Introdução ao OpenGL
Parte IV

Gilda Aparecida de Assis

gildaaa1@gmail.com

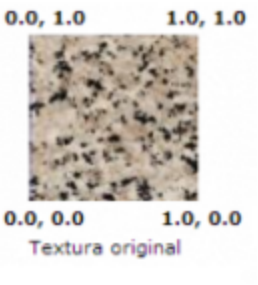
Agradecimentos ao professor Eduardo Filgueiras Damasceno por disponibilizar o material para construção da disciplina

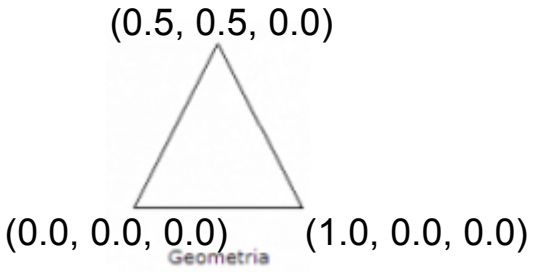
Mapeamento de Textura

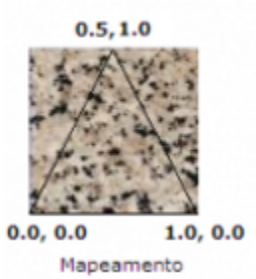
- É a aplicação de uma ou mais imagens sobre as faces de um objeto 3D
  - Uma vez mapeadas a um polígono, as texturas estão sujeitas a todas as transformações que ocorrem naquele polígono, elas rotacionam, movem ou mudam de escala com o polígono.
  - Carregar textura na memória
  - Criar textura
  - Associar textura com dados da memória
  - Associar vértices do polígono e os da textura

Mapeamento de Textura

- Para cada vértice 3D de um polígono, devem ser fornecidas as coordenadas 2D de textura.
  - As coordenadas de textura são (s,t) ou (u,v) do canto inferior esquerdo (0,0) ao superior direito (1,1). Cada vértice do polígono contém sua coordenada de textura e o OpenGL alonga ou achata a imagem para ajustar à geometria







Carregar textura na memória

- http://www.malcolmmclean.site11.com/www/datadensity/DataDensity.html
- Copiar os arquivos bmp.c e bmp.h para a pasta do projeto
  - Na janela Solution Explorer: Source Files ->Add ->Existing Item ->Selecionar bmp.c
  - Na janela Solution Explorer: Header Files ->Add ->Existing Item ->Selecionar bmp.h
  - Alterar cabeçalho de bmp.c para:

```
#define _CRT_SECURE_NO_WARNINGS //evitar erros com funções como fopen
#include <stdio.h>
#include <stdlib.h>
```

Carregar textura na memória

- Acrescente extern “C” em cada protótipo de função em bmp.h

```
extern "C" int bmpgetinfo(char *fname, int *width, int *height);
extern "C" unsigned char *loadbmp(char *fname, int *width, int *height);
```

- Copiar o arquivo bmp “quadrado e potência de 2” (largura=altura=2^n) para pasta do projeto

Carregar textura na memória

```
#include "C:\ ...\GL\glew.h"
#include "C:\...\GL\freeglut.h"
#include <stdio.h>
#include "bmp.h"

int largura, altura, tipoBMP;
GLuint IDtextura;
unsigned char *textura;
```

Carregar textura na memória

```
int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGBA);
    glutInitWindowPosition(1, 1);
    glutInitWindowSize(800, 600);
    glutCreateWindow("Minha janela GLUT");
    printf("Janela GLUT criada");
    printf("Testando Opengl...");


    textura = loadbmp("brick_64_64_24bits.bmp", &largura, &altura);
    if (*textura == 0)
        printf("Erro na carga do bmp pra memória");
    else
        printf("%i, %i\n", largura, altura);
```

Arquivo BMP

Os arquivos BMP podem ser classificados conforme a quantidade de bits para representar 1 pixel (bit/Pixel).

- 1 bit/pixel (2 cores),
- 4 bits/pixel (16 cores),
- 8 bits/pixel (256 cores),
- 24 bits/pixel (true color com até 16 milhões de cores),
- 32 bits (true color com até 4 bilhões de cores).

Arquivo BMP

a) Cabeçalho de arquivo

Contém a assinatura e informações sobre o tamanho e layout do arquivo (disposição dos dados dentro do arquivo)

b) Cabeçalho de mapa de bits

Contém as informações da imagem. Define as dimensões, compressão (se houver) e informações sobre as cores.

c) Paleta ou mapa de cores (opcional)

Somente estará presente em arquivos de imagens que usam 16 ou 256 cores (4 e 8 bits/pixel).

d) Área de dados da imagem

Dados dos pixels a serem exibidos.

