Viewing 3D

SCC0250 - Computação Gráfica

Profa. Maria Cristina F. Oliveira

Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de São Paulo (USP)

19 de setembro de 2023

Sumário

Sumário

Visão Geral

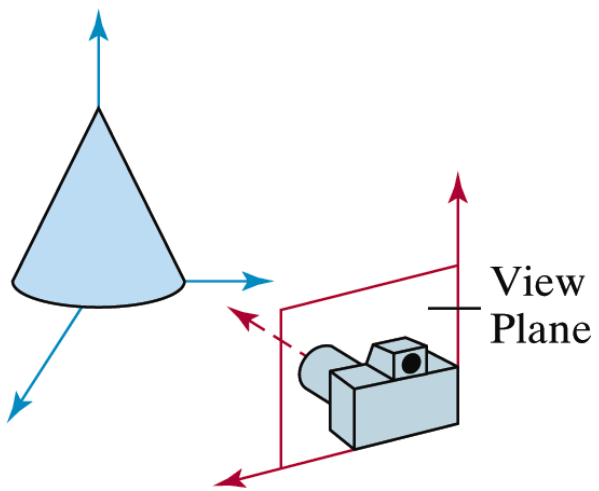
Viewing Pipeline 3D

As funções de *viewing* processam a descrição geométrica da cena por meio de um conjunto de transformações a fim de gerar uma visão específica da cena a ser exibida na superfície do dispositivo de saída. Isso envolve várias etapas:

- Definição dos parâmetros da câmera
- Definição da *View matrix* e transformação WCS->VCS
- Transformação de projeção
- Identificação das partes visíveis da cena
- Cálculo dos efeitos de iluminação

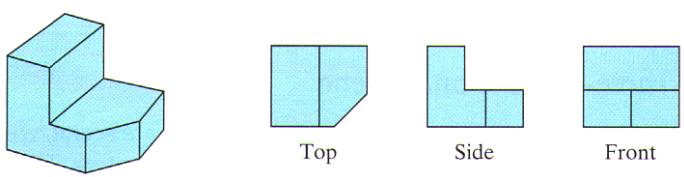
Visualizando uma Cena 3D

- Para obter uma imagem a partir de uma cena 3D dada no sistema de *coordenadas do mundo* (WCS) é necessário, primeiramente:
  - Definir um sistema de referência em função dos parâmetros de observação (câmera, ou observador) (VCS)
  - Definir a posição e orientação do **plano de projeção** – análogo ao plano do filme de uma câmera fotográfica

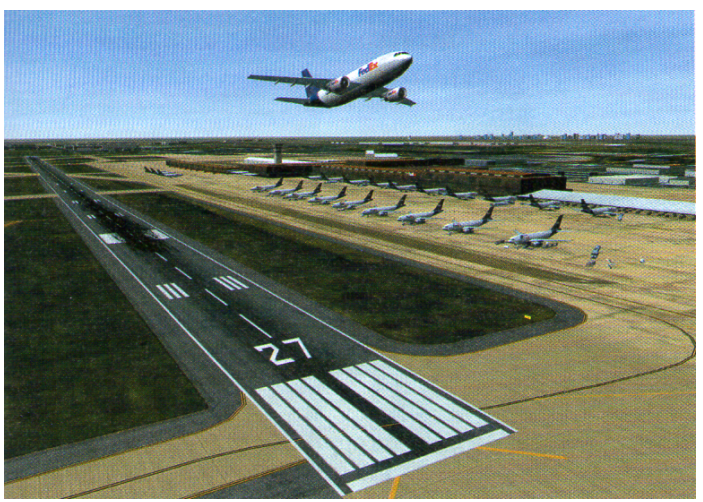


Projeções

- Há duas maneiras de projetar uma cena em um plano de projeção
  - **Projeção Paralela:** os pontos dos objetos são projetados ao longo de linhas paralelas – muito usado em desenhos arquitetônicos e de engenharia (preserva comprimentos e ângulos)
  - **Projeção Perspectiva:** os pontos dos objetos são projetados ao longo de linhas convergentes – cenas mais realísticas (objetos distantes da posição de observação parecem menores)



(a) Projeção Paralela



(b) Projeção Perspectiva

Depth Cueing (Noção de Profundidade)

- Informação de profundidade é, em geral, muito importante na composição de uma cena 3D: identifica quais partes dos objetos são visíveis para o observador e quais não são (faces da frente e de trás do objeto em relação ao observador)

