着色回顾：

应用在表面上：Flat shading

应用在顶点上：Gourald Shading

应用在像素上：Phong Shading

1. 在三角形内部做插值：重心坐标（barycentric coordinate）

为什么要在三角形内插值：

很多操作是在三角形顶点进行的，而希望三角形内部是平滑的过渡

插值什么内容：

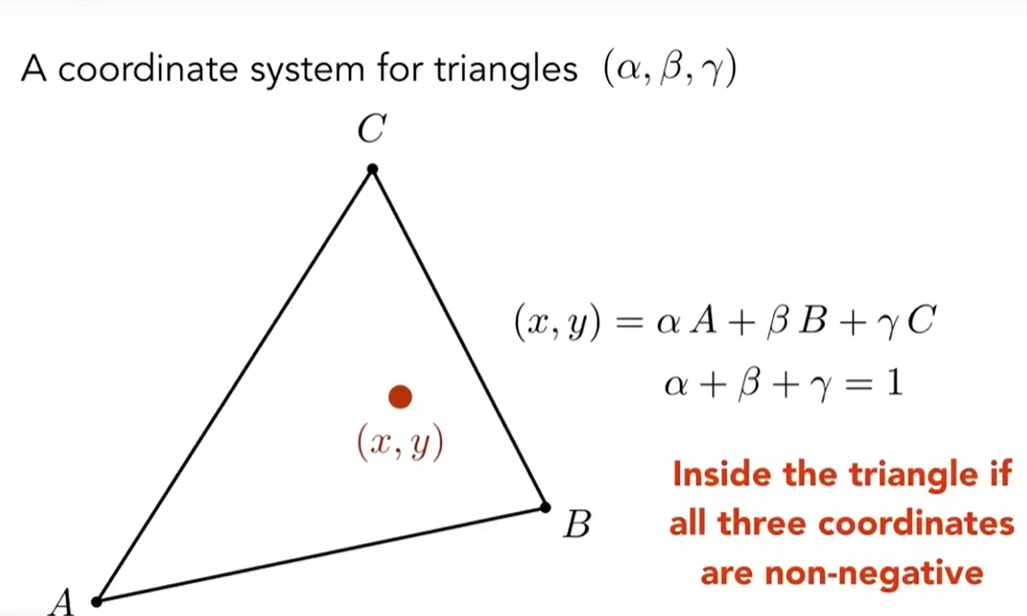
纹理坐标，颜色，法向量

怎么做：

重心坐标：定义在三角形上，三角形中的任何一个点都可以表示为ABC三个坐标的线性组合。

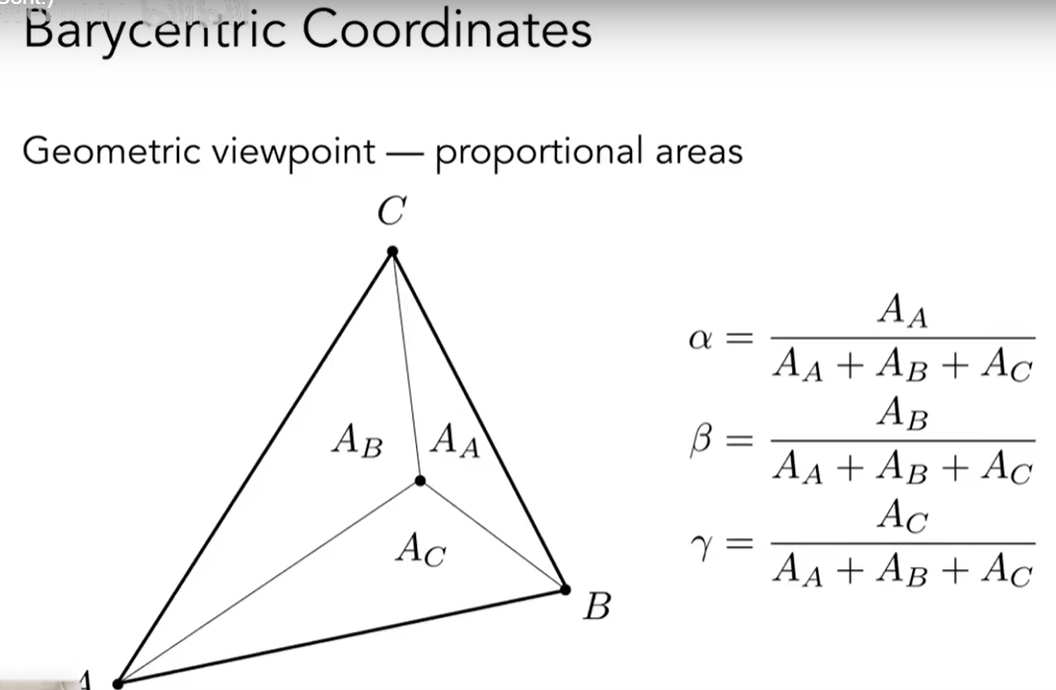
如果三个系数都是非负，则该点一定在三角形内。

系数之和必须为1，如果不为1，则该点不在三角形所在平面内



平面上任意一点都可以用A,B,C坐标表示，则A点自己的坐标是(1, 0, 0)

