

# Viewing Pipeline 3D

Maria Cristina F. de Oliveira  
Rosane Minghim  
2010

## Viewing (onde estamos no pipeline)

- Pipeline
  - Geração dos modelos do objetos  
Modelagem (*modeling*)  
(geometria + topologia)
  - Posicionamento na Cena e Descrição da Cena (*Viewing*)
  - Rendering  
(Geração da Imagem)

## Viewing

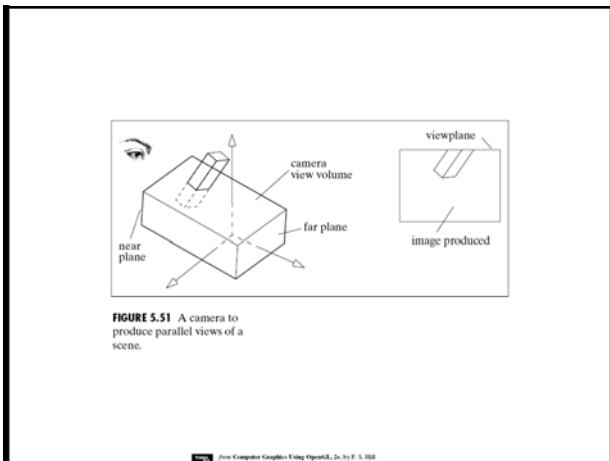
- 2D: Modelo da Cena 2D/Janela 2D/recorte 2D/viewport
- 3D: Posicionamento da câmera, volume de visualização, recorte 3D, projeção, *viewport*
  - outros elementos, como iluminação, remoção de superfícies ocultas, *depth cueing*
  - *Depth cueing* = percepção de profundidade: uso de *shading*, *textura*, *cor*, *fog*, etc.) para dar uma indicação da distância de um objeto ou superfície em relação ao observador
  - <http://www.vis.uni-stuttgart.de/depthcue/>

## Viewing Pipeline (OpenGL)

- Todos os vértices, de todos os objetos, entram no 'pipeline gráfico'
- Cada vértice é processado por três matrizes:
  - Modelview
  - Projeção
  - Viewport

FIGURE 5.52 The OpenGL pipeline (slightly simplified).

FIGURE 5.50 Simple viewing used in OpenGL for 2D drawing.



## Viewing Pipeline 3D

- No caso 3D, o *pipeline* requer:
  - A definição de um volume de interesse na cena 3D (SRU)
  - O mapeamento de seu conteúdo para o SRV (transformação de visualização)
  - A projeção do conteúdo do volume de interesse em um plano (transformação de projeção)
  - Mapeamento da janela resultante na *viewport* normalizada e depois para coordenadas do dispositivo

8

## Viewing Pipeline 3D: Analogia Câmera

- Imaginamos um observador que vê a cena através das lentes de uma câmera virtual
  - "fotógrafo" pode definir a posição da câmera, sua orientação e ponto focal, abertura da lente...
  - câmera real obtém uma projeção de parte da cena em um plano de imagem 2D (o filme)
- Analogamente, a imagem obtida da cena sintética depende de vários parâmetros que determinam como esta é projetada para formar a imagem 2D no monitor
  - posição da câmera, orientação e ponto focal, tipo de projeção, posição dos "planos de recorte" (*clipping planes*), ...

9

## Viewing Pipeline 3D: Analogia Câmera

- Três parâmetros definem completamente a câmera
  - Posição: aonde a câmera está
  - Ponto focal: para onde ela está apontando
  - Orientação: controlada pela posição, ponto focal, e um vetor denominado *view up*
- Outros parâmetros
  - Direção de projeção: vetor que vai da posição da câmera ao ponto focal
  - Plano da imagem: plano no qual a cena será projetada, contém o ponto focal e, tipicamente, é perpendicular ao vetor direção de projeção

