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 creará un entorno en 3D de tipo residencial apoyándose de OpenGL y Maya para el modelado de los objetos, además de la implementación de animaciones hechas mediante transformaciones lineales.

Adicionalmente se incluirá una análisis de costos en donde se incluirán los detalles correspondientes a su creación, y presupuesto.

ACTIVIDADES

06/12/2020 - 10/12/2020	11/12/2020 - 16/11/2020	17/12/2020 - 30/12/2020	01/01/2021 - 22/01/2021	22/01/2021 - 26/01/2021
Planteamiento del proyecto	Presentación de la propuesta	Comienzo de actividades con los objetos propuestos	Afinación de detalles con los modelos.	Optimización de archivos, y código.
Identificación de objetivos	Búsqueda de recursos que ayuden al aprendizaje del software Maya	Incorporación de los modelos en OpenGL	Optimización de los modelos para ahorrar tiempo de carga, y tamaño.	Carga de modelos finales al repositorio.
Programación del calendario de actividades	Búsqueda de referencias útiles para la realización de los objetos	Realización de las primeras pruebas	Planteamiento sobre las animaciones.	Realización del manual de usuario.
Bosquejo de la casa, y búsqueda de inspiración para el diseño	Carga de avances al repositorio.	Carga de avances al repositorio.	Programación de las animaciones.	Realización del manual técnico.
			Pruebas de las animaciones, y correcciones de cámara,	Comprobación del funcionamiento del proyecto, con la última carga.

DOCUMENTACIÓN DEL CÓDIGO

- ### CARGA DE MODELOS OBJ A OPENGL

```
Model ourModel  ((char *)"Models/HouseModel/HouseModel.obj");
Model ourModel2 ((char*) "Models/Couch/Couch.obj");
Model ourModel3((char*) "Models/TvStand/tvStand.obj");
Model ourModel4((char*)"Models/Bed/Bed.obj");
Model ourModel5((char*)"Models/Basket/Basket.obj");
Model deskRoom((char*) "Models/Desk/desk.obj");
Model drawerDesk((char*)"Models/Desk/cajon.obj"); // Cajon
Model chairRoom((char*)"Models/Chair/chair.obj");
Model carModel((char*)"Models/Car2/car.obj");
Model bearModel((char*)"Models/Bear/bear.obj");

Model drawerModel((char*)"Models/Drawer/drawer.obj");
Model lampModel((char*)"Models/Lamp/lamp.obj");
Model closetModel((char*)"Models/Closet/closet.obj");
Model tableModel((char*)"Models/Table/table.obj");
Model couch2Model((char*)"Models/Couch/Couch2ndfloor.obj");
Model couch3Model((char*)"Models/Couch/Couch2ndfloor2.obj");
Model plantModel((char*)"Models/Plant/plant.obj");
Model doorModel((char*)"Models/Door/door.obj");
Model kitchenModel((char*)"Models/Kitchen/kitchen.obj");
Model tableKit((char*)"Models/TableKit/tablekit.obj");
Model teapot((char*)"Models/Teapot/teapot.obj");
```

- **DIBUJADO DE LOS MODELOS, Y ACOMODO POR MEDIO DE TRASLACIÓN**

```
// HOUSE MODEL LOADING AND DRAWING
glUniformMatrix4fv( glGetUniformLocation( shader.Program, "model" ), 1, GL_FALSE, glm::value_ptr( model ) );
ourModel.Draw(shader);

backup = model;

if (door) {
    // DOOR MODEL LOADING AND DRAWING
    glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
    doorModel.Draw(shader);
}

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
ourModel2.Draw(shader);



// BASKET LOADING
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
ourModel3.Draw(shader);


// BED LOADING AND DRAWING
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
ourModel4.Draw(shader);

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));

// DESK LOADING
ourModel5.Draw(shader);
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
deskRoom.Draw(shader);


//DRAWER LOADING
model = glm::translate(model, glm::vec3(0.0f+ drawer_x, 0.0f, 0.0f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
drawerDesk.Draw(shader);
model = backup;

// CHAIR LOADING AND DRAWING
model = glm::translate(model, glm::vec3(0.0f+ chair_y, 0.0f , 0.0f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
chairRoom.Draw(shader);
```

```
// CHAIR LOADING AND DRAWING
model = glm::translate(model, glm::vec3(0.0f+ chair_y, 0.0f , 0.0f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
chairRoom.Draw(shader);
model = backup;

model = glm::mat4(1);
model = glm::translate(model, iniPosCar+ glm::vec3(car_x, car_y, car_z));
model = glm::rotate(model, rotCar, glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
carModel.Draw(shader);

model = backup;

// BEAR LOADING
model = glm::translate(model, glm::vec3(0.0f,0.0f + bear_y, 0.0f));
model = glm::rotate(model, rotBear, glm::vec3(0.0f, 0.0f, 1.0f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
bearModel.Draw(shader);
model = backup;

// DRAWER LOADING
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
drawerModel.Draw(shader);

// LAMP LOADING
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
lampModel.Draw(shader);

// CLOSET LOADING
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
closetModel.Draw(shader);

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
tableModel.Draw(shader);

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
couch2Model.Draw(shader);

model = backup;

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
couch3Model.Draw(shader);
```

```
lampModel.Draw(shader);

// CLOSET LOADING
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
closetModel.Draw(shader);

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
tableModel.Draw(shader);

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
couch2Model.Draw(shader);

model = backup;

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
couch3Model.Draw(shader);

model = glm::translate(model, glm::vec3(3.478f, -8.179f, 4.460f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
plantModel.Draw(shader);

model = backup;

model = glm::translate(model, glm::vec3(5.178, -8.179f, 4.460f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
plantModel.Draw(shader);

model = backup;

model = glm::translate(model, glm::vec3(-3.186f, -4.893f, 4.52f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
plantModel.Draw(shader);

model = backup;
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
kitchenModel.Draw(shader);
model = backup;

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
tableKit.Draw(shader);

model = backup;

glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE, glm::value_ptr(model));
teapot.Draw(shader);
```

