基于物理着色（一）

文刀秋二

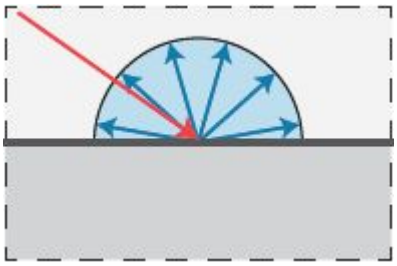
计算机图形学 话题的优秀回答者

基于物理着色（Physically Based Shading）就是计算机图形学中用数学建模的方式模拟物体表面各种材质散射光线的属性从而渲染照片真实图片的技术。基于物理的材质模型现在已经广泛的被各种渲染器和游戏引擎使用，它不仅可以让最终的画面更加真实，同时也更容易设计出让Artists能够更直观的修改和编辑材质属性的系统。Disney Principled BRDF就是一个很好的范例，Hyperion和最新的RenderMan都采用了这个技术。虚幻引擎4中的材质系统也是基于这个模型的简化版（Real Shading in UE4）。

我不想花太多笔墨详细的剖析Disney的Principled BRDF，相反的我希望通过列举几种常见的反射类型和它们现实中的对应各种材质以及介绍用于模拟他们的数学模型来High Level的说明可用的数学工具，理解了这些自然也就能理解Disney BRDF的设计思想，而你最后可以自己决定如何用这些工具设计你自己的材质系统。写到一半发现想写的内容篇幅有点多，所以暂时决定分成三个小章节：

1. 第一节介绍最基本的三种反射模型，漫反射，镜面反射和折射
2. 第二节，基于物理着色（二）- Microfacet材质和多层材质，介绍通过微表面（Microfacet）建模的粗糙表面模型，以及多层次材质的模拟（例如现实中的塑料、陶瓷等，漫反射材质上面有一层透明的镜面图层等）
3. 第三节，基于物理着色（三）- Disney和UE4的实现则是在以上的基础上谈谈Disney Principled BRDF的设计思路，参数意义，以及一些漂亮的渲染结果。
4. 第四节，基于物理着色（四）- 次表面散射则简单介绍了次表面散射材质的相关理论以及算法。

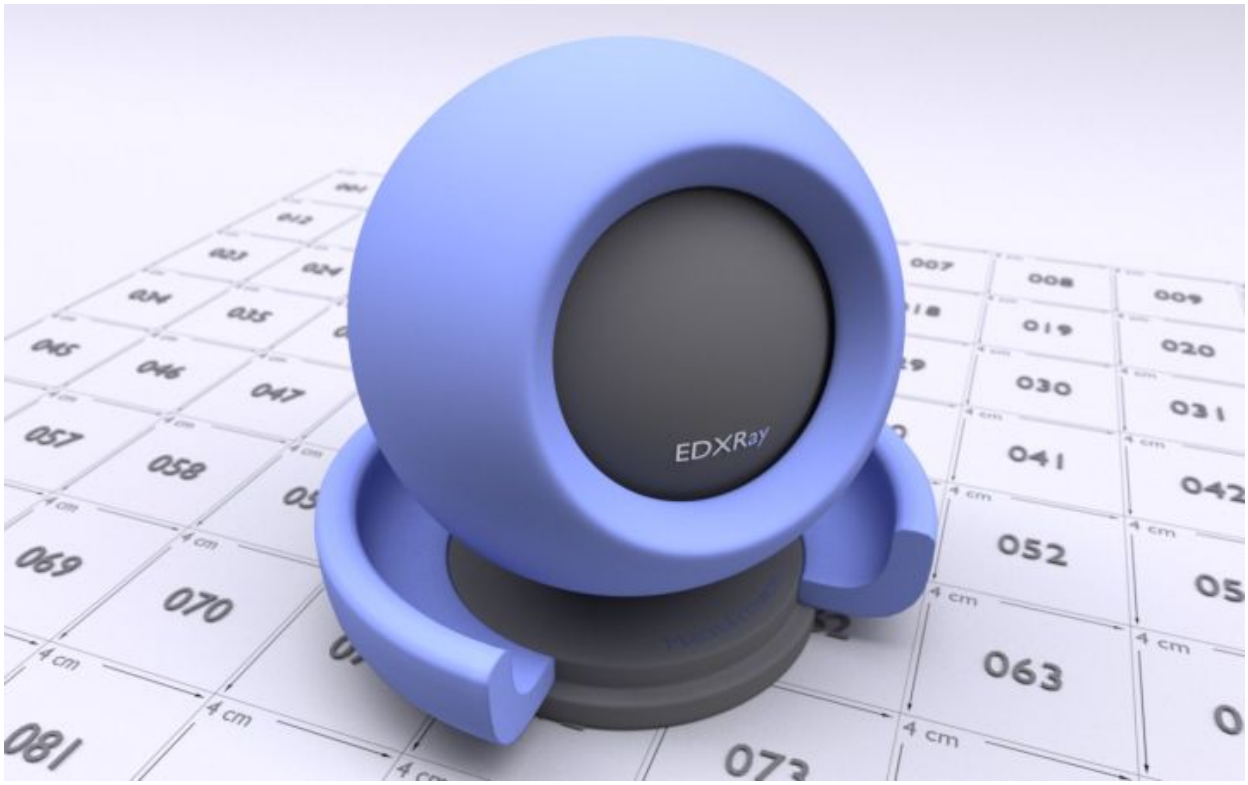
那么现在首先说说**漫反射**。漫反射是最容易模拟的模型，现实中的例子也很多，例如石头，水泥，砖块等。最简单的Lambertian很简单粗暴的认为光线被均匀的反射到表面上方的半球中。



