

# 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 "CGFImage-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. "CGFImage-tp6-T3G10-2.4.png", ou "CGFImage-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 nos vários elementos básicos criados nas aulas anteriores (**LightingScene.js**, planos, cubos, semi-esferas e cilindros texturados).

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 MySubmarine e ambiente inicial

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

1. Crie uma classe **MySubmarine** que represente um submarino. Esta classe será responsável, inicialmente, pelo desenho do submarino. Nesta fase inicial do trabalho 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. Altere a cor de fundo da cena para azul.
3. Crie um plano e aplique-lhe uma textura que simule o fundo oceânico; optimize a utilização da textura, fazendo-a repetir várias vezes ao longo do plano.
4. Coloque o relógio num poste na posição ( 8, 5, 0).
5. Aplique as transformações necessárias para colocar o submarino no centro da cena, a apontar (aproximadamente) para o poste.

(1.5  ) (1.5  ) 

## 2. Controlo do Submarino

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. Estudar a biblioteca javascript dat.GUI através dos seguintes exemplos:

<https://workshop.chromeexperiments.com/examples/gui/#1--Basic-Usage>

1. Crie um mecanismo para controlar o submarino 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", respectivamente. Para este efeito, deve criar as variáveis ou métodos na classe da cena 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 (respectivamente 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 Submarino

Neste exercício deverá criar o modelo do submarino com mais detalhe, para substituir a geometria usada no ex. 1. O submarino será composto por vários elementos, tal como descrito abaixo e representado na figura 1 (dimensões aproximadas). Os elementos devem poder ter texturas aplicadas, texturas essas que deverão ser criadas/selecionadas pelos estudantes.

