Reporte de Práctica de Laboratorio de Computación Gráfica e Interacción Humano-Computadora

Información de la práctica

Nombre del alumno:	Martínez Pérez Brian Erik		
Número de Cuenta:	319049792	Práctica número:	8
Título de la práctica:	Materiales e Iluminación II en OpenGL		
Fecha de entrega:	27 de octubre de 2025		

Introducción: (descripción inicial sobre la práctica)

Esta práctica tiene la intención de generar la creación de escenas 3D realistas a través de la Iluminación Fotorrealista. El objetivo principal es simular el sombreado básico para simular la interacción compleja de la luz con los objetos. Para cumplir con esto se toman en cuentas los temas de la practica anterior, el Modelo de Iluminación de Phong, que se logra mediante la suma de los componentes ambiental, difuso y especular. Estos componentes se toman en cuenta para la generación de 3 distintas fuentes de iluminación, que son direccional, puntual y focal (spotlight), cada uno de ellos produce un efecto distinto en la iluminación y las sombras que producen sus objetos. Es por eso por lo que en esta práctica se implementan los 3 tipos de iluminación y se visualizan sus efectos en distintos modelos 3d.

Resumen Teórico: (introducción sobre los fundamentos teóricos en los que se basa la práctica y cuáles son los objetivos teóricos que persigue.)

Modelo de Iluminación de Phong es un modelo que descompone la luz en tres componentes: Luz Ambiental (luz de fondo), Luz Difusa (color principal) y Luz Especular (brillo/reflejo).

El componente ambiental representa la luz indirecta o de fondo que ha sido dispersada por el entorno y que, por lo tanto, ilumina todas las superficies de manera uniforme.

El componente difuso representa la luz que se dispersa por igual en todas las direcciones después de golpear una superficie mate. Esta es la parte que le da al objeto su color base.

El componente especular representa el brillo o reflejo concentrado que se ve en las superficies pulidas.

La luz “direccional” simula una fuente de luz que se encuentra infinitamente lejos de la escena, como el sol.

La luz “puntual” simula una fuente de luz que irradia en todas las direcciones desde un único punto en el espacio, como una bombilla sin pantalla.

La luz focal, o “spotlight”, simula una fuente de luz que emite luz en una dirección específica dentro de un cono limitado, como una linterna o un faro de coche.

Objetivos:

- Entender la diferencia entre las distintas fuentes de iluminacion y comprender sus caracteristicas.
- Implementar las fuentes de ilumincacion en open gl.
- Aplicar las fuentes de iluminacion y comprender sus componentes aplicados en un modelo cargado 3d.
- Visualizar los efectos de las fuentes de iluminacion sobre un modelo con transparencia.

Reporte de Práctica de Laboratorio de Computación Gráfica e Interacción Humano-Computadora

Descripción de la práctica: (En esta sección se deben describir todos los pasos ejecutados durante la sesión práctica realizada en el laboratorio.)

Partiendo del proyecto de la práctica anterior, lo primero que realizamos fue el cargado de los archivos necesarios para poder aplicar las distintas fuentes de luz a los modelos que utilicemos, mantenemos los mismos archivos “lighting.frag”, “lighting.vs, ”lamp.frag“, “lamp.vs”, “modelLoading.frag” y “modelLoading.vs” en la sección de archivos de origen que se utilizaran como shaders. Además de agregar el código principal “fuentesDeLuz.cpp” en la sección de Archivos de origen.

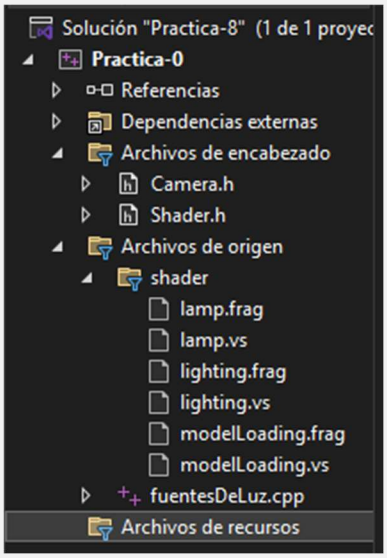

Imagen 1.1 – organización de los archivos utilizados en la solución del proyecto.

