OpenGL: Transformações Geométricas e Viewing

Transformações Geométricas nos espaços 2D e 3D

P.A.E. Eric Macedo Cabral[1] Profa. Rosane Minghim[2]

[1]cabral.eric@usp.br, [2]rminghim@icmc.usp.br

Instituto de Computação e Matemática Computacional (ICMC)
Universidade de São Paulo (USP)

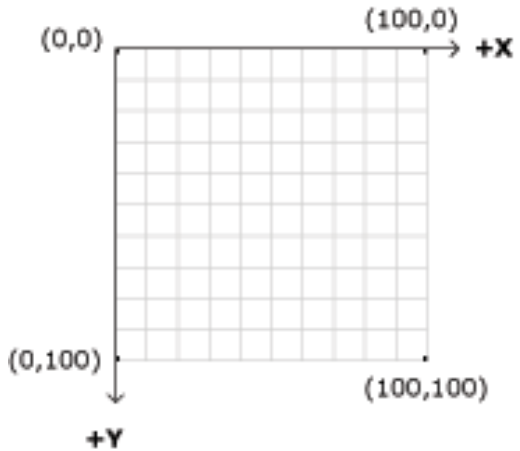
27 de Setembro de 2018

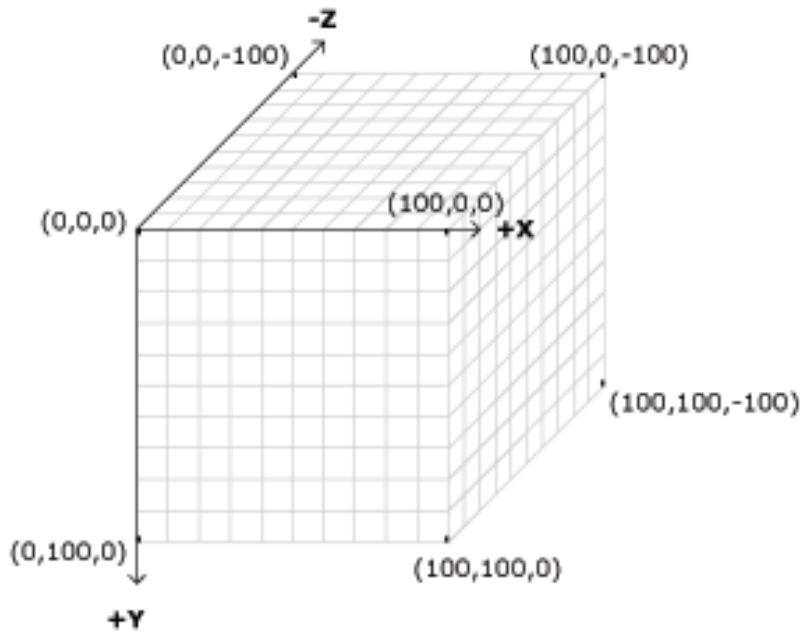
Sumário

Roteiro

Ambiente Visual

- No monitor do computador é adotado o SRT (Sistema de Referência da Tela).
  - A origem fica no canto superior esquerdo do monitor.





Ambiente Visual 2D I

Janela de Seleção (*Window*)

É a área que define a porção do universo que desejamos mapear na tela.

Método OpenGL que define a janela de seleção:

```
void gluOrtho2D(GLdouble left, GLdouble right, GLdouble bottom, ↩
    GLdouble top)
```

- **left:** Limite inferior do eixo X.
- **right:** Limite superior do eixo X.
- **bottom:** Limite inferior do eixo Y.
- **top:** Limite superior do eixo Y.

Ambiente Visual 2D II

Janela de Exibição (*Viewport*)

