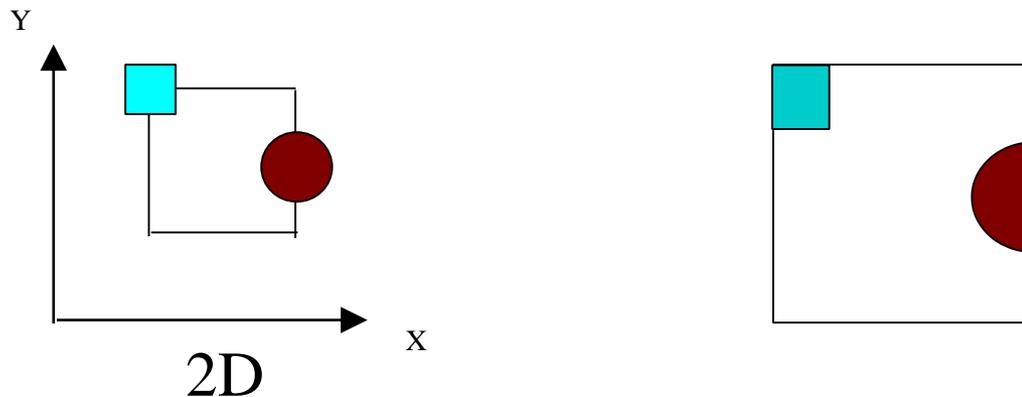

Visualização 3D

Soraia Raupp Musse

Pipeline de Visualização

- Em 2D as coisas são mais simples
 - Simplesmente especificar uma janela do mundo 2D e uma viewport na superfície de visualização
- A complexidade começa em 3D, pelo fato de termos uma dimensão a mais, mas também pelo fato do dispositivo de exibição ser 2D



Pipeline 2D

- SRO
 - SRU
 - SRW
- (recorte 2D)
- SRV
 - SRD

Em 3D? Câmera Virtual

<https://www.khanacademy.org/partner-content/pixar/virtual-cameras/virtual-cameras-1/v/virtual-cameras>



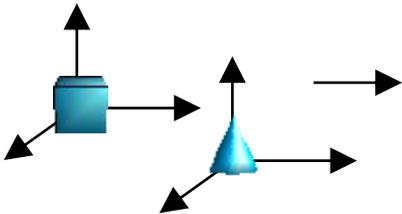
Em 3D? Câmera Virtual

- A metáfora que se utiliza é a de uma câmera virtual em nosso universo
- Podemos mover a câmera para qualquer posição e orientação no espaço
- A câmera pode tirar fotos simples ou tornar-se uma filmadora, que mostra nosso universo de diferentes pontos de vista
- Nossa câmera é um programa de computador que produz, a partir de uma descrição dos objetos, a imagem na tela do computador
- Utiliza um sistema de coordenadas próprio



Pipeline de Visualização 3D

- Para tirarmos uma foto virtual, precisamos executar os seguintes passos



1 - Objetos em um sistema de coord. Próprio (SRO - 3D)

