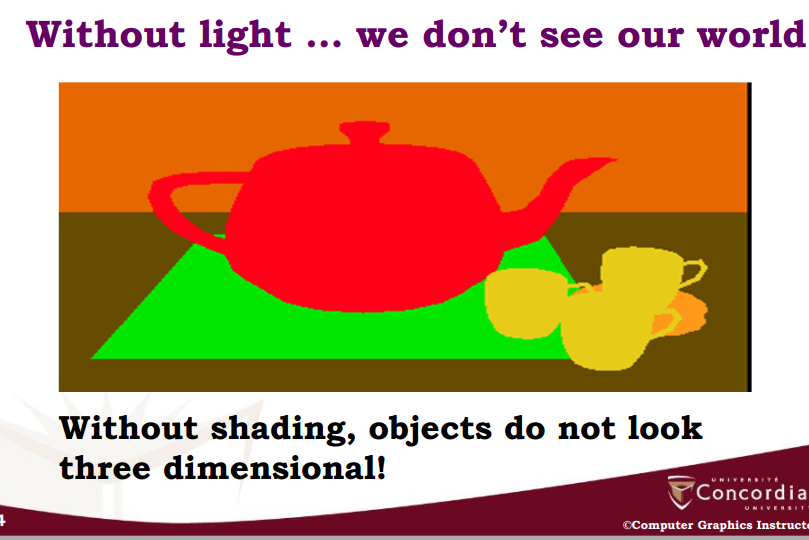


没有light 一片黑，没有shading不立体

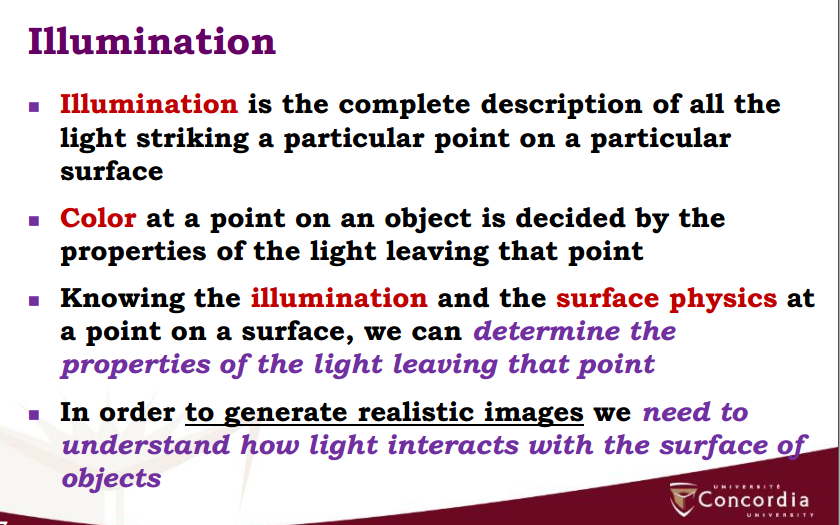


Illumination

Illumination:照明是对所有照射在特定表面上特定点的光的完整描述 //集合而不是一个点,指射入的光

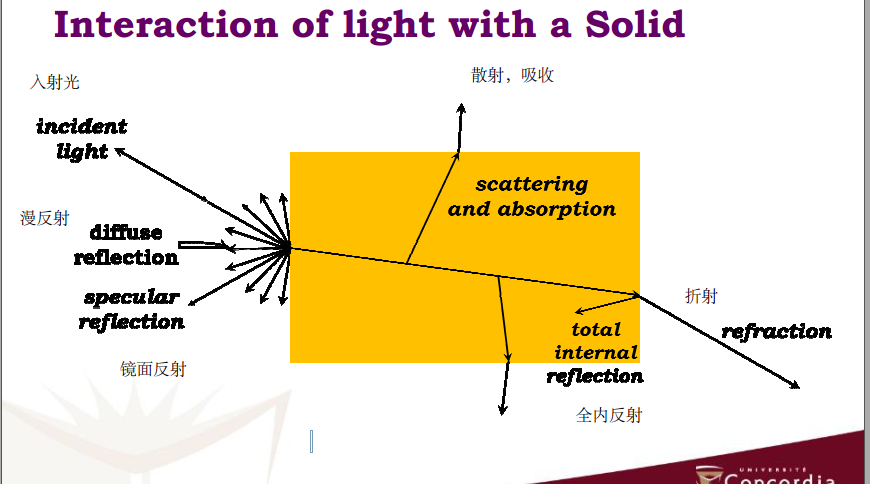
Color:这个点的颜色取决于离开这个点的光的性质

知道illumination与surface physics 表面物体性质，我们能决定离开这个点的光的性质



为了生成真实的图像，我们需要知道光是怎么和物体表面交互的

Interaction of light 光的交互



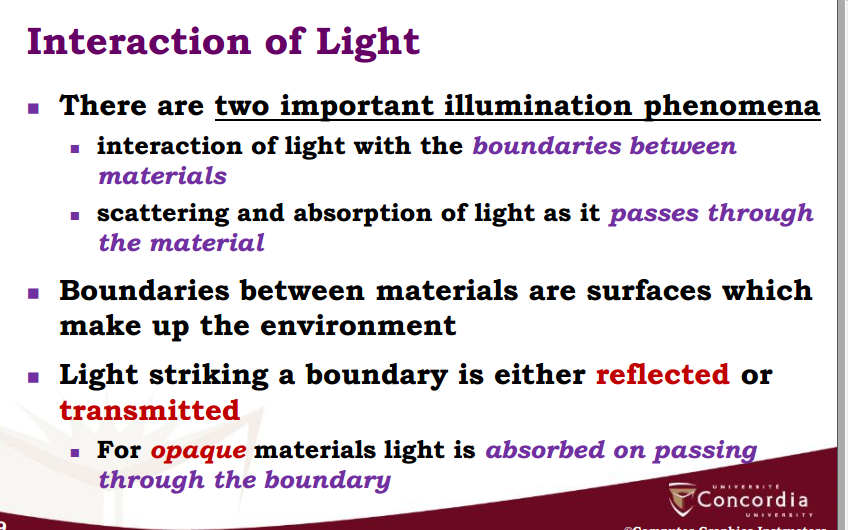
光的交互

有两个重要的illumination phenomena 照明现象

光与物质边界的交互//反射

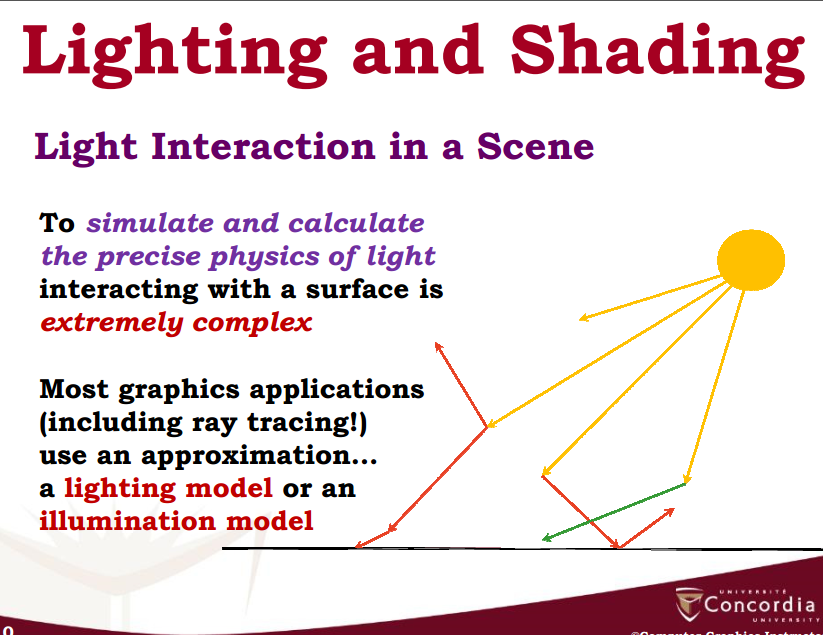
光的分散与吸收当他穿过物质的时候