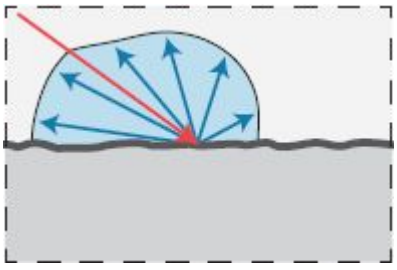
表面散射的性质通常用BRDF（双向反射分布函数）来表示。一句话不严谨的概括BRDF既是输入一个入射的方向和一个出射的方向，输出一个出射光线和入射光线能量的比值。不想细扣公式，直接上代码。

```
float Lambertian(Vector3 In, Vector3 Out)
{
    // 乘以PI的倒数是因为BRDF在半球内的积分需要为1，满足能量守恒
    return INV_PI;
}
```

因为Lambertian模型均匀的反射出能量，所以返回值是一个常量和出射入射方向无关。下面是一个用Lambertian材质渲染的图片。



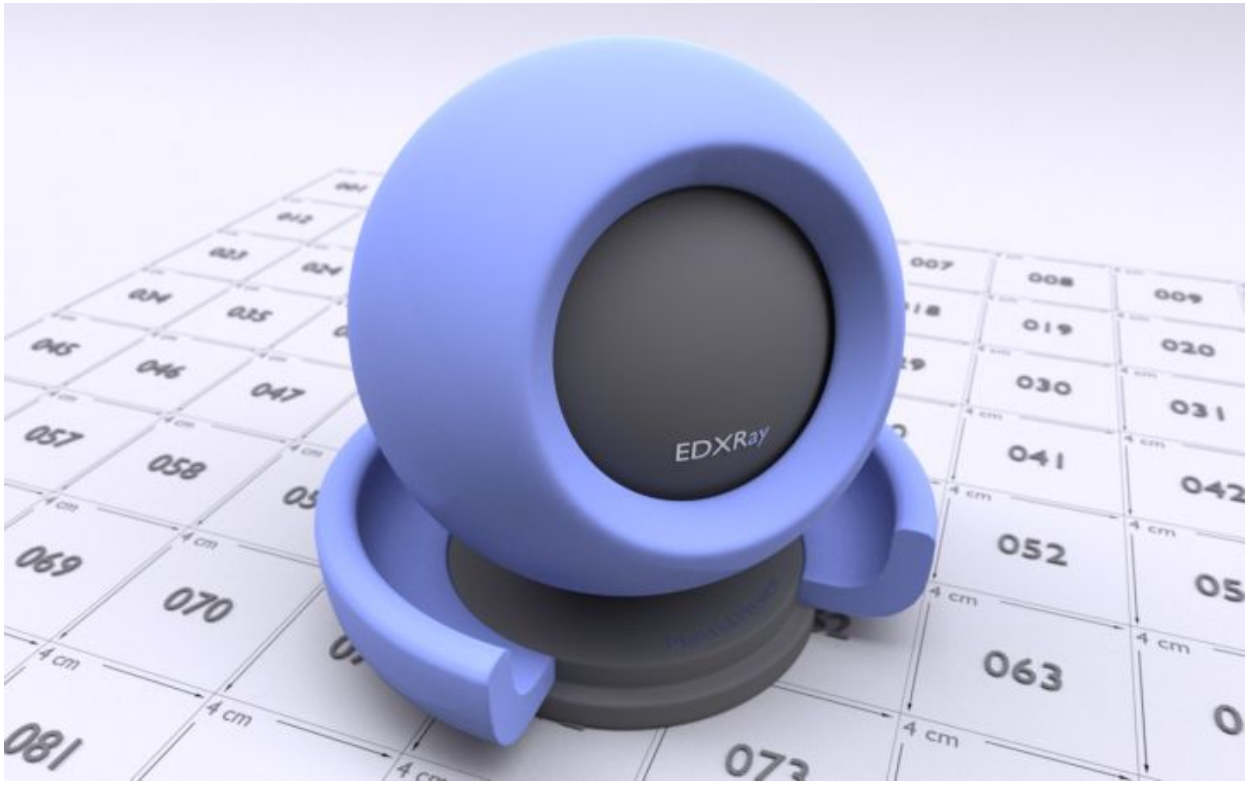
Lambertian并不能很好的模拟现实中的许多漫反射材质，现实中完美的均匀反射当然几乎是不存在的，有些现实中的材质会在视线方向接近和表面平行的时候反射更多的光线等。所以当然有一些更复杂的数学模型去模拟更复杂的漫反射，例如Oren Nayar模型。



