

Computação gráfica

Síntese de imagem (rendering)

UERN - Curso de Ciência da Computação

Prof.: Wilfredo Blanco Figuerola



Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Algoritmo simples

Para cada pixel da tela

 Lance um raio;

 Para cada objeto da cena

 Calcule a interseção do raio com este o objeto;

 Armazene a interseção mais próxima;

 Se o raio interceptou algum objeto

Calcule a contribuição das luzes na cor deste ponto;

 Pinte o pixel com esta cor.

 senão

 Pinte o pixel com de fundo.

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação

- ▶ Uma vez calculado o ponto de intercepção entre o raio e objetos da cena gráfica, então determinamos a cor correspondente deste ponto (pixel)
- ▶ Calculo da energia luminosa que emana da superfície dos objetos e chega ate a câmera (muito complexo)
- ▶ Como as superfícies **refletem** a luz.

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação (Phong)

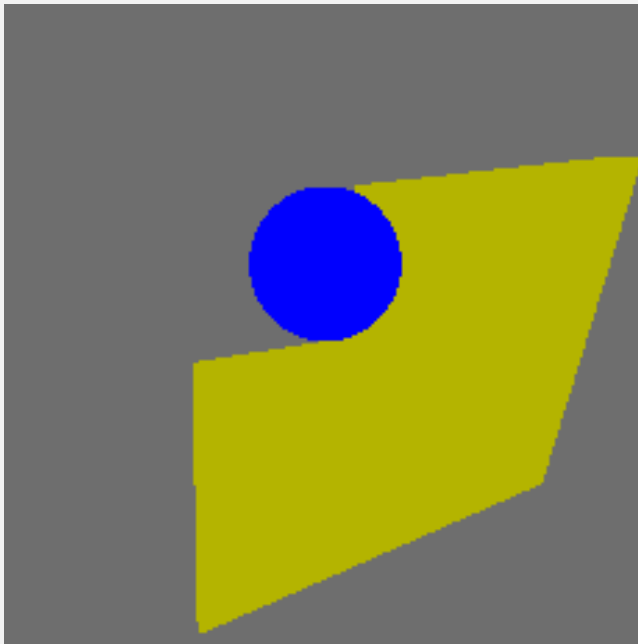
- ▶ Modelo de iluminação de Phong (Bui Tuong Phong, na Universidade de Utah, em 1973)
- ▶ Componentes:
 - ▶ Cor sem iluminação. (I_a)
 - ▶ Cor da reflexão difusa. (I_d)
 - ▶ Cor da reflexão especular. (I_s)

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação (Phong)

► Cor sem iluminação



Arquivo de modelos e dados (.rt4)

#SCENE

110. 110. 110. null => cor_de_fundo

...

MATERIAL

0. 0. 100. => cor_ambiente (*I_a*)

0. 0. 255. => comp. difusa (*k_d*)

255.255. 255. => comp. Especular (*k_s*)

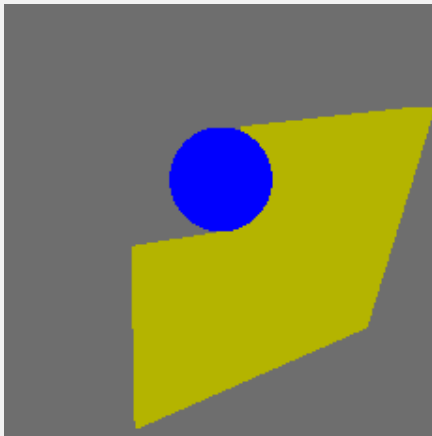
50. => coeficiente, caracteriza o maior ou menor espalhamento do ponto brilhante (*n*)

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação (Phong)

► Cor sem iluminação (1)



- Representa bem Geometria e Oclusão
- Sem interação com a luz perdemos a percepção 3D

```
// Para cada objeto calcular intercepção
for(i=0;i<=sceGetObjectCount( scene )-1;i++)
{
    Valor_t[i] = ...
    objIntercept( sceGetObject( scene, i),eye,ray);
}

// Buscar o menor "t" => objeto "j"

// se (j>=0)
//   Calcula o ponto usando eq. do raio ( eye, ray, Valor_t[j]);
//   Calcula vetor normal normalizado do objeto no ponto
//   retornar cor_ambiente;
// fim

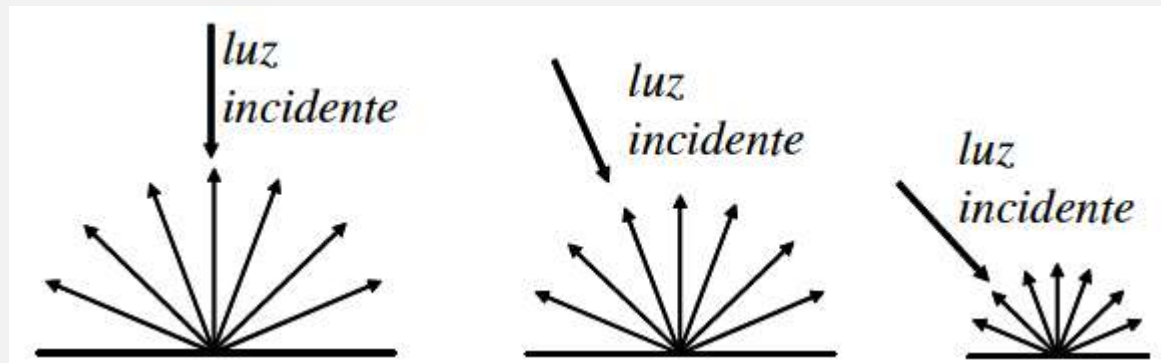
//retornar cor_de_fundo.
```