10

## Viewing Pipeline 3D: Analogia Câmera

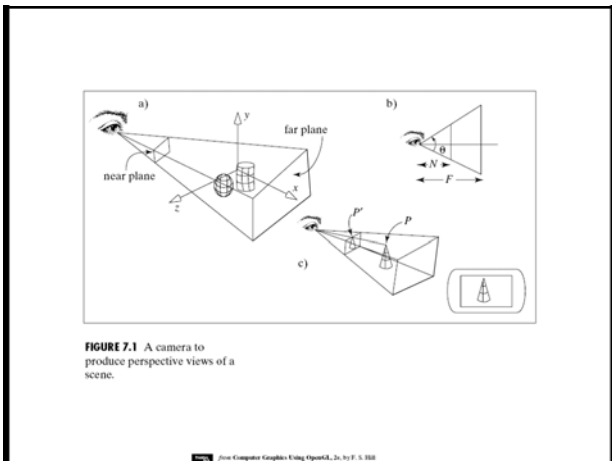
	With a Camera	With a Computer
Observação:	posiciona câmera	posiciona volume de observação
Cena:	posiciona modelo	posiciona modelo
Projeção:	escolhe lentes	escolhe formato vv
Viewport:	escolhe tamanho foto	escolhe porção da tela

fonte: curso CG Arizona State University, Dianne Hansford

## Viewing Pipeline 3D: Analogia Câmera

Fonte Figura: Schröder, The Visualization Toolkit, 1998

12



## Viewing Pipeline 3D: Analogia Câmera

- O método de projeção controla como os objetos da cena (atores) são mapeados no plano de imagem
  - Projeção ortográfica, ou paralela: processo de mapeamento assume a câmera no infinito, i.e., os raios de luz que atingem a câmera são paralelos ao vetor de projeção
  - Projeção perspectiva: os raios convergem para o ponto de observação, ou centro da projeção. Nesse caso, é necessário determinar o ângulo de visão da câmera
  - Os planos de recorte delimitam a região de interesse na cena
    - Anterior (near plane): elimina objetos muito próximos da câmera
    - Posterior (far plane): elimina objetos muito distantes

14

## Viewing Pipeline 3D

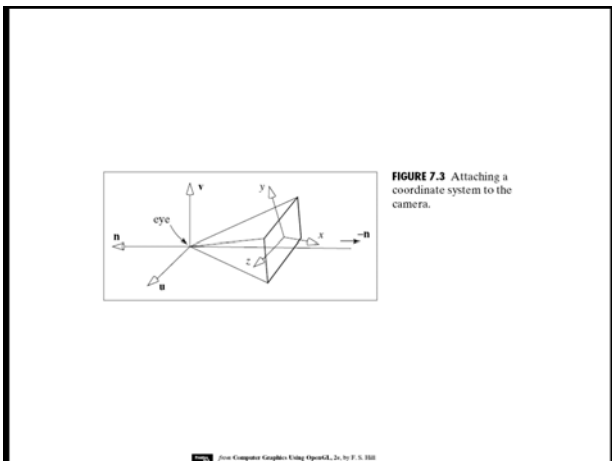
- Retomando: o *pipeline* requer a transformação da cena especificada no WCS (SRU) para o VCS (SRV)
  - O SRV descreve a cena como vista pela câmera...
  - O primeiro passo nesse processo consiste em especificar o VCS. Como?
    - Necessário especificar origem e os três eixos de referência...

15

## Especificação do VCS

- Origem do sistema
  - Posição da câmera (PRP: Projection Reference Point)
- Associados à câmera:
  - Vetor direção de projeção (**N**), que dá a direção do ponto focal, e vetor view-up (**V**), que indica o 'lado de cima' da câmera (ambos devem ser perpendiculares entre si!)
  - Plano de imagem, no qual a cena 3D será projetada, perpendicular ao vetor direção de projeção
- Eixos:
  - eixo z associado ao vetor direção de projeção, eixo y associado ao vetor view-up, eixo x...

