



Computação Gráfica (MIEIC) 2020/2021 *Projeto Final - Parte* **A** v1.0 (2021/03/26)

Objetivos

- Aplicar os conhecimentos e técnicas adquiridas até à data
- Criação de uma cena complexa, com diferentes geometrias, materiais, texturas e luzes
- Exploração de shaders, interação, controlo por teclado e animação

Descrição

Pretende-se com este projeto a criação de uma cena que combine os diferentes elementos explorados nas aulas anteriores, acrescentando alguns novos elementos. Para este trabalho deve usar como base o código que é fornecido no Moodle, que corresponde a uma cena vazia. Terá posteriormente de adicionar alguns dos objetos criados anteriormente.

A cena será no final genericamente constituída por:

- Um elemento que se move controlado pelo utilizador
- Uma paisagem envolvente (*cube map*)





Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

dada alguma liberdade quanto à composição dos mesmos na cena, para que cada grupo possa criar a sua própria cena.

O enunciado está dividido em duas partes, com uma proposta de um plano de trabalhos no final de cada uma delas. Nesta versão do enunciado está apenas disponível a primeira parte, que servirá para criar os elementos-base necessários para criar o ambiente final de acordo com o tema proposto na segunda parte (a publicar posteriormente).

À semelhança dos trabalhos anteriores, será necessário fazerem capturas de ecrã em alguns pontos do enunciado, bem como assinalar versões do código no *Git* com *Tags*. Os pontos onde tal deve ser feito estão assinalados ao longo do documento e listados numa check list no final deste enunciado, sempre assinalados com os ícones 

(captura de uma imagem) e  (tags apenas). **Neste projeto, apenas haverá uma submissão de código no Moodle no final, mas têm de fazer as tags intermédias no Git.**

Parte A

1. *Objeto controlável*

1.1. *MyMovingObject* - *Versão preliminar*

Crie a classe ***MyMovingObject***, que irá representar o objeto controlável na cena. A constituição final do elemento controlável será detalhada na Parte B do enunciado. Nesta fase deve criar-se apenas uma representação básica que permita identificar a sua posição e orientação. Para esse efeito, sugerimos que use uma forma simples, como um triângulo não-equilátero como os ***MyTriangle***



Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

- ser controlada na origem,
- Ter tamanho unitário
 - Ter o eixo central alinhado com o eixo positivo dos ZZ, ou seja: ter a frente a apontar na direção de +Z, como demonstrado na **Figura 1**.

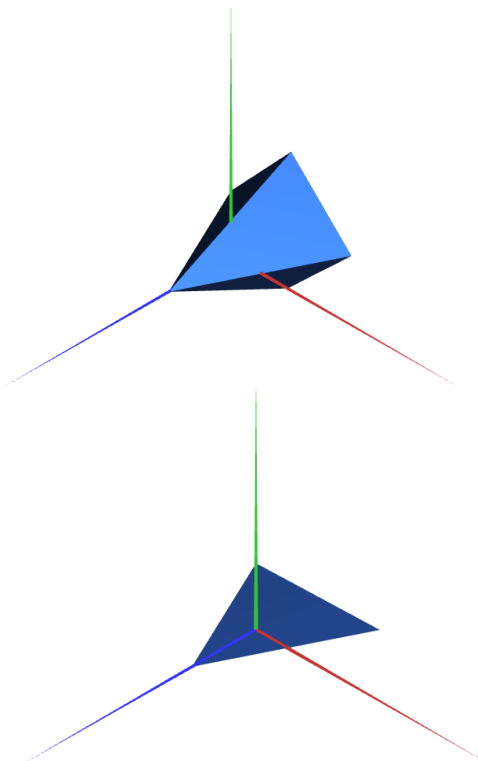


Figura 1: Exemplificação da versão preliminar de MyMovingObject
a) Pirâmide: b) Triângulo no plano XZ

1.2. **Controlo de elemento controlável por teclado**

Para poder controlar o movimento do elemento controlável na cena utilizando teclas, será necessário alterar:

- A **interface**, para detetar as teclas pressionadas
- A **cena**, para invocar funções de controlo no



Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

mesmo

Apresentam-se em seguida de forma mais detalhada os passos necessários para o efeito.

NOTA IMPORTANTE: Se usar *copy-paste* com o código seguinte, alguns símbolos podem não ser bem reconhecidos (apesar de parecerem iguais) e causar erros em Javascript. Evite fazê-lo, ou verifique bem o código.

1. Altere a classe ***MyInterface***, adicionando os seguintes métodos para processar várias teclas ao mesmo tempo:

initKeys() {	
// create reference from the scene to the GUI	
this.scene.gui=this;	
// disable the processKeyboard function	
this.processKeyboard=function()	
{};	
// create a named array to store which keys are being pressed	
this.activeKeys=	
{};	
}	
processKeyDown(event)	
{	
// called when a key is pressed down	
// mark it as active in the array	
this.activeKeys[event.code]=true;	
};	



Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

```

        this.activeKeys[event.code]=false;
    };

    isKeyPressed(keyCode)
    {
        if(
            this.activeKeys[keyCode]
            === true &&
            (keyCode == "keyL" ||
            keyCode == "keyP")) {

            this.activeKeys[keyCode]
            = false;

            return true;
        }

        return
        this.activeKeys[keyCode]
        || false;
    }

```

NOTA: No final da função *init()* da **MyInterface**, chame a função *initKeys()*.

