

Computação Gráfica (MIEIC)

Tópico 4


Aplicação de texturas


Objetivos

- Definir coordenadas de textura de forma adequada.
- Explorar os diferentes modos de aplicação de textura.
- Combinar o uso de materiais com texturas para obter uma aparência realista.

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, capturar uma imagem da aplicação para disco (p.ex. usando Alt-PrtScr em Windows ou Cmd-Shift-3 em Mac OS X para capturar para a clipboard e depois gravar para ficheiro num utilitário de gestão de imagens à escolha). No final de cada aula, devem renomear as imagens para o formato "**ex4-t<turma>g<grupo>-n.png**", em que **turma** e **grupo** corresponde ao número de turma e grupo definido no ficheiro de grupos TP, e **n** corresponde ao número fornecido no exercício (p.ex. "**ex4-t1g01-1.png**").

Nas tarefas assinaladas com o ícone  (código), devem criar um ficheiro **.zip da pasta que contém o vosso código (tipicamente na pasta 'ex4', se tiverem código noutras pastas incluam-no também)**, e nomeá-lo como "**ex4-t<turma>g<grupo>-n.zip**", (com turma, grupo e n identificados tal como descrito acima "**ex4-t1g01-1.zip**").

No final (ou ao longo do trabalho), um dos elementos deverá submeter os ficheiros via Moodle, através do link disponibilizado para o efeito. Bastará apenas um elemento do grupo submeter o trabalho.

Preparação do Ambiente de Trabalho

Devem descarregar o código disponibilizado para este trabalho no Moodle, e colocar a pasta **ex4** contida no ficheiro .zip ao mesmo nível dos trabalhos anteriores.

Se desenvolveu a classe **MyUnitCubeQuad** como o exercício extra da aula prática 2 (**ex2**), poderá copiar o ficheiro dessa classe para a pasta desta aula. A classe **MyQuad** utilizada para os planos do cubo composto é fornecida no código base desta aula prática, pelo que poderá optar por usar esse ficheiro (verifique se as classes são consistentes e compatíveis).

1. Aplicação de texturas

O mapeamento de texturas é uma forma de atribuir informação armazenada em formato *bitmap* a diferentes zonas das superfícies 3D desenhadas. Um dos seus usos mais comum é o de mapear partes ou a totalidade de uma imagem a uma geometria, de forma a acrescentar detalhe visual sem aumentar o número de vértices e sem acrescentar complexidade à geometria (outros tipos de

mapeamento incluem, por exemplo, *bump mapping* e *normal mapping*, mas que não serão explorados neste trabalho).

No contexto de OpenGL/WebGL, uma textura de duas dimensões pode resultar do carregamento de uma imagem *bitmap*, e que é carregada para um buffer, que posteriormente pode ser acessado usando duas dimensões vulgarmente identificadas como *s* e *t* (ou noutros contextos como *u* e *v*), e cujas coordenadas são normalizadas entre 0 e 1 (ver fig. 1).

Nota: É importante reparar que a representação dos eixos de coordenadas de textura são apresentadas de forma diferente entre as aulas teóricas e práticas. Especificamente, a origem (0,0) corresponde ao *canto inferior esquerdo* na aula teórica, pois o *loader* de texturas em **OpenGL** inicia nesse ponto. No entanto, no contexto das aulas práticas (**WebGL**), a origem corresponde ao *canto superior esquerdo*.