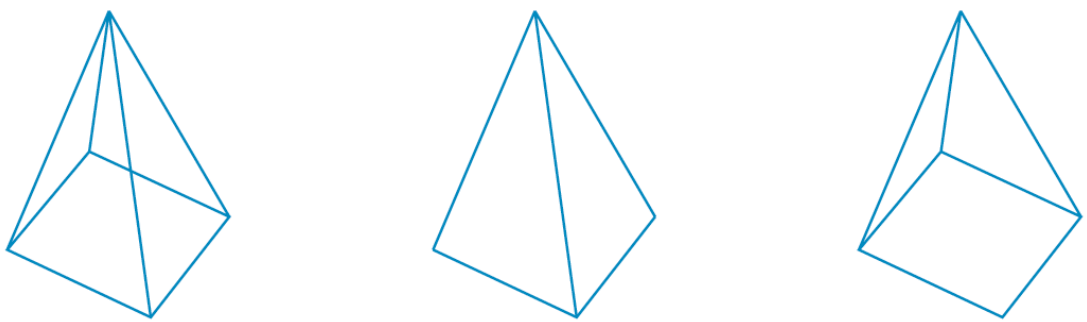


Figura: (a) Interpretação ambígua devido à ausência de informação de profundidade: objeto pode ser interpretado como em (b) ou como em (c)

Identificando Linhas e Superfícies Visíveis

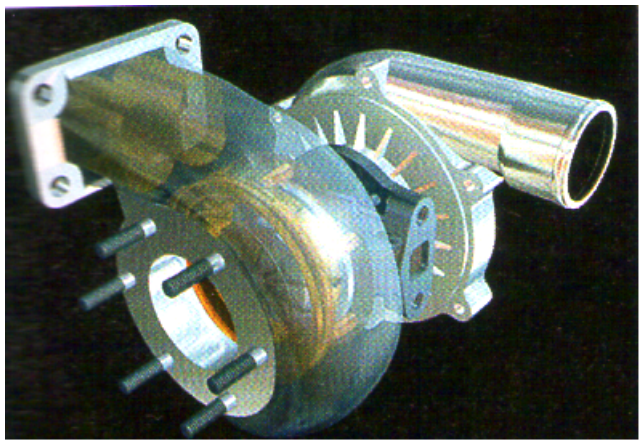
Cenas Wireframe ('desenho aramado´)

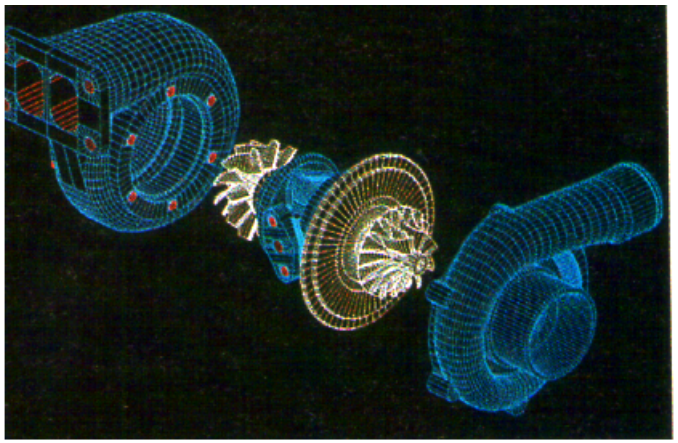
- Existem maneiras de tornar renderizações *wireframe* mais realísticas
  - Colorir as linhas visíveis de maneira diferente das não-visíveis
  - Mostrar as linhas não visíveis como pontilhadas
  - Remover as linhas não visíveis – também remove informação sobre a forma do objeto

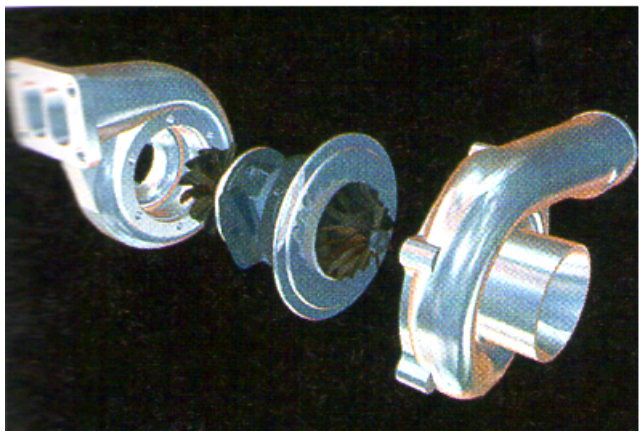
Cenas Realísticas

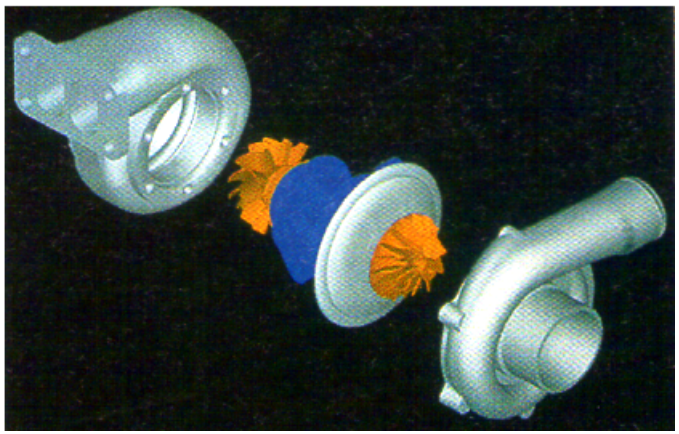
- Para cenas realísticas, somente as superfícies visíveis de um objeto são renderizadas, as partes não visíveis são completamente eliminadas
  - Os pixels da tela terão informação tão somente das cores das superfícies voltadas para o observador

