Universidad Nacional Autónoma de México

Facultad de Ingeniería

“Proyecto Final – Modelo Casa
Manual Técnico”

Asignatura: LABORATORIO DE COMPUTACIÓN GRÁFICA

Grupo:

Profesor: ING. CARLOS ALDAIR ROMAN BALBUENA

Alumno:

García Quezada Cristian Gabriel

Semestre: 2022-1
Fecha de entrega:21/11/2021

Objetivo

El alumno deberá aplicar y demostrar los conocimientos adquiridos durante todo el curso.

Descripción

El alumno deberá seleccionar una fachada y un espacio que pueden ser reales o ficticios y presentar imágenes de referencia de dichos espacios para su recreación 3D en OpenGL.

En la imagen de referencia se debe visualizar 7 objetos que el alumno va a recrear virtualmente y donde dichos objetos deben ser lo más parecido a su imagen de referencia, así como su ambientación.

Alcance del proyecto

Para el proyecto se espera tardarse un periodo de 2 o 3 semanas, ya que la primera se dedicará totalmente a elegir lo que se modelará y aprender un poco de maya, desde un inicio se sabe que será complejo y no se aprenderá todo lo que se puede hacer en Maya, además no poder hacerlo perfectamente ya que es algo que requiere practica constante.

Como resultado se espera una casa similar a la de la propuesta, aunque claro se harán algunas modificaciones, después de todo solo es de referencia, para la cantidad de objetos modelados se crearan unos pocos mas ya que se tratara de cumplir con la cantidad solicitada para el proyecto de teoría (que son más objetos), además de crear directamente 2 cuartos en vez de 1, esas cosas con el objetivo de que el proyecto de teoría se facilite más, ya que solo faltarían animaciones.

Limitantes

La principal limitante es maya, ya que hace falta practica para poder sacarle todo el provecho posible, en eso va incluido el modelado y el texturizado adecuado, también incluido que algunas veces desde mi punto de vista ya esta bien pero puede que no del todo. Otra limitante podría ser en las animaciones complejas ya que no sé exactamente como podrían hacerse, pero intentare que salgan lo mejor posible.

En caso de que se desee descargar el proyecto completo de github y ejecutar el visual, por razones desconocidas se modifica el tipo de arquitectura a x64, por lo que se debe poner en x86 (esto se modifica se encuentra al lado de Depurar)

****No es limitante en sí, pero se debe mencionar que debido a la cantidad y peso de los modelos e imágenes tarda un poco en cargar la ventana****

Propuesta

La casa que se usara de referencia es la que se ve a continuación, la cual fue sacada directo de internet, además debido a su forma se decidió implementar dos cuartos para aprovechar el espacio, los cuales también se dejan a continuación e igualmente fueron sacados de internet.

Casa parte exterior

dormitorio

sala

Cabe mencionar que en ningún lado de las imágenes se ve una puerta por lo que se pondrá en la parte posterior de la casa, además de otra pequeña puerta que haga la conexión entre los dos cuartos.

Los objetos que se decidió modelar son los siguientes :

Dormitorio

- cama

- librero

-estantería

-lampara (esta se modificará para que se vea diferente)

-sillón

- banco

-*** con posibilidad a los cuadros****

Sala

- lampará (pero se usará el modelo anterior)

-sillón pequeño (pero se usará el modelo anterior)

-sillón grande (este se hará con el mismo diseño y textura que el anterior)

-florero

-repisa

-estantería

-tapete

*Además se crearán otros modelos como una mesa y sillas, pero de esos no hay referencia así que serán vistos en el ejecutable

Documentación

Primeramente se muestran los encabezados que se usaran en el programa para usar las librerías

```
#include <iostream>
#include <cmath>

// 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"
```

