

Computação Gráfica (L.EIC)

Tópico 3

Iluminação e Materiais



Objetivos

Manipular as componentes envolvidas na iluminação, nomeadamente as luzes, as normais e as componentes de reflexão dos materiais.

Preparação do Ambiente de Trabalho

Para este trabalho deve usar o código de base que é fornecido no Moodle para esta aula. Será pedido num dos exercícios para incluir no código base os objetos que criou no trabalho 2 (tp2), nomeadamente o **MyTangram** e **MyUnitCube** (assume-se que utilizou a nomenclatura definida no enunciado do trabalho 2).

Trabalho prático

À semelhança do trabalho anterior, será necessário fazerem capturas de ecrã em alguns pontos do enunciado, bem como assinalar versões do código na *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). No final deve ser submetido um zip no Moodle com a última versão.

Experiências

A cena criada no código base contém um plano (**MyPlane**), pouco visível e com cor avermelhada, e duas luzes desligadas. Na interface gráfica no canto superior direito encontram uma série de controlos para a geometria, materiais e luzes, que devem usar para os pontos seguintes.

Ambiente

- Embora as luzes estejam desativadas, o plano é visível devido à **componente ambiente** do material e do valor da **iluminação ambiente global** da cena. Altere entre os objetos no dropdown 'Selected Object' para verificar a ausência de definição de arestas, uma vez que a mesma cor é atribuída às superfícies independentemente da sua orientação.
- Crie um controlo na interface que permita variar a intensidade da iluminação ambiente global da cena (definida na função **initLights**), usando um **slider**, e verifique as diferenças na cena.
Nota: Verifique a documentação de **CGScene.setGlobalAmbientLight()**.

Difusa

- Reinicie a cena, e mude o material aplicado para 'Red Diffuse', tornando o plano invisível (este material não tem componente ambiente). Ative apenas a luz 0, ativando a checkbox 'Enabled' da mesma. O eixo e as luzes tornam-se mais visíveis, mas o plano mantém-se invisível. Verifique a visibilidade dos outros objetos disponíveis no dropdown nestas condições.
- Com o plano selecionado, varie a posição em Z da luz 0, utilizando os **sliders** em 'Light 0 / Position', de forma a que a mesma passe para a frente do plano e este seja iluminado.
- Coloque a luz 0 na posição [2, 2, 1] e rode a câmara, verificando que o gradiente de cor nos objetos não varia com a posição do observador.

Especular

- Reiniciando a cena para voltar às configurações iniciais, ative a luz 1, e mude o material aplicado no dropdown para 'Red Specular', que criará um gradiente vermelho no plano. Rode ligeiramente a câmara para ver a variação na cor no gradiente, que está dependente das **componentes especulares** do material aplicado e da luz ativa, assim como da posição da luz e da câmara relativamente às superfícies. Poderão mudar o material para 'Red Diffuse', para comparação.
- Mude o material para 'Custom' e mude as cores das componentes ambiente e difusa para preto (#000000) e a componente especular para amarelo (#ffff00). Deverá ver o plano com uma pequeno reflexo amarelo. Rode a câmara de forma a que o centro da **reflexão** esteja aproximadamente no centro do quadrado.
- Varie o valor de 'Shininess' do material 'Custom', e verifique as diferenças na **intensidade** e amplitude da reflexão especular.
- Varie a complexidade do plano, e verifique a diferença no aspeto da reflexão especular.

Combinação de componentes de iluminação

- Reinicie a cena, mude o material aplicado no dropdown para 'Custom', e verifique as cores das componentes deste material no grupo 'Custom Material'. O plano deverá ter uma cor azul escura esbatida.
- Coloque a luz 0 na posição [1, 1, 1], e ative a luz 0, que criará um gradiente de vermelho a azul no plano.
- Varie a posição em Z da luz 0, utilizando os **sliders** em 'Light 0 / Position' para verificar a variação no gradiente do plano.
- Coloque a luz na posição [0, 0, 0.2], e varie a complexidade do objeto, aproximando a câmara para observar melhor os detalhes no plano.






Atenuação

- Reinicie a cena, aplique novamente o material 'Custom', e ative a luz 1, colocando-a na posição [0, 0, 0.2]. Altere o valor de Z para afastar a luz do plano. O plano deverá parecer mais iluminado, embora a luz esteja mais afastada.
- Reduza a componente de **atenuação constante** da luz 1 para 0.5, para uma variação de intensidade mais consistente da reflexão com a distância.
- Experimente diferentes combinações das três componentes de atenuação e observe as diferenças na iluminação do objeto com a variação da distância.

Exercícios

Parte 1 - Iluminação e materiais do Tangram

Acrescente ao código base os ficheiros das classes **MyTangram** (e as classes utilizadas para construir o tangram) e **MyUnitCube**, tal como criados na aula anterior (aula prática 2), colocando os ficheiros Javascript respetivos na pasta deste exercício, e acrescentando a importação dos mesmos na classe **MyScene**.

