

Iluminação

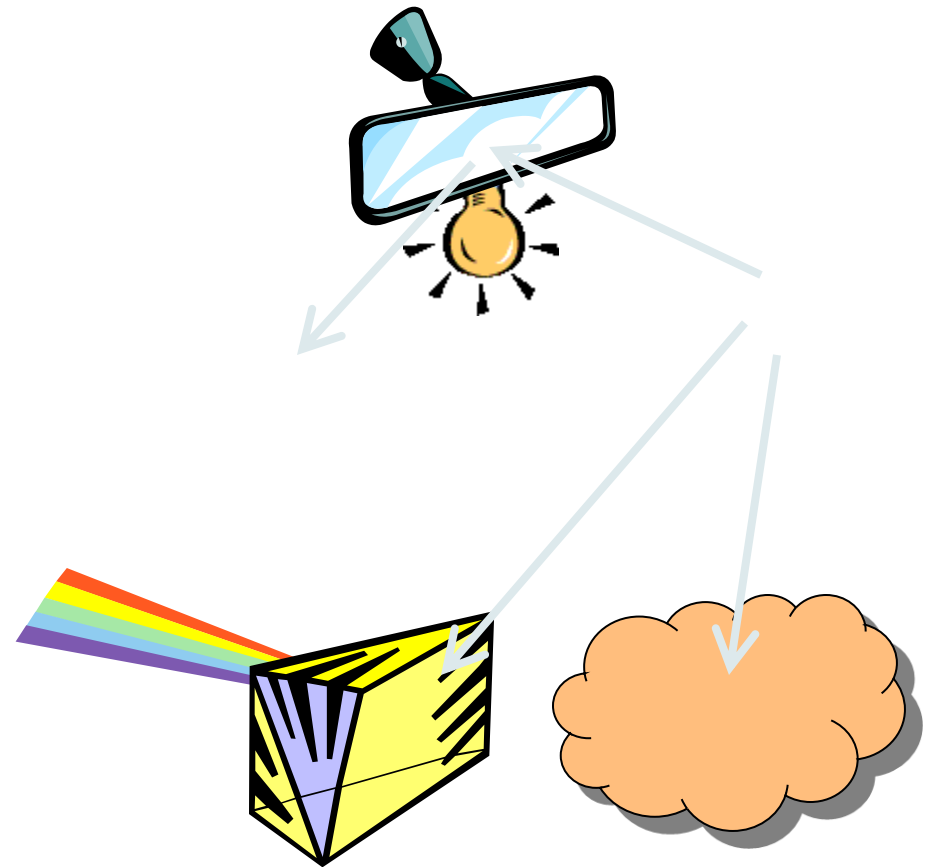
Márcio Bueno

{cgtarde,cgnoite}@marciobueno.com)

Fonte: Material do Prof. Claudio Esperança
e do Prof. Paulo Roma Cavalcanti

Iluminação

- ▶ Estudo de como a luz interage com objetos de uma cena
 - ▶ Emissão
 - ▶ Transmissão
 - ▶ Absorção
 - ▶ Refração
 - ▶ Reflexão



Modelo Físico

- ▶ Luz modelada como radiação eletromagnética
- ▶ Leva em conta todas as interações (todos os caminhos da luz)
- ▶ Intratável computacionalmente

