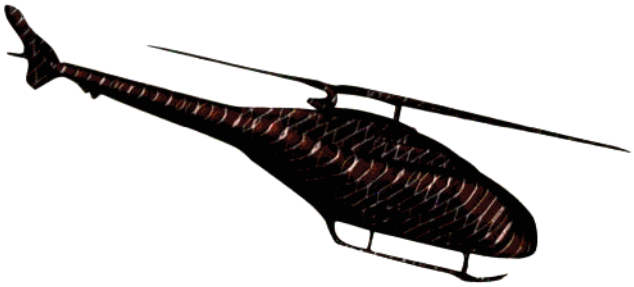
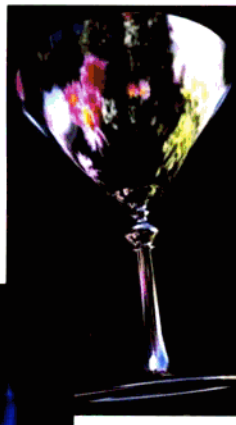
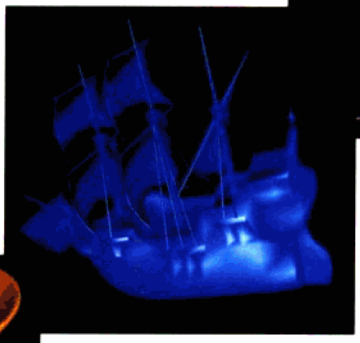
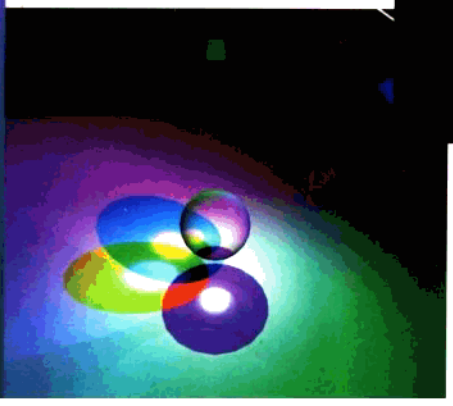


OpenGL编程 入门与提高

贾志刚／编著







中国环境科学出版社

前　言

OpenGL 是由 SGI 公司为其工作站开发的一个工业标准计算机图形软件接口，并由 Microsoft 集成到 Windows NT 和 Windows 95 中，适用于多种硬件平台及操作系统。用户可以方便地利用这个图形库创建高质量的三维彩色图像及三维动画。OpenGL 早期用于科研上的建模、模拟等方面，随着个人计算机性能的迅速提高，已经广泛地用于 CAD、可视化、计算机游戏等方面的三维图形制作、动画设计。OpenGL 作为一个热门的三维图形软件已经得到了程序设计人员的认可。

作为国内第一本详细介绍 OpenGL 资料，《精通 OpenGL》（贾志刚编著，电子工业出版社 1998 年 8 月出版）讲述了 OpenGL 的程序设计及函数的使用方法，但是，该书没有提供详细的计算机图形学的背景知识，也没有详细地介绍在 Windows 环境下的程序设计，给初学 OpenGL 的读者带来了学习、认识上的不方便。本书就是为解决上述问题编写的。

本书注重对计算机图形学知识及 OpenGL 编程技巧的介绍，结合大量图例对一些复杂的计算机图形学概念做深入浅出的解释，使初学者能够全面、深入地理解本书的内容，并在具体的编程过程中达到一定的水平。另外，本书还详细介绍如何在 Windows 环境下使用 Visual C++ 进行 OpenGL 程序设计，同时提供使用 Visual Basic, Delphi 语言进行 OpenGL 编程的具体指导。本书是一本比较理想的 OpenGL 入门书，对于要把 OpenGL 应用程序从工作站移植到 Microsoft Windows 环境的程序员也有帮助。为了便于读者使用本书的程序，本书提供包含所有源程序及相应的可执行文件软盘。这些程序均经过严格的检查、调试。

本书的内容可以分为如下部分：

第一部分作为学习、使用 OpenGL 的预备知识。介绍计算机图形学及 OpenGL 的基本概念。OpenGL 的工作原理、三维计算机图形学基础、三维坐标系统及在计算机屏幕上的实现过程。

