计算机图形学
Computer Graphics

陈中贵
chenzhonggui@xmu.edu.cn
http://graphics.xmu.edu.cn/~zgchen

Graphics@XMU

*课件来源于中科大黄章进副教授讲义

第二章

OpenGL编程

第一节

OpenGL简介

主要内容

- 图形 API的发展
- OpenGL的体系结构
  - 状态机 state machine
- 函数
  - 类型
  - 格式
- 安装编译

GKS(1980s)

- IFIPS (1973) 组织了两个委员会建立图形API的标准
  - GKS(Graphical Kernel System)
    - 二维，同时包含很好的工作站模型
  - Core: 同时应用于二维和三维
  - GKS成为ISO标准，稍后成为ANSI标准(1980s)
- GKS很难推广到三维（GKS-3D)
  - 远远落后于硬件的发展

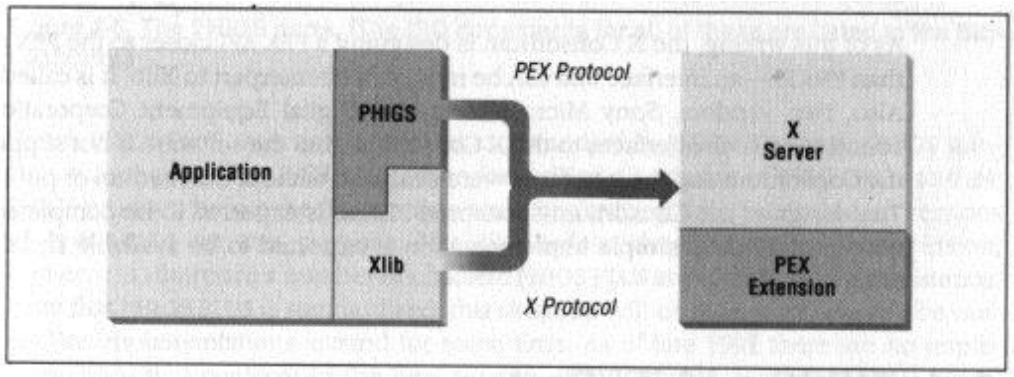
IFIPS: International Federation of Information Processing Societies

PHIGS（1990s）

- 程序员层次交互式图形系统 (Programmers Hierarchical Interactive Graphics System)
  - 来自于CAD团体的3D图形API
  - 保留模式：绘制前在数据库里保存场景的层次结构
  - 基本几何图元和网格模型
  - 不支持光照
- PHIGS+
  - NURBS曲面
  - 支持光照，但不支持纹理
- PHIGS和PHIGS+都是ANSI标准和ISO标准

PHIGS与X

- X Window系统（也常称为X11或X）R7.6
  - 1984年 DEC/MIT的雅典娜计划Data Equipment Company 数据设备公司
  - 图形用户界面(GUI)环境底层框架，硬件无关
  - 客户端-服务器(client-server)架构模型
  - GNONE、KDE
- PEX（ PHIGS Extension to X ）把两者组合
  - 不易应用



IRIS GL

- Silicon Graphics(SGI: 1981-2006)
  - Stanford的Jim Clark博士带领7个研究生创办
  - Geometry Engine（几何引擎）：硬件(VLSI)实现几何流水线，极大改良了图形工作站
- IRIS GL(Integrated Raster Imaging System Graphics Library)
  - 立即模式绘制
  - 可非常简单地设计出三维交互图形应用程序

