



# 计算机图形学

---

# Open GL

# 主要内容

---

- 什么是Open GL?
- Open GL的发展历史
- Open GL的基本特点
- Open GL的工作流程
- Open GL库函数与运行方式
- OpenGL程序结构

# 什么是Open GL ?

---

- **Open Graphics Library**开放式图形库
- 一个三维的计算机图形和模型库
- 标准的**3D**图形接口
- 官方网站: [www.opengl.org](http://www.opengl.org)



# Open GL 的发展历史

---

- 人们对三维图形技术的研究已经经历了一个很长的历程，而且涌现了许多三维图形开发工具，其中**SGI**公司推出的为其图形工作站开发的**IRIS GL**三维图形库表现尤为突出，它易于使用且功能强大。
- 但是移植性不好，**SGI**公司便在**IRIS GL**的基础上开发了**Open GL**。
- 现在**Open GL**被认为是高性能图形和交互式场景处理的标准。

# Open GL 的发展历史

- **SGL (Silicon Graphics, Inc.)** 公司成立于**1982**年，是一个生产高性能计算机系统的跨国公司，总部设在美国加州旧金山硅谷。
- **SGL**公司在业界率先集成了**RISC**技术、均衡多重处理技术、数字化媒体技术、计算机图形技术、**UMA**及**CCNUMA**体系结构等计算机领域的核心科技，形成了自己的独特风格，开创了视算科技及信息处理的新方向。
- **SGL**公司是美国**Fortune**杂志所列美国最大**500**家公司/生产企业之一，年产值超过**40**亿美元。
- <http://www.sgi.com>

# Open GL 的发展历史

---

- 1992年，SGI公司发布了Open GL 1.0版。
- 1995年，Open GL ARB 发布了1.1版。
- 2014年，Open GL 4.5版
  - 标准的主要制订者并非原来的SGL，而是逐渐在ARB中占据主动地位的3D labs。
  - ARB成员以投票方式产生标准，并制成规范文档公布，各软硬件厂商据此开发自己系统上的实现。
  - 只有通过了ARB规范全部测试的实现才能称为Open GL。

# Open GL 的发展历史

---

- 与微软的竞争

- Open GL被设计成独立于硬件，独立于窗口系统，在运行各种操作系统的各种计算机上都可用，并能在网络环境下以C/S模式工作，是专业图形处理、科学计算等高端应用领域的标准图形库。
- 它在低端应用上的主要竞争对手是MS-Direct3D，该图形库是以COM接口形式提供的，所以极为复杂，稳定性差；另外微软公司拥有该库版权，目前只在Windows平台上可用。

# Open GL 的发展历史

- 与微软的竞争

- Direct3D的优势在速度上，但现在低价显卡都能提供很好的Open GL硬件加速，所以做3D图形开发使用Direct3D已没有特别的必要。
- 在专业图形处理特别是高端应用方面目前还没有出现以Direct3D技术为基础的例子，而游戏等低端应用也有转向Open GL的趋势。

- 微软对Open GL的支持

- Windows NT始于3.51
- Windows9x始于Win95 OEM Service Release 2
- 更高版本都支持



# Open GL 的发展历史

- Windows下常用的Open GL库有两种：MS实现的和SGI实现的
  - **MS-Open GL**调用会自动检测是否存在显示卡制造商提供的**ICD(Installable Client Device)**驱动程序，有则调用**ICD**中的例程，否则使用**CPU**进行计算，所以能**利用显示卡的OpenGL加速能力**。对开发者来说使用方法并没有区别，只是有**ICD**驱动时更快些。
  - **SGI**的版本是**纯软件实现**，不能利用硬件加速并且**SGI**已经在**1999**年宣布停止支持，但这套图形库便于调试程序，仍有不少开发者使用。