16



## Especificação do VCS

- Dados os vetores **N** e **V**, os vetores unitários podem ser calculados como indicado ao lado

$$\mathbf{n} = \frac{\mathbf{N}}{|\mathbf{N}|} = (n_1, n_2, n_3)$$

$$\mathbf{u} = \frac{\mathbf{V} \times \mathbf{N}}{|\mathbf{V} \times \mathbf{N}|} = (u_1, u_2, u_3)$$

$$\mathbf{v} = \mathbf{n} \times \mathbf{u} = (v_1, v_2, v_3)$$

18

## Conversão WCS->VCS

- Temos 2 espaços vetoriais (sist. coordenadas) em  $\mathcal{R}^3$ , definidos por duas bases ortonormais
  - WCS, espaço  $x_w, y_w, z_w$  ( $\mathbf{i}, \mathbf{j}, \mathbf{k}$ )
  - VCS, espaço  $x_v, y_v, z_v$  ( $\mathbf{u}, \mathbf{v}, \mathbf{n}$ )
- Para transformar a descrição geométrica dos objetos do WCS para o VCS: aplicamos a transformação que alinha os eixos do VCS com os eixos do WCS

19

## Conversão WCS->VCS

- A origem do VCS está em  $P = (x_0, y_0, z_0)$  no WCS
  - translada para coincidir as origens
  - Aplica a matriz de rotação necessária para alinhar os eixos

20

## Conversão WCS->VCS

- Matriz de translação (alinha a origem do VCS a do WCS)

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 1 & -z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

21

## Conversão WCS->VCS

- Matriz de rotação (alinha os eixos do VCS aos do WCS) é dada por

$$\mathbf{R} = \begin{bmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

22

## Conversão WCS->VCS

- A matriz completa de transformação é

$$\mathbf{M}_{\text{WCS, VCS}} = \mathbf{R} \cdot \mathbf{T} = \begin{bmatrix} u_x & u_y & u_z & -\mathbf{u} \cdot \mathbf{P}_0 \\ v_x & v_y & v_z & -\mathbf{v} \cdot \mathbf{P}_0 \\ n_x & n_y & n_z & -\mathbf{n} \cdot \mathbf{P}_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

23

## Manipulação da Câmera



Fonte Figura: Schröder, The Visualization Toolkit, 1998

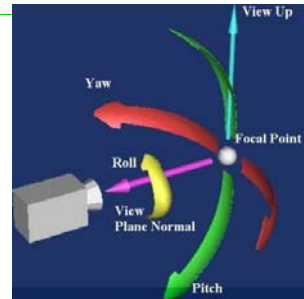
24

## Manipulação da Câmera

- *Azimuth*: rotaciona a posição da câmera ao redor do seu vetor *view up*, com centro no ponto focal
- *Elevation*: rotaciona a posição ao redor do vetor dado pelo produto vetorial entre os vetores *view up* e direção de projeção, com centro no ponto focal
- *Roll (Twist)*: rotaciona o vetor *view up* em torno do vetor normal ao plano de projeção

25

## Manipulação da Câmera



Fonte Figura: Schröder, The Visualization Toolkit, 1998

26

## Manipulação da Câmera

- *Yaw*: rotaciona o ponto focal em torno do vetor *view up*, com centro na posição da câmera
- *Pitch*: rotaciona o ponto focal ao redor do vetor dado pelo produto vetorial entre o vetor *view up* e o vetor direção de projeção, com centro na posição da câmera
- *Dolly (in, out)*: move a posição ao longo da direção de projeção (mais próximo ou mais distante do ponto focal)
- *Zoom (in, out)*: altera o ângulo de visão, de modo que uma região maior ou menor da cena fique potencialmente visível