Para el código principal tenemos definidas las posiciones de las luces puntuales, en este caso inican desde el origen.

```
// Positions of the point lights
glm::vec3 pointLightPositions[] = {
    glm::vec3(0.0f,0.0f, 0.0f),
    glm::vec3(0.0f,0.0f, 0.0f),
    glm::vec3(0.0f,0.0f,  0.0f),
    glm::vec3(0.0f,0.0f, 0.0f)
};
```

Imagen 1.2 – código para la definición de la posición de las luces.

Definimos la geometría que tendrá la fuente de luz, en este caso generamos un cubo de color blanco, es por eso que tenemos 6 valores por fila, 3 para la posición y 3 para el color.

```
float vertices[] = {
    -0.5f, -0.5f, -0.5f,  0.0f,  0.0f, -1.0f,
     0.5f, -0.5f, -0.5f,  0.0f,  0.0f, -1.0f,
     0.5f,  0.5f, -0.5f,  0.0f,  0.0f, -1.0f,
     0.5f,  0.5f, -0.5f,  0.0f,  0.0f, -1.0f,
    -0.5f,  0.5f, -0.5f,  0.0f,  0.0f, -1.0f,
    -0.5f, -0.5f, -0.5f,  0.0f,  0.0f, -1.0f,

    -0.5f, -0.5f,  0.5f,  0.0f,  0.0f,  1.0f,
     0.5f, -0.5f,  0.5f,  0.0f,  0.0f,  1.0f,
     0.5f,  0.5f,  0.5f,  0.0f,  0.0f,  1.0f,
     0.5f,  0.5f,  0.5f,  0.0f,  0.0f,  1.0f,
    -0.5f,  0.5f,  0.5f,  0.0f,  0.0f,  1.0f,
    -0.5f, -0.5f,  0.5f,  0.0f,  0.0f,  1.0f,

    -0.5f,  0.5f,  0.5f, -1.0f,  0.0f,  0.0f,
    -0.5f,  0.5f, -0.5f, -1.0f,  0.0f,  0.0f,
    -0.5f, -0.5f, -0.5f, -1.0f,  0.0f,  0.0f,
    -0.5f, -0.5f, -0.5f, -1.0f,  0.0f,  0.0f,
    -0.5f, -0.5f,  0.5f, -1.0f,  0.0f,  0.0f,
    -0.5f,  0.5f,  0.5f, -1.0f,  0.0f,  0.0f,
```

Imagen 1.3 – Código para definir la forma y el color de la fuente de luz.

Configuramos la luz direccional, en esta sección definimos el valor que tendrán la componente ambiental, difusa y especular.

```
 Directional light
Uniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.direction"), -0.2f, -1.0f, -0.3f);
Uniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.ambient"), 0.0f, 0.0f, 0.0f);
Uniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.diffuse"), 0.0f, 0.0f, 0.0f);
Uniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.specular"), 0.0f, 0.0f, 0.0f);
```

Imagen 1.4 – Código para la configuración de la luz direccional.

Para la luz puntual se define de otra forma en la que será dinámica en su movimiento, ya que esta tiene la característica de aplicar rayos de luz solo en cierta dirección, por lo dependiendo el lugar hacia donde se aplique, tendrá una diferente iluminación sobre el objeto sobre el que se le aplique.

```
// Point light 1
glm::vec3 lightColor;
lightColor.x = abs(sin(glfwGetTime() * Light1.x));
lightColor.y = abs(sin(glfwGetTime() * Light1.y));
lightColor.z = sin(glfwGetTime() * Light1.z);


glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[0].position"), pointLightPositions[0].x, pointLightPositions[0].y, pointLightPositions[0].z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[0].ambient"), lightColor.x, lightColor.y, lightColor.z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[0].diffuse"), lightColor.x, lightColor.y, lightColor.z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "pointLights[0].specular"), 1.0f, 1.0f, 0.0f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "pointLights[0].constant"), 1.0f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "pointLights[0].linear"), 0.045f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "pointLights[0].quadratic"), 0.075f);
```

Imagen 1.5 – Código para la configuración y el movimiento de la luz puntual.