OpenGL

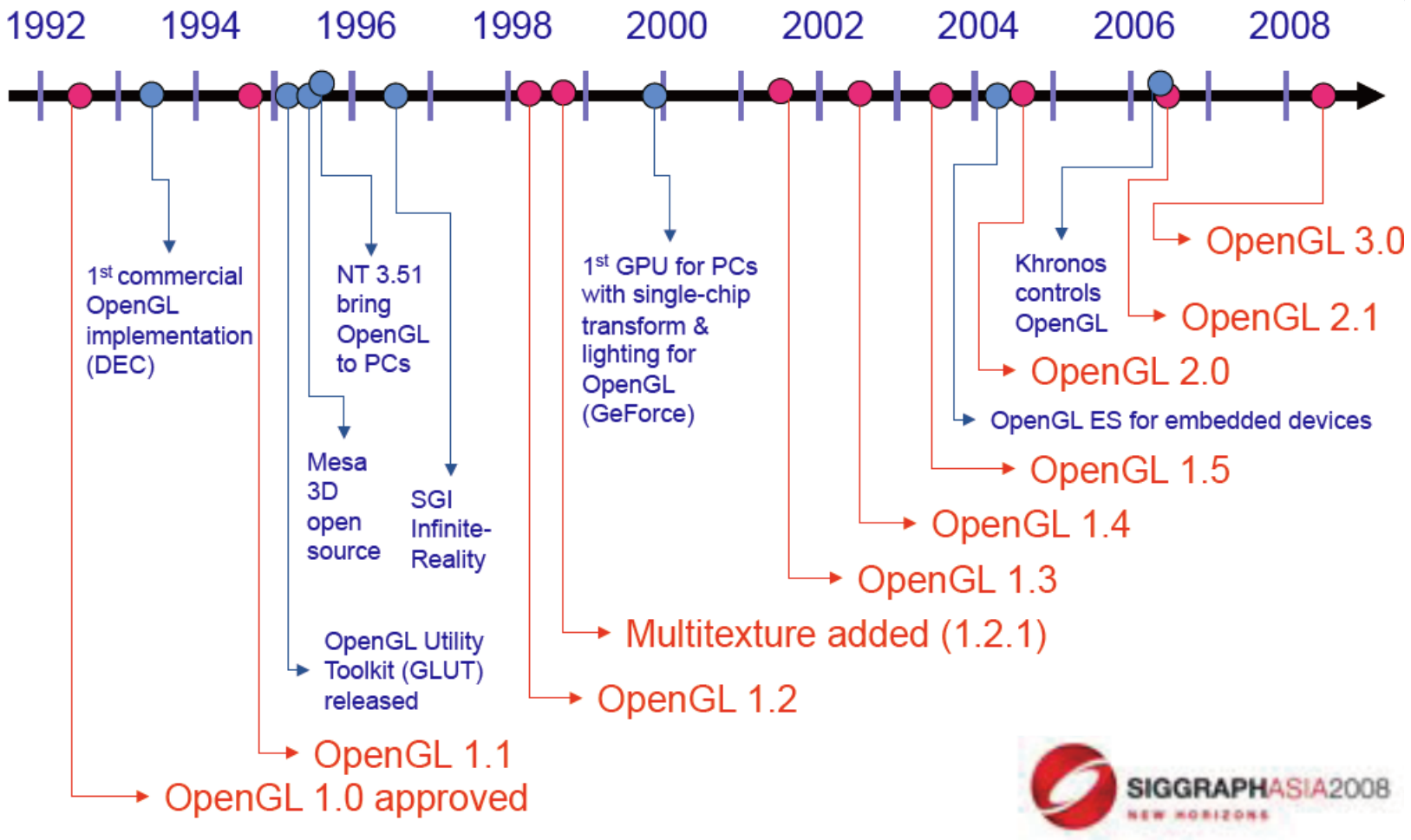
- 1992年 SGI领导的OpenGL Architectural Review Board(OpenGL ARB)发布1.0版
  - 平台无关的API:
  - 易于使用
  - 与硬件非常贴近，从而可以充分发挥其性能
  - 着重在于渲染 (rendering)
  - 没有提供窗口和输入接口，从而避免依赖于具体的窗口系统

OpenGL的发展

- 早期是由 ARB掌控其发展
  - 成员包括SGI, Microsoft, Nvidia, HP, 3DLabs, IBM, ……
  - 相对稳定 2.1(2006.7)/3.3(2010.3)/4.5(2014.8)
  - 发展反映了新的硬件能力
    - 3D纹理映射和纹理对象(1.2, 1998)
    - 顶点着色器、片段着色器(2.0, 2004)
    - 几何着色器(3.2, 2009)
    - Tessellator(4.1, 2011)
    - Compute Shader(4.3, 2012)
- 通过扩展支持平台相关的特性
- 2006年，ARB被Khronos工作组取代

