UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

Cuestionario previo y Practica 1

MATERIA:

Computación gráfica

ALUMNO(S):

PADILLA HERRERA CARLOS IGNACIO

PROFESOR(A):

Luis Sergio Valencia Castro

GRUPO:

2

Padilla Herrera Carlos Ignacio
Cuestionario Previo Práctica 2

Introducción a OpenGL.

Objetivo: El alumno se familiarizará con las funciones gráficas básicas de OpenGL para la construcción de primitivas.

Cuestionario Previo: (puede ser entregado en equipo de dos personas, recuerden que deben incluir una conclusión de manera individual)

1. Investigue qué es un IDE en el ámbito de programación.

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, o sea, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. El lenguaje Visual Basic, por ejemplo, puede ser usado dentro de las aplicaciones de Microsoft Office, lo que hace posible escribir sentencias Visual Basic en forma de macros para Microsoft Word.

2. Investigue que es GLUT y cómo ayuda o complementa a OpenGL.

Es una biblioteca de utilidades para programas OpenGL que principalmente proporciona diversas funciones de entrada/salida con el sistema operativo. Entre las funciones que ofrece se incluyen declaración y manejo de ventanas y la interacción por medio de teclado y ratón. También posee rutinas para el dibujado de diversas primitivas geométricas (tanto sólidas como en modo *wireframe*) que incluyen cubos, esferas y teteras. También tiene soporte para creación de menús emergentes.

3. ¿Para qué sirve el comando glLoadIdentity()?

Reemplaza la matriz actual con la matriz identidad

4. ¿Qué es un pixel?

Unidad básica de una imagen digitalizada en pantalla a base de puntos de color o en escala de grises.

5. Investigue la definición de PPI (pixels per inch), e indique la diferencia que tiene con la resolución de un dispositivo. ¿Cuál definición considera más exacta para saber la calidad de un monitor? Argumente su respuesta e investigue y anote la resolución del monitor de su computadora principal o laptop así como su PPIs.

La resolución de una imagen es el número de pixeles por unidad de medida, usualmente el número de pixeles por pulgadas o pixeles por centimetro . Si tu imagen es 72 pixeles de ancho y le indicas que sea 72 pixeles por pulgada, entonces te estás refiriendo que es una pulgada de ancho.

6. Anote el número de grupo de la materia Computación Gráfica (Teoría) y su dirección de correo electrónico.

Grupo 1 ING. JOSE ROQUE ROMAN GUADARRAMA **ing.roquerg@gmail.com**

Conclusiones

OpenGL es una buena opción para generación de efectos visuales, escenas, etc, debido a su portabilidad, estandarización y potencia gráfica. OpenGL esta pasando a segundo plano ante los nuevos programas de diseño que realizan tareas automáticamente. Se prevee que con la versión 2.0 se produzca un relanzamiento.

Referencias

https://www.ecured.cu/IDE_de_Programaci%C3%B3n
https://hardzone.es/2018/03/03/resoluciones-monitor-cuales-son/

Reporte Práctica 1. (Individuales)

1. Cree proyectos y realice los pasos de configuración del ambiente de desarrollo en equipos de cómputo diferentes al del laboratorio. Con base en ello conteste:

a) ¿Pudo conseguir que el código de muestra funcione?

Sí, no tuve mayores dificultades en la instalación de Visual studio ni en la configuración de GLUT

b) Liste los problemas que tuvo o intentó resolver para hacer funcionar el código. (Recuerde que se debe comentar la variable **root**) no hubo mayores dificultades para hacer funcionar el codigo proporcionado para la práctica 1

2. De un breve comentario de la práctica uno, indicando los pasos que se le complicaron y sugerencias para mejorar desarrollo de la misma.

