



Iluminação

Modelos de Iluminação em Tempo Real;
Iluminação e Materiais em OpenGL



Iluminação

- Tópicos:
 - Fundamentos de iluminação
 - Aplicação em OpenGL
 - Modelos de Shading
 - Materiais em OpenGL
 - Iluminação em OpenGL



Iluminação - Fundamentos

- Em CG a iluminação simula o modo como os objectos reflectem a luz em tempo real ... ou não.
- A versão em tempo real é uma aproximação empírica da iluminação real, por vezes sem profundas bases teóricas que a sustente, no entanto com resultados práticos bastante aceitáveis para alguns fins.
- Esta aproximação deve-se a dois factores:
 - As equações da iluminação real não são totalmente conhecidas
 - Mesmo os modelos simplificados da realidade são extremamente complexos



Iluminação - Fundamentos

- Nos modelos mais simples, assume-se que o cálculo da intensidade reflectida por um objecto depende somente da relação entre a superfície a ser iluminada e a fonte de luz.
- Os restantes objectos não participam



Iluminação - Fundamentos

- Consideremos uma fonte de luz com uma posição determinada que emite luz em todas as direcções, por ex: lâmpada, Sol.
- A fonte de luz emite raios de luz uniformemente distribuídos.
- Podemos ter em conta dois factores:
 - orientação do objecto em relação à fonte de luz
 - distância à fonte de luz



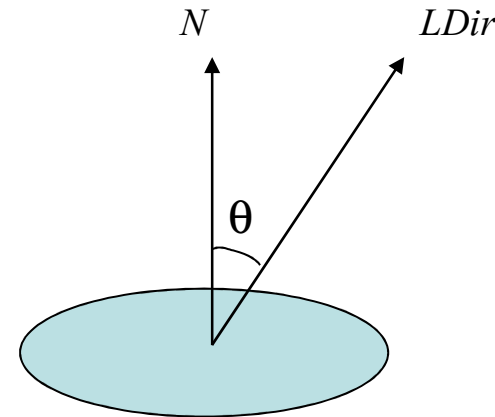
Iluminação - Fundamentos

- Reflexão difusa (Lambert)
 - A intensidade reflectida por um ponto é proporcional ao ângulo entre a direcção da fonte de luz e a normal da superfície do objecto nesse ponto.

intensidade da luz

$$I_d = L_d * K_d * \cos(\theta)$$

coeficiente de reflexão difusa





Iluminação - Fundamentos

- Atenuação baseada na distância

$$I = f_{att} * I_d$$

- A escolha *correcta* seria:

$$f_{att} = \frac{1}{d^2} \longleftarrow \text{distância à fonte de luz}$$

- Ou seja, a intensidade é inversamente proporcional ao quadrado da distância entre o objecto e a fonte de luz



Iluminação - Fundamentos

- A escolha *correcta* no entanto não produz os efeitos desejados.
 - Luz próxima => grandes variações
 - Luz distante => intensidade demasiado pequena
- A realidade não se resume a uma fonte de luz sem interacção entre os objectos.
- Numa situação real a luz reflectida por um objecto tem influência nos outros objectos.



Iluminação - Fundamentos

- Uma solução de compromisso é:

$$f_{att} = \min\left(\frac{1}{c_1 + c_2d + c_3d^2}, 1.0\right)$$

c_1, c_2, c_3 : constantes associadas à fonte de luz.

c_1 : constante que evita que o denominador fique muito pequeno quando a luz está muito perto.

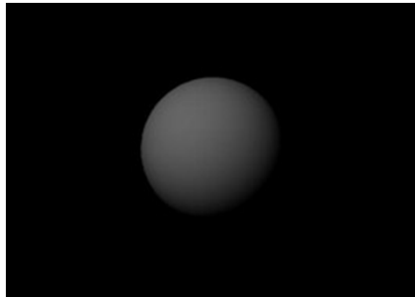
O cálculo do mínimo obriga a diminuir ou manter a intensidade, i.e.

$$f_{att} \leq 1.0$$



Iluminação - Fundamentos

- Só os pontos que estão virados para a luz é que recebem uma contribuição da luz.



- Os pontos que não recebem luz directa ficam completamente escuros.
- Adequado para um modelo do sistema solar mas ...



Iluminação - Fundamentos

- A forma mais simples para colmatar esta lacuna consiste em utilizar iluminação ambiente
- Todos os objectos recebem, para além da componente difusa, uma componente que se denomina por "ambiente".
- A componente ambiente afecta todos os pontos de um objecto de igual forma.
- Para cada objecto é definida uma constante, que indica a quantidade de luz reflectida



Iluminação - Fundamentos

- Iluminação Ambiente
- Esta forma de iluminação simula de uma forma básica as interacções entre os objectos e a luz, iluminando todos os objectos por igual

$$I_a = K_a * L_a$$

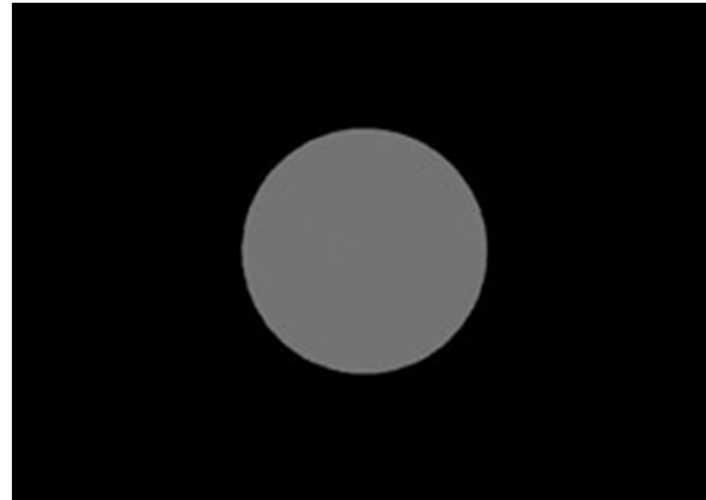
Diagram illustrating the components of the ambient lighting equation:

- K_a is labeled "Propriedade do material" (Material property).
- L_a is labeled "Propriedade da luz" (Light property).



Iluminação - Fundamentos

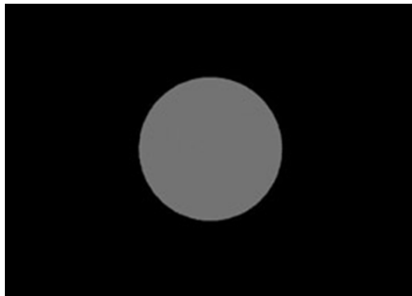
- A iluminação ambiente não tem em consideração a relação espacial entre os objectos e a fonte de luz.
- Todos os objectos são iluminados de forma uniforme independentemente da sua posição ou orientação.
- A própria luz não tem posição nem orientação definidas.





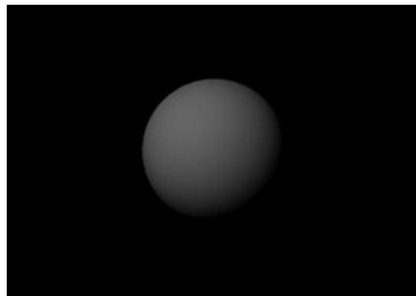
Iluminação - Fundamentos

iluminação ambiente



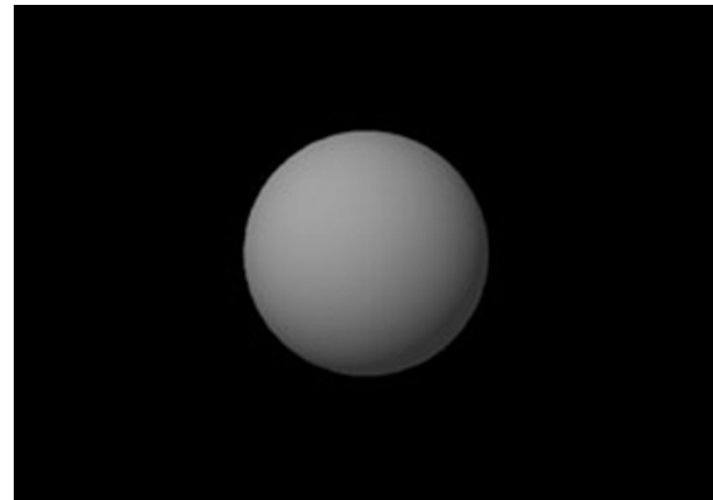
+

iluminação difusa



=

Difusa + Ambiente





Iluminação - Fundamentos

- Ficamos portanto com:

$$I = I_a + I_d = L_a * K_a + L_d * K_d * \cos(\theta)$$

- Caso ambos os vectores estejam normalizados pode-se substituir o coseno pelo produto interno,

$$I = L_a * K_a + L_d * K_d * (N \cdot L)$$

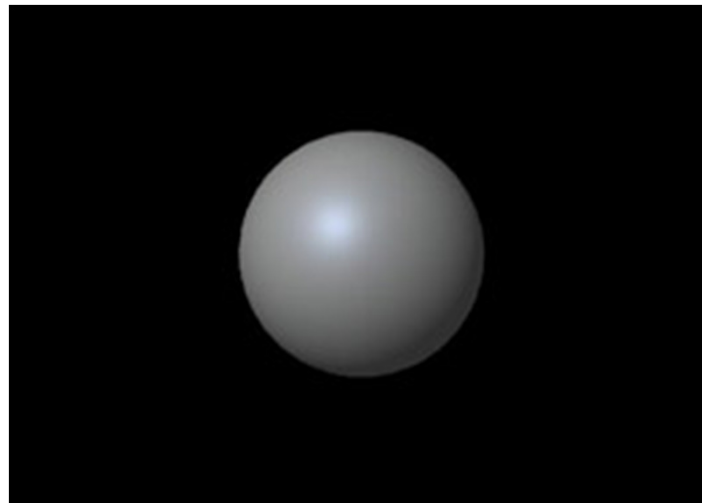
Produto Interno $A \cdot B = A_x * B_x + A_y * B_y + A_z * B_z$

Nota: só se consideram valores positivos do produto interno ou coseno



Iluminação - Fundamentos

- Reflexão Especular - Phong



- Ao iluminar materiais brilhantes verifica-se uma mancha mais clara cuja posição depende da posição do observador.



