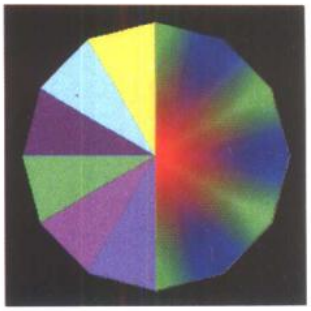
国防工业出版社
www.ndip.com.cn

李颖 薛海斌 朱伯立 朱仲立编著

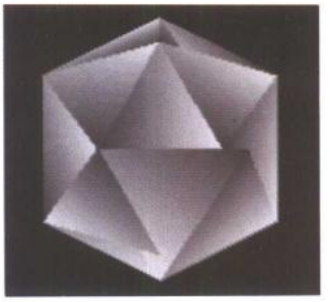
OpenGL 技术应用实例精粹

工程师工具软件应用系列

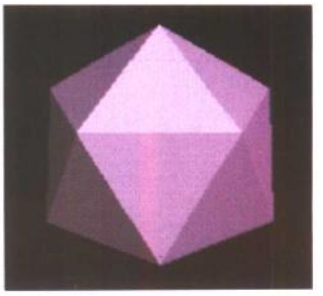


彩图 1

使用不同的颜色模型绘制的彩色调色板，右半部调色板使用光滑明暗处理模型，左半部调色板使用平面明暗模型，请参看第三章。



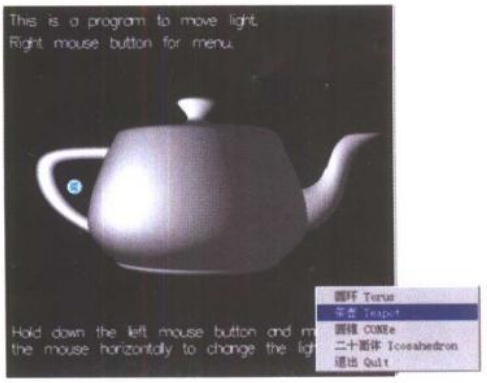
(a)



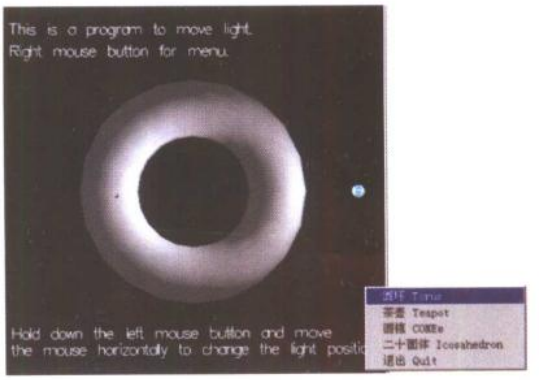
(b)

彩图 2

上图是使用多边形图原创建的二十面体:(a)图是没有经过光照的二十面体;(b)图是定义了光照和材质的二十面体。从图中可以看到，未使用光照的二十面体并没有三维立体的感觉，这里要注意的是二十面体中各平面法向量的计算方法，请参看第六章和第八章。



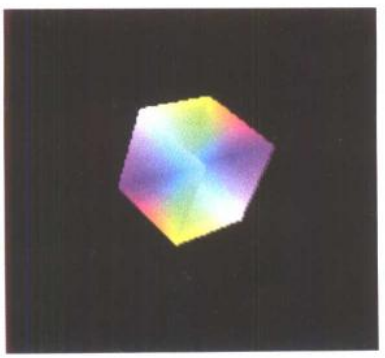
(a)

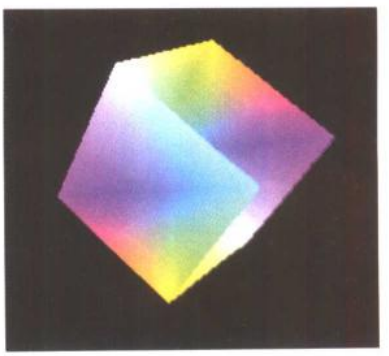


(b)

