Creando de Interfaces de Usuario

Práctica 3

Utilización de GLUT(2)

Héctor Garbisu Arocha
Curso 2015/16
Grado en Ingeniería Informática
Escuela de Ingeniería Informática
Universidad de Las Palmas de Gran Canaria

Índice

Tarea 1. Dibujar Textos

Para dibujar textos en la pantalla, GLUT provee dos funciones que permiten dibujar un sólo carácter cada vez; una función para gráficos de mapa de píxeles y otra para vectoriales. A dichas funciones se les debe dar una fuente y el carácter que se desea dibujar. Los datos relativos a la posición y el color se han de definir previamente, como en otros tipos de gráfico.

La preparación del entorno, inicialización de GLUT y OpenGL es igual que en la práctica anterior. Se nos ha suministrado un fichero main.cpp gracias al que sólo necesitamos programar los cambios referentes a esta práctica en particular.

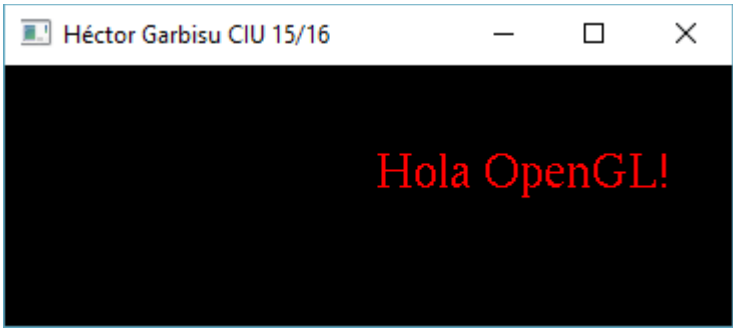
Como la tarea consiste únicamente en mostrar algo por pantalla y la única interacción del usuario es la redimensión de la ventana, en la función main basta con indicar glutDisplayFunc(Display), siendo Display una función que nosotros definimos para dibujar un texto de una u otra forma.

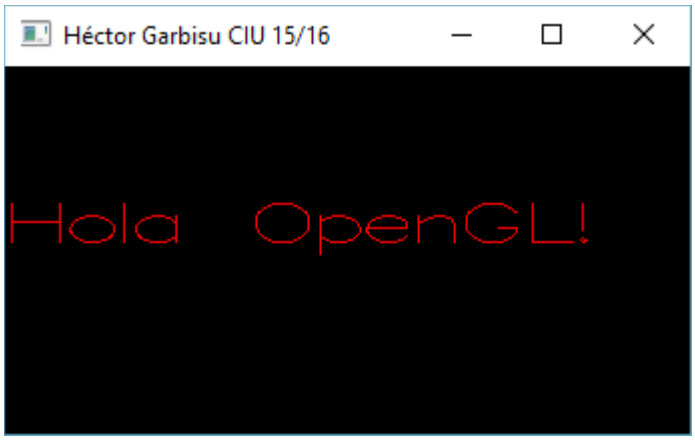
Caso A. Texto en mapa de bits

1. Borra el framebuffer
2. Se especifica el color de dibujo
3. Escribe cada carácter en posiciones adyacentes

Caso B. Texto en forma vectorial

1. Borra el framebuffer
2. Se especifica el color del dibujo
3. Se carga la matriz identidad en GL_MODELVIEW
4. Se traslada y se escala la vista del modelo
5. Se escribe cada carácter en posiciones adyacentes