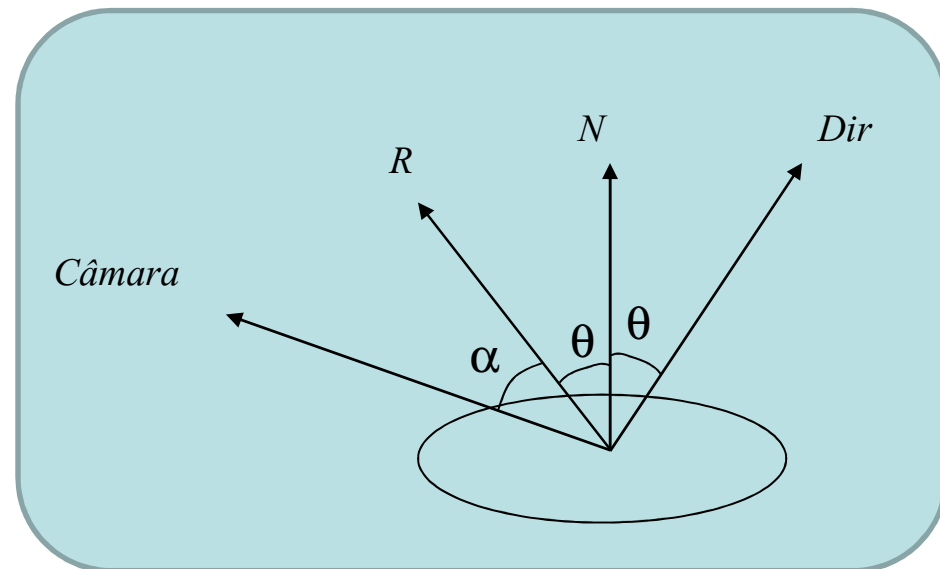
Iluminação - Fundamentos

- Iluminação Especular

- A intensidade especular é proporcional ao ângulo entre o vector reflectido pela superfície (R) e o vector da câmara, ou seja α
- Cálculo do vector de reflexão:

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

$$I_s = L_s * K_s * (R \cdot Câmera)^s$$

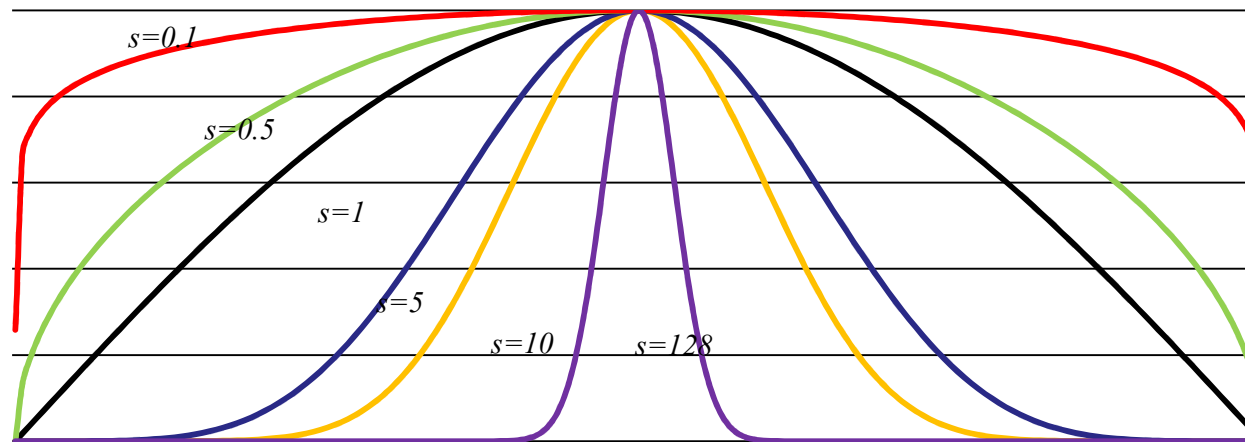
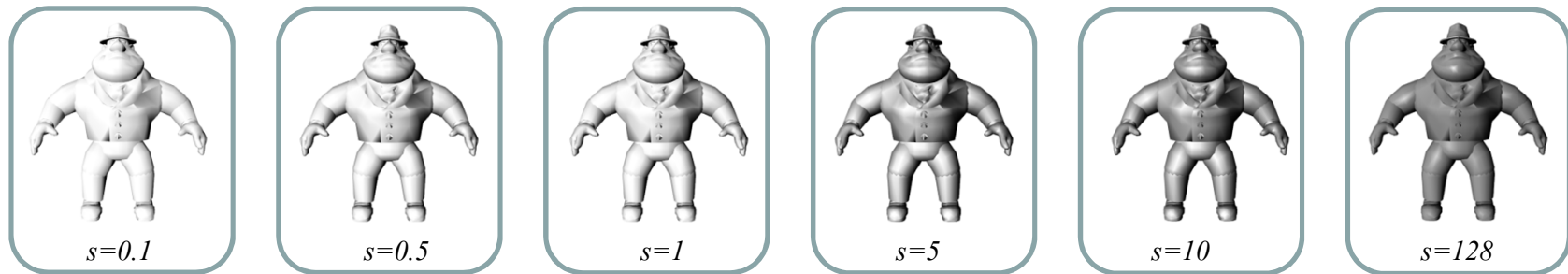


Nota: só se consideram os valores positivos do produto interno



Iluminação - Fundamentos

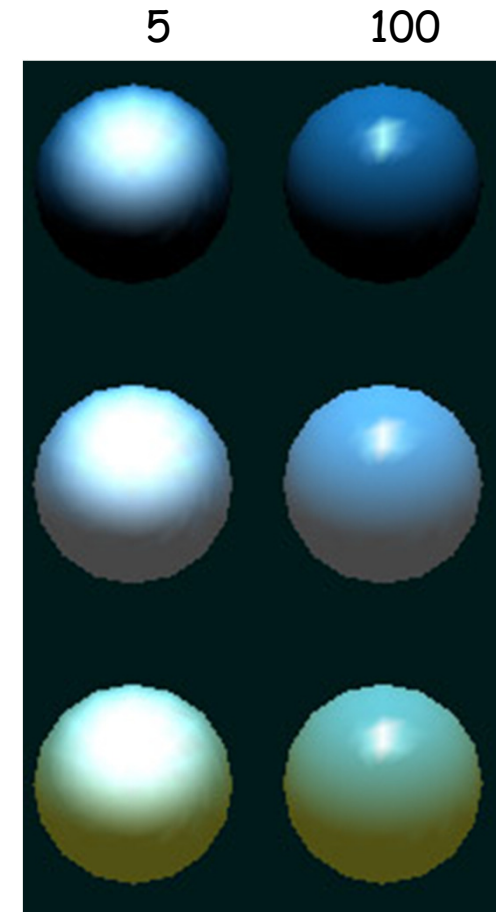
- Componente Especular





Iluminação - Fundamentos

- O coeficiente de especularidade $[0,128]$ determina a dimensão da mancha brilhante.
- Quanto maior o valor, mais pequena é a mancha.
- Materiais metálicos tendem a ter valores altos, enquanto que materiais baços definem-se com valores baixos.