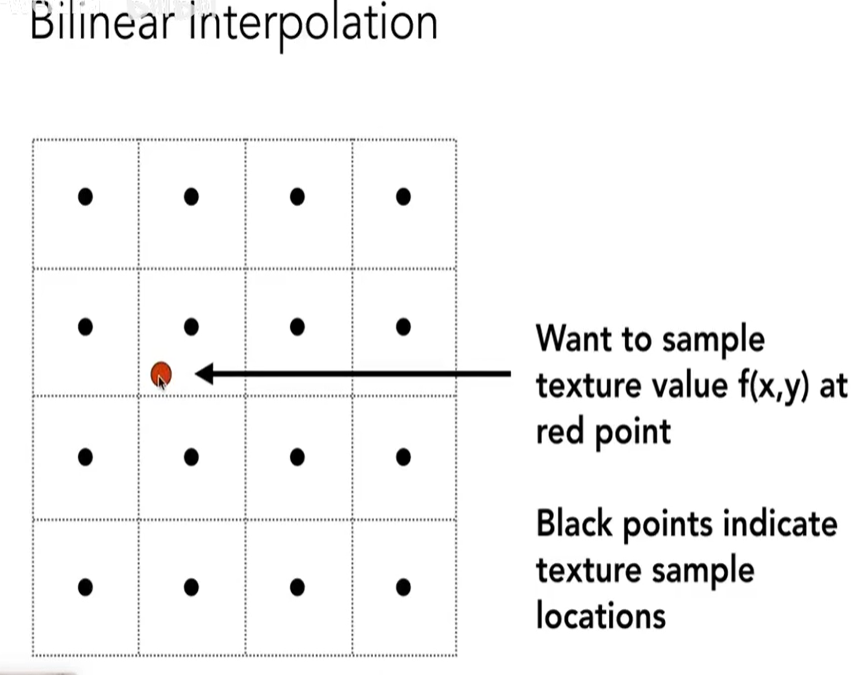
重心的系数如何计算：α：A点对面的小三角形和大三角形的比。β，γ类推。



重心坐标的问题：在投影下不能保证重心坐标不变

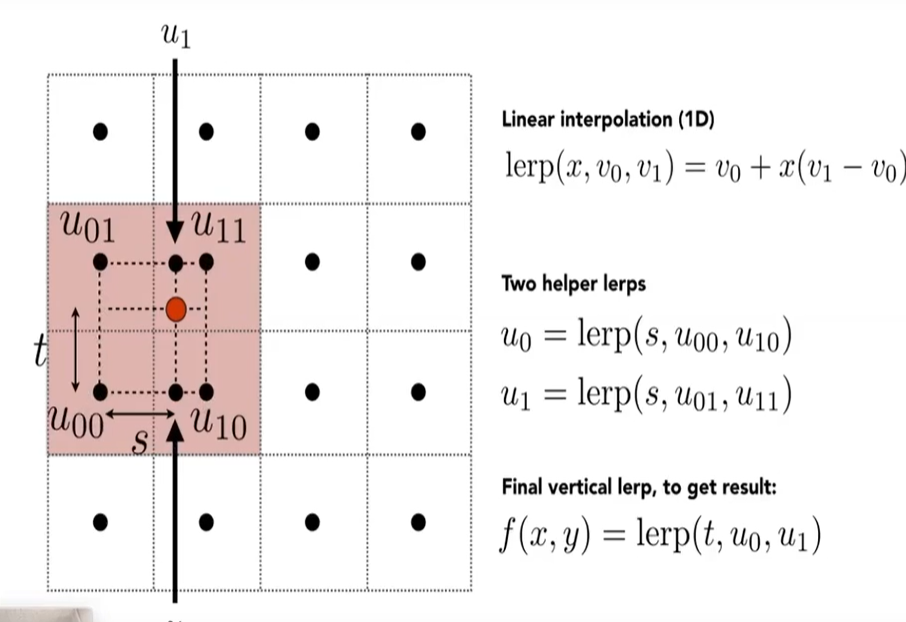
