

Textura

Introdução

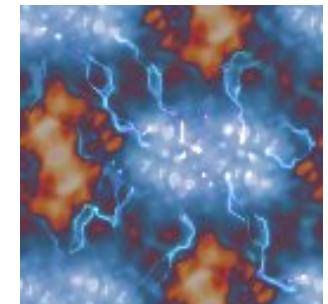
- A técnica de pegar uma textura qualquer e aplicar sobre um objeto é conhecida como **mapeamento de texturas**.
- Uma textura bem elaborada consegue trazer para o cenário do jogo ou animação um pouco mais de realidade. Praticamente todos os jogos ou animações 3D possuem texturas.
- Por exemplo, uma caixa quadrada de madeira em 3D seria apenas um polígono 3D sem a textura da caixa de madeira.



Alguns Tipos de Texturas

■ ***Surface Texture***

- ◆ Um dos tipos de textura mais simples.
- ◆ São basicamente matrizes bidimensionais que são mapeadas pixel a pixel para o objeto.
- ◆ Podem ser geradas por algoritmos ou a partir de imagens criadas previamente.
- ◆ Muito utilizada também como papel de parede podendo replicar a imagem para toda a superfície do objeto.



Alguns Tipos de Texturas

■ *Solid Textures*

- ◆ Enquanto nas texturas de superfície o principal método de obtenção das texturas é uma foto, nas texturas sólidas, elas normalmente são obtidas diretamente por algoritmos.
- ◆ Para gerar texturas mais interessantes são utilizadas algumas funções de ruído, que vão desde um deslocamento para algumas cores até a utilização de fractais.
- ◆ As texturas sólidas também são muito usadas para simular nuvens e fogo, produzindo efeitos muito interessantes.



Alguns Tipos de Texturas

- **Bump Mapping**
- Pode ser traduzido como mapeamento de rugosidade, porém, seu uso não está restrito ao ato de querer dar uma aparência rugosa a uma determinada superfície.
- É também usado quando se deseja obter um efeito áspero, dentado, perfurado, sem que se altere a superfície, dando a ilusão de que o objeto possui tais características.



Mapeamento de Texturas em OpenGL

- O uso de texturas requer a execução de dois passos distintos: a **CARGA** e a **APLICAÇÃO** da textura.
- OpenGL exige que as dimensões das imagens de Texturas sejam potências de 2 e ela não possui nenhuma função para ler textura de arquivo.

Mapeamento de Texturas em OpenGL

- Cada textura que você carrega via OpenGL é necessário identificá-la por um ID.
- Esse número você pode controlar manualmente, ou seja, você será responsável em identificar cada textura nova que você carregar, ou pode fazer o OpenGL gerar esses números automaticamente pela função **glGenTextures()**. Nada garante que esses números serão sequenciais.
- Após gerar o número de identificação para as texturas, você precisa informar ao OpenGL qual será sua textura corrente, ou seja, a que você irá usar.
- A função responsável por essa ligação é **glBindTexture()**.

Exemplo de Carregamento de Texturas

```
// abre arq. da textura usando função da biblioteca  
glaux  
  
AUX_RGBImageRec *imagemTextura = NULL;  
imagemTextura =  
auxDIBImageLoad("surface2.bmp");  
  
/* Habilita o uso de texturas */  
glEnable ( GL_TEXTURE_2D );
```

Carregamento de Texturas

- A função glGenTextures é responsável por gerar o número ID para as texturas. Ela recebe como argumentos a quantidade de IDs que ela deve gerar e onde deverão ser armazenados os IDs gerados.
- Como iremos utilizar apenas uma textura precisamos de um único ID. Nesse caso temos:
 - ◆ 1: quantidade de IDs que a função deverá gerar.
 - ◆ **texture_id**: Variável onde será armazenado os IDs.
- A variável que irá armazenar os IDs é um *array*. Assim fica flexível para utilizar mais texturas.
- Exemplo:

```
/* Define quantas texturas serão usadas no programa */
GLuint texture_id[MAX_NO_TEXTURES]; // vetor com os números das texturas
glGenTextures (1, &texture_id[0]); // 1 = uma textura;
                                // texture_id = vetor que guarda os números das texturas
```