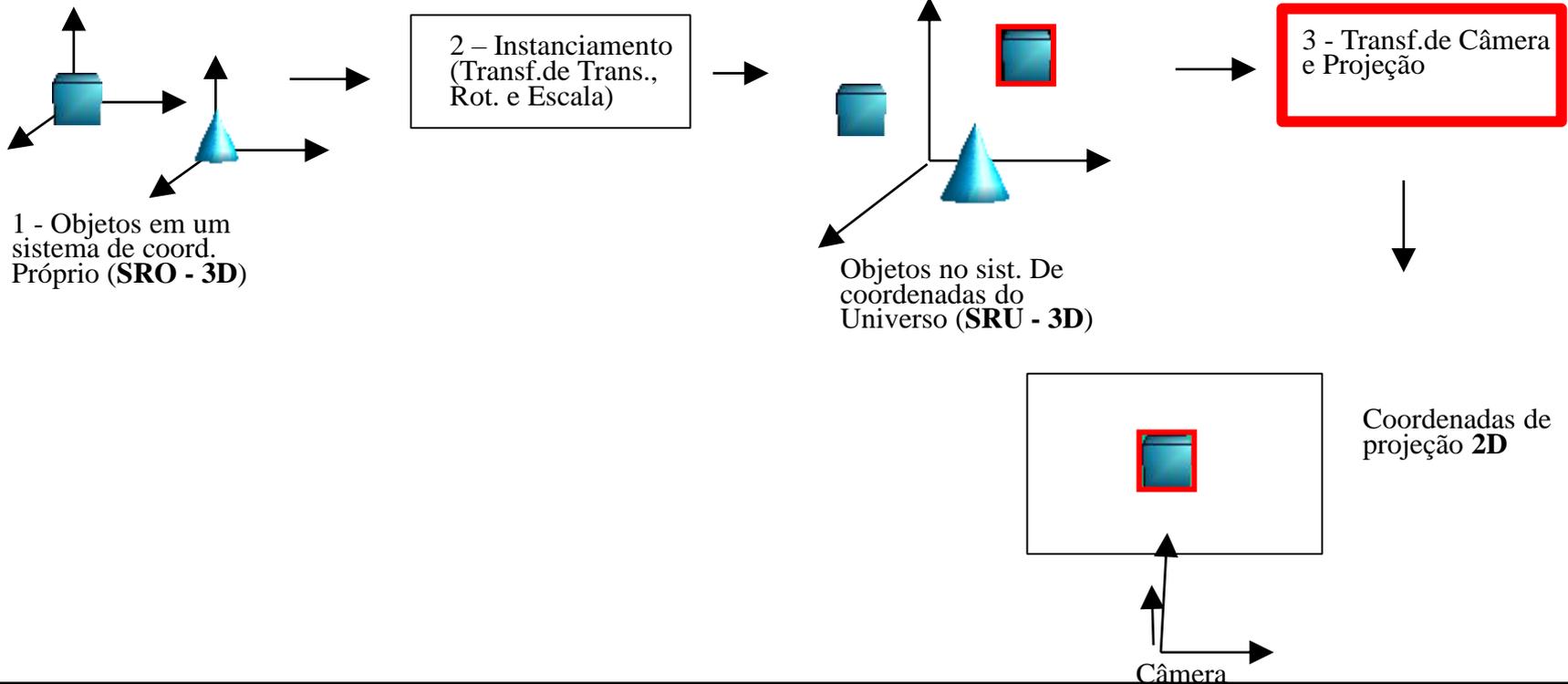
Pipeline de Visualização 3D

- Para tirarmos uma foto virtual, precisamos executar os seguintes passos



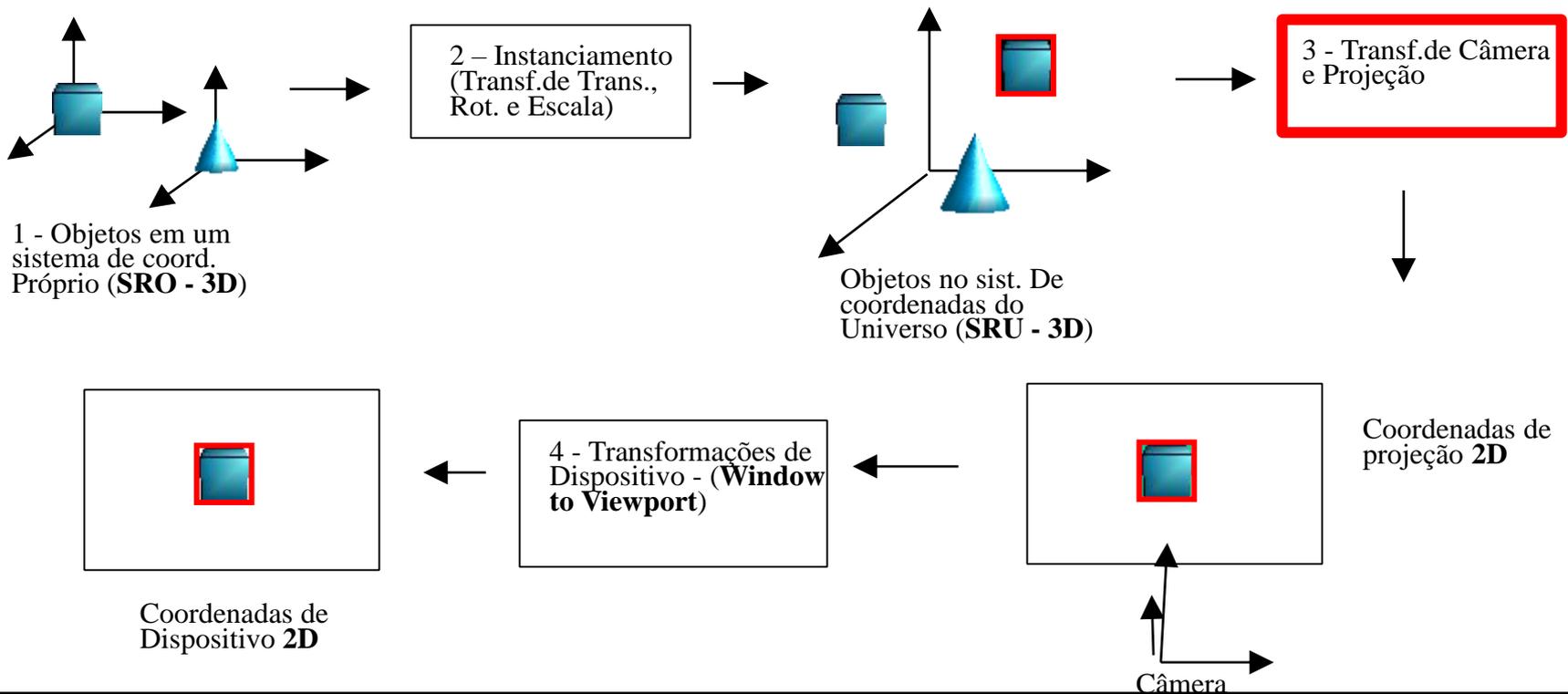
Pipeline de Visualização 3D

- Para tirarmos uma foto virtual, precisamos executar os seguintes passos

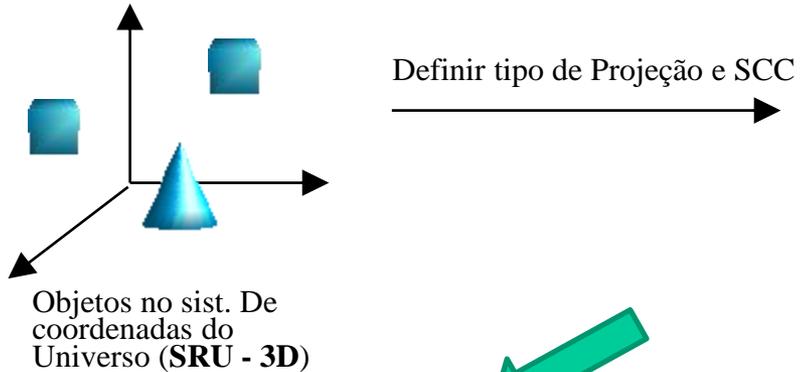


Pipeline de Visualização 3D

- Para tirarmos uma foto virtual, precisamos executar os seguintes passos

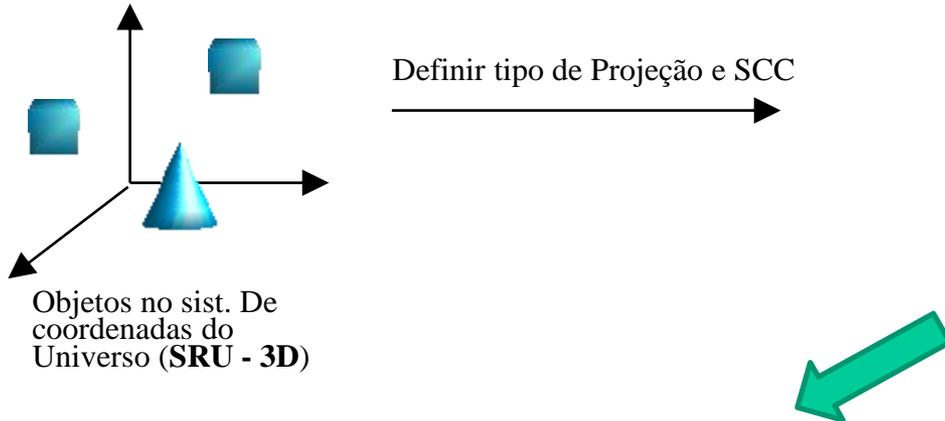


Câmera e projeção



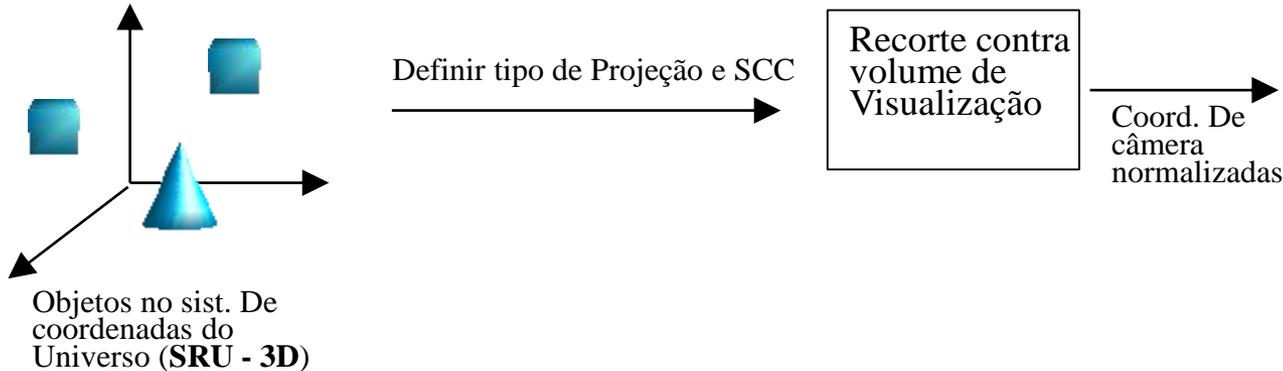
- Definir o tipo de projeção e sistema de coordenadas de câmera (SCC) implica em como será o volume de visualização.

Pipeline de Visualização – 3D



- Definir o tipo de projeção e sistema de coordenadas de câmera (SCC) implica em como será o volume de visualização.

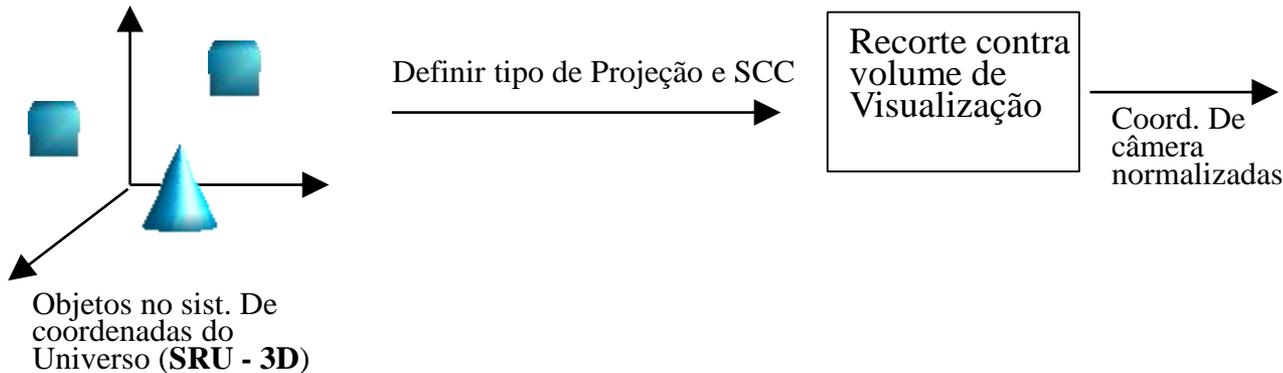
Pipeline de Visualização – 3D



- Definir o tipo de projeção e sistema de coordenadas de câmera (SCC) implica em como será o volume de visualização. Objetos estarão em coordenadas de câmera.
- Recorte contra o volume de visualização 3D. Coordenadas de câmera 3D normalizadas e recortadas.



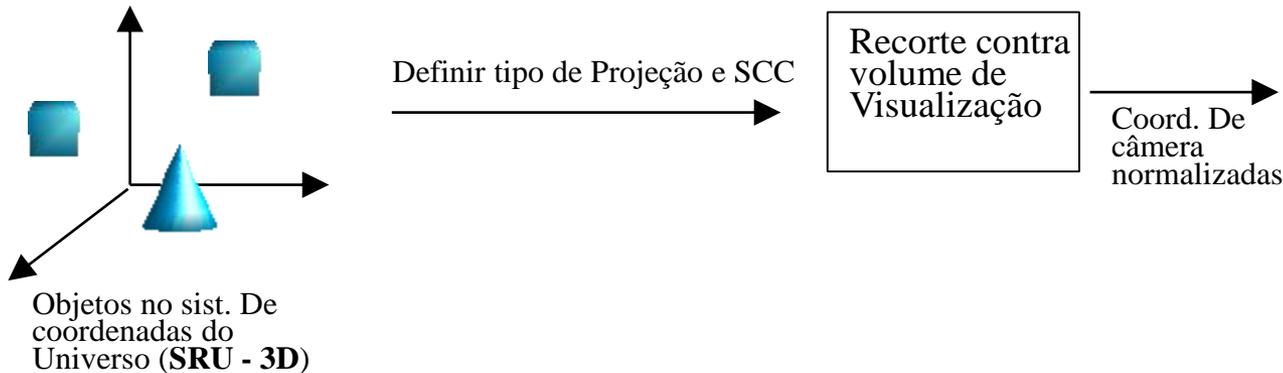
Pipeline de Visualização – 3D



- Definir o tipo de projeção e sistema de coordenadas de câmera (SCC) implica em como será o volume de visualização. Objetos estarão em coordenadas de câmera.
- Recorte contra o volume de visualização 3D. Coordenadas de câmera 3D normalizadas e recortadas.

