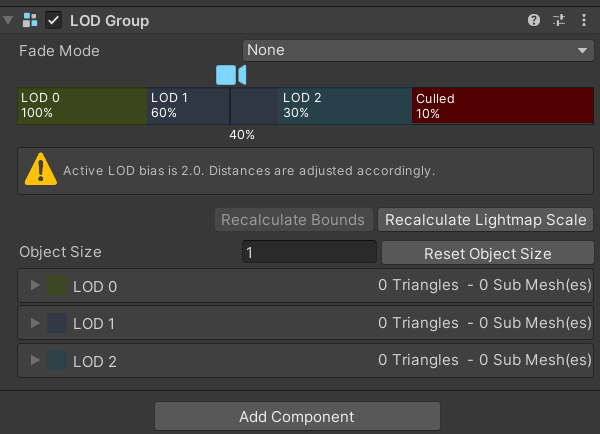
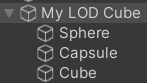
为物体添加LOD组件以实现LOD，比如在下面的实现中，LOD0在物体覆盖窗口60%的面积时有效：

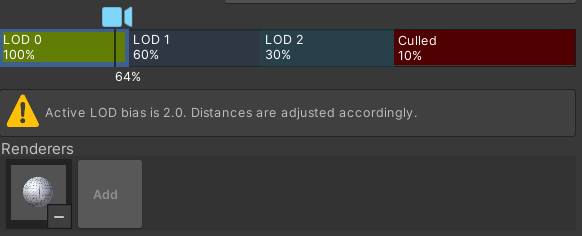


这里的警告表示Quality里为该LOD设置了2.0的放大系数，也就是说LOD0在30%

以上覆盖时有效。

将构建层级的物体作为子节点放在该物体的节点下，并拖拽到对应的LOD层级中，Unity就能自动识别并实现LOD了。



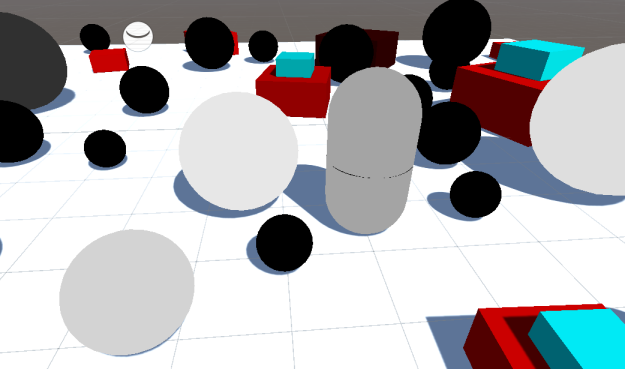


注意一个物体可以被放在多个组别中，因此可以通过LOD实现细节的展示和关闭。

在烘焙时，其他物体只考虑LOD0的内容，但是其余LOD的自身光照也会被烘焙进来。

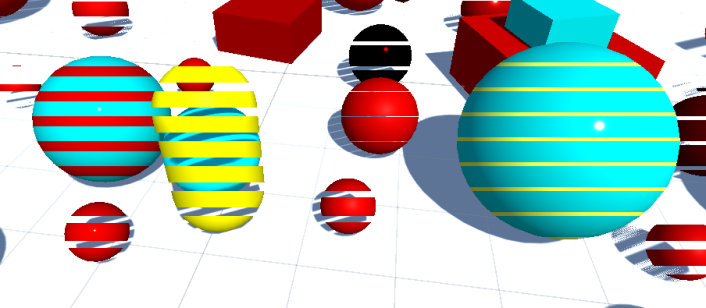
把FadeMode置为Cross Fade，并调整FadeTransitionWidth，可以自定义过渡区域。在过渡区域里，着色器可以实现自定义的混合，关键字是LOD\_FADE\_CROSSFADE。

