Rendering Equation

💡本文的顺序：介绍辐射度量学（为引入渲染方程打下基础）=> 介绍渲染方程 => 通过渲染方程理解常见的图形学算法

1.Radiometry - 辐射度量学

（1）Radiant Energy and Flus（Power）

Radiant energy：the energy of electromagnetic radiation.It is measured in units of joules

辐射能：辐射的能量，以焦耳为单位

$$Q[J = Joule] \tag{1}$$

Radiant flux/power：the energy emitted, reflected, transmitted or received, per unit time

辐射通量/辐射功率：每单位时间释放、反射、透射或接受的能量，以瓦特或流明为单位

$$\phi \equiv \frac{dQ}{dt}[W = Watt][lm = lumen] \tag{2}$$

光是有能量的，在单位时间内光源发出或者平面接受的能量，就是辐射通量

（2）Radiant Intensity

Radiant Intensity：the power per unit **solid angle** emitted by a point light source

辐射强度：每单位立体角由点光源发出的功率【注意：就是上图中第1个】

$$\begin{gathered} I(\omega) \equiv \frac{d\Phi}{d\omega} \\ [\frac{W}{sr}][\frac{lm}{sr} = cd = candela] \end{gathered} \tag{3}$$

所以，如果一个点光源能够均匀向外散射能量，那么：

$$I(\omega)_{point\_light} = \frac{\Phi}{4\pi} \tag{4}$$

（3） Irradiance

Irradiance（E，辐照度）：the power per unit area incident on a surface point

辐照度的含义是：某个表面在单位面积上接受到的辐射通量，所以说dA并不是一个可变的量（不是说这个表面面积越大，dA就越大）

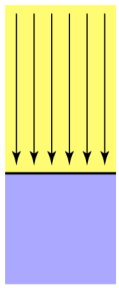
$$\begin{gathered} E(x) \equiv \frac{d\Phi(x)}{dA} \\ [\frac{W}{m^2}][\frac{lm}{m^2} = lux] \end{gathered} \tag{5}$$

但是不是也可以写成：也就是总磁通量除以总面积

$$E(x) = \frac{\Phi(x)}{A} \tag{6}$$

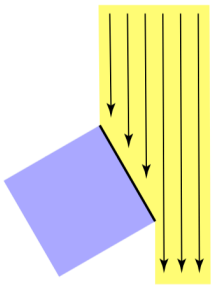
注意：**这里dA代表的是单位照射面积，而不是垂直于入射光线的投影面积**「Irradiance和radiance公式中的dA是等价的」

between light direction and surface normal.
(Note: always use a unit area, the cosine applies on Φ)



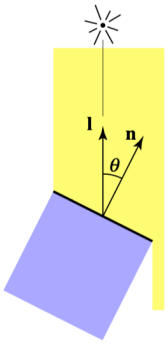
Top face of cube receives a certain amount of power

$$E = \frac{\Phi}{A}$$



Top face of 60° rotated cube receives half power

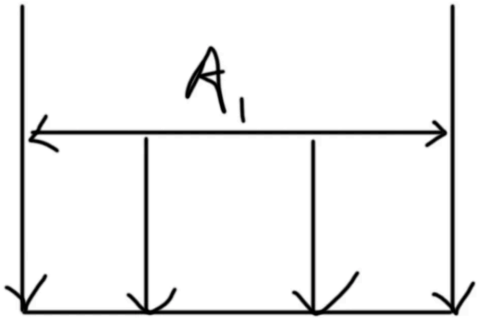
$$E = \frac{1}{2}\frac{\Phi}{A}$$



In general, power per unit area is proportional to $\cos\theta = l \cdot n$

$$E = \frac{\Phi}{A}\cos\theta$$

注意：👆上图中，这个cosθ实际上是乘在Φ旁边的，而不是A旁边的。我们可以这样理解：



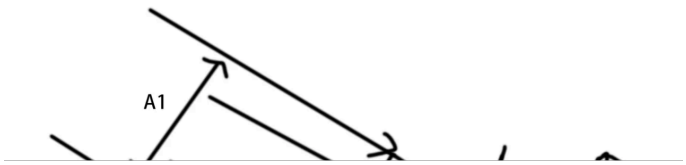


图1：光线垂直射向面积为A1的平面，我们将辐射通量看成一根根光线，假设一单位面积有n根光线，那么A1能接收到A1*n根光线；图2：光线斜着射向平面，光线与平面法线的夹角为theta，那么实际上他接受不了A2*n跟光线，而是(A2*cosθ)*n根光线；我们能看到，当斜着射向平面时，平面所接收到的辐射通量是有衰减的

=> 这就是朗伯余弦定理，一般我们认为光通量(radiance flux)是有方向性的，它也许不是单一方向，但是它每根光线都有它对应的方向，**所以我们需要分析平面能接收到的所有光线，并计算垂直于光线方向的平面**

所以我们再回看公式：

$$E(x) = \frac{d\Phi(x)}{dA} \tag{7}$$

E(x)：x代表平面的位置，E(x)是指x位置处的辐照度，其中平面面积为dA，接收到的辐射通量为dΦ(x)；也就是单位时间内，通过dA的光能量

这里的dΦ(x)要注意，根据我们上面的推理，这里的dΦ(x)应该等于：

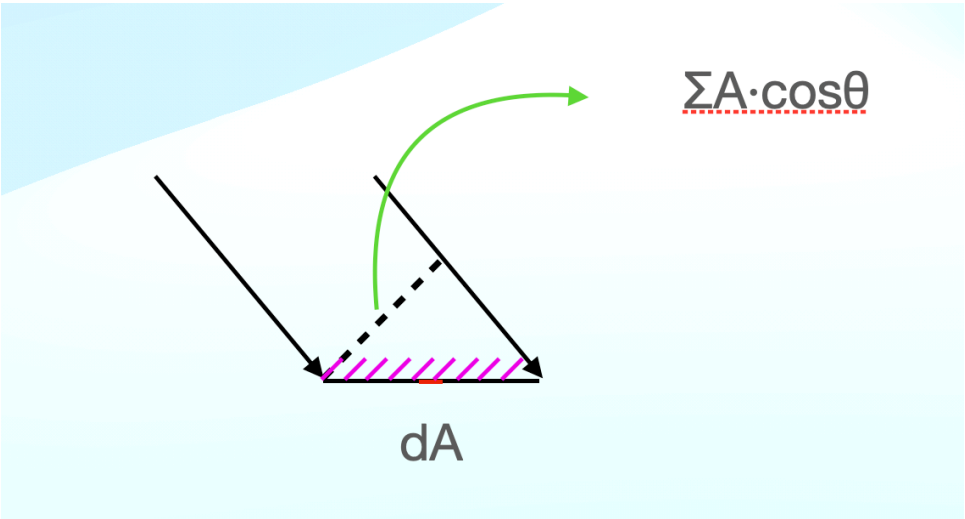
$$\begin{gathered} d\Phi(x) = \int d\Phi(x,\omega_i)\cdot cos\theta\, d\omega_i \\ E = \frac{d\Phi(x)}{dA} = \frac{\int d\Phi(x,\omega_i)\cdot cos\theta\, d\omega_i}{dA} \\ \text{假设：各个方向的磁通量相同} \\ E = \frac{d\Phi(x,\omega_i)\cdot \sum cos\theta \cdot \int d\omega_i}{dA} = \frac{d\Phi(x,\omega_i)\cdot \int d\omega_i}{dA}\cdot \sum cos\theta \end{gathered} \tag{8}$$