2. Na classe **MyScene** acrescente o seguinte método *checkKeys()* e acrescente uma chamada ao mesmo no método *update()*.

```

checkKeys() {
    var text="Keys
pressed: ";
    var
    keysPressed=false;

    // Check for
    key codes e.g. in
https://keycode.info/

    if
    (this.gui.isKeyPressed("KeyW"))

```



Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

if
(this.gui.isKeyPressed("KeyS"))
{
text+="
S ";
keysPressed=true;
}
if
(keysPressed)
console.log(text);
}

Execute o código e verifique as mensagens na consola quando "W" e "S" são pressionadas em simultâneo.

3. Prepare a classe ***MyMovingObject*** para se deslocar:

- 3.1. Acrescente no construtor variáveis que definam:

- a orientação do elemento controlável no plano horizontal (ângulo em torno do eixo YY)
- a sua velocidade (inicialmente a zero)
- a sua posição (x, y, z)

- 3.2. Crie a função ***MyMovingObject.update()*** para atualizar a variável de posição em função dos valores de orientação e velocidade.

- 3.3. Use as variáveis de posição e orientação na função



Projeto Final 2020/2021 - Parte A


Atualização automática a cada 5 minutos

alterar o ângulo de orientação, e para aumentar/diminuir o valor da velocidade (em que **val** podem ser valores positivos ou negativos).

3.5. Crie a função *reset()* que deverá recolocar o elemento controlável na posição inicial, com rotação e velocidade nula.

4. Altere o método *checkKeys()* da **MyScene** de forma a invocar os métodos *turn()*, *accelerate()* ou *reset()* do elemento controlável para implementar o seguinte comportamento em função das teclas referidas:

- **Aumentar ou diminuir a velocidade** conforme se pressionar “W” ou “S”, respectivamente.
- **Rodar o elemento controlável** sobre si mesmo para a esquerda ou direita se pressionar as teclas “A” ou “D” respectivamente.
- Pressionar a tecla “R” deverá invocar a função *reset()* do elemento controlável.

Crie uma tag no repositório neste ponto.  (1)

2. **Criação de objetos de base**



Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

Com este exercício pretende-se que crie um *cube map* que sirva como ambiente de fundo da cena. Um *cube map* pode ser definido como:

- um cubo de grandes dimensões (bastante superiores à da cena visível),
- com material próprio: com componente especular, ambiente e difusa nulas, e componente emissiva forte,
- apenas com as faces interiores visíveis,
- e às quais são aplicadas texturas que representam a envolvente da cena (uma paisagem, por exemplo; ver **Figura 2**).

Para este efeito devem usar como base o ***MyUnitCubeQuad*** e seis texturas diferentes, uma para cada face do cubo.

Inclua na pasta do projeto uma cópia do código da classe ***MyUnitCubeQuad***. Modifique essa cópia para criar uma classe ***MyCubeMap***, de forma a ser visível por dentro, e com a aplicação das respectivas texturas.

O *cube map* deve ser unitário e ao ser usado na cena, deve ser escalado de forma a medir **50 unidades de lado**. Adicionalmente, deve ser desenhado tendo como base/centro a posição da câmara. Pode utilizar para o efeito o membro *position* da câmara guardada na cena: `this.camera.position[0]`, `...[1]`, `...[2]`.

O código de base inclui dois exemplos de conjuntos de 6 imagens para mapear nos seis *quads*. Devem procurar/criar pelo menos mais uma paisagem envolvente à vossa escolha, para permitir depois ao utilizador escolher entre essa e a de exemplo através da interface gráfica (ver ponto 2.2).



Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

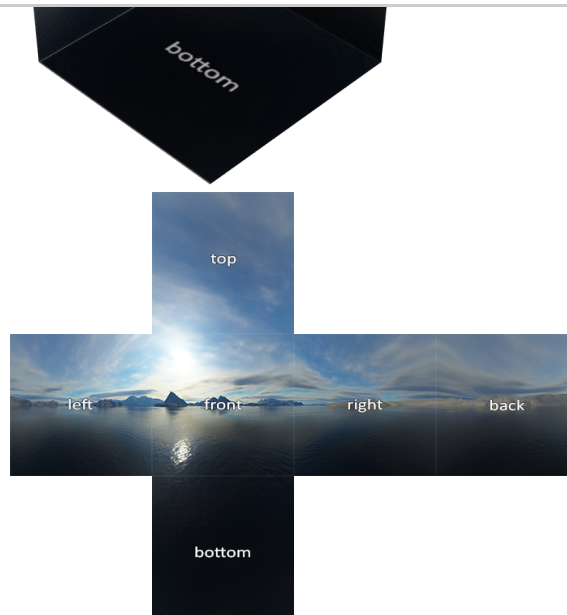



Figura 2: Imagem do tipo skybox, e a sua aplicação num cubemap visualizado de fora

Crie uma tag no repositório neste ponto.  (2)

2.2. *MyCylinder* - Cilindro (sem topos)

Crie uma nova classe ***MyCylinder*** que aproxime um cilindro de raio de uma unidade, com um número variável de "lados" (**slices**), e altura de uma unidade em Y. A base do cilindro deve ser coincidente com o plano XZ e centrada na origem.

