Aula 10

Professores:

Anselmo Montenegro
Esteban Clua

Conteúdo:

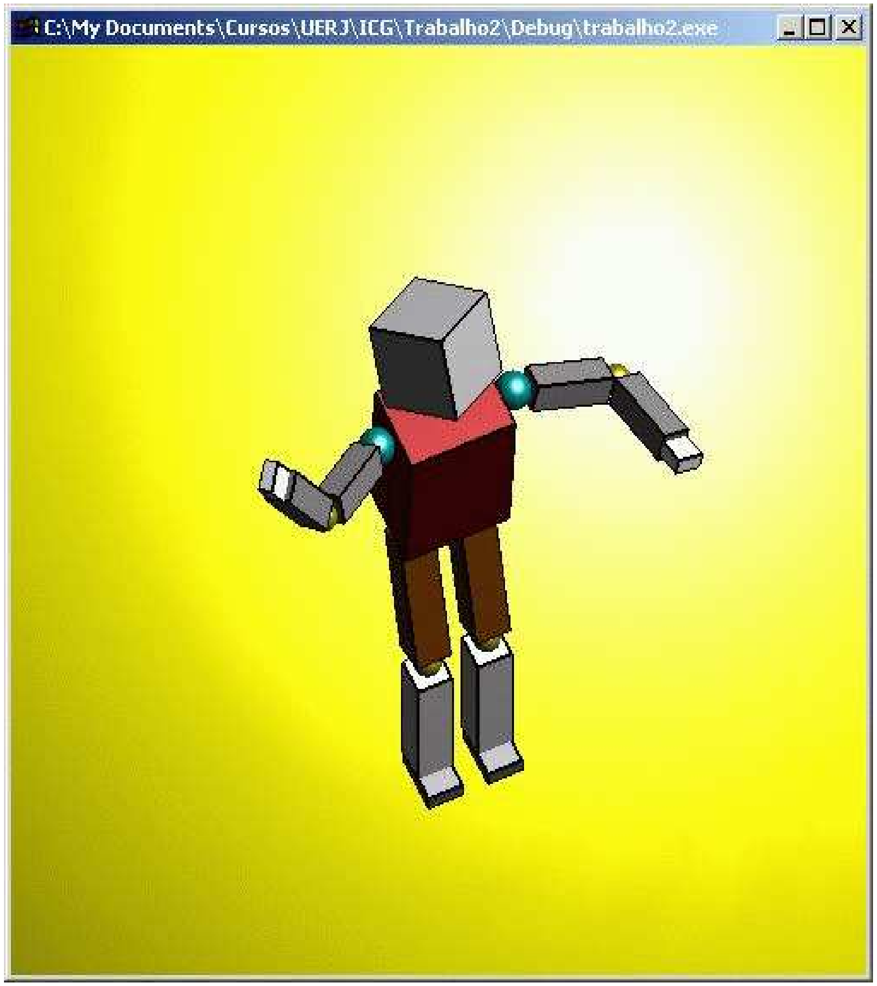
- Projeções e câmera virtual

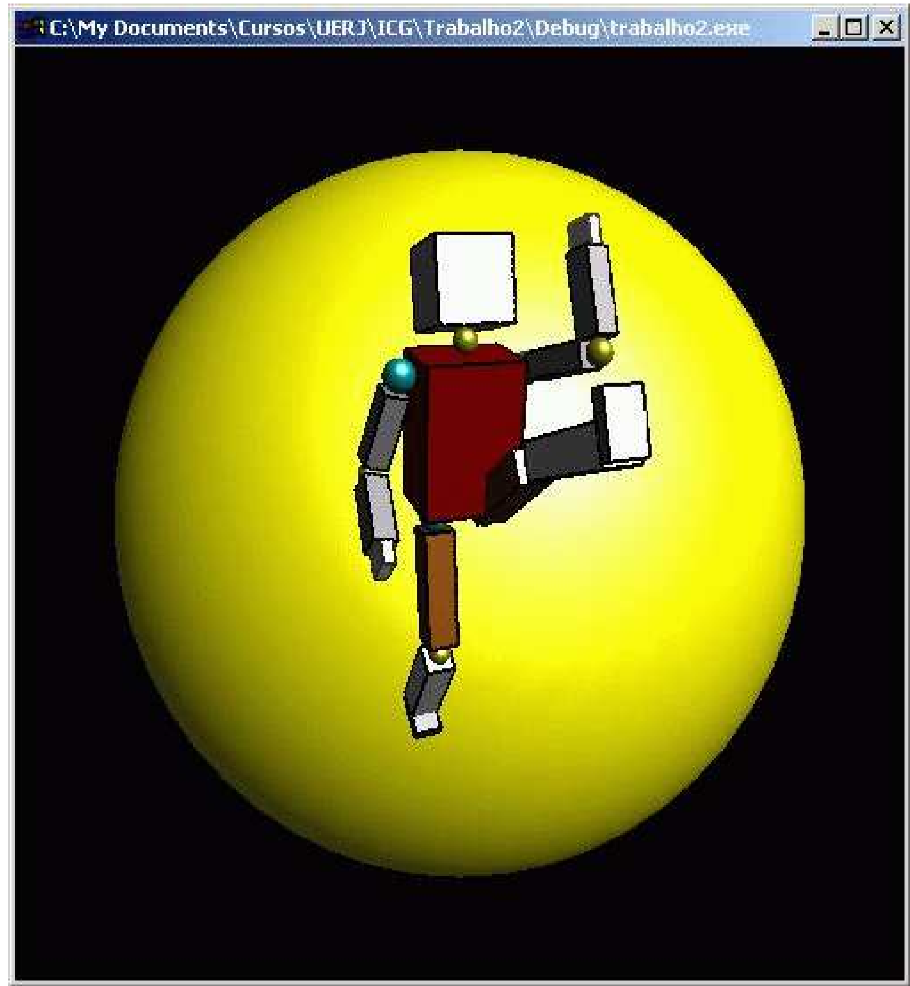
Introdução

- A geometria afim é capaz de descrever boa parte dos objetos e operações da Computação Gráfica.
- Entretanto, há operações que requerem um outro tipo de geometria.
- Este é o caso das ***transformações de visualização***.

Geometria projetiva

Motivação



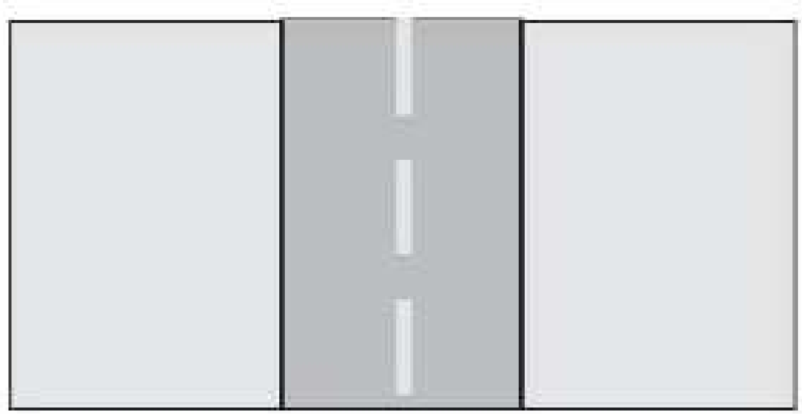


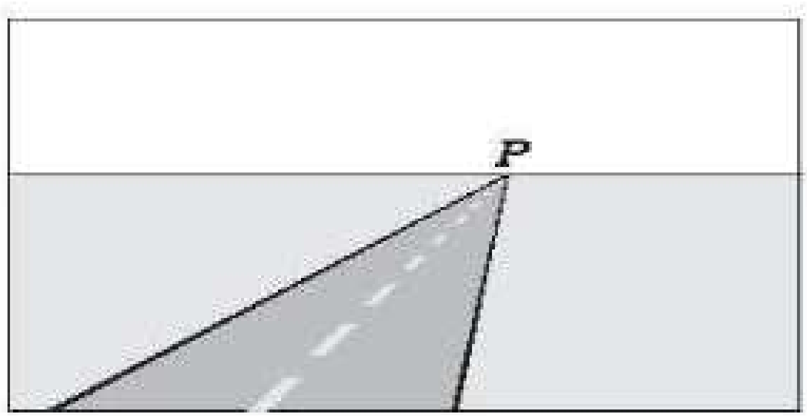
Fundação CECIERJ
Consórcio cederj

Geometria projetiva

Motivação

- Nas figura (a) vemos uma vista aérea de um pista e na figura (b) uma foto tomada de um ponto sobre o terreno.
- A figura (b) é uma ***transformação projetiva*** de (a).
- Preserva retas mas ***não preserva paralelismo***.


(a)



(b)

Geometria projetiva

Espaço projetivo

- O espaço projetivo de dimensão n, com origem em um ponto $O \in R^{n+1}$, é definido como o conjunto de retas que passam por O (menos a própria origem em O).

- O espaço projetivo de dimensão n é normalmente indicado por RP^n.

Geometria projetiva

Espaço projetivo

- A geometria projetiva é uma extensão da geometria euclidiana.
- É comum representar um ponto $(x_1,x_2,\ldots,x_{n+1})$ nesse espaço pela notação $(\boldsymbol{x}, x_{n+1})$ onde $\boldsymbol{x} \in R^n$.

Geometria projetiva

Espaço projetivo

- Sub-espaços de dimensão $m < n$ de RP^{n}, são identificados com sub-espaços de dimensão $m+1$ em R^{n+1}.

- **Exemplo:** retas projetivas em RP^2 são identificadas com planos que passam pela origem em R^3.

Geometria projetiva

Espaço projetivo

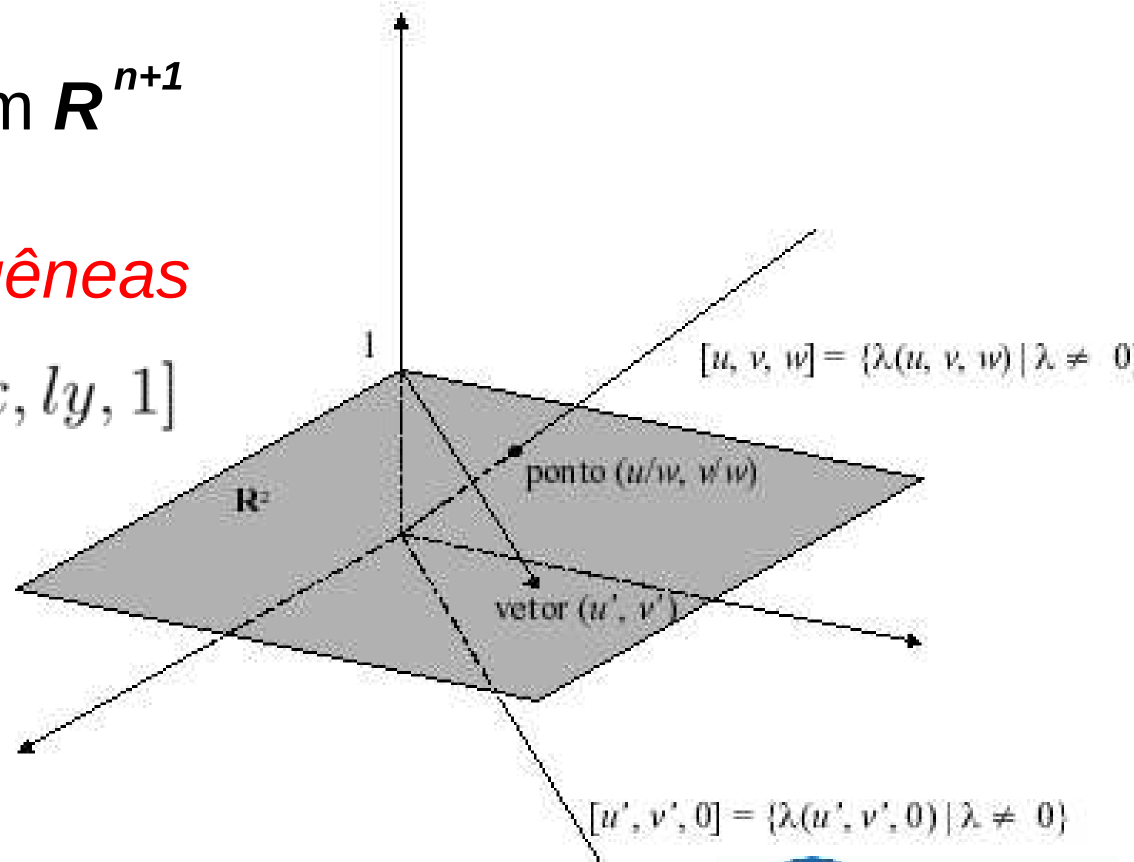
- Espaço projetivo $\boldsymbol{RP}^{n}$

$\updownarrow$

- Retas pela origem em $\boldsymbol{R}^{n+1}$

- *Coordenadas homogêneas*

$$(x, y) \leftrightarrow [x, y, 1] \cong [lx, ly, 1]$$



Geometria projetiva

Espaço projetivo

- A identificação de pontos do espaço afim com pontos do espaço projetivo leva a partição de ***RP*** ^{n} em dois conjuntos.

