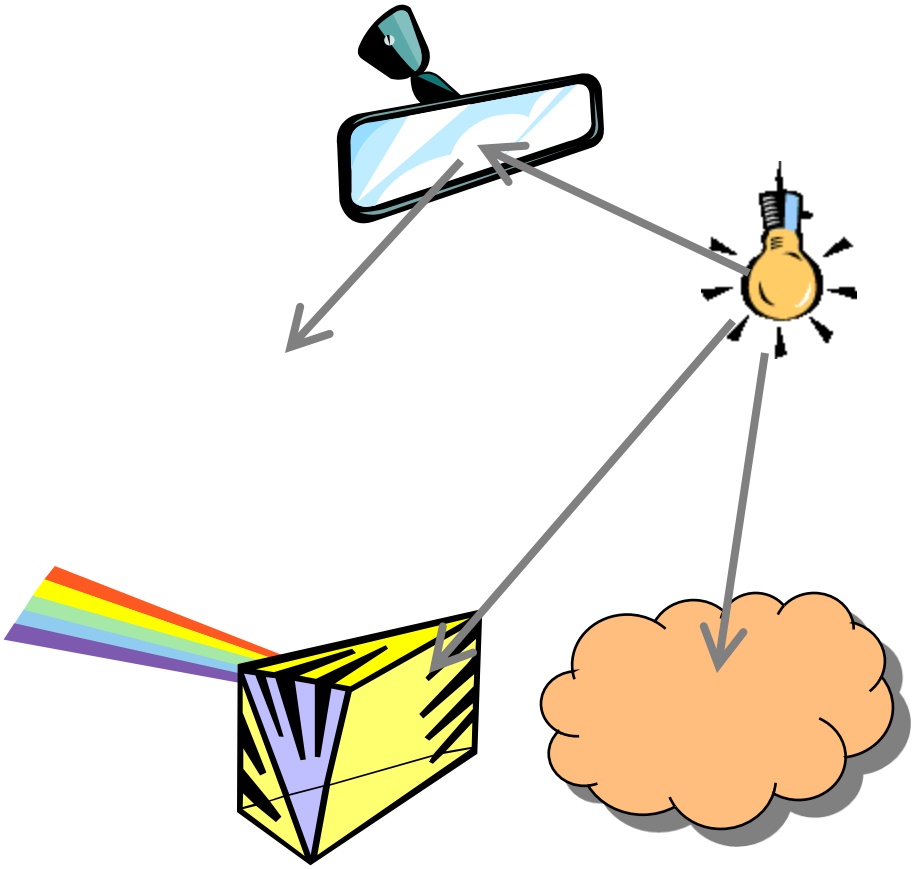
Introdução à Computação Gráfica
Iluminação

Adaptação: João Paulo Pereira
António Costa

Autoria: Claudio Esperança
Paulo Roma Cavalcanti

Iluminação

- Estudo de como a luz interage com os objectos de uma cena
  - Emissão
  - Transmissão
  - Absorção
  - Refracção
  - Reflexão



Iluminação

- Modelos físicos
  - Luz modelada como radiação electromagnética
  - Leva em conta todas as interacções (todos os caminhos da luz)
  - Intratável computacionalmente

Modelos de Iluminação em CG

- Tipicamente, luz é amostrada num número discreto de primárias (cor)
- Modelos locais
  - Apenas caminhos do tipo *fonte luminosa* → *superfície* → *olho* são tratados
  - Simples
  - Ex.: OpenGL
- Modelos globais
  - Muitos caminhos (*ray tracing,* radiosidade)
  - Complexos

Iluminação em OpenGL

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

Iluminação em OpenGL

- Suporte de efeitos atmosféricos como
  - *Fog*
  - Atenuação
- Modelo de iluminação é computado apenas nos vértices das superfícies
  - Cor dos restantes pixels é interpolada linearmente (sombreamento de *Gouraud*)

Fontes de Luz

- Para ligar uma fonte: **`glEnable (source);`**
  - ***`source`*** é uma constante cujo nome é ***`GL_LIGHT`*****`i`**, 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 os lados da superfície que 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***

