# IA725 Computação Gráfica I
# 1° semestre de 2006

Profª. Wu, Shin-Ting
ting@dca.fee.unicamp.br
Bloco A – sala 317

Prof. José Mario De Martino
martino@dca.fee.unicamp.br
Bloco A – sala 317-A

# Agenda

| Aula | Dia | Tema | Projeto |
|---|---|---|---|
| 17 | 09/05/06 | Cor | |
| 18 | 12/05/06 | Cor | |
| 19 | 16/05/06 | Iluminação | |
| 20 | 19/05/06 | Iluminação | |
| 21 | 23/05/06 | Iluminação | |
| 22 | 26/05/06 | Textura | |
| 23 | 30/05/06 | Textura | Versão 0.2 |
| 24 | 02/06/06 | Rasterização | |
| 25 | 06/06/06 | Rasterização | |
| 26 | 09/06/06 | Recorte | |
| 27 | 13/06/06 | Recorte | |
| - | 16/06/06 | | |
| 28 | 20/06/06 | Visibilidade | |
| 29 | 23/06/06 | Visibilidae | |
| 30 | 27/06/06 | **Prova** | Versão 0.3 |
| | 30/junho | | Versão Final |

# Visibilidade

- Referências
  - Computer Graphics Principles and Practice (2nd Edition)
  - J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes
  - Addison-Wesley – 1990

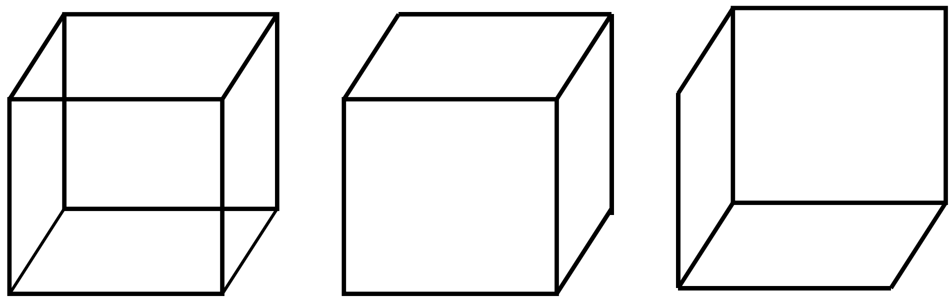
  - Procedural Elements for Computer Graphics
  - D. F. Rogers
  - McGraw-Hill – 1988

  - 3D Computer Graphcis (3rd Edition)
  - Alan Watt
  - Addison-Wesley – 2000

# Visibilidade

- Algoritmos de Remoção de Linhas e Superfícies Escondidas

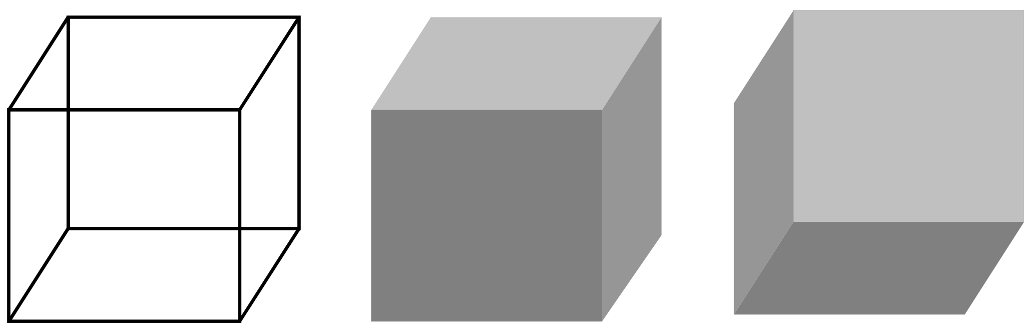
  - Motivação: A visualização *wireframe* de um objeto 3D pode ser ambígua, nem sempre permitindo a clara identificação de qual face está na frente de outra.



  - Os algoritmos de Remoção de Linhas Escondidas (*hidden line*) procuram eliminar esta ambigüidade, removendo as linhas que se encontram escondidas por outras faces

# Algoritmos de Remoção de Linhas e Superfícies Escondidas

- Os algoritmos de Remoção de Superfícies Escondidas (*hidden surface*) procuram remover faces que se encontram atrás de outras.



# Remoção de Linhas Escondidas

- Algoritmo de Roberts (Linhas Escondidas)
  - Remoção das linha escondidas de volumes convexos, cuja superfície é definida/aproximada por polígonos - poliedros convexos.
  - Filosofia:
    - Primeiro: Eliminar as arestas/planos que são escondidas pelo próprio volume.
    - Segundo: Os volumes restantes são comparados para a determinação de quais partes dos volumes são visíveis.