27

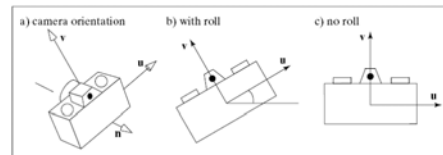
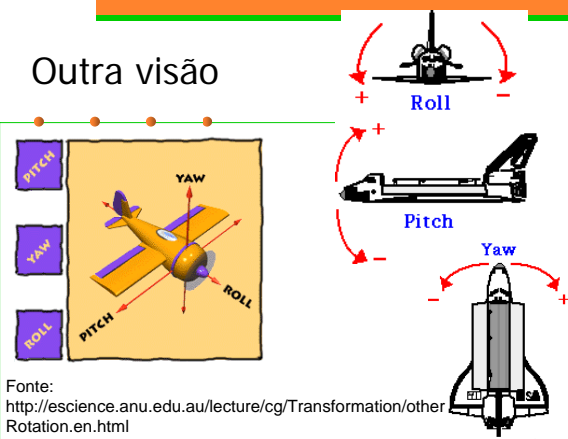


FIGURE 7.6 Various camera orientations.

From Computer Graphics Using OpenGL, 2e, by F. S. Hill

## Outra visão



Fonte:  
<http://escience.anu.edu.au/lecture/cg/Transformation/other/Rotation.en.html>

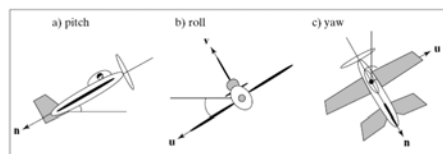


FIGURE 7.4 A plane's orientation relative to the "world."

From Computer Graphics Using OpenGL, 2e, by F. S. Hill

FIGURE 5.52 The OpenGL pipeline (slightly simplified).

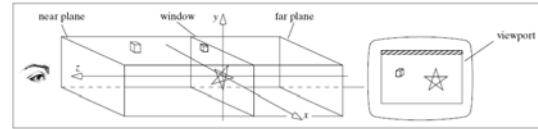
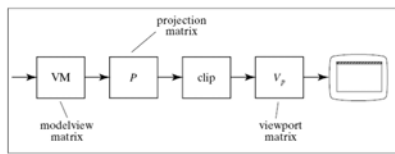


FIGURE 5.50 Simple viewing used in OpenGL for 2D drawing.

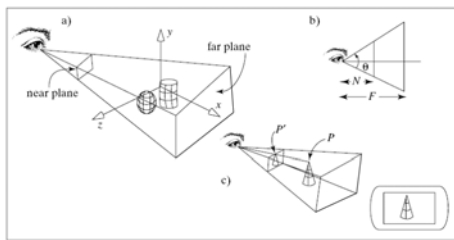


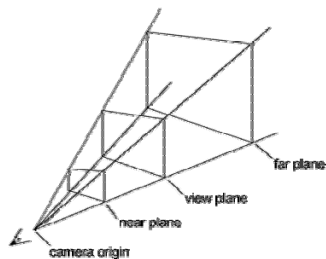
FIGURE 7.1 A camera to produce perspective views of a scene.

## Transformação de Projeção

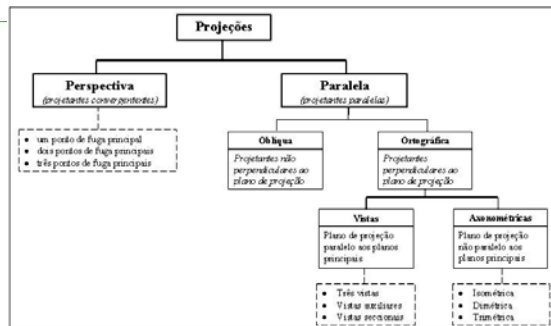
- Tendo a cena descrita no VCS, o próximo passo no *pipeline* consiste em projetar o conteúdo do volume de visualização no plano de imagem
  - Volume de visualização: 'viewing frustum': define a região de interesse na cena
  - Antes da projeção é aplicado um processo de 'recorte' (*clipping*), em que as partes dos objetos que estão fora do VF são descartadas
  - Recorte 3D – em relação aos planos de recorte (*clipping planes*)