第二部分是学习 OpenGL API 的基础。结合一些简单的程序，介绍借助 OpenGL 的 AUX 库在程序中调用 OpenGL 函数。结合具体程序，详细介绍、分析在 Windows 环境下调用 OpenGL 的过程，以及如何查找错误、调试 OpenGL 程序。

第三部分为 OpenGL 的核心内容。介绍使用 OpenGL 绘制具有色彩及光照的三维物体的基本内容，这些内容包括：OpenGL 几何要素的使用、设置物体的颜色、OpenGL 的矩阵变换、设置光源及材质的属性、绘制有光照的场景。作为使用 OpenGL 更深入的内容，介绍绘制有特殊视觉效果的图像，这些特殊效果包括：获得半透明效果的混合操作、绘制反走样图形、在场景中使用雾效果、进行纹理映射，在物体上绘制纹理、对物体的抖动操作、利用累加缓冲区产生的运动模糊、得到景深效果等。绘制 NURBS 曲线、曲面；为了设计交互式的 OpenGL 程序，将介绍 OpenGL 的选择、反馈机制。

第四部分内容详细介绍如何实现基于 MFC 的 OpenGL 编程，以及在编程过程中的一些注意事项。介绍如何使用 OCX 控件，实现在 Microsoft Visual Basic 和 Borland Dephi 语言环

境下，调用 OpenGL 函数。

参加本书编写工作的同志还有：王群、赵双勤、张砚臣、左键、田志伟、朱文羽、沈金来、郑文宇，在此表示感谢。

在本书的编写、校对以及出版过程中，得到了环境科学出版社的大力支持与帮助，在此表示诚挚的谢意。

由于编写时间紧迫，书中疏漏与错误难免，恳请用户和读者予以指正。反馈意见请发至电子信箱：jiazg@cnscpv. org. cn

作 者

1999 年 4 月

目　录

第一章 OpenGL 简介

1.1 关于 OpenGL

OpenGL 是一个工业标准的三维计算机图形软件接口。本质上，是一个极其快速并可以方便移植的三维图形和建模库，OpenGL 可以用于创建漂亮的三维图形(与光线跟踪法视觉质量相同)，使用 OpenGL 提供的最大优越性是比光线跟踪法快一个量级。其算法由著名的计算机图形学和动画界领导者 SGI 公司开发并优化。OpenGL 的主要特点是：

①可以在网络上工作，即客户机/服务器型，显示图形的计算机（客户机）可以不是运行图形应用程序的计算机（服务器），客户机与服务器可以是不同类型的计算机，但两者要服从相同的协议。

②可以在多种硬件平台上工作（如个人计算机、工作站），OpenGL 的应用程序有非常好的移植性。

如果在专门为三维图形显示而优化设计的计算机硬件（如配有完全支持 OpenGL 硬件加速的显示卡的计算机）使用 OpenGL，可以大大提高绘图效率，但是仅由软件(称为通用)实现 OpenGL 也是可能的，Microsoft windows NT 和 Windows 95 实现属于这一类。目前，计算机硬件发展很快，有些在几年前还是极其昂贵的专业级显示卡，现在花费几百元钱就可以买到。越来越多的 PC 图形硬件生产商在其产品中加入 3D 加速功能，尽管它更多地受 3D 游戏市场的推动，但同时也推动了基于 Windows 2D 图形加速卡的发展(优化诸如画线、位图填充和操作)，现在不会有人在新机器上仅使用普通 VGA 卡，3D 加速显示卡变得越来越普及。

OpenGL 应用范围很广，不仅用于科研领域中的可视化、三维建模和三维游戏设计，也广泛应用于 CAD 工程、建筑应用软件，甚至用计算机生成 Hollywood 电影中的恐龙。因此，在占有大量市场的操作系统中(如 Microsoft Windows)引入一个工业标准三维图形 API（Application Program Interface，即应用程序接口）有重要意义，三维图形将很快成为消费者和商用软件的一个典型成分。本书正是为满足初学 OpenGL 的读者要求而编写的，全面介绍 OpenGL 的基本概念，由浅入深讲述 OpenGL 的编程技巧。

