygon mesh optimization (多边形网格优化中的顶点分离), 62
vertex-edge events in radiosity method (辐射度方法中的顶点-边事件), 338-341
vertices (顶点)
attributes (属性), 36-37
in polygonal mesh representation (在多边形网格表示中), 34, 35
videodisk (视频光盘), 467
view coordinate systems in graphics pipeline (绘图流程中的观察坐标系), 143-146
view dependence /independence in global illumination (全局照明中的依赖于/独立于观察的), 300-301
view interpolation in global illumination (全局照明中的观察插值), 300-301
view interpolation in IBR (IBR中的观察插值), 458-463
view mapping parameters in PHIGS (PHIGS 中的观察映射参数), 159-162
view morphing in IBR (IBR中的观察变形), 460-463
view orientation parameters in PHIGS (PHIGS 中的观察方向参数), 159, 160
view plane distance in PHIGS (PHIGS 中的观察平面距离), 162
view plane in graphics pipeline (绘图流程中的观察平面), 145, 150
view plane normal in PHIGS(PHIGS中的观察平面法向), 158, 162
view plane window in PHIGS (PHIGS 中的观察平面窗口), 157, 163
view point in graphics pipeline (绘图流程中的观察点), 145
view reference coordinate in PHIGS (PHIGS中的观察参考坐标), 157, 158
view reference point in PHIGS (PHIGS 中的观察参考点), 157, 162
view space (观察空间)
operations in (运算), 147-156
culling (挑选), 147-148

3D screen space (3D屏幕空间), 149-152
view volume (视见体), 147-149
view volume and depth (视见体和深度), 152-156
view up vector in PHIGS (PHIGS中的观察向上向量), 158
view volume (视见体)
in graphics pipeline (在绘图流程中), 145, 147-149
clipping polygons (裁剪多边形), 168-171
and depth (和深度), 152-156
in PHIGS (在PHIGS 中), 161, 163
view-dependent texture mapping (依赖于观察的纹理映射), 470
viewing direction in volume rendering (体绘制中的观察方向), 379
viewing geometry in specular reflection (镜面反射中的观察几何学), 216
Visible Human Project (可见的人体投影), 371-373
Visualization in Scientific Computing (ViSC) (科学计算中的可视化), 370, 419
visualization of volume data, Figure 13.3 (Colour Plate) (体数据的可视化, 图13-3 (彩色插图)), 373-377
data and reality, relationship (数据和真实性关系), 376-377
data properties (数据属性), 375-376
volume filling in particle animation (粒子动画中的体填充), 530
volume rendering (体绘制), 370-391
3D texture (3D纹理), 391
perspective projection (透视投影), 390
semi-transparent gel option (半透明胶选项), 377-380
compositing pixels along ray (沿着光线合成像素), 379-380
viewing direction, transforming (观察方向, 变换), 379
voxel classification (体素分类), 378-379
semi-transparent gel plus surfaces (半透明胶加表面), 380-384
explicit extraction, Figures 13.10, 13.11 (Colour Plates) (显式抽取, 图13-10, 13-11 (彩色插图)), 382-384
structural considerations (对结构的考虑), 384-390
ray casting (光线投射), 385-388
voxel projection (体素投影), 388-390
structures , taxonomy of (结构的分类), 385
and visualization of volume data, Figure 13.3 (Colour Plate) (和体数据可视化, 图13-3 (彩色插图)), 373-377
data and reality, relationship (数据和真实性关系), 376-377
data properties (数据属性), 375-376
Voronoi diagrams (Voronoi图), 526
voxels (体素), 51
classification (分类), 378-379
creating objects (创建物体), 56
rendering description (绘制描述), 140-141
in volume rendering (在体绘制中)
filtering (过滤), 389
projection (投影), 388-390
shape, Figure 13.1 (Colour Plate) (形状, 图13-1 (彩色插图)), 371
size (尺寸), 377

W

Wallace, J.R., 320, 325
Walter, B., 304-305
Ward, G.J., 241, 299, 303
Warnock, J., 189
warping in IBR (IBR中的变形), 444
3D, 452-456
Watt, A., 117, 119, 131, 501
Watt, M., 117, 119, 131, 501
Weghorst, H., 355-356, 357, 521
Weiler, K., 271, 364
Westover, L., 389-390
Whitted, J., 125, 128, 342, 357
Whitted ray tracing (Whitted 光线跟踪), 342-369
basic algorithm (基本算法), 343-347
components (组分), 344
hidden surface removal (隐藏面消除), 346-347
shadows (阴影), 345-346
tracing rays (跟踪光线), 343-344
comparisons, Figures 18.10, 18.11, 18.12, 18.13 (Colour Plates) (比较, 图18-10, 18-11, 18-12, 18-13 (彩色插图)), 539-541
efficiency measures in (高效测量), 354-364
adaptive depth control (自适应深度控制), 354-355
bounding objects (限定物体), 355-357
bounding volume hierarchies (限定体层次), 357-358
first hit speed up (首次击中加速), 355
octrees, use of (八叉树的使用), 360-363
ray space subdivision (光线空间细分), 363-364
secondary data structures (二次数据结构), 357-363
spatial coherence, use of (空间连贯性的利用), 358-360
features of (其特性), 342
in global illumination (在全局照明中), 284-286, 293

optics of rainbow（彩虹的光学），367-369
polygon objects（多边形物体），352-354
ray coherence, use of（光线连贯性的利用），364-367
recursion in（递归），347-350
seven ray study, Figure 12.4（Colour Plate）（七种光线研究，图12-4（彩色插图）），350-352
Wilhelms, J., 522
Williams, L., 257, 271, 458-459
wireframe visualization（烟火可视化），30, 34
shading options（明暗处理选项），182
Witkin, A., 504, 516
Wolberg, G., 387
world coordinate systems in graphics pipeline（绘图流程中的世界坐标系），143

X

X-ray computer tomography, volume rendering in（体绘制中X射线计算机照相），371-373, 374
material classification（材料分类），378

Y

Yellot, I., 409
YIQ space, colour in（颜色的YIQ空间），423, 427

Z

Z-buffer algorithm（Z缓冲器算法），189-190
complex scenes（复杂场景），196-198
and compositing（和合成），191-192
and CSG representation（和CSG 表示），190-191
and rendering（和绘制），192-193
scan line（扫描线），193
on shadows（阴影上的），271-274
anti-aliasing（反走样），273-274
Zhukov, S., 241-242