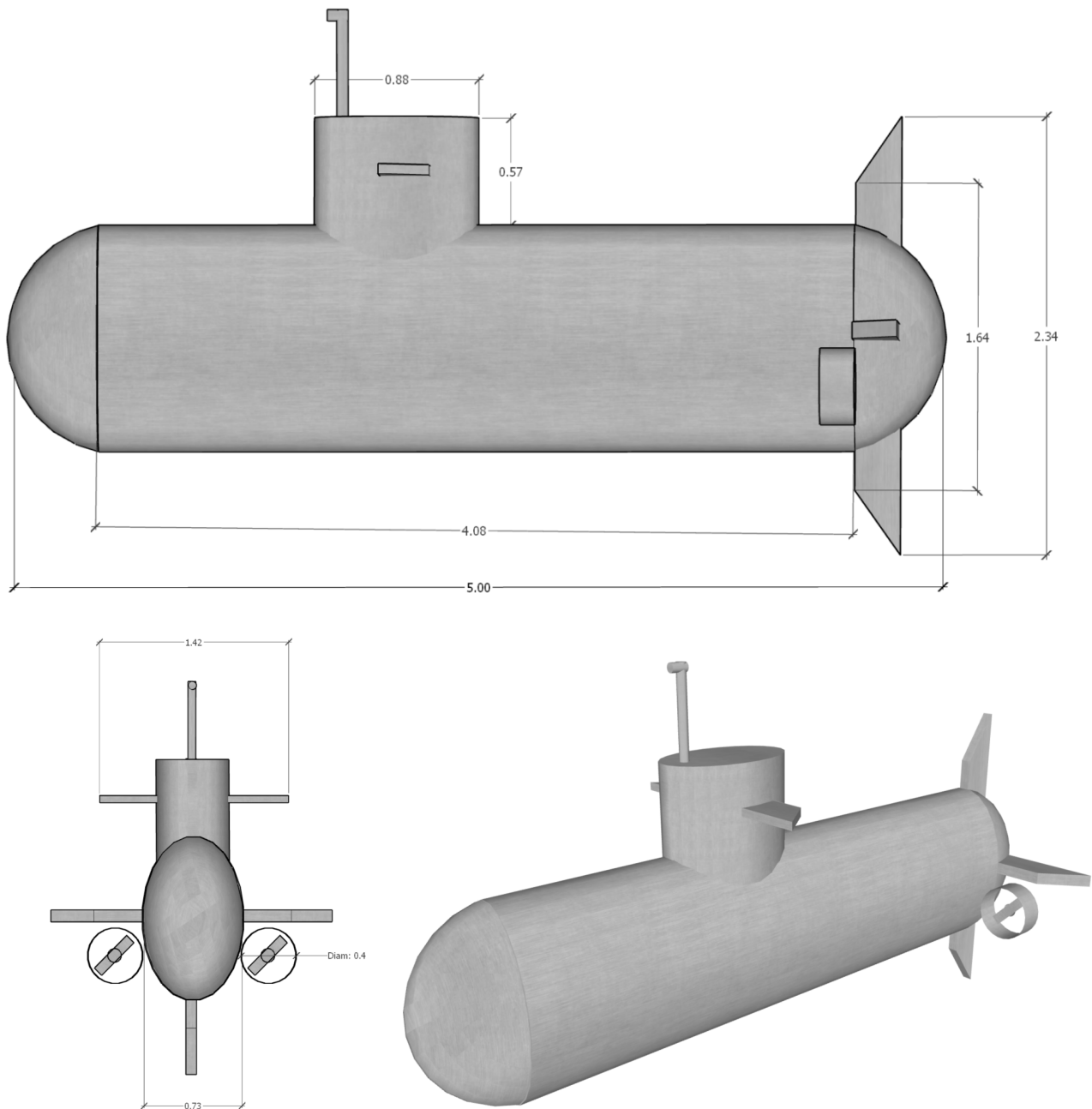


Figura 1: Vista de lado, de trás e em perspetiva do submarino, com dimensões indicativas aproximadas

- Construa o submarino com os seguintes componentes:
  - um cilindro e duas (semi-) esferas para o corpo principal, e um cilindro com topo para a torre superior, tudo "comprimido" no eixo dos xx;
  - dois cilindros para o periscópio;
  - três trapézios para as "barbatanas" da torre e da parte traseira (cada um atravessa o corpo de um lado ao outro);

- para as hélices, um paralelepípedo com uma semi-esfera no centro; à volta, um cilindro com faces interiores e exteriores, sem topos.

(4.1  ) 

- Construa uma interface para a seleção das texturas, integrada na GUI da aplicação. Deve para o efeito usar um controlo do tipo “drop-down”. Para implementar este tipo de controlos, sugere-se:
  - Declarar na cena um array **submarineAppearances** que contenha as várias appearances possíveis
  - Declarar um dicionário **submarineAppearanceList** que mapeie as strings identificando cada appearance ao seu índice em **submarineAppearances**.
  - Declarar na cena uma variável **currSubmarineAppearance** que identifique o índice da appearance selecionada/atual
  - Adicionar um controlo na interface que fique associado a **currSubmarineAppearance** e a **submarineAppearanceList**  
(exemplos em <http://workshop.chromeexperiments.com/examples/gui/#2--Constraining-Input>)
  - Ajustar o código de desenho da cena ou do submarino para que seja usada a **appearance** correta.
- Execute e altere as texturas usando a GUI

(4.3  ) (4.3  ) 

## 5. Animação do Submarino

Neste exercício procura-se animar os vários elementos que compõem o Submarino.

1. Adapte o controlo do submarino para que as teclas “W” e “S” passem a controlar a velocidade do submarino, em vez da sua posição diretamente. Ou seja, o submarino manterá a sua velocidade se nenhuma daquelas teclas for pressionada, e a velocidade aumentará de cada vez que “W” for pressionada, e diminuirá de cada vez que “S” for pressionada (podendo ser negativa, andando assim para trás).
2. Por forma a simular o movimento das hélices estas deverão girar em sentidos contrários com uma velocidade base de 1 rotação por segundo quando o submarino está na sua velocidade mínima. A velocidade de rotação das hélices deve ser proporcional à velocidade do submarino.



3. Os lemes verticais deverão rodar de acordo com o pressionar das teclas de rotação do submarino: por exemplo, quando/enquanto “A” é pressionada, o leme deve “abrir” para o lado esquerdo, para que o submarino rode para a esquerda (assumindo que tem velocidade não-nula). Quando a tecla deixar de estar pressionada, o leme deve regressar à posição central.
4. Programe as teclas “Q” e “E” para subir e descer o submarino, animando os lemes horizontais e o submarino de forma análoga à orientação esquerda/direita.



5. Ao ser pressionada a tecla ‘P’, o periscópio subirá até um valor máximo e com a tecla ‘L’ descenderá até um valor mínimo.

