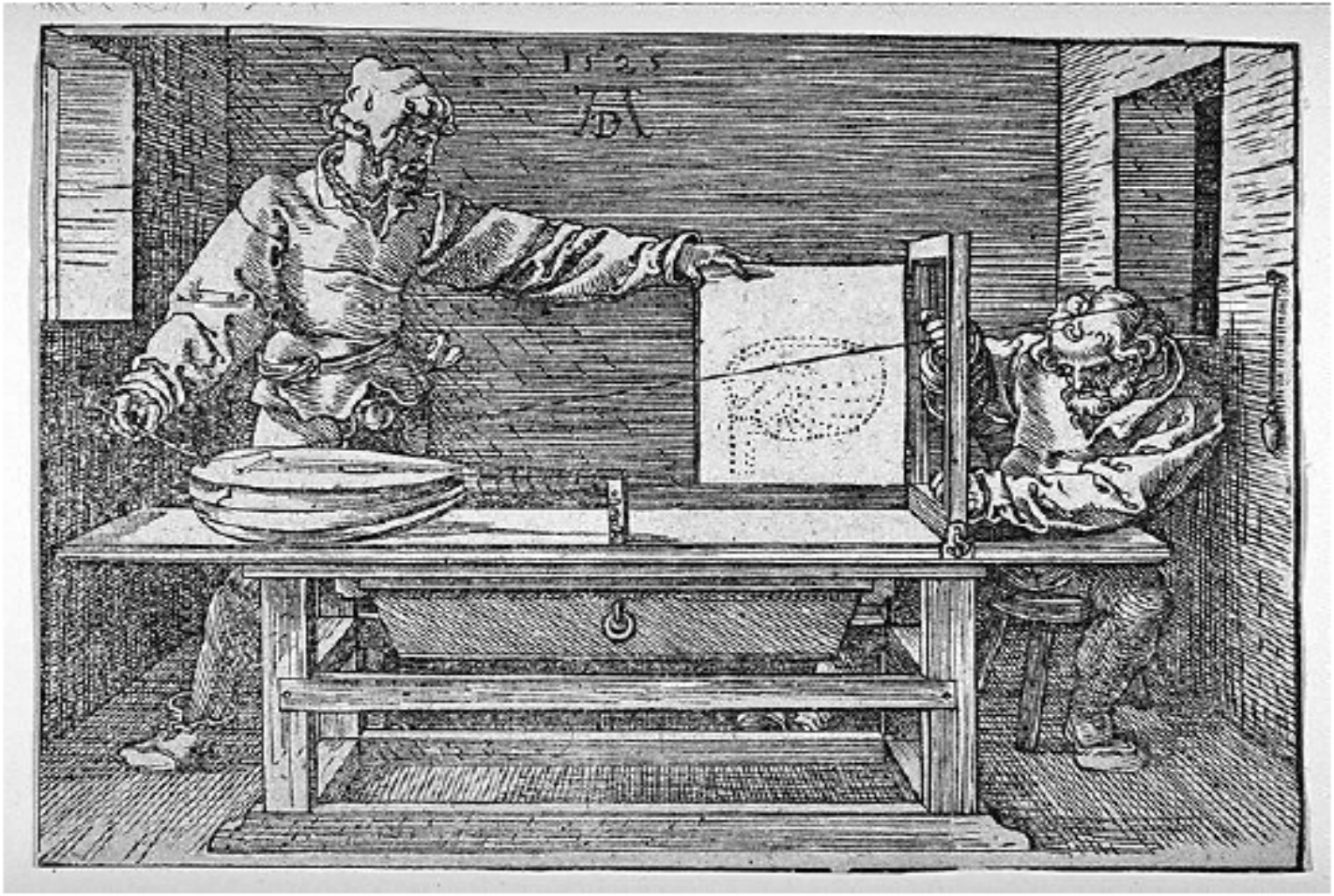
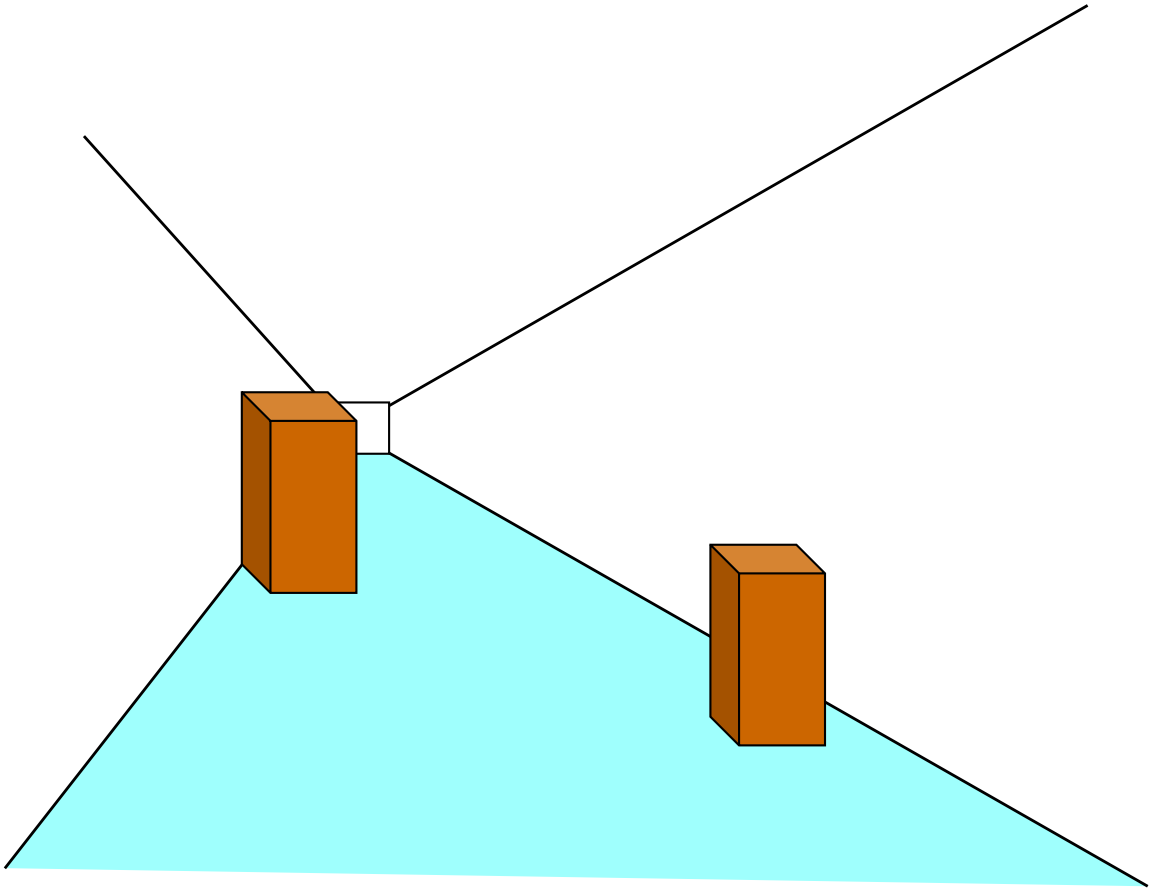
Algoritmo de Rastreamento de Raios

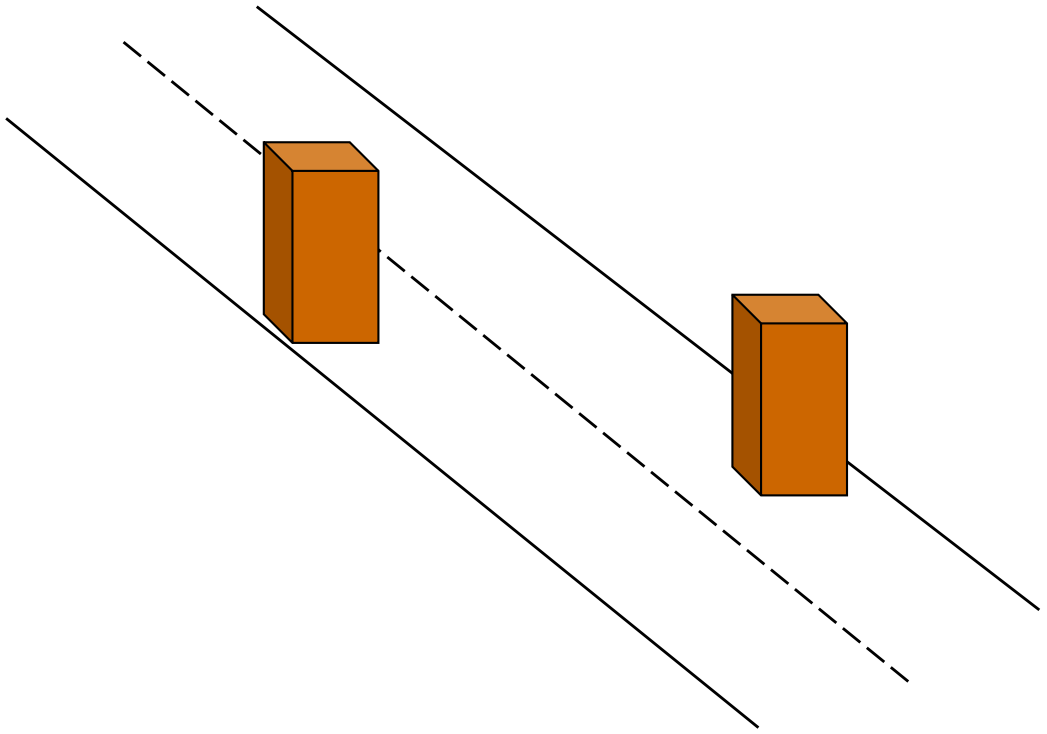


Man Drawing a Lute (The Draughtsman of the Lute), woodcut 1525, Albrecht Dürer.

Perspectiva e tamanhos relativos

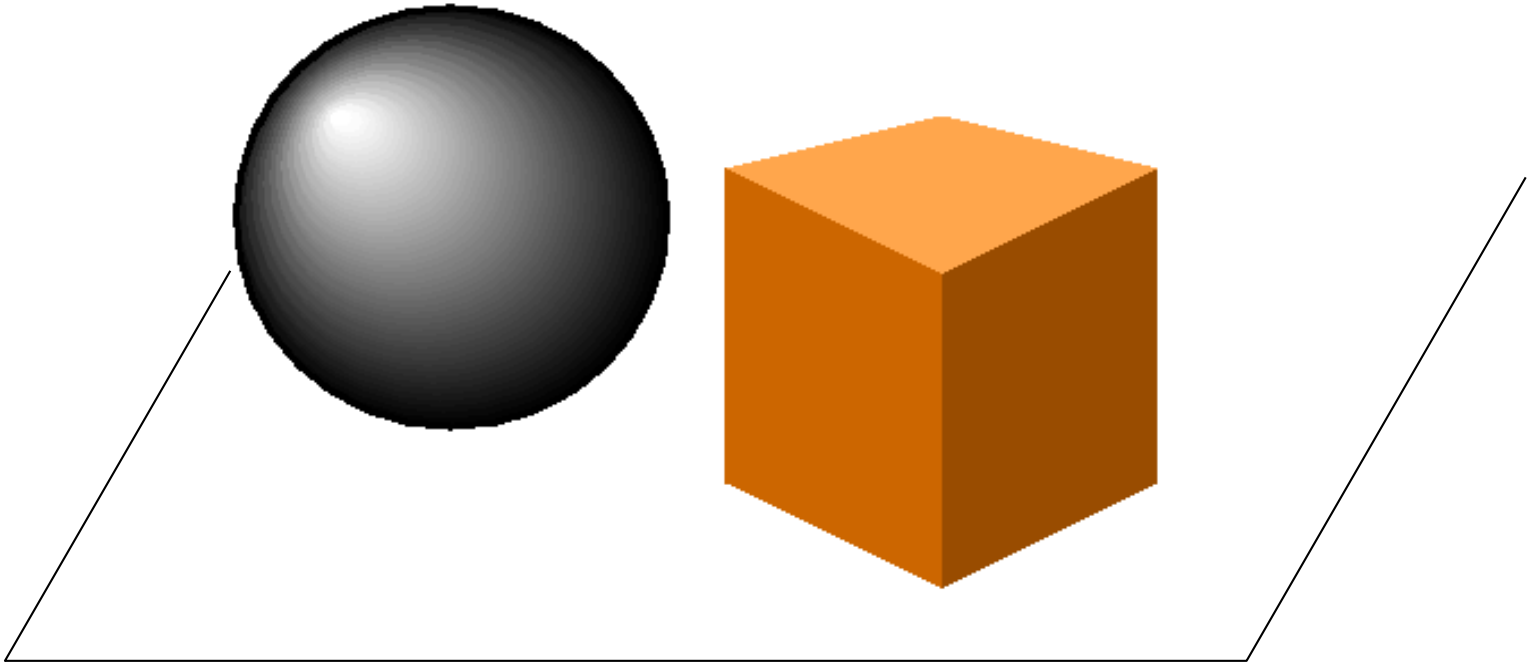


Perspectiva e tamanhos relativos

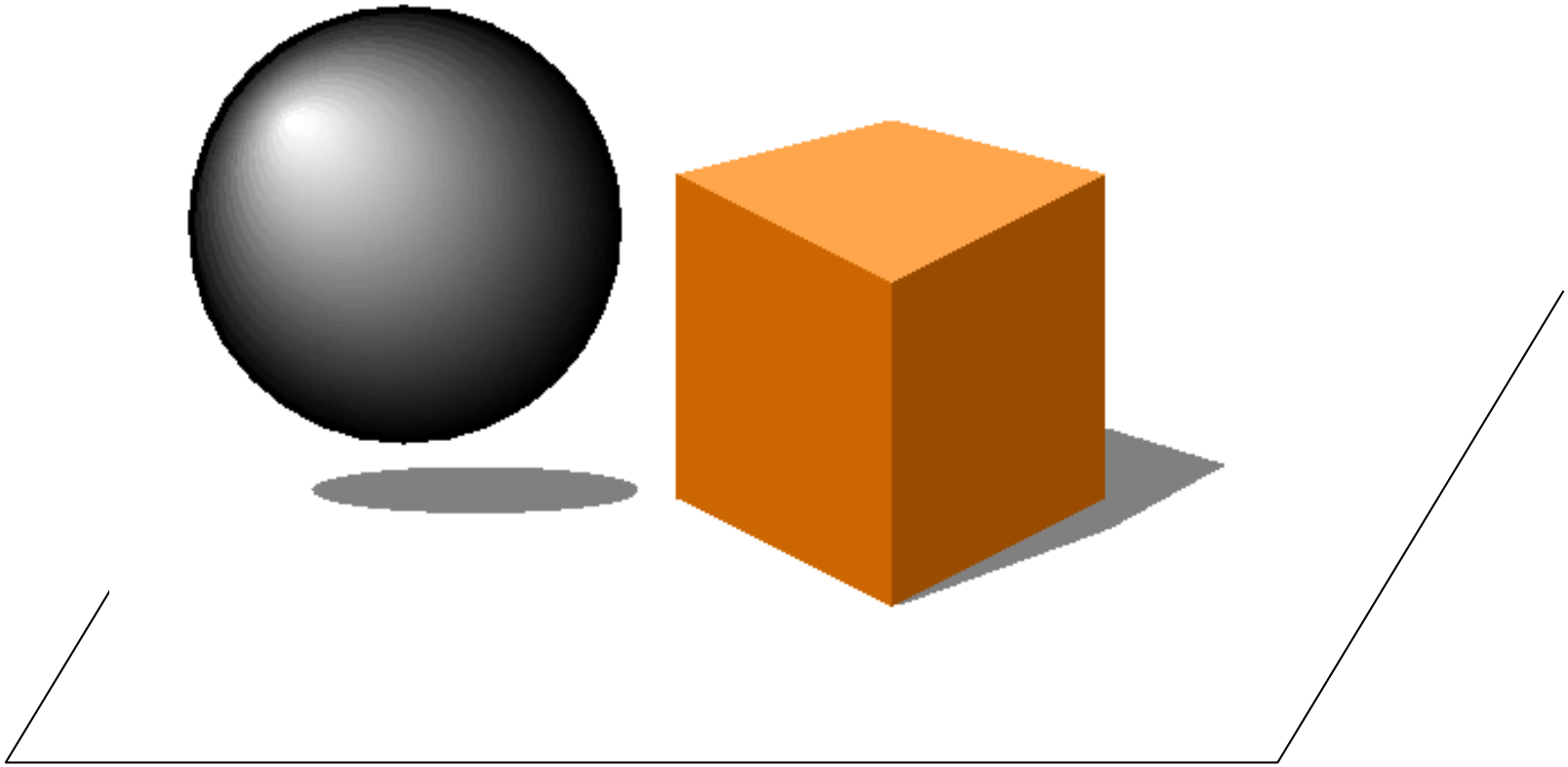


Iluminação e posição

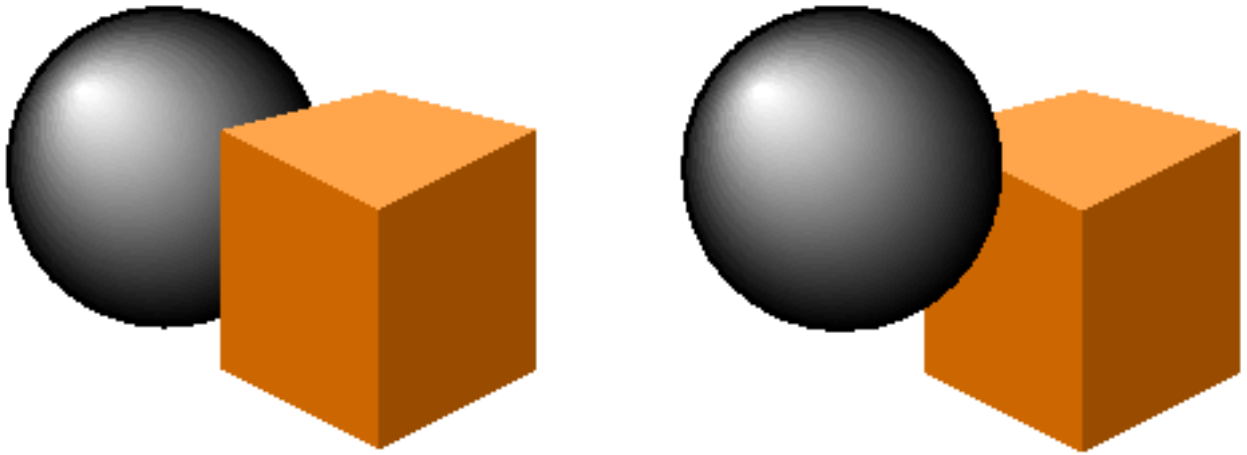
O que está na frente?
A que distância do apoio?



Sombra



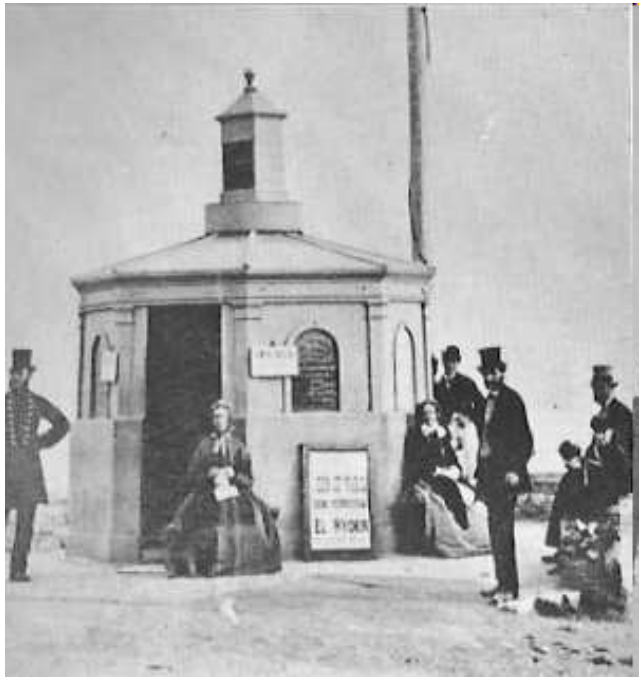
Oclusão



Efeitos Passivos

- Inerentes a aparência do mundo externo
- Independem dos nossos olhos
- Fotografias parecem 3D

Camera obscura

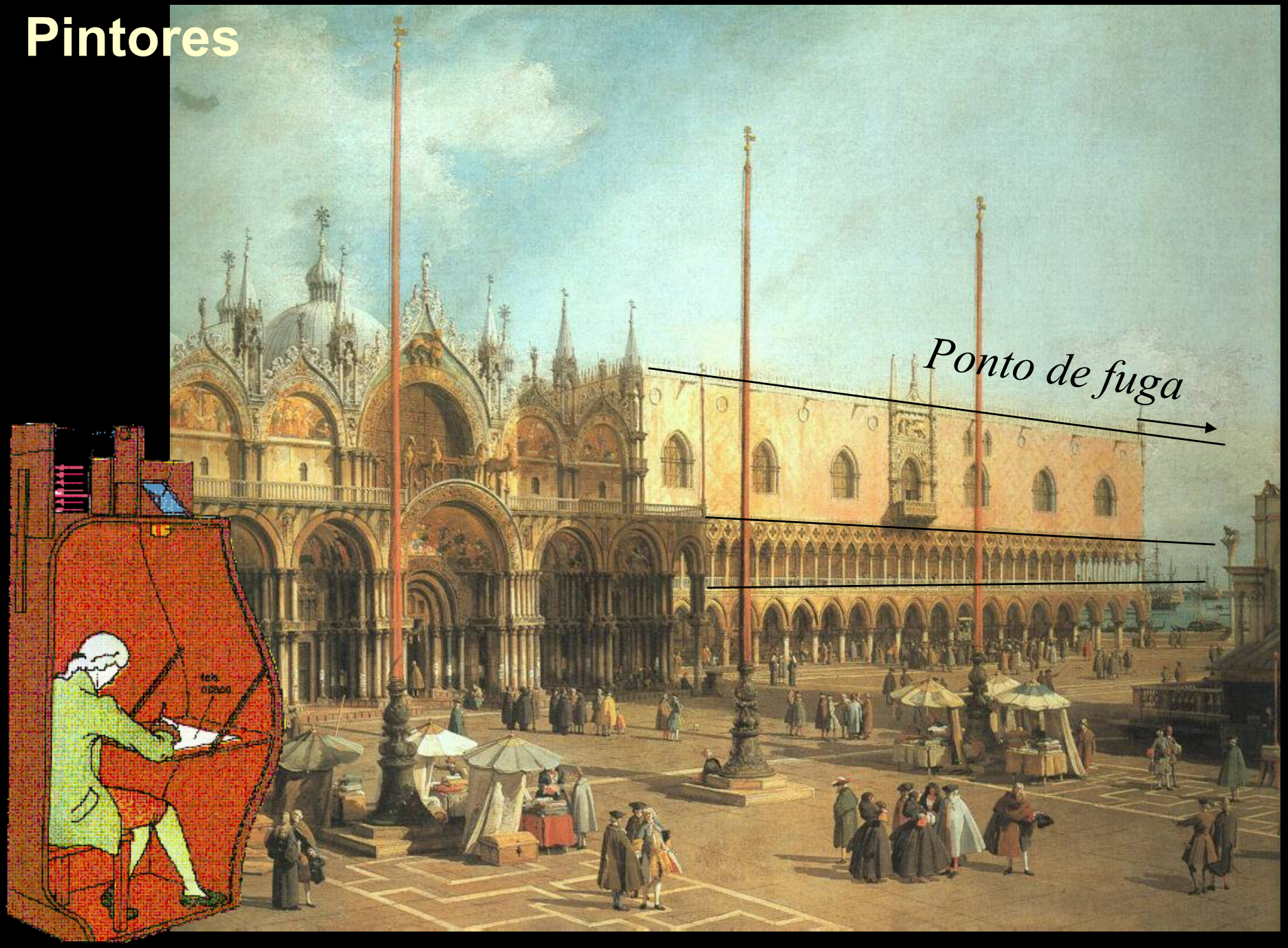


Plymouth, UK

Camera Obscura - San Francisco

The Camera Obscura at the Cliff House is one of several remaining camera obscuras in the world. The device is an ancient precursor to modern photography, and well worth a visit, especially if you haven't previously visited a camera obscura.

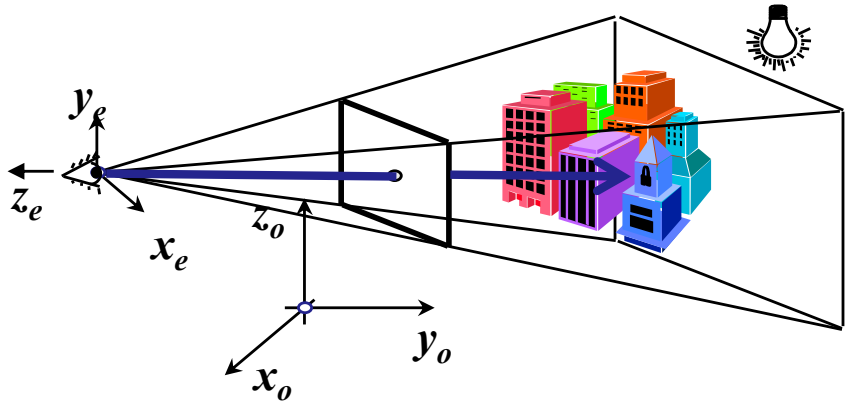
Pintores



Canaletto (Giovanni Antonio Canal) (1697-1768).

Dois enfoques: Raios

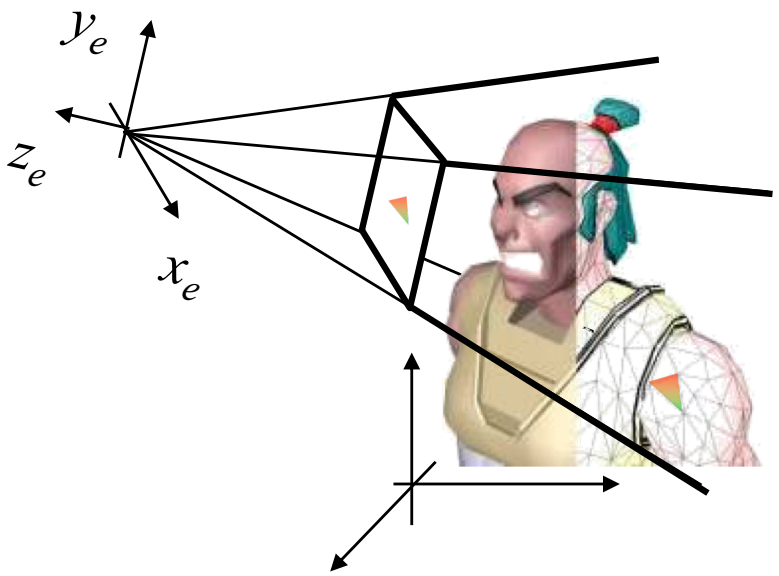
Traçado de Raios

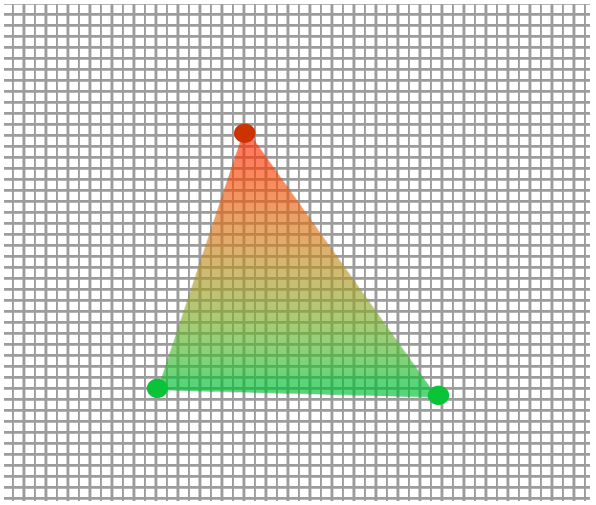


Para cada pixel da tela
Defina um raio
Para cada objeto da cena
Calcule o objeto visível
Para cada luz da cena
Lance um raio
Para cada objeto da cena
Teste se o objeto faz sombra
Calcule sua iluminação

Complexidade maior que $\mathbf{O}(num.\ de\ pixels \times num.\ de\ objetos^2)$

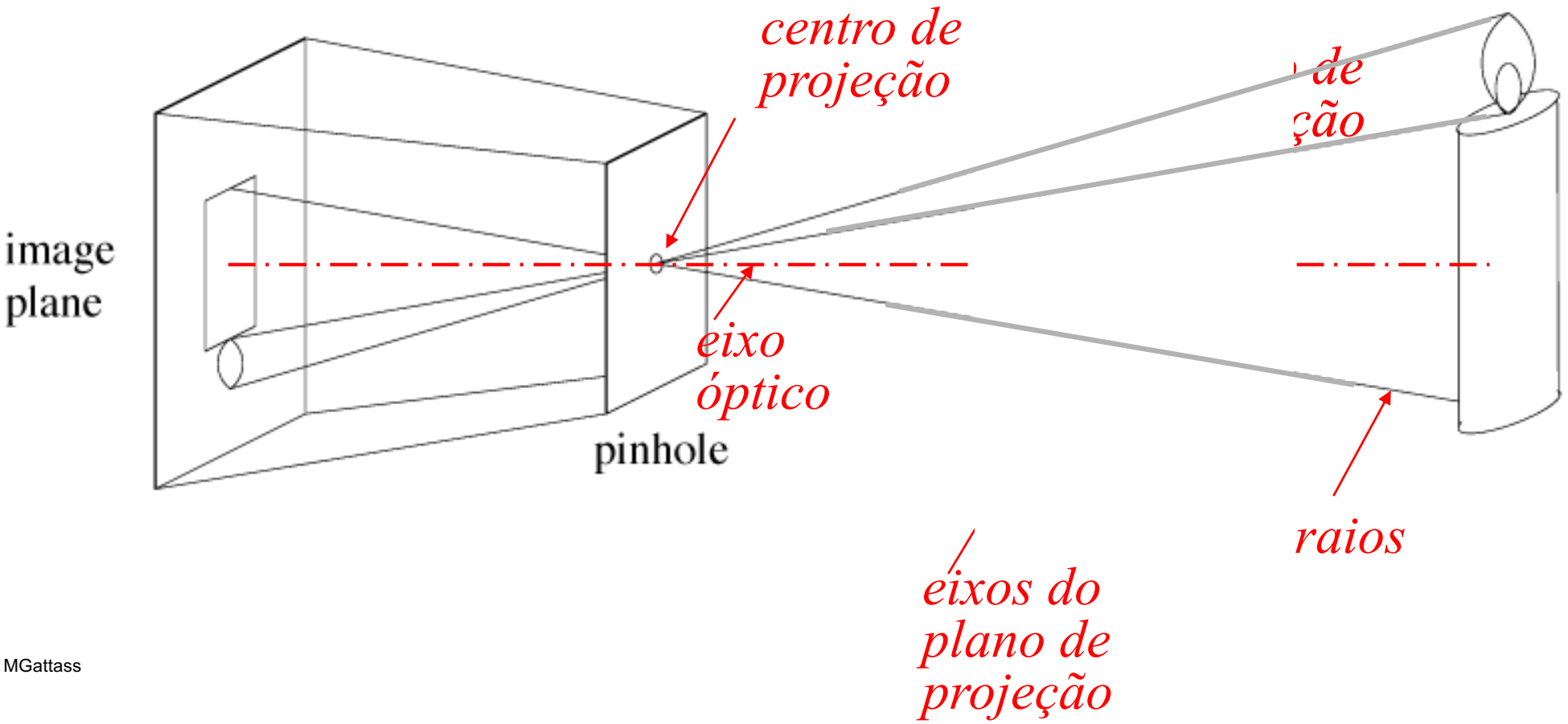
Dois enfoques: ZBuffer



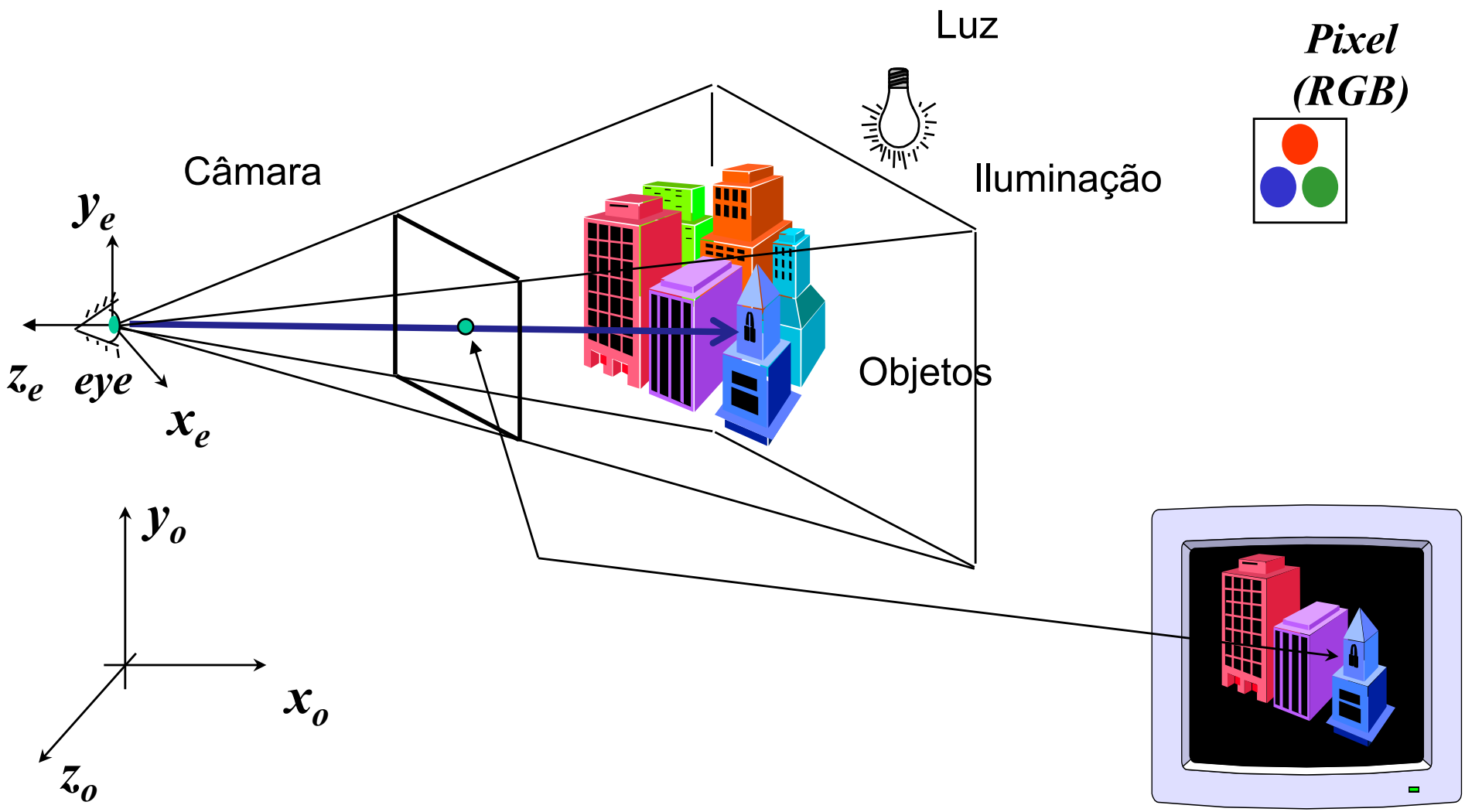


Calcule a cor de cada vertice
Projete cada triângulo
Projete os vértices no plano de projeção
Rasterize o triângulo gerandoo os fragmentos que vão para cada pixel
Aqueles fragmentos que estiverem a frente dos framentos já depositados recebem nova cor e a profundidade do pixel é atualizada.