La ultima fuente de luz será la luz de reflector en ella se define sus componentes, esta tendrá la diferencia que se mostrará junto con el movimiento de la cámara sintética, lo que en ella se muestra como la simulación en una linterna.

```
SpotLight
Uniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.position"), camera.GetPosition().x, camera.GetPosition().y, camera.GetPosition().z);
Uniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.direction"), camera.GetFront().x, camera.GetFront().y, camera.GetFront().z);
Uniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.ambient"), 0.0f, 0.0f, 0.0f);
Uniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.diffuse"), 0.0f, 0.0f, 0.0f);
Uniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.specular"), 0.0f, 0.0f, 0.0f);
Uniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.constant"), 1.0f);
Uniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.linear"), 0.0f);
Uniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.quadratic"), 0.0f);
Uniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.cutOff"), glm::cos(glm::radians(0.0f)));
Uniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.outerCutOff"), glm::cos(glm::radians(0.0f)));
```

Imagen 1.6 - Código para la configuración y el movimiento de la luz de reflector.

En la imagen 1.7 se muestra la instrucción que muestra la propiedad de los materiales llamada “shininess”, esta propiedad muestra las zonas de iluminación blancas al exponerse cerca de una fuente de iluminación.

```
// Set material properties
glUniform1f(glGetUniformLocation(lightingShader.Program, "material.shininess"), 16.0f);
```

Imagen 1.7 – código para agregar la propiedad “shininess” a los materiales.

Para visualizar como afectan la luz a los modelos, se cargan 2 modelos para observar sus efectos, en este caso se usan el modelo del perro rojo y una superficie de pasto.

```
//Carga de modelo
view = camera.GetViewMatrix();
model = glm::mat4(1);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
Piso.Draw(lightingShader);
```

Imagen 1.8 – codigo para dibujar el modelo del pasto.

```
model = glm::mat4(1);
//glEnable(GL_BLEND);//Avtiva la funcionalidad para trabajar el canal alfa
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
glUniform1i(glGetUniformLocation(lightingShader.Program, "transparency"), 0);
Dog.Draw(lightingShader);
//glDisable(GL_BLEND);  //Desactiva el canal alfa
glBindVertexArray(0);
```

Imagen 1.9 – código para el modelo del perro rojo.

Al ejecutar el código tenemos un ambiente poco iluminado ya que la única fuente de luz con parámetros reales modificados para iluminar en la luz puntual.

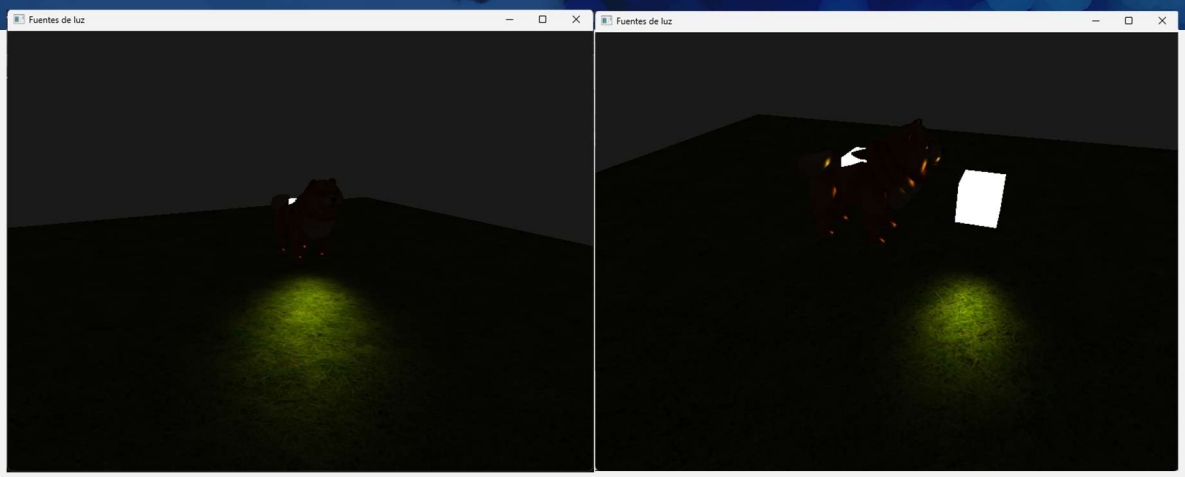

Imagen 2.1 – Primer ejecución del código.

