



29 de maio de 2018

Universidade de São Paulo  
Instituto de Ciências Matemáticas e de Computação  
Bacharelado em Ciências de Computação

Computação Gráfica  
Profa. Agma J. M. Traina  
Estagiário PAE: Daniel Y. T. Chino

## 2º Trabalho Prático

### Fase 1

Grupo:

Bárbara Côrtes	9763050
Lucas Turci	9763085

## Introdução

Este projeto consiste em criar um programa interativo de visualização de objetos 3D, em que é permitido ao usuário transladar, rotacionar e redimensioná-los. O usuário pode optar entre uma esfera, um cubo ou um bule (todos em arame), e o objeto selecionado é exibido em quatro portas de visão, cada uma com uma configuração de espelhamento. A linguagem escolhida para o desenvolvimento deste projeto foi Python.

## Instruções de Execução

Antes de executar o programa é necessário possuir Python 2 e as bibliotecas PyOpenGL e PyOpenGL\_accelerate instaladas. Elas foram instaladas no Ubuntu 16.04 LTS utilizando o comando:

```
sudo pip install PyOpenGL PyOpenGL_accelerate
```

Caso, após instalar desta maneira, o programa ainda não execute corretamente, tente o comando:

```
sudo apt-get install python-opengl
```

Para a execução, basta digitar no terminal:

```
python bule.py
```

## Instruções de Uso

Ao iniciar o programa, o usuário pode escolher se deseja visualizar o objeto apresentado usando uma projeção ortogonal ou em perspectiva. Além deste, no universo estão representados os eixos  $X$ ,  $Y$  e  $Z$ , com cores vermelha, amarela e azul, respectivamente. O objeto é posicionado na origem e o ponto de observação ( $O$ ) é fixado em  $O = (15.0, 15.0, 15.0)$ . Dessa forma, o eixo  $X$  aparece no canto direito e o  $Z$  no canto esquerdo da porta de visão. Esse ponto de observação foi escolhido de maneira a garantir que o usuário conseguisse enxergar o objeto em sua posição inicial por inteiro e sem que ele ficasse muito distante.

