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

OPENGL – PRÁTICA 05

Transformações Hierárquicas

São transformações diferentes aplicadas em objetos que seguem uma hierarquia. Por exemplo, podemos aplicar diferentes transformações sobre partes do corpo humano. Rotacionar o corpo, transladar o braço, entre outras. As transformações de mais baixa hierarquia (braços) acumulam as transformações da hierarquia mais alta (corpo).

As transformações geométricas hierárquicas em OpenGL permite dividir os objetos em vários componentes e utilizar as funções `glPushMatrix` e `glPopMatrix` para que seja possível realizar a combinação de transformações geométricas ou aplicar uma transformação em apenas um componente do objeto.

O entendimento de escopo das transformações (**Prática 04**) é fundamental para realização das transformações hierárquicas, pois para desenhar um único objeto hierárquico pode ser necessário tanto combinar as transformações geométricas como aplicar uma transformação em apenas um componente deste objeto. No momento de combinar as transformações é muito importante definir a ordem na qual os componentes de um objeto devem ser desenhados. Utilize o **Projeto01** para compreender os conceitos apresentados.

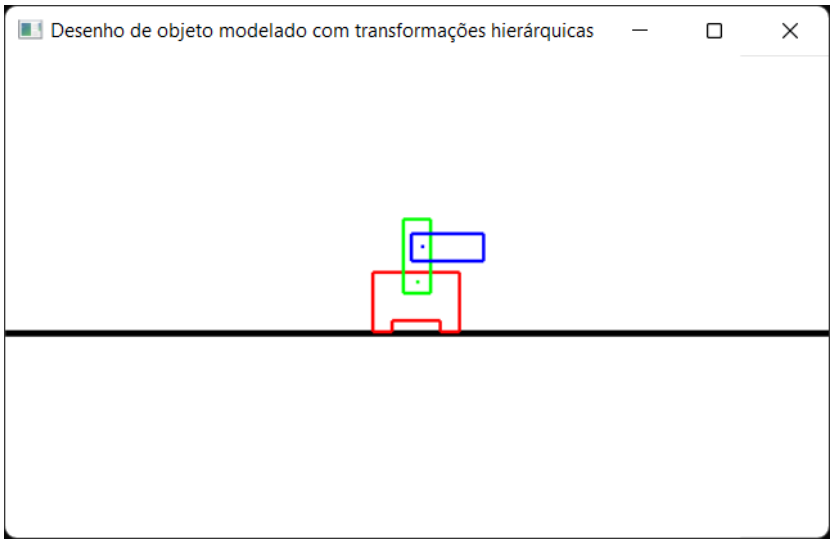

Figura 1 – Projeto 01

Comportamento do programa:

Quando a base do objeto é transladada (para esquerda ou direita) os objetos que representam os braços também acompanham a movimentação. Entretanto, se um dos braços é rotacionado a base permanece no mesmo lugar.

- **Setas** para **esquerda** e para **direita** movimentam a base (**vermelha**)
- **Home** e **End** rotacionam o braço 1 (**verde**)
- **PageUP** e **PageDn** rotacionam o braço 2 (**azul**)

Animação em 2D

A animação tradicional consiste em exibir uma sequência de imagens de forma que o olho humano perceba apenas um movimento contínuo e não as imagens individuais. A velocidade ou taxa de exibição de imagens (em inglês, frame rate) pode varia de acordo com a mídia utilizada. É importante garantir que a aparência de uma

animação não mude rapidamente em relação ao número de quadros exibidos por segundo. Para que não ocorra o que se chama de *aliasing* temporal é preciso evitar um movimento descontínuo de objetos. O que ocorre quando há um grande deslocamento durante um curto espaço de tempo.

Para implementar o efeito de animação em OpenGL é necessário redesenhar a cena de forma continua e produzir o equivalente a uma sequência de quadros. Independente da técnica utilizada é preciso uma forma para forçar a atualização contínua da tela. Isso pode ser realizado através de uma função de *callback* que é chamada em intervalos de tempo predeterminados. Entretanto, para evitar que imagem fique piscando na tela é necessário utilizar dois *buffers* de exibição.

