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 08

NOMBRE COMPLETO: Aguilar Pérez José Ramón

Nº de Cuenta: 317515048

GRUPO DE LABORATORIO: 02

GRUPO DE TEORÍA: 04

SEMESTRE 2025-2

FECHA DE ENTREGA LÍMITE: 12/abril/2025

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

1.- Ejercicios realizados.

- **Agregar un spolight (que no sea luz de color blanco ni azul) que parta del cofre de su coche y al abrir y cerrar el cofre ilumine en esa dirección.**
- **Agregar luz de tipo spotlight para el coche de tal forma que al avanzar (mover con teclado hacia dirección de X negativa) ilumine con un spotlight hacia adelante y al retroceder ((mover con teclado hacia dirección de X positiva) ilumine con un spotlight hacia atrás.**
- **Agregar otra luz de tipo puntual ligada a un modelo elegido por ustedes (no lámpara) y que puedan prender y apagar de forma independiente con teclado tanto la luz de la lámpara como la luz de este modelo.**

Para el primer y segundo ejercicio, se crearon las spotlights correspondientes para el cofre y la luz trasera y delantera. Dentro del archivo *Window.cpp* y *Window.h* se agregó una bandera llamada *LuzAuto*, la cual servirá para identificar a que dirección se está moviendo el coche (Adelante (Tecla Y) = 1, Atrás (Tecla U) = 2). La luz del cofre estará siempre activa. Después, se posicionan las spotlights enfrente y atrás del modelo del coche.

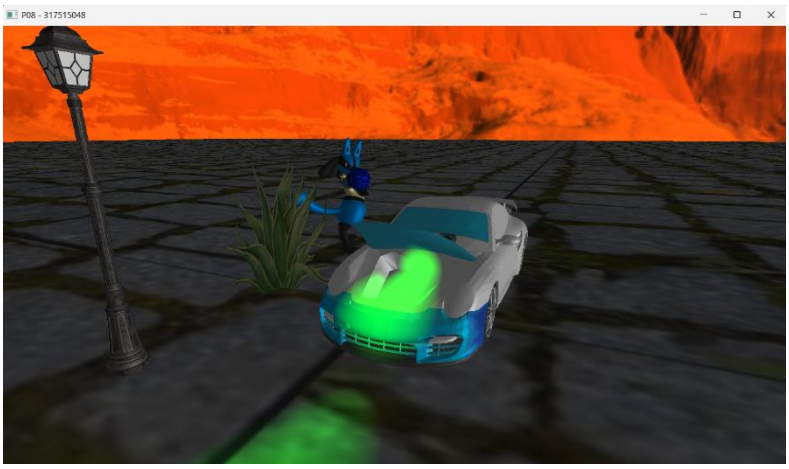
```
//Luz del cofre
spotLights[0] = SpotLight(0.0f, 1.0f, 0.0f,
    1.0f, 3.0f,         //primer termino -> intensidad de iluminacion
    0.0f, 0.0f, 0.0f,   //Posicion foco invisible
    -1.0f, 0.0f, 0.0f,  //Direccion donde apunta
    1.0f, 0.0f, 0.0f,   //Ecuacion de segundo grado
    15.0f);             //Angulo del cono
spotLightCount++;
//Luz frontal
spotLights[1] = SpotLight(0.0f, 0.0f, 1.0f,
    1.0f, 3.0f,         //primer termino -> intensidad de iluminacion
    0.0f, 0.0f, 0.0f,   //Posicion foco invisible
    -1.0f, 0.0f, 0.0f,  //Direccion donde apunta
    1.0f, 0.0f, 0.0f,   //Ecuacion de segundo grado
    15.0f);             //Angulo del cono
spotLightCount++;
//Luz trasera
spotLights[2] = SpotLight(1.0f, 0.0f, 0.0f,
    1.0,  3.0f,         //primer termino -> intensidad de iluminacion
    0.0f, 0.0f, 0.0f,   //Posicion foco invisible
    1.0f, 0.0f, 0.0f,   //Direccion donde apunta
    1.0f, 0.0f, 0.0f,   //Ecuacion de segundo grado
    15.0f);             //Angulo del cono
spotLightCount++;
```