彩图 3

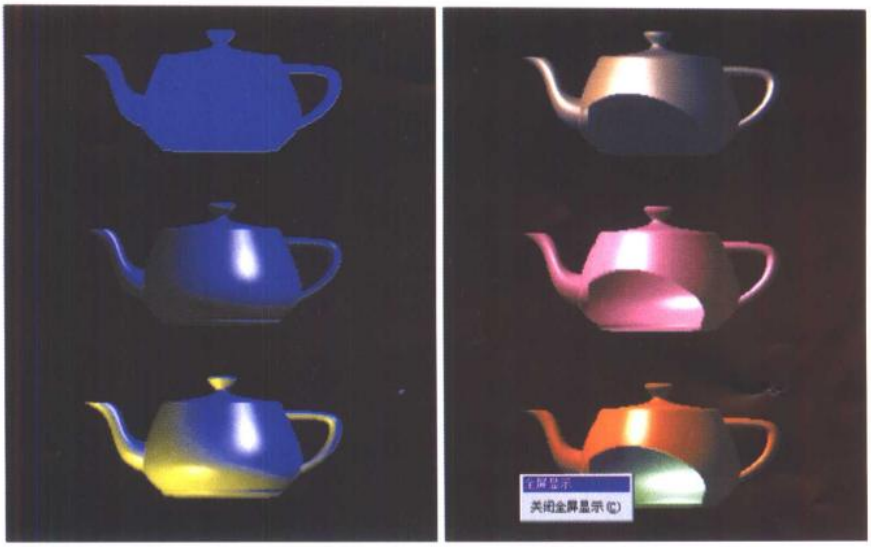
上图是围绕静止物体移动光源，光源用网格球体代替，当用鼠标移动球体时，光源的位置发生改变，可以选择菜单项选择不同的物体以观察光照效果，见第六章。





彩图 4

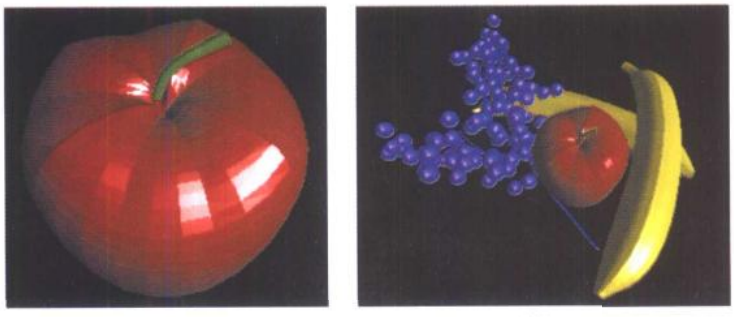
左图是使用双缓存绘制的由远及近旋转的彩色立方体，关于如何使用双缓存，请参看第八章。



(a) (b)

彩图 5

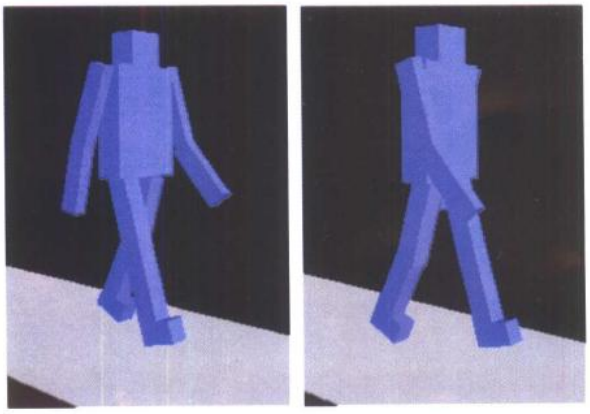
不同光照及材质属性下的作用效果。(*a*)第一个茶壶没有光照，因此看起来就如同二维平面一样，并没有三维立体的感觉；第二个茶壶定义了一个蓝色的方向光源；第三个茶壶定义了两个光源，除了一个蓝色的方向光源外，还有一个黄色的聚光源。(*b*)第一个茶壶只对外侧定义了材质属性，因此只使用了单面光照射；第二个茶壶的内侧和外侧定义了相同的材质属性，第三个茶壶的内侧和外侧定义了不同的材质属性，这两个茶壶均使用了双面光照射。参看第六章。



(a) (b)

彩图 6

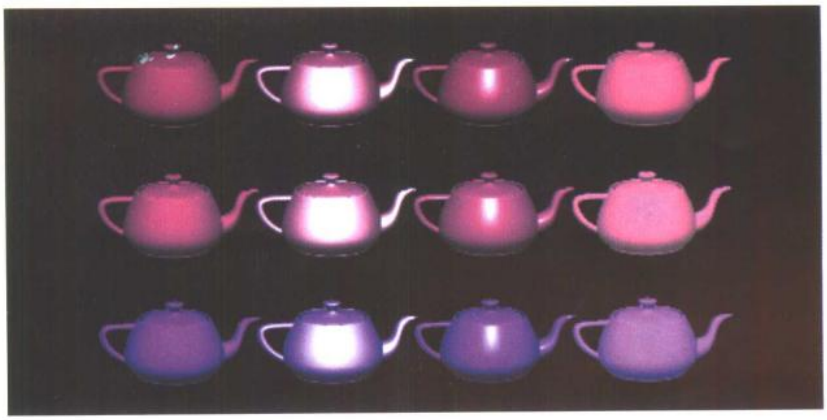
左图是使用 3DS 模型绘制的场景，关于如何将3DS模型转换为OpenGL模型，可以参看第八章。



