第十二章　光照和着色

12.1　什么是着色

在之前的课程中我们已经学会了如何在计算机中表示三维的几何并且用线框的方式将其绘制到屏幕上，使得我们可以形象地理解三维几何．但是现实中的物体除了几何属性之外，还有**材质**（**material**）属性．比如下图所展示的这样，几何上同样是一个球，我们还是能立马判断出它们由不同的材质组成．

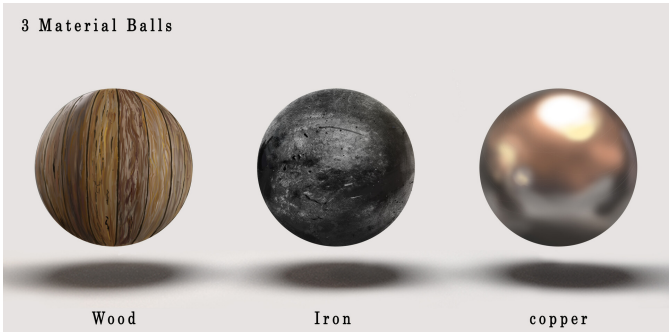


图 12.1: ©Joseph FS Leng, Art Station

我们能认出这三个球，是因为第一个球有木头的纹理，第二个球有凹凸不平粗糙的铁锈，第三个表面光滑能够反射出场景的颜色，并且顶上有金属状的高光．物体本身的颜色、纹理、粗糙度等等属性，决定了物体在光照下反射出的颜色，也就决定了一个物体在我们眼中的样子．在计算机图形学中，为几何体的表面加上材质的过程就是**着色**（**shading**），或者我们也可以说着色是绘制几何体表面的颜色使得它看起来像我们设定的材质的过程．

12.2　光照

一个物体除非自己发光，否则都需要反射来自外界的光线进入人眼才能被我们看到，为了之后讨论方便，我们首先研究一下源头的光照，也就是**光源**．

平行光　生活中最常见的光源就是太阳．因为太阳离地球很远，所以来自太阳的光线可以近似当成**平行光**（**directional light**）．我们可以用光照的方向和强度这两个量来描述平行光．其中方向表示为世界坐标中的归一化三维向量 **d**，强度定义为*垂直于光入射方向单位面积上接收到的光照射的功率*．这里我们强调光强需要在光的垂直方向测量，如果接收光的表面与入射光线之间存在夹角 θ，如图12.2所示，相同的平面面积接收到的光强与 θ 相关．由于平面面积投影到光的垂直方向需要乘以一个因子 $\cos\theta$，所以图12.2右边的平面接收到的光强就是左边的 $\cos\theta$ 倍，也就意味着光强比左边直射时更小．如果平面与光的方向完全

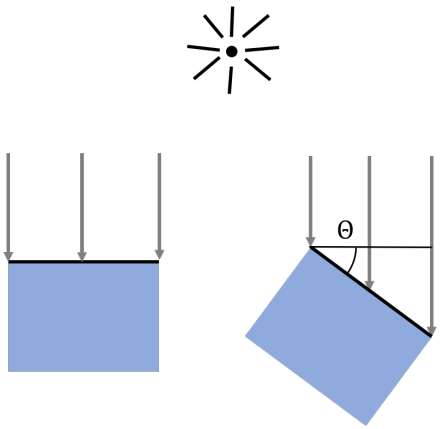


图 12.2: 平行光

平行，$\cos\theta = 0$，表示接收到的光强为 0，这也符合我们的直觉．我们用一个颜色 I_d 表示平行光的光强，其三个分量表示 RGB 三个频率范围内的光强分量，这样当入射光线与平面法线之间的夹角是 θ 时，平面接收到的光强就是 $I_d \cos\theta$.

环境光　在晴朗的白天，我们能清晰地看到由平行光照射产生的阴影，阴影的外面由平行光直接照射，阴影内由于物体阻挡不能被平行光照射．但是在阴影的暗部也不是完全黑暗的，这一部分区域就是由**环境光**（**ambient light**）照亮的．不同于平行光，环境光来自四面八方，是太阳光经过云层大气的散射以及其他物体的反射得到的，因此环境光是一种**间接光照**（**indirection illumination**）．在最简单的模型中，我们可以认为环境光在各个方向是均匀的，于是只需要一个光强 I_a 就能定义环境光．

点光源　除了太阳这一自然光源，生活中也有很多像蜡烛、灯泡、吊灯这样的人造光源．其中蜡烛、灯泡等可以近似描述为**点光源**（**point light**）．点光源没有方向性，但是与环境光不同，点光源的强度会随着距离衰减．在远处接收到的光强要远远小于光源中心附近的光强，这个衰减的比率可以计算得到是 $1/r^2$，r 表示观测点与点光源的距离．

