

玄雨

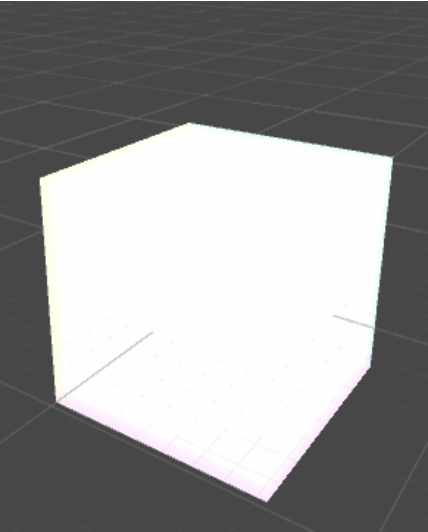
代码优化世界

博客园 首页 新随笔 联系 订阅 管理

UnityShader快速上手指南（二）

简介

前一篇介绍了如果编写最基本的shader，接下来本文将会简单的深入一下，我们先来看下效果吧



呃，gif效果不好，实际效果是很平滑的动态过渡

实现思路

- 1.首先我们要实现一个彩色方块
 - 2.让色彩动起来
- over

实现一个RGB CUBE

先看代码吧：

```
Shader "LT/Lesson2"
{
    Properties {
        _OffsetX ("Offset X", Range (-1.5, 1.5) ) = 0
        _OffsetY ("Offset Y", Range (-1.5, 1.5) ) = 0
        _OffsetZ ("Offset Z", Range (-1.5, 1.5) ) = 0
    }
    SubShader
    {
        Pass
        {
            CGPROGRAM
            #pragma vertex vert
            #pragma fragment frag
            #include "UnityCG.cginc"

            struct VertexOutput {
                float4 pos : SV_POSITION ;
                fixed4 col : COLOR ;
            };

            uniform float _OffsetX;
            uniform float _OffsetY;
```

公告

0 visitors
Jun. 01st - Jun. 30th

Click to Enlarge Map

总访问量：
02934

昵称：玄雨
园龄：3年7个月
粉丝：7
关注：1
[+加关注](#)

2016年6月						
<	日	一	二	三	四	五
	29	30	31	1	2	3
	5	6	7	8	9	10
	12	13	14	15	16	17
	19	20	21	22	23	24
	26	27	28	29	30	1
	3	4	5	6	7	8

搜索

找找看

谷歌搜索

常用链接

- 我的随笔
- 我的评论
- 我的参与
- 最新评论
- 我的标签
- 更多链接

我的标签

- unity(15)
- shader(3)
- android(3)
- ugui(2)
- 多语言本地化(2)
- 自定义布局(1)
- UI(1)
- java(1)
- opengl(1)
- tcp(1)
- 更多

```

uniform float _OffsetZ;

VertexOutput vert ( appdata_base input )
{
    VertexOutput result;
    result.pos = mul(UNITY_MATRIX_MVP , input.vertex ) ;
    result.col = input.vertex + float4( _OffsetX, _OffsetY, _OffsetZ, 0);
    return result;
}

fixed4 frag ( VertexOutput input ) : COLOR
{
    return input.col;
}

ENDCG
}
}
}

```

恩~~，首先呢，我们这次输出的颜色不同的位置颜色不同，所以我们需要一个同时能存位置和颜色的结构体：

```

struct VertexOutput {
    float4 pos : SV_POSITION ;
    // 位置信息，后面的：SV_POSITION是必须的,当然你也可以换成：POSITION
    fixed4 col : COLOR ;
    // 颜色信息，后面的：COLOR不是必须的，你可以随便取名字比如：FUCK
    // 但是嘛，为了代码方便阅读，还是写成COLOR吧
};

```

然后呢我们只有这样一个模型：



24个顶点（每个面顶点单算的），12个三角形，两个空的UV（这个是unity自带的cube模型）

这个模型是没有任何颜色信息，所以我们需要自己在shader中生成他的颜色

出于方便考虑，我们将这个模型的顶点（XYZ）变成RGB的颜色，因为刚好三个值都有变化嘛

于是有了这样的代码

```
result.col = input.vertex + float4( _OffsetX, _OffsetY, _OffsetZ, 0);
```

前面顶点位置就不作处理了，直接换算成Unity坐标就完了

```
result.pos = mul(UNITY_MATRIX_MVP, input.vertex );
```

