计算机技术丛书

Open GL 三维图形设计与制作

白建军　朱亚平　梁辉　姚东　等编著

人民邮电出版社

计 算 机 技 术 丛 书

OpenGL 三维图形设计与制作

白建军　朱亚平
梁　辉　姚　东　等编著

人 民 邮 电 出 版 社

内 容 提 要

本书是一本专门介绍三维图形设计与制作的流行软件——OpenGL 的图书。OpenGL 目前已被广泛地用于可视化技术、实体造型、CAD/CAM、模拟仿真等诸多领域内，它是一套独立于操作系统和硬件环境的三维图形库，有着强大的图形功能和良好的跨平台移植能力，已成为事实上的图形标准。本书对 OpenGL 的使用方法和三维图形编程技巧进行了详细的介绍。

本书共分 12 章，内容包括：绪论、OpenGL 基本几何对象绘制、OpenGL 中的坐标变换、OpenGL 颜色、OpenGL 中的光照处理、OpenGL 位图和图像、OpenGL 中的纹理映射、OpenGL 效果处理、OpenGL 帧缓存与动画、显示列表、OpenGL 求值器和 NURBS、选择模式与反馈模式等等。

本书内容实用性较强，可供计算机用户、三维图形设计制作人员以及大专院校相关专业的师生阅读参考。

计算机技术丛书

Open GL 三维图形设计与制作

◆ 编　　著　白建军　朱亚平　梁　辉　姚　东　等

责任编辑　王晓明

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号

北京密云春雷印刷厂印刷

新华书店总店北京发行所经销

◆ 开本：787 × 1092　1/16

印张：32.25

字数：808 千字　　1999 年 8 月第 1 版

印数：1 - 6 000 册　　1999 年 8 月北京第 1 次印刷

ISBN 7-115-08021-6/TP·1252

定价：47.00 元

丛书前言

世界上发达国家普遍重视发展以计算机和通信为核心的信息技术、信息产业和信息技术的应用，一些经济发达国家信息产业发展迅速。

当前，我国处于国民经济高速发展时期。与此相伴随，必将有信息技术、信息产业和信息技术应用的高速发展。各行各业将面临信息技术应用研究与发展的大课题以及信息化技术改造的大任务、大工程。

为了适应信息技术应用大众化的趋势，提高应用水平，我们组织编写、出版了这套“计算机技术丛书”。这套丛书以实用化、系列化、大众化为特点，介绍实用计算机技术。

这套丛书采取开放式选题框架，即选题面向我国不断发展着的计算机技术应用的实际需要和国际上的实用新技术，选题不断增添又保持前后有序。

这套丛书中有的著作还拟配合出版软件版本，用软盘形式向读者提供著作中介绍的软件，以使读者方便地使用软件。

我们希望广大读者为这套丛书的出版多提意见和建议。

前　言

OpenGL 是由 SGI 公司的 IRIS GL 发展而来的。最初 IRIS GL 被设计用来作为 SGI 工作站的图形引擎，后来根据用户希望将 IRIS GL 移植到开发系统的愿望，SGI 公司决定扩展 IRIS GL 的功能，以便使其能够更加容易地移植到不同的硬件和操作系统。1992 年 SGI 公司推出了 OpenGL，它是一套独立于操作系统和硬件环境的三维图形库，有着强大的图形功能和良好的跨平台移植能力，已成为事实上的图形标准。OpenGL 目前已被广泛地应用于可视化、实体造型、CAD/CAM、模拟仿真等诸多领域。

随着微机技术的不断发展，微机的性能越来越好，其图形处理能力也大大提高。Microsoft 和 SGI 等公司，不失时机地将 OpenGL 在 Windows 95、Windows NT 平台上予以实现。Microsoft 在 Visual C＋＋2.0 以上版本中内置了 OpenGL，这更为 OpenGL 在微机上的应用创造了有利条件，使广大微机用户能够享受 OpenGL 所带来的强大图形功能。