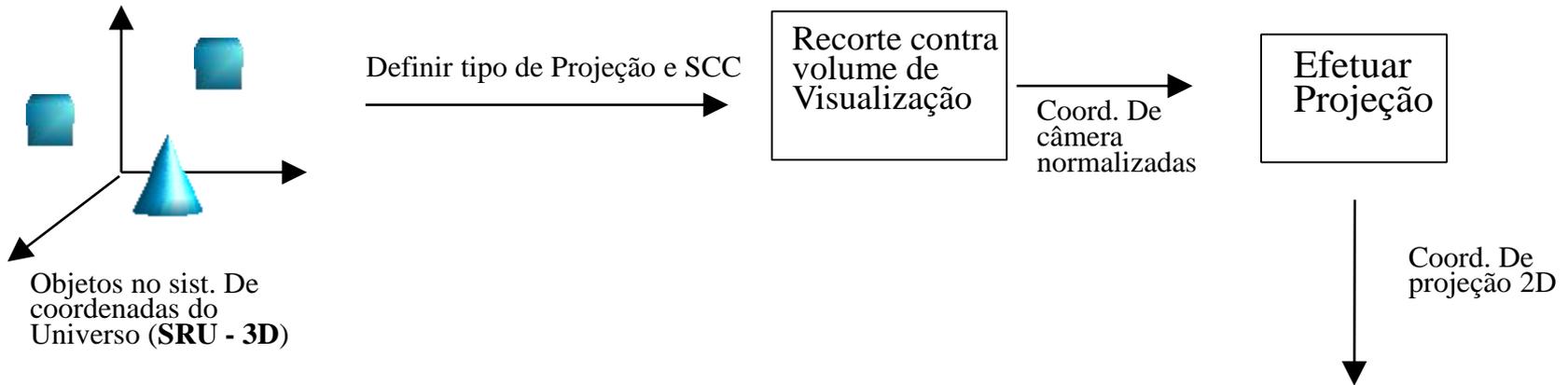


Pipeline de Visualização – 3D



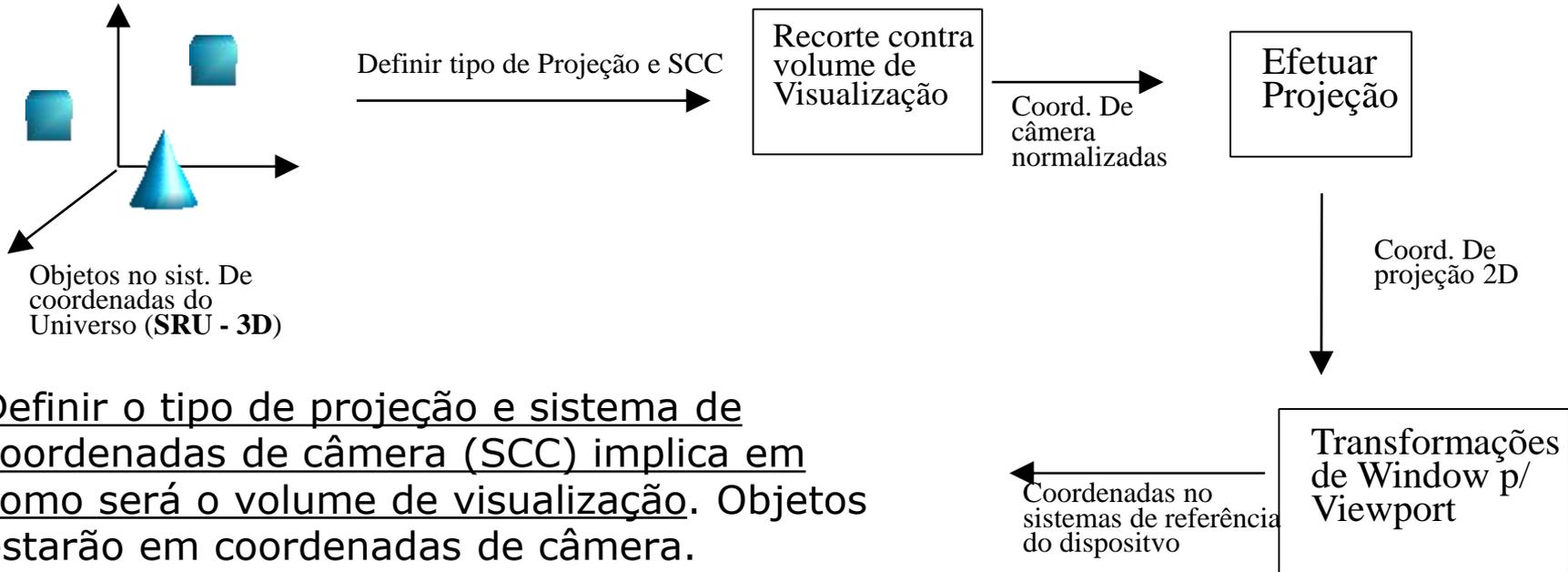
- Definir o tipo de projeção e sistema de coordenadas de câmera (SCC) implica em como será o volume de visualização. Objetos estarão em coordenadas de câmera.
- Recorte contra o volume de visualização 3D. Coordenadas de câmera 3D normalizadas e recortadas.

Pipeline de Visualização – 3D



- Definir o tipo de projeção e sistema de coordenadas de câmera (SCC) implica em como será o volume de visualização. Objetos estarão em coordenadas de câmera.
- Recorte contra o volume de visualização 3D. Coordenadas de câmera 3D normalizadas e recortadas.
- Efetuar projeção (livrar-se de uma dimensão). Coordenadas de projeção.

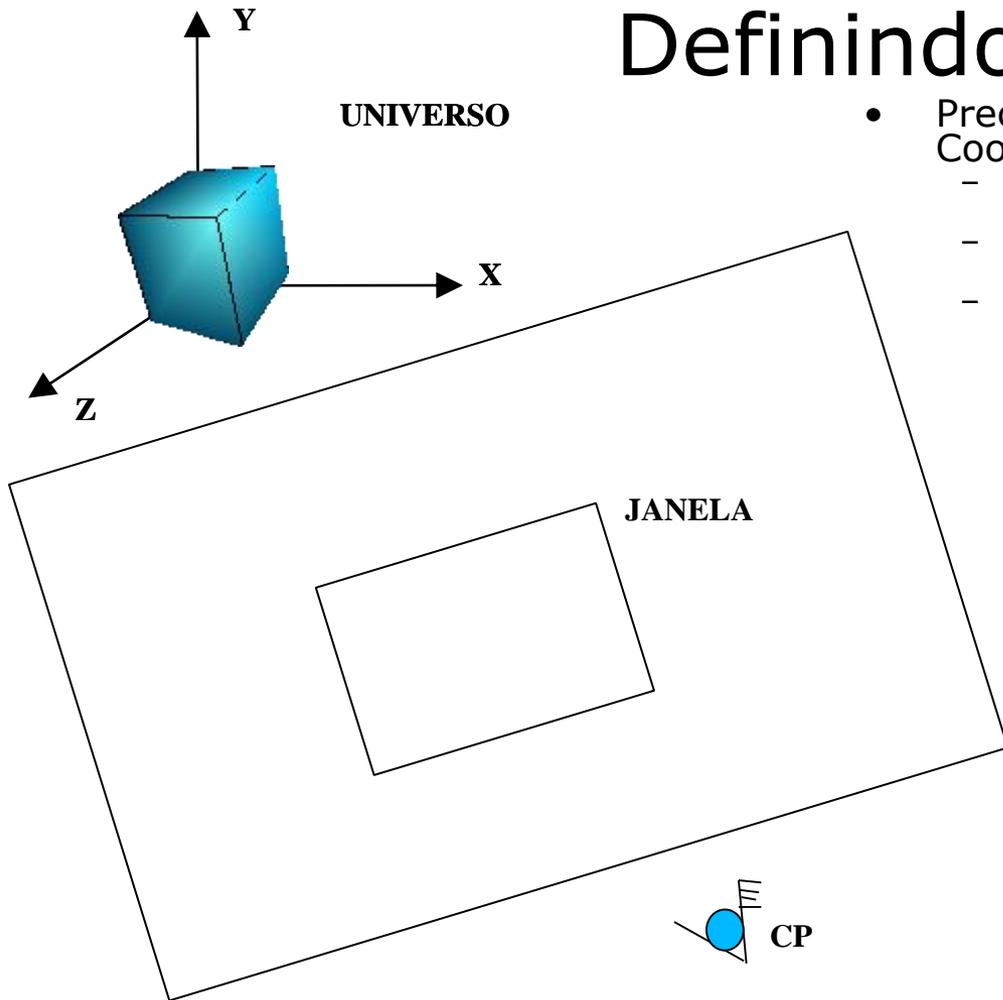
Pipeline de Visualização – 3D



- Definir o tipo de projeção e sistema de coordenadas de câmera (SCC) implica em como será o volume de visualização. Objetos estarão em coordenadas de câmera.
- Recorte contra o volume de visualização 3D. Coordenadas de câmera 3D normalizadas e recortadas.
- Efetuar projeção (livrar-se de uma dimensão). Coordenadas de projeção.