物质与表明面的边界组成了环境



光打在边界上，不是反射就是透射

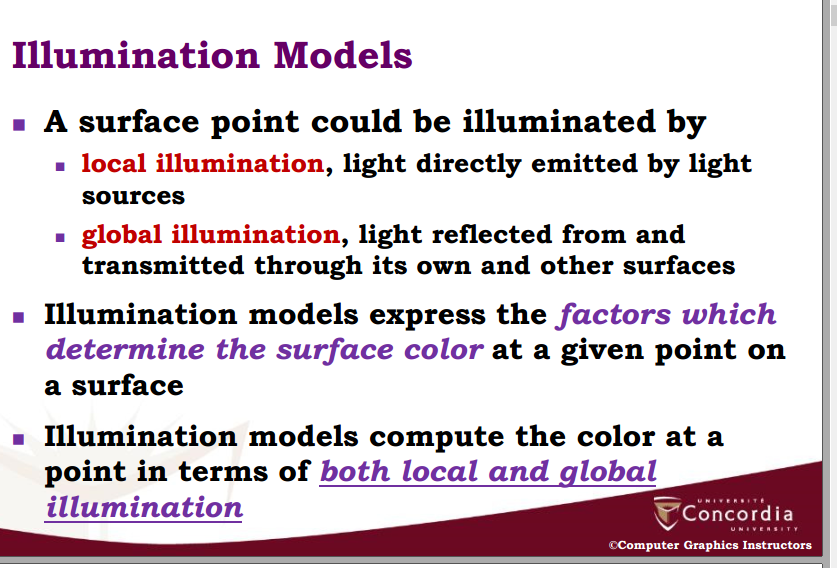
对于不透明的材料，光在试图穿越边界的时候被吸收



模拟物理现象很难

更多图形应用使用近似，一个light model或者illumination model

Illumination model



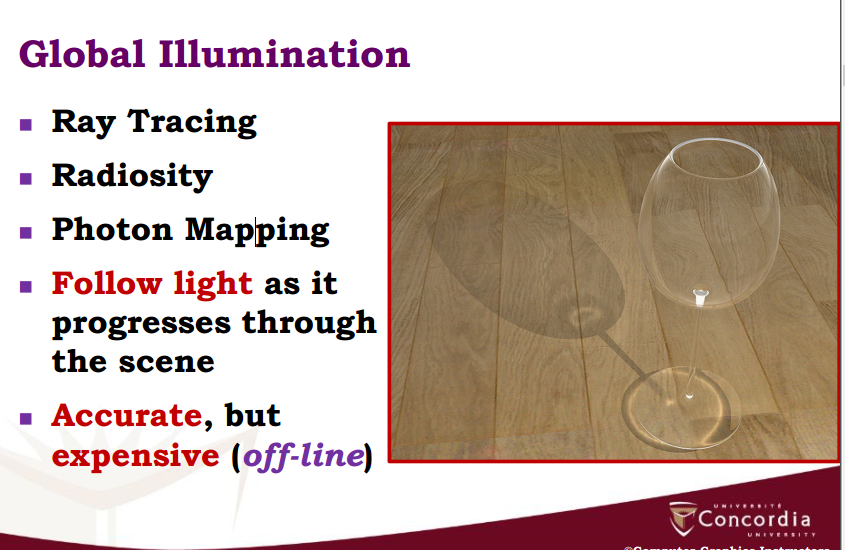
一个表面的点可以被用两种方式照射

Local illumination， 直接由光源所射的光

Global illumination， 那些由其他表面所反射透射的光

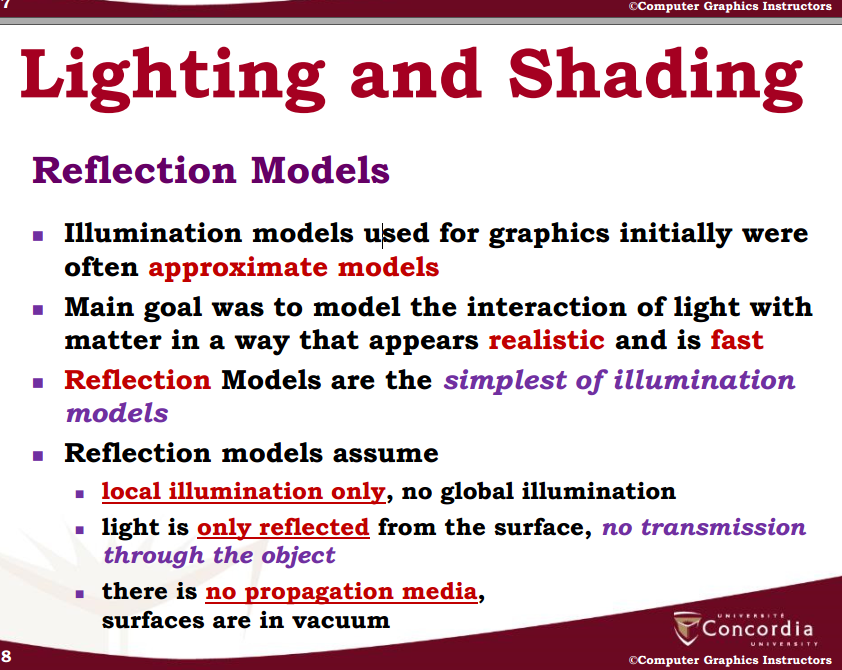
Illumination model 用系数来决定 物体表面的颜色

Illumination model 计算一个物体表面的颜色通过local illumination global illumination





Ray tracing，加了倒影啥的//其他物体的反射也在上面



