Computação Gráfica - Unidade 3

Interface, Transformações 2D e Seleção, Programação orientada a eventos. Elementos de interface, Eventos e atributos de elementos de interface. Funções callback (teclado e mouse). Transformações de sistemas de coordenadas Transformações geométricas 2D, Algoritmos de seleção, Boundaring Box.

Objetivo: demonstrar conhecimento no desenvolvimento de sistemas com interface gráfica com o usuário. Interpretar, especificar e desenvolver aplicações simples com transformações geométricas.
(aulaRabiscos.drawio.svg)

Atividades - Aula

Rabiscos

em branco

Material

Algoritmo de Seleção

Algoritmo de seleção

Livro utilizado

Nº Obra Sistema	244342
Número de chamada	516.00285, F475i, CG (Anote para localizar o material)
Autor	Figueiredo, Luiz Henrique de
Título	Introdução a geometria computacional /Luiz Henrique de Figueiredo e Paulo César Pinto Carvalho. -Rio de Janeiro : IMPA, 1991. - 111p. :il.
Notas	Trabalho apresentado no 18. Colóquio Brasileiro de Matemática.
Assunto	Geometria Processamento de dados
Autor Secundário	César, Paulo,1952-
Autor Secundário	Instituto de Matemática Pura e Aplicada (Brasil).

Registro	Volume	Biblioteca	Situação	Usuário	Data Início	Data Fim
CG210707		Central	Disponível			
CG210708		Central	Empréstimo	133658	25/04/2011	02/05/2011
CG210709		Central	Disponível			
CG210710		Central	Disponível			
CG210711		Central	Disponível			
CG210712		Central	Disponível			
CG210713		Central	Disponível			

Algoritmo de seleção

O algoritmo a seguir testa se um ponto de seleção $p_{sel} = (x_{sel}, y_{sel})$ (ponto clicado pelo usuário) está dentro ou fora de um polígono não convexo P. O polígono é formado por uma lista de n vértices ou pontos $p_i = (x_i, y_i)$.

A idéia principal do algoritmo é a partir do ponto de seleção p_{sel} é disparado um "tiro". Se o tiro cruzar a fronteira do polígono uma quantidade ímpar de vezes, o ponto de seleção está dentro do polígono. Caso contrário está fora. A direção do tiro forma uma reta que chamamos de L. Para facilitar o cálculo de intersecção de L com cada lado $\overline{p_i p_{i+1}}$ do polígono, consideramos a reta L horizontal.

A figura 1 ilustra o tiro disparado sobre a reta L a partir do ponto de seleção p_{sel}. Os pontos p_i e p_{i+1} definem o lado de índice i (lado $\overline{p_i p_{i+1}}$) do polígono. Note que exitem 3 intersecções do tiro com a fronteira do polígono, indicando que p_{sel} está dentro.

Algoritmo de seleção

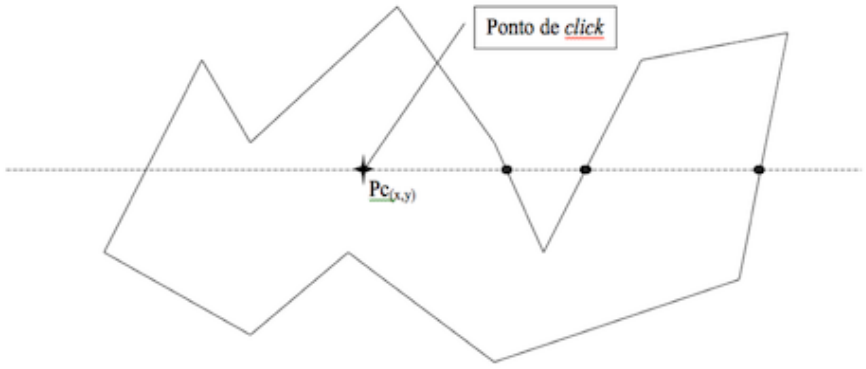


Figura 1: Ponto de seleção dentro do polígono

Agora vamos ao algoritmo. O laço principal, com contador i, refere-se a cada lado do polígono. Se o lado for horizontal, o teste de intersecção da reta L com o lado é um teste simples de comparação de cordenadas. Sempre que o ponto de seleção estiver sobre um dos lados, podemos parar o algoritmo considerando que o usuário selecionou o polígono.