(a) (b)

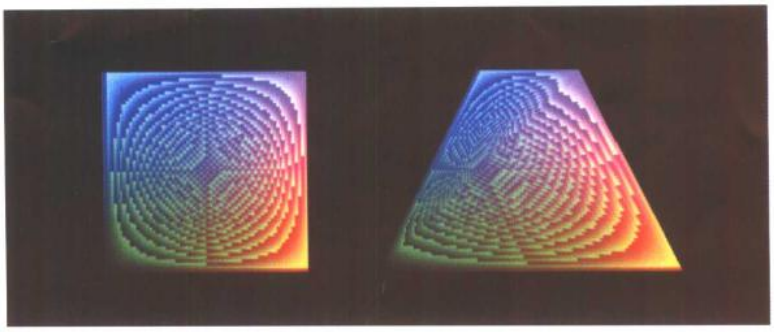
彩图 7

左图绘制了一个行走的机器人，机器人身体的各部位用长方体代替，使用glPushMatrix()和glPopMatrix()命令保存和恢复各状态变量的值，因此身体的各部位可以在原点进行绘制，然后再移动到希望的位置上，机器人的行走可以使用glRotatef()命令实现。参看第二章和第八章。



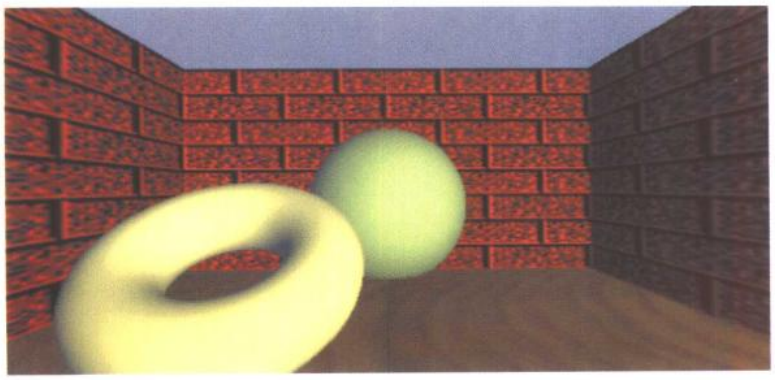
彩图 8

左图绘制了十二个不同材质属性的茶壶。为了提高程序的性能，在改变茶壶的环境反射材质属性时使用了glColorMaterial()函数。第一列使用了红色的漫反射、没有镜面反射成分和亮度；第二列增加了白色的镜面反射，并使用了低亮度；第三列使用了高亮度，因此有更聚集的高光；第四列使用了红色的漫反射，没有镜面反射和亮度，但增加了散射光。参看第六章。