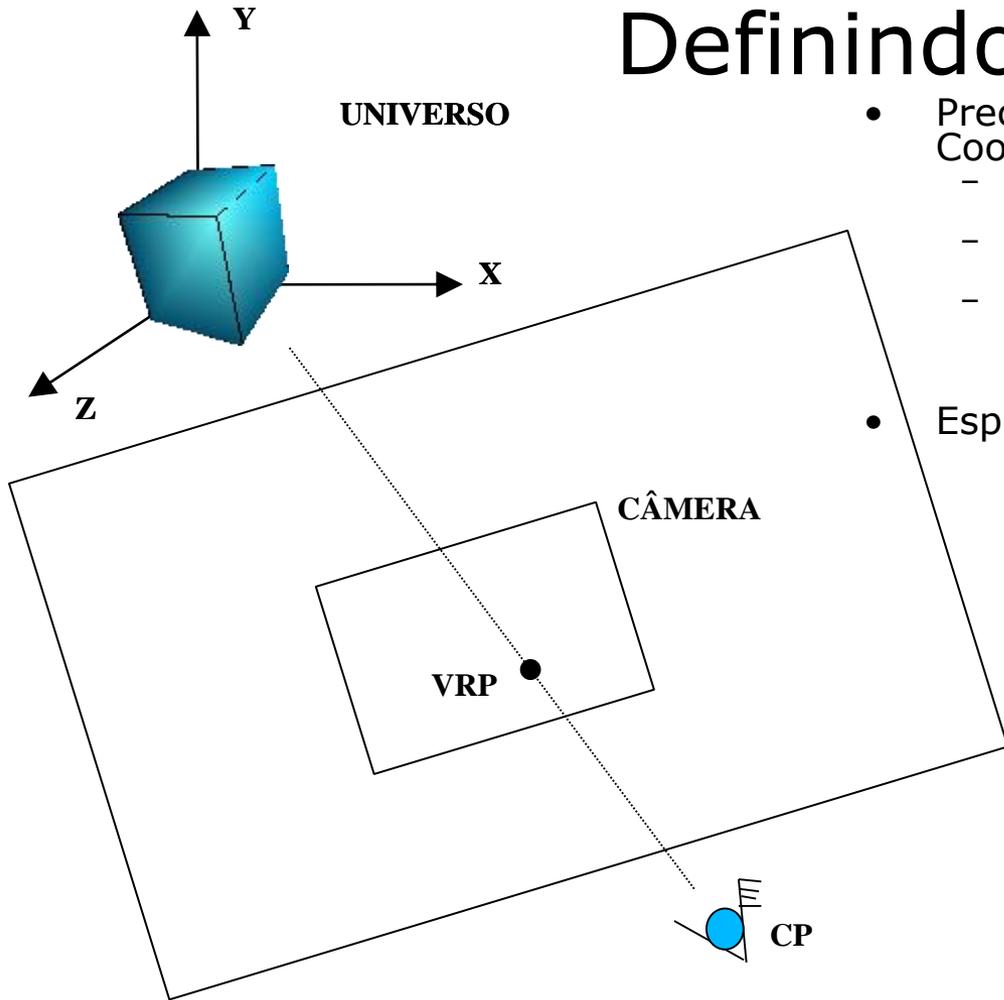
Definindo SCC

- Precisamos especificar os Sist. De Coordenadas da Câmera (SCC)
 - Plano de Projeção (posição e orientação)
 - Dentro deste plano, a JANELA
 - A posição do olho (Centro de Projeção)



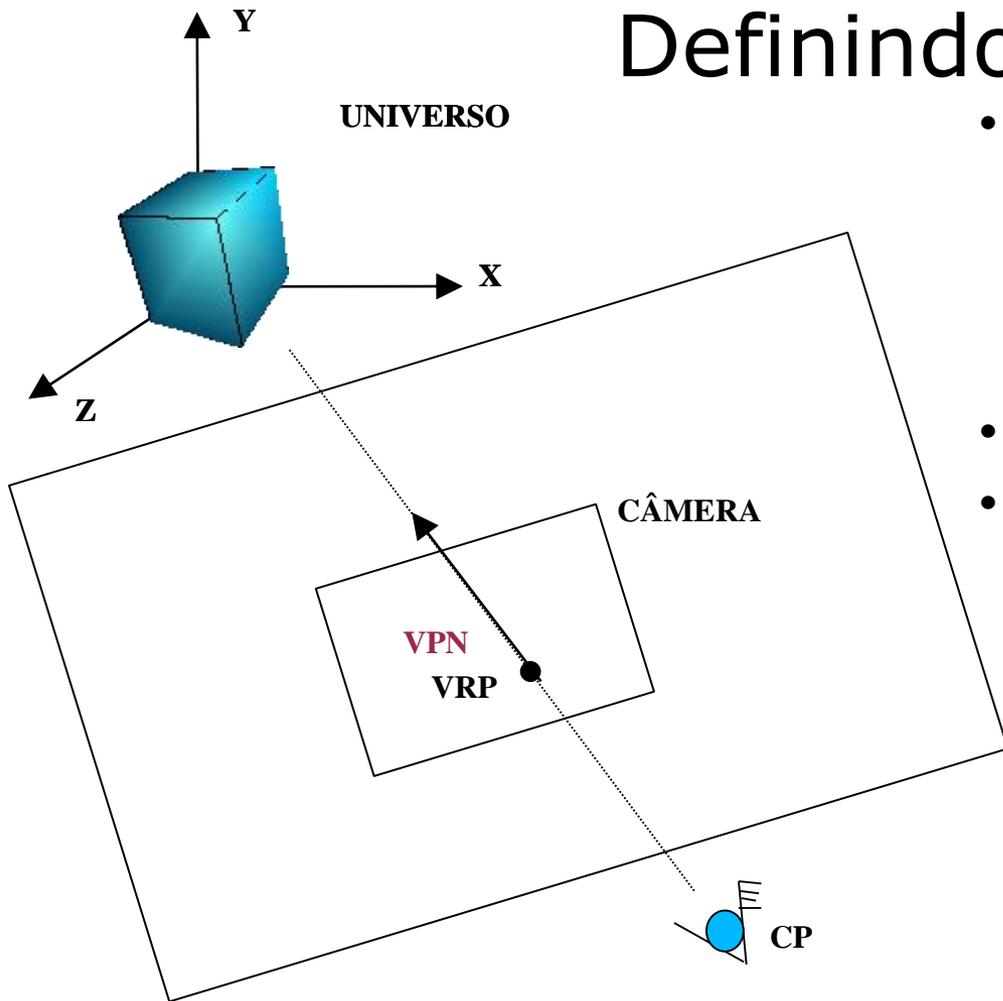
Definindo SCC

UNIVERSO



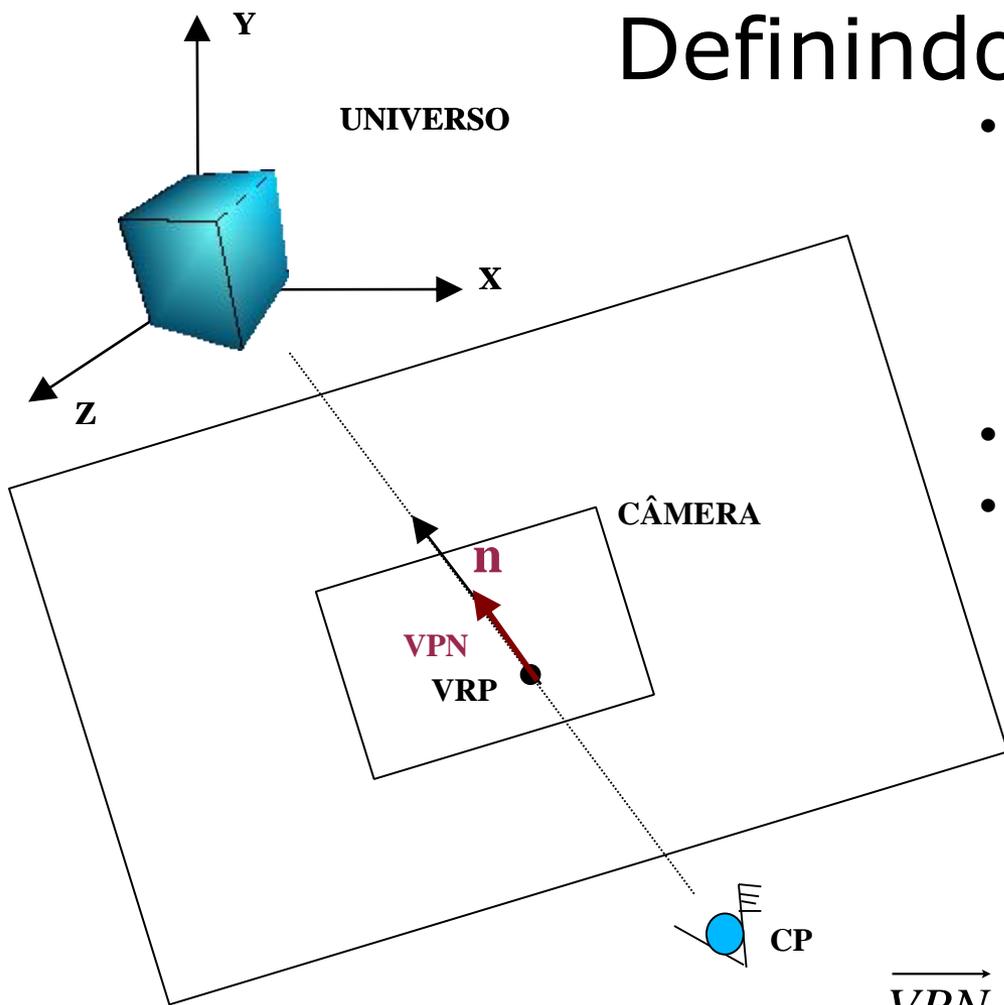
- Precisamos especificar os Sist. De Coordenadas da Câmera (SCC)
 - Plano de Projeção (posição e orientação)
 - Dentro deste plano, a JANELA
 - A posição do olho (Centro de Projeção)
- Especificar VRP (View reference point)

Definindo SCC



- Precisamos especificar os Sist. De Coordenadas da Câmera (SCC)
 - Plano de Projeção (posição e orientação)
 - Dentro deste plano, a JANELA
 - A posição do olho (Centro de Projeção)
- Especificar VRP (View reference point)
- VPN - View Plane Normal (lookat)

Definindo SCC



- Precisamos especificar os Sist. De Coordenadas da Câmera (SCC)
 - Plano de Projção (posição e orientação)
 - Dentro deste plano, a JANELA
 - A posição do olho (Centro de Projção)
- Especificar VRP (View reference point)
- VPN - View Plane Normal (lookat)