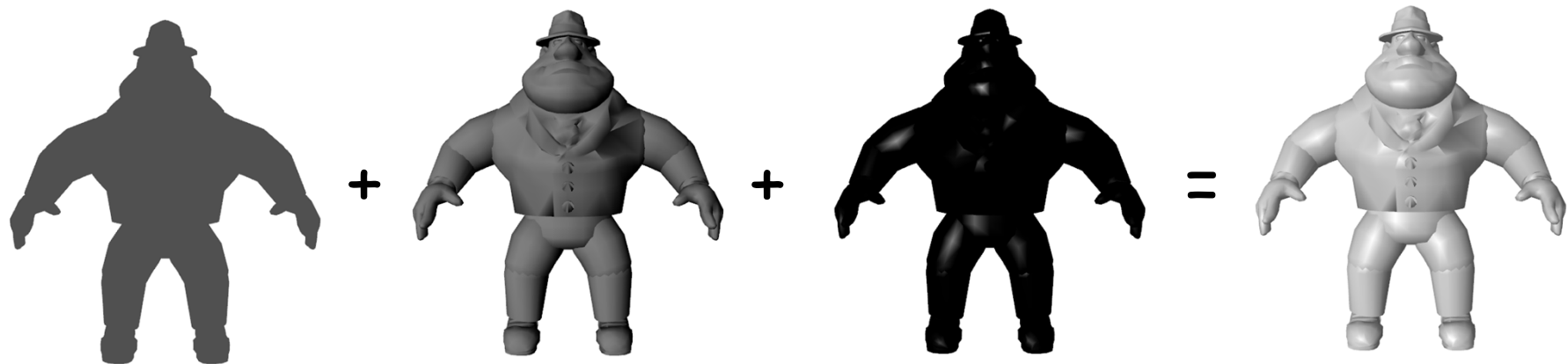




Iluminação - Fundamentos

- Resultado Final

$$I = I_a + f_{att} * [I_d + I_s]$$





Iluminação - Fundamentos

- O OpenGL adiciona dois componentes à equação anterior:
 - K_e : cor emissiva do ponto
 - L_{ga} : luz ambiente global

$$I = K_e + K_a L_{ga} + \sum f_{att} (I_a + I_d + I_s)$$

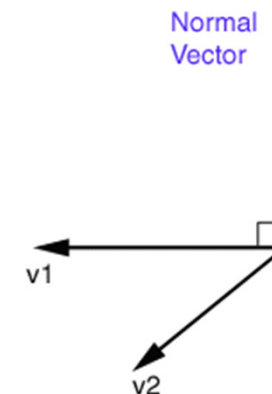
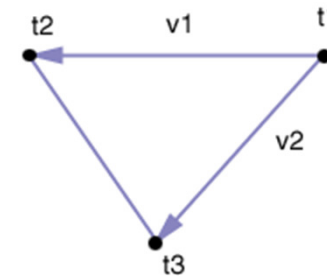


Iluminação - Fundamentos

- Vamos assumir que a iluminação é calculada para cada vértice.
- Para cada vértice é necessário definir a sua normal.
- Podemos considerar a normal como sendo um vector unitário perpendicular à superfície do polígono.
- Considerando um triângulo, podemos definir a sua normal como sendo o produto externo entre duas arestas:

$$n = v_1 \times v_2$$

- Nota: O vector obtido através do produto externo deve ser normalizado.





Iluminação - OpenGL

- Código OpenGL:

```
glBegin(GL_TRIANGLE);  
    glNormal3f(0.2, 1, 0);  
    glVertex3f(0, 0, 0);  
    glNormal3f(0, 1, 0);  
    glVertex3f(0, 0, 1);  
    glNormal3f(0, 1, 0.2);  
    glVertex3f(1, 0, 0);  
glEnd();
```

No caso de se utilizar a mesma normal para todos os vértices escreve-se:

```
glBegin(GL_TRIANGLE);  
    glNormal3f(0, 1, 0);  
    glVertex3f(0, 0, 0);  
    glVertex3f(0, 0, 1);  
    glVertex3f(1, 0, 0);  
glEnd();
```



Iluminação

- Tópicos:
 - Fundamentos de iluminação
 - Aplicação em OpenGL
 - Modelos de Shading
 - Materiais em OpenGL
 - Iluminação em OpenGL



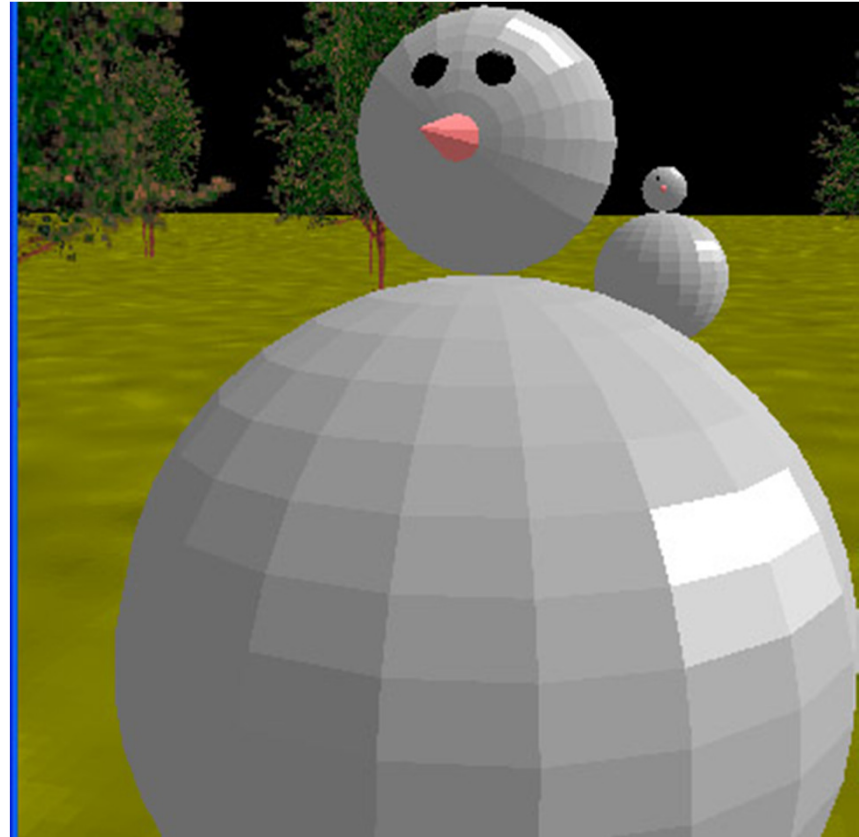
Modelos de *Shading*

- Processo para colorir um polígono (ou superfície) utilizando um determinado modelo de iluminação.
- Alguns modelos de iluminação:
 - Flat (constante)
 - Interpolação
 - Gouraud
 - Phong (não disponível em OpenGL)



Shading - Flat

- Neste modelo o polígono tem iluminação constante em toda a sua superfície.
- Uma normal para cada polígono
- Produz um resultado facetado





Shading - Flat

- Este modelo só faz sentido se:
 - A distância entre a fonte de luz e o polígono for infinita, de forma a que $N.L$ é constante ao longo do polígono.
 - O utilizador encontra-se também a uma distância infinita, para que não haja variação da componente especular ao longo do polígono.
 - A modelação é uma representação fiel da superfície a modelar, i.e. não é uma aproximação.



Shading - Flat

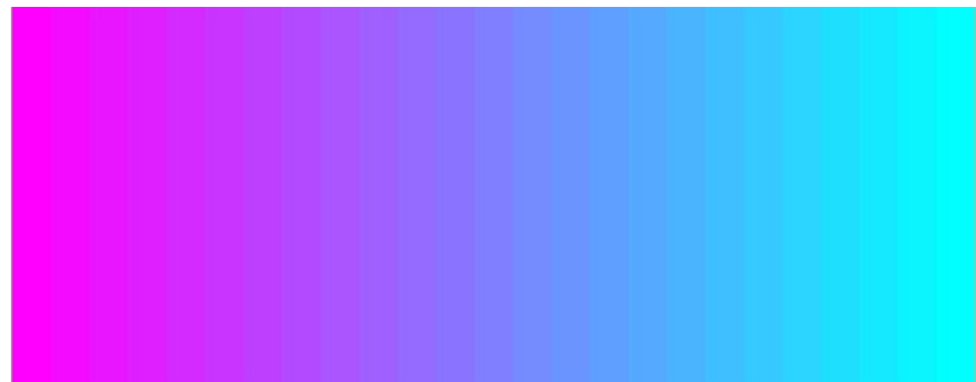
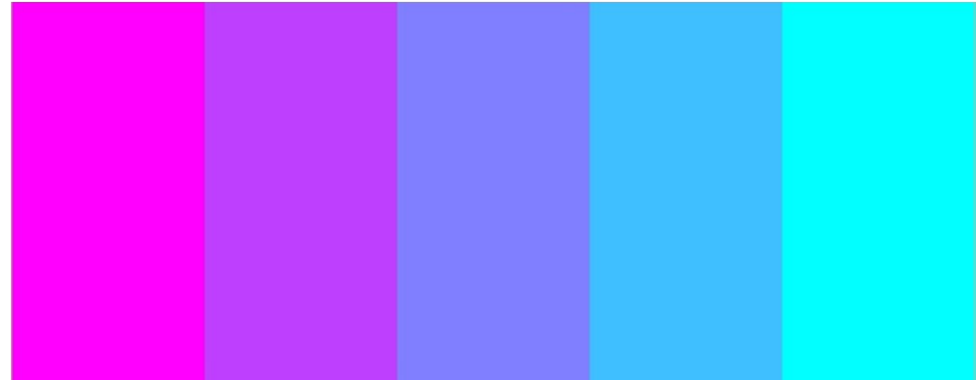
- Problema: Aspecto facetado!
- Solução: Definir uma malha poligonal mais fina?
- Esta solução tem desvantagens óbvias:
 - Implica um número mais elevado de polígonos o que pode diminuir o desempenho.
 - O aspecto facetado é de facto intensificado devido ao efeito das bandas de Mach.



Shading Flat

- Mach Band -

Fenómeno provocado pela disparidade entre a diferença real de intensidade e a intensidade percebida.





