Texturas en Opengl:

Para usar un mapa de texturas hay que seguir los pasos descritos a continuación:

1. Crear un objeto de textura y especificar una textura para ese objeto.
2. Indicar cómo se aplicará la textura a cada píxel.
3. Habilitar el mapa de texturas.
4. Dibujar la escena, proporcionando tanto las coordenadas de texturas como las coordenadas geométricas.

Hay que recordar que los mapas de texturas sólo trabajan en modo RGBA.

Crear un objeto de textura y especificar una textura para ese objeto:

Una textura generalmente se piensa bidimensional, pero también puede ser uni o tri-dimensional. Los datos que describen una textura pueden consistir en 1, 2, 3 o cuatro elementos por téxel representando cualquier cosa entre una modulación constante y una 4-upla (R, G, B, A).

Indicar cómo se aplican las texturas a cada píxel:

Se pueden elegir cualquiera de las cuatro funciones posibles para computar el valor final RGBA desde el fragmento de color y el dato de la imagen para la textura. Una posibilidad consiste en usar simplemente el color de la textura como color final, es el modo *replace*. Otro método consiste en usar el modo *modulate* o escalar los fragmentos de color, esta técnica es útil cuando se quieren combinar los efectos de iluminación con texturas. Finalmente, se pueden mezclar colores constantes con el fragmento de color basados en los valores de la textura.

Habilitar el mapa de texturas:

Antes de dibujar la escena es necesario habilitar las texturas usando **glEnable() glDisable()** con alguna de las constantes simbólicas GL_TEXTURE_1D, GL_TEXTURE_2D o GL_TEXTURE_3D para el texturado 1, 2, o tridimenesional respectivamente. (Si dos o tres modos de texturas están habilitados, se usa la dimensión mayor. Para obtener una mayor claridad en los programas se debería habilitar sólo la que se quiera usar).

Dibujar la escena proporcionando tanto las coordenadas de texturas como las coordenadas geométricas:

Es necesario indicar cómo se deberían alinear las texturas con respecto a los fragmentos a los que se les va aplicar antes de pegarlo. Esto es, hay que especificar tanto las coordenadas de textura y las coordenadas geométricas cuando se especifican los objetos de la escena. Para un mapa de textura bidimensional, se puede usar por ejemplo, el rango de coordenadas de textura de 0.0 a 1.0 en ambas direcciones, pero las coordenadas de los items a ser texturados pueden ser cualesquiera. Para aplicar una textura de ladrillos a una pared, por ejemplo, asumiendo que ésta es cuadrada, el código debería asignar las coordenadas (0, 0), (1,0), (0, 1), (1,1) a las 4 esquinas de la pared. Si la pared es grande necesitaremos hacer varias copias de la textura sobre ella. Si se hace esto la textura debe ser diseñada de tal forma que los ladrillos en la arista izquierda cacen con los ladrillos en la arista derecha, y lo mismo entre los superiores e inferiores.

También se debe indicar cómo deberían ser tratados las coordenadas que caigan fuera del rango [0.0, 1.0]. ¿Se repiten las texturas para cubrir el objeto o se anclan en un valor del contorno?.

Especificando la textura:

El comando **glTexImage2D()** define una textura bidimensional. Los comandos **glTexImage1D()** y **glTexImage3D()** definen texturas uni y tridimensionales respectivamente.

glTexImage2D(GLenum *target*, GLint *level*, GLint *internalFormat*,
GLsizei *width*, GLsizei *height*, GLint *border*,

GLenum *format*, GLenum *type*,
const GLvoid ***texels*);**

Define una textura bidimensional. El parámetro target debe valer GL_TEXTURE_2D o GL_PROXY_TEXTURE_2D. Se debe usar el parámetro *level* si se quieren usar múltiples resoluciones en el mapa de texturas; si sólo se va a usar una resolución *level* debe ser 0.

El siguiente parámetro, *internalFormat*, indica qué valores de componentes R, G, B, A, luminancia o intensidad se usarán para describir los texels de la imagen. Lo valores posibles para *internalFormat* son un entero de 1 a 4 o alguna de las constantes simbólicas: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE_12, GL_LUMINANCE_16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA_4, GL_LUMINANCE12_ALPHA_12, GL_LUMINANCE16_ALPHA_16, ..., GL_RGB, GL_R3,_G3_B2, GL_RGB4, GL_RGB5, ...

Si *internalFormat* es una de las 38 constantes simbólicas, lo que queremos es especificar componentes y tal vez resolución de las componenetes. Por ejemplo, si *internalFormat* es GL_R3_G3_B2, estamos solicitando 3 bits de rojo, 3 bits de verde, y 2 bits de azul, pero OpenGL no garantiza que nos de esto, sino la configuración que más se aproxime a ésta.

Los parámetro *width* y *height* dan las dimensiones de la imagen (textura), *border* indica el ancho del borde, que es 0 (sin borde) o 1. Tanto *width* como *height* deben corresponderse con un valor: $2^m + 2^b$, donde *m* es un entero no negativo (que puede tomar valores diferentes para *width* y *height*), y *b* es el valor de *border*. El tamaño máximo de una textura depende de la versión de OpenGL, pero debe ser al menos de 64x64 (o 66x66 con bordes).