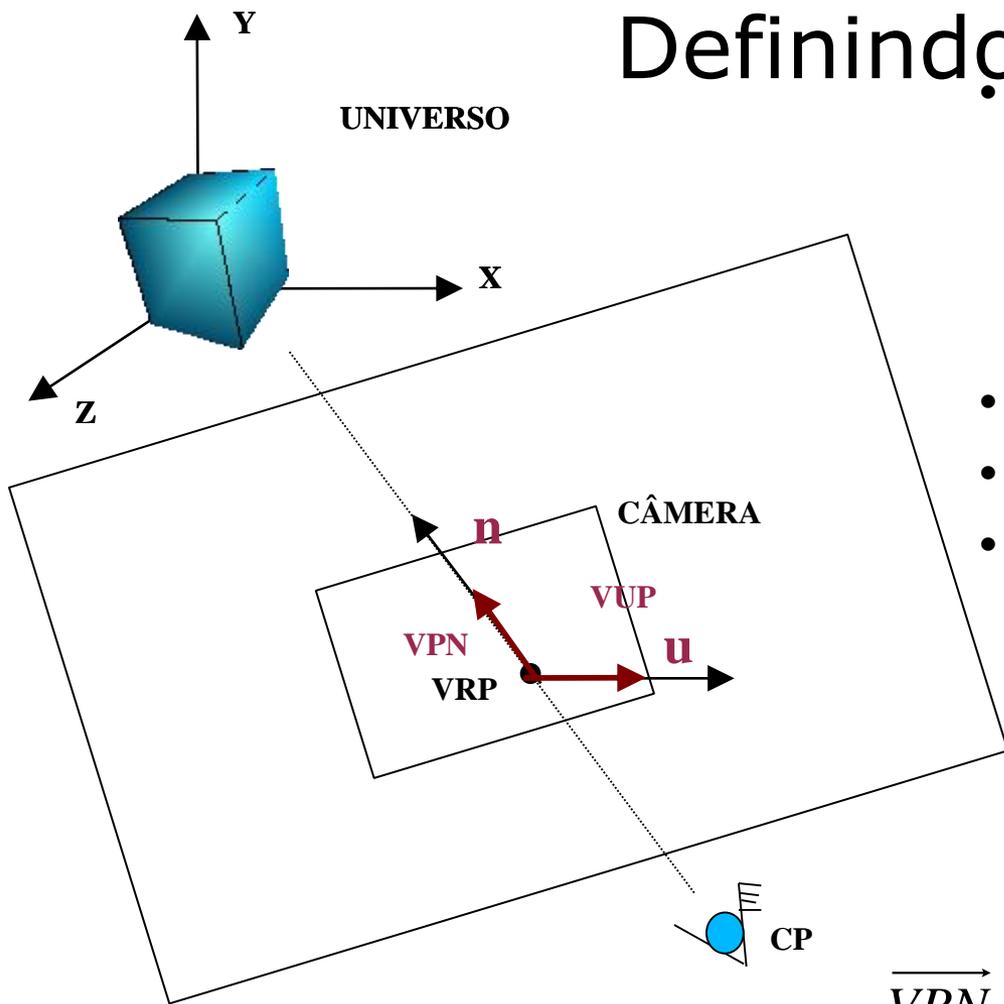
$$\vec{n} = \frac{\overrightarrow{VPN}}{|\overrightarrow{VPN}|}$$

Definindo SCC

Precisamos especificar os Sist. De Coordenadas da Câmera (SCC)

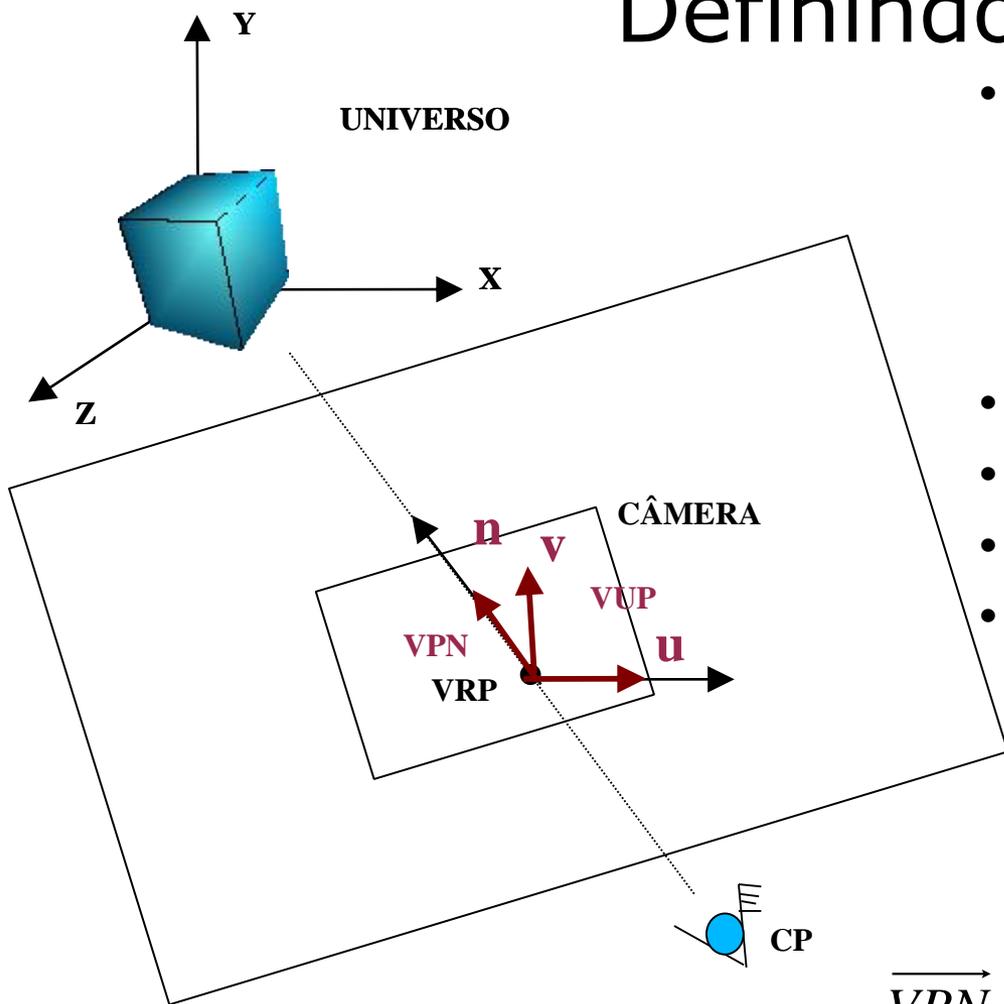
- Plano de Projeção (posição e orientação)
- Dentro deste plano, a JANELA
- A posição do olho (Centro de Projeção)

- Especificar VRP (View reference point)
- VPN - View Plane Normal (lookat)
- Segundo eixo do SRC ou SCC é VUP



$$\vec{n} = \frac{\overrightarrow{VPN}}{|\overrightarrow{VPN}|} \quad \vec{u} = \overrightarrow{VUP}$$

Definindo SCC

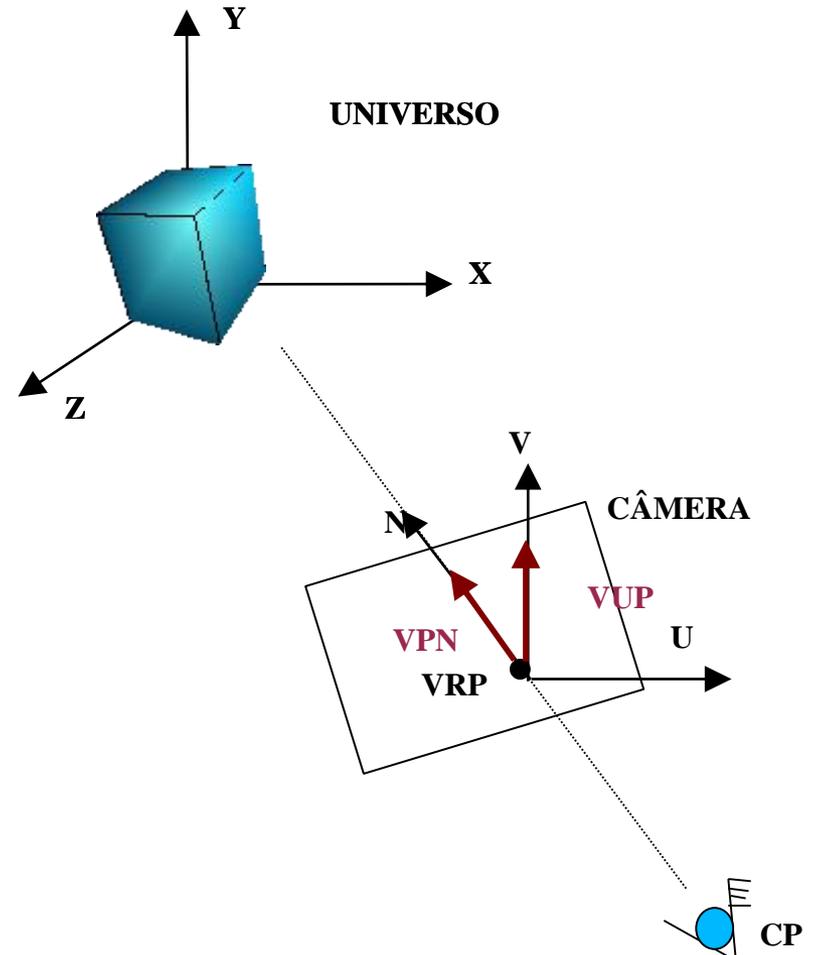
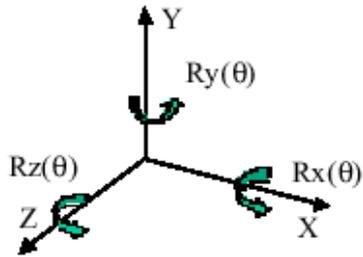