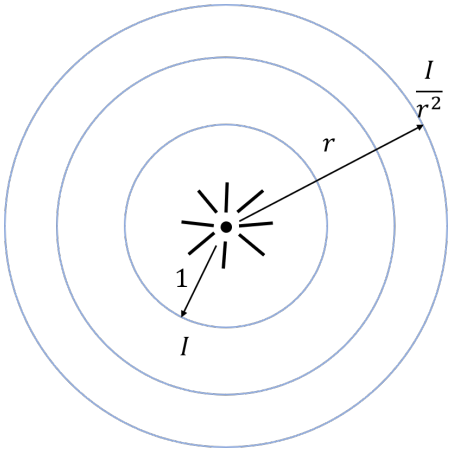


图 12.3: 点光源

我们可以形象地理解为什么衰减比率随距离平方成反比．如果我们用一个以点光源为中心，半径为 r 的球罩住点光源，那么这个球将接受所有来自点光源的能量．球的表面积

是 $4\pi r^2$，由于对称性球球表面各处单位面积接收的光功率相等，都是总功率除以表面面积，于是距离 r 处的光强自然与 r^2 成反比．描述点光源，我们需要记录点光源的位置 $\mathbf{p}$，以及单位长度处的强度 I_p，这样在距离为 r 的地方接收到的光强就能表示为 I_p/r^2．

环境光、平行光、点光源因其数学描述上的简洁是三种图形学中最常用的光源，除此之外还有面光源、体光源、聚光灯等等．这些光源在描述上更为复杂，但也能产生更为复杂美妙的光影效果．事实上，任何复杂的面光源、体光源都可以看成由无数点光源组成，所以理论上我们只需要把复杂光源分解为很多个点光源就能实现复杂光源的照明．但是在算力的约束下我们不能这么暴力地去做，所以我们仍然需要细致考虑不同光源的光照模型，这里我们就不再深入讨论了．

12.3　反射模型

解决完光照之后，我们再来研究一下物体反射光线的数学模型，更具体的来说，我们需要研究光线经过物体表面反射之后入射到摄像机中的光强．与研究光照类似，我们先考虑几个简单情况下的反射规律．

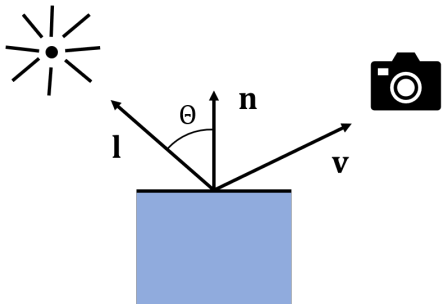


图 12.4: 漫反射

漫反射　当物体的表面极度粗糙不平时，入射到表面上的光线会被反射到各个方向，称之为**漫反射**（**diffuse reflection**）．当物体表面只有完全随机的漫反射时，称其为一个**朗伯体**（**Lambert body**）．朗伯体的特点在于其在各个角度看起来反射的光强都是相同的，这符合我们日常生活中对于粗糙表面的直觉．其实如果我们仔细观察太阳，会发现太阳边缘和中心的亮度是一样的，整个像一个均匀发光的圆盘，这其实也是因为太阳发射光线满足朗伯体的条件，称之为朗伯发光体．于是对于完全的漫反射而言，其反射到摄像机中的光强可以写为：

$$L_d = k_d(I_a + I_d \max(0, \mathbf{n} \cdot \mathbf{l})), \tag{12.1}$$

其中 k_d 是表面的漫反射颜色，$\mathbf{n}$ 是表面法向，$\mathbf{l}$ 是光源的方向，如图12.4所示．这里我们进一步解释一下公式的涵义：

1. k_d 是表面漫反射的颜色，尽管表面在空间角度上均匀反射入射光线，但是在频率空间上并不是均匀的，有的频率被表面吸收，有的频率被反射，漫反射的颜色也就表示表面在 RGB 三个分量上的反射系数．
2. I_a 和 I_d 分别表示环境光和平行光的强度，存在点光源时只需要把 I_d 替换为 I_p/r^2 即可．
3. $\mathbf{n} \cdot \mathbf{l}$ 得到的就是 $\cos\theta$，也就是我们在前面平行光部分介绍的表面与入射光线方向不垂直带来的系数，与 0 求 max 表示只有当光源在法向一侧时表面才能被照射到，否则就处在阴影中．