对于方向光：

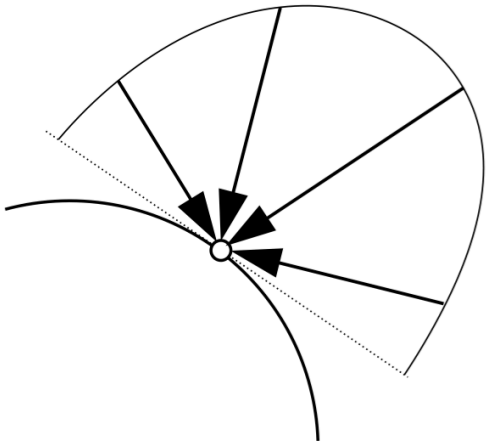
$$\begin{gathered} \text{方向光：} d\Phi(x) = d\Phi(x,\omega_i)\cdot cos\theta \\ E = \frac{d\Phi(x)}{dA} = \frac{d\Phi(x,\omega_i)}{dA}\cdot cos\theta \end{gathered} \tag{9}$$

根据式子，我们明白了这个cosθ是从何而来，现在我们建立了E与Σcosθ的关系，如果细心我们可以发现另一个式子：dA·Σcosθ=垂直于光方向的投影面积，所以说，垂直于光方向的投影面积与E之间的关系是成正比的

我们以这个图为例：解释为什么是垂直于光方向的投影面积



所以👇图中，最上面的实心弧线越长，在其他条件不变的情况下，irradiance(E)越大



🌟联想到：

- 朗伯余弦定理，也就解释的通了，我们之前在blinnphong中所涉及的**光的强度，实际上就是Irradiance**；实际上当平面不垂直于光线时，所获取到的光线条数(辐射通量)就会变少
- 同时，blinnphong模型中的点光源会根据距离进行衰减，如果光的强度是Intensity，按道理来说是不衰减的，因为是flux/单位立体角，单位立体角可以延伸出去；所以再次证明，**光的强度，实际上就是Irradiance**，因为点光源向四周发射，那么垂直于光的面积就是球的表面积$4\pi r^2$，所以光是按照距离的平方进行衰减
    - 是因为光源发出的总能量是不变的，但总面积增大了，根据公式2

（4）Radiance

Radiance（L）：the power emitted reflected transmitted or received by a surface，per unit solid angle，per projected unit area

辐射度(亮度)：每单位立体角和每单位投影面积上，由表面反射、发射或接收的能量。

$$L(p,\omega) \equiv \frac{d^2\Phi(p,\omega)}{d\omega\, dA\, cos\theta} \tag{10}$$

公式中dAcosθ就是垂直于光线的面积{注意：irradiance和radiance中的dA的定义其实是不一样的，irradiance中dA是表示的垂直面积，而radiance中的dA表示的是接受平面}

radiance的实际含义就是：该单位面积的平面，接收到从某一个方向来的光，是多少

> 所以，对于一个平面所接收到的光照的计算，我们可以拆成许多微平面，每个微平面又能从四面八方接受能量，所以要对角度和面积进行积分，能够得到所有的光照

irradiance & radiance 的关系：

$$
\begin{aligned}
dE(p,\omega) &= L_i(p,\omega)cos\theta\ d\omega \\
E(p) &= \int_{H^2} L_i(p,\omega)cos\theta\ d\omega
\end{aligned}
\tag{11}
$$