# Algoritmo de Roberts (Linhas Escondidas)

- Representação de um poliedro convexo: interseção de um conjunto de planos.
  - Equação do plano no espaço 3D:

$$ax + by + cz + d = 0$$

  - Forma matricial:

$$\begin{bmatrix} x & y & x & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = 0$$

  - Um poliedro convexo pode ser representado por uma matriz, onde cada coluna representa um dos planos que limita o volume:

$$V = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \\ b_1 & b_2 & & b_n \\ c_1 & c_2 & & c_n \\ d_1 & c_2 & & d_n \end{bmatrix}$$

# Algoritmo de Roberts (Linhas Escondidas)

- O algoritmo de Roberts adota a seguinte convenção:
  - Um ponto interno S = (x,y,z,1) ao volume (poliedro convexo) deve satisfazer a seguinte relação

$$S \cdot V \geq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \text{Ou seja} \qquad \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} a_1 & a_2 & \cdots & a_n \\ b_1 & b_2 & & b_n \\ c_1 & c_2 & & c_n \\ d_1 & c_2 & & d_n \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Ou ainda

$$\begin{bmatrix} a_1 x + b_1 y + c_1 z + d_1 \\ a_2 x + b_2 y + c_2 z + d_2 \\ \vdots \\ a_n x + b_n y + c_n z + d_n \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

- Observação: É necessário que a descrição do poliedro satisfaça esta convenção.

# Algoritmo de Roberts (Linhas Escondidas)

- Exemplo de representação de volume: Cubo de unitário centrado na origem.
  - As equações dos 6 planos que definem este cubo são:

$$x_1 = \frac{1}{2} \quad \Rightarrow \quad x_1 + 0y_1 + 0z_1 - \frac{1}{2} = 0 \quad \text{ou} \quad -x_1 + 0y_1 + 0z_1 + \frac{1}{2} = 0$$

$$x_2 = -\frac{1}{2} \quad \Rightarrow \quad x_2 + 0y_2 + 0z_2 + \frac{1}{2} = 0 \quad \text{ou} \quad -x_2 + 0y_2 + 0z_2 - \frac{1}{2} = 0$$

$$y_3 = \frac{1}{2} \quad \Rightarrow \quad 0x_3 + y_3 + 0z_3 - \frac{1}{2} = 0 \quad \text{ou} \quad 0x_3 - y_3 + 0z_3 + \frac{1}{2} = 0$$

$$y_4 = -\frac{1}{2} \quad \Rightarrow \quad 0x_4 + y_4 + 0z_4 + \frac{1}{2} = 0 \quad \text{ou} \quad 0x_4 - y_4 + 0z_4 - \frac{1}{2} = 0$$

$$z_5 = \frac{1}{2} \quad \Rightarrow \quad 0x_5 + 0y_5 + z_5 - \frac{1}{2} = 0 \quad \text{ou} \quad 0x_5 + 0y_5 - z_5 + \frac{1}{2} = 0$$

$$z_6 = -1/2 \quad \Rightarrow \quad 0x_6 + 0y_6 + z_6 + \frac{1}{2} = 0 \quad \text{ou} \quad 0x_6 + 0y_6 - z_6 - \frac{1}{2} = 0$$

# Algoritmo de Roberts (Linhas Escondidas)

- Considerando que a origem (0, 0, 0,1 ) é um ponto interno ao cubo, as equações dos planos que satisfazem a convenção do algoritmo são:

$$x_1 = \frac{1}{2} \Rightarrow x_1 + 0y_1 + 0z_1 - \frac{1}{2} = 0 \quad \text{ou} \quad \boxed{-x_1 + 0y_1 + 0z_1 + \frac{1}{2} = 0}$$

$$x_2 = -\frac{1}{2} \Rightarrow \boxed{x_2 + 0y_2 + 0z_2 + \frac{1}{2} = 0} \quad \text{ou} \quad -x_2 + 0y_2 + 0z_2 - \frac{1}{2} = 0$$

$$y_3 = \frac{1}{2} \Rightarrow 0x_3 + y_3 + 0z_3 - \frac{1}{2} = 0 \quad \text{ou} \quad \boxed{0x_3 - y_3 + 0z_3 + \frac{1}{2} = 0}$$

$$y_4 = -\frac{1}{2} \Rightarrow \boxed{0x_4 + y_4 + 0z_4 + \frac{1}{2} = 0} \quad \text{ou} \quad 0x_4 - y_4 + 0z_4 - \frac{1}{2} = 0$$

$$z_5 = \frac{1}{2} \Rightarrow 0x_5 + 0y_5 + z_5 - \frac{1}{2} = 0 \quad \text{ou} \quad \boxed{0x_5 + 0y_5 - z_5 + \frac{1}{2} = 0}$$

$$z_6 = -1/2 \Rightarrow \boxed{0x_6 + 0y_6 + z_6 + \frac{1}{2} = 0} \quad \text{ou} \quad 0x_6 + 0y_6 - z_6 - \frac{1}{2} = 0$$

# Algoritmo de Roberts (Linhas Escondidas)

- Uma vez que

$$S \cdot V = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \\ 1/2 \\ 1/2 \\ 1/2 \\ 1/2 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- Observação: Pela convenção, a normal de um plano aponta para o interior do volume.

