

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)

1

Transformações  
Modelagem

Iluminação  
(*Shading*)

Transformação  
Câmera

Recorte

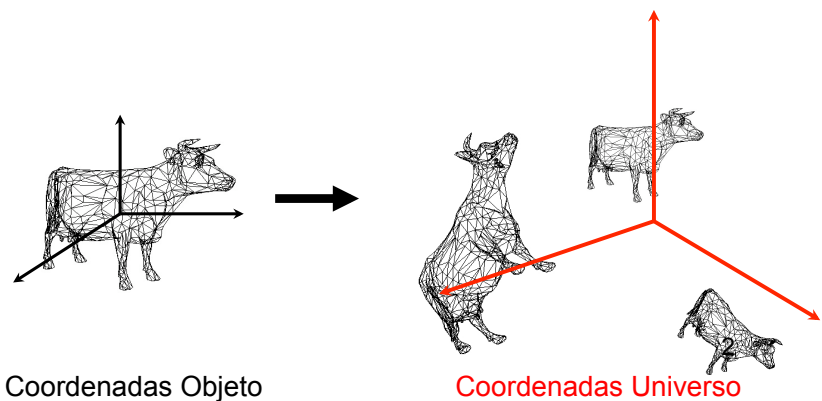
Projeção

Rasterização

Visibilidade

✓ Objetos definidos no seu próprio sistema de coordenadas

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



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

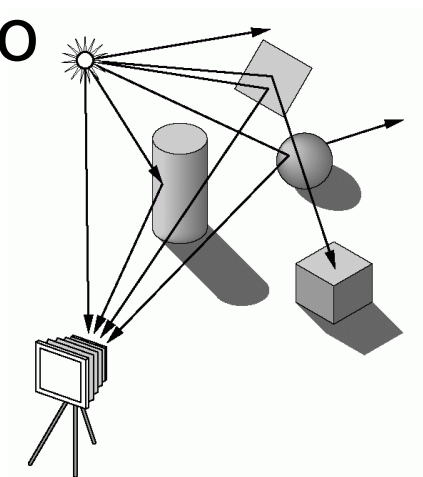


3

## Modelo de Iluminação

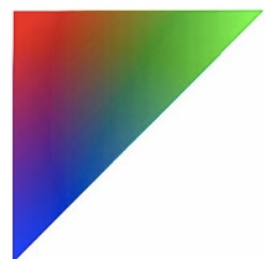
Objetivo principal

*Cálculo de iluminação nos vértices baseado-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*



