



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:** Gabriel Patricio Balam Flores

**Nº de Cuenta:** 320280324

**GRUPO DE LABORATORIO:** 03

**GRUPO DE TEORÍA:** 04

**SEMESTRE 2026-1**

**FECHA DE ENTREGA LÍMITE:** 26/oct

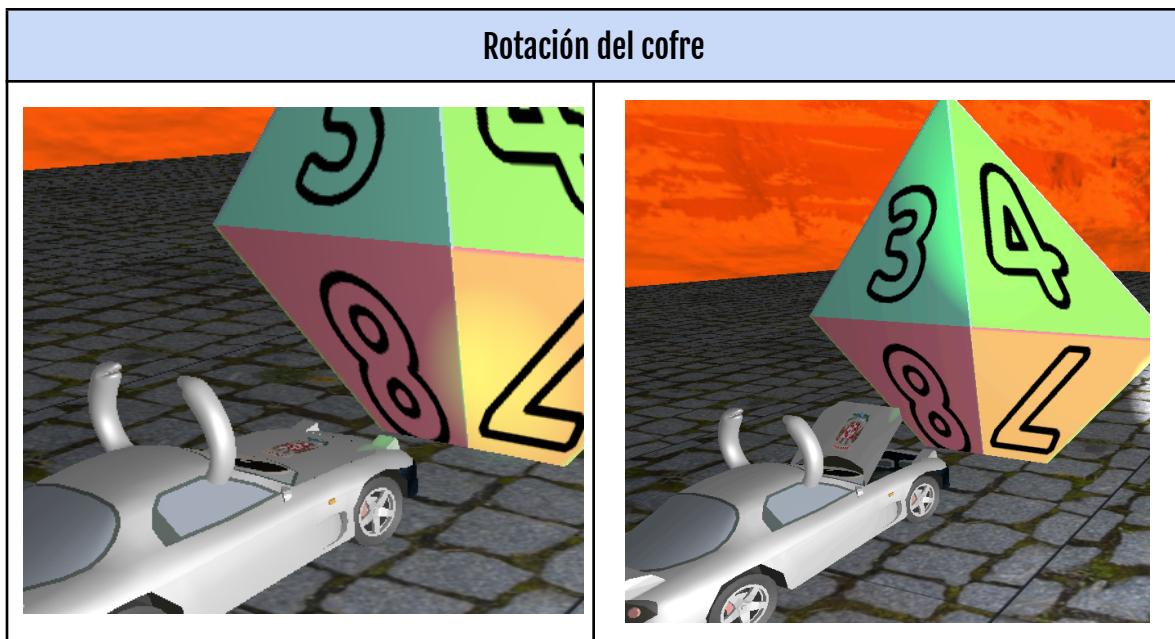
**CALIFICACIÓN:** \_\_\_\_\_

## 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.

### Ejercicio #01

El ejercicio consistía en descubrir cómo rotar la luz en tiempo de ejecución. Fue el más tardado de todos, principalmente por la investigación que tuve que realizar para lograrlo. Al final, la solución resultó bastante sencilla: solo era necesario enviar la matriz del modelo junto con el ángulo de la luz, y listo.



Se rota con la tecla [ I ].

Variable de dirección	Incrementos
<pre>glm::vec3 baseLightDirFaro(1.0f, 0.0f, 0.0f);</pre> <pre>GLfloat articulacion1;</pre> <pre>GLfloat dir1;</pre>	<pre>if (key == GLFW_KEY_I)</pre> <pre>{</pre> <pre>    theWindow-&gt;articulacion1 += 2.5 * theWindow-&gt;dir1;</pre> <pre></pre> <pre>    if (theWindow-&gt;articulacion1 == 60)</pre> <pre>        theWindow-&gt;dir1 *= -1.0f;</pre> <pre>    else if (theWindow-&gt;articulacion1 == 0)</pre> <pre>        theWindow-&gt;dir1 *= -1.0f;</pre> <pre>}</pre>

