UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

EJERCICIOS DE CLASE Nº 09

NOMBRE COMPLETO: Barragán Pilar Diana

Nº de Cuenta: 318147981

GRUPO DE LABORATORIO: 03

GRUPO DE TEORÍA: 04

SEMESTRE 2025-1

FECHA DE ENTREGA LÍMITE: 15 de octubre de 2024

CALIFICACIÓN: _____

Ejercicio de clase práctica 9: Animación Básica.

1. **Separar del arco la parte del letrero.**

Dado que mi modelo de arco no contaba con un letrero busque el letrero por separado como otro .obj y acomode ambos en 3ds Max de tal manera que tanto el arco como el letrero estuvieran bien posicionados y escalados, para después simplemente colocarlos en el origen para exportarlos, para posteriormente importarlos al código en OpenGL.

```
Model Kitt_M;
Model Llanta_M;
Model Blackhawk_M;
Model Arco;
Model Letrero;
```

```
Blackhawk_M = Model();
Blackhawk_M.LoadModel("Models/uh60.obj");
Arco = Model();
Arco.LoadModel("Models/arco.obj");
Letrero = Model();
Letrero.LoadModel("Models/letrero.obj");
```

Figura 1. Importación al código del arco y el letrero por separado.

2. **Hacer que el letrero baje y suba recorriendo toda la altura del arco, esto de acuerdo a un contador de tiempo que durará 2 segundos (2 segundos en bajar, 2 segundos a nivel del piso, 2 segundos subiendo, 2 segundos en la altura máxima y así sucesivamente.**

Para lograr que el letrero suba y baje utilice algo parecido a lo que realizamos en clase, puesto que con la bandera de que si avanza o no dividimos en dos casos que son: cuando estoy en el punto máximo y el letrero va bajando y cuando estoy en el punto mínimo y el letrero debe subir. Del primer y segundo caso también se seccionan para lo que se debe realizar en el siguiente ejercicio de rotar el letrero.

La posición del letrero se inicializa en el punto máximo del arco que en mi caso fue de 3.0f y si nos encontramos en el primer caso de querer que el letrero baje la línea de código movLetrero -= movOffset * deltaTime va a tener un símbolo negativo pues se irá decrementando para que el letrero baje, cuando llegue al punto más bajo que es al tocar el piso, en este caso en 0.1, entonces debe subir por lo que la bandera de avanzar va a cambiar. En el segundo caso, realizamos prácticamente lo mismo solo que en vez de tener un contador negativo ponemos uno positivo para que vaya aumentando la traslación en y, lo que hace que el letrero suba.

Para que el movimiento durará dos segundos simplemente cambie el valor de movOffset ya que con este calculamos que tan rápido va a ser el movimiento con el que cae al piso para multiplicarse después por el tiempo transcurrido, al asignarle el valor de 0.029f la velocidad con la que cae y sube es de 2 segundos.

```
movLetrero = 3.0f;
movOffset = 0.029f;
rotllanta = 0.0f;
rotLetrero = 0.0f;
rotllantaOffset = 1.0f;
avanza = true;
////Loop mientras no se cierra la ventana
while (!mainWindow.getShouldClose())
{
    GLfloat now = glfwGetTime(); //Tiempo actual o tiempo de inicio
    deltaTime = now - lastTime; //Tiempo que transcurre
    deltaTime += (now - lastTime) / limitFPS;
    lastTime = now;
```

Figura 2. Modificación de movOffset para que el movimiento sea en 2 segundos.

```
if (avanza)
{

    if (movLetrero > 0.1f) {
        //printf("movLetrero%f \n ", movLetrero);
        if (movLetrero>2.9f && movLetrero<3.1f) {
            if (rotLetrero < 360.0f) {
                rotLetrero += rotllantaOffset * deltaTime;
            }
            else {
                movLetrero -= movOffset * deltaTime;

            }
        }
        else {
            movLetrero -= movOffset * deltaTime;
            rotLetrero = 0.0f;
        }

    }
    else {
        avanza = !avanza;
    }

}else {
    if (movLetrero <= 3.0f) {

        if (movLetrero > 0.09f && movLetrero < 0.12f) {
            //printf("movLetrero%f \n ", movLetrero);
            if (rotLetrero < 360.0f) {
                rotLetrero += rotllantaOffset * deltaTime;
            }
            else {
                movLetrero += movOffset * deltaTime;

            }
        }
        else {
            movLetrero += movOffset * deltaTime;
            rotLetrero = 0.0f;
        }
    }
    else {
        avanza = !avanza;
    }
}
```

Figura 3. Código para que suba y baje el letrero, así como para que rote 360 grados.

```
//Arco
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, -0.2f, 0.0f));
modelaux = model;
model = glm::scale(model, glm::vec3(0.6f, 0.6f, 0.6f));
model = glm::rotate(model, -90 * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Arco.RenderModel();

//Letrero
//Llanta delantera izquierda
model = modelaux;
model = glm::translate(model, glm::vec3(0.0f, movLetrero, 0.3f));
//model = glm::translate(model, glm::vec3(0.0f, 0.1f, 0.3f));
model = glm::rotate(model, -90 * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
model = glm::rotate(model, rotLetrero * toRadians, glm::vec3(0.0f, 0.0f, 1.0f));
model = glm::scale(model, glm::vec3(0.6f, 0.6f, 0.6f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Letrero.RenderModel();
```

Figura 4. Translate y rotate en el letrero para que este suba y baje, además de rotar.

3. **El letrero rotara 360° sobre su propio centro estando en la parte superior y en la parte inferior alrededor del eje X (suponiendo que vemos al arco en dirección de Z negativo)**

Para que el letrero rotará en su propio eje al encontrarse en el punto máximo y al en el punto mínimo, como se puede ver en la Figura 3 al ya haber dividido en dos casos principales, simplemente estos los volví a dividir entre que se llegará al punto máximo y de esto cumplirse entonces se va a girar el letrero hasta que de una vuelta de 360 grados para después seguir bajando y posteriormente vendría el otro caso en el que llegamos al punto mínimo entonces al estar en este punto se realiza lo mismo que en el otro caso, girando el letrero hasta que de la vuelta de 360 grados y así sucesivamente se vaya repitiendo este patrón.

Para aplicarlos movimientos al modelo, simplemente en el traslate se utilizo la variable movLetrero en el eje y para que la traslación diera el efecto de subir y bajar. Por otro lado, para que rote en la función rotate se multiplica la variable rotLetrero que tiene los grados para que vaya girando y esto se aplica en el eje de z.

En este ejercicio no tuve inconveniente alguno a la hora de compilar y ejecutar el código.

CONCLUSIONES:

Al realizar este ejercicio práctico me fue posible comprender como es que se puede aplicar una animación simple a distintos objetos en Opengl, modificando y aplicando todo lo visto en las clases anteriores como lo es la rotación, traslación, importación de modelos jerarquía entre muchas otras para dar movimiento con el uso de condicionales y contadores.