# Algoritmo de Roberts (Linhas Escondidas)

- O produto escalar de ponto interior e cada vetor que define um plano do cubo deve ser positivo - caso contrário multiplicar o vetor do plano por -1, para atender a convenção do algoritmo de Roberts.

$$S \cdot V = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ -1/2 & 1/2 & -1/2 & 1/2 & -1/2 & 1/2 \end{bmatrix} = \begin{bmatrix} -1/2 \\ 1/2 \\ -1/2 \\ 1/2 \\ -1/2 \\ 1/2 \end{bmatrix}$$

- Assim, o volume em concordância com a representação convencionada pelo algoritmo de Roberts é definido por:

$$\begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 \end{bmatrix}$$

# Algoritmo de Roberts (Linhas Escondidas)

- A determinação da equação dos planos que definem o poliedro convexo pode ser feita, por exemplo, a partir de :
  - 3 pontos não colineares.
  - Da normal ao plano e um ponto no plano.

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício
  - Determinar da equação de um plano a partir de 3 pontos não colineares, $P_1$ $(x_1, y_1, z_1)$, $P_2$ $(x_2, y_2, z_2)$ e $P_3$ $(x_3, y_3, z_3)$.

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício

A equação de um plano é dada por :

$ax + by + cz + d = 0$

Esta equação pode ser normalizada, resultando em :

$a'\,\mathrm{x} + b'\,\mathrm{y} + c'\,\mathrm{z} = \text{-}1$

onde

$$a' = \frac{a}{d} \quad b' = \frac{b}{d} \quad c' = \frac{c}{d}$$

Os 3 pontos devem satisfazer a equação do plano, portanto

$$\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix} \begin{bmatrix} a' \\ b' \\ c' \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

Fazendo

$$X = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}, \quad C = \begin{bmatrix} a' \\ b' \\ c' \end{bmatrix}, \quad D = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

Temos que

$C = X^{-1}\,D$

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício
  - Determinar a equação do plano, a partir de vetor normal ao plano e um ponto no plano

Dada a normal

$$\bar{n} = a\,\bar{i} + b\,\bar{j} + c\,\bar{k}$$

onde :

$\bar{i}$, $\bar{j}$ e $\bar{k}$ são vetores unitários nas direções x, y e z respectivamente.

e um ponto S = (x1, y1, z1) no plano, a equação do plano é dada por

$$a\,x + b\,y + c\,z + d = 0$$

com

$$d = -(a\,x_1 + b\,y_1 + c\,z_1)$$

# Algoritmo de Roberts (Linhas Escondidas)

- A determinação da normal também pode ser efetuada pelo produto vetorial envolvendo os 3 pontos no plano.

- Outra alternativa, é utilizar técnica proposta por M. Newell que calcula/estima a normal para polígonos planares e "quase" planares (Rogers p. 208-209).

$$a = \sum_{i=1}^{n} \left(y_i - y_{(i+1)\bmod n}\right)\left(z_i + z_{(i+1)\bmod n}\right)$$

$$b = \sum_{i=1}^{n} \left(z_i - z_{(i+1)\bmod n}\right)\left(x_i + x_{(i+1)\bmod n}\right)$$

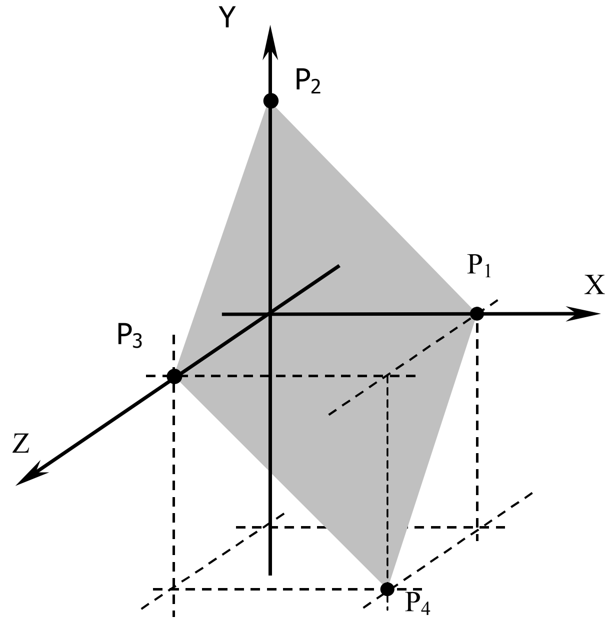
$$c = \sum_{i=1}^{n} \left(x_i - x_{(i+1)\bmod n}\right)\left(y_i + y_{(i+1)\bmod n}\right)$$

onde $n$ é o número de vértices do polígono

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício
  - Determinar a equação do plano definido pelos vértices:

    P1 (1, 0 , 0)
    P2 (0, 1, 0)
    P3 (0, 0, 1)
    P4 (1, -1, 1)




  - Utilizando:
    - Método dos 3 pontos (P1, P2, e P4)
    - Calculando Normal (P1-P2) ⊗ (P3-P1)
    - Calculando Normal Método de M. Newell

# Algoritmo de Roberts (Linhas Escondidas)

- No algoritmo de Roberts, o poliedro convexo é representado pela matriz V que especifica os planos que compõem a face do volume.

$$V = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \\ b_1 & b_2 & & b_n \\ c_1 & c_2 & & c_n \\ d_1 & c_2 & & d_n \end{bmatrix}$$

- Esta matriz pode ser calculada após a transformações de modelagem e visualização (antes da projeção) dos vértices do poliedro, ou ainda com os valores dos vértices antes da transformação. Neste caso, a matriz V deve sofrer a transformação apropriada.

# Algoritmo de Roberts (Linhas Escondidas)

- Transformação de modelagem e visualização do poliedro através da transformação da matriz V.

  - Dado os n vértices do poliedro e uma transformação geométrica T temos

$$P' = T \cdot P$$

onde

$$P' = \begin{bmatrix} x'_1 & x'_2 & \cdots & x'_n \\ y'_1 & y'_2 & & y'_n \\ z'_1 & z'_2 & & z_n \\ 1 & 1 & \cdots & 1 \end{bmatrix} \quad T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad P = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & & y_n \\ z_1 & z_2 & & z_n \\ 1 & 1 & \cdots & 1 \end{bmatrix}$$

  - A transformação acima também pode ser escrita como

$$P'^T = (T \cdot P)^T = P^T \cdot T^T$$

# Algoritmo de Roberts (Linhas Escondidas)

- Dado os n vértices do poliedro é possível escrever a seguinte relação

$$\begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ \vdots & & & \vdots \\ x_n & y_n & z_n & 1 \end{bmatrix} \begin{bmatrix} a_1 & a_2 & \cdots & a_p \\ b_1 & b_2 & & b_p \\ c_1 & c_2 & & c_p \\ d_1 & d_2 & \cdots & d_p \end{bmatrix} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1p} \\ f_{21} & f_{22} & & f_{2p} \\ \cdots & & \ddots & \\ f_{n1} & f_{n2} & & f_{np} \end{bmatrix}$$

Onde

Ponto $(x_i, y_i, z_i)$ é o vértice i do poliedro

$(a_j, b_j, c_j, d_j)$ são os coeficentes do plano j que limita o poliedro

fij é valor da substituição do vértice i na equação do plano j.

- A relação acima pode ser escrita como

$$P^T \cdot V = D$$

# Algoritmo de Roberts (Linhas Escondidas)

- Os pontos transformados devem satisfazer igualmente as equações dos planos transformados, portanto, os pontos transformados e a matriz V transformada satisfazem

$$P'^T \cdot V' = D$$

- Agora

$$P^T \cdot V = D$$

$$P'^T \cdot V' = D$$

Portanto

$$P'^T \cdot V' = P^T \cdot V$$

E como

$$P'^T = P^T \cdot T^T$$

Temos

$$P^T \cdot T^T \cdot V' = P^T \cdot V \quad \Rightarrow \quad V' = \left(T^T\right)^{-1} \cdot V$$

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício

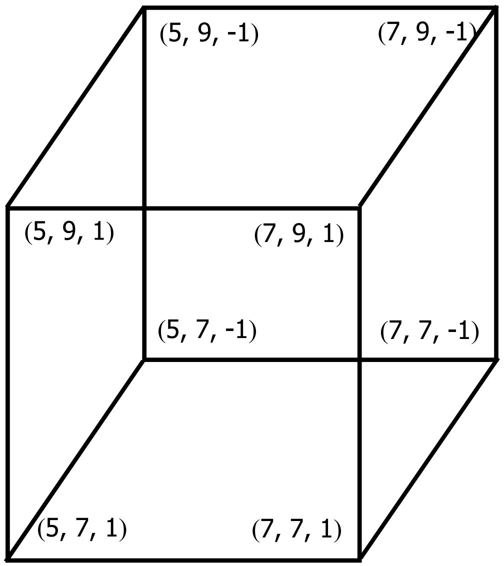
  - Considere um cubo com arestas de 2 unidades centrado na origem que é transladado de 6 unidades na direção x e de 8 unidades na direção y. Calcule a Matriz V':

    a. Utilizando os pontos transformados.

    b. Utilizando os pontos originais e transformando a Matriz V.