（5）BRDF - Bidirectional Reflectance Distribution Function

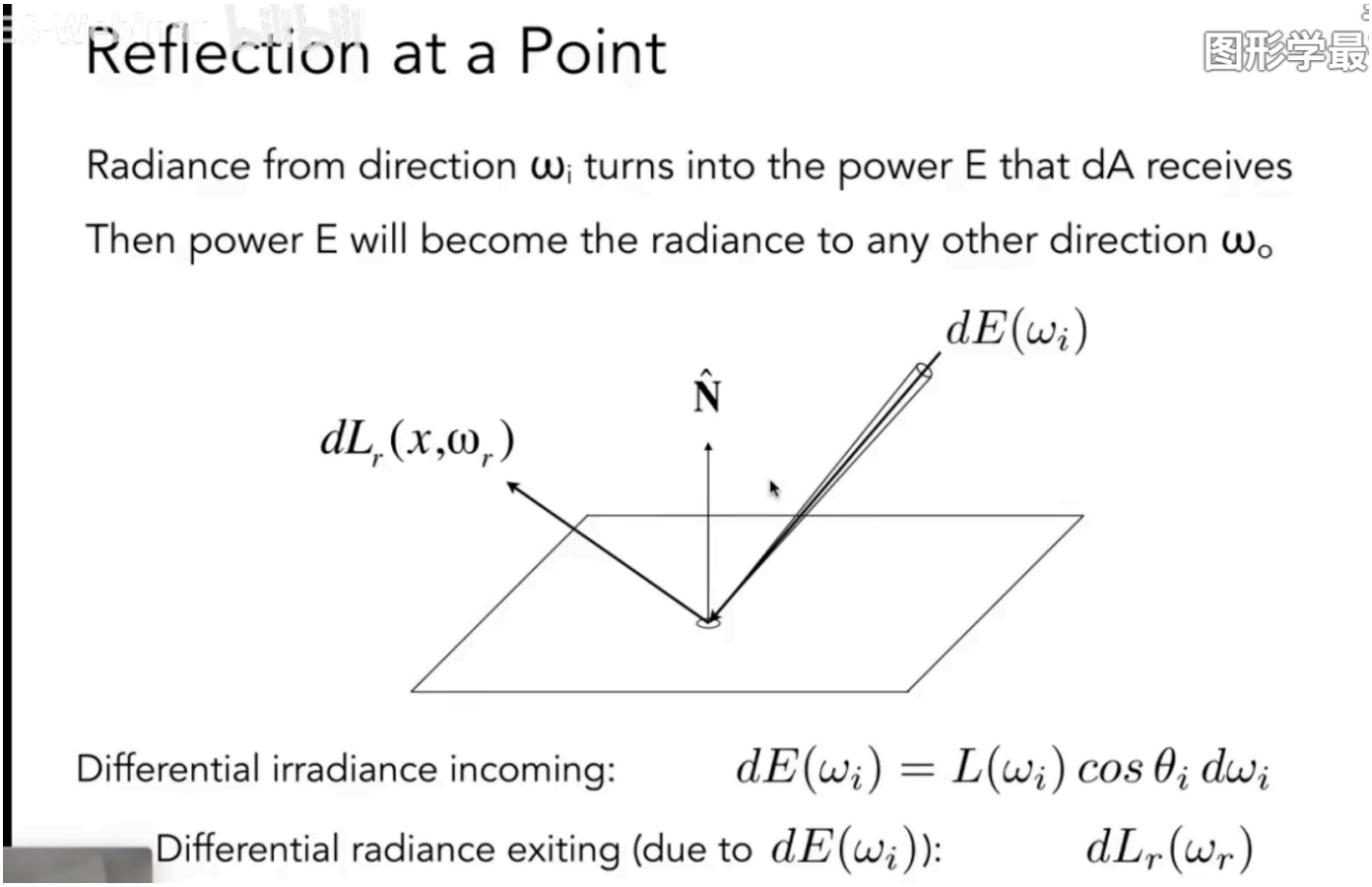
brdf：radiance from direction wi turns into the power E that dA receives，Then power E will become the radiance to any other direction wo

也就是radiance的积分，作为dA表面接收到的能量E，然后又朝特定方向发射radiance

注意：这里是wi方向入射的irradiance，与wr方向出射的radiance之比，也就是说，brdf有两个参数，入射方向和出射方向

$$
f_r(\omega_i \rightarrow \omega_r) = \frac{dL_r(\omega_r)}{dE_i(\omega_i)} = \frac{dL_r(\omega_r)}{L_i(\omega_i)cos\theta_i d\omega_i}
\tag{12}
$$

为什么入射会选择irradiance,而出射选择radiance,可以这么理解：
入射的意思是光线打到平面上，且平面要吸收能量，那radince其实不能代表吸收能量这层含义，因为还要乘上平面与法线的夹角，也就是朗伯余弦定理，而irradiance就能代表这层含义
但是irradiance的表示中没有立体角，因为irradiance是表示通过该平面的所有磁通量，那么只来自于一个方向wi的磁通量，在这里只能表示为dE
另外,我们会发现dE=Lcos(theta)dw，就有朗伯余弦定理



入射方向：我们需要求到达该平面的irradiance(E)

2.渲染方程

（1）rendering equation

$$L_o(p,\omega_o) = L_e(p,\omega_o) + \int_{H^2} L_i(p,\omega_i) f_r(p, w_i \to w_r) cos\theta_i V(p,\omega_i) d\omega_i \tag{13}$$

渲染方程：渲染方程是计算着色点p朝着wo发射出的radiance
着色点p它可能会自发光，所以有一个Le项，表示该点向wo方向发出的radiance
还有一部分能量是反射而来的，反射而来的这部分能量:我们知道brdf是一个入射方向和一个出射方向，那么我们就对这个进行积分，计算每个入射方向对当前出射方向反射的irradiance
dL出 = fr * dE = fr * (dL * cos * dwi)

（2）光源类型

一个点光源：照射物体p只有一条光路

$$L_o(p,\omega_o) = L_e(p,\omega_o) + L_i(p,\omega_i) f_r(p, w_i \to w_r) cos\theta_i d\omega_i \tag{14}$$

多个点光源：就是多条光路，求和

$$L_o(p,\omega_o) = L_e(p,\omega_o) + \sum L_i(p,\omega_i) f_r(p, w_i \to w_r) cos\theta_i d\omega_i \tag{15}$$

面光源：积分面光源对应的立体角「TODO：LTC面光源渲染」

$$L_o(p,\omega_o) = L_e(p,\omega_o) + \int L_i(p,\omega_i) f_r(p, w_i \to w_r) cos\theta_i d\omega_i \tag{16}$$

“光源”有可能是场景中的物体：多次弹射

$$L_o(p,\omega_o) = L_e(p,\omega_o) + \int L_i(p,\omega_i) f_r(p, w_i \to w_r) cos\theta_i d\omega_i \tag{17}$$
$$L_i：\text{光源、场景中物体}(\text{递归计算} L_o)$$

抽象简化成：

「🤔这里左右两边L看上去不一样，但它是矩阵级数收敛的结果，这里不细研究了～」

E：自放光 KE：一次反射 K^2E：两次反射

$$\begin{gathered} l(u) = e(u) + \int l(v) K(u,v) dv \\ L = E + KL \\ L = (1-K)^{-1} E \\ L = (I + K + K^2 + K^3 + \dots) E \\ L = E + KE + K^2 E + K^3 E + \dots \end{gathered} \tag{18}$$

3.数学前提

（0）积分的近似

🌟**积分的近似**：

$$\int_{\Omega} f(x)g(x)dx \approx \frac{\int_{\Omega_G} f(x)dx}{\int_{\Omega_G} dx} \cdot \int_{\Omega} g(x)dx \tag{19}$$

