Moreno Rodríguez Fernando Iván
Gpo 9
315295447

# UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO


Proyecto Final – Manual Técnico

Moreno Rodríguez Fernando Iván
Gpo 9
315295447

Objetivos

I. Aplicar y demostrar los conocimientos adquiridos durante el curso de Computación Grafica e Interacción Humano Computadora.
II. Plasmar y conjuntar herramientas de computación grafica en OpenGL en el entorno de Visual Studio.
III. Aprender a manejar software con herramientas graficas 3D como Blender o Maya, así como modelar y exportar objetos con ayuda de estos.

Alcance del Proyecto

Como todo proyecto, podemos administrar mejor el tiempo si este lo dividimos en subtareas, de las cuales nos encargaremos para poder dar como terminado este trabajo.

- Familiarización con entorno y herramientas
- Modelado Fachada
- Modelado Objetos
- Ambientación
- Animaciones
- Casos de prueba
- Exportación y creación de ejecutable
- Documentación

Tanto los modelados de los objetos como sus animaciones pueden ser las más complicadas. Se designarán las horas respectivas a cada una de estas tareas

- Familiarización con entorno y herramientas → 8 horas
- Modelado Fachada → 4 horas
- Modelado Objetos → 1 c/u → 7 horas
- Ambientación → 2 horas
- Animaciones → 7 horas
- Casos de prueba → 6 horas
- Exportación y creación de ejecutable → 1 horas

- Documentación → 3 horas

Esto da un total de 38 horas, las cuales, si fuese un entregable para una empresa, se convertirían en 50 aproximadamente, procurando y dejando tiempo suficiente para algún percance.

Preferimos aterrizar la información en el Diagrama de Gantt de la siguiente forma. Entonces, diremos que se extenderán las fechas de la siguiente manera:

- Familiarización con entorno y herramientas, modelado fachada, modelado objetos – **23 septiembre a 31 de septiembre**
- Ambientación, animaciones y casos de prueba – **1 noviembre 15 de noviembre**
- Exportación, creación de ejecutable y documentación – **16 noviembre a 21 noviembre**

La fecha de inicio la dimos a mediados de noviembre, debido a que en ese momento considero que tengo los conocimientos necesarios para ir empezando parte de este proyecto. El proyecto finalizaría el día 21 de noviembre.

Documentación de código

Librerías y extensiones ocupadas:

```
#include <iostream>
#include <cmath>
#include <string>
#include <windows.h>

// GLEW
#include <GL/glew.h>

// GLFW
#include <GLFW/glfw3.h>

// Other Libs
#include "stb_image.h"

// GLM Mathematics
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>

//Load Models
#include "SOIL2/SOIL2.h"


// Other includes
#include "Shader.h"
#include "Camera.h"
#include "Model.h"
#include "Texture.h"

```

Moreno Rodríguez Fernando Iván
Gpo 9
315295447

Funciones para animaciones y atributos de cámara.

```
// Function prototypes
void KeyCallback(GLFWwindow *window, int key, int scancode, int action, int mode);
void MouseCallback(GLFWwindow *window, double xPos, double yPos);
void DoMovement();
void animacion();
void animacionpelota();
void animacionbandera();
void animaciontecho();
void animacionpuerta();
void animacionanotacion();


// Window dimensions
const GLuint WIDTH = 800, HEIGHT = 600;
int SCREEN_WIDTH, SCREEN_HEIGHT;

// Camera
Camera  camera(glm::vec3(0.0f, 3.0f, 0.0f));
GLfloat lastX = WIDTH / 2.0;
GLfloat lastY = HEIGHT / 2.0;
bool keys[1024];
bool firstMouse = true;
float range = 0.0f;
float rot = 90.0f;
```

Inicialización de variables encargadas de animaciones

```
//Animacion pelota
float movBallX = 0.0;
float movBallY = 0.0;
float rotKit = 90.0;

bool ball_on = false;
bool recorrido1balon = true;
bool recorrido2balon = false;
bool recorrido3balon = false;

//Animacion banderas
float SubirBanderas = 0.0;
bool bandera_enable = false;
bool arriba_bandera = true;
bool abajo_bandera = false;

//Animacion techo
float Vtecho = 0.0;
bool techo_enable = false;
bool abrir_techo = true;
bool cerrar_techo = false;

//Animacion puertas
float MovPuerta = 0;
bool puerta_enable = false;
bool arriba_puerta = false;
bool abajo_puerta = true;


//Animacion anotacion
float scTD = 0;
float tTD = 0;
float scDW = 0;
float tDW = 0;
bool td_enable = false;
bool mov1 = true;
bool mov2 = false;
bool mov3 = false;
```

Moreno Rodríguez Fernando Iván
Gpo 9
315295447

Para este caso, utilizamos valores flotantes, encargados de ser los que van recreando el movimiento, y de booleanos sirviéndonos como delegación de recorridos y poder gestionar que movimiento es el que se desea.

