# 5.10 光栅化阶段

光栅化（rasterization）阶段的主要任务是为投影后的3D三角形计算像素颜色。

## 5.10.1 视口变换

在裁剪之后，硬件会自动执行透视除法，将顶点从齐次裁剪空间变换到规范化设备空间（NDC）。一旦顶点进入NDC空间，构成2D图像的2D *x*、*y*坐标就会被变换到后台缓冲区中的一个称为视口的矩形区域内（回顾4.2.8节）。在该变换之后，*x*、*y*坐标将以像素为单位。通常，视口变换不修改*z*坐标，因为*z*坐标还要由深度缓存使用，但是我们可以通过**D3D11\_VIEWPORT**结构体的**MinDepth**和**MaxDepth**值修改z坐标的取值范围。**MinDepth**和**MaxDepth**的值必须在0和1之间。

## 5.10.2 背面消隐

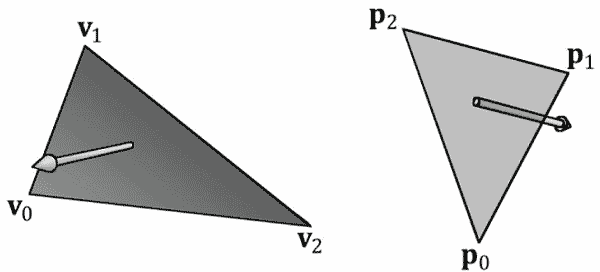
一个三角形有两个面。我们使用如下约定来区分这两个面。假设三角形的顶点按照**v**0、**v**1、**v**2的顺序排列，我们这样来计算三角形的法线**n**：

**e**0 = **v**1 - **v**0

**e**1 = **v**2 – **v**1



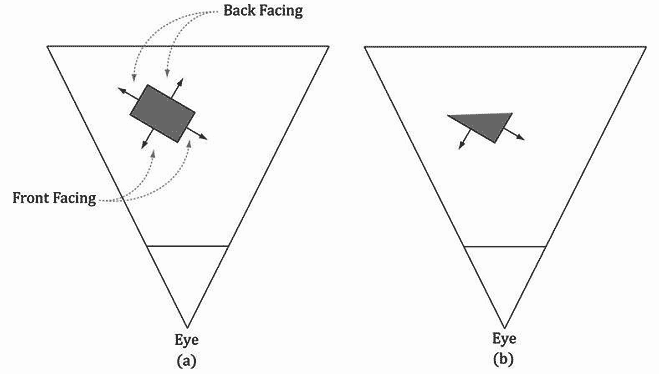
带有法线向量的面为正面，而另一个面为背面。图5.30说明了这一概念。

****

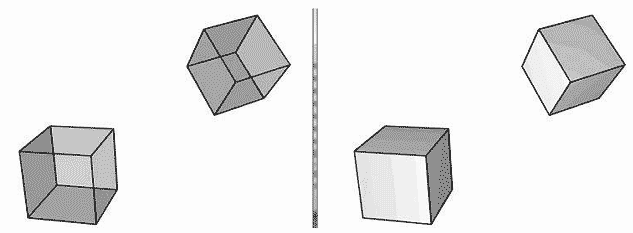
**图5.30 左边的三角形正对我们的观察点，而右边的三角形背对我们的观察点。**

当观察者看到三角形的正面时，我们说三角形是朝前的；当观察者看到三角形的背面时， 我们说三角形是朝后的。如图5.30所示，左边的三角形是朝前的，而右边的三角形是朝后的。而且，按照我们的观察角度，左边的三角形会按顺时针方向环绕，而右边的三角形会按逆时针方向环绕。这不是巧合：因为按照我们选择的约定（即，我们计算三角形法线的方式），按顺时针方向环绕的三角形（相对于观察者）是朝前的，而按逆时针方向环绕的三角形（相对于观察者）是朝后的。

现在，3D空间中的大部分物体都是封闭实心物体。当我们按照这一方式将每个三角形的法线指向物体外侧时，摄像机就不会看到实心物体朝后的三角形，因为朝前的三角形挡住了朝后的三角形；图5.31和图5.32分别以2D和3D形式说明了一概念。由于朝前的三角形挡住了朝后的三角形，所以绘制它们是毫无意义的。背面消隐（backface culling）是指让管线放弃对朝后的三角形的处理。这可以将所要处理的三角形的数量降低到原数量的一半。

****

**图5.31 (a)一个带有朝前和朝后三角形的实心物体。(b)在剔除了朝后的三角形之后的场景。注意，背面消隐不会影响最终的图像，因为朝后的三角形会被朝前的三角形阻挡。**

****

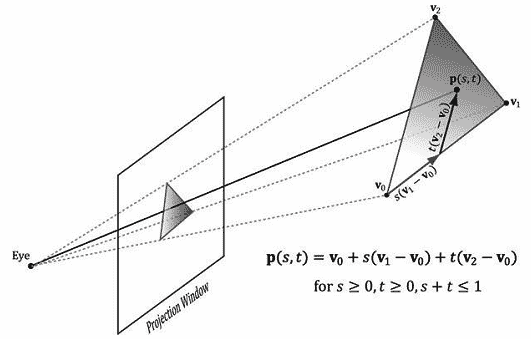
**图5.32 （左图）当以透明方式绘制立方体时，我们可以看到所有的6个面。（右图）当以实心方式绘制立方体时，我们无法看到朝后的3个面，因为朝前的3个面挡住了它们——所以朝后的三角形可以被直接丢弃，不再接受后续处理，没人能看到些朝后的三角形。**

默认情况下，Direct3D将（相对于观察者）顺时针方向环绕的三角形视为朝前的三角形，将（相对于观察者）逆时针方向环绕的三角形视为朝后的三角形。不过，这一约定可以通过修改Direct3D渲染状态颠倒过来。

## 5.10.3 顶点属性插值

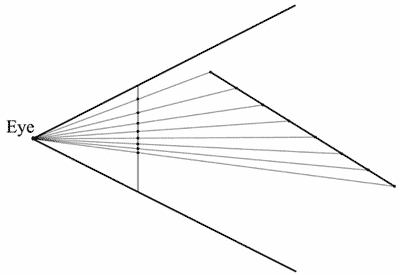
如前所述，我们通过指定三角形的3个顶点来定义一个三角形。除位置外，顶点还可以包含其他属性，比如颜色、法线向量和纹理坐标。在视口变换之后，这些属性必须为三角形表面上的每个像素进行插值。顶点深度值也必须进行插值，以使每个像素都有一个可用于深度缓存算法的深度值。对屏幕空间中的顶点属性进行插值，其实就是对3D空间中的三角形表面进行线性插值（如图5.33所示）；这一工作需要借助所谓的透视矫正插值（perspective

correct interpolation）来实现。本质上，三角形表面内部的像素颜色都是通过顶点插值得到的。

****

**图5.33：通过对三角形顶点之间的属性值进行线性插值，可以得到三角形表面上的任一属性值p(s,t)。**

我们不必关心透视精确插值的数学细节，因为硬件会自动完成这一工作；不过，有兴趣的读者可以在[Eberly01]中查阅相关的数学推导过程。图5.34介绍了一点基本思路：

****

**图5.34 一条3D线被投影到投影窗口上（在屏幕空间中投影是一条2D线）。我们看到，在3D线上取等距离的点，在2D屏幕空间上的投影点却不是等距离的。所以，我们在3D空间中执行线性插值，在屏幕空间需要执行非线性插值。**