Algoritmo: PontoEmPoligono(P, p_{sel})

$N_{int} \leftarrow 0$;
para i=1 **até** n **faça**
  se $y_i \neq y_{i+1}$ **então**
    $(x_{int}, y_{int}) \leftarrow$ pto de intersecao do lado $\overline{p_i p_{i+1}}$ com a reta L;
    se $x_{int} == x_s$ **então**
      p_{sel} está sobre o lado $\overline{p_i p_{i+1}}$: PARE;
    senão
      se $x_{int} > x_{sel}$ **e** $y_{int} > min(y_i, y_{i+1})$ **e** $y_{int} \leq max(y_i, y_{i+1})$ **então**
        $N_{int} \leftarrow N_{int} + 1$;
  senão
    se $y_{sel} == y_i$ **e** $x_{sel} \geq min(x_i, x_{i+1})$ **e** $x_{sel} \leq max(x_i, x_{i+1})$ **então**
      p_{sel} está sobre o lado horizontal $\overline{p_i p_{i+1}}$: PARE;
se N_{int} *é ímpar* **então**
  p_{sel} é interior a P;
senão
  p_{sel} é exterior a P;

Algoritmo 1: Algoritmo de ponto em polígono

FURB
Unidade 03 – Interface, Transformações 2D e Seleção
Prof. Dalton Reis

Algoritmo: PontoEmPoligono(P, p_{sel})

$N_{int} \leftarrow 0$;
para i=1 **até** n **faça**
  se $y_i \neq y_{i+1}$ **então**
    $(x_{int}, y_{int}) \leftarrow$ pto de intersecao do lado $\overline{p_i p_{i+1}}$ com a reta L;
    se $x_{int} == x_s$ **então**
      p_{sel} está sobre o lado $\overline{p_i p_{i+1}}$: PARE;
    senão
      se $x_{int} > x_{sel}$ **e** $y_{int} > min(y_i, y_{i+1})$ **e** $y_{int} \leq max(y_i, y_{i+1})$ **então**
        $N_{int} \leftarrow N_{int} + 1$;
  senão
    se $y_{sel} == y_i$ **e** $x_{sel} \geq min(x_i, x_{i+1})$ **e** $x_{sel} \leq max(x_i, x_{i+1})$ **então**
      p_{sel} está sobre o lado horizontal $\overline{p_i p_{i+1}}$: PARE;
se N_{int} *é ímpar* **então**
  p_{sel} é interior a P;
senão
  p_{sel} é exterior a P;

A seguir, listamos os principais elementos do pseudo-código abaixo:

- i: contador do laço principal, refere-se a cada um dos n lados do polígono ($i \leftarrow 1$ até n);
- N_{int}: número de intersecções a serem contabilizadas;
- $\overline{p_i p_{i+1}}$: cada lado do polígono;
- P: o polígono (lista de pontos ou vértices);
- p_{sel}: ponto de seleção (ponto que o usuário clicou;
- p_{int}: ponto de intersecção da reta L com o lado $\overline{p_i p_{i+1}}$;

Seleção: intersecção

Cálculo da Equação da Reta - Inclinação e interseção

Para encontrar os parâmetros *a* e *b* da reta *y=ax+b* basta considerar que *a* representa a sua inclinação e *b* o valor da ordenada *y* da reta para o qual a abscissa *x* é nula.

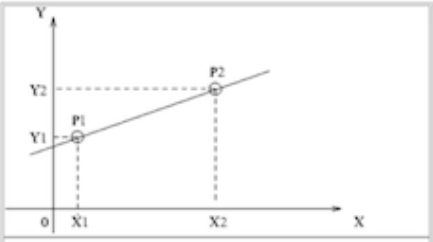


Figura: Uma reta passando por dois pontos P_1 e P_2.

Temos:

$$y_2 - y_1 = a(x_2 - x_1) \quad \Rightarrow \quad a = \frac{y_2 - y_1}{x_2 - x_1}$$

$$y_1 = \frac{y_2 - y_1}{x_2 - x_1} x_1 + b \quad \Rightarrow \quad b = y_1 - \frac{y_2 - y_1}{x_2 - x_1} x_1$$

