# 凹凸映射(bump mapping)

**作用：**

使用一张纹理来修改模型表面的法线，以便为模型提供更多的细节。这种方法不会真的改变模型的顶点位置，只是让模型看起来像是凹凸不平。

**实现方法：**

1：使用一张高度纹理(height map)来模拟表面位移(displacement),然后得到一个修改后的法线值。这种方法叫——**高度映射(height mapping)。**

2：使用一张法线纹理(normal map)来直接存储表面法线，这种方法叫——法线映射(normal mapping)。(常用选择)。

**实现思路：**

第二种方法：

a.在顶点函数中实现：

*a1.将主纹理和凹凸纹理的坐标信息，分别保存在uv.xyzw分量中(通宵使用同一组纹理坐标即可，即两纹理共用\_Mathf.\_ST)。*

*a2.将模型空间下，切线方向、副切线方向、法线方向按行排列来得到从模型空间到切线空间的变换矩阵。(内置宏 TANGENT\_SPACE\_ROTATION)。*

*a3.使用内置函数UnityObjSpaceLightDir()和UnityObjSpaceViewDir()得到模型空间下的光照和视角方向。*

*a4.再利用变换矩阵rotation把它们从模型空间变换到切线空间中。*

b.在片元函数中实现：

b1.归一化光照向量和视角向量。

b2.对bumpmap进行采样，得到切线空间下的法线向量。如果该法线纹理的类型没有设置成Normal map，就需要把法线纹理中，经过映射后的像素值，反映射回来。

b3.

详见7.2.1章节。代码2/BumpMapping

# 遮罩纹理(mask texture)

**作用：**可以让美术人员更加精准(像素级)地控制模型表面的各种性质。

1：允许我们可以保护某些区域，免于某些修改。例如有时我们希望模型表面某些区域的反光强烈些，而某些区域弱些，为了得到更加细腻的效果，我们就可以使用一张遮罩纹理来控制光照。

2：在制作地表材质时需要混合多张图片，例如表现草地的纹理、表现石子的纹理、表现祼露土地的纹理等，使用遮罩纹理可以控制如何混合这些纹理。

**思路流程：**

通过采样得到遮罩纹理的纹素值，然后使用其中某个或某几个通道的值(例如texel.r)来与某种表面属性进行相乘，这样当该通道为0时，可以保护表面不受属性的影响。

# 透明效果

在unity中，有两种方法来实现透明效果：

1:透明测试(Alpha Test),此方法其实无法得到真正的半透明效果。

2：透明度混合(Alpha Blending)。

**透明度测试原理：**它采用一种‘霸道极端’的机制，只要一个片元的透明度不满足条件(通常是小于某个阈值)，那么它对应的片元就会被舍弃，不做任何处理，也不会对颜色缓冲产生任何影响。否则，按正常的处理，进行深度测深度写入等操作。总结，透明度测试不需要关闭深度写入，要么完全透明，要么完全不透明，因上不能得到真正的半透明效果。

**透明度混合：**这种方法可以得到真正的半透明效果。它会使用当前片元的透明度作为混全因子，与已经存储在颜色缓冲中的颜色值进行混合，得到新的颜色。但是，透明度混合需要关闭深度定入，这使得我们要非常小心物体的渲染顺序。需要注意的是，透明度混合只关闭了深度写入，没有关闭深度测试。这意味着，当使用透明混合渲染一个片元时，还是会比较它的深度值与当前深度缓冲中的深度值，如果它的深度值距离摄像机更远，那么就不会再进行混合操作。这一点决定了，当一个不透明物体出现在一个透明物体的前面，而我们先渲染了不透明物体，它仍然可以正常地遮挡住透明物体。也就是说，对于透明度混合来说，深度缓冲是只读的。

**半透明深度写入：**