CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CAMPUS VII - UNIDADE TIMÓTEO
Curso: Engenharia de Computação
Disciplina: Computação Gráfica
Professor: Odilon Corrêa

OPENGL – PRÁTICA 06

Visualização Tridimensional

Na visualização em três dimensões o SRU (Sistema de Referência do Universo) passa a ser composto por três eixos ortogonais (Figura 01) entre si (***x***, ***y*** e ***z***) e pela origem (0.0, 0.0, 0.0).

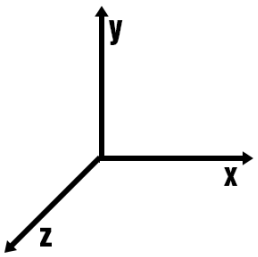

Figura 01 – Sistema de referência do universo 3D utilizado em OpenGL

Para identificar como o eixo ***z*** é posicionado em relação à ***x*** e ***y***, podemos utilizar a regra da "mão direita" ou "mão esquerda". O padrão no OpenGL é utilizar a regra da "mão direita" (Figura 02).

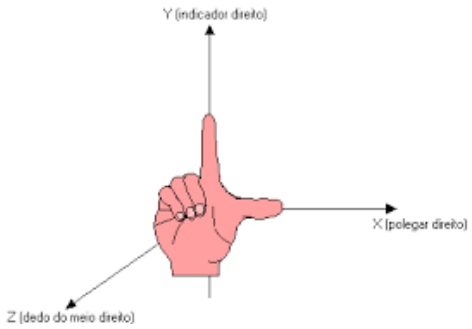

Figura 02 – Regra da mão direita para orientação dos eixos em 3D

O processo de visualização 3D é mais complexo, pois compreende um número maior de etapas. Essa complexidade existe porque quase todos os dispositivos de saída, tais como monitores, são 2D. Assim é preciso definir como a cena 3D será visualizada e projetada em uma imagem 2D.

A primeira etapa do processo de visualização 3D é a definição da cena 3D. Nesta etapa cada um dos objetos que farão parte do mundo 3D é incluído e posicionado no SRU. Esse posicionamento é feito por meio de operações de escala, rotação e translação.

O próximo passo consiste na especificação do observador virtual. Que é um componente fundamental do processo de visualização 3D. Este observador define de que local se deseja que a cena 3D seja exibida, por exemplo, de cima ou do lado direito. A especificação do observador inclui a sua posição e orientação, ou seja, onde o observador está e para onde ele está olhando dentro do universo. A imagem gerada a partir da posição e orientação do observador é estática e pode ser comparada a uma foto. A posição da câmera é dada por um ponto (***x***, ***y*** e ***z***) em relação ao mesmo universo no qual os objetos estão posicionados (SRU), e sua orientação é dada por um ponto-alvo (***x***, ***y*** e ***z***) e um vetor, aqui chamado de ***up***. A Figura 03 ilustra estes conceitos ao posicionar a câmera de maneira diferentes e ocasionando a geração de duas imagens distintas. Na Figura 03.a, os objetos aparecem da mesma maneira que estão posicionados. E na Figura 03.b os objetos aparecem inclinados 90º para esquerda.

