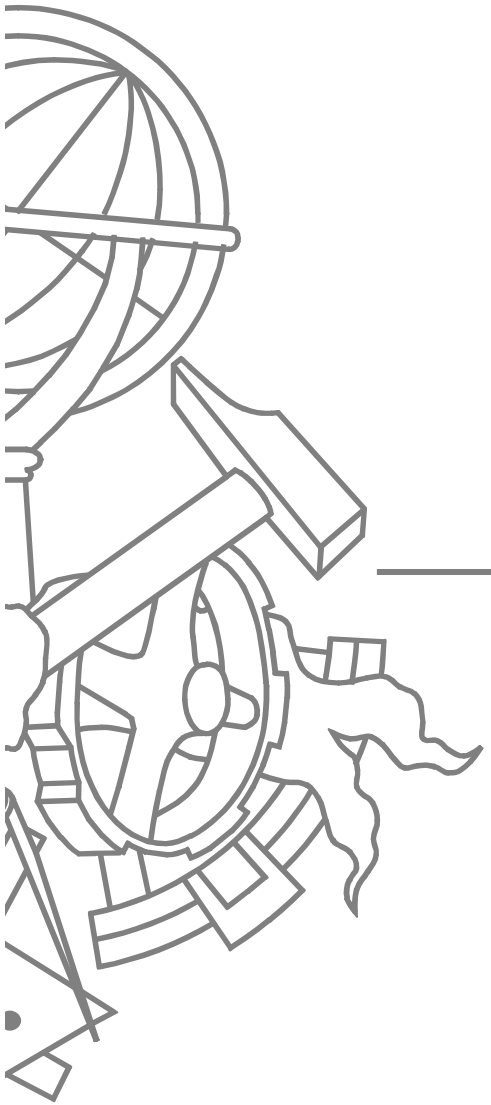


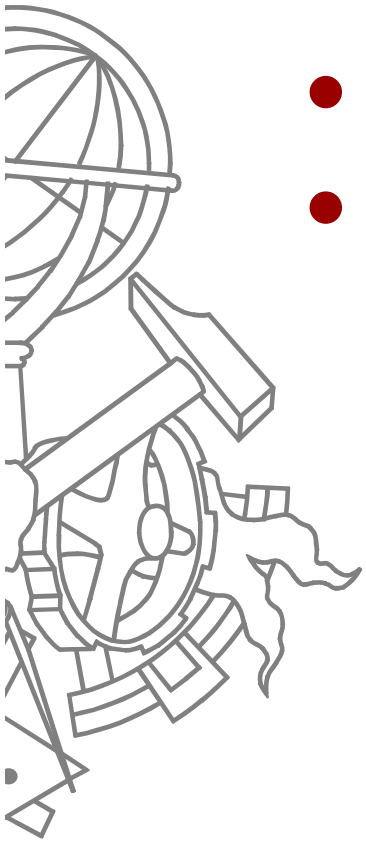
Texturas (introdução)

Aula 7

Sistemas Gráficos e Interactivos
Instituto Superior de Engenharia do Porto

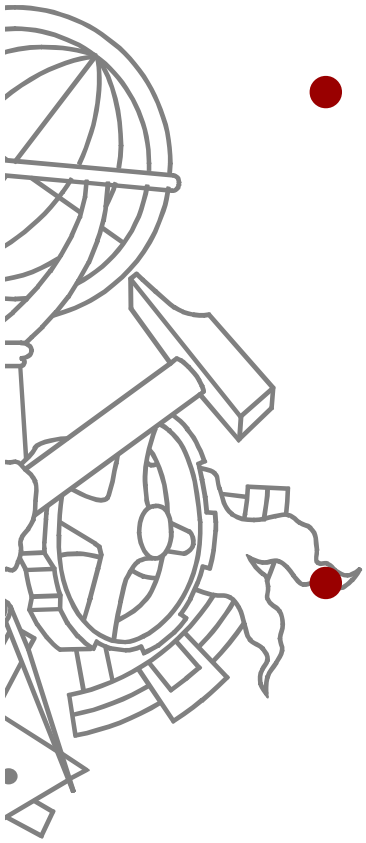
Paulo Gandra de Sousa
psousa@dei.isep.ipp.pt