什么时候取等：f(x)是常值函数，或者叫常数

什么时候取约等：（这里f和g是对称的）

- case 1：f/g函数的积分域足够小（support很小）
- case 2：f/g函数足够光滑（也就是近似于常数，波动不大）

（1）积分/求和顺序

$$\int \cdots \sum \cdots \approx \sum \cdots \int \cdots \tag{20}$$

图形学中一般认为积分和求和是可以互相更换顺序的，一般认为是相等的便于推导公式

4.shadow

「利用渲染方程来理解shadow的原理」：shadow_map中我们直接将V项提取出来：

$$L_o(p,\omega_o) = L_e(p,\omega_o) + \int_{H^2} L_i(p,\omega_i) f_r(p, w_i \to w_r) cos\theta_i V(p,\omega_i) d\omega_i$$

$$L_o(p,\omega_o) = L_e(p,\omega_o) + \frac{\int_{\Omega^+} V(p,w_i) d\omega_i}{\int_{\Omega^+} d\omega_i} \int_{H^2} L_i(p,\omega_i) f_r(p, w_i \to w_r) cos\theta_i d\omega_i \tag{21}$$

$$\text{甚至可以写成：} L_o(p,\omega_o) = L_e(p,\omega_o) + V(p,\omega_i) \cdot L_i(p,\omega_i) f_r(p, w_i \to w_r) cos\theta_i d\omega_i$$

shadow_map是一个（点）光源，来判断每个物体是否能被照射，在渲染方程中，这种情况甚至不需要积分

5.环境光遮蔽 - AO

（1）SSAO - 屏幕空间环境光遮蔽

略～后续补充

6.环境光照

（1）IBL

环境光照的主流方式：spherical map & cube map，存储在球面上或者立方体上

IBL：基于图像的照明「图像就是指spherical map 和 cube map」

分析渲染方程：**IBL不考虑遮挡**，所以Visibility项可以舍去

$$L_o(p,\omega_o) = L_e(p,\omega_o) + \int_{H^2} L_i(p,\omega_i) f_r(p, w_i \to w_r) cos\theta_i d\omega_i \tag{22}$$

「实时渲染没办法像离线渲染那样去进行蒙特卡洛追踪，对于求积分来说也是很困难的，我们需要尽量简化渲染方程」

（2）Split Sum

两种特殊的材质：这两种材质都允许通过**拆分积分**来进行近似

- diffuse：它会反射到各个方向，各个方向的反射强度应该差不多，所以说它很smooth，brdf接近于常数
- glossy：它会集中反射到某些立体角区间，这个区间往往特别小，也就是积分域很小

$$L_o(p,\omega_o) = \frac{\int_{\Omega_{f_r}} L_i(p,w_i) d\omega_i}{\int_{\Omega_{f_r}} d\omega_i} \int_{H^2} f_r(p, w_i \to w_r) cos\theta_i d\omega_i \tag{23}$$

Split Sum 1st：L拆出来：其实就是对某个范围采样L再进行平均，也就是**过滤/模糊**操作 => 我们可以提前将环境光图进行不同程度的模糊

Split Sum 2nd：右边brdf也需要积分，得需要预计算

Microfacet BRDF：

$$f(i,o) = \frac{F(i,h)G(i,o,h)D(h)}{4(n,i)(n,o)} \tag{24}$$

F：菲涅尔项 -> 石里克近似

$$\begin{aligned} R(\theta) &= R_0 + (1 - R_0)(1 - cos\theta)^5 \\ R_0 &= (\frac{n_1 - n_2}{n_1 + n_2})^2 \end{aligned} \tag{25}$$

D：法线分布 -> 比如：Beckmann distribution

$$D(h) = \frac{e^{-\frac{tan^2\theta_h}{\alpha^2}}}{\pi\alpha^2 cos^4\theta_h} \tag{26}$$

暂时不考虑G项，我们发现brdf可以抽成三个变量：α、θ_V、R_0，三维度的预计算还是太高了，继续降维度 => 通过以下方式，将R_0拆到积分外面：

「这里的**θ_V是出射方向**或者是观察方向于法线的夹角，而不是入射光线，所以这里的θ_V和积分结尾θ_i是不同的，在积分过程中θ_i是不断变化的值」

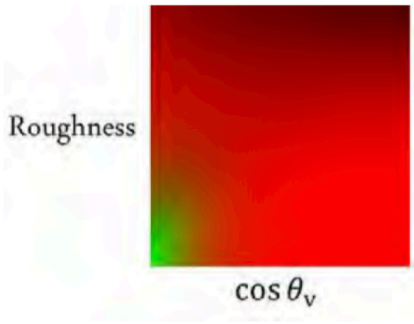
「写成分数形式其实不太直观：其实就是将F项从fr拆出来，那么fr/F就是剩下的D、G等项」

$$\begin{aligned} f_r &= \frac{f_r}{F} \cdot (R_0 + (1 - R_0)(1 - cos\theta)^5 \\ &= \frac{f_r}{F} \cdot (R_0 \cdot (1 - (1 - cos\theta)^5) + (1 - cos\theta)^5) \\ &= R_0 \cdot \frac{f_r}{F}(1 - (1 - cos\theta)^5) + \frac{f_r}{F}(1 - cos\theta)^5 \end{aligned} \tag{27}$$

带入刚刚拆解渲染方程得到的fr的积分中：

$$\begin{aligned}\int_{H^2} f_r(p, w_i \rightarrow w_r) cos\theta_i d\omega_i \approx R_0 \int_{\Omega^+} \frac{f_r}{F}(1-(1-cos\theta_i)^5) cos\theta_i d\omega_i \\ + \int_{\Omega^+} \frac{f_r}{F}(1-cos\theta_i)^5 cos\theta_i d\omega_i \\ \text{抽象成} \approx R_0 \cdot A + B\end{aligned} \tag{28}$$

因此fr的积分只剩下α、θ_V项，α代表roughness，θ可以直接存储它的余弦值cosθ；我们可以得到“两张”**固定的**二维表格（二维数组/纹理），分别存储A和B的预计算结果，甚至可以用**一张图的两个通道**实现：



最后输入α、θ，查询纹理上的A和B，再结合R_0，我们就能构造出fr的积分结果

（3）shadow under Environment Lighting

「shadow，我们是指物体与光源直接是否存在遮挡关系，如果中间有遮挡物，则光源无法照亮物体，产生阴影；对于IBL的cubemap来说，四面八方都有光源，如果我们去考虑每个方向是否有遮挡关系，那么积分又会多一个维度，所以IBL不能做干脆就不做，不考虑遮挡关系，直接拿cubemap照亮场景」

对于IBL来说，其实相当于一个many light的问题，也就相当于有无数个光源去照亮场景（之前做阴影时我们只考虑一个光源的影响），同时Visibility项也是一个难点，因为场景中有无数个光源，那么该物体与光源之间是否被遮挡，难以将V项从积分中提取出来。

解决方案：

- 使用最具有代表性/最亮的光源生成阴影
- Imperfect shadow map
- Light cuts - 将反射物当成光源，将反射物进行归类当成一个面光源
- RTRT - realtime raytracing
- **Precomputed radiance transfer - PRT**