然后我们来说传入参数中的appdata_base

对于VertexOutput vert (appdata_base input)这个函数命名

学过C语言的应该知道前面是返回值类型 括号里面是传入值类型和名字吧

然后这个appdata_base呢是定义在#include "UnityCG.cginc"的一个结构体

（强行带节奏引入了UnityCG.cginc，其实也可以像前面一篇一样使用float4 position : POSITION，只是这里为了早点引入UnityCG.cginc而已）

我们可以看一下UnityCG.cginc的部分代码：

```

// Dynamic & Static lightmaps contain indirect diffuse lighting, thus ignore SH
#define UNITY_SHOULD_SAMPLE_SH ( defined (LIGHTMAP_OFF) && defined(DYNAMICLIGHTMAP_OFF) )

struct appdata_base {
    float4 vertex : POSITION;
    float3 normal : NORMAL;
    float4 texcoord : TEXCOORD0;
};

struct appdata_tan {
    float4 vertex : POSITION;
    float4 tangent : TANGENT;
    float3 normal : NORMAL;
    float4 texcoord : TEXCOORD0;
};

struct appdata_full {
    float4 vertex : POSITION;
    float4 tangent : TANGENT;
    float3 normal : NORMAL;
    float4 texcoord : TEXCOORD0;
    float4 texcoord1 : TEXCOORD1;
    float4 texcoord2 : TEXCOORD2;
    float4 texcoord3 : TEXCOORD3;
};

```

随笔档案

- 2016年5月 (4)
- 2016年4月 (2)
- 2016年3月 (2)
- 2016年1月 (2)
- 2015年12月 (1)
- 2015年9月 (2)
- 2015年8月 (2)
- 2015年5月 (3)

最新评论

1. Re:UnityShader快速上手指南（二）
支持一下！
--马三小伙儿
2. Re:UnityShader快速上手指南（一）
学习了！
--马三小伙儿
3. Re:Unity实现滑页嵌套（解决ScrollRect嵌套冲突问题）
@玄雨嗯!谢谢...
--WenanLee
4. Re:Unity实现滑页嵌套（解决ScrollRect嵌套冲突问题）
@WenanLee最近项目比较忙，例子工程被改来做其他用途了，没有可以上传的项目了，还有目前这个思路实现的核心代码已经在blog上了，你要想完善的话自己搭建一个项目吧 (o_v o)顺带给你说我自己使用的.....
--玄雨
5. Re:Unity实现滑页嵌套（解决ScrollRect嵌套冲突问题）
玄雨你好,这个项目你可以上传到GitHub上吗?我想试着把你这个完善一下。
--WenanLee

阅读排行榜

1. Unity Android交互过坑指南(440)
2. Unity实现滑页效果（UGUI）(438)
3. Unity Sprite Packer 问题集合(332)
4. Unity实现屏幕抖动效果（通过Camera Viewpoint实现）(245)
5. Android自定义Toast(237)

评论排行榜

1. Unity实现滑页效果（UGUI）(6)
2. Unity实现滑页嵌套（解决ScrollRect嵌套冲突问题）(4)
3. Unity实现屏幕抖动效果（通过Camera Viewpoint实现）(2)
4. UnityShader快速上手指南（二）(1)
5. UnityShader快速上手指南（一）(1)

推荐排行榜

1. UnityShader快速上手指南（二）(4)
2. UnityShader快速上手指南（一）(4)
3. Unity实现滑页效果（UGUI）(3)
4. Unity多语言本地化改进版(2)
5. UnityShader快速上手指南（三）(2)

```
#if defined(SHADER_API_XBOX360)
    half4 texcoord4 : TEXCOORD4;
    half4 texcoord5 : TEXCOORD5;
#endif
    fixed4 color : COLOR;
};
```

整个有点小长,我只粘贴一部分, 其实就是一大堆Unity的定义而已,
我们再来看看

```
struct appdata_base {
    float4 vertex : POSITION; //位置
    float3 normal : NORMAL; //法线
    float4 texcoord : TEXCOORD0; // 纹理
};
```

其实这是一个简化的模型数据, 包含了一些常用的参数, 如果我们写的shader主要是给手机使用的话, 这些数据基本也就够了, 而且目前我们也用了他的位置的信息, 当然你也可以传入一个appdata_full 类型, 区别不大

至于_OffsetX, _OffsetY, _OffsetZ三个外接属性的定义就不多做赘述了
然后我们就通过

```
VertexOutput vert ( appdata_base input )
{
    VertexOutput result;
    result.pos = mul(UNITY_MATRIX_MVP , input.vertex ) ;
    result.col = input.vertex + float4( _OffsetX, _OffsetY, _OffsetZ, 0);
    return result;
}
```