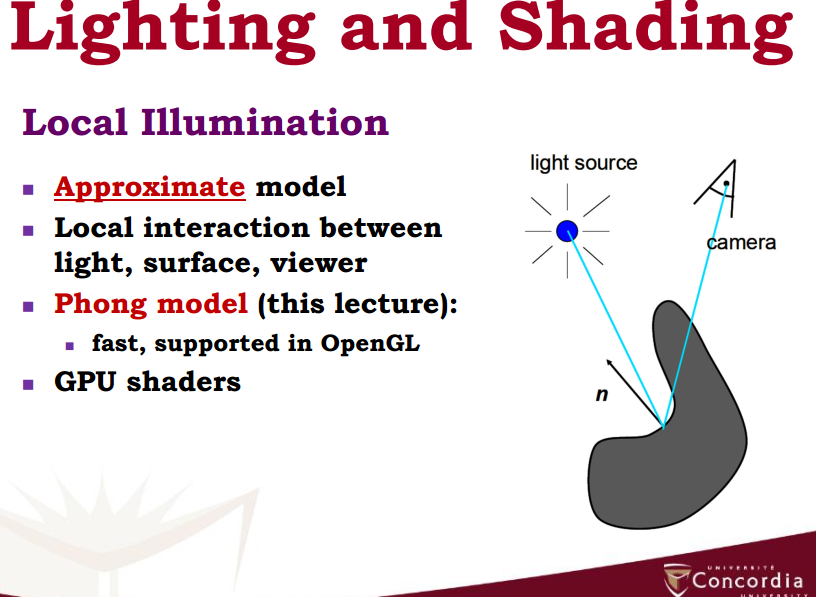
Reflection model

Reflection model是最简单的approximate model of illumination model

他假定只有local illumination没有global

光只有表面反射，没有透射

没有propagation media传播介质 //空气水啥的， 表面处于真空



Local illumination

是approxiamate model

是光源，表面，camera之间的local interaction

用Phone model //这节课内容

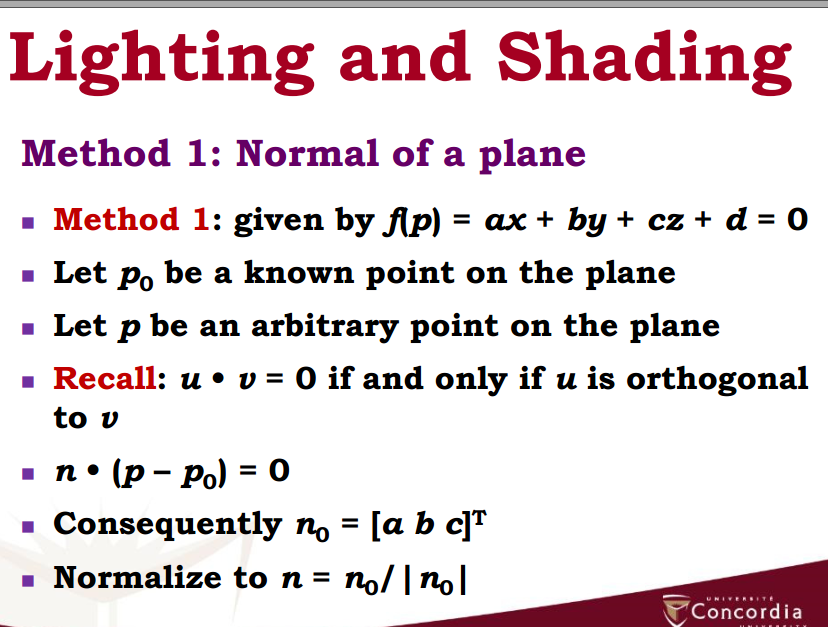
颜色取决于 表面垂直向量， 相对camera位置，相对光源位置，表面反射性质

PAGE 20

我们必须计算并指出normal vector，哪怕在opengl里

Normal of a plane

方法1：也是最简单的，当我们有平面方程的时候使用这个方法



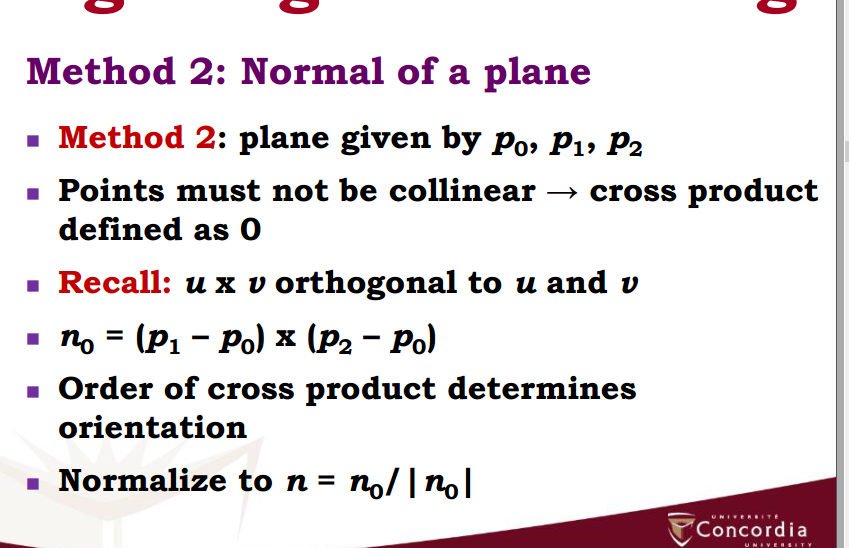
