Universidad Autónoma de querétaro

Programación

Facultad de Ingeniería

Práctica 4: Cuento

Integrantes:
Rodríguez Ruíz Stefanny,
Mendoza Frías Luis
Fernando, Martínez Barras
Alexis, Mandujano Jiménez
Daniel Cruz

Ingeniería en Nanotecnología
Profesor: José de Jesús
Santana Ramírez

Abril 11, 2019

1 Objetivo

Realizar un cuento utilizando la librería graphics.h y todas sus funciones.

2 Introducción

Las imágenes gráficas mostradas en un monitor de computadora se componen de pequeños puntos llamados píxeles, los cuales están distribuidos en la pantalla en filas; existe una cantidad específica de filas y cada fila tiene una cantidad específica de píxeles. La cantidad de píxeles usada en la pantalla se conoce como resolución. Cada modo gráfico tiene una resolución particular.
Para habilitar el monitor en modo gráfico y utilizar sus píxeles y funciones de gráficos, es necesario incluir el encabezado include ¡graphics.h¿ que contiene las declaraciones y funciones relacionadas con graficación e inicializar el monitor en modo gráfico y utilizar sus píxeles con la función initgraph().

3 Diagrama de flujo

INICIO

Llamar funciones

Void display

Void texto

Desplegar 1^{er} escena (texto)

Void Control Raton

Ir sumando clicks

Si

a=1

Desplegar nueva escena

a=2

Nueva escena

a=10

Cierra ventana

Fin

Hasta...