Shading - Interpolação

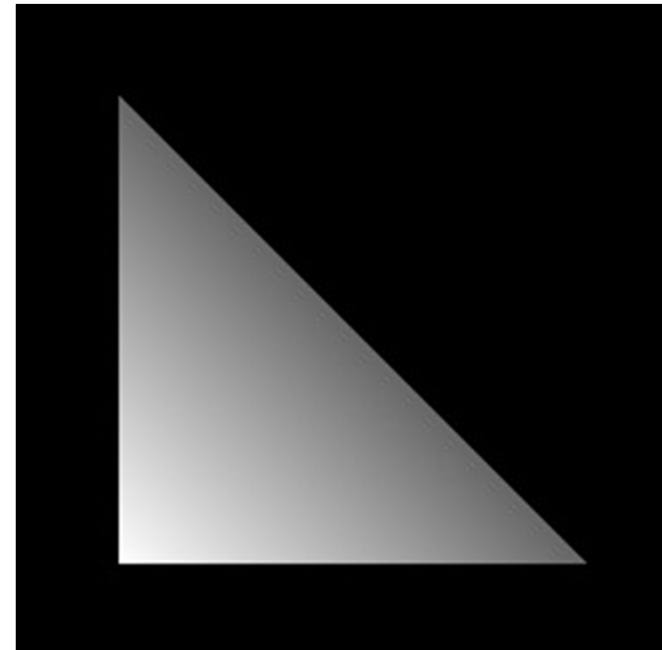
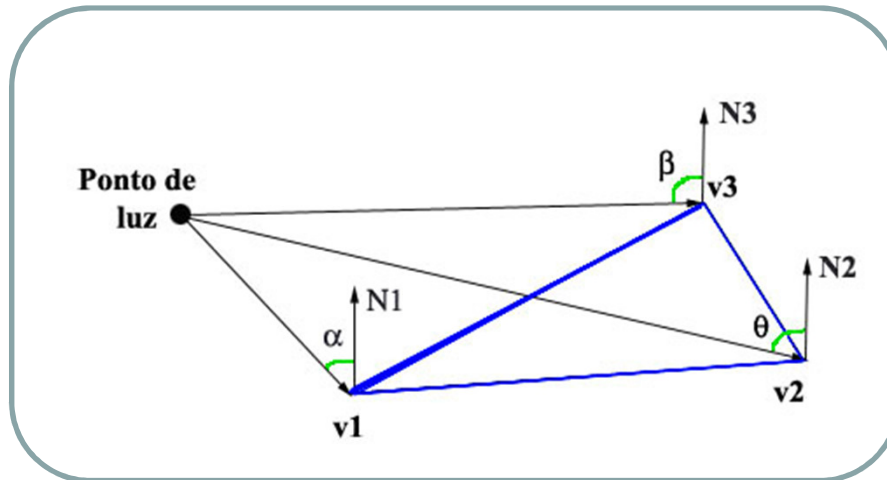
- Neste modelo, proposto por Gouraud, determina-se a intensidade da luz para cada vértice utilizando a normal respectiva.
- A intensidade dos restantes pontos do polígono é calculada por interpolação
- Desta forma elimina-se a primeira restrição do modelo FLAT: a distância do polígono à luz não necessita de ser infinita.



Shading - Interpolação

A intensidade varia ao longo do polígono.

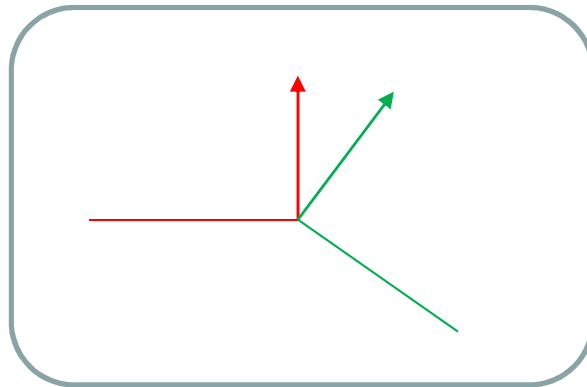
A intensidade do vértice é proporcional ao ângulo entre a sua normal e a direcção da luz.





Shading - Interpolação

- Problema: Superfície continua facetada.



- As normais nos pontos de descontinuidade são diferentes!
- Polígonos com orientações diferentes têm intensidades diferentes nas suas arestas.



Shading - Gouraud

- Muitos dos objectos a modelar são constituídos por superfícies curvas, e a modelação poligonal é apenas uma aproximação.
- Objectivo: Aproximar uma superfície curva por uma malha poligonal
- Mas, se cada polígono for iluminado individualmente ...
- ... mantem-se a aparência facetada, e por consequência torna-se fácil distinguir um polígono dos seus vizinhos, cuja orientação é diferente.



Shading - Gouraud

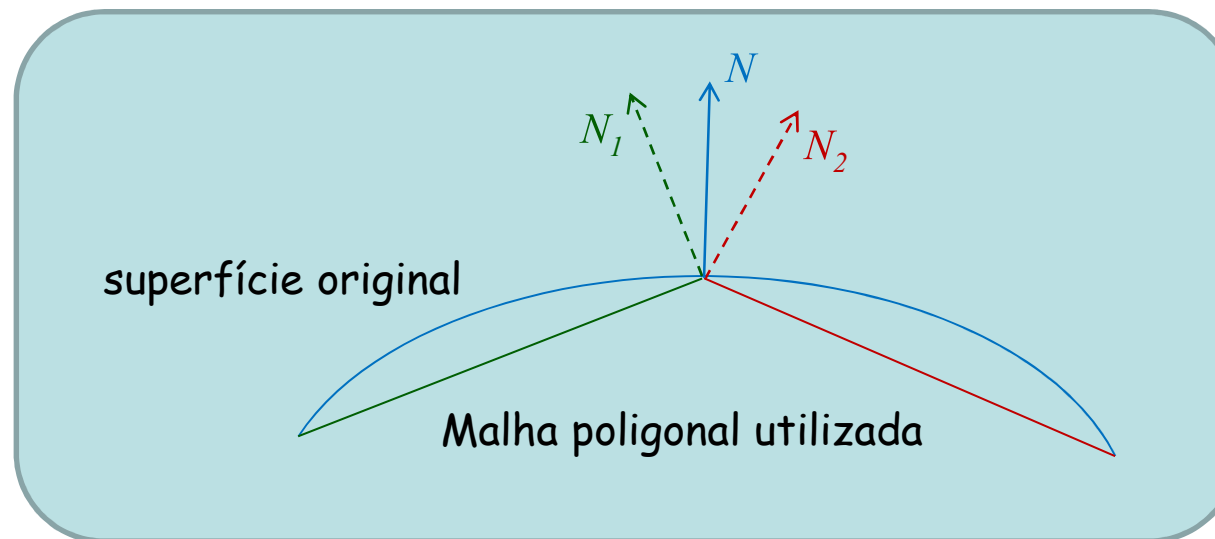
Para resolver este problema, Gouraud posteriormente sugeriu que ...

- ... cada vértice tivesse uma normal que representasse, não a orientação do polígono, ...
- ... mas sim a normal da superfície que a malha poligonal pretende aproximar.



Shading - Gouraud

- Isto implica que as normais da superfície original a aproximar sejam conhecidas para cada vértice.



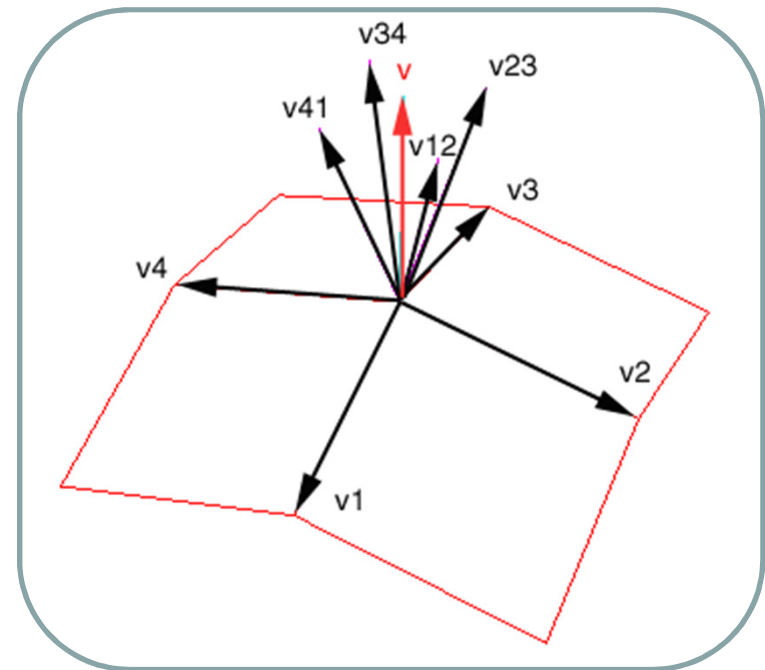
N - Normal da superfície original

N_1, N_2 - Normais individuais de cada polígono



Shading - Gouraud

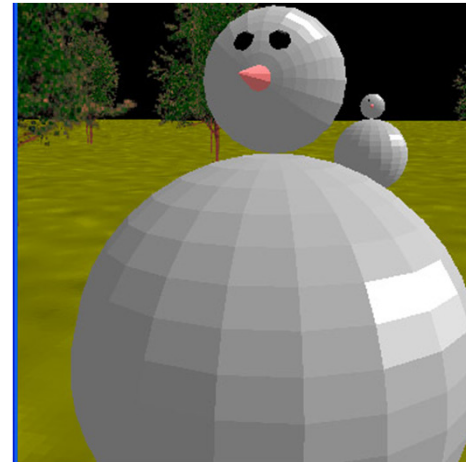
- No caso das normais da superfície não serem conhecidas, e não for possível o seu cálculo, ...
- ... é possível obter uma aproximação através da média (normalizada) das normais de cada polígono individual que partilhe o vértice.