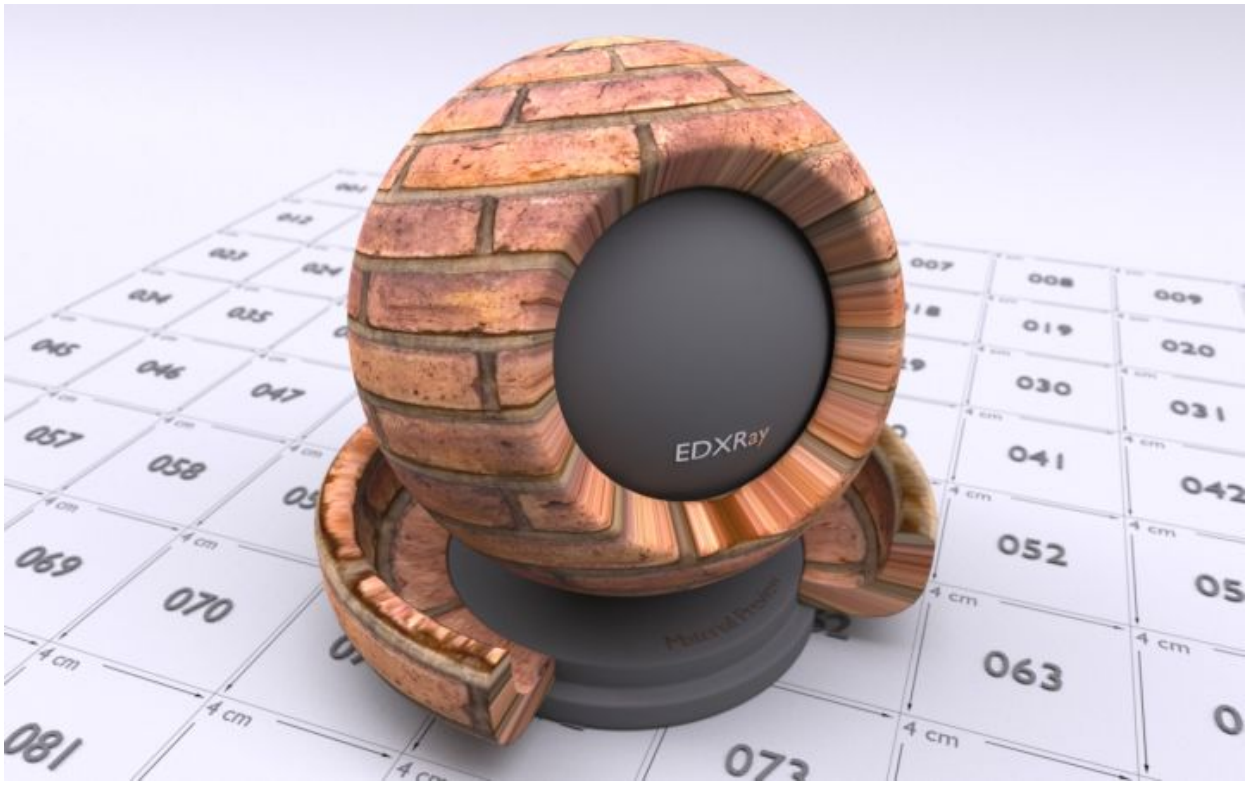
Disney的Principled BRDF简单的用两个Fresnel项来增加入射角度低时反射的光线能量强度来模拟这种效果。好处是计算量相对小而且能打到足够的近似。贴一个我自己实现的伪代码，公式在最上面文档的链接5.3章节有，自行查阅。

```
float DisneyDiffuse(Vector3 In, Vector3 Out)
{
    float oneMinusCosL = 1.0f - AbsCosTheta(In);
    float oneMinusCosLSqr = oneMinusCosL * oneMinusCosL;
    float oneMinusCosV = 1.0f - AbsCosTheta(Out);
    float oneMinusCosVSqr = oneMinusCosV * oneMinusCosV;

    // Roughness是粗糙度，IDotH的意思会在下一篇讲Microfacet模型时提到
    float IDotH = Dot(In, Normalize(In + Out));
    float F_D90 = 0.5f + 2.0f * IDotH * IDotH * Roughness;

    return INV_PI * (1.0f + (F_D90 - 1.0f) * oneMinusCosLSqr * oneMinusCosLSq
                (1.0f + (F_D90 - 1.0f) * oneMinusCosVSqr * oneMinusCosVSqr *
}
```

