# Transformações Geométricas para Visualização 3D

por

**Marcelo Gattass**
**Departamento de Informática**
**PUC-Rio**

**(adaptado por Luiz Fernando Martha para a disciplina CIV2802 – Sistemas Gráficos para Engenharia)**

# Visualização 3D
# Questões a serem consideradas

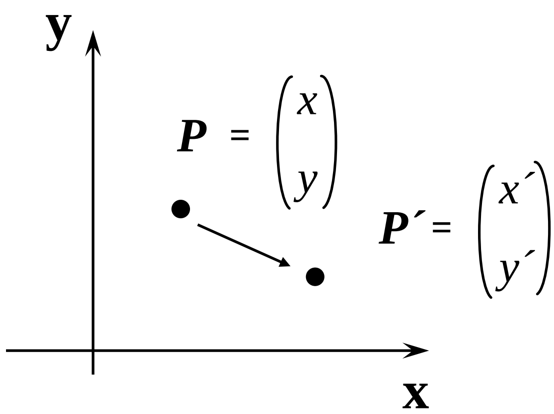
- Projeção ortográfica ou cônica (perspectiva)

- Iluminação (*lighting*) – sobreamento (*shading*)

- Tratamento de superficies ocultas (*hidden surfaces*)
  - No espaço de modelagem
    - *Cull Face* (só desenha faces com normais para frente)
    - Sorteamento da profundidade dos centros das faces
  - No espaço da tela
    - *Z-buffer* (*depth test*)

- Controle dinâmico do posicionamento da câmera

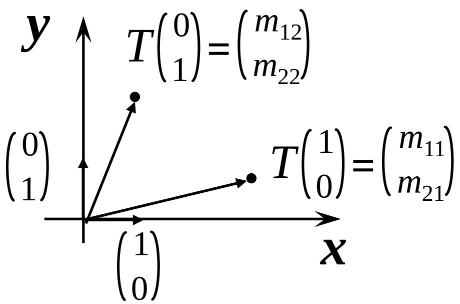
# Transformações Lineares
# $R^2 \rightarrow R^2$




$$\boxed{T(a_1\,\boldsymbol{P}_1 + a_2\,\boldsymbol{P}_2) = a_1\,T(\boldsymbol{P}_1) + a_2\,T(\boldsymbol{P}_2)}$$

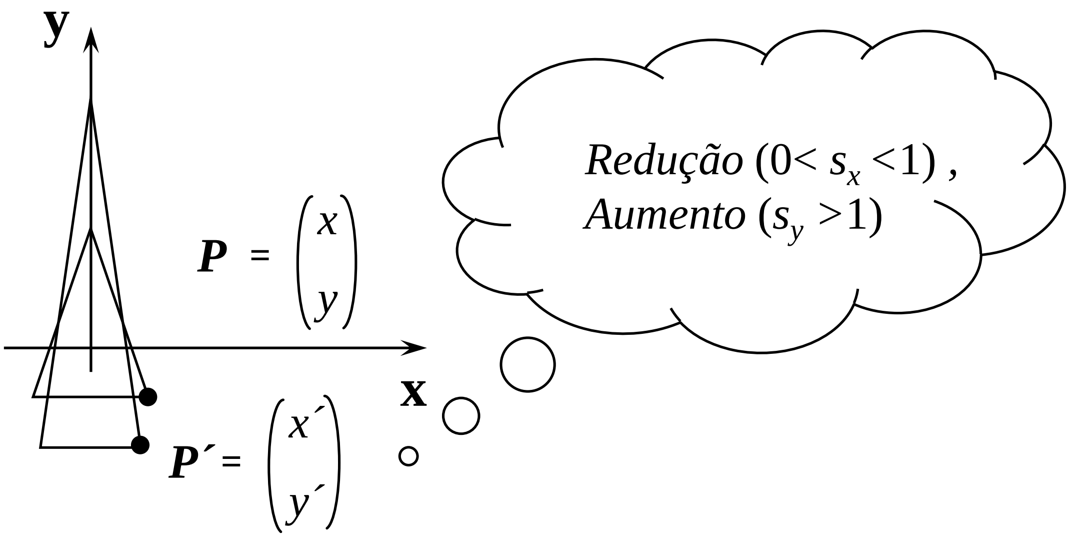
*Mostre que:*

A) $T\,(\mathbf{0}) = \mathbf{0}$

B) 


$$\Leftrightarrow \qquad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

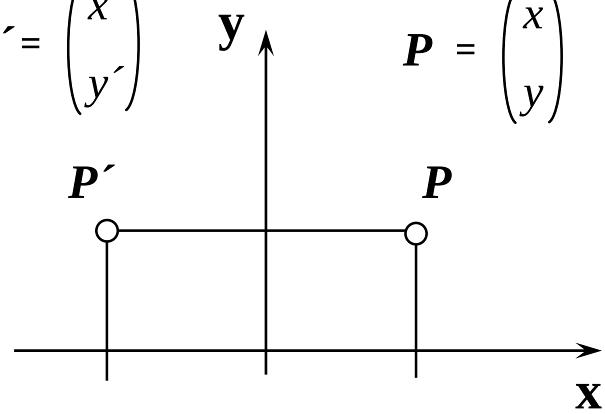
# Transformações Lineares (escala)




$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

# Transformações Lineares (espelhamento)

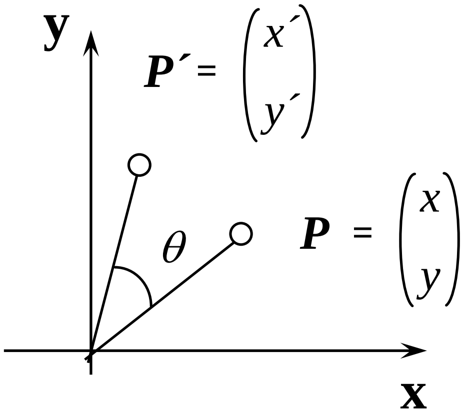
$$P' = \begin{pmatrix} x' \\ y' \end{pmatrix} \qquad P = \begin{pmatrix} x \\ y \end{pmatrix}$$




$$x' = -x$$
$$y' = \ \ y$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

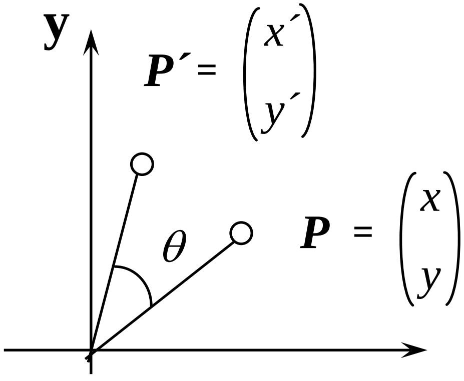
# Transformações Lineares (rotação)

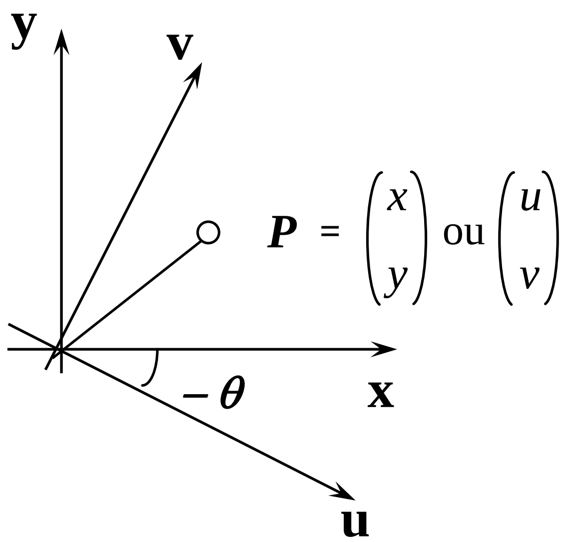



$$x' = x.\cos\theta - y.\text{sen}\,\theta$$
$$y' = x.\text{sen}\,\theta + y.\cos\theta$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} \cos\theta & -\text{sen}\,\theta \\ \text{sen}\,\theta & \cos\theta \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

# Transformações Lineares (rotação .vs. mudança de base)



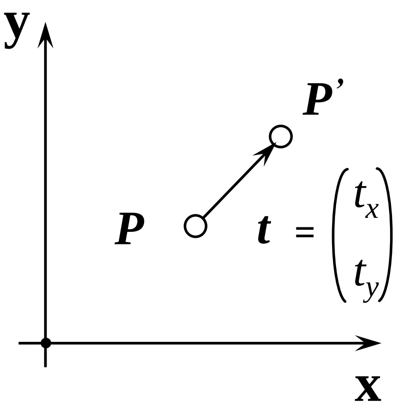




$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{bmatrix} \cos\theta & -\mathrm{sen}\,\theta \\ \mathrm{sen}\,\theta & \cos\theta \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{bmatrix} u_x & u_y \\ v_x & v_y \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

**A rotação de um ponto de $\theta$ tem o mesmo efeito da mudança de base por rotação de $-\theta$.**
**A matriz que implementa a mudança de base por rotação tem em cada linha as componentes dos vetores unitários da nova base descritos na base antiga.**

# Transformações Geométricas (Translação)

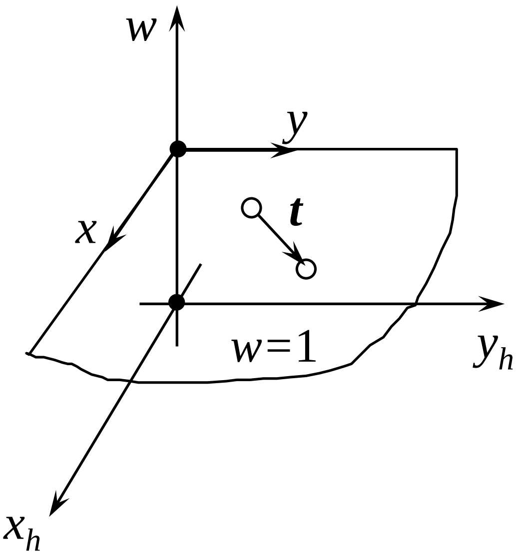



$$P' = \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$ $\Leftarrow$ *Não pode ser escrito na forma* ⊠