Seleção: intersecção

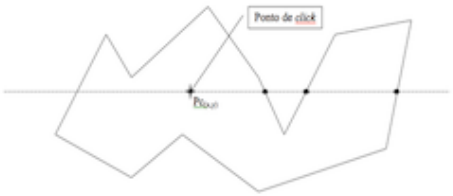
Equação Paramétrica da Reta

DSC/BCC – Computação Gráfica

$$P = P_1 + (P_2 - P_1)\cdot t$$

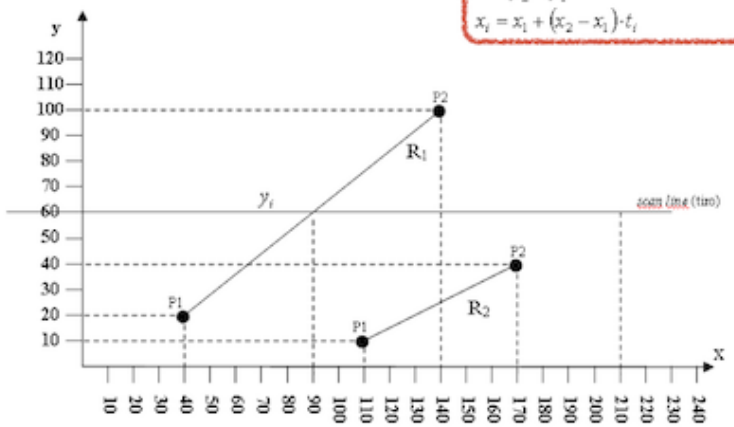
Se: $y_i = y_1 + (y_2 - y_1)\cdot t_i$ então: $t_i = \dfrac{y_i - y_1}{y_2 - y_1}$ $(t_i \in [0 \cdot\cdot 1]$ existe intersecção)

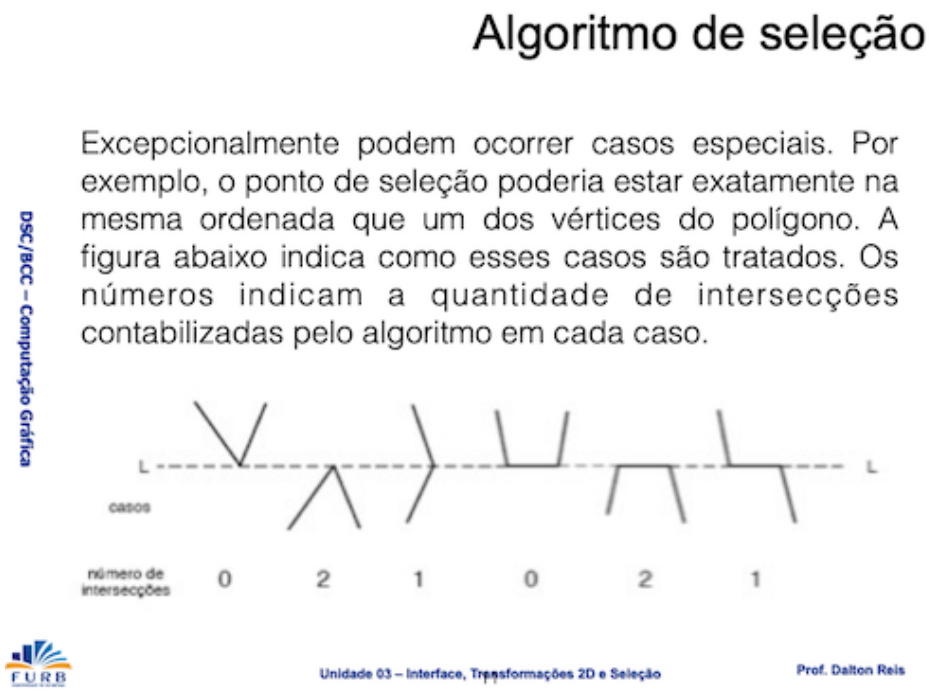
$x_i = x_1 + (x_2 - x_1)\cdot t_i$ então: $x_i = x_1 + (x_2 - x_1)\cdot \dfrac{(y_i - y_1)}{(y_2 - y_1)}$



$t_i = \dfrac{y_i - y_1}{y_2 - y_1}$ $(t_i \in [0 \cdot\cdot 1]$ existe intersecção)
$x_i = x_1 + (x_2 - x_1)\cdot t_i$

DSC/BCC – Computação Gráfica





[VisEdu-CG]

Grafo de Cena

Cena:

- representação abstrata de todos os componentes que são apresentados em um mundo virtual;
- todo componente que será representado no ambiente e exercerá um comportamento que afeta outros componentes deverá ser anexado à um nó de cena dentro de uma hierarquia.

Componentes:

- Objetos Gráficos;
- Iluminação;
- Câmera;

- etc.

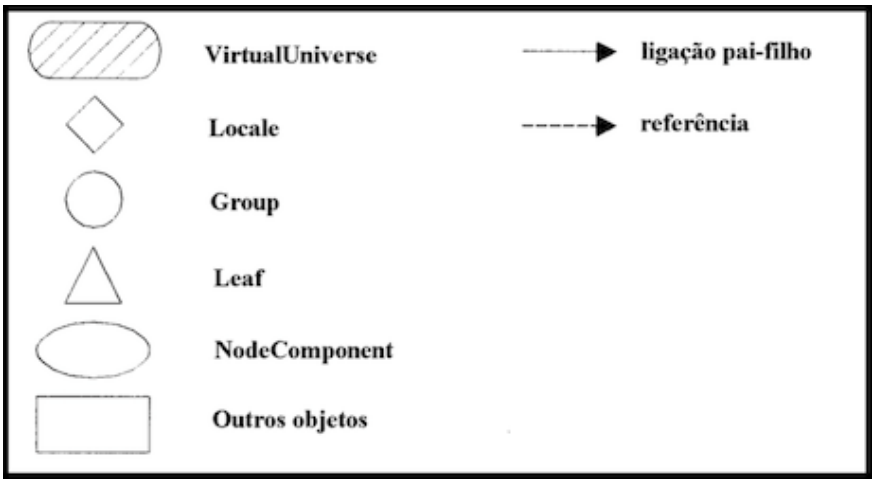
Grafo de Cena:

- estrutura composta por arcos e nós, em forma de árvore, utilizada para especificar e documentar programas que representam uma cena gráfica;
- composto por um conjunto de símbolos que representam instâncias de objetos de classes específicas.

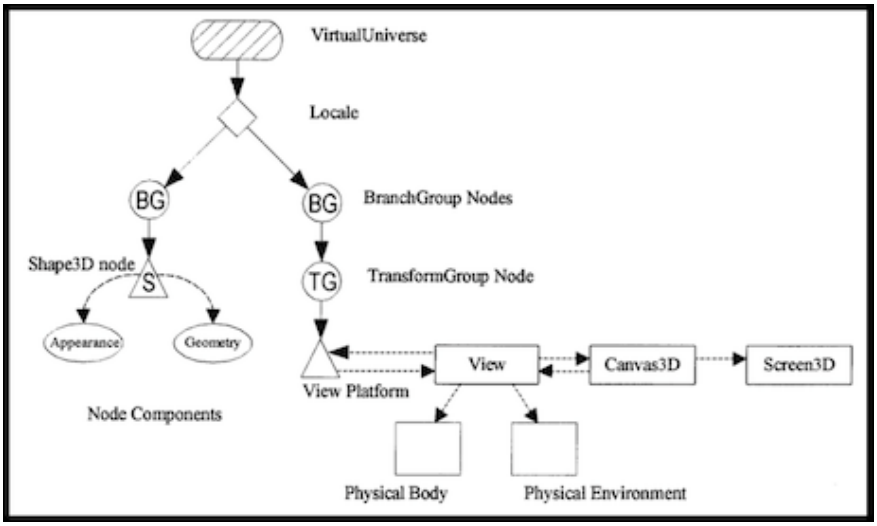
São estruturas de dados, organizadas através de classes, onde por meio de hierarquia de objetos e atributos, pode-se mais facilmente especificar cenas complexas. Cada objeto ou atributo é representado por uma classe, que possui informações sobre sua aparência física, dentre outros fatores. Um grafo de cena trata problemas que geralmente surgem em composições ou gerenciamentos de cenas (POZZER, 2007).
Popularizado pelo SGI Open Inventor, o grafo de cena protege o desenvolvedor de se preocupar com os detalhes que compõe a renderização, fazendo com que ele foque nos objetos do qual deseja renderizar, ao invés de se preocupar com a lógica da renderização em si (WALSH, 2002).

Conforme Silva, Raposo e Gattas (2004, p. 3) [1], "[...] grafos de cena são ferramentas conceituais para representação de ambientes virtuais tridimensionais nas aplicações de computação gráfica.". Isso significa que o grafo é uma espécie de mapa para a cena construída, mostrando quais objetos gráficos fazem parte dela, quais objetos possuem filhos, quais suas características (cor, textura, posicionamento etc.). Azevedo, Conci e Vasconcelos (2022, p. 183) [2] também afirmam que "[...] é comum que os objetos sejam descritos como malhas poligonais, compostas por conjuntos de vértices e arestas.". Sendo assim, objetos gráficos são formas compostas por coordenadas que são mapeadas e representadas no mundo gráfico.