Para poder observar el efecto de la luz direccional, asignamos los siguientes valores a sus propiedades. Y en la ejecución podemos ver como este tipo de luz, ilumina todo su entorno, ya que sus rayos se concentran en sola dirección, pero se esparce, provocando que con los valores que dimos se ilumine todo el ambiente.

```
glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.direction"), -0.2f, -1.0f, -0.3f);
glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.ambient"), 0.5f, 0.5f, 0.5f);
glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.diffuse"), 0.6f, 0.6f, 0.6f);
glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.specular"), 0.3f, 0.3f, 0.3f);
```

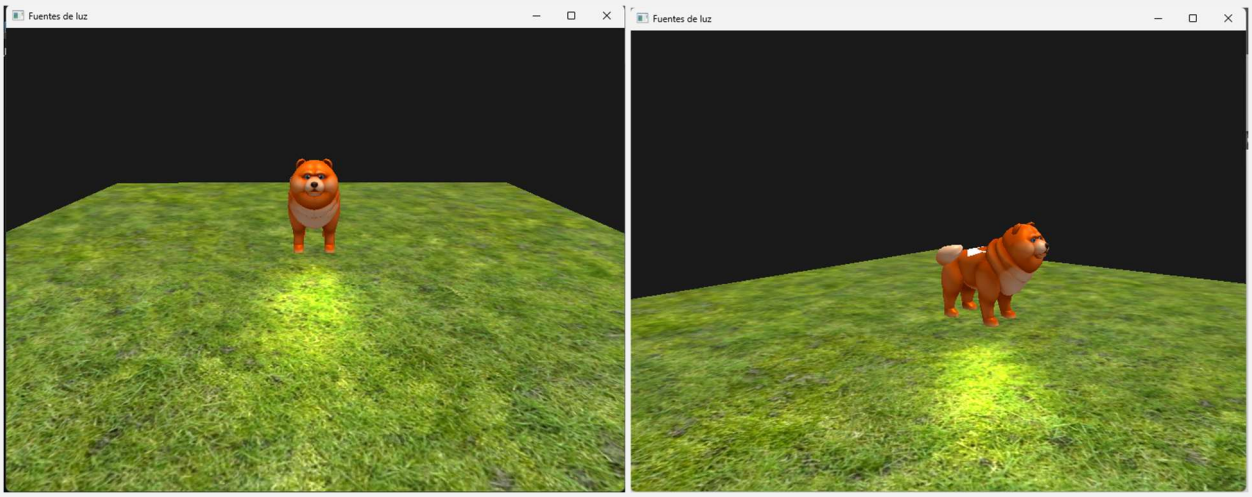

Imagen 2.2 – efectos de la fuente de luz direccional.

La siguiente fuente de luz que probamos fue la puntual, esta tiene un aspecto de un foco, ya que los rayos de luz que produce se dirigen en todas las direcciones, pero no se dispersan e iluminan todo el ambiente, solo lo que se encuentre dentro de su ángulo de iluminación. Además, la probamos en distintos ángulos para poder observar como solo ilumina lo que esta dentro de su rango.