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$ $\Leftarrow$ *Ruim para implementação* ⊠

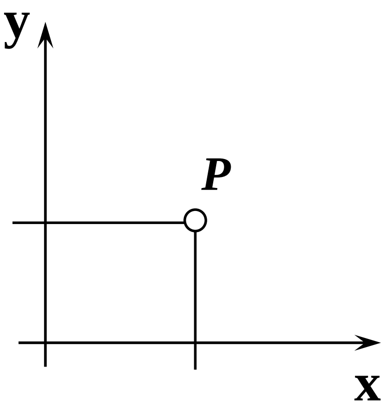
# Vantagens das coordenadas homogêneas (Translação)




$$\boldsymbol{P'} = \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}}_{[\boldsymbol{T}]} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

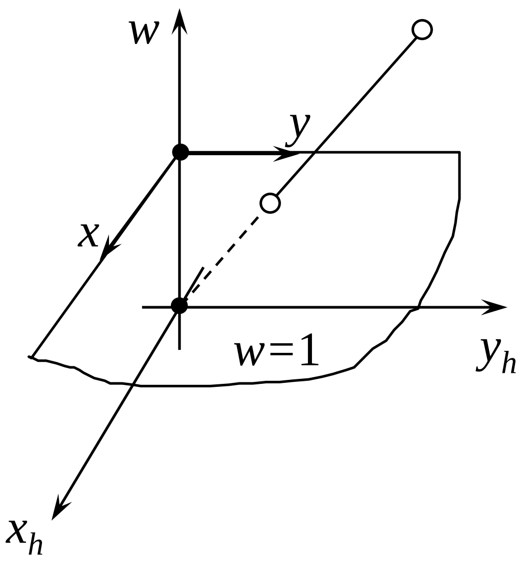
**Matriz de Translação**

# Coordenadas homogêneas




$$\boldsymbol{P} = \begin{pmatrix} x \\ y \end{pmatrix} \triangleq \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \triangleq \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} \triangleq \begin{bmatrix} x_h \\ y_h \\ w \end{bmatrix}$$

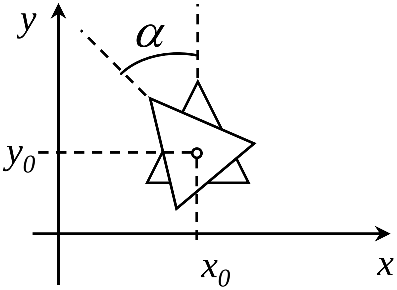
$$\boxed{\begin{array}{ll} x = x_h/w & \\ & w>0 \\ y = y_h/w & \end{array}}$$

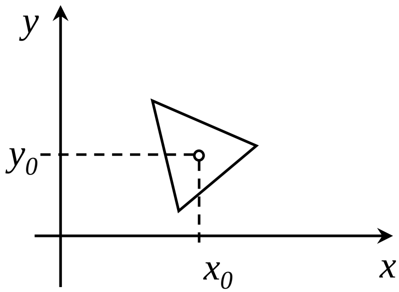



**Ex.:**

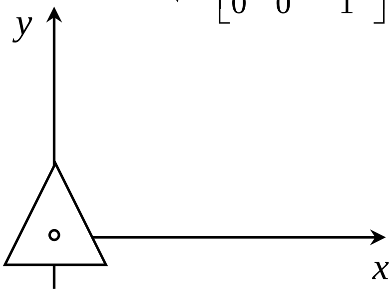
$$\begin{pmatrix} 3 \\ 2 \end{pmatrix} \triangleq \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix} \triangleq \begin{bmatrix} 6 \\ 4 \\ 2 \end{bmatrix} \triangleq \begin{bmatrix} 9 \\ 6 \\ 3 \end{bmatrix}$$

# Concatenação

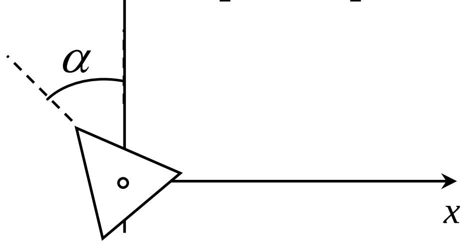



$$\begin{bmatrix} 1 & 0 & -x_0 \\ 0 & 1 & -y_0 \\ 0 & 0 & 1 \end{bmatrix}$$



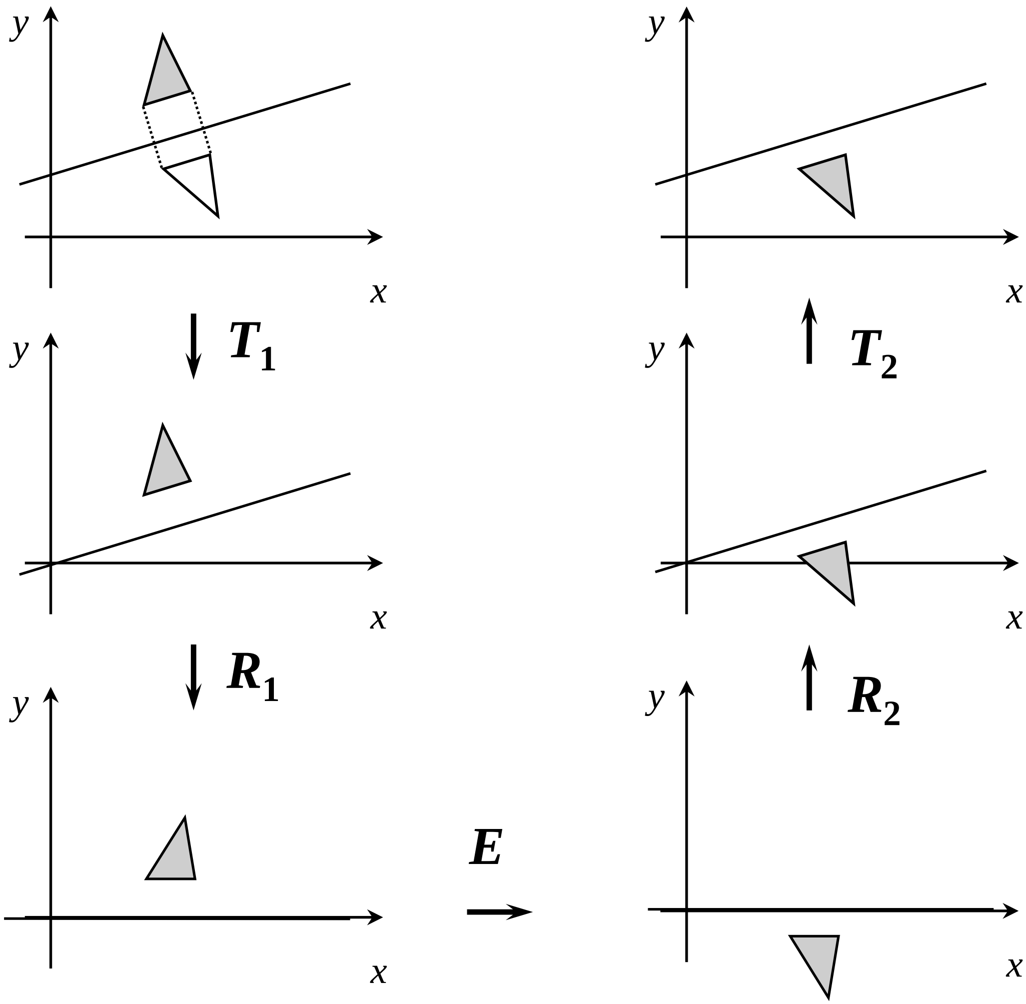

$$\begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$




$$\begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

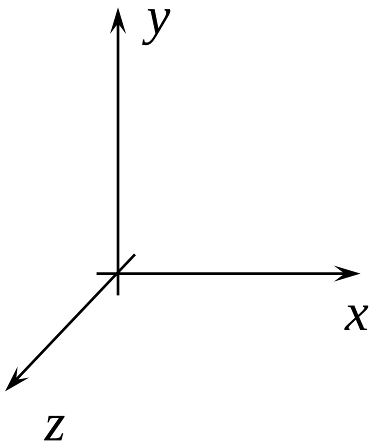
$$\boxed{\begin{Bmatrix} x' \\ y' \\ 1 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_0 \\ 0 & 1 & -y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} x \\ y \\ 1 \end{Bmatrix}}$$

# Concatenação de Transformações




$$P' = T_2\, R_2\, E\, R_1\, T_1\, P$$

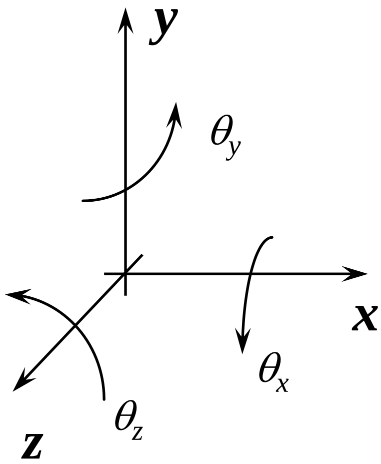
# Transformações em 3D
# (translações e escalas)




$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \left[\begin{array}{ccc:c} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \left[\begin{array}{ccc:c} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

# Transformações em 3D
# (rotações em torno dos eixos cartesianos)

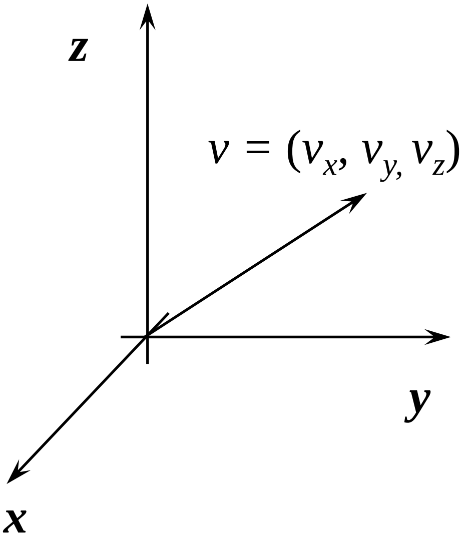



$$
\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & -\operatorname{sen}\theta_x & 0 \\ 0 & \operatorname{sen}\theta_x & \cos\theta_x & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}
$$

$$
\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} \cos\theta_y & 0 & \operatorname{sen}\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\operatorname{sen}\theta_y & 0 & \cos\theta_y & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}
$$

$$
\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} \cos\theta_z & -\operatorname{sen}\theta_x & 0 & 0 \\ \operatorname{sen}\theta_x & \cos\theta_x & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}
$$