纹理小了，需要放大，采用双线性插值：

给定一个非整数坐标，如何求出纹理值？



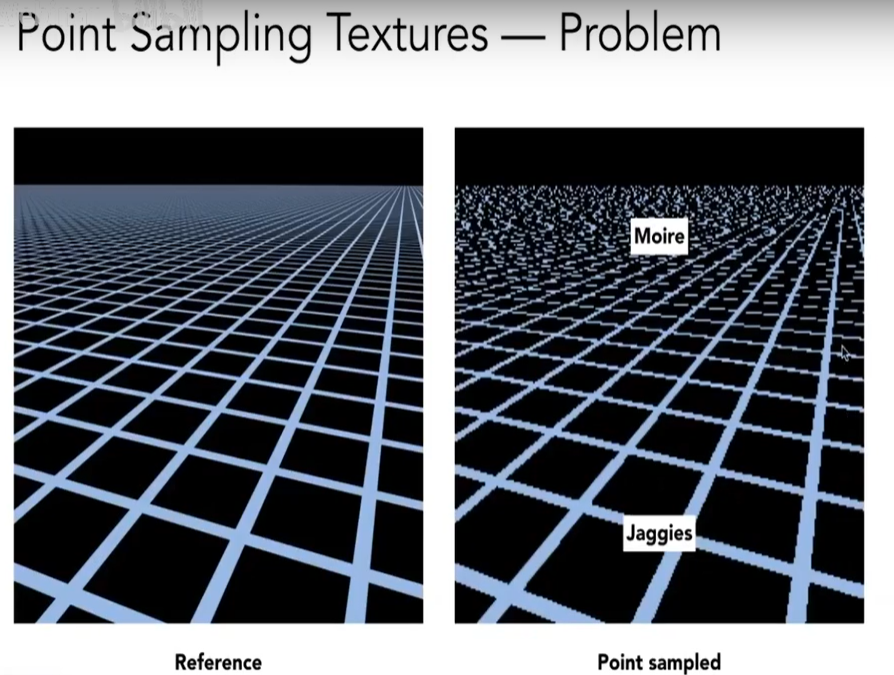
找他临近的4个点

双线性插值（水平竖直各做一次：