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação (Phong)

- ▶ Cor de reflexão difusa (2)
 - ▶ Depende da posição relativa entre a normal ao objeto naquele ponto e a direção de onde vem a luz.



Superfícies lambertianas:

- Superfícies foscas,
- Cor não depende da posição do observador.

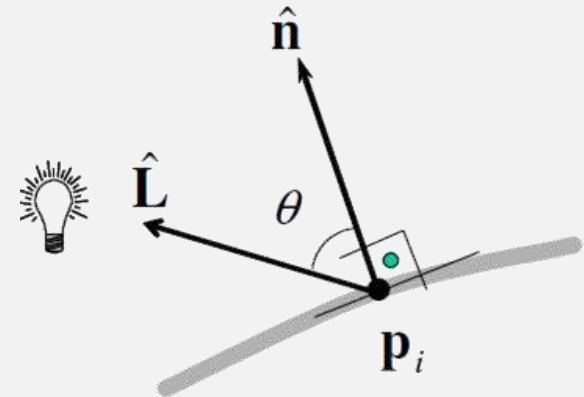
Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação

► Cor de reflexão difusa (2)

$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} l_r k_{dr} \cos \theta \\ l_g k_{dg} \cos \theta \\ l_b k_{db} \cos \theta \end{pmatrix}$$



- $(l_r, l_g, l_b)^T$ são as intensidades RGB da luz
- $(k_{dr}, k_{dg}, k_{db})^T$ é chamada a cor difusa do material
- θ é o ângulo entre a normal e a direção da luz

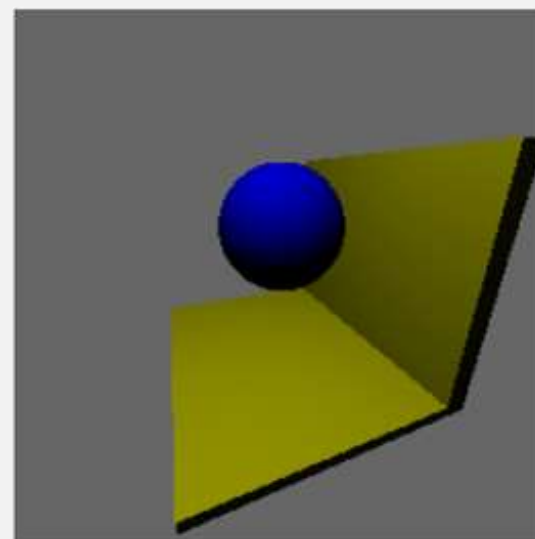
Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação

- Cor de reflexão difusa (2)

$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} l_r k_{dr} (\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}) \\ l_g k_{dg} (\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}) \\ l_b k_{db} (\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}) \end{pmatrix} = \begin{pmatrix} l_r k_{dr} \\ l_g k_{dg} \\ l_b k_{db} \end{pmatrix} (\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}) = \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} (\hat{\mathbf{n}} \cdot \hat{\mathbf{L}})$$