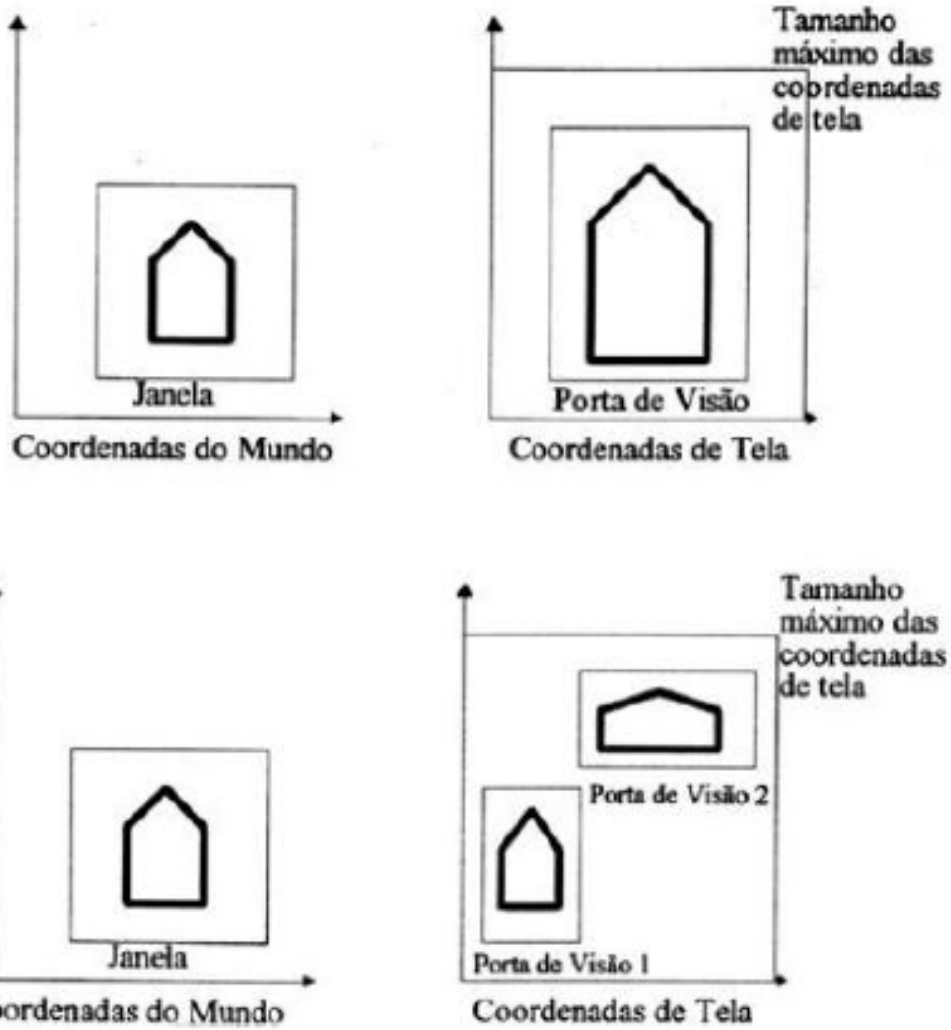
É a área do monitor que deseja-se exibir o conteúdo da janela de seleção (*Window*).

Método OpenGL que define a janela de exibição:

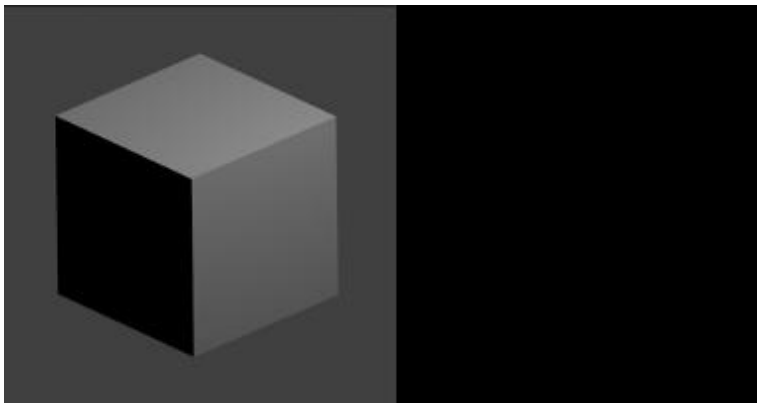
```
void glViewport(GLint x, GLint y, GLsizei width, GLsizei height)
```