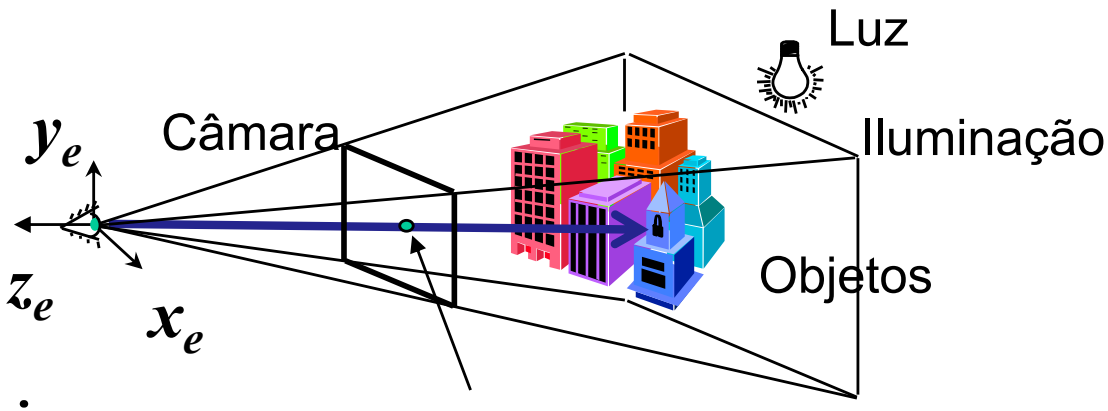
A Câmara “Pinhole” e seu modelo



Traçado de Raios

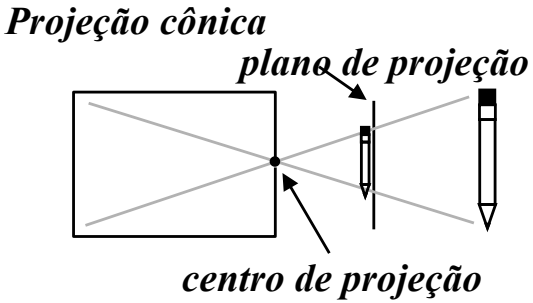


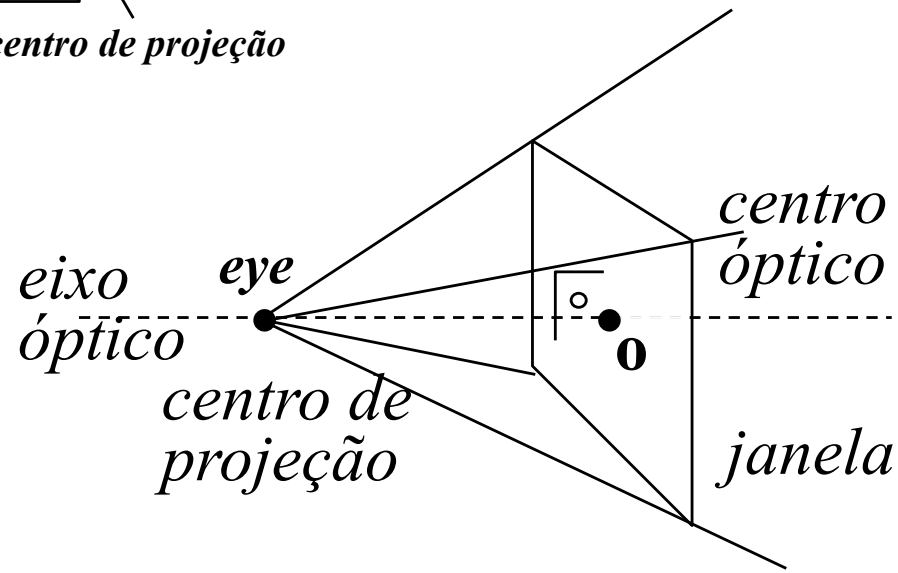
Algoritmo básico

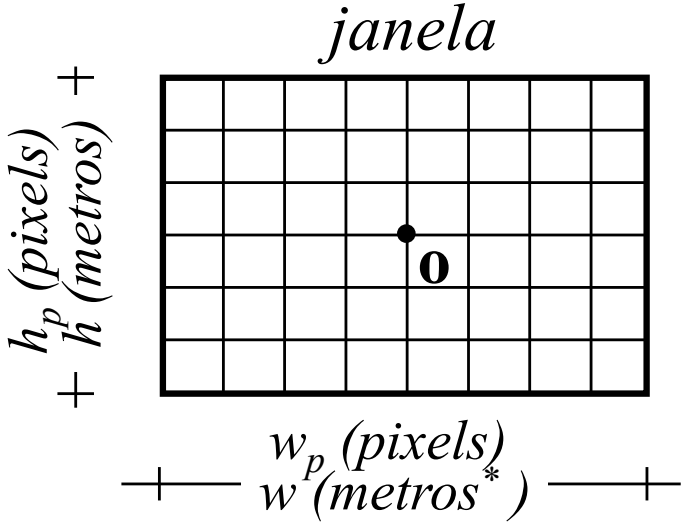



```
Para cada pixel da tela;
   Lance uma raio;
   Para cada objeto da cena
       Calcule a interseção do raio com este o objeto;
       Armazene a interseção mais próxima;

   Se o raio interceptou algum objeto
       Calcule a contribuição das luzes neste ponto;
       Pinte o pixel com esta cor;
```

Definição de uma câmera



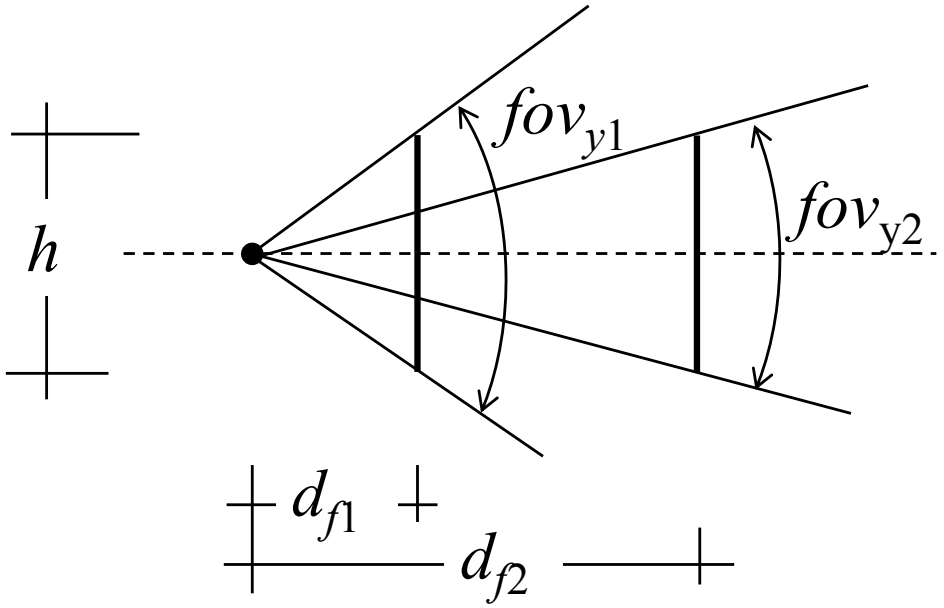




$$w = \frac{w_p}{h_p} h$$

(*) *metros ou outra unidade de comprimento*

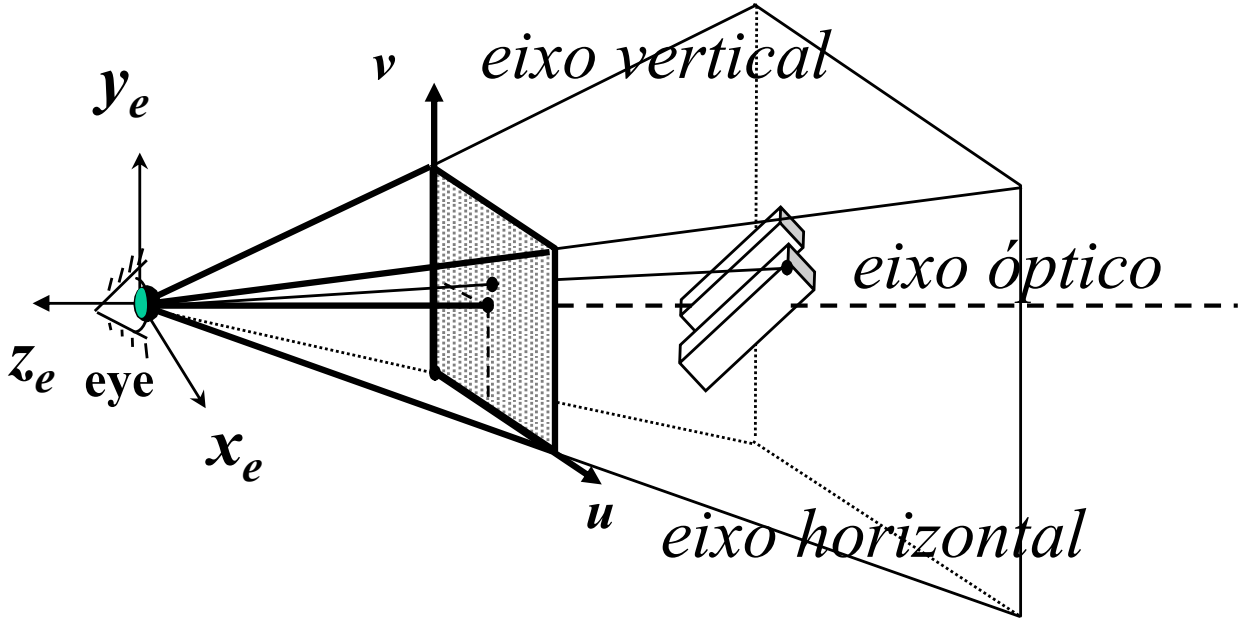
Abertura de uma câmera

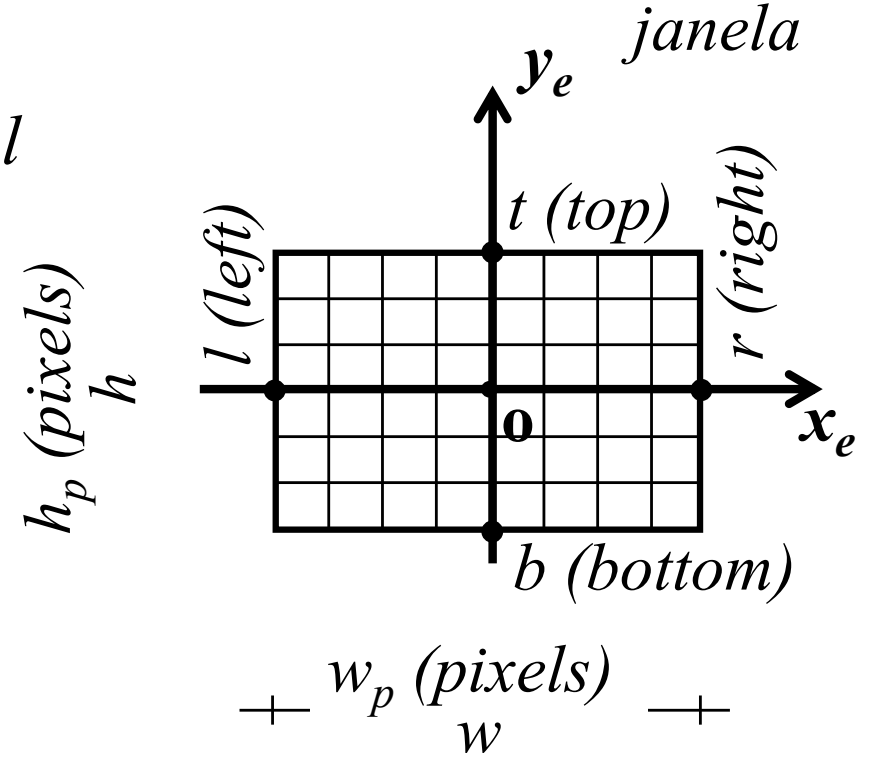


$$\frac{h}{2d_f} = \tan\left(\frac{fov_y}{2}\right)$$

$$h = 2d_f \tan\left(\frac{fov_y}{2}\right)$$

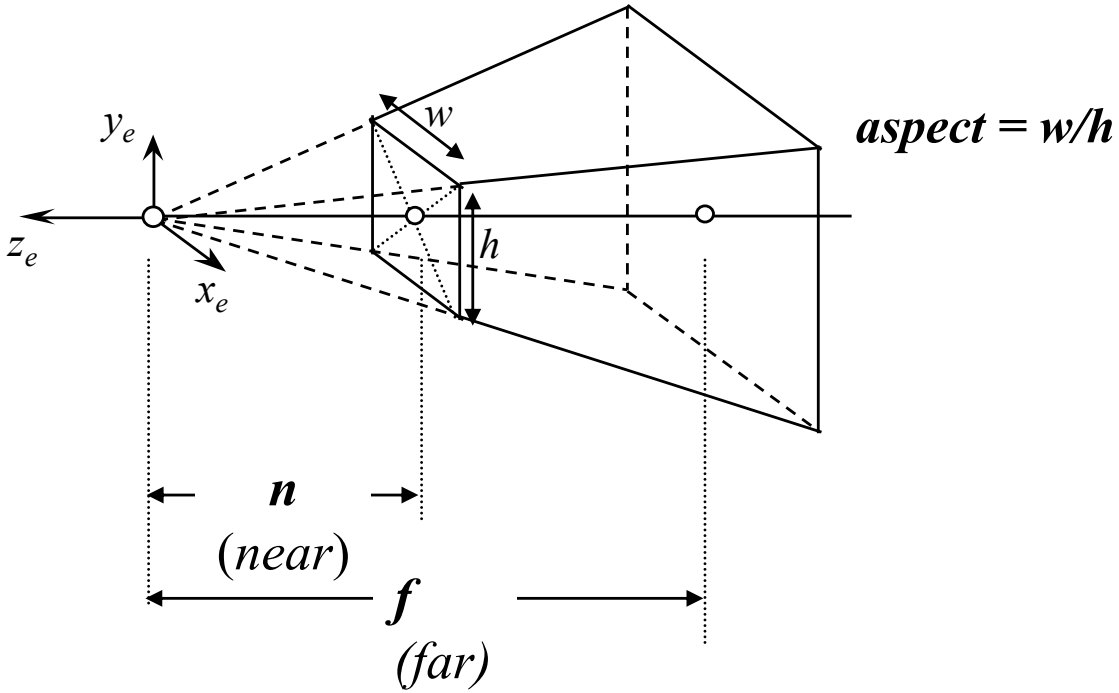
Eixos de uma câmera

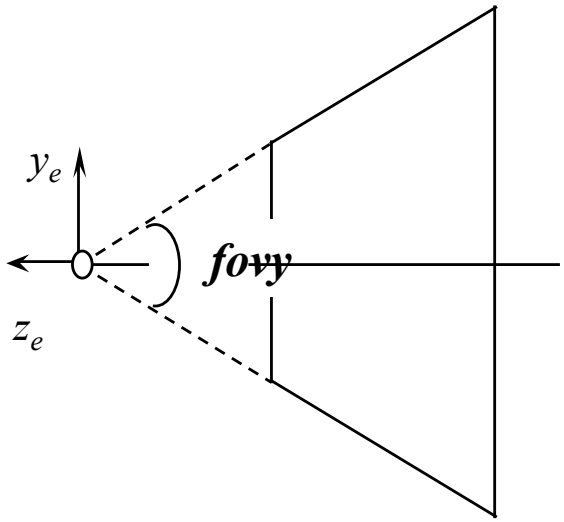




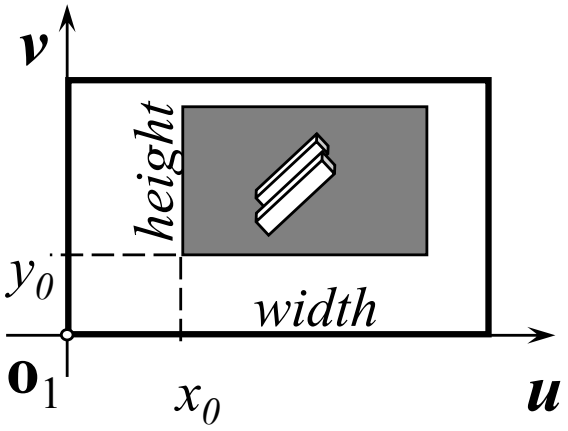
Parâmetros de funções do OpenGL

```
void glPerspective(GLdouble fovy,GLdouble aspect,GLdouble near_,GLdouble far_);
```



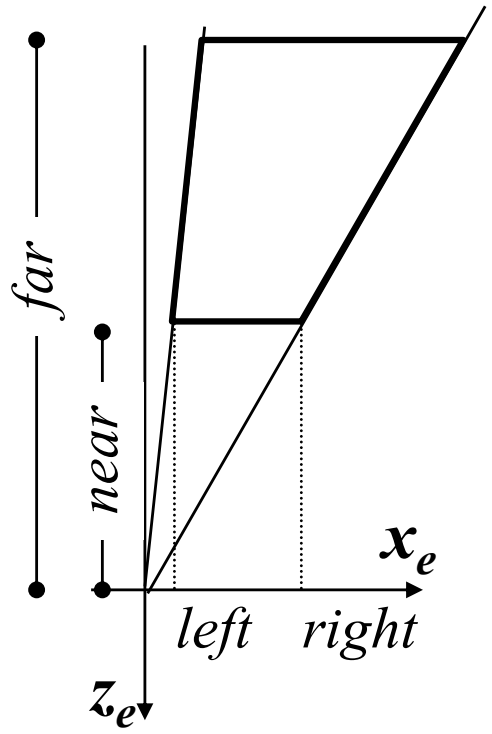


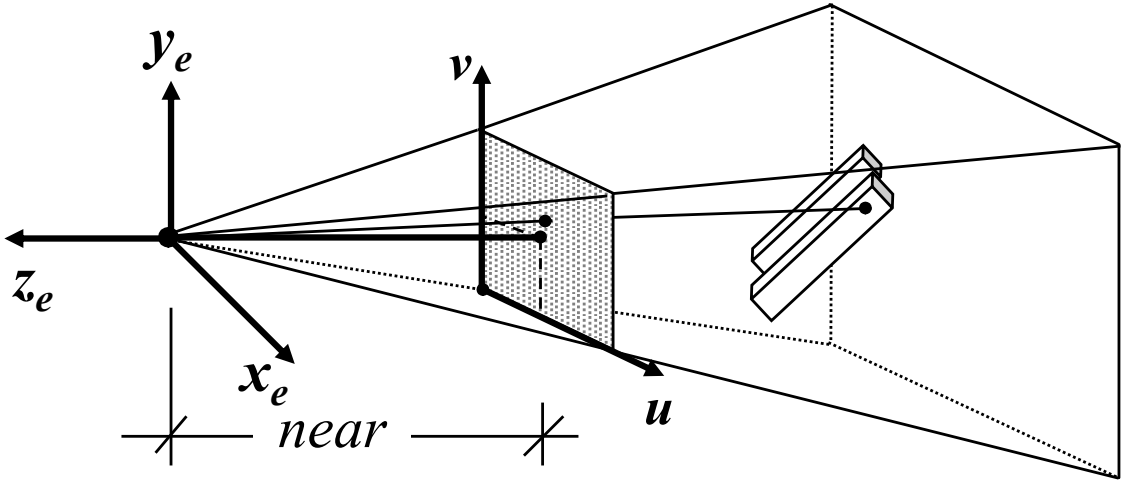

```
void glViewport(GLint x0, GLint y0,
                GLsizei width, GLsizei height );
```



Parâmetros intrínsecos (do OpenGL) parte 2 – Câmera mais geral

```
void glFrustum(  GLdouble left, GLdouble right, GLdouble bottom, GLdouble top,
                 GLdouble near_, GLdouble far_ );
```





Parâmetros internos ou intrínsecos

Primários:

n = *plano próximo*

f = *plano distante*

$w_p \times h_p$ *pixels*

fov_y = *campo de visão*

Derivados:

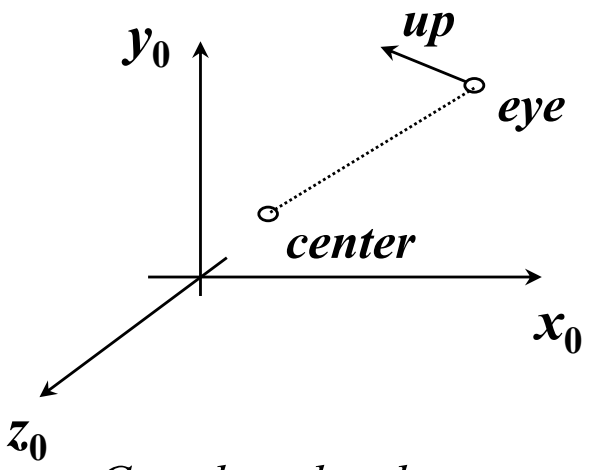
$$d_f = n$$

$$h = 2d_f \tan\left(\frac{fov_y}{2}\right)$$

$$w = \frac{w_p}{h_p} h$$

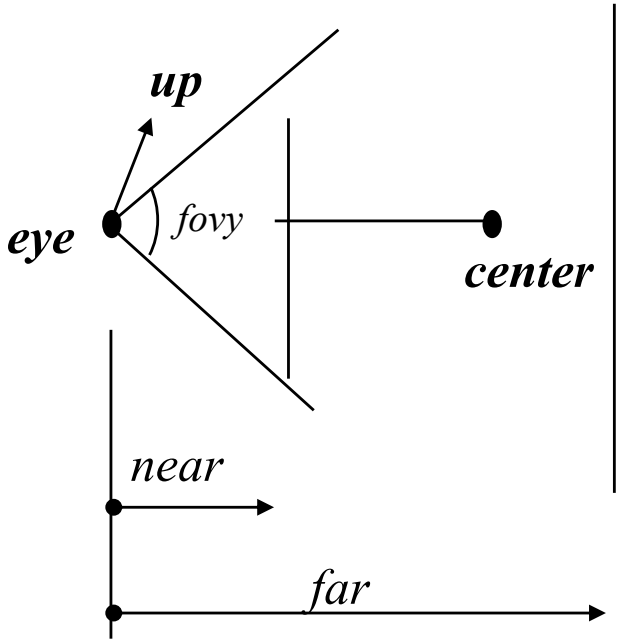
Posicionamento da câmera (parâmetros externos ou extrínsecos)

eye	=	*centro óptico*
center	=	*ponto de visada,*
up	=	*direção para cima*

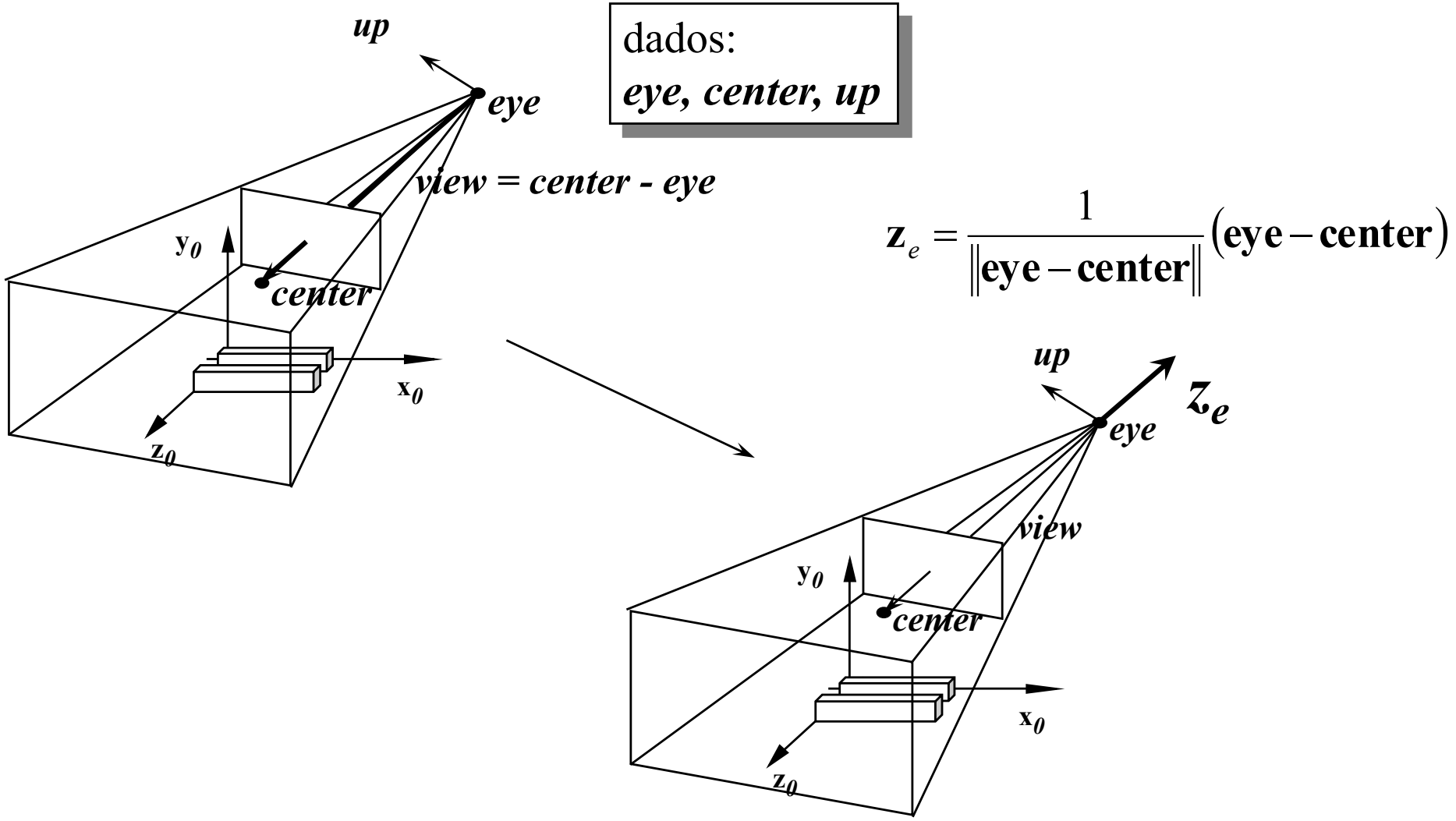


Coordenadas dos Objetos

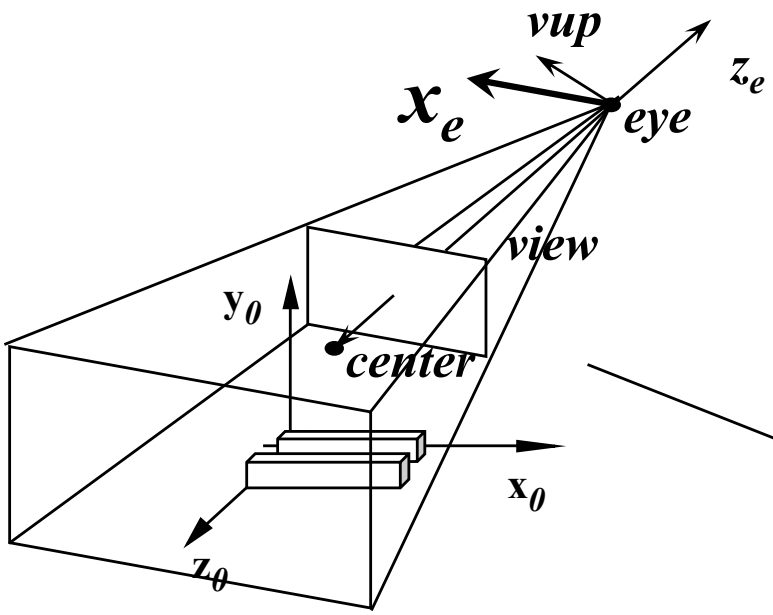
vista lateral



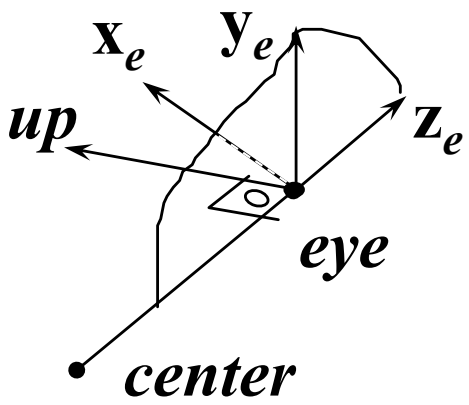
Calculo do sistema do olho - $x_e\, y_e\, z_e$