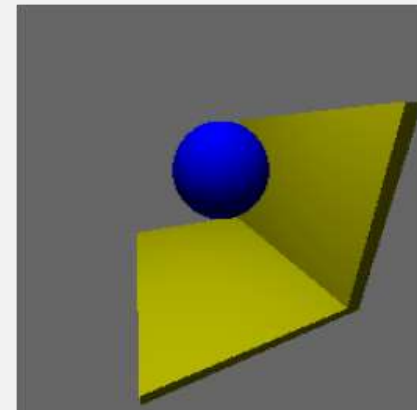
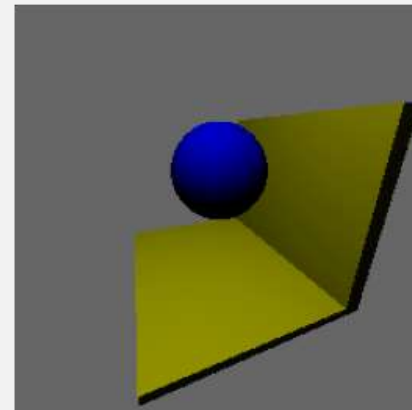
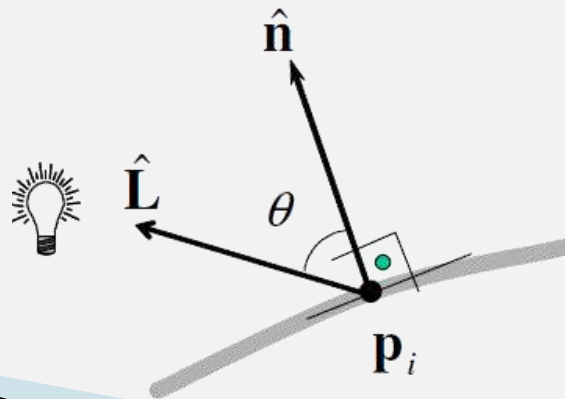
Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação

- ▶ Cor de reflexão ambiente + difusa

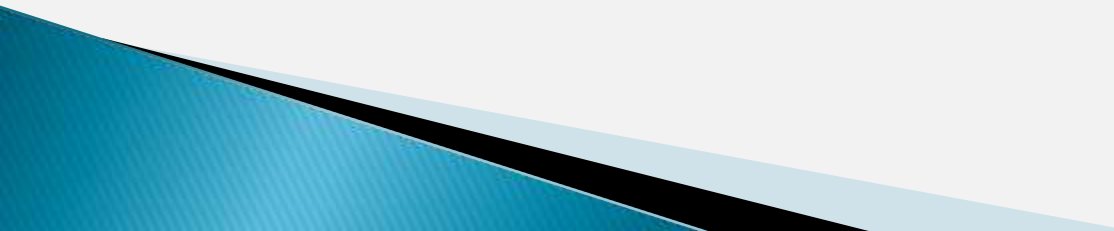
$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} + \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} (\hat{\mathbf{n}} \cdot \hat{\mathbf{L}})$$



Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

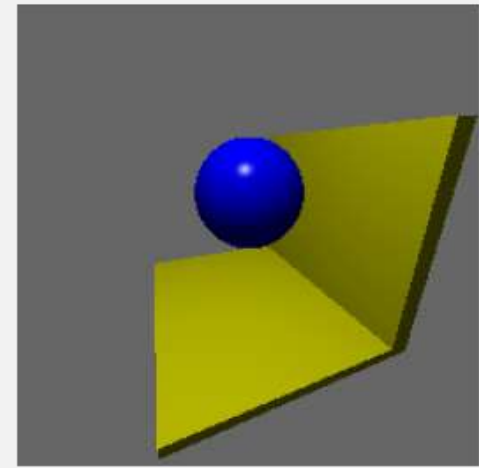
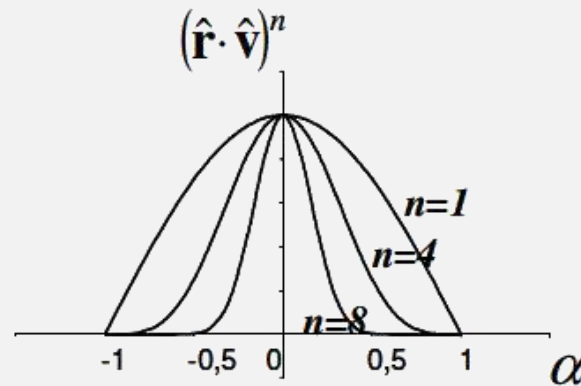
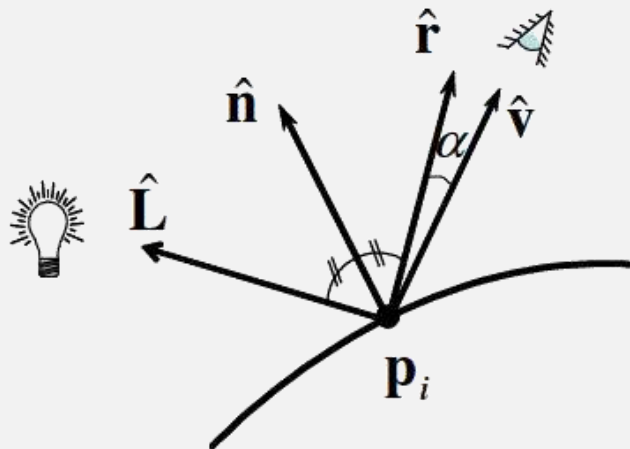
Modelo de iluminação