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício a




$$V' = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 7 & -5 & 9 & -7 & 1 & 1 \end{bmatrix}$$

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício b

$$V' = \left(T^T\right)^{-1} \cdot V \qquad T = \begin{bmatrix} 1 & 0 & 0 & 6 \\ 0 & 1 & 0 & 8 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 6 & 8 & 0 & 1 \end{bmatrix} \qquad \left(T^T\right)^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -6 & -8 & 0 & 1 \end{bmatrix} \qquad V = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$V' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -6 & -8 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 7 & -5 & 9 & -7 & 1 & 1 \end{bmatrix}$$

# Algoritmo de Roberts (Linhas Escondidas)

- Eliminação de linhas escondidas de um volume por outro volume.
  - Dados os vértices P1 e P2 de um polígono, a aresta pode ser representada por (equação paramétrica):

$$P(t) = P_1 + (P_2 - P_1)\,t \qquad 0 \le t \le 1$$

ou ainda

$$\bar{v} = \bar{s} + \bar{d}\,t$$

*onde* :

$\bar{v} = P(t)$ é vetor posição sobre a aresta

$\bar{s} = P_1$ é o vetor posição inicial

$\bar{d} = (P_2 - P_1)$ é o vetor direção da aresta

# Algoritmo de Roberts (Linhas Escondidas)

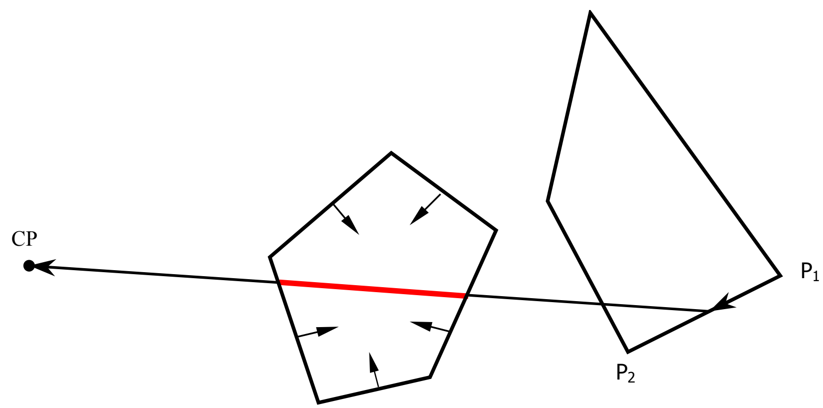
- A equação paramétrica da (semi)reta unindo o ponto sobre a aresta e o centro de projeção é:

$$\bar{u} = \bar{v} + \bar{g}\alpha = \bar{s} + \bar{d}\, t + \bar{g}\alpha \quad 0 \le t \le 1, \quad \alpha \ge 0$$

onde :

$\bar{g}$ é o vetor direção do ponto ao centro de projeção.

- Se algum ponto da semi-reta $\bar{u}$ que une o ponto da aresta com o centro de projeção estiver dentro de algum volume, o ponto da aresta é invisível, já que esta atrás deste volume.



# Algoritmo de Roberts (Linhas Escondidas)

- Um ponto está dentro de um volume quando o produto escalar com a matriz V (após a devida transformação de visualização) que descreve o volume é positivo (por construção):

$$h = \bar{u} \cdot V = \bar{s} \cdot V + t\,\bar{d} \cdot V + \alpha\,\bar{g} \cdot V > 0 \qquad 0 \leq t \leq 1, \quad \alpha \geq 0$$

Ou ainda

$$h = p + t\,q + \alpha\,w > 0$$

*com*

$$p = \bar{s} \cdot V$$

$$q = \bar{d} \cdot V$$

$$w = \bar{g} \cdot V$$

  - Observação: h, p, q e w são matrizes 1xn. Onde n é a quantidade de planos que definem o volume.

# Algoritmo de Roberts (Linhas Escondidas)

- Assim, temos n inequações

$$h_j = p_j + t\, q_j + \alpha\, w_j > 0 \quad j = 1, 2 \cdots, n \quad 0 \le t \le 1, \quad \alpha \ge 0$$

  - Valores de t e $\alpha$ que satisfaçam simultaneamente as inequações $\Rightarrow$ pontos invisíveis.

  - Observação $\alpha \ge 0$ (direção ao observador).

