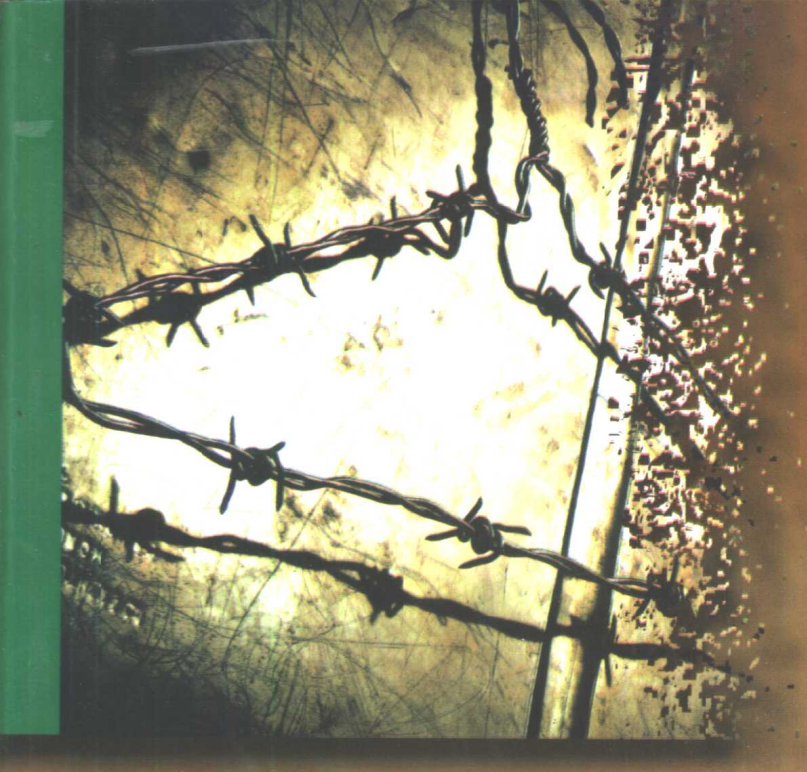

李颖 薛海斌 朱伯立 朱仲立 编著

Open GL 函数与范例解析手册

工程师工具软件应用系列

国防工业出版社

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 函数与范例解析手册

工程师工具软件应用系列

内 容 简 介

本书主要介绍的是 OpenGL 的最新版本——OpenGL 1.2 版的命令函数。OpenGL 主要有三个函数库：GL 库、GLU 库和 GLUT 库，这三个库的所有函数在本书都有详细全面的介绍。同时，为了说明某些常用函数的使用方法，书中还给出了完整的程序示例，以便读者能够快速掌握这些函数的使用方法。

本书可以作为 OpenGL 程序设计者的辅助参考书，主要适用于 OpenGL 程序设计人员。

图书在版编目(CIP)数据

OpenGL 函数与范例解析手册/李颖等编著 .—北京：国防工业出版社，2002.1
(工程师工具软件应用系列)
ISBN 7-118-02588-7

Ⅰ.O... Ⅱ.李... Ⅲ.图形软件，OpenGL－程序设计 Ⅳ.TP391.41

中国版本图书馆 CIP 数据核字(2001)第 041286 号

国防工业出版社出版发行
(北京市海淀区紫竹院南路 23 号)
(邮政编码 100044)
北京奥隆印刷厂印刷
新华书店经售
*
开本 787×1092 1/16 印张 28¾ 662 千字
2002 年 1 月第 1 版 2002 年 1 月北京第 1 次印刷
印数：1—3000 册 定价：39.00 元

前　言

作为三维图形硬件的软件接口，OpenGL 的主要目的是为了向帧缓存中绘制二维和三维几何物体，这些物体以一系列顶点或像素来描绘，顶点定义了几何物体，而像素则定义的是图像，OpenGL 对这些数据进行处理，以便把这些数据都转换为像素，从而构成帧缓存中最终所希望的图像，因此，利用 OpenGL 函数，程序员可以创建高质量的静态和动态三维彩色图像。

本书主要介绍的是 OpenGL 1.2 版的命令函数，在这一版中的 OpenGL 主要由以下三个库组成：GL 库、GLU（OpenGL Utility Library）库和 GLUT 库（OpenGL Utility Library Toolsbox）。GL 库中包含的是 OpenGL 最基本的命令函数，利用这些函数可以执行如数据输入、指定光照和颜色、生成纹理坐标、转换窗口坐标、逻辑缓存操作，以及获取 OpenGL 状态变量等任务。GL 库中的函数均以 gl 字母开头。GLU 库是 OpenGL 的实用库，它是对 OpenGL 核心函数的补充，利用 GLU 库中的函数可以执行如管理纹理中使用的图像、转换坐标、绘制二维几何物体，以及绘制 NURBS 曲线和曲面等任务。GLU 库中的函数均以 glu 字母开头。GLUT 库是 OpenGL 的实用工具箱，它使得开发 OpenGL 程序变得更加容易，GLUT 库的函数主要执行如处理多窗口绘制、处理回调驱动事件、生成层叠式弹出菜单、绘制位图字体和笔画字体，以及各种窗口管理等任务。GLUT 库的函数均以 glut 字母开头。

