Módulo 11

Sistemas Gráficos e Interação

Instituto Superior de Engenharia do Porto

Filipe Pacheco

ffp@isep.ipp.pt

Texturas

Conteúdo

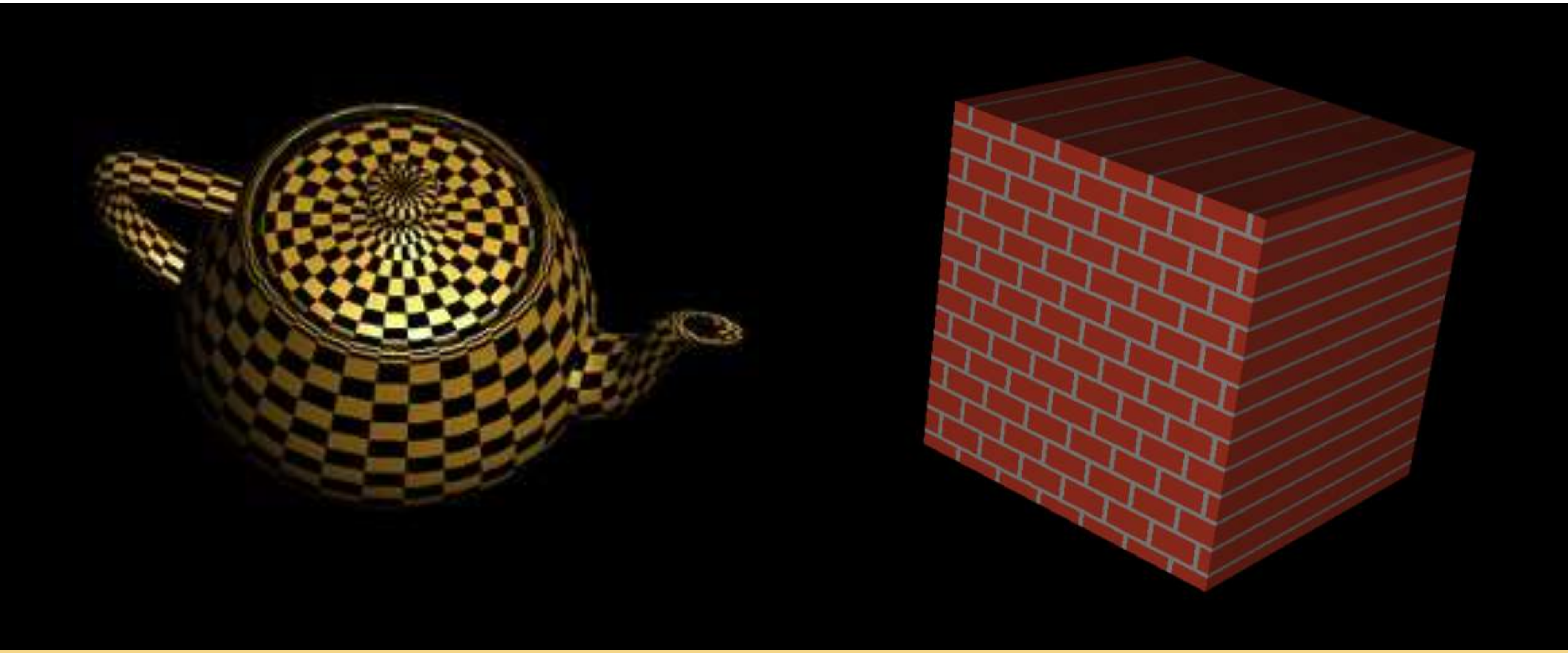
- Configuração de texturas
- Utilização de texturas 2D
  - superfícies planas
  - superfícies esféricas

Problemas

- How to realistically draw a scene with objects whose surface is not smooth?
  - Imitate natural materials, for example, wood, marble, ...
- How to draw repetitive pattern objects without having to draw lots of individual objects?
  - For example, brick wall, windows in building, etc

O que são texturas?