# Open GL 的发展历史

- 硬件性能的提高和Open GL本身的不断发展，使得Open GL不再只属于专用图形工作站。
- 如今，开发人员可以在各种硬件平台利用Open GL进行图形软件开发。
  - Open GL可以运行在当前各种流行操作系统之上，如Win系列、 Mac OS、 Unix、 Linux等。
  - 各种流行的编程语言都可以调用Open GL中的库函数，如C、 Fortran、 Pascal 、 C++、 Java 。
  - Open GL完全独立于各种网络协议和网络拓扑结构。

# Open GL 的发展历史

- **Microsoft、SGI、ATT、IBM、SUN、HP**等几家在计算机市场占主导地位的大公司都采用了**Open GL**图形标准。
- 由于**Microsoft**在**Wins**中提供**Open GL**标准，使得**Open GL**在微机中得到了广泛应用。
- 尤其是在**Open GL**三维图形加速卡和微机图形工作站推出后，人们可以在微机上实现**CAD**设计、仿真模拟、三维游戏等，从而使应用**Open GL**及其应用软件来创建三维图形变得更有机会、更为方便。

# Open GL 的基本特点

---

- OpenGL应用领域十分宽广
  - 军事
  - 电视广播
  - CAD/CAM/CAE
  - 娱乐
  - 艺术造型
  - 医疗影像
  - 虚拟现实

# Open GL 的基本特点

- Open GL具有以下特点

- 工业标准

- OARB（OpenGL Architecture Review Board）联合会领导Open GL技术规范的发展，**Open GL**有广泛的支持，它是业界唯一的真正开放的、跨平台的图形标准。

- 可靠度高

- 用OpenGL技术开发的图形软件与硬件无关，只要硬件支持Open GL API标准就行：**Open GL**应用可以运行在支持**Open GL API**标准的任何硬件上。

# Open GL 的基本特点

---

## — 可扩展性

- Open GL是低级的图形API，具有充分的可扩展性。许多Open GL开发商在Open GL核心技术规范的基础上，**增强了大量图形绘制功能**，从而使Open GL能紧跟最新硬件发展和计算机图形绘制算法的发展。
- 对于硬件特性的升级可以体现在Open GL扩展机制以及Open GL API中，一个**成功的Open GL扩展**会被融入在未来的Open GL版本之中。



# Open GL 的基本特点

---

## — 可伸缩性

- 基于**Open GL API**的图形应用程序可以运行在许多系统上，包括各种用户电子设备、PC、工作站以及超级计算机。

## — 容易使用

- **Open GL**可以利用已有的其它格式的数据源进行三维物体建模，大大提高了软件开发效率；
- 采用**Open GL**技术，开发人员几乎可以不用了解硬件的相关细节，便可以利用**Open GL**开发照片质量的图形应用程序。



# Open GL 的基本特点

---

## — 灵活性

- 尽管**Open GL**有一套独特的图形处理标准，但各平台开发商可以自由地开发适合于各自系统的**Open GL**执行实例。
- 在这些实例中，**Open GL**功能可由特定的硬件实现，也可用纯软件例程实现，或者以软硬件结合的方式实现。

## — 详细文档

- **Open GL**标准严格、规范、详细。
  - 《**OpenGL**入门教程》
  - 《**OpenGL**编程指南》





# Open GL 的基本特点

- 客观世界和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。
- Open GL提供了以下基本操作
  - 绘制物体
    - 现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。
    - Open GL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。

# Open GL 的基本特点

---

- Open GL提供了以下基本操作
  - 坐标变换
    - 可以说，无论多复杂的图形都是由基本图元组成并经过一系列变换来实现的。
    - Open GL提供了一系列基本的变换
      - 视图变换
      - 模型变换
      - 投影变换
      - 视口变换

# Open GL 的基本特点

- Open GL提供了以下基本操作
  - 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。
    - 其光照模型是**整体光照模型**。
  - 着色
    - Open GL提供了两种物体着色模式
      - **RGBA**颜色模式
      - 颜色索引模式。