Modelos de Iluminação

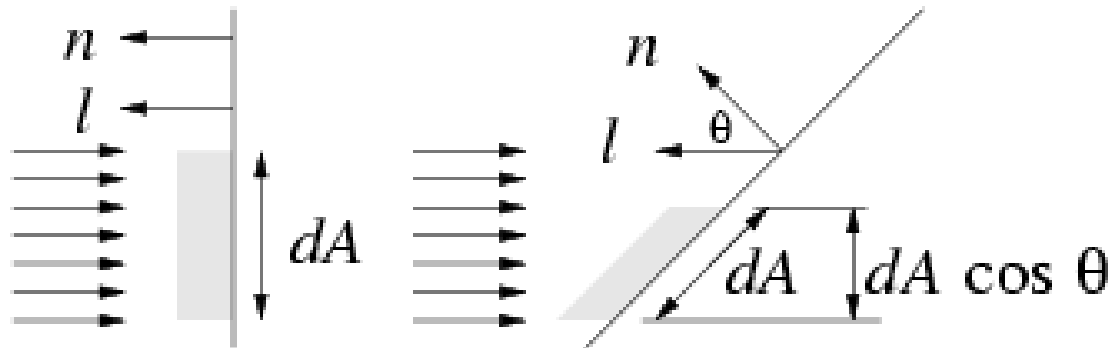
- ▶ Tipicamente, a luz é amostrada em um número discreto de primárias (comprimentos de onda)
- ▶ Modelos locais (primeira ordem)
 - ▶ Apenas caminhos do tipo *fonte luminosa* → *superfície* → *olho* são tratados
 - ▶ Simples
 - ▶ Ex.: OpenGL
- ▶ Modelos globais
 - ▶ Muitos caminhos (*ray tracing*, *radiosidade*)
 - ▶ Complexos

Modelo de Booknight

- ▶ **Considera apenas a reflexão difusa.**
 - ▶ Iluminação recebida em um ponto de uma superfície é refletida uniformemente em todas as direções
- ▶ **Contribuição proveniente da iluminação recebida de forma indireta é modelada como uma constante.**
- ▶ **Baseia-se apenas na reflexão *lambertiana*.**

Iluminação Difusa

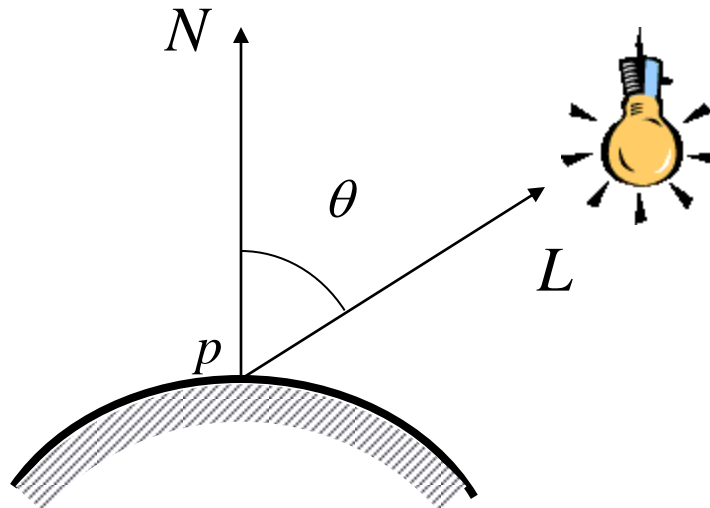
- ▶ Característica de materiais foscos.
- ▶ Lei de Lambert (fluxo de energia):
 - ▶ a luminosidade aparente da superfície não depende da posição do observador, mas apenas do cosseno do ângulo entre a normal e a direção da luz



Modelo Difuso

- ▶ Intensidade em um ponto p é dada por:

$$I_p = I_a k_a + I_d k_d \cos \theta = I_a k_a + I_d k_d \left(\frac{L \cdot N}{|L||N|} \right)$$