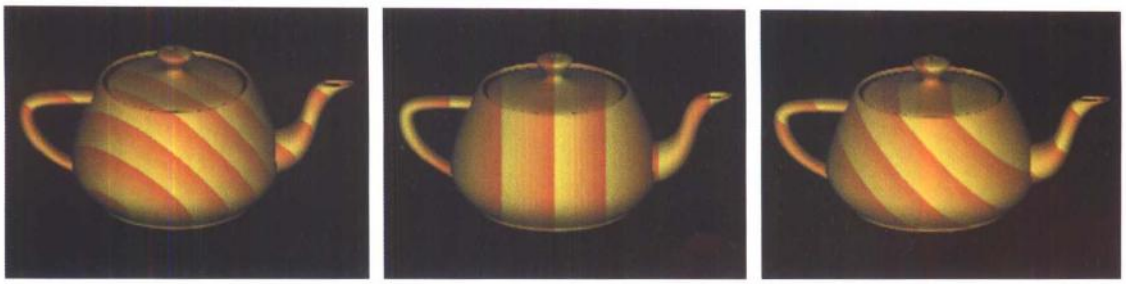
彩图 9

左图是在正方形和梯形上使用的简单的二维图形纹理，关于纹理的定义和使用参看第九章。



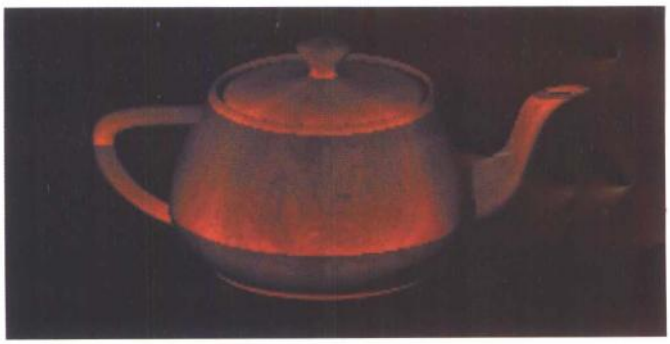
彩图 10

上图是使用纹理映射绘制的场景，场景中使用了位图文件作为纹理，墙壁使用了砖墙纹理，而地板使用了木质的地板纹理。关于如何定义纹理坐标及使用纹理映射，参看第九章。



彩图11

上图是利用自动纹理坐标生成技术在茶壶表面生成的等间隔的轮廓线，这里使用的是一维纹理，(*a*)图是顶点到平面x+y+z=0的距离；(*b*)图是顶点到平面x=0的距离；(*c*)图是顶点到平面x+z=0的距离。关于如何创建轮廓图，参看第九章。

