# UnityShader中的内置时间变量

# 纹理动画

#### 序列帧动画

**优点：**不予要进行任何物理计算就能得到非常细腻的动画效果

**缺点：**需要制作所有关键帧的美术图

注意：

1 序列帧动画 大部分需要进行 透明度融合，所以 Tags的设置都要按照透明度融合设置，

2 Unity中的纹理坐标，是左下角为原点，向上，向右变大。 如果 序列帧纹理 是从上到下，从左到右，进行的；那么采样的y 应该是 减去偏移量

float time = floor(\_Time.y \* \_Speed);

// 时间 对 行的商是 行数，余数是 列数

float row = floor(time / \_HorizonAmount); // 从0 开始

float col = time - row \* \_VerticalAmount; // 从0 开始

// 将纹理坐标缩放到一个格子

half2 uv = float2(i.uv.x / \_HorizonAmount, i.uv.y / \_VerticalAmount);

// 计算格子的偏移量

uv.x = uv.x + col / \_HorizonAmount;

uv.y = uv.y – (row + 1) / \_VerticalAmount;

3 最后一步 row+1 的原因：如果是0行0列，那么读取的坐标是第一行的，因为我们的序列纹理是从上到下，也就是说uv的数据是 序列纹理的最后一行，则第一排的显示的不对。加一之后读取的是 -1 行的数据，因为纹理是重复的，所以就是序列纹理的第一行。

#### 滚动背景

v2f vert(appdata\_base v)

{

v2f o;

o.pos = UnityObjectToClipPos(v.vertex);

o.uv.xy = TRANSFORM\_TEX(v.texcoord, \_MainTex) + frac(float2(\_ScrollX, 0.0)\*\_Time.y);

o.uv.zw = TRANSFORM\_TEX(v.texcoord, \_DetailTex) + frac(float2(\_Scroll2X, 0.0)\*\_Time.y);

return o;

}

fixed4 frag(v2f i) : SV\_Target

{

fixed4 firstLayer = tex2D(\_MainTex, i.uv.xy);

fixed4 secondLayer = tex2D(\_DetailTex, i.uv.zw);

fixed4 c = lerp(firstLayer, secondLayer, secondLayer.a);

c.rgb \*= \_Multiplier;

return c;

}

# 顶点动画

v2f vert(a2v v) {

v2f o;

//对顶点的X 进行变换

float4 offset;

offset.yzw = float3(0.0, 0.0, 0.0);

offset.x = sin(\_Frequency \* \_Time.y +

v.vertex.x \* \_InvWaveLength +

v.vertex.y \* \_InvWaveLength +

v.vertex.z \* \_InvWaveLength) \* \_Magnitude;

o.pos = UnityObjectToClipPos(v.vertex + offset);

o.uv = TRANSFORM\_TEX(v.texcoord, \_MainTex);

// 进行纹理动画

o.uv += float2(0.0, \_Time.y \* \_Speed);

return o;

}

## 广告牌

根据视角方向来旋转一个多边形，使得多边形看起来好像总是面对着摄像机。

**广告牌技术的本质就是构建旋转矩阵，基向量是 表面法线，指向上的方向，指向右的方向。还要指定一个 锚点。**

**求旋转矩阵的基向量，**

**如果固定法线： 右向量 = 法线x上向量；归一化右向量；上向量 = 法线X右向量。**

Tags {"Queue"="Transparent" "IgnoreProjector"="True" "RenderType"="Transparent" "DisableBatching"="True"}

"DisableBatching"="True"

**批处理会合并所有相关的模型，这些模型格子的模型空间会被丢失。而我们需要使用物体的模型空间下的位置来作为锚点进行计算。**

### 注意事项：

**1 在模型空间下进行一些顶点动画，那么批处理往往会破坏这种动画效果。禁用批处理，会增加一些DrawCall，性能有损耗，尽量避免使用模型空间下的一些绝对位置和方向。**

**2 如果需要对包含顶点动画的物体添加阴影，如果继续使用Diffuse等包含的Pass来渲染，则阴影显示不正确，需要自己写ShadowCasterPass，在Pass中加入顶点动画。**

**3 半透明的物体，我们把Fallback设置为 Transparent/VertexLit, 它没有定义ShadowCasterPass，所以没有阴影。**