- An image that will be superimposed on the surface of graphic objects

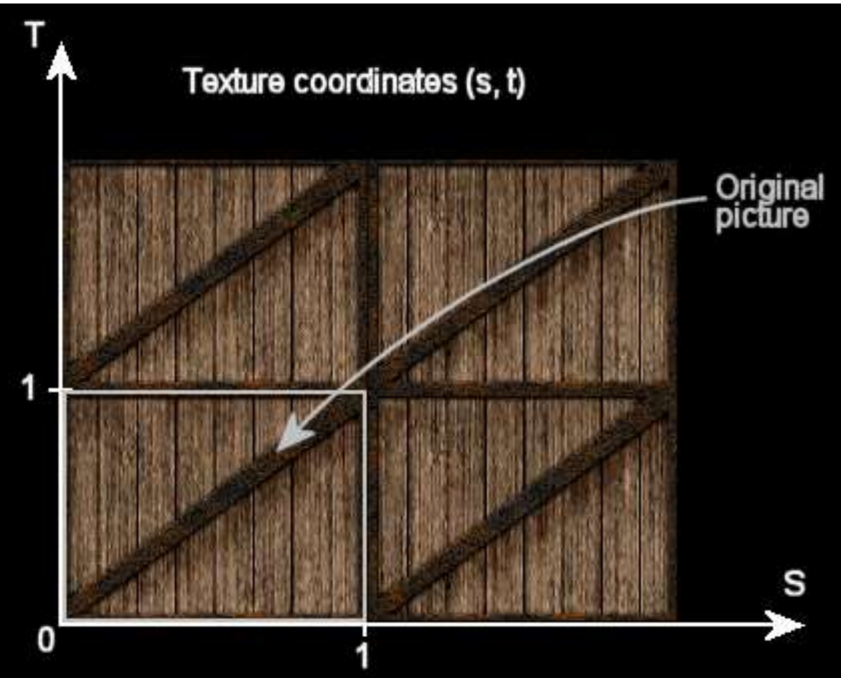


Passos necessários

1. Create a texture object and specify a texture (image) for that object.
2. Define how the texture will be applied pixel by pixel.
3. Enable texture mapping.
4. Draw the scene, providing the geometric and texture coordinates.

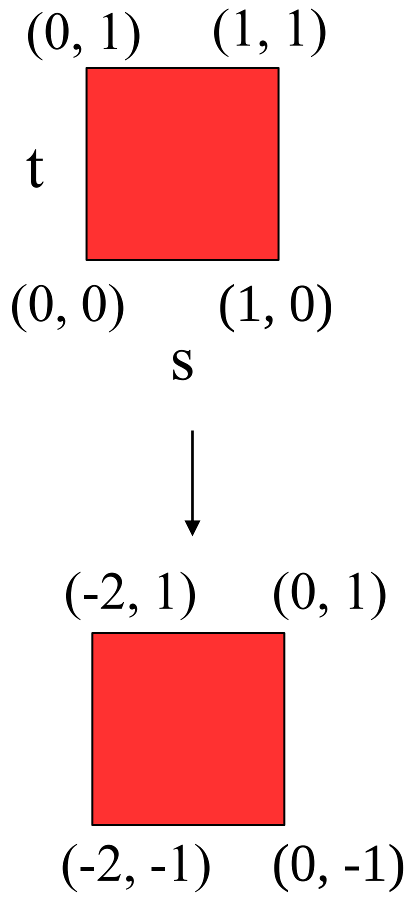
Coordenadas <S, T>

- Texture image is mapped into a coordinate space <S, T>
- between [0, 1]

