

CIV2802 – Sistemas Gráficos para Engenharia
2024.1

Transformações Geométricas 2D



Luiz Fernando Martha

André Pereira



Transformações Geométricas

$$\begin{aligned} T: \mathbf{R}^n &\rightarrow \mathbf{R}^m \\ \{P\} &\rightarrow \{P'\} = T\{P\} \end{aligned}$$

Transformação Linear:

$$T(\alpha \{P\} + \beta \{Q\}) = \alpha T\{P\} + \beta T\{Q\}$$

$$\begin{aligned} \forall \alpha, \beta \in \mathbf{R} \\ \{P\}, \{Q\} \in \mathbf{R}^n \end{aligned}$$

$$\Rightarrow \begin{cases} T\{0\} = \{0\} \\ \{P'\} = T\{P\} = [M]\{P\} \end{cases}$$

Transformações Geométricas

Exemplo: $\mathbb{R}^3 \rightarrow \mathbb{R}^2$

$$T: \mathbb{R}^3 \rightarrow \mathbb{R}^2$$

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} \rightarrow \begin{Bmatrix} x' \\ y' \end{Bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{Bmatrix} x \\ y \\ z \end{Bmatrix}$$

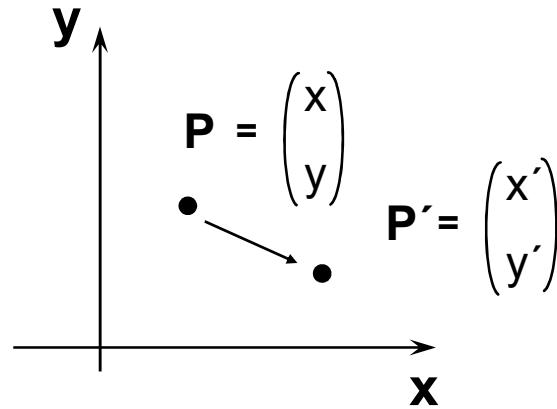
$$T \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix} = [M] \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} m_{11} \\ m_{21} \end{Bmatrix}$$

$$T \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix} = [M] \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix} = \begin{Bmatrix} m_{12} \\ m_{22} \end{Bmatrix}$$

$$T \begin{Bmatrix} 0 \\ 0 \\ 1 \end{Bmatrix} = [M] \begin{Bmatrix} 0 \\ 0 \\ 1 \end{Bmatrix} = \begin{Bmatrix} m_{13} \\ m_{23} \end{Bmatrix}$$

Transformações Lineares

$$\mathbb{R}^2 \rightarrow \mathbb{R}^2$$



$$T(a_1 \mathbf{P}_1 + a_2 \mathbf{P}_2) = a_1 T(\mathbf{P}_1) + a_2 T(\mathbf{P}_2)$$

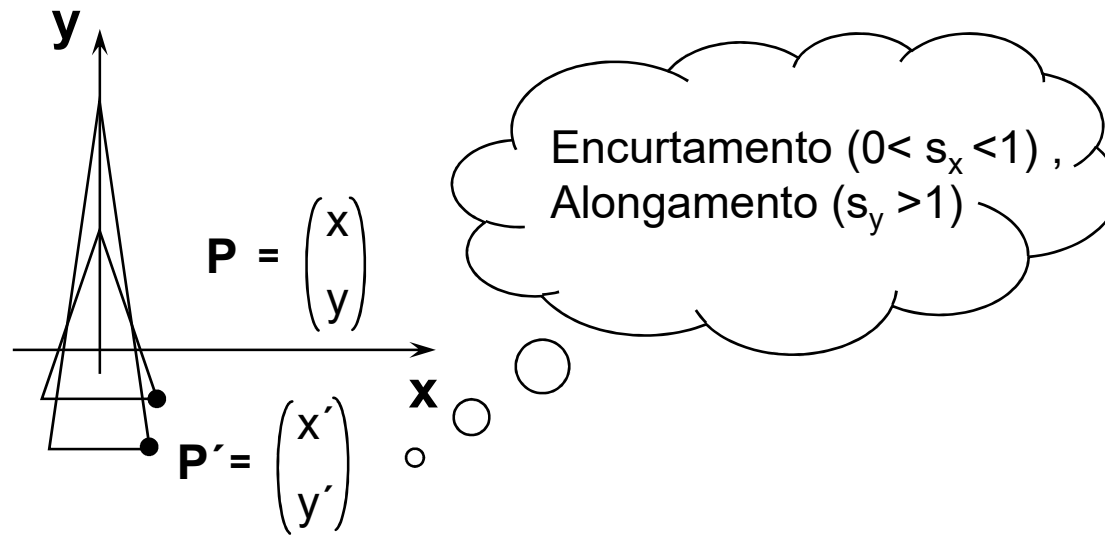
Mostre que:

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

B)

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

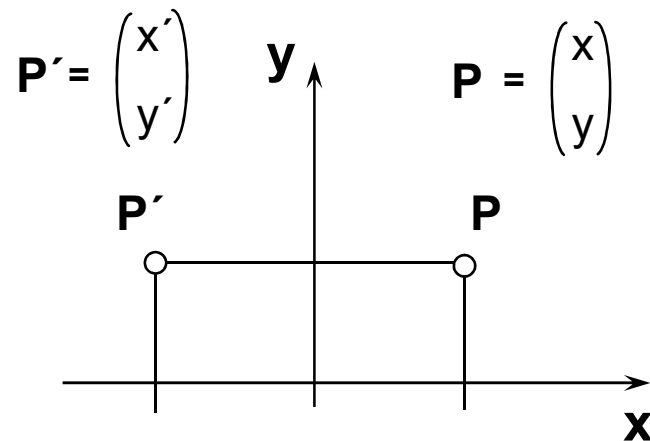
Transformação Linear (escala em relação à origem)



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

Transformações Lineares

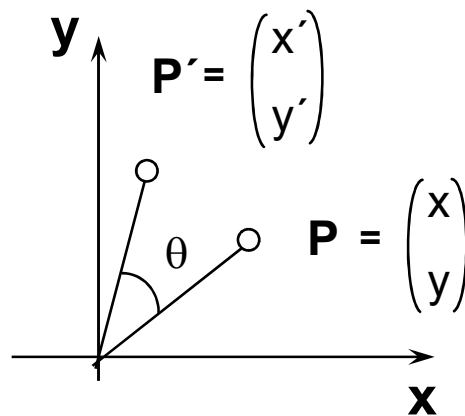
(espelhamento em relação ao eixo y)



$$\begin{aligned}x' &= -1 \cdot x \\ y' &= y\end{aligned}$$

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

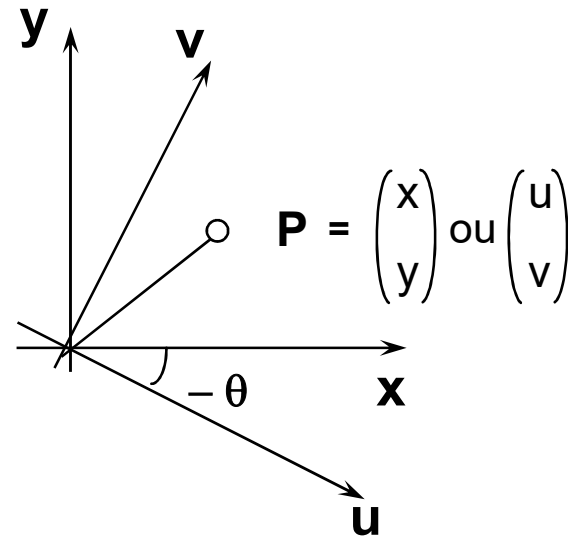
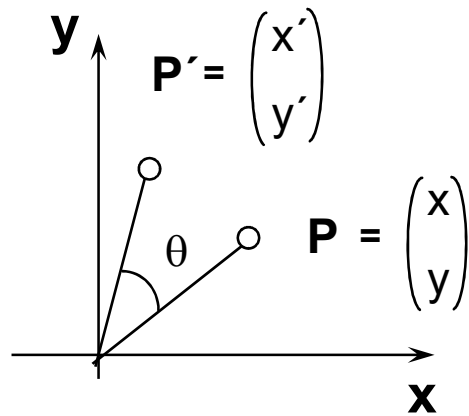
Transformações Lineares (Rotação em relação à origem)



$$\begin{aligned}x' &= x \cdot \cos \theta - y \cdot \sin \theta \\y' &= x \cdot \sin \theta + y \cdot \cos \theta\end{aligned}$$

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

Transformação Linear (rotação vs. mudança de base)