Conteúdo



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

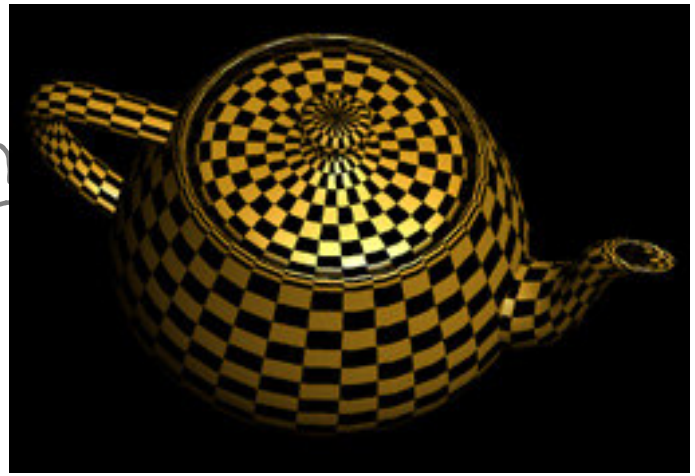
Problemas

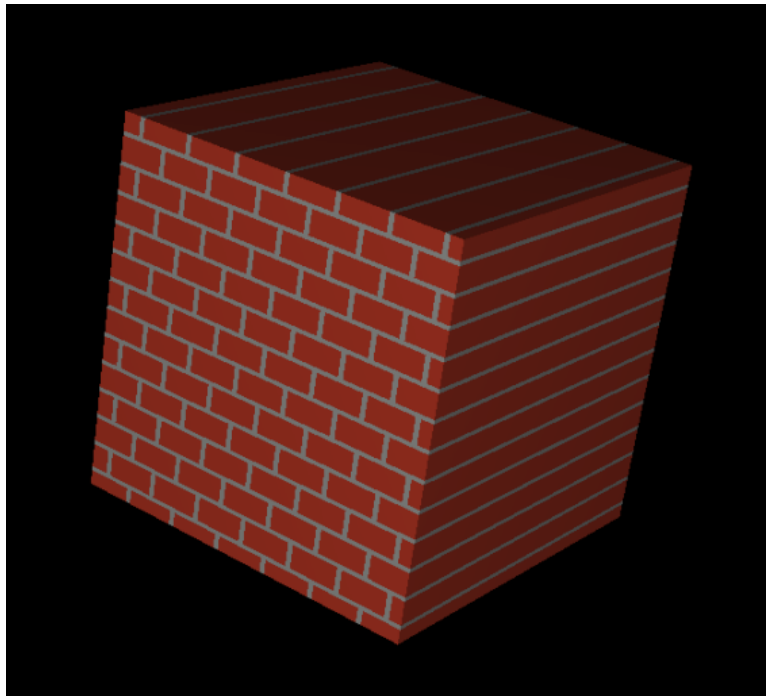


- Como desenhar de forma realista uma cena com objectos cuja superfície não é lisa?
  - Imitar materiais naturais, por exemplo, madeira, mármore, ...
- Como desenhar objectos com padrão repetitivo sem ter que desenhar imensos objectos individuais?
  - Por exemplo, parede de tijolos

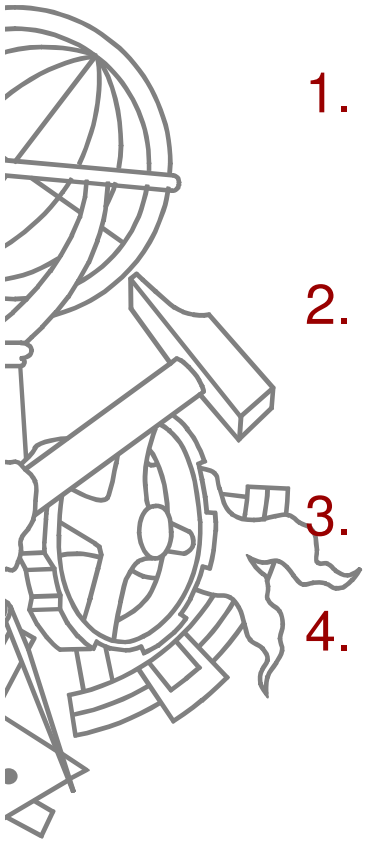
O que são texturas?

- Uma imagem que será sobreposta na superfície dos objectos gráficos



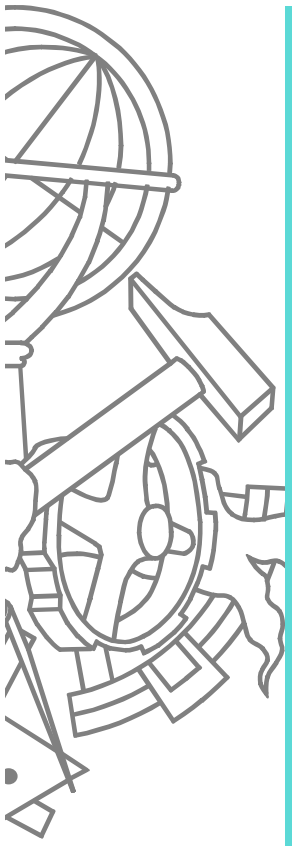


Passos necessários



1. Create a texture object and specify a texture for that object.
2. Indicate how the texture is to be applied to each pixel.
3. Enable texture mapping.
4. Draw the scene, supplying both texture and geometric coordinates.