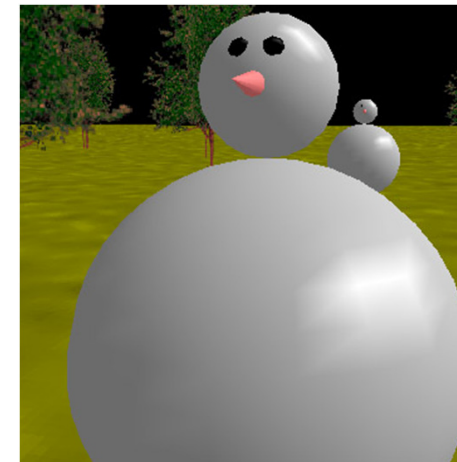


Shading - Gouraud

modelo com flat shading



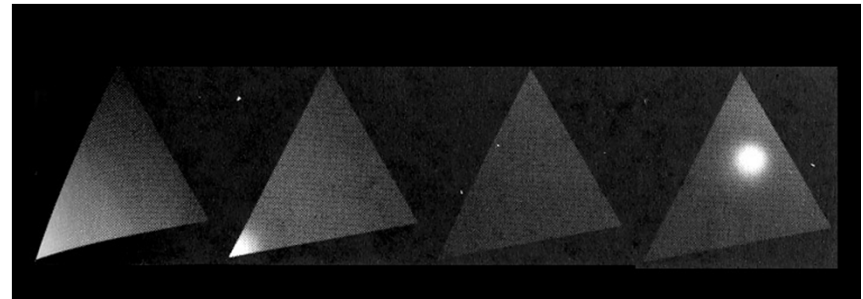
Gouraud shading





Shading - Gouraud

- O modelo de Gouraud não elimina completamente o problema das bandas de Mach, embora as reduza consideravelmente.
- As manchas especulares não são reproduzidas fielmente

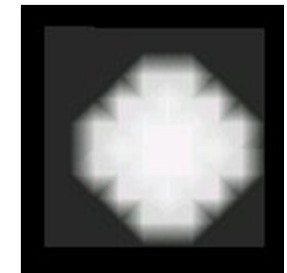




Shading - Gouraud

- Problema: Dependência da iluminação nos vértices do polígono.
- Um polígono parcialmente iluminado, em que nenhum dos vértices é iluminado é representado como se a totalidade do polígono não fosse iluminado.
- Solução: Malha mais fina?

luz circular inclui
só um canto do
polígono

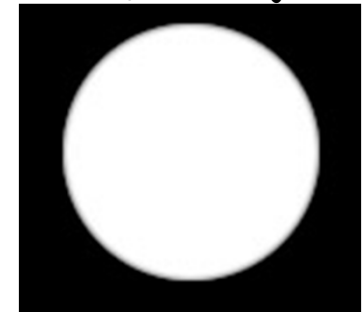


malha mais
fina

luz circular
inteiramente
dentro do polígono



situação desejada





Shading - Phong

- Phong propõe:

Interpolar Normais em vez de Intensidades

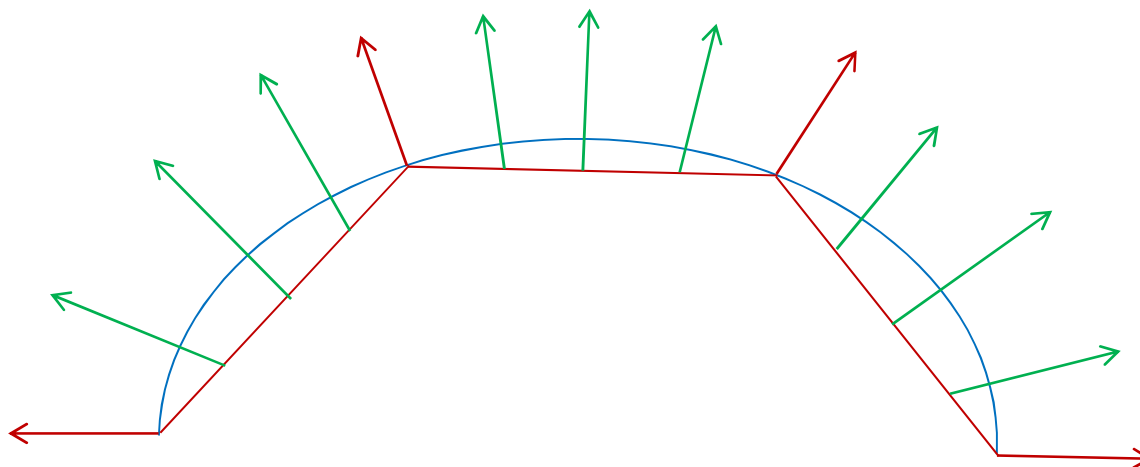
- Problema: Tempo Real? (já ultrapassado 😊)



Shading - Phong

- Phong propõe:

Interpolar Normais em vez de Intensidades



- ↗ Normais por vértice, iguais às da superfície
- ↗ Normais interpoladas para os pontos dos polígonos



Shading Phong

- Phong vs. Gouraud
 - Gouraud:
 - Por vértice:
 - Cálculo da normal e intensidade (baseada na normal computada)
 - Para os outros pontos do polígono:
 - Cálculo da intensidade por interpolação dos valores por vértice
 - Phong:
 - Por vértice:
 - Cálculo da normal
 - Para os outros pontos do polígono:
 - Cálculo da normal por interpolação dos valores dos vértices
 - Cálculo da intensidade com base na normal interpolada



Shading

- Gouraud



- Phong





Shading

- Problema: Silhueta
- Independente da qualidade do modelo de shading!





Shading

- Problema: Nos modelos apresentados os objectos não interagem em termos de iluminação. São modelos locais.
- Consequências:
 - Objectos não causam sombras noutros objectos (podem-se conseguir soluções satisfatórias em tempo real)
 - A luz reflectida por um objecto não é tida em conta na iluminação de outro objecto (demasiado pesado de um ponto de vista computacional por enquanto)

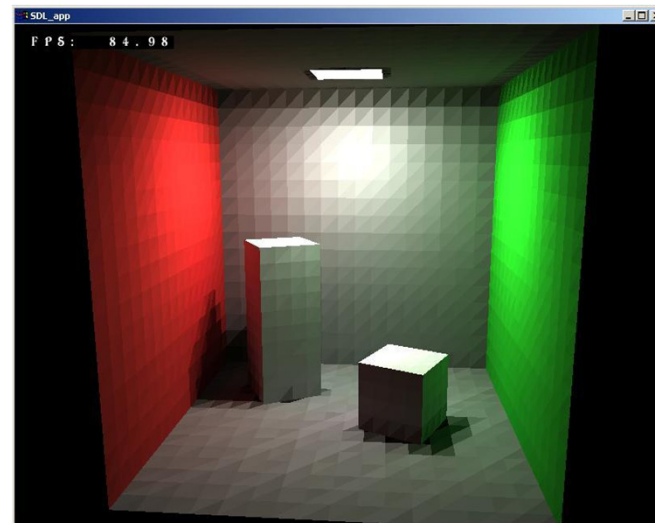
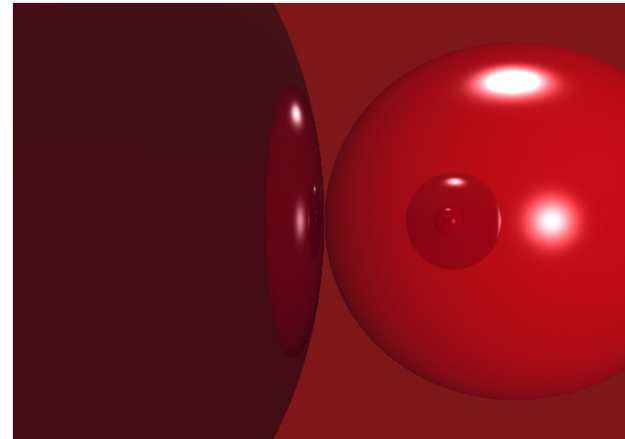


A luz vem da esquerda



Shading






- Modelos mais complexos, que contemplem iluminação global, não são utilizáveis em tempo real:
 - Ray tracing
 - cálculo de interações especulares
 - cálculo de sombras
 - Radiosidade
 - cálculo de interações difusas
 - Photon Mapping
 - Virtual Point Lights

