Logic_视觉班课堂笔记[CC014]

- **日期**: 2019年6月12日星期三
- **授课**: CC老师
- **课程次数**: 视觉班第14次课--共计(22次课)
- **主题**: OpenGL ES 主题

课程内容

- 基本图形硬件流水线设计
- GLSL 语法
- GPU & CPU 关系
- 索引绘图
- 案例(1). OpenGL ES GLSL 渲染金字塔

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代码复习
2. 请各位同学在案例(1)基础上完成 纹理填充;

3. 请各位同学在案例(1)基础上完成 纹理与颜色混合填充;

- **要求**:
    i. 能独立完成
    ii. 能将案例总结成思维导图

一.课程回顾

- 纹理翻转
- 案例回顾

二.课程笔记

- 变量和数据类型

```
//布尔型. true,false;
bool bDone = false;

//有符合整型数据
int iValue = 42;

//无符号整型数据
uint uiValue = 3929u;

//浮点型
float fValue = 3.14159f;
```

- 向量数据类型

```
//1. 向量声明--4分量的float 类型向量;
vec4 V1;

//2. 声明向量并且对其进行构造
vec4 V2 = vec4(1,2,3,4);

//3. 向量运算,加,赋值给另外一个向量,与标量相乘.
vec4 v;
vec4 vOldPos = vec4(1,2,3,4);
vec4 vOffset = vec4(1,2,3,4);

//注意: 接下来就假设所有参与运算的变量我已经对齐定义且赋值;

v = vOldPos + vOffset;
v = vNewPos;
v += vec4(10,10,10,10);
v = vOldPos * vOffset;
v *= 5;

//4. 向量中元素的获取 可以通过x,y,z,w来获取向量向量中的元素值
v1.x = 3.0f;
```

```
v1.xy = vec2(3.0f,4.0f);
v1.xyz = vec3(3,0f,4,0f,5.0f);

//5. 可以通过颜色控制.r,g,b,a
v1.r = 3.0f;
v1.rgba = vec4(1.0f,1.0f,1.0f,1.0f);

//6.可以通过纹理坐标stpq;
v1.st = vec2(1.0f,0.0f);

//7.注意!赋值混合是否合法.不可以.
v1.st = v2.xt;  //不可以
v1.st = v2.xy;  //可以,没有开发意义.会同行喷死!

//8.向量支持调换(swizzle)操作. 2个或者2个以上向量元素来进行交换.
v1.rgba =  v2.bgra;
v2.bgra =  v1.rgba;

//赋值操作
v1.r = v2.b;
v1.g = v2.g;
v1.b = v2.r;
v1.a = v2.a;

//9.向量还支持一次性对所有分量操作
v1.x = vOtherVerex.x +5.0f;
v1.y = vOtherVerex.y +4.0f;
v1.z = vOtherVerex.z +3.0f;

v1.xyz = vOtherVerex.xyz + vec3(5.0f,4.0f,3.0f);
```

- 矩阵

```
//1.创建矩阵
mat4 m1,m2,m3;

//2.构造单元矩阵
mat4 m2 = mat4(1.0f,0.0f,0.0f,0.0f
                    0.0f,1.0f,0.0f,0.0f,
                    0.0f,0.0f,1.0f,0.0f,
                    0.0f,0.0f,0.0f,1.0f);

mat4 m4 = mat4(1.0f);

mat4 m3 = mat4(0.5,0.5,0.5,0.5,
```

```
                    0.5,0.5,0.5,0.5,
                    0.5,0.5,0.5,0.5,
                    0.5,0.5,0.5,0.5,)

//2.
m1 = m2 * m3;
```

- const

```
const float zero = 0.0;
```

- 结构体

```
struct forStruct{

 vec4 color;
 float start;
 float end;

}fogVar;

fogVar = fogStruct(vec4(1.0,0.0,0.0,1.0),0.5,2.0);

vec4 color = fogVar.color;
float start = fogVar.start;
```

- 数组

```
float floatArray[4];
vec4 vecArray[2];

//注意
float a[4] = float[](1.0,2.0,3.0,4.0);
vec2 c[2] = vec2[2](vec2(1.0,2.0),vec2(3.0,4.0));
```

- 函数

定义函数给3个修饰符.
in: (没有指定时,默认限定修饰符), 传递进入函数中,函数不能对其进行修改.

inout:(传入相应数值,并且可以在函数中进行修改.)

out : (函数返回时,可以将其修改)

```
vec4 myFunc(inout float myFloat, out vec4 m1, mat4 m2){

//函数中计算

}

vec4 diffuse(vec3 normal ,vec3 light, vec4 baseColor)
{
   return baseColor * dot(normal,light);

}

//注意: GLSL 函数中没有递归!!!
```

- 控制语句

```
if(color.a < 0.25)
{
    color *= color.a;
}else
{
    color = vec4(1.0,1.0,1.0,1.0);
}

//循环只支持 while 循环/do ...while... /for
但是! OpenGL ES 开发中,尽量减轻逻辑判断.尽量降低循环迭代使用!
```

三.课程总结

> 详细请参考课后总结视频,这里只是做为关键字记录.

四.课程答疑

> 详细请参考课后答疑视频,这里只是做为关键字记录