# Open GL 的基本特点

---

- Open GL提供了以下基本操作
  - 反走样(antialiasing)
    - Open GL提供了点、线、多边形的反走样技术：通过改变点、线、多边形周围像素的颜色，使其平滑化，达到消除锯齿效果
  - 颜色融合(blending)
    - 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，可采用融合技术。

# Open GL 的基本特点

- Open GL提供了以下基本操作
  - 雾化(fog)
    - 正如自然界中存在烟雾一样，Open GL提供了“fog”的基本操作来达到对场景进行雾化的效果。
    - 根据离视点的距离，计算对象的颜色值来建立深度的视觉。其效果与现实生活一样，当物体较远时背景较淡。
  - 操作位图、字体和图像

# Open GL 的基本特点

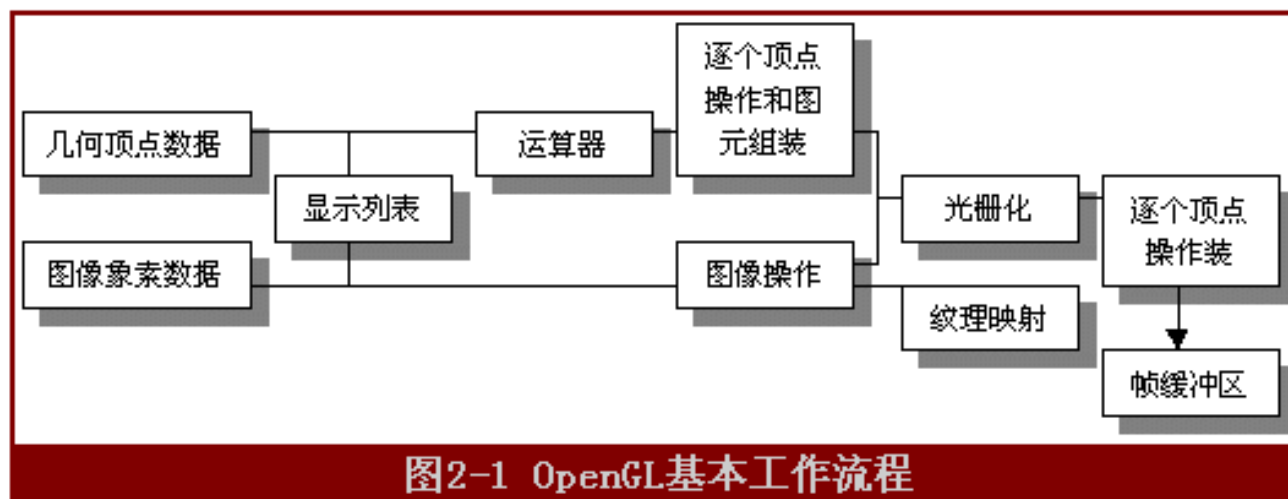
- Open GL提供了以下基本操作
  - 纹理映射(texture mapping)
    - 在计算机图形学中，把包含颜色、alpha值、亮度等数据的矩形数组称为纹理。
    - 纹理映射可以理解为将纹理粘贴在所绘制的三维模型表面，以使三维图形显得更生动。
  - 动画
    - 出色的动画效果是Open GL的一大特色；
    - 提供了双缓存区技术来实现动画绘制。

# Open GL 的基本特点

- Open GL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。
- 有许多优秀的三维图形软件可以较方便地建立物体模型，但又难以对建立的模型进行控制，若把这些模型转化为Open GL程序，则可随心所欲地控制这些模型来制作三维动画，实现仿真数据的可视化和虚拟现实。

# Open GL 的工作流程

- Open GL 工作流程就是一个从定义几何要素到把像素段写入帧缓冲区的过程。





# Open GL 的工作流程

- 在屏幕上显示图像的主要步骤

- 构造几何要素（点、线、多边形、图像、位图），创建对象的数学模型；
- 在三维空间中放置对象，选择合适的场景观察点；
- 计算对象的颜色，颜色可能直接定义，或由光照条件及纹理间接给出；
- 光栅化—把对象的数学模型和颜色信息转换到屏幕的像素。