1.2 OpenGL 的历史

OpenGL 仅出现在几年前，是一个比较新的工业标准。OpenGL 的前身的 SGI 公司的 GL，IRIS GL 是该公司高端 IRIS 图形工作站上三维编程的 API，在 SGI 的计算机上，不仅有经过优化的显示复杂图形的特殊软件，提供异常快速的矩阵变换，硬件支持深度缓冲，还有其他一些特性，但当 SGI 把 IRIS GL 移植到其他硬件平台时出现了问题。OpenGL 是 SGI 为提高 IRIS GL 可移植性而努力的结果，它有 GL 的功能，又是“开放的”，可以适应其他硬件平台的操

作系统。SGI 仍然保留 GL，但不再加强其性能，只是修补一下它的错误。

对 OpenGL 的改进由 ARB 决定，其成员有 SGI、DEC、IBM、Intel、Microsoft 等，几乎所有国际知名的计算机公司。1992 年 6 月，引入了 OpenGL 1.0 版，5 天之后，Win32 开发会上，SGI 演示了运行在 IRIS Indigo 工作站上的 OpenGL，在制造商展厅引起了广泛关注，SGI 和 Microsoft 合作把 OpenGL 引入 Windows NT 的版本。1995 年 12 月，由 OpenGL ARB(Architecture Review Board)通过了 OpenGL 1.1 版，这一版本的 OpenGL 加强了性能并引入了一些新功能。1998 年 3 月，发布了 OpenGL 1.2 版，该版本向上兼容 OpenGL 1.1 版，主要加强了纹理映射和像素处理能力。

1.3 OpenGL 的工作原理

OpenGL 是一种过程图形语言，而不是描述式的，它不描述场景以及显示的方式。实际上，一定外观或效果要通过必需的程序步骤来描述，这些步骤涉及到包含 120 个左右的命令和函数。有高度可移植性的 API 用于绘制诸如三维点、线、多边形等几何要素。另外，OpenGL 支持光照、阴影处理、纹理映射、动画以及其他特殊效果。

OpenGL 不包含任何窗口管理、用户交互、文件 I/O 的函数。 这样，才能使 OpenGL 保持与操作系统无关、高度可移植的特性。每一个主机环境，如 Microsoft 的 Windows 有自己专用函数用于实现把窗口或位图的绘图控制传递给 OpenGL。

OpenGL 初次出现在 Microsoft Windows NT 3.5，其后 Windows 95 也加入了支持 OpenGL 的一系列 DLL 文件。本书主要讲述 Microsoft OpenGL 的基本实现。首先讲述三维图形学基础，然后讲述如何在 Windows NT、Windows 95 环境下编译、联接 OpenGL 程序，再后讲解 Microsoft 提供的可以使 OpenGL 图形 API 与 Microsoft GDI 一起工作的 WGL 函数。

1.3.1 图形体系——软件与硬件

使用 OpenGL 与在 Windows 中用 GDI 绘图有些不一样。事实上，选择当前笔、刷、字体和其他 GDI 实体都不会影响 OpenGL。正如 GDI 用设备描述表控制窗口绘图，OpenGL 用绘图描述表。绘图描述表与设备描述表相关，而设备描述表与 Windows 相联，由此，OpenGL 可以在窗口绘图。

正如前面所说的，在有硬件加速的系统上运行 OpenGL 会大幅度提高绘图命令的效率，PC 图形卡生产商也不断在显示卡中添加对 OpenGL 的支持。另外，正确编写的 OpenGL 应用程序在硬件加速系统和纯软件实现的系统上都可以运行。

图 1-1 说明在 Windows 下硬件加速，包括通常 GDI 加速、Direct Draw 加速和 OpenGL 加速的工作原理。在最左端，应用程序调用 GDI，WINSRV.DLL，Win32 DDI (Device Driver Interface)，Win32 DDI 直接与图形卡驱动程序通讯，在图形卡中完成加速。Direct Draw 用于图形硬件直接存取优化，完全跳过 GDI，除了中间有一个薄硬件提取层(HAL)和对不支持的特性做软件模拟外，直接与硬件打交道。Direct Draw 通常用于游戏， 直接操纵图形内存用于快速 2D 图形和动画。在最右端可以看到，OpenGL 和其他 3D API 通过 3D DDI 调用，3D DDI 为硬件制造厂加速 OpenGL 和诸如 Reality Labs 3D API 游戏。另外，特定 OpenGL 硬件加速(如 GLINT 芯片组)可以安装自己的 OpenGL 客户机驱动程序，用于特定的 *DDI*。

