

Computação Gráfica (MIEIC)

Trabalho Prático 6


Projeto Final



Objetivos

- Aplicar os conhecimentos e técnicas adquiridas até à data
- Utilizar elementos de interação com a cena, através do teclado e de elementos da interface gráfica

Trabalho prático

Ao longo dos pontos seguintes são descritas várias tarefas a realizar. Algumas delas estão anotadas

com o ícone  (captura de imagem). Nestes pontos deverão, com o programa em execução, capturar uma imagem da execução. Devem nomear as imagens capturadas seguindo o formato "CGImage-tp6-TtGgg-x.y.png", em que TtGgg referem-se à turma e número de grupo e x e y correspondem ao ponto e subponto correspondentes à tarefa (p.ex. "CGImage-tp6-T3G10-2.4.png", ou "CGImage-tp6-T2G08-extra.jpg").

Nas tarefas assinaladas com o ícone  (código), devem criar um ficheiro .zip do vosso projeto, e nomeá-lo como "CGFCode-tp6-TtGgg-x.y.zip", (com TtGgg, x e y identificando a turma, grupo e a tarefa tal como descrito acima). Quando o ícone  surgir, é esperado que executem o programa e observem os resultados. No final, devem submeter todos os ficheiros via Moodle, através do link disponibilizado para o efeito. Devem incluir também um ficheiro **ident.txt** com a lista de elementos do grupo (nome e número). Só um elemento do grupo deverá submeter o trabalho.

Preparação do Ambiente de Trabalho

Este trabalho deve ser baseado numa cópia do trabalho anterior (uma sala de aula com, pelo menos dois planos, duas mesas, duas paredes, chão e um cilindro, e um relógio animado).

Iremos acrescentar uma classe de interface que criará uma área de interface gráfica com alguns elementos de interação, e que será também responsável por gerir eventos de teclado. Para tal, é fornecido o ficheiro **MyInterface.js** que devem incluir no projeto da seguinte forma:

- Colocar o ficheiro na mesma diretoria dos restantes ficheiros Javascript do projeto
- editar o ficheiro **main.js** e
 - adicionar '**MyInterface.js**' à lista de ficheiros a incluir
 - substituir no código da função main a referência a **CGFInterface** por **MyInterface**
- Editar o vosso ficheiro de cena (**LightingScene.js**) e
 - acrescentar no método **LightingScene.init**, as seguintes variáveis:

```
this.option1=true; this.option2=false; this.speed=3;
```
 - acrescentar ao ficheiro o seguinte método:

```
LightingScene.prototype.doSomething = function ()  
{ console.log("Doing something..."); };
```

1. Criação da classe MyDrone

Neste exercício procura-se criar uma geometria para representar um *drone* que servirá de *avatar*.

1. Crie uma classe **MyDrone** que represente um *drone*. Esta classe será responsável, inicialmente, pelo desenho do drone. Nesta fase será constituída apenas por um triângulo paralelo ao plano XZ, com coordenadas (0.5, 0.3, 0), (-0.5, 0.3, 0), (0, 0.3, 2), ou seja, um triângulo a apontar para +ZZ.
2. Aplique as transformações necessárias para colocar o drone no centro da sala, a apontar (aproximadamente) para o quadro da esquerda.

(1.2 ) (1.2 ) 

2. Controlo do Drone

Neste exercício procura-se criar um mecanismo de controlo para o avatar acima criado. Consulte a classe **MyInterface.js** para ver exemplos de utilização que ajudarão na resolução destes pontos.

1. Crie um mecanismo para controlar o drone utilizando as teclas: rodar para a esquerda ou para a direita, conforme a tecla premida é "A" ou "D", e mover-se no sentido para onde está virado ou no sentido contrário, conforme se pressione "W" ou "S", respetivamente. Para subir e descer, devem utilizar-se as teclas "I" e "J", respetivamente. Deve criar as variáveis ou métodos necessários para suportar estes movimentos na classe da cena e do drone de forma a poder alterá-las ou invocá-las na classe de interface.