- Tomemos o ***RP*** ^{2} como base de nosso raciocínio:
  - $RP^2 = \{(x,1)\} \cup \{(x,0)\}$

Geometria projetiva

Espaço projetivo

- Os pontos da *{(x,1)}* são os pontos do plano euclidiano *z =1*. Esses pontos são chamados ***pontos afins*** ou próprios.

- O pontos da forma *{(x,0)}* são denominados ***pontos ideais*** ou do ***infinito***.

Geometria projetiva

Espaço projetivo

- ***Não existem retas paralelas na geometria projetiva.***

- Para isto, basta ver que uma reta projetiva é dada por uma plano que passa pela origem.

- Como dois desses planos sempre se interceptam então, duas retas projetivas nunca são paralelas.

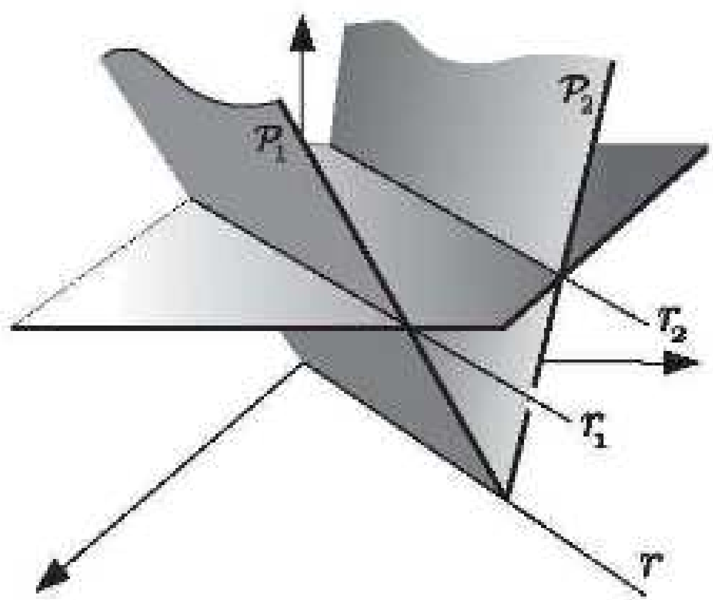
Geometria projetiva

Espaço projetivo

- O que ocorre com duas retas paralelas no plano afim?

  - Duas retas paralelas r_1 e r_2 no plano afim $z=1$ se interceptam em um ponto no infinito.

  - As retas r_1 e r_2 correspondem as retas projetivas P_1 e P_2 dados pelos planos na figura abaixo.



Geometria projetiva

Coordenadas homogêneas

- É fácil introduzir ***coordenadas*** para o espaço projetivo.
- Dado um ponto $p \in RP^n$, tomamos as coordenadas $(x_1,x_2,...,x_n,x_{n+1})$ de um ponto $p' \in R^{n+1}$ na reta que representa p.
- Desta forma tomamos coordenadas euclidianas para um ponto projetivo.

Geometria projetiva

Coordenadas homogêneas

- Entretanto, $\lambda p'$, $\lambda \in R$ e $\lambda \neq 0$, representa o mesmo ponto projetivo *p*.

- Desse modo, $\lambda (x_1, x_2, \ldots, x_n, x_{n+1})$ também representa coordenadas de *p*.

Geometria projetiva

Coordenadas homogêneas

- Isto significa que as coordenadas projetivas ***são definidas a menos de um fator escalar não nulo***.

- A notação utilizada na literatura para coordenadas homogêneas é:

$$[x_1, ..., x_n, x_{n+1}] = \lambda[x_1, ..., x_n, x_{n+1}]$$

Geometria projetiva

Transformações projetivas

- Uma transformação projetiva $T: RP^n \rightarrow RP^n$, leva pontos projetivos em pontos projetivos.

- *T* deve transformar uma reta passando pela origem em outra reta que passa pela origem.

- Logo *T* deve ser uma ***transformação linear invertível*** de R^{n+1} em R^{n+1}.

Geometria projetiva

Transformações projetivas

- Daí decorre que *T* preserva elementos lineares do espaço projetivo.

- *T* é representada por uma matriz de ordem *n+1*.

- *T* é definida a menos de um fator de escala, pois,
  $\lambda T(p) = T(\lambda p) = T(p), \lambda \in Re\lambda \neq 0$.

Geometria projetiva

Transformações projetivas

- Matriz que representa uma transformação
  $T: RP^2 \rightarrow RP^2$ no plano projetivo:

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Geometria projetiva

Transformações projetivas

- Suponha que a matriz seja dada por: $\begin{pmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{pmatrix}$
- Aplicando a um ponto do infinito *(x,y,0)* e a um ponto afim *(x,y,1)*, temos respectivamente:

$$\begin{pmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \\ 0 \end{pmatrix}=\begin{pmatrix} ax+cy+e \\ bx+dy+f \\ 0 \end{pmatrix} \text{ e } \begin{pmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}=\begin{pmatrix} ax+cy+e \\ bx+dy+f \\ 1 \end{pmatrix}$$

- Logo, leva pontos próprios em pontos próprios e pontos ideais em pontos ideais (***transformação afim***)

Geometria projetiva

Transformações projetivas

- Suponha que a matriz seja dada por: $\begin{pmatrix} a & c & e \\ b & d & f \\ r & s & 1 \end{pmatrix}$
  (com $r \neq 0$ ou $s \neq 0$)

- Aplicando a um ponto do infinito $(x,y,0)$ e a um ponto afim $(x,y,1)$, temos respectivamente:

$$\begin{pmatrix} a & c & e \\ b & d & f \\ r & s & 1 \end{pmatrix}\begin{pmatrix} x \\ y \\ 0 \end{pmatrix} = \begin{pmatrix} ax+cy+e \\ bx+dy+f \\ rx+sy \end{pmatrix} \text{ e } \begin{pmatrix} a & c & e \\ b & d & f \\ r & s & 1 \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} ax+cy+e \\ bx+dy+f \\ rx+sy+1 \end{pmatrix}$$

- Há pontos ideais levados em pontos afins e pontos afins levados em pontos ideais.

Geometria projetiva

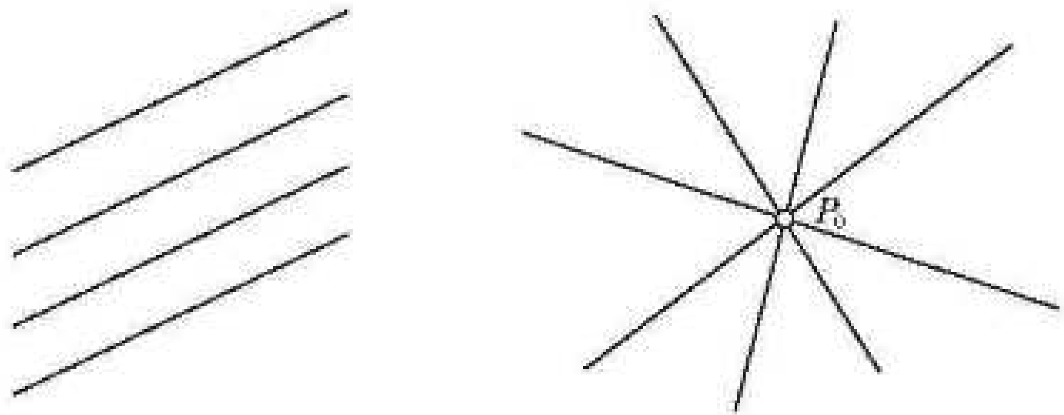
Transformações projetivas

- Suponha que um ponto ideal é levado em um ponto afim P_0.
- A família de retas paralelas, que se interceptam no ponto ideal, são transformadas em um feixe de retas que incidem em P_0.

Geometria projetiva

Transformações projetivas

- P_0 é denominado ***ponto de fuga da transformação***.



Ponto ideal transformado em ponto real.

Geometria projetiva

Transformações projetivas

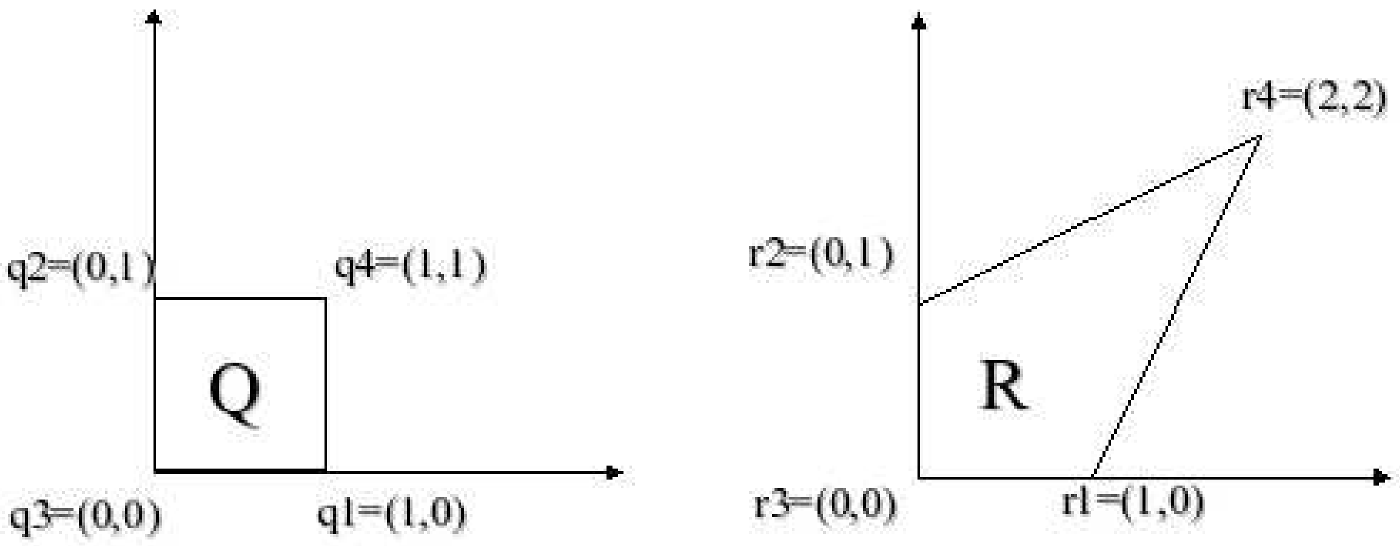
- Uma transformação projetiva *P* em RP^2 fica caracterizada quando são conhecidas as imagens por *P* de *4* pontos em "posição geral" (*3* quaisquer não estão em linha reta).

- Generalização para RP^n: $n+2$ pontos em posição geral.

Geometria projetiva

Transformações entre dois quadriláteros

- ***Transformar*** um quadrilátero *Q* em um quadrilátero *R*.