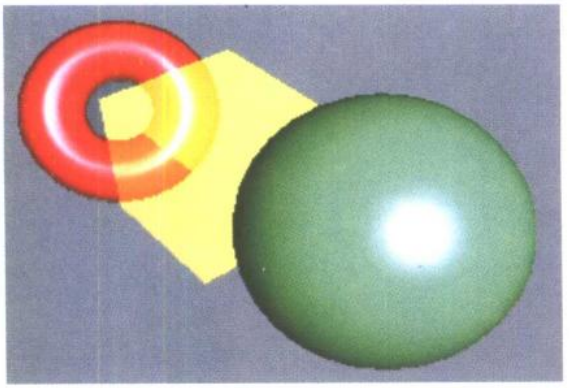


彩图12

左图是使用融合(GL_MODULATE)方式生成的贴木质纹理的茶壶，场景中使用了光照，关于如何使用纹理的映射方式，参看第九章。

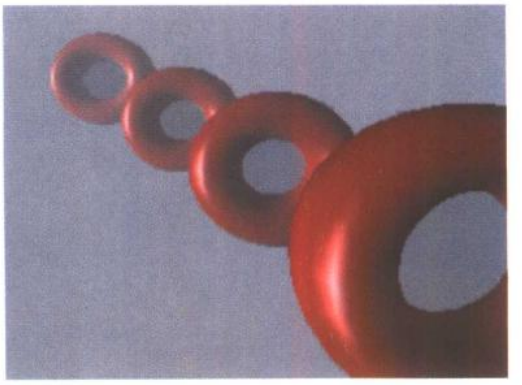
彩图13

上图是未开启反走样和开启反走样后绘制的蜘蛛网格，很明显，使用反走样后的直线看起来要平滑得多。关于如何使用反走样，参看第十章。



彩图 14

上图是使用融合绘制的有半透明物体的场景，其中立方体有50%的透明度，而球体和圆环则是不透明的。关于如何使用融合操作绘制半透明的物体，参看第十章。

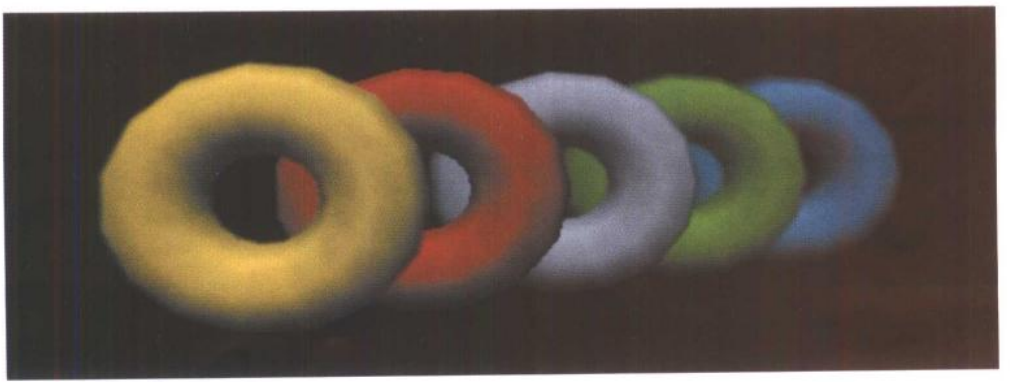


彩图 15

上图绘制的是有雾化效果的场景，场景中使用了透视投影，随着视点到物体距离的增加，雾化的浓度越来越强。关于如何使用雾化绘制大气场景，参看第十章。

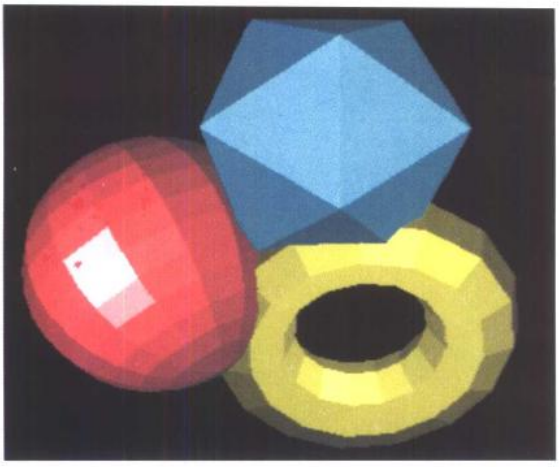
彩图 16

左图是使用二维求值器绘制的Bézier曲面，并打开二维纹理求值器为Bézier曲面应用了位图纹理，关于如何绘制Bézier曲面，参看第十一章。

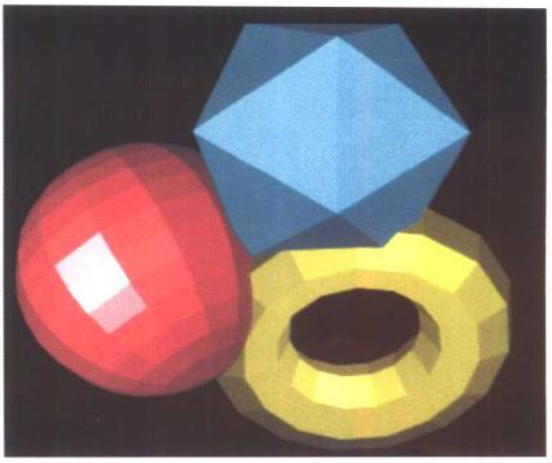


彩图 17

上图是用累加缓存绘制的有景深效果的场景，相机聚焦在红色的圆环上，远离焦点的物体随着距离的增加越来越模糊。关于如何使用累加缓存绘制景深效果，参看第十二章。



(a)



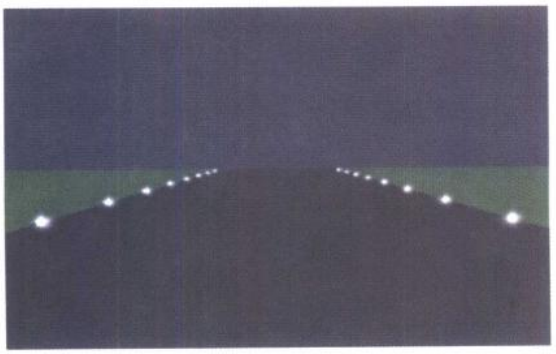
(b)

(c)

(d)

彩图 18

场景中绘制了几个平面明暗模式的物体。(*a*)图中的场景没有反走样;(*b*)图中的场景使用累加缓存执行反走样:场景进行多次绘制,每一次绘制用小于 1 个象素的跳动值跳动场景,然后再将整个图像进行累加,并求取均值。(*c*)图和(*d*)图分别是(*a*)图和(*b*)图的放大,(*c*)图中未反走样的场景有明显的锯齿边,而(*d*)图中经过反走样后,物体的边界比较平滑。关于如何使用累加缓存执行场景的反走样,参看第十二章。



彩图 19

左图是使用纹理映射和融合技术模拟的有照明感的灯光。关于如何实现灯光的模拟,参看第十四章。

彩图 20

左图是使用纹理映射和融合技术实现的火焰和烟雾的模拟,在火焰的上方有的浓烟。关于如何实现火焰和烟雾效果,参看第十四章。

前　言

OpenGL 是图形硬件的软件接口，由于它稳定性好、可移植性强，因此目前已经成为跨平台最广泛的三维图形引擎，而且也是目前应用最广泛的三维图形标准。利用 OpenGL，程序员不仅可以绘制简单的二维形体，而且还可以编制有各种不同效果的交互式三维动画场景，在建筑学、医学、CAD/CAM/CAE，以及广告业和娱乐业中，已经开发出了很多实用的OpenGL应用程序。由于OpenGL可以在Unix、Windows NT、Windows 95/98 以及 Mac 等多种不同的平台上执行，因此如果需要跨平台开发应用程序，可以考虑使用 OpenGL。