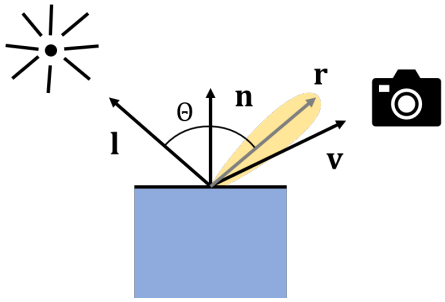


图 12.5: 镜面反射

镜面反射　与漫反射相对的就是**镜面反射**（**specular reflection**）．理想的镜面反射满足反射定律：光线的出射角等于入射角，如图12.5中的 $\mathbf{l}$ 和 $\mathbf{r}$ 所示．理想的镜面反射要求物体表面非常光滑，一般情况下我们可以认为反射光线分布于理想反射光线 $\mathbf{r}$ 附近的圆锥内，中心最强，边缘比较弱．这样的结果就是我们可以在理想镜面反射的周围看到一圈高光，比如图12.1最右边球上的高光．那么如何用数学表示反射光的分布呢？我们可以自然想到衡量 $\mathbf{v}$ 和理想镜面反射光线 $\mathbf{r}$ 的接近程度，给观察到的光强乘以一个对应的衰减因子，在 $\mathbf{v}$ 与 $\mathbf{r}$ 重合时为 1，在 $\mathbf{v}$ 和 $\mathbf{r}$ 距离很远的时候为 0．这听起来很好，但是实现的时候会碰到一个比较麻烦的问题：如何计算 $\mathbf{r}$？我们需要将 $\mathbf{l}$ 分解到 $\mathbf{n}$ 的方向和垂直于 $\mathbf{n}$ 的方向，延 $\mathbf{n}$ 的方向不变，垂直于 $\mathbf{n}$ 的方向反号，这样就能得到 $\mathbf{r}$．这样听起来并不复杂，但是我们使用一个技巧来规避掉直接求解 $\mathbf{r}$ 从而简化计算．

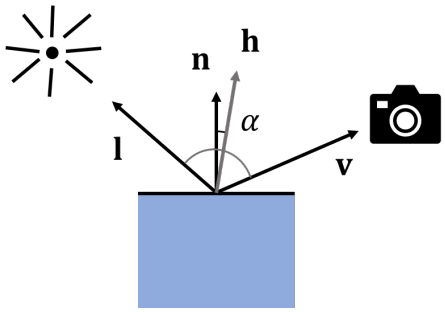


图 12.6: 半程向量

如图12.6所示，与其衡量 $\mathbf{r}$ 与 $\mathbf{v}$ 之间的差距，我们定义一个半程向量 $\mathbf{h} = \text{normalized}((\mathbf{l}+\mathbf{v})/2)$，并且计算 $\mathbf{h}$ 与 $\mathbf{n}$ 之间的夹角 α．当 $\mathbf{v}$ 正好沿着 $\mathbf{r}$ 的方向时，半程向量 $\mathbf{h}$ 应该正好沿着 $\mathbf{n}$ 方向，而随着 $\mathbf{v}$ 逐渐远离 $\mathbf{r}$，α 也在逐渐增大．于是 α 可以正确度量 $\mathbf{v}$ 和 $\mathbf{r}$ 之间的差异，同时 $\mathbf{h}$ 在计算上要更简单，所以我们可以用 α 来计算光强衰减的因子．这个因子要求在 $\alpha = 0$ 时取到 1，在 α 增大的时候的逐渐减小，于是 $\cos\alpha$ 就是一个自然的选择，并且 $\cos\alpha = \mathbf{n}\cdot\mathbf{h}$ 同样容易计算．这样我们就可以给出镜面反射的公式：

$$L_s = k_s I_d \max(0, \mathbf{n}\cdot\mathbf{h})^p. \tag{12.2}$$

公式里 k_s、I_d、max 的含义与漫反射公式12.1中一致，注意到在公式的最后多乘了一个 p 次方，这个参数 p 是用来控制镜面反射的集中度的．如图12.7所示，当 p 比较小时 $\cos^p\alpha$ 衰减比较慢，表示光线比较分散，表面比较粗糙；当 p 比较大时，$\cos^p\alpha$ 衰减快，表示光线集中，表面比较光滑．