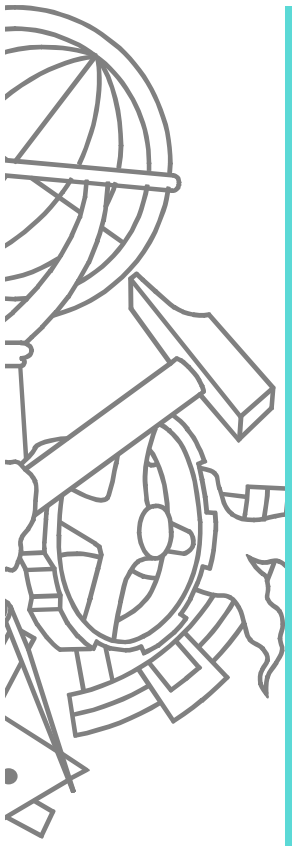
Activar texturas no OpenGL



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

// 2 - configurar aspectos gerais de texturas
glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE,
   GL_REPLACE);
...
}
```

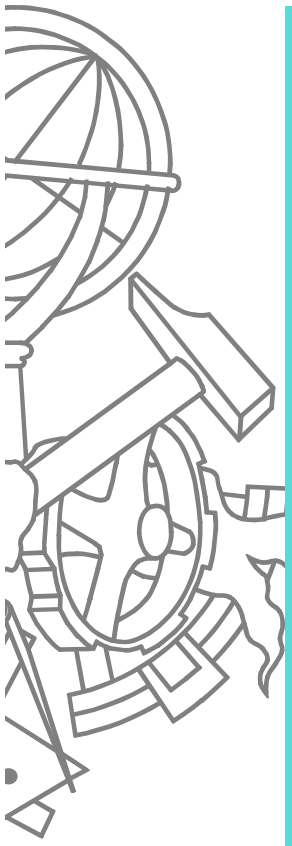
- GL_REPLACE – usa **apenas** cor da textura
- GL_MODULATE – cor da textura * cor do material
- GL_DECAL – interpolação de cor usando alpha

Definir textura



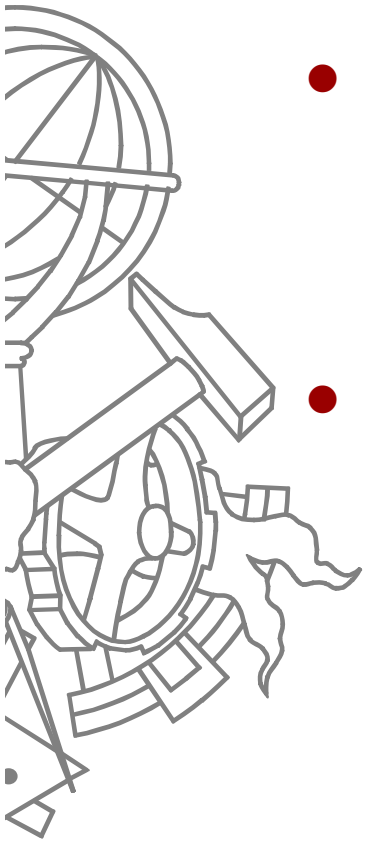
```
void init()
{
...
// 3 - criar objecto textura
GLuint texName;
glGenTextures(1, &texName);

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

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

Configurar textura



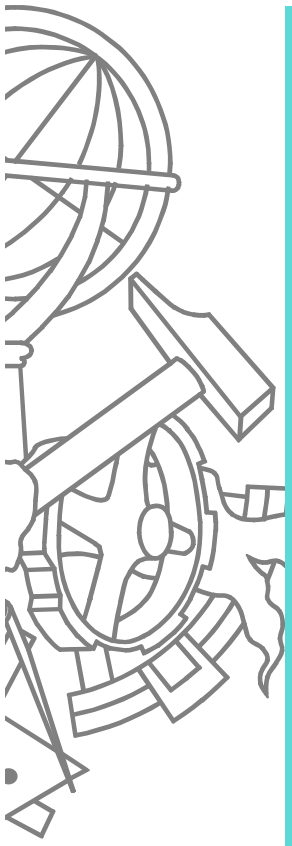
```
void init()
{
...
// 5 - configurar textura
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S,
   GL_CLAMP);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,
   GL_CLAMP);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
   GL_LINEAR);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
   GL_LINEAR);
...
}
```

glTexParameteri



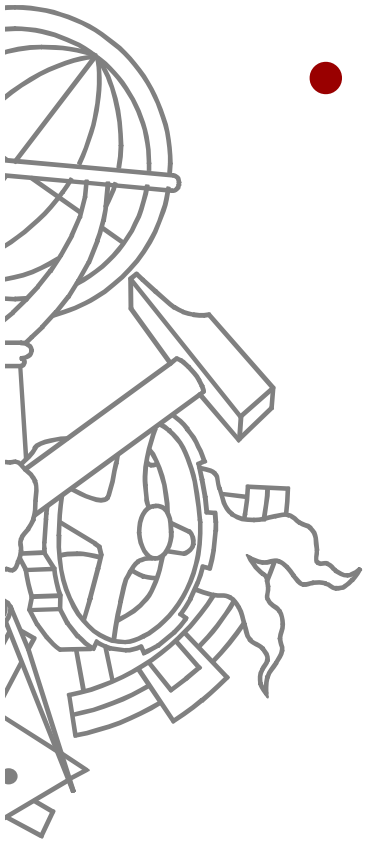
- GL_TEXTURE_WRAP_S e GL_TEXTURE_WRAP_T
  - GL_REPEAT
  - GL_CLAMP
- GL_TEXTURE_MAG_FILTER e GL_TEXTURE_MIN_FILTER
  - GL_NEAREST
  - GL_LINEAR
  - GL_NEAREST_MIPMAP_LINEAR
  - GL_LINEAR_MIPMAP_LINEAR
  - ...

Definir imagem da textura



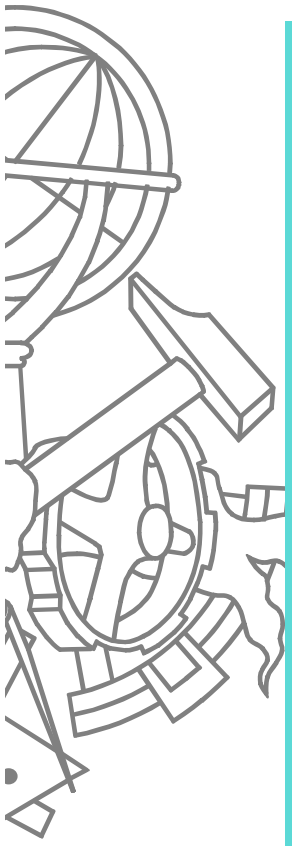
```
void init()
{
...
// 6 - carregar textura de ficheiro
GLbyte image[][][];
GLuint imageWidth, imageHeight;
...

// 7 - definir imagem
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB,
       imageWidth, imageHeight, 0, GL_RGB,
       GL_UNSIGNED_BYTE, &image[0][0][0]);
...
}
```