- ▶ Cor de reflexão especular (3)
 - ▶ Nem todas as superfícies (metálico polido) são foscas como preconiza o modelo lambertiano.
 - ▶ Dependendo da posição com relação à fonte luminosa, um observador vê pontos brilhantes.
 - ▶ Nestes pontos brilhantes a superfície age como um espelho da luz.
- 

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação

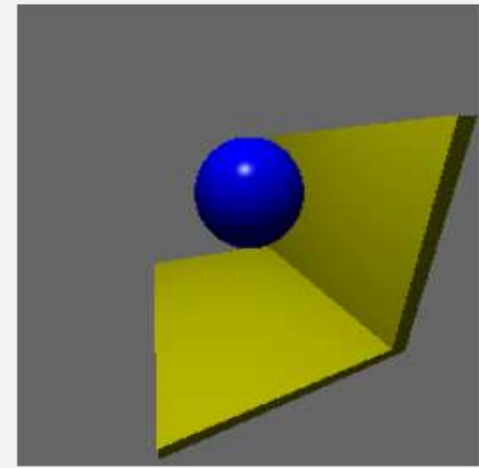
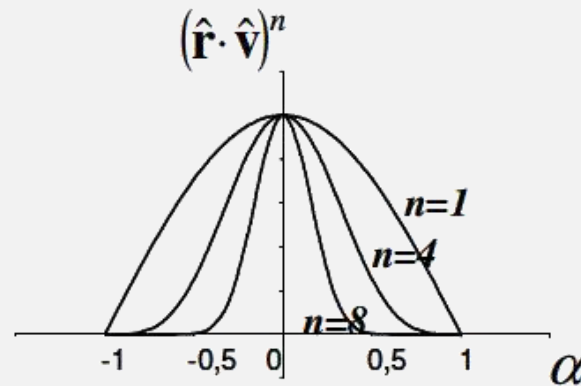
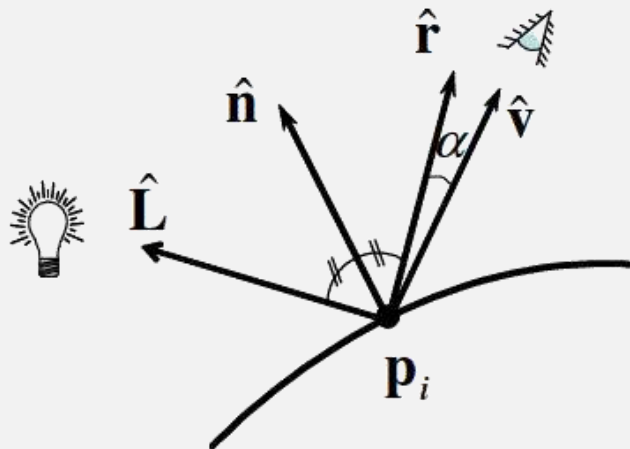


- ▶ os pontos brilhantes são vistos em torno da direção refletida \hat{r} mostrada.
- ▶ Esta direção é obtida refletindo-se o vetor que aponta para a fonte luminosa, \hat{L} , em torno da normal \hat{n} .

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação



- ▶ os pontos brilhantes são vistos em torno da direção refletida \hat{r} mostrada.
- ▶ Esta direção é obtida refletindo-se o vetor que aponta para a fonte luminosa, \hat{L} , em torno da normal \hat{n} .

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação

- Cor de reflexão ambiente + difusa + especular (3)

$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} + \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} (\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}) + \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{sr} \\ k_{sg} \\ k_{sb} \end{pmatrix} (\hat{\mathbf{r}} \cdot \hat{\mathbf{v}})^n$$