Identificando Linhas e Superfícies Visíveis









Rendering de Superfície

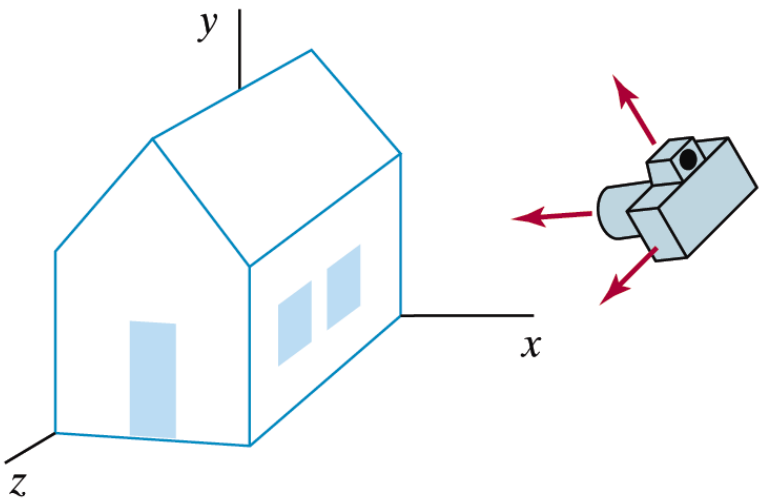
- Efeitos realísticos são introduzidos por meio de um modelo de iluminação
- Define-se a luz ambiente, a localização e a cor das diferentes fontes de luz
- Especifica-se também as características dos materiais: transparente, opaco, rugoso, etc.

Rendering de Superfície

Sumário

Viewing Pipeline 3D

- O processo de sintetizar uma imagem de uma cena 3D é análogo ao de tirar uma foto
  - Escolhe-se uma posição de observação, aonde será posicionada a câmera
  - Define-se a orientação da câmera em relação à cena
    - Para onde ela está 'olhando' (direção de observação)
    - Como está orientada em relação à cena: direção *up*

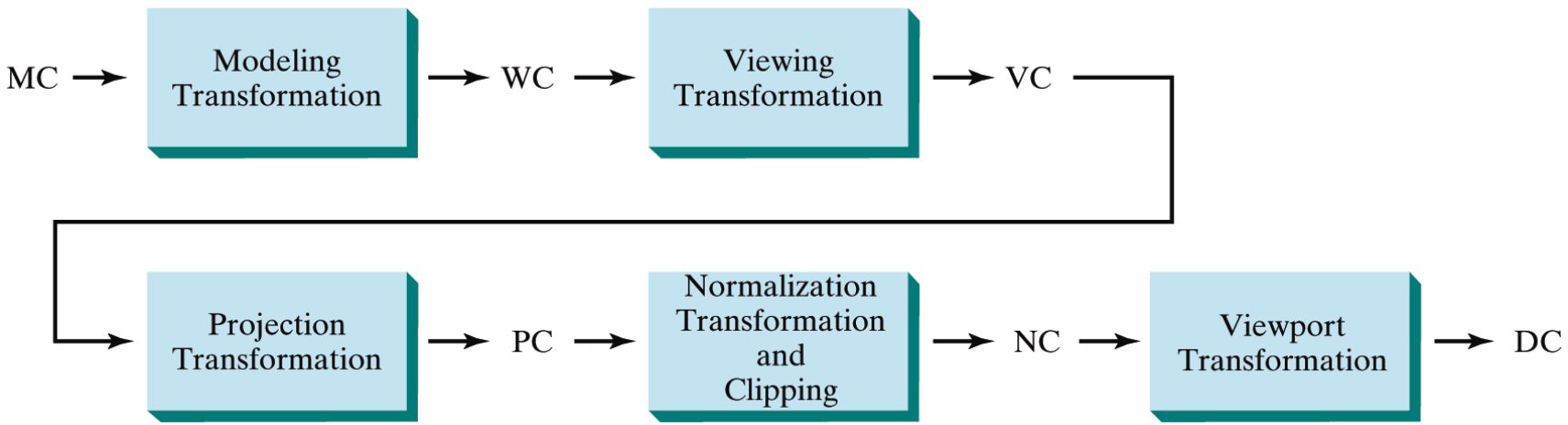


Viewing Pipeline 3D

- Etapas do *Viewing Pipeline 3D*
  - Posicionar a câmera
  - Definir um plano de projeção no qual a cena será projetada
  - Definir uma **janela de recorte 2D** neste plano, para selecionar a região da cena a ser renderizada: **viewport 2D**
  - É preciso mapear o conteúdo da **viewport 2D** para o dispositivo de exibição (em coordenadas da tela)
  - A cena será recortada também em relação a um volume no espaço (**volume de recorte**) a partir da definição de planos de recorte no espaço 3D