（4）PRT

「PRT能够handle环境光照处理不了阴影的问题」

PRT知识体系：

- 背景知识
  - Frequency and filtering
  - Basis functions
- 实时环境光照/全局光照
  - SH：Spherical Harmonics

- Prefiltered env. lighting
- PRT

Fourier Transform - 傅立叶变换

> 参考文章：https://blog.csdn.net/wangjiangrong/article/details/114322213

傅立叶级数展开：

『

什么是正交：正交是垂直的拓展概念，正交就是内积为0；我们如何定义函数是否正交，引入正交函数（集），也就是乘积和积分的形式，也就是Product Integral【Product Integral本质是一个点乘】

> 维基百科：https://zh.wikipedia.org/wiki/%E6%AD%A3%E4%BA%A4

正交函数（集）：对于两个函数f和g，可以定义如下的内积：

$$< f, g >_w = \int_a^b f(x)g(x)w(x)dx \tag{29}$$

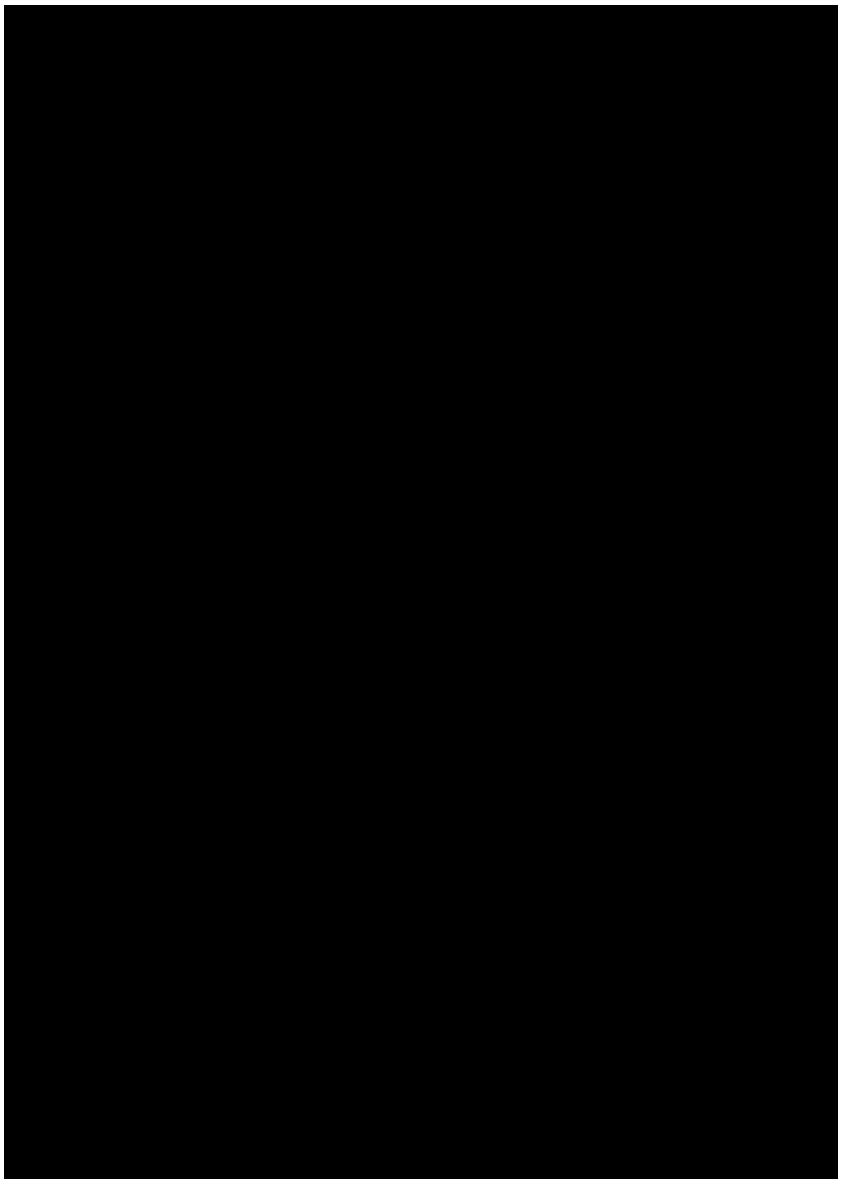
这里引入一个非负的权函数w(x)，这个内积叫做w(x)的内积；如果**两个函数带权w(x)正交**：

$$\int_a^b f(x)g(x)w(x)dx = 0 \tag{30}$$

基函数是正交的，可以依次证明：

- （1）任意余弦与余弦正交
- （2）任意正弦与正弦正交

- （3）任意正弦与余弦正交
- > 参考：（全网独家发布）傅里叶级数（一）——正交性与完备性（全网首篇证明完备性）的证明、傅里叶变换的直观解释以及相关结论在广义积分甚至勒贝积分下的若干深入补充讨论 - 数学达人上官正申的文章 - 知乎 https://zhuanlan.zhihu.com/p/472769406
- 涉及到一个积化和差（或者和差化积）的公式：
-



』

低频信息：变化不剧烈的信息；图片的低频信息：就是拿一个线去“扫描”图像，图像信息波动不大，波动平缓，那就是低频信息

频域：在不同频率下观察信号，我们可以**通过傅立叶变化将空间域/时域的函数f(x)转换为频域上的函数F(w)**的

「注意频域的图像：**靠近中间的区域表示低频的系数**，如果一个频域图像的白点集中在中心，说明这个图像低频信息较多；为什么图像会有一条竖线和横线：这是因为我们平铺图像时，图像的最右侧和最左侧相接，上下侧相接，这里频率变化最快」

大概了解一下：通过低通滤波（Low-pass filter）只保留低频信号，其实就是取频域图像的中间圆圈（中间是低频信息）；在时域上做卷积，相当于在频域上做乘积/或者叫过滤，如下图：