# Transformações em 3D (rotação em torno de um eixo qualquer que passa pela origem)

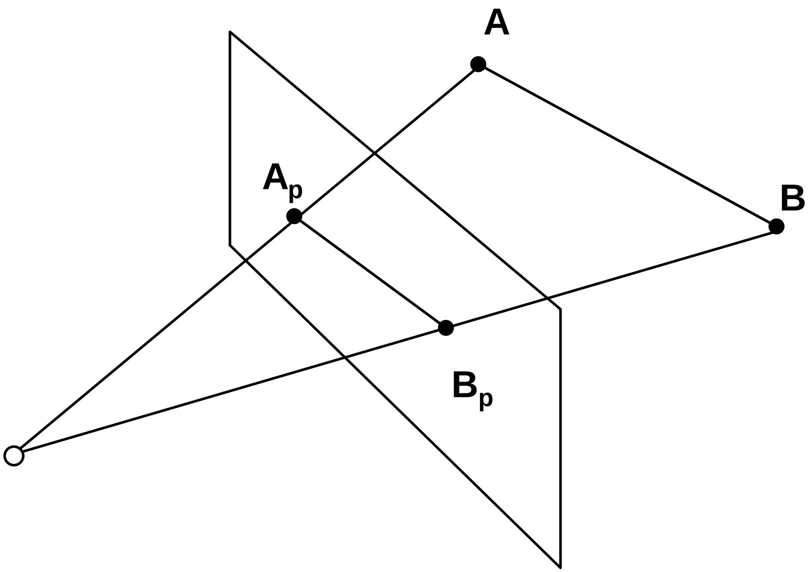



$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & 0 \\ m_{21} & m_{22} & m_{23} & 0 \\ m_{31} & m_{32} & m_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

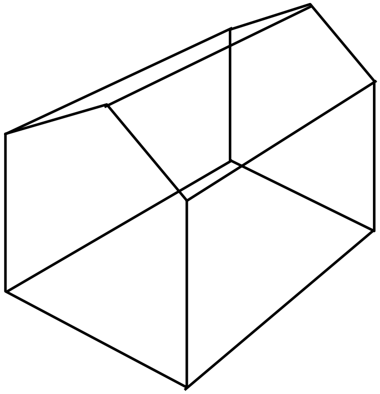
$$m_{11} = v_x^2 + \cos\theta\,(1 - v_x^2)$$
$$m_{12} = v_x v_y (1 - \cos\theta) - v_z \,sen\, \theta$$
$$m_{13} = v_z v_x (1 - \cos\theta) + v_y \,sen\, \theta$$
$$m_{21} = v_x v_y (1 - \cos\theta) + v_z \,sen\, \theta$$
$$m_{22} = v_y^2 + \cos\theta\,(1 - v_y^2)$$
$$m_{23} = v_y v_z (1 - \cos\theta) - v_x \,sen\, \theta$$
$$m_{31} = v_x v_z (1 - \cos\theta) - v_y \,sen\, \theta$$
$$m_{32} = v_y v_z (1 - \cos\theta) + v_x \,sen\, \theta$$
$$m_{22} = v_z^2 + \cos\theta\,(1 - v_z^2)$$

# Projeções Clássicas

# Projeções Planas Cônicas

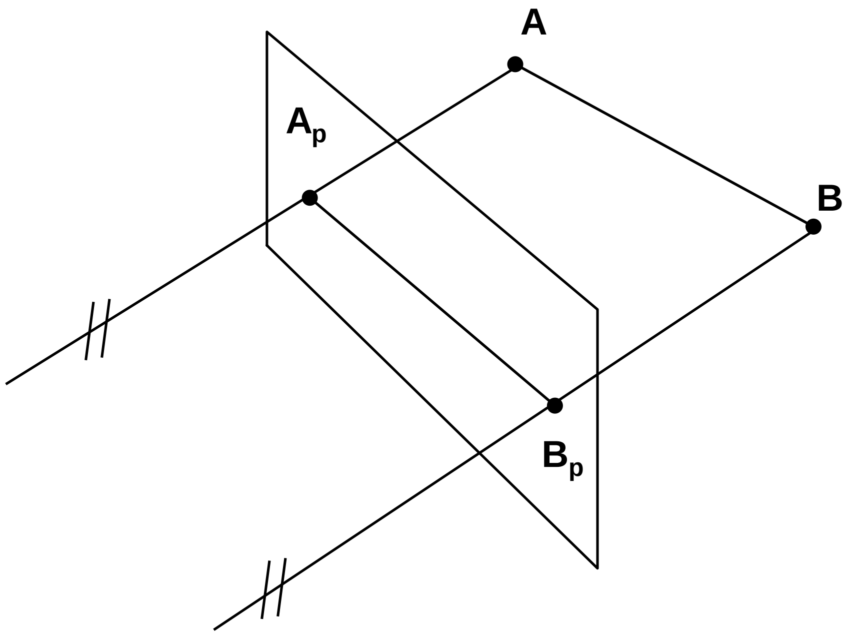



☺ *realista*



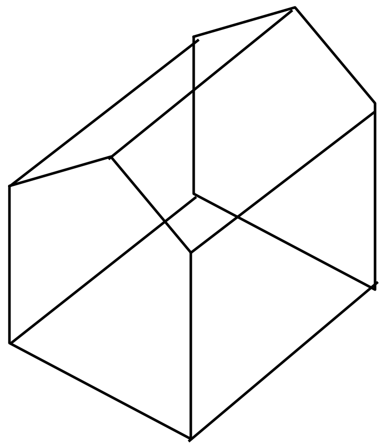
☹ *não preserva escala*
☹ *não preserva ângulos*

# Projeções Planas Paralelas




☹ *pouco realista*

☺ *preserva paralelismo*
☺ *possui escala conhecida*

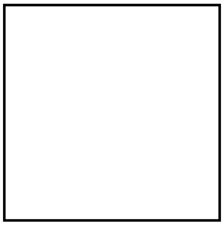


# Classificação das projeções planas

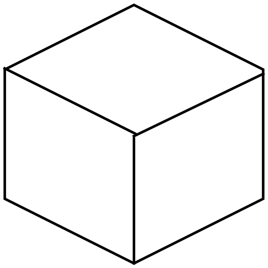
- Paralelas
  - » ortográficas **dp // n**
    - – plantas
    - – elevações
    - – iso-métrica
  - » oblíquas **dp** não é paralela a **n**
    - – cavaleiras
    - – *cabinet*
- Cônicas
  - » 1 pto de fuga
  - » 2 ptos de fuga
  - » 3 ptos de fuga

# Projeções de um cubo

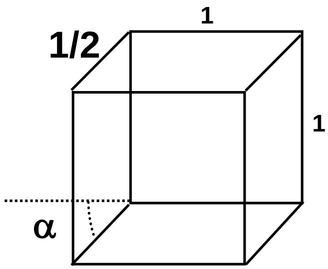
- **Paralelas**



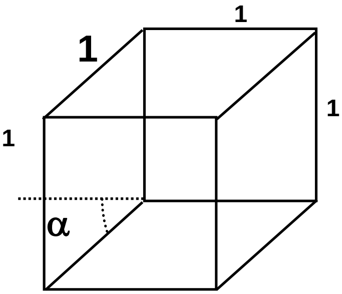
**planta ou elevação**



**iso-métrica**

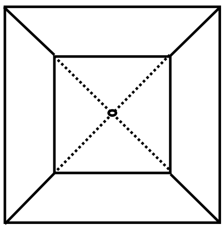



**Cabinete ($\alpha$=45º ou 60º)**

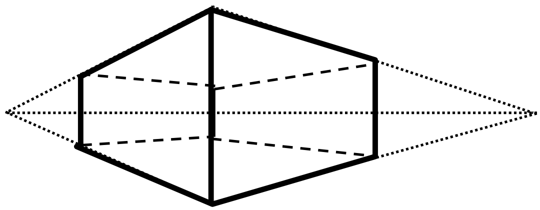



**Cavaleira ($\alpha$=45º ou 60º)**

- **Cônicas**



**1 pto de fuga**



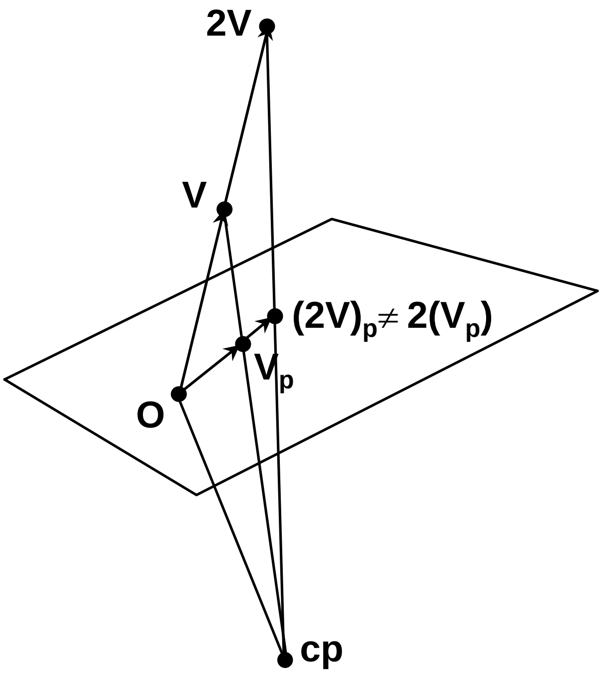
**2 ptos de fuga**

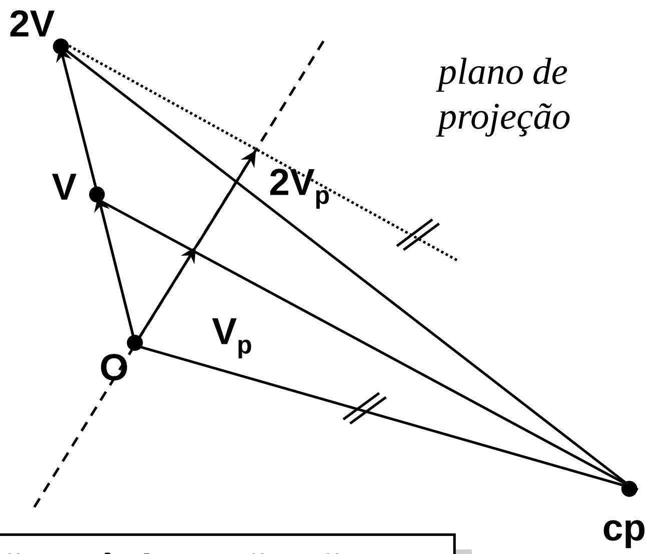
# Projeção plana é uma transformação linear?