O usuário pode selecionar o objeto que deseja visualizar com as teclas 1, 2 e 3. A tecla 1 escolhe a esfera (padrão), 2 escolhe o cubo e 3 o bule. Ele também pode realizar as transformações geométricas pelo teclado - conforme as orientações da imagem abaixo - através, ou das teclas Q, W, E (para incrementar uma das coordenadas), ou A, S e D (para decrementar uma delas), ou - e + para reduzir ou ampliar a o objeto. Caso alguma das teclas Q, W, E, A, S ou D sejam pressionadas junto com a tecla SHIFT, é alterado o ângulo de rotação do objeto em torno de seu eixo (veja a *Figura 1* abaixo para melhor compreensão).

```
COMANDOS
=====

\ESCOLHA DE OBJETO
    (1) --> Escolhe a esfera
    (2) --> Escolhe o cubo
    (3) --> Escolhe o bule

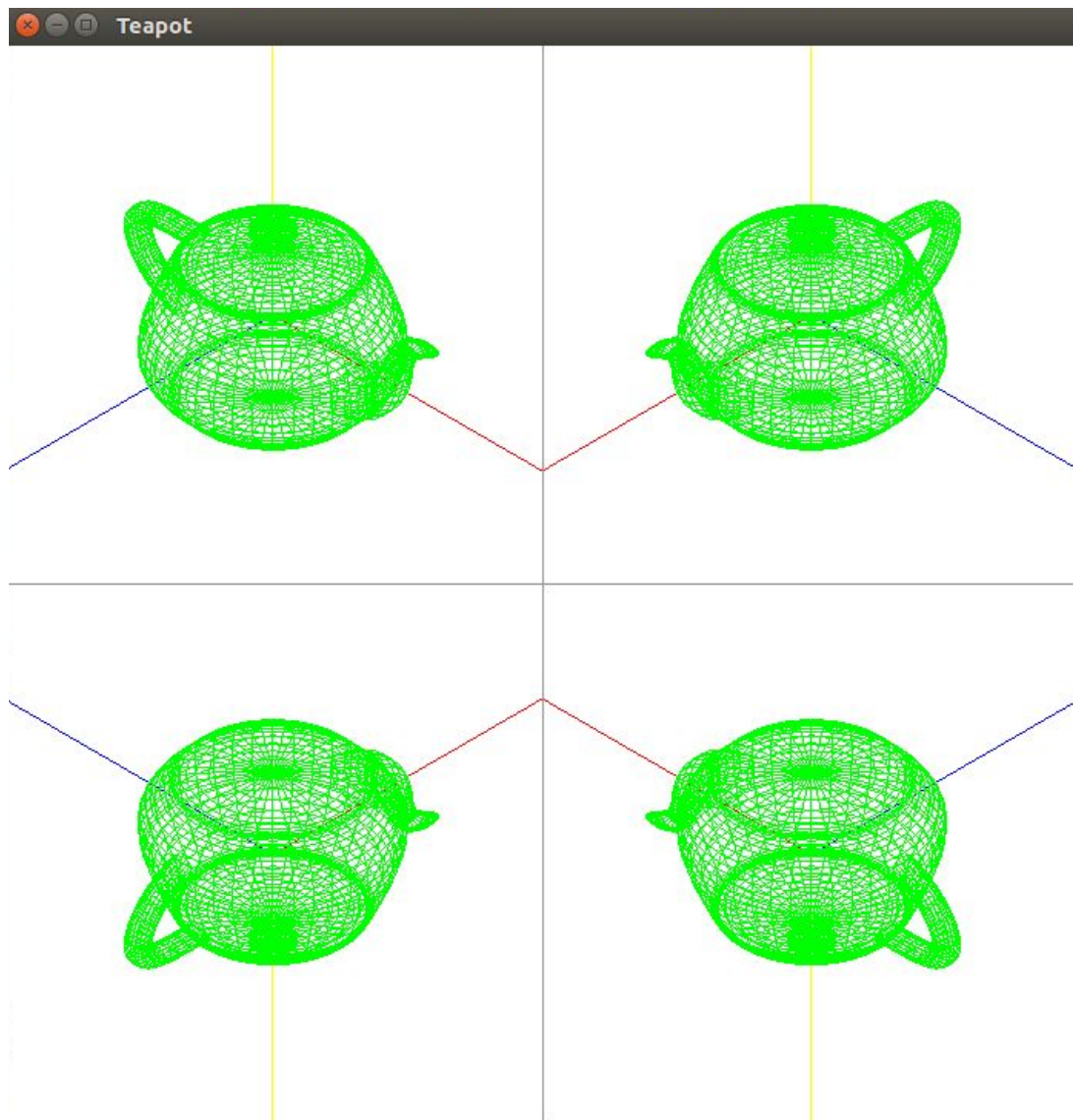
TRANSLACAO
    (q) --> positiva no eixo x
    (a) --> negativa no eixo x
    (w) --> positiva no eixo y
    (s) --> negativa no eixo y
    (e) --> positiva no eixo z
    (d) --> negativa no eixo z

ROTACAO
    (Q) (shift + q) --> sentido anti horario no eixo x
    (A) (shift + a) --> sentido horario no eixo x
    (W) (shift + w) --> sentido anti horario no eixo y
    (S) (shift + s) --> sentido horario no eixo y
    (E) (shift + e) --> sentido anti horario no eixo z
    (D) (shift + d) --> sentido horario no eixo z

ESCALA
    (+) (shift + =) --> aumenta a escala
    (-) --> diminui a escala
```

*Figura 1: Interface textual do programa exibindo instruções de uso*

Após ler as instruções, o usuário deverá pressionar a tecla ENTER para continuar a execução, que irá criar a janela de exibição do objeto. Essa janela pode ser visualizada a seguir, na *Figura 2*.