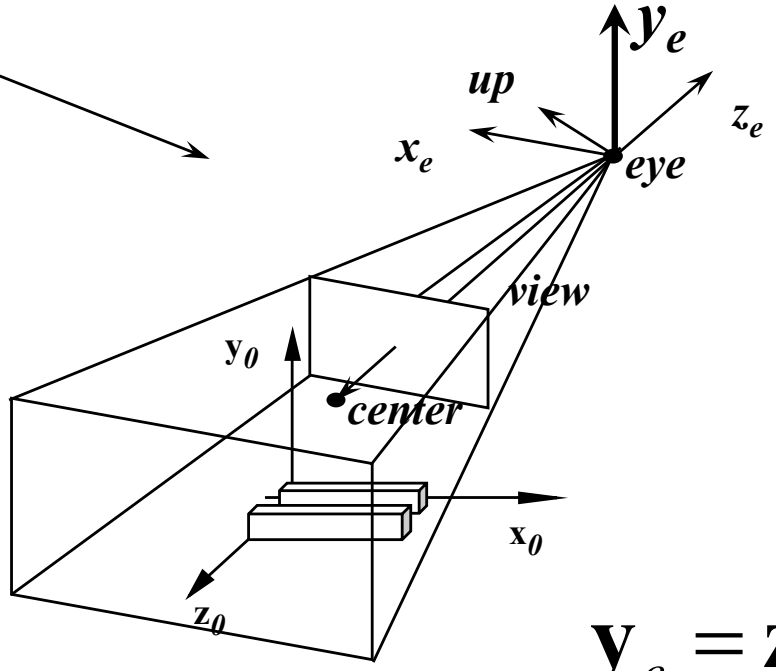


Calculo do sistema do olho - $x_e\, y_e\, z_e$



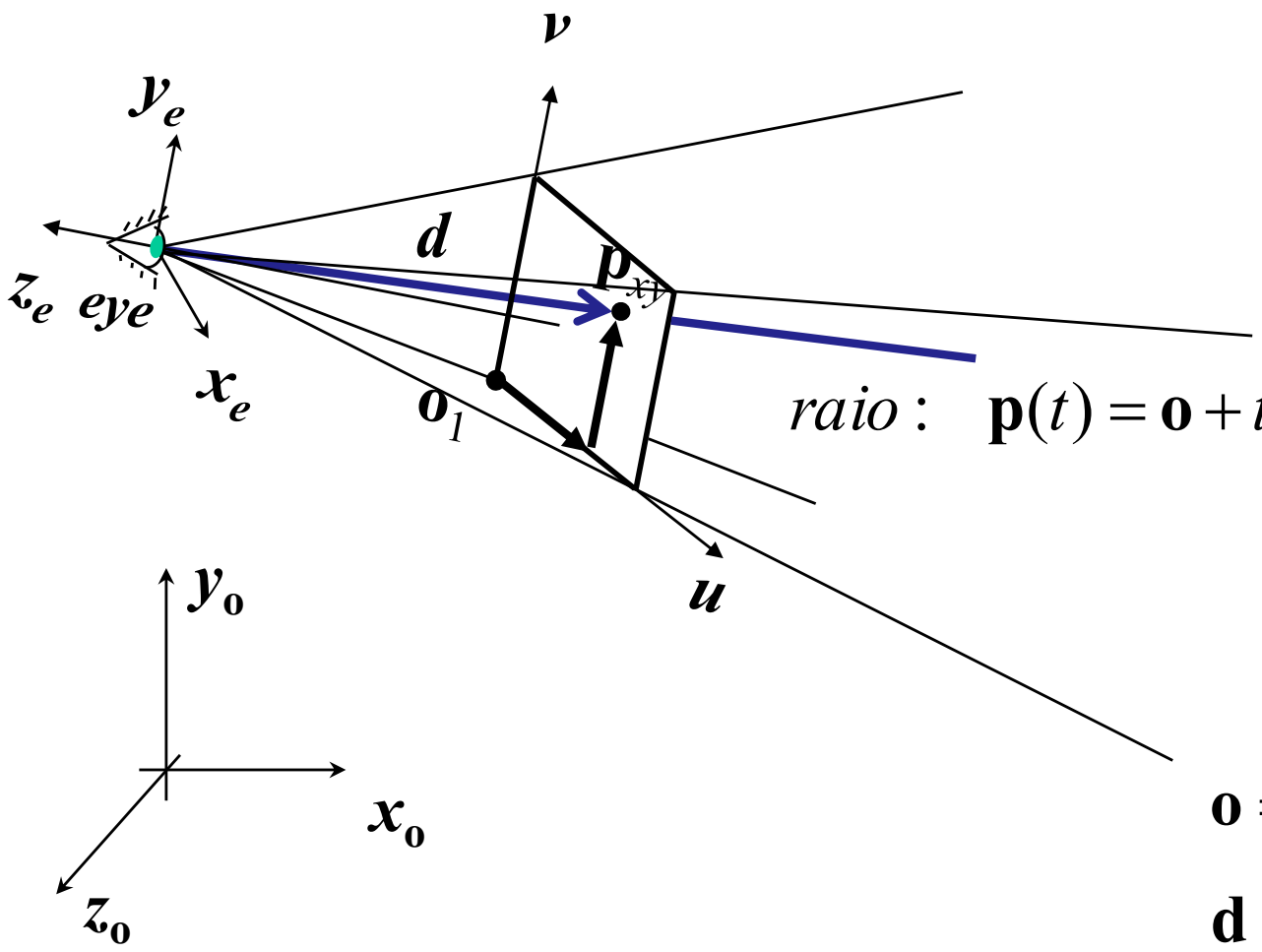
$$\mathbf{x}_e = \frac{1}{\|\mathbf{up} \times \mathbf{z}_e\|}(\mathbf{up} \times \mathbf{z}_e)$$





$$\mathbf{y}_e = \mathbf{z}_e \times \mathbf{x}_e$$

Um modelo de câmera

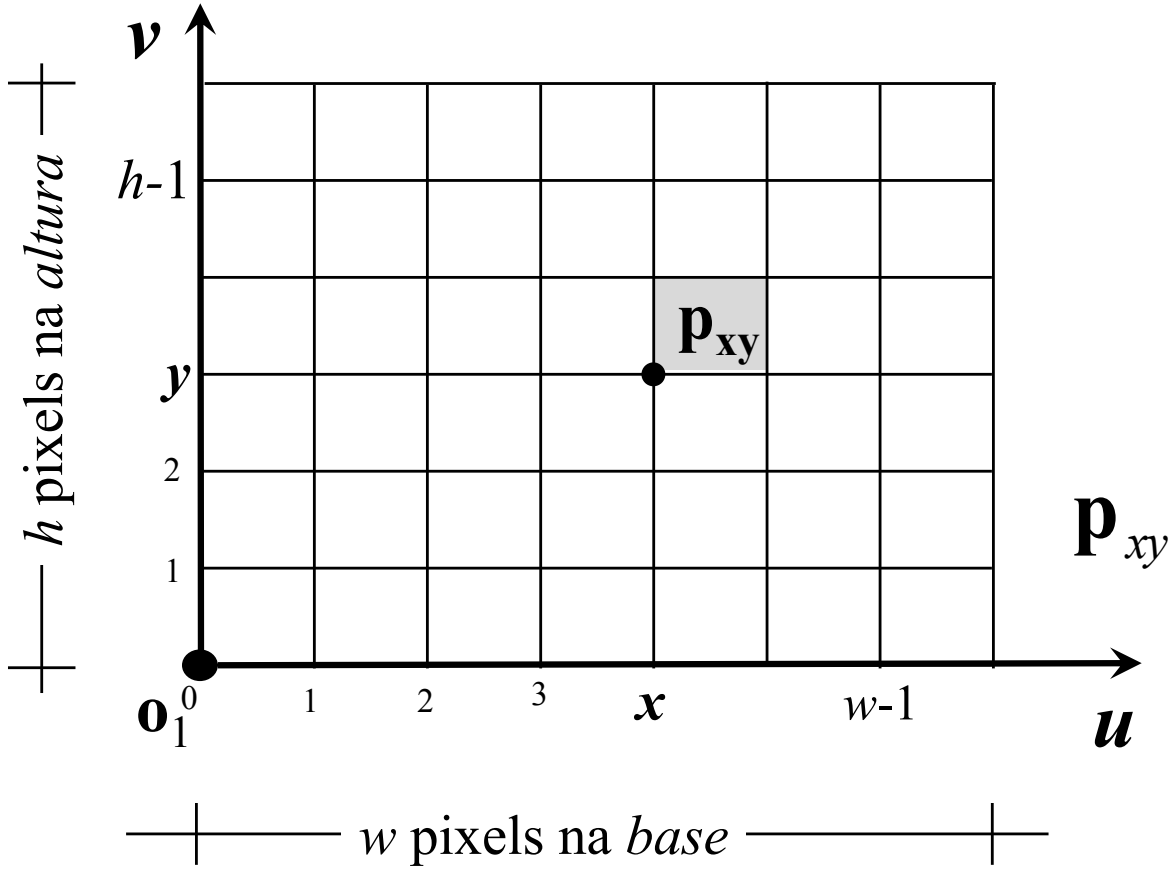


$$raio: \quad \mathbf{p}(t) = \mathbf{o} + t\mathbf{d}$$

$$\mathbf{o} = \mathbf{eye}$$

$$\mathbf{d} = \mathbf{p}_{xy} - \mathbf{eye}$$

Lançamento de Raios



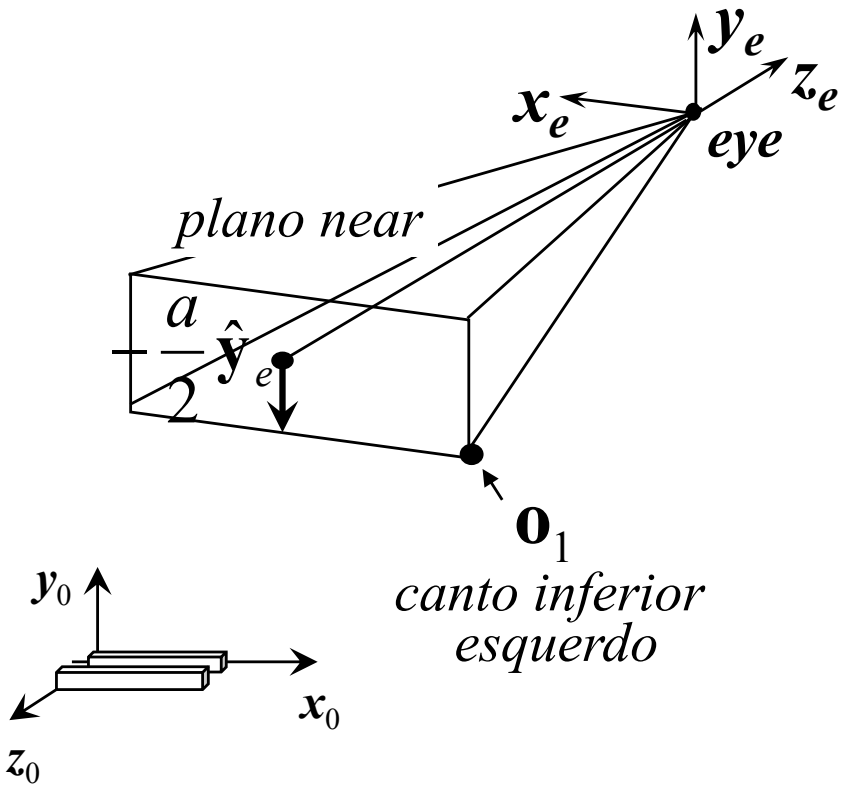
$$u(x) = \frac{x}{w_p} w$$

$$v(y) = \frac{y}{h_p} h$$

$$\mathbf{p}_{xy} = \mathbf{o}_1 + u(x)\hat{\mathbf{u}} + v(y)\hat{\mathbf{v}}$$

$$\mathbf{p}_{xy} = \mathbf{o}_1 + w\frac{x}{w_p}\hat{\mathbf{x}}_e + h\frac{y}{h_p}\hat{\mathbf{y}}_e$$

Canto inferior esquerdo da janela no plano *near* (ou *far*)

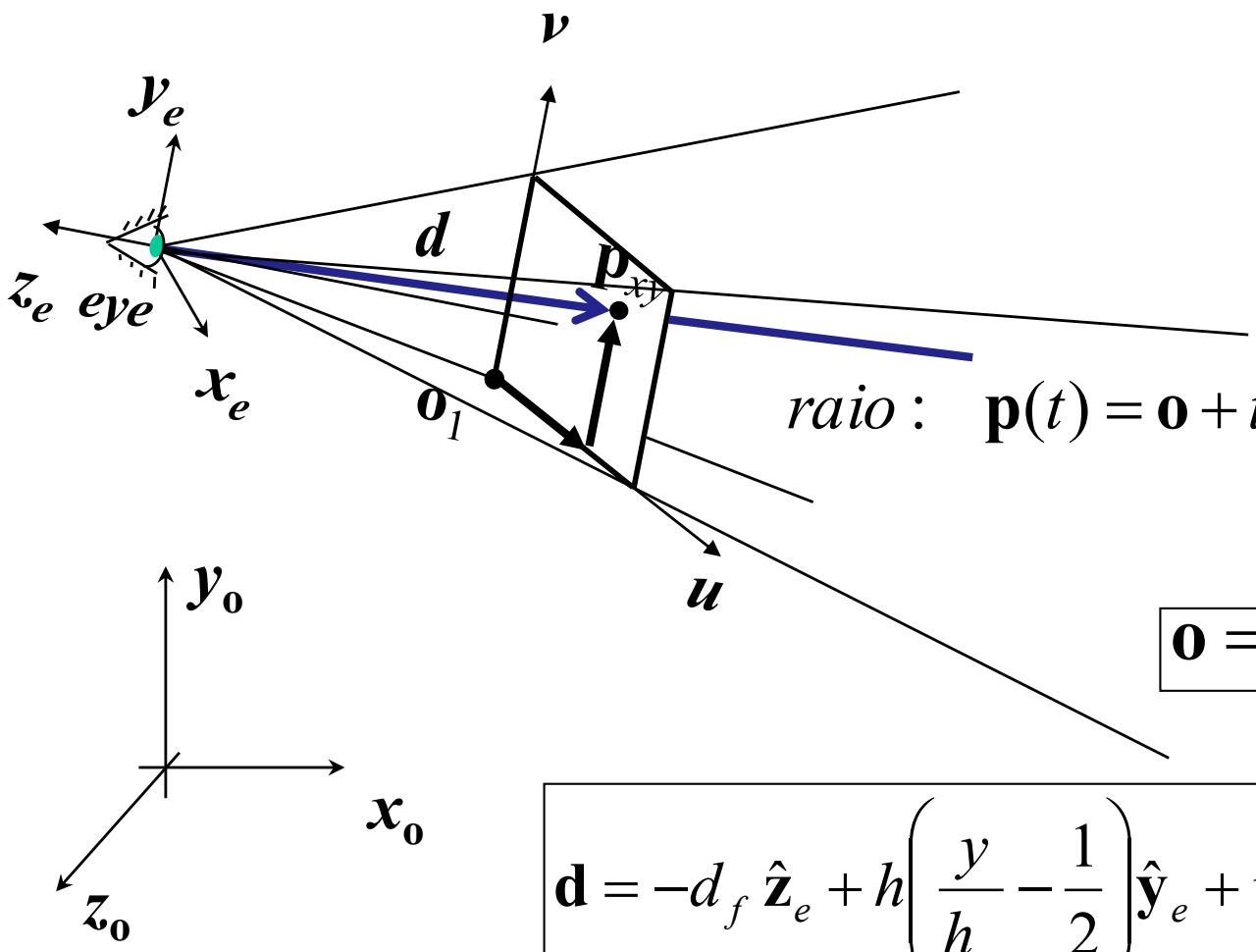


$$\mathbf{o}_1 = \mathbf{eye} - d_f\,\hat{\mathbf{z}}_e - \frac{h}{2}\hat{\mathbf{y}}_e - \frac{w}{2}\hat{\mathbf{x}}_e$$

$$\mathbf{p}_{xy} = \mathbf{o}_1 + w\frac{x}{w_p}\hat{\mathbf{x}}_e + h\frac{y}{h_p}\hat{\mathbf{y}}_e$$

$$\boxed{\mathbf{d} = -d_f\,\hat{\mathbf{z}}_e + h\left(\frac{y}{h_p} - \frac{1}{2}\right)\hat{\mathbf{y}}_e + w\left(\frac{x}{w_p} - \frac{1}{2}\right)\hat{\mathbf{x}}_e}$$

Resultando



$$raio: \quad \mathbf{p}(t) = \mathbf{o} + t\mathbf{d}$$

$$\boxed{\mathbf{o} = \mathbf{eye}}$$

$$\boxed{\mathbf{d} = -d_f\,\hat{\mathbf{z}}_e + h\left(\frac{y}{h_p} - \frac{1}{2}\right)\hat{\mathbf{y}}_e + w\left(\frac{x}{w_p} - \frac{1}{2}\right)\hat{\mathbf{x}}_e}$$

Tipo Abstrato de Dados: Camera em C

```
struct _Camera {
  /* Definição da câmera */
  Vector eye, center, up;
  float  fovy;
  float  n,f;
  int    wp,hp;

  /* Parametros derivados */
  float  df;
  float  w,h;
  Vector xe,ye,ze;
};

typedef struct _Camera  Camera;
```

```
Camera* camCreate( Vector eye, Vector at, Vector up,
                   double fovy, double _near, double _far, int wp, int hp
);
```

```
Ray camGetRay( Camera camera, double x, double y );
```

Objeto câmera

Inicialização (pré-processamento):

Dados: fov_y, w_p, h_p, n, f, ***eye***, ***center***, ***up***

$$d_f = n \qquad h = 2d_f \tan\left(\frac{fov_y}{2}\right) \qquad w = \frac{w_p}{h_p} h$$

$$\mathbf{z}_e = \frac{1}{\|\mathbf{eye} - \mathbf{center}\|}(\mathbf{eye} - \mathbf{center}) \qquad \mathbf{x}_e = \frac{1}{\|\mathbf{up} \times \mathbf{z}_e\|}(\mathbf{up} \times \mathbf{z}_e) \qquad \mathbf{y}_e = (\mathbf{z}_e \times \mathbf{x}_e)$$

Lançamento de raios: $\mathbf{o} + t\mathbf{d}$

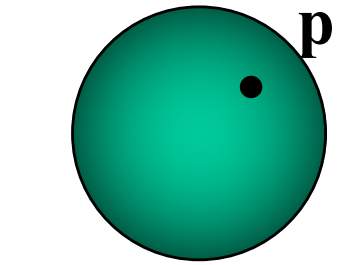
Dados: x, y

$$\mathbf{o} = \mathbf{eye}$$

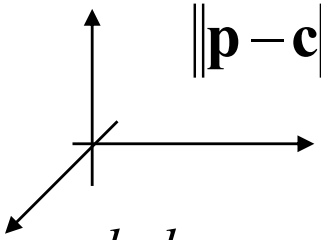
$$\mathbf{d} = -d_f \hat{\mathbf{z}}_e + h\left(\frac{y}{h_p} - \frac{1}{2}\right)\hat{\mathbf{y}}_e + w\left(\frac{x}{w_p} - \frac{1}{2}\right)\hat{\mathbf{x}}_e$$

Modelagem dos Objetos

Implícita:



$$\|\mathbf{p}-\mathbf{c}\| = r$$

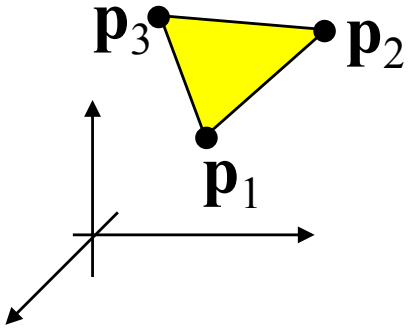


dados:

centro **c**

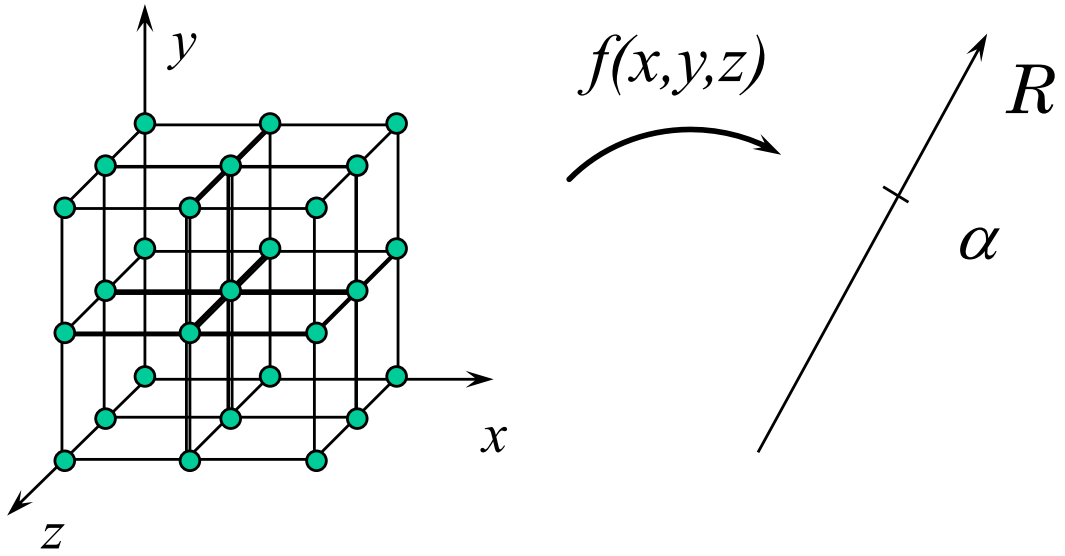
raio r

Por fronteira:



Superfícies implícitas em grades Cartesianas

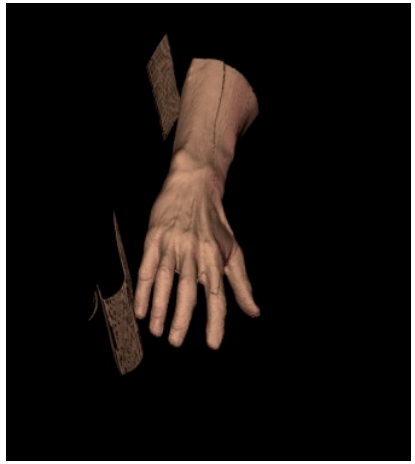
- Seja f(x,y,z), uma função amostrada em R^3

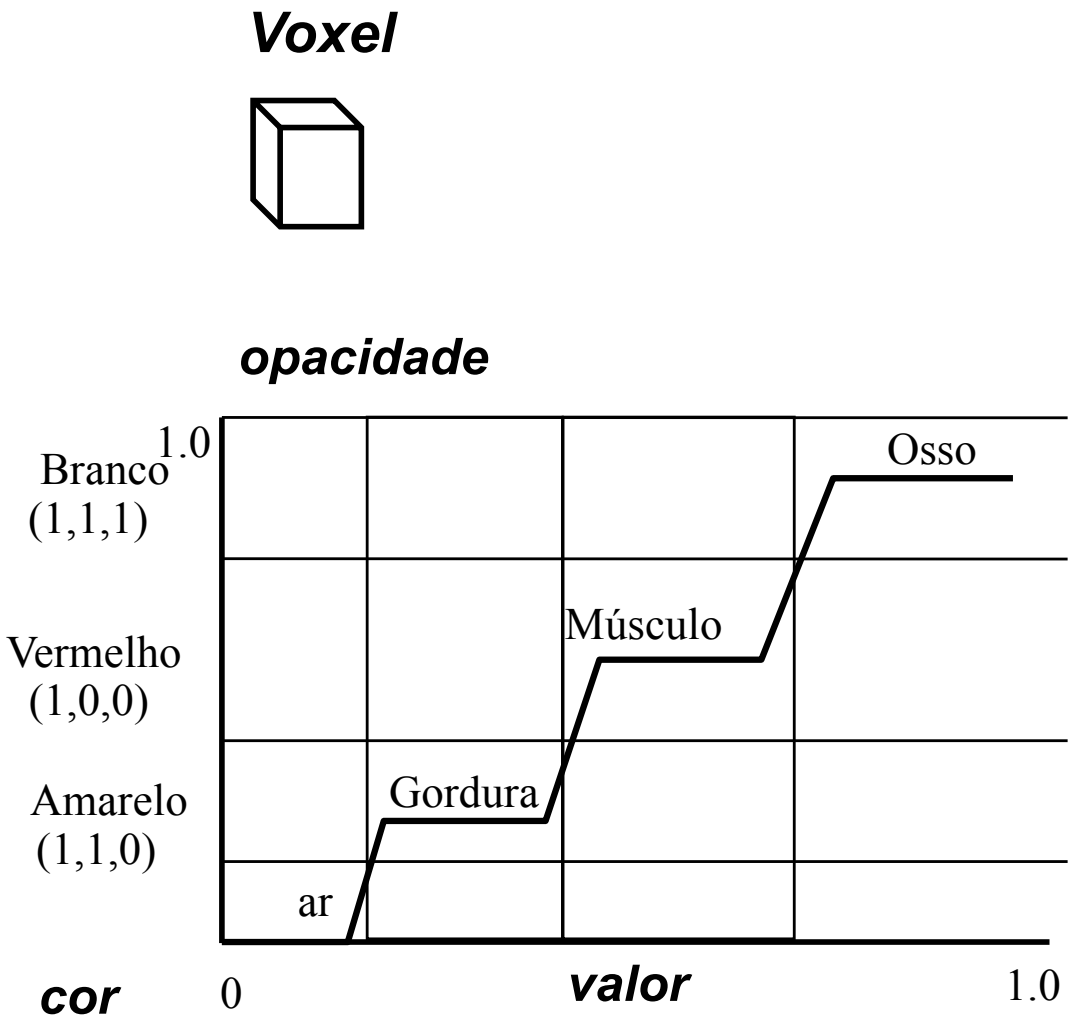


Objetivo:

– Visualizar uma **isosuperfície** S definida por f(x,y,z) = α.

Classificação do Voxel





Interseção de um raio com uma esfera

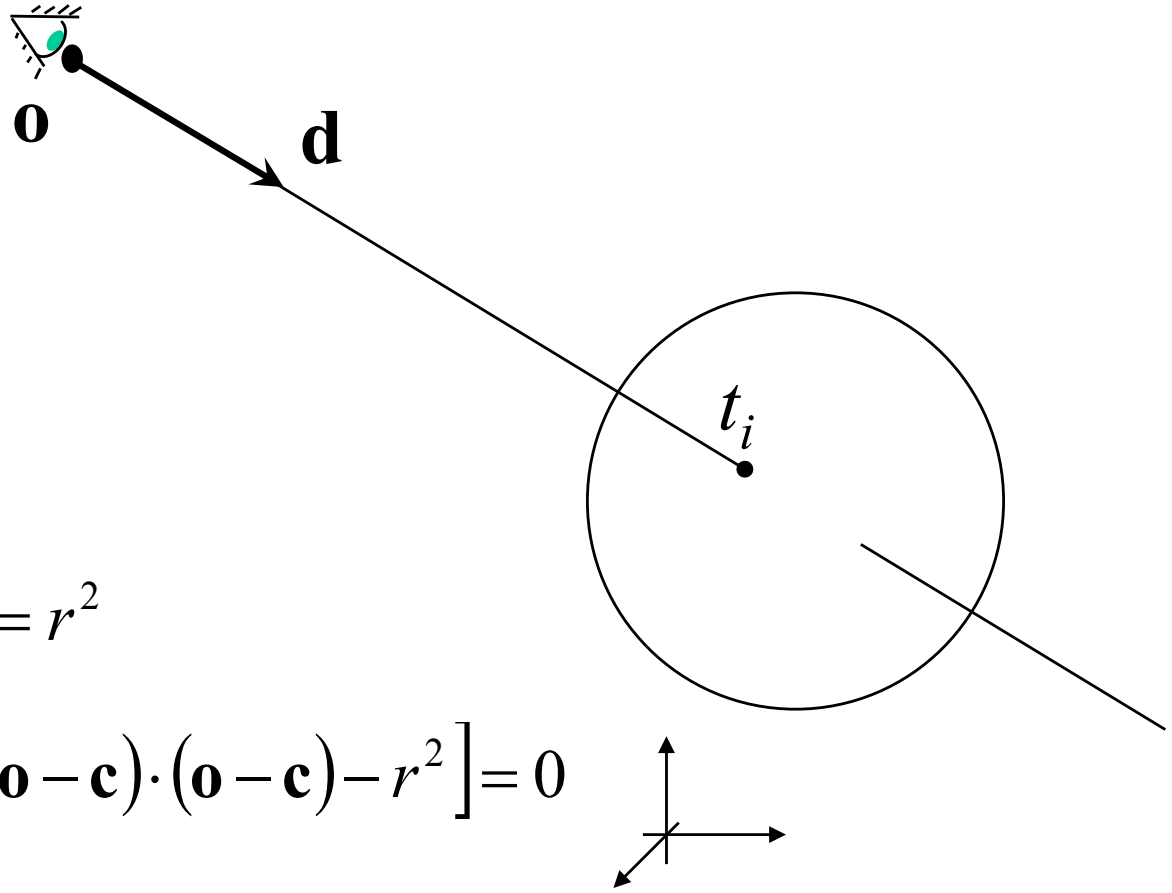
$$Raio: \quad \mathbf{p}(t) = \mathbf{o} + t\mathbf{d}$$

$$Esfera: \quad \|\mathbf{p}(t_i) - \mathbf{c}\|^2 = r^2$$

$$\|\mathbf{o} + t_i\mathbf{d} - \mathbf{c}\|^2 = r^2$$

$$((\mathbf{o} - \mathbf{c}) + t_i\mathbf{d}) \cdot ((\mathbf{o} - \mathbf{c}) + t_i\mathbf{d}) = r^2$$

$$[\mathbf{d} \cdot \mathbf{d}]\, t_i^2 + [2\mathbf{d} \cdot (\mathbf{o} - \mathbf{c})]\, t_i + [(\mathbf{o} - \mathbf{c}) \cdot (\mathbf{o} - \mathbf{c}) - r^2] = 0$$



$$a\, t_i^2 + b\, t_i + c = 0 \qquad t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Objeto esfera: métodos (dados **o**, **d**, **c**, *r*)

Interseção:

$$a = \mathbf{d}\cdot\mathbf{d}$$

$$b = 2\mathbf{d}\cdot(\mathbf{o}-\mathbf{c})$$

$$c = (\mathbf{o}-\mathbf{c})\cdot(\mathbf{o}-\mathbf{c}) - r^2$$

$$se \quad \Delta = b^2 - 4ac > 0$$

$$t_1 = \frac{-b-\sqrt{\Delta}}{2a}$$

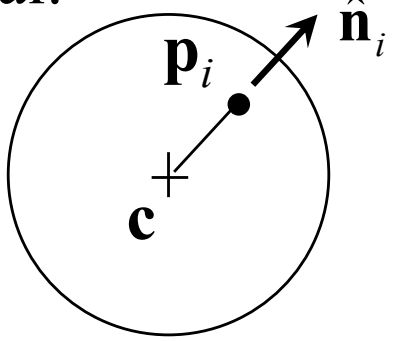
$$t_2 = \frac{-b+\sqrt{\Delta}}{2a}$$

$$t_i = \min(t_1, t_2)$$

$$se\ t_i > 0:$$

$$\mathbf{p}_i = \mathbf{p}(t_i) = \mathbf{o} + t_i\mathbf{d}$$

Normal:



$$\hat{\mathbf{n}}_i = \frac{1}{\|\mathbf{p}_i - \mathbf{c}\|}(\mathbf{p}_i - \mathbf{c})$$

Interseção com o plano do triângulo

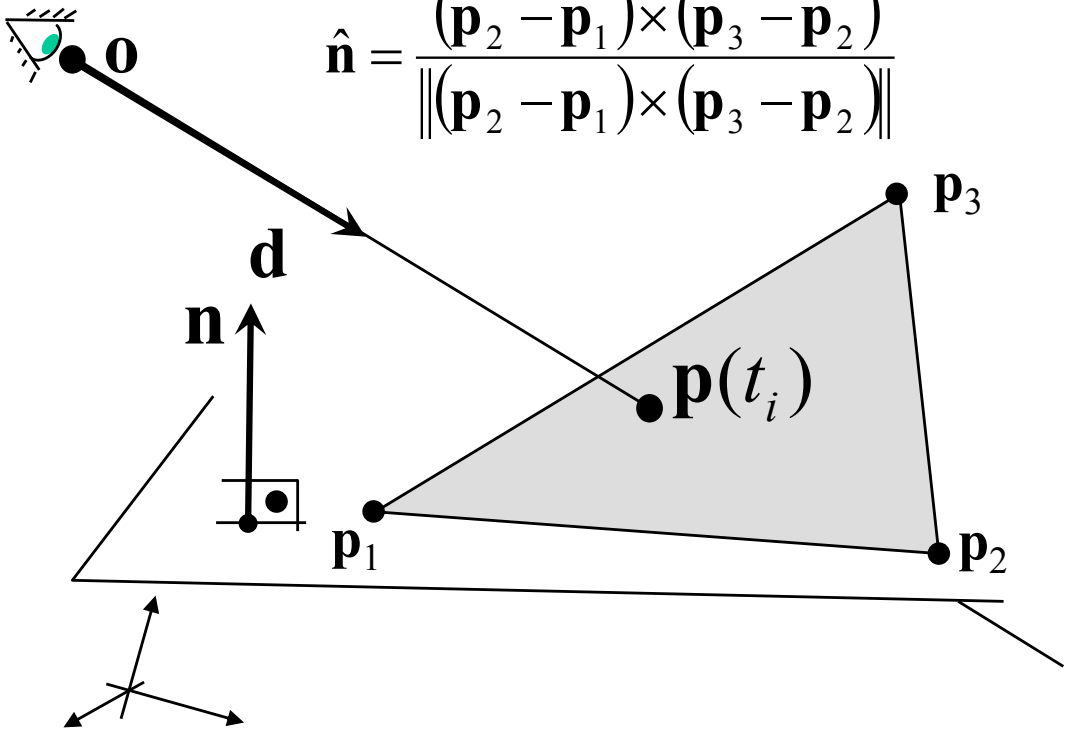
$$Raio: \quad \mathbf{p}(t) = \mathbf{o} + t\mathbf{d}$$

$$Plano: \quad (\mathbf{p}(t_i) - \mathbf{p}_1) \cdot \mathbf{n} = 0$$

$$(\mathbf{o} + t_i\mathbf{d} - \mathbf{p}_1) \cdot \mathbf{n} = 0$$

$$t_i\mathbf{d} \cdot \mathbf{n} + (\mathbf{o} - \mathbf{p}_1) \cdot \mathbf{n} = 0$$

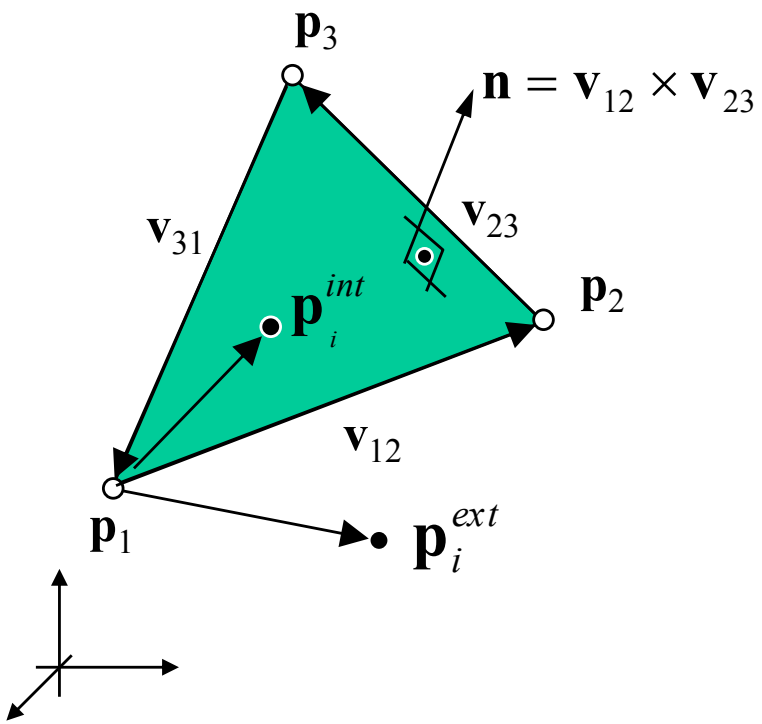
$$\hat{\mathbf{n}} = \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_2)}{\|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_2)\|}$$



$$\boxed{t_i = \frac{(\mathbf{p}_1 - \mathbf{o}) \cdot \mathbf{n}}{\mathbf{d} \cdot \mathbf{n}}}$$

$$\boxed{\mathbf{p}_i = \mathbf{p}(t_i) = \mathbf{o} + t_i\mathbf{d}}$$

Ponto interno a triângulo

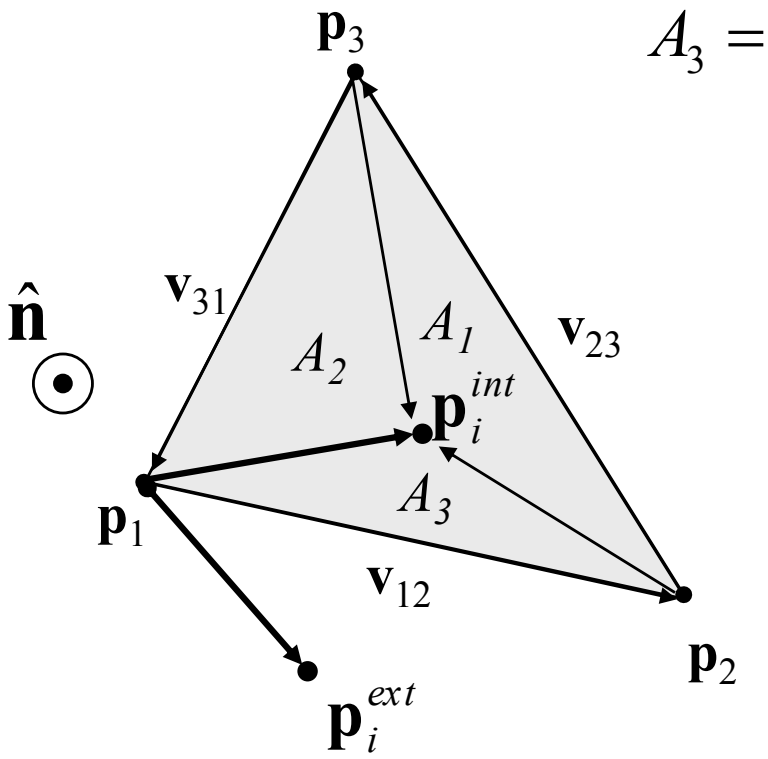


$$\mathbf{n} \cdot \left(\mathbf{v}_{12} \times (\mathbf{p}_i^{int} - \mathbf{p}_1) \right) > 0$$

$$\mathbf{n} \cdot \left(\mathbf{v}_{12} \times (\mathbf{p}_i^{ext} - \mathbf{p}_1) \right) < 0$$