- **Solução:**

$$T = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ -1 & -1 & 3 \end{pmatrix}$$

Geometria projetiva

Verificação

$$T(1,0)=T(1,0,1)=\begin{pmatrix}2 & 0 & 0\\ 0 & 2 & 0\\ -1 & -1 & 3\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix}=\begin{pmatrix}2\\0\\2\end{pmatrix}=\begin{pmatrix}1\\0\\1\end{pmatrix}=\begin{pmatrix}1\\0\end{pmatrix}$$

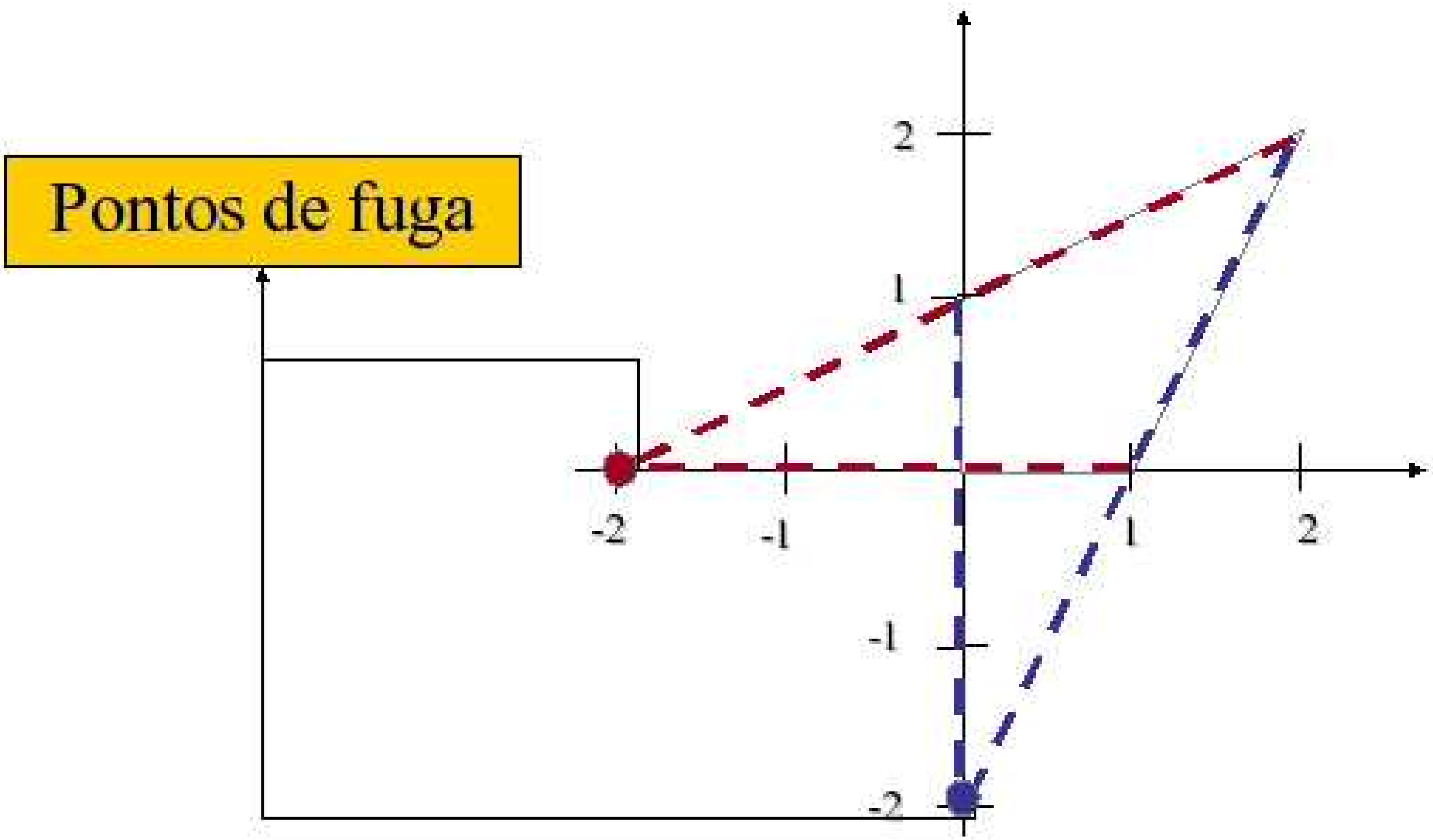
$$T(0,1)=T(0,1,1)=\begin{pmatrix}2 & 0 & 0\\ 0 & 2 & 0\\ -1 & -1 & 3\end{pmatrix}\begin{pmatrix}0\\1\\1\end{pmatrix}=\begin{pmatrix}0\\2\\2\end{pmatrix}=\begin{pmatrix}0\\1\\1\end{pmatrix}=\begin{pmatrix}0\\1\end{pmatrix}$$

$$T(0,0)=T(0,0,1)=\begin{pmatrix}2 & 0 & 0\\ 0 & 2 & 0\\ -1 & -1 & 3\end{pmatrix}\begin{pmatrix}0\\0\\1\end{pmatrix}=\begin{pmatrix}0\\0\\3\end{pmatrix}=\begin{pmatrix}0\\0\\1\end{pmatrix}=\begin{pmatrix}0\\0\end{pmatrix}$$

$$T(1,1)=T(1,1,1)=\begin{pmatrix}2 & 0 & 0\\ 0 & 2 & 0\\ -1 & -1 & 3\end{pmatrix}\begin{pmatrix}1\\1\\1\end{pmatrix}=\begin{pmatrix}2\\2\\1\end{pmatrix}=\begin{pmatrix}2\\2\end{pmatrix}$$

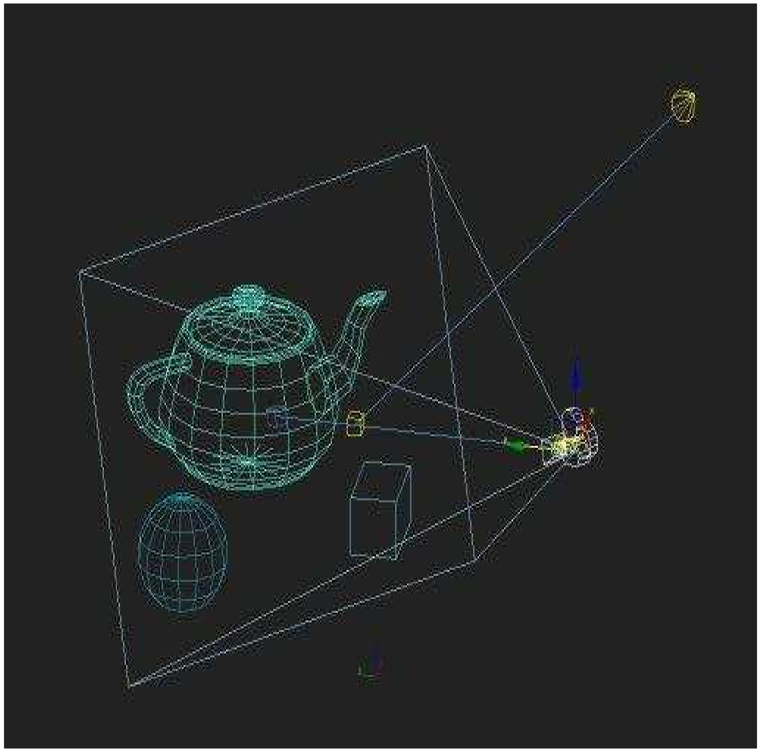
Geometria projetiva

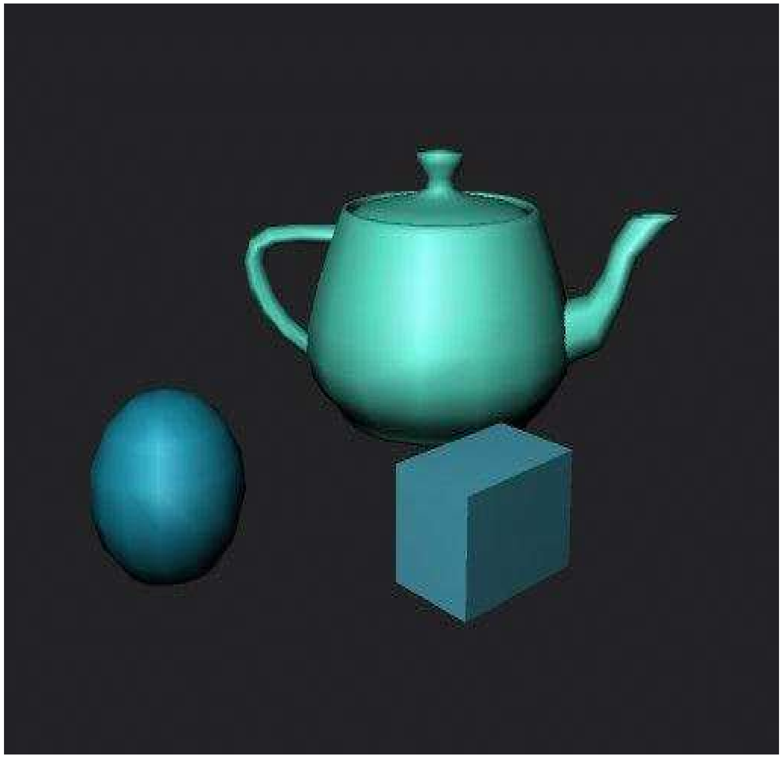
Pontos de fuga da transformação



Visualização de cenas 3D

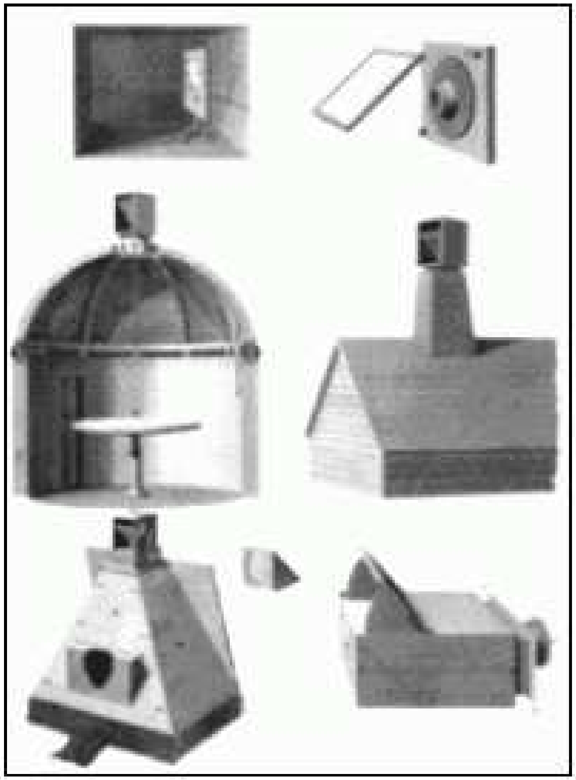
- A geração de imagens a partir de modelos de tridimensionais é uma das etapas mais importantes em Computação Gráfica.





Visualização de cenas 3D

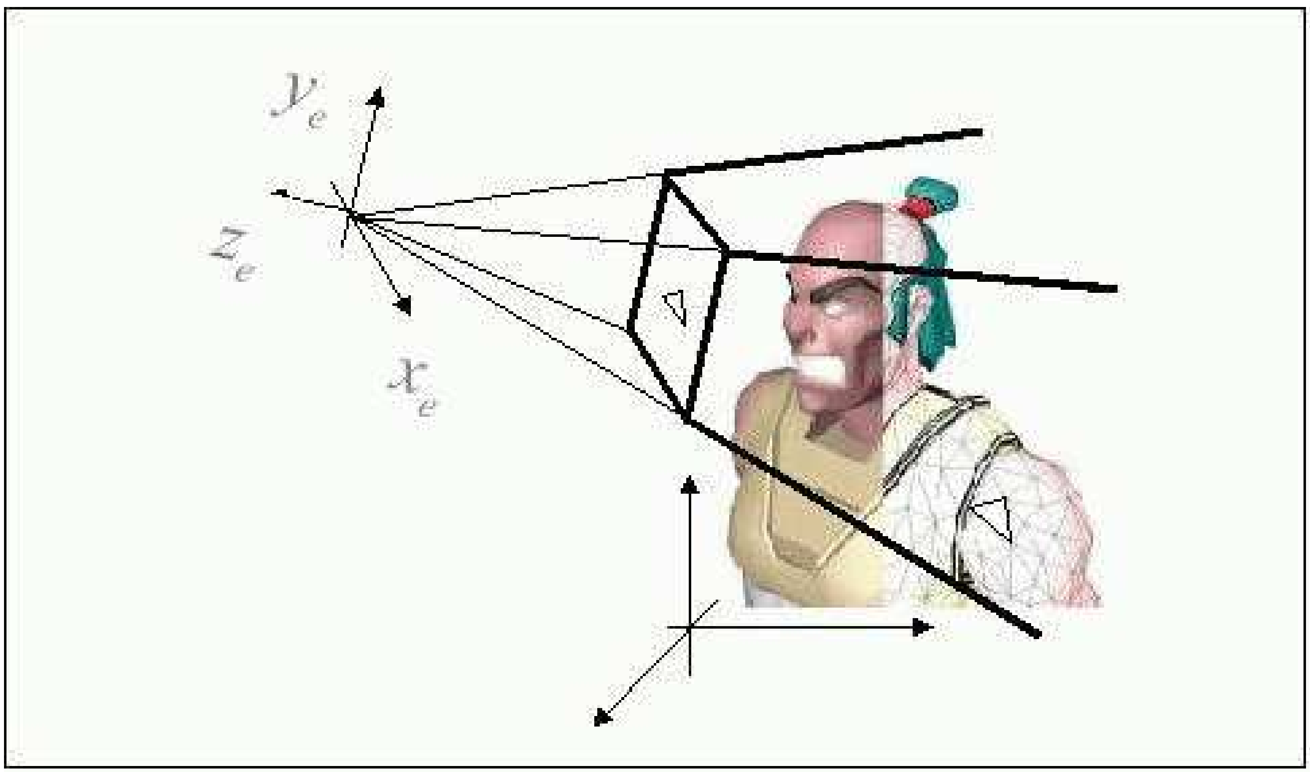
- O processo de geração de imagens a partir de cenas virtuais é ***análogo à geração de imagens através de câmeras fotográficas***.