Criar Textura

- Habilitar textura glEnable (GL_TEXTURE_1D ou GL_TEXTURE_2D ou GL_TEXTURE_3D);
- Cada textura tem um identificador único.
  - void glGenTextures (GLsizei n, GLuint *textures);
  - *n*: quantos objetos de textura serão criados
  - *textures*: Identificadores desses objetos criados em um array de inteiros.
- Em seguida, informar à OpenGL que objeto de textura será utilizado
  - void glBindTexture(GLenum type, GLuint texture);
  - *type*: GL_TEXTURE_1D, GL_TEXTURE_2D ou GL_TEXTURE_3D
  - *texture*: identificador da textura.

Associar textura com dados da memória

- Após a carga, dizer à OpenGL que imagem é textura.
- Para transformar um array de bytes (imagem) em texturas :
- void glTexImage1D(GLenum target, GLint level, GLint internalformat, GLsizei width, GLint border, GLenum format, GLenum type, void *data);

- void glTexImage2D(GLenum target, GLint level, GLint internalformat, GLsizei width, GLsizei height, GLint border, GLenum format, GLenum type, void *data);

- void glTexImage3D(GLenum target, GLint level, GLint internalformat, GLsizei width, GLsizei height, GLsizei depth, GLint border, GLenum format, GLenum type, void *data);

Associar textura com dados da memória

void glTexImage2D(GLenum target, GLint level, GLint internalformat, GLsizei width, GLsizei height, GLint border, GLenum format, GLenum type, void *data);

Target: GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_1D_ARRAY, GL_PROXY_TEXTURE_1D_ARRAY, GL_TEXTURE_RECTANGLE, GL_PROXY_TEXTURE_RECTANGLE, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, GL_PROXY_TEXTURE_CUBE_MAP.

Level: Nível de Detalhe. Nível 0 é o nível base da imagem.

InternalFormat: Número de componentes de cores na textura. GL_DEPTH_COMPONENT, GL_DEPTH_STENCIL, GL_RED, GL_RG, GL_RGB, GL_RGBA, formatos dimensionados (GL_RGB16I, etc) e comprimidos (GL_COMPRESSED_RGBA, etc).

Width, height: largura e altura da imagem (deve ser potência de 2).

Border: Deve ter o valor 0.

Associar textura com dados da memória

void glTexImage2D(GLenum target, GLint level, GLint internalformat, GLsizei width, GLsizei height, GLint border, GLenum format, GLenum type, void *data);

Format: Formato dos dados dos pixels. Valores possíveis: GL_RED, GL_RG, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_RED_INTEGER, GL_RG_INTEGER, GL_RGB_INTEGER, GL_BGR_INTEGER, GL_RGBA_INTEGER, GL_BGRA_INTEGER, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_DEPTH_STENCIL.

Type: Especifica o tipo de dado dos pixels. GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

Data: Ponteiro para os dados da imagem na memória.

Associar textura com dados da memória

- Especificar os filtros de textura:
  - São usados para interpolar pixels de textura.
  - Dois tipos em OpenGL:
  - GL_TEXTURE_MIN_FILTER para polígonos que são menores que a imagem de textura;
  - GL_TEXTURE_MAG_FILTER para polígonos que são maiores que a imagem de textura.
- Para especificar estes filtros :
  - glTexParameteri (GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER ou GL_TEXTURE_MAG_FILTER, filtro), onde filtro pode ser GL_NEAREST, GL_LINEAR, GL_NEAREST_MIPMAP_NEAREST, etc.

Associar vértices do polígono e os da textura

```
void DesenhaTriangulo(void) {
    glBegin(GL_TRIANGLES);
        glTexCoord2f(0.0, 0.0);
        glVertex3f(0.5, 0.5, 0.0);
        glTexCoord2f(0.5, 0.5);
        glVertex3f(0.0, 0.0, 0.0);
        glTexCoord2f(1.0, 1.0);
        glVertex3f(1.0, 0.0, 0.0);
    glEnd();
}
```

Variáveis globais

```
#include "C:\Users\Acer\Documents\Visual Studio 2013\Projects
#include "C:\Users\Acer\Documents\Visual Studio 2013\Projects
#include <stdio.h>
#include "bmp.h"

int largura, altura, tipoBMP;
GLuint IDtextura;
unsigned char *textura;
```

Função main(...)

```
int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGBA);
    glutInitWindowPosition(1, 1);
    glutInitWindowSize(800, 600);
    glutCreateWindow("Minha janela GLUT");
    //Aqui vai o código que carrega bmp
    /* ... */

    glEnable(GL_TEXTURE_2D);
    glGenTextures(1, &IDtextura);
    glBindTexture(GL_TEXTURE_2D, IDtextura);
    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, largura, altura,
        0, GL_BGR, GL_UNSIGNED_BYTE, textura);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);

    glutDisplayFunc(renderScene);
    glutMainLoop();
    return 0;
}
```

Carregar textura memória

Criar textura

Associar textura e dados memória