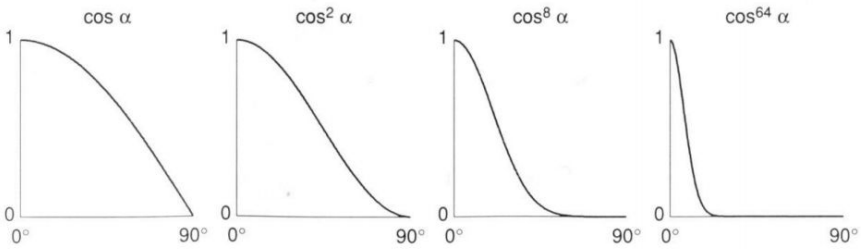


图 12.7: 不同 p 的比较

Blinn-Phong 反射模型　一般的物体我们可以认为同时存在漫反射和镜面反射，结合公式12.1和12.2我们就得到了一般物体的反射模型，称为 **Blinn-Phong 反射模型**，这是美国计算机科学家 James F. Blinn 在 1977 年改进美国越南裔学者 Bui-Tuong Phong 的光照模型提出的 [1]：

$$L = k_d(I_a + I_d \max(0, \mathbf{n} \cdot \mathbf{l})) + k_s I_d \max(0, \mathbf{n} \cdot \mathbf{h})^p. \tag{12.3}$$

至于 Blinn-Phong 之前的 Phong 模型，其实就是前面介绍的在计算镜面反射时计算视线与反射光线之间的夹角，而非半程向量与法线之间的夹角．到这里我们就得到了最简单一般物体的光照模型，不同的材质通过 k_d、k_s 和 p 这三个参数体现．仅依靠这三个系数当然不能完整描述所有材质的光学属性，但是我们先不妨进行下一步，看看用这样简单的光学模型能够实现怎样的效果．

我们并没有从严格的物理出发导出这些公式，而是直接给出了这些看起来非常直接的结果．从历史上来看，这些公式确实就是先被以结果为首要目标的图形学研究者们尝试出来，然后再被更物理真实的模型所统一的．尽管如此，你依然可以问自己一些细节问题，比如为什么公式 12.2 与公式 12.1 相比少了入射角 $\cos\theta$ 一项的贡献？

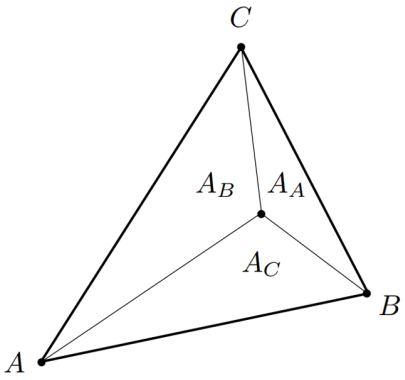
12.4　光栅化

与绘制 2D 图形一样，我们通过**光栅化**来将虚拟的几何和材质转化为屏幕上的像素．我们的几何表面是一堆三角形面片组成的网格，那么与之相对应的我们就需要解决两个问题：一是如何确定每个三角形面片上每个像素的颜色，二是如何处理三角形之间的遮挡关系．我们先来关注第二个问题．

在 2D 的图形的绘制中，我们介绍过使用**深度缓存**来处理三角形之间的遮挡．在每个像素中我们除了记录颜色信息，同时还记录了深度信息；每个三角形都需要进行逐像素的深度检测，最终只保留深度最小的颜色．这个算法应用到 3D 看起来非常直观，但是我们忽略了一个可能导致错误的问题：如何确定每个像素的深度？因为我们知道每个像素在屏幕空间中的位置，也知道每个顶点经过变换后的平面位置和深度，最直接的想法就是利用它们在屏幕空间中的位置通过插值得到深度，使用我们在之前二维图像绘制中介绍的双线性插值的方法，或者等价的使用**重心坐标**．

如图12.8所示，对于三角形中的任意一点，我们可以使用其分割出的三个小三角形之间的比值计算出其相对于三个顶点的权重．具体来说，相对于点 A 的权重 α 是与 A 点不相邻的小三角形与大三角形的面积比值．我们可以很容易地验证，当顶点位于点 A 时，满足 $\alpha = 1, \beta = \gamma = 0$，$\alpha + \beta + \gamma = 1$ 始终满足．并且由于面积随着顶点位置线性变化，三个权重也会随着位置线性变化．因此，顶点上的深度、颜色等信息，可以使用 α、β、γ 三个权

重加权平均三角形顶点上的值得到.



$$\alpha = \frac{A_A}{A_A + A_B + A_C}$$
$$\beta = \frac{A_B}{A_A + A_B + A_C}$$
$$\gamma = \frac{A_C}{A_A + A_B + A_C}$$

图 12.8: 投影变换与插值

然而，不管我们使用什么在屏幕空间的线性插值方法，得到的深度是**不准确**的，我们可以通过图12.9中简单的例子分析一下为什么.