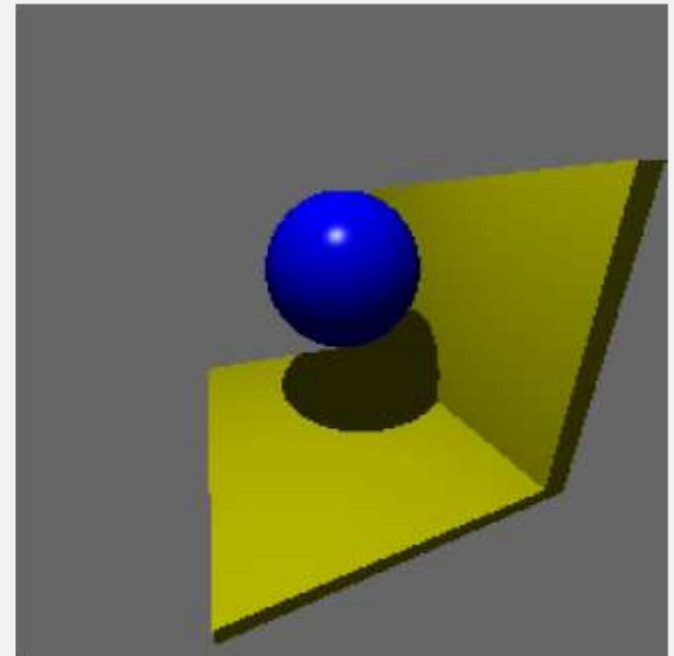
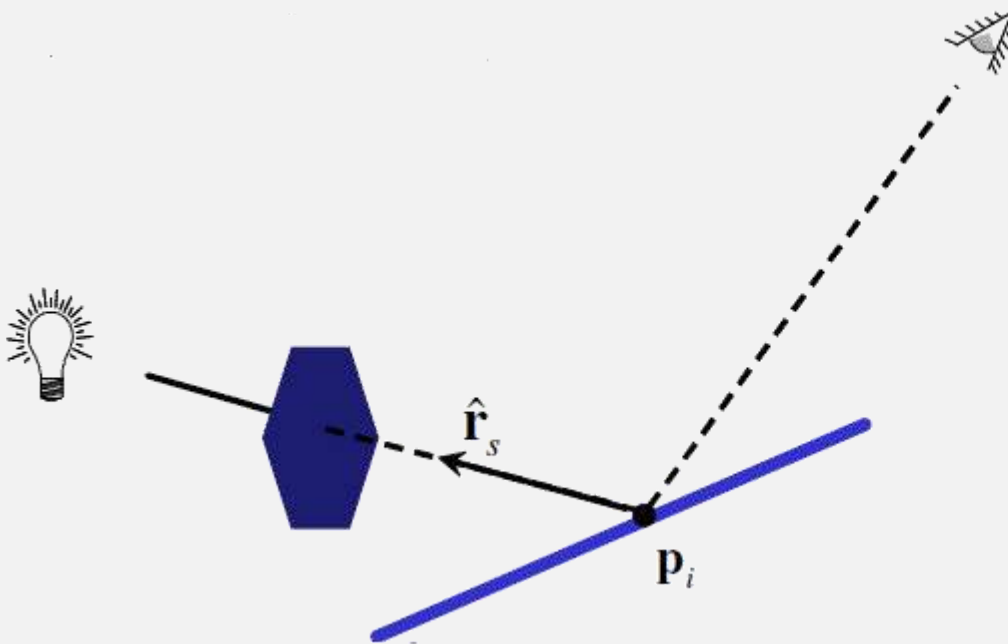
$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} + \sum_{\text{luzes}} \left(\begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} (\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}) + \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{sr} \\ k_{sg} \\ k_{sb} \end{pmatrix} (\hat{\mathbf{r}} \cdot \hat{\mathbf{v}})^n \right)$$

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Modelo de iluminação

- ▶ Luz e sombra



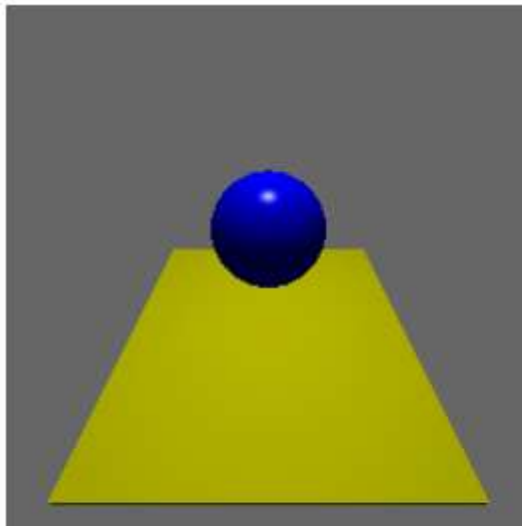
Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