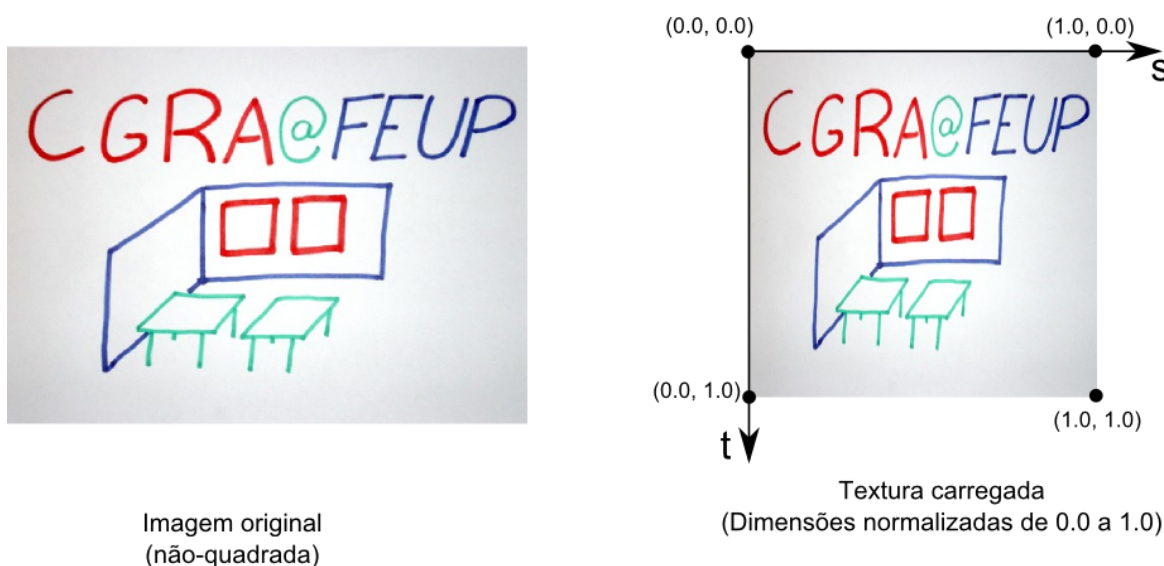


Figura 1: Imagem e correspondente textura carregada

Uma textura previamente carregada pode ser aplicada a uma dada geometria - no caso mais básico, um triângulo - fazendo o mapeamento entre os vértices da geometria e os pontos da imagem que lhes estarão associados, definindo para cada vértice uma coordenada de textura (ver **fig. 2, a)** e **b)**).

Conceptualmente, podemos considerar que estamos a definir o "recorte da imagem" que será aplicado ao triângulo em questão, sendo que caso o "recorte" não tenha as mesmas proporções do triângulo original, a imagem será distorcida de acordo (ver fig. 2, c) e d)).

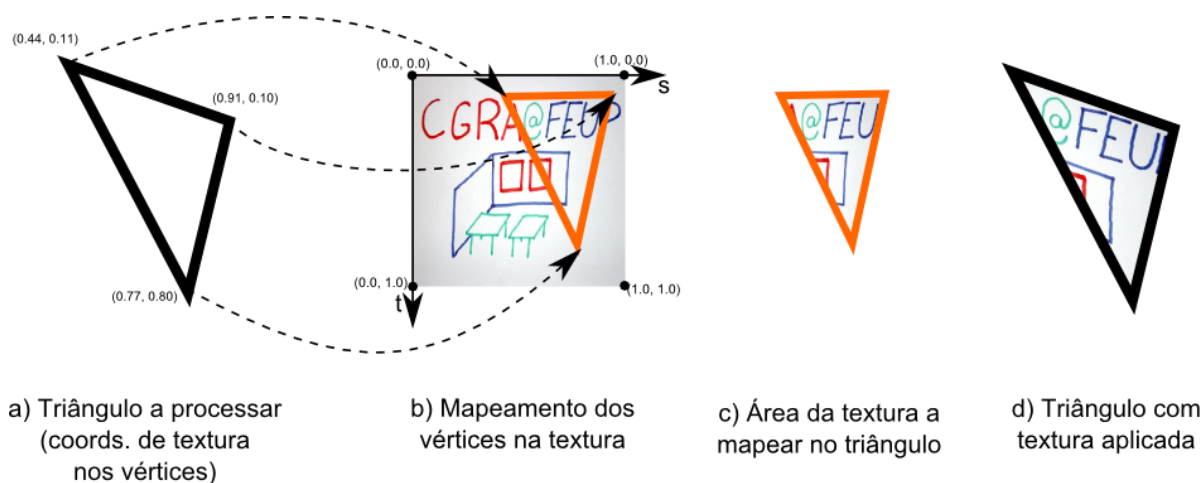


Figura 2: Mapeamento entre triângulo e textura definindo coordenadas de textura por vértice.

2. Modos de Wrapping de Texturas

No exemplo explorado até agora, as coordenadas de textura associadas a cada vértice encontram-se na gama normalizada de 0.0 a 1.0. No entanto, é possível indicar valores fora dessa gama, quando pretendemos, por exemplo, ter várias repetições da mesma imagem num polígono, ou mapear a totalidade da imagem apenas numa parte do polígono.

A forma como os valores fora da gama $[0..1]$ são utilizados na aplicação de uma textura é controlada definindo o modo de *wrapping*. Os modos de *wrapping* suportados variam um pouco entre versões de OpenGL, no caso do WebGL os modos possíveis são 'REPEAT', 'CLAMP_TO_EDGE' e 'MIRRORED_REPEAT'. Na figura 3 estão ilustrados alguns exemplos de como manipular as coordenadas de textura em cada modo para obter diferentes efeitos. Note que o modo de *wrapping* pode ser diferente nas duas dimensões *s* e *t*.