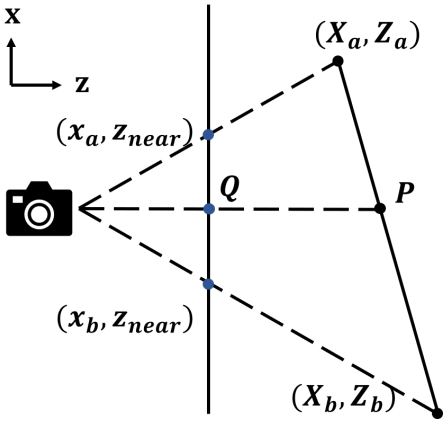


图 12.9: 投影变换与插值

如图12.9所示，我们考虑二维情况，这时相机的近平面就是一条直线，$(\mathbf{X}_a, \mathbf{Z}_a)$ 到 $(\mathbf{X}_b, \mathbf{Z}_b)$ 之间的线段经过投影变换变换到了 $(\mathbf{x}_a, \mathbf{z}_{\text{near}})$ 到 $(\mathbf{x}_b, \mathbf{z}_{\text{near}})$. 假设 $\mathbf{x}_a$ 和 $\mathbf{x}_b$ 上下对称，我们考虑它们中点像素 Q 的深度. 如果我们直接用屏幕空间的 Q 点做插值那么就会得到 Q 点的深度是 $(\mathbf{Z}_a + \mathbf{Z}_b)/2$. 然而 Q 点的深度应该由其在原线段上的点 P 决定，P 显然并不位于原线段的中点上，因此真实的 Q 点的深度并不是 $(\mathbf{Z}_a + \mathbf{Z}_b)/2$，而应该是对 $\mathbf{Z}_a$ 的权重大于 $1/2$，对 $\mathbf{Z}_b$ 的权重小于 $1/2$. 发生这个错误的原因在于透视投影变换是非线性的，重心坐标依赖于三角形的面积（二维中是长度），而透视投影并不能保证所有点面积均匀缩放. 正确的做法是在透视投影前做插值，我们需要反求每个像素在投影前的位置，然后在三维空间中线性插值. 这样做能保证正确性，但是如果每一个像素都需要这样做一遍，计算的开销就比较大了. 事实上，如果我们进一步分析透视投影的性质，我们可以得到一个修正的插值方法，只用很小的代价并且只在屏幕空间进行，称为**透视矫正插值**（**Perspective-Correct Interpolation**）. 假设我们需要对三角形的三个顶点上的值 f_a, f_b, f_c 进行插值，透视矫正投影的公式表述为：

$$f = \frac{\frac{\alpha}{w_a} f_a + \frac{\beta}{w_b} f_b + \frac{\gamma}{w_c} f_c}{\frac{\alpha}{w_a} + \frac{\beta}{w_b} + \frac{\gamma}{w_c}}, \tag{12.4}$$

其中 α, β, γ 是使用屏幕坐标求得的重心坐标，w_a, w_b, w_c 是透视投影之后得到的齐次坐标的第四个分量．这个公式表明透视投影对于插值的影响只是在重心权重上多乘了一个 $\frac{1}{w}$ 的因子．参照透视投影的矩阵公式，变换后齐次坐标的第四个分量 w 实际上就是顶点在变换前的深度 z．公式12.4的具体证明请参照这篇文章[3]．将 f_a, f_b, f_c 替换为各个顶点的深度 z，w 也替换为 z，再对公式12.4稍作变换就能得到透视矫正下像素深度的插值公式：

$$\frac{1}{z} = \frac{\alpha}{z_a} + \frac{\beta}{z_b} + \frac{\gamma}{z_c}. \tag{12.5}$$

这个公式直观地告诉我们，如果使用透视矫正的方法插值深度，等价于在屏幕空间对 $1/z$ 进行线性插值，这也说明了我们之前对深度 z 直接进行线性插值是错误的．

在使用正确的方法插值得到每个像素的深度后，我们就可以使用前面介绍的深度缓存方法正确处理遮挡关系．然而深度缓存技术也不是万能的，最大的问题在于计算的代价．为了绘制一个大场景，我们需要对每个三角形都计算深度之后才知道这个三角形是否会绘制到屏幕上，在三角形数量很多时计算代价比较高．为了减少判断次数，最常用的方法就是背面剔除（*Back Face Culling*）．如果在场景中我们绘制的都是封闭的几何体，那么显然几何体的背面是永远不会出现在屏幕上的．因此，我们只需要判断三角形的法向与视线方向是否同向，也就是 $\mathbf{v} \cdot \mathbf{n} > 0$．如果同向，那么这个三角形就是背面，我们也就不用费力再对它判断深度缓存和着色了．