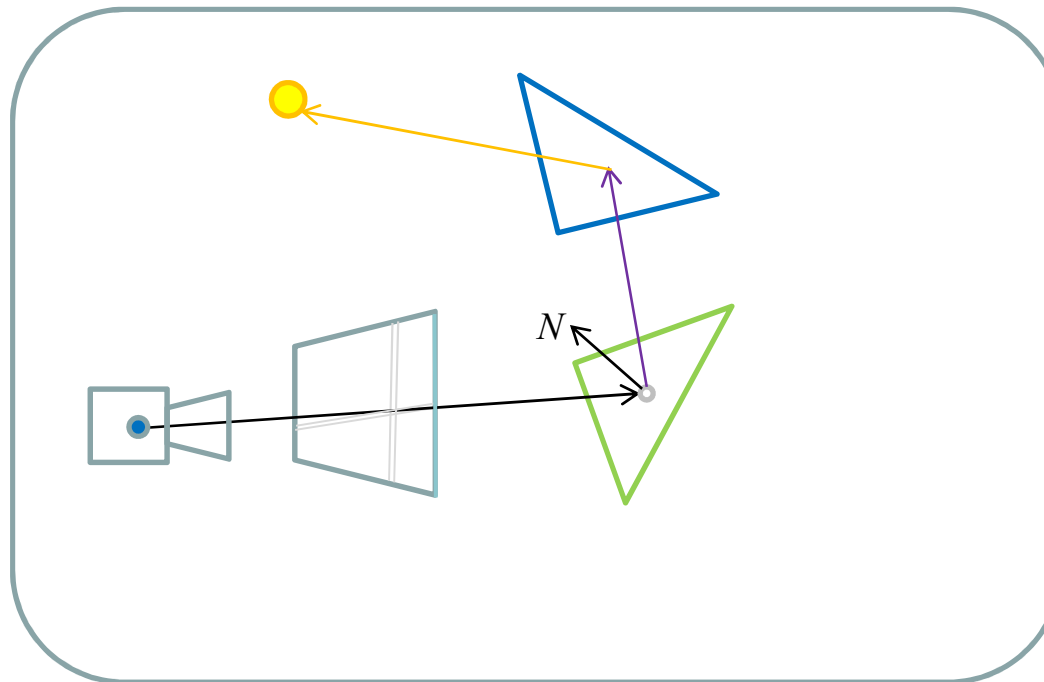




Ray Tracing

- Esquema Geral: Superfícies Especulares

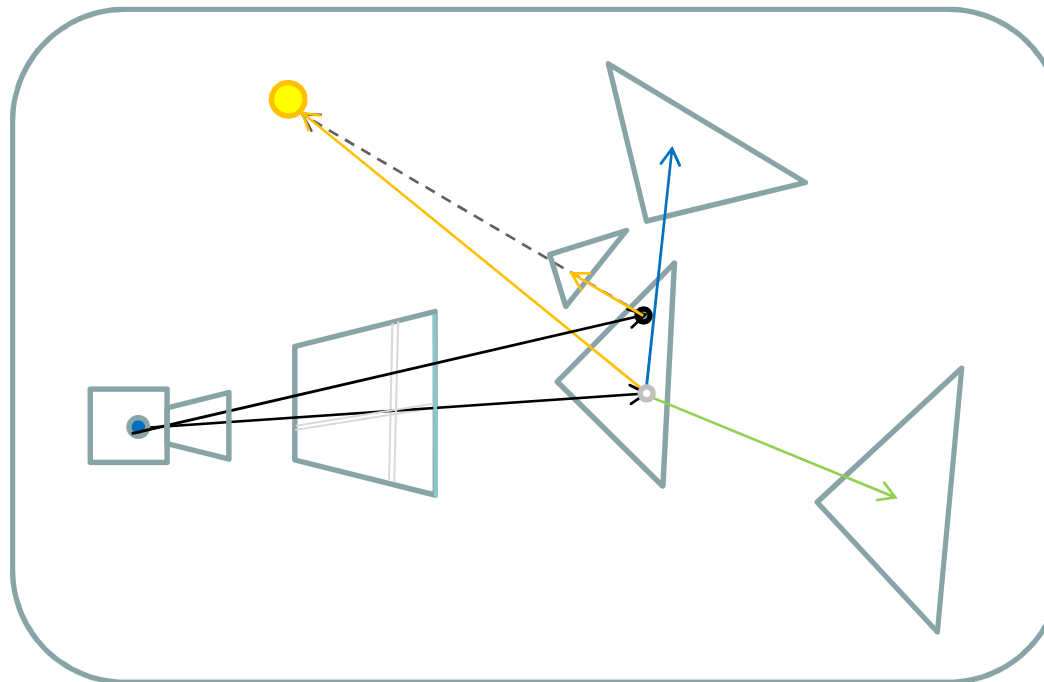
-  Superfície Difusa
-  Superfície Especular
-  Raio Primário
-  Raio Refletido
-  Raio Sombra





Ray Tracing

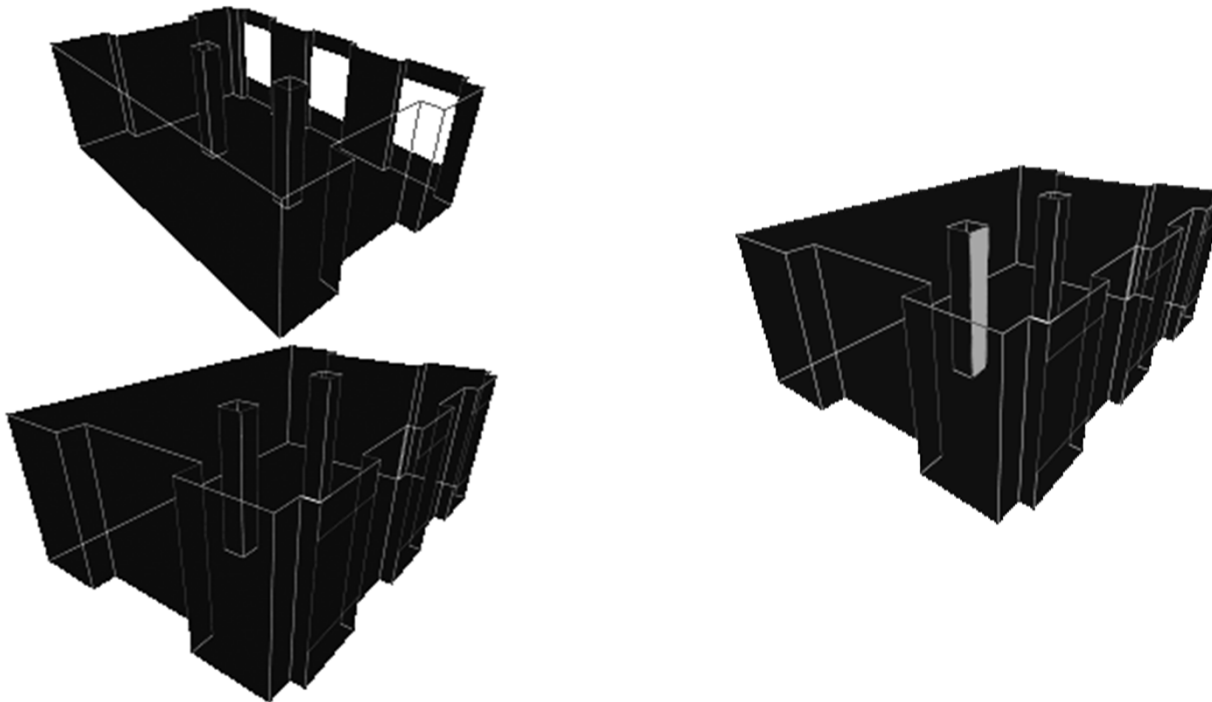
- Esquema Geral: Superfícies Difusas + Transparência





Radiosidade

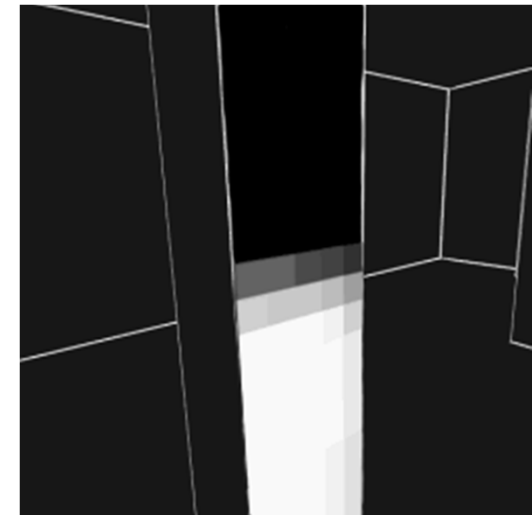
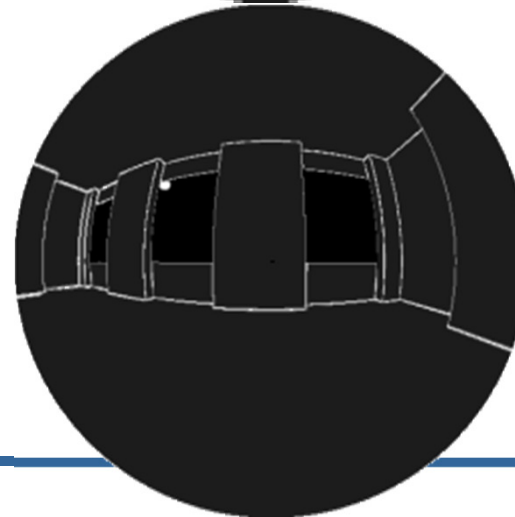
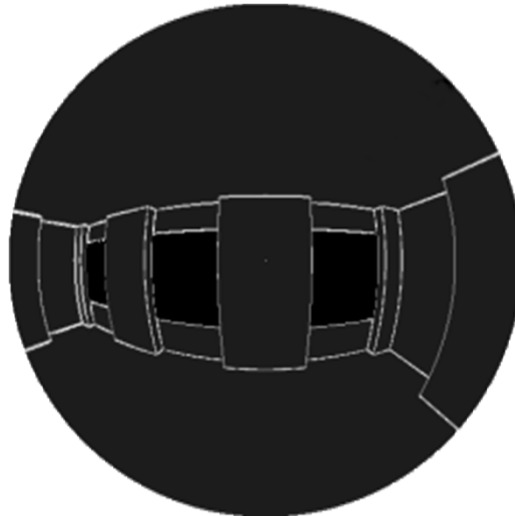
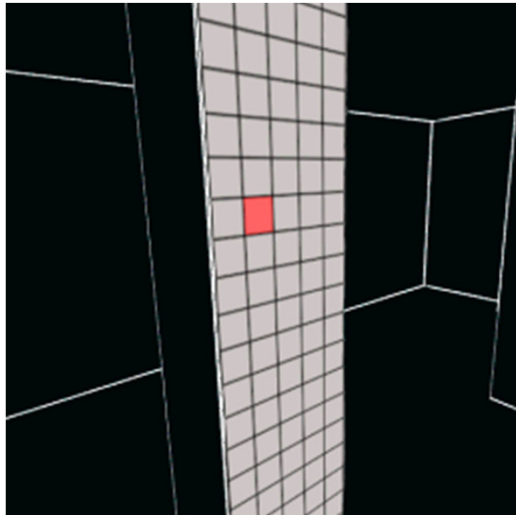
- Iluminação calculada em "patches"