直接在着色时返回淡出的系数，观察结果：

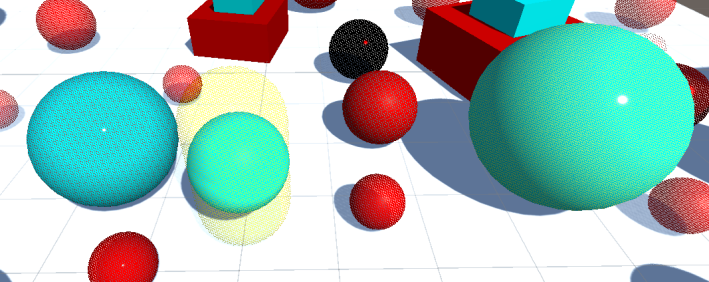


可以使用类似于半透明阴影的实现方法，实现Dither的淡化转变。

原理是每一定数量的像素进行一次Clip，产生交替条纹：（并且注意到两个LOD中有一个因子是负数）

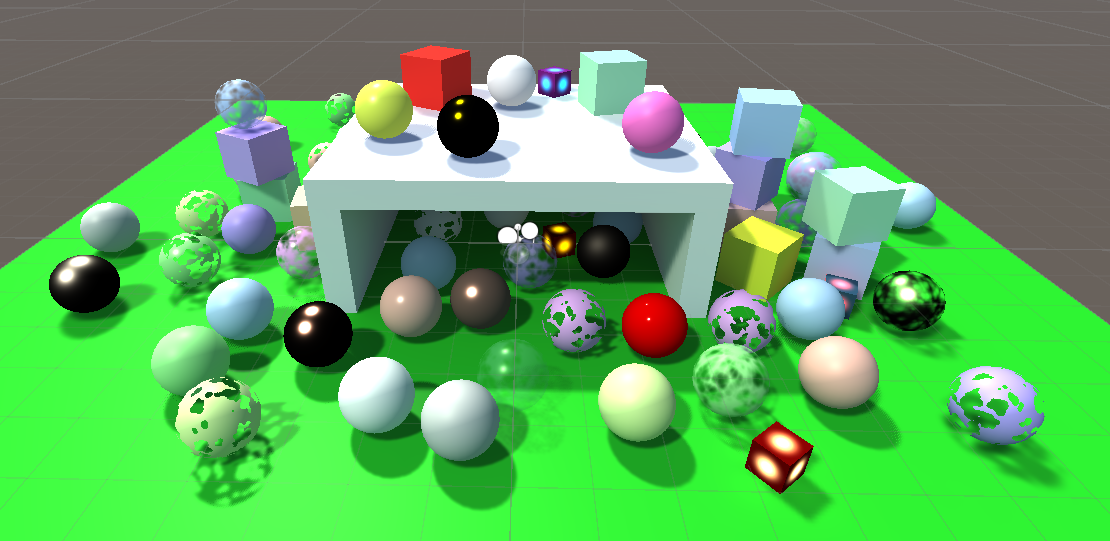


使用程序生成的噪声控制交替：



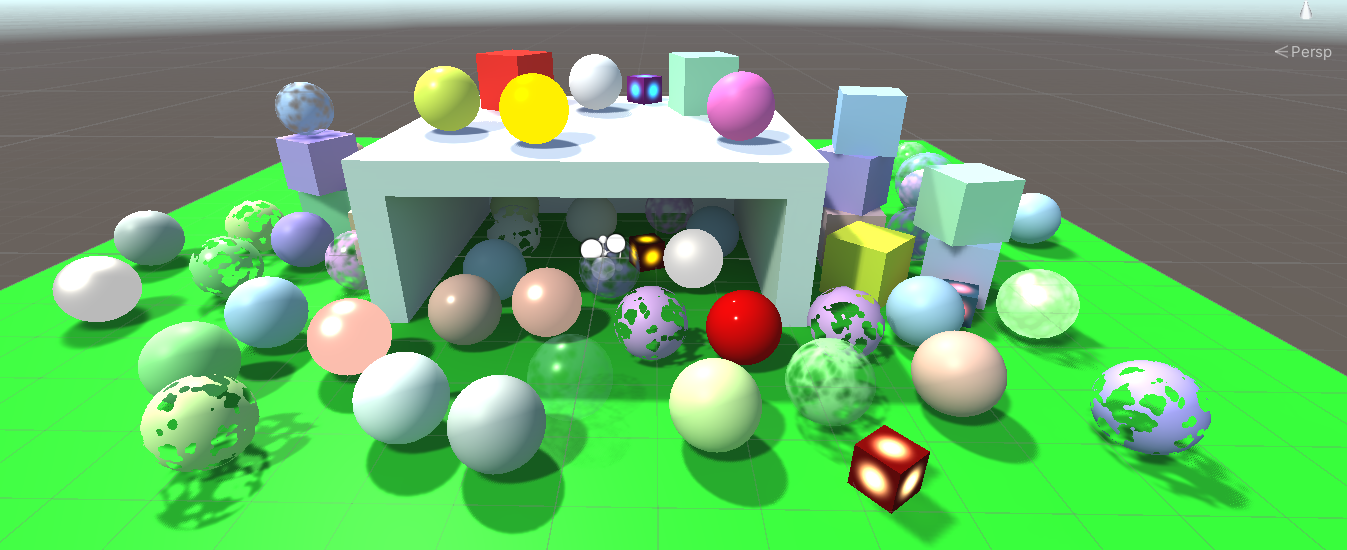
勾选Animated Cross-Fading时，LOD超过阈值后，会快速进行过渡动画的播放，而不会管过度宽度的设置，该动画的播放时间由**[LODGroup](http://docs.unity3d.com/Documentation/ScriptReference/LODGroup.html)**.crossFadeAnimationDuration决定。