1.3.2 基本实现的限制

除非受硬件支持，Microsoft 的 OpenGL 基本实现有一些限制，不支持向单色打印机或小于 16 色彩色打印机打印 OpenGL 图形，不支持用于各种窗口的硬件调色板，仅有一个调色板，必须在多个应用程序间任选。

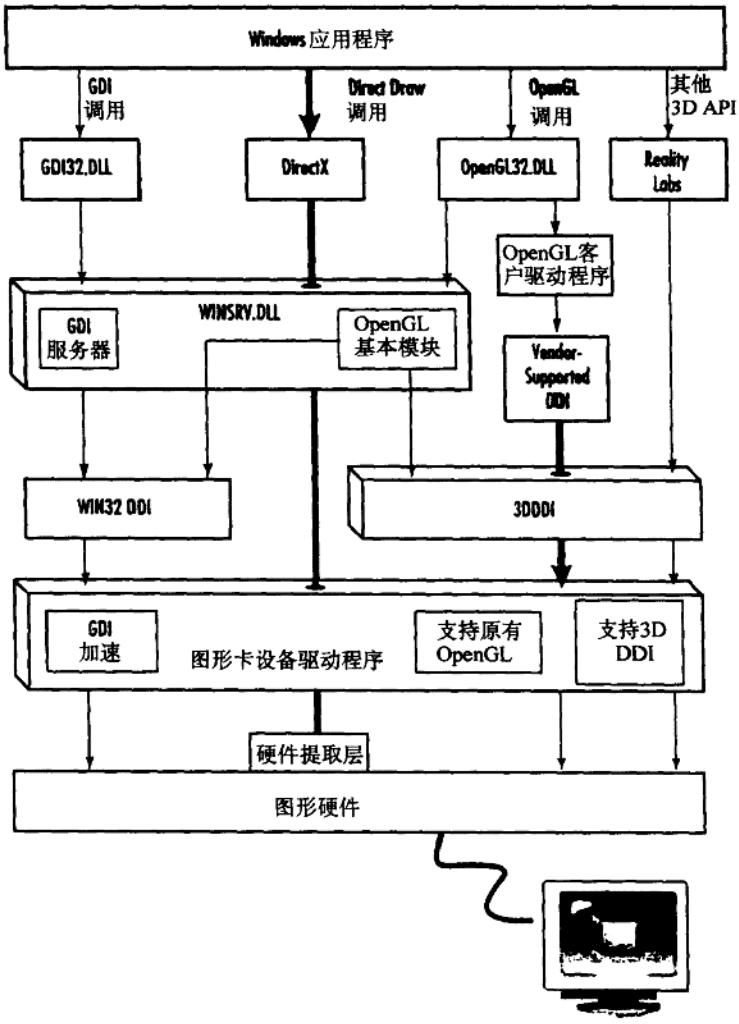


图 1-1 Windows 图形加速的原理图

第二章　三维图形学基础

为方便初学者使用、学习，在使用 OpenGL 创建三维图形之前，要先讲述一些三维图形学术语、三维图形的基本概念和坐标系统，以及如何把计算机屏幕的二维图像理解成三维图形。已经开始使用 OpenGL 有三维图形学经验的读者可以跳过本章。

2.1　三维感受

三维计算机图形实际上是在计算机屏幕上一个有深度(第三维)的二维图像。为真正看到三维物体，要用双眼分别提供物体的图像，每个眼睛接受的图像与视网膜上随时间变化的照片相似。因为得到的角度不同，这两个图像略有差别，大脑把这两个有差别的图像组合起来，就得到一个三维图片。