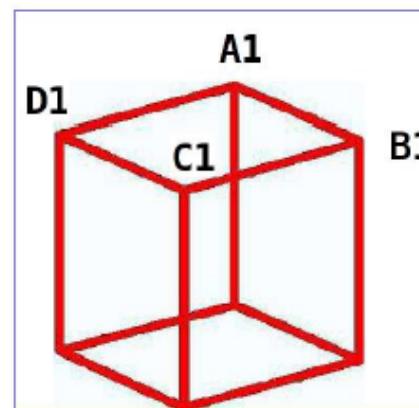
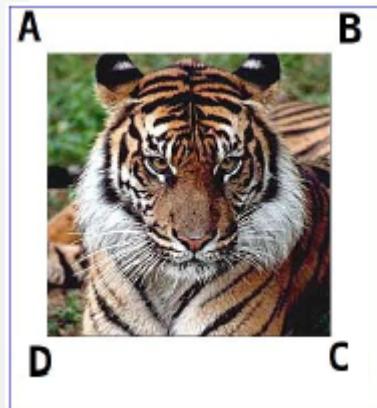
Carregamento de Texturas

- A função glBindTexture é responsável por informar ao OpenGL qual textura iremos usar.
- Recebe como argumentos um *target* e o nome da textura que iremos usar.
- Nesse caso temos:
 - ◆ **Target:** GL_TEXTURE_2D, que define que será usada uma textura 2D (bitmaps)
 - ◆ **Nome / ID da Textura:** texture_id , que define qual textura será usada
- O target pode ter os seguintes argumentos: GL_TEXTURE_1D, GL_TEXTURE_2D ou GL_TEXTURE_3D.
- Exemplo:

```
// Define a textura corrente  
glBindTexture ( GL_TEXTURE_2D, texture_id[0] );
```

Aplicação das Texturas

- Para a aplicação é necessário criar uma relação entre os vértices da textura e os vértices dos polígonos, pelos quais se deseja aplicar a textura escolhida.
- Na figura do lado esquerdo as letras A,B, C e D definem os vértices da textura e os vértices A1, B1, C1 e D1 do lado direito definem os vértices do polígono 3D onde deve ser aplicada a textura.



Aplicação das Texturas

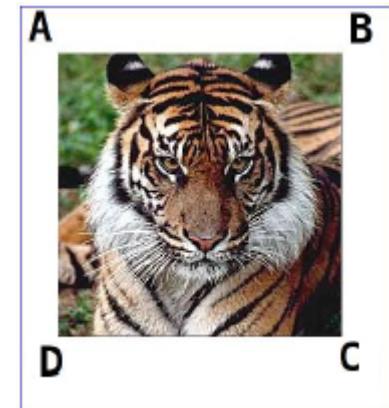
- O processo de mapeamento de texturas em OpenGL consiste em "aplicar" a imagem 2D sobre o polígono 3D de forma que os pontos A, B, C e D sejam encaixados sobre os pontos A1, B1, C1 e D1.
- Para permitir essa correspondência entre a imagem 2D e o polígono 3D usa-se a função `glTexCoord2f()` antes da definição do ponto 3D, ou seja, define-se primeiro as coordenadas da textura e depois as do polígono.
- Exemplo:

```
glTexCoord2f(0.0f, 0.0f); glVertex3f(-1.0f, -1.0f, 1.0f);
glTexCoord2f(1.0f, 0.0f); glVertex3f( 1.0f, -1.0f, 1.0f);
glTexCoord2f(0.0f, 1.0f); glVertex3f(-1.0f, -1.0f, -1.0f);
glTexCoord2f(1.0f, 1.0f); glVertex3f(-1.0f, 1.0f, -1.0f);
```