## Agregar movimiento

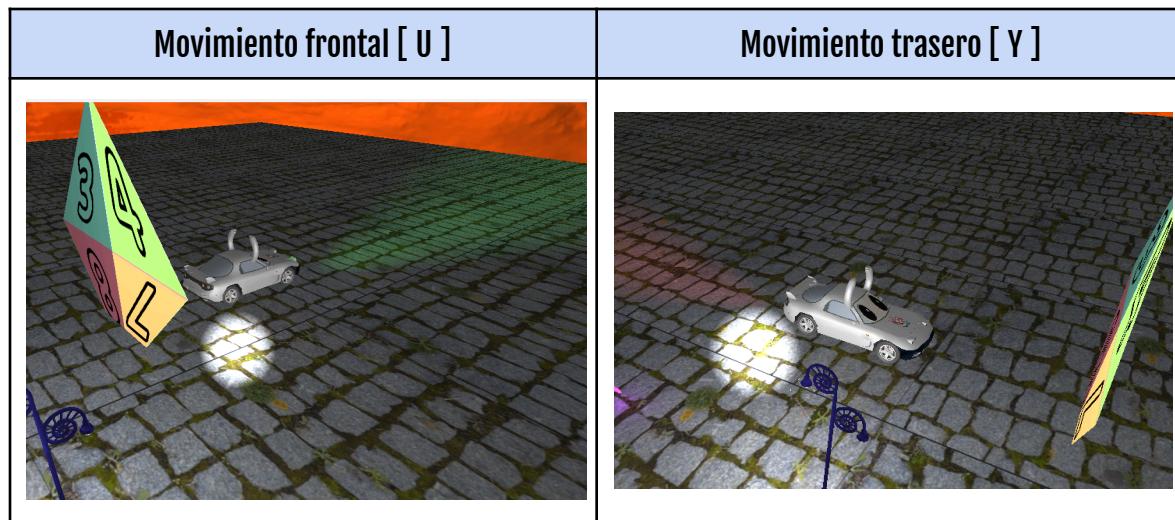
```
// Para el cofre
model = auxmodel;
model = glm::translate(model, glm::vec3(0.777f, 0.3625f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacion1()), glm::vec3(0.0f, 0.0f, 1.0f));

// Movimiento y rotación de la luz
spotLights2[0].SetFlash(glm::vec3(model* glm::vec4(baseLightPosFaro, 1.0f)),
glm::normalize(glm::vec3(model* glm::vec4(baseLightDirFaro, 0.0f))));

glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Carro_cofre.RenderModel();
```

## Ejercicio #02

Este ejercicio implica la gestión de dos luces que se encienden y apagan en función de la dirección del movimiento del coche. La implementación se basa en una bandera que indica la dirección del vehículo y una condición 'if' que determina qué configuración de luces debe activarse.



Declarar la luz	Incrementos
<pre>//Luz faro trasero spotLights[1] = SpotLight(0.451f, 0.0784f, 0.0667f, 1.0f, 2.0f, 0.0f, 0.0f, 0.0f, -1.0f, 0.0f, 0.0f, 2.0f, 0.0f, 0.0f, 25.0f); spotLightCount++; glm::vec3 baseLightPosFaroAtras(-2.0f, 0.0f, 0.0f);</pre>	<pre>if (key == GLFW_KEY_U) {     theWindow-&gt;muevex += 1.0;     theWindow-&gt;adelante = true; }</pre>

```

//luz faro delantero
spotLights[2] = SpotLight(0.0667f, 0.451f, 0.0784f,
    1.0f, 2.0f,
    0.0f, 0.0f, 0.0f,
    1.0f, 0.0f, 0.0f,
    2.0f, 0.0f, 0.0f,
    25.0f);
spotLightCount++;
glm::vec3 baseLightPosFaro(1.0f, 0.0f, 0.7f);
glm::vec3 baseLightDirFaro(1.0f, 0.0f, 0.0f);

//Segundo arreglo de luces puntuales
spotLights2[0] = spotLights[2];
spotLights2[1] = spotLights[0];
spotLights2[2] = spotLights[1];

```

```

if (key == GLFW_KEY_Y)
{
    theWindow->muevex -= 1.0;
    theWindow->adelante = false;
}
else
{
    theWindow->muevex += 1.0;
    theWindow->adelante = true;
}