(2.1 ) 

3. GUI

Neste exercício procura-se criar uma interface gráfica (GUI) com alguns controlos para alterar parâmetros da cena em tempo de execução.

1. Adicione à GUI um grupo intitulado "Luzes" (remova/comente/substitua o grupo do exemplo). Acrescente ao novo grupo, por cada fonte de luz utilizada, uma checkbox. Cada checkbox (estado on/off) deve permitir alterar o estado (respetivamente acesa/apagada) da fonte de luz que lhe diz respeito.



2. Adicione um botão que pause/retome o mecanismo de animação do relógio da cena;

(3.2 ) (3.2 ) 

4. Modelação do Drone

Neste exercício procura-se criar uma geometria algo mais complexa para substituir a representação anterior do drone. O drone deverá ser composto por vários elementos, todos eles com capacidade para conter texturas aplicadas. As texturas deverão ser criadas/selecionadas pelos estudantes. Pretende-se que o drone tenha uma estrutura semelhante à da Fig. 1.



1. Sobre o corpo do Drone:
 - a. Deverá ser composto por dois cilindros cruzados em forma de + (mais), representando os braços, sendo a parte da frente uma das pontas do +.
 - b. Em cada ponta dos braços existirá um cilindro que servirá de base para as hélices.
 - c. O centro do Drone será composto por uma semi-esfera. (poderá usar cilindros mas sofrerá uma penalização na cotação).
2. O drone terá duas “pernas”, sendo que cada uma tem:
 - a. uma base paralelepípedica
 - b. Duas tiras (superfícies) unindo a base ao corpo principal. Estas tiras devem ser curvilineas (ex. Parábolicas, segmentos circulares, b ezier). A utiliza  o de outras geometrias mais simples   poss vel, mas n o ter  a cota  o total.
3. O Drone ter  quatro h lices iguais como indicado na imagem, o centro da h lice   composto por uma semi-esfera e as p s das h lices s o compostas por cilindros achatados.
(NOTA: n mero de slices e stacks dever  ser reduzido atendendo   escala dos objectos)



4. Construa uma interface para a sele  o das texturas, integrada na GUI da aplica  o. Deve, para o efeito, usar um controlo do tipo “drop-down”. Para implementar este tipo de controlos, sugere-se:
 - a. Declarar na cena um array **droneAppearances** que contenha as v rias appearances poss veis
 - b. Declarar um dicion rio **droneAppearanceList** que mapeie as strings identificando cada appearance ao seu  ndice em **droneAppearances**.
 - c. Declarar na cena uma vari vel **currDroneAppearance** que identifique o  ndice da appearance selecionada/atual.
 - d. Adicionar um controlo na interface que fique associado a **currDroneAppearance** e a **droneAppearanceList**
(exemplos em <http://workshop.chromeexperiments.com/examples/gui/#2--Constraining-Input>)
 - e. Ajustar o c digo de desenho da cena ou do drone para que seja usada a **appearance** correta.
 - f. Inclua pelo menos uma textura que permita facilmente identificar a parte frontal do drone.

5. Execute e altere as texturas usando a GUI.



5. Animação do Drone

Neste exercício procura-se animar o Drone e as suas hélices.

1. O movimento de um Drone real é controlado pela direção e velocidade de rotação das suas hélices, de acordo com o diagrama da Fig. 2.
 - a. Considere três velocidades de rotação diferentes - lenta (L), normal (N) e rápida (R) - correspondendo a 0.2, 1 e 10 rotações por segundo, respectivamente.
 - b. Anime as hélices do seu Drone de forma a replicar os tipos de movimento exemplificados na Fig. 2