Câmera virtual

- A diferença é que em C.G., os objetos, as luzes e a câmera são descritos por modelos matemáticos.

- Por este motivo, a câmera em C.G. é denominada ***câmera virtual***.



Câmeras e geometria projetiva

Coordenadas Homogêneas

- O modelo matemático que rege os processos de geração de imagens, tanto em câmeras reais quanto em câmeras virtuais é o de ***projeção***.

- Por este motivo, a ***Geometria Projetiva*** tem um papel fundamental na geração de imagens a partir de objetos tridimensionais.

Câmeras e geometria projetiva

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

Modelo de câmera

- Uma câmera pode ser caracterizada matematicamente através de seus parâmetros ***intrínsecos*** e ***extrínsecos***.

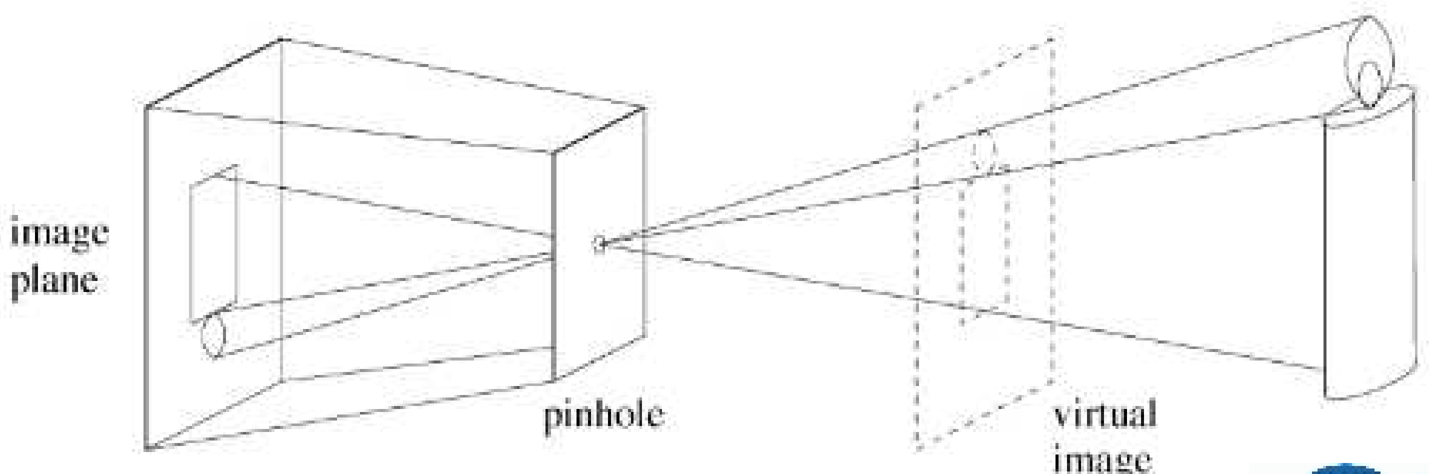
- Os ***parâmetros intrínsecos*** correspondem aos parâmetros internos da câmera como a distância focal, tamanho do pixel e as distorções de lente.

- Os **parâmetros extrínsecos** correspondem a orientação e posição da câmera em relação a um sistema de referência no mundo.

Modelo de câmera

Câmera de furo

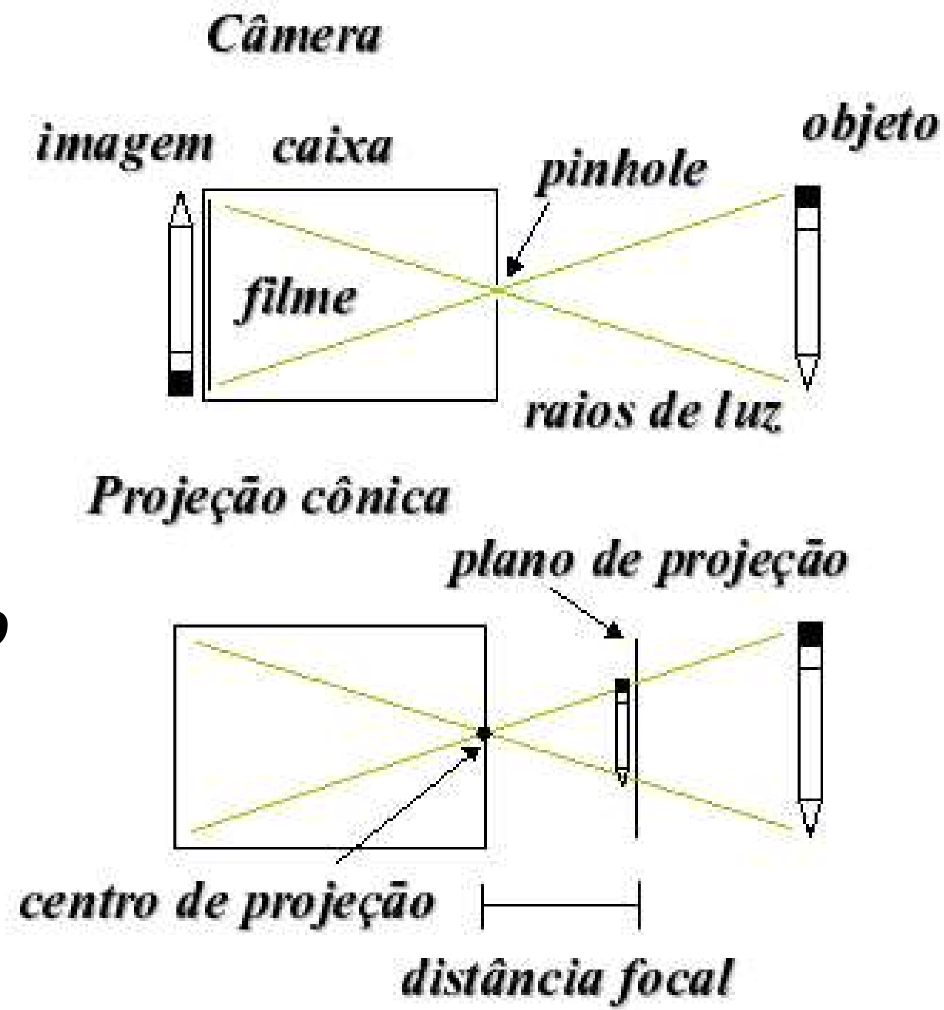
- O modelo que utilizaremos para a definição da câmera virtual é baseado em uma ***câmera de furo***.
- Neste modelo, a luz passa pelo orifício *O* em um dos lados de uma caixa e projeta a imagem do plano oposto.



Modelo de câmera

Câmera de furo

- A geometria do modelo de câmera de furo se reduz a ***projeção cônica***.
- Para evitar que a imagem seja invertida, deslocamos o plano de projeção que é posicionado entre o ***centro de projeção*** e o objeto.
- O único parâmetro intrínseco é a ***distância focal***.



Especificação de câmera virtual

- Existem diversas formas de especificar uma câmera virtual que segue o modelo da câmera de furo (pinhole).

- O modelo de câmera e o esquema utilizado para sua especificação, aqui apresentados, são baseados na câmera virtual utilizada na biblioteca *OpenGL*.

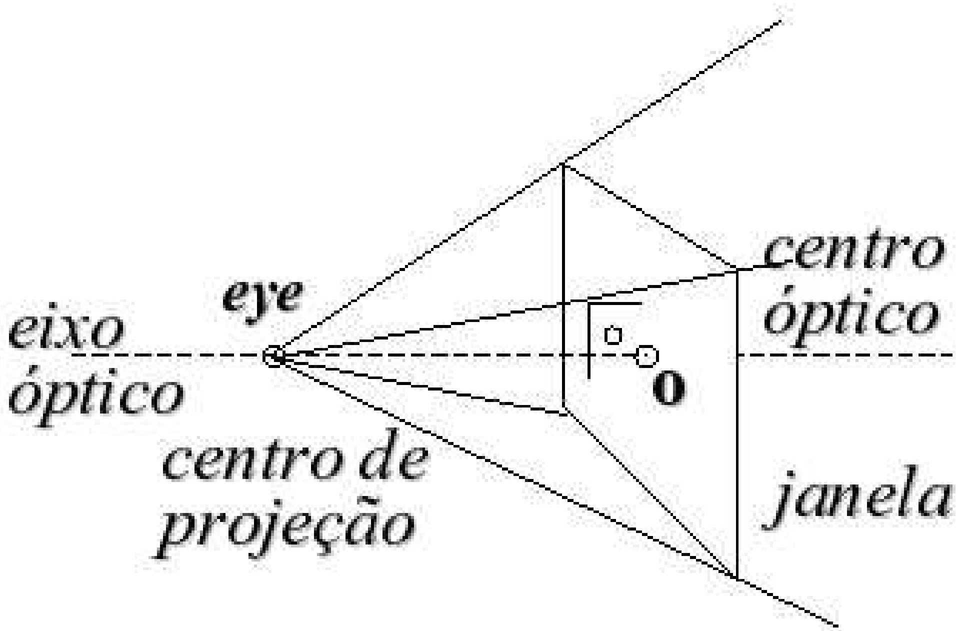
Modelo de câmera virtual

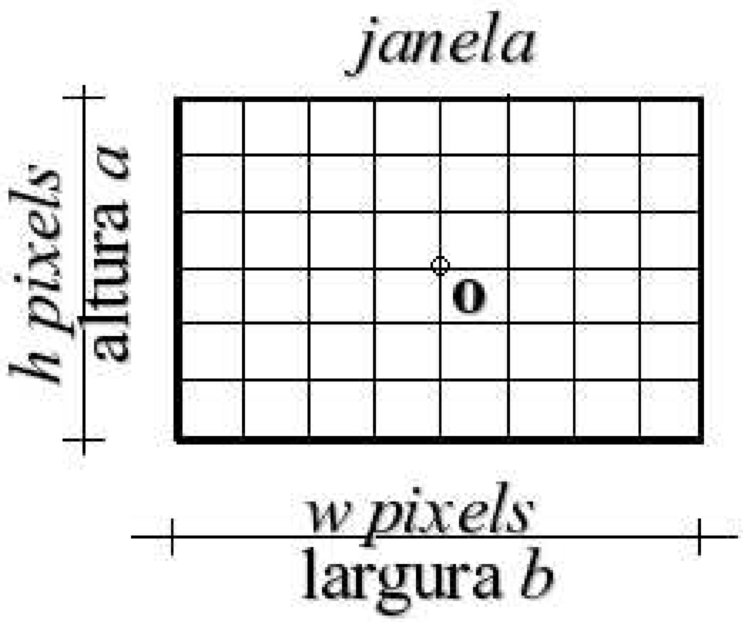
- Os parâmetro intrínsecos da câmera são definidos pelo ***centro de projeção***, o ***eixo óptico*** e as ***dimensões da tela virtual*** (um retângulo de *w x h* pixels).

- O eixo óptico é determinado pela reta que passa pelo centro de projeção e fura a tela virtual em um ponto denominado ***centro óptico*** ou ***ponto principal***.