Los parámetros *format* y *type* describen el formato y tipo de datos de la imagen. El parámetro *format* puede ser GL_COLOR_INDEX, GL_RGB, GL_RGBA, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, o GL_LUMINANCE_ALPHA. El parámetro *type* puede ser GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, GL_BITMAP, o uno de los formatos empaquetados de píxel.

Finalmente, *texels*, contiene los datos de la imagen. Estos datos describen la imagen así como su borde.

El formato interno de la textura puede afectar al rendimiento de las operaciones de textura. Para algunas implementaciones, puede ser más rápido utilizar GL_RGBA que GL_RGB, por lo que se recomienda cerciorarse de este tipo de custiones según la versión de OpenGL que se está usando.

El formato interno de la textura también controla qué cantidad de memoria consume la imagen. Por ejemplo, una textura de formato interno GL_RGBA8 usa 32 píxels por texel, mientras que una textura de fomato interno GL_R3_G3_B2 usa sólo 8 bits por téxel.

Aunque el mapeo de texturas no está definido par el modo de color indexado, se pueden especificar texturas en con una imagen en modo GL_COLOR_INDEX. En este caso, se aplican operaciones para convertir los índices a valores RGBA mediante la paleta de colores que se haya usado para formar la imagen.

El número de texels tanto para el ancho como el alto de la imagen sin incluir bordes debe ser una potencia de 2. Si nuestra imagen no cumple este requisito podemos utilizar **glScaleImage()** para alterar los tamaños de nuestras texturas.

Filtros:
Los mapas de texturas son cuadrados o rectangulares, pero después se mapean a polígonos o superficies y se transforman en coordenadas de pantalla, los téxels individuales de una textura raramente corresponden con píxels individuales en la imagen en la pantalla. Dependiendo de las transformaciones usadas para mapear las texturas, un único píxel en la pantalla se puede corresponder con cualquier cosa entre una porción de téxel (magnificación) o una gran colección de texels (minimización). Además, no está claro exactamente qué valores de texels se debería usar y cómo deberían ser promediados o interpolados. Por ello, OpenGL proporciona distintas opciones de filtros para determinar estos cálculos. Las distintas opciones proporcionan diferentes balances entre velocidad y calidad de la imagen. También se pueden especificar métodos de filtrado distintos para magnificación y minimización.
Tenemos que especificar 4 parámetros con 4 llamadas sucesivas a **glTexParameter*()**

glTexParameter{if]{v}(GLenum *target,*GLenum *pname,* TYPE {*}*param***)** Establece varios parámetros que controlan como se trata una textura, como se aplica a un fragmento o se almacena en un objeto de textura. El parámetro *target* es GL_TEXTURE_1D, GL_TEXTURE_2D, o GL_TEXTURE_3D. Los valores posibles para *pname* y *param* se dan en la tabla a continuación:

Parámetro	**Valores**
GL_TEXTURE_WRAP_S	GL_CLAMP, GL_CLAMP_TO_EDGE, GL_REPEAT
GL_TEXTURE_WRAP_T	GL_CLAMP, GL_CLAMP_TO_EDGE, GL_REPEAT
GL_TEXTURE_WRAP_R	GL_CLAMP, GL_CLAMP_TO_EDGE, GL_REPEAT
GL_TEXTURE_MAG_FILTER	GL_NEAREST, GL_LINEAR
GL_TEXTURE_MIN_FILTER	GL_NEAREST,GL_LINEAR, GL_NEAREST_MIMAP_NEAREST, GL_NEAREST_MIPMAP_LINEAR, GL_LINEAR_MIMAP_NEAREST, GL_LINEAR_MIPMAP_LINEAR
GL_TEXTURE_BORDER_COLOR	cuatro valores cualesquiera en [0.0, 1.0]
GL_TEXTURE_PRIORITY	[0.0, 1.0] para los objetos de textura actuales
GL_TEXTURE_MIN_LOD	cualquier valor en punto flotante
GL_TEXTURE_MAX_LOD	cualquier valor en punto flotante
GL_TEXTURE_BASE_LEVEL	cualquier entero no negativo
GL_TEXTURE_MAX_LEVEL	cualquier entero no negativo.

Se pueden asignar coordenadas de textura fuera del rango [0, 1] y anclarla o repetirla. Con teturas repetidas, si tienes un plano grande con coordenadas de textura variando de 0.0 a 10.0 en ambas direcciones, por ejemplo, se obtendrán 100 copias de la textura sobre el plano. Durante la repetición, las partes enteras de las coordenadas se ignoran , y

copias de la textura forman mosaicos sobre la superficie. En las aplicaciones en las que las texturas se repiten la parte alta debe emparejar con la baja y la izquierda con la derecha.
La otra posibilidad es anclar las coordenadas de texturas: cualesquiera valores mayores que 1.0 se convierten en 1.0, y cualesquiera valores menores que 0.0, se establecen a 0.0. El anclaje es útil en aplicaciones en las que se quiere que una única copia de la textura aparezca sobre una superficie grande. Si las coordenadas de textura de la superficie varían en el rango [0.0, 10.0] en ambas direcciones, aparece una copia de la textura en la esquina inferior izquierda.
Esto se selecciona con GL_TEXTURE_WRAP* Y GL_CLAMP, GL_CLAMP_TO_EDGE, GL_REPEAT.
GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER indican si se está especificando el método de filtrado para magnificación o minimización. El tercer argumento en este caso indica el método de filtrado. Si se elige GL_NEAREST, el texel con la coordenada más cercana al centro