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 Nº 07

NOMBRE COMPLETO: Sanchez Calvillo Saida Mayela

Nº de Cuenta: 318164481

GRUPO DE LABORATORIO: 02

GRUPO DE TEORÍA: 04

SEMESTRE 2025-2

FECHA DE ENTREGA LÍMITE: 05 de abril 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.- Agregar movimiento con teclado al helicóptero hacia adelante y atrás.

Para este ejercicio se agregó la misma línea de código que el carro ya tiene para que se pueda mover de adelante hacia atrás.

```
//Helicoptero
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, 10.0f, 6.0));
spotLights[3].SetPos(glm::vec3(0.0f + mainWindow.getmuevex(), 5.0f, 6.0));
//model = glm::scale(model, glm::vec3(0.3f, 0.3f, 0.3f));
model = glm::translate(model, glm::vec3(0.0f + mainWindow.getmuevex(), 0.0f, 0.0f));
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));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Blackhawk_M.RenderModel();
```

2.- Crear luz Spotlight de helicóptero de color amarilla que apunte hacia el piso y se mueva con el helicóptero

Para esto se creo nuestra luz tipo Spotlight color amarilla y primero se observó en que posición se encontraba el helicóptero sin escala para así poder posicionar nuestra luz correctamente.

Se agrego un ángulo mayor para que el área iluminada desde el helicóptero fuera mayor.

```
//luz fija (Helicoptero)
spotLights[3] = SpotLight(1.0f, 1.0f, 0.0f,    //Luz amarilla
    1.0f, 2.0f,
    0.0f, 5.0f, 6.0f,
    0.0f, -1.0f, 0.0f,
    1.0f, 0.0f, 0.0f,
    25.0f);
spotLightCount++;
```

Después, como en el ejercicio, se usó SetPos para agregar nuestra luz al movimiento de nuestro helicóptero recordando que esta luz se encuentra en la posición 3 de nuestro arreglo

```
//Helicoptero
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f, 10.0f, 6.0));
spotLights[3].SetPos(glm::vec3(0.0f + mainWindow.getmuevex(), 5.0f, 6.0));
//model = glm::scale(model, glm::vec3(0.3f, 0.3f, 0.3f));
model = glm::translate(model, glm::vec3(0.0f + mainWindow.getmuevex(), 0.0f, 0.0f));
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));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Blackhawk_M.RenderModel();
```

3.- Añadir en el escenario 1 modelo de lámpara texturizada y crearle luz puntual blanca

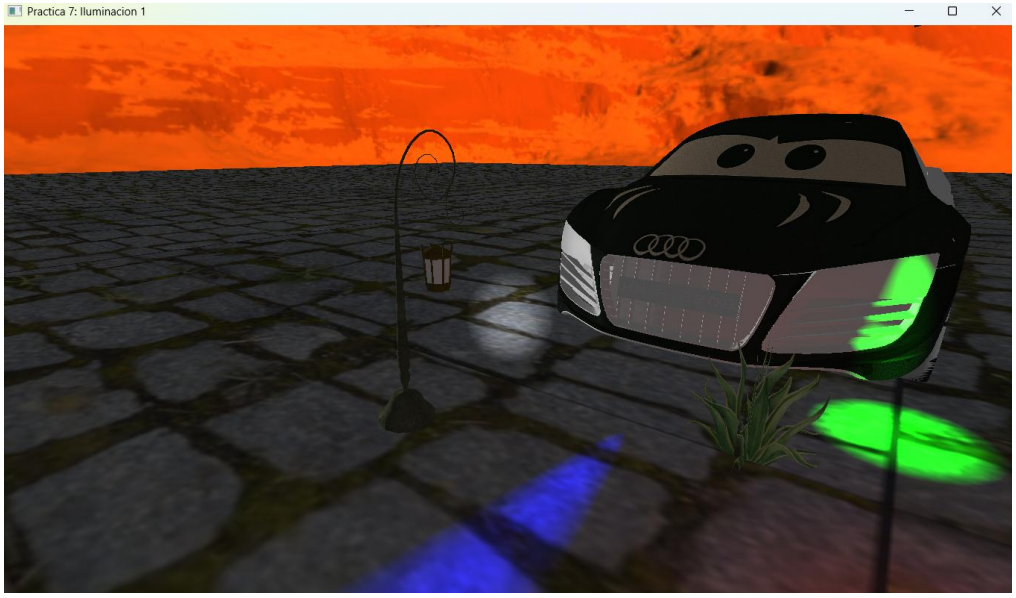
Primero buscamos un modelo de una lampara texturizada y en nuestro caso usamos una farola bonita con textura. La importamos, la acomodamos y la renderizamos en nuestro código como cuando renderizamos el dado o el carro, creando el modelo y cargándolo a nuestro código.

```
Model Lampara;
```

```
Lampara = Model();
Lampara.LoadModel("Models/lampara.obj");
```

```
//Farola
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-5.0f, -1.0f, -15.0));
model = glm::rotate(model, -90 * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Lampara.RenderModel();
```

