Transformações Modelagem

Iluminação (*Shading*)

Transformação Câmera

Recorte

Projeção

Rasterização

Visibilidade

Pipeline de renderização

Adaptação e melhoramentos de uma aula sobre o mesmo assunto (MIT - EECS 6.837 Durand and Cutler)

Transformações Modelagem

Iluminação (*Shading*)

Transformação Câmera

Recorte

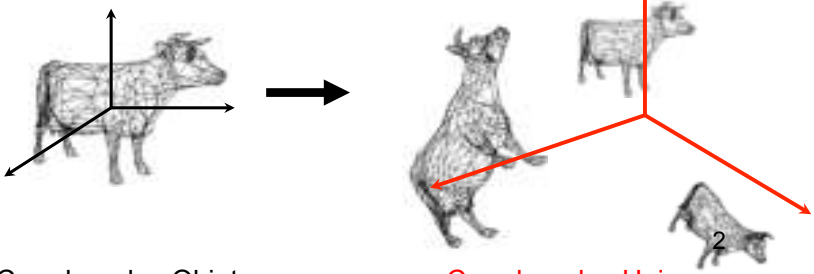
Projeção

Rasterização

Visibilidade

✓Objetos definidos no seu próprio sistema de coordenadas

✓Transformações de modelagem orientam os modelos geométricos num sistema comum de coordenadas (UNIVERSO)



Coordenadas Objeto

Coordenadas Universo

Transformações Modelagem

Iluminação (*Shading*)

Transformação Câmera

Recorte

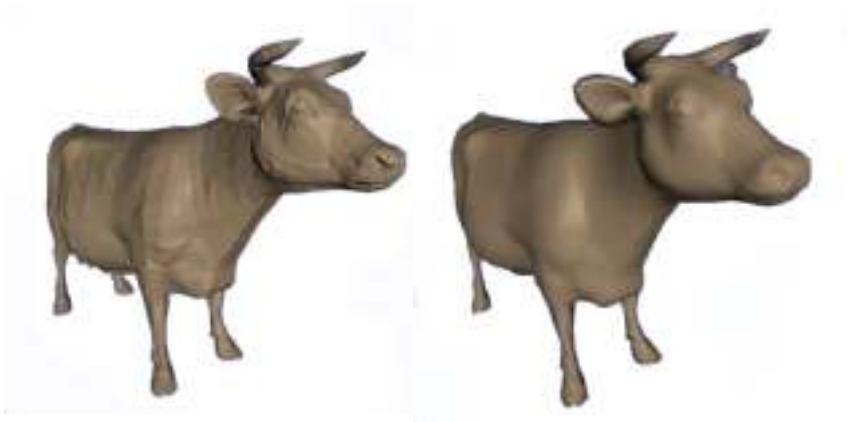
Projeção

Rasterização

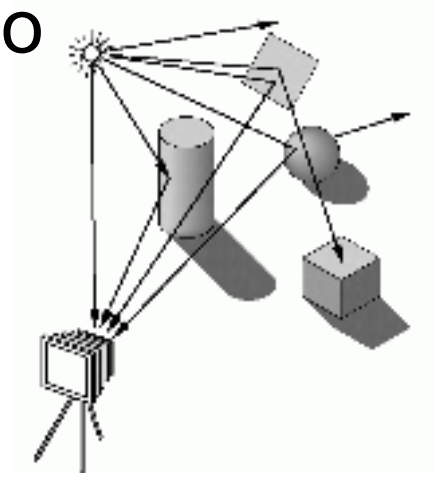
Visibilidade

✓Vértices iluminados de acordo com as propriedades geométricas e de material

✓Modelo de Iluminação Local



Modelo de Iluminação



Objetivo principal

Cálculo de iluminação nos vértices baseando-se na posição, orientação e características das superfícies e fontes de luz que as iluminam