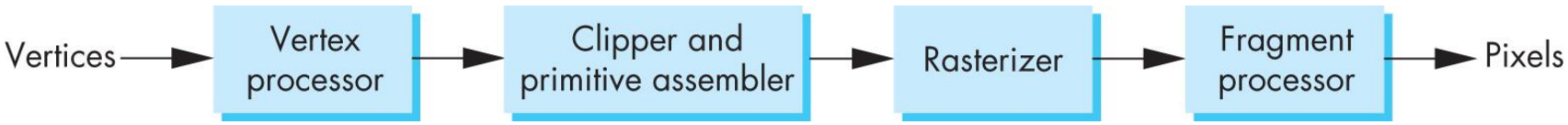
http://en.wikipedia.org/wiki/OpenGL

OpenGL的发展



现代OpenGL

- 利用GPU而不是CPU来获得性能提升
- 通过称为着色器(shader）的程序来控制
- 应用程序的工作就是把数据发送到GPU
- GPU执行所有的渲染任务



OpenGL 3.1

- 完全基于shader
  - 没有缺省的shader
  - 每个应用程序必须提供vertex shader和fragment shader
- 取消立即模式（immediate mode）
- 减少状态变量
- 大部分OpenGL 2.5版本增加的功能废弃
- 非向后兼容

其他版本

- OpenGL ES
  - 嵌入式系统
  - 1.0版：简化的OpenGL 2.1
  - 2.0版：简化的OpenGL 3.1
    - Shader based
- WebGL
  - OpenGL ES 2.0的Javascript实现
  - 被新版本的网络浏览器支持
- OpenGL 4.4 （2013）
  - 支持geometry shaders、tessellator和compute shaders

Direct3D

- DirectX: 微软开发的多媒体编程接口
- Direct3D: DirectX的3D图形API
  - 1.0: 1995
  - 2.0: Windows 95 OSR2& NT 4.0, 1996
  - 9.0c: Windows Xp SP2, 2004
  - 10.1: Windows Vista SP1, 2008
  - 11: Windows 7&Vista, 2009
  - 11.1: Windows 8, 2011
  - 11.2 : Windows 8.1, 2013

OpenGL vs Direct3D

OpenGL

- 跨平台的开放式标准API
  - 可扩展机制
- 可硬件加速的3D渲染系统
  - 底层实现（驱动）管理硬件
- 专业图形应用、科研
  - 跨平台，可移植
- 适合图形学教学

Direct3D

- Windows平台的专利API
  - 一致性好
- 3D硬件接口
  - 应用程序管理硬件资源
- 计算机游戏
  - 高性能硬件存取能力
- Direct3D 7.0能匹敌，8.0（2001）开始胜出

OpenGL库

- OpenGL核心库(OpenGL Core Library)
  - 函数名gl开头
  - Windows: opengl32.dll (WINDOWS\SYSTEM32)
    - Windows Xp支持OpenGL 1.1，Vista支持1.4
    - Direct3D的封装，需安装驱动来实现硬件加速
  - 大多数Unix/Linux系统：GL库 (libGL.a)
- OpenGL实用库(OpenGL Utility Library, GLU)
  - OpenGL的一部分，函数名以glu开头
    - Windows: glu32.dll
  - 利用OpenGL实用库提供一些功能，避免重复编写代码
  - 二次曲面、NURBS、多边形网格化等

OpenGL库

- 与窗口系统的连接
  - X Window系统：GLX
  - Windows: WGL
  - Macintosh: AGL
  - 粘合OpenGL和窗口系统