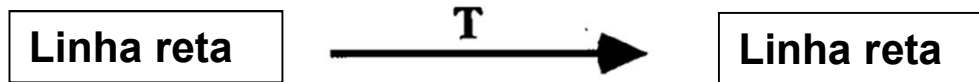
$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{bmatrix} \cos \theta & -\text{sen } \theta \\ \text{sen } \theta & \cos \theta \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

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

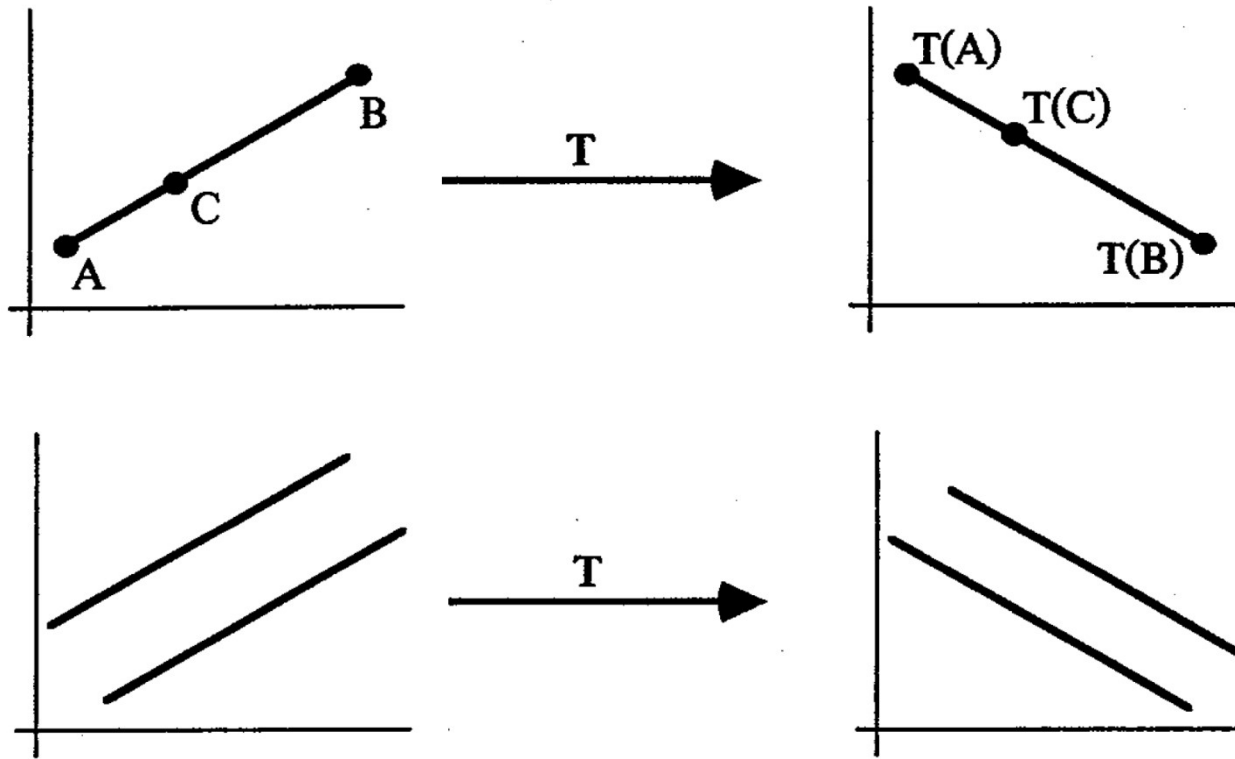
Rotação de um ponto por um ângulo θ tem o mesmo efeito da rotação dos eixos da base por um ângulo $-\theta$. A matriz que implementa a transformação é construída por linhas, colocando os cossenos diretores dos vetores da nova base em relação à antiga em cada linha.

Transformações Lineares

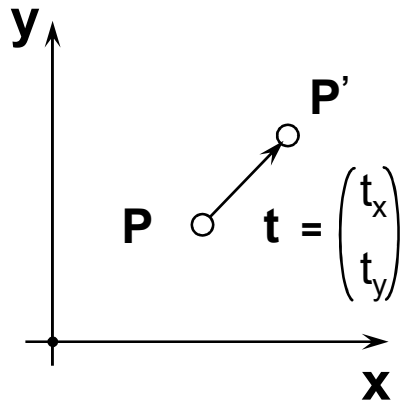
Propriedades



(Propriedade muito importante em computação gráfica)



Transformação por Translação



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

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



Não pode ser escrita dessa maneira

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



Não é prático para implementação

Transformações Afins (transformations lineares com translação)

São transformações que preservam colinearidade, razão e paralelismo.

