#include <GL/glut.h>

#include <stdlib.h>

void display(void)

{

glClearColor(0.0,0.0,0.0,0.0);

//color de fondo: negro

glClear(GL\_COLOR\_BUFFER\_BIT);

//BORRAMOS PANTALLA

glMatrixMode(GL\_PROJECTION);

//MODO DE PROYECCION

glLoadIdentity();

//cargamos la matrix identidad

glOrtho(-1.0,1.0, -1.0,1.0, -1.0,1.0);

glMatrixMode(GL\_MODELVIEW);

//MODO DE MODELADO

glBegin(GL\_QUADS);

//dibujamos un CUADRADO

glColor3f(0.0,1.0,1.0);

glVertex3f(-0.5,0.5,-0.5);

//coordenadas del primer vertice(SUPERIOR-IZDA)

glVertex3f(-0.5,-0.5,0.5);

//coordenadas del 2do vertice(INFERIOR-IZDA)

glVertex3f(0.5,-0.5,0.5);

//coordenadas del 3er vertice(INFERIOR-DCHA)

glVertex3f(0.5,0.5,-0.5);

//coordenadas del 3er vertice(INFERIOR-DCHA)

glEnd();

glFlush(); }

main(int argc, char \*\*argv)

{

glutInit(&argc,argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGBA);

glutInitWindowSize(640,480);

glutInitWindowPosition(50,50);

glutCreateWindow("FIGURAS");

glutDisplayFunc(display);

glutMainLoop();

return 0;

}

***PRACTICA # 2***

**REALIZA LOS SIGUIENTES PUNTOS**

* Con glClearColor ordenamos a OpenGL que borre los buffer indicados como parámetro. Para borrar la pantalla(el buffer de color) lo indicamos con la constante GL\_COLOR\_BUFFER\_BIT
* Establecemos el volumen de visualización, en modo de proyección y con glOrtho.
* Vamos a dibujar un cuadrado escribe el código correspondiente.
* Guardamos, compilamos y ejecutamos

1. OBSERVA QUE PASA?

**SE BORRAN LOS BUFFERS, ELEGIMOS MODO DE PRYECCION, CARGAMOS UNA MATRIZ DE IDENTIDAD Y LE ESPECIFICAMOS LOS LIMITES EN X y Y DE LA PANTALLA, SE EMPIEZA A DIBUJAR CUADRADO INICIANDO POR EL VERTICE SUPERIOR IZQ HACIA EL VERTICE SUPERIOR DERECHO.**

***PRACTICA # 3***

* Para ello vamos a sustituir la llamada a glOrtho por un llamada gluPerspective, que establecerá el modo de visualización en perspectiva.
* gluPerspective(60.0,1.0,1.0,100.0);

(Proyección perspectiva. El ángulo de visualización es de 60 grados, la razón ancho/alto es 1(son iguales) la distancia mínima es de z=1.0, y la distancia máxima es z=100.0

* Guardamos, compilamos y ejecutamos

1. OBSERVA QUE PASA?

**NO SE MIRA NADA DA DO QUEESAMOS MUY LEJOS DEL CUADRADO Y ESTE TIENE UNAS DIMENSIONES MUY PEQUEÑAS**

//NO SE VE NADA ? EN LA VISUALIZACION PERSPECTIVA, LA POSICION DEL OBSERVADOR DETERMINA COMO SE VEN LAS COSAS

//VAMOS A TOMAR QUE EL OBSERVADOR, YA QUE HEMOS INDICADO LO CONTRARIO SE ENCUENTRA EN LA POSICION(0,0,0)

//VAMOS A RETRASAR UN POCO EL CUADRADO, PARA ALEJARLO DEL OBSERVADOR SINO, SE DIBUJA JUSTO EN EL MISMO LUGAR

//QUE ESTE, Y NO PUEDE VERSE NADA AÑADIR EL CODIGO

glTranslatef(0.0,0.0,-2.0);

// alejamos el observador dos unidades del eje z

* Guardamos, compilamos y ejecutamos

1. OBSERVA QUE PASA ?

SIGUE SIN OBSERVARSE NADA. TALVEZ PORQUE LOS LIMITES DE LOS EJES SON GRANDES Y LAS DIMENSIONES DEL CUADRADO SON MENORES A 1.

* Dibuja un triangulo con las coordenadas

(0.0, 0.7, 0.0), (-0.9,-0.7, 0.0), (0.9,-0.7, 0.0);

1. OBSERVA QUE PASA ? ESCRIBE O DIBUJA

* COMO SE PUEDE OBSERVAR , NO SE TIENE EN CUENTA EN LA PARTE INFERIOR DEL TRIANGULO DEBERIA ESTAR TAPADA POR EL CUADRO, PORQUE SU COORDENADA Z, QUE INDICA LA PROFUNDIDAD ES MAYOR***( -0.7*** PARA EL CUADRADO Y ***0*** PARA EL TRIANGULO). PARA PODER VISUALIZAR LOS OBJETOS CON OCLUSION, TENEMOS QUE ACTIVAR EL Z-Buffer
* El Z-Buffer es un array que guarda la profundidad (coordenada Z) para todos los píxels

del área de dibujado, de manera que cuando un píxel va a ser dibujado, se mira si su

coordenada Z es menor que la coordenada en el Z-Buffer para ese píxel en pantalla.

Para activarlo, hemos de añadir las siguientes líneas de código, antes del dibujado:

glDepthFunc(GL\_LEQUAL);

glEnable(GL\_DEPTH\_TEST);

glClearDepth(1.0);

* Con la llamada a glEnable(GL\_DEPTH\_TEST) habilitamos la comprobación de la profundidad en el dibujado.

CON glDepthFunc(GL\_LEQUAL) decimos que el nuevo píxel se dibuje si su coordenada

Z es más cercana al punto de visualización, o igual, que la que hay actualmente en ZBuffer.

CON glClearDepth(1.0) decimos que cada vez que quedamos borrar el buffer de profundidad, se inicialicen sus posiciones al valor 1.0.

Junto con esto, al borrar la pantalla, le indicamos al programa que inicialice el Z-Buffer con:

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

* Finalmente, sólo nos queda indicar al programa que, cuando se cree la ventana, se

reserve espacio para el Z-Buffer. Esto lo indicamos al crear la ventana con:

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGBA | GLUT\_DEPTH);

* GUARDAMOS COMPILAMOS Y EJECUTAMOS

1. POR ULTIMO OBSERVA QUE PASA