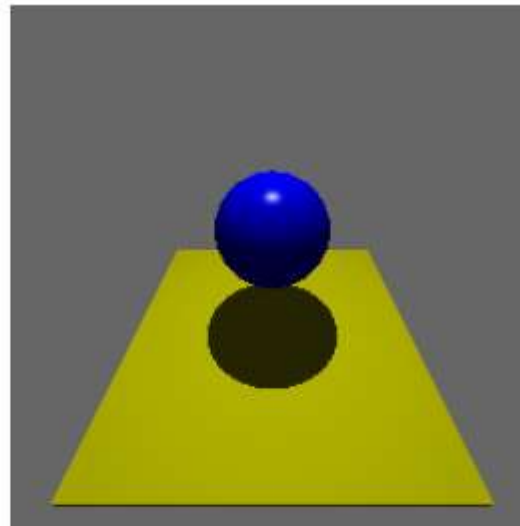
Modelo de iluminação

superfícies refletoras e objetos transparentes

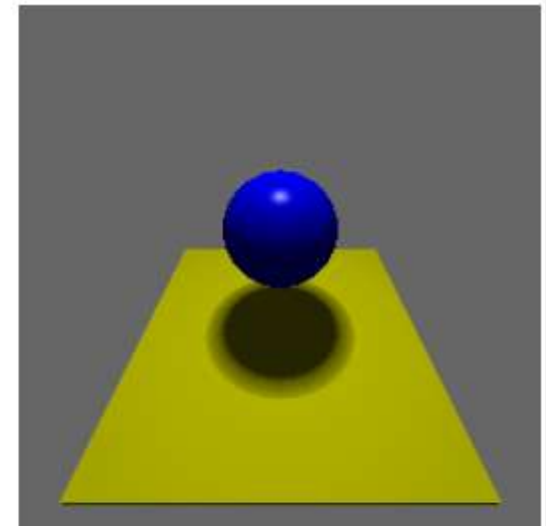
- ▶ Luz e sombra
 - ▶ Importância artística e de realismo visual
 - ▶ Incrementar a nossa percepção da posição relativa



(a) sem sombra



(b) sombra de luz pontual

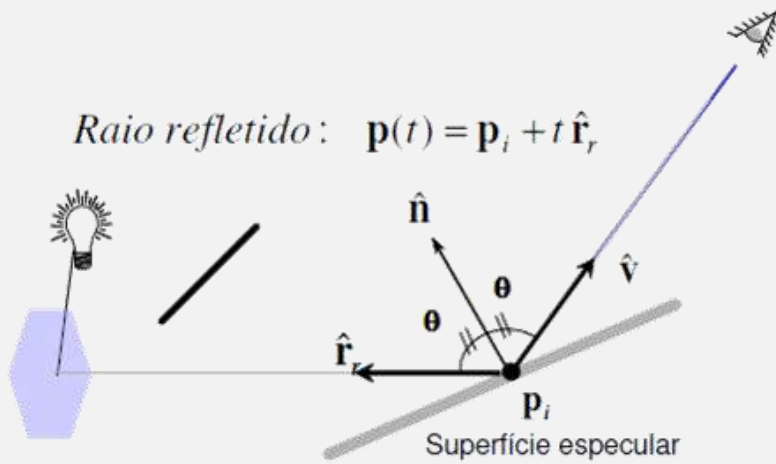


(c) sombras suaves

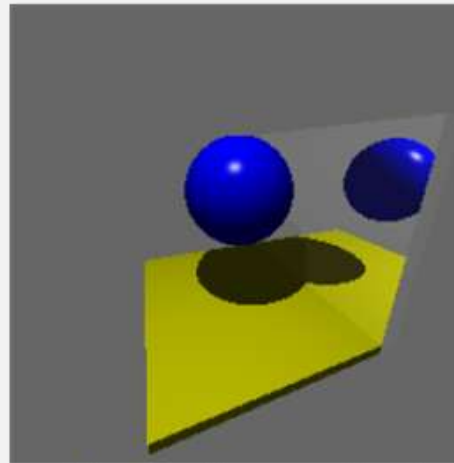
Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

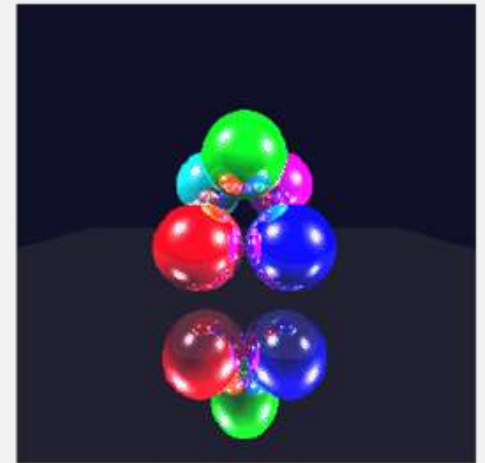
Superfícies refletoras e objetos transparentes