本书介绍的是 OpenGL 的最新版本：OpenGL1.2 版，该版本包括了 250 多个不同的图形函数，每个函数又有几种变型，因此这个版本包括了大约 300 个函数。本书按照 OpenGL 的特点共分了十五个章节，全书详细论述了 OpenGL 的运行机制及各种不同效果的实现方式。第一章和第二章是 OpenGL 的基础知识，介绍了 OpenGL 的主要特性、OpenGL 的工作流程、库函数以及 OpenGL 的编程结构。第三章至第七章介绍了 OpenGL 中 2D 形体的绘制方式，以及视景转换和显示列表等知识，利用视景转换，程序员可以以任意角度观看所绘制的场景，并可以使用不同的投影方式模拟不同的场景效果；而利用显示列表，程序员则可以优化程序的执行性能，提高程序的运行速度。除此之外，还详细描述了如何在 OpenGL 场景中为所绘制的形体使用颜色、光照和材质属性，这是很重要的内容，因为只有在场景中使用了光照，所绘制的形体看起来才会有三维立体的感觉。第八章介绍了 OpenGL 中的三维建模，给出了用几何图原构造三维模型以及利用 3DS 模型建模的方式，同时还论述了 OpenGL 是如何使用双缓存来实现三维动画演示的。第九章介绍了如何在 OpenGL 中使用纹理映射，利用纹理映射，程序员可以创建有真实感的物体模型。第十章介绍了 OpenGL 中特殊效果的实现方式，包括融合、雾化和反走样，使用融合可以生成半透明的物体，使用雾化可以模拟真实世界中的大气效果，如雾、霜、烟等，使用反走样可以改善物体的细节效果，也就是减少由于显示方式造成的锯齿边。第十一章介绍了如何在 OpenGL 中绘制曲线和曲面，这是很重要的，因为真实世界中的物体并不都是由直线和平面组成的，掌握曲线和曲面的绘制方法可以创建更加丰富多彩的模型。第十二章介绍了 OpenGL 中的帧缓存，使用累加缓存可以实现场景反走样，以及模拟相机景深程序。第十三章介绍了 OpenGL 中的交互技术，使用交互技术可以创建交互式的三维应用程序。第十四章给出了利用上面章节中介绍的 OpenGL 技术实现不同效果的综合实例，包括如何模拟火焰和烟雾，如何模拟有动感的海面和云彩，以及如何模拟闪烁的灯光等，这些特殊效果是非常实用的，读者可以仔细阅读程序中这些效果的实现方式，并将它们应用到自己的应用程序中。第十五章介绍了利用 Visual C++中的 MFC 编制 OpenGL 程序的方式，并给出了两个 MFC 编程实例。此外，附录 A 给出了 GLUT

库的部分函数的命令原型。

本书中的所有章节在介绍 OpenGL 的各种效果及实现方式时均给出了程序实例，这些程序收录在本书所带的光盘中，除了第十五章中的两个 MFC 程序需要读者自己编译外，其它各章节的程序均不必再进行编译，所有应用程序均在 Windows98 的 Microsoft Visual C++6.0 集成环境下调试成功。

本书力求通俗易懂，对于读者需要注意及重视的地方，进行了多次强调，目的就是希望读者通过阅读本书后能够掌握 OpenGL 的编程方法和技巧，因此本书给出的实例程序都是非常典型的，希望读者在阅读本书后能够进入 OpenGL 的多彩的三维世界。

本书的编写工作由李颖、薛海斌、朱伯立、朱仲立完成，李颖、薛海斌调试了本书的所有例程，此外，李学峰、李威、郝志勤、关士成、徐勇、胡勇、赵磊、谢方军、于英杰、于小权、谢欣海、齐占元、张敏等同志参与了本书的编写工作，刘春慧、陈雨、李威等对本书的编辑排版做了大量工作，在此一并表示感谢。本书对于计算机辅助设计人员，以及需要编制交互式三维真实场景的软件人员是一本非常有用的参考书。

由于作者的水平有限，加之时间仓促，书中可能会出现错误的地方，恳切希望读者批评指正。

作　者

2000 年 9 月

目　录

第一章 OpenGL 概述

本章的主要目的是介绍 OpenGL 的一些基本知识，包括 OpenGL 的由来，什么是 OpenGL，OpenGL 的特点，以及 OpenGL 可以为用户提供哪些功能，以便使读者对 OpenGL 有个初步的了解。