Coordenadas baricêntricas

$$\hat{\mathbf{n}} = unit\left(\mathbf{v}_{12} \times \mathbf{v}_{31}\right)$$



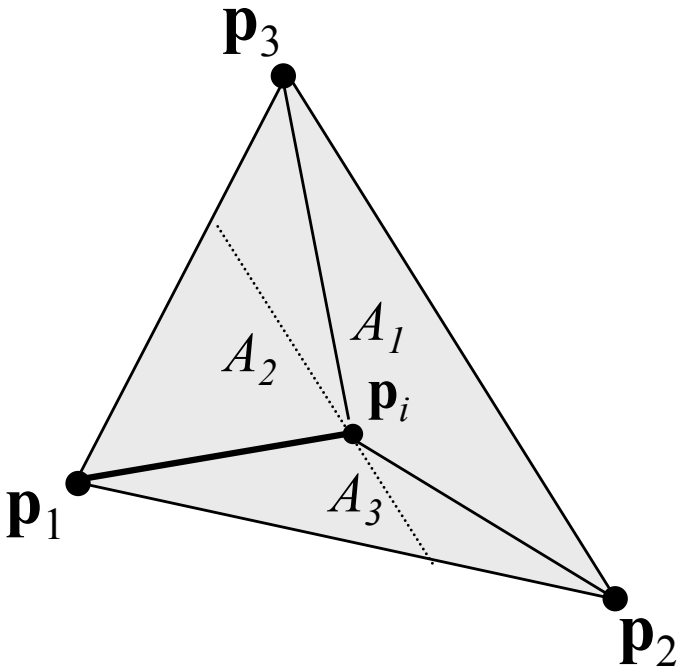
$$A_1 = \hat{\mathbf{n}} \cdot \left(\mathbf{v}_{23} \times (\mathbf{p}_i - \mathbf{p}_2)\right)/2$$
$$A_2 = \hat{\mathbf{n}} \cdot \left(\mathbf{v}_{31} \times (\mathbf{p}_i - \mathbf{p}_3)\right)/2$$
$$A_3 = \hat{\mathbf{n}} \cdot \left(\mathbf{v}_{12} \times (\mathbf{p}_i - \mathbf{p}_1)\right)/2$$

$$A_T = A_1 + A_2 + A_3$$

$$L_1 = A_1 / A_T$$
$$L_2 = A_2 / A_T$$
$$L_3 = A_3 / A_T$$

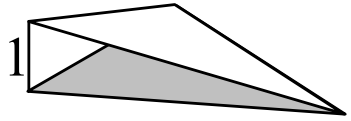
$\mathbf{p}_i$ *é interior se* L_1, L_2 *e* $L_3 \in [0..1]$

Coordenadas baricêntricas como funções interpolantes

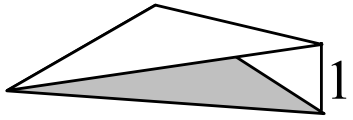


$$\begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = L_1 \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} + L_2 \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} + L_3 \begin{pmatrix} x_3 \\ y_3 \\ z_3 \end{pmatrix}$$

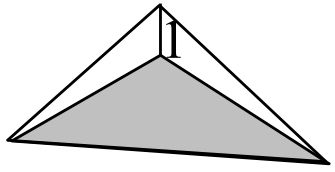
L_1



L_2

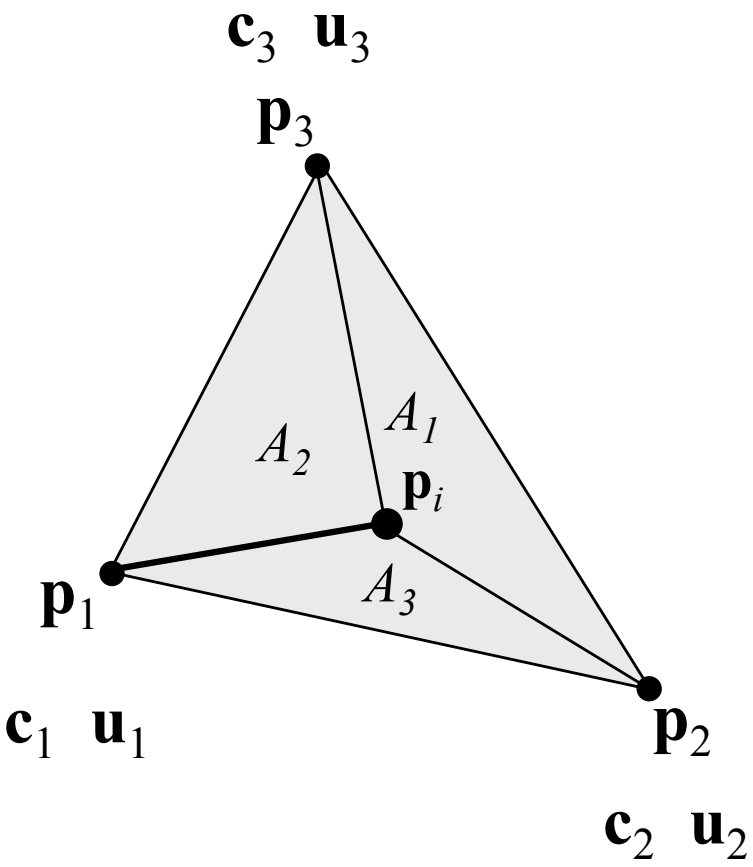


L_3



$$L_1 + L_2 + L_3 = 1$$

Interpolação de cor e coordenada de textura através das coordenadas baricêntricas



$$\begin{Bmatrix} r_i \\ g_i \\ b_i \end{Bmatrix} = L_1 \begin{Bmatrix} r_1 \\ g_1 \\ b_1 \end{Bmatrix} + L_2 \begin{Bmatrix} r_2 \\ g_2 \\ b_2 \end{Bmatrix} + L_3 \begin{Bmatrix} r_3 \\ g_3 \\ b_3 \end{Bmatrix}$$

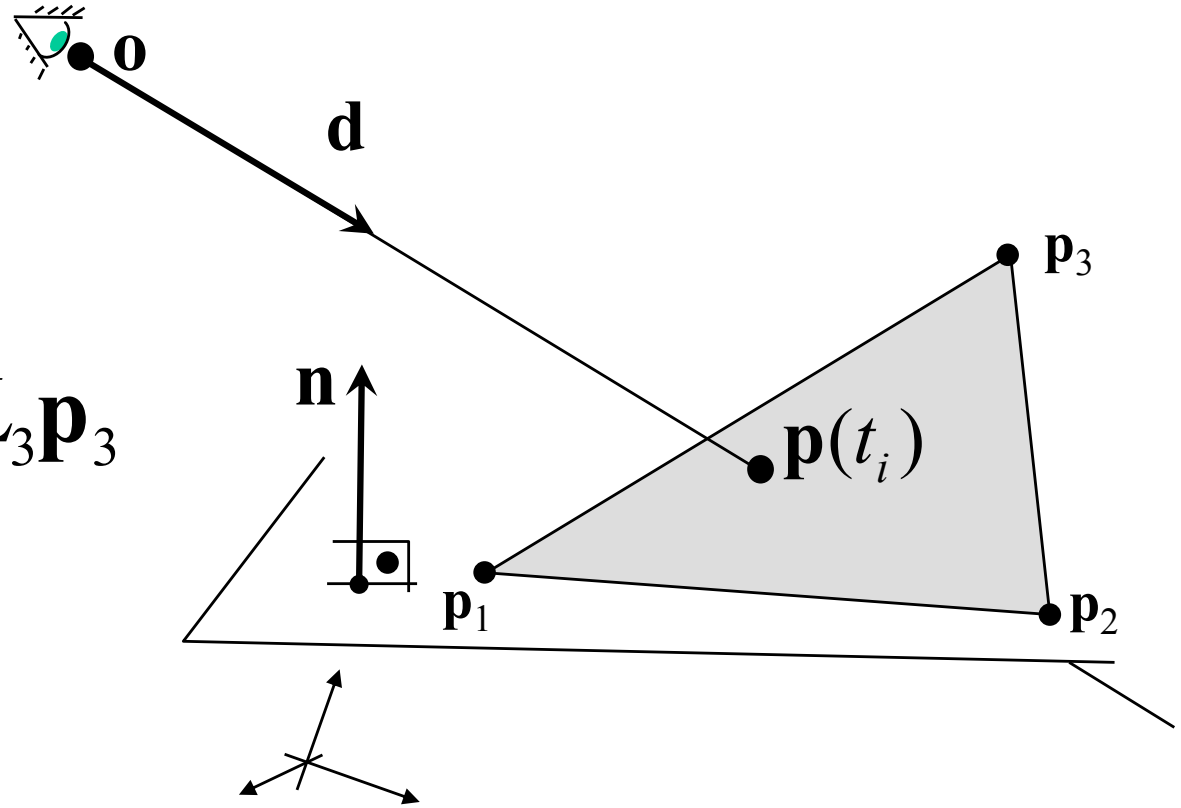
ou:

$$\begin{Bmatrix} u_i \\ v_i \end{Bmatrix} = L_1 \begin{Bmatrix} u_1 \\ v_1 \end{Bmatrix} + L_2 \begin{Bmatrix} u_2 \\ v_2 \end{Bmatrix} + L_3 \begin{Bmatrix} u_3 \\ v_3 \end{Bmatrix}$$

Forma otimizada de calculo

$$\mathbf{p}_i = \mathbf{o} + t_i\mathbf{d}$$

$$\mathbf{p}_i = L_1\mathbf{p}_1 + L_2\mathbf{p}_2 + L_3\mathbf{p}_3$$

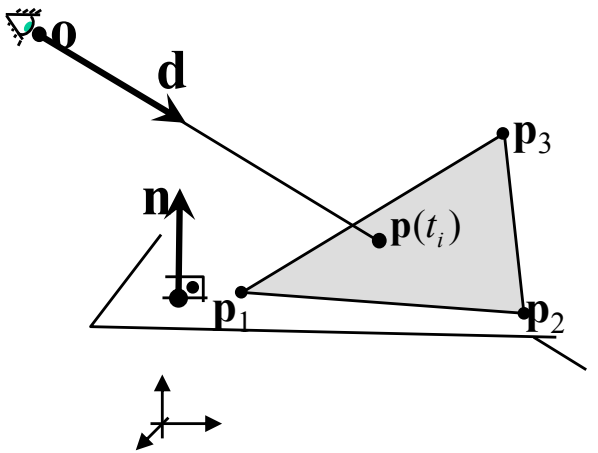


$$\mathbf{p}_i = \mathbf{o} + t_i\mathbf{d} = (1 - L_2 - L_3)_1\mathbf{p}_1 + L_2\mathbf{p}_2 + L_3\mathbf{p}_3$$

Tomas Möller, Ben Trumbore
Fast, Minimum Storage Ray/Triangle Intersection

M

Forma otimizada de cálculo



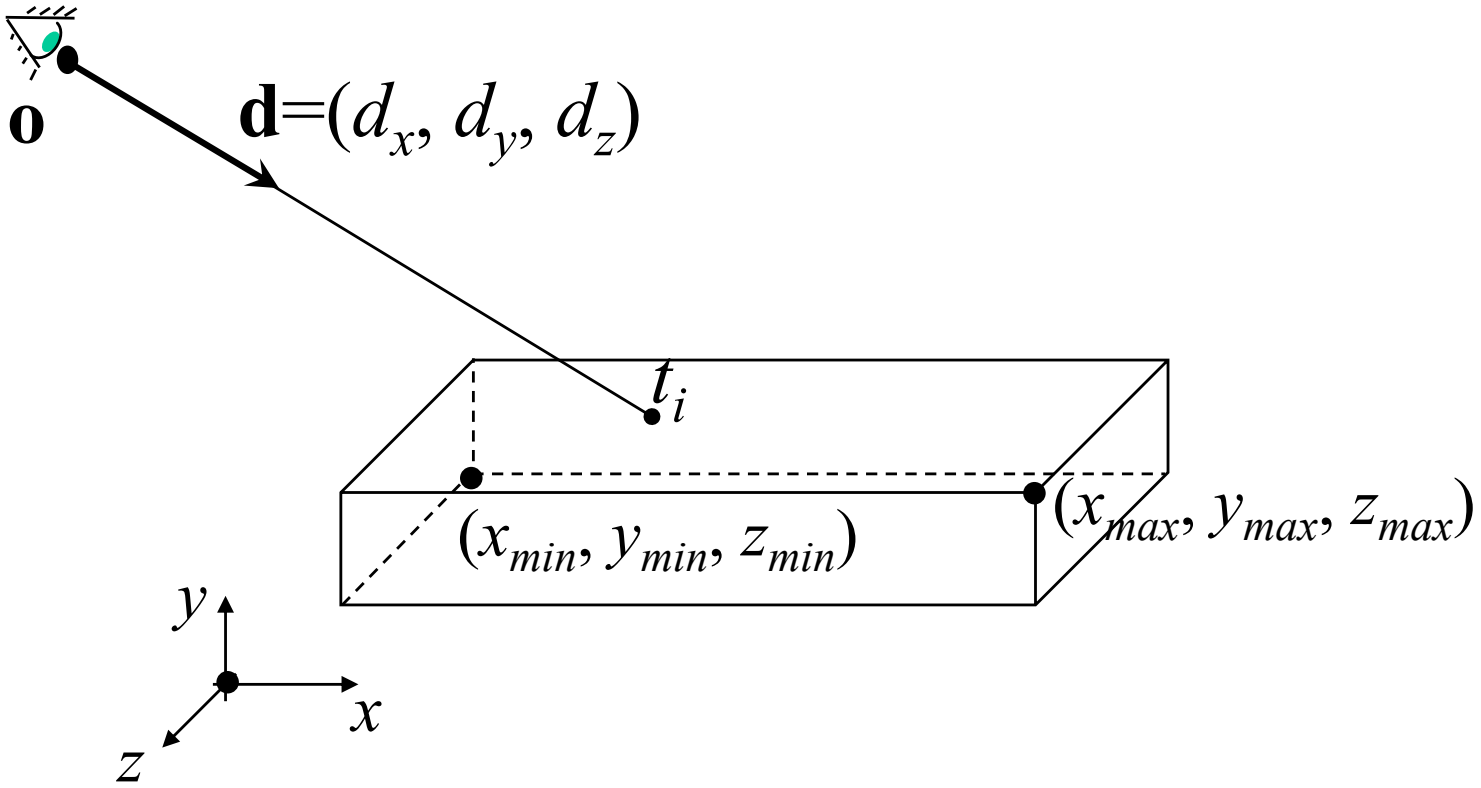
$$\mathbf{o} + t_i\mathbf{d} = (1 - L_2 - L_3)_1\mathbf{p}_1 + L_2\mathbf{p}_2 + L_3\mathbf{p}_3$$

$$-t_i\mathbf{d} + L_2(\mathbf{p}_2 - \mathbf{p}_1) + L_3(\mathbf{p}_3 - \mathbf{p}_1) = \mathbf{o} - \mathbf{p}_1$$

$$\begin{bmatrix} -\mathbf{d} & (\mathbf{p}_2 - \mathbf{p}_1) & (\mathbf{p}_3 - \mathbf{p}_1) \end{bmatrix} \begin{pmatrix} t_i \\ L_2 \\ L_3 \end{pmatrix} = (\mathbf{o} - \mathbf{p}_1)$$

$$\mathbf{Ax} = \mathbf{b}$$

Caixa alinhada com os eixos

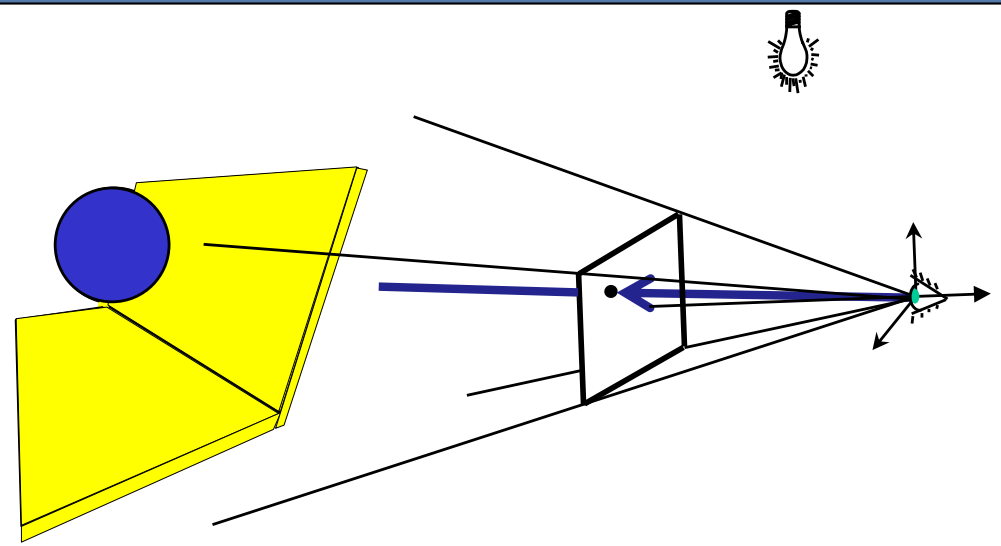


$$d_x > 0 \Rightarrow x = x_{\min}$$

$$d_y < 0 \Rightarrow y = y_{\max}$$

$$d_z < 0 \Rightarrow z = z_{\max}$$

Motivação: Uma cena simples



Camera:

eye = (100,40,40), **center** = (0,0,0), **up**=(0,1,0), fov=90°, near = 30, far=230, *w=230, h=230.*

Esfera:

c = (0,20,0), *r = 25,* cor azul = (0,0,1)

Caixas alinhadas com os eixos:

$\mathbf{p}_0$ = (-80,-50,-50), $\mathbf{p}_1$ = (50,-45,50) e cor amarela = (0.7,0.7,0)

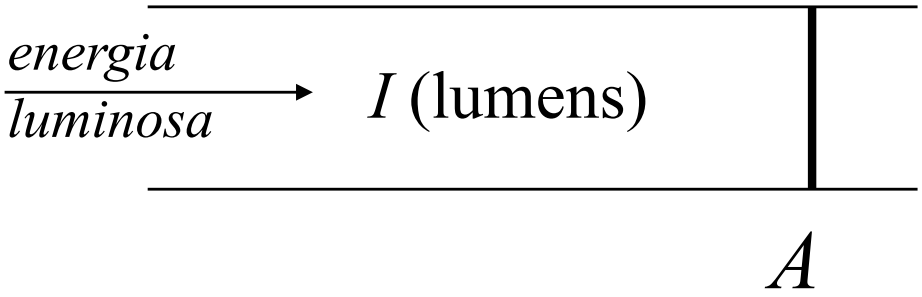
$\mathbf{p}_0$ = (-80,-50,-60), $\mathbf{p}_1$ = (50,50,-50) e cor amarela = (0.7,0.7,0)

Luz Pontual:

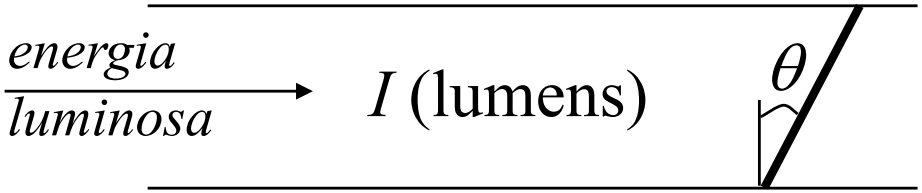
Posição=(60,120,40) e intensidade RGB *l=(0.8,0.8,0.8)*

O que conseguiriamos se simplesmente atribuissemos aos *pixels* a cor dos objetos?

Área aparente



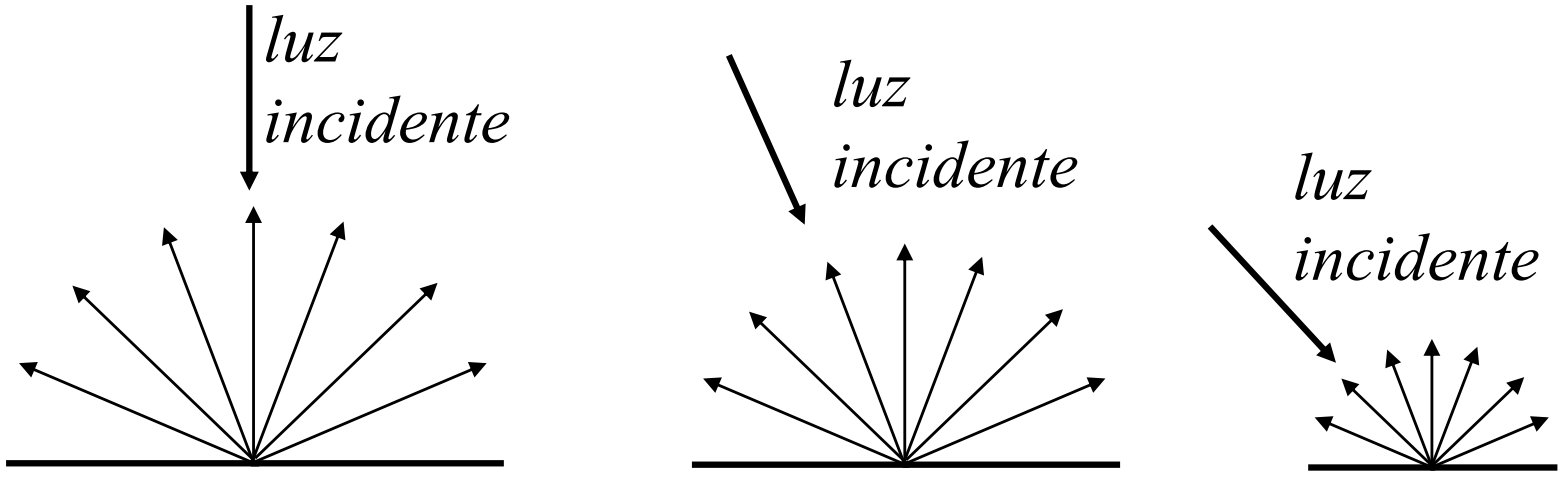
$$i = \frac{I}{A}\ lumens/m^2$$



$$i' = \frac{I}{A}\cos\theta\ lumens/m^2$$

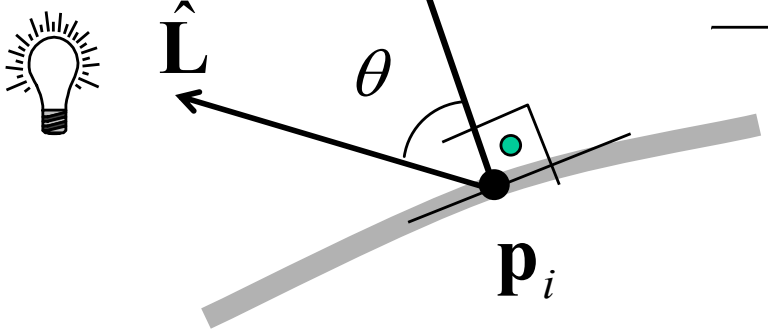
$$A' = \frac{A}{\cos\theta}$$

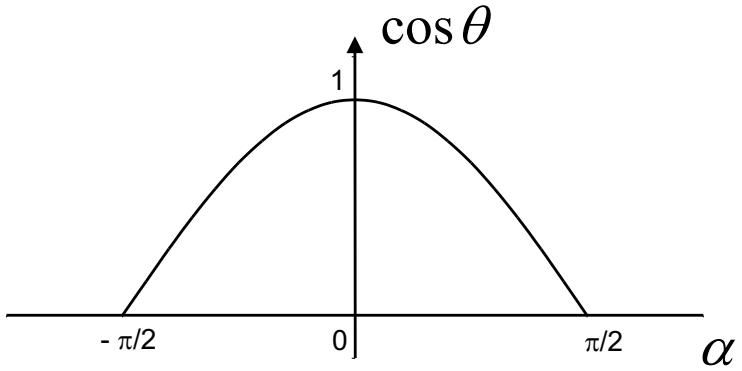
Modelo de reflexão de superfíces Lambertianas



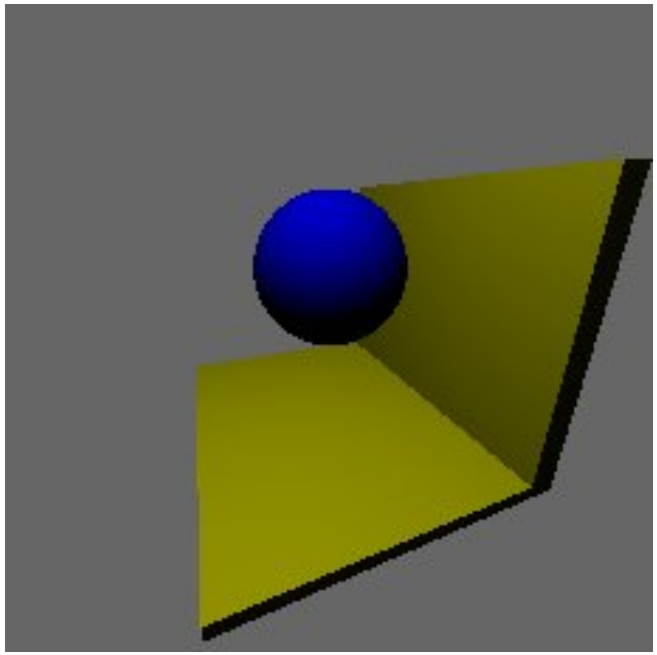
1. *Reflete igualmente em todas as direções*
2. *A intensidade é proporcional ao co-seno*

Componente de reflexão difusa





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



Outras maneiras de se escrever:

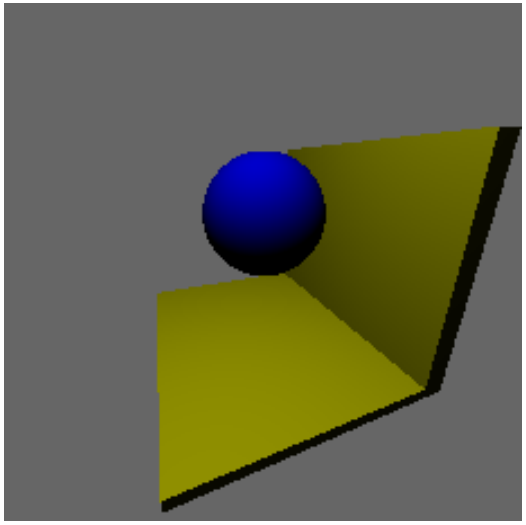
$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} l_r k_{dr} \cos\theta \\ l_g k_{dg} \cos\theta \\ l_b k_{db} \cos\theta \end{pmatrix}$$

$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} l_r k_{dr} \left(\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}\right) \\ l_g k_{dg} \left(\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}\right) \\ l_b k_{db} \left(\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}\right) \end{pmatrix} = \begin{pmatrix} l_r k_{dr} \\ l_g k_{dg} \\ l_b k_{db} \end{pmatrix} \left(\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}\right) = \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} \left(\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}\right) \qquad \textbf{Eq. 1}$$

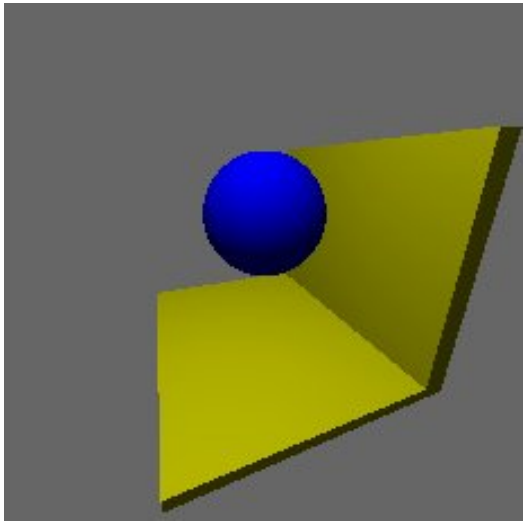
$$I,\ l,\ k \in [0\ ,1\ ]$$

Luz difusa mais ambiente:

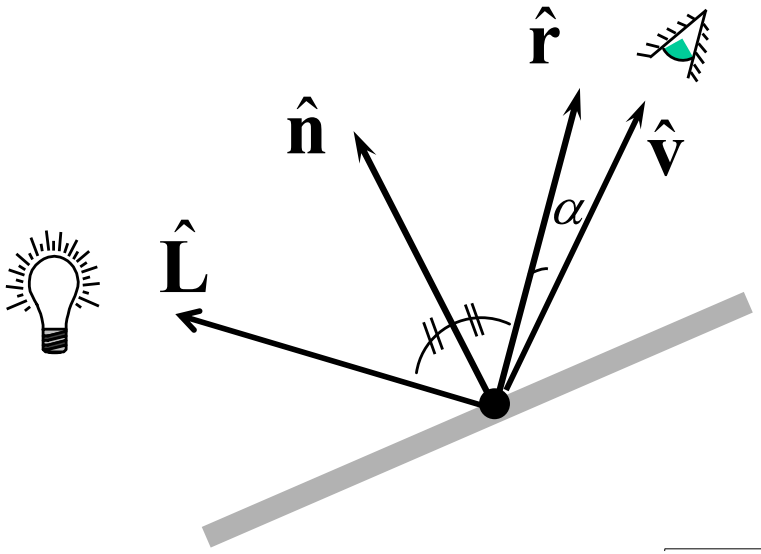
$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} \left(\hat{\mathbf{n}} \cdot \hat{\mathbf{L}} \right)$$

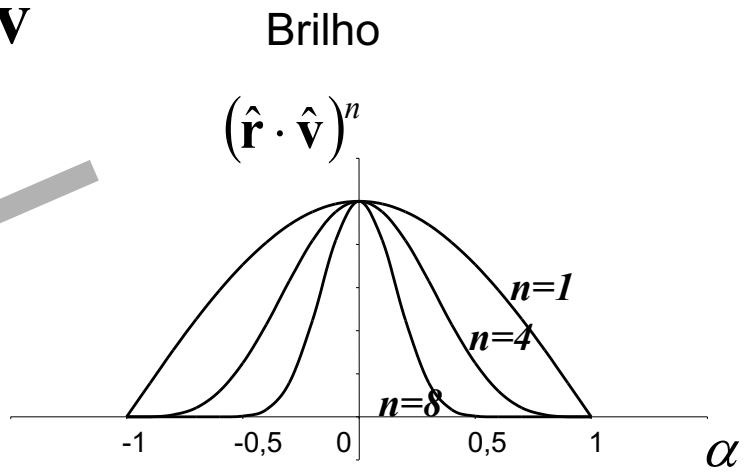


$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} + \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} \left(\hat{\mathbf{n}}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



Componente de reflexão especular



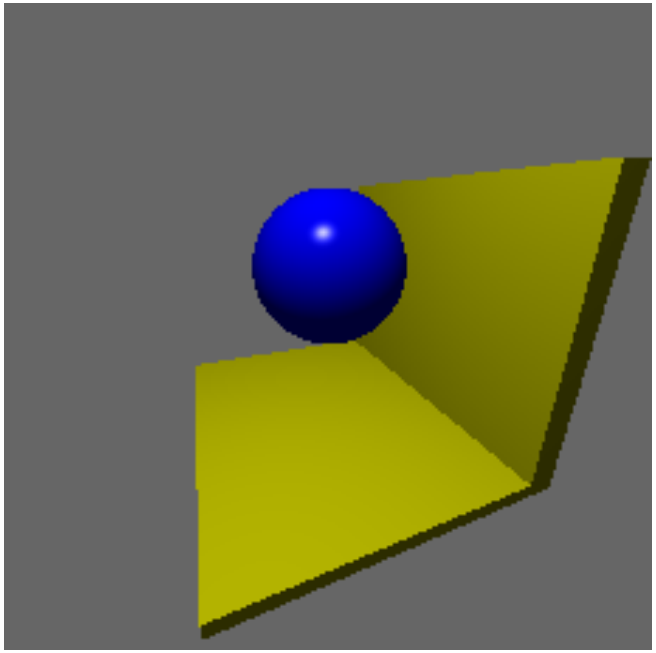


$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix}_{especular} = \begin{pmatrix} l_r k_{sr} (\cos\alpha)^n \\ l_g k_{sg} (\cos\alpha)^n \\ l_b k_{sb} (\cos\alpha)^n \end{pmatrix}$$