为什么学习频域：为了理解**product integral**，存在一定的滤波意义

- 两个函数乘起来再求积分（对应相乘加起来），可以理解为卷积或者过滤操作，可以把f(x)和g(x)分别看成两个频谱相乘
- 🤔我觉得可以这么理解：一个函数可以拆成多个基函数，基函数之间是正交的，那么现在两个函数相乘，相当于基函数对应相乘再求和 => 那么如果一个函数的信息集中在低频，相当于高频的系数接近0，那对应相乘会把另一个函数的高频信息弄丢，这也就是低通滤波的作用
-

$$\int_{\Omega} f(x)g(x)dx \tag{31}$$

基函数（Basic Function）：比如傅立叶级数展开

$$f(x) = \sum_{i} c_i \cdot B_i(x) \tag{32}$$

球谐函数（Spherical Harmonics）：球谐函数是一系列二维的基函数，定义在球面上的 => 二维：球面坐标系的θ、ϕ（俯仰角和方位角）

球谐函数具体怎么写出来要使用勒让德多项式，在课程中更关注球谐应用层面

> 拓展阅读：球谐函数基础一 - 和合的文章 - 知乎 https://zhuanlan.zhihu.com/p/467466131 - 解释球谐函数具体是什么，球谐的函数实际上很简单

每一行的函数族具有相同的频率；第l行有2l+1个函数，分别标号为-l～l；前n阶有个(n+1)^2函数；

问第l阶第m个函数的在整个球谐数组的下标是多少：我们可以根据m=0时，下标为l*(l+1)，再用m做偏移，所以下标为：

$$index_{(l,m)} = l \times (l+1) + m \tag{33}$$

最重要的性质：B_i是第i个球谐基函数，c_i是其对应的系数，我们能通过对f(x)与某个球谐基函数进行product integral，得到对应的系数 => 这种操作我们叫做**投影**，将f(x)投影到某个基函数上

$$c_i = \int_{\Omega} f(\omega)B_i(\omega)d\omega \tag{34}$$

依次计算每阶的系数，我们可以恢复出原函数

对于diffuse brdf来说,前三阶球谐已经能大致描述brdf(diffuse);
那么既然只拿前3阶描述brdf,那也就只需要用前3阶来描述光照

回到渲染方程：

$$L(o) = \int_{\Omega} L(i)V(i)\rho(i,o)max(0,n\cdot i)di \tag{35}$$

目前是三个函数相乘的积分,假设要实时计算积分,工作量很大：
根据当前的wo,分别去采样三个函数在不同入射方向的值然后积分起来

PRT的“解题思路”：

将渲染方程中L项与其他项拆开，分为lighting和light transport项，light transport项可以认为是顶点的固有属性，是我们可以预计算出来的 => 这种方式**只能处理静态场景**（因为Visibility项是预计算而固定）

Diffuse Case：brdf为常值

- $$
\begin{aligned}
L(o) &= \rho \int_{\Omega} L(i)V(i)max(0, n \cdot i)di \\
L(i) &= \sum l_i B_i(i) \\
L(o) &\approx \rho \int_{\Omega} \sum l_i B_i(i)V(i)max(0, n \cdot i)di \\
L(o) &\approx \rho \sum l_i \int_{\Omega} B_i(i)V(i)max(0, n \cdot i)di \\
&\text{观察右侧积分的形式, 相当于将} light\ \ transport \text{投影到基函数上} \\
L(o) &\approx \rho \sum l_i T_i
\end{aligned} \tag{36}
$$

- 所以我们需要分别计算light和light transport投影到球谐函数上的各系数，我们将各系数根据对应阶数相乘就得到了我们的结果（点乘）
- 🤔有一种更简单的思考方式：我们直接将light和light transport项投影到对应基函数上，再利用球谐函数的正交性得到是对应系数的点积
  - $$
\begin{aligned}
L(o) &= \rho \int_{\Omega} L(i)V(i)max(0, n \cdot i)di \\
L(o) &= \rho \int_{\Omega} (\sum l_i B_i) \cdot (\sum T_i B_i)di \\
L(o) &= \rho \int_{\Omega} (l_0 B_0 + l_1 B_1 + \ldots + l_n B_n) \cdot (T_0 B_0 + T_1 B_1 + \ldots + T_n B_n)di \\
L(o) &= \rho \int_{\Omega} (l_0 B_0 \cdot T_0 B_0) + \rho \int_{\Omega} (l_1 B_1 \cdot T_1 B_1) + \ldots \quad \text{【系数相同, 积分为1】} \\
&+ \rho \int_{\Omega} (l_0 B_0 \cdot T_1 B_1) + \ldots \text{【系数不相同, 积分为0】} \\
&= \sum l_i T_i
\end{aligned} \tag{37}
$$

球谐函数的优秀性质：

- **正交性**
  - 球谐函数的基函数投影到自己积分为1，投影到其他基函数积分为0 => 正交性

$$
\begin{aligned}
\int_{\Omega} B_i(i) \cdot B_j(i)di &= 1 \quad (i = j) \\
\int_{\Omega} B_i(i) \cdot B_j(i)di &= 0 \quad (i \neq j)
\end{aligned} \tag{38}
$$

- 投影和重建操作很简单
- 支持简单的旋转：比如我们旋转光源，那么旋转后光源对应的基函数系数有一个特殊的规律，就是系数是同阶基函数的线性组合

具体的预计算过程：

- 预计算
  - light：投影到每个基函数上，投影/积分计算中，对球面进行采样，采样方向得到L，再乘以基函数在该方向上的值，求和
  - light transport：我们是**逐顶点**进行计算，投影到每个基函数上，投影/积分计算中，对球面进行处采样，采样方向得到Visibility项，以及cosθi，再乘以基函数在该方向上的值，求和
  - 预计算的结果：假如说我们用前3阶球谐函数来模拟，那么light就是4×3个系数，（×3是因为光是rgb），那么light transport就是4×3×顶点数目
- shader中还原：light transport我们作为顶点属性数组传入，light作为uniform传入，在shader中进行对应相乘，得到数据

其他问题：

- 问题1：给一个cubemap,每一个像素对应的立体角是多大,或者叫投影到单位球面的面积是多大?
  - > 参考文章：https://www.cnblogs.com/redips-l/p/10976173.html
  -
  - 假设我们的cubemap的一个面在z=1上，对于投影面积我们通过三步求得：
    - 计算cubemap上点P(x,y,1)投影到球面上的坐标P'
    - 对于P'分别求关于x和y的偏导数，偏导数的叉乘的模长就是**微表面的面积（叉乘的模长**的几何意义：向量围成的平行四边形面积）
    - 我们在x和y方向上对微表面的面积积分，得到我们所要的面积
  -