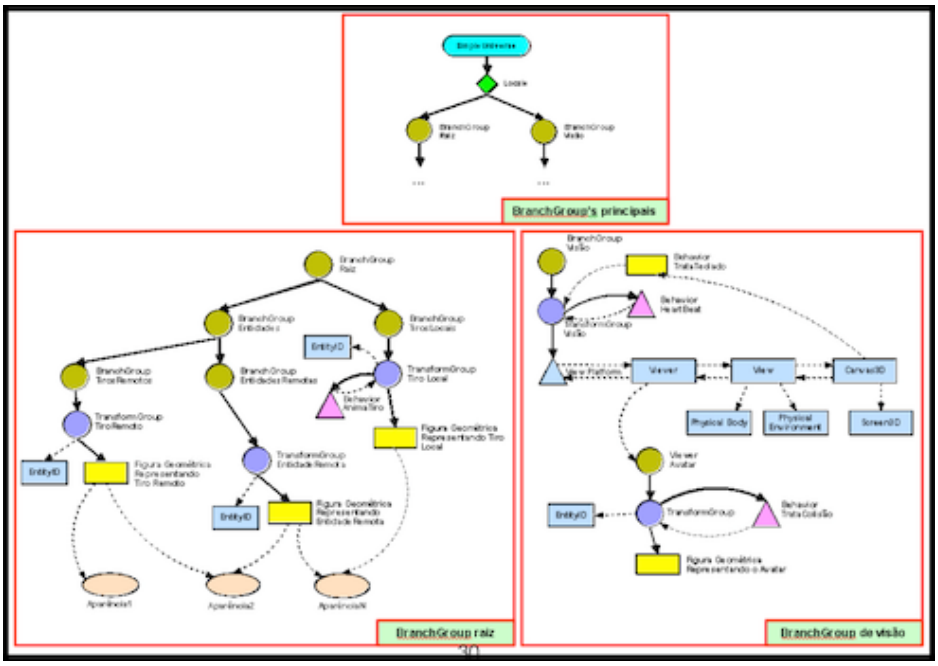
Simbologia



Formalismo



Especificação de um Ambiente Gráfico

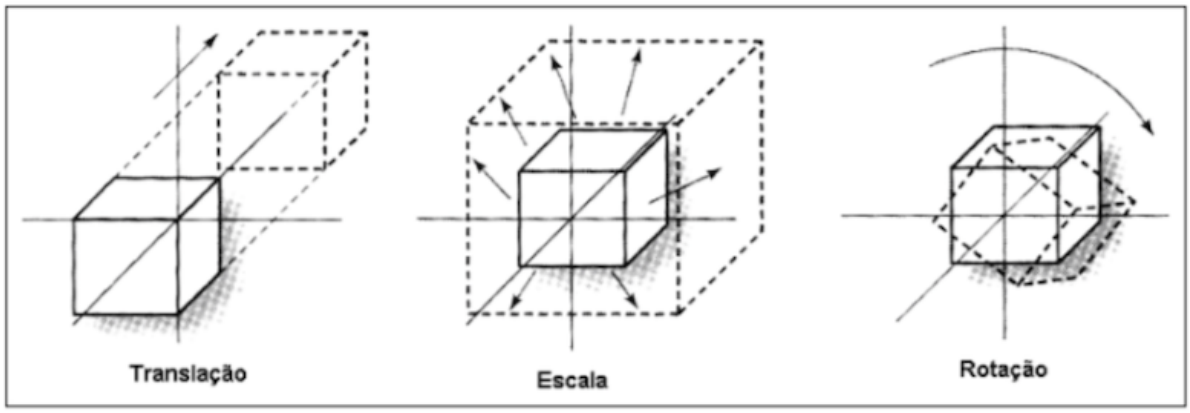


Grafo de Cena: Transformações geométricas

- OpenGL não implementa o grafo de cena
- Grafo de cena: estrutura a cena facilitando o processamento gráfico
- Estrutura básica:
    - Mundo (com lista de objetos gráficos)
    - Objeto gráfico
        - Forma: geometria e topologia
        - Aparência: cor, material, ...
        - Boundary box
        - Transformações geométricas
        - Objetos "filhos": é um novo obj. gráfico (herda transformações do pai)

Transformações Geométricas

As Transformações Geométricas (translação, escala e rotação) do Ponto Médio é a prova de que qualquer segmento de reta pode ser transladado, escalado e rotacionado pela simples Transformação de seus pontos extremos.
Observação: as transformações são sempre em relação à origem.



> *[!IMPORTANT]*
>
> *As transformações de translação, escala e rotação utilizam as operações de soma, multiplicação e seno/cosseno, respectivamente. Mas não são operações aritméticas, mas sim operações geométricas. E no caso de transformações geométricas homogêneas se utiliza **matrizes** para fazer estas operações. A matriz tem a dimensão do espaço gráfico (x,y,z) mais um, onde este mais um é o espaço homogêneo. E para se fazer transformação (translação, escala e rotação) se **multiplica** um ponto por esta matriz.*

https://github.com/dalton-reis/disciplina_CG_2025_1/blob/main/CG_Biblioteca/Transformacao4D.cs#L11-L15

Matriz: propriedades

- se deve respeitar a regra das dimensões para poder multiplicar matrizes entre si

Exemplo de Multiplicação de Matrizes

Matriz Identidade

A matriz $identidade$ é o elemento neutro na multiplicação de matrizes.

$$\begin{bmatrix} a & e & i & m \\ b & f & j & n \\ c & g & k & o \\ d & h & l & p \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} a & e & i & m \\ b & f & j & n \\ c & g & k & o \\ d & h & l & p \end{bmatrix} \tag{1}$$

https://github.com/dalton-reis/disciplina_CG_2025_1/blob/main/CG_Biblioteca/Transformacao4D.cs#L71-L78

Translação homogênea 3D: origem

> *[!WARNING]*
> *Esta transformação é em relação a origem do sistema de referência.*

A translação de um ponto $P(x,y,z)$ no plano ocorre pela adição ás coordenadas de P dos valores de deslocamento dx, dy e dz, e por ser uma adição o elemento neutro é zero.