A matriz de uma transformação afim tem a seguinte forma:

$$[T] = \begin{bmatrix} a & b & e \\ c & d & f \\ 0 & 0 & 1 \end{bmatrix}$$

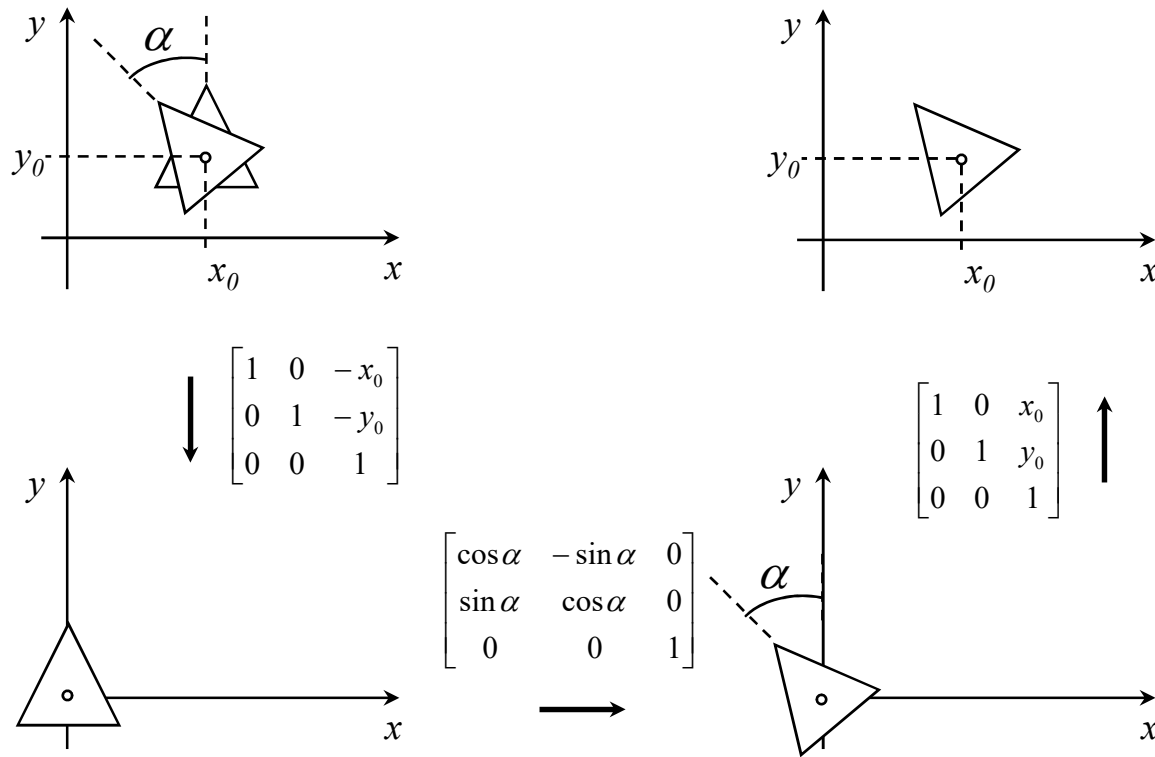
de tal forma que

$$\begin{bmatrix} a & b & e \\ c & d & f \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} x \\ y \\ 1 \end{Bmatrix} = \begin{Bmatrix} x' \\ y' \\ 1 \end{Bmatrix}$$

Sendo assim, pode-se trabalhar apenas com uma matriz 2x3:

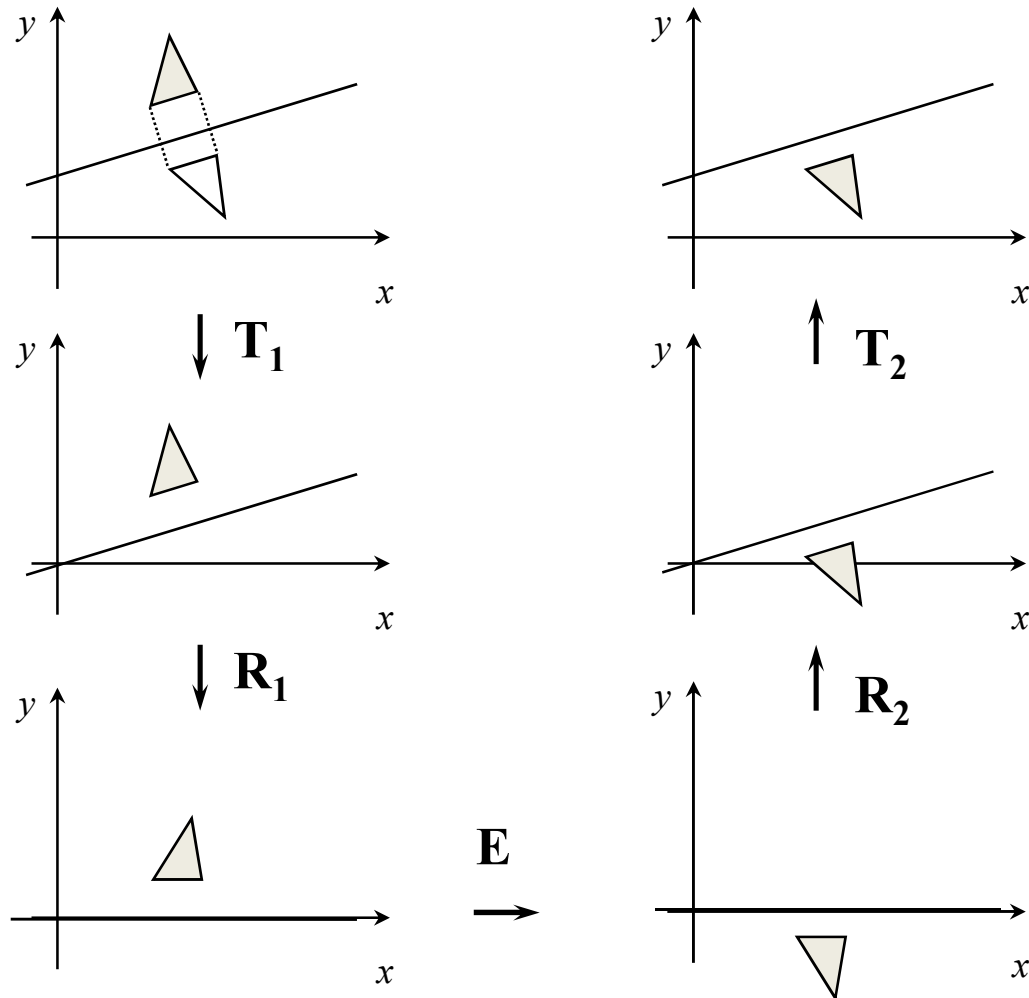
$$\begin{Bmatrix} x' \\ y' \end{Bmatrix} = \begin{bmatrix} a & b & e \\ c & d & f \end{bmatrix} \begin{Bmatrix} x \\ y \\ 1 \end{Bmatrix}$$

Concatenação



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

Concatenação de Transformações



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

Definições

Sistema de Coordenadas do Mundo (*World Coordinate System*) - É o espaço de coordenadas do objeto (modelo). É o espaço onde o modelo de aplicação é definido.

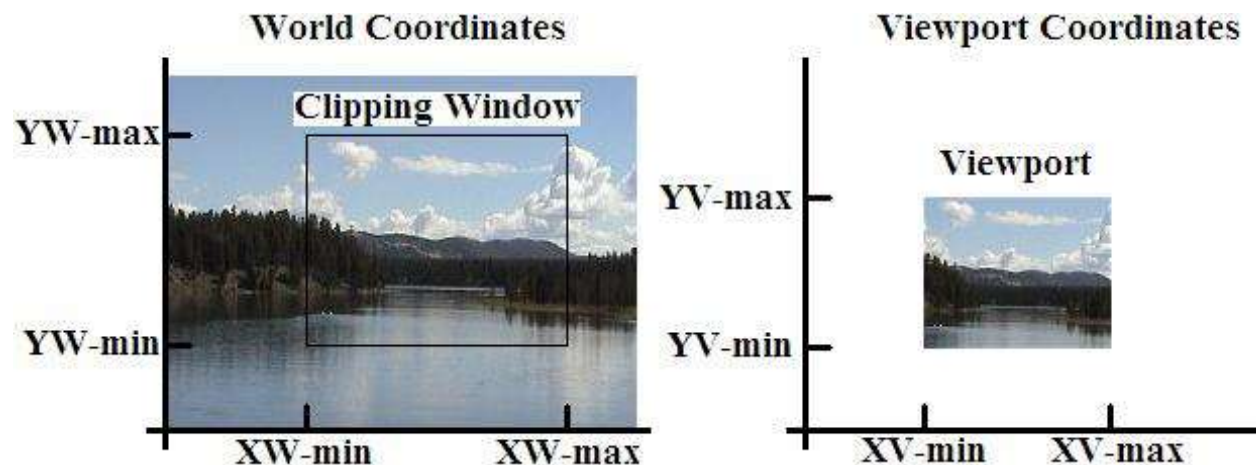
Sistema de Coordenadas da Tela (*Screen Coordinate System*) - É o espaço de coordenadas onde a imagem é mostrada.