Posteriormente se configura el tamaño de la pantalla que será de 800*600, además de algunas funciones que se usaran, incluida la cámara y variables que ocuparemos en el proyecto

```
// Function prototypes
void KeyCallback(GLFWwindow* window, int key, int scancode, int action, int mode);
void MouseCallback(GLFWwindow* window, double xPos, double yPos);
void DoMovement();

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

// Camera
Camera  camera(glm::vec3(-2.0f, 2.0f, -15.0f));//-50.0f, 1.0f, -50.0f));camx,camy,
GLfloat lastX = WIDTH / 2.0;
GLfloat lastY = HEIGHT / 2.0;
bool keys[1024];
bool firstMouse = true;
float range = 0.0f;
float spotAngle = 0.0f;
// Light attributes
glm::vec3 lightPos(0.0f, 0.0f, 0.0f);
bool active;
```

```
//Variables de animacion
bool anim = false;
float posicion = 0.0;
float rot = 0.0f;
float movx = 0.0f;
float movz = 0.0f;
//float rot = 0.0f;
float rot1 = 90.0f;
float movsilla_x = 0.0f;
bool tec = false;
float movluz_y = 0.0f;
bool luz = false;
float movluz_z = 0.0f;
bool luz1 = false;
float movsab_y = 0.0f;
bool sab = false;

bool recorrido1=true;
bool recorrido2=false;
bool recorrido3=false;
bool recorrido4=false;
bool recorrido5=false;
float rotvehiculo = 0.0;
```

A continuación se muestra la función por medio de la cual se crea la ventana

```
// Create a GLFWwindow object that we can use for GLFW's functions
GLFWwindow* window = glfwCreateWindow(WIDTH, HEIGHT, "Iluminacion 2", nullptr, nullptr);

if (nullptr == window)
{
    std::cout << "Failed to create GLFW window" << std::endl;
    glfwTerminate();

    return EXIT_FAILURE;
}
```

Los shaders que se ocuparan así como modelos empleados son los siguientes

faltan algunos modelos que se ocuparon, pero en el código del proyecto están completos

```
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 Piso((char*)"Models/Esfera/Piso.obj");
Model sabana((char*)"models/cama/sabana.obj");
Model cama((char*)"models/cama/cama.obj");
Model base((char*)"models/cama/base.obj");
Model lampara((char*)"models/lampara/lampara.obj");
Model librero((char*)"models/librero/librero.obj");
Model mueble((char*)"models/mueble/mueble.obj");
Model banco((char*)"models/banco/banco.obj");
Model repisa((char*)"models/repisa/repisa.obj");
Model sillonc((char*)"models/sillonc/sillonc.obj");
Model sillong((char*)"models/sillong/sillong.obj");
Model casa((char*)"models/casa/casa.obj");
Model techo((char*)"models/techo/techo.obj");
Model mesa((char*)"models/mesa/mesa.obj");
Model silla((char*)"models/silla/silla.obj");
Model silla2((char*)"models/silla2/silla2.obj");
Model arbol((char*)"models/arbol/arbol.obj");
```

Seguido a lo anterior el VBO y el VAO para los skybox, así mismo como la llamada a sus caras para texturizar **el skybox elegido fue principalmente porque se buscó varios pero todos se pixeleaban mucho, por lo que se veía mal, así que se decidió usar el mostrado más adelante **

```
//SkyBox
GLuint skyboxVBO, skyboxVAO;
glGenVertexArrays(1, &skyboxVAO);
glGenBuffers(1, &skyboxVBO);
glBindVertexArray(skyboxVAO);
glBindBuffer(GL_ARRAY_BUFFER, skyboxVBO);
glBufferData(GL_ARRAY_BUFFER, sizeof(skyboxVertices), &skyboxVertices, GL_STATIC_DRAW);
glEnableVertexAttribArray(0);
glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 3 * sizeof(GLfloat), (GLvoid*)0);

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

GLuint cubemapTexture = TextureLoading::LoadCubemap(faces);
```