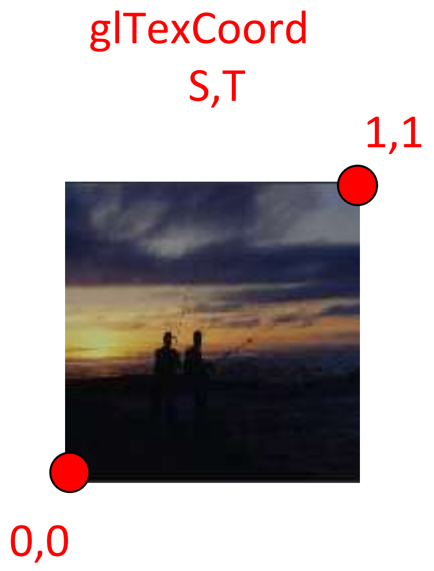


Atribuir a textura a um objecto

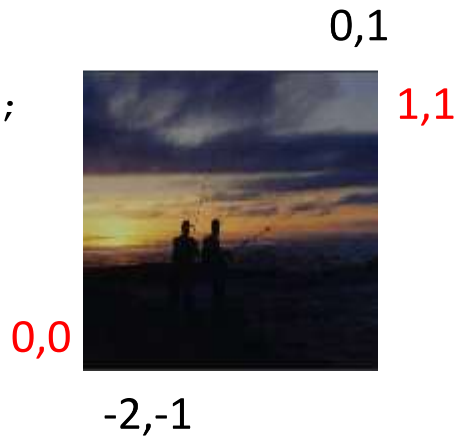
```
void display()
{
...
// activar textura
glBindTexture(GL_TEXTURE_2D, texName);

// desenhar objecto com textura
glBegin(GL_QUADS);
        glTexCoord2f(0.0, 0.0);
        glVertex3f(-2.0, -1.0, 0.0);
        glTexCoord2f(0.0, 1.0);
        glVertex3f(-2.0, 1.0, 0.0);
        glTexCoord2f(1.0, 1.0);
        glVertex3f(0.0, 1.0, 0.0);
        glTexCoord2f(1.0, 0.0);
        glVertex3f(0.0, -1.0, 0.0);
glEnd();
...
}
```



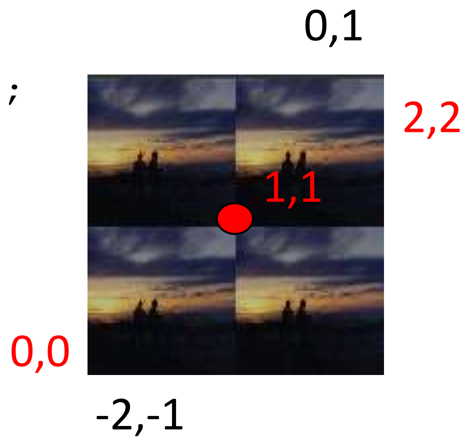
glTexCoord2f



```
glBegin(GL_QUADS);
        glTexCoord2f(0.0, 0.0);
        glVertex3f(-2.0, -1.0, 0.0);
        glTexCoord2f(0.0, 1.0);
        glVertex3f(-2.0, 1.0, 0.0);
        glTexCoord2f(1.0, 1.0);
        glVertex3f(0.0, 1.0, 0.0);
        glTexCoord2f(1.0, 0.0);
        glVertex3f(0.0, -1.0, 0.0);
glEnd();
```

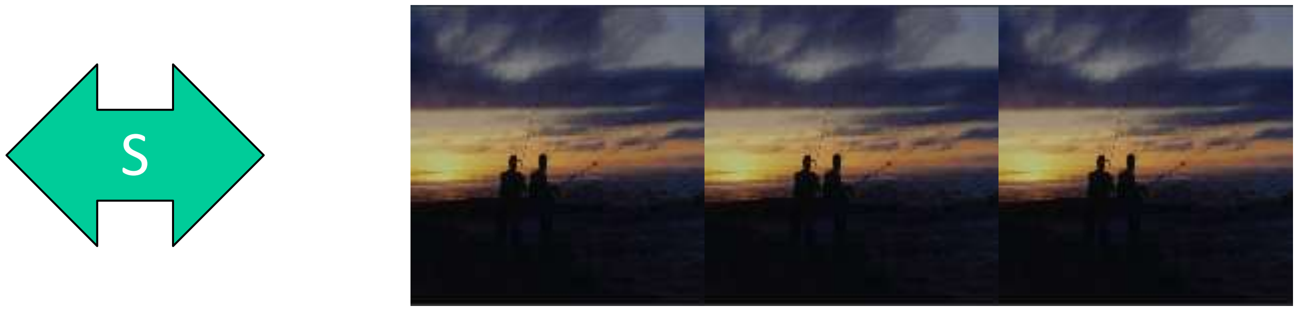


```
glBegin(GL_QUADS);
        glTexCoord2f(0.0, 0.0);
        glVertex3f(-2.0, -1.0, 0.0);
        glTexCoord2f(0.0, 2.0);
        glVertex3f(-2.0, 1.0, 0.0);
        glTexCoord2f(2.0, 2.0);
        glVertex3f(0.0, 1.0, 0.0);
        glTexCoord2f(2.0, 0.0);
        glVertex3f(0.0, -1.0, 0.0);
glEnd();
```