(5.5  ) (5.5  ) 

## 6. Torpedos e alvos

1. Crie a classe MyTarget que representa um objeto - alvo - a ser atingido por torpedos. A escolha do tipo e aparência do objeto é livre (podem inclusive criar objetos de diferentes tipos). A posição de um alvo deve ser inicializada no construtor, e armazenada no objeto.
2. Instancie uma lista de alvos (pelo menos dois) espalhados pelo fundo do mar.
3. Crie a classe MyTorpedo, que representa um torpedo. Esta classe deve armazenar a sua posição e orientação, e uma referência a um alvo (inicialmente não atribuída). A representação de um torpedo deverá ser construída com (ver Figura 2):
  - um cilindro e duas (semi-) esferas para o corpo principal;
  - dois trapézios para as "barbatanas" da parte traseira (cada um atravessa o corpo de um lado ao outro);

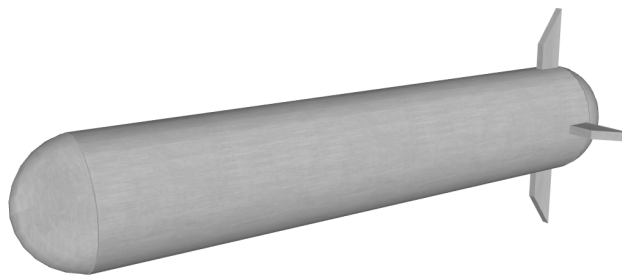


Figura 2: Vista de lado do torpedo; comprimento aproximado: uma unidade

4. Implemente o lançamento de um torpedo dirigido a um alvo, despoletado quando o utilizador pressionar a tecla "F", e de forma a executar o seguinte:
  - Criar um novo torpedo, e posicioná-lo na parte de baixo do submarino, ao centro.
  - Associar o torpedo ao primeiro alvo da lista ("lock target")
  - Iniciar uma animação para o torpedo desde a sua posição inicial até ao seu alvo, segundo uma curva de Bézier (detalhes abaixo); a duração da animação deve ser determinada em função da distância em linha reta entre a posição inicial e final (sugere-se uma relação de uma unidade por segundo, logo se a distância for de 5 unidades, a animação deve ser de 5 segundos)
  - Quando o torpedo chegar ao alvo, ambos devem explodir (a representação desta explosão é livre) e desaparecer

De cada vez que o utilizador pressionar a tecla "F", um novo torpedo deve ser gerado e apontado ao próximo alvo da lista de alvos.

**Detalhes do trajeto em curva de Bézier:** O trajeto do torpedo desde a sua posição inicial até à posição do alvo deve ser implementado aplicando a representação paramétrica das curvas de Bézier (ver slide 27 das aulas teóricas) para calcular a nova posição do torpedo a cada atualização, durante o período de animação.

Neste contexto:

- $P_1$  será a posição inicial do torpedo,
- $P_2$  será um ponto a 6 unidades de distância da posição inicial, na direção da frente do submarino
- $P_3$  será um ponto 3 unidades acima da posição do alvo, na vertical
- $P_4$  será a posição do alvo
- O valor de  $t$  deve ser calculado em cada instante da atualização da animação, de forma a ser 0 no início da animação, e 1 no final da duração da animação (determinada como indicado acima)

5. A orientação do torpedo pode ser calculada de forma simplificada calculando o vetor resultante da diferença entre a posição atual e a anterior (ou a próxima) do torpedo.

(6.5  ) (6.5  )