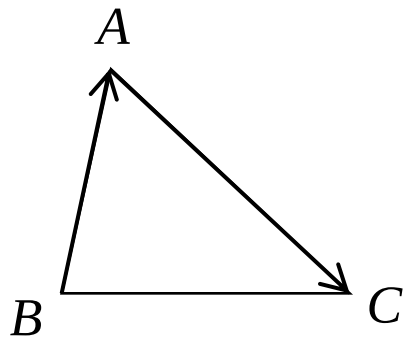
Geometria

- Além das propriedades da luz e do material, a geometria do objecto é também importante
  - A posição dos vértices em relação ao olho e à fonte luminosa contribui para o cálculo dos efeitos atmosféricos
  - A *normal* é fundamental
    - Não é calculada automaticamente
    - Precisa de ser especificada com **`glNormal ()`**

Cálculo do Vector Normal

- Triângulo
  - Dados três vértices,

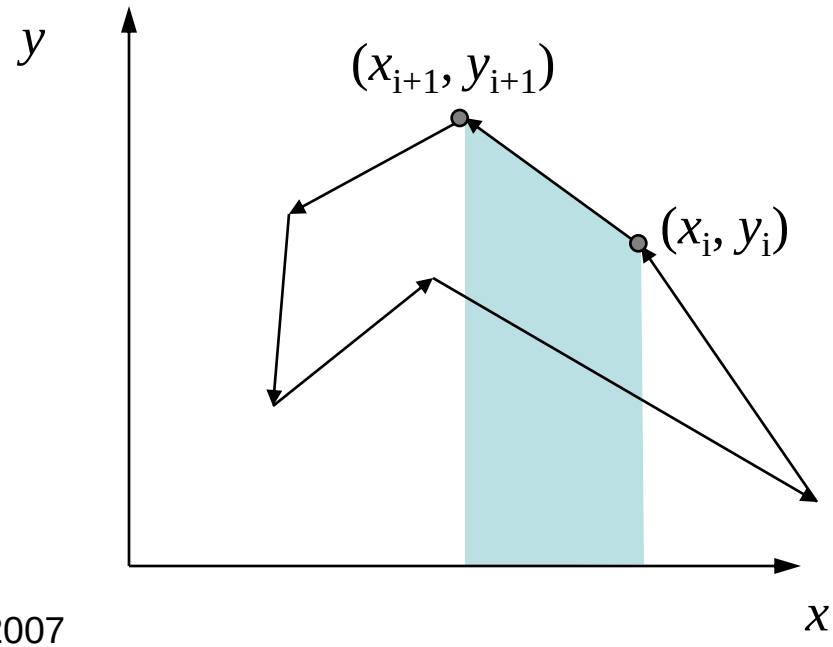
$$\vec{n} = \text{normalizar}((A-B)\times(C-A))$$



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

Cálculo do Vector Normal

- Polígono planar (cont.)
  - Coeficientes *a, b, c* da equação do plano são proporcionais às áreas do polígono projectado nos planos *yz*, *zx* e x*y*