Fundação CECIERJ Consórcio cederj

Modelo de câmera virtual



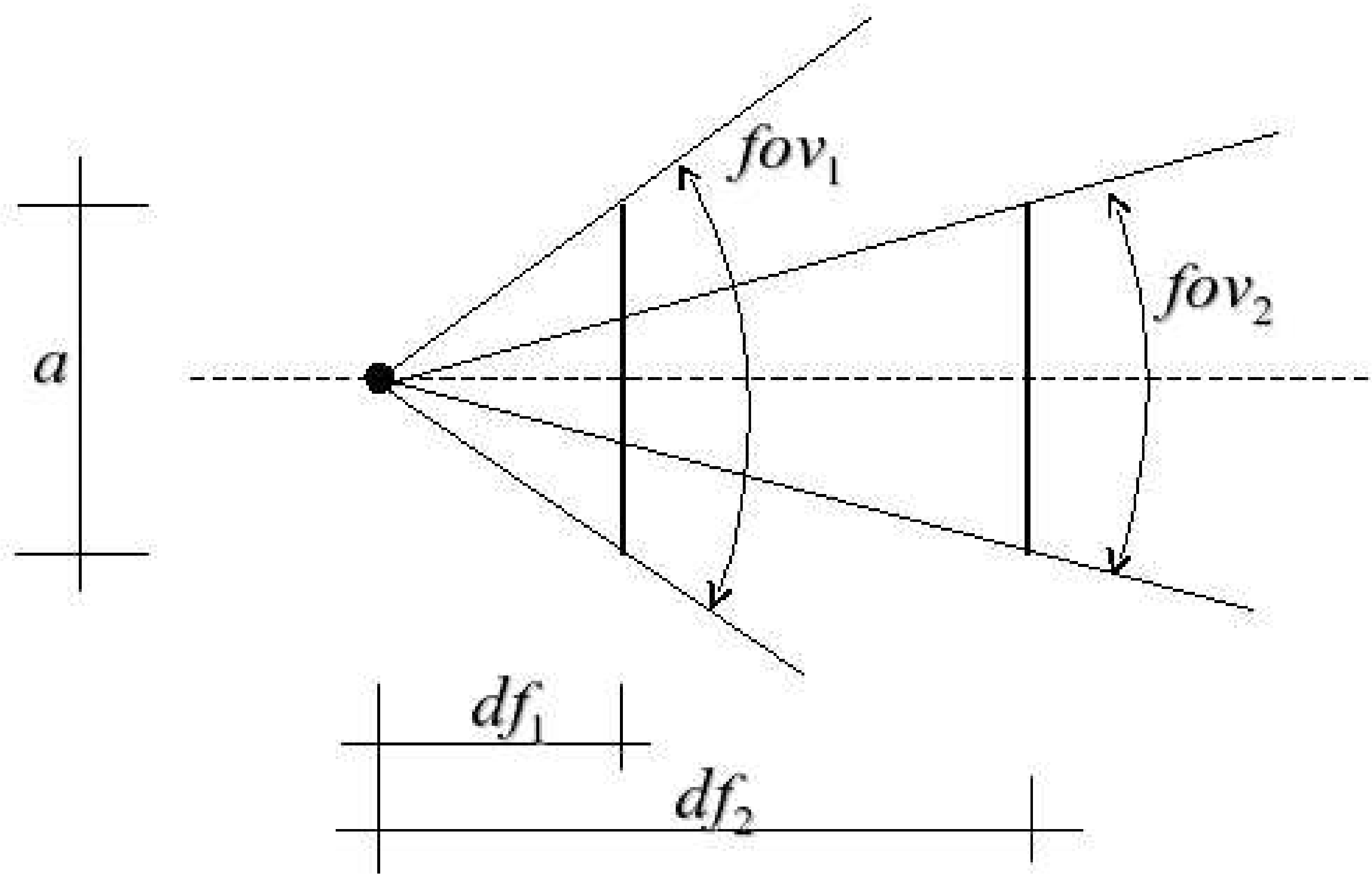


Modelo de câmera virtual

- Um caso bastante comum é aquele em que o eixo óptico é perpendicular à tela virtual e intercepta exatamente seu centro.

- Nestes casos o tamanho do retângulo e a sua distância ao centro de projeção definem a ***abertura da câmera*** ou ***campo de visão*** (*fov*).

Modelo de câmera virtual

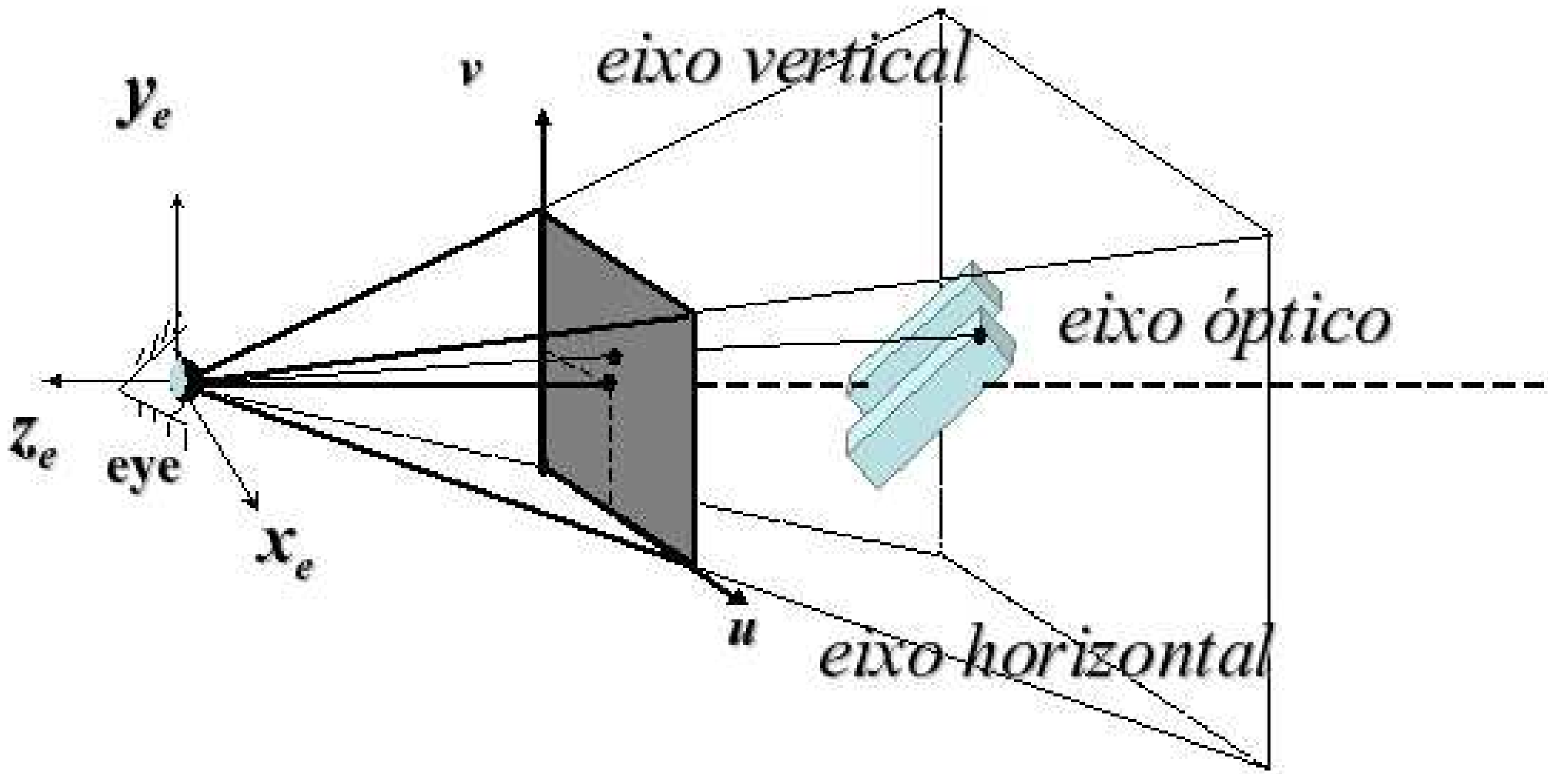


Modelo de câmera virtual

- O eixo óptico e as direções dos lados do retângulo da tela especificam três direções que definem os ***eixos da câmera*** $x_e y_e z_e$ e os ***eixos da imagem uv***.

- A escolha do eixo z_e voltado para trás é feito para que o referencial tenha orientação positiva (mão direita), isto é, seja ***dextrógiro***.

Modelo de câmera virtual



Especificação da câmera virtual

- Os seguintes parâmetros são parâmetros internos da câmera virtual:

  df - distância focal
  fov - campo de visão
  a e *b* - altura e largura da tela
  w e *h* - numero de pixels na horizontal e vertical

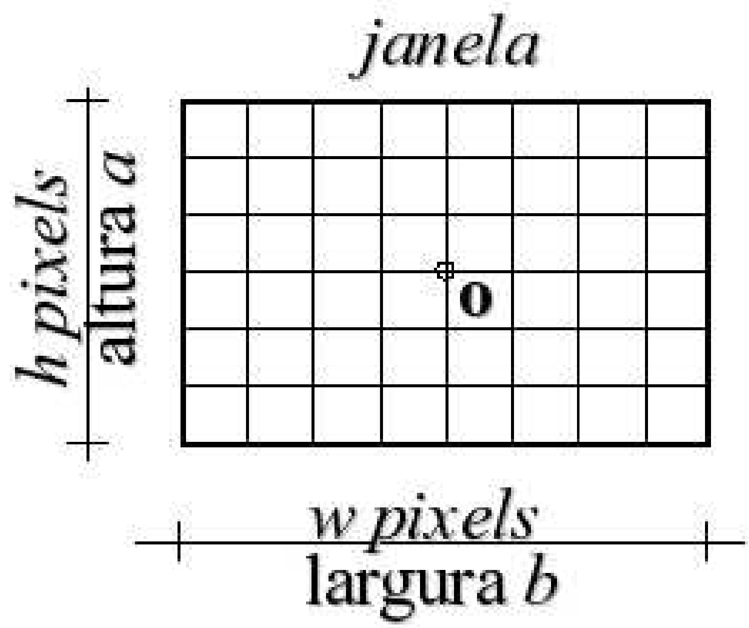
- Esses parâmetros ***não são independentes***.
- As relações entre os parâmetros permitem escolher quais especificam a câmera e quais ficam definidos automaticamente.

Especificação da câmera virtual

- Na maioria dos dispositivos o pixel é quadrado.

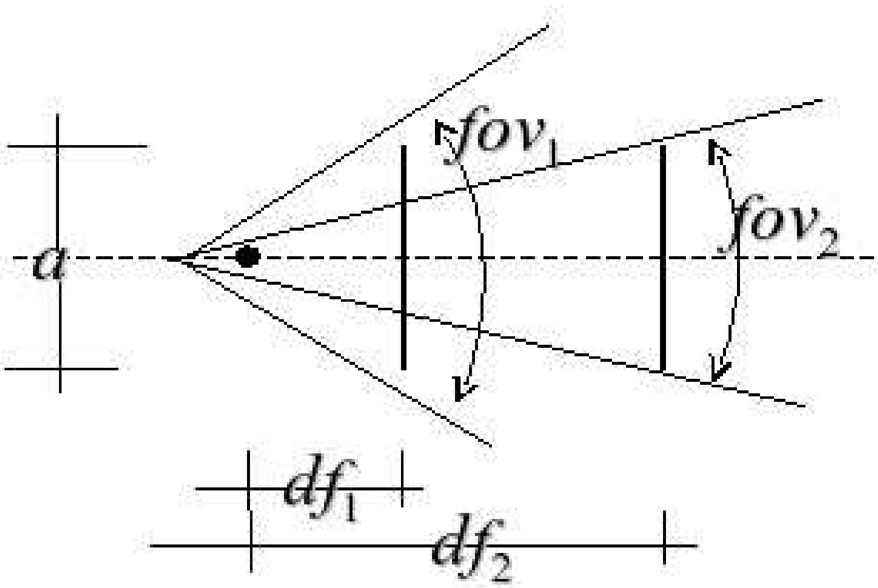
- Nestes casos, a razão entre os lados da janela é dada por:

$$b = \frac{w}{h} a$$



Especificação da câmera virtual

- Uma outra expressão importante é a que relaciona o campo de visão ***fov***, o número de pixels na vertical a e a distância focal *f*.



$$\frac{a}{2\,df} = \tan\left(\frac{fov}{2}\right)$$

$$a = 2\,df\,\tan\left(\frac{fov}{2}\right)$$

Especificação da câmera virtual

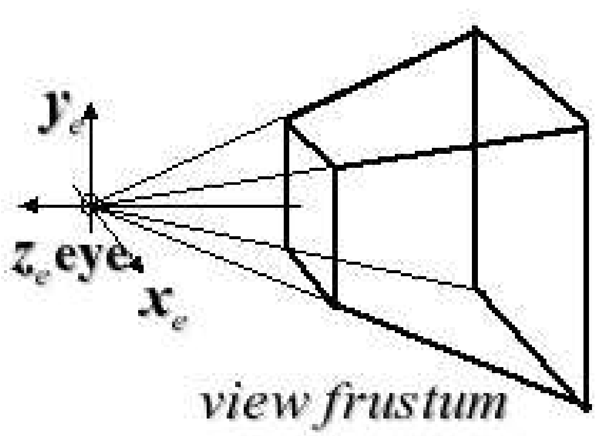
- Uma boa escolha para parametrizar uma câmera é utilizar os parâmetros *fov* e a razão de aspecto ***w/h*** entre a largura e altura da tela.