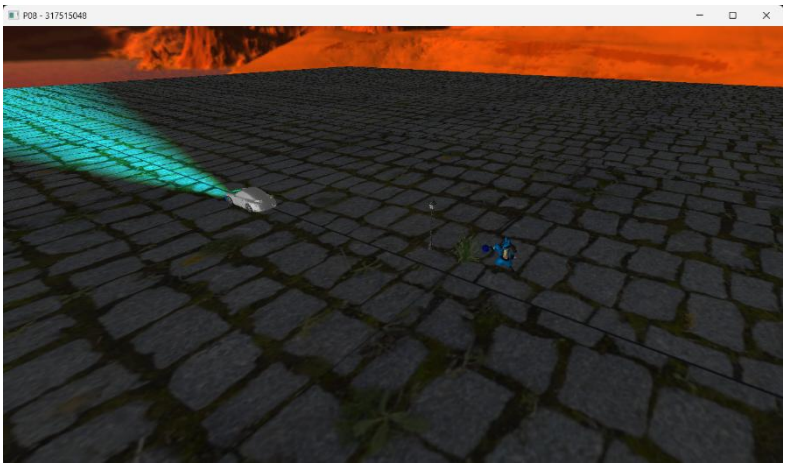
```
//Movimiento del coche
if (key == GLFW_KEY_Y)
{
    theWindow-> muevex -= 1.0;
    theWindow-> LuzAuto = 1.0f;
}
if (key == GLFW_KEY_U)
{
    theWindow-> muevex += 1.0;
    theWindow-> LuzAuto = 2.0f;
}
```

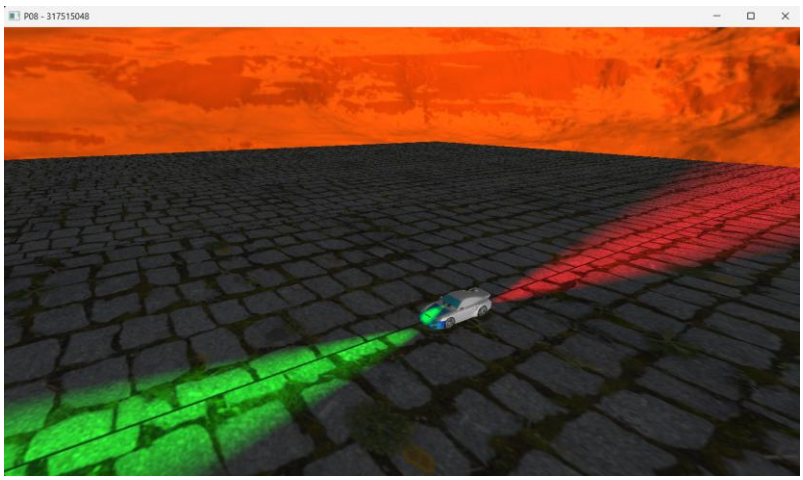
```
//Porsche
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(0.0f + mainWindow.getmuevex(), 0.1f, 1.0f)); // Ade
spotLights[1].SetPos(glm::vec3(-3.0f + mainWindow.getmuevex(), 0.0f, 1.0f)); //Luz frontal
spotLights[2].SetPos(glm::vec3(3.0f + mainWindow.getmuevex(), 0.0f, 1.0f)); //Luz trasera
model = glm::scale(model, glm::vec3(0.25f, 0.25f, 0.25f));
model = glm::rotate(model, 90 * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
modelaux = model;
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Porsche_M.RenderModel();
```

Dentro del main, se agrega las condicionales las cuales utilizan la bandera *LuzAuto* para encender el spotlight correspondiente. En el caso en el que se está moviendo hacia atrás se debe apagar la luz frontal, la cual está en medio del arreglo, por lo que se hace un arreglo temporal el cual solo tiene a la luz del cofre y la luz trasera.

```
//************************************ Luz del cofre siempre encendida ********************
if (mainWindow.getLuzauto() == 1.0) {
    shaderList[0].SetSpotLights(spotLights, spotLightCount - 1); //Se enciende la luz del cofre
}
else if (mainWindow.getLuzauto() == 2.0){
    SpotLight tempLights[2];         // Arreglo temporal
    tempLights[0] = spotLights[0];  // Luz cofre
    tempLights[1] = spotLights[2];  // Luz trasera
    shaderList[0].SetSpotLights(tempLights, 2);
}
else {
    shaderList[0].SetSpotLights(spotLights, spotLightCount - 2);
}
```