Aplicação das Texturas

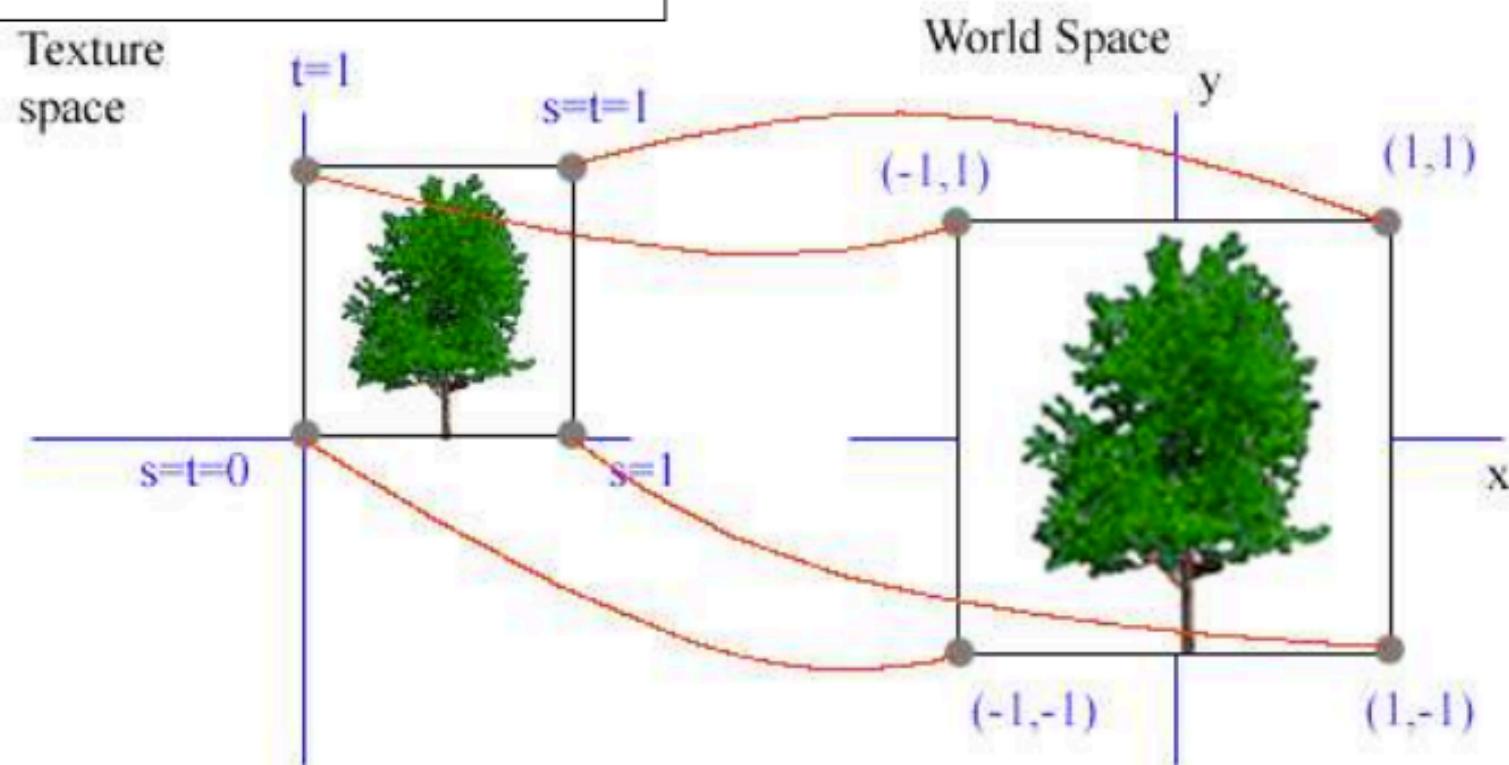
- O sistema de coordenadas da textura tem como **(0,0)** o ponto inferior esquerdo da imagem e como **(1,1)** o ponto superior direito, ou seja, na imagem anterior temos as seguintes coordenadas para os pontos A, B, C e D.