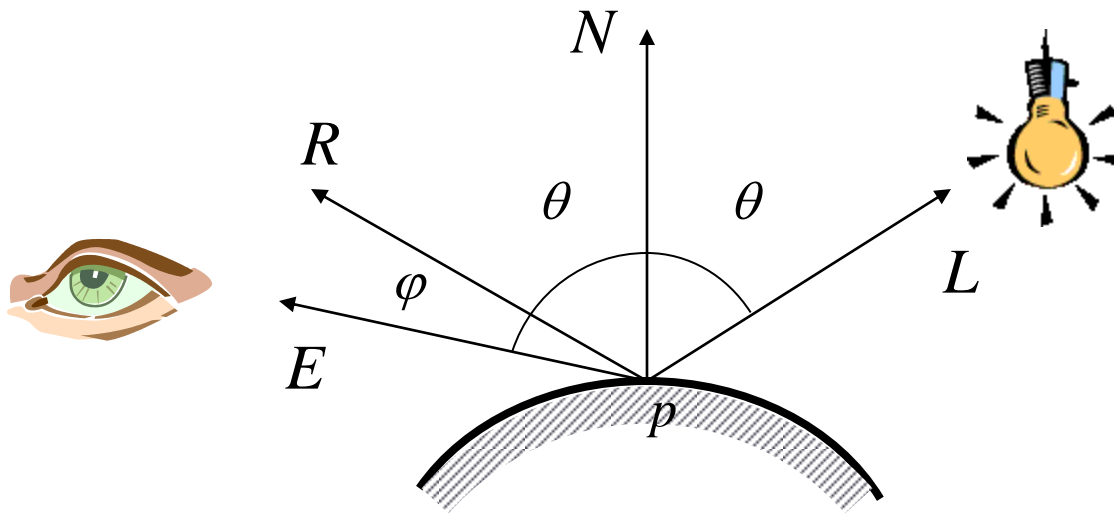
Iluminação Especular

- ▶ Simula a reflexão à maneira de um espelho (objetos altamente polidos).
- ▶ Depende da disposição entre observador, objeto e fonte de luz.
- ▶ Em um espelho perfeito, a reflexão se dá em ângulos iguais
 - ▶ Observador só enxergaria a reflexão de uma fonte pontual se estivesse na direção certa.
- ▶ No modelo de Phong, simulam-se refletores imperfeitos assumindo que luz é refletida segundo um *cone* cujo eixo passa pelo observador.

