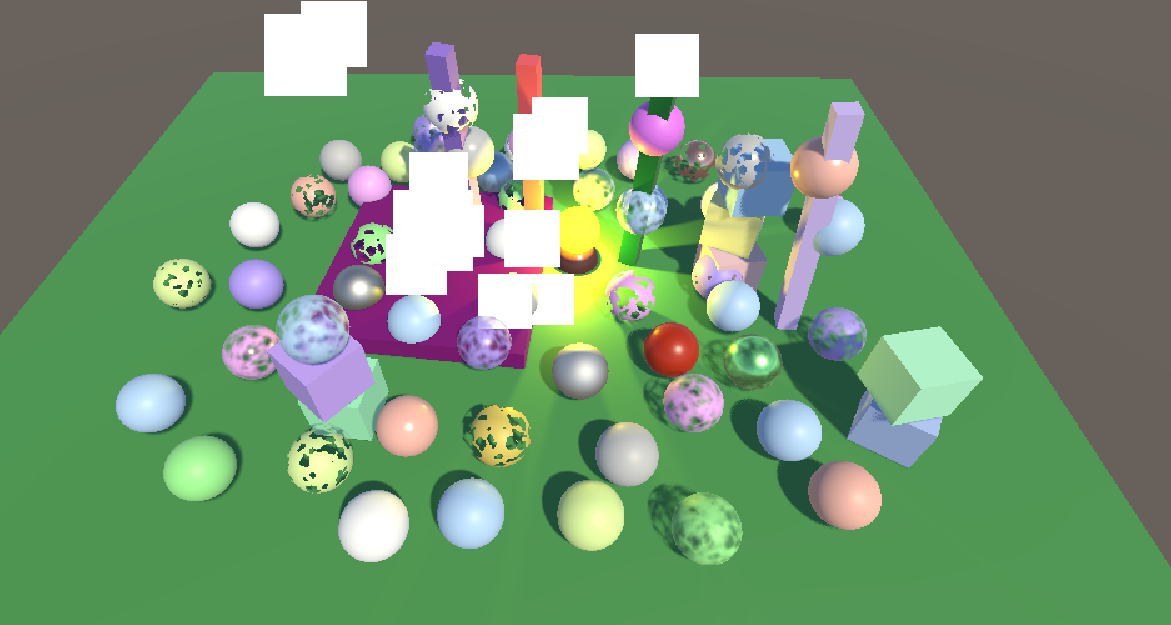
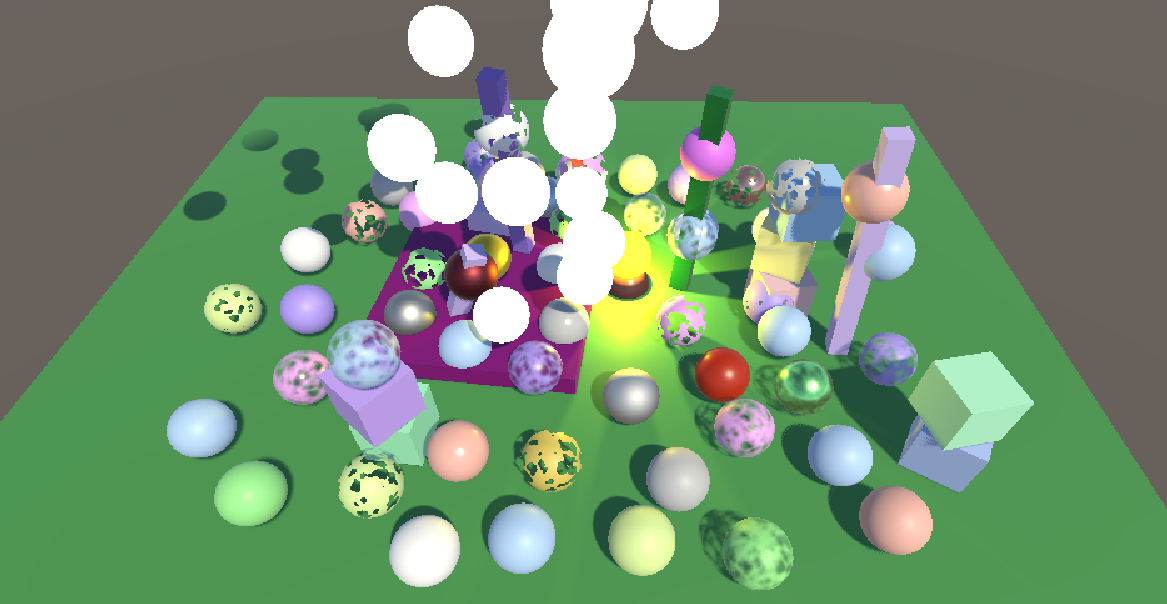
通过GameObject/Effects/Particle System创建粒子系统，赋予unlit材质，得到如下结果：