# Open GL 的工作流程

- OpenGL的绘制过程多种多样，内容非常丰富，主要提供以下几种对三维物体的绘制方式
  - 线框绘制方式(Wire frame): 绘制三维物体的网格轮廓线。
  - 深度优先线框绘制方式(Depth cued): 采用线框方式绘图，使远处的物体比近处的物体暗一些，以模拟人眼看物体的效果。
  - 反走样线框绘制方式(Antialiased): 采用线框方式绘图，绘制时采用反走样技术，以减少图形线条的参差不齐。
  - 平面明暗处理方式(Flat shading): 对模型的平面单元按光照进行着色，但不进行光滑处理。

# Open GL 的工作流程

- 光滑明暗处理方式(Smooth shading): 对模型按光照绘制的过程进行光滑处理, 该方式更接近于现实。
- 加阴影和纹理的方式(Shadow and Texture): 在模型表面贴上纹理甚至加上光照阴影效果, 使三维场景像照片一样逼真。
- 运动模糊绘制方式(Motion blurred): 模拟物体运动时人眼观察所觉察到的动感模糊现象。
- 大气环境效果(Atmosphere effects): 在三维场景中加入雾等大气环境效果, 使人有身临其境之感。
- 深度域效果(Depth of effects): 类似于照相机镜头效果, 模拟在聚焦点处清晰。

# Open GL库函数与运行方式

- Open GL开发组件

- Windows下的Open GL组件有两种，SGI提供的和Microsoft提供的。
- 大体上没有什么区别，都是由三大部分组成：
  - 函数的说明文件：gl.h、glu.h、glut.h和glaux.h
  - 静态链接库文件：glu32.lib、glut32.lib、glaux.lib和opengl32.lib
  - 动态链接库文件：glu.dll、glu32.dll、glut.dll、glut32.dll和opengl32.dll

# Open GL库函数与运行方式

- Open GL的库函数大致可以分为六类
  - 核心库，实用库，辅助库，工具库，**Windows(X窗口)专用库， Win32 API函数库**
  - Open GL核心库（基本）
    - 包含有**115**个函数，函数名的前缀为**gl**。  
**glVertex2f( , , )**
    - 用于**常规的、核心的图形处理**。由于许多函数可以接收不同数据类型的参数，因此派生出来的函数原形多达**300**多个。

# Open GL库函数与运行方式

## – Open GL实用库

- 包含有43个函数，函数名的前缀为glu。

### gluLookAt()

- 通过调用核心库的函数，为开发者提供相对简单的用法，实现一些较为复杂的操作。如：坐标变换、纹理映射、绘制椭球、茶壶等简单多边形。
- Open GL中的核心库和实用库可以在所有的Open GL平台上运行。

# Open GL库函数与运行方式

## – Open GL工具库

- 包含大约30多个函数，函数名前缀为**glut**。  
**glutInitWindowSize(100,100)**
- 提供基于窗口的工具，如：多窗口绘制、空消息和定时器，以及一些绘制较复杂物体的函数。
- 由于**glut**中的窗口管理函数是不依赖于运行环境的，因此Open GL中的工具库可以在所有的Open GL平台上运行。

# Open GL库函数与运行方式

## – Open GL辅助库

- 包含有31个函数，函数名前缀为**aux**。  
**auxInitDisplayMode()**
- 提供窗口管理、输入输出处理以及绘制一些简单三维物体。
- **Open GL**辅助库可使编程简单明了，提供初学者入门的函数。
- **Open GL**中的辅助库不能在所有的**Open GL**平台上运行。



# Open GL库函数与运行方式

## – Windows专用库 (X窗口专用库)

- 包含有16个函数，函数名前缀为wgl  
wglCreateContext()/glXCreateContext()
- wgl函数将OpenGL与Windows 视窗系统联接起来管理绘图描述表，显示列表，执行函数和文字位图。
- Windows专用库只能用于Windows 环境中。