Janela do Mundo (*World Window* ou *Clipping Window*) - É o retângulo no sistema de coordenadas do mundo que define a região que é mostrada.

Janela de Interface - É a janela aberta na tela em que a imagem será mostrada.

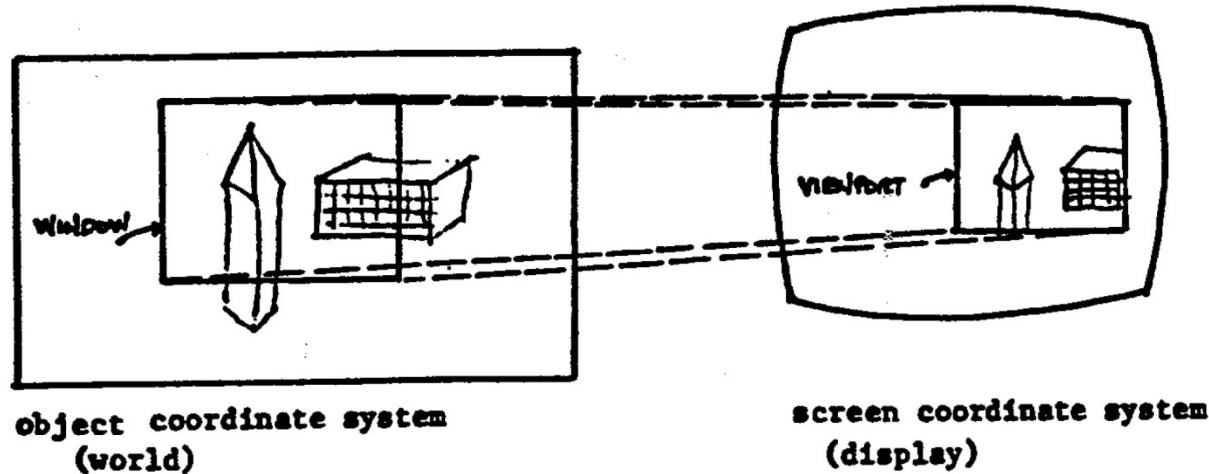
Viewport - É a porção retangular da janela de interface que define onde a imagem é mostrada (usualmente é toda a janela de interface, mas em alguns casos pode ser modificada para uma porção da janela de interface).

Transformação *Window-Viewing* - É o processo de mapeamento de uma *world window* em coordenadas do mundo para coordenadas da tela de um *Viewport*.



(Fonte: Hankins, 2014 – Lecture Notes on Computer Graphics at MTSU)