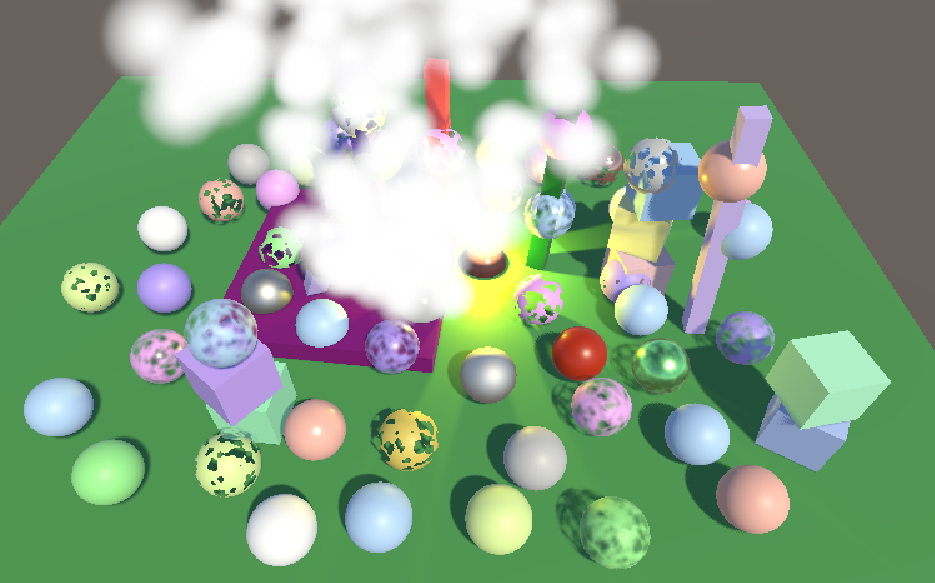
复制Unlit Shader，作为粒子材质，因为粒子是动态的，所以不需要烘焙用的meta Pass。

可以将粒子系统设置为绘制mesh。注意，此时GPU instancing不起作用，因为粒子系统使用的是Procedural Drawing，把Mesh合并为一个单一网格。