GLUT

- OpenGL实用工具库（OpenGL Utility Toolkit Library, GLUT）
  - 提供所有窗口系统的共同功能
    - 创建窗口
    - 从鼠标和键盘获取输入
    - 菜单
    - 事件驱动
- 代码可以在平台间移植，但是GLUT缺乏一些现代GUI的控件和功能
  - 无滚动条
  - 可用FLTK、SDL

http://www.opengl.org/resources/libraries/glut/

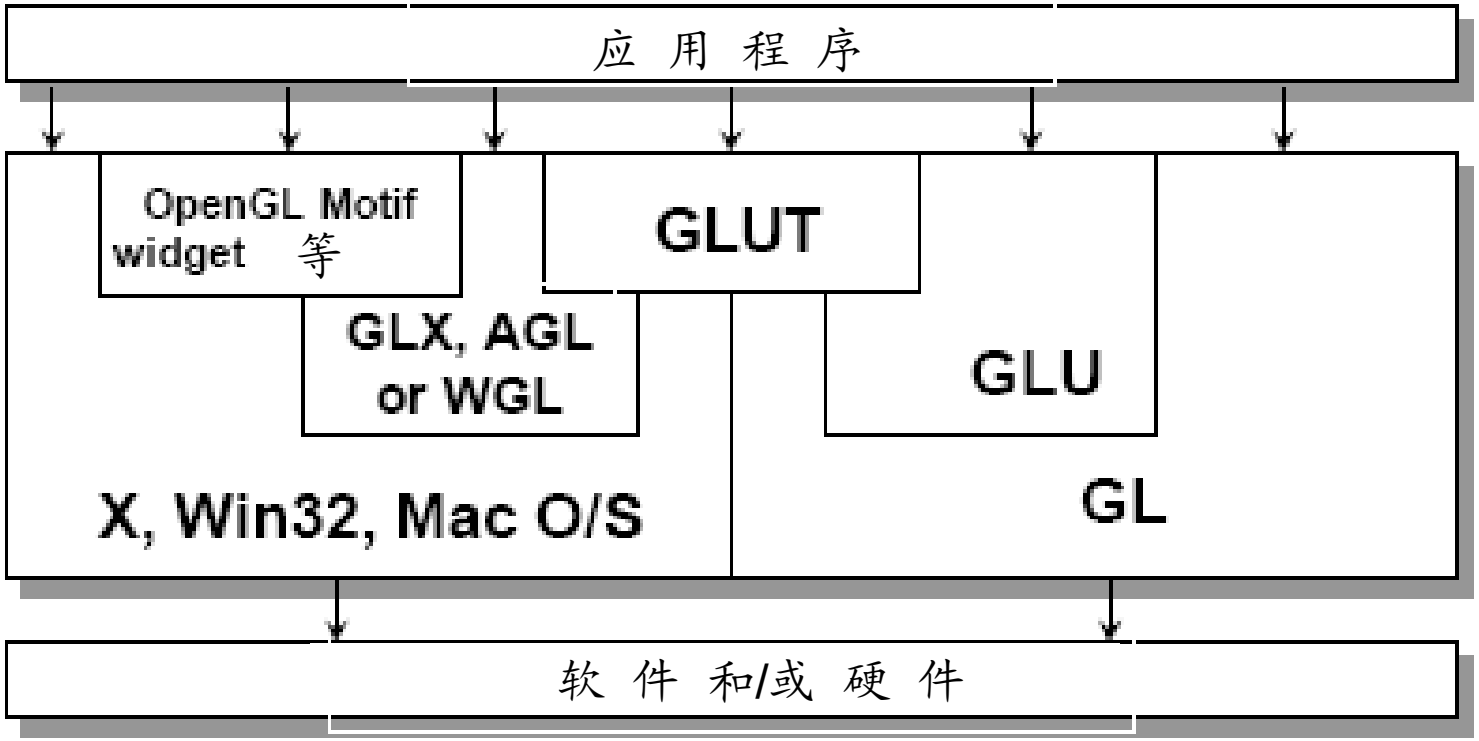
freeglut

- GLUT库已经很久没有更新
  - 可以和OpenGL 3.1一起使用
  - 有些功能不能使用，因为需要废弃的函数
- Freeglut是类似GLUT的开源扩展
  - 增加的功能
  - 上下文检查