glTexImage2D

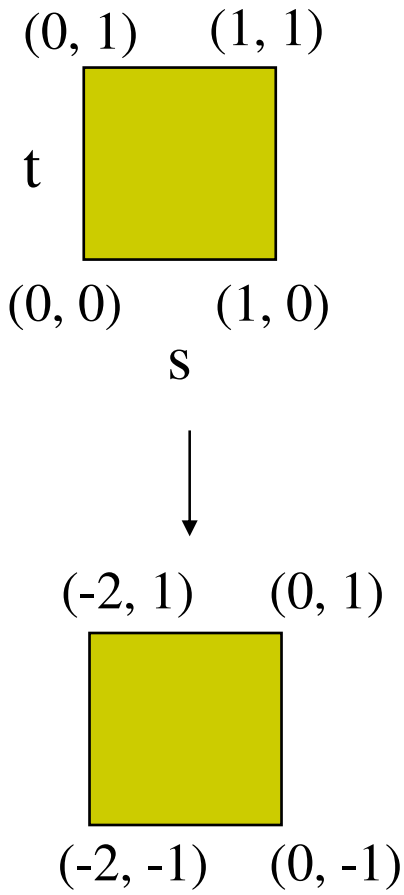


- `void glTexImage2D(target, level, internalformat, width, height, border, format, type, pixels)`
  - Width e Height TEM que ser potência de base 2
  - Format define o formato do array de pixels
    - GL_RGB, GL_RGBA
  - Type e pixels apontam para a memória contendo a imagem

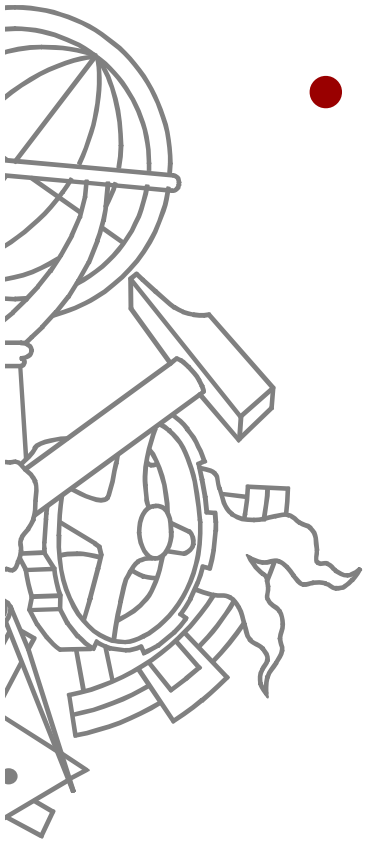
Atribuir a textura a um objecto



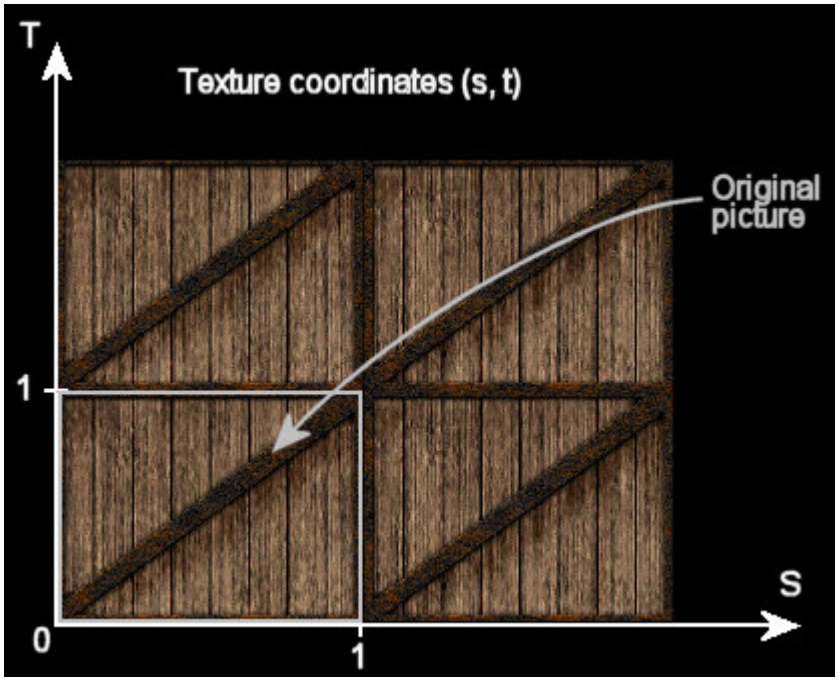
```
void display()
{
...
// activar textura
glBindTexture(GL_TEXTURE_2D, texName);

// 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();
...
}
```



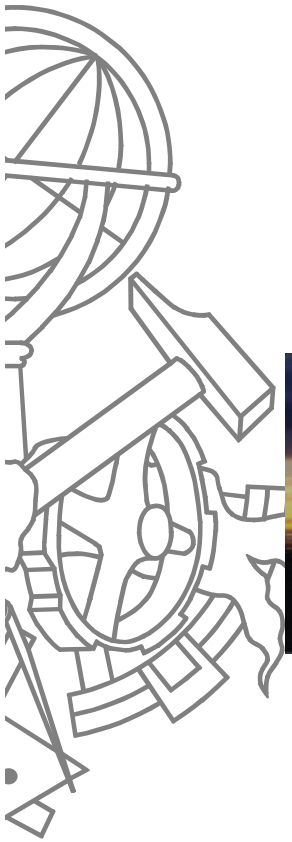
Coordenadas <S, T>

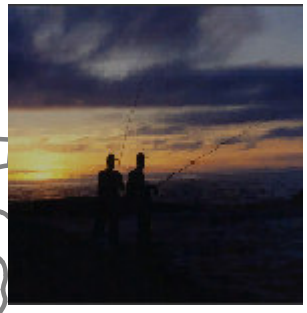


- Imagem da textura é mapeada num espaço de coordenadas <S, T>
  - entre [0, 1]



glTexCoord2f





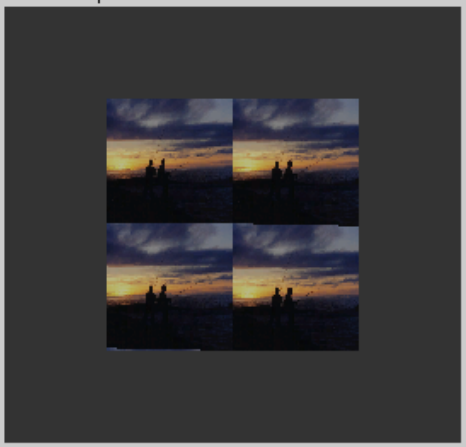
```
glBegin(GL_QUADS);
        glTexCoord2f(0.0, 0.0);
        glVertex3f(-2.0, -1.0, 0.0);
        glTexCoord2f(0.0, 1.0);
        glVertex3f(-2.0, 1.0, 0.0);
        glTexCoord2f(1.0, 1.0);
        glVertex3f(0.0, 1.0, 0.0);
        glTexCoord2f(1.0, 0.0);
        glVertex3f(0.0, -1.0, 0.0);
glEnd();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