给你一个plane，让p0作为一个Plane上的已知点，让p作为任意点

那么n\*(p-p0)=0,因为只有两个向量相乘等于0才垂直

结果就是

单位化

方法2，已知平面三个点



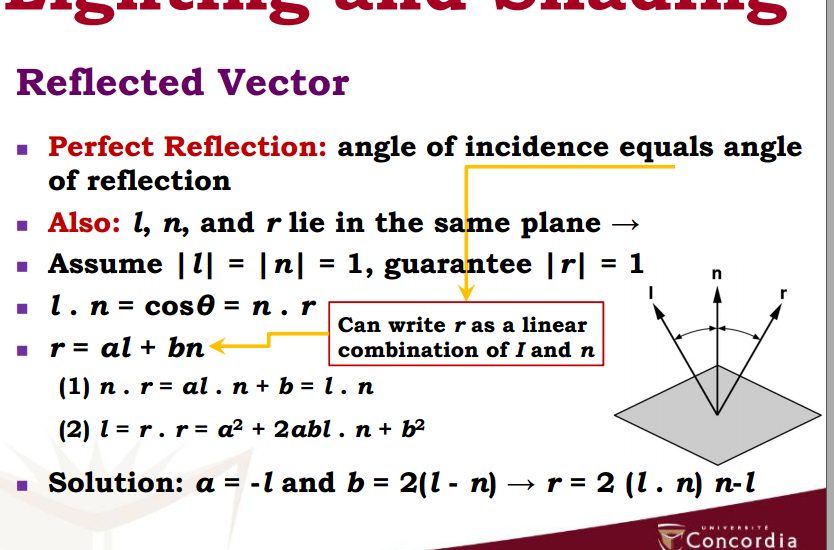
三个点必然不能在一条线上

由三个点组成一个平面，那么n0就是两个点cross两个点的差值

//也就是两个向量CROSS 乘

Reflected vector

我们知道了垂直向量，就可以求反射vector



Perfect reflection，完美反射，入射角等于反射角

假设所有的射线大小等于1

那么ln=cos Theta \*|l|\*|n|=nr //因为角度一样

然后我们将r表示成n与l的结合//因为不平行

R=al+bn

A与b只是两个待解决的系数，他们可以用l与n得到r

最后我们既能得到a 与b

R=2(l-n)n-l

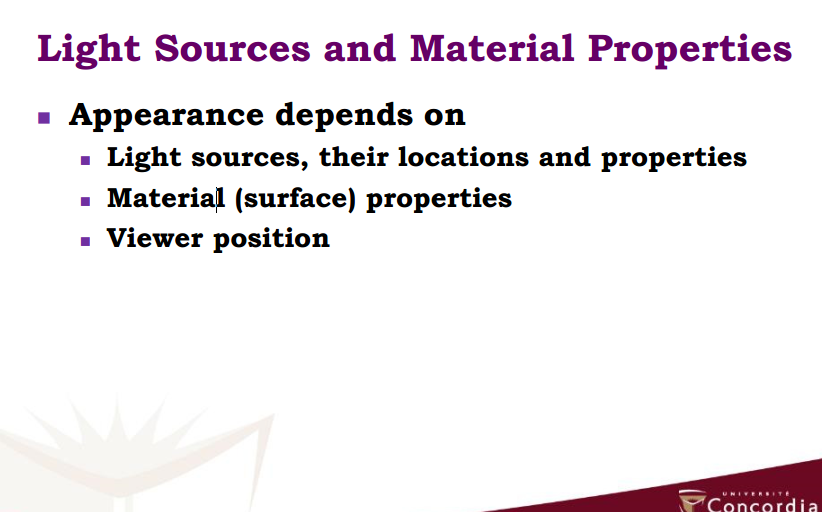
光源与物质属性

外表取决于

光源位置性质

Material 性质

Viewer位置



光源分类