Modelo de Phong

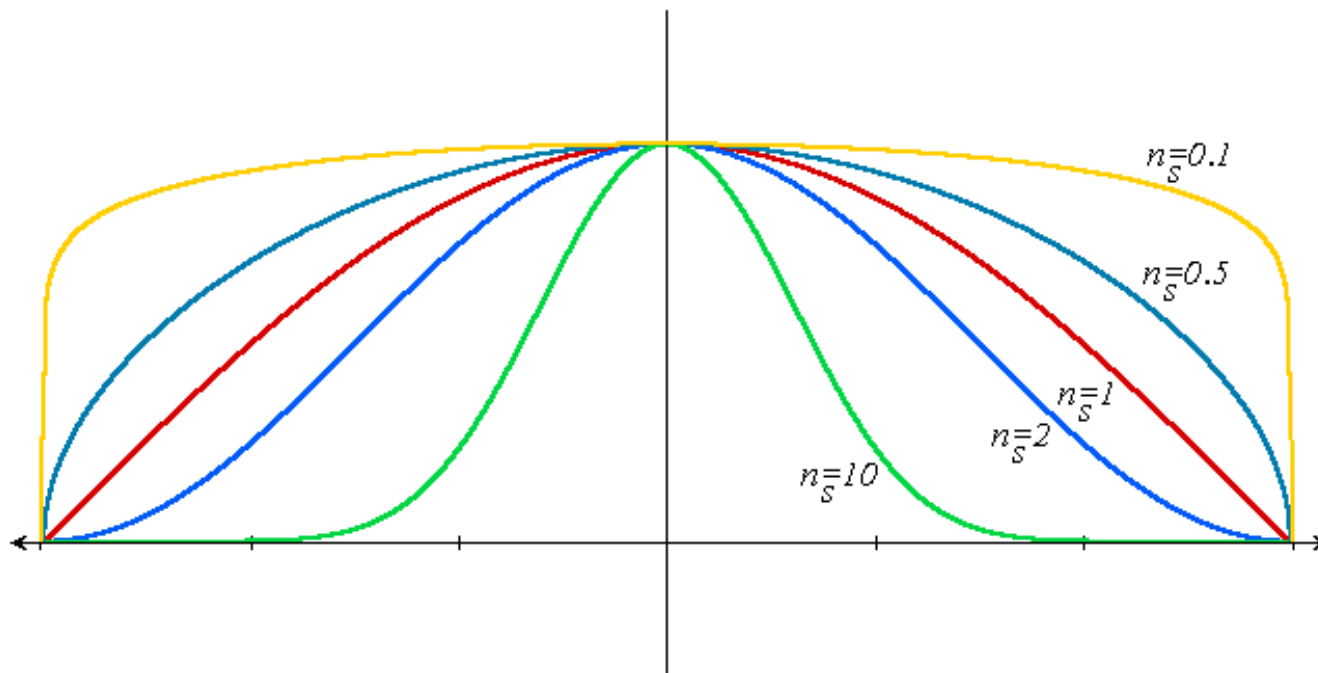
- Contribuição especular é dada por

$$I_p = I_e k_e \cos^n \varphi = I_e k_e \left(\frac{R \cdot E}{|R||E|} \right)^n$$



Coeficiente de Especularidade

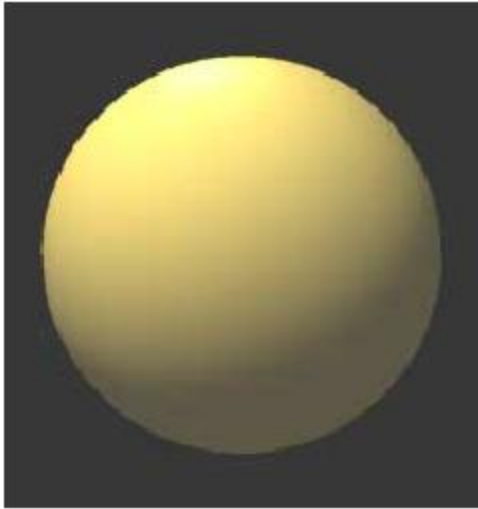
- ▶ Indica quão polida é a superfície
 - ▶ Espelho ideal tem especularidade infinita
 - ▶ Na prática, usam-se valores entre 5 e 100



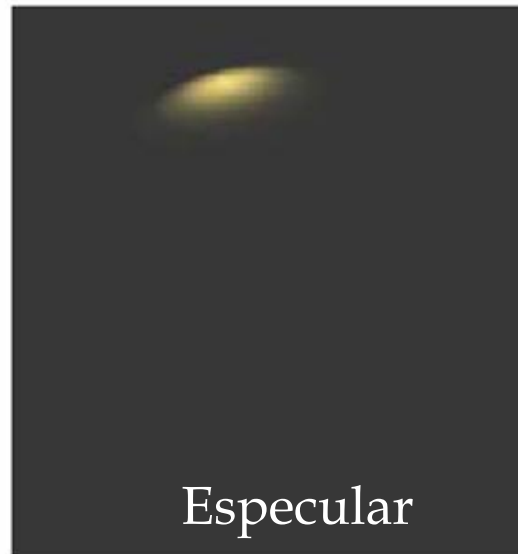
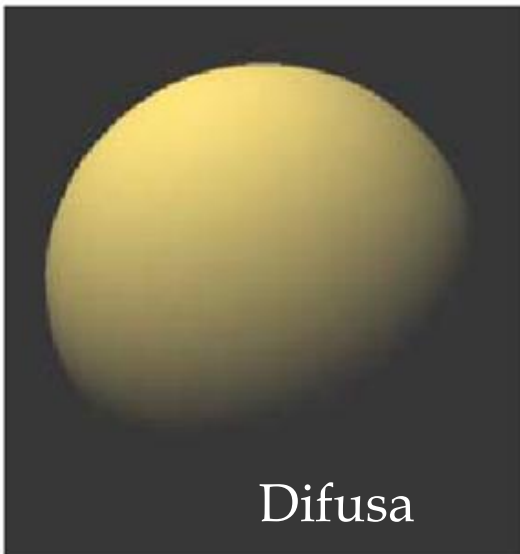
Componentes do Modelo de Phong