4

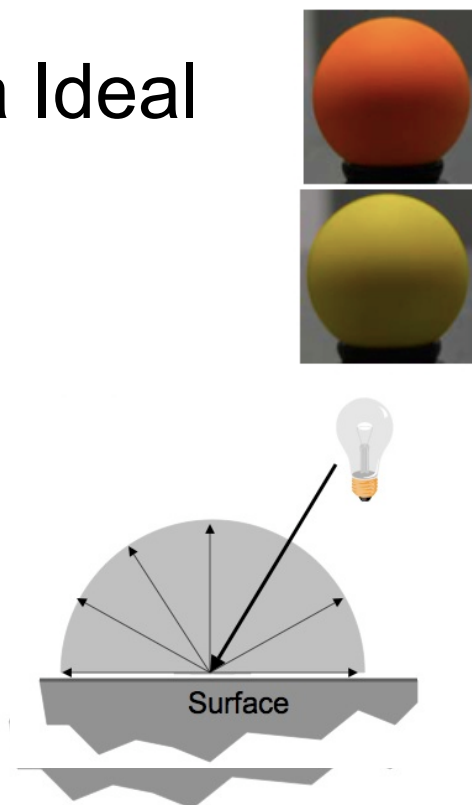
# Considerações Iniciais

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

5

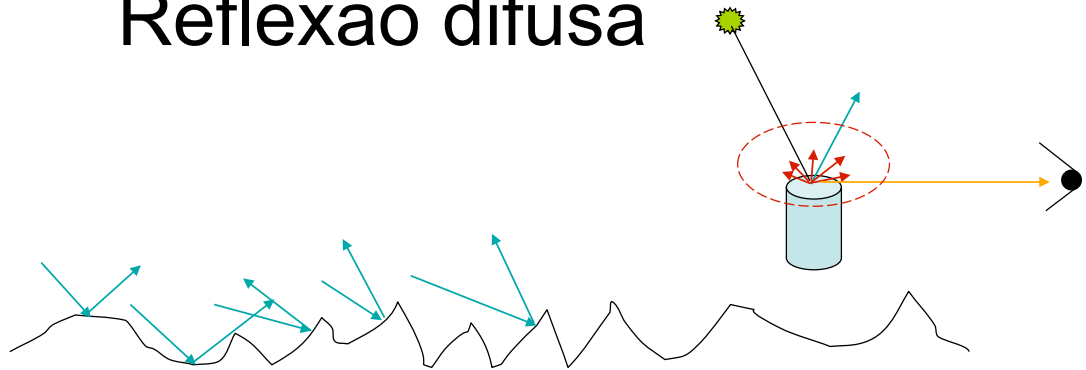
## Reflexão Difusa Ideal

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



6

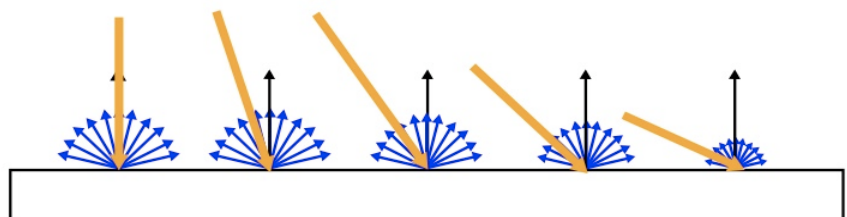
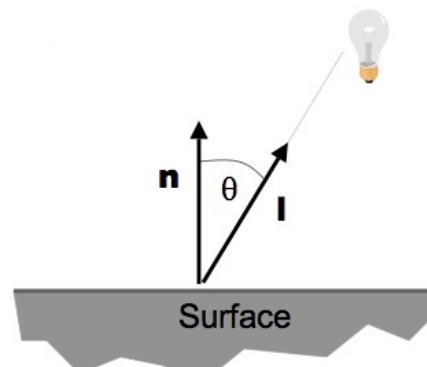
# Reflexão difusa



- 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

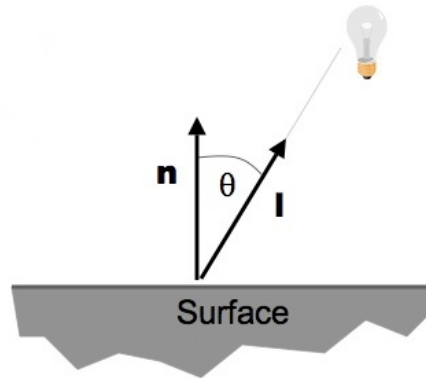
## Lei dos Cossenos de Lambert

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



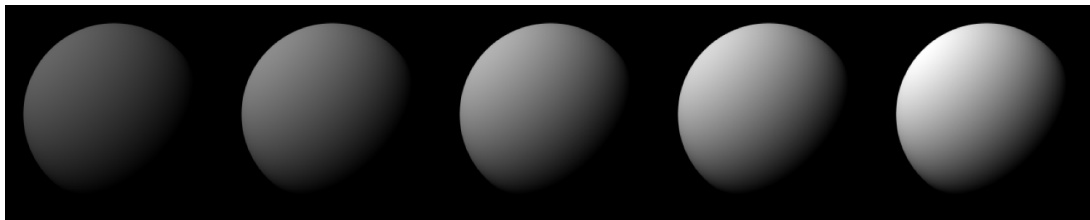
## Computacionalmente...