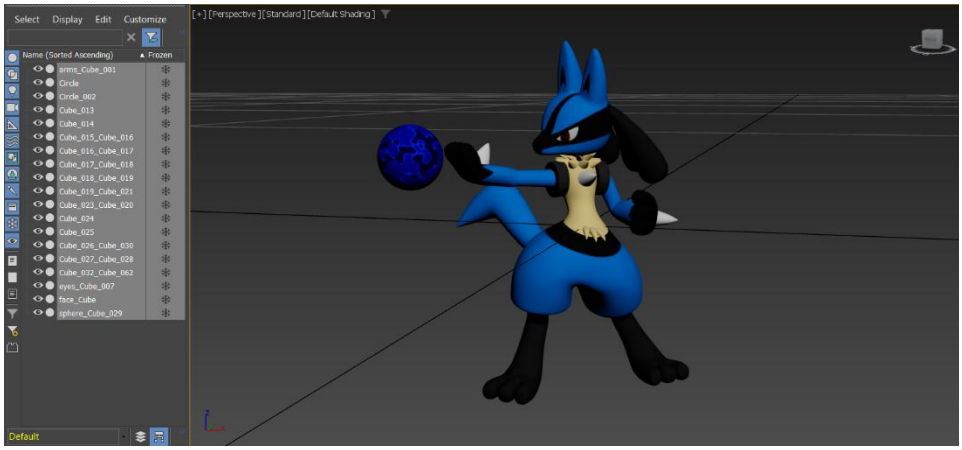




Adicionalmente, se agregó otra condicional, la cual considera la posición del cofre: si está abierto, se enciende la spotlight del cofre, y si está cerrado se apaga. Las otras spotlight siguen funcionando igual. Debido a las especificaciones adicionales dadas en Discord, sobre que la luz del cofre siempre está activa, esta versión está comentada, basta con comentar la otra para ver las diferencias entre versiones.

```
//****************************** Luz del cofre se enciende cuando se abre el cofre, se apaga cuando se cierra

if (mainWindow.getLuzauto() == 0.0f) {
    if (mainWindow.getcofre() > 10.0) {
        shaderList[0].SetSpotLights(spotLights, spotLightCount - 2); //Se enciende la luz del cofre
    }
    else if (mainWindow.getcofre() < 0.0) {
        shaderList[0].SetSpotLights(spotLights, 0); //Se apaga la luz del cofre
    }
}
else {
    if (mainWindow.getLuzauto() == 1.0f) {
        SpotLight tempLights[1];         // Arreglo temporal
        tempLights[0] = spotLights[1];  // Luz trasera
        shaderList[0].SetSpotLights(tempLights, 1);
    }
    else if (mainWindow.getLuzauto() == 2.0f) {
        SpotLight tempLights[1];         // Arreglo temporal
        tempLights[0] = spotLights[2];  // Luz trasera
        shaderList[0].SetSpotLights(tempLights, 1);
    }
}
```

Para el último ejercicio, se descargó un modelo del pokemón Lucario, el cual está realizando el ataque Esfera Aural, esta esfera será el objeto que se iluminará con luz puntual además de la lámpara previamente implementada en la práctica pasada. El modelo se trabajó en 3dsMax, donde se le aplicaron las texturas y se movió su pivote al origen para mayor comodidad a la hora de manipular el modelo en el programa principal.



Una vez que se optimizó el modelo, se agregó el .OBJ y .MTL a la carpeta *Models* del proyecto, mientras que las texturas de Lucario se agregaron a la carpeta *Textures* del proyecto. Dentro del programa, se declara el modelo de Lucario y su ruta del archivo. Mientras que en el *main,* se renderiza el modelo en un lugar donde no estorbe ni al carro ni a la lámpara.

```
Model Lucario_M;
```

```
Lucario_M = Model();
Lucario_M.LoadModel("Models/lucario.obj");
```

```
//********************** Instancia Lucario **********************
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(5.0f, 1.3f, -4.0));
model = glm::scale(model, glm::vec3(2.5f, 2.5f, 2.5f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Lucario_M.RenderModel();
```

Luego, se creó la luz puntual para la esfera aural. Esta luz es de color azul y se encuentra dentro del arreglo de luces puntuales junto a la de la lámpara creada en el ejercicio de clase. Se añaden banderas las cuales registran cada vez que se prende y se apaga la lámpara y la esfera. Para cada objeto se asignan las siguientes teclas:

- Lámpara: C -> Enciende, V -> Apaga.
- Esfera Aural: B -> Enciende, N -> Apaga.