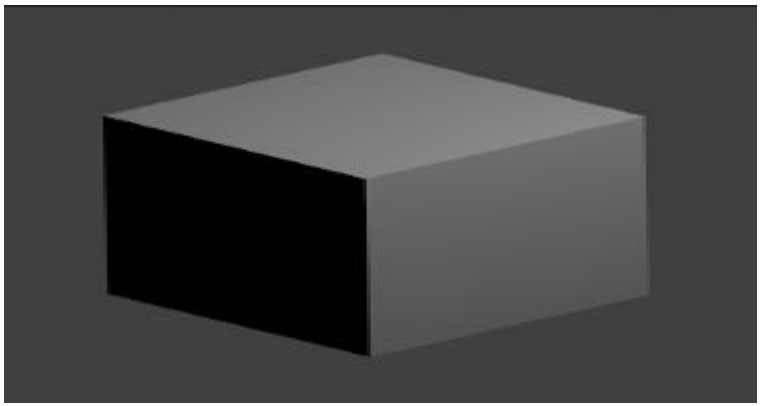
- **x:** Posição no eixo x referente à coordenadas da tela.
- **y:** Posição no eixo y referente à coordenadas da tela..
- **width:** Proporção de largura da janela de exibição.
- **height:** Proporção de altura da janela de exibição.

