使用多重问题

第一步

Fragment shader 定义纹理采样器:

uniform sampler2D CC\_Texture0

uniform sampler2D CC\_Texture1

第二步：

this.tex1Location = gl.getUniformLocation(p, "CC\_Texture0"); this.tex2Location = gl.getUniformLocation(p, "CC\_Texture1");

gl.activeTexture(gl.TEXTURE0); //webgl中一共32个，可以看cocos2d列的常量

gl.bindTexture(gl.TEXTURE\_2D, this.tex1.getName());

gl.uniform1i(this.tex1Location, 0); //把CC\_Texture0指向gl.TEXTURE0 gl.activeTexture(gl.TEXTURE1);

gl.bindTexture(gl.TEXTURE\_2D, this.tex2.getName());

gl.uniform1i(this.tex1Location, 1);

详解：

可以这样简单的理解为：显卡中有N个纹理单元（具体数目依赖你的显卡能力），每个纹理单元（GL\_TEXTURE0、GL\_TEXTURE1等）都有GL\_TEXTURE\_1D、GL\_TEXTURE\_2D等，如下[代码](http://www.xuebuyuan.com/" \o "代码" \t "http://www.xuebuyuan.com/_blank)：

struct TextureUnit

{

GLuint targetTexture1D;

GLuint targetTexture2D;

GLuint targetTexture3D;

GLuint targetTextureCube;

...

};

TextureUnit textureUnits[GL\_MAX\_TEXTURE\_IMAGE\_UNITS]

GLuint currentTextureUnit = 0;

默认情况下当前活跃的纹理单元为0.

void glActiveTexture(GLenum textureUnit)

{

currentTextureUnit = textureUnit - GL\_TEXTURE0 ;

}

**glActiveTextue 并不是激活纹理单元，而是选择当前活跃的纹理单元。**

void glBindTexture(GLenum textureTarget, GLuint textureObject)

{

TextureUnit \*texUnit = &textureUnits[currentTextureUnit];

switch(textureTarget)

{

case GL\_TEXTURE\_1D: texUnit->targetTexture1D = textureObject; break;

case GL\_TEXTURE\_2D: texUnit->targetTexture2D = textureObject; break;

case GL\_TEXTURE\_3D: texUnit->targetTexture3D = textureObject; break;

case GL\_TEXTURE\_CUBEMAP: texUnit->targetTextureCube = textureObject; break;

}

}

从示例代码中可以看到：当绑定纹理目标时，所作用的是当前活跃的纹理单元。

//地球着色器

precision mediump float;

varying vec2 vTextureCoord;//接收从顶点着色器过来的参数

varying vec4 vAmbient;

varying vec4 vDiffuse;

varying vec4 vSpecular;

uniform sampler2D sTextureDay;//纹理内容数据

uniform sampler2D sTextureNight;//纹理内容数据

void main()

{

//给此片元从纹理中采样出颜色值

vec4 finalColorDay;

vec4 finalColorNight;

finalColorDay= texture2D(sTextureDay, vTextureCoord);

finalColorDay = finalColorDay\*vAmbient+finalColorDay\*vSpecular+finalColorDay\*vDiffuse;

finalColorNight = texture2D(sTextureNight, vTextureCoord);

finalColorNight = finalColorNight\*vec4(0.5,0.5,0.5,1.0);

if(vDiffuse.x>0.21)

{

gl\_FragColor=finalColorDay;

}

else if(vDiffuse.x<0.05)

{

gl\_FragColor=finalColorNight;

}

else

{

float t=(vDiffuse.x-0.05)/0.16;

gl\_FragColor=t\*finalColorDay+(1.0-t)\*finalColorNight;

}

}

Int uDayTexHandle=GLES20.glGetUniformLocation(mProgram, "sTextureDay");

Int uNightTexHandle=GLES20.glGetUniformLocation(mProgram, "sTextureNight");

GLES20.glActiveTexture(GLES20.GL\_TEXTURE0);

GLES20.glBindTexture(GLES20.GL\_TEXTURE\_2D, texId);

GLES20.glUniform1i(uDayTexHandle, 0);

GLES20.glActiveTexture(GLES20.GL\_TEXTURE1);

GLES20.glBindTexture(GLES20.GL\_TEXTURE\_2D, texIdNight);

GLES20.glUniform1i(uNightTexHandle, 1);