通过研究三角形遮挡关系的处理，我们顺便解决了像素插值的问题．下面我们再回头尝试解决最后一个问题：决定每个像素的颜色，也就是确定**着色模型**．

12.5　着色模型

通过前面光照和材质的介绍，在给定光照、物体几何和材质之后，决定像素颜色最关键的就是表面的法向．第一种着色方法就是使用三角形的面法向，逐三角形确定颜色，这种着色方法称为**平面着色**（**Flat Shading**），如图12.10所示．

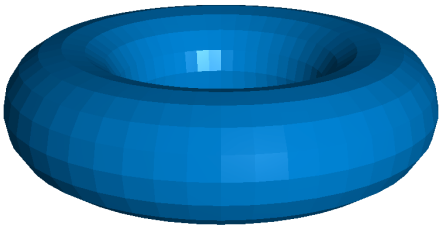


图 12.10: 平面着色 ©From Wikipedia

在平面着色中，每个三角形只有一个法向，每个三角形只有一个颜色，于是我们能在渲染结果里看到很多独立的面片，但是我们还是可以直观看到整个形状上明暗的过渡以及高光．如果我们的三角面片足够多，我们就能得到一个光滑的表面，但是如何在三角形数量不够的时候依然让表面看起来光滑呢？这就引入了第二种着色方法：计算每个顶点的颜色，然后在三角形中间插值．这种着色方法称为 **Gouraud 着色**（**Gouraud Shading**），是由法裔计算机科学家 Henri Gouraud 在 1971 年提出 [2]，结果如图12.11所示．

在 Gouraud 着色中，我们不再是逐三角形着色，而是逐顶点着色．我们使用每个顶点的顶点法向计算光照颜色，然后在三角形内部用重心坐标进行插值．从结果可以看到，在曲

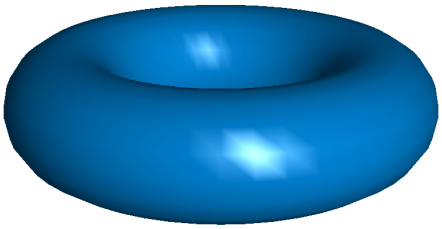


图 12.11: Gouraud 着色 ©From Wikipedia

面的大部分区域里我们得到了光滑的颜色过渡，但是在高光区域我们能明显看到三角形插值的痕迹．同样，如果我们增加三角形的数量，我们依然能够得到光滑的渲染结果，但是有没有方法能够在当前的精度下进一步减小离散化带来的误差？我们首先思考一下顶点颜色插值的合理性．如果三角形近似的曲面正对着光源方向，那么三角形中心的高光应该要强于顶点处的高光，这表明三角形中心的颜色并不能写成顶点颜色的线性插值，因为插值出来的亮度不可能强于被插值的顶点的亮度．因此对颜色插值并不是一个好的策略，我们需要考虑三角形内部颜色随位置的非线性变化．既然不能通过插值得到颜色，我们就只有对三角形上的每一个点都计算一个颜色，这样我们的着色方案就不再是逐顶点的，而是逐像素的．这种着色方法称为 **Phong 着色**（**Phong Shading**）, 由美国越南裔学者 Bui-Tuong Phong 在 1973 年提出 [4]，如图12.12所示．注意这里的 Phong 着色模型不同于前面的 Blinn-Phong 反射模型，前者描述的是着色方法，后者描述的是光照模型．

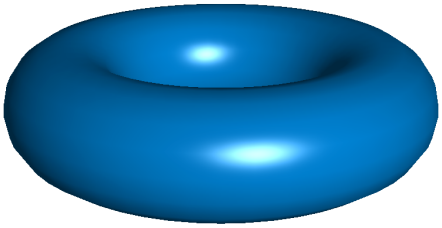


图 12.12: Phong 着色 ©From Wikipedia

在 Phong 着色中我们需要单独计算每个像素的颜色，也就需要每个像素的法向．像素的法向与深度一样，需要在透视投影之前插值才能得到正确结果，因此我们需要使用公式12.4进行透视矫正的重心坐标插值，同时在插值之后还需要归一化．从图12.12可以看出，Phong 着色能够正确画出椭圆形的高光区域，不再有 Gouraud 着色的不自然高光了．