接下来实现反射，目前场景中的物体不会被环境光照影响，在金属物体上更明显：



我们之前实现的GI是diffuse项的，下面进一步去支持specular项。

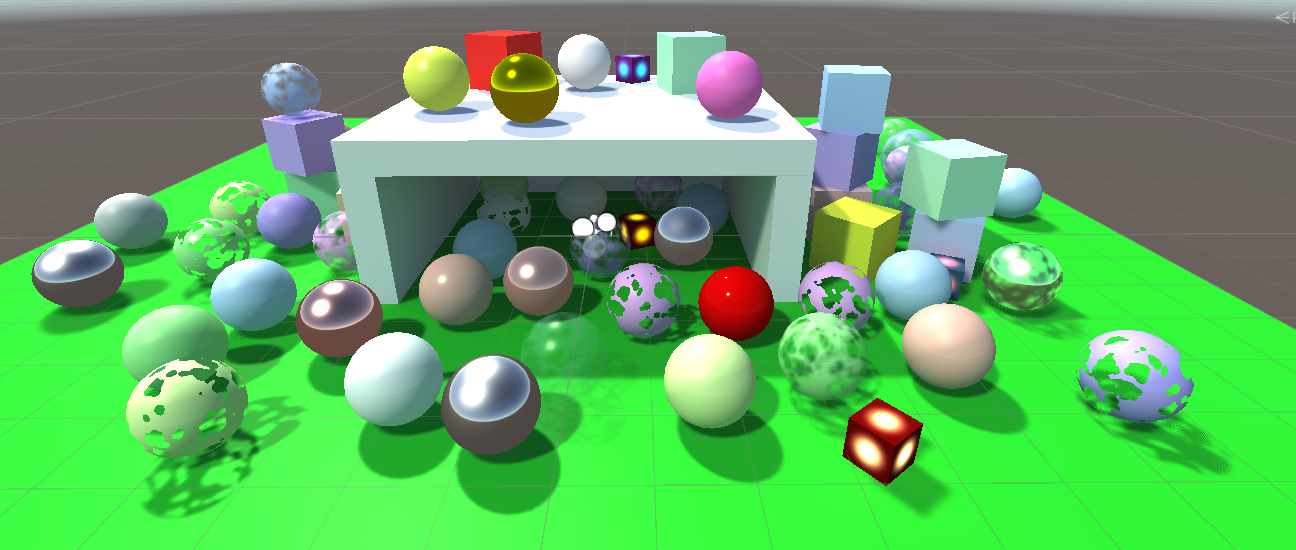
先使用白色作为贡献色，计算gi的specular贡献：



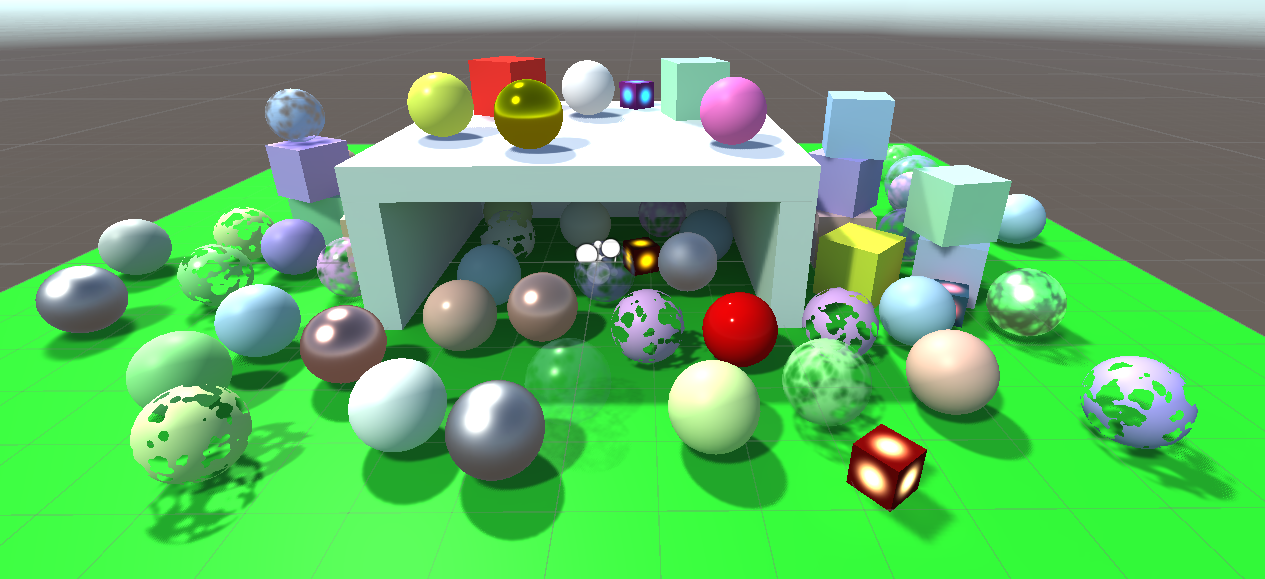
采样环境的立方体贴图时，采样方向为视线方向在表面的反射方向。

记得在drawsettings里把反射选项打开。