本书作者长期从事计算机图形方面的编程工作，有着较丰富的三维图形应用程序的编程经验。作者编写本书的目的，是为了使读者能通过具体的程序和翔实的内容介绍，来掌握 OpenGL 三维图形编程的方法和技巧。考虑到目前 Visual C++编程环境的广泛应用，本书中所采用的大多数例子均是在 Visual C++上编译通过的。另外为了满足不同行业读者的需要，本书所涉及的内容覆盖面较为广泛。作者真诚地希望本书能成为对广大读者有用的一本参考书。

本书主要由白建军、朱亚平、梁辉、姚东、白建顺编写。另外参加本书编写及整理资料的人员还有：冯峰、王雪梅、陈卫国、王爱民、边晓东、朱廷、刘彬、许清、梦容客、苍鹰、刘保国、魏金刚、冯良、王红梅、乔蕊青、高丰、李梦燕、须瑞、梁永强、安健、雷洪亮、鹿篪、马民、范为民、周志勇、许欣、高云霞、胡萍、董玫、马晓红、武勇、许殊、赵钢柱、陶洁、刘鸿燕、房蕾、徐军、马增顺、高冈、滦因杰。

由于计算机软件的发展很快，加上作者的水平有限，书中的错误之处在所难免，希望广大读者批评指正。

作　者

1999 年 5 月

目 录

第一章 绪论

1.1 OpenGL 简介

OpenGL 即开放性图形库（Open Graphic Library），是一个三维的计算机图形和模型库，它是由 SGI 公司为其图形工作站开发的 IRIS 演变而来的。OpenGL 适用于多种硬件平台及操作系统，用户用这个图形库不仅能方便地制作出有极高质量的静止三维彩色图像，还能创建出高质量的动画效果。

由于 OpenGL 在三维真实感图形制作中具有的优秀性能，一批在计算机市场上占主导地位的大公司（如 Microsoft、IBM、DEC、SUN 等）都将其作为自己的图形标准，从而使 OpenGL 已成为新一代的三维图形工业标准。

OpenGL 的前身是 SGI 公司为其图形工作站开发的 IRIS GL，为使其能适用于多种硬件，SGI 开发了 OpenGL。1992 年 7 月，SGI 公司发布了 OpenGL1.0 版本；1995 年 12 月，又发布了 OpenGL 的 1.1 版本，在 OpenGL 中引入了一些新功能，包括改进打印机支持、RGBA 及色彩指数模式、提高顶点位置、引入新的纹理特性等。

由于 OpenGL 本身就是一个与硬件无关的编程接口，所以 OpenGL 的应用程序有较好的移植性，这也是 OpenGL 中没有包括处理窗口和用户输入命令的原因。OpenGL 的另一个显著特点是它的网络功能。在网络环境下，即使创建图形程序的计算机不是运行图形程序的计算机，只要它们遵守同样的通信协议，OpenGL 就可正常工作。

近几年，微型机的硬件性能有了很大的提高，配以 32 位操作系统的微型机，其性能已超过了早期工作站的水平，使得原本只能在工作站实现的图形功能已经可以移植到微机平台上。而 Visual Studio 等集成软件开发环境的出现，也使在微机上实现三维图像更加方便。