Radiosidade

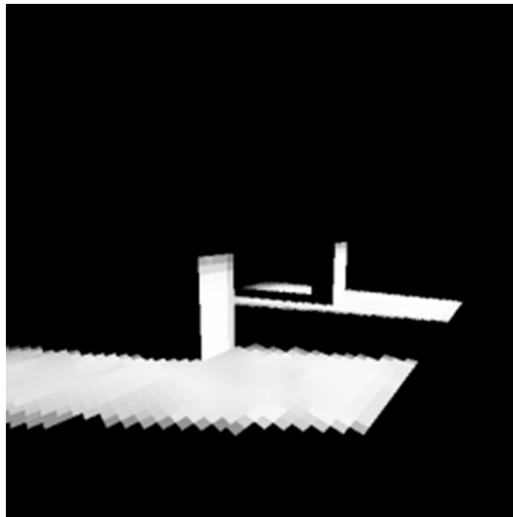
- Patches





Radiosidade

- Passo 1 e nova vista a partir do patch

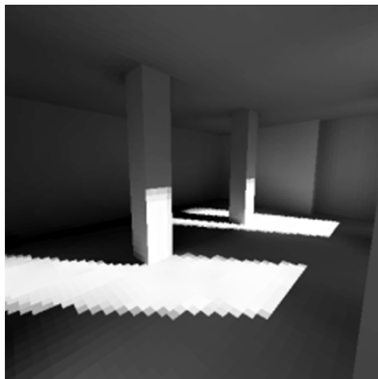




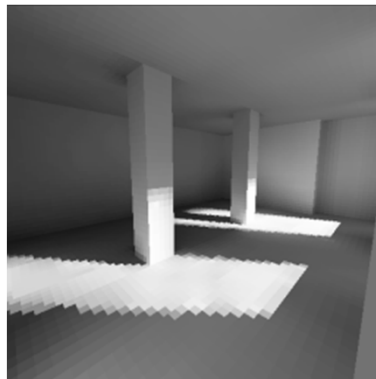
Radiosidade

- Processo Iterativo

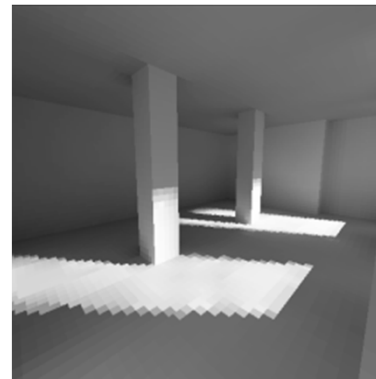
passo 2



passo 3

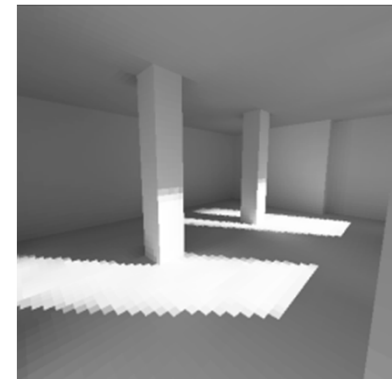


passo 4



...

passo 16



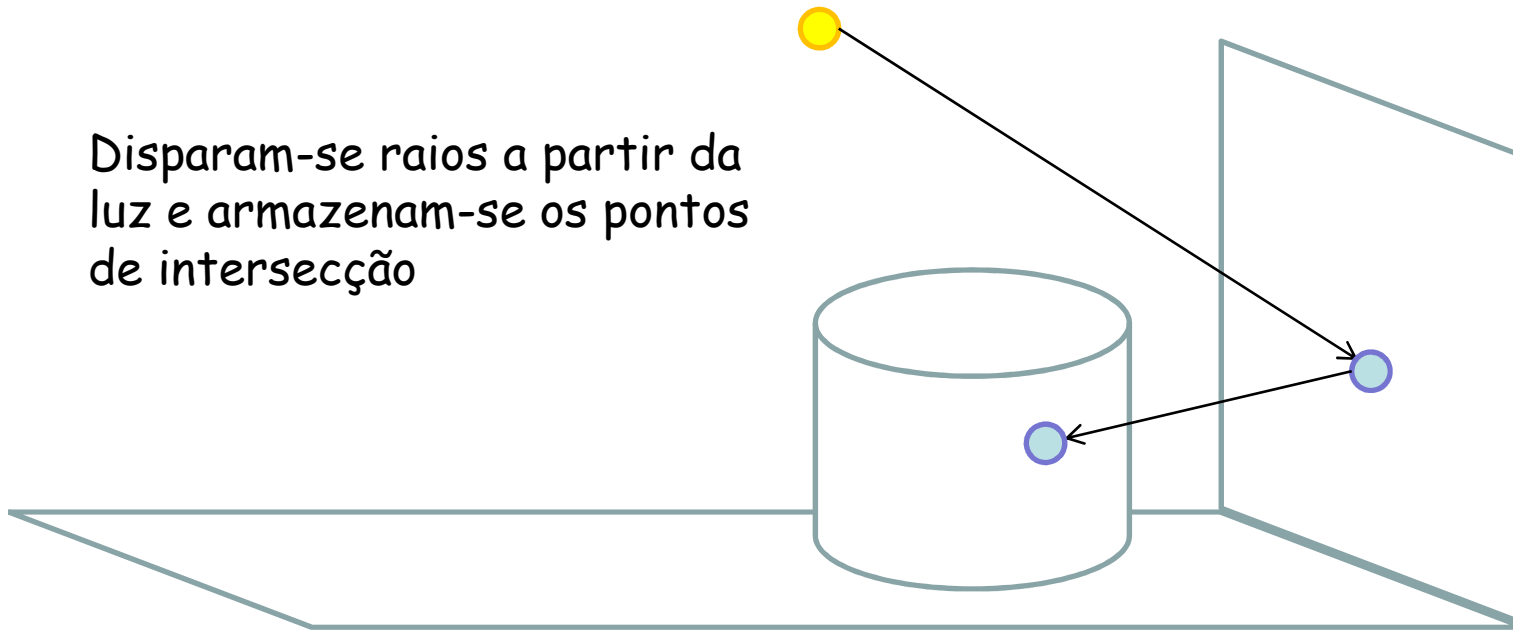
(todas as imagens do site de Hugo Elias)



