Proyecto 2. Manual técnico

Objetivo

El alumno conocerá y aplicará las técnicas de modelado, implementación de modelos y animación de manera práctica demostrando su ingenio y creatividad.

Alcances:

Se planea realizar un ambiente virtual basado en un conjunto de imágenes guía que nos servirán para representar algún lugar en específico.

Cronograma:

Actividades	**Fechas**					
	Diciembre			**Enero**		
	6-10	**10-16**	**17-31**	**1-13**	**13-18**	**18-25**
Planeación						
Definición de objetivos y alcances	X					
Definición del cronograma	X					
Capacitación de software de modelado Blender		X				
Determinacion de objetos a modelar y cuales descargar.			x			
Modelado de la cama, el buró y la silla			x			
Modelado del sillon, el retrato y el teléfono				X		
Ejecución						
Obtención de los modelos: escalera, puerta y ventanas				X		
Obtención del modelo de refrigerador, cocina y detalles						X
Diseño de fachada				X		
Animación					X	
Respaldo de proyecto en github					X	

Análisis de costos:

Gastos únicos:

	Concepto
Computadora de desarrollo:	$15,000
Modelos:	$500
Software de modelado:	$0
Capacitación del software:	$1,000
Total	$16,500

Gastos mensuales:

	Concepto mensual	Meses de uso	Total
Luz	$30	2	$60
Internet	$200	2	$400
Alimentos(Snacks)	$500	2	$1,000
Transporte	$20	1	$20
		Total	$1,480

Costo total del proyecto: $17,980

Precio de venta: $35,000

Programa:

Para cargar los modelos, se utilizó el siguiente conjunto de instrucciones:

```
// Load models
Model casa( (char *)"Models/casa/casa.obj");
Model cajon((char*)"Models/buro/cajon.obj");
Model telefono((char*)"Models/telefono/telefono.obj");
Model retrato((char*)"Models/retrato/retrato.obj");
Model silla((char*)"Models/silla/silla.obj");
Model puertaPrin((char*)"Models/door/puerta.obj");
Model banquito((char*)"Models/banco/banco.obj");
Model jarron((char*)"Models/jarron/jarron.obj");
```

Para realizar las transformaciones en OpenGL se usaron las siguientes líneas:

Casa y entorno estatico:

```
// Draw the loaded model
glm::mat4 model(1);
model = glm::translate( model, glm::vec3( 0.0f, tras, 0.0f ) ); // Translate it down a bit so it's at the center of the scene
//model = glm::scale( model, glm::vec3( 0.02f, 0.02f, 0.02f ) );    // It's a bit too big for our scene, so scale it down
//model = glm::rotate(model, (float)glfwGetTime(), glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv( modelLoc, 1, GL_FALSE, glm::value_ptr( model ) );

casa.Draw( shader );
```

Cuadro de pared, con primitivas de OpenGL:

```
// Bind diffuse map
glActiveTexture(GL_TEXTURE0);
glBindTexture(GL_TEXTURE_2D, texture1);

glBindVertexArray(VAO);
model = glm::mat4(1);
glDrawElements(GL_TRIANGLES, 6, GL_UNSIGNED_INT, 0);

glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glBindVertexArray(0);
```

Para el cajon:

```
// Draw the loaded model
model= glm::mat4(1);
model = glm::translate(model, glm::vec3(0.0f, tras, 0.0f-movCajon)); // Translate it down a bit so it's at the center of the scene
//model = glm::scale( model, glm::vec3( 0.02f, 0.02f, 0.02f ) );    // It's a bit too big for our scene, so scale it down
//model = glm::rotate(model, (float)glfwGetTime(), glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));

cajon.Draw(shader);
```

El teléfono y el retrato:

```
// Draw the loaded model
model = glm::mat4(1);

model = glm::translate(model, glm::vec3(0.75945f, 4.5712f+tras, -1.4055f)); // Translate it down a bit so it's at the center of the scene
//model = glm::scale( model, glm::vec3( 0.02f, 0.02f, 0.02f ) );    // It's a bit too big for our scene, so scale it down
model = glm::rotate(model, glm::radians(rotLlamada), glm::vec3(1.0f, 0.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));

telefono.Draw(shader);

model = glm::mat4(1);
model = glm::translate(model, glm::vec3(0.417576f, 4.52347f + tras, -1.28148f)); // Translate it down a bit so it's at the center of the scene
//model = glm::scale( model, glm::vec3( 0.02f, 0.02f, 0.02f ) );    // It's a bit too big for our scene, so scale it down
model = glm::rotate(model, glm::radians(movRetrato), glm::vec3(1.0f, 0.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));

retrato.Draw(shader);
```