## Viewing Frustum

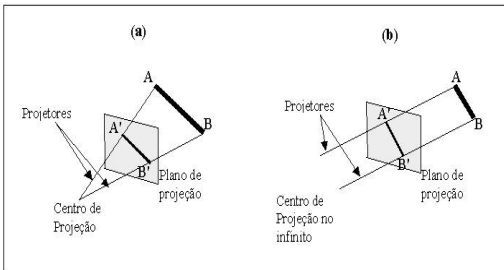
Volume de visualização, projeção perspectiva



## Taxonomia das projeções

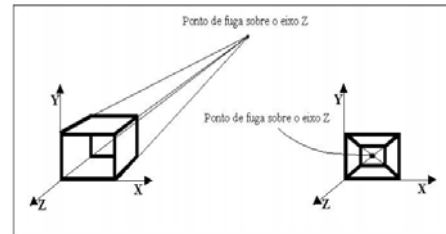


## Projeções paralela e perspectiva



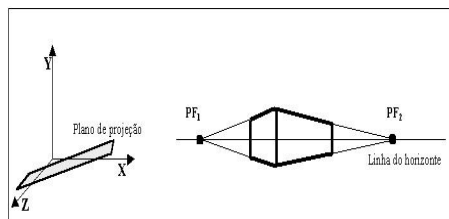
37

## Projeção perspectiva um ponto de fuga



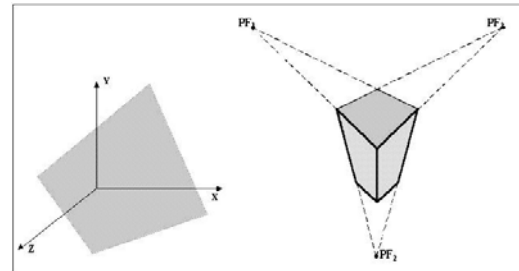
38

## Projeção perspectiva dois pontos de fuga



39

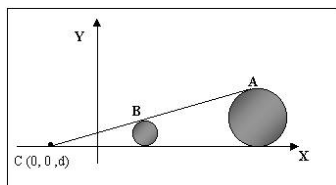
## Projeção perspectiva três pontos de fuga



40

## Características da Perspectiva

- Encurtamento perspectivo
  - Objetos ficam menores a medida que se distanciam do centro de projeção

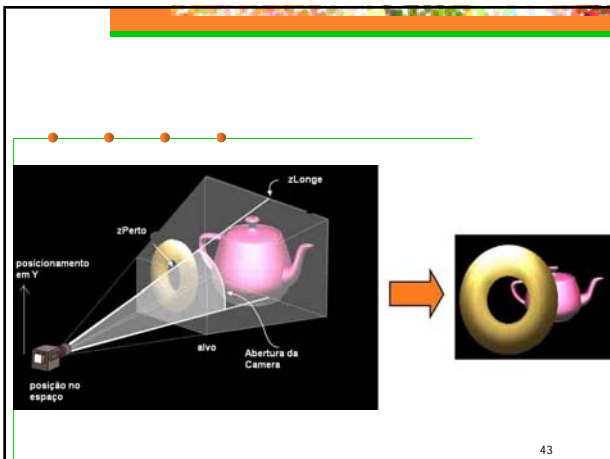


41

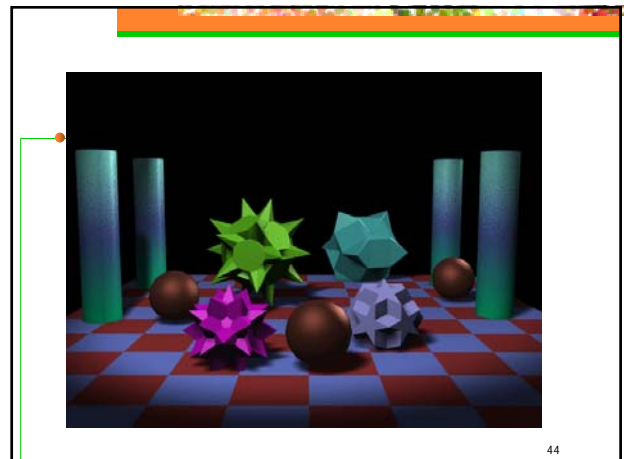
## Características da Perspectiva

- Pontos de Fuga
  - Retas não paralelas ao plano de projeção parecem se interceptar em um ponto no horizonte
- Confusão Visual
  - Objetos situados atrás do centro de projeção são projetados de cima para baixo e de trás para a frente
- Distorção Topológica
  - Pontos contidos no plano paralelo ao plano de projeção que contém o centro de projeção são projetados no infinito