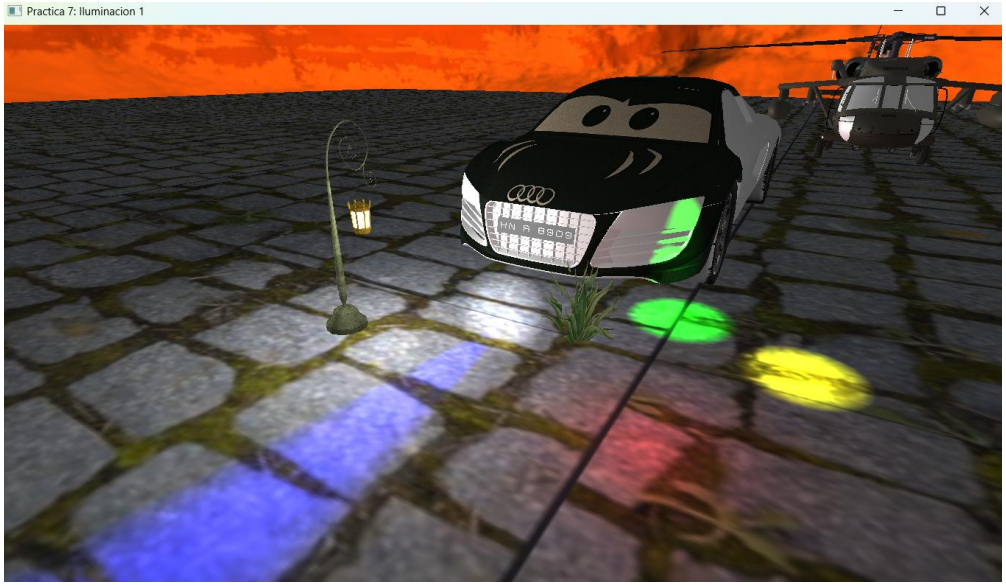
Al renderizarla se ve así



Ahora tenemos que agregar nuestra luz puntual blanca y eso es creando nuestra luz PointLight en la posición que deseemos, en particular colocándola donde se supone que la farola alumbra en la vida real.

```
//luz puntual (farola)
pointLights[1] = PointLight(1.0f, 1.0f, 1.0f,    //Luz blanca
    0.5f, 1.0f,
    -5.2f, 5.0f, -12.3f,
    0.5f, 0.09f, 0.01f);
pointLightCount++;
```

En este código se movió la ubicación de la luz y la ecuación cuadrática con la que logramos que el área que ilumine sea mayor considerando que nuestra farola es alta.



2.- 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

La familiarización con estos nuevos bloques de código fue buena ya que el profesor la explico de una forma fácil de entender. No hubo mucho problema, solo con el hacer que una luz se moviera con nuestro objeto, pero eso se resolvió en el ejercicio.

3.- Conclusión:

a. Los ejercicios del reporte: Complejidad, Explicación.

Se cubrió tanto la teoría (tipos de luces) como la práctica (implementación en código y shaders). Sin embargo, hubo confusión inicial al diferenciar entre PointLight y SpotLight en cuestión de cuando usar uno y otra, lo cual se resolvió analizando el shader y ajustando parámetros.

b. Conclusión

Esta práctica permitió comprender cómo la iluminación define el realismo en gráficos 3D, especialmente mediante el control de luces puntuales y su atenuación. Aunque hubo desafíos técnicos (como sincronizar la posición de la luz con el modelo y ajustar ecuaciones), se logró un resultado funcional donde la luz de la farola ilumina un área específica del piso y la atenuación cuadrática ahora es configurable para simular diferentes fuentes lumínicas.