



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° 8

NOMBRE COMPLETO: Gómez Enríquez Agustín

N° de Cuenta: 317031405

GRUPO DE LABORATORIO: 3

GRUPO DE TEORÍA: 5

SEMESTRE 2026-1

FECHA DE ENTREGA LÍMITE: 26 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 un spotlight (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.

```
58 // --- NUEVO: control del cofre del Bugatti y faros/spotlights del coche ---
59 float hoodAngleDeg = 0.0f; // ángulo actual del cofre
60 float targetHoodAngleDeg = 0.0f; // objetivo (0° cerrado, 90° abierto)
61 const float hoodSpeedDeg = 1.0f; // velocidad de animación (grados/segundo, se siente lenta/suave)
62
63 glm::vec3 bugattiBasePos = glm::vec3(12.0f, -1.0f, 17.0f); // misma base que ya usas al dibujar el bugatti
64 // Pivote local del cofre (bisagra). Ajusta finamente si no coincide perfecto con tu malla:
65 glm::vec3 hoodPivotLocal = glm::vec3(0.0f, 0.58f, -0.80f);
66 // Punto "punta del cofre" (para poner la luz amarilla). También puedes afinarlo:
67 glm::vec3 hoodTipLocal = hoodPivotLocal + glm::vec3(0.0f, 0.0f, 2.0f);
68
69 // Para detectar avance/retroceso del coche (muevex):
70 float prevMuevex = 0.0f;
71 bool forwardBeam = false; // avanzar = X negativa
72 bool reverseBeam = false; // retroceder = X positiva
73
```

Creamos nuestras variables globales para trabajar la luz led cofre y las luces que van adelante y detrás del carro.

```
practica8 (Ámbito global)
337 // [1] LUZ DEL COFRE (amarilla, NO blanca/azul)
338 spotLights[1] = SpotLight(
339     1.0f, 0.95f, 0.2f, // color (R,G,B) -> amarillo cálido
340     5.0f, 1.2f, // intensidades: ambient, diffuse
341     0.0f, 0.0f, 0.0f, // pos placeholder (se actualiza cada frame)
342     0.0f, 0.0f, 0.0f, // dir placeholder (se actualiza cada frame)
343     1.0f, 0.022f, 0.0019f, // atenuación realista
344     18.0f // edge/apertura
345 );
346 spotLightCount++;
347
348 // [2] SPOTLIGHT ADELANTE (se enciende al avanzar, X negativa)
349 spotLights[2] = SpotLight(
350     1.0f, 1.0f, 1.0f, // blanco
351     8.0f, 2.0f, // intensidades
352     0.0f, 0.0f, 0.0f, // pos placeholder
353     0.0f, -1.0f, 0.0f, // dir placeholder
354     1.0f, 0.022f, 0.0019f, // atenuación
355     16.0f
356 );
357 spotLightCount++;
358
359 // [3] SPOTLIGHT ATRÁS (se enciende al retroceder, X positiva)
360 spotLights[3] = SpotLight(
361     1.0f, 1.0f, 1.0f, // blanco (puedes poner rojizo si quieres "luz de reversa")
362     8.0f, 2.0f, // intensidades
363     0.0f, 0.0f, 0.0f, // pos placeholder
364     0.0f, -1.0f, 0.0f, // dir placeholder
365     1.0f, 0.022f, 0.0019f, // atenuación
366     16.0f
367 );
368 spotLightCount++;
```

Ahora dentro del main vamos a instanciar nuestras 3 luces spotlight, recordemos comentar las luces que ya estaban en el escenario para no tener creadas tantas luces y hacer que se crashee nuestro programa.

```

391 // --- NUEVO: control por teclas del cofre ---
392 // U = abrir (hasta 90°), 0 = cerrar (hasta 0°)
393 if (mainWindow.getKeys()[GLFW_KEY_O]) targetHoodAngleDeg = 90.0f;
394 if (mainWindow.getKeys()[GLFW_KEY_I]) targetHoodAngleDeg = 0.0f;
395
396 // Interpolación suave (clamp) hacia el objetivo
397 if (fabs(targetHoodAngleDeg - hoodAngleDeg) > 0.01f) {
398     float step = hoodSpeedDeg * deltaTime;
399     if (hoodAngleDeg < targetHoodAngleDeg) hoodAngleDeg = fminf(hoodAngleDeg + step, targetHoodAngleDeg);
400     else hoodAngleDeg = fmaxf(hoodAngleDeg - step, targetHoodAngleDeg);
401 }