本书中的函数是按照 GL 库、GLU 库和 GLUT 库的顺序编排的，对于每个库中的函数均按照函数功能进行了分类，并放在独立的章节中，同时给出了部分函数的程序示例代码，书中的示例程序均调试通过，可以作为读者学习和掌握 OpenGL 的很好的范例。本书的编写和程序调试工作由李颖、薛海斌和朱伯立完成。

本书对于需要使用 OpenGL 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。

作　者

2001 年 4 月

目　录

第一篇　GL库函数

第二篇 GLU 库函数

第三篇　GLUT 库函数

第一篇　GL 库函数

第一章　使用颜色

1.1　glShadeModel —— 选择平面明暗模式或光滑明暗模式

C 语言描述

void glShadeModel(GLenum *mode*)

参数

mode　指定表示明暗模式的符号值，可以选择 GL_FLAT 和 GL_SMOOTH，缺省值为 GL_SMOOTH。

说明

OpenGL 图原可以为平面明暗模式或光滑明暗模式，缺省时，光滑明暗模式在图原光栅化时插值计算顶点的颜色，实际上为每个最终的像素片元应用不同的颜色。平面明暗模式只选择一个顶点的计算颜色，并将该颜色应用到光栅化单个图原所产生的所有像素片元上。无论哪种情况，如果激活光照，计算到的顶点颜色都是光照后的结果颜色，如果关闭光照，计算到的颜色就是指定顶点时的当前颜色。

对于点，平面明暗模式和光滑明暗模式是没有区别的，当发出 glBegin 命令时，从 1 开始计数顶点和图原的数目，在平面明暗模式下，线段 i 的颜色就是计算到的顶点 $i+1$ 的颜色。同样从 1 开始计数，平面明暗模式下多边形的颜色由下面计算到的顶点颜色来确定，如表 1-1 所示。

表 1-1　平面明暗模式下多边形颜色的确定

第 i 个多边形的图原类型	顶　点
单个多边形($i=1$)	1
相邻的三角形	$i+2$
三角形扇形	$i+2$
单独的三角形	$3i$
相邻的四边形	$2i+2$
单独的四边形	$4i$

平面明暗模式和光滑明暗模式由 glShadeModel 指定，*mode* 变量分别为 GL_FLAT 和 GL_SMOOTH。

下列函数用来检索与 glShadeModel 相关的信息：

glGet，变量为 GL_SHADE_MODEL。

错误

如果 *mode* 的数值是 GL_FLAT 或 GL_SMOOTH 之外的任何值，则会产生 GL_INVALID_ENUM 错误。

如果在 glBegin 和对应的 glEnd 语句之间调用 glShadeModel 函数，则会产生 GL_INVALID_OPERATION 错误。

参阅

glBegin，glEnd，glColor，glLight，glLightModel

1.2 glColor —— 设置当前颜色

glColor3b，glColor3d，glColor3f，glColor3i，glColor3s，glColor3ub，glColor3ui，glColor3us，glColor4b，glColor4d，glColor4f，glColor4i，glColor4s，glColor4ub，glColor4ui，glColor4us，glColor3bv，glColor3dv，glColor3fv，glColor3iv，glColor3sv，glColor3ubv，glColor3uiv，glColor3usv，glColor4bv，glColor4dv，glColor4fv，glColor4iv，glColor4sv，glColor4ubv，glColor4uiv，glColor4usv

C 语言描述

```
void glColor3b(GLbyte red,
               GLbyte green,
               GLbyte blue)
void glColor3d(GLdouble red,
               GLdouble green,
               GLdouble blue )
void glColor3f(GLfloat red,
               GLfloat green,
               GLfloat blue)
void glColor3i(GLint red,
               GLint green,
               GLint blue)
void glColor3s(GLshort red,
               GLshort green,
               GLshort blue)
void glColor3ub(GLubyte red,
```

```
                GLubyte green,
                GLubyte blue)
void glColor3ui(GLuint red,
                GLuint green,
                GLuint blue)
void glColor3us(GLushort red,
                GLushort green,
                GLushort blue)
void glColor4b(GLbyte red,
                GLbyte green,
                GLbyte blue,
                GLbyte alpha )
void glColor4d(GLdouble red,
                GLdouble green,
                GLdouble blue,
                GLdouble alpha)
void glColor4f(GLfloat red,
                GLfloat green,
                GLfloat blue,
                GLfloat alpha)
void glColor4i(GLint red,
                GLint green,
                GLint blue,
                GLint alpha)
void glColor4s(GLshort red,
                GLshort green,
                GLshort blue,
                GLshort alpha)
void glColor4ub(GLubyte red,
                GLubyte green,
                GLubyte blue,
                GLubyte alpha)
void glColor4ui(GLuint red,
                GLuint green,
                GLuint blue,
                GLuint alpha)
void glColor4us(GLushort red,
                GLushort green,
                GLushort blue,
```