- Precisamos especificar os Sist. De Coordenadas da Câmera (SCC)
 - Plano de Projeção (posição e orientação)
 - Dentro deste plano, a JANELA
 - A posição do olho (Centro de Projeção)
- Especificar VRP (View reference point)
- VPN - View Plane Normal (lookat)
- Segundo eixo do SRC ou SCC é VUP
- Terceiro eixo é produto vetorial de u e n

$$\vec{n} = \frac{\overrightarrow{VPN}}{|\overrightarrow{VPN}|} \quad \vec{u} = \overrightarrow{VUP} \quad \vec{v} = \vec{n} \times \vec{u}$$

Efeitos de câmera

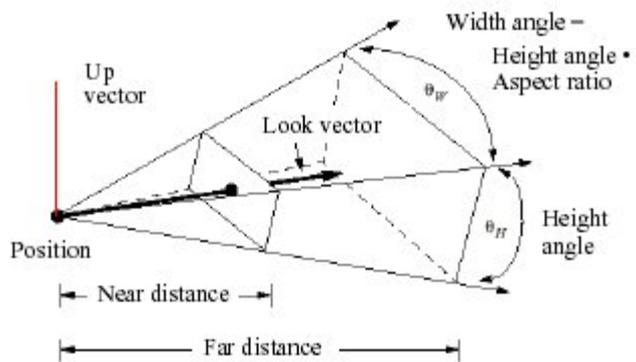
- Qual o efeito de inverter o VUP?
- Mudar a orientação da câmera



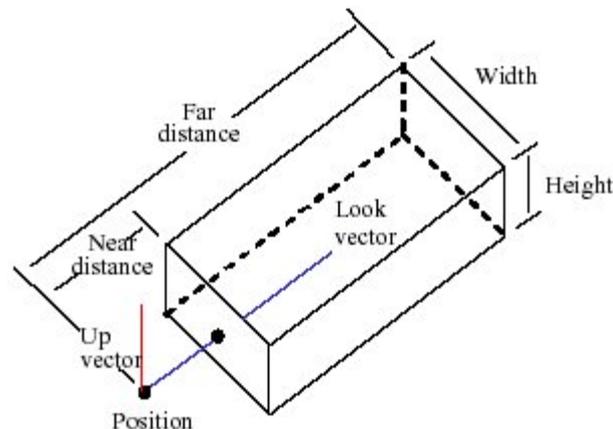
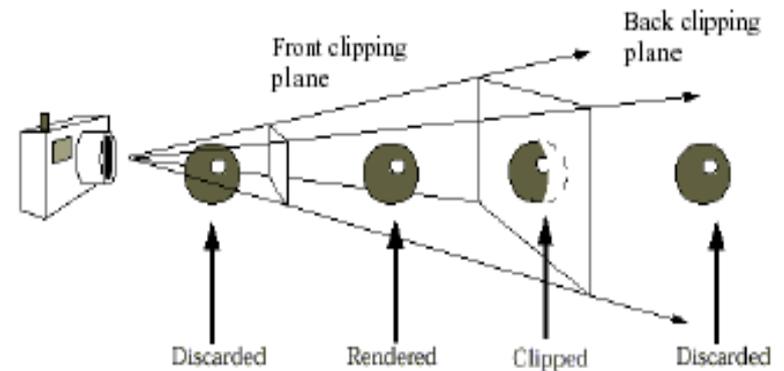
- Mudar a posição da câmera

View Volume

- Finalmente, precisamos definir uma área do espaço que será considerada (o view volume) - Frustum
- Somente os objetos dentro do view volume serão considerados



Perspectiva - Frustum

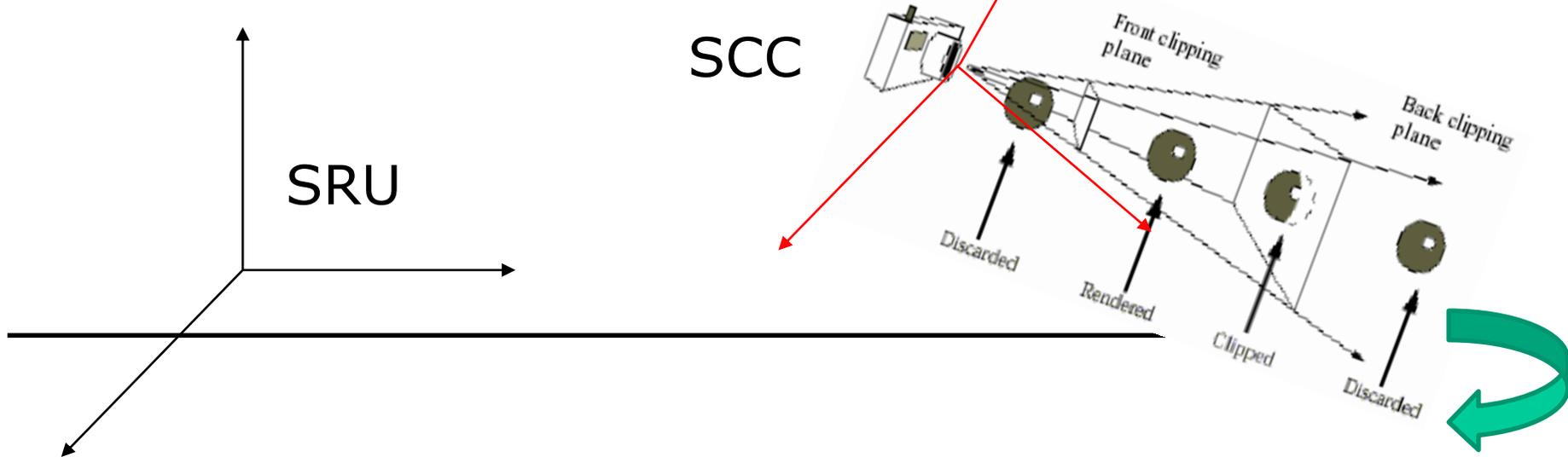


Paralela - Paralelepípedo



Mapeamento das Coordenadas de Universo no SCC

- Uma vez que o SCC é estabelecido, todo o processamento gráfico subsequente é feito neste sistema
- Transformação do Universo para SCC feita através de uma única matriz M
- Esta transformação envolve no máximo 3 translações até VRP e 3 rotações para ajustar o sistema de universo na posição do SCC

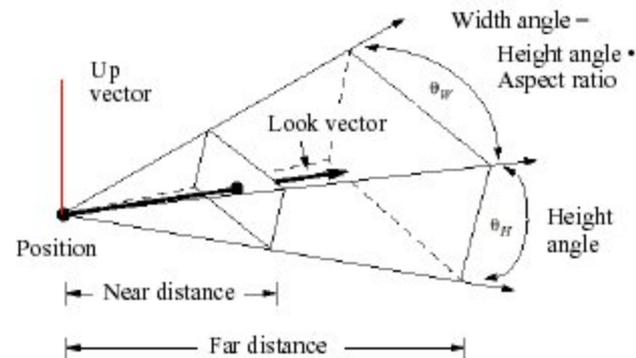


Recorte 3D

- Para o recorte em 3D, veremos a extensão do algoritmo de Cohen-Sutherland em 2D
- O algoritmo de Cohen-Sutherland classifica as extremidades das linhas de acordo com sua posição no espaço em relação ao volume de visualização
- Em 3D, esta classificação é formada por 6 bits de acordo com o seguinte
 - bit 1: ponto está acima do volume
 - bit 2: ponto está abaixo do volume
 - bit 3: ponto está à direita do volume
 - bit 4: ponto está à esquerda do volume
 - bit 5: ponto está atrás do volume
 - bit 6: ponto está à frente do volume

Recorte 3D

- Em 3D, esta classificação é formada por 6 bits de acordo com o seguinte
 - bit 1: ponto está acima do volume
 - bit 2: ponto está abaixo do volume
 - bit 3: ponto está à direita do volume
 - bit 4: ponto está à esquerda do volume
 - bit 5: ponto está atrás do volume
 - bit 6: ponto está à frente do volume
- Todos os bits em 0: Trivialmente aceito
- And \neq 0: Trivialmente rejeitado
- Outros casos, a linha deve ser recortada



FUNCIONA PARA FRUSTUM? Considera-se planos...