Viewing Pipeline 3D

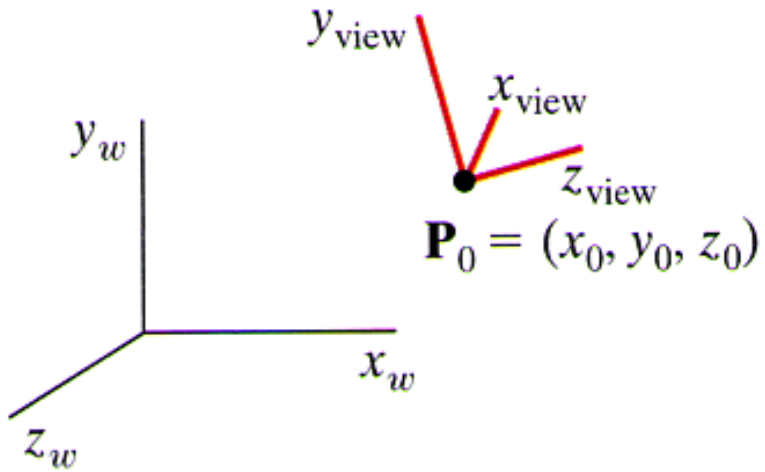
- A **posição de observação**, o **plano de projeção**, a **janela de recorte** e os **planos de recorte** são especificados em relação ao sistema de **coordenadas de mundo** (WCS)



Sumário

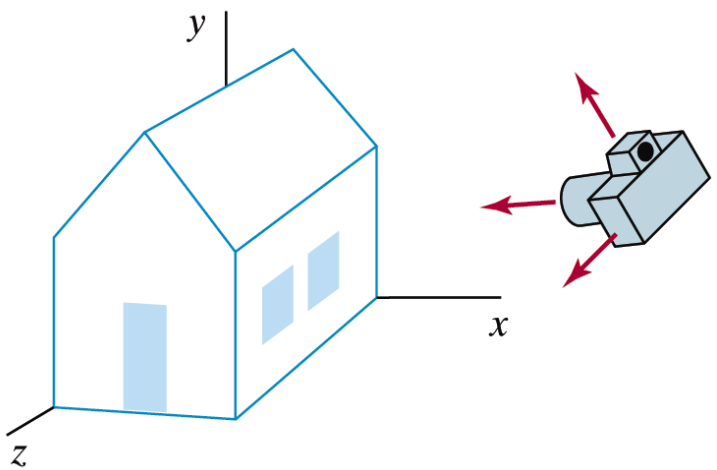
Definindo o VCS e o plano de projeção

- Para estabelecer o VCS (*viewing coordinate system*) é preciso ter a origem e os 3 eixos que o definem (dados no WCS)
- Vamos usar 3 parâmetros: posição da câmera, vetor direção de observação e vetor de orientação da câmera (*view-up*)
- A partir deles definimos a origem do VCS e dois dos seus eixos (z_{view} e y_{view})



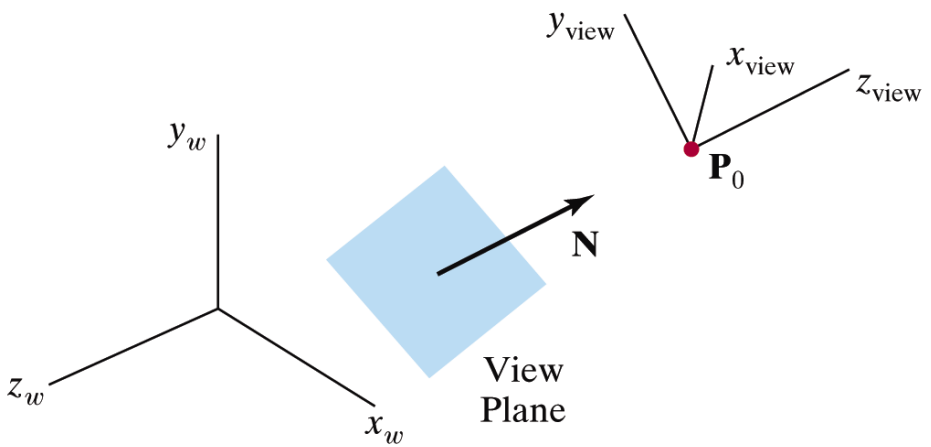
Definindo o VCS e o plano de projeção

- A **posição da câmera** $\mathbf{P}_0 = (x_0, y_0, z_0)$ (posição do observador) define a origem do VCS
- O **vetor view-up** $\mathbf{V}$ (orientação da câmera) define a direção do eixo y_{view}
- O vetor direção de observação define o eixo z_{view} (note que y_{view} e z_{view} precisam ser perpendiculares entre si)
- O terceiro eixo x_{view} é obtido a partir do produto vetorial entre esses dois