1.1 OpenGL 的由来

在今天，3D 已经深入到了世界的各个角落，从电影艺术到动画世界，从工程应用到虚拟现实，甚至在席卷全球的 Internet 浪潮中，我们都能看到它的身影。事实上，这些精美的动画带给我们震撼性的视觉感受的同时，人们不禁要问，在背后支撑它们的到底是什么呢？答案很简单，那就是风靡全球的三维图形技术，其中，作为三维图形技术代表的 OpenGL 已经成为最广泛采用的图形标准，它便是这些图形技术中的佼佼者。

OpenGL 最初是 SGI 公司为其图形工作站开发的可以独立于窗口系统、操作系统和硬件系统环境的图形开发环境，用户可以完全不去理会具体的硬件系统、窗口和操作系统的结构和指令形式，只要按照 OpenGL 规定的格式书写应用程序，就可以在任何支持该语言的硬件平台上执行。OpenGL 的前身是 SGI 的 IRIS GL。IRIS 是 SGI 公司首要的图形工作站的名称，而 IRIS GL 过去是这些工作站上编制实时动态场景的“图形语言”。IRIS GL 仅为 SGI 的图形工作站专有，不是一个开放标准。那时，SGI 公司虽然制造了世界上速度最快的 3D 计算机，但是很少有人能够承受它昂贵的价格，更不用提使用 GL 语言编程所需要的专门技巧了。后来，SGI 公司认识到了这一点。为了拓宽 3D 市场并扩大自己在其中的份额，SGI 开始对 IRIS GL 改造以使其适应多种不同的平台，从而产生了性能优越，具有跨平台和高度可重用性的 OpenGL。鉴于此，已经有超过 30 个大公司和研究机构加盟或表示接受 OpenGL 作为标准图形软件接口，许多软件和硬件的制造商与 SGI 公司一起成立了 OpenGL 体系结构审查委员会(简称 OpenGL ARB)，从而使 OpenGL 成为一个开放的标准。目前 ARB 的成员包括：SGI、Microsoft、Intel、IBM 和 DEC 等公司。

1.2 OpenGL 的主要特性

作为开发可移植的交互式 2D 和 3D 图形应用程序的首选环境，OpenGL 从 1992 年引入至今已经成为业界最广泛使用和受到最多支持的 2D 和 3D 图形应用程序接口(API)，现已有成千上万的应用程序运行在各种各样的平台上，包括飞行训练、分子结构研究、影视广告艺术、CAD/CAM/CAE、游戏娱乐、医学成像、计算机动画和虚拟现实等。在

这些领域中，程序员利用 OpenGL 制作出有吸引力的图形图像，这些当然都是与 OpenGL 优良的特性分不开的，这些特性包括：

- **工业标准** OpenGL 的规范由独立的国际协会——OpenGL 体系结构审查委员会进行指导，因此，OpenGL 是唯一开放的、中立于制造商的，能够运行在多个平台上的图形标准。
- **稳定性** OpenGL 在各种平台上的实现时间已经超过了八年，除了 OpenGL 规范得到良好的控制外，建议的更新也及时地通知给开发商，而且 OpenGL 向后兼容，确保已有的应用程序在版本更新后不过时。
- **可靠性** 不管在什么样的操作系统或窗口系统上运行，任何 OpenGL 的应用程序在任何兼容 OpenGL API 的硬件上都会显示出一致的视觉效果。
- **可扩展性** OpenGL 的设计考虑得非常周到，兼顾到了以后的升级。应用开发者和硬件厂商(比如图形加速卡)能够利用 OpenGL API 中的扩充机制支持硬件的各种新特征。
- **可扩充性** 基于 OpenGL API 的程序能够运行在从消费电器到 PC 机、工作站，甚至超级计算机上，因此，OpenGL 程序可以扩充到开发者选择的任何机器类型上运行。
- **易用性** OpenGL 命令(函数)具有很强的逻辑性，因此开发者可以凭直觉开发三维场景，并且用 OpenGL 编制图形软件与使用其它图形库相比，代码量要小得多。另外，OpenGL 驱动程序封装了底层硬件信息，从而使应用程序开发人员无须考虑指定硬件的特点。