在图 2-1 物体远离时，夹角 θ 变小，这一三维效应可以通过加大两个图像间夹角进行放大。如果遮住一个眼睛，将无法感受距离远近。

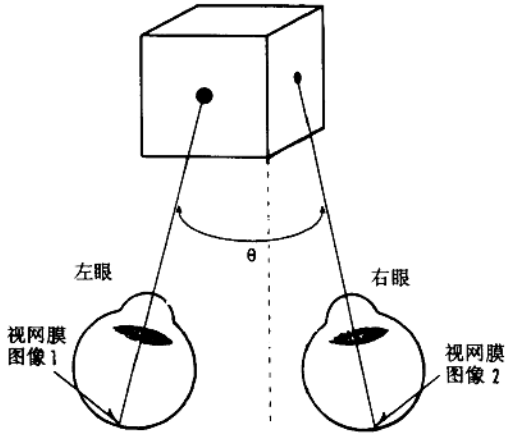


图 2-1　用眼睛观察物体

下面讲述在计算机图形学中是如何感受三维效果的：

●2D+透视投影=3D

为什么遮住一个眼睛后，三维世界并没有变成一个平面世界呢？因为许多三维世界的效果也出现在二维世界，这可以使大脑分辨出深度。对这个问题的最简单的理解就是近大远小，这一效应称为透视，透视与颜色变化、纹理、光照、明暗和颜色强度相互作用在一起才构成对三维图像的感受。

透视本身也可以得到三维外观，图 2-2 是一个简单的方块线框图，即使没有着色和明暗处理，方块仍然有三维物体外观，如果注视方块的时间足够长，方块的前后将换位置，因为大脑会因为图中没有绘制任何表面而混淆了。

●消隐线

图 2-2 包括的信息足可以反映出三维物体的轮廓，但还无法分辨出方块的前面和后面，如果图 2-2 是实体，则看不到方块背面的角点，即使是线框方块，前面的一些线也会遮住后面的部分线框。若在二维图形上模拟这种情况，一定要消掉被前面遮住的那些线，称为消隐线，如图 2-3 所示。

●颜色和明暗处理

图 2-3 仍然不像是真实的物体，方块表面的颜色与背景完全相同，仅能看到物体的前缘，而真实的方块应该有一些颜色或纹理，如木块。在计算机或纸上，若仅给方块着色并在二维上绘制它，得到类似于图 2-4 所示的效果。

如果把边缘都画成不同的颜色，就会体验到三维效果，也可以利用光照给各个侧面有相同颜色的方块做明暗处理，如图 2-5 所示。

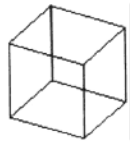


图 2-2　一个简单的方块线框图

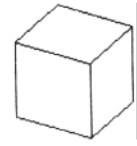


图 2-3　方块线框图消隐

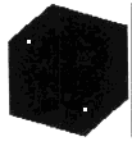


图 2-4　着色的方块线

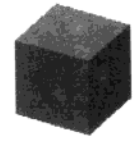


图 2-5　明暗处理的方块

●光照和阴影

正确使用光照效果是 OpenGL 绘制完美的图片过程中一个必不可少的要素。光照对三维物体有两个重要的影响，第一，若以某一角度观察或照亮一个颜色单一的表面，使其看上去有明有暗；其二，由于绝大多数实体不透光，会投下一个阴影。

有 2 个光源影响三维物体：一个是泛光，没有方向，仅仅是能给实体造成明暗效果的均匀光照。泛光使远处边缘看上去暗一些；另一个光源来自称为“灯光”的光源，可以改变实体的明暗，也可以产生阴影。图 2-6 所示为受光源照射的方块所产生的阴影。

图 2-6　方块的光照和阴影

2.2　坐标系统

既然可以在二维表面(计算机屏幕)感受三维效果，那么如何在屏幕上绘制这些物体呢？在计算机屏幕上绘制点、线、多边形，要根据行、列绘出一个位置。例如，标准 VGA 屏幕由左至右有 640 个像素，由上至下有 480 个像素，屏幕中心的点为(320，240)，即由屏幕左起 320 个像素，由上向下第 240 个像素。

在 OpenGL 中，要为所使用的窗口定义坐标系统，并把所指定的坐标映射到物理屏幕像素上。下面讲述如何在二维绘图中实现，进而把这一原理扩展到三维坐标系统。

1．二维直角坐标

二维图形最常用的坐标系统是直角坐标。由 x，Y 的两个值定义直角坐标，x 坐标为水平方向位置，y 为垂直方向位置的测度。

直角坐标的原点为 x=0，y=0，也可以写成坐标对(0，0)形式，图 2-7 描述的即是二维直角坐标，x 轴与 y 轴互相垂直。可以用不同的窗口映射方法，使 OpenGL 对用户定义的坐标在绘图时有不同的理解。本书后面将讲述这些真实空间坐标向窗口坐标的不同映射方法。