Eq.2

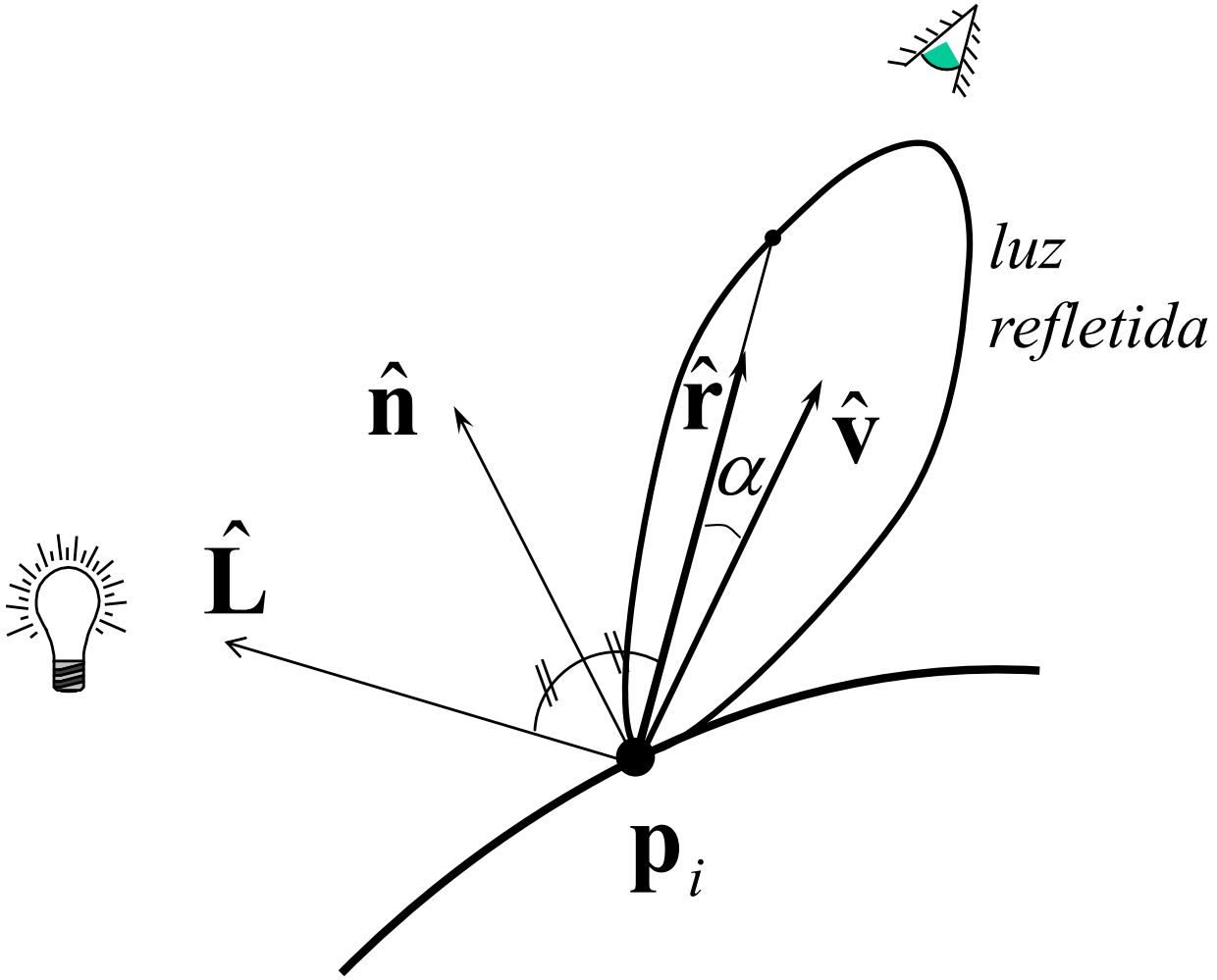
$I, l, k \in [0, 1]$

Reflexão especular

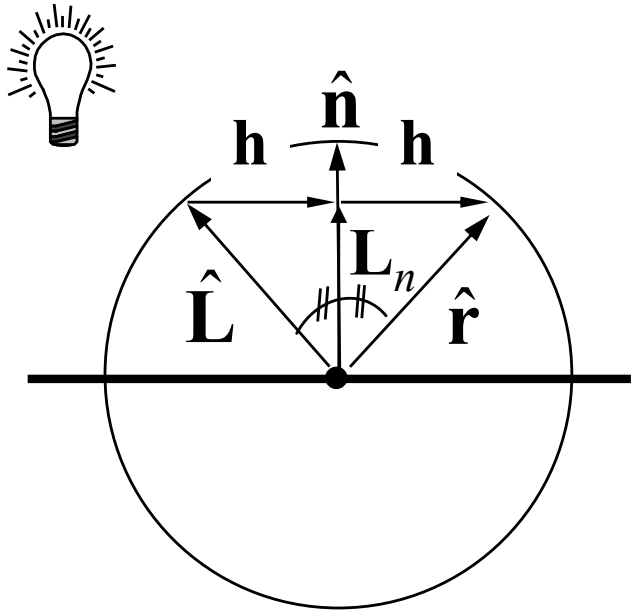


$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} + \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} \left(\hat{\mathbf{n}}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

Reflexão especular



Cálculo da reflexão de um vetor sobre outros



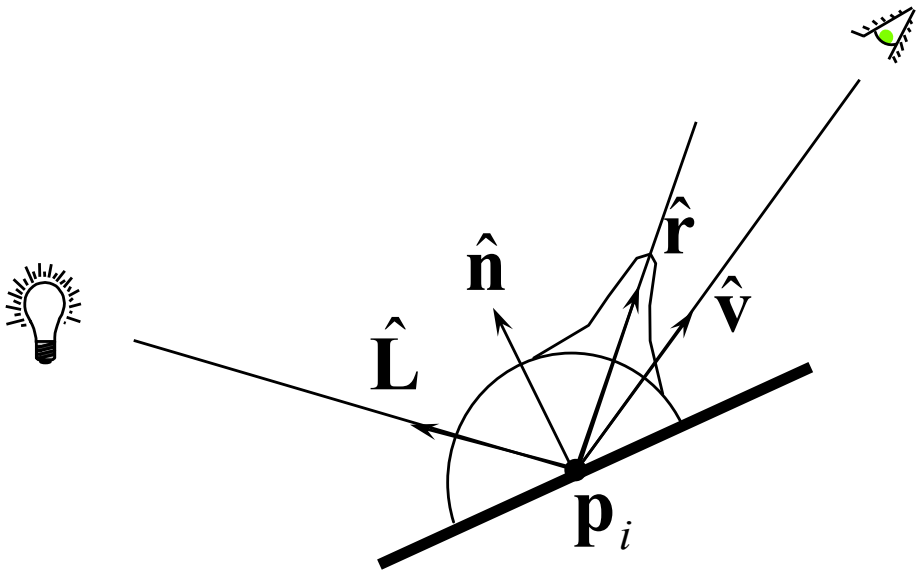
$$\mathbf{L}_n = (\hat{\mathbf{L}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}}$$

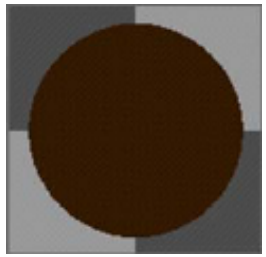
$$\mathbf{h} = \mathbf{L}_n - \mathbf{L}$$

$$\hat{\mathbf{r}} = \mathbf{L}_n + \mathbf{h}$$

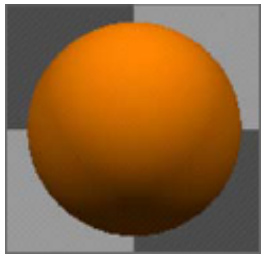
$$\boxed{\hat{\mathbf{r}} = 2\left(\hat{\mathbf{L}} \cdot \hat{\mathbf{n}}\right)\hat{\mathbf{n}} - \hat{\mathbf{L}}}$$

Distribuição da luz direta sobre um ponto

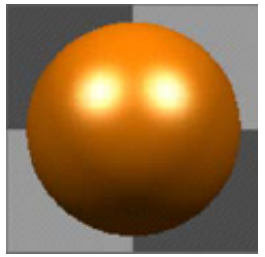




Ambient



Diffuse



Specular

$$C_\lambda = C_{amb_\lambda} + C_{luz_\lambda} k_{dif\,\lambda} \left(\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}\right) + C_{luz_\lambda} k_{s\lambda} \left(\hat{\mathbf{r}} \cdot \hat{\mathbf{v}}\right)^n$$

Modelo de várias luzes

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

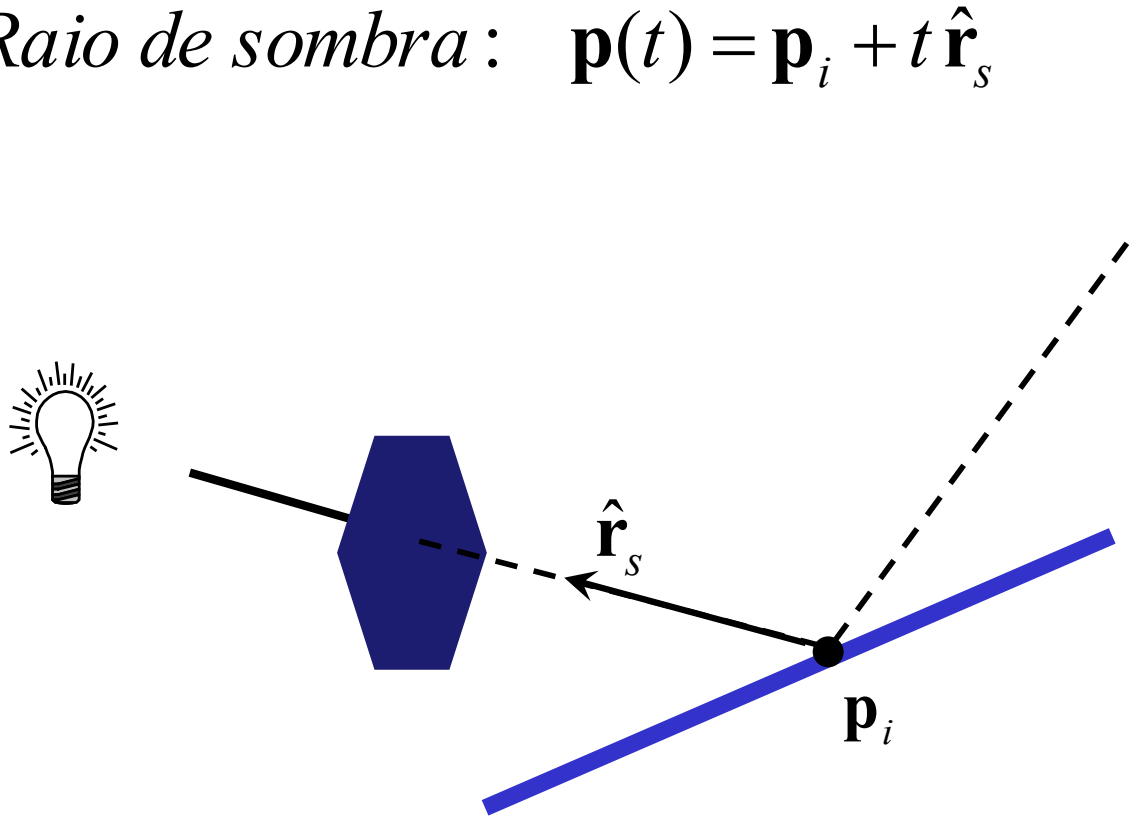
$$\hat{\mathbf{r}} = 2(\hat{\mathbf{L}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}} - \hat{\mathbf{L}} \qquad \textit{para cada fonte de luz}$$

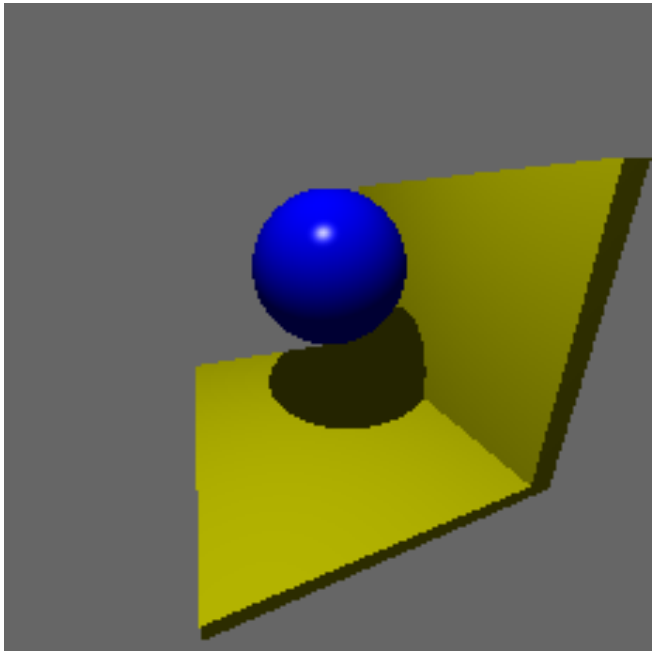
$$\hat{\mathbf{r}}_r = 2(\hat{\mathbf{v}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}} - \hat{\mathbf{v}} \qquad \textit{uma reflexão apenas}$$

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

Sombra

$$\textit{Raio de sombra}: \quad \mathbf{p}(t) = \mathbf{p}_i + t\,\hat{\mathbf{r}}_s$$





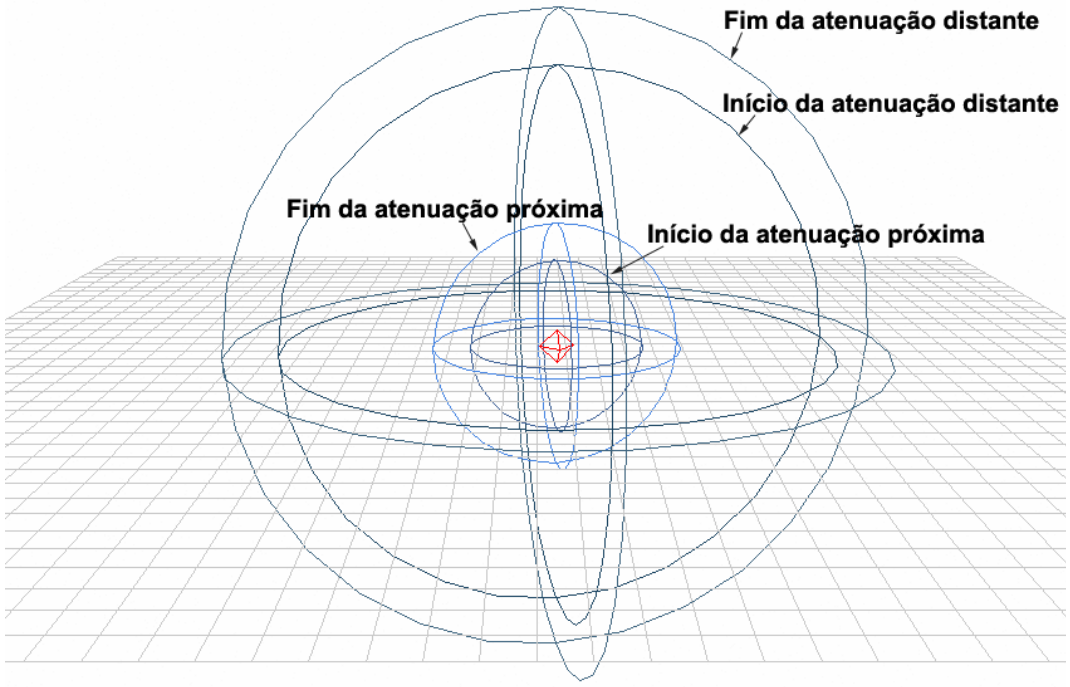
A luz não chega a superfície

Modelo de várias luzes e sombra

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

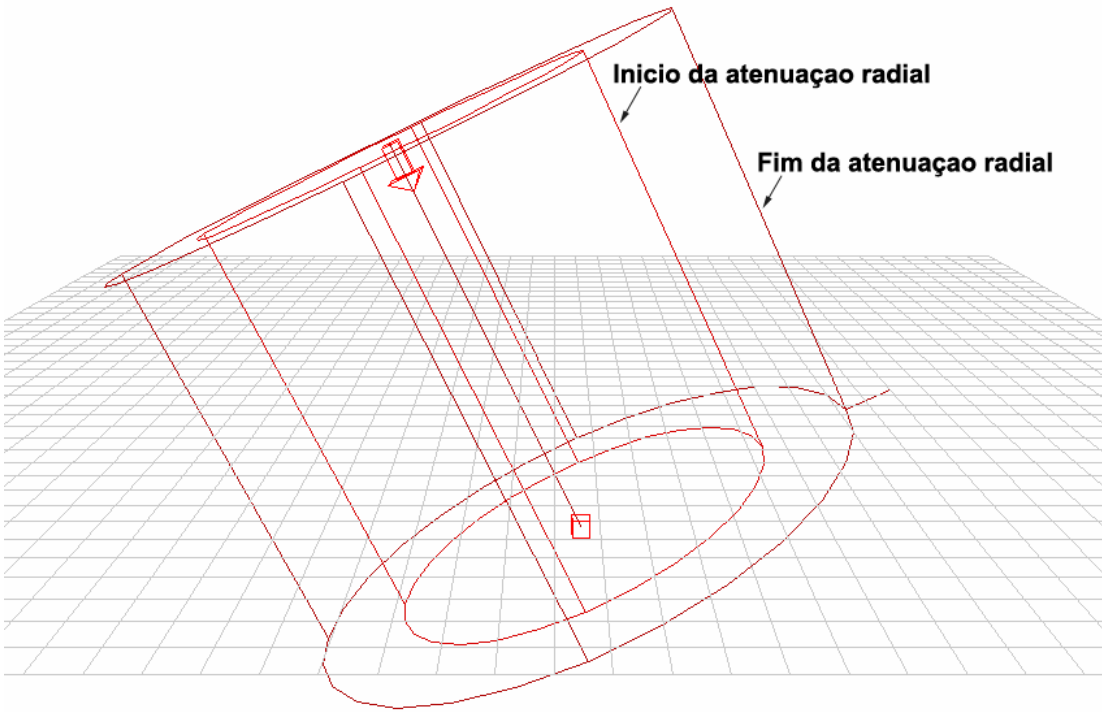
$$f_s = \begin{cases} 0 & se\ sombra \\ 1 & caso\ contrário \end{cases}$$

Fontes de luz especiais: Onidirecional



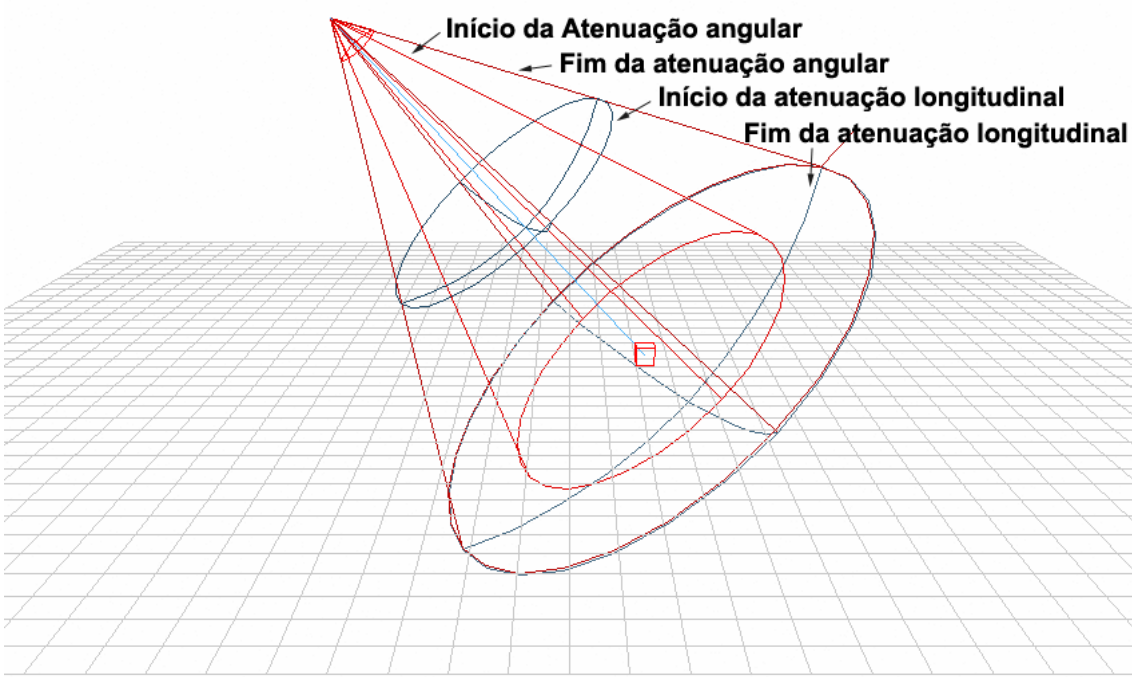
Fonte: Mauricio Hofmam

Fontes de luz especiais: Direcional



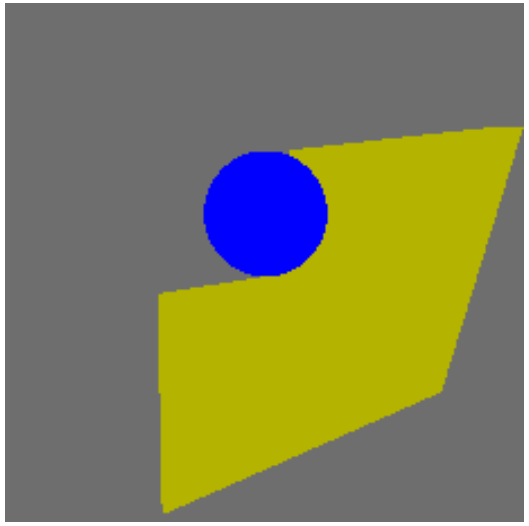
Fonte: Mauricio Hofmam

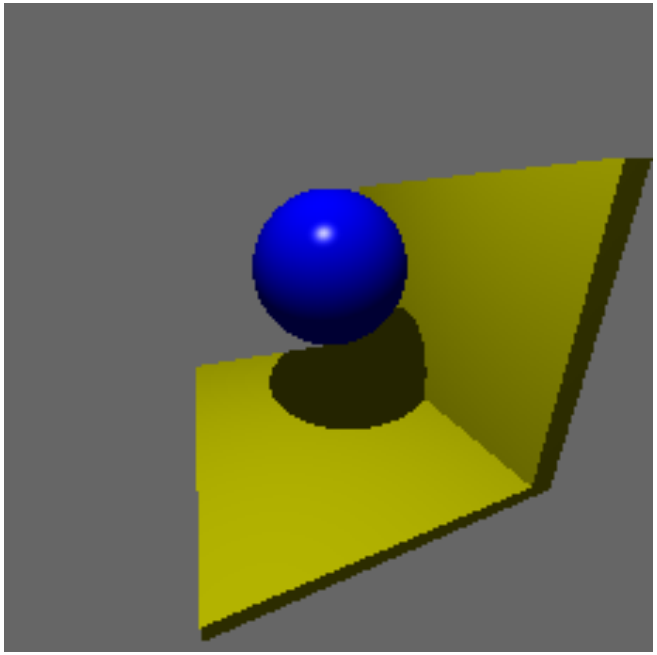
Fontes de luz especiais: Farolete

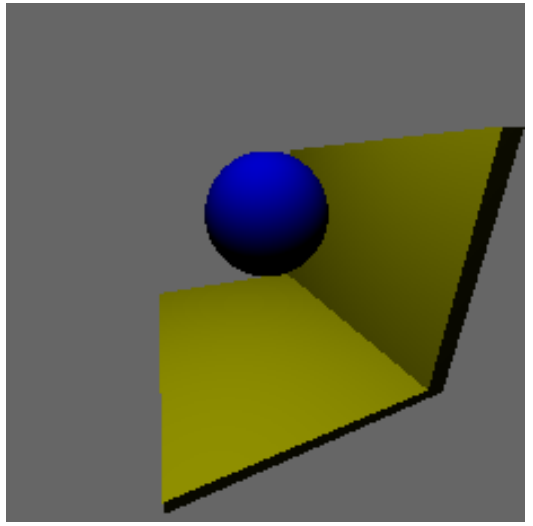


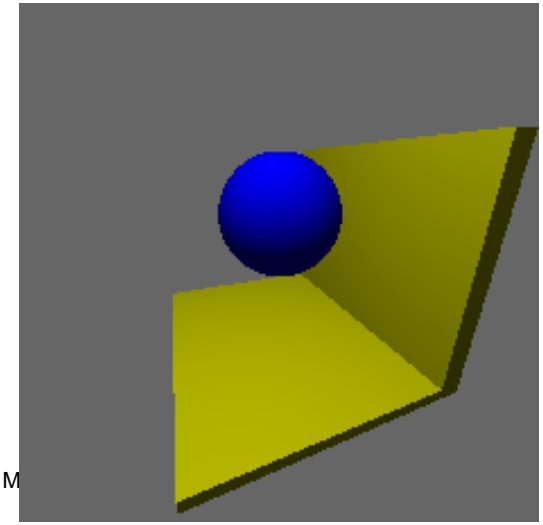
Fonte: Mauricio Hofmam

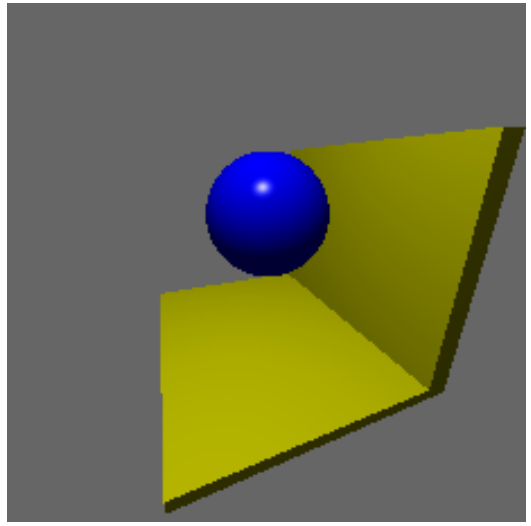
Uma revisão





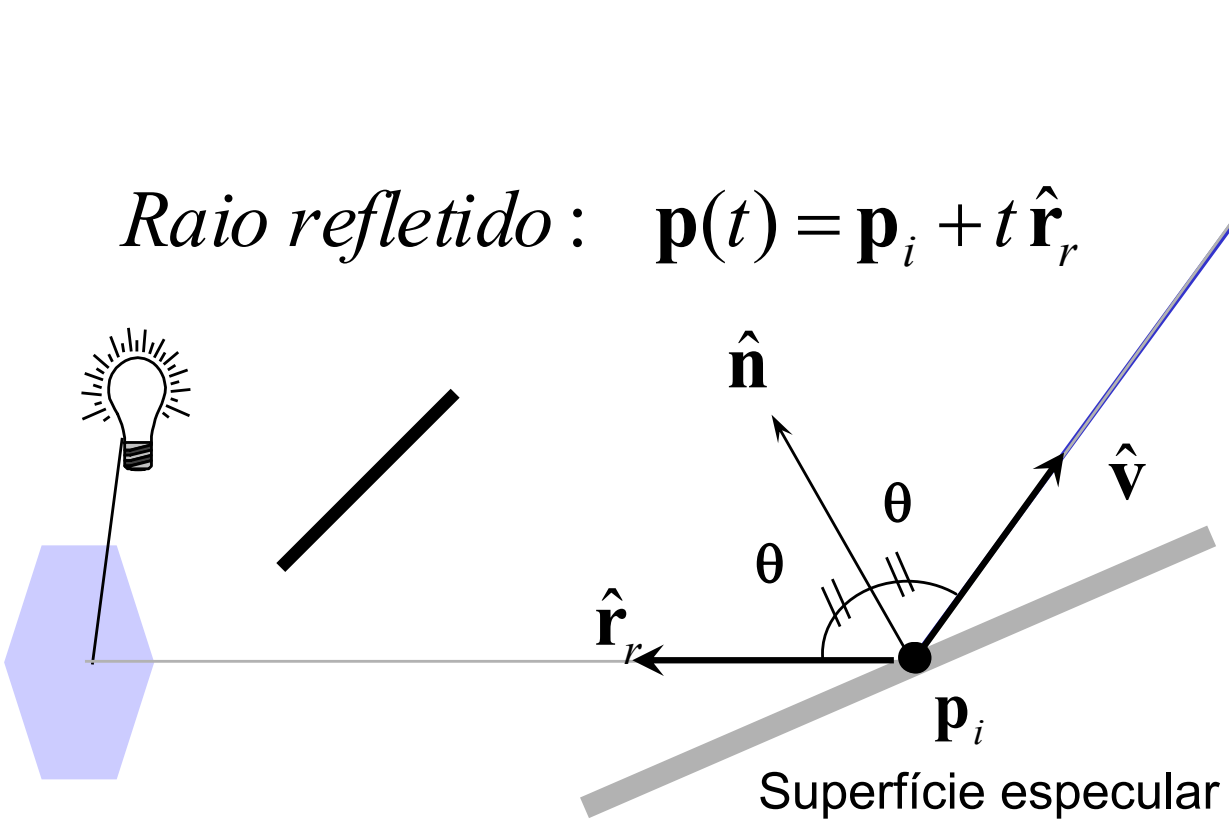


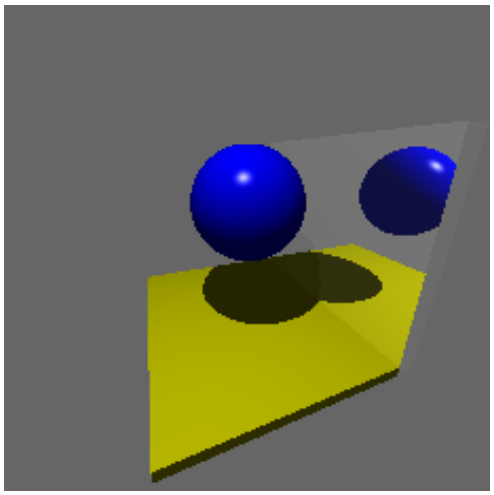


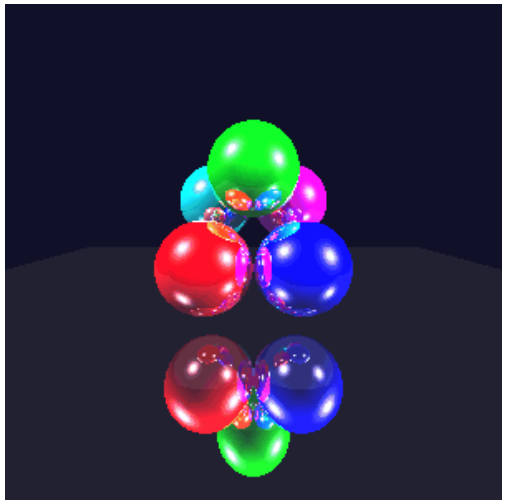


Reflexão de outros objetos

$$\textit{Raio refletido}: \quad \mathbf{p}(t) = \mathbf{p}_i + t\,\hat{\mathbf{r}}_r$$