```
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();
```

Demo

Texture

Screen-space view



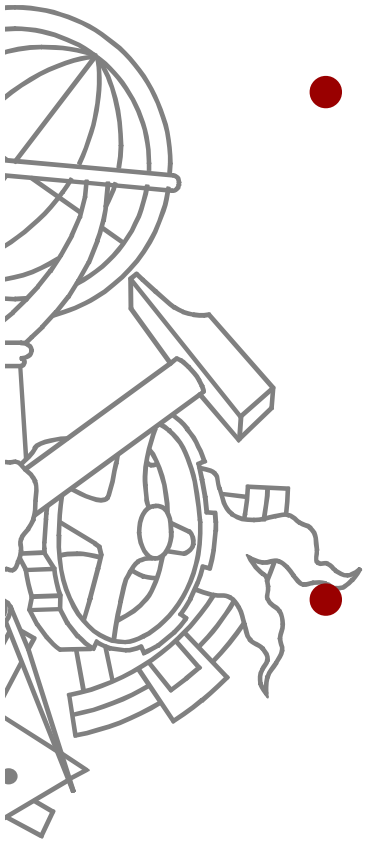
Texture-space view

Command manipulation window

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

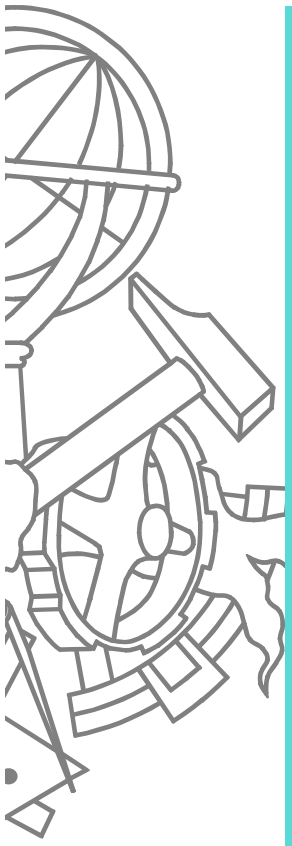
Click on the arguments and move the mouse to modify values.

Leitura de texturas



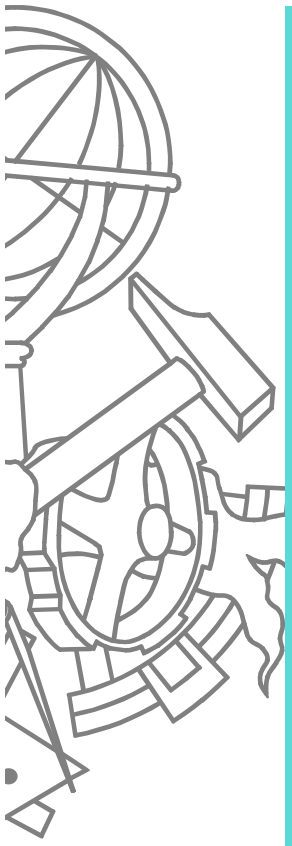
- Qualquer ficheiro de imagem desde que obedeça às regras de dimensão ser potência de 2
  - 64 x 64, 32 x 8, …
- Código da demo LerImagens
  - JPEG, BMP, PPM

Leitura de BMP



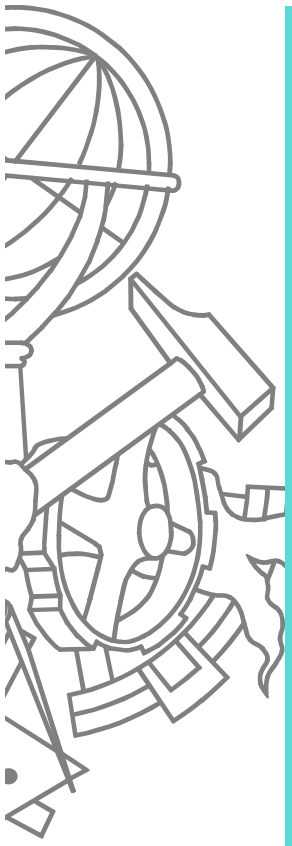
```
#include <GL/glaux.h>
...
AUX_RGBImageRec *imagemBMP;
...
imagemBMP = auxDIBImageLoad("textura.bmp");
glBindTexture(GL_TEXTURE_2D, texName);
glTexParameteri(...);
...
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA,
  imagemBMP->sizeX, imagemBMP->sizeY, GL_RGB,
  GL_UNSIGNED_BYTE, imagemBMP->data);
...
```

Leitura de JPEG



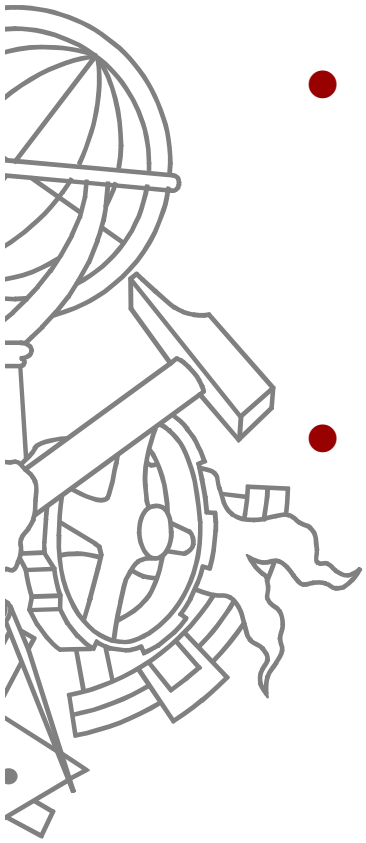
```
typedef struct {
  int           sizeX, sizeY, bpp;
  char          *data;
}JPGImage;