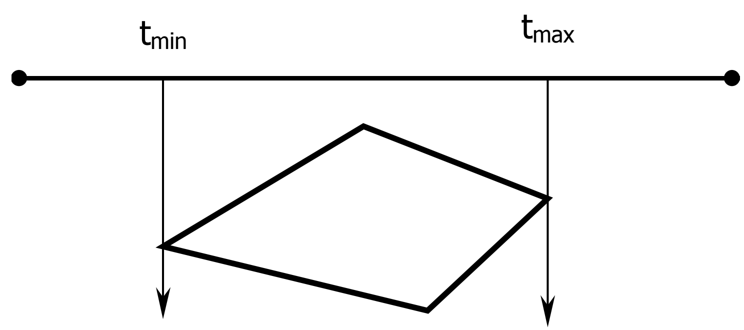
# Algoritmo de Roberts (Linhas Escondidas)

- Método de solução:

  - Resolver duas as duas as equações associadas. Resolver ((n-1) * n) /2 equações. Checar cada solução contra as outras inequações solução válida

$$h_j > 0 \quad \forall j, \quad 0 \le t \le 1, \quad \alpha \ge 0$$

  - Pegar o valor mínimo e o valor máximo de t.



# Algoritmo de Roberts (Linhas Escondidas)

- Exercício
  - Calcular trecho da aresta entre os pontos P1 = [-2 0 -2 1] e P2 = [2 0 -2 1] que se encontra escondido por um cubo unitário centrado na origem. Considerar um projeção ortográfica com centro de projeção em $z = \infty$.

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício
  - Calcular trecho da aresta entre os pontos P1 = [-2 0 -2 1] e P2 = [2 0 -2 1] que se encontra escondido por um cubo unitário centrado na origem que foi rodado 30º no sentido anti-horário em torno do eixo Y. Considerar um projeção ortográfica com centro de projeção em $z = \infty$.

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício
  - Calcular trecho da aresta entre os pontos P1 = [1 0 -1 1] e P2 = [0 0 -1 1] que se encontra escondido por um cubo unitário centrado na origem. Considerar um projeção ortográfica com centro de projeção em $z = \infty$.

# Algoritmo de Roberts (Linhas Escondidas)

- Exercício
  - Calcular trecho da aresta entre os pontos P1 = [1 0 2 1] e P2 = [-1 0 -2 1] que se encontra escondido por um cubo unitário centrado na origem. Considerar um projeção ortográfica com centro de projeção em $z = \infty$.

# Remoção de Superfícies Escondidas

- Algoritmo Z-Buffer (Superfícies Escondidas)
  - O algoritmo Z-Buffer define uma estratégia para a eliminação de superfícies escondidas de fácil implementação em *hardware.*
  - O Z-Buffer é uma extensão do *Framebuffer*.
  - O *Framebuffer* é uma memória que contém para cada *pixel* o valor de sua cor/ intensidade.
  - O Z-Buffer, além do valor da cor/intensidade, permite o armazenamento também da informação de profundidade (distância ao observador localizado no eixo z) do elemento visível naquele *pixel*.

# Algoritmo Z-Buffer (Superfícies Escondidas)

- Observar que o cálculo da profundidade também pode ser efetuado de forma incremental:

$$ax + by + cz + d = 0$$

$$z = \frac{-(ax + by + d)}{c} \quad com \quad c \neq 0$$

Para y = constante (linha de varredura)

a profundidade do pixel no ponto $x_1 = x + \Delta x$ é

$$z_1 - z = \frac{-(ax_1 + by + d)}{c} + \frac{(ax + by + d)}{c} = \frac{-a(x_1 - x)}{c} = -\frac{a}{c}\Delta x$$

ou seja

$$z_1 = z - \frac{a}{c}\Delta x$$

Considerando $\Delta x = 1$ temos

$$z_1 = z - \frac{a}{c}$$

# Algoritmo Z-Buffer (Superfícies Escondidas)

- Algoritmo

1. Inicializar framebuffer com cor de fundo.
2. Inicializar z-buffer com valor mínimo de Z (mais distante).
3. “Rasterizar” os polígonos em qualquer ordem.
4. Para cada pixel (x,y) de um polígono calcular a profundidade z(x,y)
5. Comparar o z calculado com o valor já armazenado no z-buffer:
   - Se z(x,y) > Zbuffer(x,y) armazena a cor do polígono no framebuffer e armazene a profundidade z(x,y) do polígono no z-buffer.
   - Caso contrário manter framebuffer e z-buffer inalterados.

# Algoritmo Z-Buffer (Superfícies Escondidas)

- Observação finais:

  - Vantagens:
    - Simplicidade - fácil implementação em hardware.
    - Trata interseções complexas.
    - Não há limitação à complexidade da cena.
    - Não há necessidade de ordenação.

  - Desvantagens:
    - Tamanho da memória.
    - Não trata transparência.
    - Força bruta - não é aconselhável a implementação em software.