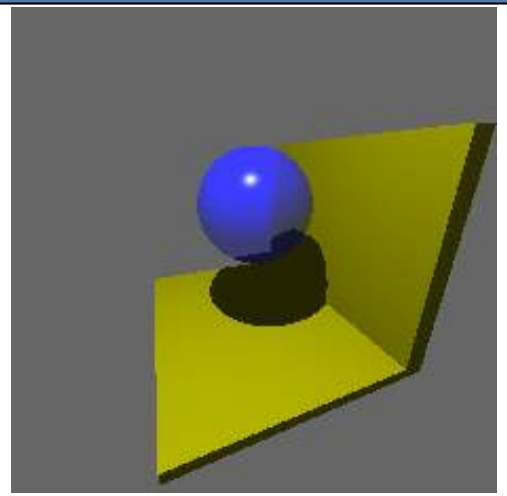


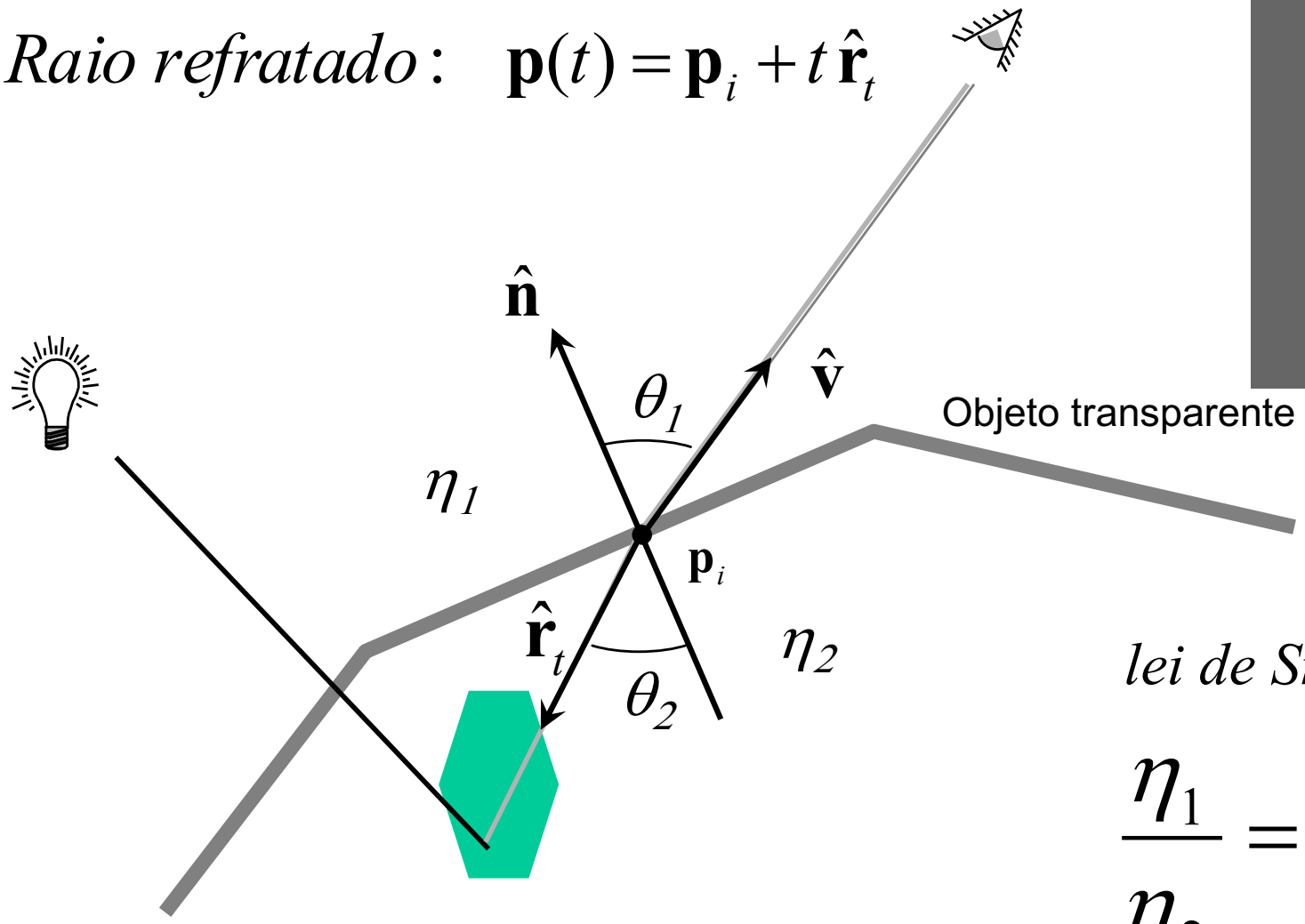




Transparência

Raio refratado: $\mathbf{p}(t) = \mathbf{p}_i + t\,\hat{\mathbf{r}}_t$

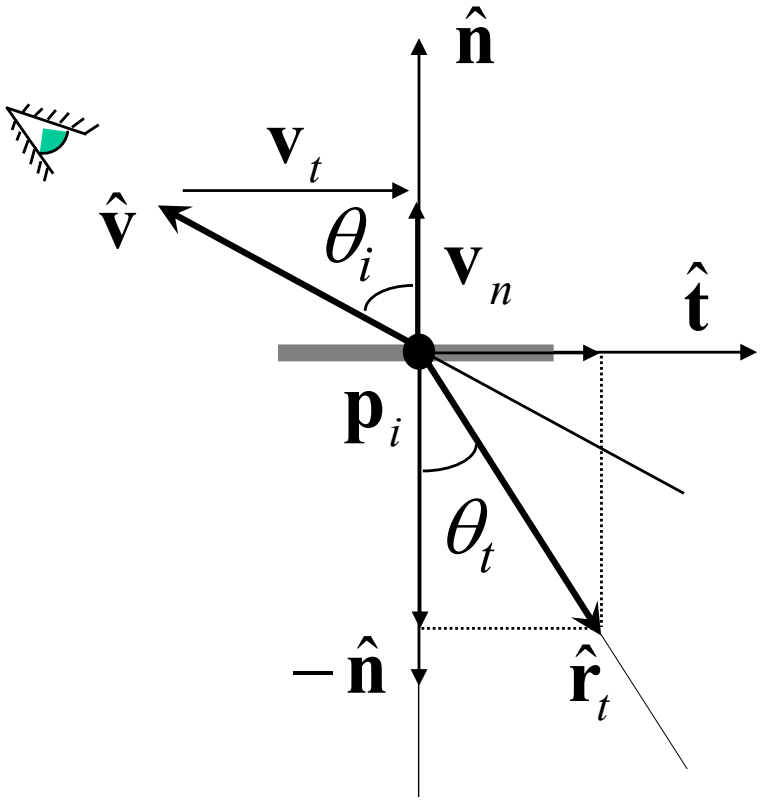




lei de Snell

$$\frac{\eta_1}{\eta_2} = \frac{\sin\theta_2}{\sin\theta_1}$$

Cálculo do Raio Refratado



$$\mathbf{v}_t = (\hat{\mathbf{v}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}} - \hat{\mathbf{v}}$$

$$\sin\theta_i = \|\mathbf{v}_t\|$$

$$\sin\theta_t = \frac{\eta_i}{\eta_t}\sin\theta_i$$

$$\cos\theta_t = \sqrt{1-\sin^2\theta_t}$$

$$\hat{\mathbf{t}} = \frac{1}{\|\mathbf{v}_t\|}\mathbf{v}_t$$

$$\mathbf{r}_t = \sin\theta_t\hat{\mathbf{t}} + \cos\theta_t(-\hat{\mathbf{n}})$$

$$\textit{Raio refratado}: \quad \mathbf{p}(t) = \mathbf{p}_i + t\,\hat{\mathbf{r}}_t$$

Advertência: Refração não é simples!

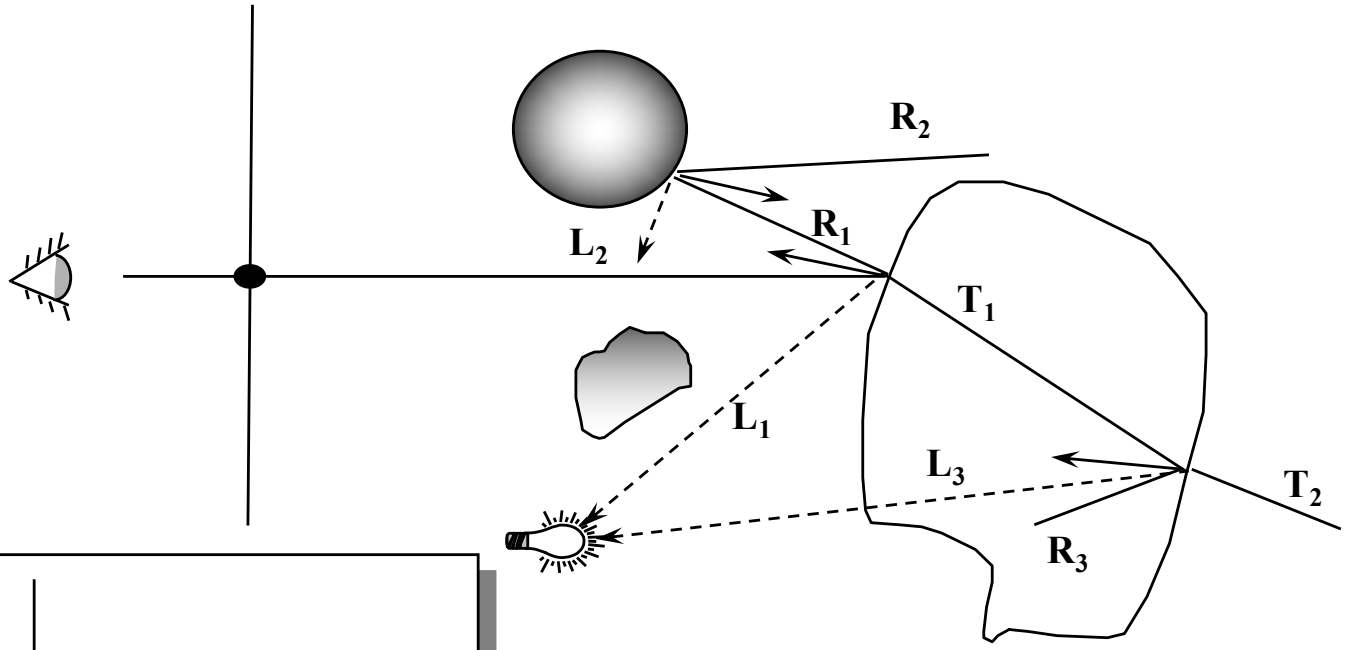
Iluminação considerando superfícies refletoras e objetos transparentes

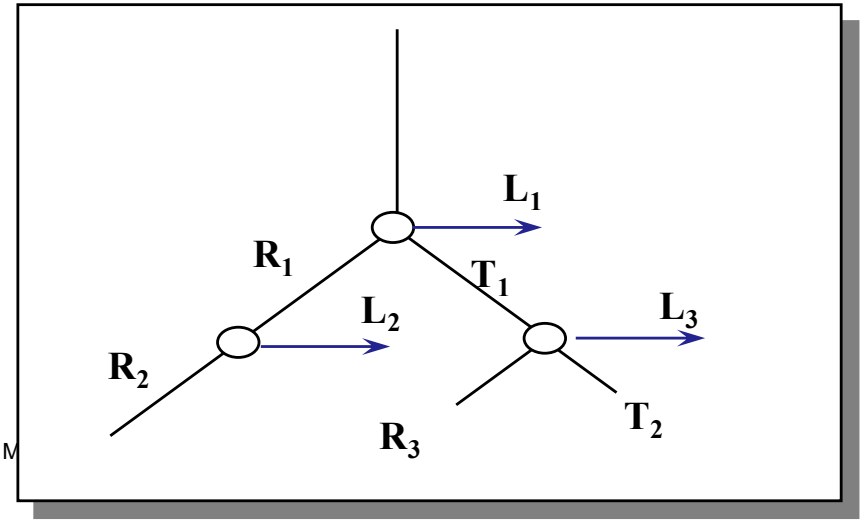
$$\begin{pmatrix} I_r \\ I_g \\ I_b \end{pmatrix} = \begin{pmatrix} I_{ar} \\ I_{ag} \\ I_{ab} \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} + \sum_{luzes} f_s \left(\begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{dr} \\ k_{dg} \\ k_{db} \end{pmatrix} (\hat{\mathbf{n}} \cdot \hat{\mathbf{L}}) + \begin{pmatrix} l_r \\ l_g \\ l_b \end{pmatrix} \otimes \begin{pmatrix} k_{sr} \\ k_{sg} \\ k_{sb} \end{pmatrix} (\hat{\mathbf{r}}_r \cdot \hat{\mathbf{L}})^n \right) + k \begin{pmatrix} I_r(\mathbf{r}_r) \\ I_g(\mathbf{r}_r) \\ I_b(\mathbf{r}_r) \end{pmatrix} + (1-o) \begin{pmatrix} I_r(\mathbf{r}_t) \\ I_g(\mathbf{r}_t) \\ I_b(\mathbf{r}_t) \end{pmatrix}$$

redução da reflexão

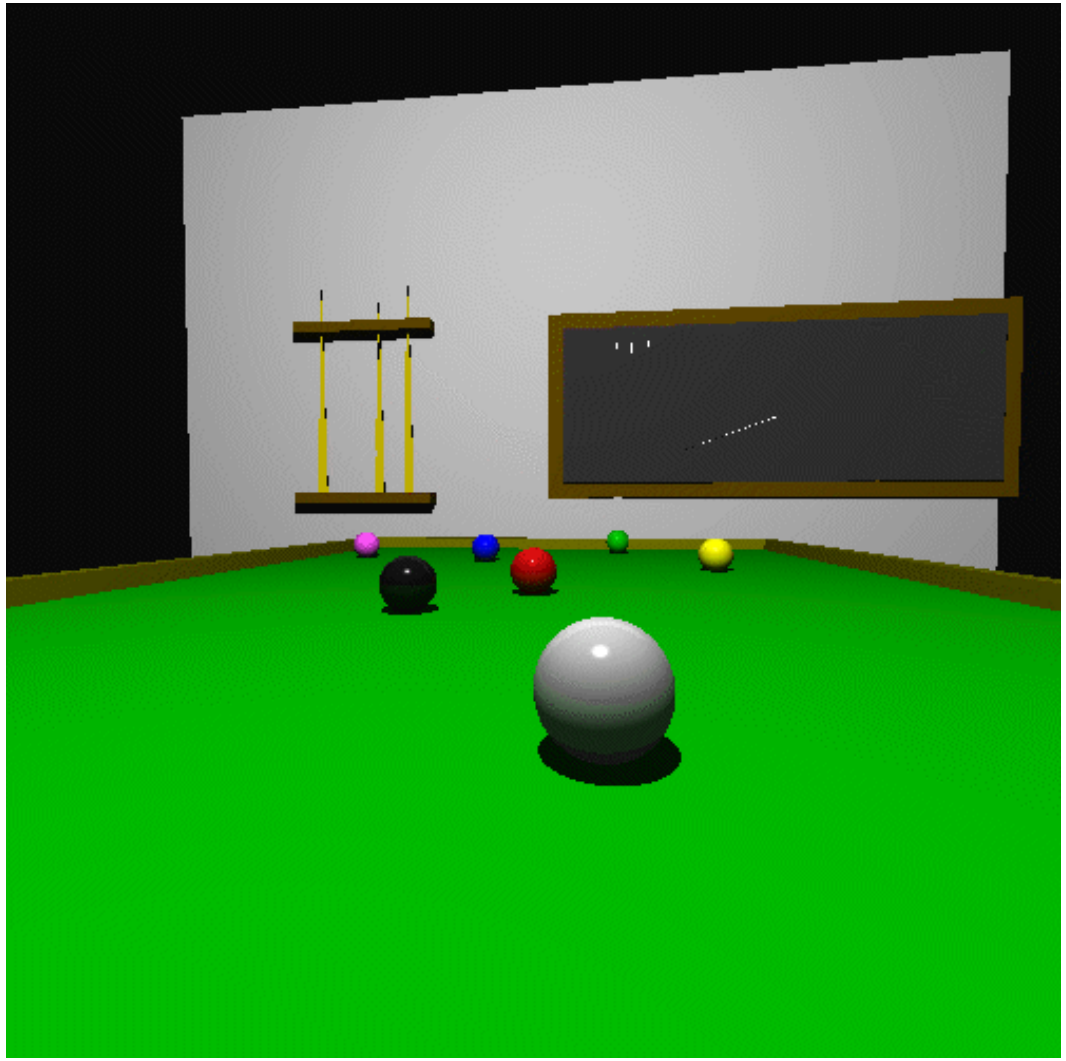
redução da transparência

Natureza recursiva do algoritmo de Rastreamento de Raios





Resultado de curso



Alunos de CGI98

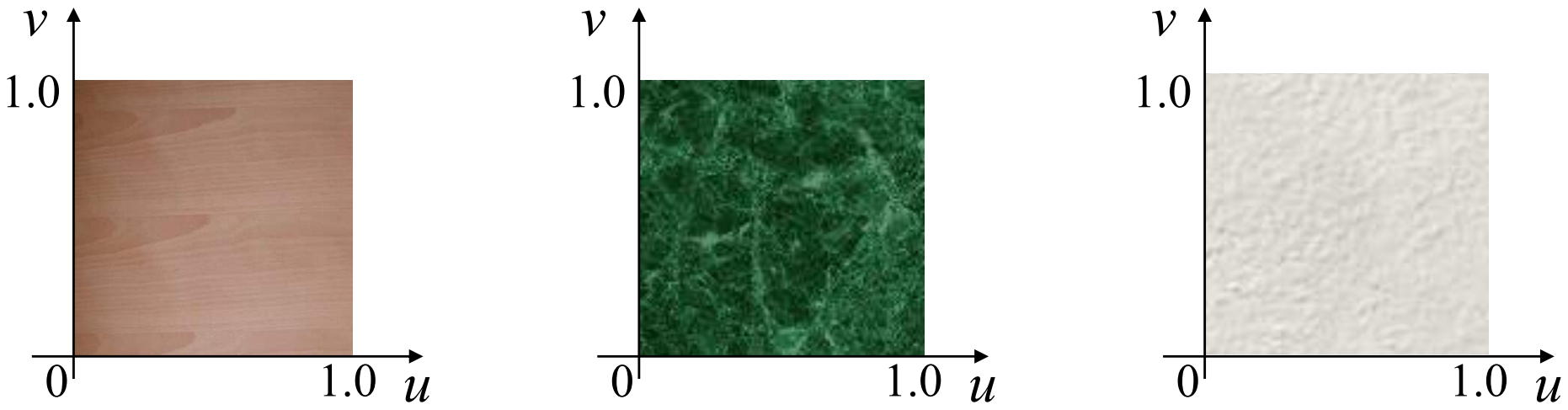
Algoritmo de traçado de raios

```
selecione o centro de projeção(eye) e uma janela no plano de projeção
for (cada pixel da tela)
 {
  determine o raio ray que vai do centro de projeção ao pixel;
  pixel = trace ( ray, 1);
  }
}
```

```
Color  trace (Scene scene, Vector3d eye, Vector3d 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;
}
```

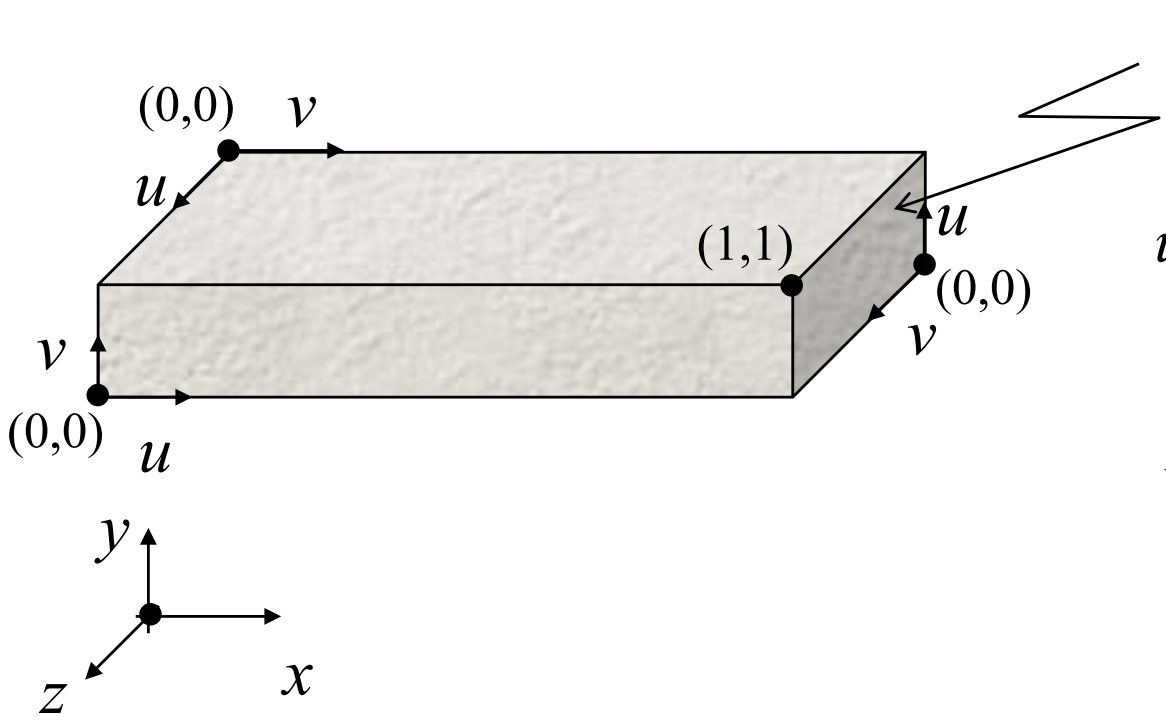
```
Color  shade (Scene scene, Object object, Vector3D ray,
                                  Vector3D point, Vector3D normal,  int  depth)
{
 color = termo ambiente do material do objeto ;

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 (Eq.1) + componente especular (Eq.2)
    }  }

if (depth >= maxDepth)  return color;

if (objeto é refletor) {
    rRay = raio na direção de reflexão;
    rColor = trace(scene, point, rRay, depth+1);
    reduza rColor pelo coeficente de reflexão especular e some a color; }

return color;
}
```

Texturas



Texturas 2D = Imagens onde o domínio é $u, v \in [0,1] \times [0,1] \subset \mathrm{R}^2$

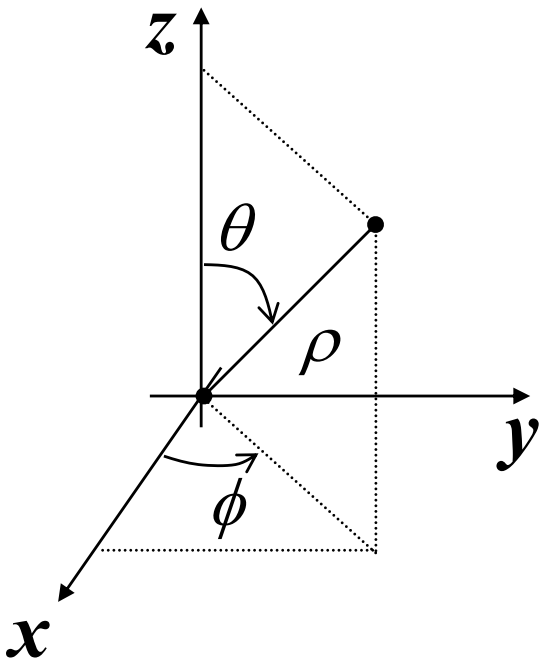
Sistemas de coordenada de textura na caixa



$$u = \frac{y - y_{\min}}{y_{\max} - y_{\min}}$$

$$v = \frac{z - z_{\min}}{z_{\max} - z_{\min}}$$

Sistema de coordenada de textura na esfera



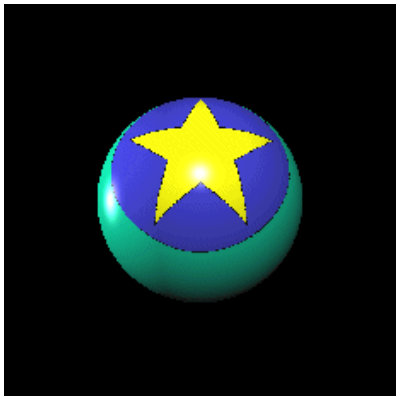
$$z = \rho \cos\theta$$

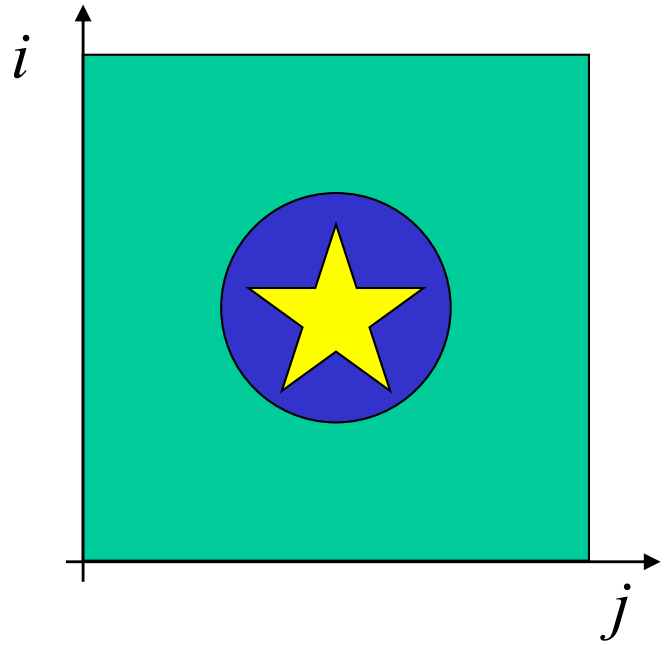
$$x = \rho \sin\theta \cos\phi$$

$$y = \rho \sin\theta \sin\phi$$

$$\tan\phi = y / x$$

$$\tan\theta = \frac{\sqrt{x^2 + y^2}}{z}$$





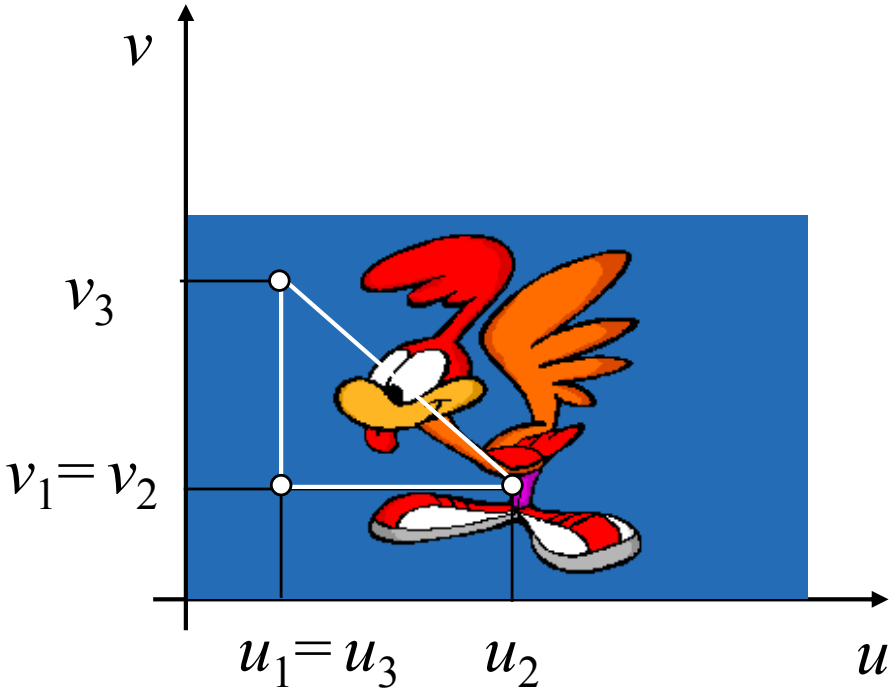
$$u = 1 + \phi/\pi$$

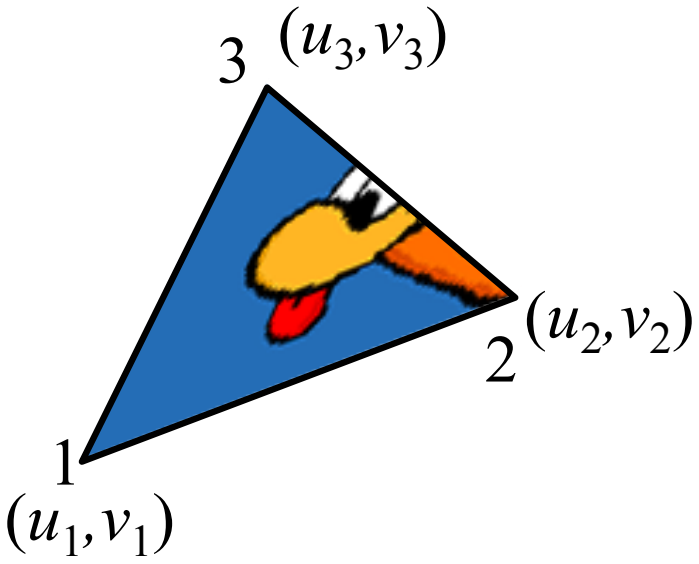
$$v = 1 - \theta/\pi$$

$$i = v(h-1)$$

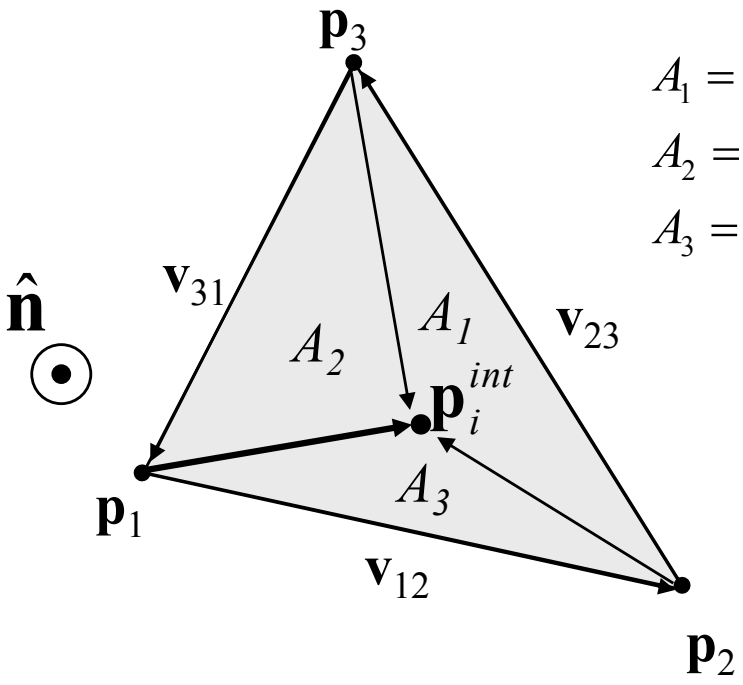
$$j = u(w-1)$$

Sistema de coordenada de textura no triângulo





Textura no triângulo e coordenadas baricêntricas

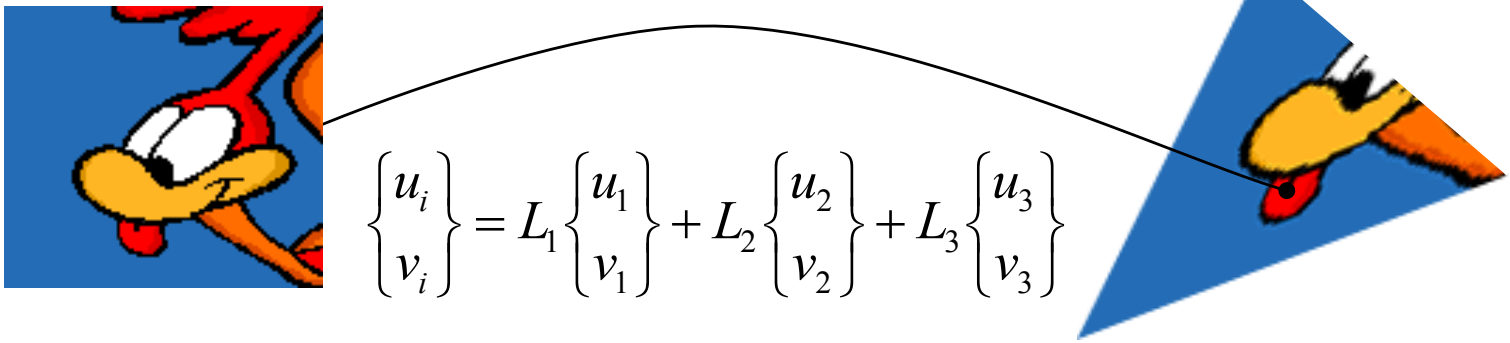


$$A_1 = \hat{\mathbf{n}} \cdot \left(\mathbf{v}_{23} \times (\mathbf{p}_i - \mathbf{p}_2) \right) / 2$$
$$A_2 = \hat{\mathbf{n}} \cdot \left(\mathbf{v}_{31} \times (\mathbf{p}_i - \mathbf{p}_3) \right) / 2$$
$$A_3 = \hat{\mathbf{n}} \cdot \left(\mathbf{v}_{12} \times (\mathbf{p}_i - \mathbf{p}_1) \right) / 2$$

$$A_T = A_1 + A_2 + A_3$$

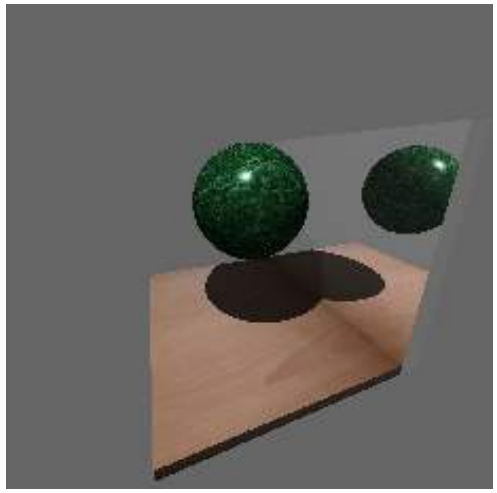
$$L_1 = A_1 / A_T$$
$$L_2 = A_2 / A_T$$
$$L_3 = A_3 / A_T$$

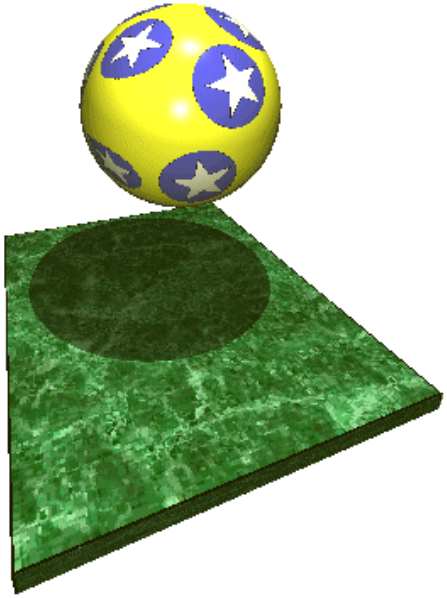
$$L_3 = 1 - (L_1 + L_2)$$

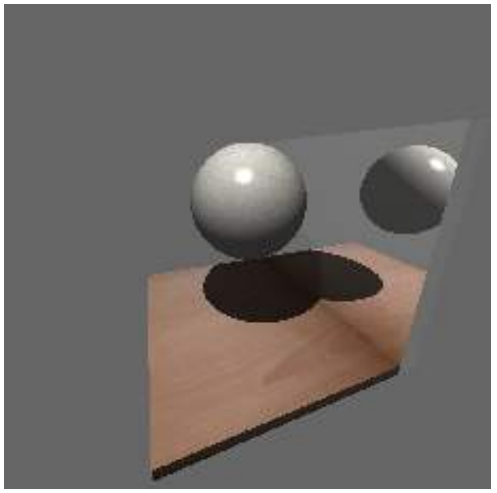


$$\begin{Bmatrix} u_i \\ v_i \end{Bmatrix} = L_1 \begin{Bmatrix} u_1 \\ v_1 \end{Bmatrix} + L_2 \begin{Bmatrix} u_2 \\ v_2 \end{Bmatrix} + L_3 \begin{Bmatrix} u_3 \\ v_3 \end{Bmatrix}$$

