**3 方向光**

**3.1 光照**

**3.1.1 受光照影响的Shader**

**3.1.2 法线向量**

**3.1.3 表面属性**

**3.1.4 光照计算**

**3.2 灯光**

**3.2.1 灯光属性**

**3.2.2 光照函数**

**3.2.3 向GPU发送灯光数据**

最后调用CommandBuffer.SetGlobalVector方法来完成数据传输。

**3.2.4 可见光**

通过cullingResults.visibleLights获取到可见光源的数据。

**3.2.5 支持多个方向光**

光照方向是通过VisibleLight.localToWorldMatrix属性来获取的，该矩阵的第三列即为光源的前向向量，要记得取反。

默认情况下Unity不会将可见光转换为线性空间，我们在CustomRenderPipeline脚本的构造函数中通过将GraphicsSettings.lightsUseLinearIntensity设为true来将光强转换到线性空间。

在Pass中将着色器编译目标级别设置为3.5，该级别越高，允许使用现代GPU的功能越多。如果不设置，Unity默认将着色器编译目标级别设为2.5，介于DirectX着色器模型2.0和3.0之间。

**3.3 BRDF**

**3.3.1 光**

材质和光线相交会发生两种物理现象：散射和吸收。

光线被吸收是由于光被转化成了其它能量，但吸收并不会改变光的传播方向。

散射不会改变光的能量，但会改变它的传播方向。

**3.3.2 Metallic和Smoothnes**

Unity内置渲染管线支持两种流行的基于物理的工作流程：金属工作流和高光反射工作流。使用不同的工作流可以实现相同的效果，只是它们使用的参数不同而已。

Metallic定义该物体表面看起来是否更像金属或非金属。Smoothness是Metallic的附属值，定义了从视觉上看该表面的光滑程度。

**3.3.3 BRDF(双向反射分部函数)属性**

用辐射率来量化光。辐射率是单位面积、单位方向上光源的辐射通量，通常用L表示。基于表面的入射光线的入射辐射率Li来计算出射辐射率Lo的过程被称为是着色的过程。这个过程可以使用BRDF（双向反射分布函数）来定量分析。BRDF可以使用f(l,v)来表示，其中l是光线入射方向，v是观察方向。

一个更直观地理解是，当一束光线沿着入射方向l到达表面某点时，f(l,v)表示了有多少部分的能量被反射到了观察方向v上。

**3.3.4 反射率**

当使用金属工作流时，物体表面对光线的反射率（Reflectivity）会受到Metallic（金属度）的影响，物体的Metallic越大，其自身反照率（Albedo）颜色越不明显，对周围环境景象的反射就越清晰，达到最大时就完全反射显示了周围的环境景象。我们调整BRDF的GetBRDF方法，用1减去金属度得到的不反射的值，然后跟表面颜色相乘得到BRDF的漫反射部分。

我们遵循能量守恒定律，表面反射的光能不能超过入射的光能，这意味着镜面反射的颜色应等于表面颜色减去漫反射颜色。

**3.3.5 粗糙度**

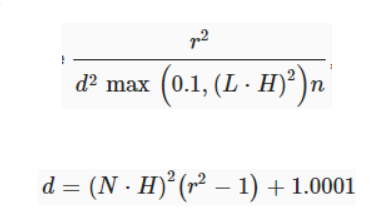
粗糙度和光滑度相反，只需要使用1减去光滑度即可。我们使用源码库中CommonMaterial.hlsl的PerceptualSmoothnessToPerceptualRoughness方法，通过感知到的光滑度得到粗糙度，然后通过PerceptualRoughnessToRoughness方法将感知到的粗糙度平方，得到实际的粗糙度，这与迪士尼光照模型匹配。

**3.3.6 视角方向**

**3.3.7 镜面反射强度**

镜面反射强度取决于视角方向和完美反射方向的对齐程度，我们使用URP中相同的公式，这是简化版Cook-Torrance模型的一种变体。

镜面反射计算公式如下：



**3.4 透明度**

当我们调整小球的Alpha值时，小球会渐渐透明化，但镜面反射也会慢慢消失。在实际情况下，比如透明的玻璃，光线会穿过它或者反射出来，镜面反射并不会消失，我们现在还不能做到这一点。

**3.4.1 Premultiplied（预乘） Alpha**

先说说什么是 Premultiplied Alpha。常见的像素格式为RGBA8888即（r，g，b，a），每个通道8位，范围在［0，255］之间。比如红色50%的透明度可以表示为（255，0，0，127），Premultiplied Alpha是把RGB的通道也乘上透明度比例，这就是（r\*a，g\*a，b\*a，a），那么红色50%透明度则变成了（127，0，0，127）。使用它的好处是可以让两个像素之间线性插值后颜色结果更合理，使得带透明通道图片的纹理可以进行正常的线性插值。

我们想要的结果是调整Alpha值，只让漫反射光照淡化，而镜面反射光照保持完整的强度。将源混合因子设置为One，目标混合因子保持不变，使用OneMinusSourceAlpha。这样会恢复镜面反射，但是漫反射的不受Aplha的影响。

解决方案是调整BRDF文件中的GetBRDF方法，将漫反射颜色乘以表面的Alpha，进行透明度预乘，而不是以后依靠GPU进行混合。

**3.4.2 预乘开关**

**3.5 ShaderGUI**

我们的材质现在支持多种渲染模式，不过切换起来比较麻烦，需要单独配置和进行一些参数调节，我们使用ShaderGUI来对材质面板进行一些扩展，可以很方便的切换各种渲染模式，来一键进行参数配置。

**3.5.1 扩展材质面板**

1. 我们使用CustomEditor来扩展材质面板，声明在Shader最下方。

2. 创建子文件夹Editor，然后创建脚本CustomShaderGUI.cs，该类继承ShaderGUI并重载OnGUI方法来扩展材质编辑器。我们需要访问3个相关对象并追踪它们，第一个MaterialEditor是用来显示和编辑材质的属性，第二个Object[]数组是正在编辑的材质的引用对象，可以通过材质编辑器的Targets属性得到，第三个参数是可以编辑的属性数组。