UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

REPORTE DE PRÁCTICA Nº9

NOMBRE COMPLETO: Araiza Valdés Diego Antonio

Nº de Cuenta: 423032833

GRUPO DE LABORATORIO: 02

GRUPO DE TEORÍA: 06

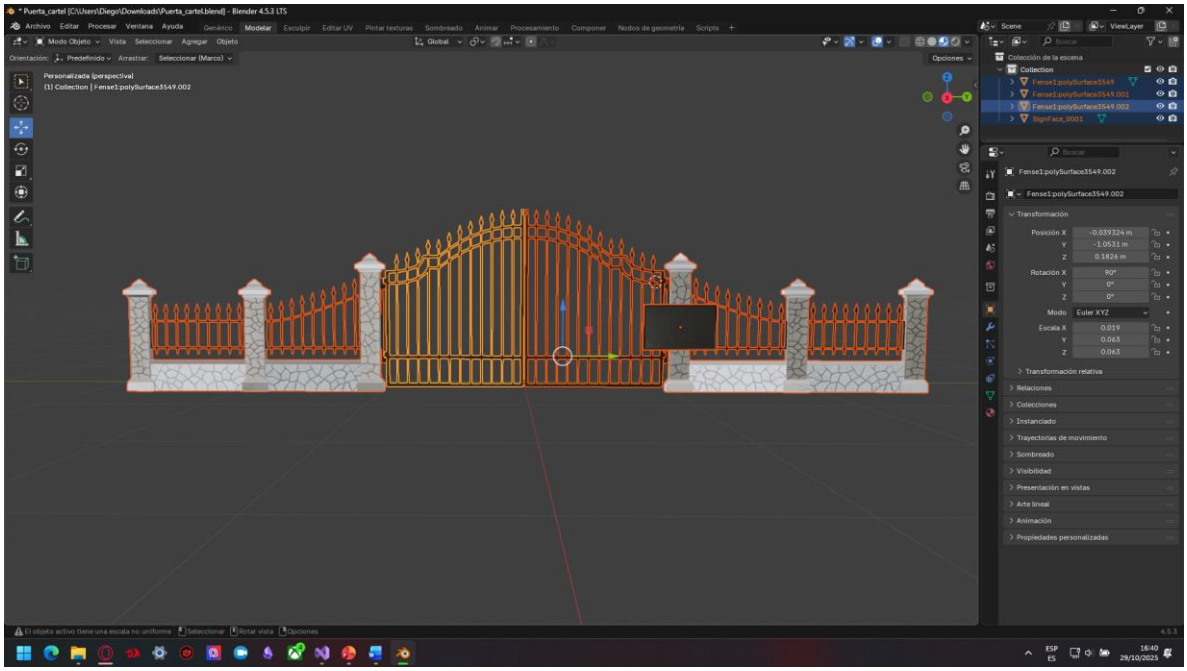
SEMESTRE: 2026-1

FECHA DE ENTREGA LÍMITE: 02/11/2025

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

1. Ejecución de los ejercicios que se dejaron, comentar cada uno y capturas de pantalla de bloques de código generados y de ejecución del programa.
   1.1. Ejercicio 1: Separar del arco la parte del letrero y las puertas o rejas.



Primero se texturizó el modelo del letrero para que pareciera una pantalla LED de texto corrido apagada y se separó cada una las puertas y el letrero de la estructura general y se exportó individualmente.

1.2. Ejercicio 2: Hacer que en el arco que crearon se muestre la palabra: PROYECTO CGEIHC desplazándose las letras de derecha a izquierda como si fuera letrero LCD/LED de forma cíclica

Primero se creó una imagen para el letrero con la tipografía utilizada en el previo de manera que dijera PROYECTO CGEIH, se aumentó la calidad, se le quitó el fondo y se optimizó para hacer uso de ella

```
unsigned int letreroIndices[] = {
  0, 1, 2,
  0, 2, 3,
};

GLfloat letreroVertices[] = {
   -0.5f, 0.0f, -0.5f,     0.0f,  0.1f,        0.0f, 1.0f, 0.0f,
    0.5f, 0.0f, -0.5f,     0.2f,  0.1f,        0.0f, 1.0f, 0.0f,
    0.5f, 0.0f,  0.5f,     0.2f,  0.9f,        0.0f, 1.0f, 0.0f,
   -0.5f, 0.0f,  0.5f,     0.0f,  0.9f,        0.0f, 1.0f, 0.0f,

};
```

```
CartelPTexture = Texture("Textures/CartelPuerta1.png");
CartelPTexture.LoadTextureA();
```