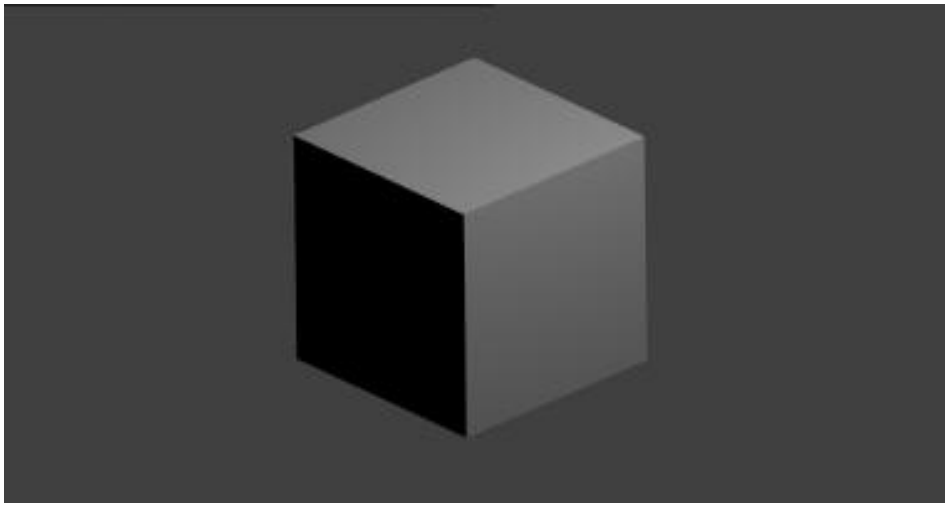
Ambiente Visual 2D III



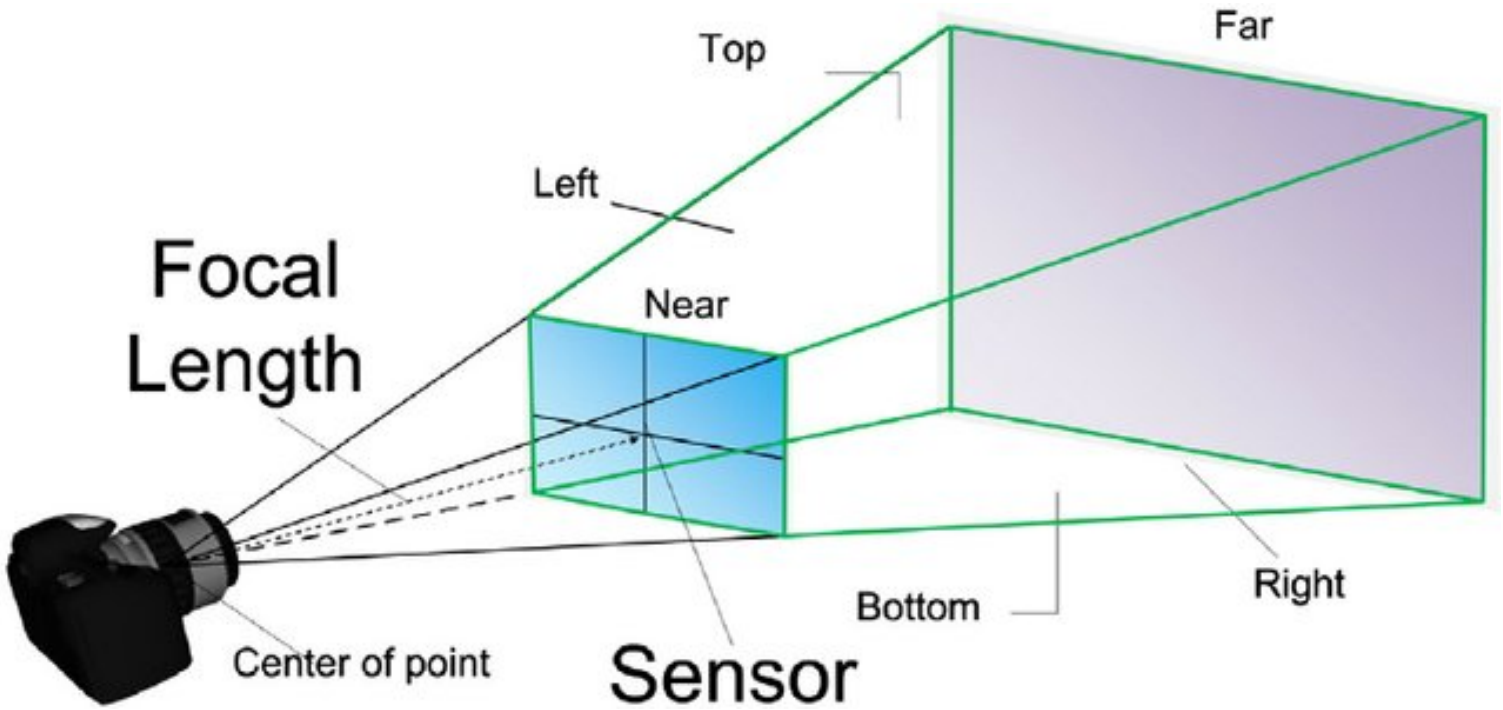
Ambiente Visual 2D IV







Ambiente Visual 3D I



Ambiente Visual 3D II

A GLU oferece a seguinte função para posicionar e orientar a câmera:

```
void gluLookAt(GLdouble obsx, obsy, obsz, alvox, alvoy, alvoz, upx, upy↩
    , upz)
```

- **obs*:** define a posição da câmera.
- **alvo*:** ponto para onde o observador está olhando.
- **up*:** vetor que indica a 'vertical' da câmera.

Ambiente Visual 3D III

Desenho 3D

- Em 3D é necessário especificar também a coordenada Z.
- São utilizadas as mesmas primitivas gráficas apresentadas em 2D:
  - GL_POINTS, GL_LINES, GL_LINE_LOOP, GL_QUADS, GL_TRIANGLES, etc.

Remoção de superfícies escondidas

- Ao renderizar objetos sólidos, devemos ter o cuidado de exibir apenas as faces que realmente devem aparecer na cena.
- Não devemos exibir as faces escondidas, localizadas atrás de outros objetos.

Ambiente Visual 3D IV

Remoção de superfícies escondidas

- A OpenGL implementa um algoritmo simples e eficiente para remoção de faces ocultas: **Z-Buffering**.
- Utiliza um buffer para cor (Color Buffer) e outro para a profundidade (Depth Buffer).
- Objetos mais próximos da câmera virtual definem a cor dos pixels da cena.

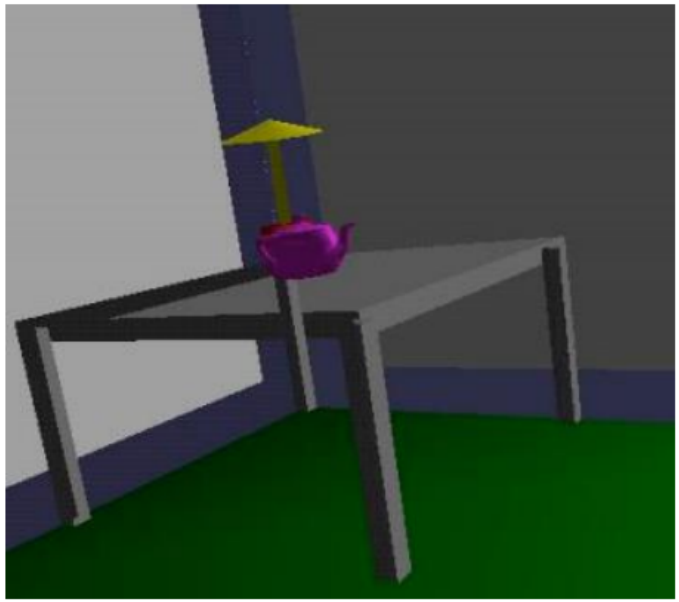
Para indicar o uso do buffer de profundidade:

```
glutInitDisplayMode( ... | GLUT_DEPTH);
```