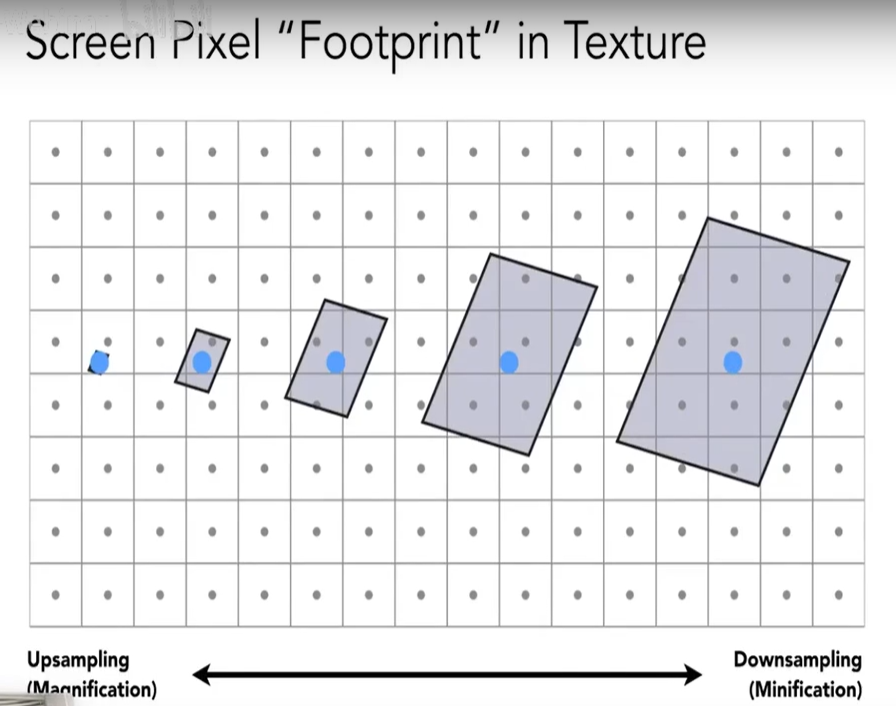
纹理大了，会有什么后果？

走样（远处摩尔纹，近处锯齿）



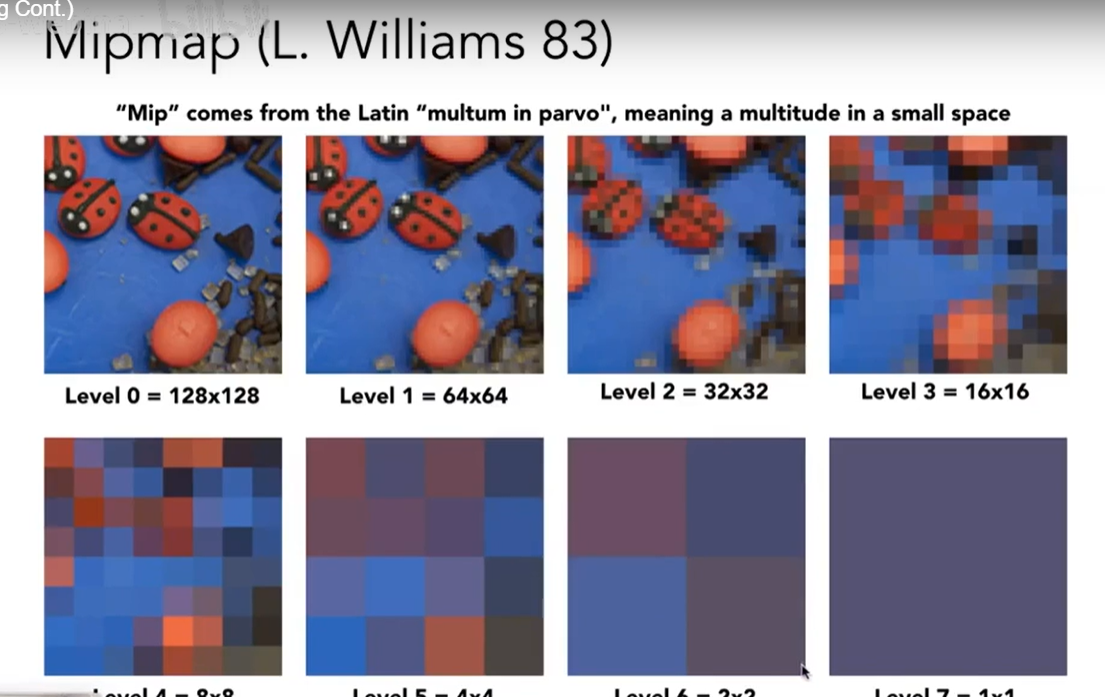
为什么？因为近处，一个像素覆盖的区域相对较小，而远处一个像素可能覆盖了一整个区域