Ou seja**:**

**T(P+Q) = T(P)+T(Q)** e

**T(αP) = αT(P) ?**

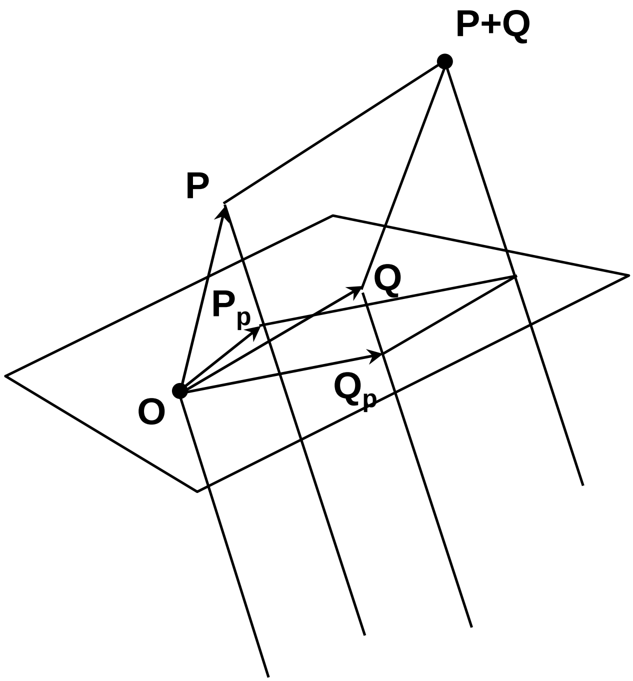


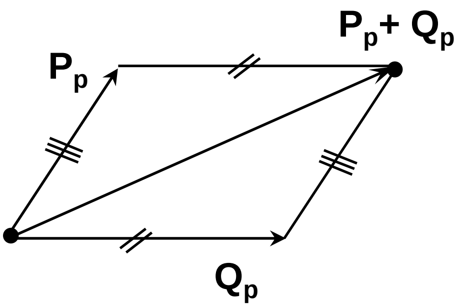

**Projeções cônicas não são TL, paralelas podem ser.**

# Projeção plana paralela é uma transformação linear?

$T(P+Q) = T(P)+T(Q)$ ?



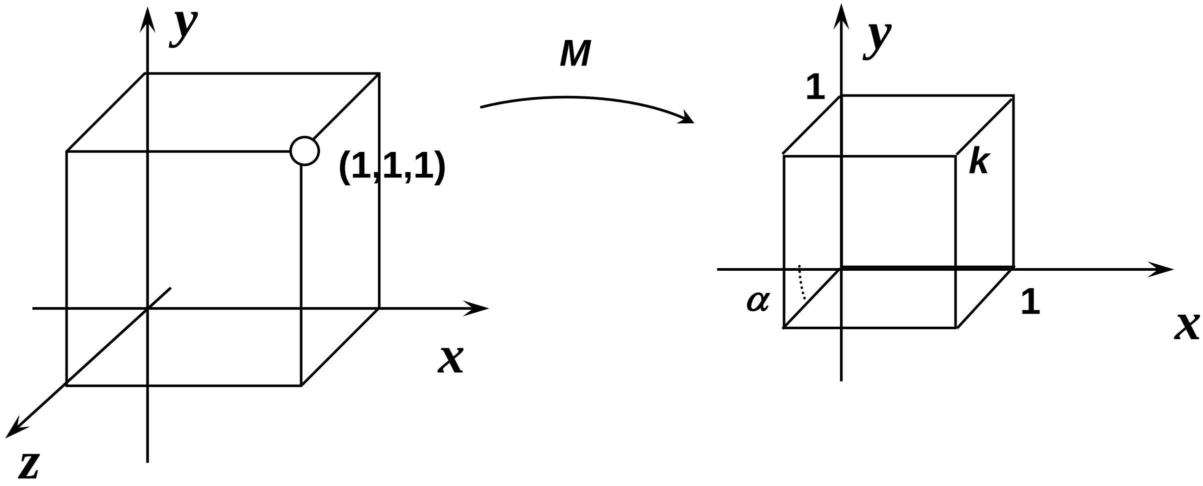

retas paralelas
projetam em paralelas




$T( 0 ) = 0$ ?

Projeção paralela em plano que passa
pela origem é uma transformação linear

# Matrizes de projeções Cavaleiras e Cabinetes




$\Re^3$ $\Re^2$

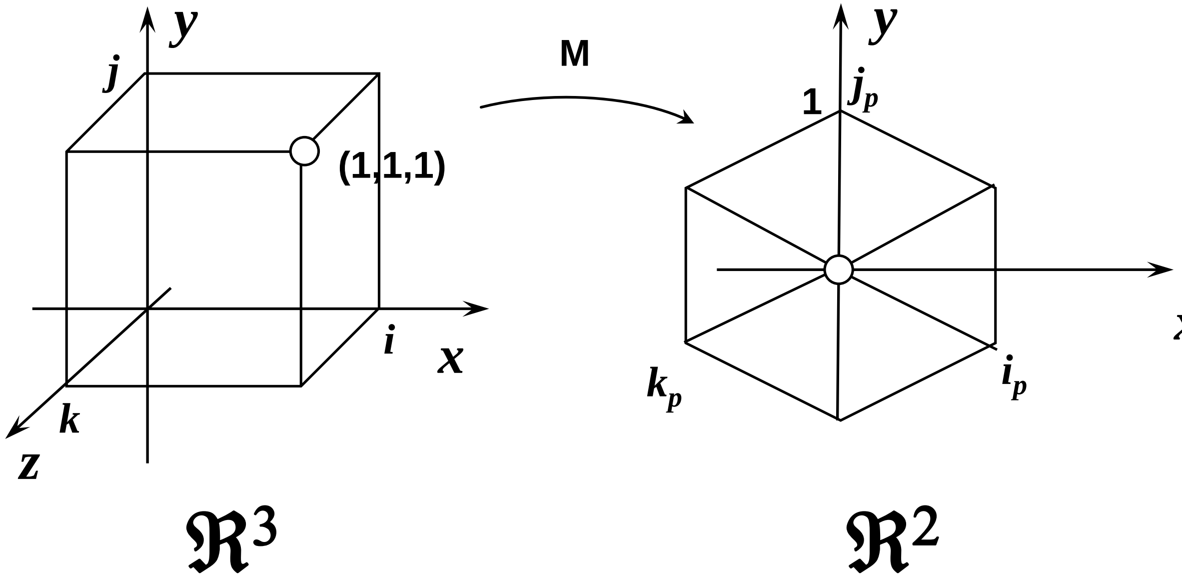
T(1,0,0) = (1,0)

T(0,1,0) = (0,1)

T(0,0,1) = ($-k \cos \alpha, -k \operatorname{sen} \alpha$)

$$M = \begin{vmatrix} 1 & 0 & -k.\cos(\alpha) \\ 0 & 1 & -k.\operatorname{sen}(\alpha) \end{vmatrix}$$

# Matrizes de projeções pseudo-isométricas




T(1,0,0) = (cos 30º , –sin 30º)

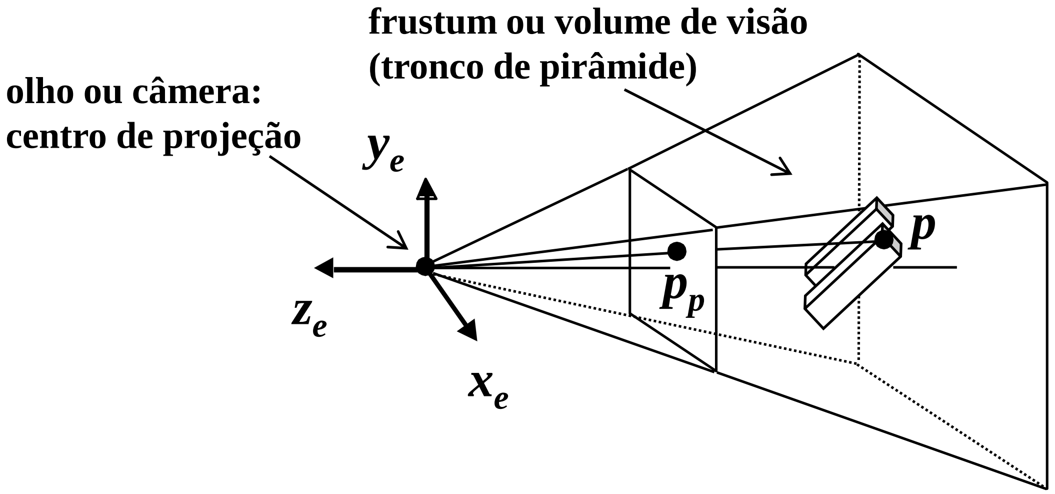
T(0,1,0) = (0,1)

T(0,0,1) = (–cos 30º, –sin 30)

$$M = \begin{vmatrix} \cos 30^\circ & 0 & -\cos 30^\circ \\ -\sin 30^\circ & 1 & -\sin 30^\circ \end{vmatrix}$$