- ▶ Emissão: contribuição que não depende de fontes de luz (fluorescência)
- ▶ Ambiente: contribuição que não depende da geometria
- ▶ Difusa: contribuição correspondente ao espalhamento da reflexão *lambertiana* (independe da posição do observador)
- ▶ Especular: contribuição referente ao comportamento de superfícies polidas

Componentes do Modelo de Phong



$$I_p = I_a k_a + I_d k_d \langle L, N \rangle + I_e k_e \langle H, N \rangle^n$$



Iluminação em OpenGL

- ▶ Assume fontes pontuais de luz
 - ▶ Onidirecionais
 - ▶ *Spot*
- ▶ Interações de luz com superfície modeladas em componentes (modelo de *Phong*):
 - ▶ Emissão
 - ▶ Ambiente
 - ▶ Difusa
 - ▶ Especular

Iluminação em OpenGL

- ▶ Suporte a efeitos atmosféricos como:
 - ▶ *Fog*
 - ▶ Atenuação
- ▶ Modelo de iluminação é computado apenas nos vértices dos polígonos.
- ▶ Suporta *Gouraud shading*
 - ▶ Cor dos pixels no interior dos polígonos é obtida por interpolação linear.

Fontes de Luz

- ▶ Para ligar uma fonte: `glEnable (source) ;`
 - ▶ *source* é uma constante cujo nome é `GL_LIGHTi`, começando com `GL_LIGHT0`
 - ▶ Quantas? Pelo menos 8, mas para ter certeza:
 - ▶ `glGetIntegerv(GL_MAX_LIGHTS, &n) ;`
- ▶ Não esquecer de ligar o cálculo de cores pelo modelo de iluminação
 - ▶ `glEnable (GL_LIGHTING) ;`

Fontes de Luz

- ▶ Para configurar as propriedades de cada fonte:
`glLightfv(source, property, value);`
- ▶ **Property** é uma constante designando:
 - ▶ Coeficientes de cor usados no modelo de iluminação:
 - **`GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR`**
 - ▶ Geometria da fonte
 - **`GL_POSITION, GL_SPOT_DIRECTION, GL_SPOT_CUTOFF, GL_SPOT_EXPONENT`**
 - ▶ Coeficientes de atenuação
 - **`GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, GL_QUADRATIC_ATTENUATION`**

Propriedades de Material

- ▶ Especificados por

 - `glMaterialfv (face, property, value)`

 - ▶ *Face* designa quais lados da superfície se quer configurar:

 - ▶ `GL_FRONT, GL_BACK, GL_FRONT_AND_BACK`

 - ▶ *Property* designa a propriedade do modelo de iluminação:

 - ▶ `GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS`

Geometria

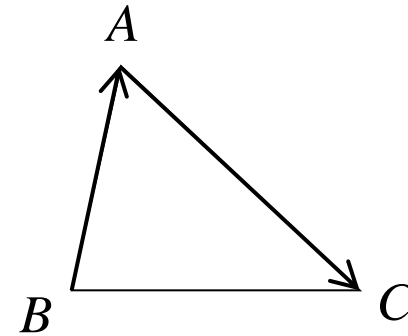
- ▶ Além das propriedades da luz e do material, a geometria do objeto também é importante
 - ▶ A posição dos vértices com relação ao olho e à fonte luminosa contribui no cálculo dos efeitos atmosféricos
 - ▶ A *normal* é fundamental
 - ▶ Não é calculada automaticamente
 - ▶ Precisa ser especificada com `glNormal ()`

