卡通shader

光照计算

思路

1、计算世界空间模型法线与世界空间光照方向的点乘结果

2、二次处理光照结果：通过if函数判断法线与光照的点乘结果：结果小于等于0的区域为模型的暗部，大于0且小于等于0.25的区域为灰部，大于0.25的区域为亮部

3、Bling-Phong模型计算高光反射

实现

1、计算光照

```
v2f vert(a2v v)
{
    v2f o;
    o.pos = UnityObjectToClipPos(v.vertex);

    o.WorldPos = mul(unity_ObjectToWorld,v.vertex); //获取世界空间模型顶点位置
    o.WorldNormal = UnityObjectToWorldNormal(v.normal);//世界空间法线
    return o;
}

fixed4 frag(v2f i):SV_Target{

    fixed3 WorldNormal = normalize(i.WorldNormal); //归一化世界空间法线
    fixed3 WorldLight = normalize(_WorldSpaceLightPos0); //归一化世界空间光照方向
    fixed3 ViewDir = normalize(_WorldSpaceCameraPos - i.WorldPos); //归一化的视角方向
    fixed Diffuse = dot(WorldLight,WorldNormal); //模型表面的光照
```

在顶点着色器中计算好世界空间模型顶点和法线后，我选择在片元着色器计算光照，这样得到的结果会比在顶点着色器中插值计算的结果的精度要高一些。得到结果用Diffuse表示

2、二次处理光照结果

```
if (Diffuse >= _Dif.x && Diffuse <=_Dif.y)
{
    Diffuse = _Dif.w; //_Dif.w 控制灰部的亮度
}
else if (Diffuse <= _Dif.x )
{
    Diffuse = _Dif.z; //_Dif.z 控制暗部的亮度
}
else
{
    Diffuse = 1; //亮部为1
}
```

得到由3个颜色组成的梯度渐变

3、计算高光反射

```
fixed3 halfDir = normalize(WorldLight + ViewDir);
fixed3 Specular = pow(saturate(dot(WorldNormal,halfDir)),_Gloss); //高光反射
//将高光颜色设置成1
if (Specular.x > 0.01)
{
    Specular.xyz = fixed3(1,1,1);
}
```

这里也是使用同样的方法将高光0到1的结果变成为1

描边效果

思路

1、在一个新的Pass里，将模型前面剔除，只渲染背面，并且将模型表面沿法线方向拓展

实现

```
pass{
    Tags { "lightMode" = "UniversalForward" "Queue" = "Transparent" "RenderType" = "Opaque"}
    cull front
```

1、将该Pass设置为Transparent，并剔除前面

```
v.normal -= 0.5;
v.vertex.xyz = normalize(v.normal) * _Pow + v.vertex;
```

2、直接在模型空间将顶点沿法线方向偏移

```
fixed4 frag(v2f i): SV_Target{

    return fixed4(_LColor,1);
}
```

3、片元着色器直接返回一个颜色变量，控制描边的颜色

效果图