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



9

## Reflexão Difusa - Exemplos



$K_d = 0.4$

$K_d = 0.55$

$K_d = 0.7$

$K_d = 0.85$

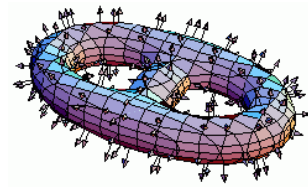
$K_d = 1$

menos difuso

mais difuso

10

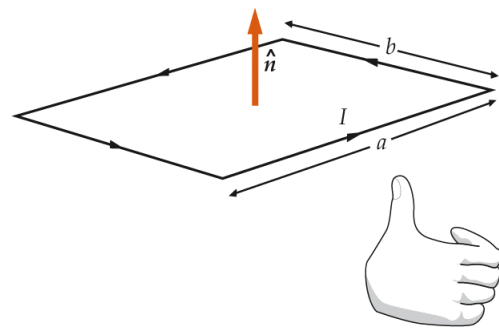
# Vetor Normal



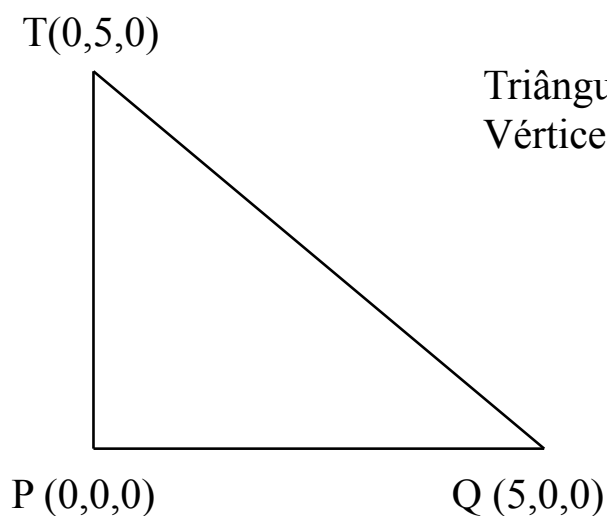
- Fundamental nos cálculos de iluminação
- Em OpenGL

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

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



## Exemplo



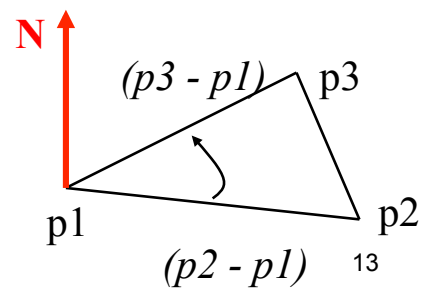
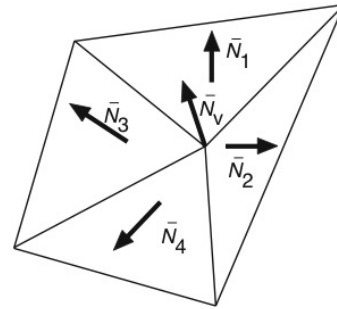
Triângulo A  
Vértices PQT

Triângulo B  
Vértices PTQ

Qual enumeração é  
correta?

# Aproximando o Vetor Normal

- Calcula-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 relação 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