A função `glutInitDisplayMode` avisa a GLUT que tipo de modo de exibição deve ser usado quando a janela é criada. Ao trocar o modo de exibição de GLUT_SINGLE para GLUT_DOUBLE os comandos de desenho são executados para criar uma cena fora da tela para depois rapidamente colocá-la na janela de visualização. Também é necessário trocar a chamada de função `glFlush` para `glutSwapBuffers`. A segunda função consegue armazenar em um buffer uma imagem enquanto uma outra imagem de outro buffer é exibida.

É possível criar um laço que continuamente altera as transformações geométricas aplicadas sobre os objetos antes de chamar a função responsável pelo desenho da cena. A função `glutTimerFunc` da biblioteca GLUT permite associar uma função de *callback* a ser executada de tempos em tempos. Utilize o **Projeto02** para compreender os conceitos apresentados.

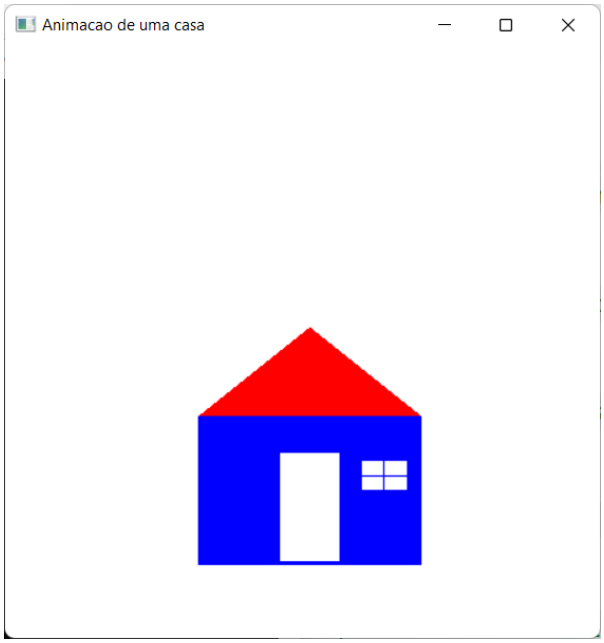

Figura 2 – Projeto 02

Observações importantes:

A função `Anima` do código do projeto atualiza as variáveis de translação que efetivamente realiza a animação. As variáveis `xStep` e `yStep` são utilizadas para controlar o sentido do movimento em relação aos eixos `x` e `y`. Note que a função é atividade pela GLUT apenas uma vez. Para se ter uma animação contínua é necessária acrescentar uma nova chamada no final da função para ativá-la novamente. A razão desse comportamento é oferecer uma forma para que a animação possa ser interrompida. Além disso, para garantir a atualização da imagem é necessário chamar a função `glutPostRedisplay`.

Tratamento de eventos de mouse

A função `glutMouseFunc` é chamada pela GLUT cada vez que ocorre um evento de mouse. Os parâmetros de entrada da função são: (`int button, int state, int x, int y`). Os parâmetros `x` e `y` informam a posição do mouse no momento que o evento ocorreu. O parâmetro `button` informa qual botão

foi usado e `state`, se o mouse foi pressionado ou liberado. Os possíveis valores para os parâmetros `button` e `state` são:

- `button`: GLUT_LEFT_BUTTON, GLUT_MIDDLE_BUTTON ou GLUT_RIGHT_BUTTON
- `state`: GLUT_UP ou GLUT_DOWN

Utilize o **Projeto03** para compreender os conceitos apresentados.