从平面着色到 Phong shading，我们发现我们需要计算的颜色数量在增加，从逐三角形到逐顶点再到逐像素，换句话说就是**着色频率**在增加，效果也在逐渐接近真实．由于 Phong 着色需要对每个三角形的每个像素计算光照，因此公式12.3和公式12.5会被频繁调用，这也是我们如此强调它们的计算效率的原因．至此我们已经介绍完了最简单的绘制三维物体表面的方法：光栅化 + 深度缓存 +Phong 着色，这是今天大部分实时图形应用的基础．

12.6　风格化渲染

如图12.13所示，我们展示了不同艺术风格的渲染结果，这些风格服务于不同的目的，比如游戏、教育、工业等等，统称为风格化渲染，或非真实感渲染（*Non-Photorealistic Rendering*,

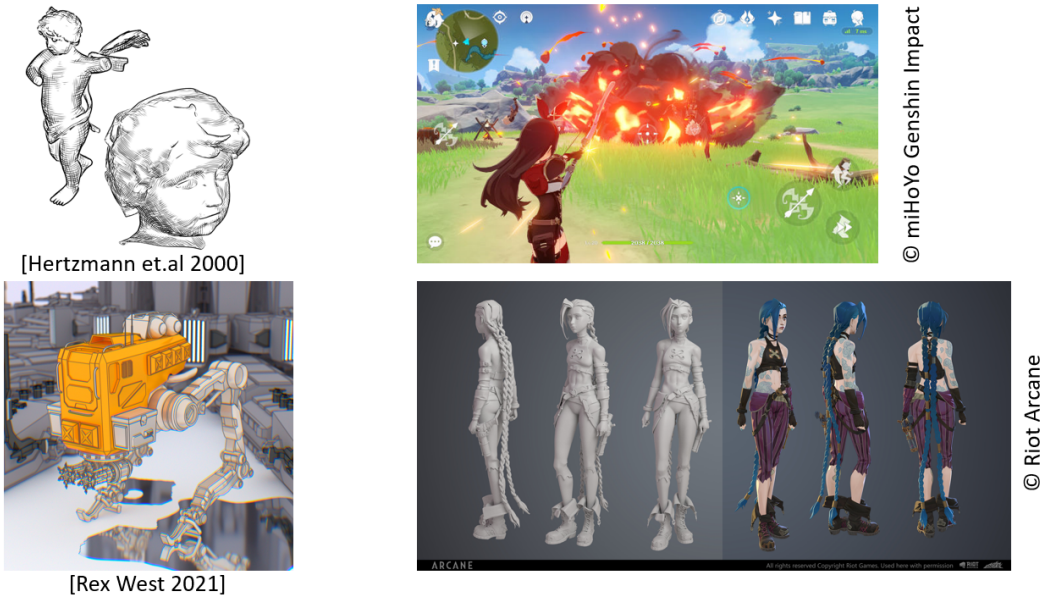


图 12.13: 风格化渲染

NPR）．由于不同的场景下渲染的质量要求和艺术效果差别巨大，风格化渲染的方法也多种多样．这里我们只在前面介绍的渲染框架内，介绍最简单的实现一种风格化的方法．

从图12.13中我们可以总结出风格化渲染的两个非常重要的特征．一是线，包括物体边缘的轮廓线，表示结构的结构线，表示光影的阴影线等等．就像画素描中使用的线一样，我们希望使用渲染的方法能够自动生成这些不同的线．第二点是艺术化的着色．在上面介绍的 Blinn-Phong 反射模型中，每个点的颜色由光源的颜色、漫反射的颜色、镜面反射的颜色等属性所决定．而在风格化渲染中，我们往往希望颜色有不同的艺术效果，比如卡通风格、冷暖色调、油画风格等等．基于这两点，下面我们举一个例子介绍如何在渲染管线中实现如图12.14的艺术效果．

图 12.14: 简单的风格化渲染结果

首先，我们需要绘制出物体的轮廓线．这一步有非常多的做法．第一种方法是我们可以通

过物体的法线方向来判断. 物体的边缘位置法线应该正好与视线方向垂直，也就是 $\mathbf{v}\cdot\mathbf{n}\approx 0$. 因此我们可以判断当着色点的法向方向与视线方向点乘在 0 附近时，就绘制边缘线的颜色. 图12.15展示了选择不同的判断范围时的绘制结果. 我们可以发现一个很明显的问题，这种方法虽然简单，但是得到的边缘性的粗细并不是一致的，如果判断范围过小，就会出现边缘线断裂，如果过大就看起来不再是一条线.