TRANSFORMAÇÃO	OPERAÇÃO	ELEMENTO NEUTRO
Translação	soma	0

Translação: Expressão

$$P(x,y,z) \Rightarrow P(x',y',z') \quad \begin{matrix} x+dx=x' \\ y+dy=y' \\ z+dz=z' \end{matrix} \tag{2}$$

Translação: Matriz

$$[x\ y\ z] + [dx\ dy\ dz] = [x'\ y'\ z'] \tag{3}$$

Translação: Matriz Homogênea

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \tag{4}$$

https://github.com/dalton-reis/disciplina_CG_2025_1/blob/main/CG_Biblioteca/Transformacao4D.cs#L82-L88

Escala homogênea 3D: origem

> *[!WARNING]*
> *Esta transformação é em relação a origem do sistema de referência.*

A escala de um ponto $P(x,y,z)$ no plano ocorre pela multiplicação das coordenadas de P por valores de escala sx, sy e sz, e por ser uma multiplicação o elemento neutro é um.

TRANSFORMAÇÃO	OPERAÇÃO	ELEMENTO NEUTRO
escala	multiplicação	1

Fator de escala > 1 amplia
Fator de escala no intervalo]0..1[reduz

Escala: Expressão

$$P(x,y,z) \Rightarrow P(x',y',z') \qquad \begin{matrix} x \times sx = x' \\ y \times sy = y' \\ z \times sz = z' \end{matrix} \tag{5}$$

Escala: Matriz

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \times \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & sz \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \tag{6}$$

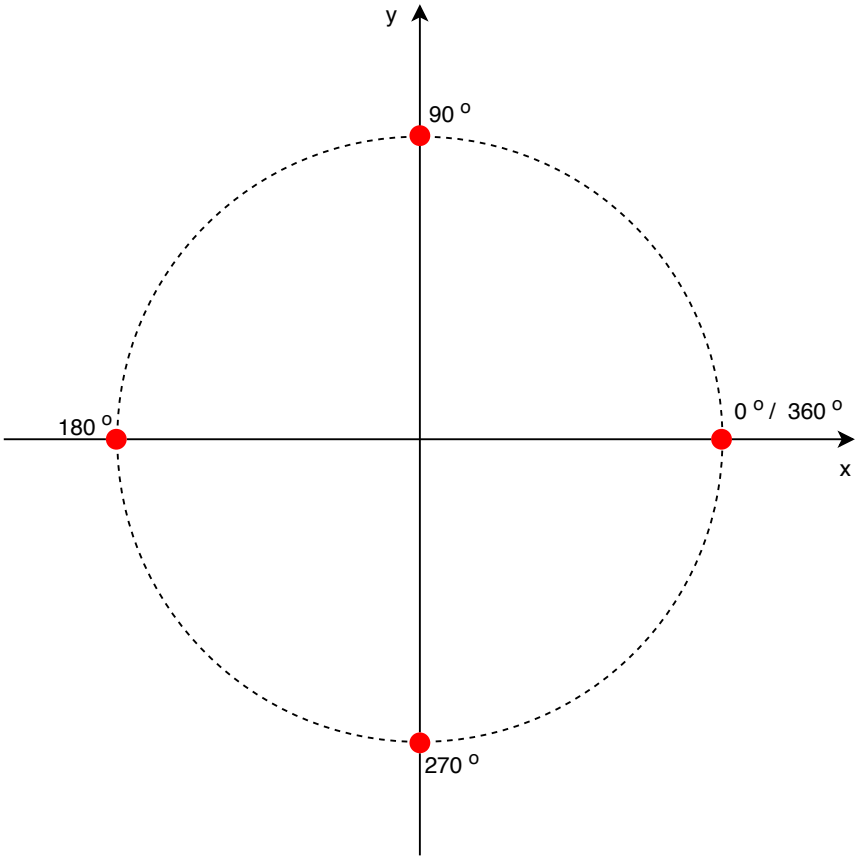
Escala: Matriz Homogênea

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \times \begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \tag{7}$$

https://github.com/dalton-reis/disciplina_CG_2025_1/blob/main/CG_Biblioteca/Transformacao4D.cs#L93-L99

Rotação homogênea 3D: origem

Sentido anti-horário:



> *[!WARNING]*
> *Esta transformação é em relação a origem do sistema de referência.*

A Rotação de um ponto $P(x,y,z)$ no plano ocorre em relação as dimensões do sistema de referência. No caso de um sistema de referência 3D as rotações serão em relação aos eixos X, Y e Z definidos pelo ângulo de rotação.