As normais de cada vértice devem ser definidas de forma a aproximar uma superfície curva, de acordo com o método de Smooth Shading de Gouraud, como demonstrado na **Figura 3**.



Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

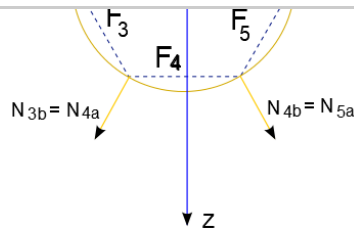


Figura 3: Ilustração das normais a atribuir a cada vértice no caso de um cilindro aproximado por seis lados.

Para cada vértice, defina as coordenadas de textura de forma a mapear uma textura à volta do cilindro. Note que como a textura dá a volta ao cilindro poderá necessitar de repetir a primeira linha vertical de vértices pois estes são, simultaneamente, o início e o fim da textura.

Crie uma tag no repositório neste

ponto.  (3)

2.3. *MySphere* *Esfera*

No código fornecido encontra uma classe *MySphere* correspondente a uma esfera com o centro na origem, com eixo central coincidente com o eixo Y e raio unitário.

A esfera tem um número variável de "lados" à volta do eixo Y (slices), e de "pilhas" (stacks), que corresponde ao número de divisões ao longo do eixo Y, desde o centro até aos "pólos" (ou seja, número de "fatias" de cada semi-esfera). A **Figura 4** tem uma representação visual da esfera.

Pretende-se que analise o código deste objeto, e acrescente o código necessário para gerar as coordenadas de textura para aplicar texturas à superfície da esfera, como demonstrado na **Figura 5**.



Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

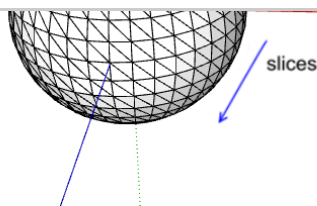


Figura 4: Imagem exemplo de uma esfera centrada na origem.

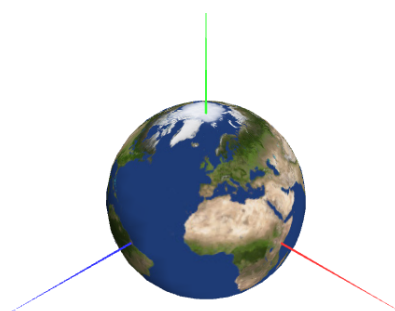
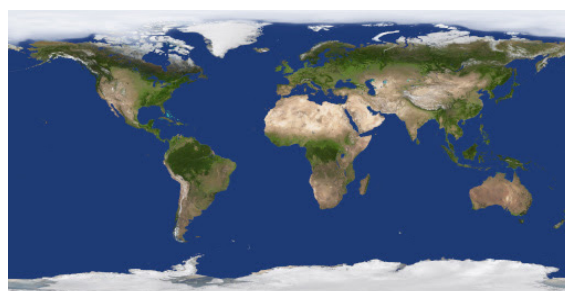



Figura 5: Exemplo de textura (disponível no código base) e sua aplicação numa esfera

Crie uma tag no repositório neste ponto.  (4)

3. *Controlos adicionais na interface*

Acrescente os seguintes controlos adicionais na interface gráfica (GUI), para poder parametrizar o movimento do elemento controlável e a aparência do *cube map*:




Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos

(sugestão: siga o exemplo das texturas no TP4).

2. Crie outro *slider* na GUI chamado **scaleFactor** (entre 0.5 e 3) que permita escalar o elemento controlável de forma a que seja mais fácil observar as suas animações.
3. Crie um *slider* na GUI chamado **speedFactor** (entre 0.1 e 3) que deve ser usado para multiplicar as velocidades dos objetos controlados pelo utilizador e outras animações, de forma a “acelerar” ou “desacelerar” toda a cena.

Crie uma tag no repositório neste ponto.  (5)

Proposta de plano de trabalho - Parte A

- Semana de 29 de março: Apresentação de parte A; início do ponto 1
- Semana de 5 de abril: Continuação do ponto 1; Ponto 2.1
- Semana de 12 de abril: Ponto 2.2 e 2.3
- Semana de 19 de abril: Finalização da parte A e apresentação da parte B

Checklist

Durante a execução da parte A deverá identificar os commits com as tags assinaladas acima, **respeitando estritamente a regra dos nomes:**



Publicada através do Google Docs

[Saber mais](#)

[Denunciar abuso](#)

Projeto Final 2020/2021 - Parte A

Atualização automática a cada 5 minutos