El profesor llevó a cabo el desarrollo de la práctica a una velocidad adecuada, donde todo se entendió y fue claro, no estoy muy seguro de como mejorar el desempeño de la clase. Tal vez un poco más dinámico con chistes o cosas así, pero el tema se entendió a la perfección

```
Padilla Herrera Carlos Ignacio
/* Este programa dibuja una "Tetera", este objeto esta definido
 * en GLUT, se crea una fuente de luz, y un material */
/***********2020-1******************/
//ALUMNO: Padilla Herrera Carlos Ignacio
//Incluimos las librerias
//#include <GL/glut.h>
//#include <stdlib.h>
#include "Main.h"

float rot = 0.0f;

void init(void)
{
	// Ubicamos la fuente de luz en el punto (1.0, 1.0, 1.0)
	//GLfloat light_position[] = { 1.0, 1.0, 1.0, 0.0 };

	// Activamos la fuente de luz
	glEnable(GL_LIGHTING);
	glEnable(GL_LIGHT0);

	glClearDepth(1.0f);					// Activamos el valor de inicio
del buffer de profundidad
	glEnable(GL_DEPTH_TEST);				// Hacemos la prueba
de profundidad
	glDepthFunc(GL_LEQUAL);				// Tipo de prueba de
profundidad a hacer
	return;
}

void reshape(int w, int h)
{
	if (!h)
		return;

	glViewport(0, 0,  (GLsizei) w, (GLsizei) h);
	// Activamos la matriz de proyeccion.
	glMatrixMode(GL_PROJECTION);
	// "limpiamos" esta con la matriz identidad.
	glLoadIdentity();
	// Usamos proyeccion ortogonal
	 //glOrtho(-200, 200, -200, 200, -200, 200);
	gluPerspective(30.0f, (GLfloat)800/(GLfloat)600, 0.03, 1000.0);
	// Activamos la matriz de modelado/visionado.
	glMatrixMode(GL_MODELVIEW);
	// "Limpiamos" la matriz
	glLoadIdentity();
	return;
}
```

```
// Aqui ponemos lo que queremos dibujar.
void display(void)
{
	// Propiedades del material
	GLfloat mat_ambient[] = { 0.7f, 0.7f, 0.7f, 1.0f };
	GLfloat mat_diffuse[] = { 1.0f,1.0f, 0.0f, 1.0f };
	GLfloat mat_specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };
	GLfloat mat_shininess[] = { 100.0f };

	GLfloat mat_diffuse2[] = { 0.0f,1.0f, 0.0f, 1.0f };

	// "Limpiamos" el frame buffer con el color de "Clear", en este
	// caso negro.
	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

	glMatrixMode( GL_MODELVIEW_MATRIX );
	glLoadIdentity();

	glTranslatef(0.0,0.0,-20.0);
	// Rotacion de 30 grados en torno al eje x
	glRotated(30.0 + rot, 1.0, 0.0, 0.0);
	// Rotacion de -30 grados en torno al eje y
	//glRotated(-30.0, 0.0, 1.0, 0.0);
	glRotated(-30, 0.0, 1.0, 0.0);
	// Dibujamos una "Tetera" y le aplico el material
	//aquí le aplica al material mat ambient y así

	glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
	glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
	glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
	glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);
	glutSolidTeapot(1.0);
	glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse2);
	glutWireCube(2.0f);
	//cubo dibujo lineas
	//gluit solid teapot dibuja la tetera. recibe como argumento el radio de la tetera.
quiero construir una tetera con radio de una unidad
	//
	//glFlush();
	glutSwapBuffers ( );
	return;
}

// Termina la ejecucion del programa cuando se presiona ESC
void keyboard(unsigned char key, int x, int y)
{
	switch (key)
	{
		case 27: exit(0);
				break;
```

```
                case 'x':
                        rot += 10;
                        // hace un incremento
                        break;
                case 'X':
                        rot -= 10;
                        // hace un incremento
                        break;
        }
         glutPostRedisplay();
         //genera un evento de dibujo. Vuelve a dibujar lo qu ehay en el escenario.
         //
         return;
}

// Main del programa.
int main(int argc, char **argv)
{
        // Inicializo OpenGL
        //crea un canvas mediante glut
         glutInit(&argc, argv);

         // Activamos buffer simple y colores del tipo RGB
         //glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB| GLUT_DEPTH);
         //Indica los recursos para canvas de dibujo. Le indica trabajar con un buffer doble.
Opengl tiene muchos buffers, estos de aqui son para almacenar información
         //buffer doble para trabajar en paralelo
         glutInitDisplayMode (GLUT_DOUBLE | GLUT_RGB| GLUT_DEPTH);

         // Definimos una ventana de medidas 300 x 300 como ventana
         // de visualizacion en pixels
         //indica el tamaaño inicial de mi ventana
         glutInitWindowSize (300, 300);

         // Posicionamos la ventana en la esquina superior izquierda de
         // la pantalla.
         // Se refiere a la esquina superior izquierda de la ventana
         glutInitWindowPosition (300, 400);

         // Creamos literalmente la ventana y le adjudicamos el nombre que se
         // observara en su barra de titulo.
         glutCreateWindow ("Tetera");

         // Inicializamos el sistema
         // init puede llamarse como guste
         init();
         //callbacks
         //funciones para mandar a  llamar otra funcion
```

```
        //cuando glut determine que se cre+o un evento de dibujo , este glut lo va a
atrapar
        //
        glutDisplayFunc(display);
        glutReshapeFunc(reshape);
        glutKeyboardFunc(keyboard);
        glutMainLoop();

        // ANSI C requiere que main retorne un valor entero.
        return 0;
}
```