Por consiguiente, se creó un objeto nuevo para el letreo (meshList[7]) y se importó la textura creada anteriormente.

```
Model EstructuraP_M;
Model PuertaIzq_M;
Model PuertaDer_M;
Model Cartel_M;
```

```
EstructuraP_M = Model();
EstructuraP_M.LoadModel("Models/EstructuraPuerta.obj");
PuertaIzq_M = Model();
PuertaIzq_M.LoadModel("Models/PuertaIzq.obj");
PuertaDer_M = Model();
PuertaDer_M.LoadModel("Models/PuertaDer.obj");
Cartel_M = Model();
Cartel_M.LoadModel("Models/Letrero.obj");
```

Después se importarón los modelos separados en el punto anterior.

```
//Estructura de la puerta
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(20.0f, -1.63f, 15.0f));
model = glm::scale(model, glm::vec3(5.0f, 5.0f, 5.0f));
modelaux = model;
//model = glm::rotate(model, -90 * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
EstructuraP_M.RenderModel();

//Puerta Izquierda
model = modelaux;
model = glm::translate(model, glm::vec3(0.039396f, 0.182048f, -1.05364f - Tpuerta));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
PuertaIzq_M.RenderModel();

//Puerta Derecha
model = modelaux;
model = glm::translate(model, glm::vec3(0.039396f, 0.182381f, 1.06414f));
model = glm::rotate(model, Rpuerta * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
PuertaDer_M.RenderModel();

//Letrero
model = modelaux;
model = glm::translate(model, glm::vec3(-0.032753f, 0.417611f, 1.16599f));
modelaux = model;
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Cartel_M.RenderModel();
```

Posteriormente se declaran y renderizan la estructura de la puerta, ambas puertas y el letrero. Asignando una traslación variable a la puerta izquierda y una rotación variable a la puerta derecha.

```
//Letras en movimiento
toffsetnumerocambiau += 0.004 * deltaTime;
if (toffsetnumerocambiau > 1.0)
    toffsetnumerocambiau = 0.0;
toffsetnumerov = 0.0;
toffset = glm::vec2(toffsetnumerocambiau, toffsetnumerov);
model = modelaux;
model = glm::translate(model, glm::vec3(-0.01f, 0.0f, -0.005f));
model = glm::rotate(model, -90 * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
model = glm::rotate(model, -90 * toRadians, glm::vec3(0.0f, 0.0f, 1.0f));
model = glm::scale(model, glm::vec3(0.5f, 1.0f, 0.30f));
glUniform2fv(uniformTextureOffset, 1, glm::value_ptr(toffset));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
CartelPTexture.UseTexture();
Material_brillante.UseMaterial(uniformSpecularIntensity, uniformShininess);
meshList[7]->RenderMesh();
```

Finalemnte, siguiendo la estructura del número cambiante se le asignó una animación al letrero, de manera que redujimos el multiplicador de detltaTime para que se actualizara varias veces (fuera una animación fluida) pero se recorriera poco hacia la derecha. Además, se trasladó, escaló y roto para ajustarlo al cartel. Finalmente se hizo uso de la textura y del objeto creado anteriormente para colocar la textura en él.

1.3. Ejercicio 3: Hacer que al presionar una tecla una de las puertas se abra por medio de una rotación y la otra puerta se abra por medio de un deslizamiento

```
    GLboolean getAbrirPuerta() { return abierta;  }

    ~Window();
private:
    GLFWwindow *mainWindow;
    GLint width, height;
    GLboolean abierta;
```

```
abierta = false;
```

```
if (key == GLFW_KEY_O && action == GLFW_PRESS)
{
    theWindow->abierta = !theWindow->abierta;
}
```

Para el ultimo ejercicio se creó un método que al presionar la tecla O intercambie el valor de una bandera.

```
float Tpuerta = 0.0f;
float Rpuerta = 0.0f;
```