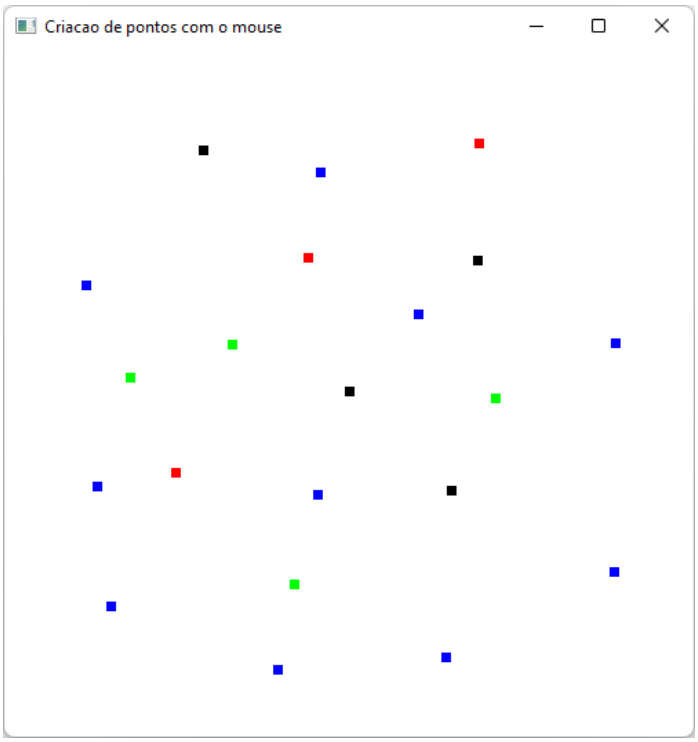

Figura 3 – Projeto 03

Comportamento do programa:

Quando o usuário clicar com o botão esquerdo do mouse um ponto na cor **preta** deve exibido na cena. Se o usuário pressionar:

- "R" ou "r" um novo ponto será exibido na cor **vermelha**
- "G" ou "g" um novo ponto será exibido na cor **verde**
- "B" ou "b" um novo ponto será exibido na cor **azul**

Exercício

Implemente um programa em OpenGL que crie uma cena (Figura 4) com um quadro quadrado e um círculo com 7 vértices.

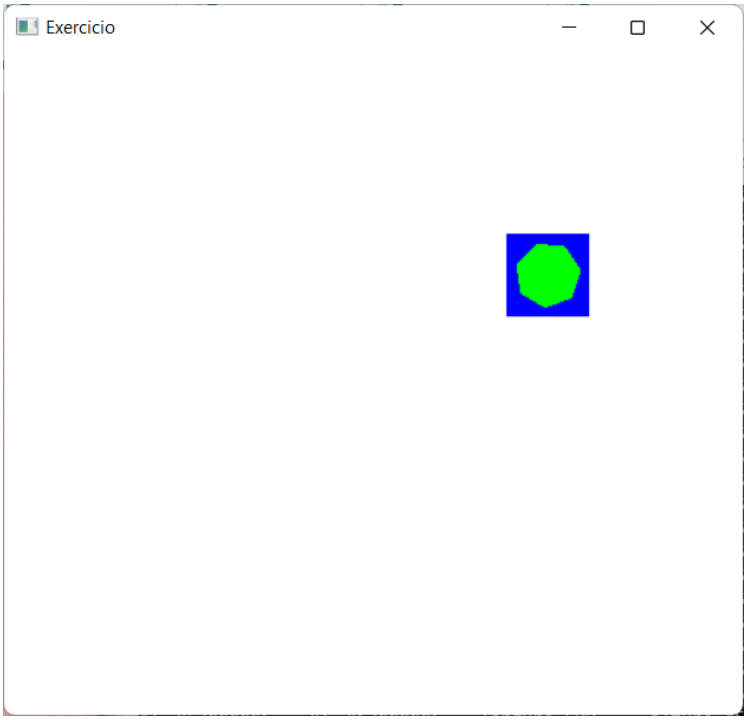

Figura 4 – Exercício

Dificuldade	Funcionamento
	**Translação** • O quadrado e círculo devem ser deslocados na diagonal até uma das bordas da janela e então mudar de direção. Esse comportamento é semelhante ao que foi implementado no **Projeto02**.
	**Rotação** • O círculo deve ser rotacionado a cada deslocamento realizado. Fica a critério do aluno definir o sentido da rotação.
	**Eventos de teclado** • As transformações geométricas devem ser paralisadas e continuadas se o usuário pressionar "P" ou "p".
	**Eventos de mouse** • As posições do quadrado e círculo devem ser alteradas quando o usuário clicar com o botão esquerdo do mouse. Os objetos devem ser transferidos para a posição do cursor do mouse.

Observações

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

A proposta do exercício foi retirada e adaptada do tutorial "Animação com OpenGL e GLUT" do site da Profª. Isabel Harb Manssour (https://www.inf.pucrs.br/~manssour/OpenGL/Animacao.html).

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)