TRANSFORMAÇÃO	OPERAÇÃO	ELEMENTO NEUTRO
Rotação	seno / cosseno	0° ou múltiplo de 360°

Como chegar na matriz de Rotação

Rotação eixo X: Matriz Homogênea

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\alpha & -sin\alpha & 0 \\ 0 & sin\alpha & cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \tag{8}$$

https://github.com/dalton-reis/disciplina_CG_2025_1/blob/main/CG_Biblioteca/Transformacao4D.cs#L103-L110

Rotação eixo Y: Matriz Homogênea

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \times \begin{bmatrix} cos\alpha & 0 & -sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ sin\alpha & 0 & cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \tag{9}$$

https://github.com/dalton-reis/disciplina_CG_2025_1/blob/main/CG_Biblioteca/Transformacao4D.cs#L114-L121

Rotação eixo Z: Matriz Homogênea

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \times \begin{bmatrix} cos\alpha & -sin\alpha & 0 & 0 \\ sin\alpha & cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \tag{10}$$

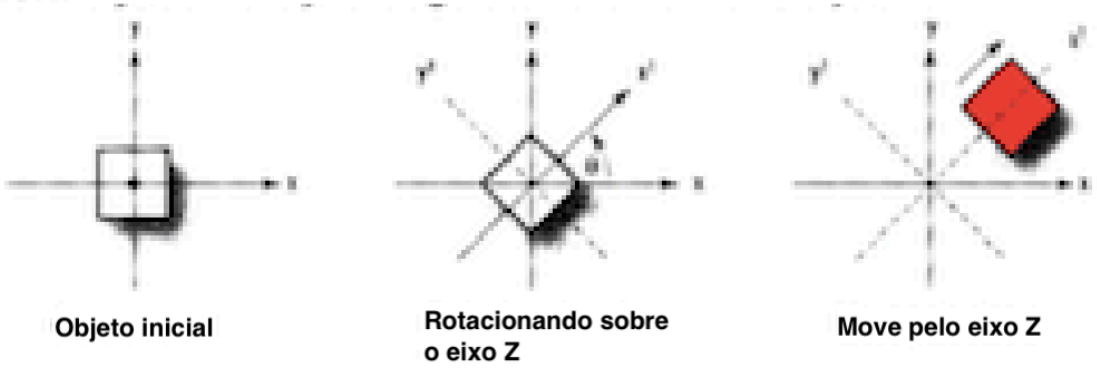
https://github.com/dalton-reis/disciplina_CG_2025_1/blob/main/CG_Biblioteca/Transformacao4D.cs#L125-L132

Composição de Transformações Geométricas

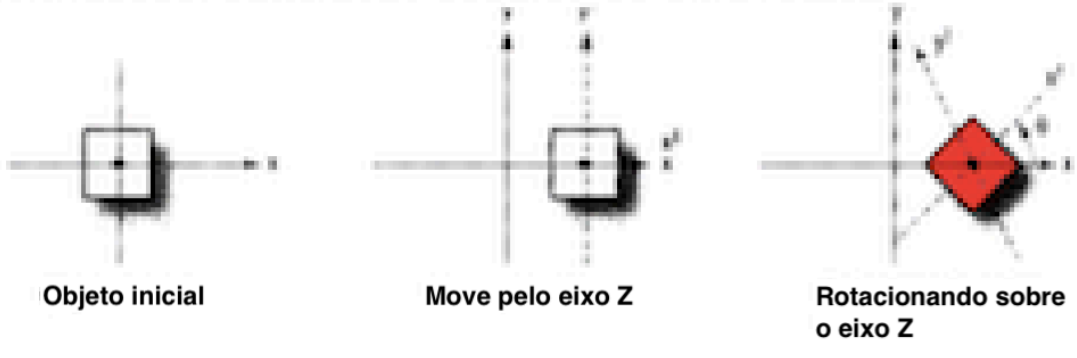
As transformações de translação, escala e rotação são **independentes** umas das outras. Mas se pode multiplicar matrizes de transformações entre si para **acumular** transformações, e assim, reduzir expressivamente o custo das transformações de pontos de uma cena. Pois em vez de se multiplicar os pontos de uma cena por uma sequência de matrizes de transformações, se multiplica estas matrizes entre si gerando uma **matriz de transformação global**, e se multiplica os pontos de uma cena por esta matriz para se ter as transformações nos objetos.