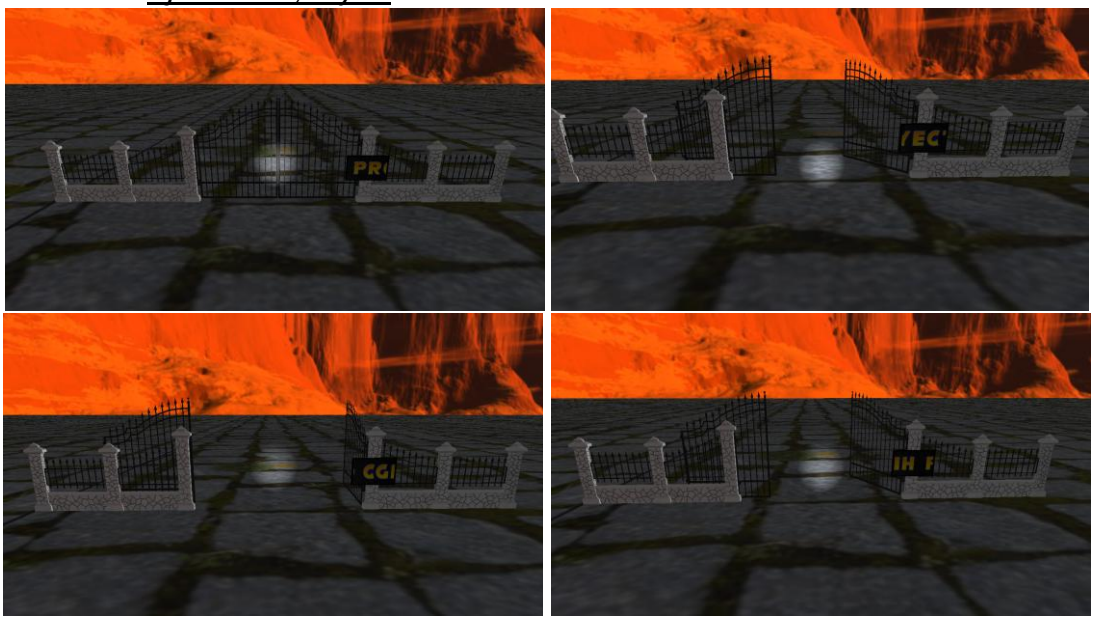
Posteriormente se crearon las variables que nos permitirán asignar rotación y traslación a las puertas.

```
if (mainWindow.getAbrirPuerta()) {
    if (Tpuerta < 1.05f) {
        Tpuerta += 0.034243f * deltaTime;
    }
    if (Rpuerta > -92.0f) {
        Rpuerta -= 3.0f * deltaTime;
    }
}
else {
    if (Tpuerta > 0.0f) {
        Tpuerta -= 0.0333f * deltaTime;
    }
    if (Rpuerta < 0.0f) {
        Rpuerta += 3.0f * deltaTime;
    }
}
```

Finalmente, declaramos un condicional que recibe la bandera declarada anteriormente para que, si las puertas se abren, se vayan recorriendo/rotando hasta llegar al punto deseado, de manera que, tarden lo mismo en completar el recorrido, para esto hicimos uso de deltaTime.

Evidencias:

1.3.1. *Ejercicio 1, 2 y 3:*



1. Liste los problemas que tuvo a la hora de hacer estos ejercicios y si los resolvió explicar cómo fue, en caso de error adjuntar captura de pantalla
   R: No hubo problemas.

2. Conclusión:
   1. Los ejercicios del reporte: Complejidad, Explicación.
      Respecto a los ejercicios realizados, fueron bastante adecuados a lo visto en clase y con los ejemplos vistos, no hubo mayor problema.

   2. Comentarios generales: Faltó explicar a detalle, ir más lento en alguna explicación, otros comentarios y sugerencias para mejorar desarrollo de la práctica.
      La explicación fue muy buena, se entendió perfectamente como animar objetos y que es y que no es animación. Sin embargo, me hubiera gustado que explique un poco más la lógica acerca de la siguiente línea:
      glUniform2fv(uniformTextureOffset, 1, glm::value_ptr(toffset));
      ya que, si bien la entendí, solo fue muy por encima.

   3. Conclusión
      Personalmente creo que se cumplieron los objetivos de la práctica. Con el ejercicio y la práctica, aprendí a crear diversas animaciones haciendo uso de banderas, condicionales y funciones, como movimiento (traslación y rotación), cambiar texturas y hacer letreros movibles (con y sin pausas visibles). Además, aprendí a utilizar la variable deltaTime.