，覆盖了纹理上很大一部分。用一个点如何代表一整块区域里颜色的变化？就不能简单地用像素中心去采样。

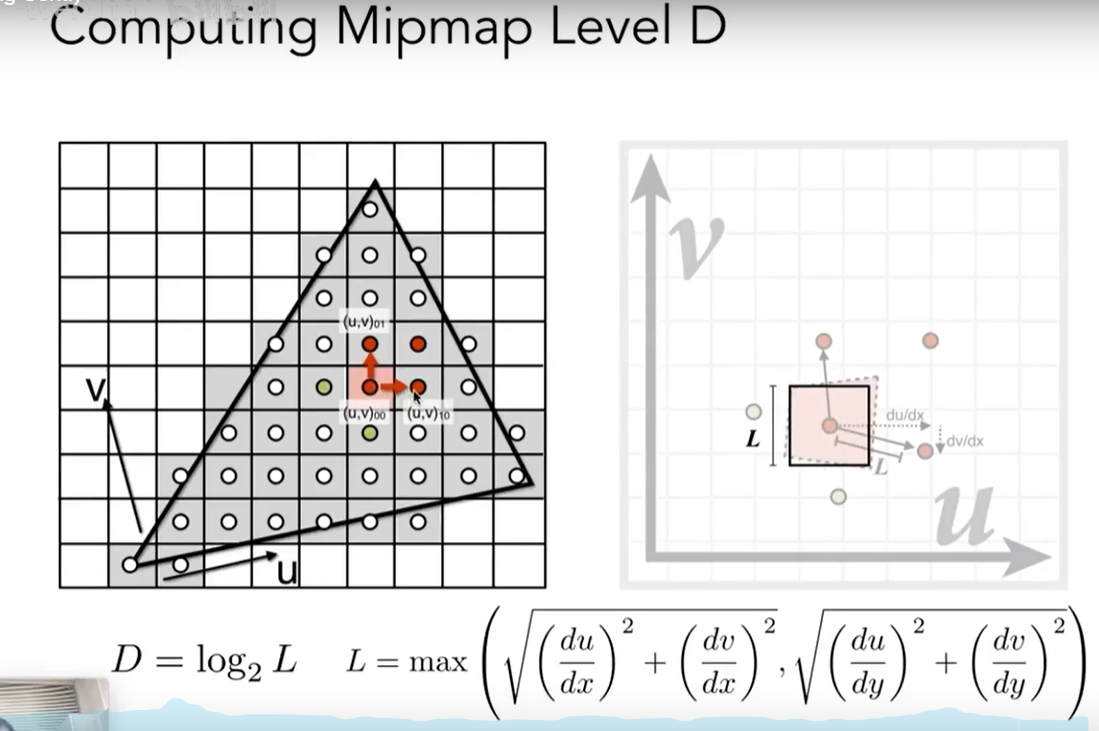


解决方法：mipmap，越远的就用越小的粗糙贴图

虽然多出了7个图，但这7个图的大小只是原图的1/3



那么，其他7个图里面，是怎么确定如何“缩略”像素的呢？



对每个点，求他上面，右边，右上角，一共四个点，分别映射到纹理上大概占据多长的距离，近似得到一个正方形。

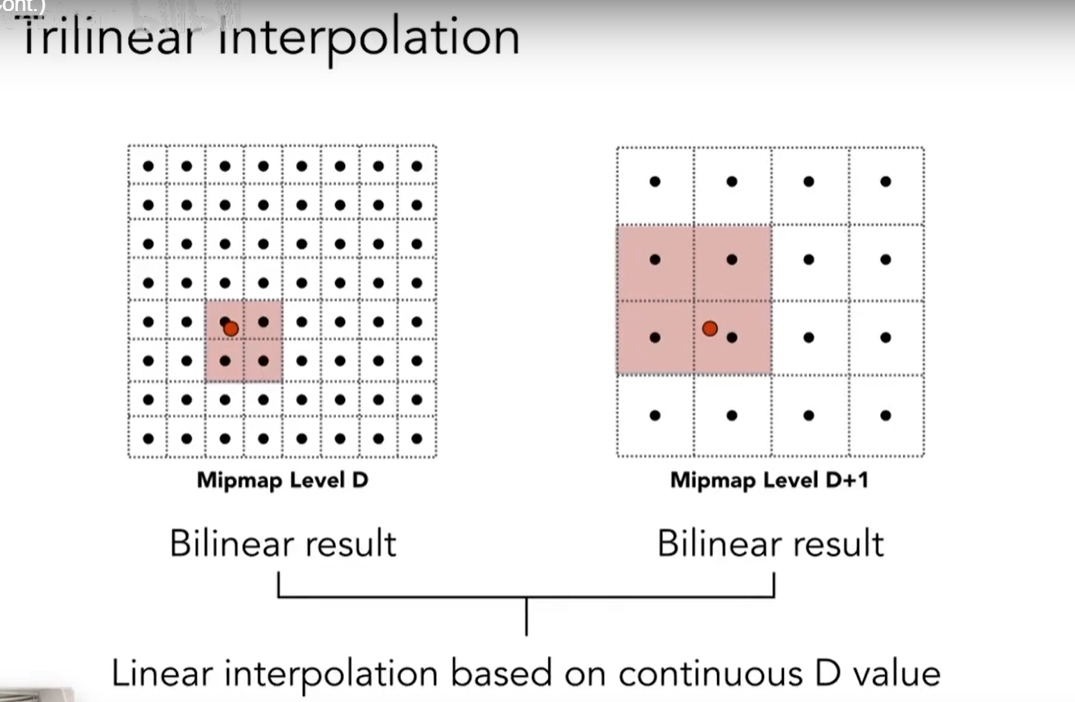