$$AreaXY_i = \frac{1}{2}(y_i + y_{i+1})(x_i - x_{i+1})$$

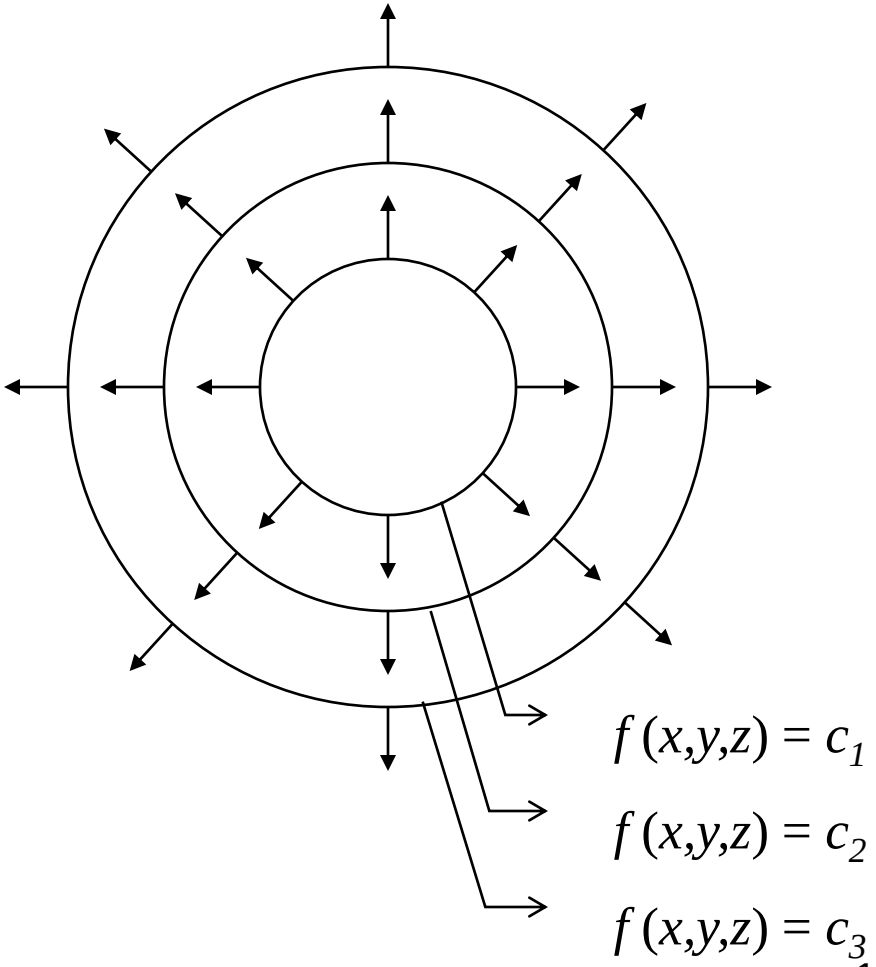
$$c = \sum AreaXY_i$$

Cálculo do Vector Normal de Superfícies Implícitas

- Normal é dada pelo vector gradiente

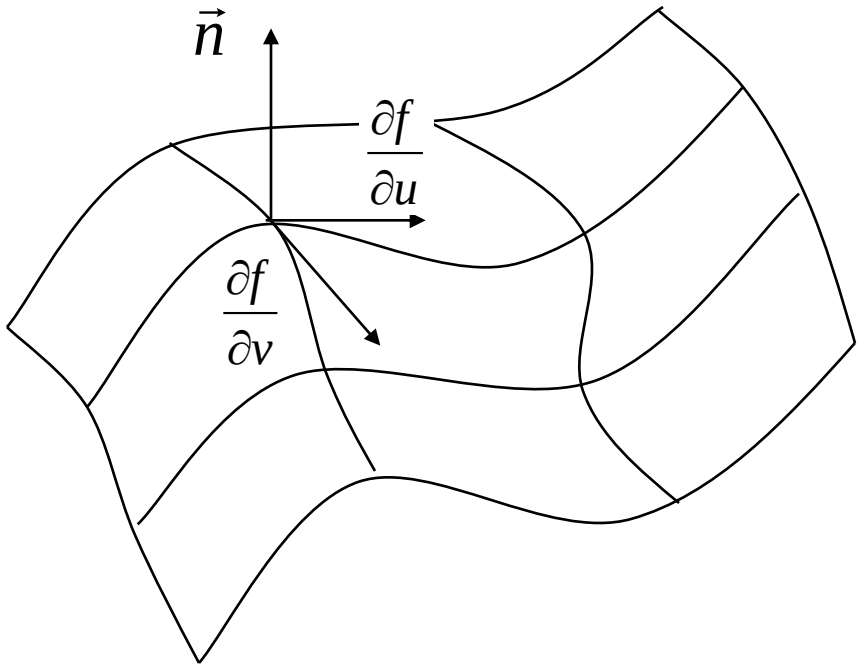
$$f(x, y, z) = 0$$

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



Cálculo do Vector Normal de Superfícies Paramétricas

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



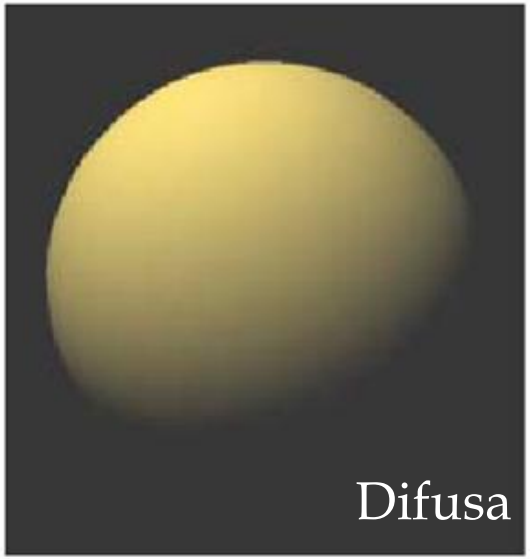
$$P = \begin{pmatrix} f_x(u,v) \\ f_y(u,v) \\ f_z(u,v) \end{pmatrix}$$

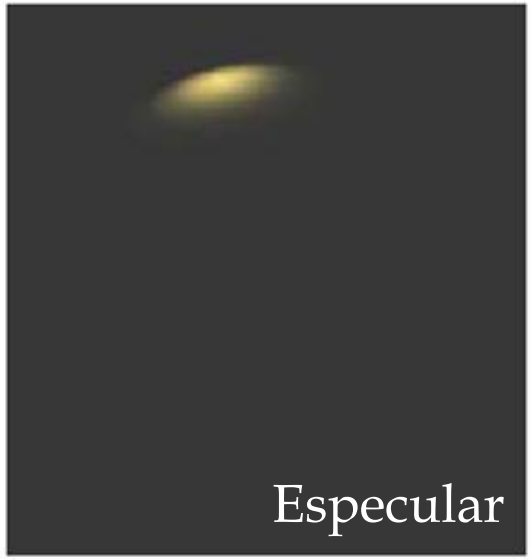
$$\vec{n} = \frac{\partial f}{\partial u} \times \frac{\partial f}{\partial v} = \begin{pmatrix} \partial f_x / \partial u \\ \partial f_y / \partial u \\ \partial f_z / \partial u \end{pmatrix} \times \begin{pmatrix} \partial f_x / \partial v \\ \partial f_y / \partial v \\ \partial f_z / \partial v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

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* (independente da posição do observador)
- Especular: contribuição referente ao comportamento de superfícies polidas

Componentes do Modelo de Phong





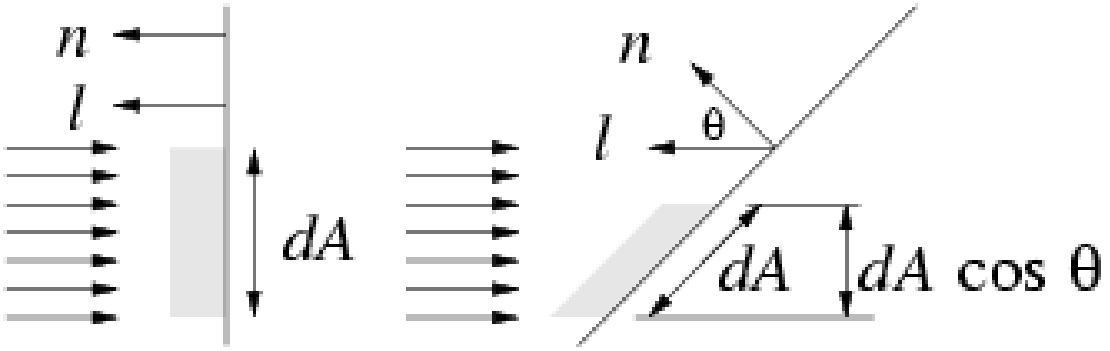
Iluminação Ambiente

- Componente que modela como uma constante o efeito da reflexão de outros objectos 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
  - **`glLightModelfv (GL_LIGHT_MODEL_AMBIENT, params)`**
- Contribuição é dada por