(a) raio refletido



(b) espelho vertical

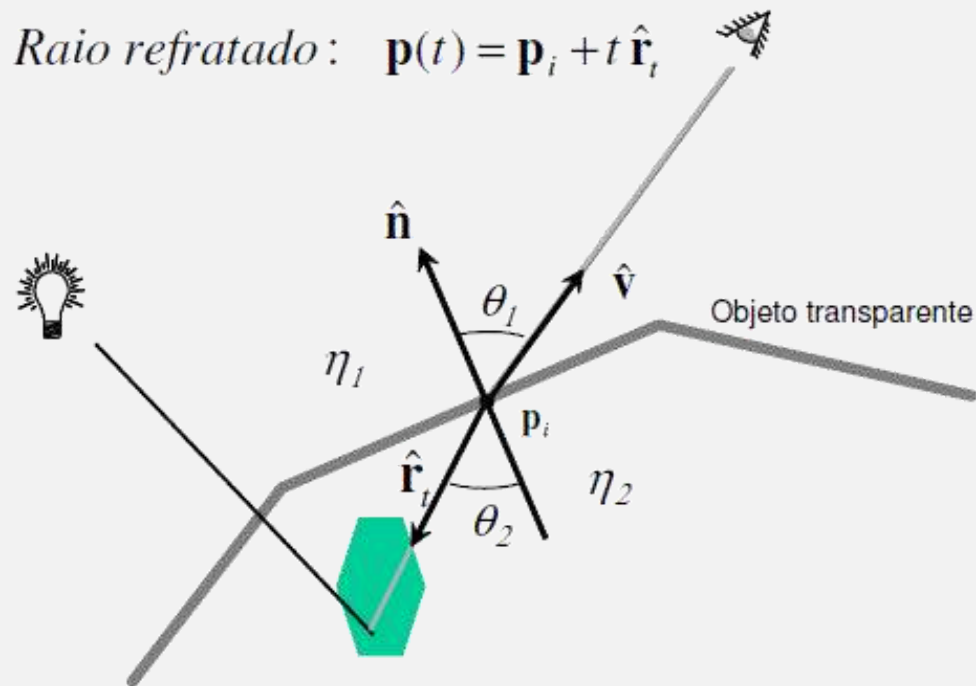


(b) 5 bolas

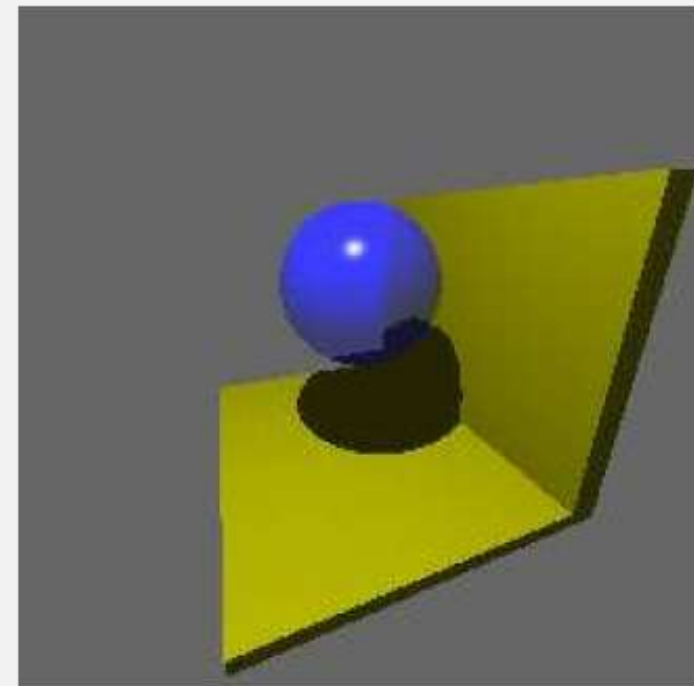
Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Superfícies refletoras e objetos transparentes



(a) Raio refratado



(b) Esfera transparente

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Natureza recursiva

```
for (cada pixel da tela)
{
    determine o raio ray correspondente ao pixel em questão;
    pixel = trace ( ray, 1 );
}
```

```
Color trace (Scene scene, Vector eye, Vector ray, int depth)
{
    determine a interseção mais próxima com um objeto
    if (intercepta objeto)
    {
        calcule a normal no ponto de interseção
        return ( shade ( scene, object, ray, point, normal, depth ));
    }
    return BACKGROUND;
}
```

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Natureza recursiva

```
Color shade (Scene scene, Object object, Vector ray,
             Vector point, Vector normal, int depth)
{
    color = cor ambiente sobre a cor difusa do material do objeto;

    for (cada luz) {
        L = vetor unitário na direção de point para a posição da luz;
        if (L•normal>0) {
            if (a luz não for bloqueada no ponto) {
                color += componente difusa + componente especular
            }
        }
    }

    if (depth >= maxDepth) return color;

    if (objeto é refletor) {
        rRay = raio na direção de reflexão;
```