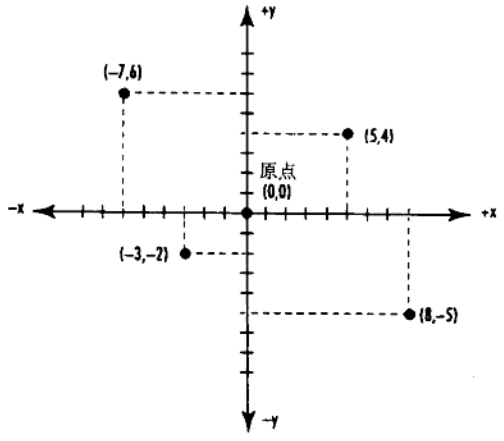


图 2-7　二维直角坐标

2．坐标裁剪

窗口长度的单位是像素，在开始绘制点、线、多边形之前，用户要告诉 OpenGL 如何把指定的坐标对变成屏幕坐标，这可由指定窗口上直角坐标空间区域来完成，这一区域称为裁剪区。在二维空间，裁剪区是窗口内 x,y 的最大、最小值；另一个方法是定义相对于窗口的原点位置，图 2-8 绘出这两个常用的裁剪区。

第一个例子中(图 2-8 左)，窗口内 x 由左到右的取值范围为 0 到 +150，y 由下到上的取值范围是 0 到+100，屏幕的中心坐标为(75，50)，第二个例子的 x 取值为-75 到+75，y 取值范围-50 到+50，屏幕的中点为(0,0)。也可以用 OpenGL 函数(或普通 GDI 绘图的 Windows 函数)把坐标系统颠倒过来。应该注意的是，缺省的 Windows 窗口是+y 轴由窗口顶部指向底部。尽管写文本是由上至下有益处，但绘图时这个缺省映射不方便。

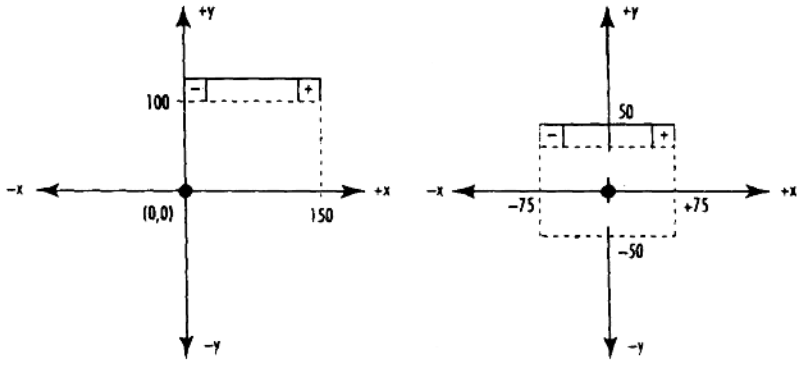


图 2-8　两个裁剪区

3．视见区——把窗口变为三维

以像素为长度单位的窗口尺寸只有在特殊情况下，裁剪区的宽、高才完全一致，因此要把逻辑上的直角坐标映射到物理上的屏幕像素坐标，这一映射可由设定视见区完成，视见区是窗口中用于绘制的裁剪区域。把裁剪区映射到窗口的某一区域，通常把视见区设为整个窗口，当然，也可以是窗口的一个部分。

图 2-9 给出一个 300×200 像素的窗口，视见区为整个窗口大小，若把裁剪区设为 x 轴的范围 0 到 150，y 轴 0 到 100，在视见窗口上，逻辑坐标会被映射到一个更大的屏幕坐标系统。逻辑坐标系统的一个单位长度与窗口的物理坐标系统 2 个单位长度相匹配。

若视见区与裁剪区相匹配，视见窗口大小仍为 300×200，那么视见区仅为窗口的左下角，如图 2-10 所示。

可以用视见区放大、缩小窗口上的图像。如果把裁剪区设置得比窗口还要大，则仅能显示一部分裁剪区。

4．绘图要素

无论是二维还是三维空间，所绘制的物体都是由更小的几何形状组合成的，这些小单元称为几何要素。几何要素是诸如点、线、多边形等二维表面，把它们装配到三维空间就可以创建三维物体，如图 2-5 的三维方块是由 6 个二维正方形构成的，几何要素的角点称为顶点。在第五章中将介绍 OpenGL 几何要素及其使用。