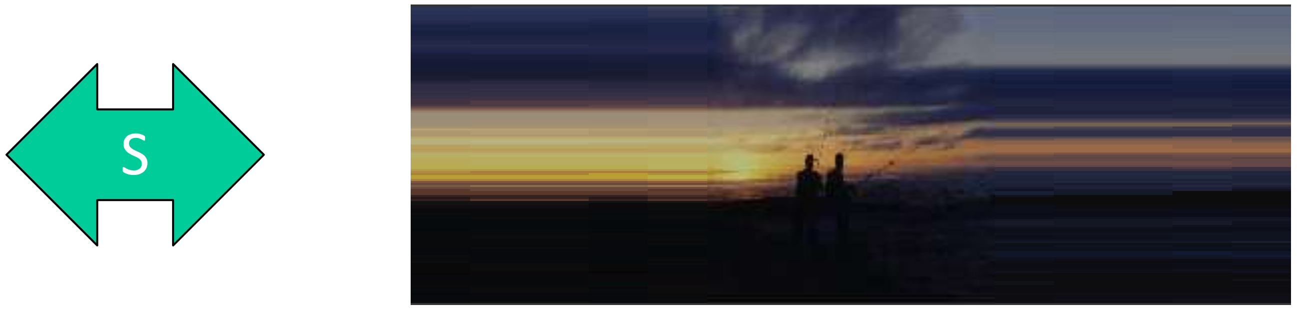


glTexParameter

- GL_TEXTURE_WRAP_S e GL_TEXTURE_WRAP_T
  - GL_REPEAT

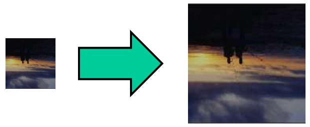


  - GL_CLAMP

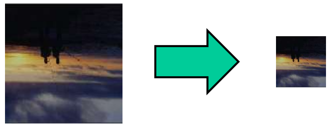


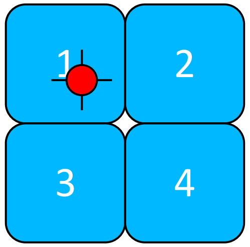
glTexParameter

- GL_TEXTURE_MAG_FILTER

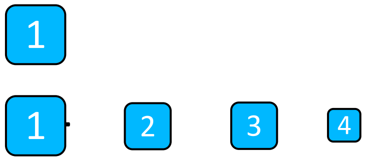


- GL_TEXTURE_MIN_FILTER





- GL_NEAREST
- GL_LINEAR



- GL_NEAREST_MIPMAP_LINEAR
- GL_LINEAR_MIPMAP_LINEAR

MipMaps
We will talk later in this module

- ...