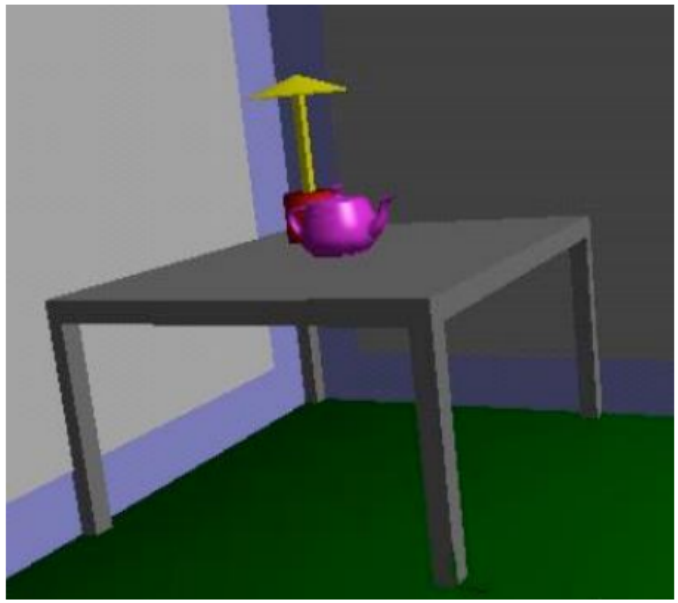
Para habilitar o uso do buffer:

```
glEnable(DL_DEPTH_TEST);
```

Ambiente Visual 3D V



Visibilidade incorreta



Visibilidade correta

Fonte: R. W. Linderman, http://web.cs.wpi.edu/ gogo/courses/cs543/

Roteiro

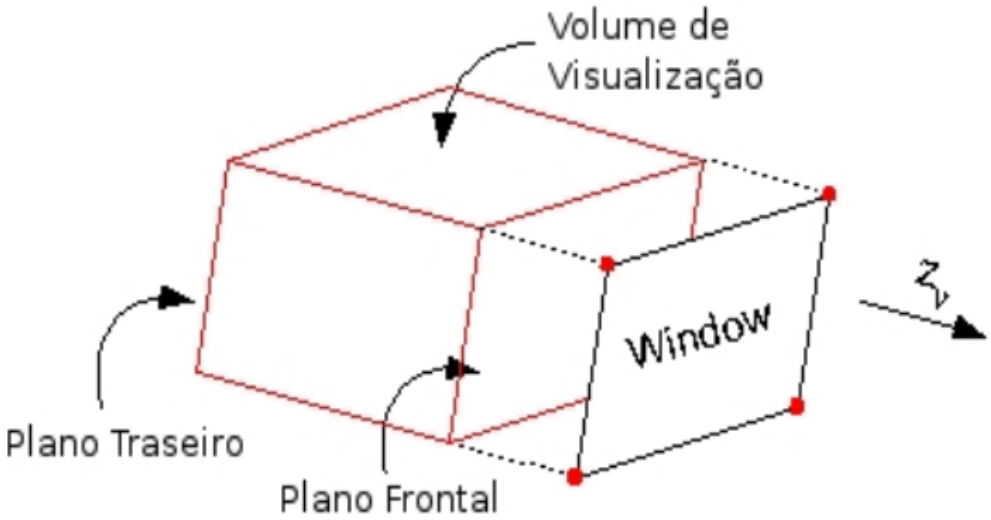
2. Projeções
   - Paralela
   - Perspectiva

Projeções I

- Obtém representações bidimensionais de objetos tridimensionais.
- A projeção é definida por raios de projeção (projetantes) que passam através de cada vértice dos objetos e interceptam o plano de projeção.
- Dois tipos de projeções:
  - Paralela Ortográfica.
  - Perspectiva.