Ambient light：没有可识别的光源或方向

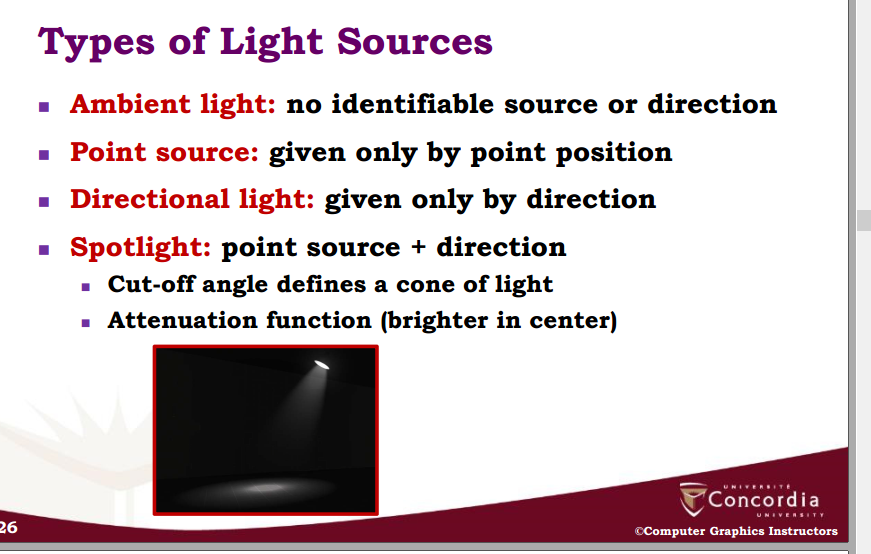
Point source:点光源，你给点就行了

Directional light: 给个方向就行

Spotlight：点+方向

截止角定义了一个光锥

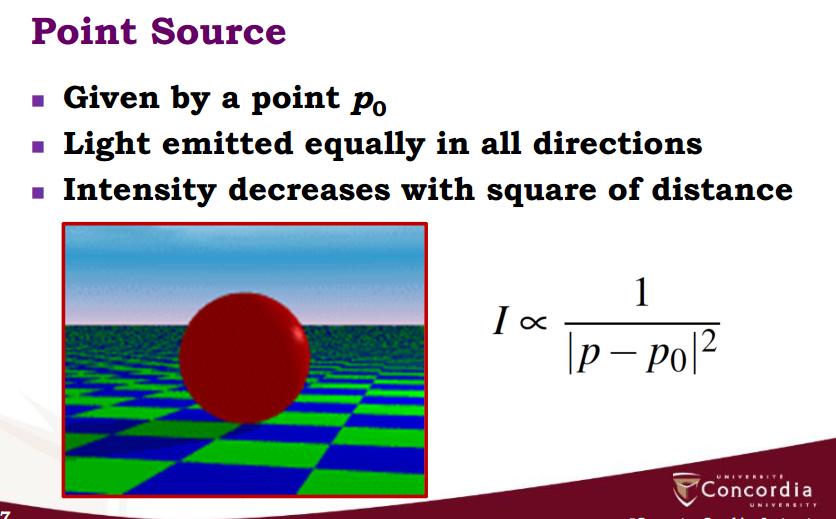
衰减function:中间亮旁边暗



点光源

给一个点，所有方向点强度一样

强度与距离点光源的平方成反比 //距离越大，强度越小



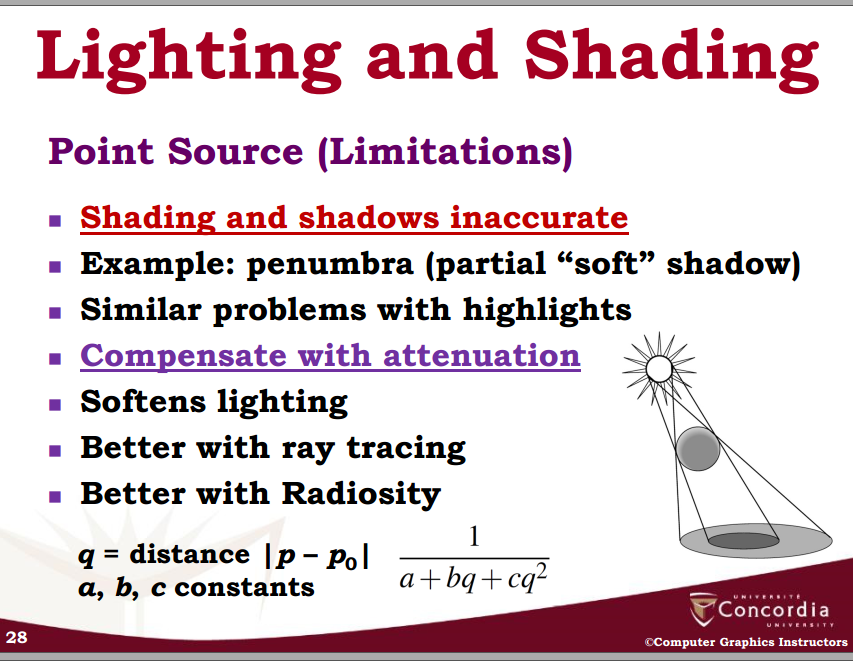
缺陷：

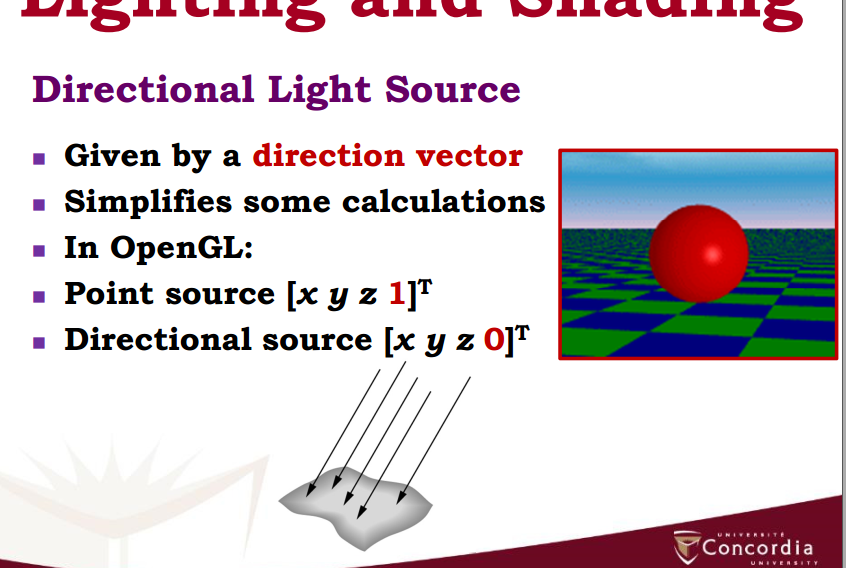
阴影不正确

例子penumbra半影，没法表达soft shadow,因为一个点没有体积//示例图是有soft shadow的情况

而且没办法表示衰减

更容易ray tracing与radiosity



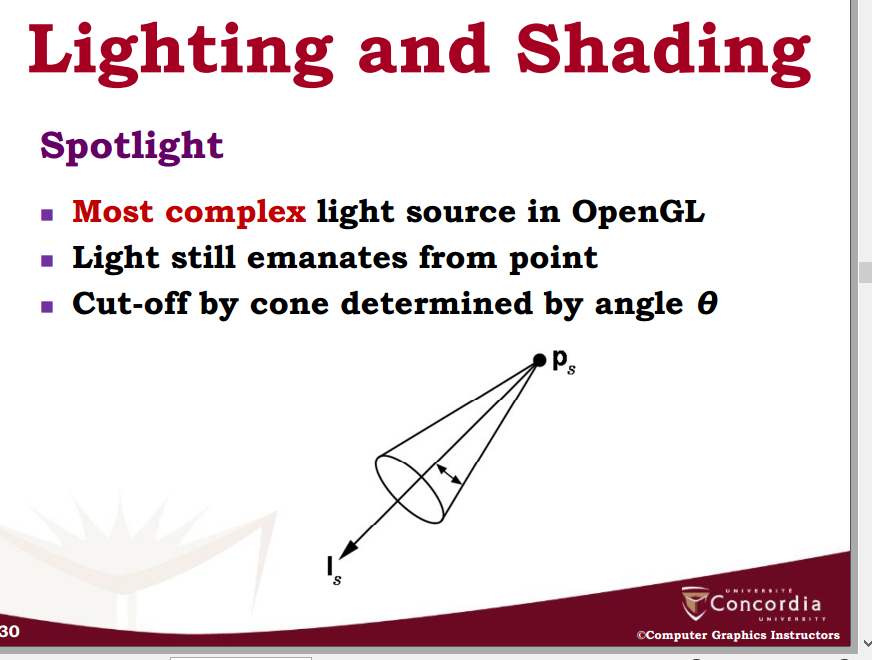


给一个向量就好

在Opengl里，direction