GLEW

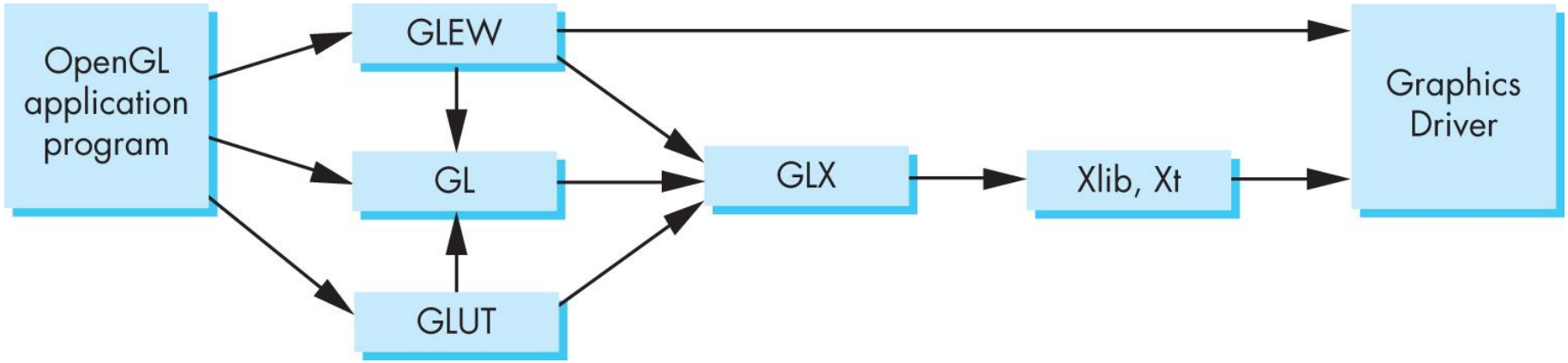
- OpenGL Extension Wrangler Library：跨平台的开源OpenGL扩展加载库
- 使得调用特定系统支持的OpenGL扩展功能更简单
- 对于windows代码来说，避免直接调用实体入口
- 应用程序只需要包含glew.h头文件，并调用glewInit()即可