Recorte 3D

- O recorte da linha implica em encontrar as intersecções da linha com os planos que definem o volume de visualização
- Os cálculos de intersecção usam a representação paramétrica das linhas
 - $P_0(x_0, y_0, z_0)$ para $P_1(x_1, y_1, z_1)$
 - $x = x_0 + t(x_1 - x_0)$
 - $y = y_0 + t(y_1 - y_0)$
 - $z = z_0 + t(z_1 - z_0)$
- Como t varia de 0 a 1, as três equações dão as coordenadas de todos os pontos na linha, de P_0 a P_1

Interseção entre retas

Dada a função $y = x + 1$ e $y = 2x - 1$,

$$x + 1 = 2x - 1$$

$$x - 2x = -1 - 1$$

$$-x = -2$$

$$x = 2$$

$$y = x + 1$$

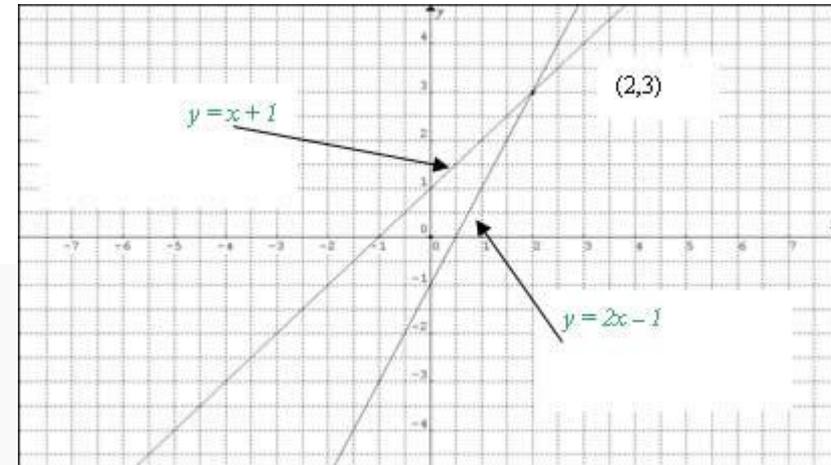
$$y = 2 + 1$$

$$y = 3$$

$$y = 2x - 1$$

$$y = 2 * 2 - 1$$

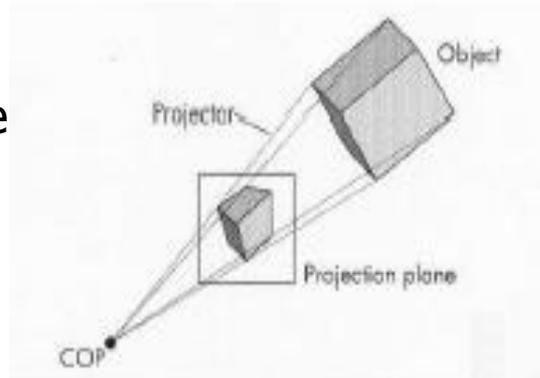
$$y = 3$$



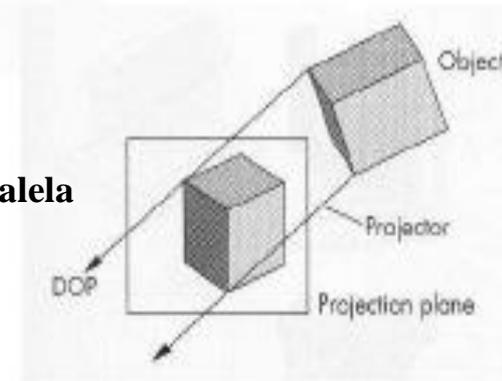
Projeções

- Genericamente, projeções transformam pontos em um sistema de coordenadas com N dimensões em pontos num sistema com dimensão menor do que N
- Para nós interessa apenas o caso de 3D -> 2D
- Projeção de um objeto no espaço através de linhas de projeção, denominadas **raios de projeção**, que têm origem em um **centro de projeção** e passam por cada parte do objeto interseccionando um **plano de projeção** onde finalmente forma-se a imagem 2D

Projeção Perspectiva

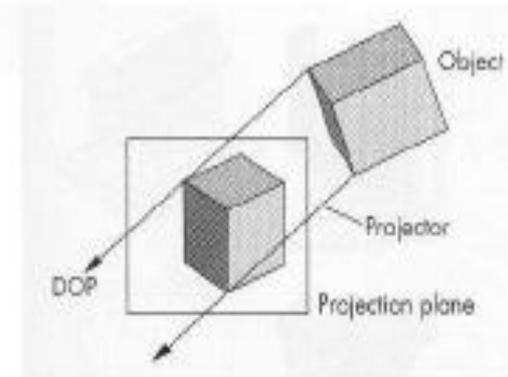
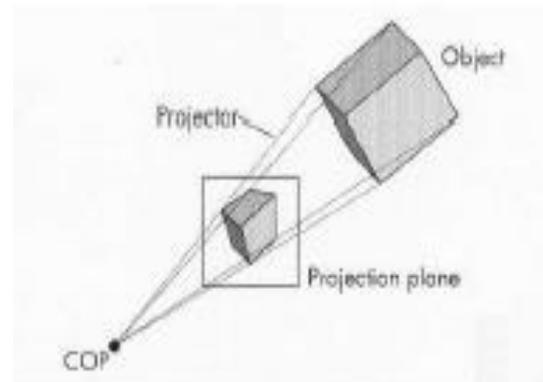


Projeção Paralela



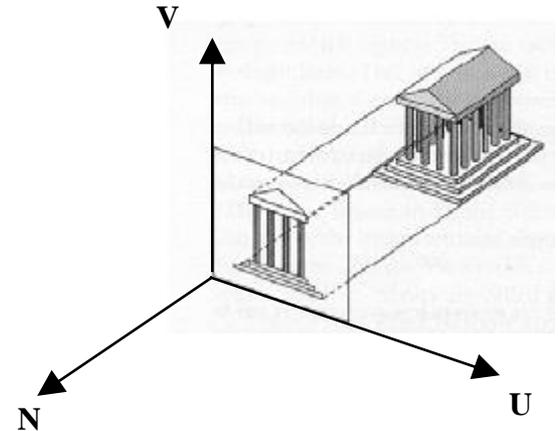
Projeções

- Diferença entre tipos de projeção
 - Centro de Projeção
- Efeito da projeção perspectiva é semelhante ao sistema visual humano
- Projeção perspectiva mais realista, mas não é útil quando precisamos medir as dimensões dos objetos
- Projeção paralela pode ser **ortográfica** ou **oblíqua** (raios de projeção não são paralelos a normal ao plano)



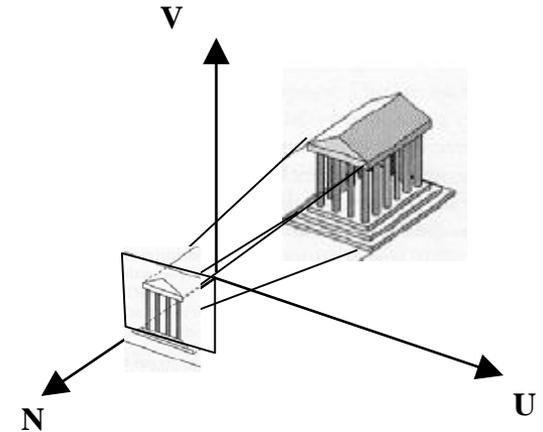
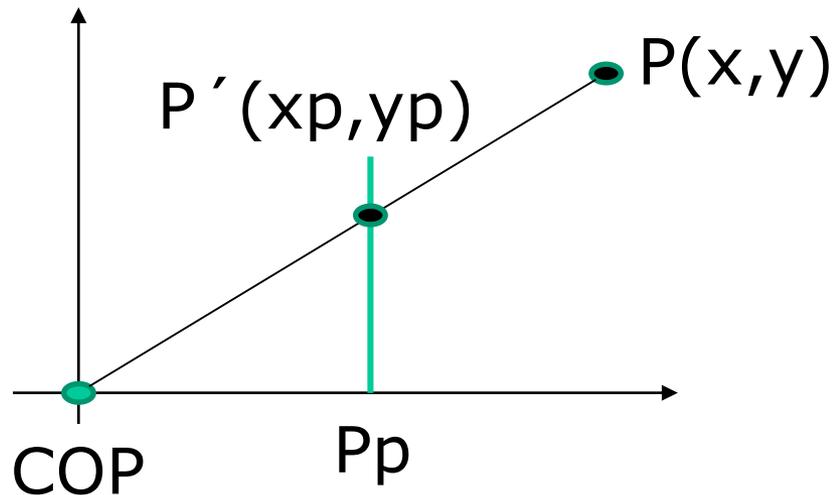
Obtendo Coordenadas de Projeção

- Precisamos “livrar-nos” de uma dimensão
 - Para projeções ortográficas (paralelas) é simples:
 - Basta descartarmos a coordenada n no SCC (assumindo o plano de projeção em $n = 0$)
 - $u_p = u$
 - $v_p = v$
 - $n_p = 0$ (ou n do pp)



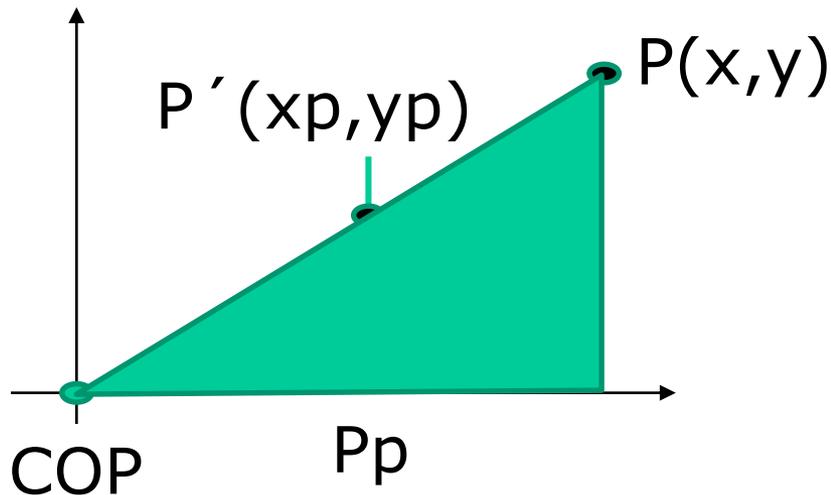
Obtendo Coordenadas de Projeção

- Para projeções perspectivas?
 - Assumindo que o plano de projeção encontra-se perpendicular ao eixo n do SCC e a uma distância d