# Projeções Cônicas

# Projeção cônica simples



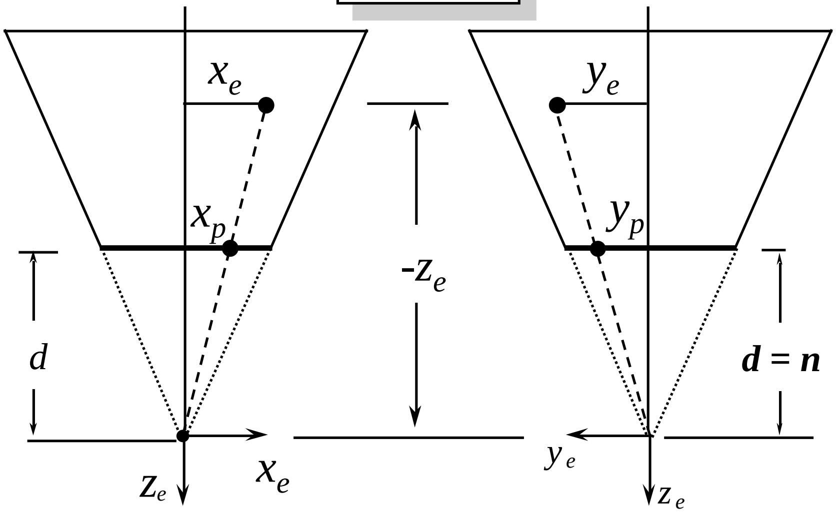

$$p = \begin{pmatrix} x_e \\ y_e \\ z_e \end{pmatrix}$$

$$p_p = \begin{pmatrix} x_p \\ y_p \\ -d \end{pmatrix}$$

$$\boxed{z_p = -d}$$

$$\frac{x_p}{x_e} = \frac{d}{-z_e}$$

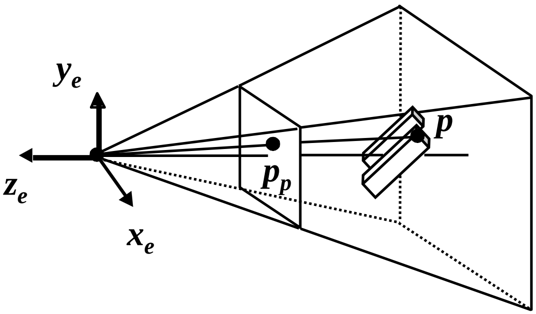
$$\boxed{x_p = \frac{d}{-z_e}\, x_e}$$




$$\frac{y_p}{y_e} = \frac{d}{-z_e}$$

$$\boxed{y_p = \frac{d}{-z_e}\, y_e}$$

# Projeção cônica simples via coordenadas homogêneas




$$\boxed{x_p = \frac{d}{-z_e}\; x_e}$$

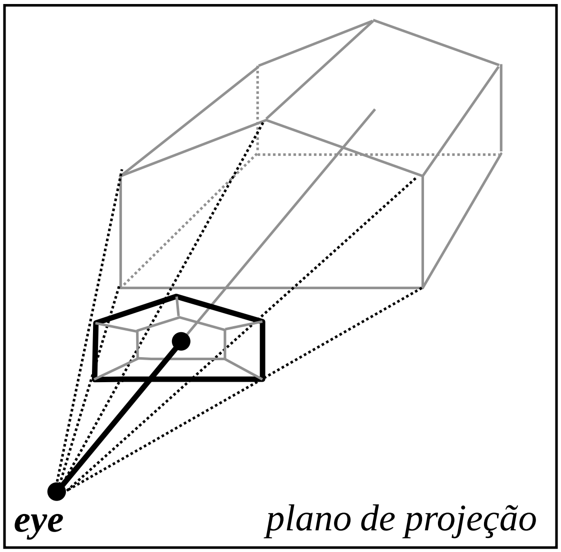
$$\boxed{y_p = \frac{d}{-z_e}\; y_e}$$

$$\boxed{z_p = -d}$$

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}}_{[\,\boldsymbol{P}\,]} \begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} d\,x_e \\ d\,y_e \\ d\,z_e \\ \underbrace{-z_e}_{w} \end{bmatrix} \xrightarrow{\div\, w} \begin{bmatrix} (d/\text{-}z_e)\,x_e \\ (d/\text{-}z_e)\,y_e \\ -d \\ 1 \end{bmatrix}$$

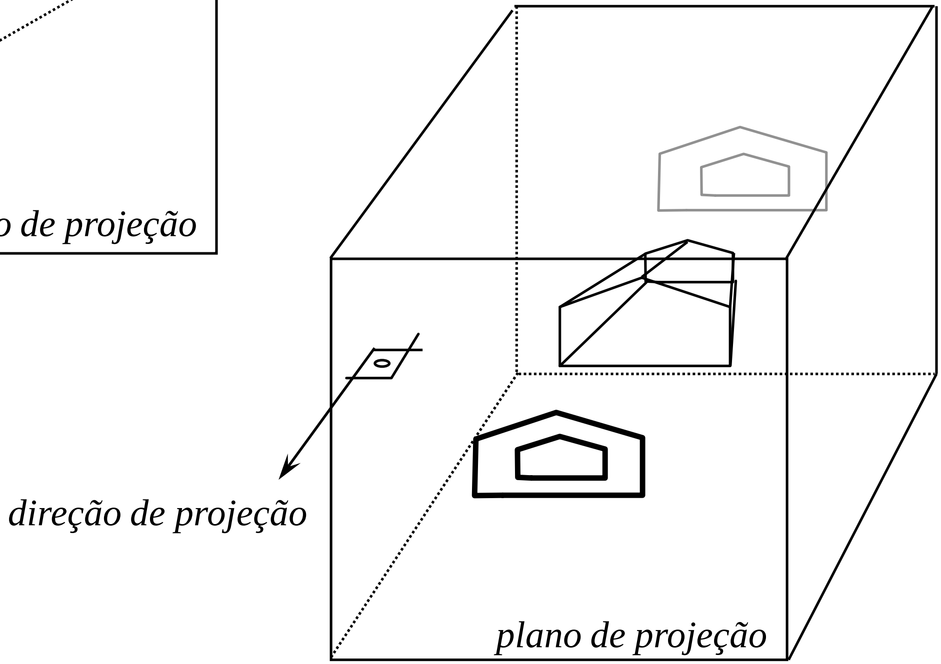
# Simplificação da projeção cônica: distorção do frustum de visão