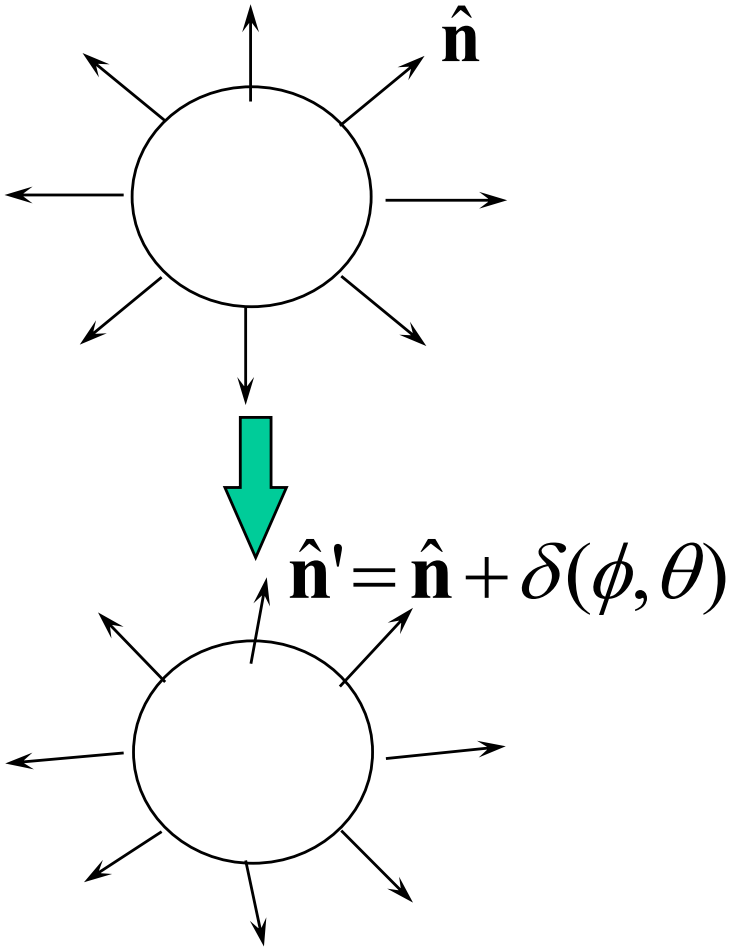
Textura



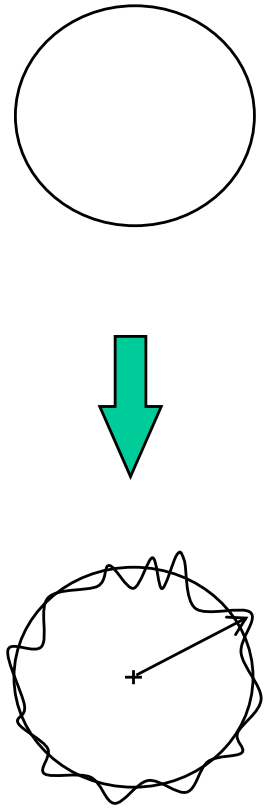




Texturas de rugosidade (*bump textures*) e **Texturas de deslocamentos** (*displacement mapping*)

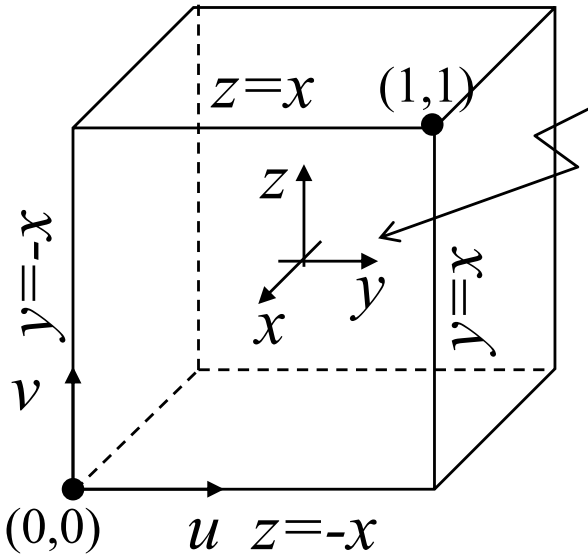


Pertubar aleatóriamente
as normais dos objetos



Pertubar aleatóriamente
as posições dos pontos

Textura ambiente (*environment maps*)

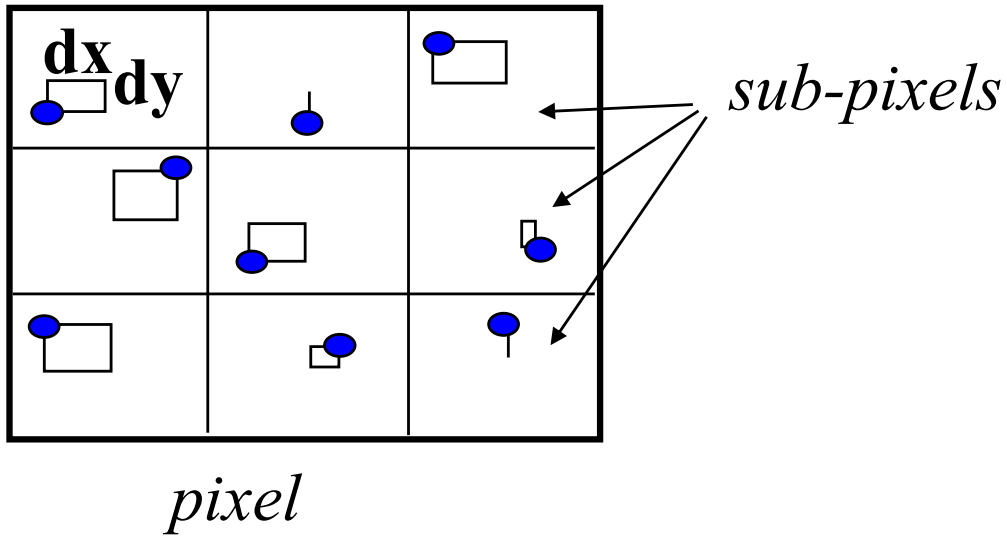


face dos raios x>y e x>z

$$u = \frac{y + x}{2x}$$

$$v = \frac{z + x}{2x}$$

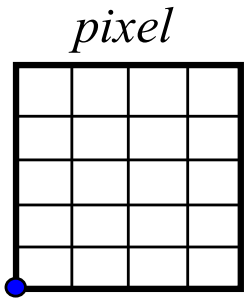
Tratamento anti-alias

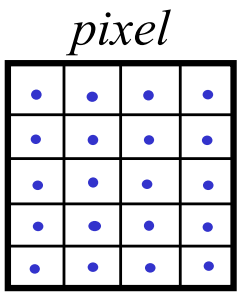


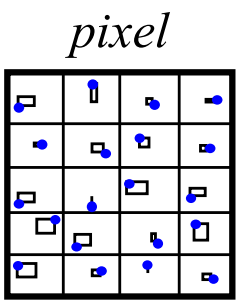
dx, dy = variáveis randômicas

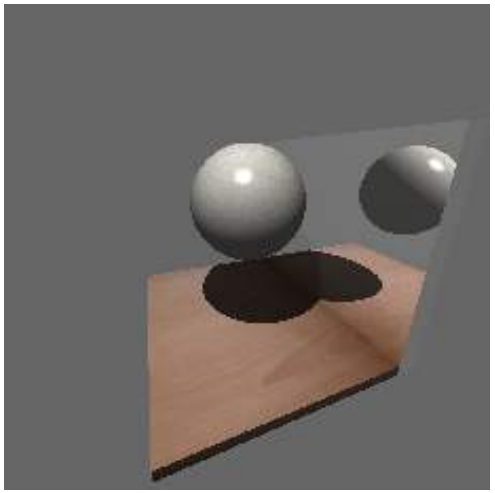
- **Lance um raio para cada sub-*pixel***
- **Faça uma média dos valores obtidos**

Anti-alias

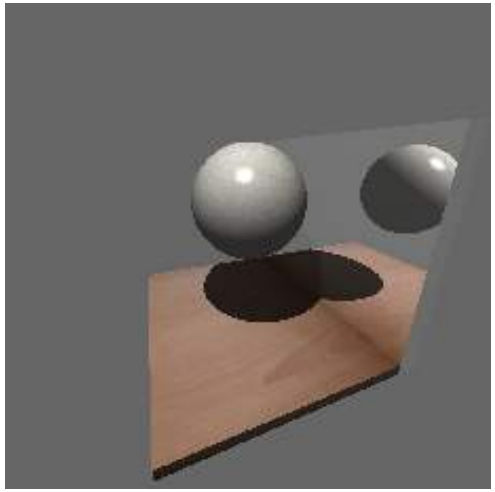




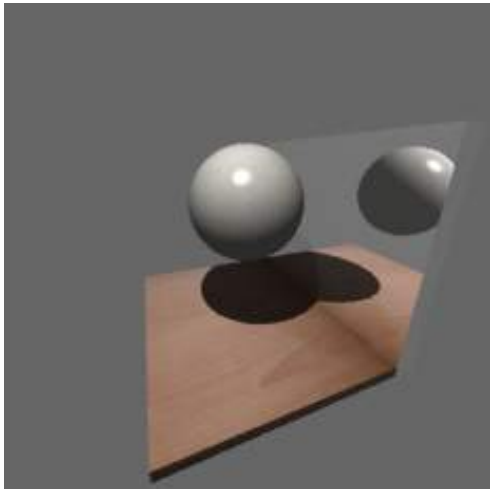




(a) original

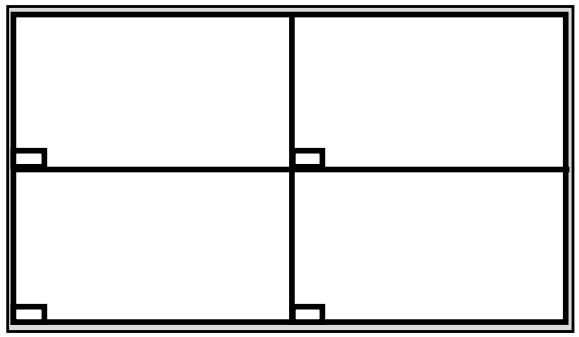


(b) uniforme

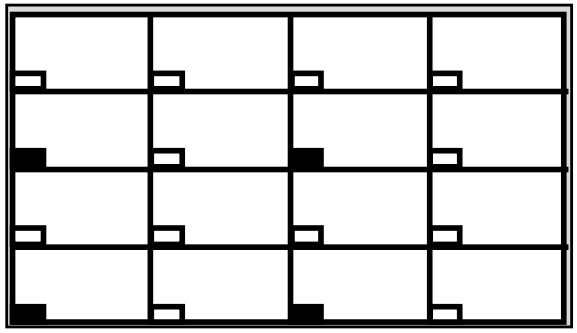


(c) "jittered"

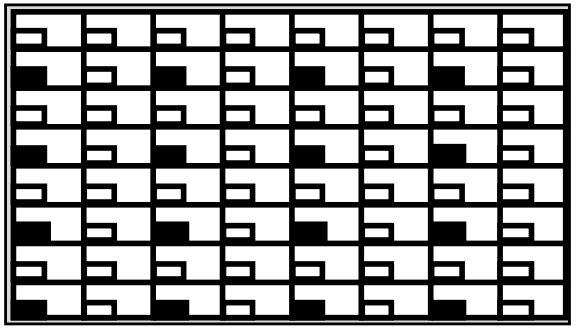
Refinamento Progresivo



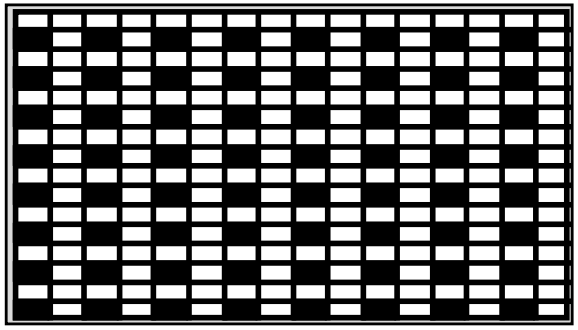
amostragem inicial



primeira subdivisão



segunda subdivisão

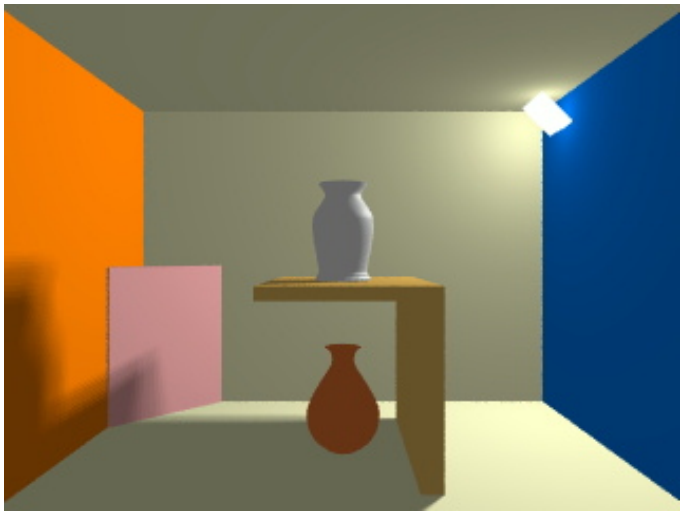


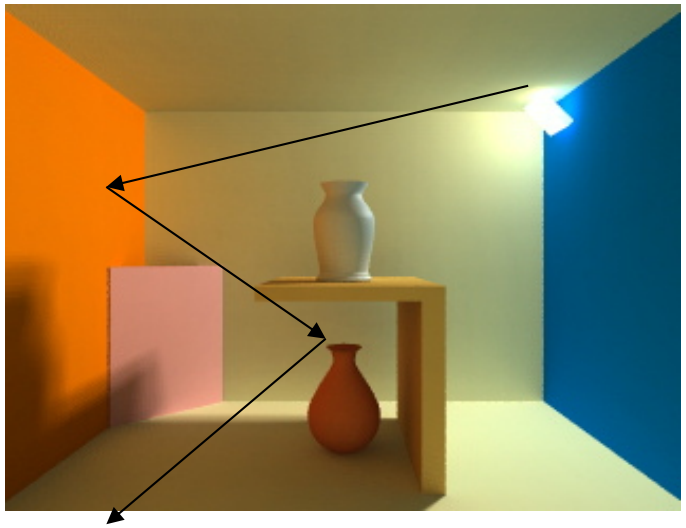
subdivisão final

pixels sendo visitados

pixels já visitados

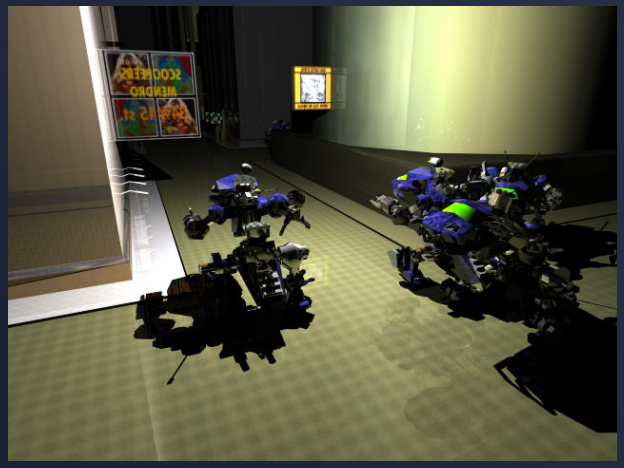
Radiosidade e *Ray Tracing*

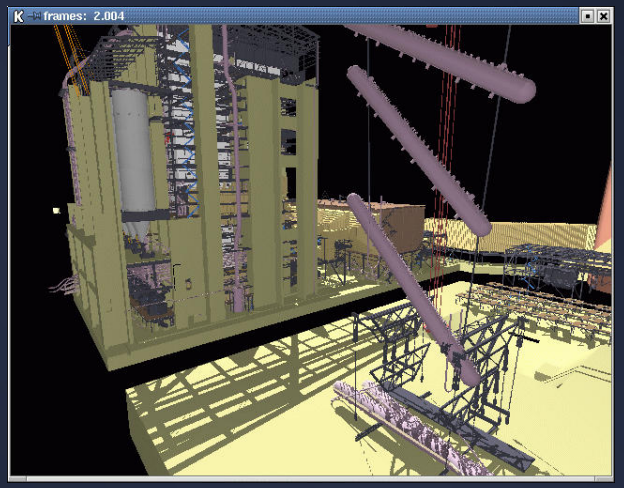


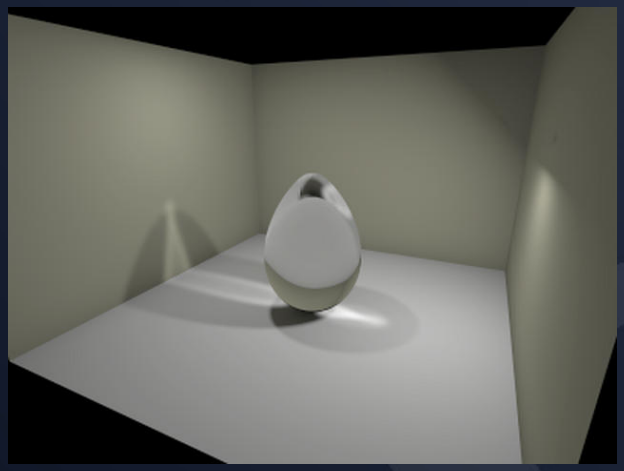


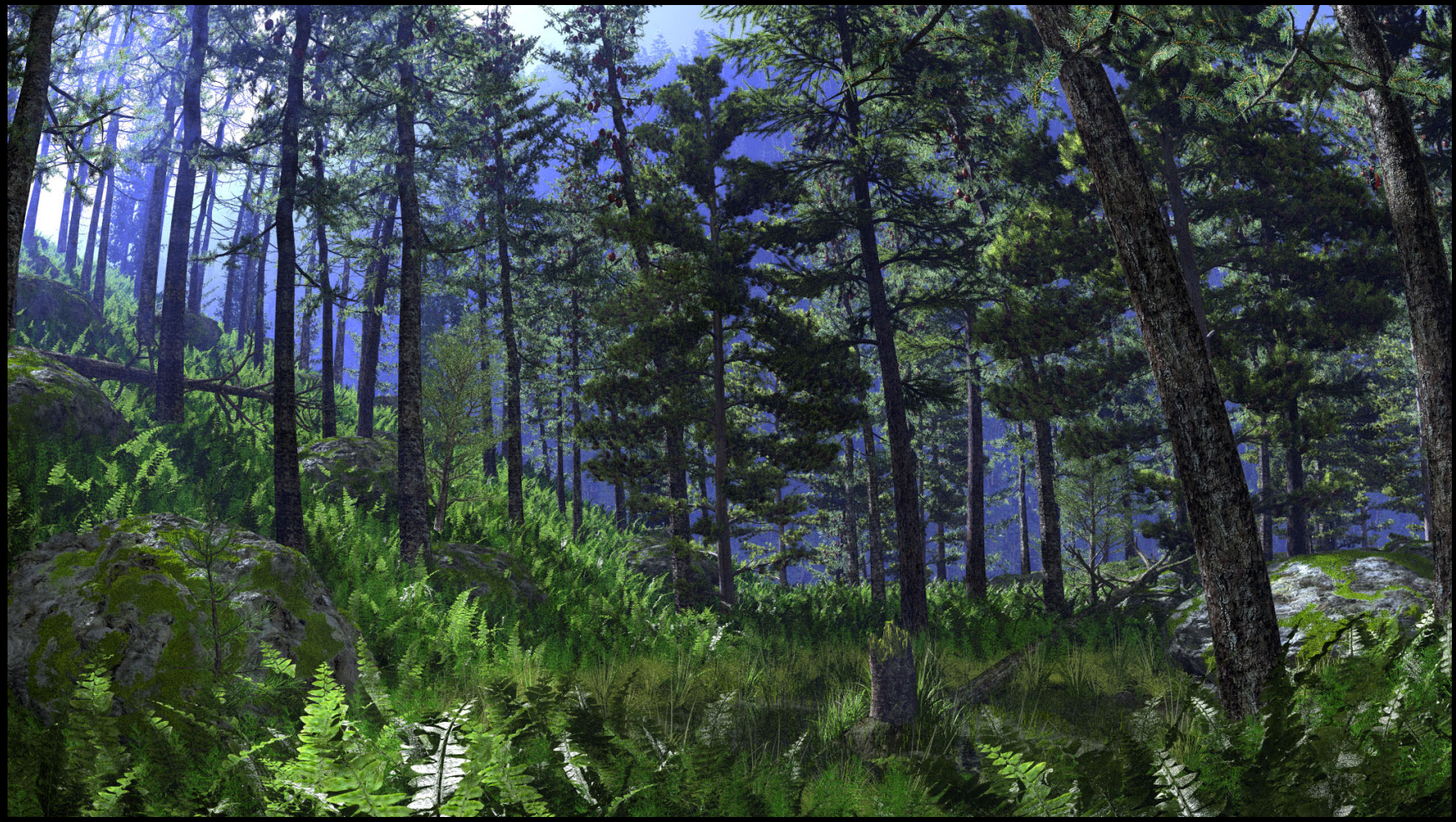
COMPUTER GRAPHIK
UNIVERSITÄT
DES SAARLANDES

Todays State of the Art - Some Snapshots









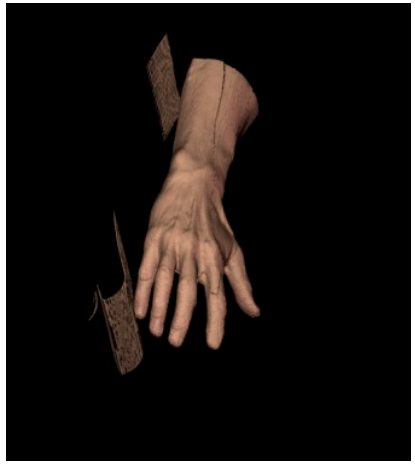

Paskire
Bire-Bire
Bodabida
AUTOBAHN - JVP 2005 - MegaPOV 1.1 - 22"46"

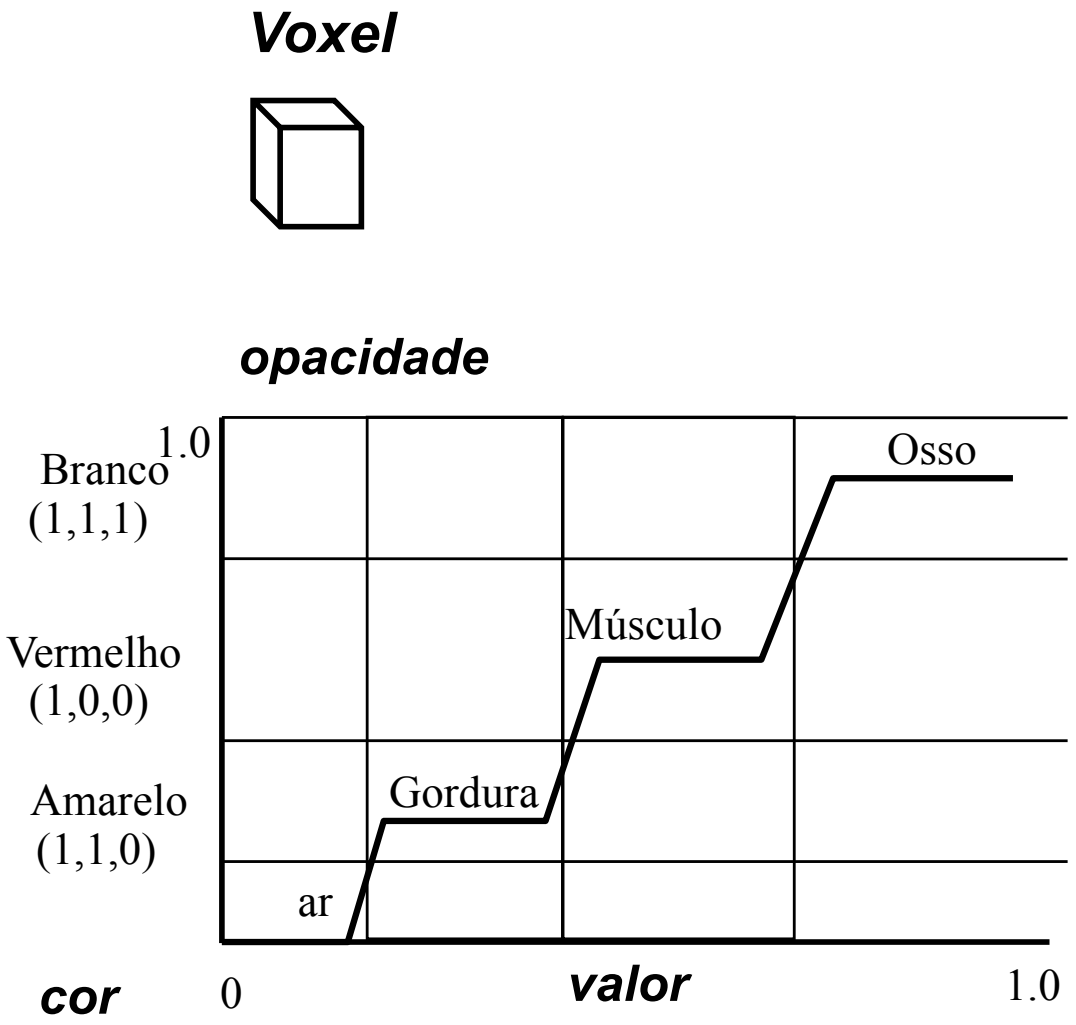
CoolCore
Gilles Tran (c) 2000 www.oyonale.com

Aceleracao do RT

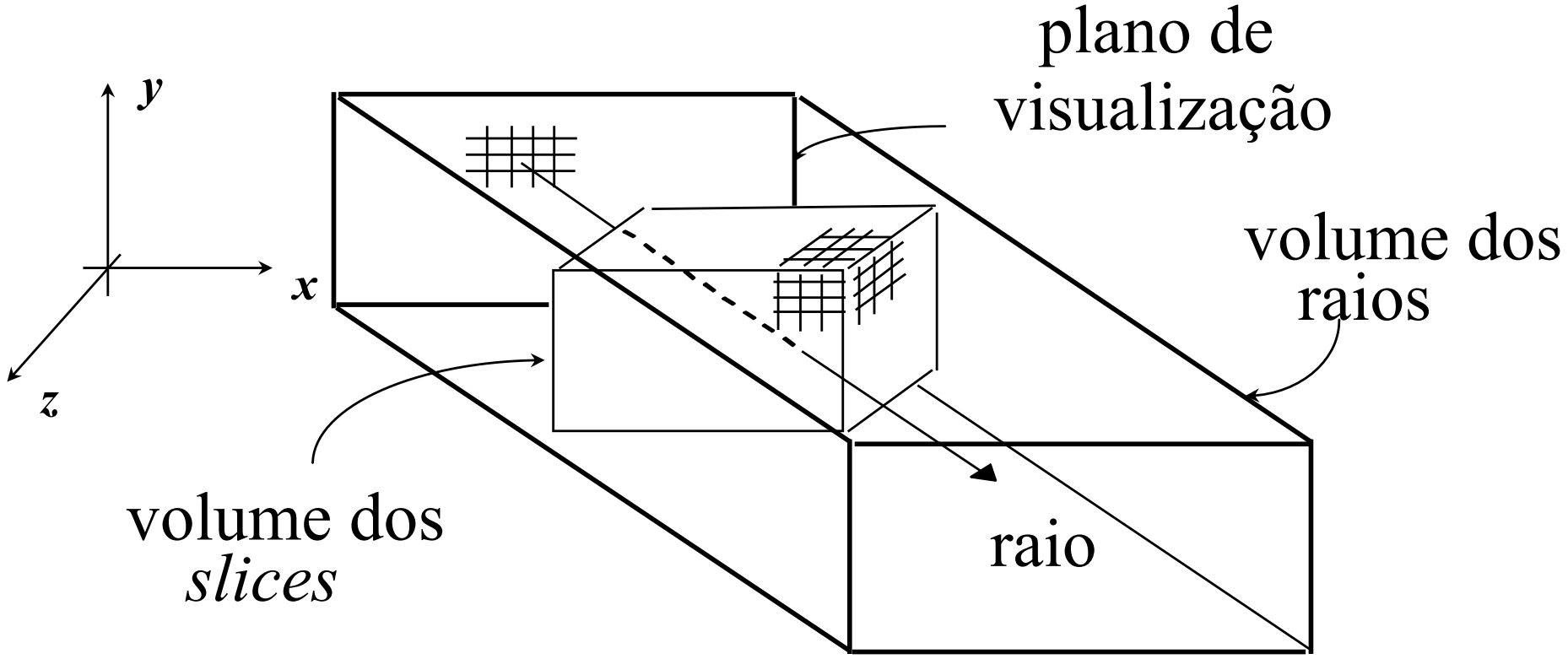
- Cálculos mais eficientes
- Uso de uma Kd Tree

Classificação do Voxel

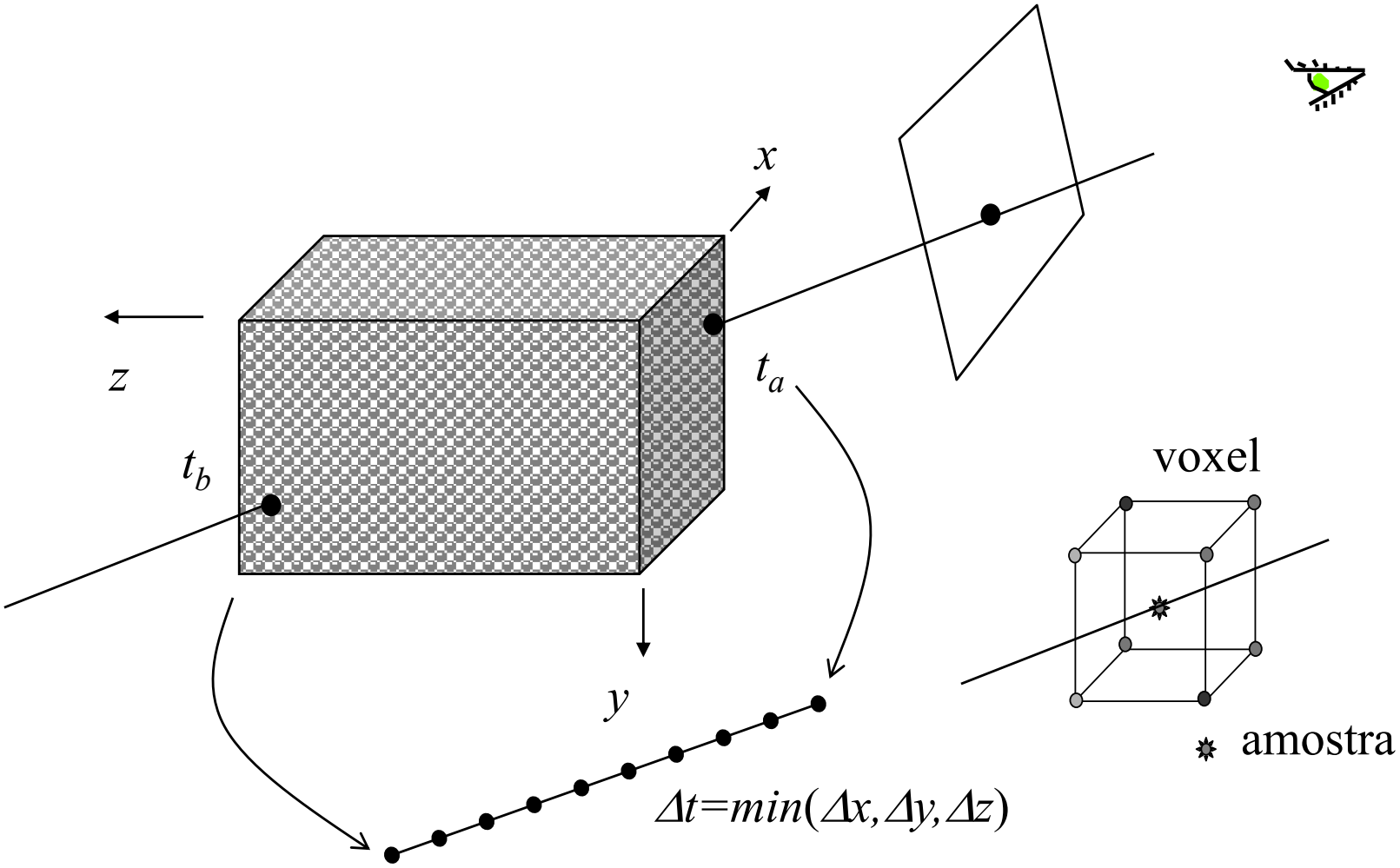




Lançamento dos Raios

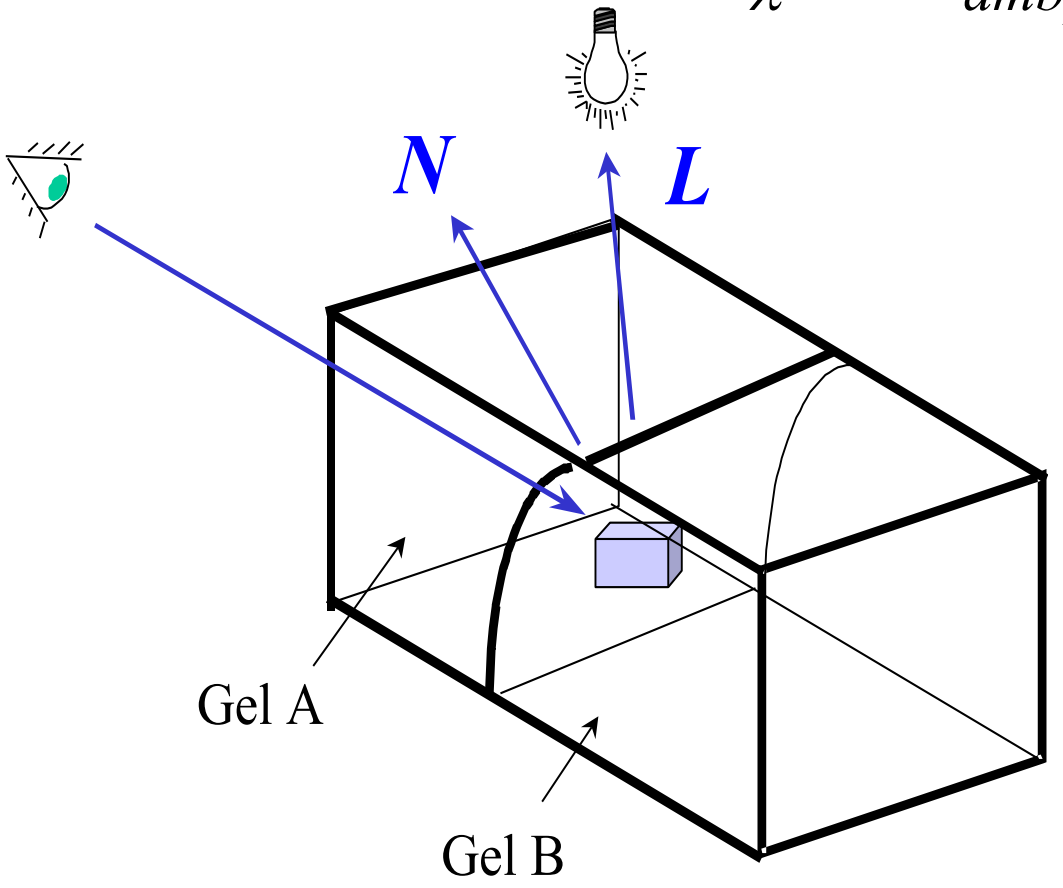


Partição dos Raios



Iluminação de um voxel

$$C_\lambda = C_{amb_\lambda} + C_{luz_\lambda} k_{dif\ \lambda} \left(\hat{\mathbf{n}} \cdot \hat{\mathbf{L}} \right)$$

