# 前提

Normal map的normal是用于描述面片中的某个点的normal，和顶点的normal不是一个东西，他描述的是更细微的表面，主要用来解决顶点不足的情况下，更真实的表面细节。

height map虽然也是描述面片中某个点的height，但是可以通过曲面细分生成更多的顶点，然后直接位移这些顶点（沿着顶点的normal，这个normal不是采样的方式获得，而是原本顶点自带的normal值），这种做法常见于地形制作。不过本章的中height map不是用来干这个事情的，而也是用来模拟细微表面的。

我们知道一个Fragment，实际上对应的是三维空间中一个面片中的一个点，记为P

Normal map是告诉你P的法线向量是朝哪个方向的，

而Height map是告诉你P的高度是多高，怎么显示一个点有多高呢？也就是模拟一个新的顶点。

# Normal map

## 切线空间是什么？

无论是哪种贴图，本质上，这些值都是局部空间的，什么意思？比如你通过sampler采样到的P点的normal，并不是世界坐标系下的normal，而是相对于面片的坐标系，这就是learn opengl 教程里面反复在提及的切线空间。

<https://learnopengl-cn.github.io/05%20Advanced%20Lighting/04%20Normal%20Mapping/>

## 为什么将lightpos以及viewpos转为切线空间？

一般而言，应该是将局部normal 转为世界 normal 会更好，但是我们要知道一个事情，normal是在fragment shader阶段获取到的，如果是用normal的世界坐标系来计算，那么你需要在fragment shader 阶段对normal进行矩阵计算。那么有没有办法避免在fragment shader阶段进行矩阵计算呢？有办法，那就是将你要做矩阵变化的数值，在vertex shader就做完，然后依靠光栅化自动插值，做法就是将 lightpos以及viewpos转为切线空间，并且这个转化是在vertex shader阶段做完，然后fragment shader阶段就不用对局部normal进行矩阵乘算了，直接去与 切线空间的 lightpos，viewpos进行计算就完事了。

Fragment因为是比vertex要多很多的（一般情况下），所以后者明显效率是高于前者的，唯一的问题就是，为什么切线空间的lightDir 和 viewDir可以通过透视插值得到（在使用normal map技术的前提下，由于normal是在fragment阶段才能获取到的，所以根本无法说在vertex阶段对normal进行插值）

# Height map

Height map的中文名叫 视差贴图，本质上，这就是原理，

# Height Map 和 Normal map的区别

理论上，Height Map中包含了面片中每个点的高度，所以自然而然的normal的信息也是包含进去，但是为什么这两者还是完全不同呢？主要是这两者的用途问题。

Height Map是告诉你计算点 P1 的颜色的时候，其实你要去算的应该是另一个点 P2的颜色。

而 Normal Map 是告诉你计算点 P1的颜色的时候，他的光照漫反射部分应该是用 贴图上的normal以及light dir 来计算。

看到了吗？这两个东西是完全不同的作用，虽然你height map包含了normal 的信息，但是在计算P1点的时候，会去计算P2的颜色，而去计算P2的漫反射的时候，这个信息完全没被用上，因为只有heightmap的时候，正常情况下，fragment不会去计算P的normal是多少，我们知道normal一般就是height的梯度的垂直方向，但是在fragment 阶段，梯度你是无法直接计算的，你不知道你前后左右的height是多少，那么，有没有办法知道呢？答案当然是有的，那就是制作多一张normal map，通过heightmap去预先制作。

所以正常情况下，输入到渲染管线前，肯定是有两张贴图的，heightmap+normalmap，只不过normalmap可以由heightmap生成的（当然你可以尝试非heightmap生成，视觉效果可能会有点奇怪）。