目前，OpenGL 的主要版本有 1.0、1.1、1.2、1.2.1，其中 1.1 版本应用最为广泛。支持 OpenGL 的平台包括 Windows 95/98、 Windows NT、Windows 2000、OS/2、Unix 及 Solaris 等操作系统，在 Microsoft 公司的 Windows 95 系列操作系统中，OpenGL 已经集成到了 Windows 95 OSR2 或更高的版本中。

1.3 进入 OpenGL

1.3.1 什么是 OpenGL

OpenGL 是图形硬件的软件接口，它由大约 250 个命令组成，用户通过这些命令指定创建交互式 2D 或 3D 程序所需的几何对象和操作。从程序员的观点来看，OpenGL 是一种图形函数库，为了使用方便，实际上每条 OpenGL 的命令函数都有几种不同的变种，因此，OpenGL 库中包含了 300 多个函数。

OpenGL 是网络透明的，它允许任务在本机调动并通过网络发送命令到远程工作站执行。这种情况下，运行程序并发送 OpenGL 绘图命令的计算机称为客户机，接受命令并执行绘图命令的计算机称之为服务器。很多情况下，特别是在 PC 机应用中，OpenGL 绘图命令不通过网络传输，仅由单台计算机完成所有运行程序和显示图形的工作，此时，该计算机既充当了客户机，又充当了服务器的角色。

OpenGL 又是能运行在多种平台上的接口，包括 UNIX、Windows 95/98、Windows

NT、Linux 等在内的操作系统都对其提供了很好的支持。但是，为了实现与平台无关，OpenGL 不提供任何窗口管理、输入管理和事件响应机制，因此，这些功能必须使用所在平台的用户接口实现。与此类似，OpenGL 不提供用于描述诸如汽车、人体、飞行器、分子之类的复杂三维物体模型的高层命令或函数，用户必须从点、线、面等几何原型中构建自己喜爱的模型。当然，程序员可以通过在 OpenGL 基本库之上建立高层库来提供这些功能，事实上，OpenInventor 做的就是这件事，详情参见“OpenGL 相关函数库”。

1.3.2 OpenGL 能够做些什么

OpenGL 能够生成逼真的三维图像，从绘制简单的 2D 物体到复杂的 3D 场景，OpenGL 都能够高效地完成绘制，下面具体地对 OpenGL 的功能进行描述。

创建二维和三维物体

任何复杂的形体都可以通过对点、线和面的组合构造出来。OpenGL 提供了描述点、线和多边形等几何形体的函数，通过它们可以绘制出任何自己所需要的二维和三维物体。OpenGL 中还有设置颜色的函数，用以对物体的表面进行着色。

布置场景并以适当的角度观看场景

创建后的三维物体需要布置在场景中的不同位置，在 OpenGL 中，可以使用平移、旋转和缩放等变换来实现。正如照片一样，每一个画面都是以一定的角度来拍摄的，OpenGL 在完成场景的绘制后，也必须调用视点变换函数来确定观看场景的位置和角度。此外，OpenGL 还提供了投影变换，实现从三维场景到二维图像的变换。OpenGL 中的投影变换分为两种，一种是正视投影，我们通常所说的三视图(工程制图中的正视图、俯视图、侧视图)均通过正视投影来实现；一种是透视投影，透视投影得到的图像符合人们在现实生活中观察周围景象的情形，也就是，距离观察者较近的物体看起来较大，而较远的物体看起来则较小。

在场景中引入光线

OpenGL 可以在场景中引入光线，这样做才能使绘制的图形有真实感和立体感。物体光照的实现主要与三个要素有关：光源、被照射物体的材质特性及该物体表面每点的法向量。OpenGL 可以提供四种光，即漫射光、环境光、镜面光和漫反射光，同时还可以指定光的颜色、位置以及强弱等参数。物体的材质特性指明了物体反射光线的方式，它直接影响物体看上去像什么。物体表面每点的法向量用来确定光线在该点的反射方向。

纹理映射

通过点、线、面等几何图原创建的物体，往往由于表面过于光滑和单调，缺少表面细节，看起来反而不真实，解决此问题的方法就是使用纹理映射。OpenGL 可以让开发人员应用纹理映射把真实图像贴到物体的表面，即把从真实世界中拍摄到的某种物体的表面细节，采用贴图的方式贴到三维场景中同种物体的表面，从而使渲染后得到的图像中的物体与真实世界中的物体惟妙惟肖。纹理映射使用的范围非常广泛，桌面、墙壁、天花板、壁画、甚至树木、天空、路面都可以使用纹理映射来表示物体的细节。

实现特殊效果

OpenGL 提供了一系列实现特殊效果的函数，通过它们可以实现半透明效果、烟雾，