42



43



44

## Projeção Paralela Ortográfica

- Transformação de projeção: como especificar uma projeção ortográfica em um plano paralelo ao plano xy do VCS?

45

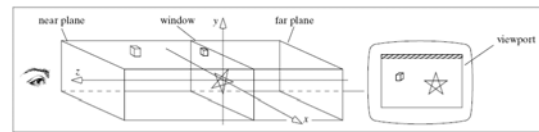


FIGURE 5.50 Simple viewing used in OpenGL for 2D drawing.

From Computer Graphics Using OpenGL, 2e, by F. S. Hill

## Projeções Paralelas

- No caso de projeções ortográficas, matrizes de transformação são triviais
- Ex. projeção no plano  $x_v y_v$  (VCS):

$$M_{ortograf} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

47

## Projeção Perspectiva

- PRP: *Projection Reference Point*
  - o ponto de referência de projeção... posicionado em  $(x_{prp}, y_{prp}, z_{prp})$  (no SRV)
  - alguns sistemas assumem que coincide com a posição da câmera (a origem do SRV)
- Problema da projeção:
  - determinar as coordenadas  $(x_p, y_p, z_p)$  de um ponto  $P = (x, y, z)$  projetado no plano de projeção.

48



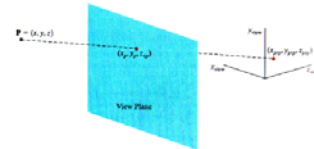
## Transformação de Projeção

- Problema da projeção:
  - determinar as coordenadas  $(x_p, y_p, z_p)$  de um ponto  $P = (x, y, z)$  projetado no plano de projeção
- Dados:
  - $(x_{prp}, y_{prp}, z_{prp})$  (no SRV)
  - O plano de projeção: perpendicular ao eixo  $z$  do SRV, posicionado em  $z = z_{vp}$

49

## Transformação de Projeção

- Algumas bibliotecas gráficas permitem que se escolha o ponto de referência de projeção  $(x_{prp}, y_{prp}, z_{prp})$



50

## Transformação de Projeção

- Considerando que a projeção do ponto  $(x, y, z)$  intercepta o plano de projeção na posição  $(x_p, y_p, z_{vp})$ , podemos escrever qualquer ponto ao longo dessa linha de projeção como

$$\begin{aligned} x_p &= x + u^*(x_{prp} - x) \\ y_p &= y + u^*(y_{prp} - y) \\ z_p &= z + u^*(z_{prp} - z) \\ u &\in [0, 1] \end{aligned}$$

51

## Transformação de Projeção

- No plano de projeção:  $z_p = z_{vp}$ . Podemos resolver  $z_p$  para obter o valor de  $u$  nessa posição...

$$u = \frac{z_{vp} - z}{z_{prp} - z}$$

52

## Transformação de Projeção

- Substituindo nas eqs. de  $x_p$  e  $y_p$  (e  $w_p$ )

$$\begin{aligned} x_p &= x \left( \frac{z_{prp} - z_{vp}}{z_{prp} - z} \right) + x_{prp} \left( \frac{z_{vp} - z}{z_{prp} - z} \right) \\ y_p &= y \left( \frac{z_{prp} - z_{vp}}{z_{prp} - z} \right) + y_{prp} \left( \frac{z_{vp} - z}{z_{prp} - z} \right) \end{aligned}$$