投影到球面上(球放在原点) = 该向量除以它的模，或者说求它的单位向量

$$
\begin{aligned}
p &= \frac{(x,y,1)}{\sqrt{x^2+y^2+1}} \\
\frac{\partial p}{\partial x} &= \frac{(y^2+1,-xy,-x)}{(x^2+y^2+1)^{\frac{3}{2}}} \\
\frac{\partial p}{\partial y} &= \frac{(-xy,x^2+1,-y)}{(x^2+y^2+1)^{\frac{3}{2}}} \\
dS = ||\frac{\partial p}{\partial x} \times \frac{\partial p}{\partial y}|| &= ||\frac{(x,y,1)}{(x^2+y^2+1)^2}|| = \frac{1}{(x^2+y^2+1)^{\frac{3}{2}}}
\end{aligned} \tag{39}
$$

在x，y上求积分，为了方便，我们可以求从 $\int_{y=0}^{t}\int_{x=0}^{s}$ 的积分$f(s,t)$

$$
\begin{aligned}
f(s,t) &= \int_{y=0}^{t}\int_{x=0}^{s}\frac{1}{(x^2+y^2+1)^{\frac{3}{2}}}dxdy \\
&= tan^{-1}\frac{st}{\sqrt{s^2+t^2+1}}
\end{aligned}
$$

那么对于四边形$ABCD$，对应的球面投影面积：$S = f(A) - f(B) - f(D) + f(C)$

再根据立体角定义：$d\omega = \frac{dA}{r^2}$

- 那么这是对于cubemap求立体角，但是对于顶点求立体角，就很简单了，因为我们会随机发射光线，所以我们认为我们是均匀的采样，那每个的立体角就是：
-

$$
\begin{aligned}
d\omega &= \frac{A}{r^2} \\
&= \frac{4\pi r^2}{r^2} = 4\pi \text{【对于球来说】}
\end{aligned} \tag{40}
$$

- 问题2：如何推导&处理球谐的旋转?
  - 球谐具有的性质：
    - 旋转不变性：

$$
\text{对原函数}f(x)\text{进行}R\text{旋转：}R(f(x)) = f(R(x)) \tag{41}
$$

    - 对每层band上的SH coefficient，可以分别在上面进行旋转，并且这个旋转是线性变换：

$$
\begin{aligned}
&\text{意味着如果给定某一层上}SH\text{系数列表}t = (t_0, t_1, \ldots, , t_{k-1}) \\
&\text{如果将某个旋转}R\text{应用在其上，存在}k\text{阶方阵}M_R\text{使得：} \\
&tM_R = t'
\end{aligned} \tag{42}
$$

  - 快速球谐旋转是通过性质2👆推导的：

    我们假设P是指某层band上投影的球谐系数列表t（P里面的自变量是x，其实是w_i，Games202作业描述上写的是n_i，这三者是等价的）：

$$
\begin{gathered}
P(x) = [t_{-l}, \ldots, t_l] \\
\text{存在旋转矩阵} M \text{使得：【性质2】} \\
P(x)M = P(R(x))
\end{gathered} \tag{43}
$$

整理得到：

$$
\begin{gathered}
[P_{-l}, \ldots, t_l]M = P(R(x)) \\
\text{记} A = [t_{-l}, \ldots, t_l], \text{如果矩阵} A \text{是可逆矩阵, 此时易得：} \\
\text{【这里的} A \text{目前是一个行向量, 我们要采样填充} 2l+1 \text{行, 将其转换为一个矩阵】} \\
M = A^{-1}P(R(x))
\end{gathered} \tag{44}
$$

⚠️这里注意：这里A是一个行/列矩阵，它不是方阵，所以严格意义来说它没有逆矩阵（就算是方阵也可能没有逆矩阵），P(R(x))同理 => **所以我们需要填充，将其构造成一个矩阵**

对于第l层的band，构造矩阵的方式是，目前有1行2l+1列，所以我们需要取2l+1行/组数据传入进去，每一组数据都是**随机选择的w_i**

=> 选取的条件：（1）我们要保证选取的w_i能让A为可逆矩阵（2）可以选取一些恰当的例子让A包含更多的0，从而减少运算量

任何一个矩阵它都不一定有逆矩阵 => 所以这个旋转叫做“Simple and Fast Spherical Harmonics Rotation”，因为这比较快速，但不太准确的做法

🤔：所以games202的推导是正确的，它这里M[P(n_{-l}), ..., P(n_l)] = [P(R(n_{-l})), ..., P(R(n_{-l}))]已经进行了2l+1个采样，但是它没有类似👆的推导过程，容易让人看的一头雾水：

具体预计算步骤：

- 构造A，并计算A的逆矩阵
- 计算P(R(n_i)) i∈[-l, l]，对于任意的R，都要重新计算投影，这里与预计算不是背道而驰吗？=> 但实际上就是这么做的，原因之一是：因为只需要旋转光照，三阶球谐×rgb也就9×3个系数；原因之二如下👇
  - 我们分析一下为什么要这么做，是否是多次一举：（1）如果我们直接将旋转后的光函数进行投影，那么对于每个球谐基函数来说，投影要采样点吧，起码>100个采样点吧，那么这个计算量相当大（2）但是我们观察我们的做法，我们只选择了2l+1个wi方向，进行投影，那么这个计算量绝对是可以接受的，可以进行实时计算

- 问题3：如何计算间接光照?
  - 光线传输方程变为：

$$
\begin{gathered}
L_{DI} = L_{DS} + \frac{\rho}{\pi}\int_S \hat{L}(x', \omega_i)(1 - V(\omega_i))max(N_x \cdot \omega_i, 0)d\omega_i \\
L = \frac{\rho}{\pi}\int_S (L(\omega_i)V(\omega_i) + \hat{L}(x', \omega_i)(1 - V(\omega_i))max(N_x \cdot \omega_i, 0)d\omega_i
\end{gathered} \tag{45}
$$

L_{DS}（diffuse shadowed）是直接光源，后面的积分是计算间接光源，间接光源是指，光源首先打到某物体，再从某物体反射打到当前着色点，所以这里Visibility项取反（1-visibility）是因为光源不会直接打到着色点，也就是说我们考虑的情况正好是该方向有遮挡物的情况

具体步骤：

- 对于每个顶点，计算L_{DS}

    - 从当前顶点发射光线，如果与其他三角形相交，在交点处通过重心坐标插值得到球谐系数，这个系数表示间接光照的球谐系数
    - 对于这个反射回来的间接光L(x', ω)，乘以几何项$N_x \cdot \omega_i$
    - 计算L_{DS}
- 问题4：为什么games202方程中要除以π?
    -
    - 简言之就是因为这个积分：
    -

$$\int_{\Omega} cos\theta d\omega = \int_0^{2\pi} \int_0^{\pi} cos\theta(sin\theta d\theta d\Phi) = \pi \tag{46}$$