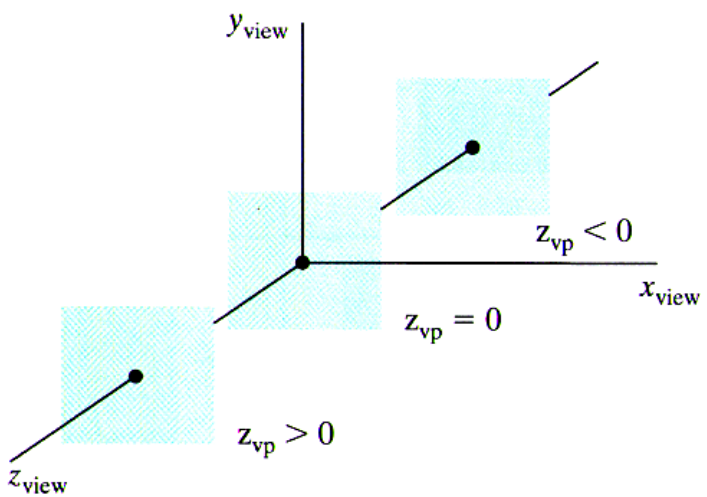
Vetor Direção de Observação e Normal ao Plano de Projeção

- Dado que o eixo z_{view} define a direção de observação, adota-se um plano de projeção perpendicular a esse eixo
- Assim, a orientação do plano de projeção e a direção positiva do eixo z_{view} podem ser definidas em termos de um **vetor N normal ao plano de projeção**



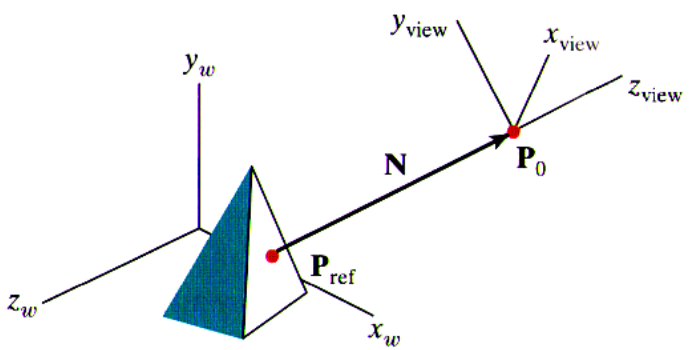
Vetor Direção de Observação e Normal ao Plano de Projeção

- Um valor escalar z_{vp} define a posição do plano de projeção ao longo do eixo z_{view}
  - Especifica a distância do plano de projeção à posição da câmera, ao longo da direção de observação, normalmente na direção negativa de z_{view}
  - Portanto, o plano de projeção é sempre paralelo ao plano $x_{view}y_{view}$



Vetor Direção de Observação e Normal ao Plano de Projeção

- O vetor normal $\mathbf{N}$ (assim como a direção de observação) pode ser especificado de diferentes maneiras, em função de um ponto de referência na cena P_{ref} dado
  - A direção de $\mathbf{N}$ pode ser definida ao longo da linha que vai da origem do VCS (P_0) ao ponto de referência (P_{ref}) (vetor $P_{ref} - P_0$)
  - Alternativamente, $\mathbf{N}$ pode ser dado como o vetor do ponto $\mathbf{P}_{ref}$ à origem do VCS $\mathbf{P}_0$ ($\mathbf{P}_0 - P_{ref}$) (como na figura)
  - Nesse caso, P_{ref} é denominado **look-at point**, e a direção de observação é oposta à orientação de $\mathbf{N}$



Vetor View-Up

- Uma vez estabelecida a direção da normal ao plano de projeção $\mathbf{N}$, tem-se a direção e orientação do eixo y_{view}
- Em seguida, podemos estabelecer o **vetor view-up** $\mathbf{V}$, que dá a direção do eixo y_{view}
- Usualmente $\mathbf{V}$ é definido selecionando uma posição relativa à origem do sistema de coordenadas do mundo (WCS): vetor vai da origem do WCS para essa posição

Vetor View-Up

- $\mathbf{V}$ pode ser definido em qualquer direção não paralela a $\mathbf{N}$
  - Uma forma conveniente seria definir uma direção paralela ao eixo y_w, $\mathbf{V} = (0, 1, 0)$
  - Se esse vetor não for exatamente perpendicular a $\mathbf{N}$, ele pode ser ajustado projetando-o sobre um plano perpendicular a $\mathbf{N}$