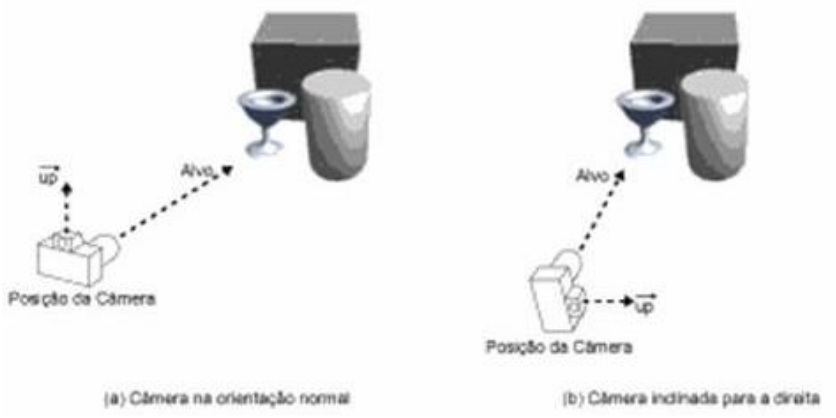

Figura 03 – Modelo de câmera sintética utilizada em OpenGL

A operação de obter representações bidimensionais de objetos tridimensionais é chamada de projeção. Objetos 3D, sem geral são representados por uma coleção de pontos (vértices). A projeção desses objetos é definida por raios de projeção (segmentos de retas) chamados de projetantes e que passam através de cada vértice do objeto e interseccionam um plano de projeção. Esta classe de projeções é conhecida como projeções geométricas planares. As projeções são usualmente divididas em dois tipos:

- **Projeção Paralela Ortográfica**: as propriedades são paralelas entre si e passam pelos pontos de definem os objetos e interseccionam o plano com um ângulo de 90º (Figura 04.a).
- **Projeção Perspectiva**: as projetantes emanam de um único ponto que está a uma distância finita do plano de projeção e pelos pontos de definem os objetos (Figura 04.b).

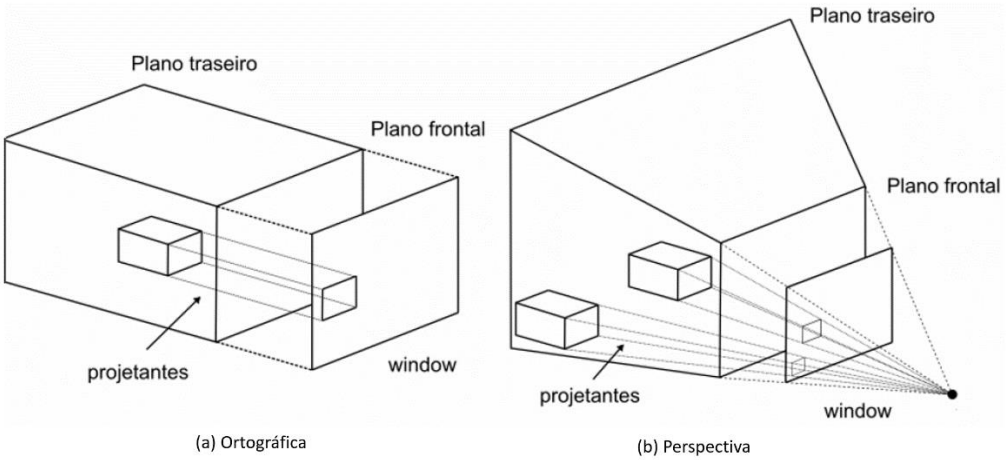

Figura 04 – Projeções de paralelepípedos usando a projeção paralela ortográfica (a) e a projeção perspectiva (b)

Na visualização 2D, a *window* é usada para determinar a porção do universo que será visualizada. Quando se está trabalhando em 3D também se considera o conteúdo da *window* para fazer o mapeamento para a *viewport*. Entretanto, primeiro é preciso definir um volume de visualização, ou seja, a região exata do universo 3D que será mapeado para a *viewport*. O tamanho é a forma do volume de visualização dependem do tipo de projeção e da *window*, visto que os seus lados são planos que passam através das bordas da *window*.

Para a projeção paralela ortográfica, os quatro lados do volume de visualização e os planos frontal e traseiro na direção do eixo z, formam um paralelepípedo. Para a projeção perspectiva o volume de visualizaçãoé um tronco de pirâmide, limitado pelos planos frontal e traseiro, cujo topo é o centro de projeção, como ilustrado na Figura 04.b. Os planos frontal e traseiro do volume de visualização são paralelos ao plano de projeção, e por meio da sua inclusão obtém-se um volume de visualização limite por seis planos, permitindo excluir parte da cena de acordo com a profundidade.

