Aula prática 6

Objetivos

Construir um programa interactivo que permita instanciar objectos 3D (inicialmente apenas cubos).

Modelar cada primitiva através duma transformação de instanciação (T.Rz.Ry.Rx.S) definida pelo utilizador.

Exercício 6.1

Pretende-se estender o exercício 5.1 para 3D, permitindo ao utilizador instanciar um objeto repetidas vezes, podendo aplicar, à última instância do objeto, uma transformação de instanciação (mModel), representada pela sequência de transformações elementares (T.Rz.Ry.Rx.S). Todas as transformações são em 3D.

A aplicação deverá disponibilizar os seguintes elementos de interface:

- slider para ajustar os valores da translação (em X, Y e Z)
- slider para ajustar o valor da rotação em torno de X, Y e Z.
- slider para ajustar o valor da escala (em X, Y e Z).

Os limites dos valores a aplicar aos sliders serão:

- Translação: [-1,1]
- Escala: [0.05,2]
- Rotação: [-180,180]

Os sliders aplicar-se-ão à `mModel` (matriz de modelação/instanciação) aplicada à primitiva corrente.

Existe ainda um botão que permite guardar definitivamente a matriz de instanciação e associá-la à primitiva que estava a ser editada, sem que se possa voltar atrás. De forma automática, instancia-se uma nova primitiva, com valores por omissão para as transformações, por forma a que a `mModel` seja a matriz identidade. Não esquecer atualizar os valores dos elementos da interface (sliders) para o seu valor refletir o estado atual.

Em cada momento, o programa deverá ser capaz de redesenhar todas as primitivas que foram guardadas, bem como a primitiva em edição.

Tarefas:

- Vamos usar como ponto de partida a sua solução do problema 5.1
- Comece por adicionar aos seus scripts javascript, os seguintes ficheiros: cube.js e sphere.js.
- No evento window.onload (a inicialização da aplicação) deverá invocar as seguintes funções: `cubeInit(gl)` e `sphereInit(gl)`. Estas funções inicializam os buffers com a geometria dos respetivos objetos. Nestes ficheiros existem ainda as funções `cubeDrawWireFrame(gl, program)` e `sphereDrawWireFrame(gl, program)` que desenham, respetivamente, um cubo e uma esfera numa representação em malha de arame. Estas funções deverão ser estudadas em detalhe.
- Modifique a sua função `render()` por forma a passar para o shader duas novas matrizes: `mProjection` e `mView`, as quais deverão ser inicializadas (na zona adquada do seu código) da seguinte forma (não se assuste se não perceber o que este pedaço de código faz. É normal, pois a respetiva matéria ainda não foi dada).

```
var at = [0, 0, 0];
var eye = [1, 1, 1];
var up = [0, 1, 0];
mView = lookAt(eye, at, up);
mProjection = ortho(-2,2,-2,2,10,-10);
```

- Não se esqueça de acrescentar ao vertex shader as respetivas matrizes e de passar os valores destas matrizes ao programa GLSL (shaders) quando for desenhar algo. O vertex shader deverá ter a seguinte linha de código (a matriz mModel é a matriz que contém a transformação de instanciação da primitiva):

```
gl_Position = mProjection * mView * mModel * vPosition;
```

- Inclua no ficheiro html os sliders em falta para controlar a transformação corrente, bem como o botão para acrescentar de forma definitiva uma primitiva à lista de primitivas bem como os respetivos *event handlers*.

Exercícios propostos

TPC:

Exercício 6.2 - Acrescente as seguintes funcionalidades:

- um botão para fazer reset à transformação corrente.
- um botão para fazer reset ao programa, voltando ao estado inicial.

Exercício 6.3 - Acrescente mais um botão, permitindo à aplicação instanciar cubos e esferas.

Exercício 6.4 - Acrescente uma nova primitiva, por exemplo um cilindro ou um donut.