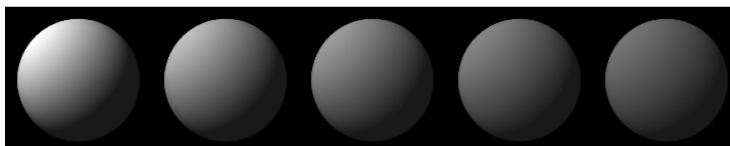
d=2.5



$k_c=k_l=0, k_q=1$



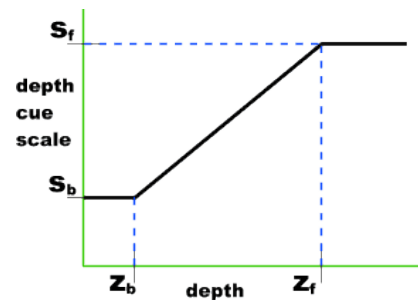
$k_c=k_l=0.25, k_q=0.5$



$k_c=0, k_l=1, k_q=0$

# Atenuação Atmosférica

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



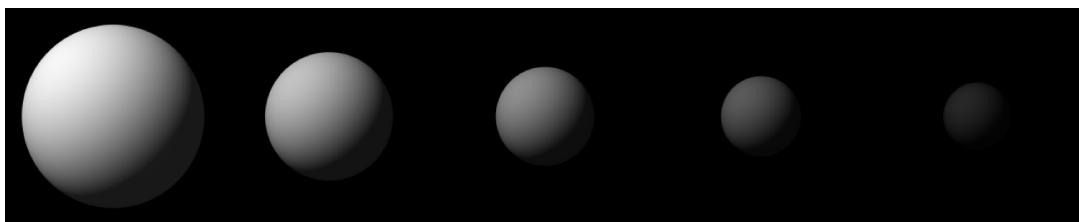
$$C'_{obj} = s_0 C_{obj} + (1-s_0) C_{atms}$$

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

15

# Atenuação Atmosférica

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



Distância da luz é constante

$I_p=1.0$ ;

$k_d=0.9$ ;

$z_f=1.0$ ;  $z_b=0.0$ ;  $s_f=1.0$ ;  $s_b=0.1$

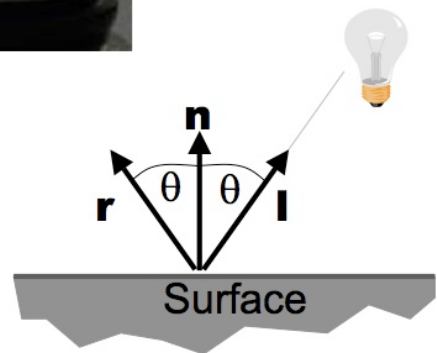
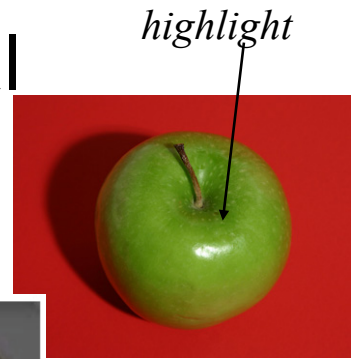
16



# 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??



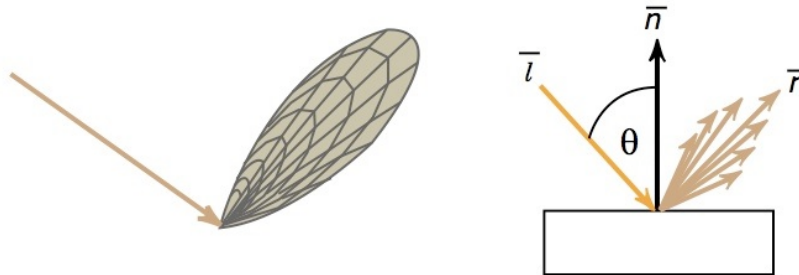
## Reflexão Especular não-ideal



- 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

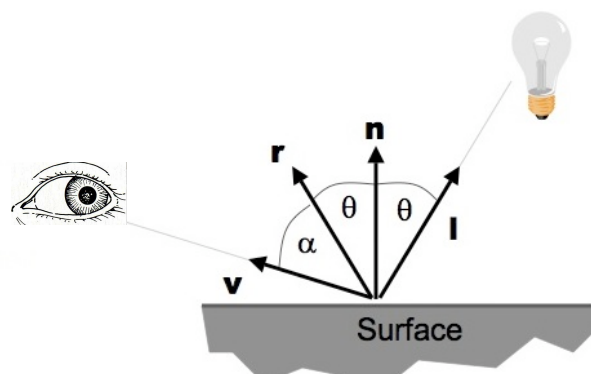


\*Phong tem duas contribuições importantes em CG. Esta é a primeira. A outra está relacionada com Modelo de Iluminação para polígonos, veremos adiante.