```

## Implementación

```

// Para el movimiento -----
mx = mainWindow.getmuevex();

if (mainWindow.getadelante())
{
    shaderList[0].SetSpotLights(spotLights2, spotLightCount -1);
}
else
{
    shaderList[0].SetSpotLights(spotLights, spotLightCount - 1);
}

// Movimiento del coche
model = glm::translate(model, glm::vec3(mx, 0.0f, 0.0f));

// Movimiento de la luz
spotLights[1].SetPos(glm::vec3(model * glm::vec4(baseLightPosFaroAtras, 1.0f)));

auxmodel = model;

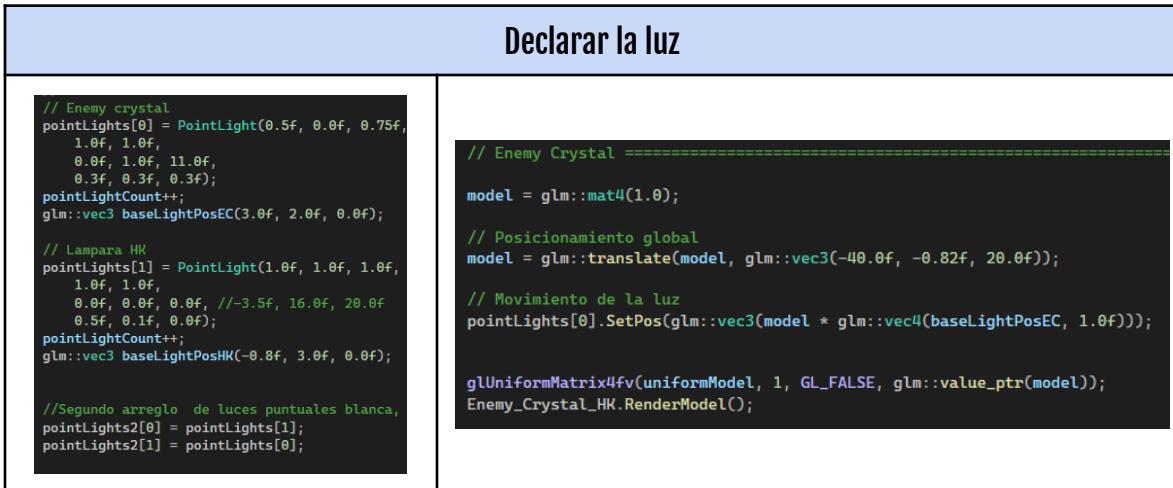
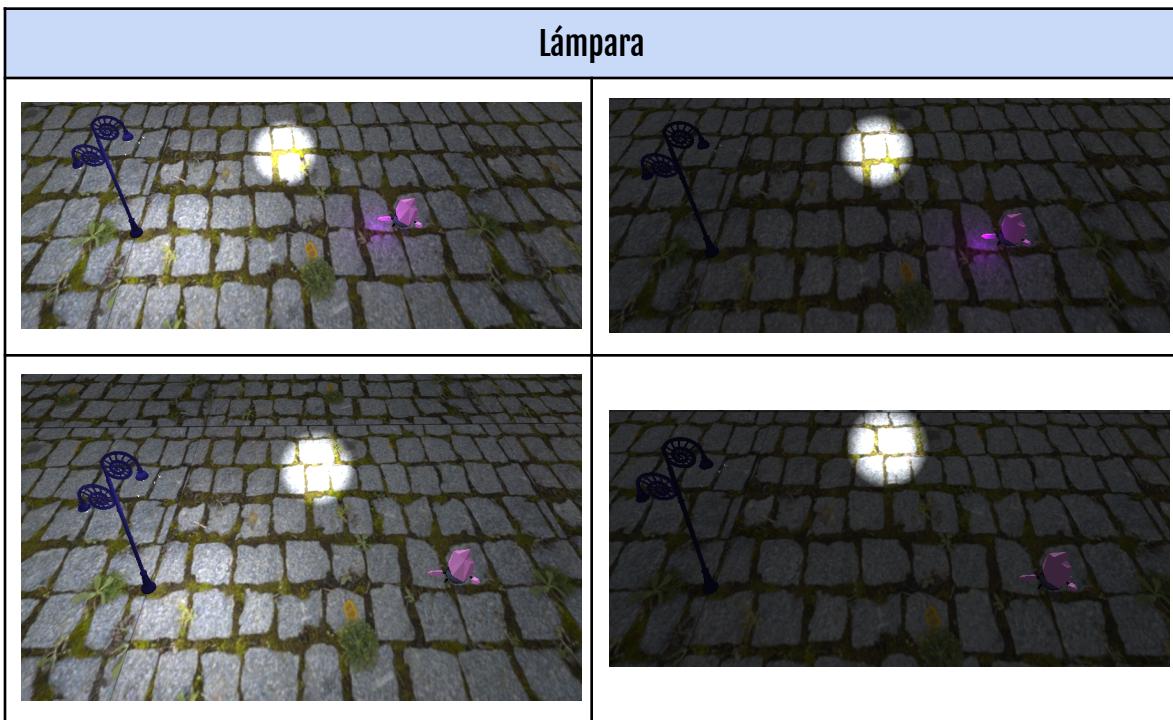
// Para el cuerpo del carro
model = glm::translate(model, glm::vec3(0.44f, 0.0f, 0.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Carro_cuerpo.RenderModel();

// Para el cofre
model = auxmodel;
model = glm::translate(model, glm::vec3(0.777f, 0.3625f, 0.0f));
model = glm::rotate(model, glm::radians(mainWindow.getarticulacionl()), glm::vec3(0.0f, 0.0f, 1.0f));

// Movimiento y rotación de la luz
spotLights2[0].SetFlash(glm::vec3(model* glm::vec4(baseLightPosFaro, 1.0f)),
    glm::normalize(glm::vec3(model* glm::vec4(baseLightDirFaro, 0.0f))));
```

