Aula 7

- Cálculo do versor normal a cada um dos triângulos de uma malha
- O Modelo de Iluminação de Phong
- Cálculo das componentes do modelo de iluminação – CPU
- Cálculo das componentes do modelo de iluminação – GPU – *"Per Vertex Shading"*
- Cálculo das componentes do modelo de iluminação – GPU – *"Per Fragment Shading"*

1.1 O Modelo de Iluminação de Phong — Implementação faseada

Analise o exemplo incompleto **WebGL_example_23.html**.

O objetivo deste exemplo é utilizar um foco de luz pontual para iluminar um modelo 3D, usando o modelo de iluminação de Phong.

A implementação do modelo de iluminação de Phong exige a implementação de funcionalidades adicionais, relativamente ao código dos exemplos anteriores.

Identifique as principais alterações relativamente aos exemplos anteriores:

- Os modelos representados em ficheiro são apenas descritos pelas **coordenadas** dos vértices definindo a malha de triângulos, deixando de haver informação de cor associada a cada vértice – ver os vários exemplos.

- Por consequência, foi alterada a função que faz a leitura de um modelo a partir de ficheiro.

- Como o cálculo da iluminação é efetuado na CPU, não há qualquer alteração ao código dos *shaders*. A cor calculada para cada vértice da malha é um *input* dos *shaders*.

- Adição de **variáveis globais** que armazenam as características do foco de luz pontual e do material definindo o modelo.

- Definição de funções adicionais no ficheiro **maths.js** para operar com pontos e vetores 3D – verificar quais são.

- Definição de uma função adicional (incompleta) no ficheiro **models.js**, para associar um vetor normal unitário a cada vértice da malha: o vetor normal ao triângulo a que o vértice pertence

- Note que estas novas funcionalidades não produzem, ainda, qualquer efeito: é necessário completar o código da função **drawModel()** para que isso aconteça

Questões:

- Que modelo está a ser representado inicialmente? Qual é a sua cor?
- Está a ser efetuado algum cálculo de iluminação?

Tarefas:

- Analise o código, no ficheiro **maths.js**, das funções adicionais para operar com pontos e vetores.
- No ficheiro **models.js**, complete a função que associa a cada vértice da malha o vetor normal unitário do respetivo triângulo:

computeVertexNormals(coordsArray, normalsArray)

Questões:

- Onde está situado o foco de luz pontual que ilumina a cena? Quais são as suas características?
- Qual é a cor atribuída ao modelo? Quais são as propriedades do material que lhe está associado?

Tarefas:

- Complete, de modo faseado, atendendo aos comentários incluídos no código, e efetuando sucessivos testes, o modelo de iluminação de Phong:
    - Iluminação ambiente
    - Reflexão difusa
    - Reflexão especular
- Teste a sua implementação usando os modelos, com diferentes níveis de detalhe, disponibilizados. Analise as características da iluminação obtida (p.ex., usando animação e a representação em perspetiva). Efetue também a visualização em modo ***wireframe***.

Sugestões:

Acrescente funcionalidades que permitam:

- Alterar a cor do foco.
- Alterar a posição espacial do foco e a sua distância à cena (foco direcional vs. foco pontual).
- Alterar os coeficientes de reflexão associados ao material definindo o modelo.

1.2 O Modelo de Iluminação de Phong — GPU — "Per Vertex Shading"

Analise o exemplo **WebGL_example_23_GPU_per_vertex.html**.

Tal como no exemplo anterior, é utilizado um foco de luz pontual para iluminar um modelo 3D, usando o modelo de iluminação de Phong.

Mas, agora, o cálculo da iluminação para cada um dos vértices da malha é efetuado na **GPU**, obrigando a modificações significativas, relativamente ao código do exemplo anterior.

Identifique as principais alterações relativamente ao exemplo anterior:

- Para cada vértice, o cálculo da iluminação é efetuado pelo ***vertex-shader***, que sofreu alterações significativas na sua estrutura e nos seus argumentos.

- A cor calculada para cada vértice da malha é, tal como anteriormente, o *input* do *fragment-shader*, que não foi modificado.

- As funções **drawModel()**, **drawScene()** e **initBuffers()** estabelecem os diferentes argumentos do *vertex-shader*.

Questões:

- Que modelo está a ser representado inicialmente? Qual é a sua cor?

- Está a ser efetuado algum cálculo de iluminação?

Tarefas:

- Analise o código do ***vertex-shader***, e verifique as diferenças / semelhanças com a função **drawModel()** do exemplo anterior.

- Analise o código da função **drawModel()**, que é mais simples do que no exemplo anterior: calcula a matriz global de transformação e estabelece os argumentos do *vertex-shader* que são próprios do modelo a ser representado.

- Analise o código da função **drawScene()**: calcula a matriz de projeção e estabelece os argumentos do *vertex-shader* que são próprios da cena a ser representada.

Sugestões:

Acrescente funcionalidades que permitam:

- Alterar a cor do foco, a sua posição espacial e a sua distância à cena (foco direcional vs. foco pontual).

- Alterar os coeficientes de reflexão associados ao material definindo o modelo.

1.3 O Modelo de Iluminação de Phong — GPU — "Per Fragment Shading"

Analise o exemplo **WebGL_example_23_GPU_per_fragment.html**.

Tal como no exemplo anterior, é utilizado um foco de luz pontual para iluminar um modelo 3D, usando o modelo de iluminação de Phong.

Mas, agora, o cálculo da iluminação é efetuado pelo ***fragment-shader***, o que implica modificações significativas no código do ***vertex-shader*** e do ***fragment-shader***.

No entanto, relativamente ao exemplo anterior, **não foi necessário efetuar qualquer alteração do código JavaScript**.

Identifique as principais alterações relativamente ao exemplo anterior:

- Para cada vértice, o ***vertex-shader*** calcula o vetor normal, o vetor na direção do foco de luz e o vetor na direção do observador.
- Esses vetores são passados ao ***fragment-shader*** onde serão interpolados.
- No ***fragment-shader*** é usado o modelo de iluminação para calcular a cor a atribuir a cada pixel.

Questões:

- Que modelo está a ser representado inicialmente? É composto por quantos triângulos? Qual é a sua cor?
- Quais são as características do foco de luz? É um foco de luz direcional ou pontual?

Tarefas:

- Para o mesmo modelo e o mesmo foco de luz, **compare** os efeitos de iluminação do exemplo anterior com os efeitos de iluminação deste exemplo. Qual deles reproduz melhor a componente especular? Por que razão?
- Analise o código do ***vertex-shader*** e do ***fragment-shader***, e verifique as diferenças / semelhanças com o código do ***vertex-shader*** do exemplo anterior.

Sugestões:

Acrescente funcionalidades que permitam:

- Alterar a cor do foco, a sua posição espacial e a sua distância à cena (foco direcional vs. foco pontual).
- Alterar os coeficientes de reflexão associados ao material definindo o modelo.