Posteriormente Modelos de Sombreamento

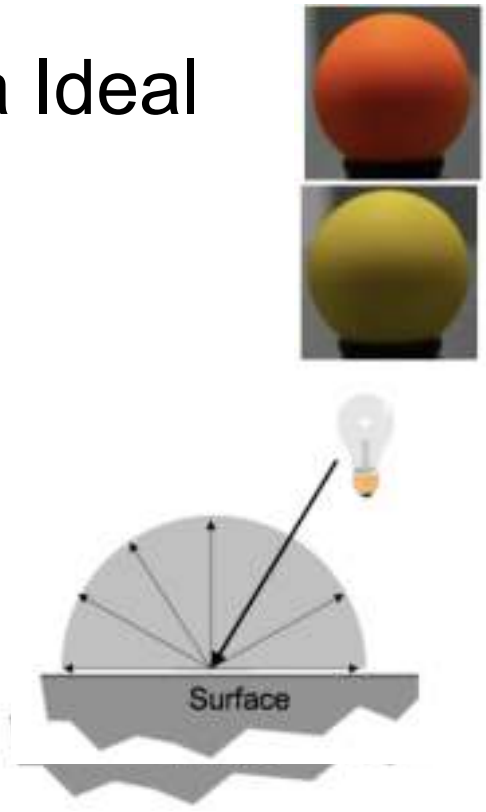
Cálculo de iluminação nos demais pixels que compõe o triângulo a partir das cores nos vértices

Considerações Iniciais

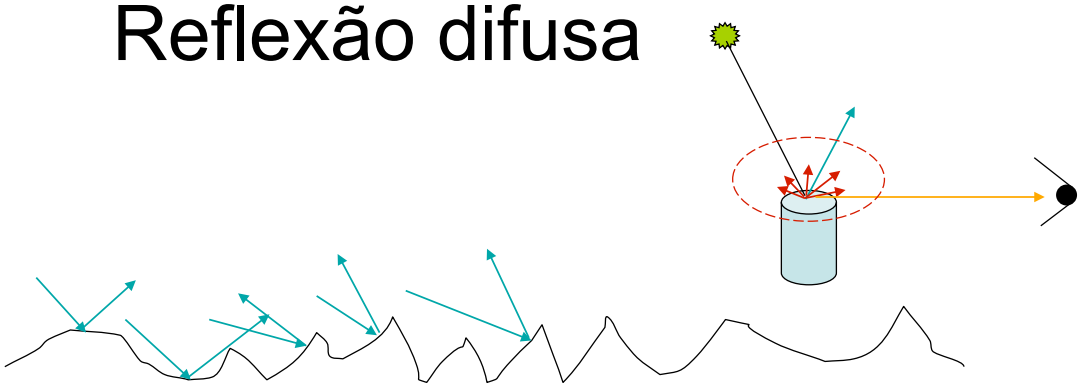
- Um comprimento de onda apenas (monocromático)
- Fontes de luz pontuais (uma direção basicamente)
- Uma fonte de luz apenas

Reflexão Difusa Ideal

- Luz refletida igualmente em todas as direções
- Exemplos: giz, quadro-negro, algumas tintas



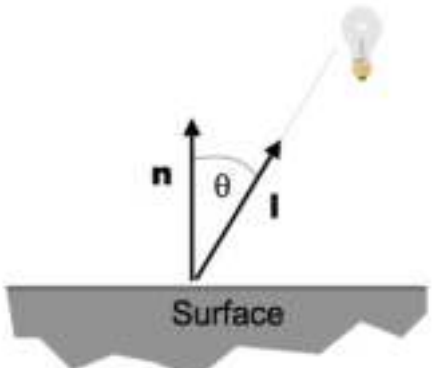
Reflexão difusa

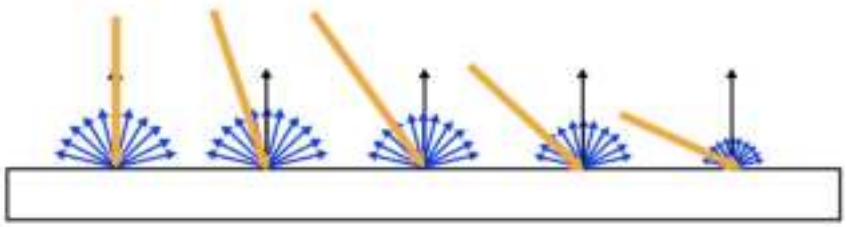


- Reflexão difusa
  - a rugosidade da superfície diminui a possibilidade de reflexão da luz incidente numa direção de reflexão “preferencial”
- Superfícies difusoras perfeitas
  - luz é refletida igualmente em todas as direções
  - intensidade da luz refletida modelada matematicamente
  - Lei de Lambert, superfícies Lambertianas