**Projeção cônica**



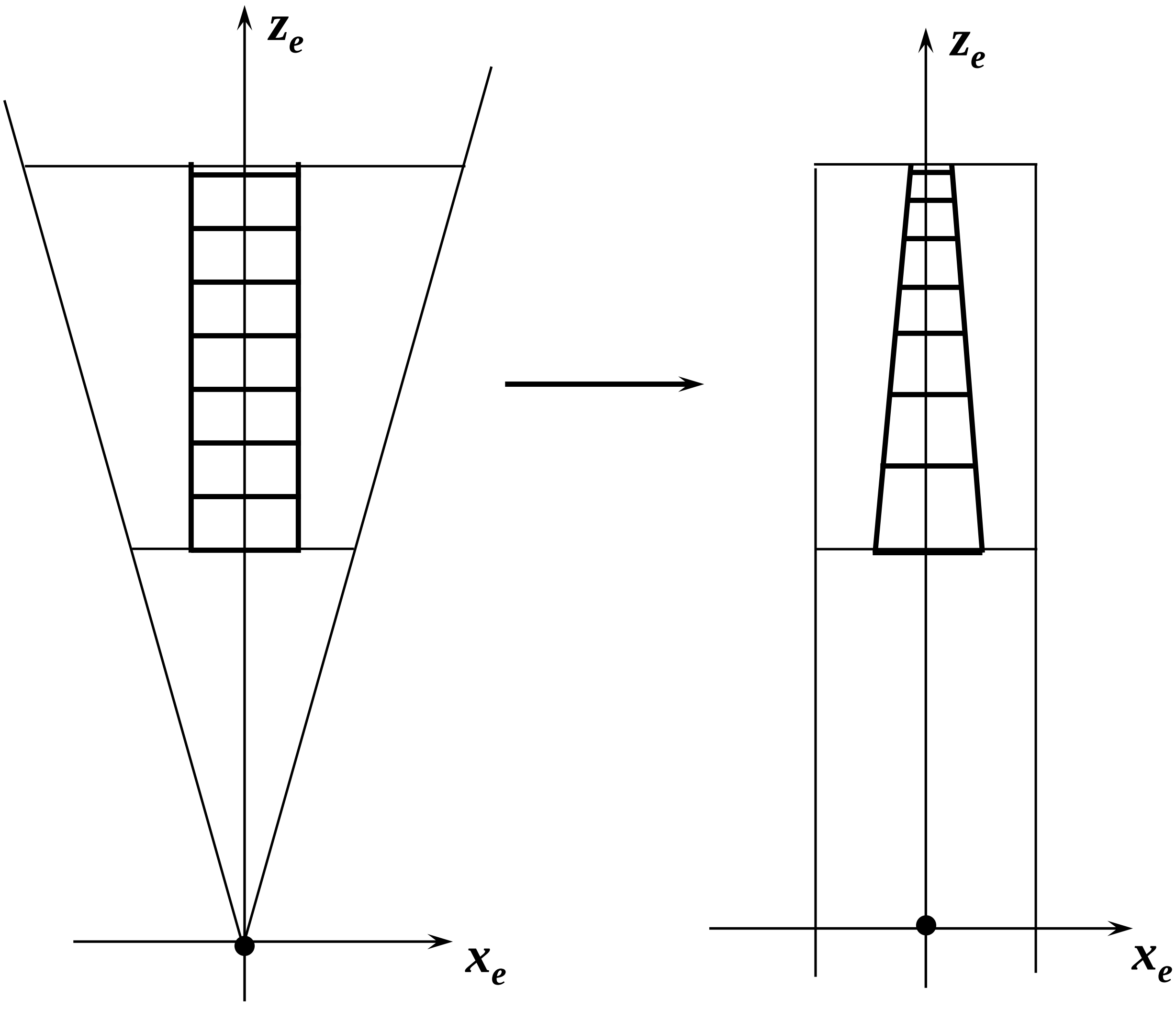

**É muito mais simples implementar uma projeção ortográfica do que uma projeção cônica.**

**Projeção ortográfica**



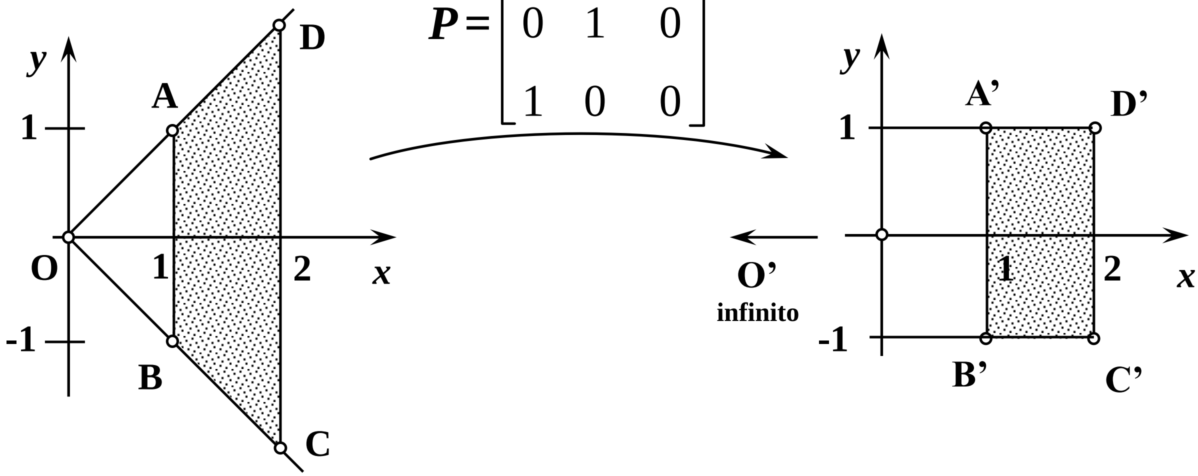
# Efeito de profundidade

**Como seria possível transformar a projeção cônica em uma projeção ortográfica considerando efeito de profundidade?**



# Solução: coordenadas homogêneas (exemplo 2D)

$$\boldsymbol{P} = \begin{bmatrix} 3 & 0 & -2 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$




$\boldsymbol{P}$ A = A'

$$\begin{bmatrix} 3 & 0 & -2 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$ ☺

$\boldsymbol{P}$ D = D'

$$\begin{bmatrix} 3 & 0 & -2 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}$$ ☺

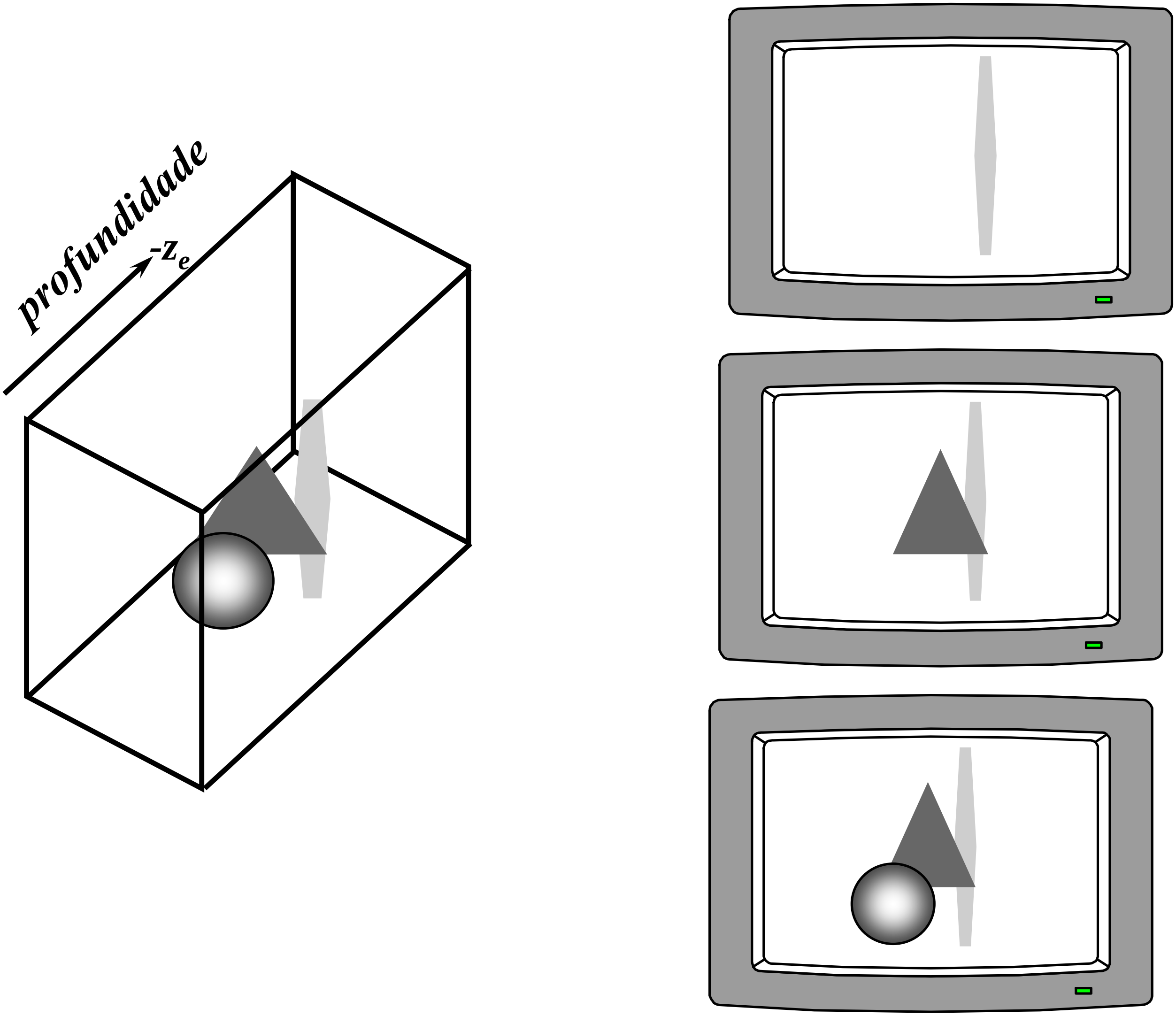
$\boldsymbol{P}$ B = B'

$$\begin{bmatrix} 3 & 0 & -2 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$ ☺

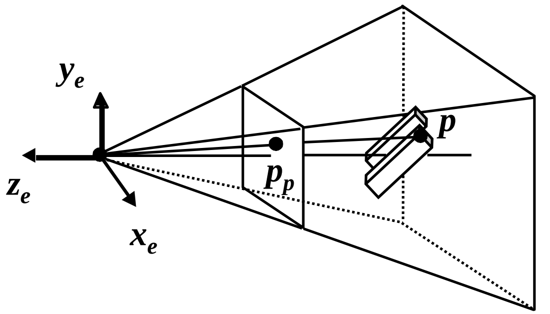
$\boldsymbol{P}$ C = C'

$$\begin{bmatrix} 3 & 0 & -2 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ -2 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$$ ☺

# Importância do efeito de profundidade: remoção de superfícies escondidas



# Projeção cônica simples preservando profundidade em *z* e planaridade




$$x_p = \frac{d}{-z_e}\, x_e$$

$$y_p = \frac{d}{-z_e}\, y_e$$

$$z_p = \textit{profundidade}$$

Condição de manutenção de planaridade de planos no espaço distorcido: $z_p = \alpha + (\beta / z_e)$

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & -\alpha & -\beta \\ 0 & 0 & -1 & 0 \end{bmatrix}}_{[\,\boldsymbol{P}\,]} \begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} d\,x_e \\ d\,y_e \\ -\alpha\, z_e - \beta \\ \underbrace{-z_e}_{w} \end{bmatrix} \xrightarrow{\div\, w} \begin{bmatrix} (d/-z_e)\, x_e \\ (d/-z_e)\, y_e \\ \alpha + (\beta / z_e) \\ 1 \end{bmatrix}$$

# Determinação da profundidade $z_p$ para garantir planaridade

O espaço de coordenadas ($x_p$,$y_p$,$z_p$) obtido pela matriz de transformação [$P$] é denominado *espaço da tela*.

A transformação do objeto para o espaço da tela é conveniente porque faz com que o processo de remoção de linhas e superfícies ocultas de uma imagem seja feito com base em retas perpendiculares ao plano de projeção (projeção ortográfica). Isto é, para saber se um ponto de uma linha ou superfície é obscurecido por outro ponto basta comparar a profundidade $z_p$ dos dois pontos: o que tiver a menor profundidade é o ponto que aparecerá na tela.

Entretanto, para que o resultado da transformação do objeto para o sistema de coordenadas da tela seja útil para a eliminação de linhas e superfícies escondidas, é necessário que se calcule a profundidade de uma linha, não somente para os seus pontos extremos, mas também para qualquer ponto interior da linha. O mesmo se aplica para pontos interiores de um plano.

# Determinação da profundidade $z_p$ para garantir planaridade (cont.)

Portanto, para que a interpolação de um ponto interior no espaço da tela seja simples, é preciso que:

- Linhas retas no sistema de coordenadas do olho sejam transformadas para linhas retas no sistema de coordenadas da tela.
- Planos no sistema de coordenadas do olho sejam transformados para planos no sistema de coordenadas da tela.