```

Agregamos nuestras teclas para mover el cofre, Abrimos con O y cerramos con I. El movimiento tendrá un tope que es en 90 grados para que tenga una función lógica al mundo real.

```

461 // Seguimiento de spotlights del coche (adelante/atrás)
462 glm::mat4 M_car = glm::mat4(1.0f);
463 M_car = glm::translate(M_car, glm::vec3(0.0f + mainWindow.getmuevex(), 0.5f, -3.0f));
464 M_car = glm::scale(M_car, glm::vec3(0.5f));
465 M_car = glm::rotate(M_car, -90.0f * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
466
467 // Posiciones locales de faro delantero y trasero (ajusta si tu malla no coincide)
468 glm::vec3 headLocal = glm::vec3(40.0f, 0.0f, -13.0f); // delantero
469 glm::vec3 tailLocal = glm::vec3(-47.0f, 0.0f, -13.0f); // trasero (estimado)
470
471 // A mundo
472 glm::vec3 headPos = glm::vec3(M_car * glm::vec4(headLocal, 1.0f));
473 glm::vec3 tailPos = glm::vec3(M_car * glm::vec4(tailLocal, 1.0f));
474
475 // Direcciones a mundo
476 glm::vec3 forwardDir = glm::normalize(glm::vec3(M_car * glm::vec4(0.0f, 0.0f, 1.0f, 0.0f)));
477 glm::vec3 backwardDir = -forwardDir;

```

No olvidemos definir las posiciones y el seguimiento de las luces que van pegadas al modelo de nuestro carro.

Actividad 2: *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. Son dos spotlights diferentes que se prenderán y apagarán de acuerdo a alguna bandera asignada por ustedes.*

```

478 // Encendido según banderas
479 if (forwardBeam) {
480     // Enciende el de adelante
481     // Si tu clase SpotLight tiene setters de intensidades:
482     // spotLights[2].SetAmbientIntensity(0.1f);
483     // spotLights[2].SetDiffuseIntensity(2.0f);
484     spotLights[2].SetFlash(headPos, forwardDir);
485 }
486 else {
487     // Apaga (si no tienes setters, deja un cono pequeño sin afectar)
488     // spotLights[2].SetAmbientIntensity(0.0f);
489     // spotLights[2].SetDiffuseIntensity(0.0f);
490     spotLights[2].SetFlash(headPos, forwardDir);
491 }
492
493 if (reverseBeam) {
494     // Enciende el de atrás
495     // spotLights[3].SetAmbientIntensity(0.1f);
496     // spotLights[3].SetDiffuseIntensity(2.0f);
497     spotLights[3].SetFlash(tailPos, backwardDir);
498 }
499 else {
500     // Apaga
501     // spotLights[3].SetAmbientIntensity(0.0f);
502     // spotLights[3].SetDiffuseIntensity(0.0f);
503     spotLights[3].SetFlash(tailPos, backwardDir);
504 }
505
506

```

Vamos a manejar el encendido y apagado de nuestras luces delanteras y traseras por medio de banderas que dependen de la dirección a la que se mueve el carro, es decir que el carro enciende la luz trasera cuando va de reversa y enciende la delantera cuando el carro avanza hacia adelante.

Ejecución:



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

Tuve demasiados problemas con las rotaciones de las spotlights pero es porque me lo auto compliqué al trabajar con variables en lugar de definir las propiedades de cada elemento desde el main y el loop. Aun así, considero que anclarlos a un pivote del cofre fue una técnica fácil de implementar y que depende del movimiento del objeto principal.

3.- Conclusión:

En esta practica logre trabajar nuevamente con las luces que fue relativamente fácil, el problema es al momento de asignar banderas y crear eventos a partir de un botón, que fue donde más problemas tuve, aun así considero que es un buen acercamiento a la iluminación dentro de OpenGL.

Bibliografía en formato APA

- de la Computación e Inteligencia Artificial, C. (n.d.). *Color e iluminación*. Uhu.Es. Retrieved October 24, 2025, from https://www.uhu.es/francisco.moreno/gii_rv/docs/Tema_6.pdf