Lei dos Cossenos de Lambert

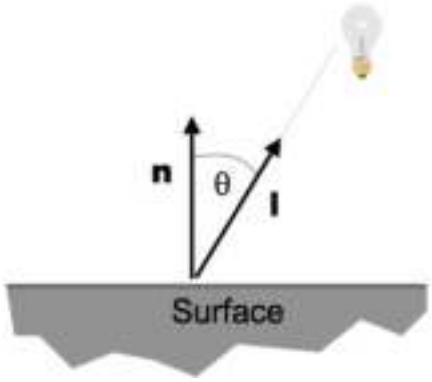
- Intensidade da reflexão difusa é proporcional ao cosseno do ângulo entre a fonte de luz e a normal à superfície



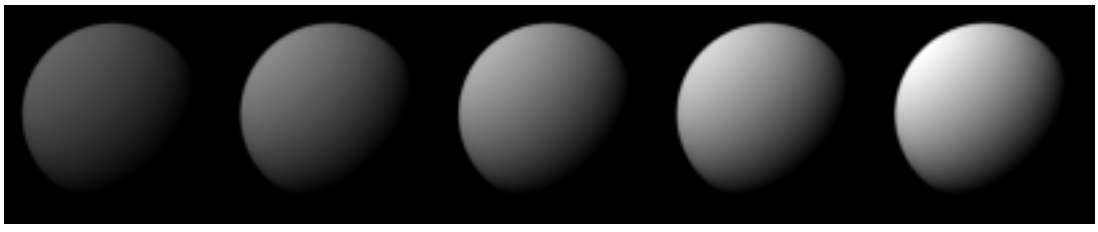


Computacionalmente…

- $I_d = I_p \cdot k_d \cdot \cos\theta$
- $\cos\theta = (n \cdot l)$
  Produto escalar dos 2 vetores NORMALIZADOS
- I_p = intensidade da fonte de luz
- k_d = coeficiente de quão difuso é o objeto [0,1]



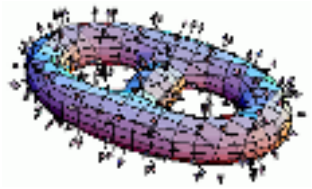
Reflexão Difusa - Exemplos



Kd= 0.4 Kd= 0.55 Kd= 0.7 Kd= 0.85 Kd= 1

menos difuso mais difuso

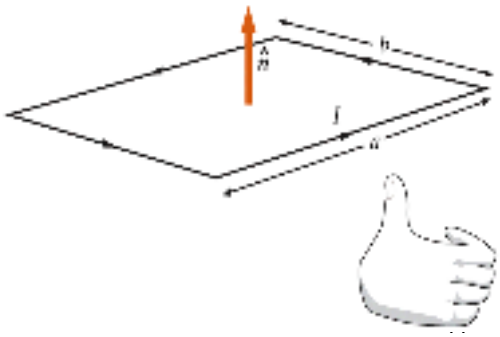
Vetor Normal



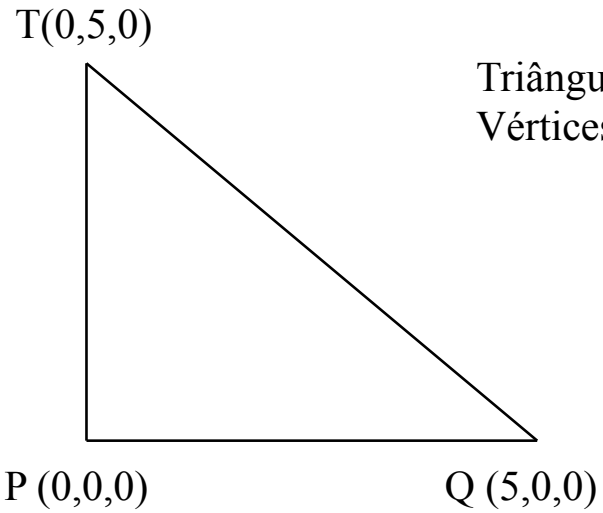
- Fundamental nos cálculos de iluminação
- Em OpenGL

```
glNormal3f( nx, ny, nz);
```

- Consistência na especificação (apontando “para fora” ao percorrer os vértices em sentido anti-horário)



Exemplo



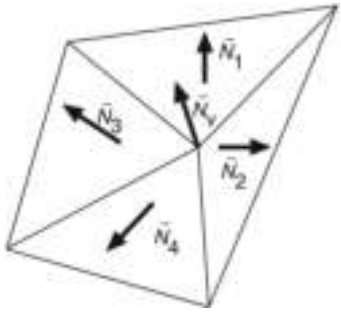
Triângulo A
Vértices PQT

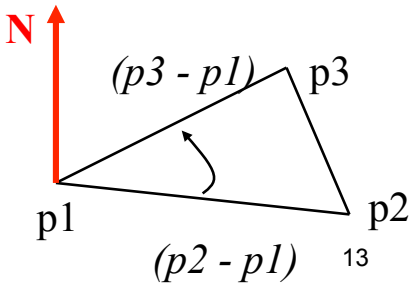
Triângulo B
Vértices PTQ

Qual enumeração é correta?