19

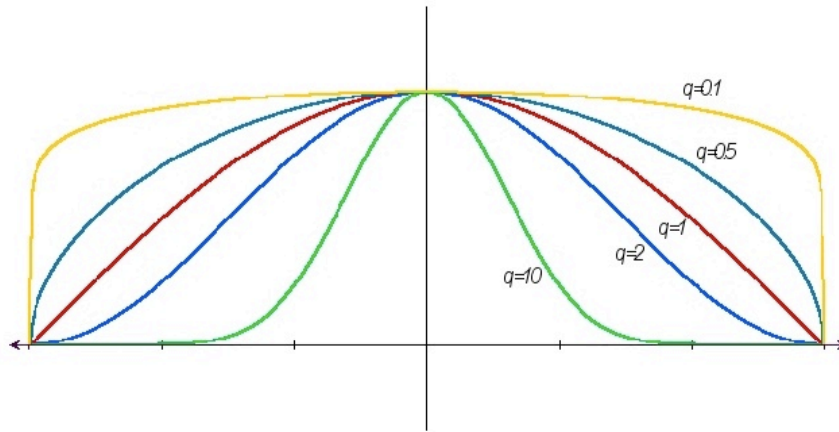
## Computacionalmente...

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



20

# Expoente especularidade Phong

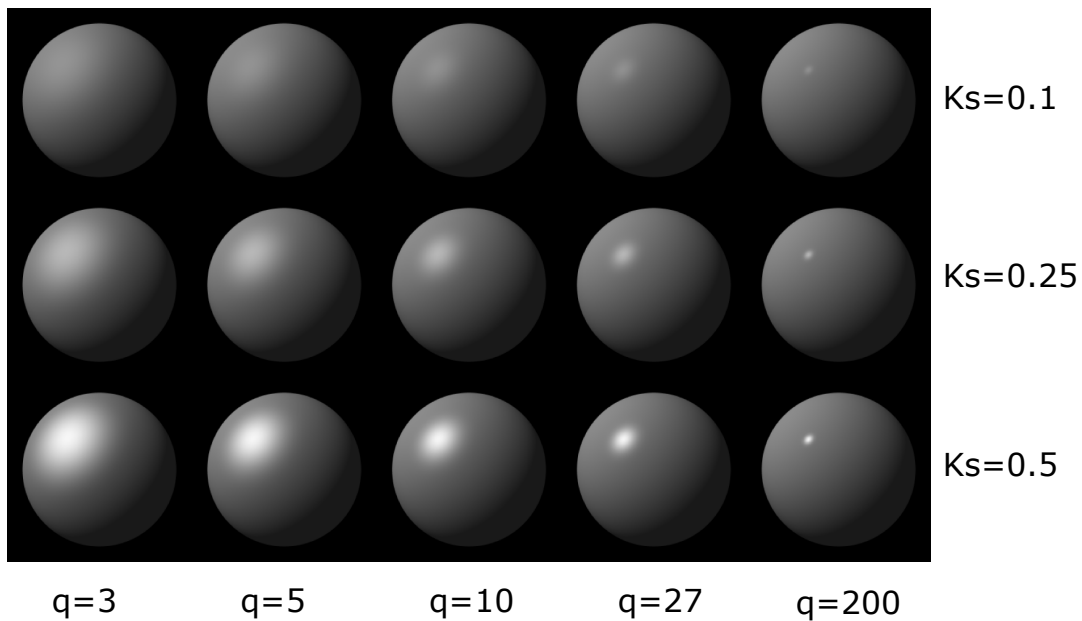


$$(\cos \alpha)^q$$

21

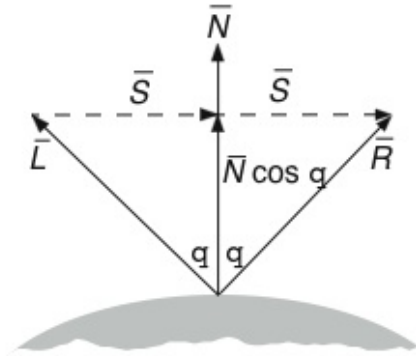
## Reflexão Especular

$K_d=0.45$



22

# Como calcular R?



Todos vetores unitários

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

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

$$R = 2 N \cos q - L$$

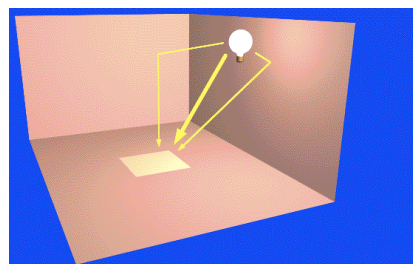
$$\cos q = (N \cdot L)$$

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