## – Win32 API函数库

- 包含有6个函数，函数名无专用前缀。
- 处理像素存储格式和双帧缓存。这6个函数将替换Windows GDI中原有的同样的函数。
- Win32API函数库只能用于Windows 环境中。

# Open GL库函数与运行方式

- Open GL程序运行方式（主要有三种）
  - Open GL硬件加速方式
    - 一些显示芯片如3Dlabs公司的GLiNT进行了优化，Open GL的大部分功能均可由硬件实现，仅有少量功能由操作系统来完成。
    - 极大地提高了图形显示的性能，并且能够获得工作站级的图形效果，但是这样的图形硬件价格十分昂贵，非一般用户所能承担。

# Open GL库函数与运行方式

- Open GL程序运行方式（主要有三种）
  - 三维图形加速模式
    - 一些中低档的图形芯片具备一定的三维加速功能，由硬件来完成一些较为复杂的图形操作：一些重要的Open GL操作，例如Z缓存等就能够直接由显卡硬件来完成，显卡所不能支持的图形功能，则通过软件模拟的方式在操作系统中进行模拟。
    - 特点：显示速度无法与硬件加速方法相比，但与采用纯软件模拟方式相比，速度要快得多。

# Open GL库函数与运行方式

- Open GL程序运行方式（主要有三种）
  - 纯软件模式
    - 对于不具备三维加速功能的显示卡，要想运行Open GL，只能采用纯软件模拟方式：由于所有复杂的Open GL图形功能均通过主机来模拟，所以速度将会受到很大的影响。
    - 正是由于有了软件模拟方式，才使得更多的用户能够领略Open GL的强大功能，并能在硬件性能较差的机器上对OpenGL进行开发。
- 采用Open GL技术，大大降低了开发高质量图形软件对软、硬件的依赖程度。

# Open GL库函数与运行方式

---

- **OpenGL对硬件的要求**
  - CPU时钟频率：90MHz以上
  - 内存：16/32/64MB以上
  - 硬盘：512MB以上
- **OpenGL对软件环境的最低要求**
  - 操作系统：Win NT4.0以上或Win 95以上
  - OpenGL库：Visual Studio 4.0以上版本已包含该库。



# Open GL程序结构

---


- 基本语法
- 状态机制
- 程序的基本结构
- 程序举例



# Open GL基本语法

- Open GL的函数虽然多达几百个，但由于有一套非常规范的语法规则，应用起来很方便。
  - 前缀代表Open GL命令的函数类型
    - 核心库：函数以gl开头，如glColor3f()。
    - 实用库：函数以glu开头，  
如gluCreatWindow()。
    - 工具库：函数以glut开头，如glutInit()。
    - 辅助库：函数以aux开头，  
如auxInitDisplayMode()。
    - Windows专用函数库：函数以wgl开头，  
如wglCreateContext()。

# Open GL基本语法

- 后缀表明Open GL命令的参数个数和数据类型。
- 有时会在函数后缀后加个“v”，如 **glVertex3fv**等,这表示该参量是一个矢量或矩阵的指针。
- 掐头去尾，中间的关键词就是Open GL函数的功能。例如
  - **glVertex2i(100,200)**表明是Open GL的基本函数（gl-），是绘点的函数（-Vertex-），是两个整型参数（-2i）。 



# Open GL基本语法

- 基本符号常量：以**GL\_**为开头，均用大写字母，并用下划线与关键词分开，如**GL\_COLOR\_BUFFER\_BIT**。
- 实用符号常量：以**GLU\_**为开头，均用大写字母，并用下划线与关键词分开，如**GLU\_RGB**。
- 数据类型：Open GL定义的数据类型以**GL**开头，例如（**gl.h**文件中可见）
  - **typedef int GLint;**
  - **typedef unsigned int GLuint;**
  - **typedef float GLfloat;**