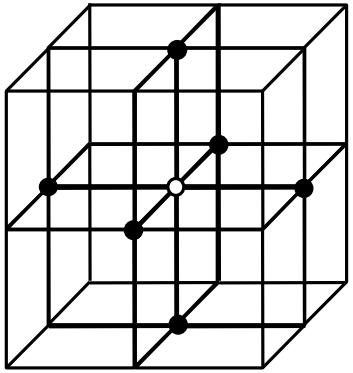


$$N = \nabla V(X)$$

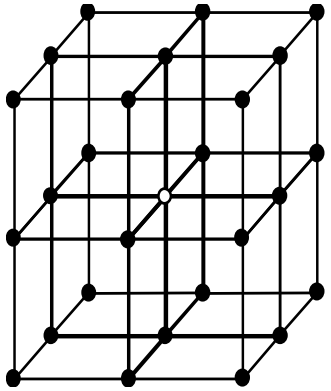
Estimativa do vetor normal

$$N = \frac{\nabla f(x,y,z)}{|\nabla f(x,y,z)|}$$

$$\nabla f(x_i, y_j, z_k) = \begin{Bmatrix} [f(x_{i+1}, y_j, z_k) - f(x_{i-1}, y_j, z_k)]/(2\Delta x), \\ [f(x_i, y_{j+1}, z_k) - f(x_i, y_{j-1}, z_k)]/(2\Delta y), \\ [f(x_i, y_j, z_{k+1}) - f(x_i, y_j, z_{k-1})]/(2\Delta z), \end{Bmatrix}$$

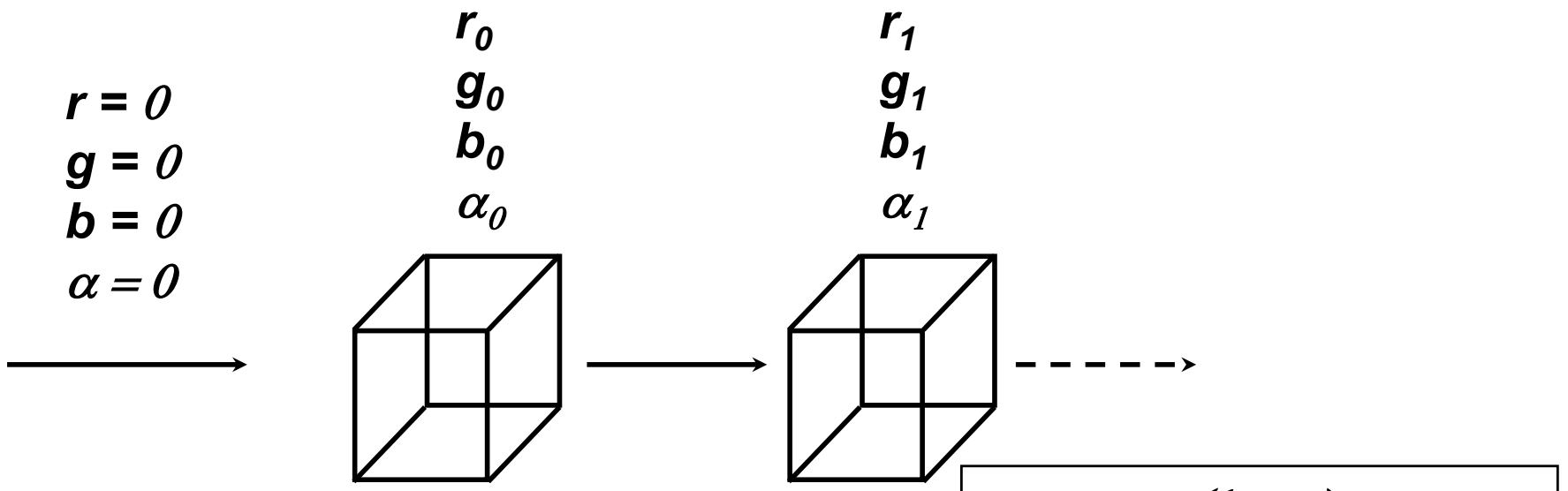


1ª ordem



2ª ordem

Influência de um Voxel

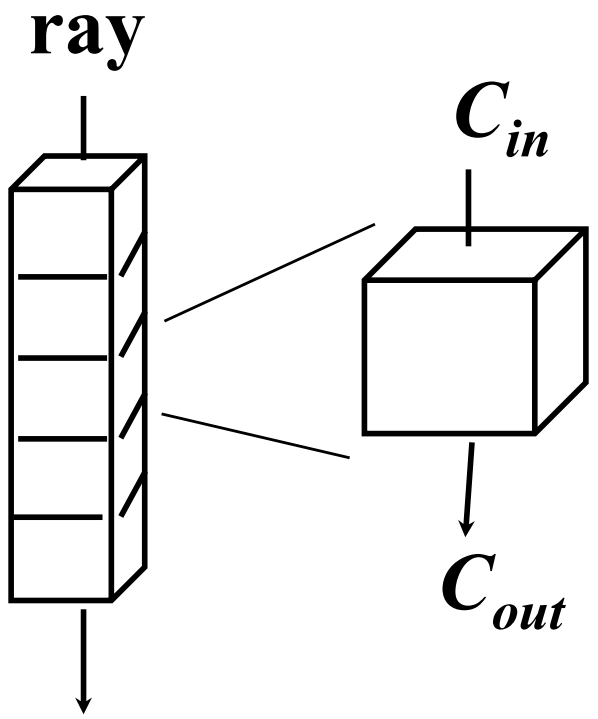


Composição no raio

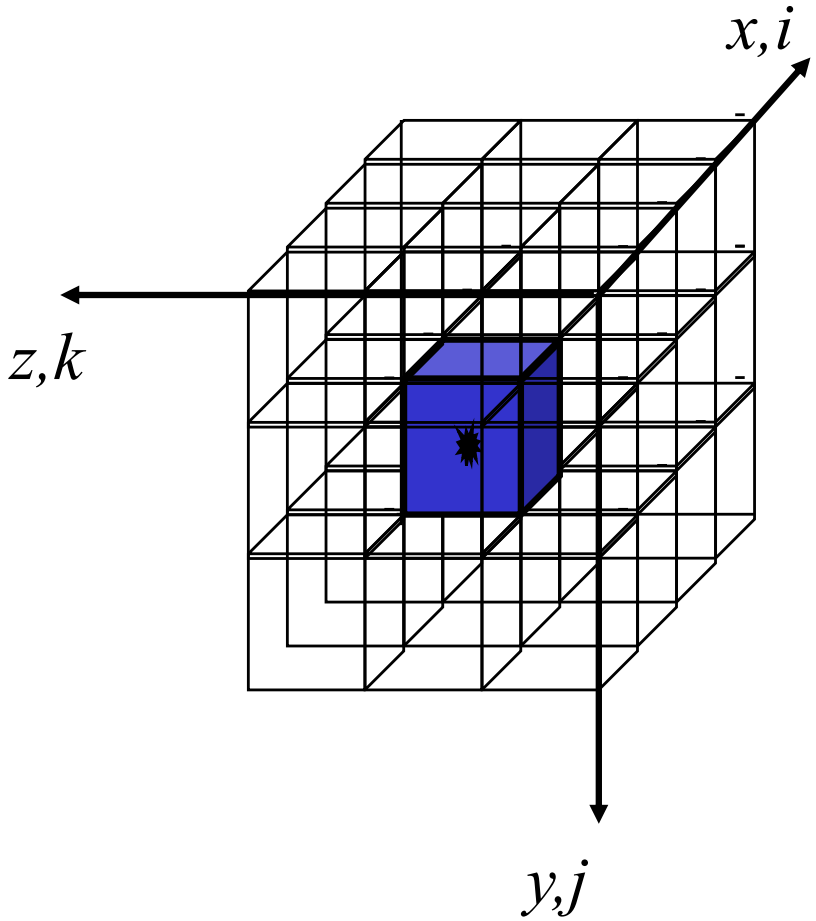
$$\alpha_{out} = \alpha_{in} + (1 - \alpha_{in})\alpha_{\nu}$$

$$C_{out} = C_{in} + (1 - \alpha_{in})\alpha_{\nu} C_{\nu}$$

$$C = C_{out}\ \alpha_{out}$$



Interpolação



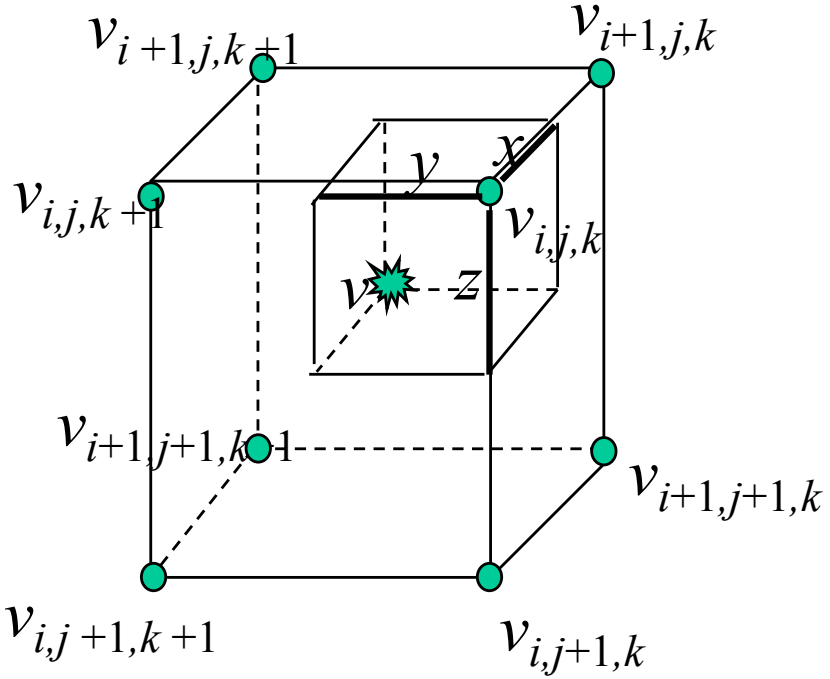
(x_a, y_a, z_a)

$$i = \left\lfloor x_a / Dx \right\rfloor \qquad x = \frac{x_a \,\%\, Dx}{Dx}$$

$$j = \left\lfloor y_a / Dy \right\rfloor \qquad y = \frac{y_a \,\%\, Dy}{Dy}$$

$$k = \left\lfloor z_a / Dz \right\rfloor \qquad z = \frac{z_a \,\%\, Dz}{Dz}$$

Interpolação no voxel



$$
\begin{aligned}
v(x,y,z) = {} & (1-x)(1-y)(1-z)v_{i,j,k} + \\
& (x)(1-y)(1-z)v_{i+1,j,k} + \\
& (1-x)(y)(1-z)v_{i,j+1,k} + \\
& (x)(y)(1-z)v_{i+1,j+1,k} + \\
& (1-x)(1-y)(z)v_{i,j,k+1} + \\
& (x)(1-y)(z)v_{i+1,j,k+1} + \\
& (1-x)(y)(z)v_{i,j+1,k+1} + \\
& (x)(y)(z)v_{i+1,j+1,k+1}
\end{aligned}
$$

Etapa de Composição

- Para cada raio:

  gera amostras de cor $C_\lambda(p_i)$ e opacidades $\alpha(p_i)$

  - reamostragem dos dados dos *voxels*, em k amostras igualmente espaçadas

- Processo de acumulação

  $I = t\, I_b + (1-t) I_0$

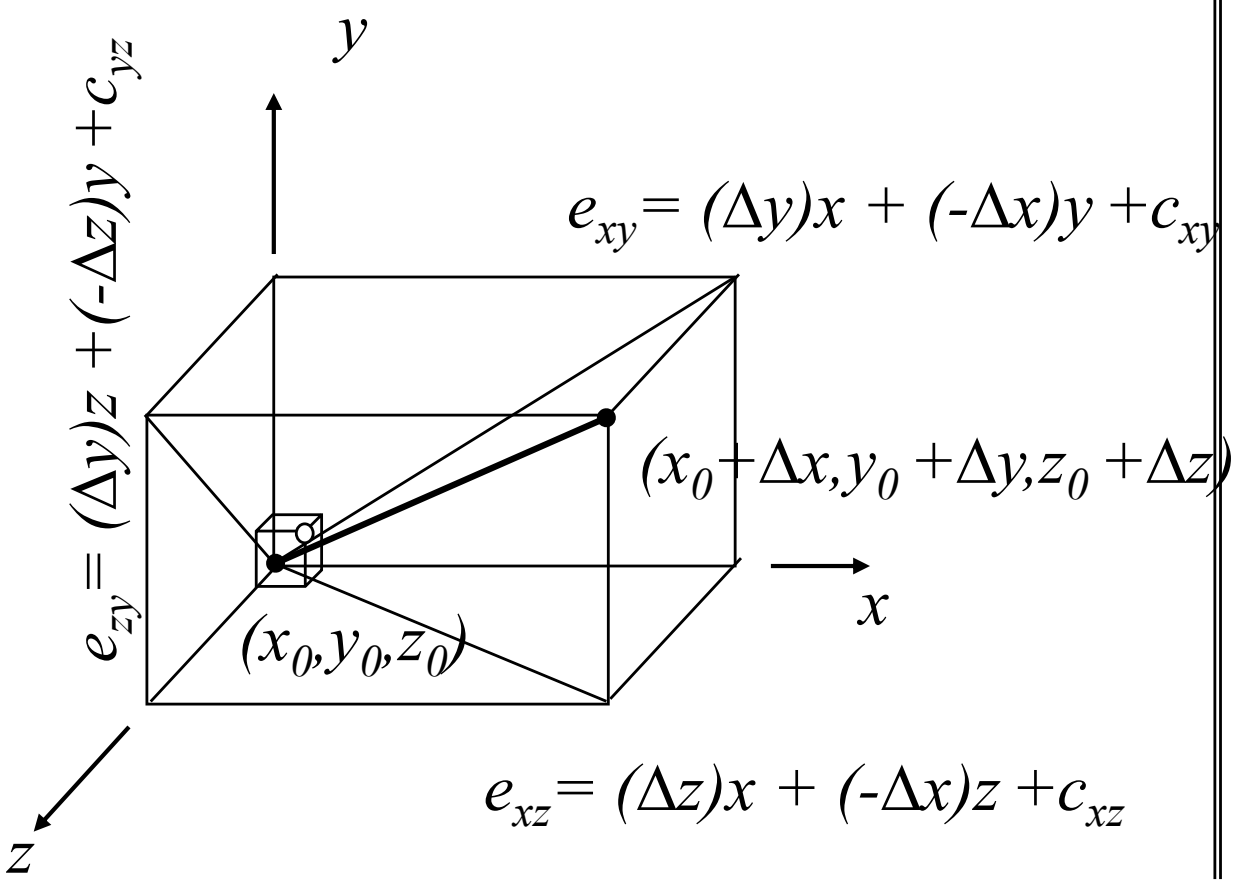
  I_0 = cor do objeto

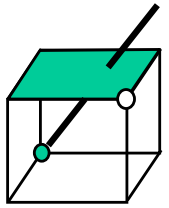
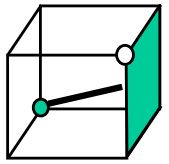
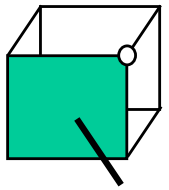
  I_b = cor do fundo

  I = cor resultante

  t = coeficiente de transparência

Tripod

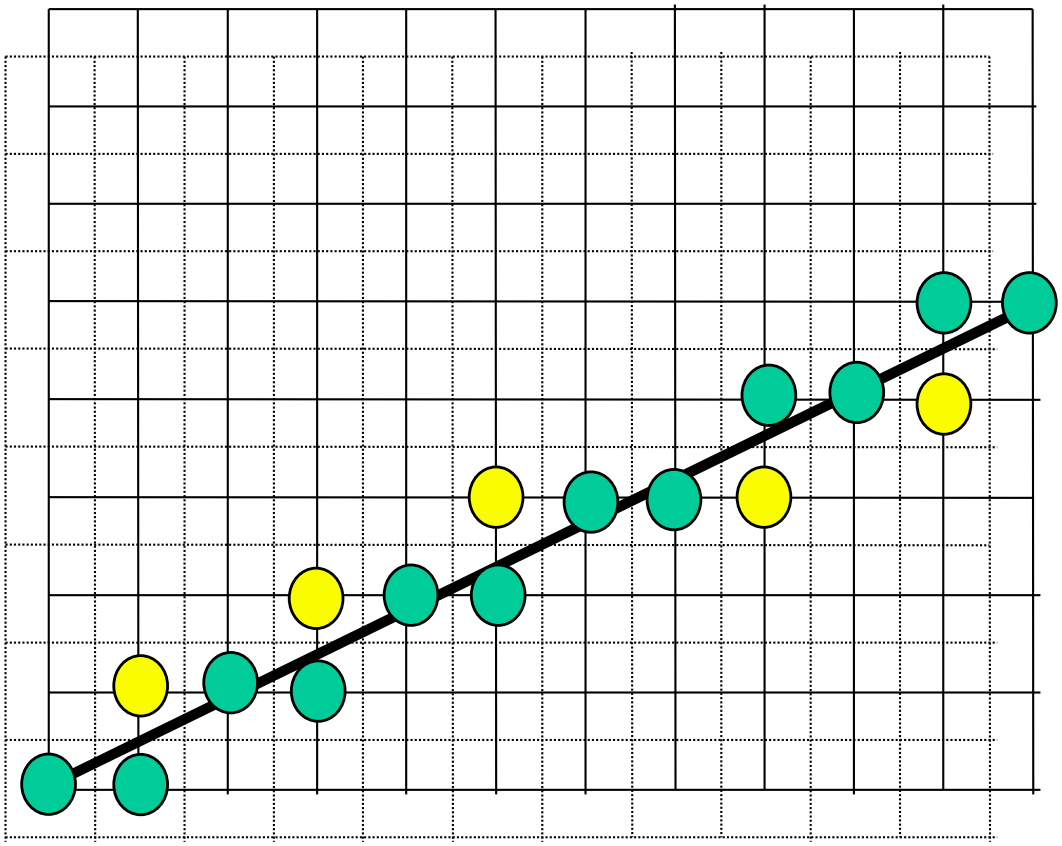


	e_{xy}	e_{xz}	e_{zy}
	+	±	+
	-	-	±
	±	+	-

Valor inicial e incremento

	$(1/2,1/2,1/2)$	$x{+}{+}$	$y{+}{+}$	$z{+}{+}$
$e_{xy} = (\Delta y)x + (-\Delta x)y + c_{xy}$	$(\Delta y - \Delta x)/2$	Δy	$-\Delta x$	
$e_{xz} = (\Delta z)x + (-\Delta x)z + c_{xz}$	$(\Delta z - \Delta x)/2$	Δz		$-\Delta x$
$e_{zy} = (\Delta y)z + (-\Delta z)y + c_{yz}$	$(\Delta y - \Delta z)/2$		$-\Delta z$	Δy

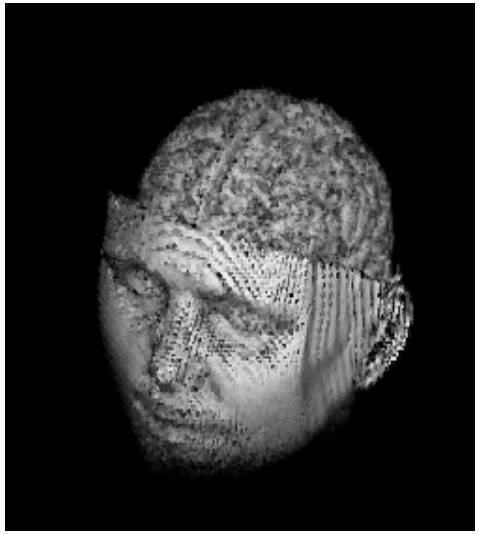
Caminhamento discreto



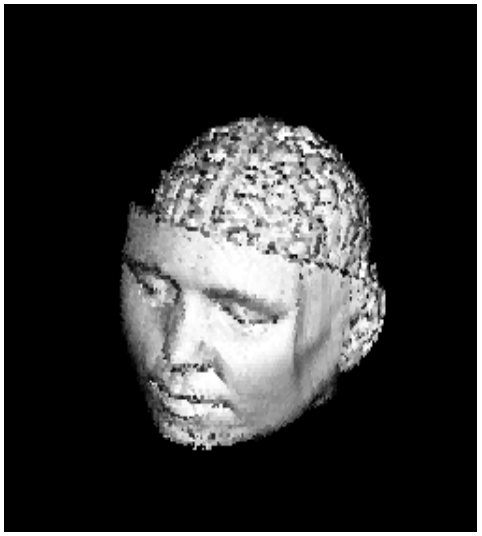
Bresenham

Cohen

Efeito da amostragem

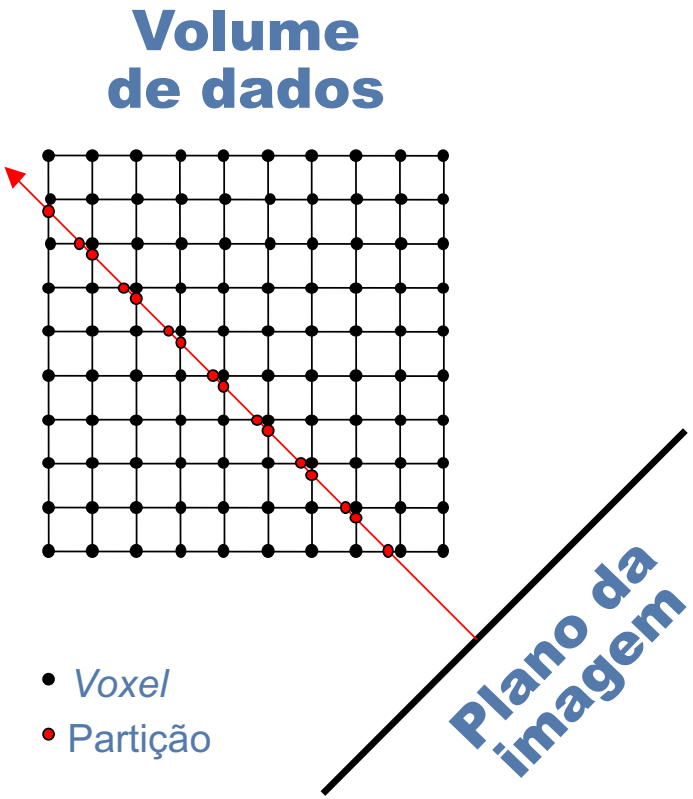


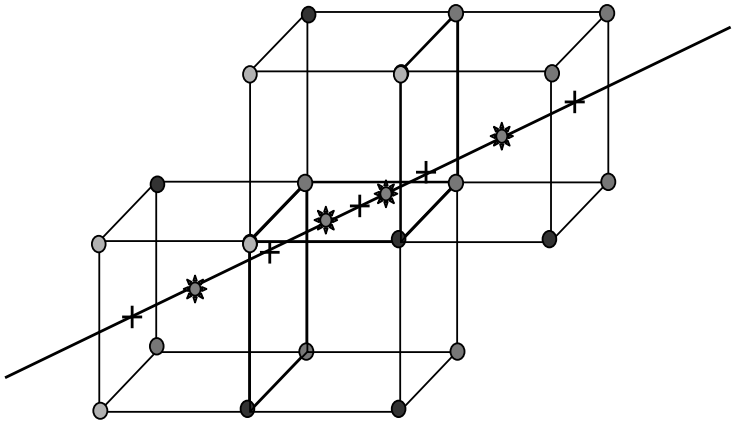
Bresenham



Tripod

Partição celular

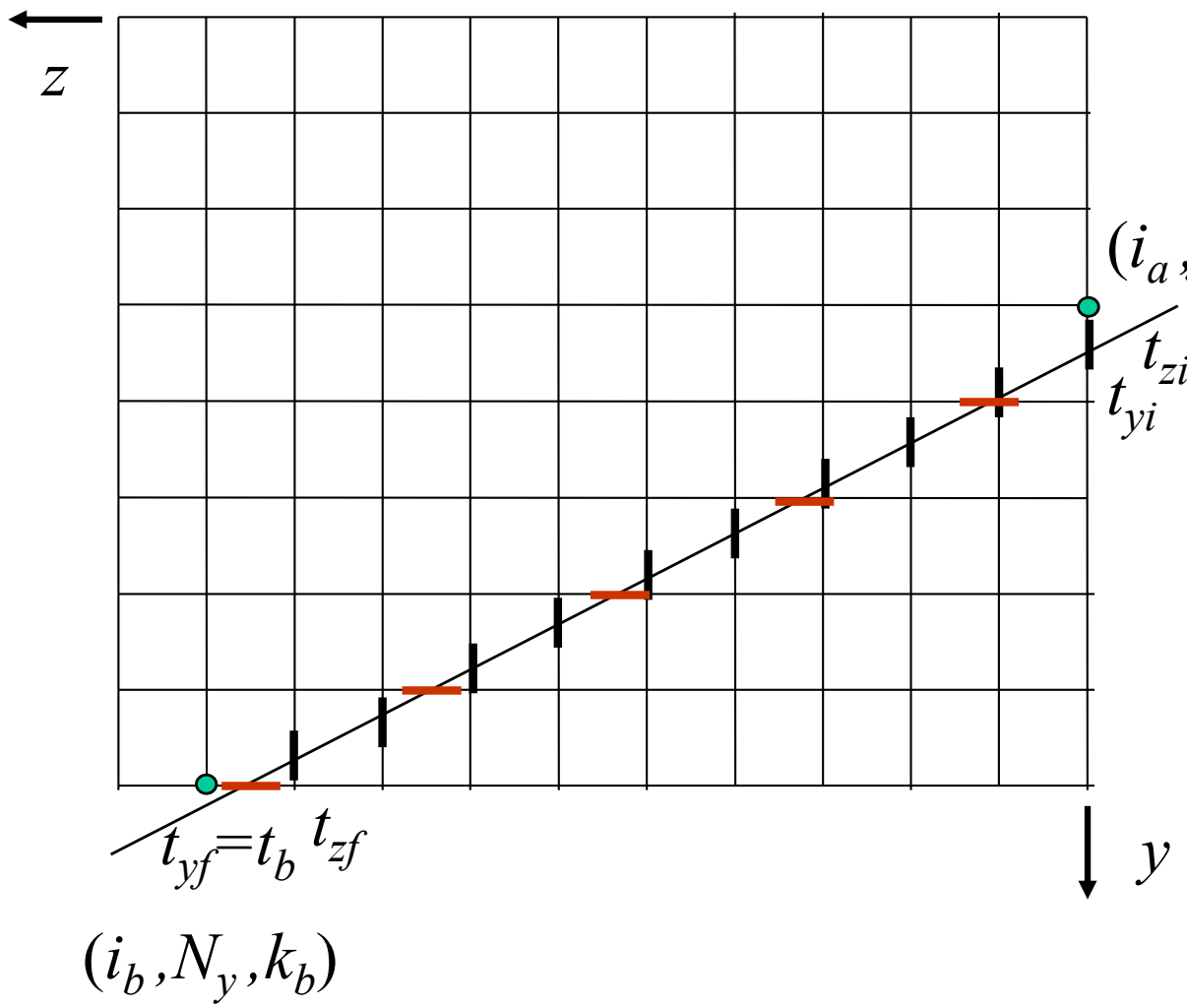




✸ uma amostra no meio da partição

\+ marcador da partição

Partição na grade



$$n_x = (i_b - i_a) - 1$$

$$n_y = (j_b - j_a) - 1$$

$$n_z = (k_b - k_a) - 1$$

$$\Delta t_x = (t_{xf} - t_{xi}) / n_x$$

$$\Delta t_y = (t_{yf} - t_{yi}) / n_y$$

$$\Delta t_z = (t_{zf} - t_{zi}) / n_z$$

Partição celular: algoritmo

Dados: *txi,tyi,tzi, txf,tyf,tzf, nx,ny,nz*

dtx = txf/nx; dty = tyf/ny; dtz = tzf/nz;

tx=txi; ty=tyi; tz=tzi;

t1 = min(*tx,ty,tz*) e *w* é o eixo do mínimo

n=nx+ny+nz;

while (*n > 0*)

tw += dtw;

n--;

t2 = min(*tx,ty,tz*) e *w* é o eixo do mínimo

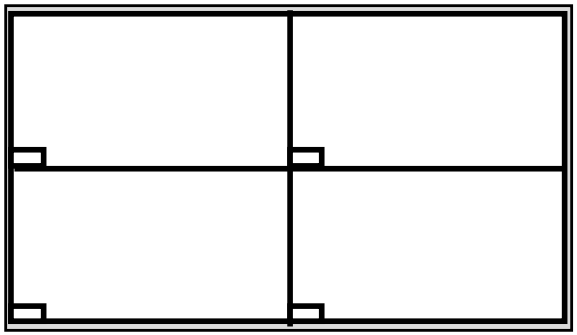
Sample ((*t1+t2)/2*);

t1=t2;

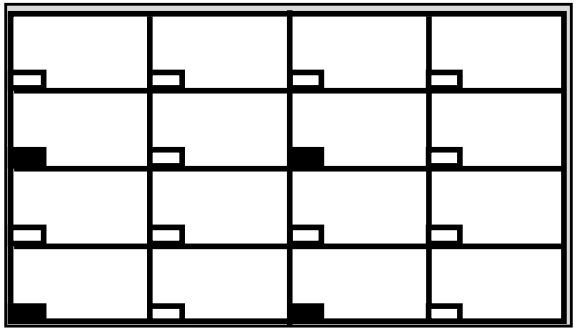
Otimizações

- Velocidade
  - Refinamento progressivo
  - Terminação adaptativa do raio
  - Estruturas Hieráquicas
- Qualidade da imagem
  - aumento do número de amostra no raio
  - lançamento de mais raios
  - melhora esquema de interpolação

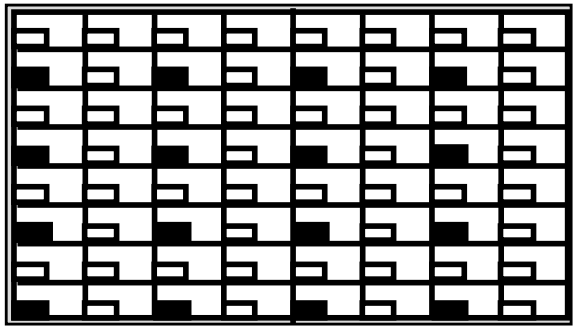
Refinamento Progresivo



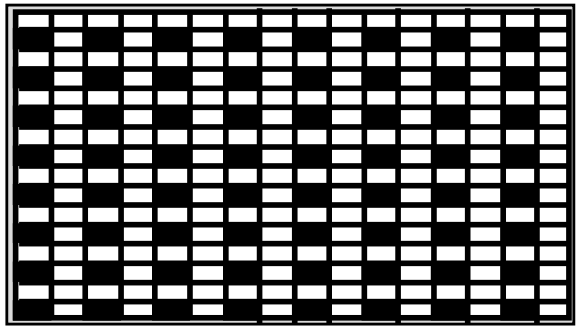
amostragem inicial



primeira subdivisão



segunda subdivisão



subdivisão final

▭ pixels sendo visitados ▬ pixels já visitados

Refinamento Progressivo: Exemplo