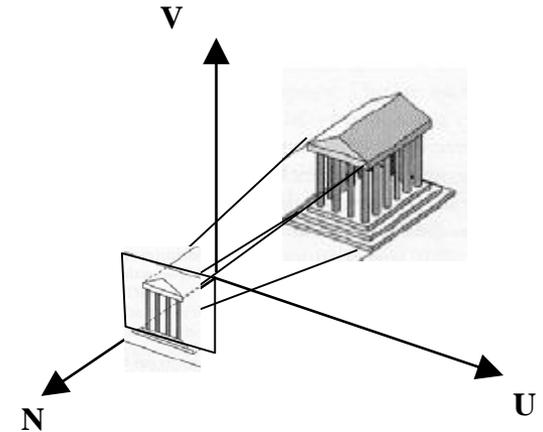


Obtendo Coordenadas de Projeção

- Para projeções perspectivas?
 - Assumindo que o plano de projeção encontra-se perpendicular ao eixo n do SCC e a uma distância d

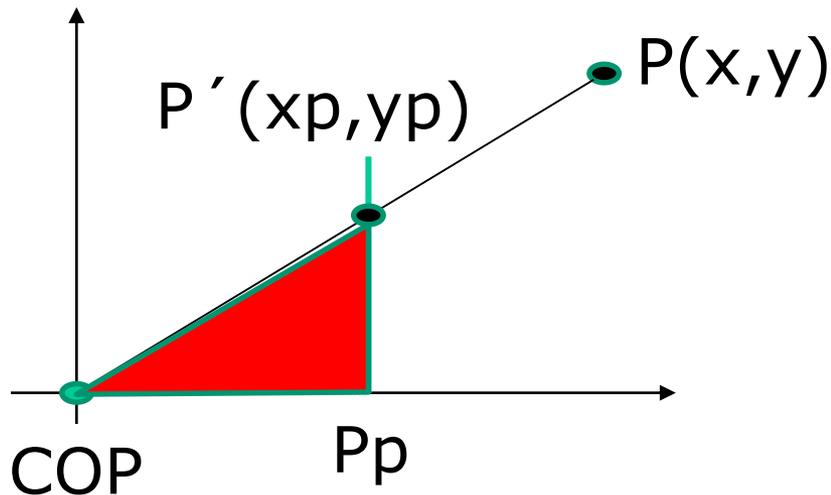


$$y/x =$$

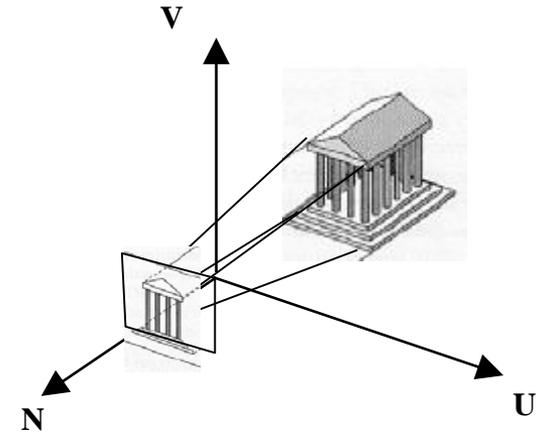


Obtendo Coordenadas de Projeção

- Para projeções perspectivas?
 - Assumindo que o plano de projeção encontra-se perpendicular ao eixo n do SCC e a uma distância d

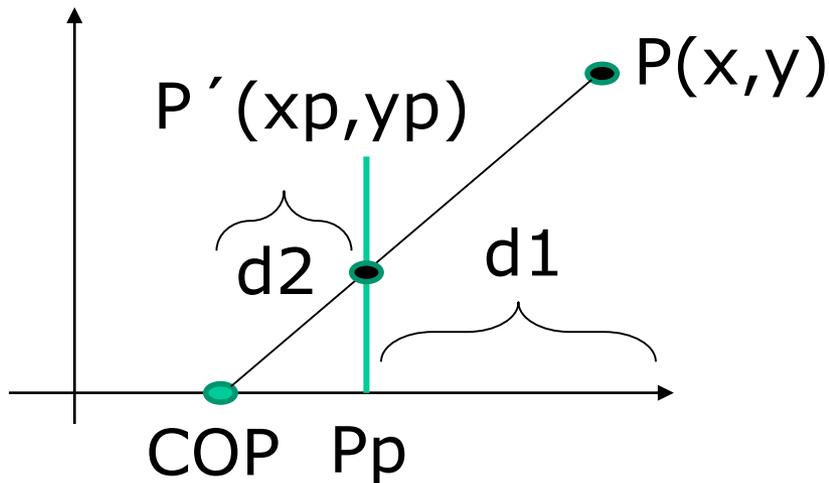


$$y/x = y_p/x_p$$

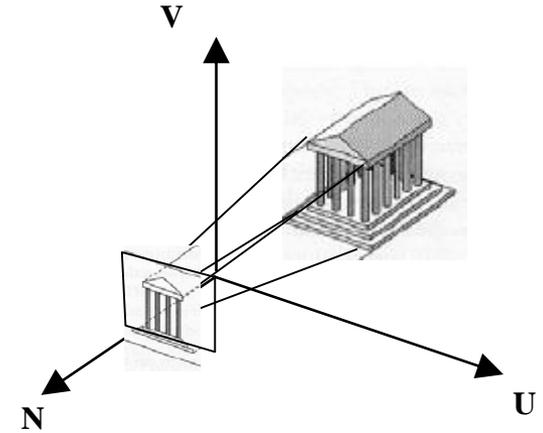


Obtendo Coordenadas de Projeção

- E se COP não estiver na origem?



$$y/(d_1+d_2) = y_p/d_2$$



Pipeline 3D

- SRO
- SRU

Definição do volume de visualização (Projeção + SCC)

- SRC
(recorte 3D)
- SRP
- SRV-SRD

Exemplo de código

```
void PosicionaObservador(void)
{
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluLookAt(0.0,80.0,200.0 (posição), 0.0,0.0,0.0 (look_at),
    0.0,1.0,0.0 (UP));
}

void EspecificaParametrosVisualizacao(void)
{
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(angle (altura do frustum em y), fAspect (x/y),0.5
    (NEAR), 500 (FAR));
    PosicionaObservador();
}
```

Transformações 3D

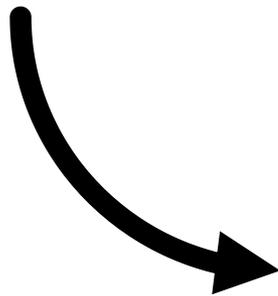
Soraia Raupp Musse

Translação – Coord. Homogêneas

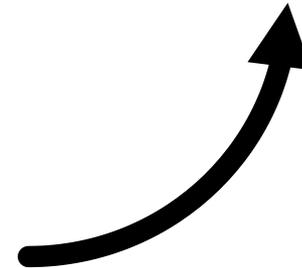
$$\vec{p}' = \vec{p} + \vec{t}$$

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\begin{cases} \frac{x'}{w'} = \frac{x}{w} + t_x \\ \frac{y'}{w'} = \frac{y}{w} + t_y \end{cases}$$



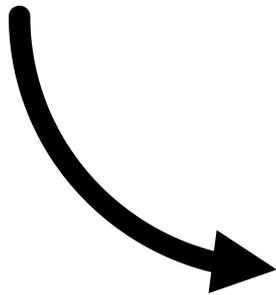
$$\begin{cases} x' = x + wt_x \\ y' = y + wt_y \\ w' = w \end{cases}$$



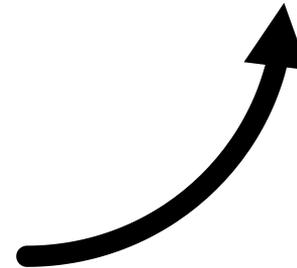
Escala – Coord. Homogêneas

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\begin{cases} \frac{x'}{w'} = s_x \frac{x}{w} \\ \frac{y'}{w'} = s_y \frac{y}{w} \end{cases}$$



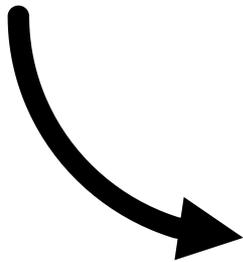
$$\begin{cases} x' = s_x x \\ y' = s_y y \\ w' = w \end{cases}$$



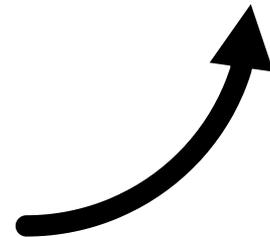
Rotação – Coord. Homogêneas

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\begin{cases} \frac{x'}{w'} = \cos \theta \frac{x}{w} - \sin \theta \frac{y}{w} \\ \frac{y'}{w'} = \sin \theta \frac{x}{w} + \cos \theta \frac{y}{w} \end{cases}$$



$$\begin{cases} x' = \cos \theta x - \sin \theta y \\ y' = \sin \theta x + \cos \theta y \\ w' = w \end{cases}$$

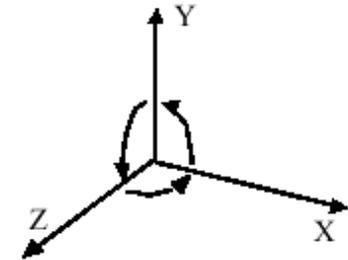


Transformações 3D

- Translação – `glTranslatef(dx, dy, dz)`

– $T(dx, dy, dz)$:

$$\begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Sistema de Coordenadas
“mão direita”

- Escala – `glScalef(Sx, Sy, Sz)`

– $S(Sx, Sy, Sz)$:

$$\begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Transformações 3D

- Rotação – `glRotatef(angle,x,y,z)`

$$R_z(\theta): \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_x(\theta): \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_y(\theta): \begin{bmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