计算出了相应顶点的颜色

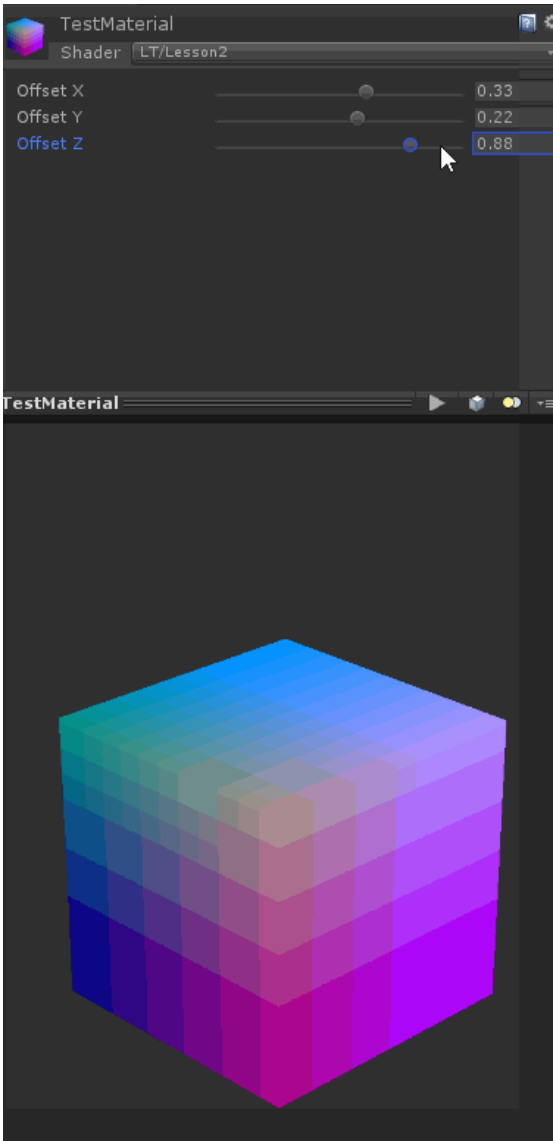
然后直接在面片渲染函数中把对应点的颜色赋值给他就行了

return input.col;

注意这里我们的传入参数变成了vert 的返回值

fixed4 frag (VertexOutput input) : COLOR

好来看下初步的效果:



这里我们就完成了一个RBG CUBE了。
下面对一些原理性的东西简单解释一下

光栅化?插值?

前面我解释过vert函数是一个顶点调用一次，这里我们的模型一共才24个顶点，但是为啥出来这么多个颜色呢，这里就跟渲染流程的光栅化有关。默认情况下，光栅化会保持平滑过渡,如果两边不匹配就会在中间插值，然后对于我们的模型而言，一个面上每个顶点的颜色都不同，所以他就会自动插入很多个顶点，并且自动渐变颜色来满足平滑过渡（也就是说如果你颜色都一样，就不会插点了，当然你也可以手动不让他插点，概念比较多，这里就不展开了，我们要快速上手嘛）

让颜色随着时间变化而变化

这里是我强行要加的一个功能，不然感觉这篇blog就太少内容咯，哈哈
有两种实现方法：

- 1.unity中通过C#代码去控制刚才开放出来的参数
- 2.shader中自己通过时间去更改颜色

我们既然是学shader，当然是在shader中进行更改啦
直接上代码：

```
result.col = input.vertex + float4( _SinTime.w + 0.5, _SinTime.w + 0.5, _SinTime.w + 0.5, 0);
```