1.2 OpenGL 入门

本节我们将通过一个简单的 OpenGL 程序来说明 OpenGL 的语法规则和相关函数库，并通过这个程序使读者对 OpenGL 有一个初步理解。

1.2.1 一个简单的 OpenGL 程序

程续列表 1.1：

```
#include<window.h>
#include<GL/gl.h>
#include<GL/glu.h>
#include<GL/glaux.h>
```

```
#include<stdio.h>
void myinit(void);
void CALLBACK myReshape(Glsizei w, Glsizei h);
void CALLBACK display(void);
//初始化
void myinit(void)
{
glClearColor(0.0,0.0,0.0,0.0);      //将窗口清为黑色
}
void CALLBACK display(void)
{
glClear(GL_COLOR_BUFFER_BIT);
// 将颜色缓存清为 glClearColor 命令所设置的颜色,即背景色
glColor4f(0.2,0.8,1.0,1.0);
// 选颜色（R，G，B）
auxWireCube(1.0);
// 绘制六面体的虚线图
glFlush();
// 强制绘图，不驻留缓存
}
// 是一个用于窗口改变大小时的处理，与绘图无关
void CALLBACK myReshape(Glsizei w, Glsizei h)
{
glViewport(0,0,w,h,)
}
void main(void)
{
auxInitDisplayMode(AUX_SINGLE|AUX_RGBA);
// 窗口显示单缓存和 RGB（彩色）模式
auxInitPosition(0,0,400,400);
// 大小 x=400 y=400 （0，0）是屏幕左上点
auxInitWindow(openglsample.c);
// 初始化窗口，参数是标题
myinit();
auxReshapeFunc(myReshape);
auxMainLoop(display);
}
```

运行结果如图 1－1 所示。

图 1－1 一个简单的例子

以上是一个很简单的 OpenGL 程序，它体现了 OpenGL 的最基本的几个方面，在下面的几节中，将分别以程序 1.1 为例，来介绍 OpenGL 的基础知识。

1.2.2 OpenGL 相关函数库

在程续列表 1.1 的开始部分，读者可以看到<gl.h>、<glu.h>、<glaux.h>这样三个头文件。这就是与 OpenGL 编程关系最密切的三个函数库的头文件。

(1) OpenGL 核心函数库，前缀为“gl”。在这个函数库中，共有 115 个不同的三维操作函数。这些函数提供了最基本的绘图命令，用来描述几何体形状、矩阵形状，进行光照、纹理、雾和反走样处理。

(2) OpenGL 应用程序函数库，前缀为“glu”。这个函数库包括 43 个辅助函数，用来管理坐标变换，多边形镶嵌，绘制 NURBS 曲线、曲面，以及处理错误等。

(3) OpenGL 编程辅助函数库，前缀为“aux”。此函数是 OpenGL 的辅助库，包括 31 个与平台无关的函数，提供了窗口管理和消息响应函数及一些简单模型的制作函数。

除此以外，当读者更深一步接触 OpenGL 时，会应用到以下另外几个与 OpenGL 相关的函数库：

(1) 对 Windows95 和 WindowsNT 系统扩展的函数库，包括 WGL 函数（前缀为“wgl”）和 Win32 函数。这些函数用来管理显示列表、字体位图、绘图描述表及扩展函数。

(2) 对 X-windows 系统扩展的函数库，前缀为“glx”，类似于 WGL 函数，主要是针对 X-windows 系统的。

1.2.3 OpenGL 语法规则

通过上一节的介绍，读者了解了 OpenGL 的相关函数库的有关内容。在本节中，我们通过程续列表 1.1 来进一步了解 OpenGL 的语法规则。

OpenGL 的语法规则其实相当简单，以例程 1.1 中的函数 glColor4f 为例，前缀“gl”，指这个函数是 OpenGL 的核心库函数，组成命令的单词首字母大写，如“Color”；后缀“4”表示有 4 个变量；“f”说明变量类型为 32 位浮点数（OpenGL 变量的数据类型参见表 1－1）。核心库函数常量是以 GL 开头，均用大写字母，并用下划线将每个关键词分开，如 GL_COLOR_BUFFER_BLT。

表 1－1　　OpenGL 命令后缀及变量的数据类型

后缀	数据类型	C 语言类型	OpenGL 类型定义
b	8 位整数	signed char	GLbyte
s	16 位整数	short	GLshort
i	32 位整数	long	GLint,GLsizei
f	32 位浮点数	float	GLfloat,GLclampf
d	64 位浮点数	double	GLdouble,Glclampd
ub	无符号 8 位整数	unsigned char	GLubyte,GLboolean
us	无符号 16 位整数	unsigned short	GLushort
ui	无符号 32 位整数	unsigned long	GLuint,GLenum,GLbitfield