53

## Casos especiais

- Pode-se simplificar os cálculos restringindo alguns parâmetros... p. ex.:
- centro de projeção sobre eixo  $z$  do SRV:
  - $(x_{prp}, y_{prp}, z_{prp}) = (0, 0, z_{prp})$

$$\begin{aligned} x_p &= x \left( \frac{z_{prp} - z_{vp}}{z_{prp} - z} \right) \\ y_p &= y \left( \frac{z_{prp} - z_{vp}}{z_{prp} - z} \right) \end{aligned}$$

## Casos especiais

- Pode-se simplificar os cálculos restringindo alguns parâmetros... p. ex.:
- centro de projeção na origem do SRV:
  - $(x_{prp}, y_{prp}, z_{prp}) = (0, 0, 0)$

$$x_p = x \left( \frac{z_{vp}}{z} \right)$$

$$y_p = y \left( \frac{z_{vp}}{z} \right)$$

## Casos especiais

- Pode-se simplificar os cálculos restringindo alguns parâmetros... p. ex.:
- Plano de projeção é plano xy do SRV, e centro de projeção sobre eixo z do SRV:
  - $(x_{prp}, y_{prp}, z_{prp}) = (0, 0, z_{prp})$ , e  $z_{vp} = 0$

$$x_p = x \left( \frac{z_{prp}}{z_{prp} - z} \right)$$

$$y_p = y \left( \frac{z_{prp}}{z_{prp} - z} \right)$$

## Casos especiais

- Em geral, plano de projeção está entre o centro de projeção e a cena
- ... outras posições são possíveis (o centro de projeção não pode estar no plano de projeção)

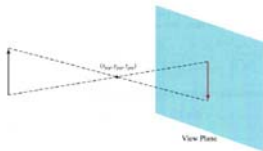


Figura: Os objetos são invertido se o ponto de referência está entre o plano de visão e a cena.

## Casos especiais

- Efeitos da perspectiva tb. dependem da distância entre o centro de projeção e o plano de projeção

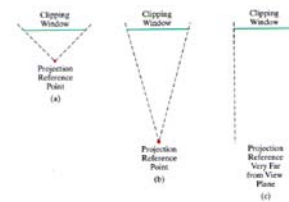


Figura: Se o centro de projeção está próximo ao plano de projeção, objetos mais próximos ao plano aparecerão muito maiores do que os distantes

## Matriz de transformação perspectiva

- Não é possível, a partir das eqs. anteriores, definir uma matriz de transformação perspectiva
  - denominadores dos coeficientes de  $x$  e  $y$  são definidos em função de  $z$

$$x_p = x \left( \frac{z_{prp} - z_{vp}}{z_{prp} - z} \right) + x_{prp} \left( \frac{z_{vp} - z}{z_{prp} - z} \right)$$

$$y_p = y \left( \frac{z_{prp} - z_{vp}}{z_{prp} - z} \right) + y_{prp} \left( \frac{z_{vp} - z}{z_{prp} - z} \right)$$

59

## Matriz de transformação perspectiva

- Pode-se superar essa limitação usando coordenadas homogêneas

$$x_p = \frac{x_h}{h}, \quad y_p = \frac{y_h}{h}$$

- Em que o parâmetro homogêneo é

$$h = z_{prp} - z$$

60

## Matriz de transformação perspectiva

- Os numeradores permanecem os mesmos

$$x_p = x \left( \frac{z_{prp} - z_{vp}}{h} \right) + x_{prp} \left( \frac{z_{vp} - z}{h} \right)$$

$$y_p = y \left( \frac{z_{prp} - z_{vp}}{h} \right) + y_{prp} \left( \frac{z_{vp} - z}{h} \right)$$

61

## Matriz de transformação perspectiva

- Pode-se definir uma matriz de transformação que projeta uma posição espacial dada em coordenadas homogêneas
- Inicialmente calcula-se as coord's homogêneas  $P_h = (x_h, y_h, z_h, h)$  de um ponto  $P = (x, y, z, 1)$  usando a matriz de projeção perspectiva
 
$$P_h = M_{pers} \cdot P$$
- A seguir as coord's são divididas por  $h$  para se obter as coord's projetadas