- **ANIMACIONES COMPLEJAS**

```
if (keys[GLFW_KEY_L]) {

    (carCircuit == false) ? carCircuit = true,cir1=true  : carCircuit = false; // INICIAR LA ANIMACION DEL COCHE

}
```

```
if (keys[GLFW_KEY_J]) { // ANIMACION CAIDA DEL OSO

    (bearCir == false)? bearCir = true: bearCir=false, rotBear=0.0f,bear_y=0, cirB1=true; // SE ACTIVA LA ANIMACION DEL OSO

}
```

- CIRCUITO DEL AUTOMÓVIL

```
void moveCar() {

    if (carCircuit) {

        if (cir1) {
            (car_z < 10.3) ? car_z = car_z + 0.1 : cir1 = false;
            (cir1 == false) ? cir2 = true, rotCar=45.5 : cir1 = true;
        }

        if (cir2) {
            (car_x < 17.8) ? car_x = car_x + 0.1 : cir2 = false;
            (cir2 == false) ? cir3 = true, rotCar = 47.1 : cir2 = true;
        }

        if (cir3) {
            (car_z > -10.5) ? car_z = car_z - 0.1 : cir3 = false;
            (cir3 == false) ? cir4 = true, rotCar = 48.6 : cir3 = true;
        }

        if (cir4) {
            (car_x > -4.5) ? car_x = car_x - 0.1 : cir4 = false;
            (cir4 == false) ? cir5 = true, rotCar = 0 : cir4 = true;
        }

        if (cir5) {
            (car_z < 7.5) ? car_z = car_z + 0.1 : cir5 = false;
            (cir5 == false) ? cir6 = true, rotCar = 45.5 : cir5 = true;
        }
        if (cir6) {
            (car_x > 0.1) ? car_x = car_x + 0.1 : cir6 = false;
            carCircuit = false;
        }

    }
}
```

- CAÍDA DEL OSO DE PELUCHE

```
void moveBear() {

    if (bearCir) {

        if (cirB1) {
            (rotBear < 0.4) ? rotBear = rotBear + 0.05 : cirB1 = false, cirB2=true;
        }

        if (cirB2) {
            (bear_y > -2.1) ? bear_y = bear_y - 0.1 : cirB2 = false;
        }
    }
}
```

ANÁLISIS DE COSTOS

- El proyecto “Elaboración de modelado 3D para residencia” se ejecutará en un periodo de 1 mes, equivalente 20 días naturales o jornadas.

Código	Concepto	Unidad	P. Unitario	Op.	Cantidad	Importe	%
Análisis:	**1**		**SERVICIO**		**1.0000**	**$28,534.24**	
ELABORACIÓN DE MODELADO 3D PARA RESIDENCIA.							
BÁSICOS							
BM01	**Ingeniero en Computación especialista en modelado 3D, incluye todo lo necesario para la realización de sus funciones.**	**JOR**					
MAT01	Juego de papelería mensual, incluye: papelería, bolígrafos, lapiceros, gomas, libretas y todos los utensilios necesarios para la realización de labores de oficina	JGO	$200.00	/	24.000000	$8.33	
MO01	Ingeniero en Computación especialista en modelado 3D.	JOR	$500.00	*	1.000000	$500.00	
MAQ20	Impresora multifuncional, incluye copias, escaneo y consumibles	HR	$38.73	*	0.200000	$7.75	
MAQ17	Computadora de escritorio de altas especificaciones, incluye software especializado.	HR	$18.01	*	8.000000	$144.08	
MAQ19	Plotter, incluye insumos para la impresión de 90x 60	HR	$44.84	*	0.200000	$8.97	
	Importe:					$669.12	
	Volumen:				20.000000	$13,382.60	52.53%
BM02	**Diseñador gráfico especialista en modelado 3D, incluye todo lo necesario para la realización de sus funciones.**	**JOR**					
MAT02	Juego de papelería mensual, incluye: papelería, bolígrafos, lapiceros, gomas, libretas y todos los utensilios necesarios para la realización de labores de oficina	JGO	$200.00	/	24.000000	$8.33	
MO2017	Diseñador gráfico especialista en modelado 3D.	JOR	$500.00	*	1.000000	$500.00	
MAQ13	Computadora Portátil, incluye Software especializado.	HR	$11.08	*	8.000000	$88.64	
MAQ20	Impresora multifuncional, incluye copias, escaneo y consumibles	HR	$38.73	*	0.200000	$7.75	
	Importe:					$604.72	
	Volumen:				20.000000	$12,094.40	47.47%
							100.00
	(CD) COSTO DIRECTO					**$25,477.00**	**%**
	(CI) INDIRECTOS				12%	$3,057.24	
						-	**28,534.24**

- Los costos indirectos contemplan Mantenimiento de oficinas, luz, agua y otros servicios.