软件组织

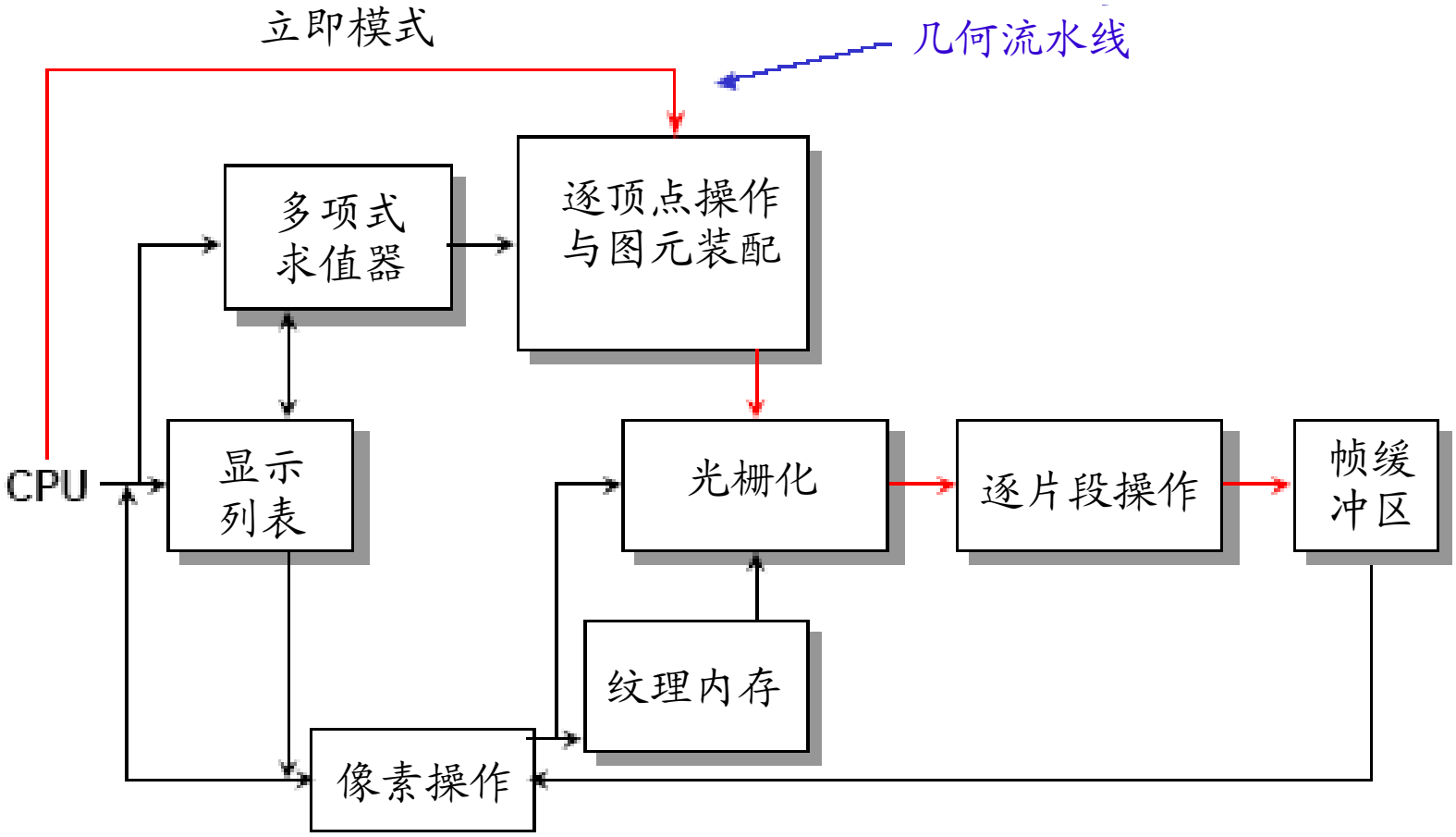

应 用 程 序
OpenGL Motif
widget 等
GLUT
GLX, AGL
or WGL
GLU
X, Win32, Mac O/S
GL
软 件 和/或 硬 件

软件组织

- X window系统下，OpenGL函数库的组织
  - GLUT会使用GLX和X库



OpenGL流水线体系



图形函数

- 图元函数(primitive) what
  - 系统可以显示的低级对象或最基本的实体，包括点、线段、多边形、像素、文本和各种曲线/曲面等
- 属性函数(attribute) how
  - 控制图元在显示器上显示的方式：线段颜色、多边形填充模式、图标题文本的字体等
- 视图函数(viewing)
  - 设置虚拟照相机的位置、朝向和镜头参数等。
- 变换函数(transformation)
  - 对对象进行诸如平移、旋转和缩放等变换操作。
- 输入函数(input) GLUT
  - 处理来自键盘、鼠标等设备的输入
- 控制函数(control) GLUT
  - 与窗口系统通信，初始化程序，处理运行时的错误等
- 查询函数(query)
  - 确定特定系统或设备的性能参数，查询相机参数、帧缓冲区等API相关的信息

OpenGL的状态

- OpenGL是一个有限状态机(state machine)的黑盒
  - 状态：持续性参数，如颜色、线型、材质属性等
  - 来自应用程序的输入改变machine的状态或者产生可见的输出
- OpenGL函数有两种类型
  - 定义图元
    - 如果图元可见，则被输出
    - 顶点如何被处理，图元的外观由状态控制
  - 改变状态
    - 属性函数
    - 视图函数
    - 变换函数
    - 在3.1以下版本，大部分状态变量由应用程序定义并发送到着色器