GLushort *alpha*)

参数

red, *green*, *blue* 指定当前新的红、绿、蓝颜色值。

alpha 指定当前新的 alpha 颜色值。只有在 glColor4 函数带 4 个变量时才指定此参数。

C 语言描述

void glColor3bv(const GLbyte * *v*)
void glColor3dv(const GLdouble * *v*)
void glColor3fv(const GLfloat * *v*)
void glColor3iv(const GLint * *v*)
void glColor3sv(const GLshort * *v*)
void glColor3ubv(const GLubyte * *v*)
void glColor3uiv(const GLuint * *v*)
void glColor3usv(const GLushort * *v*)
void glColor4bv(const GLbyte * *v*)
void glColor4dv(const GLdouble * *v*)
void glColor4fv(const GLfloat * *v*)
void glColor4iv(const GLint * *v*)
void glColor4sv(const GLshort * *v*)
void glColor4ubv(const GLubyte * *v*)
void glColor4uiv(const GLuint * *v*)
void glColor4usv(const GLushort * *v*)

参数

v 指定一个指向包含红、绿、蓝和 alpha 值的数组指针。

说明

OpenGL 既存储当前的单个颜色索引值，又存储当前的 4 个 RGBA 颜色值。glColor 函数设置新的 RGBA 颜色值，这个函数有两种主要的形式：glColor3 和 glColor4。glColor3 形式明确指定新的红、绿、蓝值，并设置当前的 alpha 值为 1.0。glColor4 形式指定所有的 4 个分量。

glColor3b、glColor4b、glColor3s、glColor4s、glColor3i 和 glColor4i 分别用 3 个或 4 个无符号字节、短整型或长整型数值做为它们的变量，如果在这些命令名称后再加上字符 v 时，那么这些颜色命令中的变量则是指向这些数值的数组指针。

当前的颜色值以浮点形式进行存储，不指定尾数和指数。如果指定的是无符号整数颜色分量，则这些值被线性映射为浮点值，最大的值映射为 1.0(全浓度)，零映射为 0.0(零浓度)。当指定的是有符号整数颜色分量时，这些值也被线性映射为浮点值，最大的正

值映射为 1.0,最小的负值映射为 -1.0,浮点值则被直接映射。

在更改当前颜色之前,浮点值及有符号的整数值均不被箝位在[0,1]之间,然而,颜色分量在进行插值或写入到颜色缓存之前都被箝位在此范围之间。

下列函数可以用来检索与 glColor 相关的信息:

glGet,变量为 GL_CURRENT_COLOR。

glGet,变量为 GL_RGBA_MODE。

注意

可以在任何时候更改当前颜色,而且,glColor 可以在 glBegin 及对应的 glEnd 语句之间调用。

参阅

glBegin,glEnd,glShadeModel,glGet,glIndex

1.3 glColorPointer —— 定义颜色数组

C 语言描述

```
void glColorPointer(GLint size,
                    GLenum type,
                    GLsizei stride,
                    GLsizei count,
                    const GLvoid * pointer)
```

参数

size 每个颜色的分量数目,该值必须为 3 或 4。

type 在颜色数组中每个颜色分量的数据类型,可以选择的数据类型为:GL_BYTE、GL_UNSIGNED_BYTE、GL_SHORT、GL_UNSIGNED_SHORT、GL_INT、GL_UNSIGNED_INT、GL_FLOAT 或 GL_DOUBLE。

stride 相邻两个颜色的字节偏移量。当 *stride* 为零时,颜色值在数组中是一个接一个排列的。

count 静态颜色的数目,从第一个颜色开始计数。

pointer 指向颜色数组中第一个颜色元素中第一个分量的指针。

说明

glColorPointer 函数指定了绘制中使用的颜色分量数组的位置和数据格式,*stride* 参数指定了从一个颜色到下一个颜色之间的字节偏移量,在有些实现中,在单个数组中存储顶点属性可能比在几个数组中存储顶点属性效率更高。*count* 参数从第一个颜色数组元