

Texturas en OpenGL

- Una textura, desde el punto de vista de su almacenamiento en memoria es un array de datos. Cada uno de los valores de este array lo llamamos 'textel'. Este array de datos representa una imagen, que utilizaremos para mapearla sobre un polígono.
- Se pueden tener texturas unidimensionales (un solo pixel de alto o de ancho) bidimensionales (imagen de tamaño $m \times n$) o tridimensionales (con volumen), aunque lo habitual será utilizar las bidimensionales.

Pasos para mapear texturas

- Indicar los parámetros de aplicación de la textura y activar el mapeado de texturas.
- Especificar la textura
- Dibujar la escena.

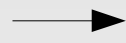
Como aplicar texturas sobre un poligono

- `glTexParameter{if}{v}(GLenum objetivo, GLenum pnombre, GLfloat param;`

pnombre

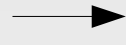
param

`GL_TEXTURE_WRAP_S`



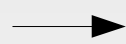
`GL_CLAMP, GL_REPEAT`

`GL_TEXTURE_WRAP_T`



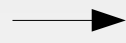
`GL_CLAMP, GL_REPEAT`

`GL_TEXTURE_MAG_FILTER`



`GL_NEAREST, GL_LINEAR`

`GL_TEXTURE_MIN_FILTER`



`GL_NEAREST, GL_LINEAR,
GL_NEAREST_MIPMAP_NEAREST,
GL_NEAREST_MIPMAP_LINEAR,
GL_LINEAR_MIPMAP_NEAREST,
GL_LINEAR_MIPMAP_LINEAR`

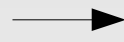
Como aplicar texturas sobre un pixel

- `glTexEnv{if}{v}(GLenum objetivo, GLenum pnombre, GLfloat param);`

pnombre

param

`GL_TEXTURE_ENV_MODE`



`GL_DECAL`

`GL_BLEND` ó `GL_MODULATE`

`GL_TEXTURE_ENV_COLOR`



Un valor de color RGBA

- Habilitar la textura

`glEnable(GL_TEXTURE2D);`

Expecificar la textura

```
glTexImage2D(GLenum objetivo, GLint nivel, GLint componentes, GLsizei ancho, GLsizei alto, GLint borde, GLenum formato, GLenum tipo, const GLvoid *pixels);
```

- objetivo → GL_TEXTURE_2D
- nivel → 0,1,2.. es el nivel de detalle de la textura. Si sólo hay una textura nivel=0
- componentes → Número de componentes de color. Color indexado=1, RGB=3, RGBA=4
- ancho → Ancho de la textura: potencia de 2
- alto → Alto de la textura: potencia de 2
- borde → Anchura del borde= 0,1, ó 2.
- formato → GL_COLOR_INDEX, GL_RGB, GL_RGBA, etc.
- tipo → GL_UNSIGNED_BYTE, GL_INT, etc.
- pixels → Puntero al array donde se almacenan los valores de la textura

Asignar coordenadas de textura.

- El último paso del proceso será dibujar los objetos. Para que la textura se visualice correctamente sobre ellos deberemos asignarle a cada uno de los vértices las coordenadas de textura. Estas coordenadas determinan qué texsel en el mapa de texturas es asignado a cada vértice. Para ello, llamaremos, antes de cada `glVertex`, a la función:

```
glTexCoord2f(coord_s, coord_t);
```

Ejemplo (en funcion init)

En la función de inicialización de valores definiremos los parámetros de pegado de la textura, y leeremos las imágenes de fichero

```
//Pegado sobre el polígono
```

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

```
//Filtros
```

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);  
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);  
glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_DECAL);
```

```
//Cargamos un fichero bmp en el array
```

```
imagen[0]=LoadBMP("file.bmp");
```

```
//Activamos las texturas
```

```
glEnable(GL_TEXTURE_2D);
```

Ejemplo (en funcion display)

Por último, en la función de dibujado, cargamos la textura antes de dibujar el polígono:

```
glTexImage2D(GL_TEXTURE_2D,0,3,imagen[0]->sizeX, imagen[0]->sizeY,  
0,GL_RGB,GL_UNSIGNED_BYTE,imagen[0]->data);
```

```
glBegin(GL_POLYGON);  
glNormal3f(0,0,1);  
glTexCoord2i(0,0); //Asignamos las coordenadas de textura del siguiente vértice.  
glVertex3f(-ancho/2,-alto/2,0);  
glTexCoord2i(0,1);  
glVertex3f(-ancho/2,alto/2,0);  
glTexCoord2i(1,1);  
glVertex3f(ancho/2,alto/2,0);  
glTexCoord2i(1,0);  
glVertex3f(ancho/2,-alto/2,0);  
glEnd();
```