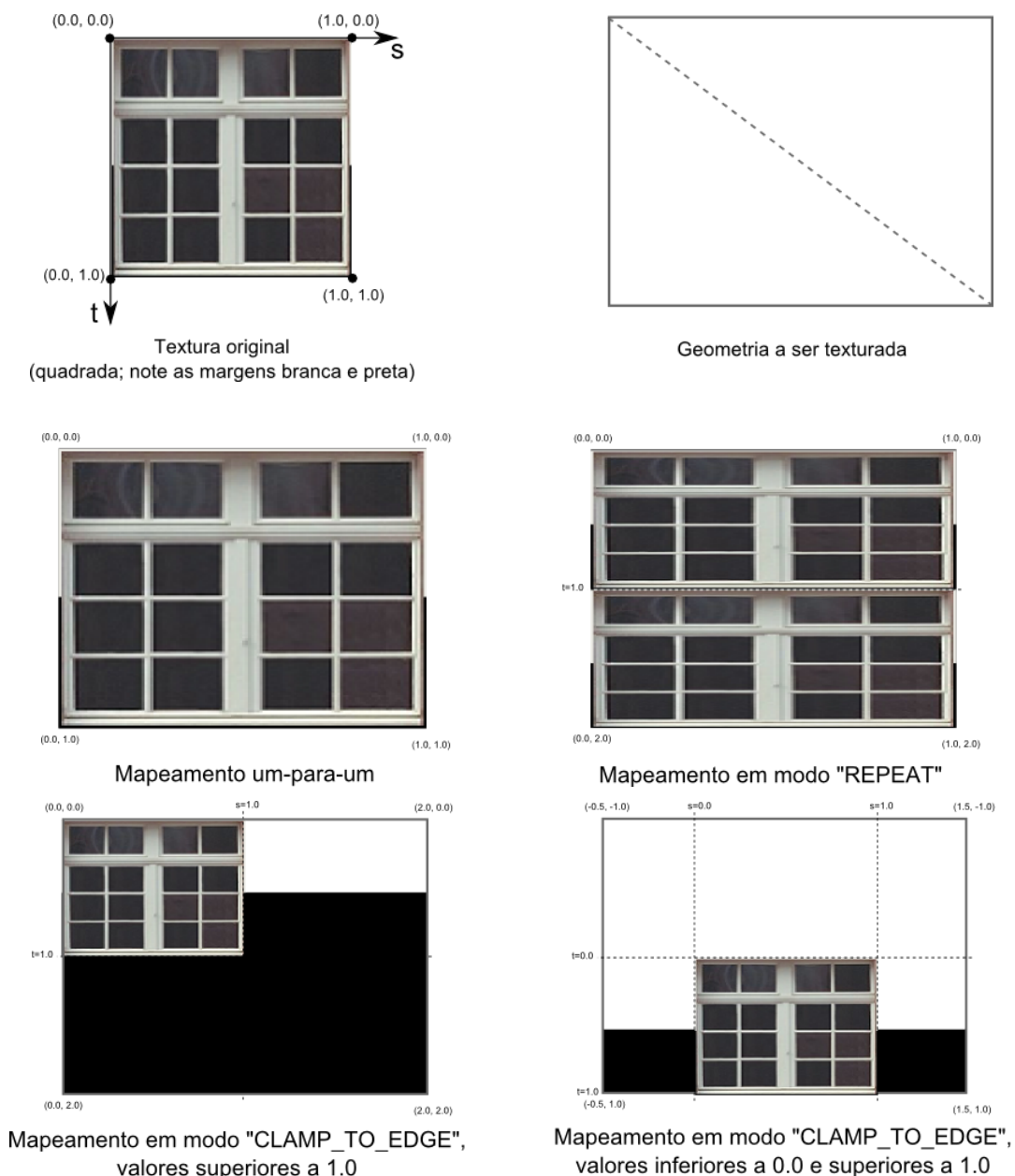


Figura 3: Aplicação de uma textura utilizando repetição ou *clamping*.

Note como no modo de 'CLAMP_TO_EDGE' as margens da imagem são estendidas ao longo das zonas de coordenadas fora da gama $[0..1]$

Experiências

Pretende-se nesta aula explorar a definição de coordenadas de textura em objetos e os diferentes modos de wrapping para a aplicação de diferentes texturas.

A cena fornecida contém um objeto do tipo **MyQuad** ao qual foi aplicado um material do tipo **CGFappearance** chamado **quadMaterial**. A interface contém um menu *dropdown* para a escolha de texturas (começando sem seleção), dois *dropdowns* para a seleção de modos de wrapping, e *sliders* para controlar as coordenadas de textura associadas aos quatro cantos do retângulo.