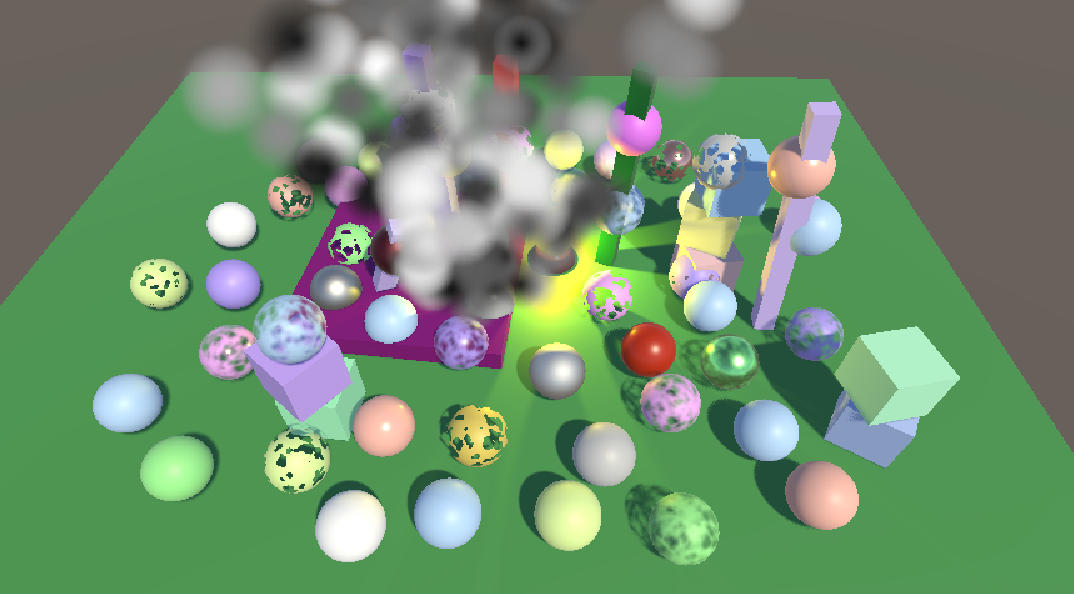
带有阴影的粒子，以球体绘制：



后续我们将专注于无阴影的billboard粒子，使用纹理的Transparent材质：

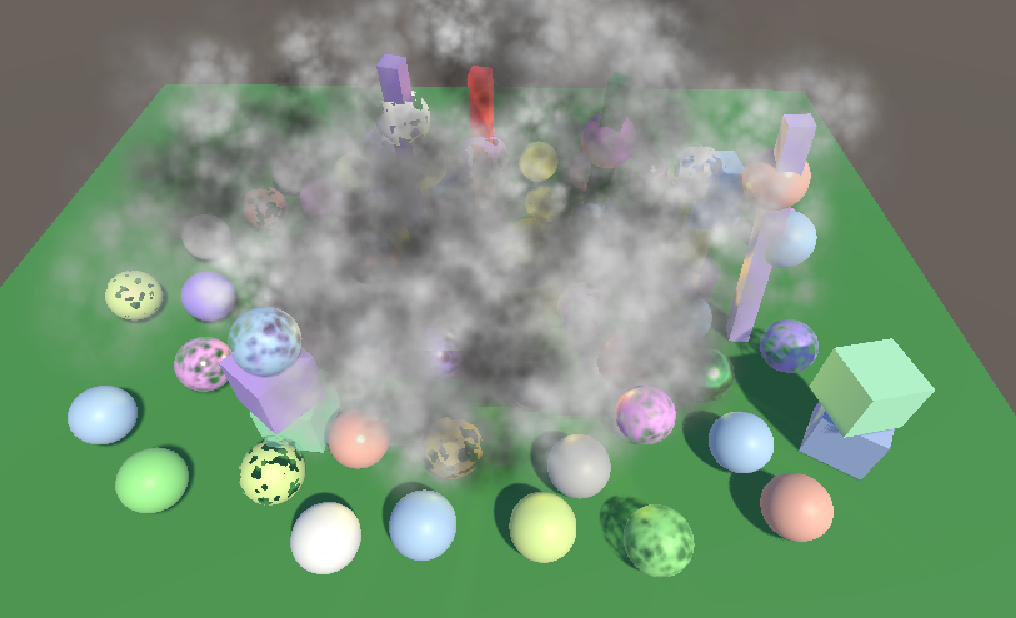


为了应用粒子颜色，在Attributes中添加color输入，并通过shader feature控制该颜色的应用，启用时，将其应用到BaseColor的输出中：

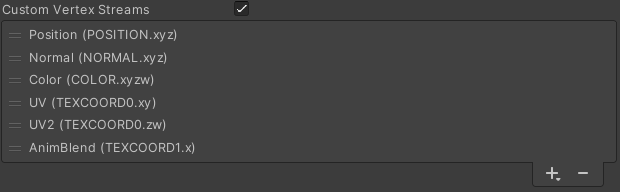


为了获得比较正确的透明度绘制结果，打开按距离排序的选项。请注意，当基于距离排序时，粒子可能会由于视图位置的变化而突然交换绘制顺序，就像任何透明对象一样。（这里有个疑问，Procedural Drawing怎么控制绘制顺序？）// todo

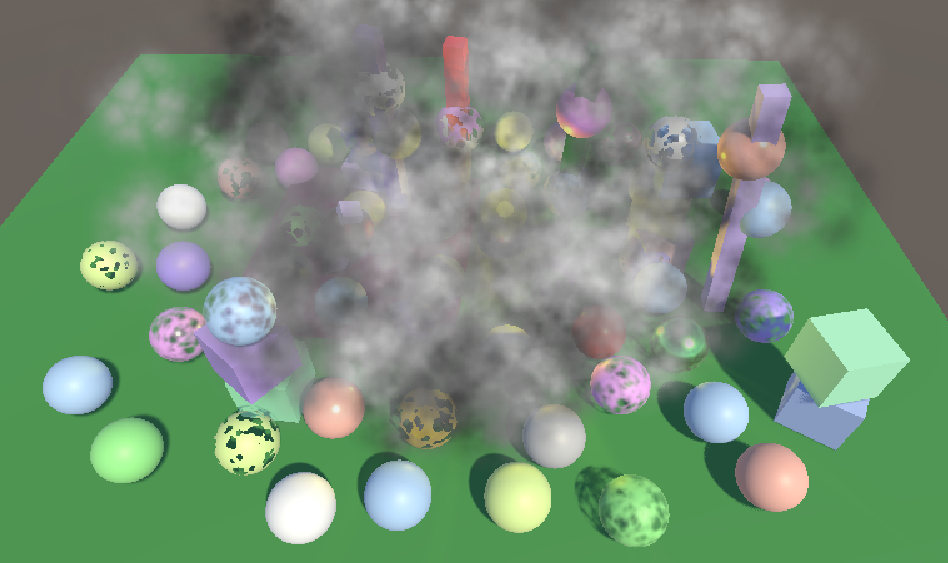
通过粒子系统的Texture Sheet Animation实现FlipBook，但是现在的切换还是直接跳转的：



为了实现混合，我们需要输入额外的UV坐标，以及一个混合因子。在Renderer的Custom Vertex Streams里添加对应接口：



然后在hlsl中添加对应的接口，从TEXCOORD0和1中获取属性并使用lerp应用即可：



下面，我们将处理靠近近平面的粒子，一种方法是设置最大可见尺寸，另一种是根据深度实现fade效果。