## Ejercicio #03

Este ejercicio fue el más fácil. Consistió en agregar dos luces de tipo puntual e implementar una funcionalidad para que cada una pudiera encenderse y apagarse de forma independiente. La solución la implementé con una estructura if que podría considerarse similar a un switch-case, ya que toma en cuenta todos los casos posibles de las luces y, dependiendo de cada uno, se envía el arreglo correspondiente al shader.



## Implementación

```
if (mainWindow.getPrendida() && mainWindow.getPrendida2())
{
    shaderList[0].SetPointLights(pointLights, pointLightCount);
}
else if (mainWindow.getPrendida() && !mainWindow.getPrendida2())
{
    shaderList[0].SetPointLights(pointLights, pointLightCount - 1);
}
else if (!mainWindow.getPrendida() && mainWindow.getPrendida2())
{
    shaderList[0].SetPointLights(pointLights2, pointLightCount - 1);
}
else
{
    shaderList[0].SetPointLights(pointLights, pointLightCount -2);
}
```

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.

## Contratiempos

Lo más complicado de la práctica fue encontrar la manera correcta de manejar la matriz model para lograr una rotación adecuada de la luz, ya que, si se gestiona de forma incorrecta, la luz termina afectándose en su traslación y en sus parámetros de intensidad.

3.- Conclusión:

- a. Los ejercicios del reporte: Complejidad, Explicación.
  - b. Comentarios generales.
  - c. Conclusión
1. Bibliografía en formato APA

## Conclusión

Esta práctica me ayudó a comprender mejor cómo manipular las luces dentro de OpenGL. Ahora soy capaz de implementar correctamente la

traslación, rotación, encendido y apagado de luces, además de crear la ilusión de tener más luces de las que realmente soporta mi computadora.

## Bibliografía

- Página principal. (26 de septiembre de 2018). *Wiki de OpenGL* . Recuperado el 29 de agosto de 2025 de [http://www.khronos.org/opengl/wiki\\_OpenGL/index.php?title=Main\\_Page&oldid=14430](http://www.khronos.org/opengl/wiki_OpenGL/index.php?title=Main_Page&oldid=14430) .