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

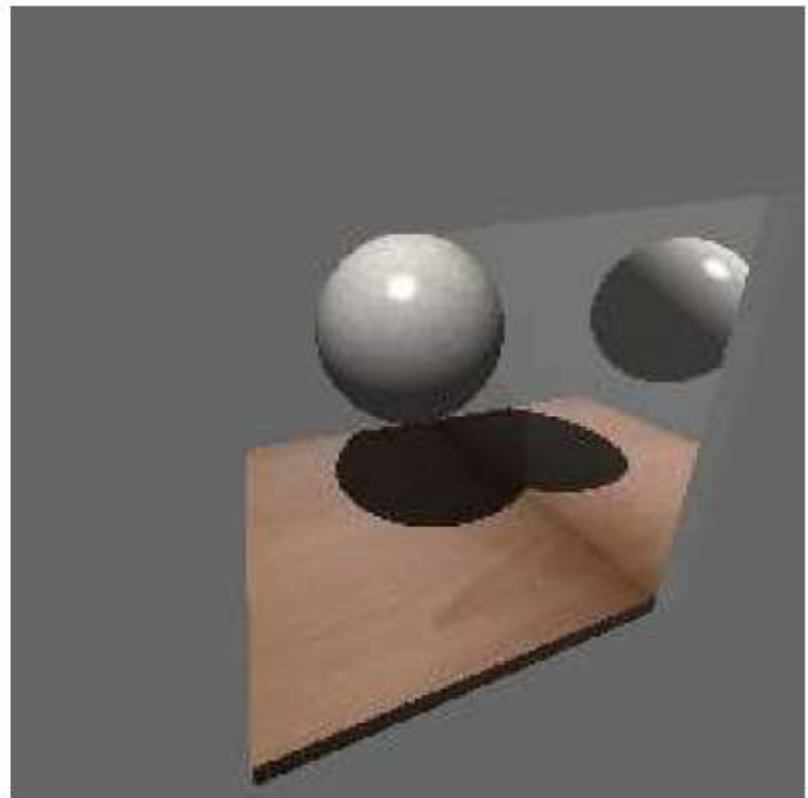
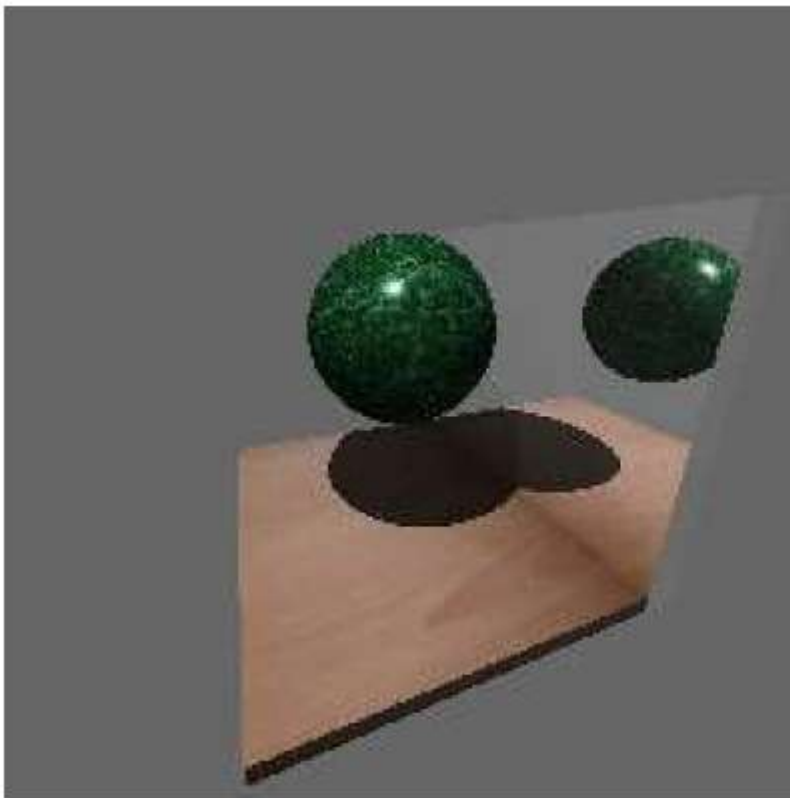
Natureza recursiva

```
    rColor = trace(scene, point, rRay, depth+1);  
    color += k* rColor;  
}  
  
if (objeto é transparente) {  
    tRay = raio na direção de refração;  
    tColor = trace(scene, point, tRay, depth+1);  
    color += (1-o)*tColor;  
}  
  
return color;  
}
```

Geração de Imagens Realísticas

Rastreamento de Raios (Ray Tracing)

Textura



Referencias

- ▶ Gattass, Marcelo Material de aula da disciplina de Computação Gráfica, PUC-RIO. 2005
- ▶ Livro: Teoria da computação Gráfica, Editora Campus Ltda,RJ 2003
- ▶ Gonzales R. C & Woods R. E. Processamento de imagens digitais. S. Paulo: Editora Edgard Blucher, 1ª edição 2000.
- ▶ Tomas Akenine-Moller & Eric Haines Real-Time Rendering: A K Peters Ltd. USA., second edition, 2002.
- ▶ Montenegro, Anselmo Material da aula da disciplina de Computação Grafica
<http://www.ic.uff.br/~anselmo/cursos/CGI/CGI20112/CGI20112.html>