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);
...
```

Leitura de PPM



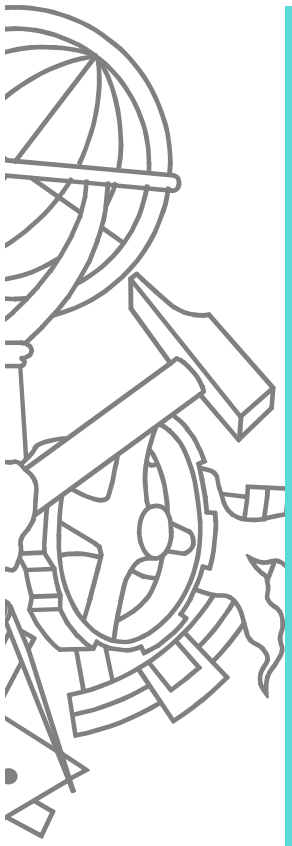
```
typedef struct {
  int           sizeX, sizeY;
  char          *data;
} PPMImage;

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);
...
```

Trabalhar com múltiplas texturas



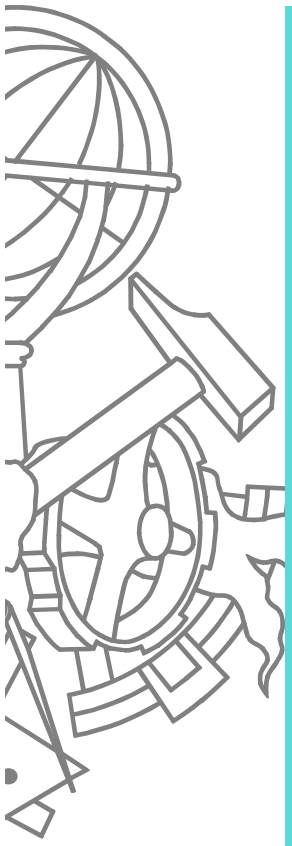
- Gerar vários nomes de texturas
  - `GLuint texNames[QT];`
  - `glGenTextures(QT, texNames);`
- Configurar cada textura – init()
  - `glBindTexture(GL_TEXTURE_2D, texNames[0]);`
  - `glTexParameteri(...);`
  - `glTexImage2D(...);`
  - `glBindTexture(GL_TEXTURE_2D, texNames[1]);`
  - `glTexParameteri(...);`
  - `glTexImage2D(...);`

Trabalhar com múltiplas texturas



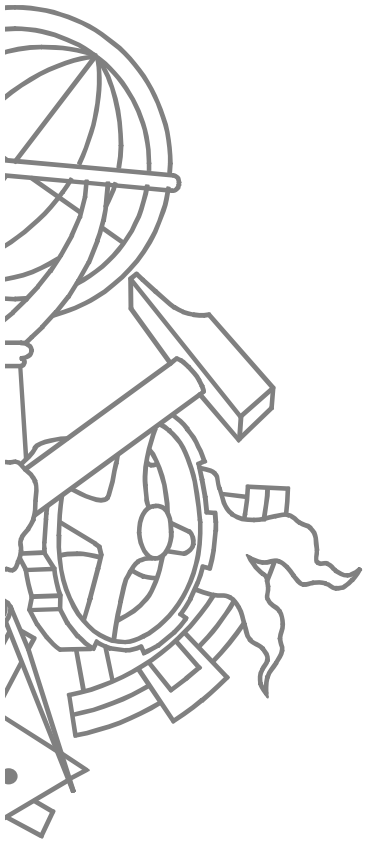
```
void display()
{
...
// activar textura #0
glBindTexture(GL_TEXTURE_2D, texNames[0]);
desenhaCubo(1);
...
// activar textura #1
glBindTexture(GL_TEXTURE_2D, texNames[1]);
desenhaCubo(0.5);
...
}
```

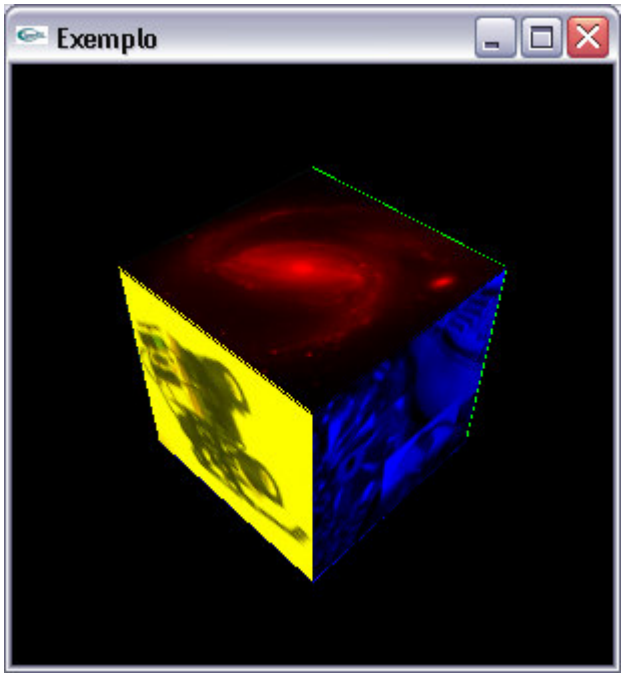
Texturas em esferas



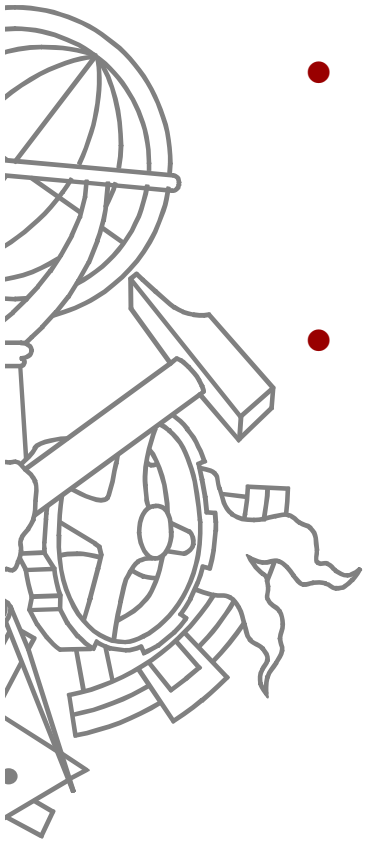
```
void display()
{
...
glBindTexture(GL_TEXTURE_2D, modelo.textura[0]);