- Crie uma instância de **MyTangram** e outra de **MyUnitCube** na função **init()** da cena, e acrescente-os à lista de objetos disponíveis que é apresentada na GUI (verifique como é feito para os objetos existentes).
- Faça algumas das experiências anteriores de iluminação com estes novos objetos, e repare que existem provavelmente inconsistências com o esperado, uma vez que não foram declaradas as **normais** para estes objetos.
- Declare as normais para os diferentes objetos, na função **initBuffers()** das suas classes, começando pelo cubo. Poderá ter de repetir vértices que, sendo partilhados por faces com orientações diferentes, terão de ter normais diferentes dependendo da face em que estão a ser usados (nomeadamente no cubo, e nas faces traseiras das peças do Tangram).
- Crie um material com cor semelhante a madeira, com baixa componente especular, e adicione à lista de materiais disponível (siga o exemplo da 'Red Diffuse' ou outro, na função **initMaterials** da cena). Teste com o cubo.
 (1) 
- Deixe na classe MyTangram** crie um material para cada uma das peças com elevada componente especular, e com cor de acordo com a figura fornecida (e aplique-o à peça respetiva).
 (2)  (1) 

Parte 2 - Desenho de um prisma

Acrescente ao projeto um novo ficheiro chamado **MyPrism.js** (por exemplo, fazendo uma cópia do **MyQuad.js**). Altere o construtor de **MyPrism** para conter as variáveis **slices** e **stacks**:

```
constructor(scene, slices, stacks)
{
    super(scene);

    this.slices = slices;
    this.stacks = stacks;

    this.initBuffers();
}
```

Complete a classe **MyPrism** para que esta consiga desenhar um prisma com um **número variável de "lados"** e de **"andares"** (**slices** e **stacks**, parâmetros já incluídos no construtor da classe, ver Figura 1) como se estivesse contido (inscrito) num cilindro de raio igual a uma unidade, base coincidente com o plano XY e centrada na origem, e com comprimento também unitário, em Z. O cilindro pode ser aberto nas extremidades (sem tampas).

NOTA: todas as normais deverão ser normalizadas.

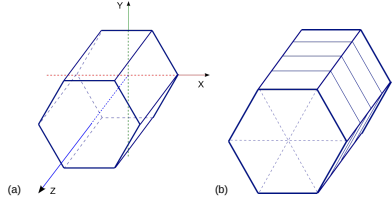


Figura 1: Prisma (a) de seis lados (**slices**) e um andar (**stack**) de seis lados e quatro andares. (a escala não é a real)

- Numa primeira versão do prisma, considere que o cilindro tem apenas um "andar" (tal como o exemplo na Figura 1 (a)). Note que, para cada face, os vetores normais dos seus vértices devem ser perpendiculares a essa face (Figura 2). **Poderá por isso ter de definir o mesmo vértice mais do que uma vez, na lista de vértices, de forma a atribuir-lhe normais diferentes.**
Comente a eventualidade de, com esta definição de normais, a iluminação calculada ser semelhante à calculada com "Constant Shading"

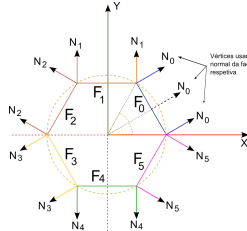


Figura 2: Ilustração das normais a atribuir a cada vértice no caso de seis lados. As normais serão equivalentes nos vértices de todas as **stacks**

- Na classe **MyScene**, importe **MyPrism.js** e esconda todos os elementos da cena atualmente visualizados, exceto os eixos. Nesta mesma classe crie na função **init** uma instância de **MyPrism**, com 8 lados, representando uma coluna vertical com as dimensões que achar apropriadas. Na função **display** de **MyScene** deve invocar a função **display** do objeto criado. Poderá ter de usar rotações, escalas ou translações antes de desenhar o prisma.
- Na alínea 1) implementou vários lados e apenas um andar. Ajuste a implementação para suportar o desenho de vários andares (**stacks**) como no exemplo na Figura 1 (b), e garanta que a coluna desenhada em 2) é agora desenhada com 20 andares.

(3) (2) ()

Parte 3 - Superfície Cilíndrica - Aplicação de Gouraud shading

- Crie uma nova classe **MyCylinder** fazendo uma cópia do ficheiro **MyPrism.js** desenvolvido no exercício anterior. Não se esqueça de alterar o nome do novo ficheiro, da classe nele contida. Altere as normais do método **initBuffers** da classe **MyCylinder**, de forma a que a normal de cada vértice seja perpendicular à superfície do cilindro perfeito em que o prisma original está inscrito. Ou seja, cada vértice que seja usado em dois lados adjacentes, terá sempre a mesma normal (Figura 3).
NOTA: todas as normais deverão ser normalizadas.

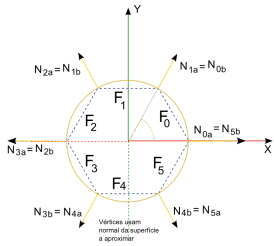







Figura 3: Ilustração das normais a atribuir a cada vértice no caso de um cilindro aproximado por seis lados.

- Simplifique a lista de vértices e de normais, de modo a eliminar duplicados (implica obviamente passar a referir o mesmo vértice mais do que uma vez na lista de índices).
- Na classe **MyScene**, importe agora o ficheiro **MyCylinder.js**. Nesta mesma classe crie uma instância de **MyCylinder** na função **init**, com **8 slices**, **20 stacks**. Na função **display** de **MyScene** deve invocar a função **display** do objeto criado.
 (4)  (3) 
- Confirme que, com esta metodologia, as transições de de iluminação nas arestas são suavizadas (smoothed), tornando-as menos evidentes, e dando uma aparência curva à superfície.

Checklist

Até ao final do trabalho deverá ter criado as seguintes imagens e **commits** do código, **respeitando estritamente a regra dos nomes**:

-  **Imagens (4):** 1, 2, 3, 4 (nomes do tipo "cg-t-turma>g<grupo>-tp3-n.png")
-  **GIT Commits/Tags (3):** 1, 2, 3 (Git Tag correspondente: "tp3-1")

Deve também submeter no final um **zip** no **moodle** com o nome no formato "cg-t-turma>g<grupo>-tp3.zip"