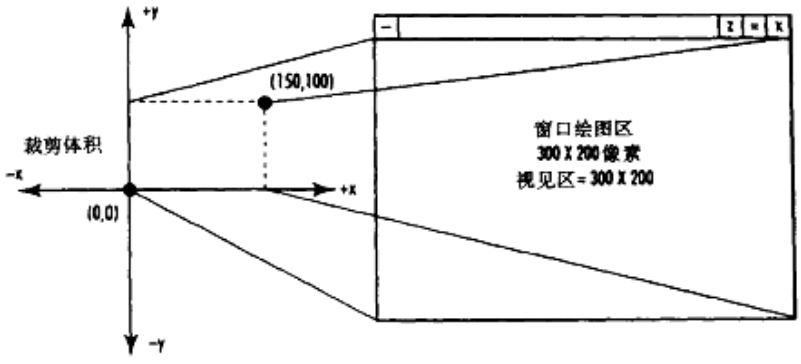


图 2-9　视见区尺寸为裁剪区的两倍

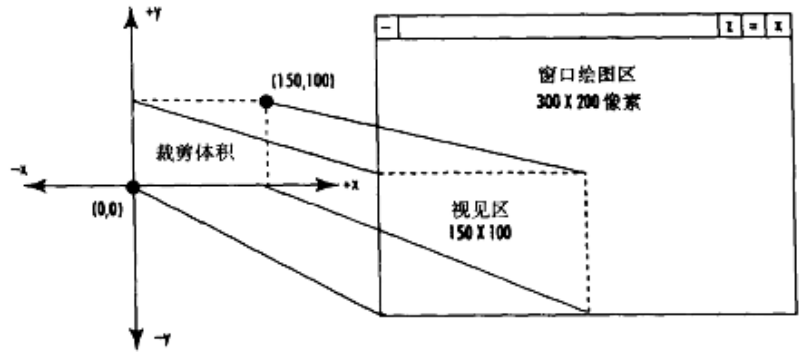


图 2-10　视见区与裁剪区面积相同

5. 三维直角坐标

把二维坐标系统扩展到三维要加上深度坐标，图 2-11 为加上 z 轴的直角坐标系统，z 轴垂直于 x, y 轴，z 轴的正方向是垂直于屏幕指向观察者的，在三维坐标系统中定义一个位置要用 x, y, z 三个坐标。

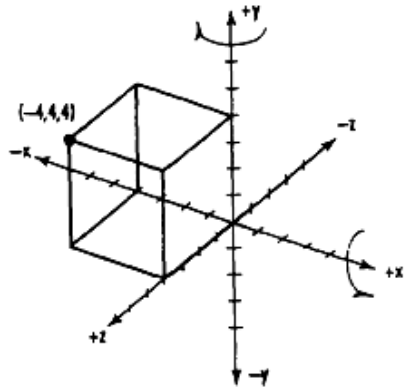


图 2-11　三维直角坐标系统

2.3 投影三维的实质

尽管在三维空间可以用直角坐标定义位置，但屏幕上的像素是二维的，OpenGL 是如何把这些三维坐标转换成屏幕上的二维坐标的呢？坐标转换是通过三角函数和简单矩阵变换完成的。有关内容将在本书第六章中讲述。然而，用 OpenGL 创建图形时，不必对这些数学公式有更深入理解，只需理解称为所谓“投影”的概念。把三维坐标投影到二维屏幕上，如图 2-12 所示为把一个屋子投影到一块玻璃上。通过定义投影，用户可以指定显示在窗口中的剪裁体积。在 OpenGL 中有两大类投影方式：正交投影与透视投影。