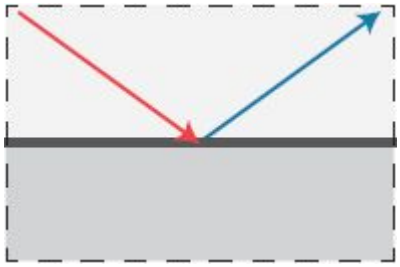
下面是用这种漫反射模型渲染的图片。注意球的顶部和Lambertian模型相比，因为入射角度低的缘故，会更加亮，具体量多少也取决于Roughness参数的大小。



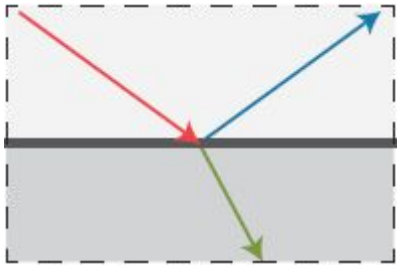
下面是一张配上砖块纹理的渲染。



接下来看看另一类比较容易模拟的材质，**光滑镜面反射**。最常见的就是镜子和玻璃一类的材质了，他们分别为导体和电介材质。镜面反射就只是将入射光线根据表面法线反射，并且只在反射方向有能量其他方向均为0。



对玻璃这种电介材质则除了反射之外还有根据物体的折射率一部分光线会折射进入物体之中。



反射和折射能量的多少是根据菲涅尔定律决定（Fresnel's Law）。图形学中常常使用用Schlock的近似Fresnel，伪代码如下。

```
float Fresnel_Schlick(float InCosine, float normalReflectance)
{
        // InCosine是入射光线和法线的夹角，normalReflectance是入射光线和法线垂直时
        float oneMinusCos = 1.0f - InCosine;
        float oneMinusCosSqr = oneMinusCos * oneMinusCos;
        float fresnel = normalReflectance +
                (1.0f - normalReflectance) * oneMinusCosSqr * oneMinusCosSqr

        return fresnel;
}
```

值得一提的是在离线渲染，光线追踪里这两种材质本身的着色很容易，只用根据反射和折射的方向射出新的光线然后递归的着色即可。在游戏实时渲染中反射通常是用屏幕空间的反射（Screen Space Reflection）和环境光照贴图两者结合实现的，折射也通常是用扭曲屏幕空间像素的方法。都会有一些Artifact，目前没有完美的解决方案。

贴两张镜面和玻璃材质的渲染。

对于电介材质来说，除了本身的颜色以外，折射率是决定外观的另一个重要因素。折射率决定了折射光线的方向以及全反射的临界角度，上面的图片采用的折射率是1.5，也就是常见的玻璃的折射率，下面这张图则是用折射率2.42的渲染结果，对应的材质是钻石。根据菲涅尔定律，折射率越高的物体反射光线与折射光线的比值也越大，所以整个看起来更加Bling Bling了。读完这篇章至少明白钻石比玻璃更加闪亮，也就够了。

当然现实中不存在完美平滑的表面。接下来提高真实感的一个手段就是对材质表面的法线进行扰动。法线贴图是方法之一。下面是一个镜面反射的物体在表面因为法线贴图变得粗糙之后的模样。

可以看到粗糙的表面更能准确的模拟现实中许多材质的视觉效果。

今天就写到这，下一节会介绍如何不用法线贴图，而用微表面（Microfacet）模型去模拟粗糙的表面，以及多层的材质模拟。

本文中光线传递的示意图出自：https://www.http://mitsuba-renderer.org/releases/current/documentation.pdf

本文中其他所有渲染结果均使用本人自己开发的渲染器EDXRay。（链接是非常旧的介绍，见谅）