Assim:

$$\boxed{x_e = \frac{-z_e}{d}\, x_p}$$

$$a.x_e + b.y_e + c.z_e + q = 0$$

$$\boxed{y_e = \frac{-z_e}{d}\, y_p}$$

$$a'.x_p + b'.y_p + c'.z_p + q' = 0$$

Substituindo $x_e$ e $y_e$ na primeira equação e dividindo por $-z_e$, tem-se:

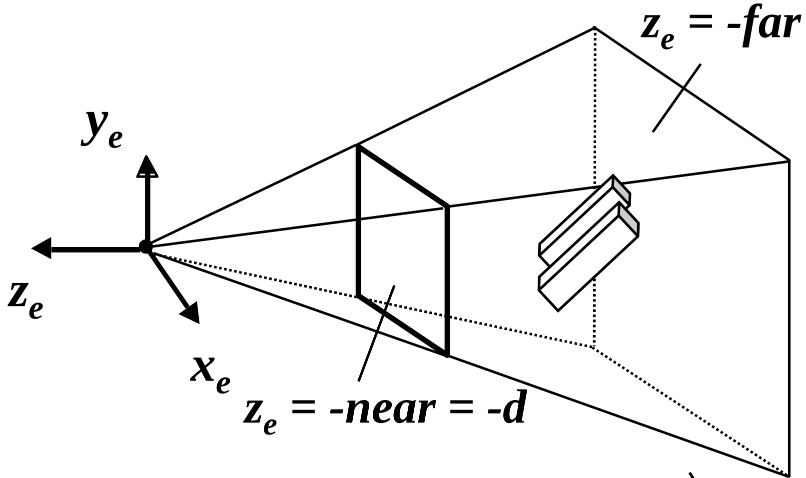
$$\frac{a}{d}x_p + \frac{b}{d}y_p - c - \frac{q}{z_e} = 0$$

Comparando esta equação com a segunda equação, vê-se que:

$$c'.zp + q' = -c - \frac{q}{z_e}$$

$$\boxed{z_p = \alpha + (\beta / z_e)}$$

# Distorção do frustum de visão para um paralelepípedo de mesma altura

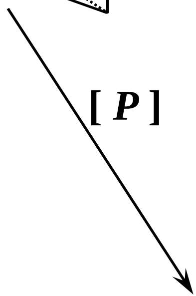



Existem infinitos valores de $\alpha$ e $\beta$ que satisfazem a condição de planaridade $z_p = \alpha + (\beta / z_e)$. Uma boa opção é manter a altura do frustum de visão na distorção. Isto faz o problema da projeção cônica recair no problema padrão de projeção ortográfica. Neste caso:

$\alpha = -(near + far)$

$\beta = -(near \times far)$

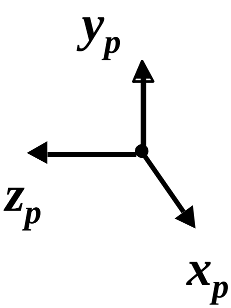
Obs.: *near* e *far* são distâncias (> 0);
*d* = *near* (usualmente adotado).

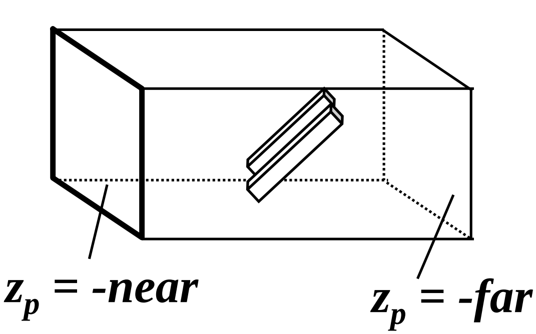



$$[\,P\,] = \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & n \times f \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

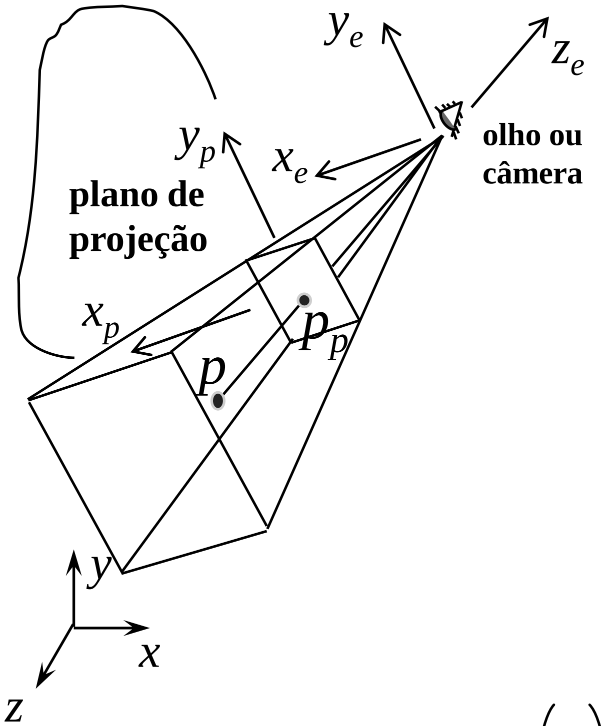
$n = near$

$f = far$



# Geometria Projetiva e Coordenadas Homogêneas em 3D




**Problema: como transformar a projeção cônica geral em uma projeção cônica simples?**

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ w \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

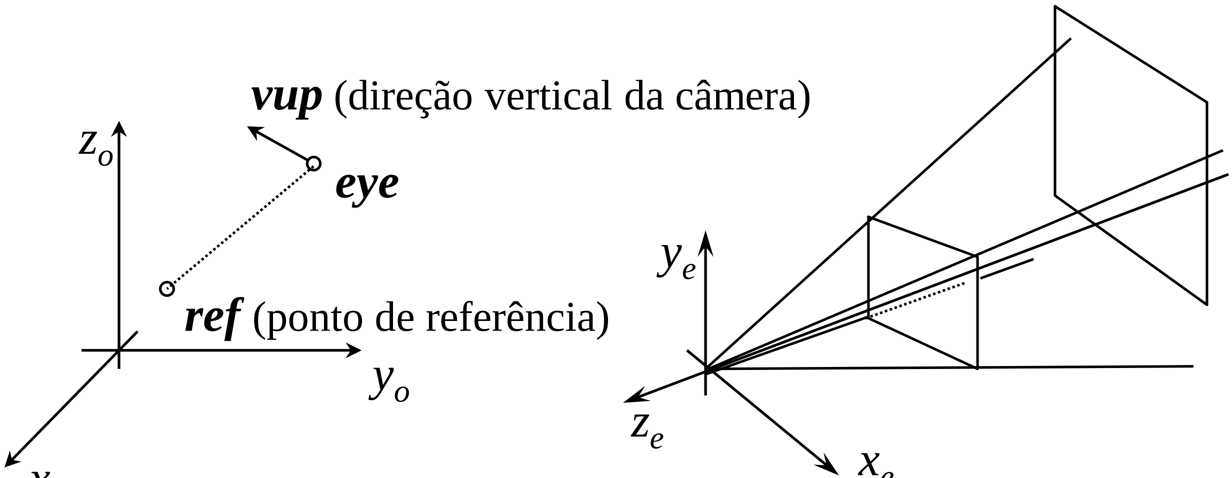
[ ***P*** ][ ? ]

$$\boldsymbol{p_p} = \begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} = \begin{pmatrix} x_h/w \\ y_h/w \\ z_h/w \end{pmatrix} \qquad \boldsymbol{p} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

# Transformação de câmera

*Dados:* ***eye***, ***ref***, ***vup*** *(definem o sistema de coordenadas do olho)*

*Determine a matriz que leva do sistema de Coordenadas dos Objetos para o sistema de Coordenadas do Olho*



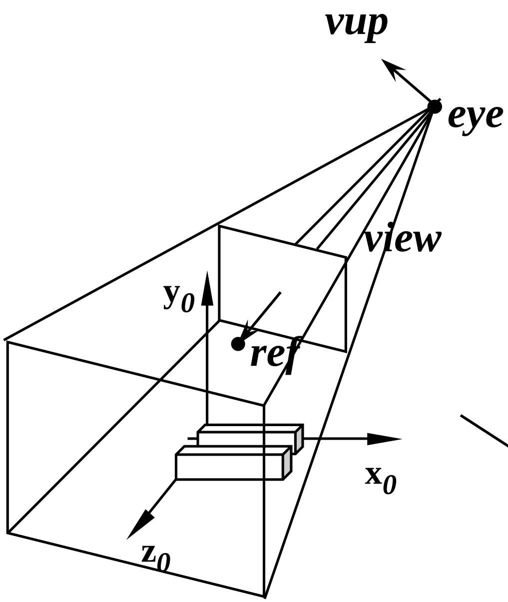

*Sistema de coordenadas dos Objetos*

*Sistema de coordenadas do Olho*

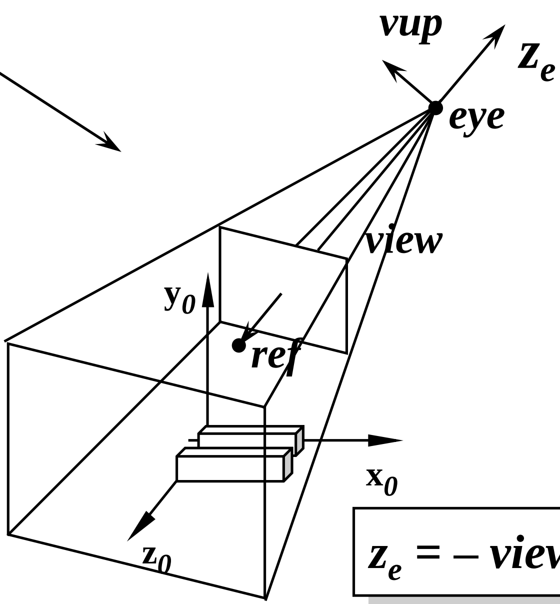
```
Parâmetros: eye ( eyex, eyey, eyez )
            ref ( refx, refy, refz )
            vup ( vupx, vupy, vupz )
```