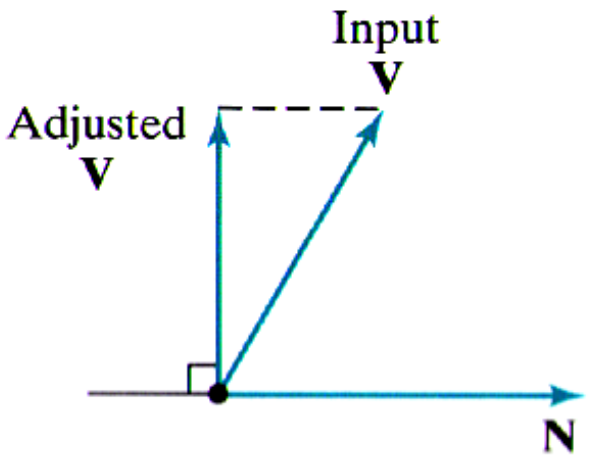


Figura: Ajuste do vetor *view-up* $\mathbf{V}$ para torná-lo perpendicular a $\mathbf{N}$

Sistema de Coordenadas da câmera (VCS): **uvn**

- Como a normal $\mathbf{N}$ define a orientação do eixo z_{view}, e o vetor *view-up* $\mathbf{V}$ define y_{view}, só falta definir a direção positiva do eixo x_{view}
  - Essa direção é dada pelo vetor $\mathbf{U}$, calculado como o produto vetorial dos vetores $\mathbf{N}$ e $\mathbf{V}$
  - Também pode-se usar o produto vetorial de $\mathbf{N}$ e $\mathbf{U}$ para ajustar o valor de $\mathbf{V}$ ao longo do eixo y_{view}

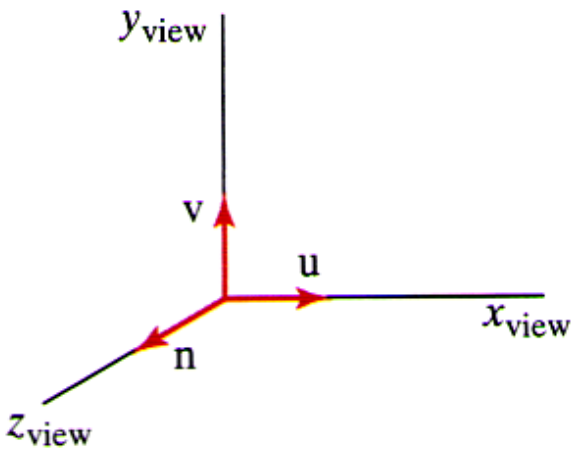
Sistema de Coordenadas da câmera (VCS): **uvn**

- Assim, para obter o sistema $\mathbf{uvn}$ fazemos

$$\mathbf{n} = \frac{\mathbf{N}}{|\mathbf{N}|} = (n_x, n_y, n_z)$$

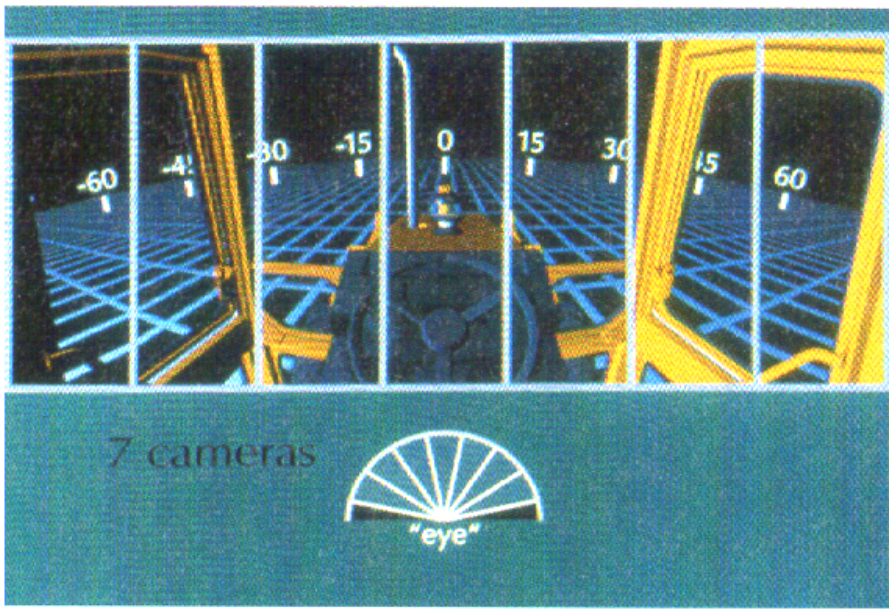
$$\mathbf{u} = \frac{\mathbf{V} \times \mathbf{n}}{|\mathbf{V} \times \mathbf{n}|} = (u_x, u_y, u_z)$$

