



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: Gabriel Patricio Balam Flores

Nº de Cuenta: 320280324

GRUPO DE LABORATORIO: 03

GRUPO DE TEORÍA: 04

SEMESTRE 2026-1

FECHA DE ENTREGA LÍMITE: 19/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 primer ejercicio de la práctica consiste en darle movimiento al helicóptero hacia adelante [J] y hacia atrás [H]. Para lograrlo, se debe agregar la variable de posición en los archivos windows.h y windows.cpp. Posteriormente, se definen los incrementos correspondientes al presionar cada una de las teclas, y con ello el movimiento queda implementado.



Variable de posición	Incrementos
<pre>GLfloat getmx() { return mx; } GLfloat muevex, mx; mx = 0.0f;</pre>	<pre>if (key == GLFW_KEY_H) { theWindow->mx += 1.0; } if (key == GLFW_KEY_J) { theWindow->mx -= 1.0;</pre>
Aregar movimiento	
<pre>// Helicoptero ===== model = glm::mat4(1.0);</pre>	

```

// Inicial
model = glm::translate(model, glm::vec3(0.0f, 50.0f, 6.0));
model = glm::scale(model, glm::vec3(5.0f, 5.0f, 5.0f));

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

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

// Movimiento de la luz
newPos = glm::vec3(model * glm::vec4(baseLightPosH, 1.0f));
spotLights[2].SetPos(newPos);

// Rotar
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();

```

Ejercicio #02

El segundo ejercicio de la práctica es muy similar al realizado en clase. Consiste en agregar una luz spotlight de color amarillo al helicóptero, la cual debe moverse junto con él. Para lograrlo, se crea una nueva luz spotlight, se agrega a la lista correspondiente y, dentro del ciclo while, se actualiza su posición multiplicando su posición inicial por la matriz de transformación del helicóptero.



Declarar la luz	Posición inicial
<pre> //luz helicoptero spotLights[2] = SpotLight(1.0f, 1.0f, 0.0f, 1.0f, 1.0f, 0.0f, 0.0f, 0.0f, -1.0f, -1.0f, 0.0f, 2.0f, 0.0f, 0.0f, 25.0f); spotLightCount++; </pre>	<pre> glm::vec3 baseLightPosH(-2.7f, -0.7f, 0.45f); </pre>

Jerarquía de luz

```
// Movimiento de la luz  
newPos = glm::vec3(model * glm::vec4(baseLightPosH, 1.0f));  
spotLights[2].SetPos(newPos);
```

Ejercicio #03

El último ejercicio de la práctica consiste en añadir el modelo de la lámpara de nuestro universo, la cual se utilizará en el proyecto final, y agregarle una luz puntual.

La parte del modelo fue sencilla, ya que este procedimiento se ha realizado en múltiples ocasiones. Primero se importa el modelo en Blender y se le asigna la textura mediante una imagen. Posteriormente, se exporta el modelo (en mi caso, en formato .obj) y se incorpora al proyecto en OpenGL. Una vez dentro, se posiciona y escala según sea necesario, y con esto queda listo.

La adición de la luz es similar al caso del helicóptero: se declara la luz y sus parámetros, y luego se posiciona en el foco de la lámpara.

Lámpara



Textura



Blanco



Declarar la luz

```
// Lampara HK
pointLights[0] = PointLight(1.0f, 1.0f, 1.0f,
    1.0f, 1.0f,
    -3.5f, 16.0f, 20.0f,
    0.5f, 0.1f, 0.0f);
pointLightCount++;
```

Modelo

```
// Lampara =====
model = glm::mat4(1.0);

// Posicionamiento global
model = glm::translate(model, glm::vec3(0.0f, -0.82f, 20.0f));
model = glm::scale(model, glm::vec3(5.0f, 5.0f, 5.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Lampara_HK.RenderModel();
```

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.

Contriempos

Esta práctica no presentó ningún problema ni contratiempo, gracias a la explicación del profesor y a la sencillez de las actividades.

3.- Conclusión:

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

Conclusión

En esta práctica aprendí a crear e implementar diferentes tipos de luces en OpenGL. También comprendí cómo aplicar jerarquías para asignar una luz a un modelo específico.

Además, aprendí a utilizar los parámetros de cada tipo de luz para configurarlas según las necesidades de la escena.

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 .