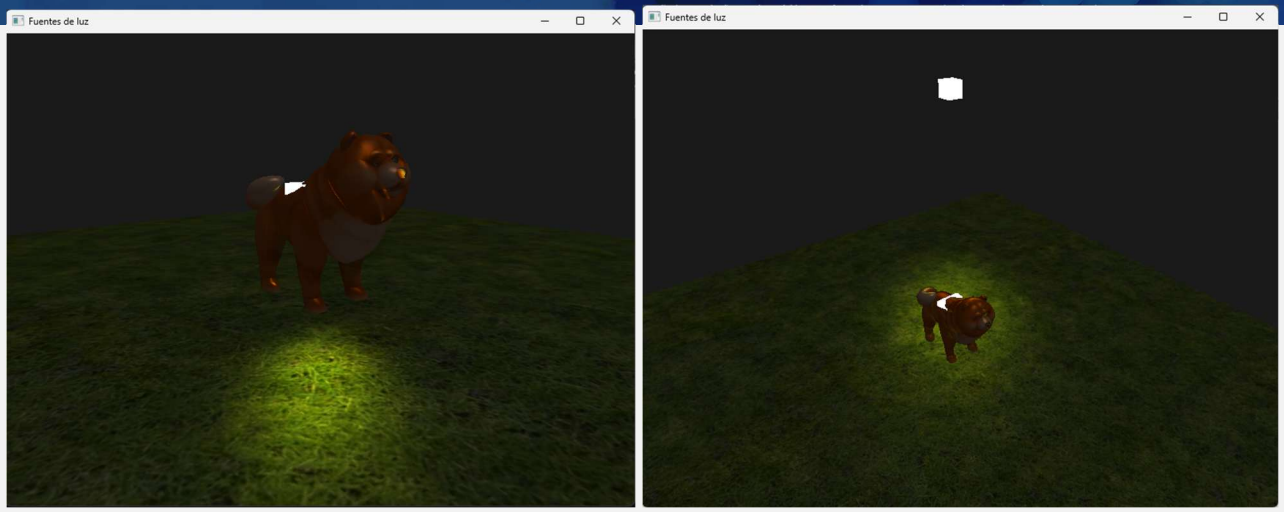

Imagen 2.3 - efectos de la fuente de luz puntual.

Para la ultima fuente de luz, la luz focal, simula la iluminación de una linterna, este tipo de fuente tiene la característica de que sus rayos de luz es que van en una dirección en especifica dentro del rango de un cono. Además, en esta implementación se tomó como prueba, que la linterna fuera dinámica y que se moviera junto con la cámara sintética. Para tener este efecto asignamos los siguientes valores a las propiedades de la luz.

```
glUniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.position"), camera.GetPosition().x,
camera.GetPosition().y, camera.GetPosition().z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.direction"), camera.GetFront().x,
camera.GetFront().y, camera.GetFront().z);
glUniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.ambient"), 0.6f, 0.6f, 0.6f);
glUniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.diffuse"), 0.2f, 0.2f, 0.8f);
glUniform3f(glGetUniformLocation(lightingShader.Program, "spotLight.specular"), 0.2f, 0.2f, 0.2f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.constant"), 1.0f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.linear"), 0.3f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.quadratic"), 0.7f);
glUniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.cutOff"), glm::cos(glm::radians(12.0f)));
glUniform1f(glGetUniformLocation(lightingShader.Program, "spotLight.outerCutOff"), glm::cos(glm::radians(18.0f)));
```

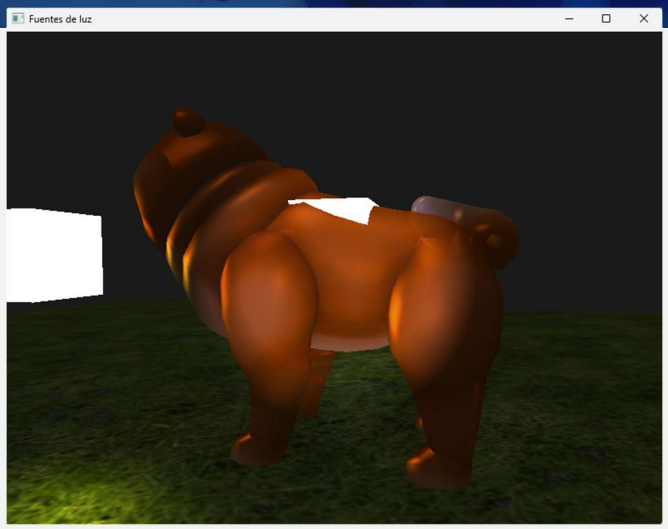

Imagen 2.4 - efectos de la fuente de luz focal.

Después de hacer las pruebas de iluminación para el modelo del perro, procedemos a realizar las mismas pruebas sobre el modelo de una pelota, esta tiene la propiedad que su textura tiene transparencia, así probamos cuales son los efectos de las distintas fuentes de iluminación sobre un modelo con transparencia.