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);
...
}
```

Demo

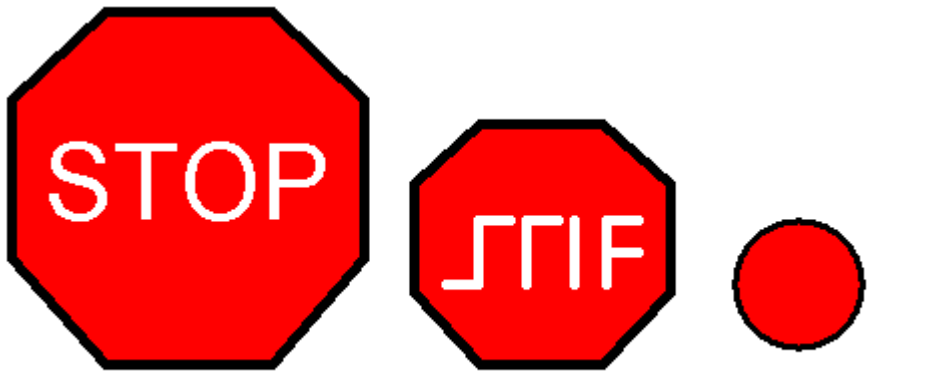




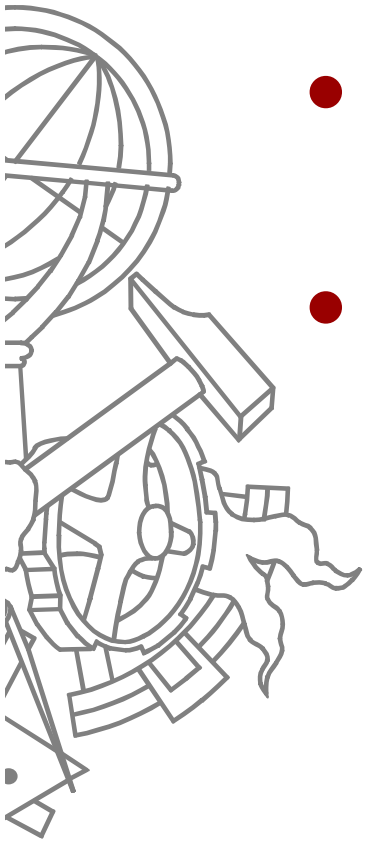
Mipmaps



- Numa cena, os objectos são vistos de diferentes pontos de vista e de diferentes distâncias.
  - a textura precisa ser reduzida de acordo com o tamanho da projecção dos objectos.
- Para evitar "defeitos visuais" resultantes da escala, não se usa uma única imagem de textura, mas uma série de mapas de textura de resoluções decrescentes: *mipmaps*.

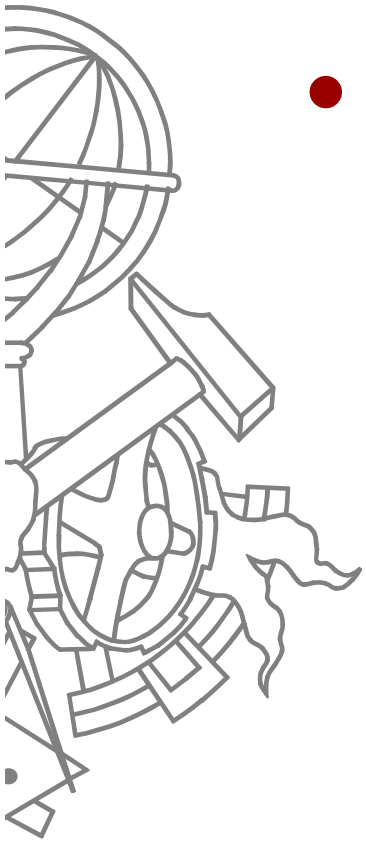


Mipmaps



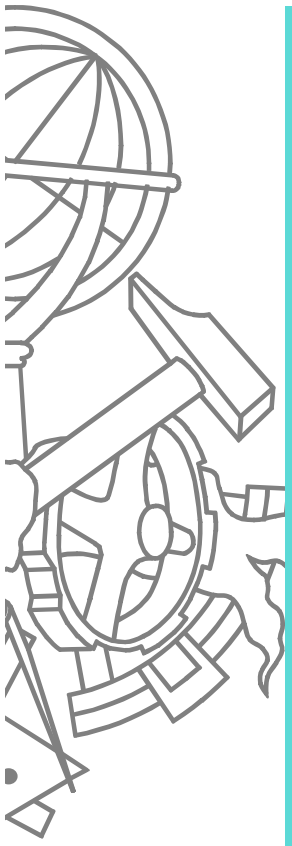
- Usar parâmetro level de glTexImage2D