Vértice da textura	Coordenadas
A	0,1
B	1,1
C	1,0
D	0,0



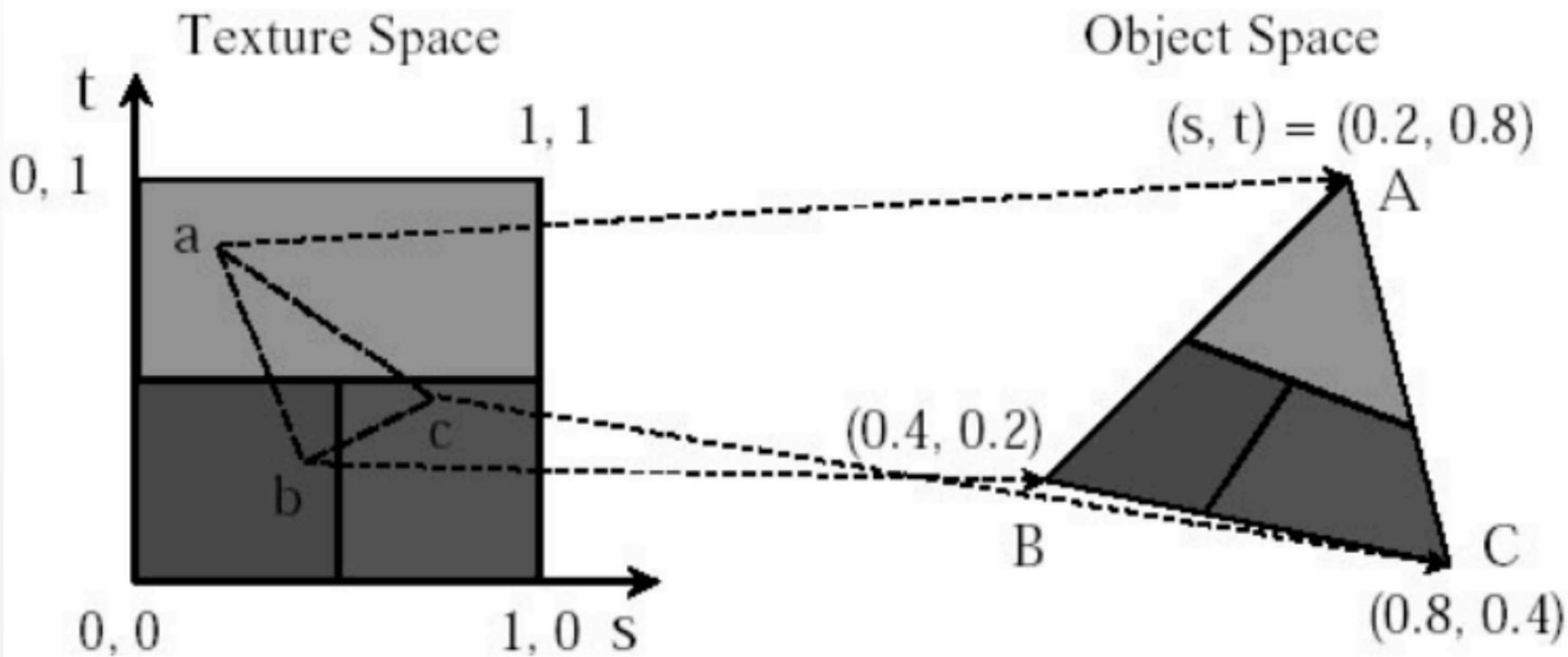
Aplicação das Texturas

```
glBindTexture(GL_TEXTURE_2D,1);
glBegin(GL_QUADS);
    glTexCoord2f(0,0); glVertex3f(-1.0f, -1.0f, 0.0f);
    glTexCoord2f(1,0); glVertex3f( 1.0f, -1.0f, 0.0f);
    glTexCoord2f(1,1); glVertex3f( 1.0f, 1.0f, 0.0f);
    glTexCoord2f(0,1); glVertex3f(-1.0f, 1.0f, 0.0f);
glEnd();
```



Aplicação das Texturas

A escolha de coordenadas no espaço das texturas é "livre".



Aplicação das Texturas

Matriz para Texturas

- Permite realizar transformações geométricas sobre a textura.

```
glMatrixMode(GL_TEXTURE);
glTranslatef(0.5,0,0);
glRotatef(45,0,0,1);

glMatrixMode(GL_MODELVIEW);
glBegin(GL_QUADS);
...
glEnd();
```



Aplicação das Texturas

- Iremos aprender como utilizar texturas BMP.
- Pode ser utilizada a biblioteca glaux.h (Windows), SLD.h (Windows/Linux), texture.h (Windows/Linux), etc.
- No nosso exemplo de hoje será utilizada uma função que lê imagens de arquivo (a função já está implementada para vocês no exemplo).

