

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	13-19
Planeación					
Definición de objetivos y alcances	X				
Definición del cronograma	X				
Capacitación de software de modelado Blender		X			
Determinación de objetos a modelar y cuales descargar. Modelado de 50% de los objetos			X		
Ejecución					
Modelado de 50% de los objetos restante				X	
Diseño de fachada				X	
Animación				X	X
Respaldo de proyecto e github					X

Pgograma:

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 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(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 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(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:

```
43 //Variables para animaciones
44 float movCajon = 0; bool cajonActive = false;
45 float movRetrato = 0; bool retratoActive = false;
46 float movPuerta = 0; bool puertaActive = false;
47
48 float rotAccidenteB, rotAccidenteJ, movAccidente = 0; bool accidenteP1 = false; bool accidenteP2 = false; bool accidenteP3 = false;
49 float rotLlamada; bool llamadaP1 = false; bool llamadaP2 = false; bool llamadaP3 = false; bool llamadaP4 = false; int numRins = 0; static int RINGS_MAX = 10;
50
51
```

Los objetos que tienen animaciones complejas son:

El teléfono:

```
383
384 void animacionLlamada() {
385     float rin = 2;
386     float lim = 15;
387     if (llamadaP1) {
388
389         rotLlamada += rin;
390         if (rotLlamada >= lim) {
391             llamadaP1 = false;
392             llamadaP2 = true;
393         }
394     }
395     if (llamadaP2) {
396         rotLlamada -= rin;
397         if (rotLlamada <= -lim) {
398             llamadaP2 = false;
399             llamadaP3 = true;
400         }
401     }
402
403     if (llamadaP3) {
404
405         rotLlamada += rin*2;
406         if (rotLlamada >= lim) {
407             llamadaP3 = false;
408             llamadaP4 = true;
409         }
410     }
411 }
```

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 complejas ya que manejan estados y cambian de dirección.

Por ultimo, 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
```