$$A = I_A M_A$$

Iluminação Difusa

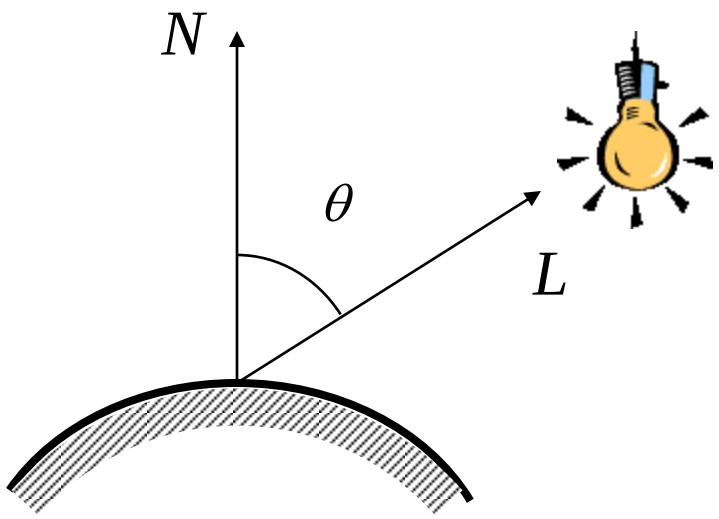
- Iluminação recebida por uma superfície e que é reflectida uniformemente em todas as direcções
- Característica de materiais baços ou foscos
- Esse tipo de reflexão é também designada por reflexão *lambertiana*
- A luminosidade aparente da superfície não depende do observador, mas apenas do cosseno do ângulo de incidência da luz



Iluminação Difusa

- Contribuição é dada por

$$D = I_D M_D \cos\theta = I_D M_D \left(\frac{L \cdot N}{|L||N|}\right)$$



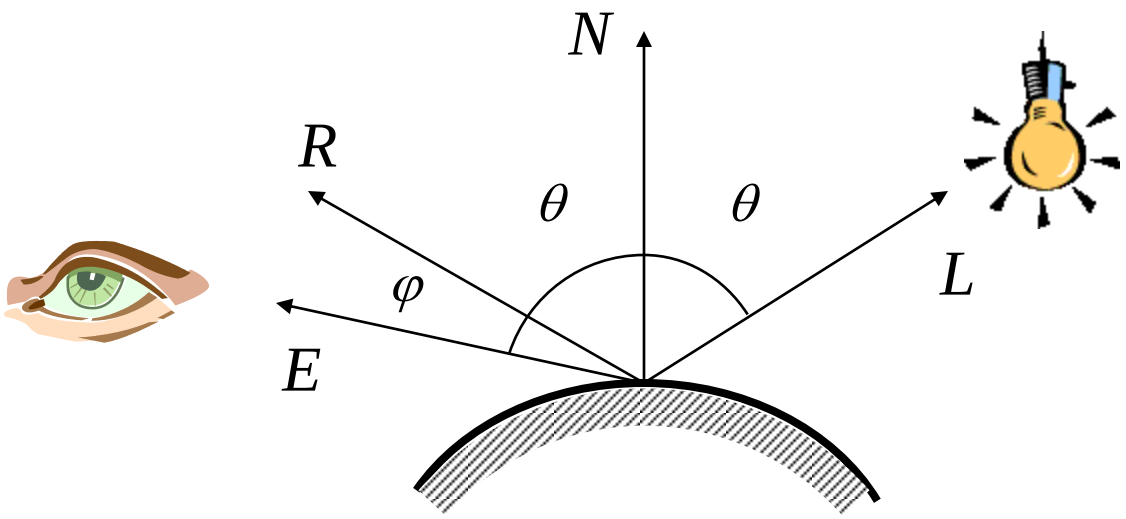
Iluminação Especular

- Simula a reflexão à maneira de um espelho (objectos altamente polidos)
- Depende da posição do observador, objecto e fonte de luz
- Num espelho perfeito, a reflexão dá-se em ângulos iguais
  - Observador só veria a reflexão de uma fonte pontual se estivesse na direção certa
- No modelo de Phong simulam-se reflectores imperfeitos, assumindo que luz é reflectida segundo um *cone* cujo eixo passa pelo observador

