

Logic_视觉班课堂笔记[CC011]

- 日期: 2019年6月3日星期一
- 授课: CC老师
- 课程次数: 视觉班第11次课--共计(22次课)
- 主题: OpenGL ES 主题

课程内容

- 案例(3)--OpenGL ES GLKit 加载立体图形(正方体)
- 案例(4)--使用CoreAnimation 加载立体图形(正方体)
- GLSL 初探

课程安排

- 08:00 - 09:00 第一节课
- 09:00 - 09:10 课间休息
- 09:10 - 10:00 第二节课
- 10:00 - 10:10 课程总结
- 10:10 - 10:30 课后答疑

课后作业:

1. 请各位同学课后完成案例(3)/案例(4) 代码复习

- 要求:
 - i. 能独立完成
 - ii. 能将案例总结成思维导图

一.上节课课程回顾

- GLKit 加载图片案例回顾

GLKViewController 关于更新

- (**void**)glkViewControllerUpdate:(**GLKViewController ***)controller;

Required method for implementing GLKViewControllerDelegate.
实现glkView **ControllerDelegate**所需的方法

This update **method** variant should be used
when not subclassing **GLKViewController**.
当不对**GLKViewController**进行进行子类化时,则使用该方法实现更新

This method will **not** be called **if** the **GLKViewController object**
has been subclassed **and** implements -(**void**)update.

如果**GLKViewController** 被子类化且实现了 -(**void**)
update方法,则不会调用 - (**void**)glkViewControllerUpdate:(**GLKViewContro**
ller *)controller; 方法

二.课程笔记

GPUImage /ijkplayer /kxmovie.

shader : 普通字符串来存储! 这个没有任何问题! 缺点不便于阅读和书写.

常用后缀

- vsh/fsh
 - vertex shader/ fragment shader 着色器shader
- glsl : shader.
- 也可以是字符串

GLuint glCreateShader(GLenum **type**);
type - 创建着色器的类型, GL_VERTEX_SHADER 或者 GL_FRAGMENT_SHADER
返回值 - 是指向新着色器对象的句柄. 可以调用 glDeleteShader 删除

void glDeleteShader(GLuint shader);
shader - 要删除的着色器对象句柄

void glShaderSource(GLuint shader , GLsizei **count** ,const GLChar * con
st *string, const GLint *length);
shader - 指向着色器对象的句柄
count - 着色器源字符串的数量, 着色器可以由多个源字符串组成, 但是每个着色器
只有一个main函数

`string` – 指向保存数量的`count` 的着色器源字符串的数组指针
`length` – 指向保存每个着色器字符串大小且元素数量为`count` 的整数数组指针。

```
void glCompileShader(GLuint shader);
```

`shader` – 需要编译的着色器对象句柄

```
void glGetShaderiv(GLuint shader , GLenum pname , GLint *params );
```

`shader` – 需要编译的着色器对象句柄

`pname` – 获取的信息参数, 可以为 `GL_COMPILE_STATUS/GL_DELETE_STATUS/GL_INFO_LOG_LENGTH/GL_SHADER_SOURCE_LENGTH/ GL_SHADER_TYPE`

`params` – 指向查询结果的整数存储位置的指针。

```
void glGetShaderInfoLog(GLuint shader , GLsizei maxLength, GLsizei *length , GLChar *infoLog);
```

`shader` – 需要获取信息日志的着色器对象句柄

`maxLength` – 保存信息日志的缓存区大小

`length` – 写入的信息日志的长度(减去null 终止符); 如果不需要知道长度. 这个参数可以为Null

`infoLog` – 指向保存信息日志的字符缓存区的指针。

```
GLuint glCreateProgram( )
```

创建一个程序对象

返回值: 返回一个执行新程序对象的句柄

```
void glDeleteProgram( GLuint program )
```

program : 指向需要删除的程序对象句柄

//着色器与程序连接/附着

```
void glAttachShader( GLuint program , GLuint shader );
```

program : 指向程序对象的句柄

`shader` : 指向程序连接的着色器对象的句柄

//断开连接

```
void glDetachShader(GLuint program);
```

program : 指向程序对象的句柄

`shader` : 指向程序断开连接的着色器对象句柄

```
glLinkProgram(GLuint program)
```

program: 指向程序对象句柄

链接程序之后, 需要检查链接是否成功. 你可以使用`glGetProgramiv` 检查链接状态:

```
void glGetProgramiv (GLuint program,GLenum pname, GLint *params);
```

program: 需要获取信息的程序对象句柄

pname : 获取信息的参数,可以是:
GL_ACTIVE_ATTRIBUTES
GL_ACTIVE_ATTRIBUTES_MAX_LENGTH
GL_ACTIVE_UNIFORM_BLOCK
GL_ACTIVE_UNIFORM_BLOCK_MAX_LENGTH
GL_ACTIVE_UNIFORMS
GL_ACTIVE_UNIFORM_MAX_LENGTH
GL_ATTACHED_SHADERS
GL_DELETE_STATUS
GL_INFO_LOG_LENGTH
GL_LINK_STATUS
GL_PROGRAM_BINARY_RETRIEVABLE_HINT
GL_TRANSFORM_FEEDBACK_BUFFER_MODE
GL_TRANSFORM_FEEDBACK_VARYINGS
GL_TRANSFORM_FEEDBACK_VARYING_MAX_LENGTH
GL_VALIDATE_STATUS
params : 指向查询结果整数存储位置的指针

从程序信息日志中获取信息

```
void glGetProgramInfoLog( GLuint program ,GLsizei maxLength, GLsizei  
i *length , GLChar *infoLog )
```

program : 指向需要获取信息的程序对象句柄

maxLength : 存储信息日志的缓存区大小

length : 写入的信息日志长度(减去null 终止符),如果不需要知道长度,这个参数可以为Null.

infoLog : 指向存储信息日志的字符缓存区的指针

```
void glUseProgram(GLuint program)
```

program: 设置为活动程序的程序对象句柄.

三.课后总结



四.课后答疑

探探:

- 谈谈OpenGL 顶点着色器/片元着色器? 什么VBO? 以及一些简单的滤镜 Shader?
- 0.3,1.0,0.7 -> 向量(方向). 随便写.
- 后面会讲不着急. GLSL

GLSL 1节.

OpenGL渲染 YUV 数据-> 会讲!

球体: 导入模型.

一般shader 顶点处理->片元处理.

就是没有提示.都可以写.写着就会了.

GLSL 内建函数.就那么几个.

解决问题算法! shader.