初步了解了 OpenGL 的语法规则后，请读者再来看一些例子。如 glVertex2f 命令，即必须以 32 位浮点数的形式提供(x,y)坐标；而 glVertex3sv，则必须提供一个由(x,y,z)三个 16 位整数组成的数组，后缀“v”表示这个命令带有一个指向数组的指针。上面两个命令可以用 glVertex*表示，由此可见星号表明可有多个实际的命令名。

接着再举一个例子来让读者体会带指针和不带指针命令的异同：

```
glColor3f(1.0, 0.0, 0.0);
float color_array[] = {1.0, 0.0, 0.0};
glColor3fv(color_array);
```

另外，有时会遇到如 glVertex3{sfd}这样的命令，这个命令表示 glVertex3s、glVertex3f、glVertex3d 三个命令。

1.2.4　对 OpenGL 的基本理解

通过前两节的介绍，相信读者对 OpenGL 的语法规则及函数库都有了一定了解。本节将对程续列表 1.1 进行较详细的说明。

1. 一个完整的 OpenGL 程序的基本结构

实际上，一个 OpenGL 程序的基本结构是很简单的，无论多么复杂的 OpenGL 程序，都可以大致分解成以下三个部分：

(1) 定义窗口：这部分将定义所绘制的三维图形在屏幕坐标中的显示位置，并定义显示图形窗口的大小和窗口的性质。在例程 1.1 中，函数 auxInitPosition(0,0,500,500)定义了显示三维图形窗口的位置和大小，这个函数表示以当前屏幕坐标系的右下角（0，0）为原点，开一个宽度和高度各为 500 的的窗口。

函数 auxIniDisplayMode(AUX_SINGLE|AUX_RGBA)定义了这个窗口的显示属性，即使用单缓存和 RGB（彩色）模式。

(2) 初始化操作：由于 OpenGL 的绘图方式是由一系列的状态决定的（关于 OpenGL 的状态机制，参见附录 A），所以初始化这些状态变量可以为下一步的图形显示做一些准备工作，使显示 OpenGL 的三维图形更方便，其中包括清缓存区、定义光照模型、定义纹理映射等基本操作的初始化以及定义视口等。如在程序列表 1.1 中：

```
void myinit(void)
{
glClearColor(0.0,0.0,0.0,0.0);
```

}

这个语句就是执行初始化操作，将屏幕背景清为黑色。

(3) 绘制及显示图形：绘制及显示图形即根据实际应用的要求，通过建立三维模型，设置运动轨迹，改变 OpenGL 的状态变量，来绘制具有高度真实感的三维图形。

例程 1.1 中，语句 auxWireCube(1.0)用来绘制六面体的虚线图，glFlush()强制完成绘图。

2. 回调(CALLBACK)函数

CALLBACK 函数是一些用来让系统调用的函数，系统调用它们来实现显示、接受键盘输入功能，读者可以将它们列在主程序初始化后（顺序一般无关紧要），系统会自动执行它们。例如：在 void CALLBACK display(void)中你写好想要绘制的三维图形，然后在主程序中使用 auxMainLoop(display);就可以让这个图形一直显示，这很类似 VC 中的 OnPaint()函数。

1.3 OpenGL 基本操作

客观世界的各种事物的形状虽然千变万化，但用计算机将之描述出来却只需要把一系列基本操作组合起来。OpenGL 提供了以下基本操作：

(1) 绘制物体：真实世界里的任何物体都可以在计算机中用简单的点、线、多边形来描述。OpenGL 提供了丰富的基本图元绘制命令，从而可以方便地绘制物体。

(2) 变换：可以说，无论多复杂的图形都是由基本图元组成并可以经过一系列变换来实现。OpenGL 提供了一系列基本的变换。如取景变换、模型变换、投影变换及视口变换。