Aproximando o Vetor Normal

- Cálcula-se para as faces e após, média das faces encontra a normal do vértice
- Para uma face: produto vetorial de vetores a partir das arestas



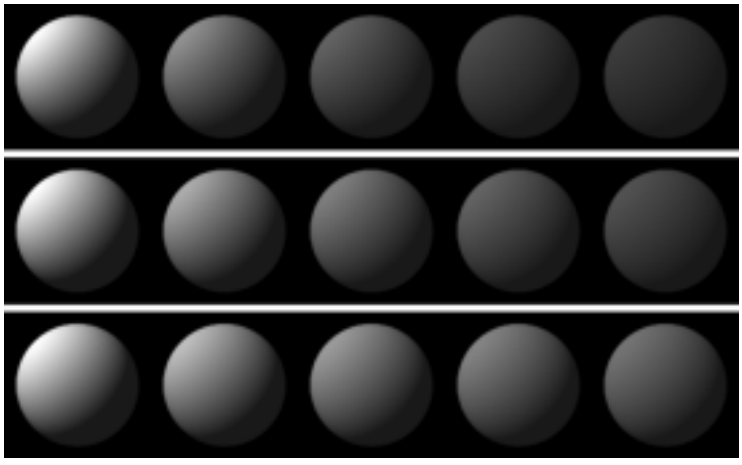


Atenuação da Fonte de Luz

- Fator de atenuação da fonte de luz (f_{att})
- Incorpora dependência em relacão à distância da fonte

$$\text{attenuation factor} = \frac{1}{k_c + k_l d + k_q d^2}$$

d=1 d=1.375 d=1.75 d=2.125 d=2.5



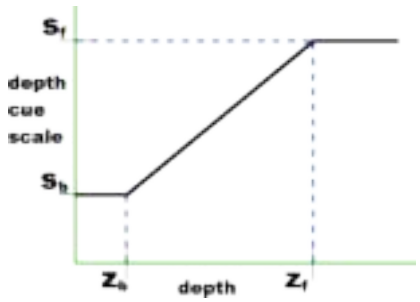
kc=kl=0, kq=1

kc=kl=0.25, kq=0.5

kc=0, kl=1, kq=0

Atenuação Atmosférica

- Modificação da intensidade calculada de acordo com distância
- Modificação determinada por fatores de escala ($\mathbf{s_f}$ e $\mathbf{s_b}$) que indicarão a *combinação* da intensidade com uma cor escolhida

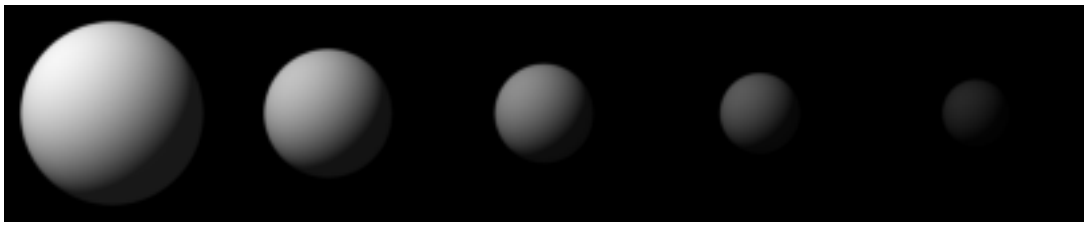


$$C'_{obj} = s_0 C_{obj} + (1-s_0) C_{atms}$$

$$s_0 = s_b + \frac{(z_0 - z_b)(s_f - s_b)}{z_f - z_b}$$

Atenuação Atmosférica

z =1 z =0.77 z =0.55 z =0.32 z =0.09



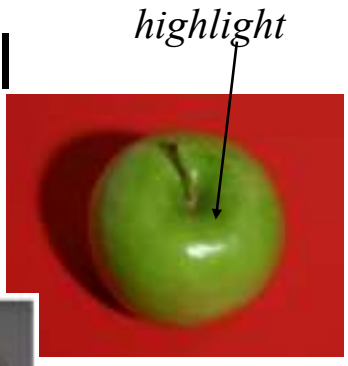
Distância da luz é constante

Ip=1.0;