呃，对，就改这一行。效果就是实现啦，大家可以自己行试一下
下面解释一小下下：

_SinTime是unity为shader内置的一个时间的sin值得变量（看名字也看的出来吧）
需要引入#include "UnityCG.cginc"（这也是为啥我前面强行带节奏的原因）
然后来普及下Unity为我们内置了哪些东西吧：

```
Transformations 变换

float4x4 UNITY_MATRIX_MVP
Current model*view*projection matrix
```

```

当前物体*视*投影矩阵。（注：物体矩阵为 本地->世界）
float4x4 UNITY_MATRIX_MV
Current model*view matrix
当前物体*视矩阵
float4x4 UNITY_MATRIX_P
Current projection matrix
当前物体*投影矩阵
float4x4 UNITY_MATRIX_T_MV
Transpose of model*view matrix
转置物体*视矩阵
float4x4 UNITY_MATRIX_IT_MV
Inverse transpose of model*view matrix
逆转置物体*视矩阵
float4x4 UNITY_MATRIX_TEXTURE0 to UNITY_MATRIX_TEXTURE3
Texture transformation matrices
贴图变换矩阵
float4x4 _Object2World
Current model matrix
当前物体矩阵
float4x4 _World2Object
Inverse of current world matrix
物体矩阵的逆矩阵
float3 _WorldSpaceCameraPos
World space position of the camera
世界坐标空间中的摄像机位置
float4 unity_Scale
xyz components unused; .w contains scale for uniformly scaled objects.
不适用xyz分量，而是通过w分量包含的缩放值等比缩放物体。

_ModelLightColor float4 Material's Main * Light color 材质的主颜色*灯光颜色
_SpecularLightColor float4 Material's Specular * Light color 材质的镜面反射（高光）*灯光颜色。
_ObjectSpaceLightPos float4 Light's position in object space. w component is 0 for
directional lights, 1 for other lights
物体空间中的灯光为，平行光w分量为零其灯光为1；
_Light2World float4x4 Light to World space matrix 灯光转世界空间矩阵
_World2Light float4x4 World to Light space matrix 世界转灯光空间矩阵
_Object2Light float4x4 Object to Light space matrix 物体转灯光空间矩阵

float4 _Time : Time (t/20, t, t*2, t*3), use to animate things inside the shaders
时间：用于Shader中可动画的地方。
float4 _SinTime : Sine of time: (t/8, t/4, t/2, t)
时间的正弦值。
float4 _CosTime : Cosine of time: (t/8, t/4, t/2, t)
时间的余弦值
float4 _ProjectionParams : 投影参数
x is 1.0 or -1.0, negative if currently rendering with a flipped projection matrix
x为1.0 或者-1.0如果当前渲染使用的是一个反转的投影矩阵那么为负。
y is camera's near plane y是摄像机的近剪裁平面
z is camera's far plane z是摄像机远剪裁平面
w is 1/FarPlane. w是1/远剪裁平面
float4 _ScreenParams : 屏幕参数
x is current render target width in pixels x是当前渲染目标在像素值中宽度
y is current render target height in pixels y是当前渲染目标在像素值中的高度
z is 1.0 + 1.0/width z是1.0+1.0/宽度
w is 1.0 + 1.0/height w是1.0+1.0/高度

```

呃，格式不是很好看的样子，这里有链接，自己去看吧<http://www.creeger.com/Components/SL-BuiltinValues.html>

有了这些东西之后呢，我们就可以简单的根据时间变化做一些动态shader了，比如什么UV流动啊，颜色动态变化啊，动态模型（是动态模型不是模型动画哈）啥的，瞬间就高大上了有木有，性能嘛取决于你写的代码（同样的代码级别下，shader速度秒杀你在c#中写）

总结

这一篇感觉写的比较乱，主要是知识点比较杂（这理由不是很好找啊....原谅我语文老师是数学老师教大的）

主要知识点是介绍一下光栅化那个插值，这个很重要https://en.wikibooks.org/wiki/Cg_Programming/Rasterization（虽然我讲的一笔带过，大家去看看官方解释吧）

然后介绍一下UnityCG.cginc，我们既然是写的unityshader，当然还是要经常使用这个库的啦，后面还有光照，空间矩阵啥的，基本我们要想做高级特效离不开这个库的，大家可以去看看这个库源码

了解了这些之后，就是单纯的算法了（比如怎么通过时间更改模型顶点位置实现好看的动画啥的）

恩~~~再次坦白下写的比较乱，如有疑问欢迎联系QQ: 821580467一起探讨

标签: [unity](#), [shader](#)

好文要顶

关注我

收藏该文





玄雨
关注 - 1
粉丝 - 2
[+加关注](#)

4

推荐

0

反对


(请你对文章做出评价)

« 上一篇: [UnityShader快速上手指南（一）](#)

» 下一篇: [UnityShader快速上手指南（三）](#)

posted @ 2016-05-30 00:08 玄雨 阅读(111) 评论(1) 编辑 收藏

评论列表

#1楼 2016-05-30 07:32 马三小伙儿 


“

支持一下！

”

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

 注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

最新IT新闻：

- 苹果今日在澳大利亚发行10.3亿美元债券
- 王雪红向股东道歉：HTC亏损 我很抱歉
- 后端傻瓜化？
- 易迅网最艰难的时候，京东为什么选择放老二一马？
- 中国的程序员培训是不是有问题？

» 更多新闻...

最新知识库文章：

- 高效编程之道：好好休息
- 快速学习者的高效学习策略
- 一个前端的自我修养
- 架构漫谈（九）：理清技术、业务和架构的关系
- 架构漫谈（八）：从架构的角度看如何写好代码

» 更多知识库文章...