A biblioteca GLU oferece uma função específica para isso, denominada de **gluLookAt**, que permite especificar a posição e orientação da câmera.

- **gluLookAt(GLdouble eyex, GLdouble eyey, GLdouble eyez, GLdouble centerx, GLdouble centery, GLdouble centerz, GLdouble upx, GLdouble upy, GLdouble upz);**

  Seus parâmetros são: ***eyex***, ***eyey*** e ***eyez*** são usados para definir as coordenadas ***x***, ***y*** e ***z***, respectivamente, da posição da câmera (ou observador); ***centerx***, ***centery*** e ***centerz*** são usados para definir as coordenadas ***x***, ***y*** e ***z***, respectivamente, da posição do alvo, isto é, para onde o observador está olhando (normalmente, o centro da cena); ***upx***, ***upy*** e ***upz*** são as coordenadas ***x***, ***y*** e ***z,*** que estabelecem o vetor ***up*** (indica o "lado de cima" de uma cena 3D).

Para utilizar a projeção perspectiva é preciso utilizar a função **gluPerspective**:

- **gluPerspective(GLdouble fovy, GLdouble aspect, GLdouble zNear, GLdouble zFar);**

  Seus parâmetros são: ***fovy*** é o ângulo, em graus, na direção ***y*** (usada para determinar a "altura" do volume de visualização); ***aspect*** é a razão de aspecto que determina a área de visualização na direção ***x***, e seu valor é a razão em ***x*** (largura) e ***y*** (altura); ***zNear***, que sempre deve ter um valor positivo maior do que zero, é a distância do observador até o plano de corte mais próximo (em ***z***); ***zFar***, que também sempre tem um valor positivo maior do que zero, é a distância do observador até o plano de corte mais afastado (em ***z***). Esta função sempre deve ser definida ANTES da função ***gluLookAt***, e no modo ***GL_PROJECTION***.

O código do **Projeto01** (Figura 05) ilustra a utilização dessas funções, desenhando um cubo com projeção perspectiva. O cubo é um objeto predefinido da biblioteca GLUT.

Se for necessário trabalhar com a projeção ortográfica, basta configurar utilizando a função **glOrtho**.

- **glOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble zNear, GLdouble zFar);**

  Seus parâmetros são: ***left*** e ***right*** especificam os limites mínimos e máximo no eixo ***x***; analogamente, **bottom** e **top** especificam os limites mínimo e máximo no eixo ***y***, enquanto ***zNear*** e ***zFar*** especificam os limites mínimo e máximo no eixo ***z***, geralmente com os valores negativos para o lado oposto da posição do observador.

O código do **Projeto02** (Figura 06) ilustra a utilização da projeção paralela ortográfica. A chamada da função **glOrtho** pode ser observada na função **EspecificaParametrosVisualizacao**.