Virtual Point Lights

- Passo 1 - Iniciar o sistema: Criar pontos de luz virtuais

Disparam-se raios a partir da luz e armazenam-se os pontos de intersecção

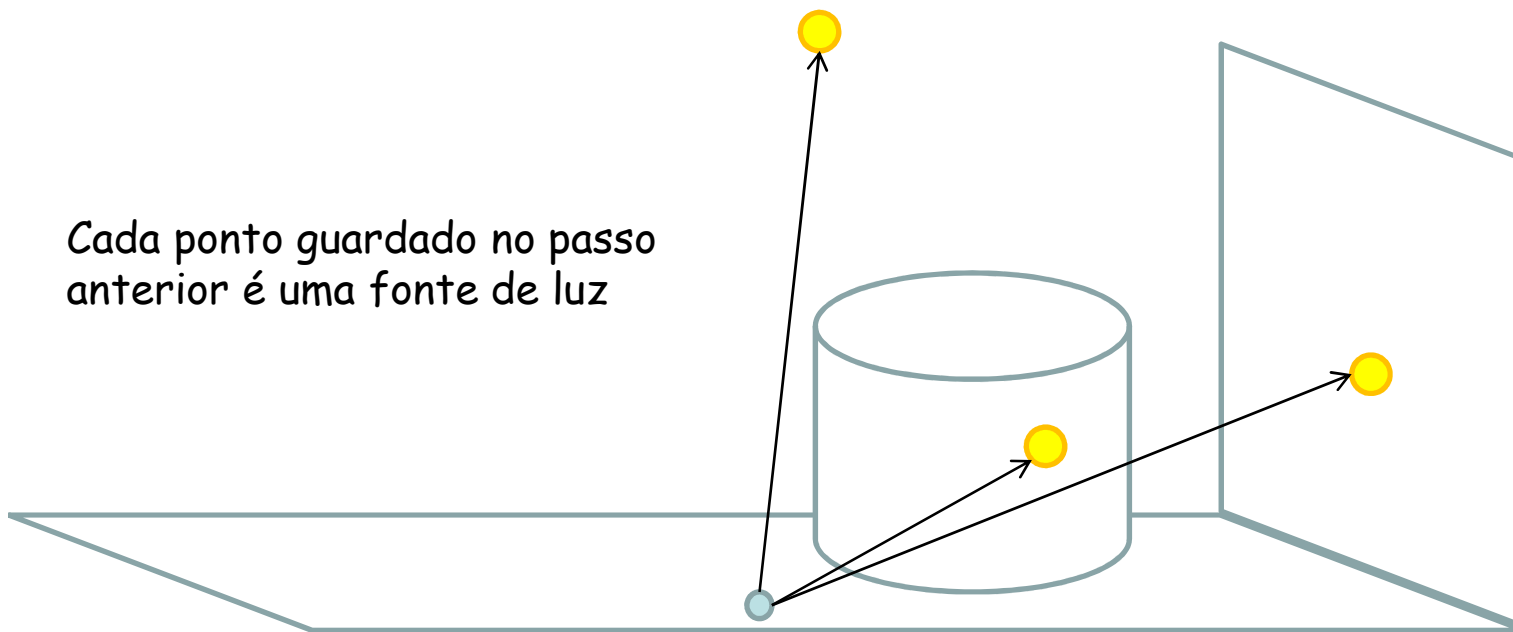




Virtual Point Lights

- Passo 2 - Render: Cada ponto recebe contribuições das diversas fontes de luz

Cada ponto guardado no passo anterior é uma fonte de luz





Iluminação

- Tópicos:
 - Fundamentos de iluminação
 - Aplicação em OpenGL
 - Modelos de Shading
 - Materiais em OpenGL
 - Iluminação em OpenGL



OpenGL - Materiais

- Componentes da cor:
 - Difusa
 - Especular
 - Ambiente
 - Emissiva



OpenGL - Materiais

- Teapot Azul



Luz Branca



Luz Vermelha



OpenGL - Materiais

- Teapot azul (0.3,0.3,1.0)

Luz Branca
(1.0,1.0,1.0)



Luz 100% Vermelha
(1.0,0.0,0.0)



Luz Predominantemente Vermelha
(1.0,0.3,0.3)





OpenGL - Materiais

- Atribuir materiais:

```
glMaterialfv(GL_FRONT, componente, array);  
glMaterialf(GL_FRONT, GL_SHININESS, valor);
```

0..128

Componente:

GL_DIFFUSE

GL_AMBIENT

GL_SPECULAR

GL_EMISSION

GL_AMBIENT_AND_DIFFUSE



Iluminação

- Tópicos:
 - Fundamentos de iluminação
 - Aplicação em OpenGL
 - Modelos de Shading
 - Materiais em OpenGL
 - Iluminação em OpenGL



OpenGL - Iluminação

- Definir propriedades da luz

```
glLight{if}(GL_LIGHTi, param, valor1, valor2, ...);  
glLight{if}v(GL_LIGHTi, param, array_valores)
```

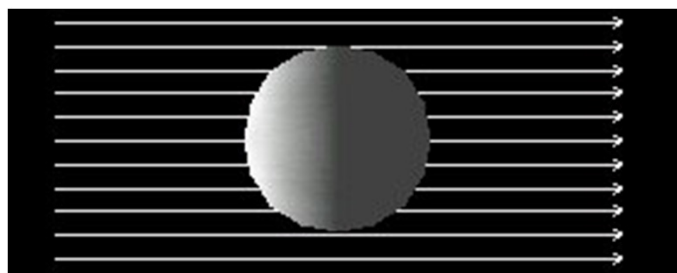


OpenGL - Iluminação

Direccional

Atributos: direcção, cor

GL_POSITION





OpenGL - Iluminação

- Definir uma luz direccional

```
GLfloat amb[3] = {0.2, 0.2, 0.2};
```

```
GLfloat diff[3] = {1.0, 1.0, 1.0};
```

```
GLfloat pos[4] = {0.0, 0.0, 1.0, 0.0};
```

0.0 indica que a luz é
direccional

← A posição indica a
direcção da luz

```
glLightfv(GL_LIGHT0, GL_POSITION, pos);
```

```
glLightfv(GL_LIGHT0, GL_AMBIENT, amb);
```

```
glLightfv(GL_LIGHT0, GL_DIFFUSE, diff);
```



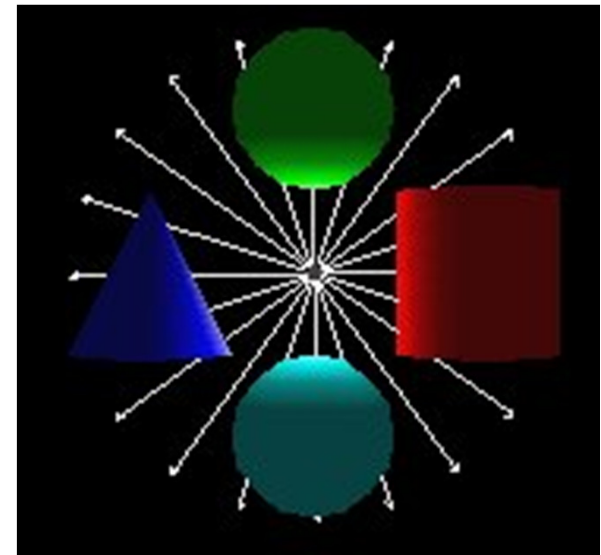
OpenGL - Iluminação

- Ponto de Luz

Atributos: posição, atenuação, cor

`GL_POSITION`

`GL_..._ATTENUATION`





OpenGL - Iluminação

- Definir um ponto de luz

```
GLfloat amb[3] = {0.2, 0.2, 0.2};  
GLfloat diff[3] = {1.0, 1.0, 1.0};  
GLfloat pos[4] = {0.0, 0.0, 10.0, 1.0};
```

1.0 indica que se trata
de um ponto de luz

```
glLightfv(GL_LIGHT0, GL_POSITION, pos);  
glLightfv(GL_LIGHT0, GL_AMBIENT, amb);  
glLightfv(GL_LIGHT0, GL_DIFFUSE, diff);
```



OpenGL - Iluminação

Foco de Luz (Spotlight)

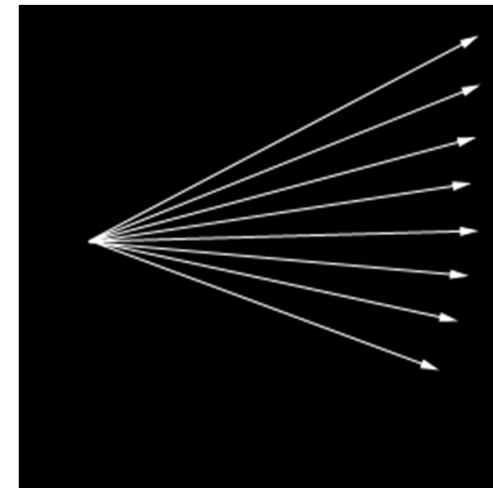
Atributos: posição, ângulo, atenuação, direcção, cor

GL_POSITION

GL_SPOT_DIRECTION

GL_SPOT_CUTOFF

GL_..._ATTENUATION





OpenGL - Iluminação

- Definir um foco de luz

```
GLfloat diff[3] = {1.0, 1.0, 1.0};
GLfloat pos[4] = {0.0, 0.0, 10.0, 1.0};
GLfloat spotDir[3] = {0.0, 0.0, -1.0};

glLightfv(GL_LIGHT0, GL_POSITION, pos);
glLightfv(GL_LIGHT0, GL_DIFFUSE, diff);
glLightfv(GL_LIGHT0, GL_SPOT_DIRECTION, spotDir);
glLightf(GL_LIGHT0, GL_SPOT_CUTOFF, 45.0);
// [0,90] ou 180
glLightf(GL_LIGHT0, GL_SPOT_EXPONENT, 0.0);
// [0,128]
```



OpenGL - Iluminação

- Ligar, desligar luzes individuais (por omissão estão desligadas)

```
glEnable(GL_LIGHTi); // i = 0..7  
glDisable(GL_LIGHTi);
```

- Ligar, desligar o quadro

```
glEnable(GL_LIGHTING);  
glDisable(GL_LIGHTING);
```



OpenGL - Iluminação

Demo sobre luzes e materiais



OpenGL - Iluminação

Demo sobre posicionamento da luz



Referências

- **Computer Graphics – Principles and Practice**, Foley, van Dam, Feiner and Hughes
- **OpenGL Reference Manual**, OpenGL ARB
- **OpenGL Programming Guide**, OpenGL ARB
- **Radiosidade**
<http://freespace.virgin.net/hugo.elias/radiosity/radiosity.htm>