3.1 Código

```
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glut.h>
#include <iostream>

using namespace std;

int a=0;
```

```
void display(void);

void printtext(int x, int y, string String);


void ControlRaton( int button, int state, int x, int y ){
    if (button==GLUT_LEFT_BUTTON && state==GLUT_UP){
        a=a+1;
        cout<<a<<endl;
        if(a==1){
            glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
            glEnable(GL_DEPTH_TEST);

            glBegin(GL_POLYGON);
            glColor3f(0.0, 1.0, 0.0);
            glVertex2f(-1.0f,1.0f);
            glColor3f(1.0, 1.0, 0.0);
            glVertex2f(1.0f,1.0f);
            glColor3f(0.0, 0.0, 1.0);
            glVertex2f(1.0f,-1.0f);
            glColor3f(0.0, 1.0, 1.0);
            glVertex2f(-1.0f,-1.0f);
            glEnd();
            glFlush();

            glColor3f(1.0, 1.0, 1.0);
            char string[64];
            sprintf(string, "Habia una vez una vaca que queria
               ser mas que una vaca ");
            printtext(60,350,string);

            glutSwapBuffers();
        }
        if(a==2){
            glClearColor(0,1,1,1);
            glClear(GL_COLOR_BUFFER_BIT);
            glMatrixMode(GL_PROJECTION);
            glLoadIdentity();

            //CASA
            glBegin(GL_POLYGON);
            glColor3f(0.0, 1.0, 0.0);
            glVertex2f(1.0f,-1.0f);
            glVertex2f(1.0f,0.0f);
            glVertex2f(-1.0f,0.0f);
            glVertex2f(-1.0f,-1.0f);
            glEnd();
```

```
//SOL
glPushMatrix();
glTranslatef(1.0,1.0,0.0);
glRotatef(45, 0, 1, 1);
glScalef(1.0, 1.0, 1.0);
glColor3f(2.0, 5.0, 0.0);
glutSolidSphere(0.3,20,20);
glPopMatrix();

//NUBE
glPushMatrix();
glTranslatef(-0.5,0.8,0.0);
glRotatef(45, 0, 1, 1);
glScalef(1.0, 1.0, 1.0);
glColor3f(1.0, 1.0, 1.0);
glutSolidSphere(0.08,20,20);
glPopMatrix();

glPushMatrix();
glTranslatef(-0.42,0.8,0.0);
glRotatef(45, 0, 1, 1);
glScalef(1.0, 1.0, 1.0);
glColor3f(1.0, 1.0, 1.0);
glutSolidSphere(0.08,20,20);
glPopMatrix();

glPushMatrix();
glTranslatef(-0.34,0.8,0.0);
glRotatef(45, 0, 1, 1);
glScalef(1.0, 1.0, 1.0);
glColor3f(1.0, 1.0, 1.0);
glutSolidSphere(0.08,20,20);
glPopMatrix();

glPushMatrix();
glTranslatef(-0.58,0.8,0.0);
glRotatef(45, 0, 1, 1);
glScalef(1.0, 1.0, 1.0);
glColor3f(1.0, 1.0, 1.0);
glutSolidSphere(0.08,20,20);
glPopMatrix();

//TECHO DE LA CASA
glBegin(GL_POLYGON);
glColor3f(1.0, 0.5, 1.0);
glVertex2f(-0.75f,-0.25f);
glVertex2f(-0.3f,0.15f);
```

```
glVertex2f(0.15f,-0.25f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(0.5, 0.3, 0.5);
glVertex2f(0.1f,-0.9f);
glVertex2f(0.1f,-0.25f);
glVertex2f(-0.7f,-0.25f);
glVertex2f(-0.7f,-0.9f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.0, 0.0, 0.5);
glVertex2f(-0.05f,-0.9f);
glVertex2f(-0.05f,-0.5f);
glVertex2f(-0.25f,-0.5f);
glVertex2f(-0.25f,-0.9f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(0.0, 5.0, 1.0);
glVertex2f(-0.45f,-0.45f);
glVertex2f(-0.65f,-0.45f);
glVertex2f(-0.65f,-0.65f);
glVertex2f(-0.45f,-0.65f);
glEnd();

//LINEAS
glBegin(GL_LINES);
glColor3f(0.0, 0.0, 0.0);
glVertex2f(-0.55f, -0.45f);
glVertex2f(-0.55f, -0.65f);
glEnd();

glBegin(GL_LINES);
glColor3f(0.0, 0.0, 0.0);
glVertex2f(-0.65f, -0.55f);
glVertex2f(-0.45f, -0.55f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.0, 1.0, 1.0);
glVertex2f(0.65f,-0.8f);
glVertex2f(0.65f,-0.7f);
glVertex2f(0.25f,-0.7f);
glVertex2f(0.25f,-0.8f);
glEnd();
```

```
glBegin(GL_POLYGON);
glColor3f(1.0, 1.0, 1.0);
glVertex2f(0.8f,-0.7f);
glVertex2f(0.8f,-0.65f);
glVertex2f(0.65f,-0.65f);
glVertex2f(0.65f,-0.7f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.0, 1.0, 1.0);
glVertex2f(0.35f,-0.9f);
glVertex2f(0.35f,-0.8f);
glVertex2f(0.3f,-0.8f);
glVertex2f(0.3f,-0.9f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.0, 1.0, 1.0);
glVertex2f(0.45f,-0.9f);
glVertex2f(0.45f,-0.8f);
glVertex2f(0.4f,-0.8f);
glVertex2f(0.4f,-0.9f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.0, 1.0, 1.0);
glVertex2f(0.55f,-0.9f);
glVertex2f(0.55f,-0.8f);
glVertex2f(0.5f,-0.8f);
glVertex2f(0.5f,-0.9f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.0, 1.0, 1.0);
glVertex2f(0.65f,-0.9f);
glVertex2f(0.65f,-0.8f);
glVertex2f(0.6f,-0.8f);
glVertex2f(0.6f,-0.9f);
glEnd();

glBegin(GL_POLYGON);
glColor3f(1.0, 1.0, 1.0);
glVertex2f(0.7f,-0.65f);
glVertex2f(0.7f,-0.6f);
glVertex2f(0.65f,-0.6f);
glVertex2f(0.65f,-0.65f);
glEnd();
```

```
        glBegin(GL_POLYGON);
        glColor3f(0.0, 0.0, 0.0);
        glVertex2f(0.6f,-0.65f);
        glVertex2f(0.65f,-0.65f);
        glVertex2f(0.65f,-0.6f);
        glEnd();

        glBegin(GL_POLYGON);
        glColor3f(0.0, 0.0, 0.0);
        glVertex2f(0.3f,-0.75f);
        glVertex2f(0.3f,-0.7f);
        glVertex2f(0.25f,-0.7f);
        glVertex2f(0.25f,-0.75f);
        glEnd();

        glBegin(GL_POLYGON);
        glColor3f(0.0, 0.0, 0.0);
        glVertex2f(0.4f,-0.8f);
        glVertex2f(0.4f,-0.75f);
        glVertex2f(0.35f,-0.75f);
        glVertex2f(0.35f,-0.8f);
        glEnd();

        glBegin(GL_POLYGON);
        glColor3f(0.0, 0.0, 0.0);
        glVertex2f(0.5f,-0.75f);
        glVertex2f(0.5f,-0.7f);
        glVertex2f(0.45f,-0.7f);
        glVertex2f(0.45f,-0.75f);
        glEnd();

        glBegin(GL_POLYGON);
        glColor3f(0.0, 0.0, 0.0);
        glVertex2f(0.6f,-0.8f);
        glVertex2f(0.6f,-0.75f);
        glVertex2f(0.55f,-0.75f);
        glVertex2f(0.55f,-0.8f);
        glEnd();

        glFlush();
    }
    if(a==3){
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glEnable(GL_DEPTH_TEST);
        glBegin(GL_POLYGON);
        glColor3f(0.0, 1.0, 0.0);
        glVertex2f(-1.0f,1.0f);
        glColor3f(1.0, 1.0, 0.0);
```

```
        glVertex2f(1.0f,1.0f);
        glColor3f(0.0, 0.0, 1.0);
        glVertex2f(1.0f,-1.0f);
        glColor3f(0.0, 1.0, 1.0);
        glVertex2f(-1.0f,-1.0f);
        glEnd();
        glFlush();

        glColor3f(1.0, 1.0, 1.0);

    }
```

3.2 Terminal