图 12.15: 通过法线绘制边缘线

第二种方法避免了边缘性粗细不一致的问题，称为程序化几何法. 在这个方法种我们绘制两遍几何体，第一遍只绘制几何体的背面，绘制颜色为边缘线的颜色，并且将几何体稍微向外扩展一点，第二遍再在背面之上绘制正面的几何体，这样没有被遮挡的部分就是边缘线了. 在这个方法中我们需要注意两点，第一点是在扩展背面的几何体时，我们需要每个顶点在法向上移动的距离在屏幕上最终是等宽的，不然还是会出现边缘线不等宽的现象. 因此在计算移动距离时，我们需要考虑投影变换的影响. 第二点是在绘制正面的几何体时，我们需要使用深度缓存，不然的话我们就可能丢失一些物体上的轮廓线（比如茶壶盖的轮廓线）.

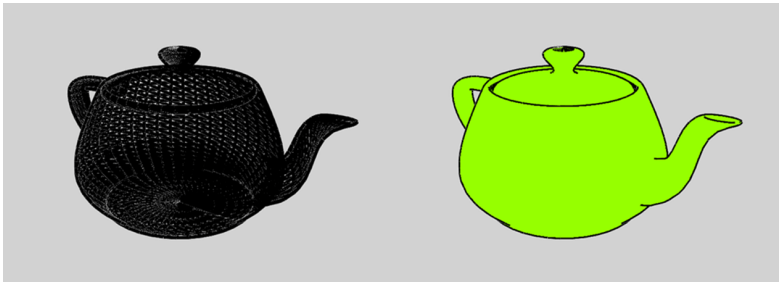
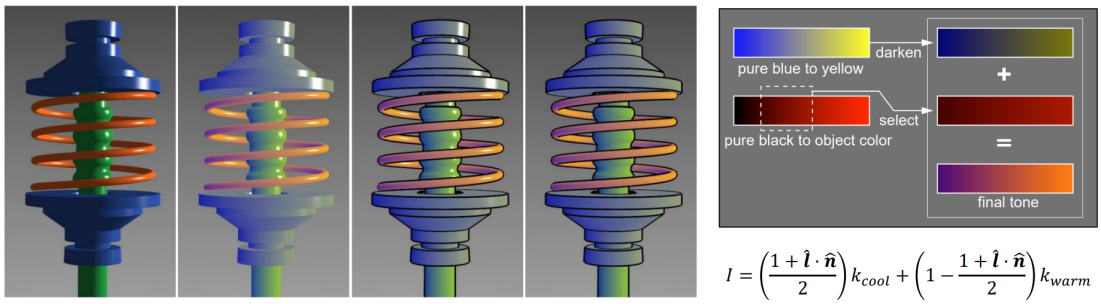


图 12.16: 程序化几何方法绘制边缘线

除此之外，我们还可以在正常绘制结束之后，再从结果的图像中提取出边缘. 提取边缘的方法可以使用前面图像章节介绍的卷积方法，也有更高级的基于神经网络的方法等等. 而为了避免颜色的干扰，我们也可以单独渲染一张法线图，提取出边缘线之后再跟正常图像合成.

解决了线的问题之后，我们接下来看着色. 在图12.14中，物体的体积感并不是通过光影来塑造的，而是通过颜色的冷暖塑造的. 由于冷暖色并没有明显的亮度区别，因此我们可以同时观察到物体的各个部分的细节，同时保有立体感，因此这种着色方法经常用于科学插图、工业设计中. 这种着色方式称为 Gooch 着色，或者冷暖色着色.

如图12.17所示，在 Gooch 着色中，我们给定了一个冷色 k_{cool} 和暖色 k_{warm}，物体上的颜色通过在这两个颜色之间插值得到. 插值的系数由物体法向与视线之间的夹角决定，具



[Gooch et.al 1998]

图 12.17: 冷暖色着色

体的插值公式为：

$$k = \left(\frac{1+\mathbf{l}\cdot\mathbf{n}}{2}\right)k_{\text{cool}} + \left(\frac{1-\mathbf{l}\cdot\mathbf{n}}{2}\right)k_{\text{warm}} \tag{12.6}$$

其中 $\mathbf{l}$ 是指向光源的方向，$\mathbf{n}$ 是表面法线方向．在此基础上，我们还可以将连续的颜色变化变成阶梯状的分段颜色，这样就得到了图12.14中的卡通效果．

这里我们介绍的风格化渲染技术只是抛砖引玉，可以看到为了实现一个效果我们都可以采用截然不同的方法，感兴趣的同学可以进一步了解不同艺术风格的实现，乃至于创造自己的艺术风格．