(3) 光照处理：正如自然界不可缺少光一样，绘制有真实感的三维物体必须做光照处理。

(4) 着色：OpenGL 提供了两种物体着色模式，一种是 RGBA 颜色模式，另一种是颜色索引模式。

(5) 反走样：在 OpenGL 绘制图形过程中，由于使用的是位图，所以绘制的图像的边缘会出现锯齿形状，称为走样。为消除这种缺陷，OpenGL 提供了点、线、多边形的反走样技术。

(6) 融合：为了使三维图形更加具有真实感，经常需要处理半透明或透明的物体图像，这就需要用到融合技术。

(7) 雾：正如自然界中存在烟雾一样，OpenGL 提供了“fog”的基本操作来达到对场景进行雾化的效果。

(8) 位图和图像：在图形绘制过程中，位图和图像是非常重要的一个方面。OpenGL 提供了一系列函数来实现位图和图像的操作。

(9) 纹理映射：在计算机图形学中，把包含颜色、alpha 值、亮度等数据的矩形数组称为纹理。而纹理映射可以理解为将纹理粘贴在所绘制的三维模型表面，以使三维图形显得更生动。

(10) 动画：出色的动画效果是 OpenGL 的一大特色，OpenGL 提供了双缓存区技术来实现动画绘制。

以上这些基本操作在以后的章节中将详细介绍。通过上几节的介绍，读者可能会发现

OpenGL 并没有提供三维模型的高级命令，它也是通过基本的几何图元——点、线及多边形来建立三维模型的。目前，有许多优秀的三维图形软件（如 3Dmax）可以较方便地建立物质模型，但又难以对建立的模型进行控制。若把这些模型转化为 OpenGL 程序，则可随心所欲地控制这些模型来制作三维动画、实现虚拟现实。

1.4 OpenGL 状态机制

本节中，我们将继续深入探讨 OpenGL 的内部机制。OpenGL 的绘图方式是由一系列状态决定的，如果设置了一种状态或模式而不改变它，OpenGL 在绘图过程中将一直保持这种状态或模式。如程续列表 1.1 中的以下语句：

```
void myinit(void)
{
glClearColor(0.0,0.0,0.0,0.0);
}
```

其中函数 glClearColor(0.0,0.0,0.0,0.0)将视口背景色清为黑色，如果不改变这种状态，视口背景色将一直保持为黑色。当然，这只是 OpenGL 众多状态中的一种，其它状态有模型变换、线及多边形的填充图案、多边形绘制模式、光源的位置及特性、被绘制物体的材质等。OpenGL 是用状态变量来描述这些状态的，并用 glEnable()和 glDisable()命令来控制这些状态的打开和关闭（关于 OpenGL 状态变量及其相关数据，请参见附录 A）

1.4.1 状态查询命令

在 OpenGL 中，每个状态变量都有一个缺省值，我们可以根据数据类型的不同，用表 1－2 所列出的 4 个命令之一来查询状态变量的值。

表 1－2　　OpenGL 状态查询命令

查询命令	说明
void glGetBooleanv(Glenum pname, GLboolean *params);	获得 Boolean 类型状态变量值
void glGetIntegerv(GLenum pname, GLint *params);	获得 Integer 类型状态变量值
void glGetFloatv(GLenum pname, GLfloat *params);	获得 Float 类型状态变量值
void glGetDoublev(GLenum pname, GLdouble *params);	获得 Double 类型状态变量值

在上述命令中，pname 是一个象征所返回状态变量的符号常量，params 是一个指向放置返回状态变量类型的指针（有关 pname 详细资料，请参见附录 A）

OpenGL 还提供了以下一些用于返回特殊状态变量的专用命令：

```
void glGetClipPlane(GLenum plane, GLdouble *equation);
GLenum glGetError(void);
void glGetLight{if}v(GLenum light, GLenum pname, TYPE *params);
void glGetMap{ifd}v(GLenum target, GLenum query, TYPE *v);
void glGetMaterial{if}v(GLenum face, GLenum pname, TYPE *params);
void glGetPixelMap{f ui us}v(GLenum map, TYPE *values);
```