También es importante mencionar la creación de los objetos, pero solo se pondrá un ejemplo ya que son varios los modelos que se crearon. Como podemos observar se muestra con todas las transformaciones a realizar para dicho modelo, las cuales consistirán en traslación, escala y rotación.

```
model = glm::mat4(1);                                                                    /////repisa sala
model = glm::translate(model, glm::vec3(-10.0f, 6.0f, -16.0f));
model = glm::scale(model, glm::vec3(0.7f, 0.4f, 0.4f));
model = glm::rotate(model, glm::radians(rot1), glm::vec3(0.0f, 90.0f, 0.0f));
//glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
glUniform1f(glGetUniformLocation(lightingShader.Program, "activatransparencia"), 0);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
repisa.Draw(lightingShader);
```

Ahora se muestran los controles de la cámara que servirán para controlar la visualización

```
// Camera controls
if (keys[GLFW_KEY_W] || keys[GLFW_KEY_UP])
{
    camera.ProcessKeyboard(FORWARD, deltaTime);
}

if (keys[GLFW_KEY_S] || keys[GLFW_KEY_DOWN])
{
    camera.ProcessKeyboard(BACKWARD, deltaTime);
}

if (keys[GLFW_KEY_A] || keys[GLFW_KEY_LEFT])
{
    camera.ProcessKeyboard(LEFT, deltaTime);
}

if (keys[GLFW_KEY_D] || keys[GLFW_KEY_RIGHT])
{
    camera.ProcessKeyboard(RIGHT, deltaTime);
}
```

Finalmente se deja la parte de las animaciones, pero también se pondrá un ejemplo.

Por el lado de animacion sencilla se tiene el movimiento de una silla, el cual se hará de la siguiente forma

-primeramente en la parte de DoMovement

```
////////////////////animacion silla/////////////////////////
if (tec) {
    if (movsilla_x >= -5.0f)
    {
        movsilla_x = movsilla_x - 0.1f;
    }
}
else {
    if (movsilla_x <= 0.0f)
    {
        movsilla_x = movsilla_x + 0.1f;
    }
}
////////////////////////////////////////////////////////////
```

-mas abajo en la parte de keycallback se coloca lo siguiente

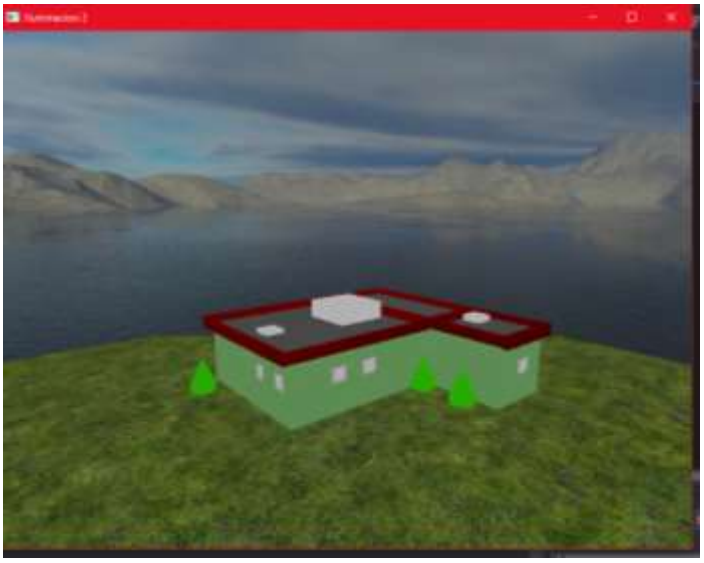
```
if (keys[GLFW_KEY_X])       //silla sala
{
    if (!tec)
        tec = true;
    else
        tec = false;
}
```

-y finalmente para llevarla a cabo, se indica en que parte del código se realizaran las modificaciones