23

## Luz Ambiente

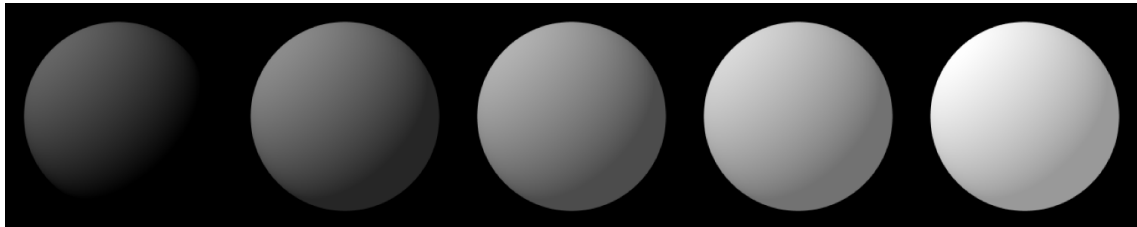
- Fonte de luz sem direção específica. Aproxima (muito mal...) as múltiplas reflexões entre as superfícies presentes no ambiente
- Objetos tem iluminação própria
- Conhecida como luz ambiente
  - $I = I_a \cdot K_a$ 
    - $I_a$  - intensidade da luz ambiente
    - $K_a$  - coeficiente de reflexão ambiente



24

# Componente Ambiente

$$I_a = I_p = 1.0, K_d = 0.4$$



$K_a = 0.0$

$K_a = 0.015$

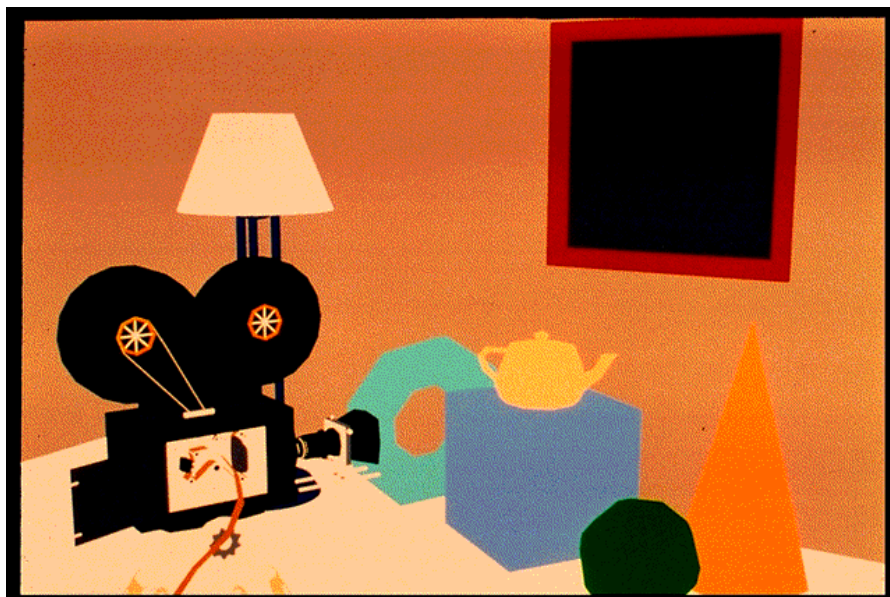
$K_a = 0.3$

$K_a = 0.45$

$K_a = 0.6$

25

# Luz Ambiente



Aqui não tem componente difusa...