GL_TEXTURE_ENV_MODE

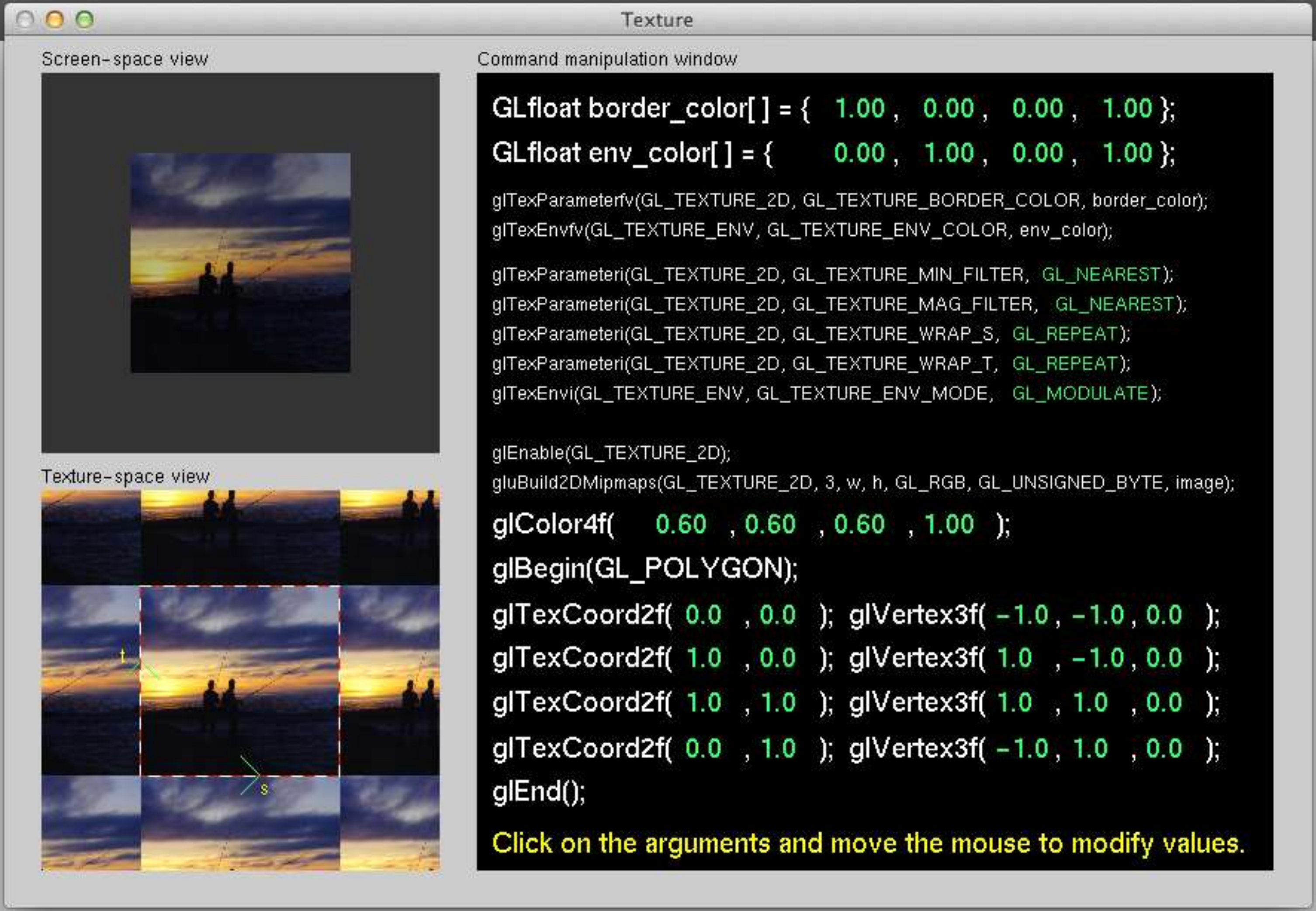
```
glTexEnvf(GL_TEXTURE_ENV,
   GL_TEXTURE_ENV_MODE, GL_REPLACE);
```

GL_REPLACE – uses only texture color

GL_MODULATE – texture color * material color

GL_DECAL – color interpolation using alpha

Demo



```
GLfloat border_color[ ] = {  1.00 ,  0.00 ,  0.00 ,  1.00 };
GLfloat env_color[ ] = {     0.00 ,  1.00 ,  0.00 ,  1.00 };

glTexParameterfv(GL_TEXTURE_2D, GL_TEXTURE_BORDER_COLOR, border_color);
glTexEnvfv(GL_TEXTURE_ENV, GL_TEXTURE_ENV_COLOR, env_color);

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,  GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,  GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S,  GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,  GL_REPEAT);
glTexEnvi(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE,  GL_MODULATE);

glEnable(GL_TEXTURE_2D);
gluBuild2DMipmaps(GL_TEXTURE_2D, 3, w, h, GL_RGB, GL_UNSIGNED_BYTE, image);
glColor4f(   0.60  , 0.60  , 0.60  , 1.00  );
glBegin(GL_POLYGON);
glTexCoord2f( 0.0  , 0.0  );  glVertex3f( -1.0 , -1.0 , 0.0  );
glTexCoord2f( 1.0  , 0.0  );  glVertex3f( 1.0  , -1.0 , 0.0  );
glTexCoord2f( 1.0  , 1.0  );  glVertex3f( 1.0  , 1.0  , 0.0  );
glTexCoord2f( 0.0  , 1.0  );  glVertex3f( -1.0 , 1.0  , 0.0  );
glEnd();
```

Click on the arguments and move the mouse to modify values.