El esqueleto del código utilizado fue recuperado de la practica ultima de la materia Computación Grafica e Interacción Humano-computadora en el semestre en curso (2022-1).

Gran parte de esto fue visto múltiples veces en clase a lo largo del semestre.

La estructura definida para guardar las keyframes son todos los atributos del personaje en movimiento:

```
int i_curr_steps = 0;
typedef struct _frame
{
    //Variables para GUARDAR Key Frames
    float posX;     //Variable para PosicionX
    float posY;     //Variable para PosicionY
    float posZ;     //Variable para PosicionZ
    float incX;     //Variable para IncrementoX
    float incY;     //Variable para IncrementoY
    float incZ;     //Variable para IncrementoZ
    float rotRodIzq;
    float rotRodDer;
    float rotCodoDer;
    float rotCodoIzq;
    float rotInc;
    float rotInc2;
    float rotInc3;
    float rotInc4;
```

Para poder cargar las keyframes de la forma adecuada, primero anime los cuadros necesarios en OpenGL e imprimí dichas posiciones; inicialice un arreglo de 10 keyframes en 0 y posterior a ello las llene con la información necesaria. De este modo, cargamos desde un inicio los frames que recorrerá el personaje:

```
void saveFrame(void)
{

    printf("frameindex %d\n", FrameIndex);

    KeyFrame[FrameIndex].posX = posX;
    KeyFrame[FrameIndex].posY = posY;
    KeyFrame[FrameIndex].posZ = posZ;

    KeyFrame[FrameIndex].rotRodIzq = rotRodIzq;
    KeyFrame[FrameIndex].rotRodDer = rotRodDer;
    KeyFrame[FrameIndex].rotCodoDer = rotCodoDer;
    KeyFrame[FrameIndex].rotCodoIzq = rotCodoIzq;

    printf("%f %f %f %f %f %f",posX,posY,posZ,rotRodIzq,rotRodDer,rotCodoDer);

    FrameIndex++;
}
```

```
for(int i=0; i<MAX_FRAMES; i++)
{
    KeyFrame[i].posX = 0;
    KeyFrame[i].incX = 0;
    KeyFrame[i].incY = 0;
    KeyFrame[i].incZ = 0;
    KeyFrame[i].rotRodIzq = 0;
    KeyFrame[i].rotRodDer = 0;
    KeyFrame[i].rotCodoDer = 0;
    KeyFrame[i].rotCodoIzq = 0;
    KeyFrame[i].rotInc = 0;
    KeyFrame[i].rotInc2 = 0;
    KeyFrame[i].rotInc3 = 0;
    KeyFrame[i].rotInc4 = 0;
}

KeyFrame[0].posX = 60.0;
KeyFrame[0].posY = 1.0;
KeyFrame[0].posZ = 0.0;
KeyFrame[0].rotRodIzq = 0.0;
KeyFrame[0].rotRodDer = 0.0;
KeyFrame[0].rotCodoDer = 0.0;
KeyFrame[0].rotCodoIzq = 0.0;

KeyFrame[1].posX = 60.0;
KeyFrame[1].posY = 1.0;
KeyFrame[1].posZ = 0.0;
KeyFrame[1].rotRodIzq = 80.0 ;
KeyFrame[1].rotRodDer = -45.0;
KeyFrame[1].rotCodoDer = -45.0;
KeyFrame[1].rotCodoIzq = 45.0;

KeyFrame[2].posX = 65.3999;
KeyFrame[2].posY = 1.0;
KeyFrame[2].posZ = 0.0999;
KeyFrame[2].rotRodIzq = -45.0;
KeyFrame[2].rotRodDer = 80.0;
KeyFrame[2].rotCodoDer = 80.0;
KeyFrame[2].rotCodoIzq = -80.f;
```