# Remoção de Superfícies Escondidas

- Algoritmo de Pintor (*list priority ou Painter's algorithm)*
  - Primeiro efetua uma ordenação em Z (ordenação por profundidade - distância ao observador localizado no eixo z) dos polígonos que compõem a cena.
  - Começando pelo polígono mais distante do observador até o mais próximo, cada um deles é convertidos em *pixels* ("rasterizado") e colocado no *framebuffer*. Observe que a informação do elemento mais próximo do observador pode substituir informação de elementos mais distantes.
  - O nome do algoritmo advém da analogia do processo de pintura - primeiro os objetos/paisagem mais distantes são pintados, em cima destes os objetos mais próximos.

# Remoção de Superfícies Escondidas

- Ray-casting (superfícies escondidas)

  - O algoritmo de Ray-casting baseia-se na observação que um determinado *pixel* receberá a cor do objeto que se encontra mais próximo do observador. Assim, a partir do observador, raios são lançados através dos *pixels* que definem a imagem. Para cada raio é calculada a interseção com os objetos da cena. O objeto que apresente a menor distância (distância entre o ponto de interseção e observador) é o objeto visível naquele pixel.

# Ray-casting (superfícies escondidas)

- A simplicidade e generalidade do Ray-casting é uma de suas principais vantagens. A operação básica da técnica é o cálculo da interseção de uma reta (raio) e as primitivas geométricas que compõem a cena.

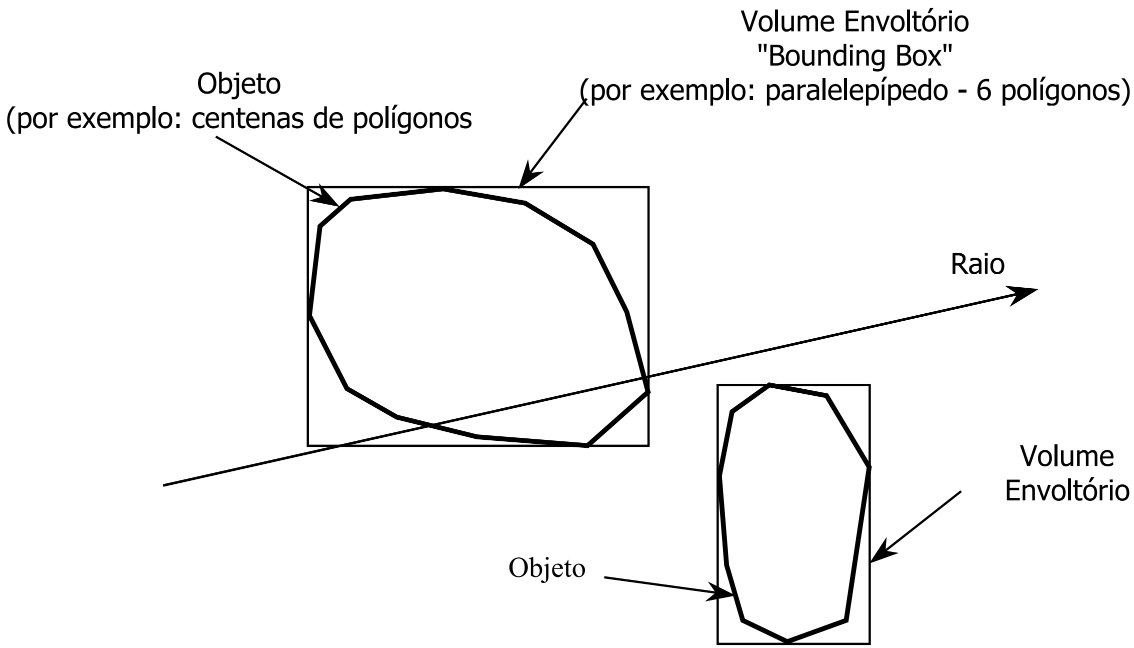
- A principal desvantagem é que, em princípio, para o cálculo do objeto mais próximo do observador é necessário calcular a interseção de cada raio com todas as primitivas da cena.

- Existem duas grandes estratégias para a aceleração da técnica de ray-casting:

  - Volume Envoltórios
  - Divisão Espacial.