那么如何确定材质某一区域使用的是哪一层的mipmap？

假设这个区域大小是1x1个像素，那么他就是位于第一层（原大小）mipmap，也就是直接显示原来纹理上的那个像素。

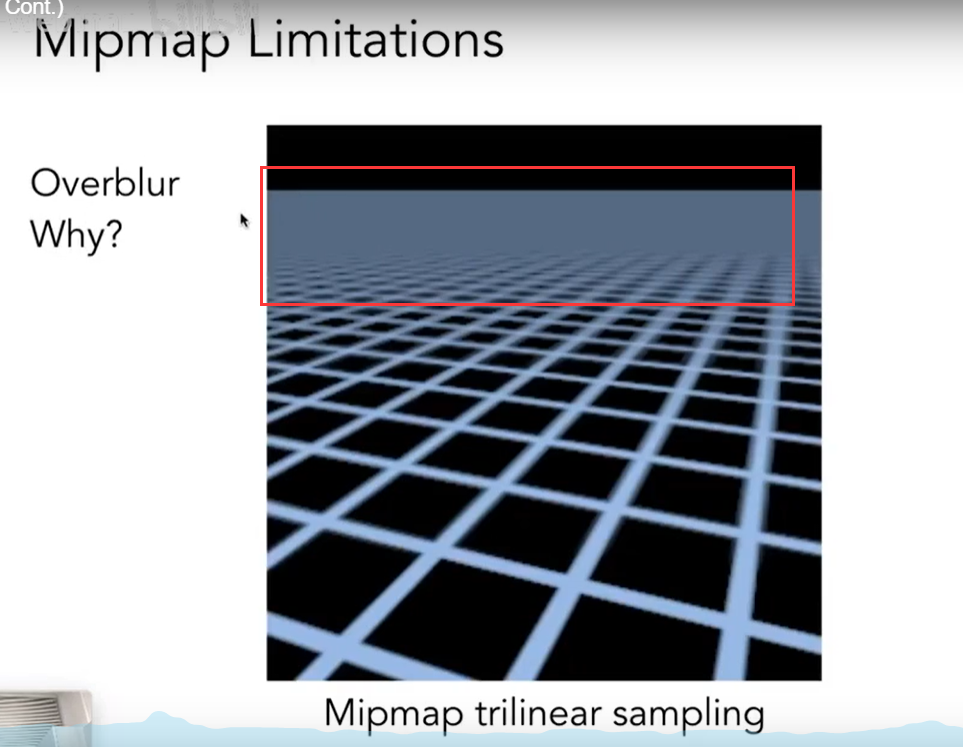
如果这个区域大小是4\*4个像素，那么经过一次mipmap之后，他会变成2\*2，经过第二层之后会变成1\*1.

由上面推导可知，**对于边长为L的区域，在log2L这一层，一定是只对应一个像素的。这样我们就可以通过查他在第几层变成1个像素的大小，立刻去查那个像素，就得到这个区域内的平均值了。**