可以看到，采样到了天空盒的情况：

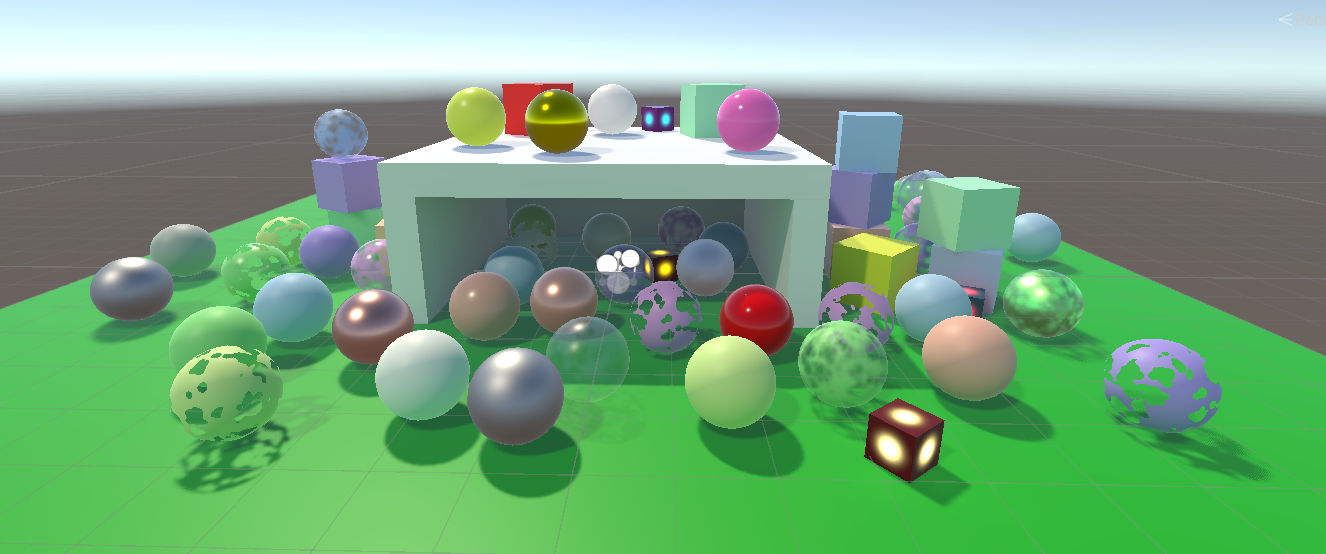


但是，我们要将粗糙度对采样的影响体现出来，基本原理是根据BRDF计算得到的粗糙度去采样不同的mipmap层级（也就是IBL的做法）。

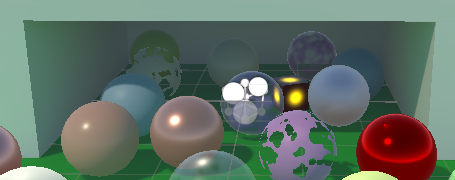


这里暂停一下，因为换了项目，所以忘记改颜色空间了，debug到了Light Color的问题，然后看了这篇才想起来要改成Linear：https://zhuanlan.zhihu.com/p/343936327