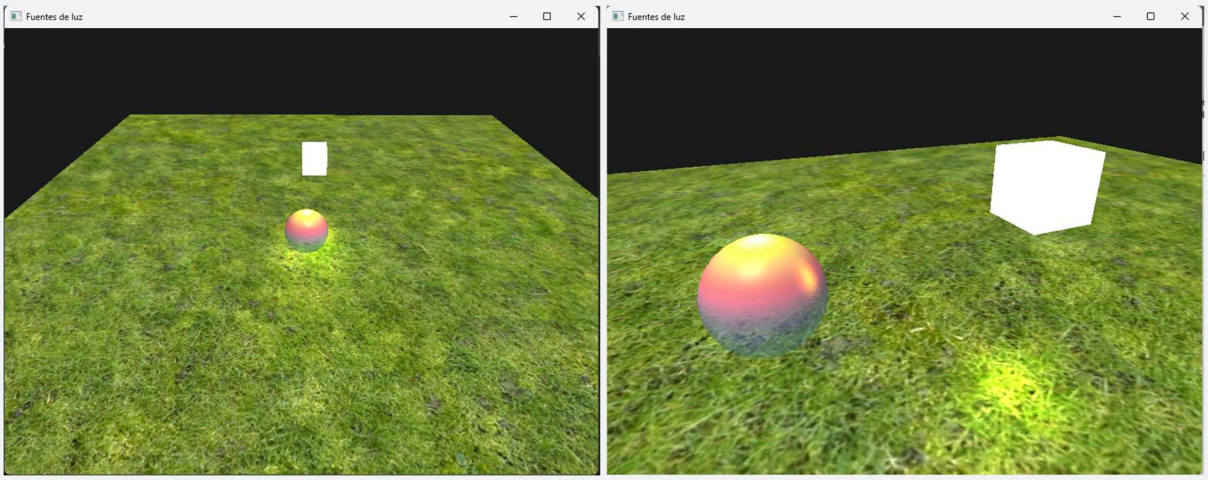

Imagen 2.5 – efectos de la fuente de luz direccional sobre un modelo con transparencia.

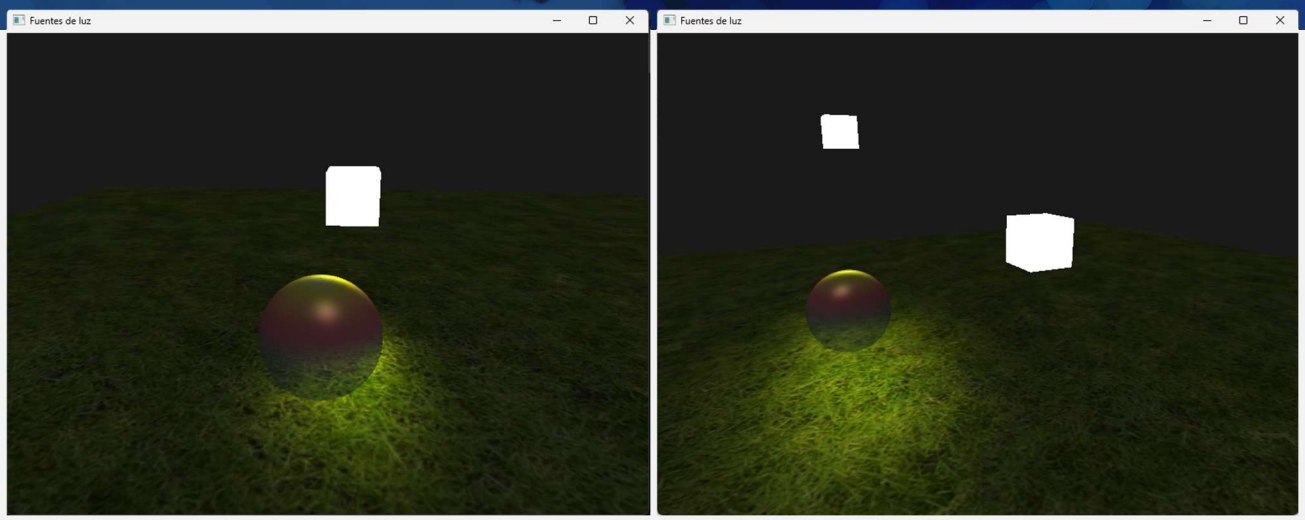

Imagen 2.6 – efectos de la fuente de luz puntual sobre un modelo con transparencia.