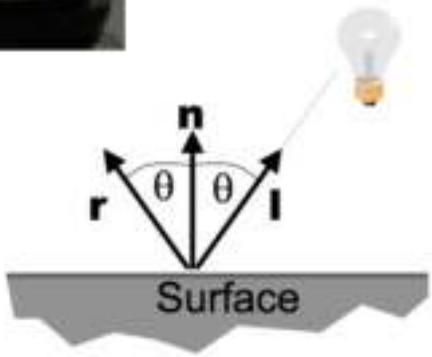
kd=0.9;

zf=1.0; zb=0.0; sf=1.0; sb=0.1

Reflexão Especular Ideal



- Efeito visual que devolve a energia luminosa numa direção preferencial
- Depende da posição do observador
- ângulo de incidência = ângulo de reflexão



Qual a cor do reflexo especular??

Reflexão Especular não-ideal

- Reflexão apresenta distribuição ao redor da direção ideal

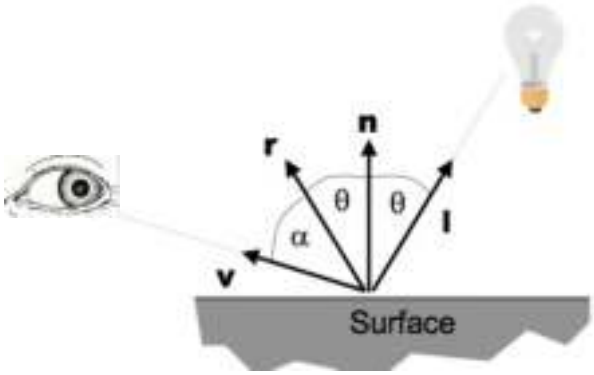
Reflexão Especular não-ideal

- Modelo empírico simples proposto por *Phong* [1975]*
- A reflexão especular varia pouco ao redor da direção especular pura

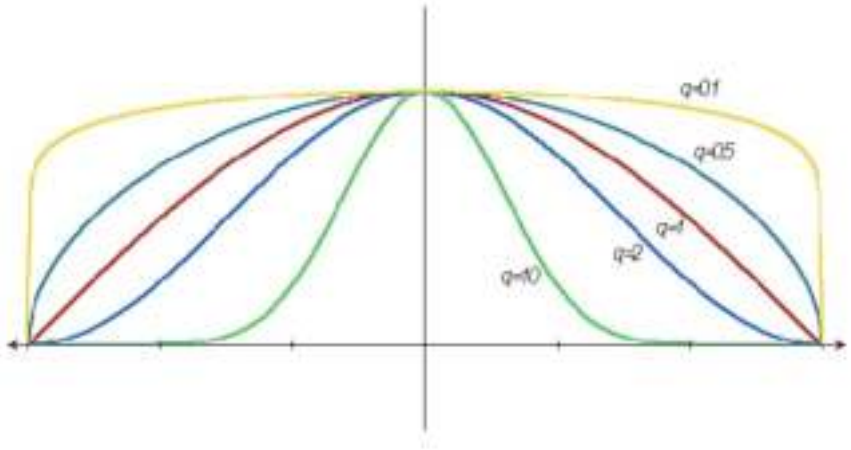
*Phong tem duas contribuições importantes em CG. Esta é a primeira. A outra está relacionada com Modelo de Iluminação para polígonos, veremos adiante.

Computacionalmente…

- $I_s = I_p \cdot k_s \cdot (\cos \alpha)^q$
- $\cos \alpha = (r \cdot v)$ Produto escalar dos vetores NORMALIZADOS
- I_p = intensidade da fonte de luz
- k_s = coeficiente de quão *especular* é o objeto [0,∞…], na prática algumas centenas