al是 0，因为是向量

Point是1，因为是位置

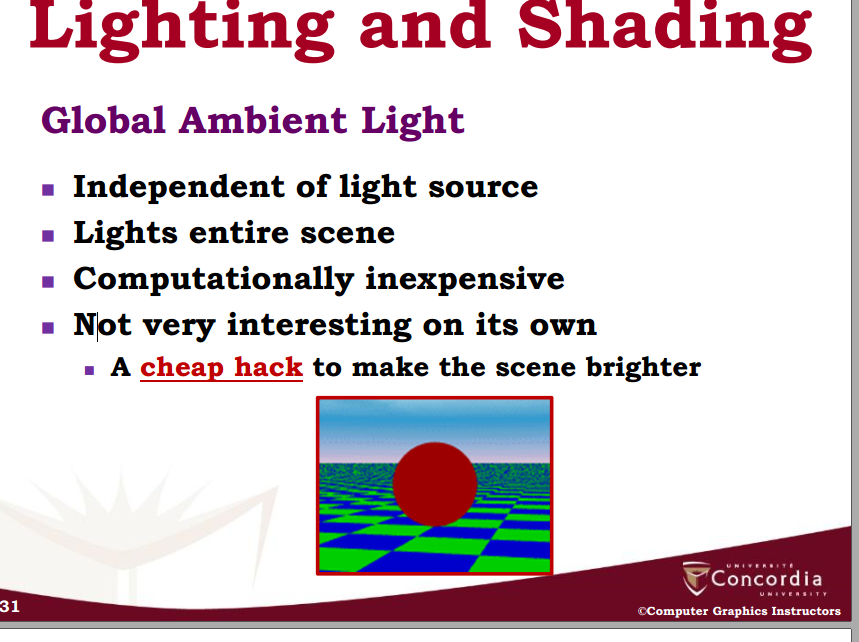


Spotlight

Opengl里最复杂的光源

光还是由点产生的

cut off取决于角度



全球环境光

与光源无关

整个场景都有光

计算起来不贵

Phong Illumination Model

计算表面任意点的颜色

在真实与有效之间找到了平衡点

Local computation//没有多个物体的交互

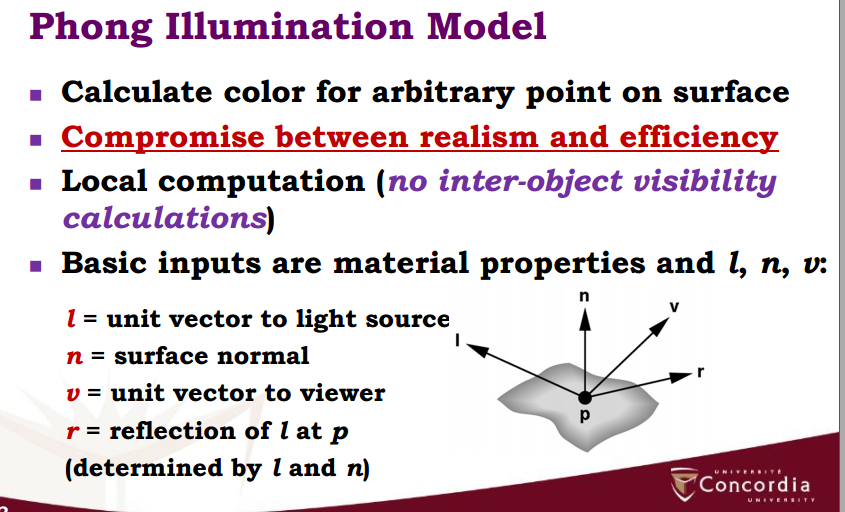
基本的Input

l:指向光源的Unit vector//长度为1

n：表面的垂直向量

v:观察者角度的单位向量

r: l在p点关于n的反射向量



Phong illumination overview

添加来自每个light source的贡献// 可以有多个光源

把最终结果控制在[0,1]内

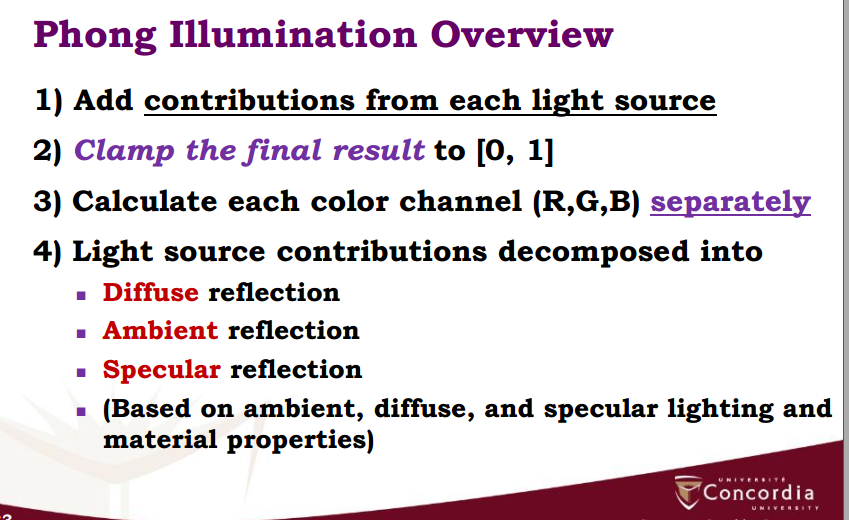
把RGB每个颜色分别计算

点光源的贡献被分为

Diffuse reflection 漫反射