- Estes parâmetros, juntamente com duas distâncias *near* e *far* em relação ao centro de projeção são os parâmetros usados pela função da *OpenGL* :

```
void glPerspective(Gldouble fovy,Gldouble aspect,
Gldouble near, Gldouble far);
```

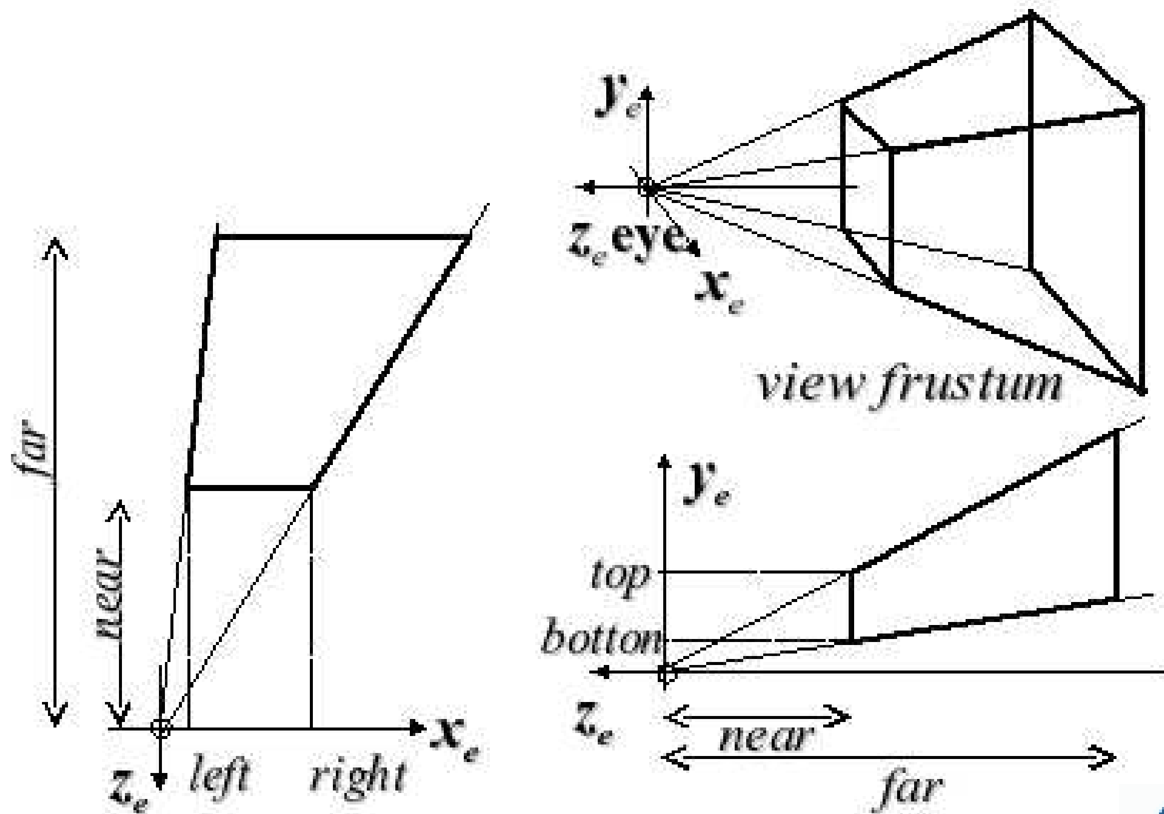
Especificação da câmera virtual

- Os parâmetros descritos determinam um ***volume de visualização (frustum)*** na forma de um tronco de pirâmide reta.



Especificação da câmera virtual

- Precisamos de um conjunto de parâmetros mais gerais quando o eixo ótico não atravessa o centro do plano de projeção.



Especificação da câmera virtual

- Podemos utilizar as coordenadas dos cantos inferior esquerdo (*left, bottom*) e superior direito (*right, top*) que definem a tela virtual, juntamente com os planos em *-near* e *-far*.

- Estes parâmetros são utilizados pela função *glFrustum* da biblioteca *OpenGL*:

```
void glFrustum(GLdouble left,GLdouble right, GLdouble
bottom, GLdouble top, GLdouble near, GLdouble far);
```

Sistema de coordenadas da câmera virtual

- Até o momento especificamos uma câmera na posição padrão, o que é de pouca utilidade.

- Precisamos agora definir os parâmetros externos para que possamos ***posicionar e orientar a câmera*** no espaço como um objeto qualquer.

Sistema de coordenadas da câmera virtual

- A posição da câmera é dada pelo próprio vetor *(eye_x,eye_y,eye_z)*.
- O eixo z_e é definido pelo vetor normalizado
- O eixo óptico, por sua vez, é dado pela reta que passa pelo ***eye*** e pelo ***center***.

$$z_e = \frac{1}{\|center - eye\|}(center - eye)$$

Sistema de coordenadas da câmera virtual

- O vetor ***Up*** é um vetor qualquer posicionado no plano $\boldsymbol{x_e y_e}$.

- Logo o vetor unitário $\boldsymbol{x_e}$ pode ser obtido através da normalização de $\boldsymbol{up \times z_e}$.

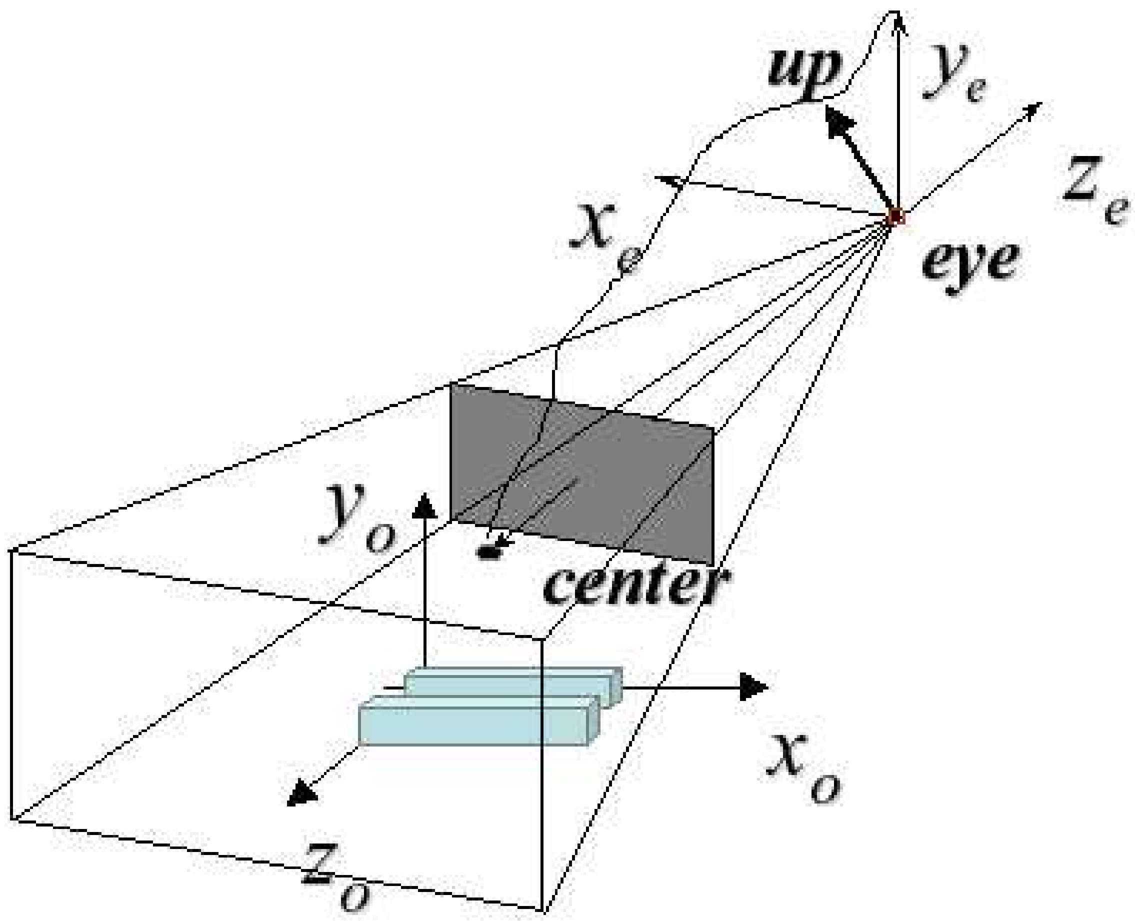
$$x_e = \frac{1}{\|up \times z_e\|}\left(up \times z_e\right)$$

Sistema de coordenadas da câmera virtual

- Finalmente, o vetor $\boldsymbol{y_e}$ é obtido pelo produto vetorial $\boldsymbol{x_e} \times \boldsymbol{z_e}$.
- O uso dos parâmetros *eye*, *center* e *up* na definição do sistema de coordenadas da câmera é o mesmo utilizado pela função ***gluLookAt*** da biblioteca utilitária ***GLU*** da *OpenGL*.

```
void gluLookAt(GLdouble eyex, GLdouble eyey, GLdouble
eyez, GLdouble centerx, GLdouble centery, GLdouble
centerz, GLdouble upx, GLdouble upy, GLdouble upz);
```

Sistema de coordenadas da câmera virtual



Transformações de visualização

- O processo de geração de imagens envolve, além do processo de projeção, ***transformações entre*** os seguintes ***sistemas de coordenadas***:

  - Sistema de coordenadas do objeto.
  - Sistema de coordenadas do mundo.
  - Sistema de coordenadas da câmera.
  - Sistema de coordenadas da tela.

Coordenadas do objeto para coordenadas do mundo

- A mudança das coordenas de um objeto no seu sistema próprio, para as coordenadas do sistema do mundo (global) é obtida através de uma matriz de transformação $\boldsymbol{M_{obj}}$.

- Estas transformações são exatamente as transformações de rotação, escala, cisalhamento e etc. que vimos anteriormente

Coordenadas do mundo para coordenadas da câmera

- A transformação das coordenadas do mundo para as coordenadas da câmera, dos vértices de uma primitiva, consiste na composição de duas transformações, nesta ordem:

  - ***Uma translação que leve o observador (eye) para a origem.***
  - ***Uma rotação que alinhe os eixos da câmera com os eixos do mundo.***

Fundação CECIERJ Consórcio cederj

Coordenadas do mundo para coordenadas da câmera

- A matriz que representa esta composição é:

$$L_{at} = R_{ew} \circ T_{ew} = \begin{bmatrix} x_{ex} & x_{ey} & x_{ez} & 0 \\ y_{ex} & y_{ey} & y_{ez} & 0 \\ z_{ex} & z_{ex} & z_{ex} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -eye_x \\ 0 & 1 & 0 & -eye_y \\ 0 & 0 & 1 & -eye_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- O entendimento da translação é trivial.

- A matriz de rotação $\boldsymbol{R_{ew}}$ é a matriz inversa da matriz $\boldsymbol{R_{we}}$ que transforma a base do sistema do mundo $\boldsymbol{x_w y_w z_w}$ para a base do sistema da câmera.

Coordenadas do mundo para coordenadas da câmera

- As colunas da matriz $\boldsymbol{R_{we}}$ são os vetores da base do mundo transformados para a base da câmera.

$$R_{we}=\begin{bmatrix} x_{ex} & y_{ex} & z_{ex} & 0 \\ x_{ey} & y_{ey} & z_{ey} & 0 \\ x_{ez} & y_{ez} & z_{ez} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, R_{ew}=R_{we}^{-1}=R_{we}^{T}=\begin{bmatrix} x_{ex} & x_{ey} & x_{ez} & 0 \\ y_{ex} & y_{ey} & y_{ez} & 0 \\ z_{ex} & z_{ex} & z_{ex} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Como $\boldsymbol{R_{we}}$ é ortogonal, sua inversa, isto é, $\boldsymbol{R_{ew}}$ é dada por $\boldsymbol{R_{we}^{T}}$.