Transformação Window-Viewport



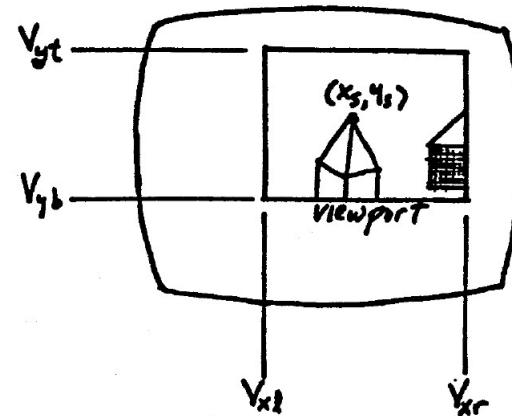
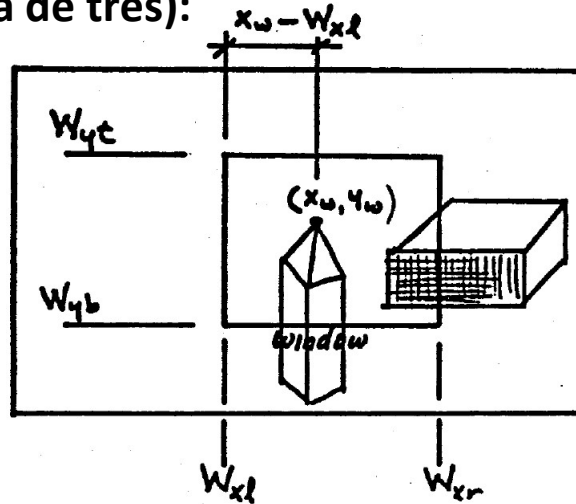
Um *viewport* é uma região na tela, usualmente retangular, dentro do qual a imagem é desenhada. O *viewport* pode ser toda a tela ou uma porção desta.

Uma *window* (janela) define a porção do espaço do objeto (mundo) a ser desenhado que vai ser vista no *viewport*. A *window* é definida nas coordenadas do espaço do objeto (*world coordinates*).

A *window* e o *viewport* são uma forma conveniente para definir a transformação da imagem do objeto na tela. Por exemplo, se nós quisermos examinar uma figura grande, nós mantemos os tamanhos da *window* e do *viewport* fixos e movemos a posição da *window*. Isto possibilita percorrer a figura com uma magnificação fixa. Para magnificar ou reduzir a imagem do objeto no *viewport*, basta reduzir ou aumentar o tamanho da *window*, respectivamente.

Transformação Window-Viewport é uma Transformação Afim

Proporção (regra de três):



$$x_s = \underbrace{\frac{(x_w - W_{xl})}{(W_{xr} - W_{xl})}}_{\text{fraction of displacement of full viewport}} \cdot \underbrace{(V_{xr} - V_{xl})}_{\text{viewport width}} + \underbrace{V_{xl}}_{\text{viewport offset}}$$

$$x_s = \frac{(V_{xr} - V_{xl})}{(W_{xr} - W_{xl})} (x_w - W_{xl}) + V_{xl}$$

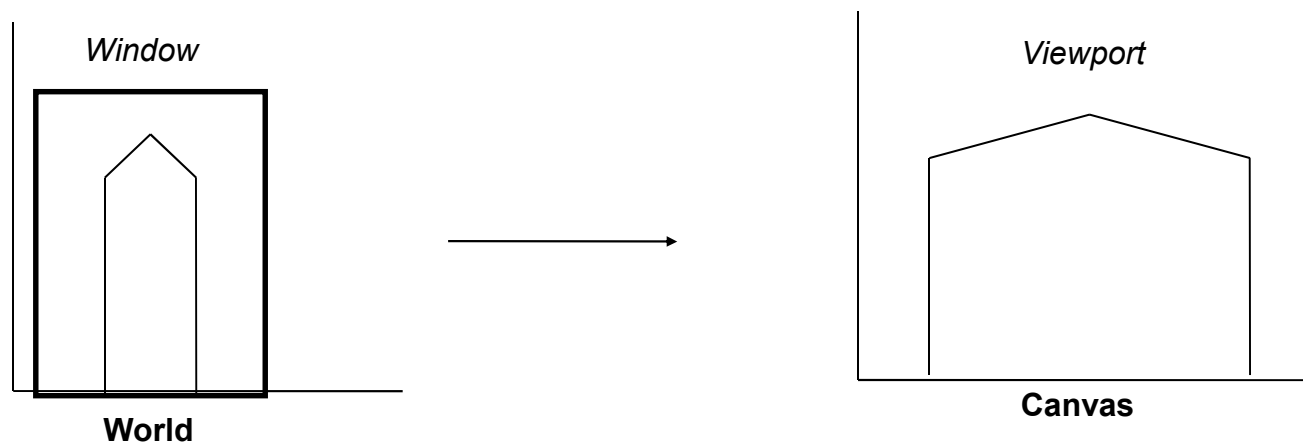
$$y_s = \frac{(y_w - W_{yb})}{(W_{yt} - W_{yb})} \cdot (V_{yt} - V_{yb}) + V_{yb}$$

$$y_s = \frac{(V_{yt} - V_{yb})}{(W_{yt} - W_{yb})} (y_w - W_{yb}) + V_{yb}$$

$$\begin{Bmatrix} x_s \\ y_s \end{Bmatrix} = \begin{bmatrix} a & 0 \\ 0 & d \end{bmatrix} \begin{Bmatrix} x_w \\ y_w \end{Bmatrix} + \begin{Bmatrix} e \\ f \end{Bmatrix} = \begin{bmatrix} a & 0 & e \\ 0 & d & f \end{bmatrix} \begin{Bmatrix} x_w \\ y_w \\ 1 \end{Bmatrix}$$