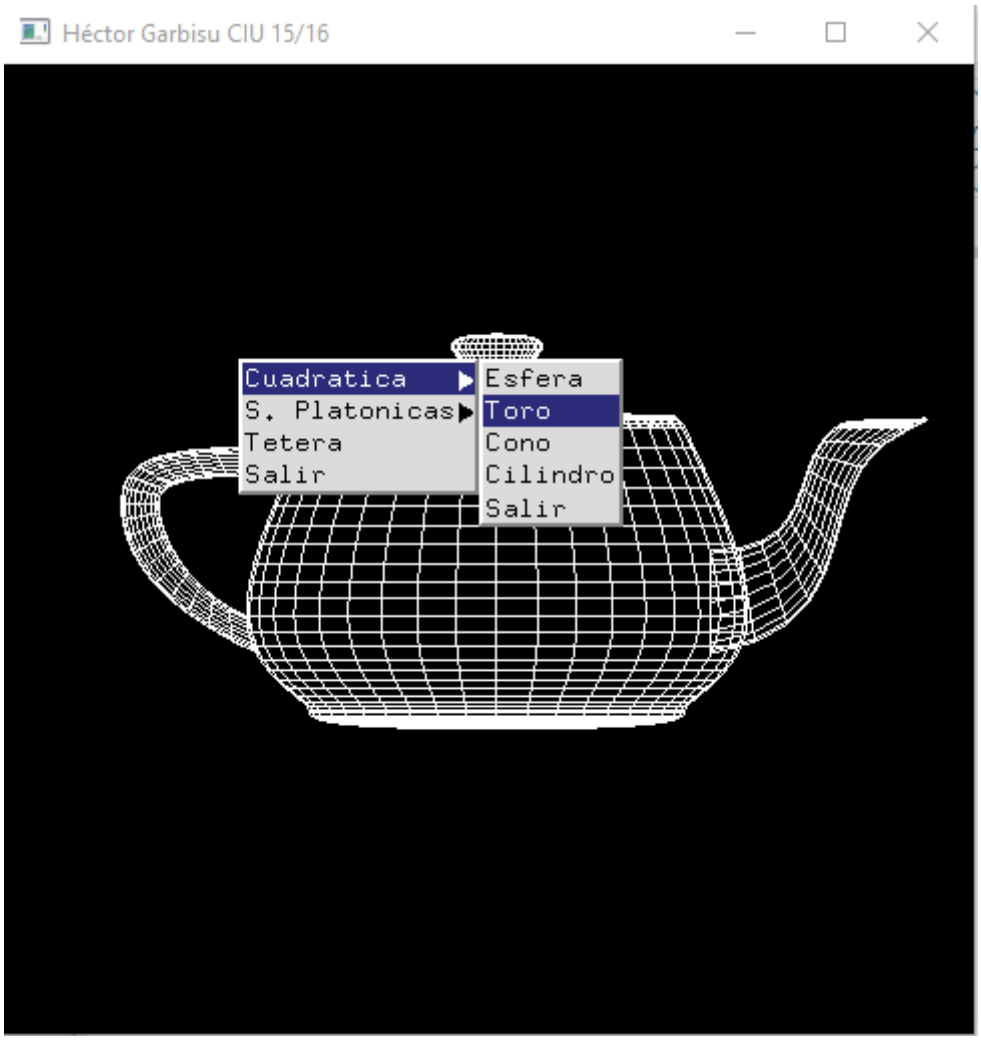
Tarea 2. Crear jerarquía de menús

De la jerarquía de menús, los primeros elementos en ser añadidos son los útlimos que aparecerían en un uso corriente del menú; luego los submenús que los contienen, y así sucesivamente. Las acciones que se desencadenan al hacer clic en uno de los elementos del menú son tres posibles: o bien se muestran los submenús, o se muestra una figura tridimensional o se sale de la aplicación.

Para generar una estructura de menús completa, primero se crea uno de los submenús más exteriores, uno que sólo vaya a contener elementos terminales. Mediante glutCreateMenu se crea, y se le asigna un identificador que se usará más adelante para encajarlo en otros menús mayores si hiciera falta.
Luego se añaden sus elementos terminales mediante glutAddMenuEntry, especificando un texto y un código.
Es importante que se creen primero los submenús exteriores, ya que glutAddMenuEntry sólo opera sobre el último menú hecho con glutCreateMenu.

Mediante glutAddSubMenu se pueden colocar unos menús debajo de otros jerárquicamente.

A cada menú creado, se le debe pasar una función encargada de reaccionar a los códigos que se habían indicado a los elementos “hoja” del árbol de menús. En nuestro caso, esa función es la misma para cada submenú, encargándose de modificar la variable que indica qué modelo tridimensional mostrar y de hacer glutPostRedisplay.