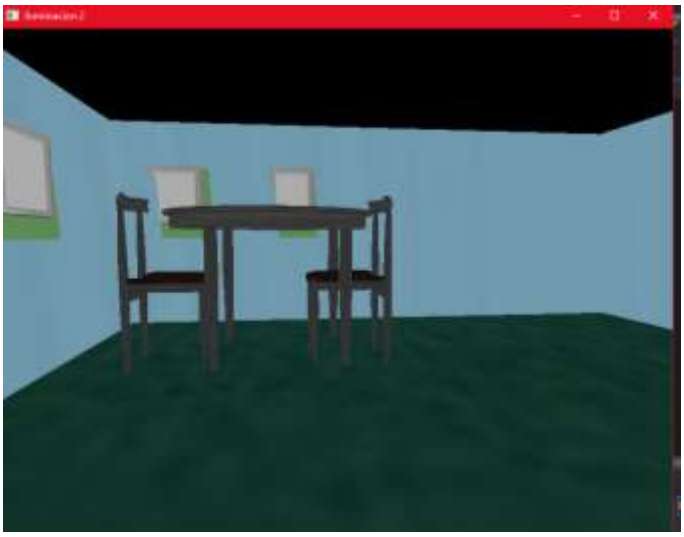
```
model = glm::mat4(1);                                                                    /////silla 2
model = glm::translate(model, glm::vec3(-5.0f + movsilla_x , 0.75f, 9.0f));
model = glm::scale(model, glm::vec3(0.88f, 0.88f, 0.88f));
//glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
glUniform1f(glGetUniformLocation(lightingShader.Program, "activatransparencia"), 0);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
```

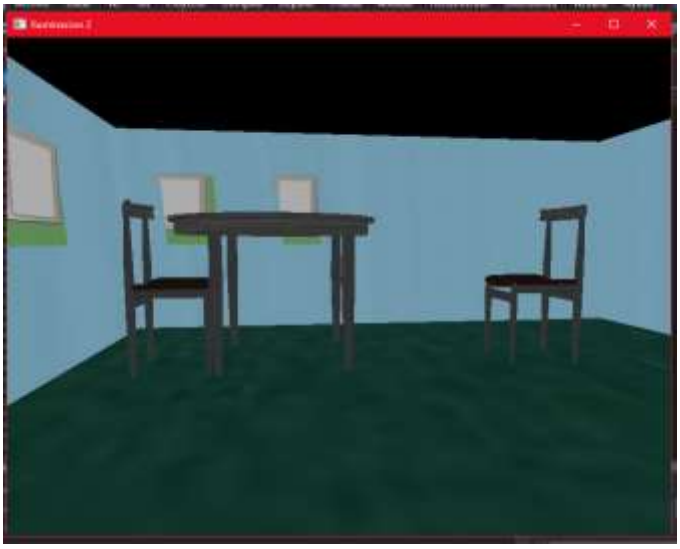
Una muestra de lo realizado es, para llevarlo a cabo es necesario presionar la tecla X para moverla, y después se puede volver a presionarla para regresar a su lugar original, esto simulara que alguien abre la silla para sentarse.

Primero se deja imagen de vista de la casa por fuera



Muestra de la casa, vista por fuera además de fondo un skybox





*Movimiento de la silla al presionar X *

Otras animaciones sencillas que se crearon fueron los movimientos de las luces, para la del cuarto es necesario presionar la tecla “Z”; para el movimiento de la luz de la lampara de la sala es con “C”; para el movimiento de la sabana de la cama es con “V”

Costos

Partiendo de un costo de $150*hora y de un costo de luz de 1.051 $/kWh y un iva de 0.16%, los costos resultantes son los siguientes

Tiempo	
Horas	48
precio x hora	150
precio horas	7200
Servicios	
luz	50.448
tomando 1.051 $/kWh	
Adicionales	
youtube	0
software	0
clases lab	0
Total	7250.448
iva	1160.07168
Total + iva	**8410.5197**

Diagrama de Gantt

ACTIVIDADES	SEMANAS							
	semana 1: 25-31 oct		semana 2: 1-7 nov		semana 3: 8-14 nov		semana 4: 15-21 nov	
Eleccion de casa y objetos a modelar								
Aprendizaje de maya								
Modelado de objetos								
Creacion de código								
Documentación de codigo								

Finalmente se dejan los enlaces de descarga:

Liga github: https://github.com/kurohaven1999/ProyectoLab_Grafica.git

Liga drive: https://drive.google.com/file/d/1b8QRM3xl_2VMXt4x5jSgUJBb6cADrD5Y/view?usp=sharing