Projeção Paralela I

- As projetantes são paralelas entre si.
- Não há alteração nas medidas do objeto.




```
void gluOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble ↩
    top, GLdouble near, GLdouble far);
```

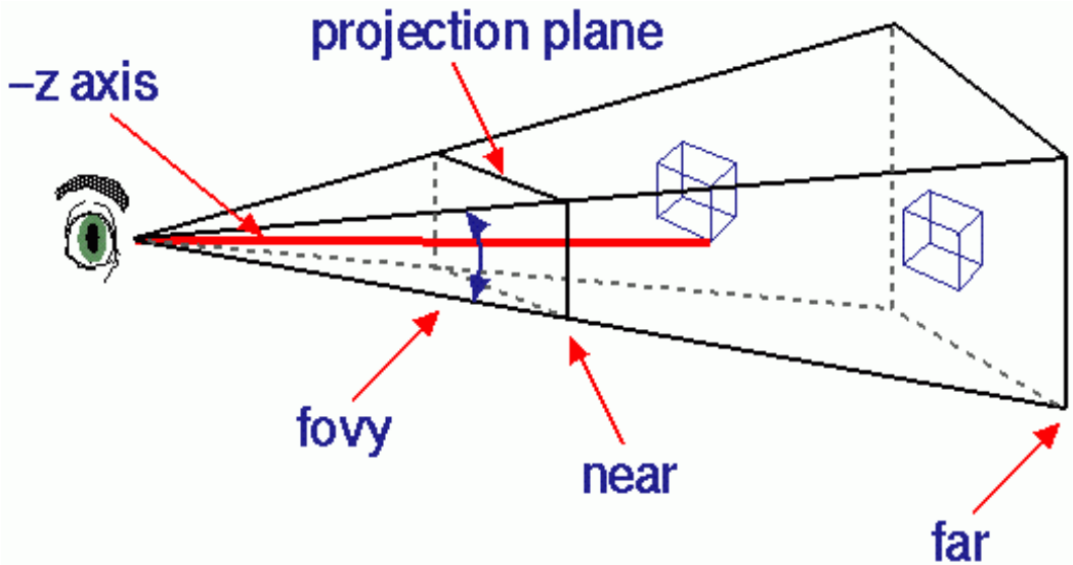
Projeção Paralela II

Exemplo de Projeção Paralela:

```
void Desenha(){
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(-65.0, 65.0, -65.0, 65.0, -400.0, 400.0);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluLookAt(40,60,100, 0,0,0, 0,1,0);
    glClearColor(1.0f, 1.0f, 1.0f, 1.0f);
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0f, 0.0f, 0.0f);
    glutWireCube(50);
    glFlush();
}
```

Perspectiva I

As projetantes emanam de um único ponto, a uma distância finita do plano de projeção.



Perspectiva II

```
void gluPerspective(GLdouble fovy, GLdouble aspect, GLdouble zNear, ↩
    GLdouble zFar);
```

- **fovy:** ângulo de abertura da câmera em y.
- **aspect:** aspecto de visualização em x.
- **zNear:** distância do observador ao plano de corte frontal.
- **zFar:** distância do observador ao plano de corte traseiro.

Perspectiva III

Exemplo de Projeção Perspectiva:

```
void Desenha(){
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(60, 1.0, 0.5, 500);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluLookAt(40, 60, 100, 0, 0, 0, 0, 1, 0);
    glClearColor(1.0f, 1.0f, 1.0f, 1.0f);
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0f, 0.0f, 1.0f);
    glutWireCube(50);
    glFlush();
}
```

Roteiro

3. Transformações Geométricas
   - Transformações Básicas
   - Transformações Hierárquicas

Transformações Geométricas I

- O objeto transformado deve inicialmente ser transladado para a origem.
- Então deve ser aplicada a transformação.
- Por fim o objeto deve ser transladado novamente para sua posição inicial.

Transformações Geométricas II

Matriz de Transformação

- Todos os comandos de transformação são compostos em uma matriz de transformação.
- Cada novo comando é acumulado, alterando a configuração da matriz.
- Ao especificar um novo vértice, a sua posição é calculada aplicando-se a matriz de transformação corrente às suas coordenadas.

A matriz de transformação é inicializada com a matriz identidade:

```
glLoadIdentity();
```

- Não altera os objetos

$$\begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix}$$

Transformações Geométricas III

Translação:

```
void glTranslatef(GLfloat x, GLfloat y, GLfloat z);
```

- x, y, z: representam a quantidade a ser transladada no respectivo eixo.
- Em 2D: $z = 0$.

