
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:** Barragán Pilar Diana

**Nº de Cuenta:** 318147981

**GRUPO DE LABORATORIO:** 03

**GRUPO DE TEORÍA: 04**

**SEMESTRE 2025-1**

**FECHA DE ENTREGA LÍMITE: 04 de octubre de 2024**

**CALIFICACIÓN: _____**

# Práctica 7: Iluminación 1

## 1. Agregar movimiento con teclado al helicóptero hacia adelante y atrás.

Para agregar movimiento al helicóptero primero guarde en un vector su posición ya que esto lo ocupe más adelante para la traslación y posicionamiento del helicóptero, así como para la luz que se moverá con él. En el vector utilice la función `mainWindow.getmuevex()` que hace que se mueva el helicóptero de adelante hacia atrás con las teclas Y y U declarado esto en Windows.cpp, sumándole al origen la posición que da el usuario al presionar las teclas antes mencionadas.

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

Utilice el arreglo de `spotLights[2]` junto con la función `SpotLight` para crear la luz del helicóptero, en ella en los tres primeros parámetros puesto que corresponden al RGB, configure las variables para que la luz fuera de color amarillo, posteriormente deje en 1 y 2 los valores de la intensidad ambiental y difusa, después posicione la luz en el origen para que sea ahí donde se crea, luego a la dirección le asigne un valor de -5 en y para que está se viera de arriba en el helicóptero hacia es piso, por ultimo deje el ángulo de la luz en 15.

```
//Luz de helicóptero
spotLights[2] = SpotLight(1.0f, 1.0f, 0.0f,
    1.0f, 2.0f,
    0.0f, 0.0f, 0.0f,
    0.0f, -5.0f, 0.0f,
    1.0f, 0.0f, 0.0f,
    15.0f);
spotLightCount++;
```

Figura 1. Declaración de la luz del helicóptero.

Más adelante para que está luz que cree se mueva junto con el helicóptero, utilice el vector de posición antes mencionado para saber en todo momento donde se ubica, luego cree un nuevo vector con la posición de la luz en la que se sumará el lugar donde se encuentra el helicóptero más la traslación para que la luz se vea justo debajo de él, finalmente esta posición se manda al método `SetPos` que actualiza la posición de la luz.

```
//Helicóptero
model = glm::mat4(1.0);
//Linea de codigo para poder mover el helicóptero con el teclado
glm::vec3 PosHeli = glm::vec3(0.0f + mainWindow.getmuevex(), 5.0f, 6.0f);
model = glm::translate(model, PosHeli);
//Vector con la posición del helicóptero más la traslación de la luz
glm::vec3 PosLuz = PosHeli + glm::vec3(-0.1f, 0.0f, 0.0f);
spotLights[2].SetPos(PosLuz);
model = glm::scale(model, glm::vec3(0.3f, 0.3f, 0.3f));
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();
```

Figura 2. Código para que la luz se mueva con el helicóptero.

3. Añadir en el escenario 1 modelo de lámpara texturizada (diferente a los que usarán en su proyecto final) y crearle luz puntual blanca.

Para añadir al escenario el modelo de lampara realice el mismo procedimiento que en practicas anteriores, primero cambie el pivote del modelo, lo posicione en el origen y lo exporte en 3ds Max, después en el código declare el objeto de tipo `Model` llamado `Lamp` para posteriormente cargar el modelo especificando la ruta donde se encuentra.

```
Model Llanta4;
Model Capo;
Model Lamp;
```

```
Capo.LoadModel("Models/capo.obj");
Lamp = Model();
Lamp.LoadModel("Models/lampara.obj");
```

Figura 3. Declaración y carga del modelo.

Continué creando la luz puntual que iba a utilizar en la lampará, con el arreglo `pointLights[0]` y la función `PointLight` ajustando los primeros tres valores que corresponden al RGB en uno para que la luz sea de color blanco, luego la intensidad ambiental y difusa las deje con los valores de 0.5 y 1, posteriormente en el vector de posición la coloque en donde se supone debe estar el foco de la lámpara, por ultimo deje en 0.1, 0.3 y 0.1 a las variables de con, lin y exp.

```
//Luz puntual de la lampara
pointLights[0] = PointLight(1.0f, 1.0f, 1.0f,
    0.5f, 1.0f,
    -5.0f, 3.7f, 0.0f,
    0.1f, 0.3f, 0.1f);
pointLightCount++;

unsigned int spotLightCount = 0;
```

Figura 4. Declaración de la luz puntual.

Para terminar, traslade el modelo de la lampara a un lugar en el plano donde se pudiera apreciar mejor, acomode su escala y renderice el modelo con `RenderModel`.

```
//Lámpara
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-5.0f, -0.1f, 0.0));
model = glm::scale(model, glm::vec3(0.3f, 0.3f, 0.3f));
color = glm::vec3(0.3f, 0.3f, 0.3f);
glUniform3fv(uniformColor, 1, glm::value_ptr(color));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
Lamp.RenderModel();
```

Figura 5. Posicionamiento y renderizado del modelo.

Al compilar el código de esta práctica no se me presento ningún inconveniente respecto a la programación de este, puesto que todo funciono con normalidad.

**CONCLUSIÓN:**

Al realizar los ejercicios asignados en la practica me fue posible comprender de mejor manera todo lo correspondiente a como se maneja la iluminación en opengl, al cambiar los parámetros de distintos tipos de luces como lo son la puntual o la de tipo reflector que se nos pidieron, conociendo los efectos que estas pueden aportar a un escenario, al igual que incorporarlos a otros objetos haciendo que estas se trasladen como fue el caso del helicóptero o que simplemente estén presentes como la lámpara.