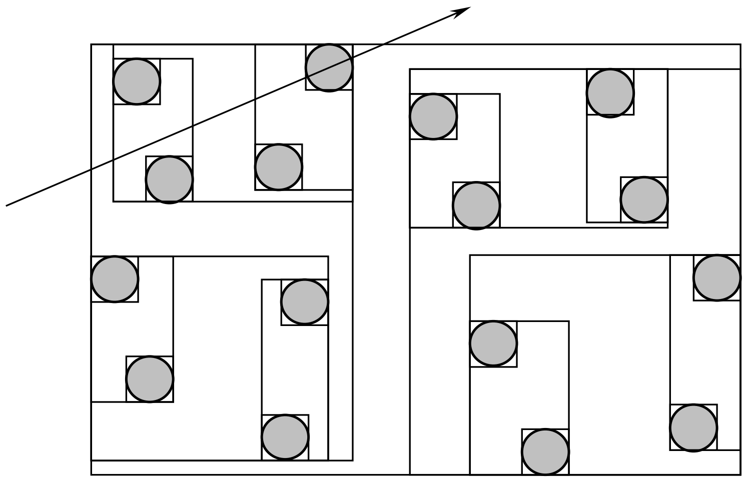
# Ray-casting (superfícies escondidas)

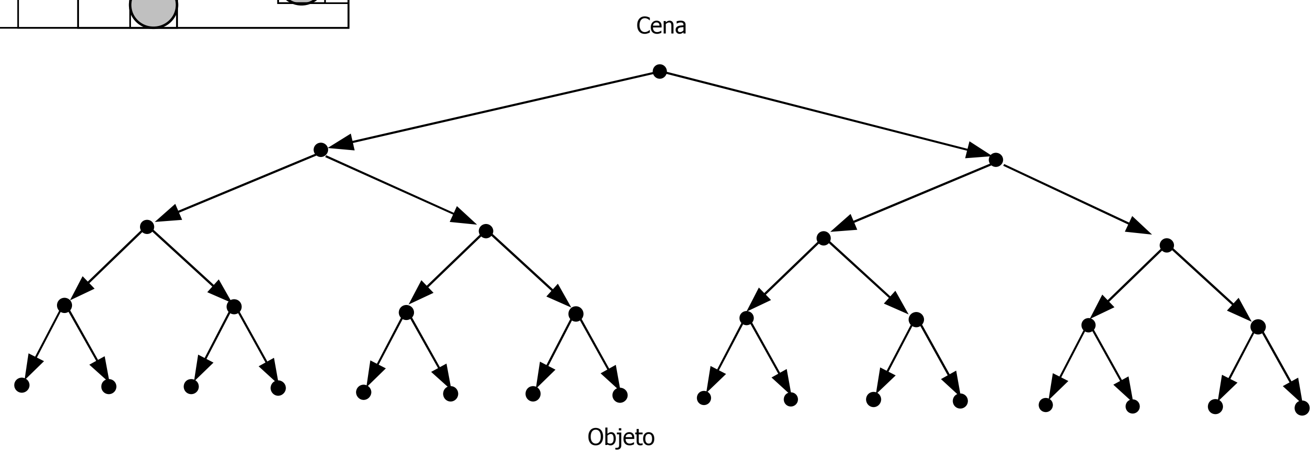
- Volume Envoltórios/Hierarquia de Volumes Envoltórios
  - A essência desta estratégia é efetuar testes de interseção com volumes envoltórios mais simples que o objeto. A interseção com o objeto apenas é calculada caso o raio atinja o volume envoltório.



# Ray-casting (superfícies escondidas)

- É possível ainda definir hierarquias de volumes envoltórios para reduzir a necessidade de testes com os objetos.





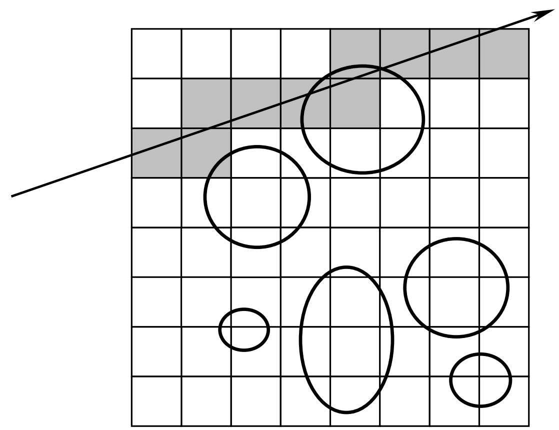
# Ray-casting (superfícies escondidas)

- Divisão Espacial

  - No método de hierarquia de volumes, uma hierarquia de volumes e construída a partir de volumes envoltórios dos objetos da cena. No método de divisão divide-se o espaço 3D da cena para a construção de volumes envoltórios. Esta divisão pode ser uniforme ou não uniforme.

# Ray-casting (superfícies escondidas)

- Divisão Espacial Uniforme
  - O espaço da cena é dividido em uma grade 3D uniforme. Cada elemento da grade recebe a identificação dos objetos que ele (parcialmente) contém. O caminho de um raio na grade é efetuado de maneira incremental utilizando-se uma extensão do algoritmo DDA.



# Ray-casting (superfícies escondidas)

- Divisão Espacial Não Uniforme
  - A divisão espacial divide o espaço em regiões de tamanho variável dependendo da densidade de ocupação local (manter o número de objetos por *voxels* constante)