



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 N° 7

NOMBRE COMPLETO: Gómez Enríquez Agustín

N° de Cuenta: 317031405

GRUPO DE LABORATORIO: 3

GRUPO DE TEORÍA: 5

SEMESTRE 2026-1

FECHA DE ENTREGA LÍMITE: 19 de octubre del 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.

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

```
21
22     float getHeliMoveZ() const { return heliMoveZ; }
23     void updateHeliMovement(float dt);    // integra J/H con deltaTime
24
25     ~Window();
26 private:
27     GLFWwindow *mainWindow;
28     GLint width, height;
29     bool keys[1024];
30     GLint bufferWidth, bufferHeight;
31     void createCallbacks();
32     GLfloat lastX;
33     GLfloat lastY;
34     GLfloat xChange;
35     GLfloat yChange;
36     GLfloat muevex;
37     bool mouseFirstMoved;
38     static void ManejaTeclado(GLFWwindow* window, int key, int code, int action);
39     static void ManejaMouse(GLFWwindow* window, double xPos, double yPos);
40
41     float heliMoveZ = 0.0f;    // acumulador Z del helicóptero
42     float heliSpeed = 1.0f;    // velocidad (unidades/seg)
43
```

Debido a que en practicas anteriores como la grúa realice el control de los movimientos desde el while del programa principal, pero considerando el comentario del profesor de optimizar código y no todo trabajarlo en el loop, opte por trabajarlo casi igual que como se tiene el movimiento del carro, que es agregando mis globales desde el window.h en el público y en el privado.

```
131     void Window::updateHeliMovement(float dt)
132     {
133         // J = avanzar, H = retroceder
134         if (keys[GLFW_KEY_J]) {
135             heliMoveZ += heliSpeed * dt;    // avanzamos
136         }
137         if (keys[GLFW_KEY_H]) {
138             heliMoveZ -= heliSpeed * dt;    // retrocedemos
139         }
140     }
```

Posteriormente trabajamos en el Window.cpp para agregar nuestra función de movimiento desde teclado con las teclas J y H respectivamente.

```

446     model = glm::mat4(1.0f);
447     //model = glm::translate(model, glm::vec3(0.0f, 5.0f, 6.0f));
448     model = glm::translate(model, glm::vec3(0.0f + mainWindow.getHeliMoveZ(), 5.0f, 6.0f));
449     model = glm::scale(model, glm::vec3(0.3f, 0.3f, 0.3f));
450     model = glm::rotate(model, -90 * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
451     model = glm::rotate(model, 90 * toRadians, glm::vec3(0.0f, 0.0f, 1.0f));
452     glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
453     Blackhawk_M.RenderModel();

```

Por último, modificamos la línea de translate que se encuentra en el loop para que al mantener presionado la tecla, la acción se ejecute hasta que se deje de presionar, (Nota: Corregí después el eje, pensé que era en Z pero se modifica el movimiento sobre el eje X, por eso el incremento va en X)

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

```

292     spotLightCount++;
293
294     spotLights[3] = Spotlight(
295         1.0f, 1.0f, 0.0f,    // color amarillo (R,G,B)
296         4.0f, 1.0f,         // intensidades: ambiental, difusa
297         0.0f, 0.0f, 0.0f,    // posición (se actualizará cada frame)
298         0.0f, -1.0f, 0.0f,    // dirección hacia el piso
299         1.0f, 0.022f, 0.0019f, // atenuación (alcance realista)
300         20.0f                // ángulo del cono (edge en grados)
301     );
302     spotLightCount++;
303