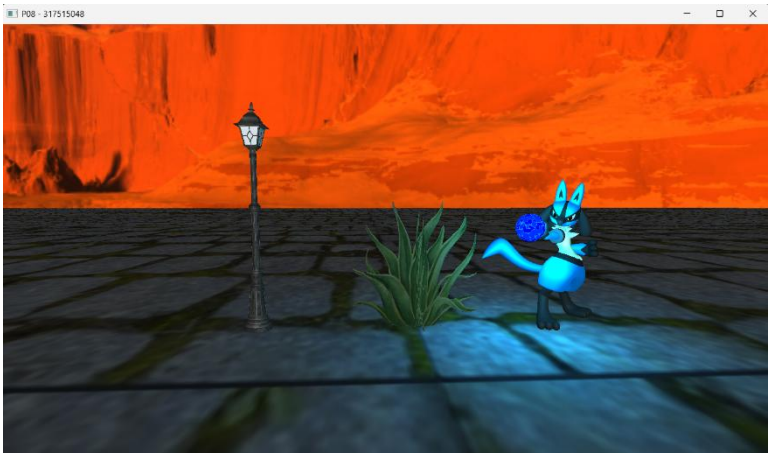
```
//contador de luces puntuales
unsigned int pointLightCount = 0;
//Luz puntual para la lámpara
pointLights[0] = PointLight(1.0f, 1.0f, 1.0f,
    0.25f, 1.0f,            //Intensidad de la
    -5.0f, 5.0f, -4.0,      //Posición
    0.3f, 0.05f, 0.01f);    //Atenuación
pointLightCount++;
//Luz puntual para Esfera Aural
pointLights[1] = PointLight(0.0f, 0.7f, 1.0f,
    0.3f, 1.0f,         //Intensidad de la luz
    3.15f, 2.3f, -2.65f,//Posición
    0.1f, 0.01f, 0.09f);    //Atenuación
pointLightCount++;

unsigned int spotLightCount = 0;
//unsigned int spotLightCount2 = 0;
```

```
//Encendido y apagado de la lámpara
if (key == GLFW_KEY_C)
{
    theWindow-> Lampara = 1.0;
}
if (key == GLFW_KEY_V)
{
    theWindow-> Lampara = 0.0;
}
//Encendido y apagado de la esfera aural
if (key == GLFW_KEY_B)
{
    theWindow->Esfera = 1.0;
}
if (key == GLFW_KEY_N)
{
    theWindow->Esfera = 0.0;
}
```

Finalmente, se agrega las condicionales que utilizan las banderas anteriores y controlan que luces puntuales se van a encender. Para esto se tienen cuatro casos: las dos luces están encendidas, sólo la lámpara está encendida, sólo la esfera está encendida y las dos luces están apagadas. En este caso, debido a su posición en el arreglo, también se hace uso de un arreglo temporal para solo apagar la luz de la lámpara y dejar encendida la luz de la esfera.

```
//Encendido y apagado de luces puntuales
if (mainWindow.getLampara() == 1.0 && mainWindow.getEsfera() == 1.0) {
    shaderList[0].SetPointLights(pointLights, pointLightCount);
}
else {
    if (mainWindow.getLampara() == 1.0 && mainWindow.getEsfera() == 0.0) {
        shaderList[0].SetPointLights(pointLights, pointLightCount - 1);
    }
    else {
        if (mainWindow.getLampara() == 0.0 && mainWindow.getEsfera() == 1.0) {
            PointLight tempLights[1];         // Arreglo temporal
            tempLights[0] = pointLights[1];  // Luz de esfera aural
            shaderList[0].SetPointLights(tempLights, 1);
        }
        else {
            shaderList[0].SetPointLights(pointLights, pointLightCount - 2);
        }
    }
}
```






2.- Problemas presentados

No se presentaron problemas a la hora de realizar los ejercicios solicitados.

3.- Conclusión:

a. Los ejercicios del reporte: Complejidad, Explicación.

La elaboración de esta práctica fue muy interesante de realizar, ya que permitió comprender de mejor manera como se pueden manipular distintos tipos de luces al mismo tiempo, además de aplicar banderas y condicionales para la manipulación del entorno.

b. Comentarios generales: Faltó explicar a detalle, ir más lento en alguna explicación, otros comentarios y sugerencias para mejorar desarrollo de la práctica

La explicación de cómo funcionan los arreglos de luces, así como la creación de banderas para poder aplicar condicionales y como apagar una luz fue clara y precisa.

c. Conclusión

Esta práctica me permitió comprender mejor cómo funcionan los arreglos de luces dentro de OpenGL. Para apagar una luz, se tiene que especificar la cantidad de luces que se mostrarán al momento de usar ya sea función SetPointLights/SetSpotLights, por lo que, si la luz a apagar se encuentra al final del arreglo, basta con restar una luz para hacer este efecto. En cambio, si la luz a apagar se encuentra en una posición del arreglo poco convencional o que se debe apagar la luz de en medio del arreglo, se deben recurrir a otras técnicas como es el uso de arreglos temporales para excluir a esta luz.

1. Bibliografía en formato APA

- 3ds Max Quick Start Guide. (s/f). Autodesk.com. Recuperado el 20 de marzo de 2025, de https://www.autodesk.com/learn/ondemand/curated/3ds-max-quick-start-guide
- De programación, T. (2016, abril 24). Iluminación - Tipos de Luces. Blogspot.com. https://acodigo.blogspot.com/2016/04/iluminacion-tipos-de-luces.html
- Lucario - Download Free 3D model by Gianmarco (@GianmArt). (2021, noviembre 16). https://sketchfab.com/3d-models/lucario-32ab2458321e495084fe3bc3b3bf6a91