# Calcula o sistema – $x_e\, y_e\, z_e$

dados:
***eye, ref, vup***

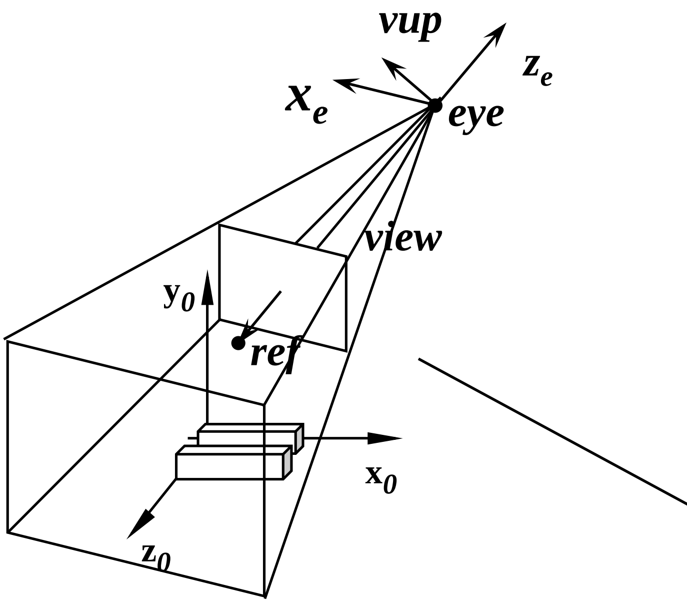



***view = ref - eye***

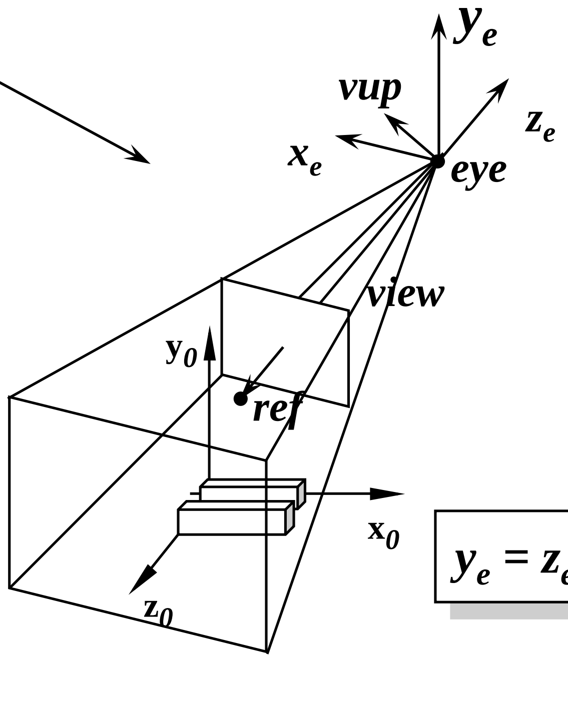


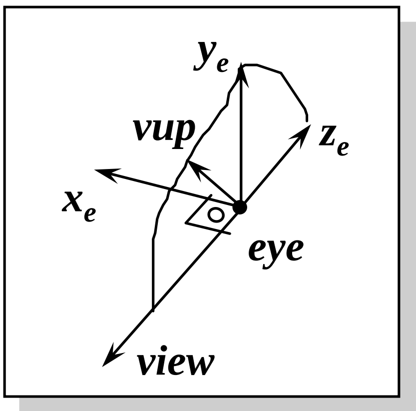

$z_e = -\, view \,/\, \|view\|$

# Calcula o sistema – $x_e\, y_e\, z_e$



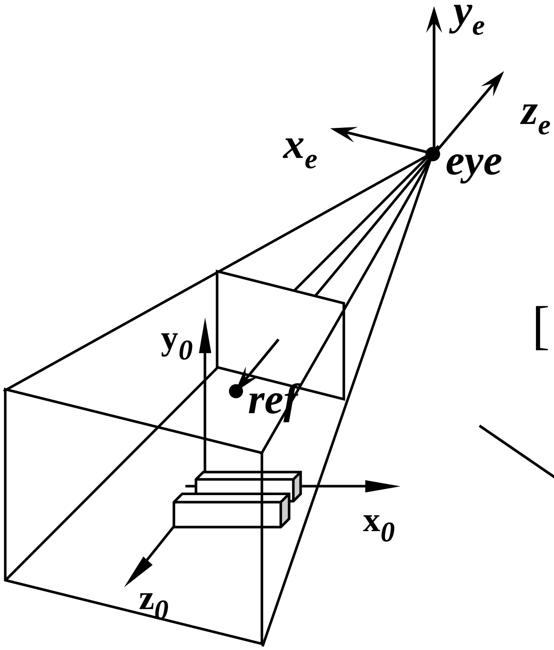

$x_e = (vup \times z_e) \,/\, \|vup \times z_e\|$



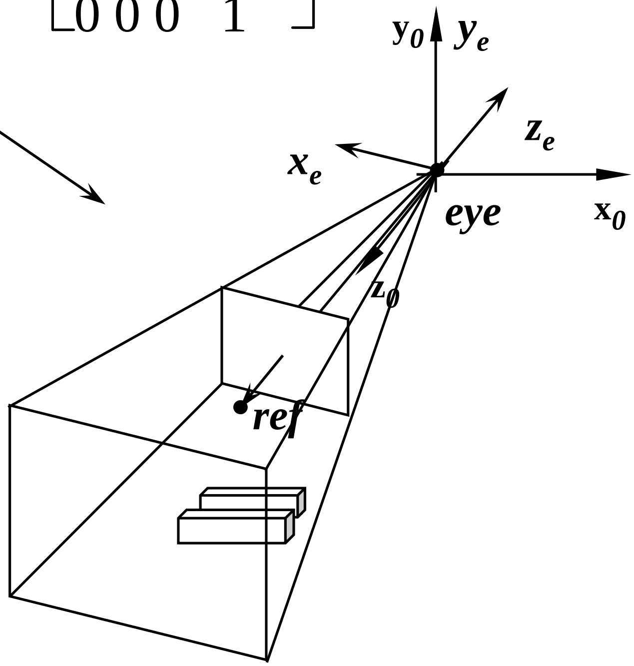




$y_e = z_e \times x_e$

# Translada o *eye* para origem (mudança de base por translação)



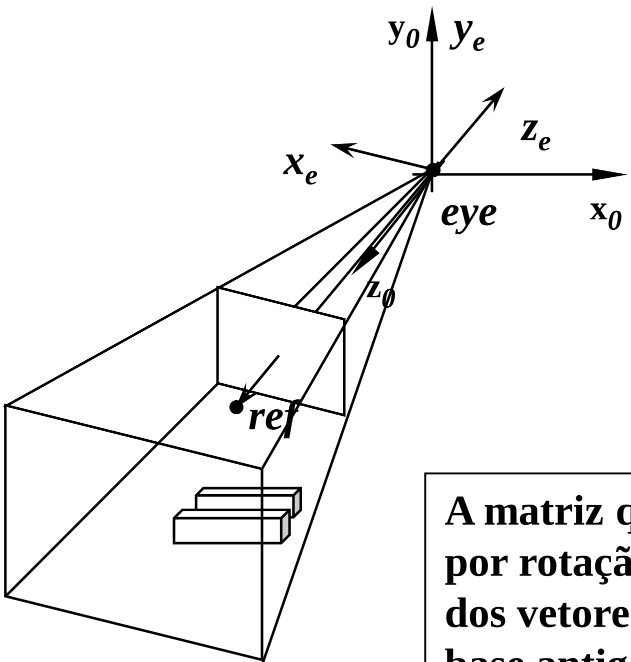

$$[\,T_c\,] = \begin{bmatrix} 1 & 0 & 0 & -eye_x \\ 0 & 1 & 0 & -eye_y \\ 0 & 0 & 1 & -eye_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

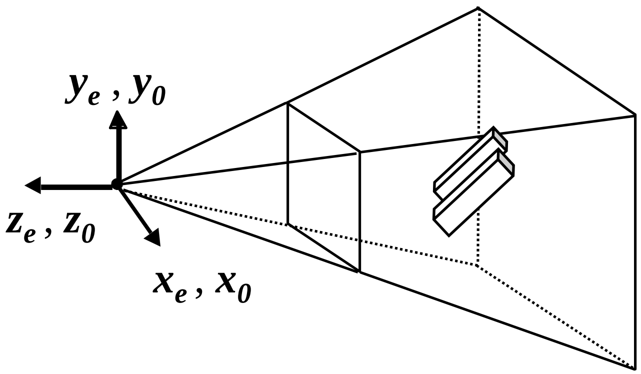



**A matriz da mudança de base por translação para a nova origem *eye* é obtida aplicando-se um vetor de translação igual a -*eye*.**

# Rotaciona $x_e\, y_e\, z_e$ para $x_o\, y_o\, z_o$

$$[\,\boldsymbol{R}\,] = \begin{bmatrix} x_{ex} & x_{ey} & x_{ez} & 0 \\ y_{ex} & y_{ey} & y_{ez} & 0 \\ z_{ex} & z_{ey} & z_{ez} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$




**A matriz que implementa a mudança de base por rotação tem em cada linha as componentes dos vetores unitários da nova base descritos na base antiga.**

# Matriz de transformação de Câmera [*C*]

$$[\,\boldsymbol{T_c}\,] = \begin{bmatrix} 1 & 0 & 0 & -eye_x \\ 0 & 1 & 0 & -eye_y \\ 0 & 0 & 1 & -eye_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boxed{\boldsymbol{z_e = -\,view\,/\,\|view\|}}$$

$$\boxed{\boldsymbol{x_e = (vup \times z_e)\,/\,\|vup \times z_e\|}}$$

$$\boxed{\boldsymbol{y_e = z_e \times x_e}}$$

$$[\,\boldsymbol{R}\,] = \begin{bmatrix} x_{ex} & x_{ey} & x_{ez} & 0 \\ y_{ex} & y_{ey} & y_{ez} & 0 \\ z_{ex} & z_{ey} & z_{ez} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boxed{[\,\boldsymbol{C}\,] = [\,\boldsymbol{R}\,]\,[\,\boldsymbol{T_c}\,]}$$

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ w \end{bmatrix} = \underbrace{\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}}_{\boxed{[\,\boldsymbol{P}\,][\,\boldsymbol{C}\,]}} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} = \begin{pmatrix} x_h/w \\ y_h/w \\ z_h/w \end{pmatrix}$$