  - Level = 0 : maior resolução
- É necessário definir os mipmaps para todas as resoluções até 1x1
  - Exemplo
    - imagem original 64x32
    - Mipmaps
      - 64x32, 32x16, 16x8, 8x4, 4x2, 2x1, 1x1

Mipmaps



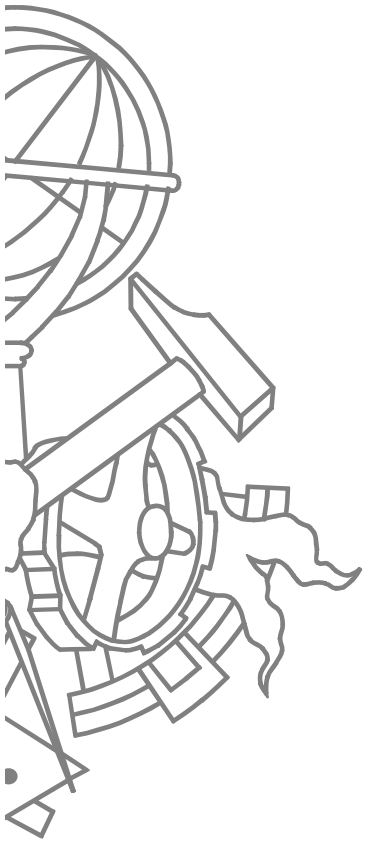
- Tendo uma imagem com a textura na maior resolução, GLU pode gerar automaticamente os mipmaps correspondentes
  - **`gluBuild2DMipmaps(GL_TEXTURE_2D, components, width, height, format, type, data)`**

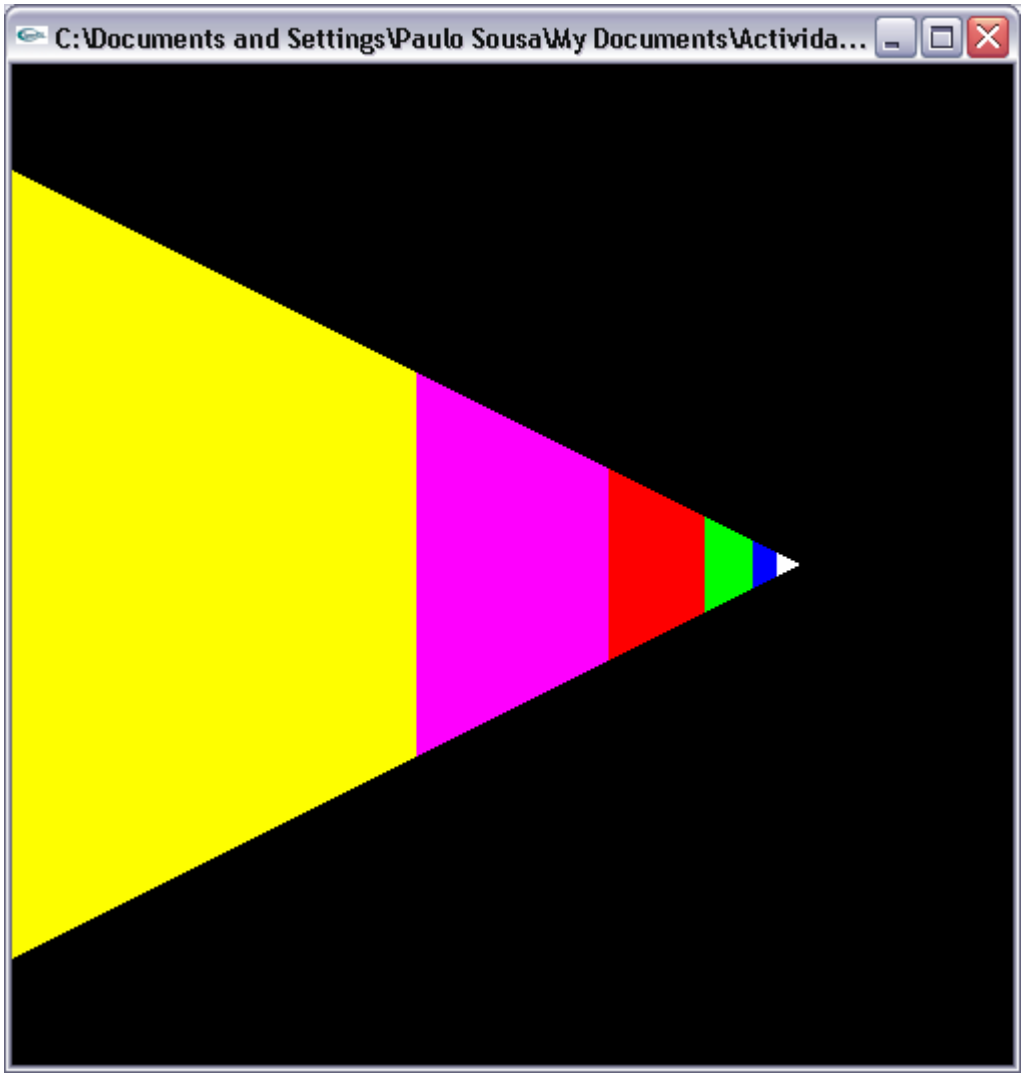
Mipmaps



```
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);
...
}
```

Demo




C:\Documents and Settings\Paulo Sousa\My Documents\Actividа...