A matriz *modelview*

- No sistema gráfico definido pela *OpenGL* existe uma única matriz que realiza a transformação de coordenadas do sistema do objeto para o sistema da câmera.

- Esta matriz é denominada ***Modelview***.

A matriz *modelview*

- Ela realiza as transformações de modelagem e a transformação das coordenadas do mundo nas coordenadas da câmera.

- Logo, a ***modelview*** é dada por $M_{view}=L_{at} \circ M_{obj}$, isto é, a composição da matriz de transf. de câmera com a matriz de modelagem do objeto.

A matriz *modelview*

- Como na OpenGL acumulamos as matrizes através de uma multiplicação à direita, devemos:

  - Definir primeiramente a ***transformação de câmera*** através da ***glLookAt***, por exemplo.
  - Aplicar as ***transformações de modelagem*** aos objetos.

- Quando vários objetos precisam ser instanciados podemos utilizar o esquema de pilha para armazenar matrizes que serão posteriormente recuperadas.

A transformação de projeção

- A transformação de projeção cônica pode ser facilmente definida na posição canônica.
- Para projetarmos um ponto genérico no plano ***near***, basta escalarmos as coordenadas por um fator que leve a coordenada ***z*** para a posição ***-n***.
- Ou seja, as coordenadas do ponto p_p na projeção de um ponto p são:

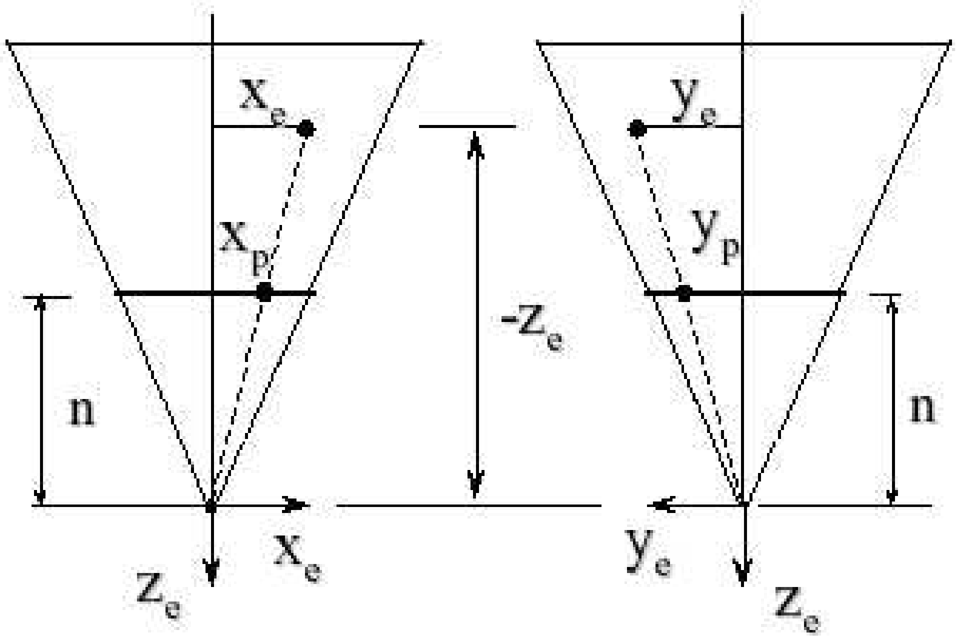
$$p_p = \begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} = \frac{n}{-z_e} \begin{pmatrix} x_e \\ y_e \\ z_e \end{pmatrix}$$

A transformação de projeção

- Esta mesma transformação também pode ser derivada por ***semelhança de triângulos***.

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

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



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

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

A transformação de projeção

- Finalmente, podemos escrever a transformação como uma transformação projetiva (em coordenadas homogêneas), conforme abaixo:

$$\begin{bmatrix} wx_p \\ wy_p \\ wz_p \\ w \end{bmatrix} = \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} nx_e \\ ny_e \\ nz_e \\ -z_e \end{bmatrix} \rightarrow \begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} = \frac{1}{-z_e} \begin{pmatrix} nx_e \\ ny_e \\ nz_e \end{pmatrix}$$

A transformação de projeção

- Um dos problemas com esta formulação é o de que ***perdemos a informação de profundidade dos pontos*** já que as coordenadas z são levadas no plano $z = -n$.

- Por este motivo, ***não seremos capazes de determinar quando uma superfície está a frente de uma outra.***

A transformação de projeção

- Para evitar este problema utilizamos uma transformação projetiva que ***leva o centro de projeção para o infinito***.

- Este processo ***transforma a transformação projetiva em uma transformação paralela ortográfica***.

- Lembremos que uma projeção cônica pode ser definida como a composição de uma transformação projetiva com uma transformação paralela.

A transformação de projeção

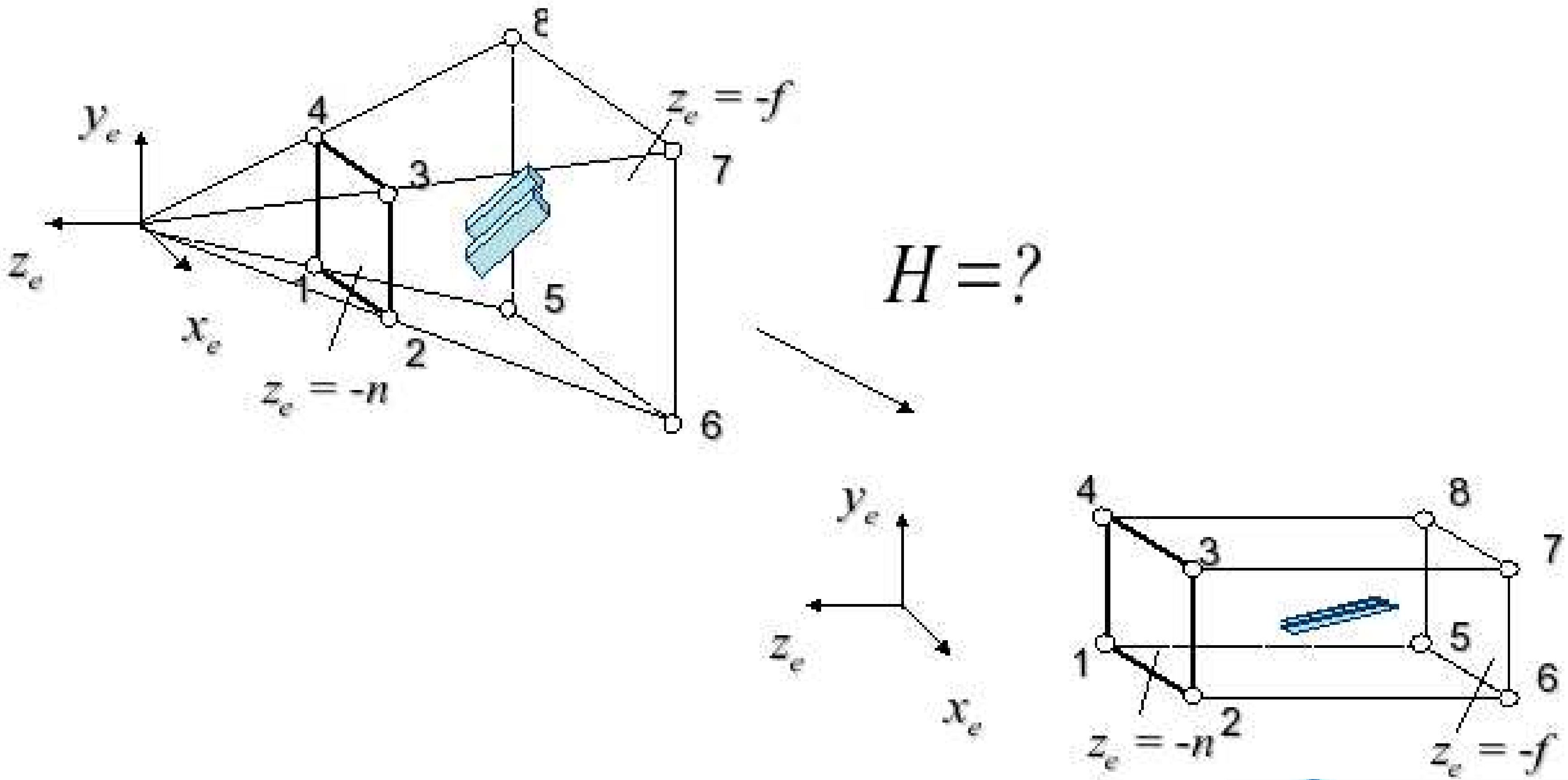
- ***A transformação projetiva torna os raios projetores paralelos.***

- As coordenadas *x* e *y* dos vértices paralelos ao plano de projeção tem seus valores determinados corretamente.

- Além disso, ***as profundidades relativas são preservadas na coordenada z***.

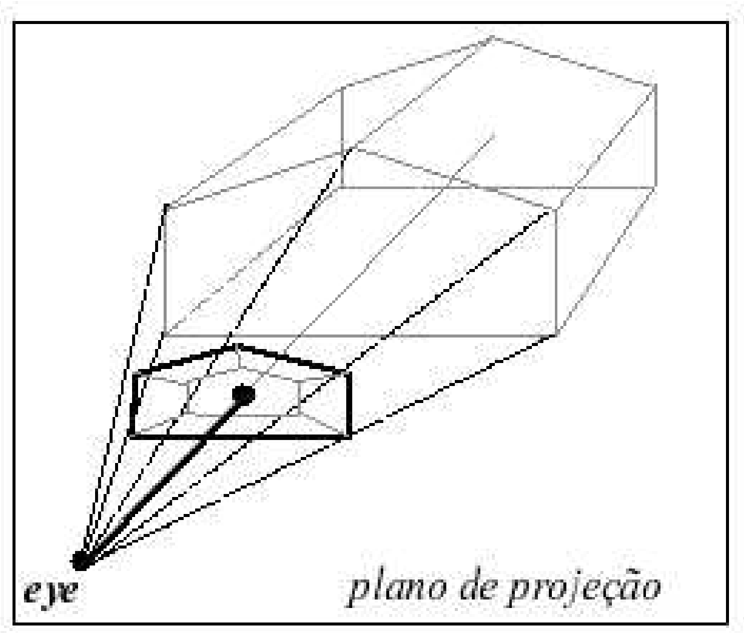
A transformação de projeção

- Vejamos como determinar a matriz que transforma um tronco de pirâmide em um paralelepípedo.

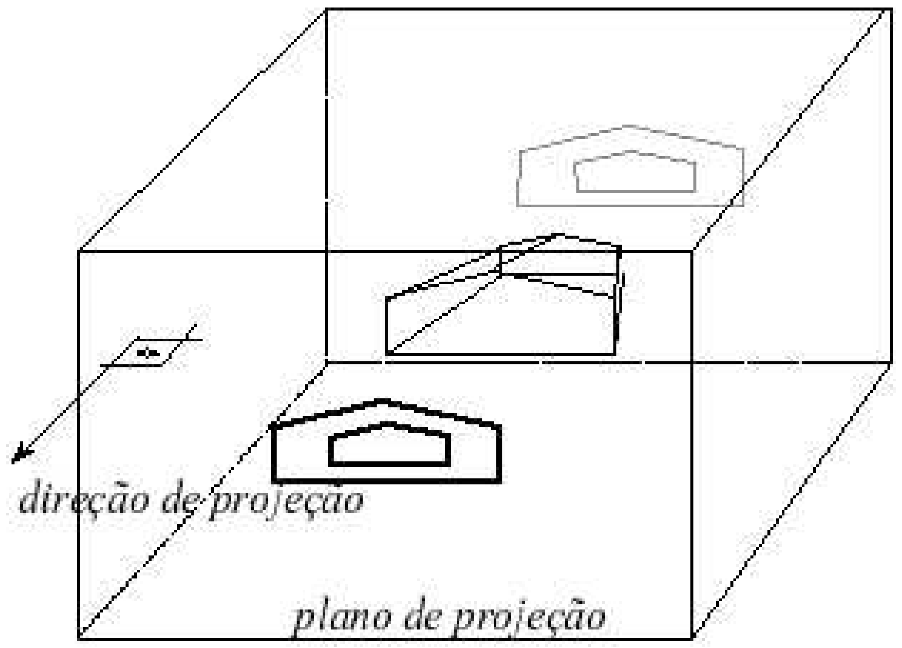


A transformação de projeção

Projeção cônica



Projeção ortográfica



A transformação de projeção

- Para determinar a matriz, partimos de uma matriz de transformação projetiva genérica em coordenadas homogêneas.

$$H = \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & nf \\ 0 & 0 & -1 & 0 \end{bmatrix} \qquad \begin{bmatrix} -nx/z \\ -ny/z \\ -(n+f)-nf/z \end{bmatrix} = \begin{bmatrix} nx \\ ny \\ (n+f)z+nf \\ -z \end{bmatrix} = \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & nf \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- (x_i,y_i,z_i) são as coordenadas cartesianas do ponto e (x_i',y_i',z_i') são as coordenadas do ponto transformado.