```

Como trabajaremos con spotlight, agregamos una cuarta luz solamente modificando el color para que se muestre de color amarillo y el resto de valores los trabajare al igual que la luz del faro del carro.

```

375     glm::vec3 color = glm::vec3(1.0f, 1.0f, 1.0f);
376
377
378     glm::mat4 M_heli = glm::mat4(1.0f);
379
380     //Luz del helicóptero
381     M_heli = glm::translate(M_heli, glm::vec3(0.0f + mainWindow.getHeliMoveZ(), 5.0f, 6.0f));
382     M_heli = glm::scale(M_heli, glm::vec3(0.5f));
383     M_heli = glm::rotate(M_heli, -90.0f * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
384     M_heli = glm::rotate(M_heli, 90.0f * toRadians, glm::vec3(0.0f, 0.0f, 1.0f));
385
386     glm::vec3 heliCenter = glm::vec3(M_heli * glm::vec4(0.0f, 0.0f, 0.0f, 1.0f));
387     glm::vec3 heliLightPos = heliCenter + glm::vec3(0.0f, 0.0f, 0.0f);
388     glm::vec3 heliLightDir = glm::vec3(0.0f, -1.0f, 0.0f);
389     spotLights[3].SetFlash(heliLightPos, heliLightDir);
390

```

Ahora para la parte del movimiento junto con el helicóptero lo trabajamos desde el loop para actualizar su posición con cada clic, y posicionamos la luz debajo del helicóptero copiando sus coordenadas.

ACTIVIDAD 3: Añadir en el escenario 1 modelo de lámpara texturizada (que usarán en su proyecto final) y crearle luz puntual blanca

```
224 Reflector_M = Model();
225 Reflector_M.LoadModel("Models/Reflector halo AdiXXioN.fbx");
226
```

El modelo de reflector que se utilizara en el proyecto final se estaba trabajando en blender, sin embargo para trabajarlo en esta práctica, lo exporte como .fbx, pero no le cargue ninguna textura.

```
256
257 //Segunda luz puntual (Reflector)
258 pointLights[1] = PointLight(1.0f, 1.0f, 1.0f,
259     1.0f, 1.0f,
260     0.0f, 0.0f, -7.0f,
261     0.3f, 0.2f, 0.1f);
262 pointLightCount++;
263
```

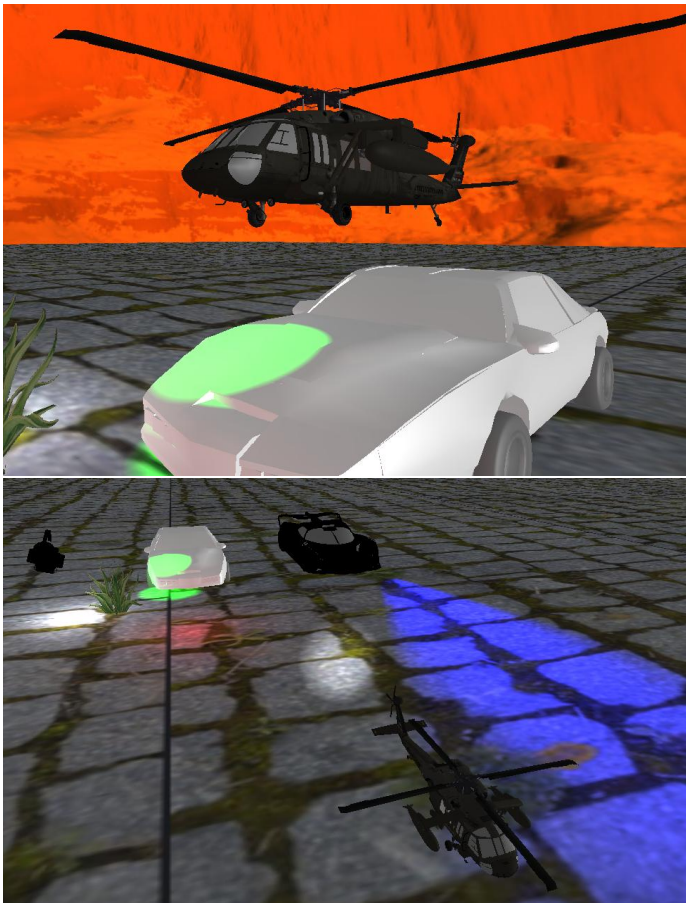
Ya que se solicita una pointLight, esta será la segunda luz que definimos en el proyecto, pero esta vez con el color blanco, y por fines prácticos coloque la intensidad al máximo y que se pudiera apreciar. Y como en esta actividad no se solicita movimiento con respecto al proyector, solo definimos una posición que coincida con la del proyector.

```
454
455 //Reflector
456 model = glm::mat4(1.0);
457 model = glm::translate(model, glm::vec3(-4.5f, 4.5f, -7.0f));
458 model = glm::scale(model, glm::vec3(7.0f, 7.0f, 7.0f));
459 model = glm::rotate(model, -165 * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
460 model = glm::rotate(model, -45 * toRadians, glm::vec3(0.0f, 0.0f, 1.0f));
461 glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
462 Reflector_M.RenderModel();
463
```