Ambient reflection 环境反射

Specular reflection 镜面反射



Diffuse reflection

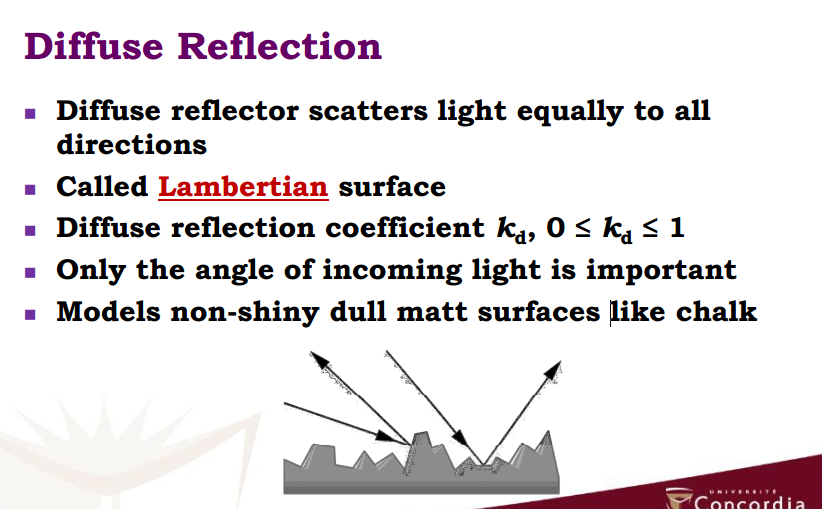
漫反射器把光均匀的散射到所有方向

叫做labertian surface

漫反射系数kd， 在0到1之间

只有入射角度重要

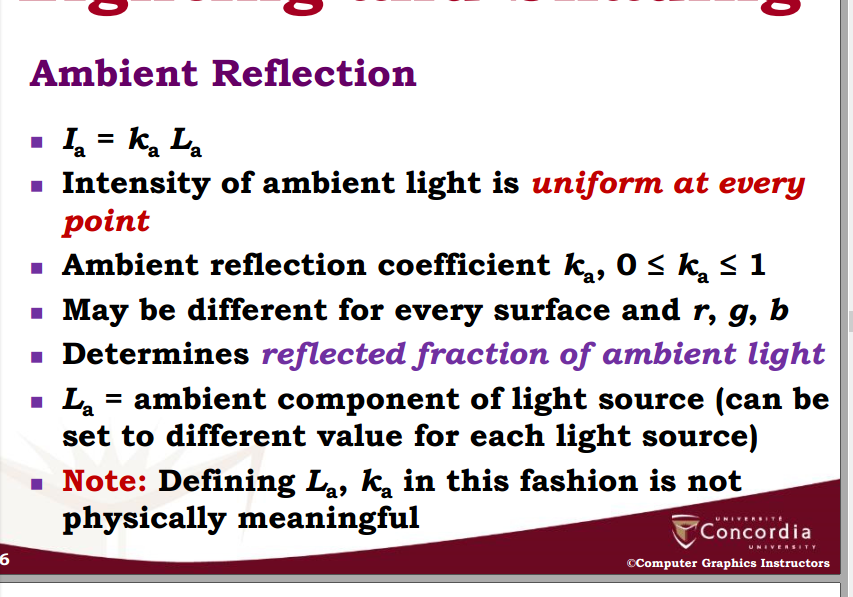
模型的无光泽的暗哑表面像粉笔



例如下面这个图，表面粗糙不像金属，就不会shiny



取决于入射角,kd是系数，ld是散射元素， abc是平面系数，

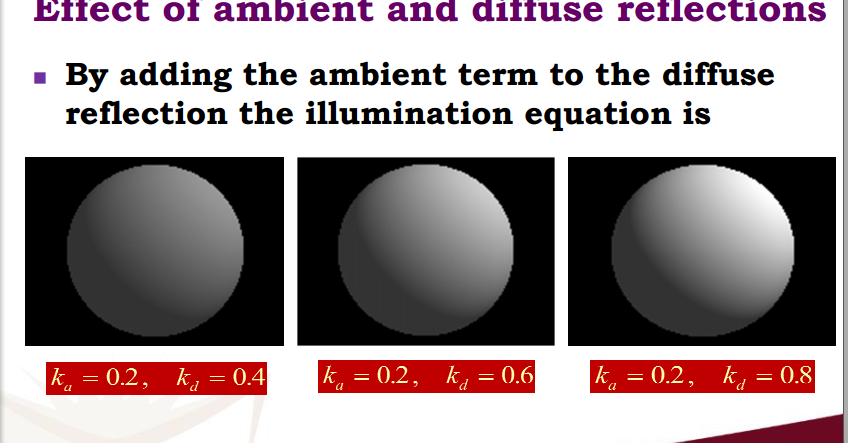


Ambient reflection

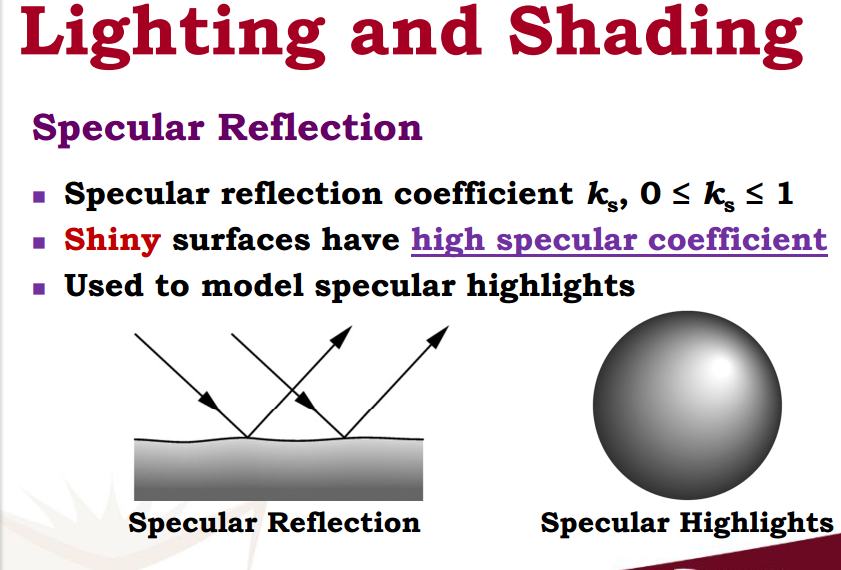
每个点的ambient light强度相同

系数ka在0到1

La就是光源的ambient component 环境组件（每个点光源的值都可以不同）

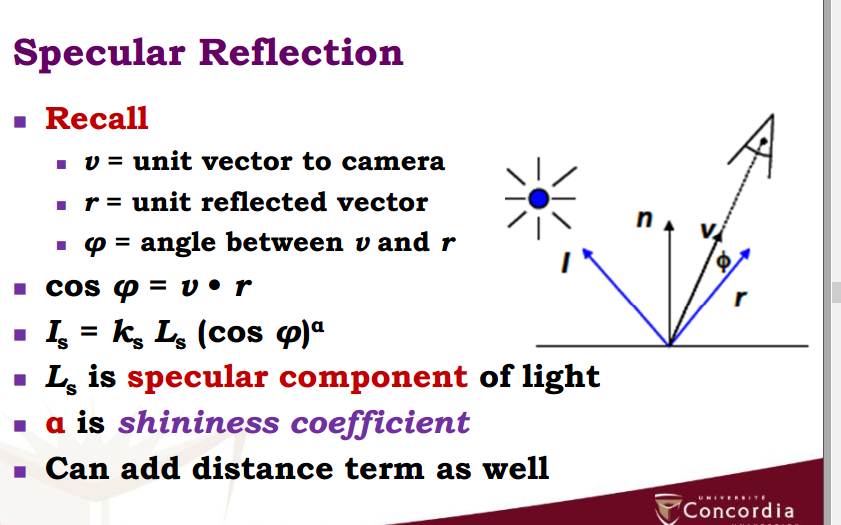