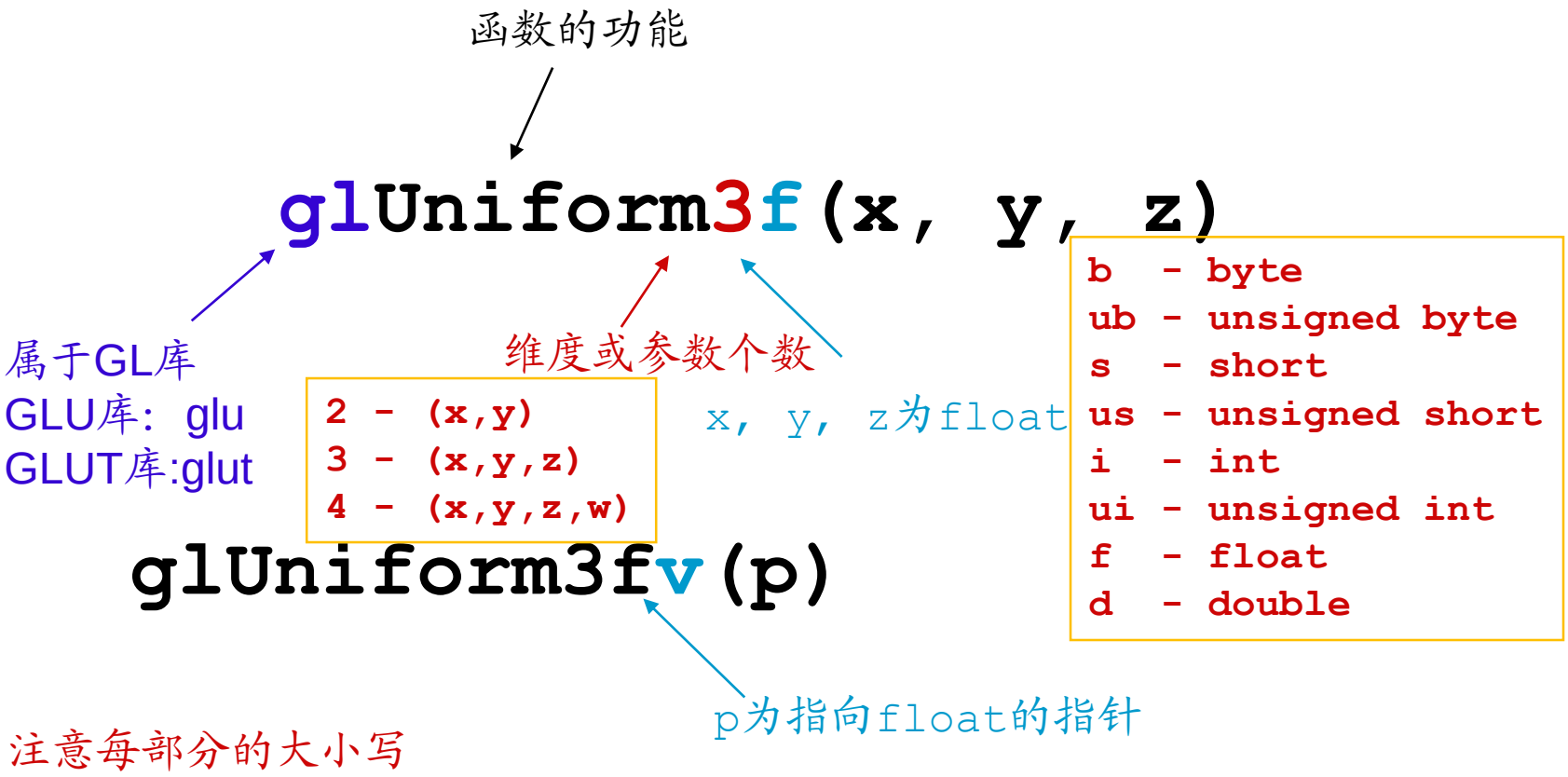
面向对象方面的缺陷

- OpenGL不是面向对象的，因此逻辑上的一个函数却对应着多个OpenGL函数:

  ```
  glUniform3f
  glUniform2i
  glUniform3dv
  ```

- 内在存储模式是相同的
- 在C++中很容易创建重载函数，但效率却成为主要问题

OpenGL函数名称的格式



OpenGL常量和数据类型

- 头文件gl.h, glu.h和glut.h中定义大量的常量
  - 例如：**glEnable(GL_DEPTH_TEST)**
  - **glClear(GL_COLOR_BUFFER_BIT)**
  - 注意：#include <GL/glut.h>自动将其他两个头文件包含到程序中
- 头文件中定义了OpenGL数据类型：Glfloat, Gldouble,…