    - 具体推导：这里的ρ表示的其实是albedo，也就是说diffuse物体的brdf：

$$diffuse物体：f_r = \frac{albedo}{\pi} \tag{47}$$

    首先我们要明白albedo的物理定义：也就是辐射度与辐照度之比

$$albedo = \frac{\sum L_o}{\sum L_i} \tag{48}$$

    我们回到渲染方程，对于diffuse物体来说：

$$\begin{aligned} L_o(p, \omega_o) &= \int f_r \cdot L_i(p, \omega_i) \cdot cos\theta d\omega \\ &= (albedo \cdot L)\frac{\int cos\theta d\omega}{k} \quad 【假设k为归一化系数】 \\ albedo \cdot L已经等于L_o了，所以k &= \int cos\theta d\omega \end{aligned} \tag{49}$$

- > 参考文章：https://zhuanlan.zhihu.com/p/342807202

Games202 - homework2：

> PS：因为写的.cpp，所以写好记得重新编译，不然没有结果

- light.txt：
    - 前三阶球谐，所以有九个系数，下面每行对应“一个”系数；但这里其实不是“一个”系数，而是三个系数，因为L有rgb值，需要分别投影
    -
- light transport.txt：
    - 如果有n个顶点，那么数据就有n行，每行是球谐系数列表，比如我们是前3阶，那么是9个数，这里一个顶点不像L有rgb值，顶点只会对应一个值
    -
- reconstruction：

```
// ps:这不是在vertex shader中!!!
// 我们计算三角形▲012上某点的着色情况
auto sh0 = m_TransportSHCoeffs[0]; // 顶点0的球谐列表
auto sh1 = m_TransportSHCoeffs[1]; // 顶点1的球谐列表
auto sh2 = m_TransportSHCoeffs[1]; // 顶点2的球谐列表

// 顶点和光照对应分量
Color3f c0 = Color3f(rL.dot(sh0), gL.dot(sh0), bL.dot(sh0)), // 顶点0的光照
        c1 = Color3f(rL.dot(sh1), gL.dot(sh1), bL.dot(sh1)), // 顶点1的光照
        c2 = Color3f(rL.dot(sh2), gL.dot(sh2), bL.dot(sh2)); // 顶点2的光照

const Vector3f &bary = its.bary; // 插值系数/三角形重心坐标
Color3f c = bary.x() * c0 + bary.y() * c1 + bary.z() * c2;

return c;
```

- shader中：假如我们使用三阶球谐，就是9个系数，我们**可以通过mat3来存储这九个数**

```
attribute mat3 aPrecomputeLT; // Light Transport(顶点属性)
uniform mat3 uPrecomputeL[3]; // Light(uniform)

float L_dot_LT(mat3 PrecomputeL, mat3 PrecomputeLT)
{
    // 将mat3拆回成3个vec3
    vec3 L_0 = PrecomputeL[0];
    vec3 L_1 = PrecomputeL[1];
    vec3 L_2 = PrecomputeL[2];
    vec3 LT_0 = PrecomputeLT[0];
    vec3 LT_1 = PrecomputeLT[1];
    vec3 LT_2 = PrecomputeLT[2];
    return dot(L_0, LT_0) + dot(L_1, LT_1) + dot(L_2, LT_2);
}

void main()
{
    for(int i = 0; i < 3; i++)
    {
        vColor[i] = L_dot_LT(aPrecomputeL[i], aPrecomputeLT);
    }
    ...
}
```

Glossy Case：

-

$$
\begin{aligned}
L(o) &= \int_{\Omega} L(i)V(i)\rho(i,o)max(0,n\cdot i)di \\
L(o) &\approx \sum l_i T_i(o) \\
T_i(o) &\approx \sum t_{ij} B_j(o) \\
\therefore L(o) &\approx \sum (\sum l_i t_{ij}) B_j(o)
\end{aligned} \tag{50}
$$

-
- brdf此时不再是常数，因为出射方向wo可以任意指定，所以此时每个顶点的Light Transport不再是固定的，而是关于o的函数
- 🤔：原先light transport是关于wi的函数，所以我们投影一次就能得到结果；但glossy case引入了brdf，让light transport是关于wi和wo的函数，那么我们先关于wi进行球谐投影，再关于wo进行球谐投影 => 投影一次能消除一个方向

球谐后续：

- 人们会使用前3、4、5阶球谐来模拟
- 球谐不太适合描述高频信息，可以使用更好的基函数进行模拟

球面坐标系

三个参数定义：半径（r），极角（θ），方位角（φ）

混合偏导数

$$
\begin{aligned}
\frac{\partial^2 z}{\partial x \partial y} &= f''_{xy}(x,y) = \frac{\partial}{\partial y}(\frac{\partial f}{\partial x}) \\
&\neq \frac{\partial z}{\partial x}\cdot\frac{\partial z}{\partial y}
\end{aligned} \tag{51}
$$

继续推导：

$$
\begin{aligned}
\frac{\partial^2 z}{\partial x \partial y} &= \frac{\partial}{\partial y}(\frac{\partial f}{\partial x}) \\
\text{假设：} L &= \frac{\partial^2 z}{\partial x \partial y}, E = \frac{\partial f}{\partial x} \\
\text{那么：} L &= \frac{\partial}{\partial y}(E) = \frac{\partial E}{\partial y}
\end{aligned} \tag{52}
$$

Solid Angles

立体角是角度在三维空间的延伸

- 角(弧度制)：弧长 / 半径
- 立体角：投影到球面上的面积 / 半径的平方

🤔记忆：可以这么记忆，对于角度来说，可以把l投影到r边上，所以是l/r；而立体角来说，是将A投影到圆心的平面上，所以是A/r^2【当然这肯定是不准确的，只是方便记忆】

立体角 - 微积分相关：

分析：该微分面积可以看成一个长方形，分别是θ和φ的方向；因为弧长=半径*角度，所以θ方向的长度为rdθ；而φ方向要注意，它投影到底面一定长度小于半径r，所以我们需要找到它的对应半径，其实我们作辅助线，在上面构造一个和底面平行的圆，这个圆的半径是rsinθ，故得证

🎨相关属性查表！

1.球谐函数

2.菲涅尔系数 - F_0