$$\mathbf{v} = \mathbf{n} \times \mathbf{u} = (v_x, v_y, v_z)$$



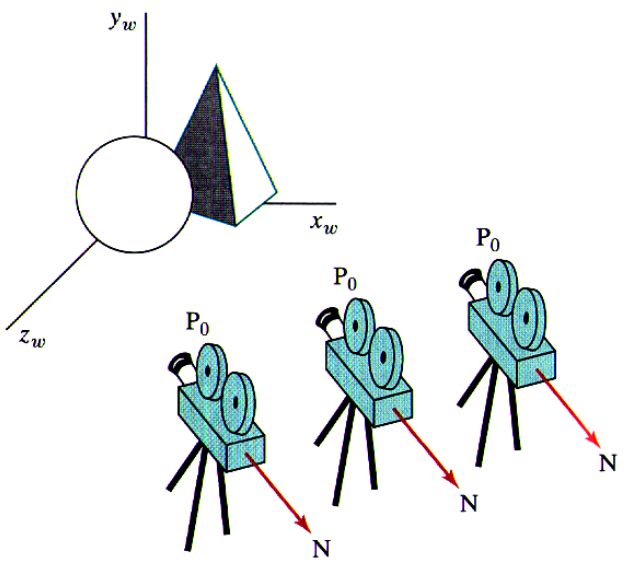
Manipulando os parâmetros de câmera

- É possível variar os parâmetros da câmera para obter diferentes efeitos visuais
  - Fixar P_0 (posição de observação) e variar $\mathbf{N}$ para apresentar múltiplas visões de uma cena a partir de uma posição fixa da câmera
  - Ao alterar $\mathbf{N}$ é preciso ajustar os demais eixos para manter o sistema orientado pela mão direita

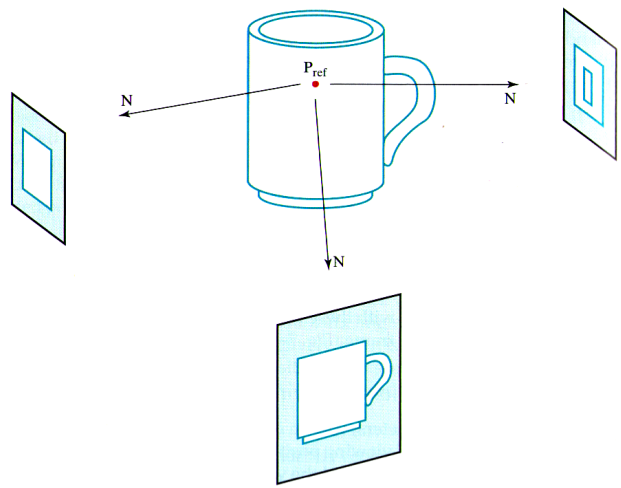


Manipulando os parâmetros da câmera

- Pode-se obter efeitos de câmera panorâmica (*pan*) fixando $\mathbf{N}$ e modificando o ponto de observação
- Também pode-se mover o ponto de observação ao redor do objeto para mostrar diferentes vistas (visão lateral, frontal, superior, etc.)



(a) Efeito de pan



(b) Visões diferentes

Sumário

Transformação do Sistema de Coordenadas de Mundo (WCS) para o Sistema de Coordenadas da Cãmera (VCS)

- Uma vez definido o VCS, o próximo passo no *Viewing Pipeline 3D* consiste em mapear a descrição da cena no sistema de coordenadas do mundo (WCS) para o sistema de coordenadas da câmera (VCS)
  - Trata-se de uma transformação entre sistemas de coordenadas
  - Transformação geométrica que alinha o VCS com o WCS
  - A matriz de transformação é denominada *view matrix*

- A transformação é descrita pela matriz que compõe:
  1. Translação para alinhar a origem do VCS com a origem do WCS
  2. Rotação para alinhar os eixos x_{view}, y_{view} e z_{view} com os eixos do WCS x_w, y_w e z_w

Transformação do Sistema de Coordenadas de Mundo (WCS) para o Sistema de Coordenadas da Cãmera (VCS)

- Se a origem do sistema de visão é $\mathbf{P}_0 = (x_0, y_0, z_0)$, a matriz de translação é dada por

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

Transformação do Sistema de Coordenadas de Mundo (WCS) para o Sistema de Coordenadas da Cãmera (VCS)

- E a matriz de rotação pode ser obtida diretamente a partir dos vetores $\mathbf{u} = (u_x, u_y, u_z)$, $\mathbf{v} = v_x, v_y, v_z$ e $\mathbf{n} = (n_x, n_y, n_z)$

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

Transformação do Sistema de Coordenadas de Mundo (WCS) para o Sistema de Coordenadas da Cãmera (VCS)