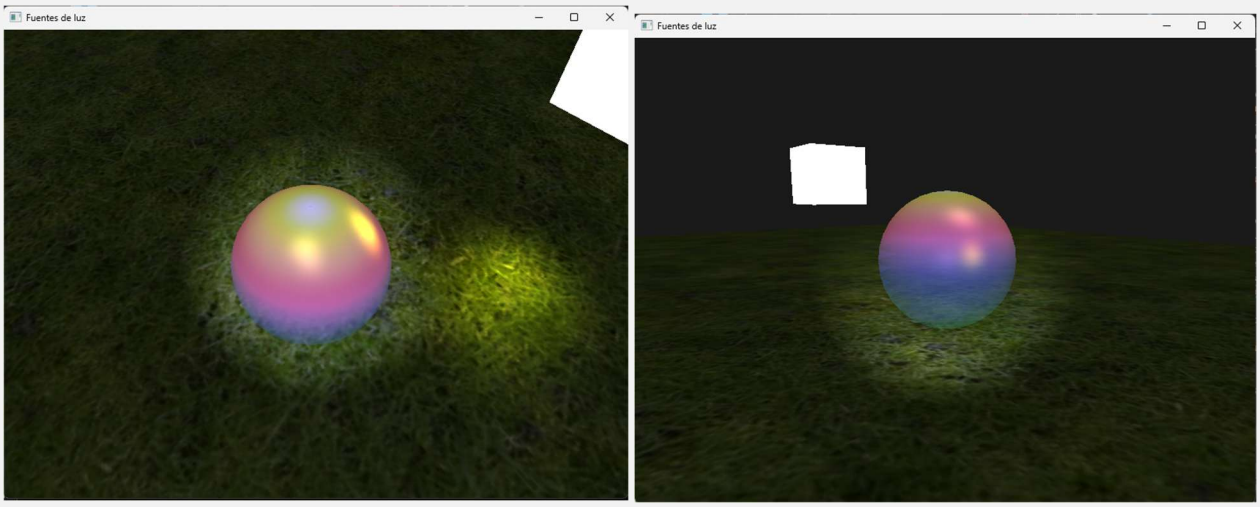

Imagen 2.7 – efectos de la fuente de luz focal sobre un modelo con transparencia.

Realizamos las mismas pruebas de fuentes de iluminación, pero ahora para un modelo cargado que nosotros propusimos para observar sus efectos.

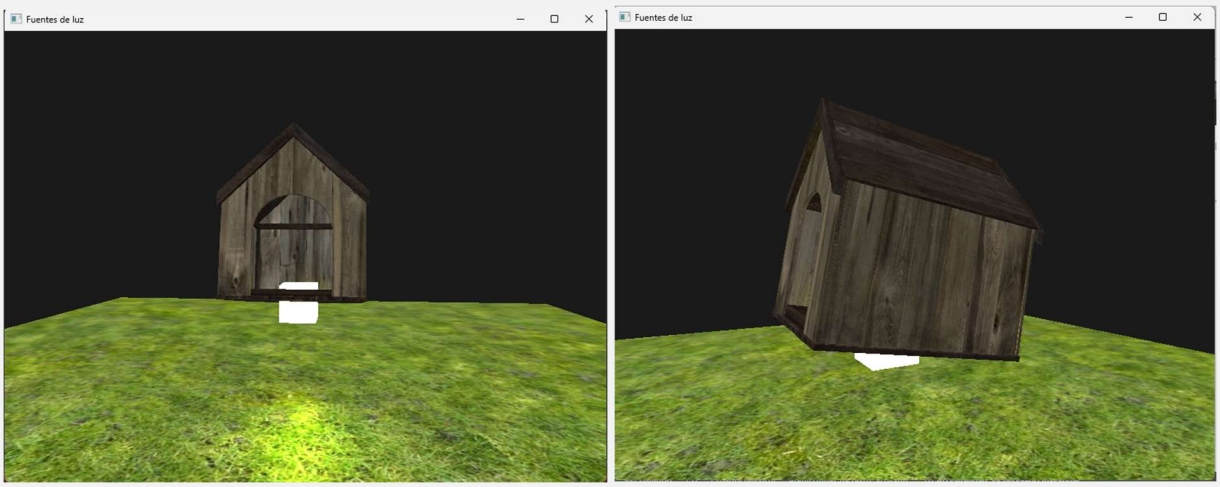

Imagen 2.8 – efectos de la fuente de luz direccional sobre un modelo cargado.