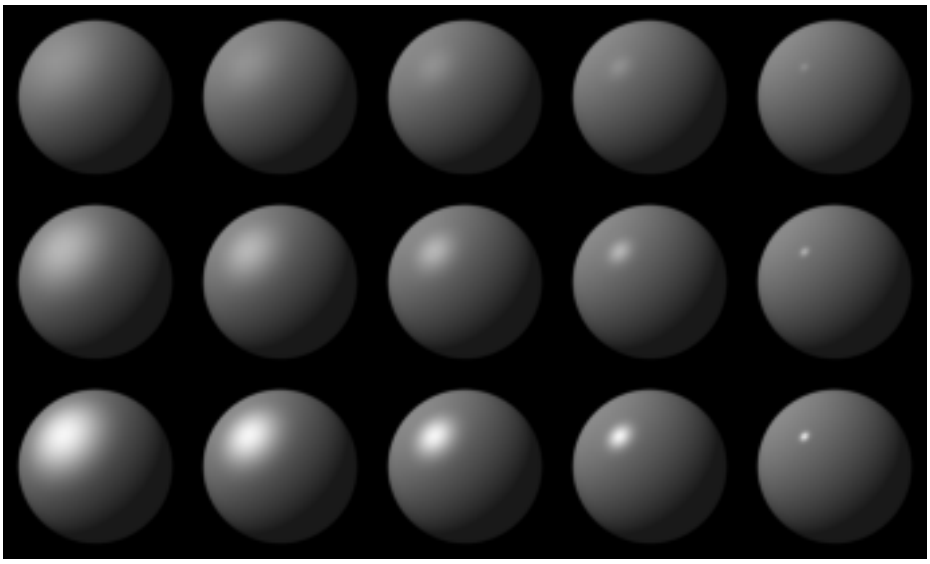
Expoente especularidade Phong



$(\cos \alpha)^q$

Reflexão Especular

Kd=0.45



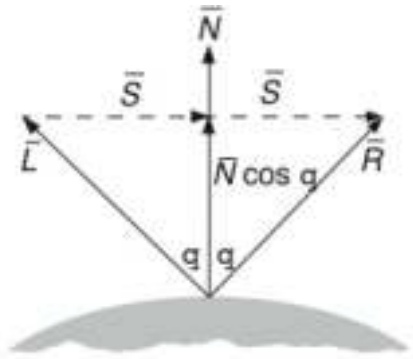
Ks=0.1

Ks=0.25

Ks=0.5

q=3 q=5 q=10 q=27 q=200

Como calcular R?



Todos vetores unitários

$R = N \cos q + S$
$S = N \cos q - L$

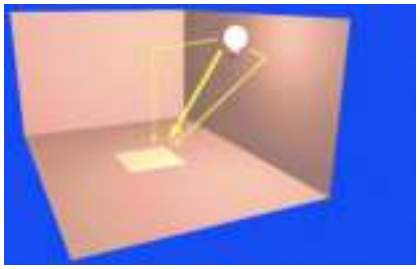
$R = 2 N \cos q - L$

$\cos q = (N \cdot L)$

$R = 2N (N.L) - L$

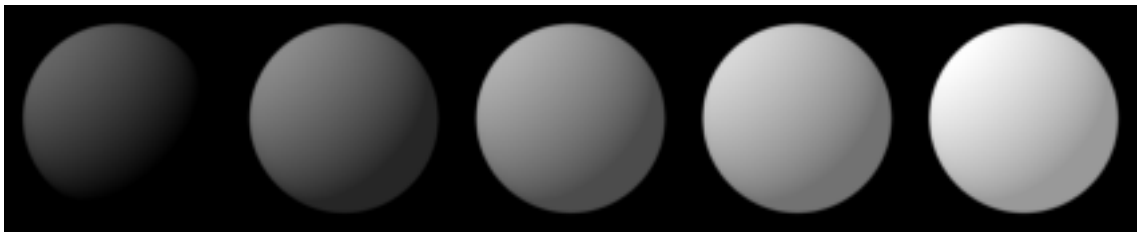
Luz Ambiente

- Fonte de luz sem direção específica. Aproxima (muito mal…) as múltiplas reflexões entre as superfícies presentes no ambiente
- Objetos tem iluminação própria
- Conhecida como luz ambiente
  - $I = I_a.K_a$
    - I_a - intensidade da luz ambiente
    - K_a - coeficiente de reflexão ambiente



Componente Ambiente

Ia=Ip=1.0, Kd=0.4



Ka=0.0 Ka=0.015 Ka=0.3 Ka=0.45 Ka=0.6

Luz Ambiente

Aqui não tem componente difusa…

Múltiplas Fontes de Luz

- Existindo *m* fontes de luz, basta somarmos os termos de cada fonte de luz
- Qual um problema em potencial desta abordagem??