Filtros

- Caso as dimensões da textura não coincidam com as dimensões do objeto deve-se proceder a uma ampliação (MAGNIFICATION) ou a uma redução (MINIMIZATION) da textura. Neste caso o OpenGL permite considerar dois tipos principais de técnica na redução/aumento da imagem: NEAREST/LINEAR.
- No primeiro (NEAREST) é considerado simplesmente o pixel mais próximo e no segundo (LINEAR) é efetuada uma média com os pixels adjacentes.
- Assim, o LINEAR é mais “perfeito”, mas obviamente mais pesado computacionalmente.

Filtros

- Os seguintes comandos implementam a ampliação e redução LINEAR

```
glTexParameteri(GL_TEXTURE_2D,  
    GL_TEXTURE_MAG_FILTER, GL_LINEAR);
```

```
glTexParameteri(GL_TEXTURE_2D,  
    GL_TEXTURE_MIN_FILTER, GL_LINEAR);
```

Filtros

- Outra propriedade possível de especificar relaciona-se com a repetição da mesma textura na superfície do objeto.
- OpenGL permite distinguir essa repetição nas direções horizontal e vertical.
- Assim, caso deseje efetuar a repetição (REPEAT) da textura na horizontal e na vertical deve-se efetuar (S é a direção horizontal e T vertical):

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S,  
    GL_REPEAT);
```

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,  
    GL_REPEAT);
```

Filtros

- Caso não deseje efetuar a repetição e a textura aplicada seja apenas a original, faz-se:

```
glTexParameteri(GL_TEXTURE_2D,  
                 GL_TEXTURE_WRAP_S,  
                 GL_CLAMP);
```

```
glTexParameteri(GL_TEXTURE_2D,  
                 GL_TEXTURE_WRAP_T,  
                 GL_CLAMP);
```

Filtros

- Exemplo:

```
glTexCoord2f(3.0,0.0); glVertex2f(1.0, 0.0);
glTexCoord2f(3.0, 3.0); glVertex2f(1.0, 1.0);
```

É possível especificar propriedades distintas para a horizontal e vertical. Por exemplo:

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S,
    GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_
    CLAMP);
```

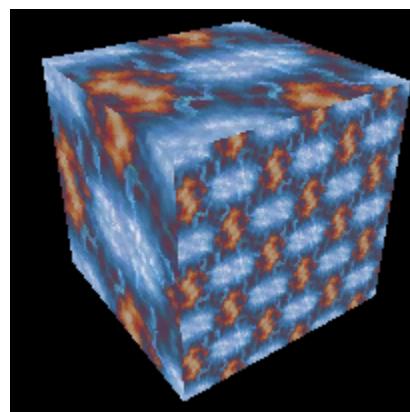
- **Obs: Para as propriedades de repetição terem efeito é necessário que as coordenadas da textura tenham valores superiores a 1.**

Especificando a Textura

- É necessário definir as características de cada imagem que será usada como textura:
`glTexImage2D(target, level, components, width, height, border,
 format, type, *pixels);`
- *target*: deve ser `GL_TEXTURE_2D`;
- *level*: especifica o nível de detalhe (0 é a imagem base);
- *components*: formato interno dos pixels (por ex, `GL_RGB`, `GL_RGBA`);
- *width*, *height*: tamanho da imagem (múltiplos de 2 e mínimo 64x64);
- *border*: borda da imagem (deve ser sempre 0);
- *format* e *type*: indicam o formato que o pixel será carregado na matriz de pixels (geralmente `GL_RGB` ou `GL_RGBA`) e o tipo de dados (`GL_UNSIGNED_BYTE`);
- *pixels*: matriz de pixels da imagem.

Exercício

- Pegue o primeiro exemplo do Moodle e carregue a textura corretamente.
- Faça com que pelo menos um lado do polígono tenha repetição da textura, como no exemplo da figura abaixo:



Mapeamento Automático de Texturas

- Para o mapeamento automático de coordenadas de texturas use a função:

```
void glTexGen{ifd}{v}(GLenum coord,  
                      GLenum pname, TYPE param)
```

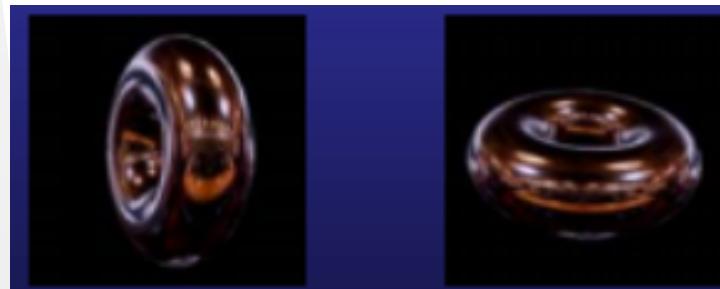
- ◆ **coord**: pode ser `GL_S`, `GL_T`, `GL_R` ou `GL_Q` para indicar se a coordenada de textura s , t , r , ou q será gerada (corresponde às coordenadas homogêneas de textura (x, y, z, w)). Em 2D usa-se somente s e t .
- ◆ **pname**: pode ser `GL_TEXTURE_GEN_MODE`, `GL_OBJECT_PLANE`, ou `GL_EYE_PLANE`.

Mapeamento Automático de Texturas

- ◆ **p a r a m :** se **p n a m e** for **GL_TEXTURE_GEN_MODE**, ele poderá ser **GL_OBJECT_LINEAR**, **GL_EYE_LINEAR** ou **GL_SPHERE_MAP**. Estas constantes determinam qual função será usada para gerar as coordenadas de textura. Com as outras opções para **pname**, **param** ou será um ponteiro para um vetor de valores ou um valor especificando parâmetros da função de geração da textura.

Mapeamento Automático de Texturas

- Em resumo, **glTexGen** gera coordenadas de textura de geometrias:
 - ◆ Espaço do objeto (**GL_OBJECT_LINEAR**):
 - ★ A textura é “colada” ao objeto, ou seja, as texturas são fixas ao objeto (aplicação tradicional).
 - ◆ Espaço do olho (observador) (**GL_EYE_LINEAR**):
 - ★ A textura é “colada” considerando o espaço do mundo, ou seja, permite fixar as texturas no espaço e o objeto move-se na textura.
 - ◆ Mapeamento esférico (**GL_SPHERE_MAPPING**):
 - ★ Baseado no vetor de reflexão (usa iluminação especular). Reflete o ambiente.



Mapeamento Automático de Texturas

- ◆ Se **pname** for **GL_OBJECT_PLANE** ou **GL_EYE_PLANE**, então **param** deverá conter os coeficientes para a função de geração de textura correspondente. Ex:

```
static GLint s_vector[4] = { 2, 0, 0, 0 };
static GLint t_vector[4] = { 0, 0, 2, 0 };
glTexGeni(GL_S, GL_TEXTURE_GEN_MODE, GL_OBJECT_LINEAR);
glTexGeniv(GL_S, GL_OBJECT_PLANE, s_vector);
glTexGeni(GL_T, GL_TEXTURE_GEN_MODE, GL_OBJECT_LINEAR);
glTexGeniv(GL_T, GL_OBJECT_PLANE, t_vector);
```

- ◆ Aqui, o modo de mapeamento **GL_OBJECT_LINEAR** mapeia as coordenadas de textura a partir das coordenadas do objeto:

$$\text{coordinate} = X * \text{vector}[0] + Y * \text{vector}[1] + Z * \text{vector}[2] + W * \text{vector}[3]$$

Mapeamento Automático de Texturas

- Devemos habilitar a geração de coordenadas de textura para a coordenada s (ou s e t) da seguinte forma:
glEnable(GL_TEXTURE_GEN_S)
- Para habilitar o cálculo para t basta incluir também:
glEnable(GL_TEXTURE_GEN_T)

Exercício

- Modifique o primeiro exemplo para conseguir colocar textura em um torus.

Exercício

- Rode o segundo exemplo que está no Moodle.
- Veja que com o botão do meio do mouse é possível testar 3 tipos de mapeamento de textura no objeto.
- Tente mudar as coordenadas do plano definidas na função ProcessMenu.

Slides

- [http://www.bdjogos.com/conteudo.php?
link=capitulo_42.php](http://www.bdjogos.com/conteudo.php?link=capitulo_42.php)
- Slides do Prof. António Ramires Fernandes – Universidade do Minho - Portugal.