# 法线纹理

为了让模型表面有凹凸质感，可以加入法线纹理，法线纹理主要是用纹理值来替代法线。算漫反射的时候，diffuse = \_LightColor0.rgb\*albedo\*max(0,dot(normal, lightdir));主要修改cosA的值，即便是白光，白纹理，只有一张法线纹理，则依旧会显示凹凸，因为呈现的是cosA的值，cosA的值又不一样，所以会有凹凸感。

法线纹理的计算：

但凡涉及到法线纹理，就要走两大部：

1 是选择切线空间还是世界空间：

1. 选择切线空间，需要求从模型空间到切线空间的转换，需要把模型空间的xyz轴按行排列。

选择切线空间是把 切线tangent作为X轴，切线和法线的叉乘binormal作为Y轴，把法线normal作为Z轴。

其中binormal = cross(normalize(tangent.xyz), normalize(normal))\*tangent.w;要归一化切线和法线，最后乘以切线的w，来规定方向。

从模型空间转为切线空间的变换为：

rotate = float3x3(tangent.xyz, binormal, normal);

//这个rotate可以用Unity的API计算 TANGENT\_SPACE\_ROTATION;

计算光照主要用的是光照方向和视角方向，因为是从模型空间到切线空间，所以需要获取光照方向，视角方向的模型空间方向，再乘rotate，获取切线空间值。

tangentLightDir = mul(rotate, ObjSpaceLightDir(vertex)).xyz;

tangentViewDir = mul(rotate, ObjSpaceViewDir(vertex)).xyz;

1. 选择世界空间，需要求从切线空间到时间空间的转换

现将切线空间的xz轴转为世界空间坐标系下的worldX, worldZ,然后获得worldY，然后按列排列就是 切线空间到世界空间的 变换矩阵。

2 获取法线纹理的法线向量：

法线存在于法线纹理中，通过tex2D函数获取颜色值，注意这里是fixed4。

fixed4 texColor = tex2D(\_BumpMap, uv);

因为颜色值是[0, 1]。而法线是分布在[-1, 1]上的需要进行映射。

texColor.xy = texColor.xy \* 2 – 1;

UnityAPI：fixed4 texColor = UnpackNormal(texColor);

如果需要添加控制凹凸程度 的变量，就是在这里乘以系数。即：

texColor.xy = (texColor.xy \* 2 – 1) \*系数;

因为法线向量是单位向量, 所以 x^2 + y^2 + z ^2 = 1。所以：

texColor.z = sqrt(1 – dot(texColor.xy, texColor.xy))。

normal = texColor.xyz;

如果是在世界坐标下，那么normal还需要从切线空间转为世界空间

normal = mul (rotation, normal);

**以上所有步骤都是为了求出法线，因为法线存在于法线纹理，存在于切线空间，所以要空间变换，所以要纹理采样并映射到[-1, 1]。**

# 渐变纹理

用于实现插画风格和基于冷到暖色调的着色技术。

渐变纹理的实现：

主要是基于半兰伯特光照模型。

正常的漫反射光: diffuse = \_LightColor0.rgb \* albedo \* max(0, dot(normal, lightDir))

因为 max(0, dot(normal, lightDir)) 将dot(normal, lightDir)的负值截断为0，半兰伯特模型是将 dot(normal, lightDir)的所有的值映射到[0,1] 包括负值。

float hbValue = dot(normal, lightDir) \* 0.5 + 0.5;

对渐变纹理采样，才用的就是hbValue:

fixed3 albedo = tex2D(\_RampTex, fixed2(hbValue, hbValue)).rgb \* 主色调

渐变纹理漫反射：diffuse = \_LightColor0.rgb \* albedo；不用在乘max(0, dot(normal, lightDir))

【-1,1】= 【0,1】\*2 -1

【0,1】=(【-1,1】+1) /2

渐变纹理一定要使用 clamp 模式，因为浮点数的关系可能映射的值不是[0,1],可能是-0.03， 1.002 等，如果使用repeat，首尾的颜色可能会相差很大，造成斑点。

# 遮罩纹理

利用一张纹理，获取颜色rgbw，分别表示对某一项的控制，例如r可以表示高光反射强度，g表示边缘光照的强度等等。

例如：

float specularMask = tex2D(\_Mask, uv).r \* 控制系数

fixed3 specular = \_LightColor0.rgb \* \_Specular.rgb \* pow(max(0, dot(halfDir, normal)), \_Gloss) \* specularMask;