- a multiplicação de matrizes é associativa mas não comutativa

Ordem das transformações: rotação seguida de uma translação



Ordem das transformações: translação seguida de uma rotação



Observe que o resultado final é diferente, ou seja, a ordem das transformações interfere no resultado após uma sequência de transformações. Caso "clássico é a transformação de escala ou rotação a um ponto fixo, onde se usa a sequência:

- translação para origem em relação ao ponto que se quer fixar a transformação;
- a transformação, no caso, escala ou rotação;
- translação inversa da origem para o ponto que se fixou a transformação.

A multiplicação de diferentes matrizes de transformação, entre si, geram a concatenação de todas as modificações em uma única estrutura, que é chamada de matriz de modelação-visualização. Ela é responsável por determinar dentro de um contexto, as posições e modificações dos objetos 3D de uma cena.

⏭ Unidade 4)

<!--

Download

Nesta pasta tem um executável da atividade 3 que pode ser usado para ajudar a entender o enunciado desta atividade.
CG-N3_Executavel

Dicas de código

Na Classe Mundo.cs adicione *ToolkitOptions*

```
  class Program
  {
    static void Main(string[] args)
    {
      ToolkitOptions.Default.EnableHighResolution = false;
      Mundo window = Mundo.GetInstance(600, 600);
      window.Title = "CG_N3";
      window.Run(1.0 / 60.0);
    }
  }
}
```

Já na Classe Objeto.cs o método *Desenhar()*

```
    public void Desenhar()
    {
#if CG_OpenGL
      GL.PushMatrix();                                    // N3-
Exe12: grafo de cena
      GL.MultMatrix(matriz.ObterDados());
      GL.Color3(objetoCor.CorR, objetoCor.CorG, objetoCor.CorB);
      GL.LineWidth(primitivaTamanho);
      GL.PointSize(primitivaTamanho);
#endif
      DesenharGeometria();
      for (var i = 0; i < objetosLista.Count; i++)
      {
        objetosLista[i].Desenhar();
      }
      GL.PopMatrix();                                     // N3-
Exe12: grafo de cena
    }
```

E ainda, na Classe Objeto.cs um exemplo de método para rotação em relação a origem do SRU

```
    public void Rotacao(double angulo)
    {
      RotacaoEixo(angulo);
      matriz = matrizTmpRotacao.MultiplicarMatriz(matriz);
    }
```

Já, na Classe Objeto.cs, um exemplo de método para rotação em torno de um ponto

```
    public void RotacaoEixo(double angulo)
    {
      switch (eixoRotacao)
      {
        case 'x':

 matrizTmpRotacao.AtribuirRotacaoX(Transformacao4D.DEG_TO_RAD *
angulo);
          break;
        case 'y':

 matrizTmpRotacao.AtribuirRotacaoY(Transformacao4D.DEG_TO_RAD *
angulo);
          break;
        case 'z':

 matrizTmpRotacao.AtribuirRotacaoZ(Transformacao4D.DEG_TO_RAD *
angulo);
          break;
      }
    }
```

```
    public void RotacaoZBBox(double angulo)
    {
      matrizGlobal.AtribuirIdentidade();
      Ponto4D pontoPivo = bBox.obterCentro;
```

```
      matrizTmpTranslacao.AtribuirTranslacao(-pontoPivo.X, -
pontoPivo.Y, -pontoPivo.Z); // Inverter sinal
      matrizGlobal =
matrizTmpTranslacao.MultiplicarMatriz(matrizGlobal);

      RotacaoEixo(angulo);
      matrizGlobal =
matrizTmpRotacao.MultiplicarMatriz(matrizGlobal);

      matrizTmpTranslacaoInversa.AtribuirTranslacao(pontoPivo.X,
pontoPivo.Y, pontoPivo.Z);
      matrizGlobal =
matrizTmpTranslacaoInversa.MultiplicarMatriz(matrizGlobal);

      matriz = matriz.MultiplicarMatriz(matrizGlobal);
    }
```

-->

---

1. SILVA, Romano J. M. da; RAPOSO, Alberto B.; GATTAS, Marcelo. Grafo de Cena e Realidade Virtual. Rio de Janeiro: PUC, 2004. Disponível em: https://web.tecgraf.puc-rio.br/~abraposo/INF1366/2007/02_GrafoDeCena_texto.pdf. Acesso em: 27 nov. 2023. ↩

2. AZEVEDO, Eduardo; CONCI, Aura; VASCONCELOS, Cristina. Computação Gráfica: Teoria e Prática: Geração de Imagens. 1. ed. rev. Rio de Janeiro: Alta Books, 2022. ↩