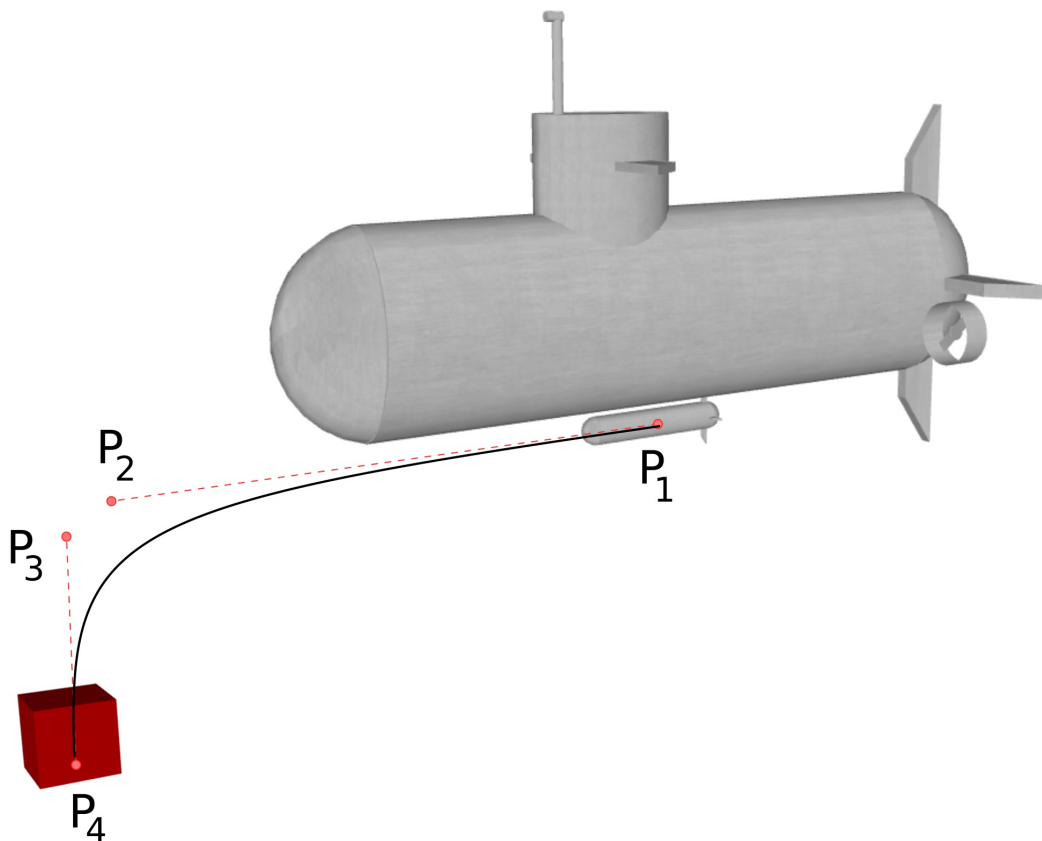


Figura 3: Trajetória do torpedo ao alvo seguindo uma curva de Bézier  
(dimensões indicativas, não necessariamente à escala)

## Notas sobre a avaliação do trabalho:

O enunciado incorpora, em cada alínea, a sua classificação máxima, correspondendo esta a um ótimo desenvolvimento, de acordo com os critérios seguintes, e que cumpra com todas as funcionalidades enunciadas.

Desenvolvimentos além dos que são pedidos apenas serão considerados no critério “criatividade” referido abaixo, não compensando nunca a falta ou falha de funcionalidades expressamente referidas no enunciado.

Para efeitos de avaliação do trabalho e tendo em atenção as cotações mencionadas anteriormente, serão considerados os seguintes critérios:

- Software (2 valores):
  - Estruturação e eficiência das rotinas mais críticas em termos de tempo de cálculo,
  - Criatividade e qualidade da Interação (intuitividade, coerência, facilidade de utilização);
- Controlo do submarino (3 valores)
- GUI (3 valores)
- Modelação do submarino (4 valores)
- Animação do submarino (4 valores)
- Torpedos e alvos (4 valores)

De acordo com a formulação constante na ficha de disciplina, a avaliação deste trabalho conta para a classificação final com um peso de:



$50\% * 40\% = 20\%$  da nota final.

### **Discussão do trabalho**

A avaliação do trabalho decorrerá durante a última aula prática, e consistirá numa apresentação/discussão de 10 minutos de cada grupo com o respetivo docente das aulas práticas.

# Checklist

Até ao final do trabalho deverá submeter as seguintes imagens e versões do código via Moodle, respeitando estritamente a regra dos nomes, bem como o ficheiro ident.txt com a identificação dos membros do grupo:

-  Imagens (6): 1.5, 3.2, 4.1, 4.3, 5.5, 6.5  
(nomes do tipo "CGFImage-tp6-TtGgg-x.y.png" )
-  Código em arquivo zip (6): 1.5, 2.1, 3.2, 4.3, 5.5, 6.5  
(nomes do tipo "CGFCode-tp6-TtGgg-x.y.zip")