1. Selecione a textura 'Board' na interface. Com o modo de wrap das coordenadas S e T em 'Repeat', altere os valores das coordenadas de textura de forma a que obtenha três colunas e duas linhas da imagem no objeto.
2. Reinicie a cena, e selecione a textura 'Floor' na interface. Mantendo o modo de wrap em 'Repeat', altere os valores das coordenadas de textura de forma a que a imagem seja invertida na vertical.
3. Altere o modo de wrap das coordenadas S e T para 'Clamp to Edge' e veja as diferenças no mapeamento da textura.
4. Reinicie a cena, e selecione a textura 'Window' na interface. Com o modo de wrap das coordenadas S e T em 'Clamp to Edge', altere os valores das coordenadas de textura de forma a que a janela apareça centrada na geometria, ocupando metade da altura e largura totais, como mostrado na figura 5.

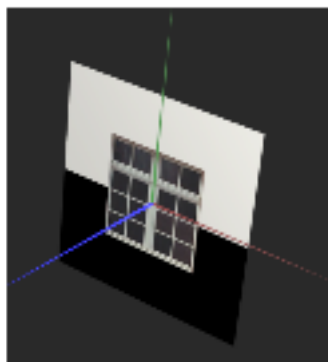


Figura 5: Janela centrada no objeto **MyQuad**.

5. Experimente alternar os modos de wrapping para S e T e observe as diferenças.

Exercícios

Inclua na pasta do código fornecido os ficheiros das classes **MyTangram** e das classes de todas as peças. Crie uma checkbox que permita esconder o objeto **MyQuad** e respectivo material **quadMaterial** de forma a que a cena fique vazia.

Aplicação de texturas ao Tangram

1. Crie um novo material na inicialização a ser aplicado, na função *display* da classe **MyTangram**, no objeto de **MyDiamond** (caso não tenha criado na TP3). Defina como textura desse material a imagem 'tangram.png' (ver exemplos no código).
2. Defina as coordenadas de textura do **MyDiamond** de forma a que as arestas da peça do losango na imagem coincidam com as arestas do objeto. Para ajudar no processo de determinação das coordenadas de textura a atribuir a cada vértice, sugere-se que abra uma cópia de 'tangram.png' num editor de imagem para anotar os eixos S e T tal como na figura 1 deste enunciado, identifique os vértices do losango nessa figura, e determine quais as suas coordenadas nesse espaço S, T (valores entre 0.0 e 1.0).

As coordenadas de textura são definidas criando na função **initBuffers** do objeto um *array* adicional **this.texCoords** com um par de coordenadas para cada vértice previamente declarado no array **this.vertices**:

```
this.texCoords=[
    s0, t0,
    s2, t2,
    ...
    sn, tn
];
```

3. Repita os dois passos anteriores para cada uma das outras peças do Tangram, de forma a que

cada peça tenha mapeada a sua representação da imagem. (1 ) (1 )



Aplicação de texturas a um cubo composto por planos

4. Crie uma nova classe **MyUnitCubeQuad**, que define um novo cubo unitário utilizando um objeto do tipo **MyQuad**, desenhado várias vezes para definir as faces. Utilize as funções de transformações geométricas para desenhar as seis faces, na função **display()** de **MyUnitCubeQuad**.

Nota: Se já criou esta classe no exercício extra da aula prática 2 (**ex2**), pode copiar o ficheiro correspondente para a pasta desta aula (**ex4**).



5. Aplique a textura 'mineSide.png' às faces laterais do **MyUnitCubeQuad**, e as texturas 'mineTop.png' e 'mineBottom.png' às faces de cima e de baixo, respetivamente.
6. Repare como as texturas ficam pouco definidas. Isso deve-se ao facto de terem originalmente dimensões de 16x16 pixels, mas na verdade estarem a cobrir uma área de desenho muito superior. Por omissão, nestes casos é feita uma **interpolação linear** das cores (**LINEAR FILTERING**, ver filtragem nos slides da teórica).

Encontre no código de exemplo o comando que permite alterar o tipo de filtragem usado (comentado originalmente na função **display()** da cena). Use-o para as texturas do cubo para atingir o efeito pretendido, ativando esse modo depois de ativar a textura e antes de desenhar

as faces a afetar. (2 ) (2 )

Checklist

Até ao final do trabalho deverá submeter as seguintes imagens e versões do código via Moodle, **respeitando estritamente a regra dos nomes**:

-  Imagens (2): 1, 2 (nomes do tipo "ex4-t<turma>g<grupo>-n.png")
-  Código em arquivo zip (2): 1, 2 (nomes do tipo "ex4-t<turma>g<grupo>-n.zip")