进一步地，如果我想mipmap更加平滑，更具体地说，例如，我想实现“在xx位置贴图为mipmap第1.8层”这样的小数点层，如何解决？答案是**线性插值**。

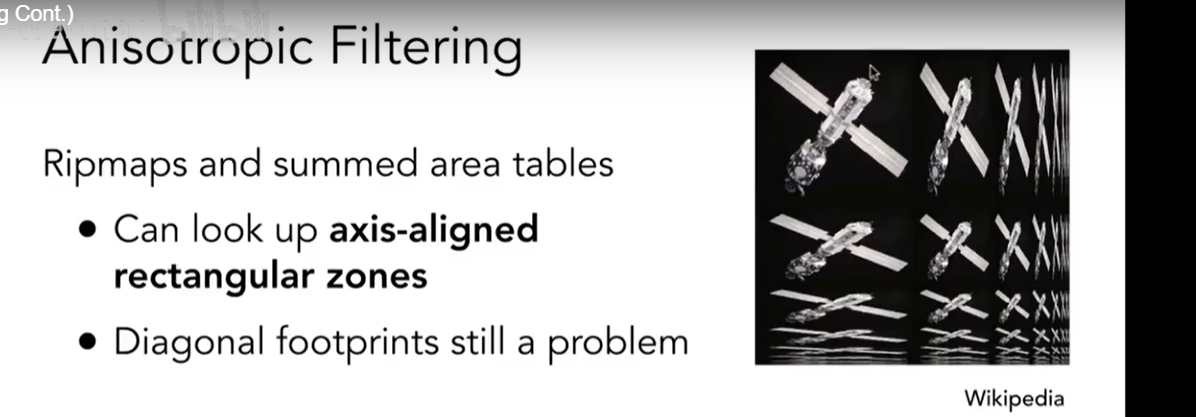


先做第D层的双线性插值，再做D+1的双线性插值，再将两个插值合在一起，做一次线性插值。这称作**三线性插值**。这样可以得到一个完全连续的材质远近变化。

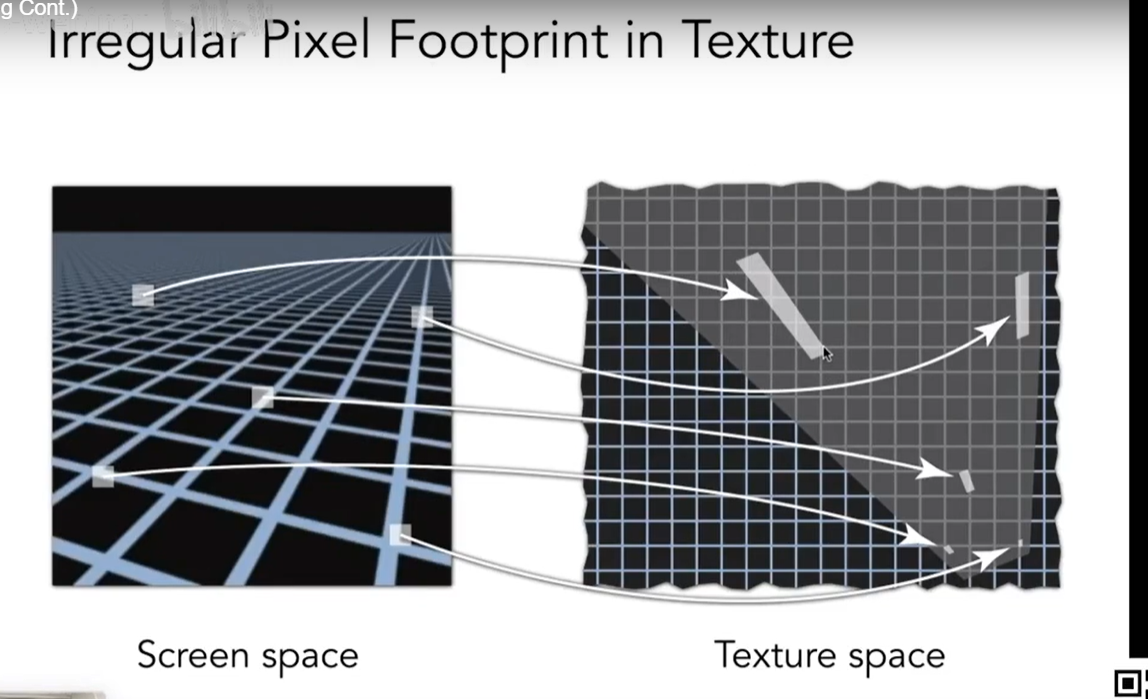


问题：mipmap远处会糊掉，overblur。

解决方法：各向异性过滤。



每一行，高度不变，宽度压缩；每一列，高度不变，宽度压缩



有的时候，屏幕中的一个点对应的可能是纹理上的一个矩形区域而不是正方形区域。各向异性过滤允许我们对长条形的纹理区域做快速的范围查询，而不用限制在正方形区域。

但是各向异性过滤对于斜矩形区域的查询仍然没有很好的解决

解决方法：**EWA过滤**。

把一个不规则的形状拆成多个不同的圆形