*Figura 2: Essa imagem mostra a janela criada após o usuário pressionar ENTER. Os eixos vermelho, amarelo e azul representam respectivamente os eixos x, y, e z. No canto superior esquerdo, o bule em projeção paralela. Acima e à direita, o bule espelhado em y; abaixo e à esquerda, o bule espelhado em x; abaixo e à direita, o bule espelhado em x e y.*

## Implementação

A seguir encontram-se breves descrições das funções implementadas pelo programa, e de seus atributos.

- *init()*:

Essa função inicializa os parâmetros do programa e é responsável por chamar a função *glutInit()* e criar a janela.

- *setprojection(mask)*:

*mask*: esse parâmetro pode receber os valores 0, 1, 2 ou 3. 0 indica que não há nenhum espelhamento. 1 indica um espelhamento em x, 2 em y e 3 em x e y, simultaneamente.

Essa função altera a matriz de projeção de maneira a definir qual tipo (paralela ou perspectiva) será utilizado para exibição da imagem, de acordo com a escolha do usuário. Além disso, essa função trata do espelhamento do objeto, invertendo o sinal dos parâmetros referentes a x ou y, dependendo da *view port* em questão.

Os parâmetros passados para as funções *glOrtho()* e *glFrustum()* foram definidos de maneira a mostrar o bule por inteiro na sua posição inicial e com certa margem para que o usuário consiga movimentá-lo. Para a projeção em perspectiva foi considerado ainda o fator de distorção da imagem.

- *drawAxis()*:

Essa função desenha os eixos x, y e z, em vermelho, amarelo e azul - respectivamente - na porta de visão atual.

- *keyPressEvent(key, x, y)*:

*key*: parâmetro que guarda a tecla que foi pressionada no teclado

*x*: coordenada x do cursor do mouse

*y*: coordenada y do cursor do mouse

Essa função é definida como a função de callback para *glutKeyboardFunc*, e realiza o tratamento de eventos do teclado ou de clique do mouse. Neste programa, é

responsável pela alteração de variáveis globais que definem as propriedades do objeto (ângulo, posição e escala), e da seleção do objeto desejado.

Ao final dessa função, é chamada a função *display()*, para que sejam mostradas na tela quaisquer transformações feitas pelo usuário.

- *displayViewPort(x, y, w, h, mask):*

*x*: coordenada *x* da origem da porta de visão.

*y*: coordenada *y* da origem da porta de visão.

*w*: largura da porta de visão.

*h*: altura da porta de visão.

*mask*: máscara de bits, em que o primeiro bit representa se há simetria pelo eixo *X*, e o segundo bit, se existe pelo eixo *y*. (ver *setProjection*).

Essa função é responsável por configurar uma porta de visão com as propriedades passadas, realizar as transformações geométricas do objeto e desenhá-lo em seguida. Ela configura a porta de visão através dos parâmetros passados com a chamada *glViewport(x, y, w, h)*. Depois, é chamada a função *setProjection* com a máscara (*mask*). Aqui também está presente a chamada de *drawAxis()*. Em seguida, são feitas as transformações de translação, rotação e escala e, por fim, é chamada a função *glutWireTeapot*, que desenha o bule.

- *drawLines():*

Essa função é responsável por desenhar duas linhas passando pelo centro da janela, com o objetivo de delimitar as quatro portas de visão.

- *display():*

Essa função é definida como a função de callback para *glutDisplayFunc*, e realiza o desenho dos objetos do programa. Também é invocada sempre que alguma transformação geométrica é efetuada. Ela invoca as funções de desenho de cada uma das portas de visão, e a função *drawLines()*, que desenha as linhas divisórias das portas de visão.

## **Resultados**

A aplicação final tornou-se uma ótima ferramenta para visualização de objetos 3D, que além de ser interativa e fácil de usar, é bastante útil para ver e compreender as transformações geométricas de translação, rotação e escala dos objetos.