Qual a solução de OpenGL?
"*…the color values are clamped to the range [0,1].*" (Red book, p. 205)

Colocando tudo junto…

Considerando vetores normalizados e **m** fontes de luz:

$$I = I_a k_a + \Sigma \{ I_{pm} [k_d (N.L) + k_s (R.V)^q] \}$$

Colocando tudo junto…

agora com cores

$$I_\lambda = I_{a\lambda} k_{a\lambda} + \Sigma \{ I_{pm\lambda} [k_{d\lambda} (N.L) + k_{s\lambda} (R.V)^q] \}$$

λ = (R,G,B)

Calculamos 3 vezes a equação acima, uma vez para cada canal

E OpenGL?

Materiais

Parâmetro	Default	Significado
GL_AMBIENT	(0.2, 0.2, 0.2, 1.0)	Cor ambiente do material
GL_DIFFUSE	(0.8, 0.8, 0.8, 1.0)	Cor difusa do material
GL_AMBIENT_AND_DIFFUSE		Cor ambiente e difusa do material
GL_SPECULAR	(0.0, 0.0, 0.0, 1.0)	Cor especular do material
GL_SHININESS	0.0	Expoente Phong
GL_EMISSION	(0.0, 0.0, 0.0, 1.0)	Cor emissão do material

Materiais

```
GLfloat material_diffuse = { 1, 0, 0, 1 };
GLfloat material_specular = { 1, 1, 1, 1 };
GLfloat material_shininess = { 100 };

glMaterialfv(GL_FRONT, GL_DIFFUSE, material_diffuse);
glMaterialfv(GL_FRONT, GL_SPECULAR,material_specular);
glMaterialfv(GL_FRONT, GL_SHININESS, material_shininess);
```

Pode ser um de `GL_FRONT, GL_BACK, GL_FRONT_AND_BACK`

Materiais

```
glColorMaterial(GLenum face, GLenum mode);
```

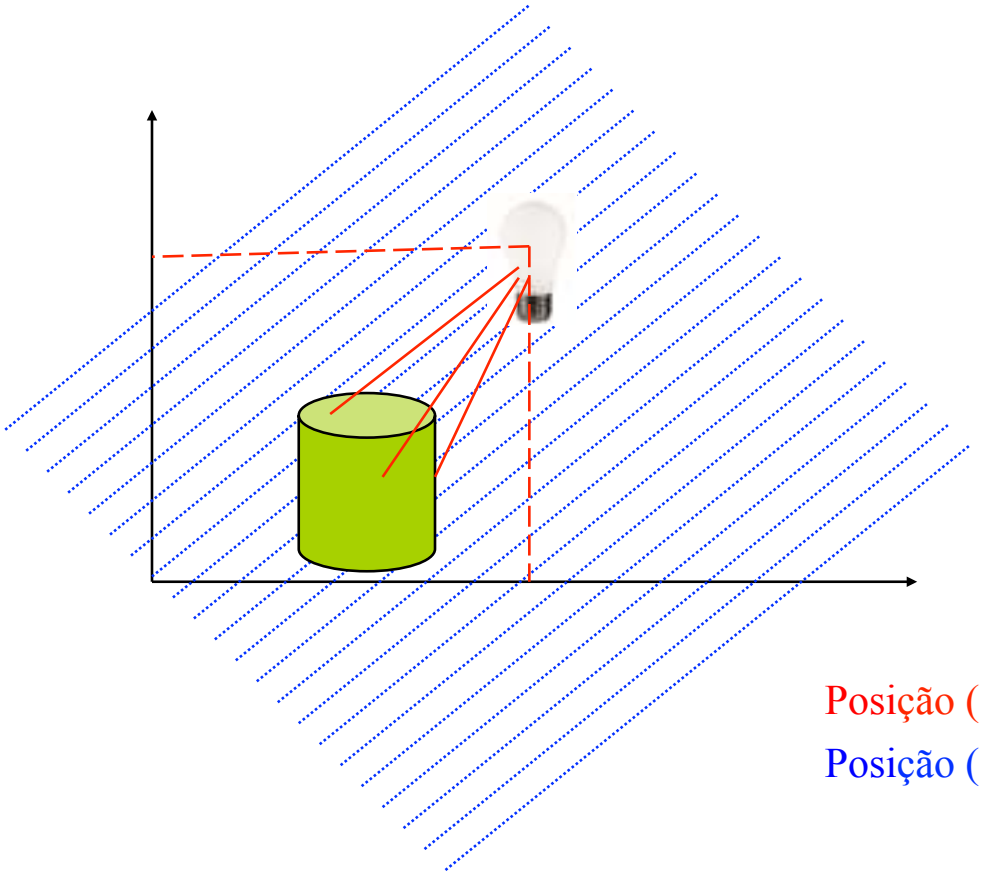
Utilizado para forçar um parâmetro de material ser o mesmo que a cor atual. Necessita ser habilitado com glEnable. Altera vários objetos ao mesmo tempo.

```
glColorMaterial(GL_FRONT, GL_DIFFUSE);
glEnable(GL_COLOR_MATERIAL);
glColor3f(0.2, 0.5, 0.8);
/* draw some objects here */
glColor3f(0.9, 0.0, 0.2);
/* draw other objects here */
glDisable(GL_COLOR_MATERIAL);
```

