**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**

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。但OpenGL ES 2.0和WebGL 1.0的图形API是不能处理可变长度的循环的，也不支持线性空间。所以我们在工程构建时可以关闭对OpenGL ES 2.0和WebGL 1.0的支持。

**BRDF(双向反射分布函数)**

在Unity的内置渲染管线中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。其中，金属工作流是默认的工作流程，对应的Shader为Standard Shader。如果想要使用高光反射工作流，需要在材质的Shader下拉框选择Standard（Specular setup）。需要注意的是，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。金属工作流也不意味着它只能模拟金属类型的材质，名字源于它定义了材质表面的金属值（是金属类型的还是非金属类型的）。高光反射工作流的名字源于它可以直接指定表面的高光反射颜色（有很强的高光反射还是很弱的）等，而在金属工作流中这个颜色需要由漫反射颜色和金属值衍生出来。在实际游戏制作过程中，我们可以选择自己更偏好的工作流来制作场景，也可以混合使用。

1. 这里将使用金属工作流，需要为Lit.shader添加两个属性，Metallic和Smoothness。其中Metallic定义了该物体表面看起来是否更像金属或非金属，如果把材质的Metallic值设为1，表明该物体几乎完全是一个金属材质，若设置为0表明该物体几乎没有任何金属特性。Smoothness是Metallic的附属值，定义了从视觉上看该表面的光滑程度，1代表完全光滑，镜面反射最明显，0代表完全粗糙。

我们可以用辐射率来量化光。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，被认为是对单一光线的亮度和颜色评估。在渲染中，通常会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，这个过程往往被称为是着色的过程。

想要得到出射辐射率Lo，需要知道物体表面一点是如何和光进行交互的，这个过程就可以使用BRDF（Bidirectional Reflectance Distribution Function，双向反射分布函数）来定量分析。大多数情况下，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（双向的含义）。

BRDF的含义有两种理解方式。第一种理解是，当给定入射角度后，BRDF可以给出所有出射方向上的反射和散射光线相对分布情况；第二种理解是，当给定观察方向（即出射方向）后，BRDF可以给出从所有入射方向到该出射方向的光线分布。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。

反射率

1. 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。