Para la puerta y la silla:

```
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(1.35797f, 4.31681f + tras, -1.82913f)); // Translate it down a bit so it's at the center of the scene
//model = glm::scale( model, glm::vec3( 0.02f, 0.02f, 0.02f ) );    // It's a bit too big for our scene, so scale it down
//model = glm::rotate(model, (float)glfwGetTime(), glm::vec3(1.0f, 0.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));

silla.Draw(shader);


model = glm::mat4(1);
model = glm::translate(model, glm::vec3(-0.67457f, 1.05998f + tras, -0.914868f)); // Translate it down a bit so it's at the center of the scen
//model = glm::scale( model, glm::vec3( 0.02f, 0.02f, 0.02f ) );    // It's a bit too big for our scene, so scale it down
model = glm::rotate(model, glm::radians(movPuerta), glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));

puertaPrin.Draw(shader);
```

Para el jarrón y el banquito:

```
model = glm::mat4(1);
model = glm::translate(model, glm::vec3(-0.319778f-movAccidente, 3.90331f + tras, -4.38809f)); // Translate it down a bit so it's at the center
//model = glm::scale( model, glm::vec3( 0.02f, 0.02f, 0.02f ) );    // It's a bit too big for our scene, so scale it down
model = glm::rotate(model, glm::radians(rotAccidenteJ), glm::vec3(0.0f, 0.0f, 1.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));

jarron.Draw(shader);

model = glm::mat4(1);
model = glm::translate(model, glm::vec3(-0.319778f, 3.90331f + tras, -4.38809f)); // Translate it down a bit so it's at the center of the scene
//model = glm::scale( model, glm::vec3( 0.02f, 0.02f, 0.02f ) );    // It's a bit too big for our scene, so scale it down
model = glm::rotate(model, glm::radians(rotAccidenteB), glm::vec3(0.0f, 0.0f, 1.0f));
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));

banquito.Draw(shader);
```

Para poder coordinar los objetos que iban a tener animación, se crearon las siguientes variables:

```
//Variables para animaciones
float movCajon = 0; bool cajonActive = false;
float movRetrato = 0; bool retratoActive = false;
float movPuerta = 0; bool puertaActive = false;

float rotAccidenteB, rotAccidenteJ, movAccidente = 0; bool accidenteP1 = false; bool accidenteP2 = false; bool accidenteP3 = false;
float rotLlamada; bool llamadaP1 = false; bool llamadaP2 = false; bool llamadaP3 = false; bool llamadaP4 = false; int numRins = 0; static int RINGS_MA
```

Los objetos que tienen animaciones complejas son:

El teléfono:

```
void animacionLlamada() {
    float rin = 2;
    float lim = 15;
    if (llamadaP1) {

        rotLlamada += rin;
        if (rotLlamada >= lim) {
            llamadaP1 = false;
            llamadaP2 = true;
        }
    }
    if (llamadaP2) {
        rotLlamada -= rin;
        if (rotLlamada <= -lim) {
            llamadaP2 = false;
            llamadaP3 = true;
        }
    }

    if (llamadaP3) {

        rotLlamada += rin*2;
        if (rotLlamada >= lim) {
            llamadaP3 = false;
            llamadaP4 = true;
        }
```

Y el florero:

```
356
357   void animacionAccidente() {
358       float vel=3;
359       if (accidenteP1) {
360           rotAccidenteB += vel;
361           rotAccidenteJ += vel;
362           if (rotAccidenteB >= 45) {
363               accidenteP1 = false;
364               accidenteP2 = true;
365           }
366       }
367       if (accidenteP2) {
368           rotAccidenteB += vel;
369           rotAccidenteJ += vel;
370           movAccidente += vel / 100;
371           if (rotAccidenteB >= 90) {
372               accidenteP2 = false;
373               accidenteP3 = true;
374           }
375       }
376       if (accidenteP3) {
377           rotAccidenteJ += vel;
378           if (rotAccidenteJ >= 95.5) {
379               accidenteP3 = false;
380           }
381       }
382   }
```

Ambos son complejos ya que manejan estados y cambian de dirección.

Por último, para hacer reset se hizo la siguiente función:

```
333
334   void reset() {
335       cajonActive = false;
336       movCajon = 0;
337       retratoActive = false;
338       movRetrato = 0;
339       puertaActive = false;
340       movPuerta = 0;
341
342       accidenteP1 = false;
343       accidenteP2 = false;
344       accidenteP3 = false;
345       rotAccidenteB = 0;
346       rotAccidenteJ = 0;
347       movAccidente = 0;
348
349       rotLlamada = 0;
350       numRins = 0;
351       llamadaP1 = false;
352       llamadaP2 = false;
353       llamadaP3 = false;
354       llamadaP4 = false;
355   }
356
```