Ajuste dos limites da Window para manter razão de aspecto do Viewport

Quando os retângulos que definem *window* e *viewport* não são proporcionais, ocorre o que se chama de distorção, fenômeno este que pode ser visualizado na figura abaixo, onde a “torre de uma igreja” passou a ser vista como um “galpão”.



Para que tal efeito não ocorra, é necessário que os lados da *window* e *viewport* sejam proporcionais.

Ajuste dos limites da Window para manter razão de aspecto do Viewport (cont.)

Considerando que o *viewport* ocupa integralmente o *canvas*, para que se consiga definir um par *window-viewport* de modo a não ocorrer a distorção e utilizar toda a superfície de visualização, deve-se proceder da seguinte forma:

1. Obter as dimensões do *canvas* (*viewport*) em pixels:

w (largura) **h** (altura)

2. Calcular a razão de aspecto do retângulo do *canvas*:

h_w = **h** / **w** (**altura** / **largura**)

3. Inicializar limites da *window* com os mínimos e máximos da caixa envolvente do modelo: **xmin**, **xmax**, **ymin**, **ymax**

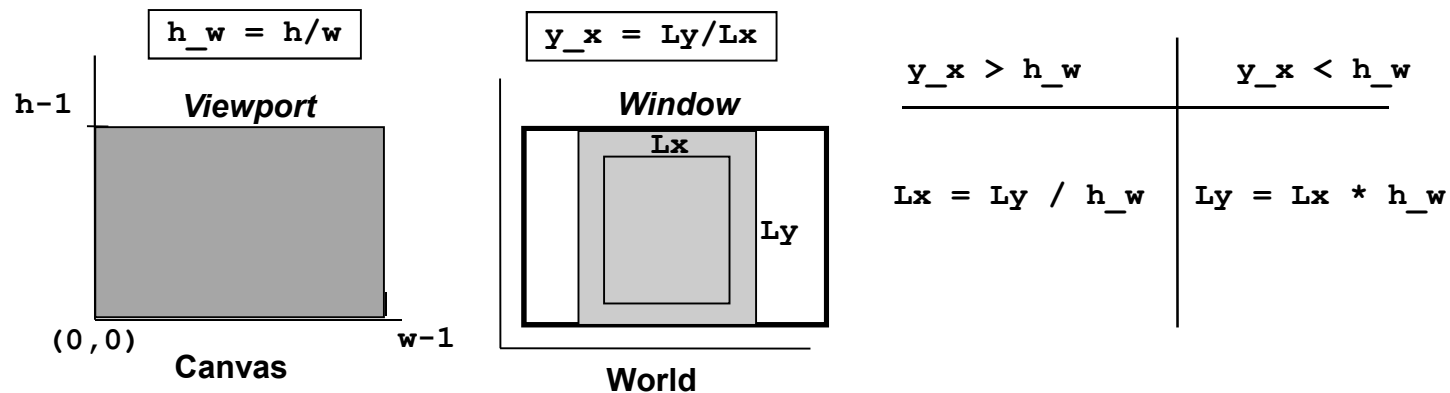
4. Calcular o centro da *window* e tamanhos mínimos com folga de 10%:

Lx = (**xmax** - **xmin**) * 1.10 **Ly** = (**ymax** - **ymin**) * 1.10

Mx = (**xmin** + **xmax**) / 2 **My** = (**ymin** + **ymax**) / 2

Ajuste dos limites da Window para manter razão de aspecto do Viewport (cont.)

5. Ajustar tamanho da *window* para que não haja distorção:



6. Calcular os limites ajustados da *window*:

$$x_{min} = Mx - Lx / 2$$

$$x_{max} = Mx + Lx / 2$$

$$y_{min} = My - Ly / 2$$

$$y_{max} = My + Ly / 2$$