Fundação CECIERJ Consórcio cederj

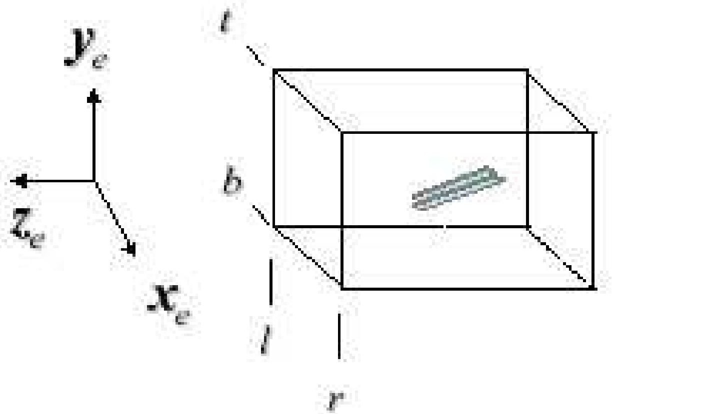
Transformação de normalização

- Para efetuar operações de forma mais simples, normalizamos o paralelepípedo de visão de forma que se torne um cubo definido por [-1,1] x [-1,1] x [-1,1].
- Para isso, aplicamos as seguintes transformações:

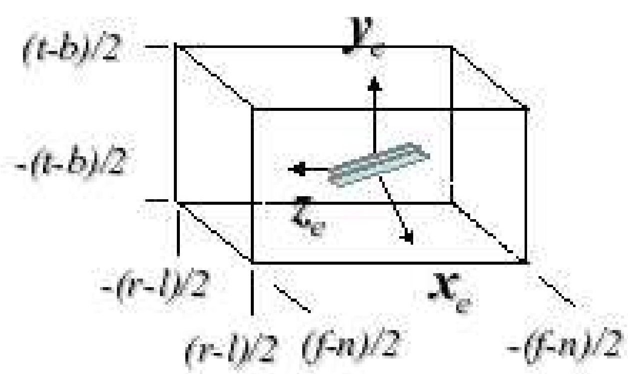
  - ***Transladamos*** o centro do paralelepípedo para a origem.
  - ***Aplicamos uma escala*** sobre paralelepípedo de forma que se torne um cubo normalizado.
  - ***Espelhamos*** o cubo resultante em relação ao plano xy para que os menores ***z*** representem os pontos mais próximos.

Fundação CECIERJ
Consórcio cederj

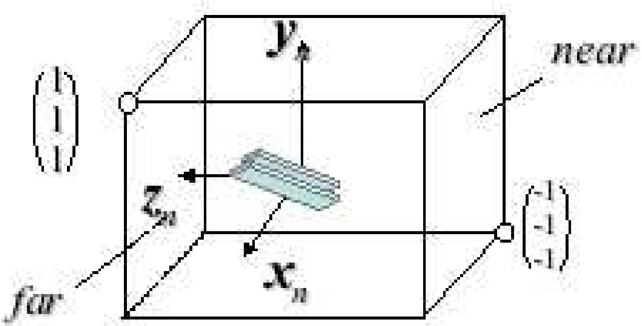
Transformação de normalização



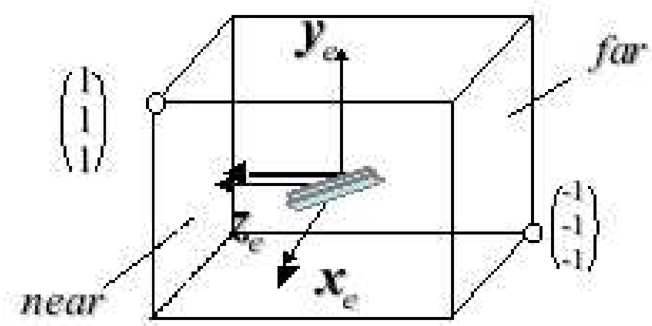
$$\begin{bmatrix} 1 & 0 & 0 & -(r+l)/2 \\ 0 & 1 & 0 & -(t+b)/2 \\ 0 & 0 & 1 & (f+n)/2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$\begin{bmatrix} 2/(r-l) & 0 & 0 & 0 \\ 0 & 2/(t-b) & 0 & 0 \\ 0 & 0 & 2/(f-n) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



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



Transformação de normalização

- A matriz associada a ***transformação de normalização*** é dada pela seguinte composição:

$$N = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2/(r-l) & 0 & 0 & 0 \\ 0 & 2/(t-b) & 0 & 0 \\ 0 & 0 & 2/(f-n) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -(r+l)/2 \\ 0 & 1 & 0 & -(t+b)/2 \\ 0 & 0 & 1 & (f+n)/2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$N = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

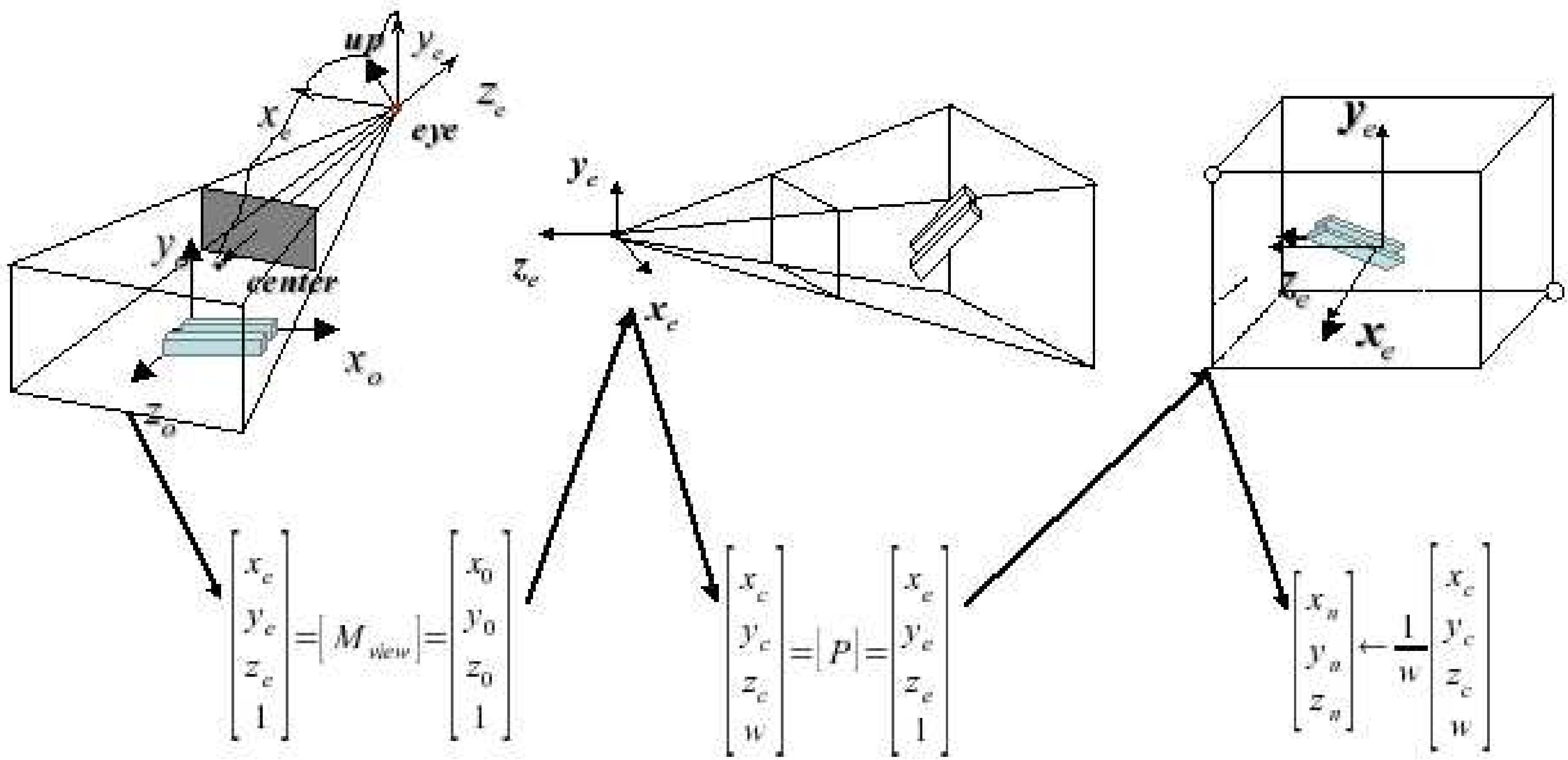
Transformação de normalização

- Multiplicando a matriz de normalização pela matriz de projeção chegamos à

$$P = NH = \begin{bmatrix} \frac{2}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2n}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

- Esta é a matriz que a especificação da OpenGL apresenta com a ***matriz correspondente a função glFrustum.***

Resumo

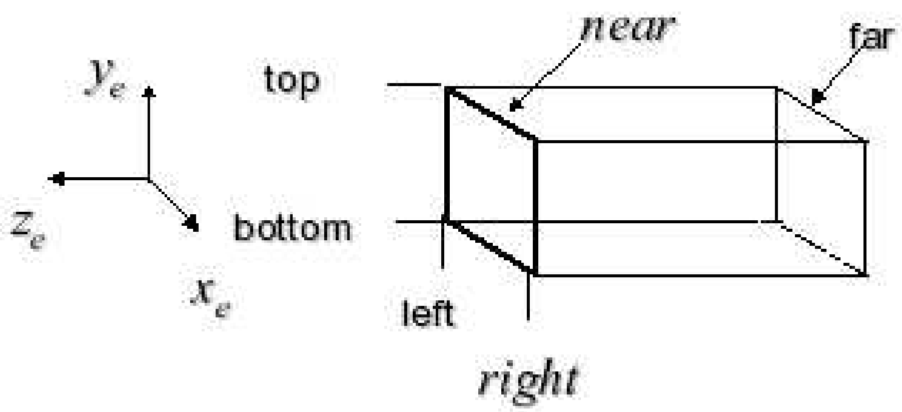


up
y_e
z_e
x_e
eye
y_o
center
x_o
z_o
y_e
z_e
x_e
y_e
z_e
x_e
$\begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} M_{view} \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$
$\begin{bmatrix} x_c \\ y_c \\ z_c \\ w \end{bmatrix} = \begin{bmatrix} P \end{bmatrix} = \begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix}$
$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} \leftarrow \frac{1}{w} \begin{bmatrix} x_c \\ y_c \\ z_c \\ w \end{bmatrix}$

Projeção ortográfica

- Na projeção ortográfica, os raios projetores não convergem para um centro de projeção.

- Ao contrário, são paralelos ao eixo *z* e ortogonais ao plano de projeção *z*=*near*.



Projeção ortográfica

- O paralelepípedo de visão associado a um projeção ortográfica da *OpenGL* é o mesmo que resulta da transformação do troco de pirâmide pela transformação *H*.

- Logo, a matriz de projeção paralela, simplemente ***leva o paralelepípedo para o cubo no espaço normalizado.***

Projeção ortográfica

- A matriz de projeção paralela é a seguinte:

$$N = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- As funções da OpenGL que produzem tal matriz são:

```
glOrtho(GLdouble left,GLdouble right,  GLdouble bottom,
GLdouble top, GLdouble near, GLdouble far)
```

```
glOrtho2D(GLdouble left,GLdouble right,  GLdouble bottom,
GLdouble top)
```

Fundação CECIERJ Consórcio cederj

Projeção ortográfica

- A projeção ortográfica, apesar de não ser tão realista tem muitas aplicações em engenharia e arquitetura.

- Ela ***preserva paralelismo entre linhas e permite a definição de escala*** tornando possível a tomada de medidas diretamente sobre a planta.

Fundação CECIERJ Consórcio cederj

Aula 10

Professores:

Anselmo Montenegro
Esteban Clua

Conteúdo:

- Projeções e câmera virtual