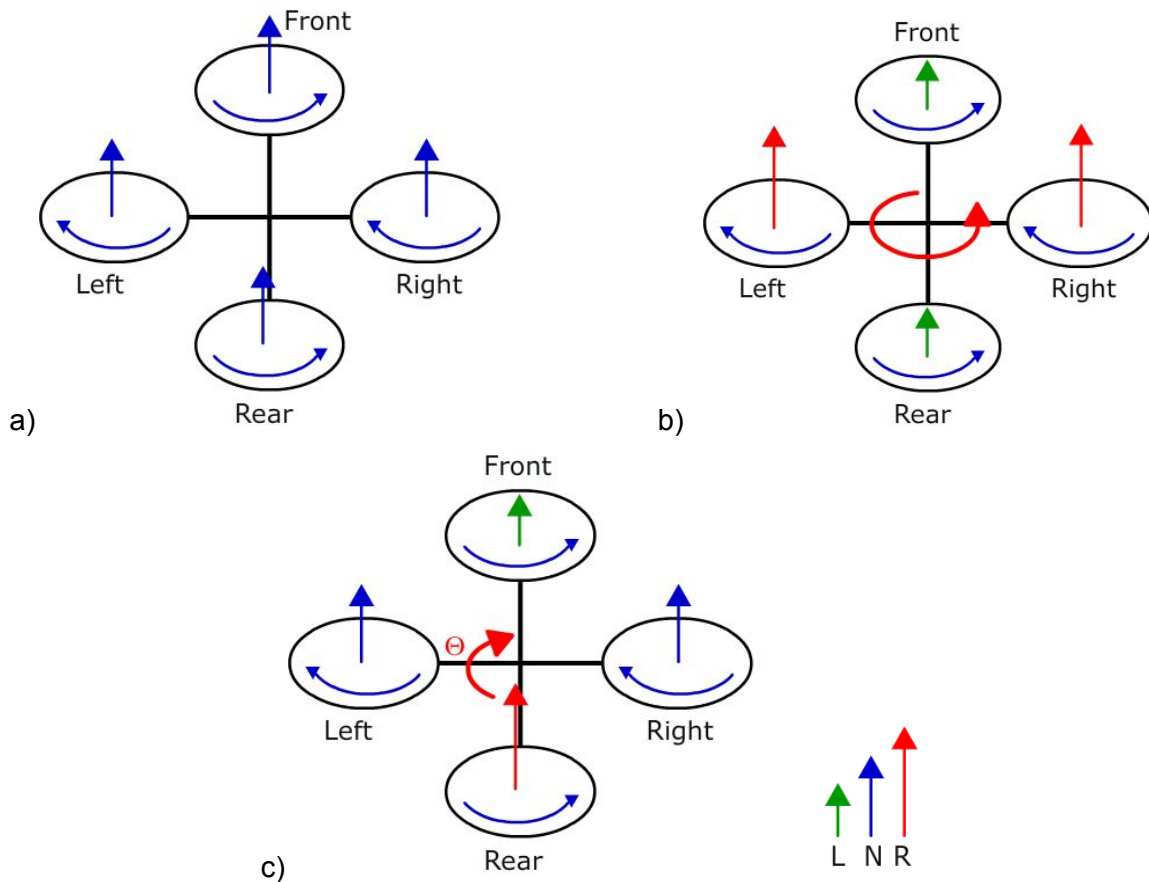


Fig. 2: Exemplos dos três modos de movimento a) Modo estático;
b) Movimento de rotação (Yaw); c) movimento para a frente com inclinação (Pitch).

2. Quando o drone avança ou recua, deve inclinar-se ligeiramente para a frente ou para trás, respetivamente. Implemente essa animação.

3. Para efeitos de visualização, deve contemplar um fator de escala na GUI que permita ver a rotação das hélices em “câmara lenta”. Esse fator deve variar entre 0.1 e 2.0. Um valor de 1.0 corresponderá às velocidades de rotação mencionadas acima, um fator de 2 fará as hélices rodarem ao dobro da sua velocidade original, um fator de 0.5 a metade. Este fator não deve alterar a velocidade de movimento do drone propriamente dito, apenas da rotação das hélices.

(5.3 ) (5.3 ) 

(Continua...)