Activar texturas no OpenGL

```
void init()
{
...
// 1 - activate textures
glPixelStorei(GL_UNPACK_ALIGNMENT, 1);
glEnable(GL_TEXTURE_2D);

// 2 – general config
glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE,
   GL_REPLACE);
...
}
```

GL_REPLACE – usa apenas cor da textura
GL_MODULATE – cor da textura * cor do material
GL_DECAL – interpolação de cor usando *alpha*

Definir textura

```
void init()
{
...
// 3 - create texture objects
GLuint texName;
glGenTextures(1, &texName);
```

one texture

```
// GLuint texNames[3];
// glGenTextures(3, texNames);
```

Example
3 textures

```
// 4 - activar textura
glBindTexture(GL_TEXTURE_2D, textName);
...
}
```

```
texNames[0]
texNames[1]
texNames[2]
```

Configurar textura

```
void init()
{
...
// 5 – configure each texture
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S,
    GL_CLAMP);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
    GL_CLAMP);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
    GL_LINEAR);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
    GL_LINEAR);
...
}
```

Definir imagem da textura

```
void init()
{
...
// 6 – load from file
GLbyte image[][][];
GLuint imageWidth, imageHeight;
...

// 7 – define the image
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, imageWidth,
    imageHeight, 0, GL_RGB, GL_UNSIGNED_BYTE,
    &image[0][0][0]);
...
}
```

glTexImage2D

- `void glTexImage2D(target, level, internalformat, width, height, border, format, type, pixels)`
  - width and height must be base power 2
  - format defines the format of the pixel array
  - GL_RGB, GL_RGBA
  - type data type of the pixel array
  - GL_BYTE, ...
  - memory pixels containing the uncompressed image

Leitura de texturas

- Any image file as long as it complies with the dimension rule that must be a power of 2
  - 64 x 64, 32 x 8, …

- LerImagens demo as code for:
  - JPEG, BMP, PPM

Leitura de BMP

```
#include <GL/glaux.h>
...
AUX_RGBImageRec *imagemBMP;
...
imagemBMP = auxDIBImageLoad("textura.bmp");
glBindTexture(GL_TEXTURE_2D, texName);
glTexParameteri(...);
...
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA,
    imagemBMP->sizeX, imagemBMP->sizeY,
    GL_RGB, GL_UNSIGNED_BYTE, imagemBMP->data);
free(imageBMP->data);
free(imageBMP);
...
```

Leitura de JPEG

```
typedef struct {
  int         sizeX, sizeY, bpp;
  char        *data;
}JPGImage;

extern "C" int read_JPEG_file(char *, char **, int *, int *, int
    *);
...
JPGImage imagemJPG;
...
read_JPEG_file("textura.jpg", &imagemJPG.data, &imagemJPG.sizeX,
    &imagemJPG.sizeY, &imagemJPG.bpp);
glBindTexture(GL_TEXTURE_2D, texName);
glTexParameteri(...);
...
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, imagemJPG.sizeX,
    imagemJPG.sizeY, GL_RGB, GL_UNSIGNED_BYTE, imagemJPG.data);
free(imagemJPG.data);
```

Leitura de PPM

```
typedef struct {
  int         sizeX, sizeY;
  char        *data;
} PPMImage;


extern "C" PPMImage *LoadPPM(char *);
...
PPMImage *imagemPPM;
...
imagemPPM = LoadPPM("textura.ppm");
glBindTexture(GL_TEXTURE_2D, texName);
glTexParameteri(...);
...
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA,
        imagemPPM->sizeX, imagemPPM->sizeY, GL_RGB,
        GL_UNSIGNED_BYTE, imagemPPM->data);
free(imagemPPM.data);
free(imagemPPM);
```

Objetos de texturas

- Generate texture objects

```
GLuint texNames[QT];
glGenTextures(QT, texNames);
```

- Configure each texture – init()

```
glBindTexture(GL_TEXTURE_2D, texNames[0]);
glTexParameteri(...);
glTexImage2D(...);

glBindTexture(GL_TEXTURE_2D, texNames[1]);
glTexParameteri(...);
glTexImage2D(...);
```

Objetos de texturas

```
void display()
{
   ...
   // ativate texture #0
   glBindTexture(GL_TEXTURE_2D, texNames[0]);
   desenhaCubo(1);
   ...
   // ativate texture #1
   glBindTexture(GL_TEXTURE_2D, texNames[1]);
   desenhaCubo(0.5);
   ...
}
```

Texturas em esferas

```
void display()
{
   ...
   glBindTexture(GL_TEXTURE_2D, modelo.textura[0]);

   GLUquadricObj* l_poQuadric = gluNewQuadric();
   gluQuadricDrawStyle(l_poQuadric, GLU_FILL);
   gluQuadricNormals(l_poQuadric, GLU_SMOOTH);
   gluQuadricTexture(l_poQuadric, GL_TRUE);

   gluSphere(l_poQuadric, 0.5, 30, 30);
   gluDeleteQuadric(l_poQuadric);
   ...
}
```