Computando o Vetor Normal

- ▶ **Triângulo**

- ▶ Dados três vértices,

$$\vec{n} = \text{normalizar}((A - B) \times (C - A))$$



- ▶ **Polígono planar**

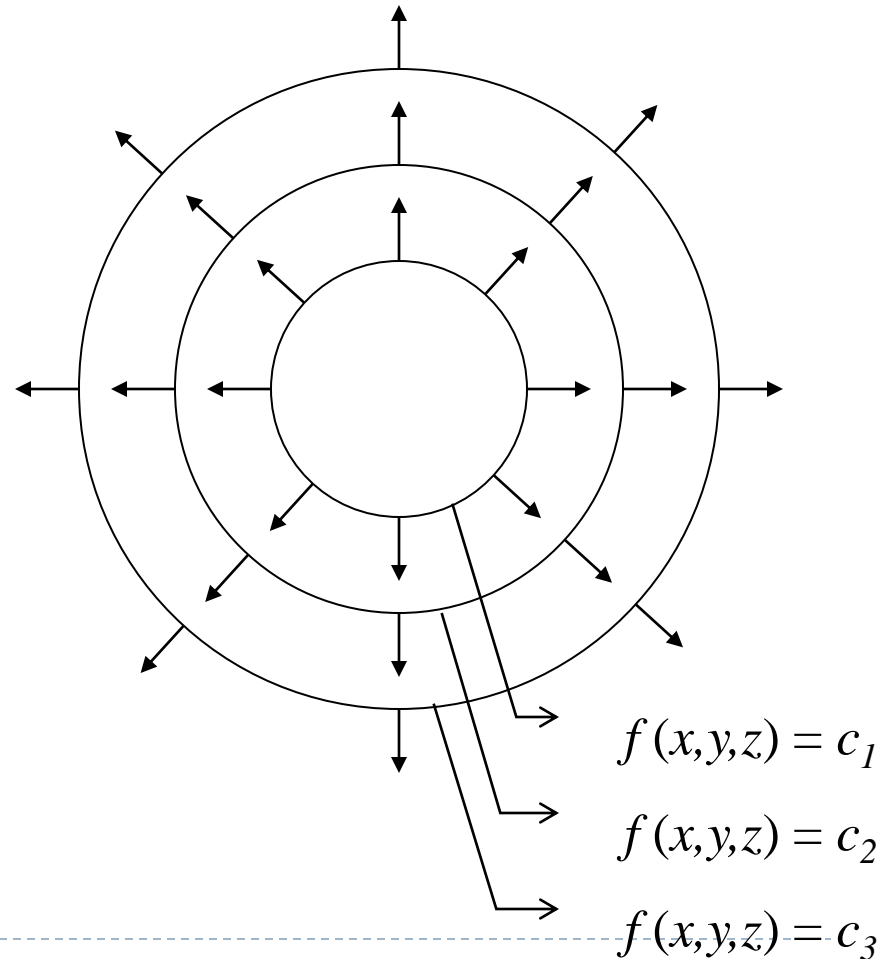
- ▶ Uma opção é usar a fórmula do triângulo para quaisquer 3 vértices
 - ▶ Sujeito a erros (vetores pequenos ou quase colineares)
- ▶ Outra opção é determinar a equação do plano
 - ▶ $ax + by + cz + d = 0$
 - ▶ Normal tem coordenadas (a, b, c)

Calculando o Vetor Normal de Superfícies Implícitas

- ▶ Normal é dada pelo vetor gradiente

$$f(x, y, z) = 0$$

$$\vec{n} = \begin{pmatrix} \partial f / \partial x \\ \partial f / \partial y \\ \partial f / \partial z \end{pmatrix}$$