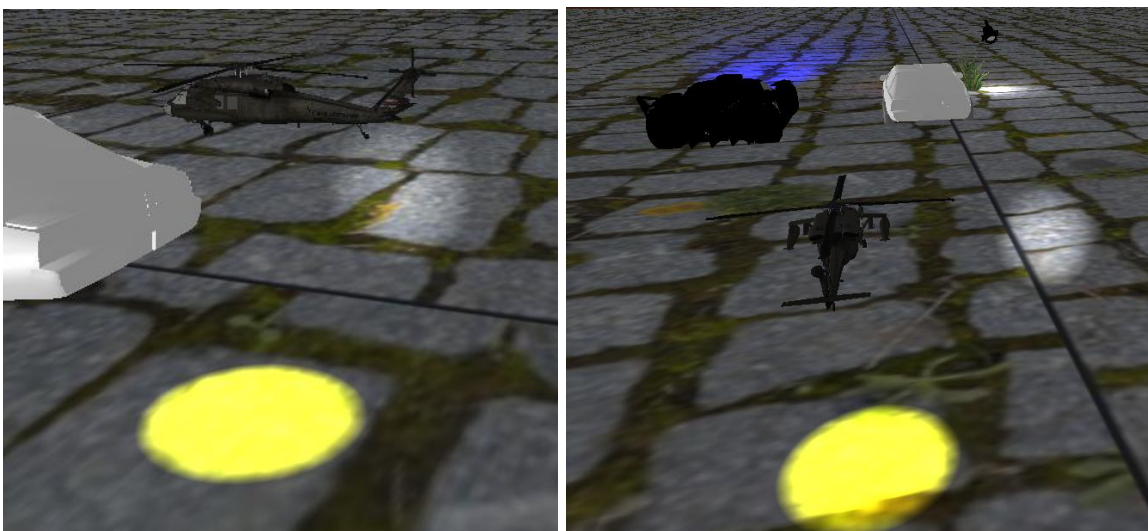
Lo agregamos como un modelo simple sin ejecuciones y solo ajustamos su posición y tamaño para que coincidan con el de los demás elementos del escenario.

EJECUCIÓN

ACTIVIDAD 1



ACTIVIDAD 2



ACTIVIDAD 3



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

Honestamente no tuve errores con la programación, pero si con las funciones de visual, puesto que sin querer presioné la tecla de INS y me editaba el texto y no pude estar haciendo copy + paste tan fácil para reciclar el código de otras instancias o funciones, pero a su vez me permitió conocer el porque y para que, de cada línea de código, mas adelante descubrí como quitar esa función del teclado y pude seguir trabajando normal.

3.- Conclusión:

Trabajar con luces me permitió conocer las características que tiene la luz por si misma, como su intensidad, el radio que ocupa, su estela de luz y el movimiento que se maneja con el objeto de que la emite. Sin embargo, aún me queda la duda de como trabajar con la estela de luz en un ambiente vertical, porque en el carro si puedo dibujar el paso por donde avanza y reducir la intensidad con la longitud, pero porque esta se refleja en los materiales, pero para trabajarlo en el aire lo intente, pero solo lo lograba insertando un cono del mismo color. así que aun queda por trabajar en la luz, pero esta primera practica fue un gran acercamiento y también a la optimización de código.

Bibliografía en formato APA

- de la Computación e Inteligencia Artificial, C. (n.d.). *Color e iluminación*. Uhu.Es. Retrieved October 18, 2025, from https://www.uhu.es/francisco.moreno/gii_rv/docs/Tema_6.pdf