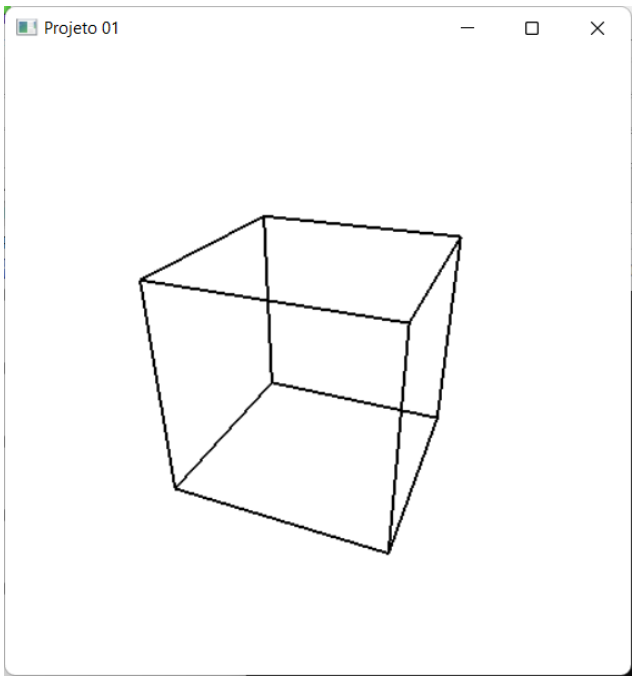

Figura 05 - Projeto 01

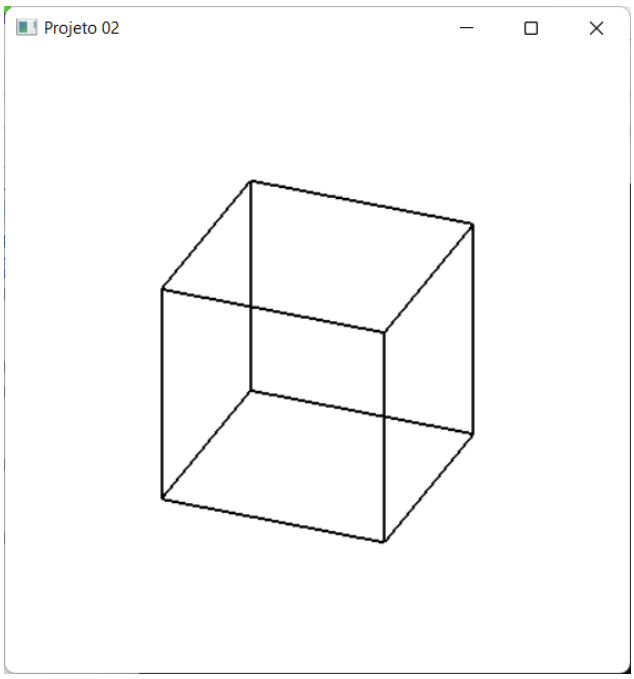

Figura 06 - Projeto 01

Exercício

Faça as modificações necessárias no código do Projeto01 e insira uma animação de rotação no objeto.

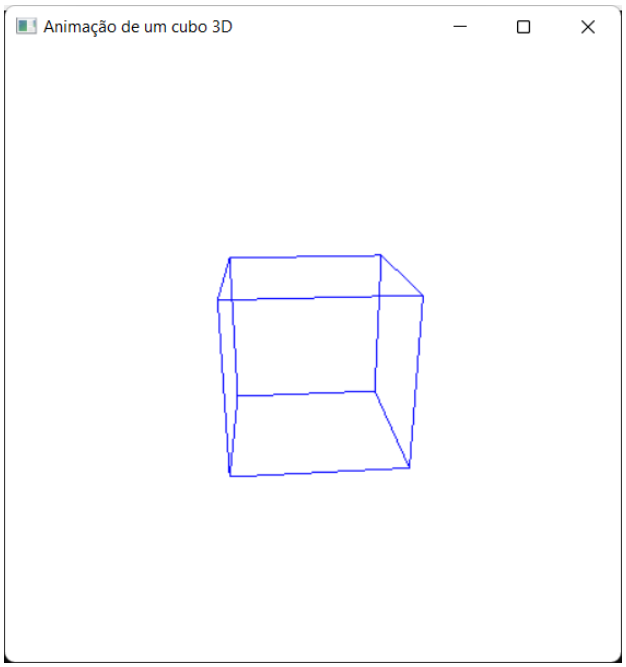

Figura 07 - Exercício

Observação

O arquivo executável (Exercicio.exe) do projeto foi disponibilidade para auxiliar na visualização e compreensão do funcionamento do exercício.

Referências

Todas as informações descritas neste roteiro foram retiradas do livro "OpenGL: Uma abordagem prática e objetiva", presente no plano didático da disciplina, e do site da Profª. Isabel Harb Manssour (https://www.inf.pucrs.br/~manssour/OpenGL/Tutorial.html)