# Open GL状态机制

- Open GL的工作方式：状态机制
  - Open GL 是一组绘图命令的API集合,各种执行命令是在某种状态或模式下进行的,一直有效地保持到改变它们之后。
  - 例如，当前颜色就是一个状态变量。
    - glColor()
  - 许多状态变量可以通过函数glEnable(), glDisable()来设置为有效或无效
    - 设置光照： glEnable(GL\_LIGHTING)

# Open GL程序的基本结构

- 基本框架分3部分
  - Part 1,初始化部分
    - 设置颜色模式、缓存模式、深度检测、光照处理
  - Part 2,设置观察坐标系下的取景模式和取景框位置及大小:
    - `glViewport()`设置在屏幕上的窗口大小
    - `glOrtho()`设置投影方式为平行投影
    - `gluPerspective()`设置投影方式为透视投影
  - Part 3, 主要部分: 使用库函数构造几何物体对象的数学描述, 包括点线面的位置和拓扑关系、几何变换、光照处理等。

# Open GL程序演示

---

- 学习OpenGL前的准备工作
  - 选择一个编译环境
    - 设置选择Visual Studio 2019作为学习OpenGL的环境
  - 安装GLUT工具包：
    - GLUT会给我们的学习带来一定的方便
    - <http://www.opengl.org/resources/libraries/glut/glutdlls37beta.zip>

# Open GL程序演示

---

## – 安装GLUT工具包

- 压缩包解压后得到5个文件
- 把glut.h到\Community\VC\Tools\MSVC\14.23.28105\include\GL。
- 把glut.lib和glut32.lib放到静态函数库所在文件夹 (\14.23.28105\lib\x86)
- 把glut.dll和glut32.dll放到操作系统目录下system32(SysWOW64)文件夹内

[https://blog.csdn.net/c\\_php\\_/article/details/102499187](https://blog.csdn.net/c_php_/article/details/102499187)

# Open GL程序演示

- Windows环境下OpenGL应用程序演示
- 该程序的作用是在一个黑色的窗口中央画一个白色的矩形

```
int main(int argc, char *argv[])
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE);
    glutInitWindowPosition(100, 100);
    glutInitWindowSize(400, 400);
    glutCreateWindow("第一个OpenGL程序");
    glutDisplayFunc(&myDisplay);
    glutMainLoop();
    return 0;
}
```



# Open GL程序演示

- 程序解析

- 需要包含头文件`#include <GL/glut.h>`, 这是GLUT的头文件
  - GLUT的头文件中已自动包含`<GL/gl.h>`和`<GL/glu.h>`
- `main(int argc, char *argv[])`, 带命令行参数的main函数: 除return外, 函数中的各语句都是GLUT工具包所提供的函数。

# Open GL程序演示

## ● 程序解析

- **glutInit:** 在其它的GLUT命令使用之前，对GLUT进行初始化
- **glutInitDisplayMode:** 设置显示方式，其中GLUT\_RGB表示使用RGB颜色；GLUT\_SINGLE表示使用单缓冲，与之对应的还有GLUT\_DOUBLE
- **glutInitWindowPosition:** 设置窗口在屏幕中的位置
- **glutInitWindowSize:** 设置窗口的大小
- **glutCreateWindow:** 根据前面设置的信息创建窗口。
- **glutDisplayFunc:** 设置一个函数，当需要进行画图时，这个函数就会被调用
- **glutMainLoop,** 进行一个消息循环，显示窗口直到关闭窗口



# Open GL程序演示

- 在`glutDisplayFunc`函数中设置了“当需要画图时，请调用`myDisplay`函数”。

```
#include <GL/glut.h>
void myDisplay(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glRectf(-0.5f, -0.5f, 0.5f, 0.5f);
    glFlush();
}
```

- `myDisplay`中的命令都是OpenGL的标准函数
  - `glClear(GL_COLOR_BUFFER_BIT)`: 清除颜色
  - `glRectf`, 画一个矩形。
  - `glFlush`, 保证前面的OpenGL命令立即执行。