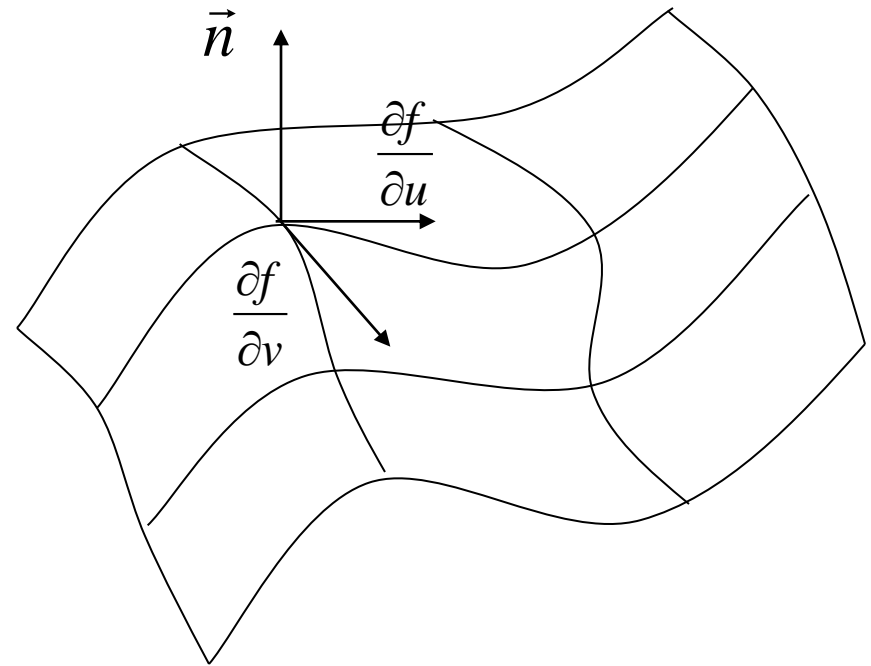


Calculando o Vetor Normal de Superfícies Paramétricas

- ▶ Normal é dada pelo produto vetorial dos gradientes em relação aos parâmetros u e v

$$P = \begin{pmatrix} f_x(u, v) \\ f_y(u, v) \\ f_z(u, v) \end{pmatrix}$$

$$\vec{n} = \frac{\partial f}{\partial u} \times \frac{\partial f}{\partial v} = \begin{pmatrix} \partial f_x / \partial u \\ \partial f_y / \partial u \\ \partial f_z / \partial u \end{pmatrix} \times \begin{pmatrix} \partial f_x / \partial v \\ \partial f_y / \partial v \\ \partial f_z / \partial v \end{pmatrix}$$



Iluminação Ambiente

- ▶ Componente que modela como uma constante o efeito da reflexão de outros objetos do ambiente
- ▶ Depende dos coeficientes `GL_AMBIENT` tanto das fontes luminosas quanto dos materiais
- ▶ É ainda possível usar luminosidade ambiente não relacionada com fontes luminosas
 - ▶ `glLightMaterialfv (GL_LIGHT_MODEL_AMBIENT, params)`
- ▶ Contribuição é dada por

$$A = I_a k_a$$

Atenuação

- ▶ Para fontes de luz posicionais ($w = 1$), é possível definir um fator de atenuação que leva em conta a distância d entre a fonte de luz e o objeto sendo iluminado
- ▶ Coeficientes são definidos pela função `glmLight` ()
- ▶ Por default, não há atenuação ($c_0=1, c_1=c_2=0$)

$$aten = \frac{1}{c_0 + c_1d + c_2d^2}$$

Juntando tudo

- ▶ A atenuação só é aplicada sobre às componentes difusa e especular.
- ▶ A fórmula que calcula a cor de um vértice devida a uma fonte luminosa i é dada por:

$$C_i = A_i + \text{aten} (D_i + S_i)$$

- Portanto, no total, a cor é dada pela contribuição da iluminação ambiente (parcela não associada com fontes de luz) somada à luz emitida e às contribuições C_i

$$C = \text{Amb} + E + \sum A_i + \text{aten} (D_i + S_i)$$