E OpenGL?
Fontes de Luz

Parâmetro	Default	Significado
GL_AMBIENT	(0.0, 0.0, 0.0, 1.0)	Cor ambiente da luz
GL_DIFFUSE	(1.0, 1.0, 1.0, 1.0)	Cor difusa da luz
GL_SPECULAR	(1.0, 1.0, 1.0, 1.0)	Cor especular da luz
GL_POSITION	(0.0, 0.0, 1.0, 0.0)	Posição da fonte

Indica se a fonte de luz é direcional ou não
0 indica raios de luz paralelos
1 indica uma posição no espaço



Posição (1,1,0,1)

Posição (1,1,0,0)

Número de Fontes

- *The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form* `GL_LIGHTi` *where*

```
0 < i < GL_MAX_LIGHTS
```


E OpenGL?
Fontes de Luz

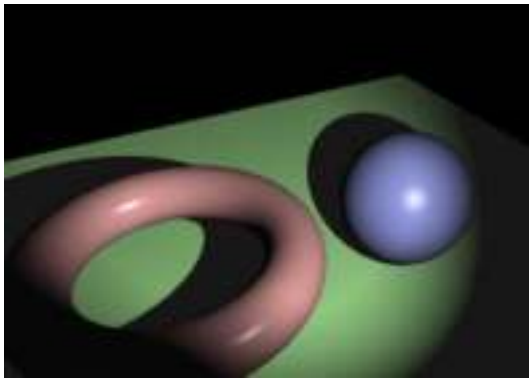
```
GLfloat light_ambient[] = { 0.0, 0.0, 0.0, 1.0 };
GLfloat light_diffuse[] = { 1.0, 1.0, 1.0, 1.0 };
GLfloat light_specular[] = { 1.0, 1.0, 1.0, 1.0 };
GLfloat light_position[] = { 1.0, 1.0, 1.0, 0.0 };

glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
glLightfv(GL_LIGHT0, GL_POSITION, light_position);

glEnable(GL_LIGHTING); <---
glEnable(GL_LIGHT0);
```

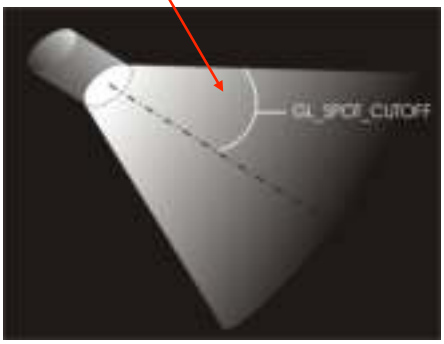
Outros Tipos de Fontes

- Spot
  - GL_SPOT_DIRECTION
  - GL_SPOT_EXPONENT
  - GL_SPOT_CUTOFF

Specifies the intensity distribution of the light. Higher spot exponents result in a more focused light source, regardless of the spot cutoff angle

```
glLightf(GL_LIGHT0, GL_SPOT_CUTOFF, 45.0);
```





Fator de Atenuação

```
glLightf(GL_LIGHT0, GL_CONSTANT_ATTENUATION, 2.0);
glLightf(GL_LIGHT0, GL_LINEAR_ATTENUATION, 1.0);
glLightf(GL_LIGHT0, GL_QUADRATIC_ATTENUATION, 0.5);
```

$$\text{attenuation factor} = \frac{1}{k_c + k_l d + k_q d^2}$$

Limitações

- Número de fontes de luz
- Bloqueio de luz não existe (alguém falou em sombras??)