OpenGL和GLSL

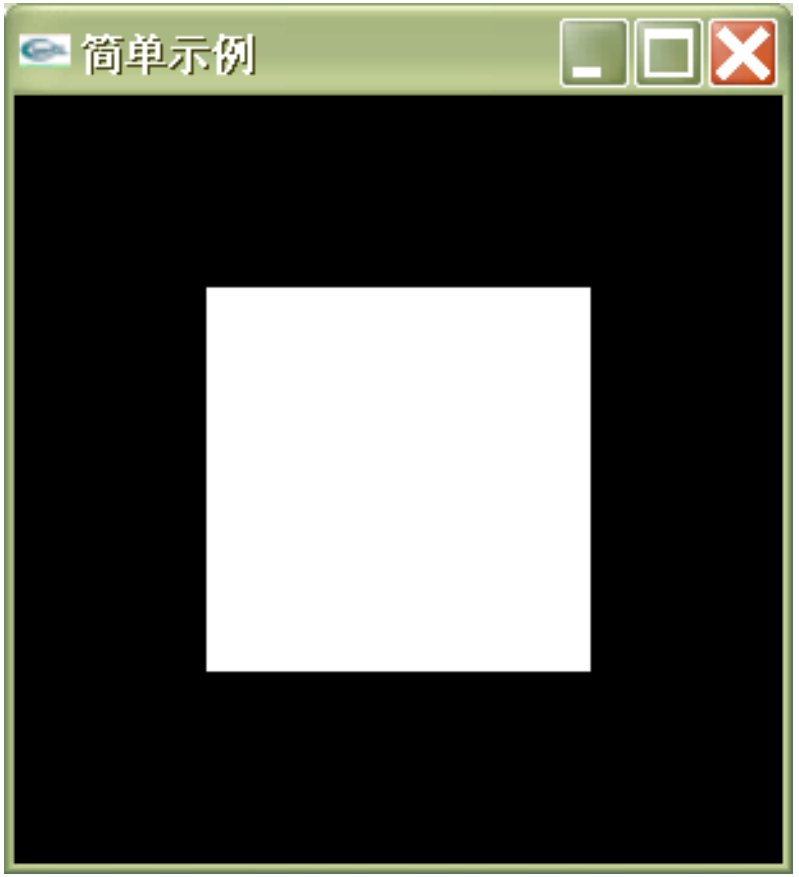
- 基于着色器的OpenGL与其说是状态机模型，不如说是数据流模型
- 大部分状态变量、属性和相关的3.1前的OpenGL函数被弃用
- 由着色器负责渲染任务
- 应用程序只需把数据传输到GPU

GLSL

- OpenGL Shading Language： OpenGL着色语言
- 类C语言
  - 2到4维的矩阵和向量类型
  - 重载的运算符
  - 类C++的构造函数
- 类似于Nvidia的Cg和Microsoft的HLSL
- 代码以源代码的形式发送到着色器
- 通过OpenGL函数编译、链接和发送信息到着色器

一个简单程序

- 在黑色背景上画一个白色矩形



simple.c

```
#include <GL/glut.h>
void display(){
        glClear(GL_COLOR_BUFFER_BIT);
        glBegin(GL_QUADS);
                glVertex2d(-0.5, -0.5);
                glVertex2d(-0.5, 0.5);
                glVertex2d(0.5, 0.5);
                glVertex2d(0.5, -0.5);
        glEnd();
        glFlush();
}
int main(int argc, char* argv[]) {
        glutInit(&argc, argv);
        glutCreateWindow("simple");
        glutDisplayFunc(display);
        glutMainLoop();
}
```

事件循环

- 程序指定了一个显示回调函数display
  - 每个GLUT程序都必须有一个显示回调函数
  - 每当OpenGL检测到显示必须被刷新时，显示回调函数就会被执行，例如窗口被首次打开时
  - main函数是以程序进入到事件循环作为结束