Para los modelos, utilizamos en total 19 objetos:

```
Shader lightingShader("Shaders/lighting.vs", "Shaders/lighting.frag");
Shader lampShader("Shaders/lamp.vs", "Shaders/lamp.frag");
Shader SkyBoxshader("Shaders/SkyBox.vs", "Shaders/SkyBox.frag");
Model BotaDer((char*)"Models/Personaje/bota.obj");
Model PiernaDer((char*)"Models/Personaje/piernader.obj");
Model PiernaIzq((char*)"Models/Personaje/piernaizq.obj");
Model Torso((char*)"Models/Personaje/torso.obj");
Model BrazoDer((char*)"Models/Personaje/brazoder.obj");
Model BrazoIzq((char*)"Models/Personaje/brazoizq.obj");
Model Cabeza((char*)"Models/Personaje/cabeza.obj");
Model Football((char*)"Models/Football/football.obj");
Model Estadio((char*)"Models/stadium/estadio.obj");
Model Goal((char*)"Models/Goal/Goal.obj");
Model Letras((char*)"Models/Letras/Letras.obj");
Model Flag((char*)"Models/Flag/flag.obj");
Model Board((char*)"Models/Board/board.obj");
Model Poste((char*)"Models/Pole/poste.obj");
Model Roof((char*)"Models/stadium/roof.obj");
Model Puertas((char*)"Models/stadium/rejas.obj");
Model Touch((char*)"Models/Board/td.obj");
Model Down((char*)"Models/Board/td2.obj");
Model Banca((char*)"Models/bench/bench2.obj");
// Build and compile our shader program
```

Moreno Rodríguez Fernando Iván
Gpo 9
315295447

Para el skybox, se utilizo una imagen para simular un paisaje de nubes en plena luz del día:

```
// Load textures
vector<const GLchar*> faces;
faces.push_back("SkyBox/right.tga");
faces.push_back("SkyBox/left.tga");
faces.push_back("SkyBox/top.tga");
faces.push_back("SkyBox/bottom.tga");
faces.push_back("SkyBox/back.tga");
faces.push_back("SkyBox/front.tga");
```

La animación mas complicada es la del personaje al patear, puesto que se compone de dos animaciones. La primera es por keyframes donde el personaje simula el patear una pelota. Una vez que se detiene, empieza la siguiente animación mediante un bool, que es la pelota siendo pateada. Esta realiza dos recorridos: de forma ascendente y descendiente:

Moreno Rodríguez Fernando Iván
Gpo 9
315295447

```
//Movimiento del personaje

if (play)
{
    if (i_curr_steps >= i_max_steps) //end of animation between frames?
    {
        playIndex++;
        if (playIndex>FrameIndex - 2)   //end of total animation?
        {
            printf("termina anim\n");
            playIndex = 0;
            play = false;
            ball_on = true;
        }
        else //Next frame interpolations
        {
            i_curr_steps = 0; //Reset counter
                              //Interpolation
            interpolation();
        }
    }
    else
    {
        //Draw animation
        posX += KeyFrame[playIndex].incX;
        posY += KeyFrame[playIndex].incY;
        posZ += KeyFrame[playIndex].incZ;


        rotRodIzq += KeyFrame[playIndex].rotInc;
        rotRodDer += KeyFrame[playIndex].rotInc2;
        rotCodoDer += KeyFrame[playIndex].rotInc3;
        rotCodoIzq += KeyFrame[playIndex].rotInc4;
        i_curr_steps++;
    }
```

```
void animacionpelota()
{

    //Movimiento de la bola
    if (ball_on)
    {
        if (recorrido1balon)
        {
            movBallY += 0.005f;
            movBallX += 0.0075f;
            if (movBallY > 4)
            {
                recorrido1balon = false;
                recorrido2balon = true;
            }
        }
        if (recorrido2balon)
        {
            movBallY -= 0.005f;
            movBallX += 0.005f;
            if (movBallY <= 0)
            {
                recorrido2balon = false;
                recorrido3balon = true;
            }
        }
        if (recorrido3balon)
        {
            movBallX = 0.0f;
            movBallY = 0.0f;
            recorrido3balon = false;
            recorrido1balon = true;
            ball_on = false;
        }

    }
}
```