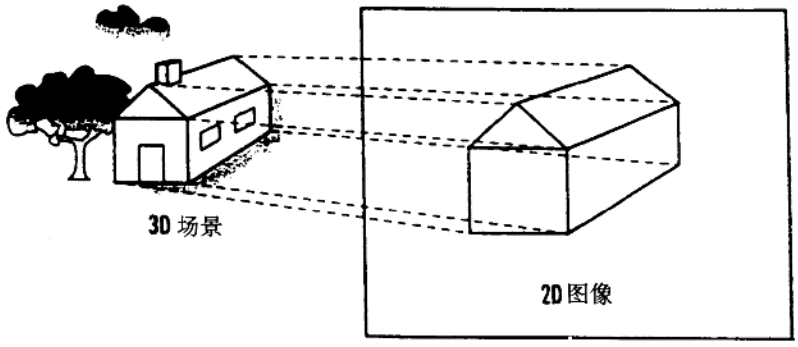


图 2-12 投影示意图

1. 正交投影

正交投影或称平行投影，这种投影方式是指定一个方形或矩形裁剪体积使用，在这个裁剪体积外的物体被丢掉，被投影物体的尺寸不变。这类投影方式常用于建筑设计或 CAD。

正交投影的裁剪体积由远、近、左、右、顶、底裁剪平面构成，在这一体积内的物体被投影到屏幕上成为二维图像。

2. 透视投影

透视投影是另一个常用的投影方式，它考虑物体近大远小的视觉效果。其裁剪体积称为截头锥体，越靠近取景体积前面的物体越接近其原有尺寸，反之亦然。这类投影在三维动画和模拟时更逼真。

2.4 本章小结

本章介绍了三维图形学基础，讲述了为什么要从不同角度感受真实三维空间，讲述了投影、消隐、着色、明暗处理、光照技巧，以及 OpenGL 把三维图形投影到二维屏幕的两种方法。

本章有意略去了 OpenGL 创建这些效果的细节。本书后面章节，将讲述如何使用这些技巧并最大限度地开发 OpenGL 的性能。

第三章　使用 AUX 库函数

本章首先介绍 OpenGL 的工作原理，再介绍给变量和函数命名的规则，如果读者已经开始写一些程序，就可以跳过第一部分，而直接学习 AUX 库函数使用部分的内容。

3.1　OpenGL API

与大家熟悉的计算机高级语言，如 C、FORTRAN 等不同，OpenGL 不是一门计算机编程语言，而仅是一个 API（Application Program Interface，即应用程序接口）。如果提到一个程序是基于 OpenGL 的或是 OpenGL 应用程序，那么这个程序是用某一编程语言写成的，只是在其中调用了 OpenGL 的一些库函数。OpenGL 的库函数有在各种编程语言下的定义，如 C 语言、FORTRAN 等。当然，可以把多个图形软件包结合起来，由 OpenGL 完成一些特定的任务，而用特定环境下的图形软件，如 Windows GDI 完成其他工作。

作为一个 API，OpenGL 库函数符合 C 语言调用规则。这就是说，用 C 语言编写的程序可以比较方便地调用 API。其原因在于：一方面，函数本身用 C 语言编写；另一方面，C 语言提供了一系列调用汇编语言或其他语言的中间函数。本书多数的程序用 C 或 C++编写，运行在 Windows NT、Windows 95 环境下。C++程序可以与 C 语言相同方式调用 C 函数和 API。在 Microsoft Visual Studio 环境下的 FORTRAN 可以很方便地使用 OpenGL 模块，并实现与 Visual C++的混合编程。其他编程语言，如第四代编程语言(称为“4GLs”)类，如 Visual Basic 也能够通过 OCX 控件调用 OpenGL（本书第十五章将进行详细讨论）。

3.1.1　OpenGL API 的构成

OpenGL API 由三个不同的库函数构成：

①辅助库 AUX，也称为“工具箱”库函数（glaux.lib）。该库函数的声明在 glaux.h 头文件中，这一库函数的内容并不是 OpenGL 定义的内容，而是为调用 OpenGL 函数提供的与平台独立框架的工具箱。此库函数的前缀为“aux”。

②OpenGL 核心库（opengl32.lib）。这些函数由 OpenGL ARB 定义，是 OpenGL 的核心函数。其头文件为 gl.h，前缀为“gl”。

③OpenGL 应用库函数（glu32.dll），头文件为 glu.h。这个库包括的函数可以使用户更方便地编写程序、完成任务，如绘制球、盘、圆柱。应用库函数实质上是用 OpenGL 命令写成的，可以在任意支持 OpenGL 的平台上使用。这些函数的前缀是“glu”。

值得注意的是，在运行 OpenGL 应用程序时，Windows 系统所在目录的 system 子目录中一定要有 opengl32.dll、glu32.dll。

本章主要介绍 AUX 库函数、OpenGL 基础以及 gl 库的一些命令。