Transformações Geométricas IV

Rotação:

```
void glRotatef(GLfloat angulo, GLfloat x, GLfloat y, GLfloat z);
```

- Angulo: especifica o ângulo de rotação (em graus).
- x, y, z: o eixo a ser realizada a rotação.
- Para o caso 2D: $x = 0$, $y = 0$, $z = 1$.

Transformações Geométricas V

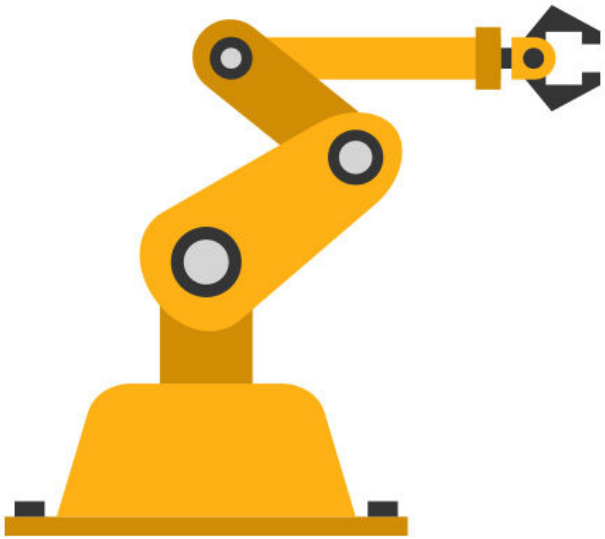
Escala:

```
void glScalef(GLfloat x, GLfloat y, GLfloat z);
```

- x, y, z: Representam o fator de escala nos respectivos eixos.
- Em 2D: $z = 0$.

Transformações Hierárquicas I

- São transformações diferentes aplicadas em objetos que seguem uma hierarquia.
- As transformações de nível mais baixo na hierarquia acumulam as transformações de níveis hierárquicos mais altos.



Transformações Hierárquicas II

- Para aplicar transformações hierárquicas em diversos objetos, é possível salvar a matriz de transformação corrente em uma pilha.
- Cada objeto transformado utiliza sua própria versão da matriz de transformação, que é empilhada desempilhada conforme a necessidade.

Transformações Hierárquicas III

```
glPushMatrix();
```

- Empilha a matriz de transformação atual.
- Não altera a matriz atual.

```
glPopMatrix();
```

- Retira a matriz do topo da pilha.
- A matriz atual é substituída pela matriz desempilhada.

Próxima Aula

Exercício:

- Fazer uma aplicação OpenGL que posicione um observador na origem do R^3 e que desenhe um objeto diferente em cada quadrante. Além disso, a aplicação deve permitir uma mudança na direção de visão do observador por meio das teclas direcionais.

Dúvidas

- **Horário de atendimento:** Quarta-feira, das 14 às 16h.
  - Laboratório VICG (Bloco 1, sala 007)
  - Enviar email com antecedência.
- **Email :** cabral.eric@usp.br
  - **Subject :** [CG2018_2]

Bibliografia

- **Básica:**
  - Hearn, D. Baker, M. P. Computer Graphics with OpenGL, Prentice Hall, 2004. **(livro texto)**
  - Neider, J. Davis, T. Woo, M. OpenGL programming guide, 2007. (**livro base para aulas práticas**)
  - Angel, E. Interactive computer graphics: a top-down approach with OpenGL, Addison Wesley, 2000.
  - Foley, J. et. al - Introduction to Computer Graphics, Addison-Wesley, 1993.
  - Kessenich, J., Sellers, G., Shreiner, D. OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.5 with SPIR-V - Ninth Edition.

Bibliografia

- **Complementar:**
  - Computer Graphics Comes of Age: An Interview with Andries van Dam. CACM, vol. 27, no. 7. 1982
  - The RenderMan – And the Oscar Goes to... IEEE Spectrum, vol. 38, no. 4, abril de 2001.
  - Material do ano passado: `https://sites.google.com/site/computacaograficaicmc2017t2/`
  - Apostilas antigas da disciplina Computação Gráfica
    - `http://www.gbdi.icmc.usp.br/material?q=system/files/apostilas.pdf`