- A matriz de transformação (*view matrix*) é, portanto, dada por

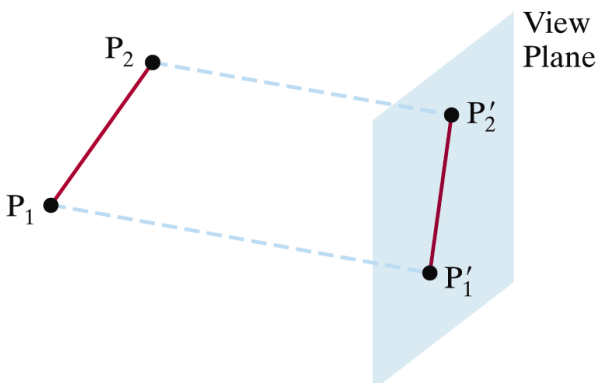
$$\begin{aligned}\mathbf{M}_{WC,VC} &= \mathbf{R}\cdot\mathbf{T} \\ &= \begin{bmatrix} u_x & u_y & u_z & -\mathbf{u}\cdot\mathbf{P}_0 \\ v_x & v_y & v_z & -\mathbf{v}\cdot\mathbf{P}_0 \\ n_x & n_y & n_z & -\mathbf{n}\cdot\mathbf{P}_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\end{aligned}$$

Sumário

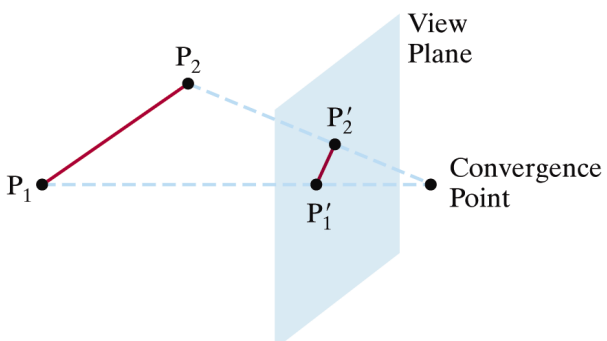
Transformações de Projeção

- Após a transformação para o VCS, a próxima etapa do *Viewing Pipeline 3D* é a projeção da cena no plano de projeção

- Em geral, os pacotes gráficos admitem
  - **Projeção Paralela:** as coordenadas são transferidas para o plano de projeção ao longo de linhas paralelas
  - **Projeção Perspectiva:** as coordenadas são transferidas para o plano de projeção ao longo de linhas que convergem para um ponto



(c) Projeção Paralela



(d) Projeção Perspectiva

Sumário

Sistema de coordenadas da câmera no OpenGL

- Os parâmetros da câmera permitem definir uma matriz que é concatenada com a matriz de transformações de modelagem *MODELVIEW* corrente
- Inicia-se especificando a *matriz modelview*

```
1 gl.glMatrixMode(GL.GL_MODELVIEW);
```

Sistema de coordenadas da câmera no OpenGL

- Para especificar os parâmetros da câmera utiliza-se a função

```
1 glu.gluLookAt(GLdouble x0, GLdouble y0, GLdouble z0,
2          GLdouble xref, GLdouble yref, GLdouble zref,
3          GLdouble Vx, GLdouble Vy, GLdouble Vz);
```

- Essa função define
  - A origem do sistema de visão $\mathbf{P}_0 = (x0, y0, z0)$ no sistema de coordenadas de mundo (a localização da câmera)
  - A posição de referência $\mathbf{P}_{ref} = (xref, yref, zref)$ (para onde a câmera aponta, também chamado *look-at*)
  - O vetor *view-up* $\mathbf{V} = (Vx, Vy, Vz)$

Sistema de coordenadas da câmera no OpenGL

- A direção positiva do eixo z_{view} do VCS está na direção $\mathbf{N} = \mathbf{P}_0 - \mathbf{P}_{ref}$
  - A direção de observação está na direção negativa do eixo z_{view}

- Os parâmetros especificados usando a função *gluLookAt(...)* são usados para compor a matriz $\mathbf{M}_{WC,VC}$
  - Matriz que alinha o sistema da câmera (VCS) com o sistema de coordenadas de mundo (WCS)

- Os parâmetros default da função *gluLookAt(...)* são
  - $\mathbf{P}_0 = (0, 0, 0)$
  - $\mathbf{P}_{ref} = (0, 0, -1)$
  - $\mathbf{V} = (0, 1, 0)$

Sistema de coordenadas da câmera no OpenGL

```
1  public void display(GLAutoDrawable drawable) {
2    GL gl = drawable.getGL();
3    GLUT glut = new GLUT();
4    GLU glu = new GLU();
5
6    //limpa o buffer
7    gl.glClear(GL.GL_COLOR_BUFFER_BIT | GL.GL_DEPTH_BUFFER_BIT);
8
9    //define que a matrix  a de modelo
10   gl.glMatrixMode(GL.GL_MODELVIEW);
11   gl.glLoadIdentity(); //carrega a matrix de identidade
12   glu.gluLookAt(1.0, 0.5, 0.5, //posio da cmera
13                 0.0, 0.0, 0.0, //para onde a cmera aponta
14                 0.0, 1.0, 0.0); //vetor view-up
15
16   //desenha um cubo
17   gl.glColor3f(1.0f, 0.0f, 0.0f);
18   glut.glutWireCube(1.0f);
19
20   //fora o desenho das primitivas
21   gl.glFlush();
22 }
```

Sistema de coordenadas da câmera no OpenGL