下面考虑菲涅尔效应，在掠射角的高光由粗糙度，以及视线和法线的夹角决定。

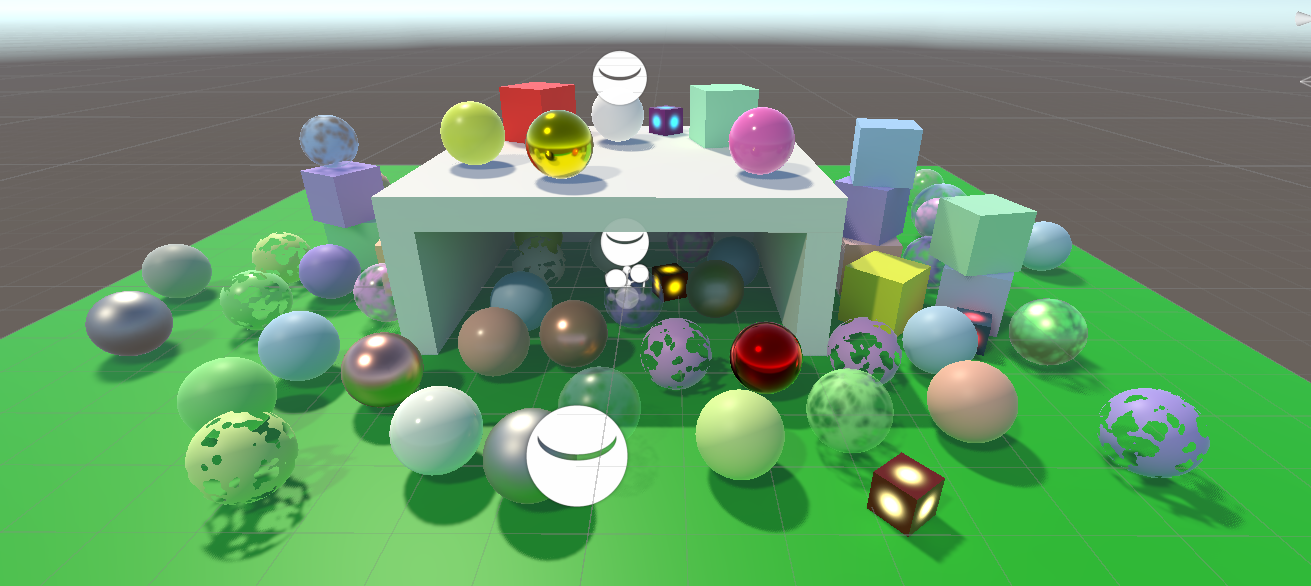


可以为材质添加单独的属性控制该现象：



使用反射探针可以把场景中的物体渲染到CubeMap里，一个场景可以使用多个反射探针，一个反射探针要绘制六次。

使用三个Realtime的反射探针：



这里物体渲染的策略是选择最重要的探针，而不是混合。

最终的渲染结果：