Kd越大越亮

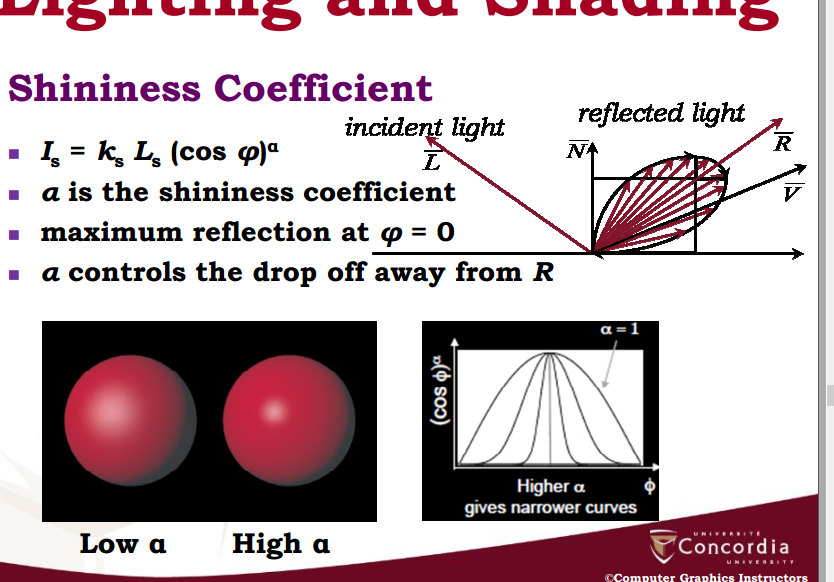


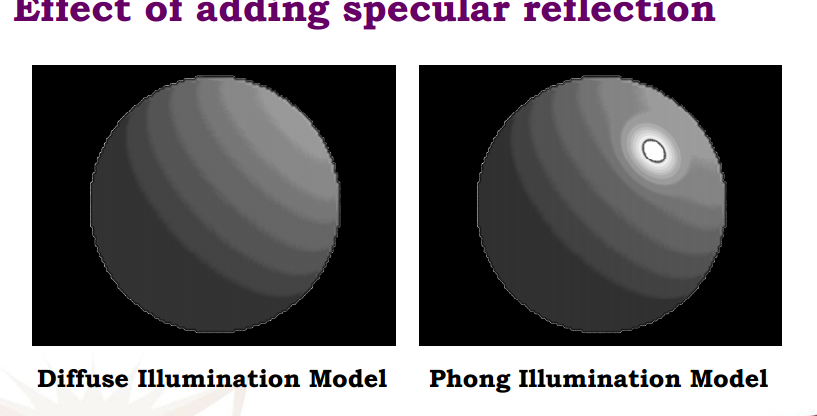
镜面反射

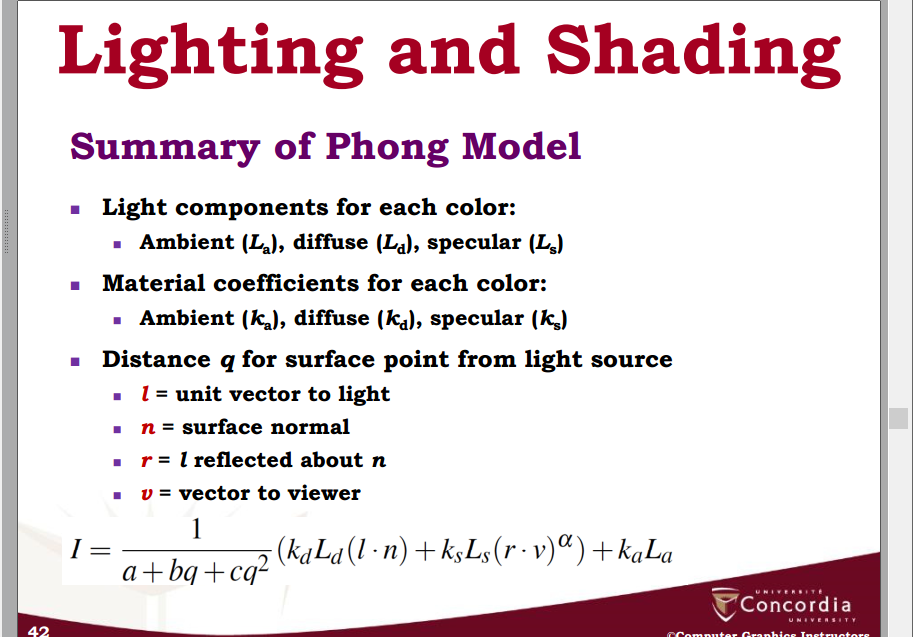
Shiny surface镜面反射系数高例如金属



A是shininess系数



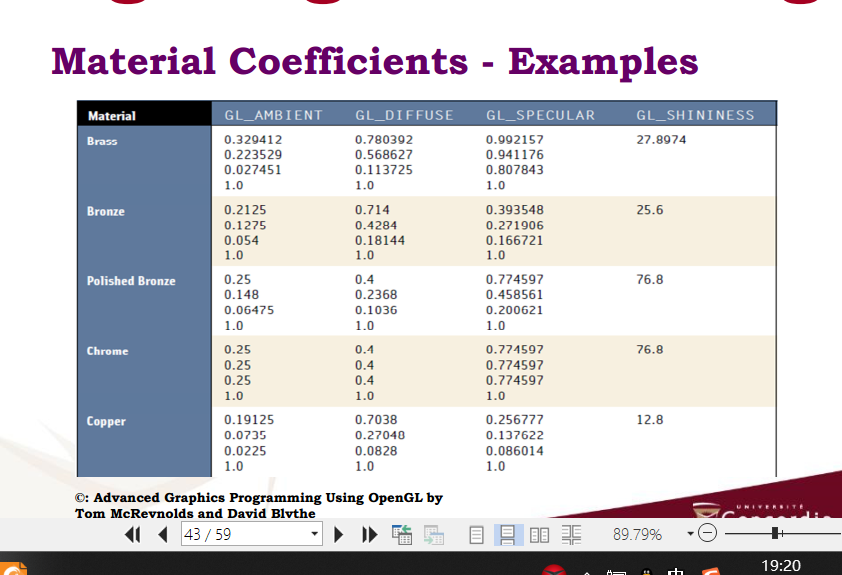


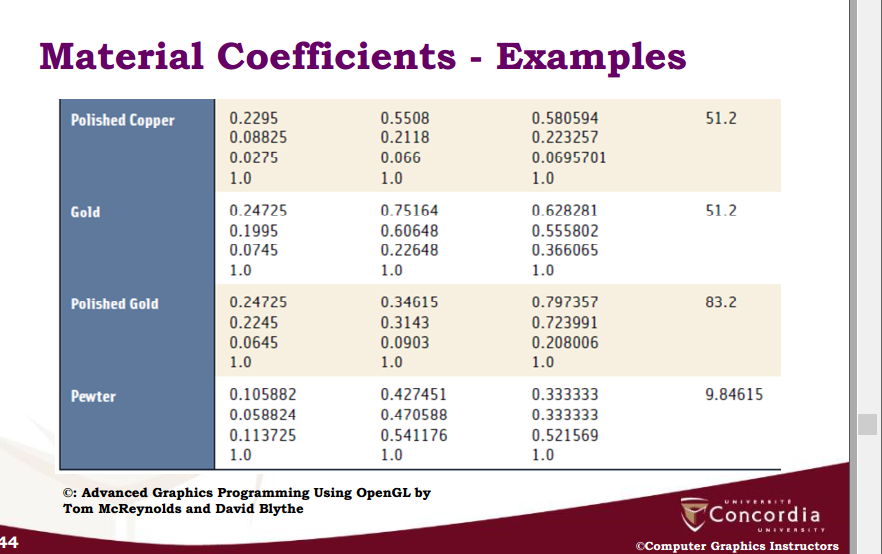


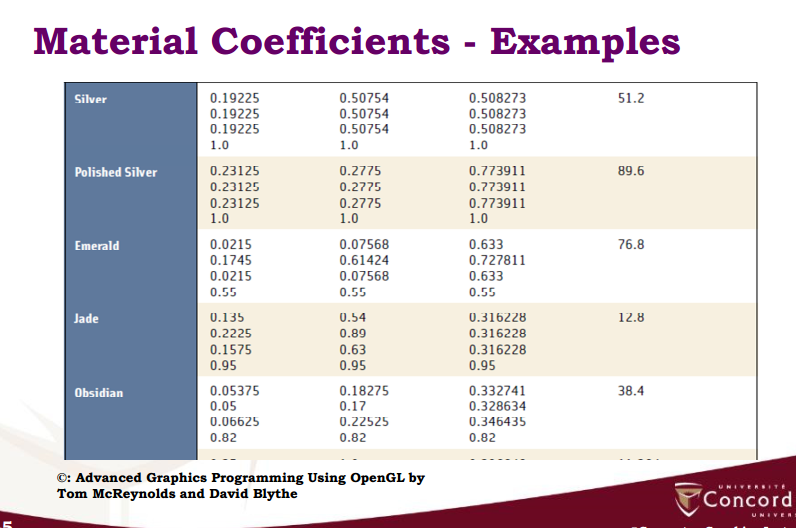
Phone model 主要就是设置这些东西

La Ld Ls就是不同的light component

Material就是表面的系数







Shading

决定每个pixies的颜色的过程，使用illumination model

最简单的方法

决定每个pixel是否可见

计算表面法向量

计算光线强度与颜色使用Illumination model，

很贵