4 方向阴影

4.1 渲染阴影

确定产生阴影区域的方法就是把光源想象成一个摄像机（暂时叫做光源相机），光源相机的位置和朝向就是光源的位置和发射光线的方向。在渲染场景之前先用光源相机对场景执行一次取景操作，使用LightMode标签为ShadowCaster的Pass，把在光源相机所在角度所有可视的片元深度信息存储在一个帧缓冲区中，称为阴影贴图（Shadow Map），其本质是一张深度图。在真正渲染时把每一个待输出片元再次放到光源相机的角度下计算深度值，如果这次计算的深度值比阴影贴图的深度值要离光源相机远，就表示它落在某个阴影区域中了。Unity就是使用的这种技术。

4.1.1 设置阴影

1. 在渲染阴影之前，我们需要配置一些属性，比如渲染阴影的最大距离和阴影贴图的大小。

**[Serializable]和[SerializeField]**

4.1.2 创建阴影类

4.1.3 带阴影的光源

4.1.4 创建阴影图集

创建阴影图集，然后调用buffer.SetRenderTarget方法来指定渲染数据存储到渲染纹理而不是帧缓冲区中。

Frame Debugger折叠标签

4.1.5 渲染阴影

1，要渲染阴影，首先要创建一个ShadowDrawingSettings实例；

2，方向光没有真实位置，我们要找出与光方向匹配的视图和投影矩阵，并给我们一个裁剪空间的立方体，该立方体与包含光源阴影的摄像机的可见区域重叠，这些数据通过cullResults.ComputeDirectionalShadowMatricesAndCullingPrimitives方法获得；

3，调用buffer.SetViewProjectionMatrices方法应用获取的视图和投影矩阵；

4，调用context.DrawShadows方法渲染阴影投射；

4.1.6 ShadowCaster Pass

DrawShadows方法只渲染Shader中带有ShaderCaster Pass通道的物体，我们在Shader中添加第二个Pass块，LightMode改为ShadowCaster。这个Pass只需写入深度数据，所以添加ColorMask 0不写入任何颜色数据，但会进行深度测试，并写入深度值。

4.1.7 支持多光源渲染阴影

多光源渲染阴影时，需要拆分图集，为每个光源使用自己的阴影贴图图块来渲染。通过使用buffer.SetViewport设置视口矩形。

4.2 采样阴影

我们渲染了阴影贴图，还需要对贴图进行采样才能接收物体的阴影。

4.2.1 阴影转换矩阵

对于每个片元，我们必须从阴影图集中对应的阴影图块采样深度数据，因此需要找到对应在世界空间的阴影纹理坐标，可以通过为每个可投影的定向光创建阴影转换矩阵并将其发送到GPU来实现这点。

通过将获得的光源的投影矩阵和视图矩阵相乘，可以创建一个从世界空间到灯光空间的转换矩阵。

4.2.2 获取方向光的阴影数据

要对光源的阴影采样，需要知道它在阴影图集的图块索引和阴影强度。

4.2.3 阴影图集采样

图集不是常规的纹理，要用TEXTURE2D\_SHADOW宏进行定义。

阴影衰减值是阴影强度和衰减因子的插值。

4.2.4 灯光的阴影衰减

将灯光的阴影衰减添加到入射光强度的计算中。