26

# 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)*

27

## Colocando tudo junto...

Considerando vetores normalizados e  $m$  fontes de luz:

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

28

# Colocando tudo junto...

## agora com cores

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

$\lambda = (R, G, B)$

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

29

## 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

30

# 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

31

# 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);
```

32



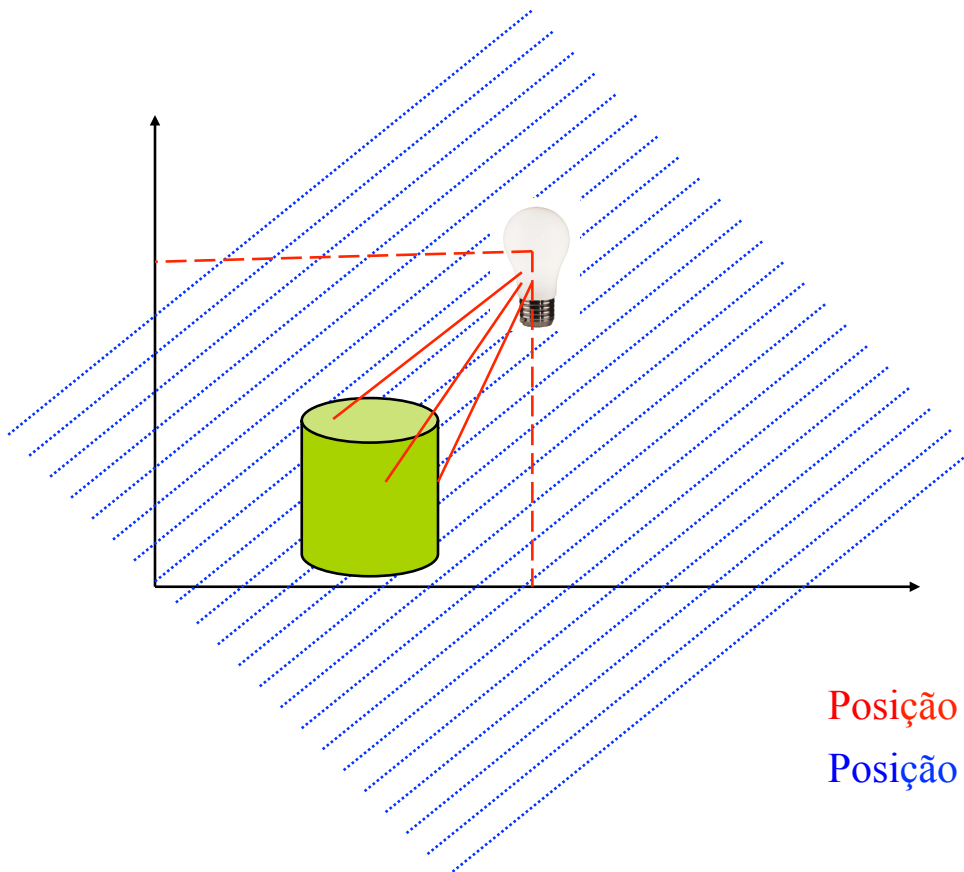
# E OpenGL?

## 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

33



34

# 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*

$$0 < i < \text{GL\_MAX\_LIGHTS}$$

35

## 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);
```

36

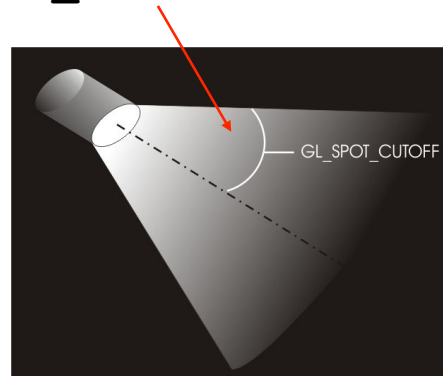
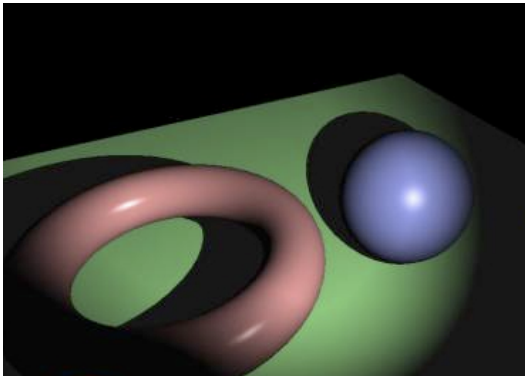
# Outros Tipos de Fontes

- 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??)