## Transformação de Projeção

- Na forma matricial homogênea

$$M_{Pers} = \begin{bmatrix} z_{prp} - z_{vp} & 0 & -x_{prp} & x_{prp} z_{prp} \\ 0 & z_{prp} - z_{vp} & -y_{prp} & y_{prp} z_{prp} \\ 0 & 0 & -z_{vp} & z_{vp} z_{prp} \\ 0 & 0 & -1 & z_{prp} \end{bmatrix}$$

63

## Transformação de Projeção

- Observação
  - Valor original da coordenada  $z$  (no VCS) deve ser mantido para uso posterior por algoritmos de remoção de superfícies ocultas

64

## Paralela vs. Perspectiva

- Projeção perspectiva
  - Tamanho varia inversamente com distância: aparência 'realista'
  - Distâncias e ângulos não são preservados
  - Linhas paralelas não são preservadas
- Projeção paralela
  - Boa para medidas exatas
  - Linhas paralelas são preservadas
  - Ângulos não são preservados
  - Aparência menos realista

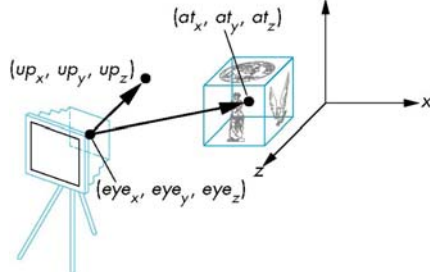
65

## No OPENGL

- `glOrtho(left, right, bottom, top, near, far);`
- `glFrustum(left, right, bottom, top, near, far);`
- `gluPerspective(angle, aspect, near, far);`
- `gluLookAt(eyex, eyey, eyez, atx, aty, atz, upx, upy, upz);`

## gluLookAt

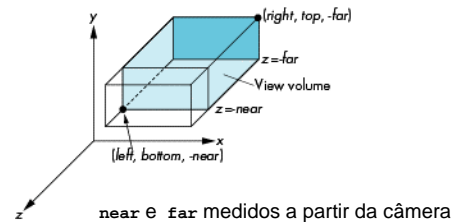
`gluLookAt(eyex, eyey, eyez, atx, aty, atz, upx, upy, upz)`



Fonte: E. Angel, Interactive Computer Graphics, 4a. Ed., Addison-Wesley 2005

## OpenGL Orthogonal Viewing

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

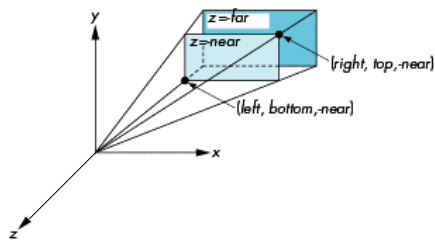


near e far medidos a partir da câmara

Fonte: E. Angel, Interactive Computer Graphics, 4a. Ed., Addison-Wesley 2005

## OpenGL Perspective

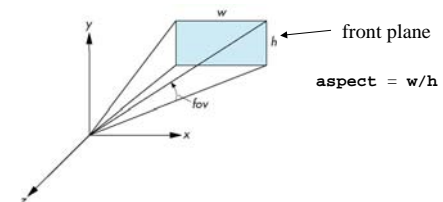
`glFrustum(left, right, bottom, top, near, far)`



Fonte: E. Angel, Interactive Computer Graphics, 4a. Ed., Addison-Wesley 2005

## Using Field of View

`gluPerspective(fovy, aspect, near, far)`



Fonte: E. Angel, Interactive Computer Graphics, 4a. Ed., Addison-Wesley 2005

## Bibliografia

- Capítulo 6 da apostila
- Cap. 7 Hearn & Baker
- Cap. 2 Conci e Azevedo
- <http://escience.anu.edu.au/lecture/cg/Transformation/index.en.html>
- Curso CG da ACM (link na pág. GBDI)
- ...