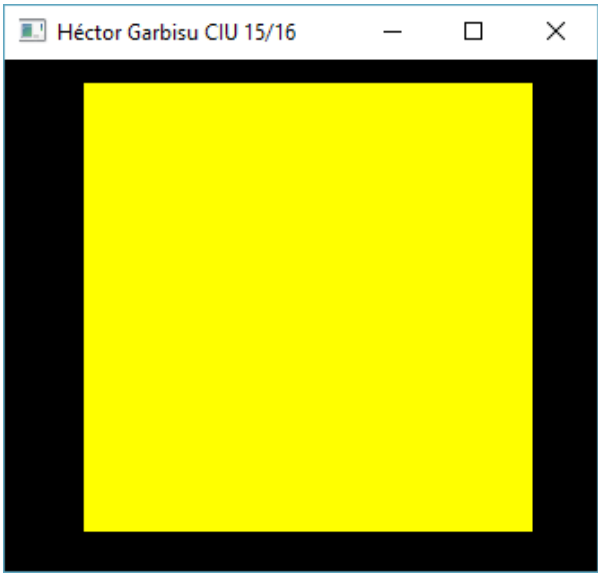
Tarea 3. Animación, ratón y timer

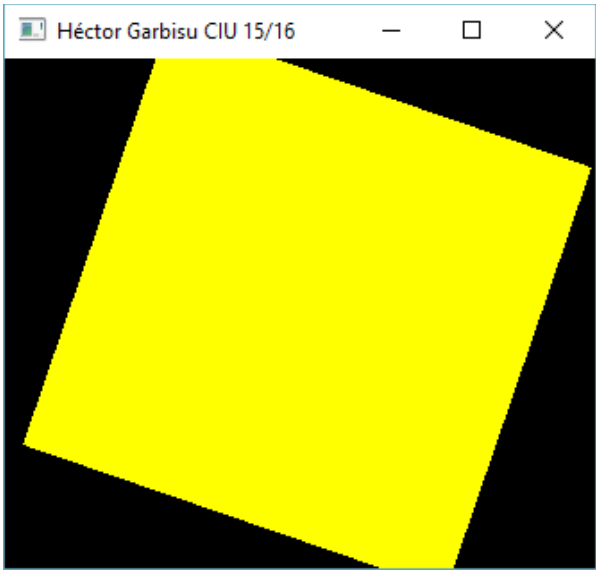
La animación que utilizaremos aquí consiste en cambiar la posición (orientación) de un objeto a intervalos regulares suficientemente cortos como para percibirse un movimiento continuo.

La interacción con el ratón ya la hemos trabajado en prácticas anteriores y el escalado también, pero no la rotación, que tan solo supondrá hacer uso de la función glRotatef.

GLUT tiene su propia función de temporización, aunque desconozco si tiene ventajas sobre otras herramientras de mismo propósito. A glutTimerFunc se le debe dar un tiempo en milisegundos, una función y un valor. Tras pasar el tiempo indicado en el primer parámetro, se llama a la función (callback), usando como parámetro de ésta el tercer parámetro indicado a glutTimerFunc.

En nuestro caso la función de timer se encargará de modificar las variables que definen la rotación del objeto, en función de otras variables que indican la dirección de la rotación.





Tarea 3. Modificación

Para esta extensión de la Tarea3 voy a hacer que el cuadrado pueda rotar a una velocidad variable, dependiendo de las veces que se haya hecho uno u otro clic. El resultado es un cuadrado que gira muy rápido en sentido antihorario cuando se ha hecho muchas veces clic izquierdo, gira muy rápido en sentido horario cuando se ha hecho muchas veces clic derecho, y está parado si se ha dado un número igual de clics izquierdos y derechos, independientemente del orden, o se pulsa ratón central.

Los cambios al código original son mínimos. La mayor diferencia es que ahora el periodo de refresco se ha forzado en 17ms, lo que equivale a unas 59 imágenes por segundo. Si bien es cierto que para un cuadrado que se mueve muy lentamente tanto refresco puede ser excesivo, la actualización que teníamos originalmente de 10 imágenes por segundo se volvía insuficiente desde que se le daba un poco de velocidad al giro del cuadrado.

```
131 void Timer(int v){
132     if (velocidad!=0){
133         angulo += velocidad;
134         if (angulo > 360.0) angulo -= 360.0;
135         if (angulo < 360.0) angulo += 360.0;
136         glutPostRedisplay();
137     }
138     glutTimerFunc(17, Timer, v);
139 }
140
141 void Raton(int boton, int estado, int x, int y){
142     if (boton == GLUT_LEFT_BUTTON && estado == GLUT_DOWN){
143         velocidad+=0.2;
144     }
145     if (boton == GLUT_RIGHT_BUTTON && estado == GLUT_DOWN){
146         velocidad-=0.2;
147     }
148     if (boton == GLUT_MIDDLE_BUTTON && estado == GLUT_DOWN){
149         velocidad = 0;
150     }
151 }
```