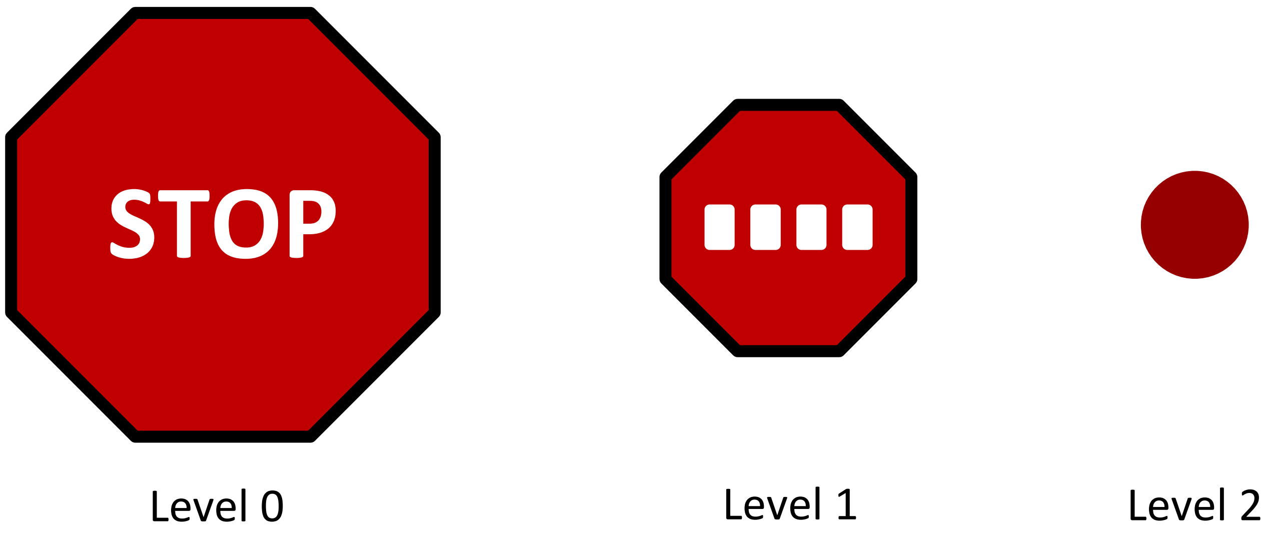
Nota:
The mapping is always the same for each Quadric object

Carregamento de texturas

- TextureLoader
  - http://members.iinet.net.au/~cleathley/openGL/TextureLoader.htm
- BMGLib
  - http://members.cox.net/scottheiman/bmglib.htm
- DXTviewer
  - http://www.ozone3d.net/dxt_viewer.php
- OpenCV
  - http://opencv.org
- FreeImage
  - http://freeimage.sourceforge.net
- …

Mipmaps

- In a scene, objects are viewed from different points of view and from different distances.
  - The texture needs to be reduced according to the projection size of the objects.
- To avoid visual defects resulting from the scale, a single texture image is not used, but a series of decreasing resolution texture maps: mipmaps.



Mipmaps

- Use level parameter of glTexImage2D
  - level = 0: higher resolution
- It is necessary to define mipmaps for all resolutions up to 1x1
- Example
  - 64x32 original image
  - Mipmaps: 64x32(0), 32x16(1), 16x8(2), 8x4(3), 4x2(4), 2x1(5), 1x1(6)

Mipmaps

- Having an image with the texture in the highest resolution, the GLU can automatically generate the corresponding mipmaps
-
```
gluBuild2DMipmaps(GL_TEXTURE_2D,
   components, width, height,
   format, type, data)
```

Mipmaps

```
void init()
{
   ...
   glGenTextures(3, modelo.textura);
   ...
   imagemBMP = auxDIBImageLoad("textura.bmp");
   glBindTexture(GL_TEXTURE_2D, modelo.textura[0]);
   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S,
         GL_REPEAT);
   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
         GL_REPEAT);
   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
         GL_LINEAR);
   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
         GL_LINEAR_MIPMAP_LINEAR);
   gluBuild2DMipmaps(GL_TEXTURE_2D, GL_RGBA,
         imagemBMP->sizeX, imagemBMP->sizeY, GL_RGB,
         GL_UNSIGNED_BYTE, imagemBMP->data);
   free(imagemBMP->data);
   free(imagemBMP);
}
```