Iluminação Especular

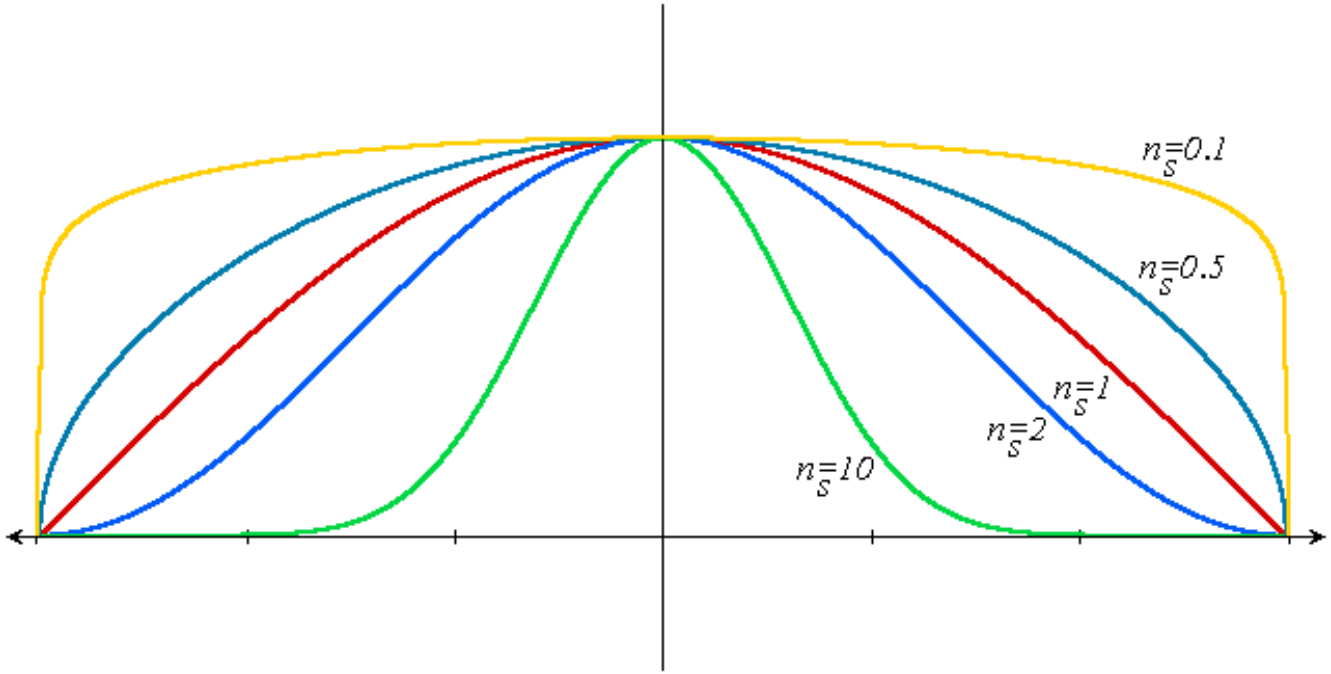
- Contribuição é dada por

$$S = I_S M_S \cos^n \varphi = I_S M_S (R \cdot E)^n$$



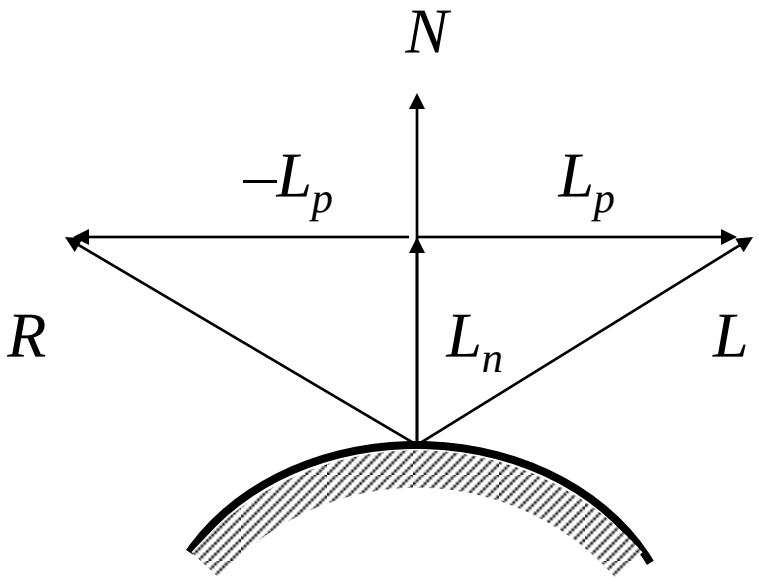
Coeficiente de Especularidade

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



Cálculo do vector da luz reflectida

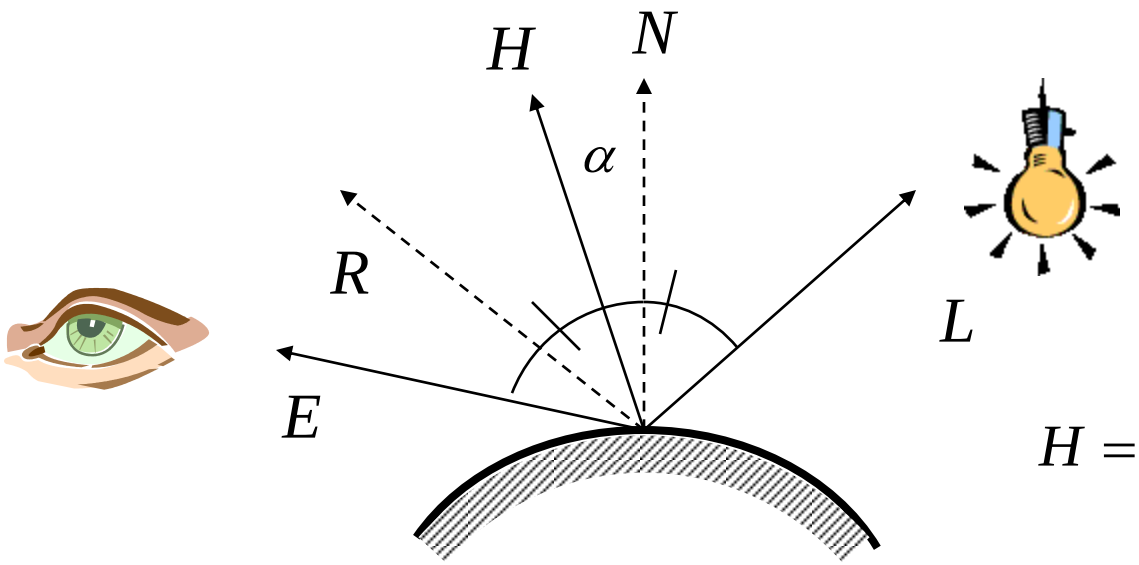
$$R = L_n - L_p = (N \cdot L)N - (L - L_n) = 2(N \cdot L)N - L$$



Componente Especular em OpenGL

- Utiliza o ângulo entre a normal e o vector *halfway*

$$S = I_S M_S \cos^n \alpha = I_S M_S (H \cdot N)^n$$



$$H = \text{normalizar}\left(\frac{E+L}{2}\right)$$
$$= \text{normalizar}(E+L)$$

Atenuação

- Para fontes de luz posicionais (w = 1), é possível definir um factor de atenuação que leva em conta a distância d entre a fonte de luz e o objecto iluminado
- Coeficientes são definidos pela função **`glLight ()`**
- Por omissão não há atenuação (c_0=1, c_1=c_2=0)

$$aten = \frac{1}{c_0 + c_1 d + c_2 d^2}$$

Juntando tudo

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

$$C_i = A_i + aten\left(D_i + S_i\right)$$

- No final, 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

$$C = Amb + E + \sum A_i + aten\left(D_i + S_i\right)$$