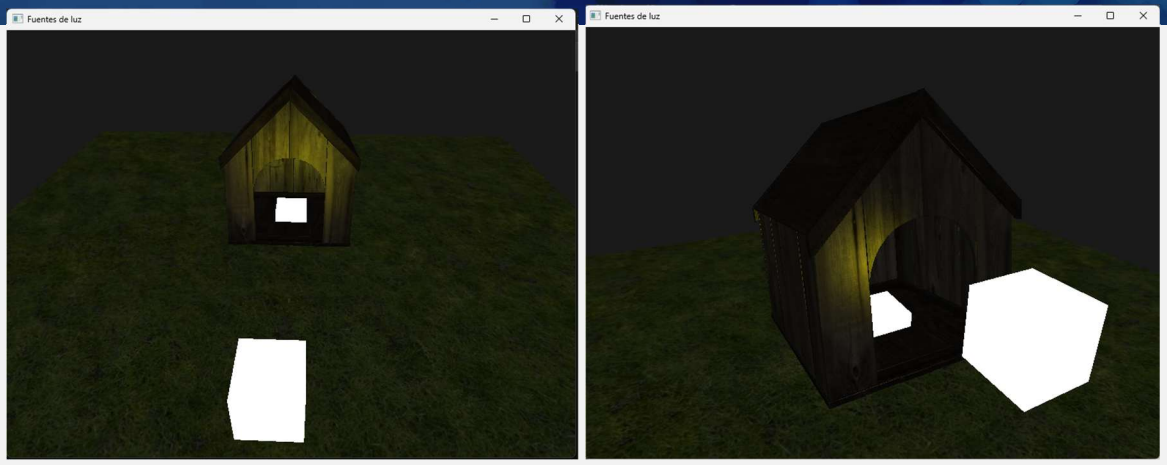

Imagen 2.9 – efectos de la fuente de luz puntual sobre un modelo cargado.

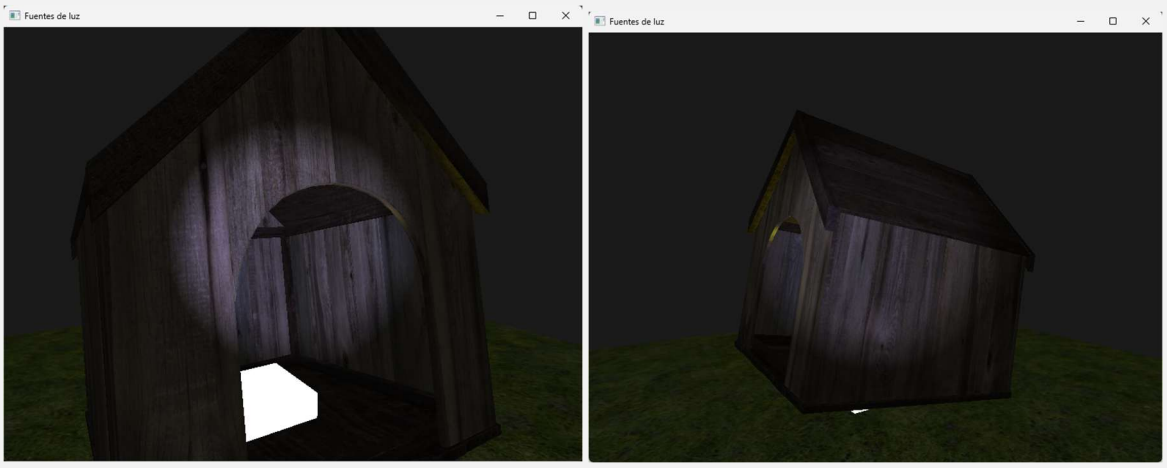

Imagen 2.10 – efectos de la fuente de luz focal sobre un modelo cargado.

Resultados: (explicar lo que se logró o aquello que no lograron realizar o comprender)

- Se logro comprender las diferencias entre los tipos de fuentes de iluminación (direccional, puntual y focal).
- Se logro encontrar los valores correctos para las distintas configuraciones de iluminación en los objetos.
- Logramos implementar cada una de las fuentes de iluminación en open GL.
- Se logro visualizar como afectan las distintas fuentes de iluminación a objetos, además se aplico a distintos modelos, para comprender aun mejor como funciona cada una de ellas.

Conclusiones:

Para esta practica logre comprender y visualizar las distintas fuentes de iluminación, esto me resulto muy importante ya que, en desarrollo de modelos realistas, la iluminación es muy importante, ya que se generan escenarios más realistas, porque no es lo mismo la iluminación en lugar abierto como un parque, a estar en un lugar más oscuro iluminado por una linterna o un foco. Es por eso por lo que me resulto muy útil realizar esta práctica. Por último, pude aplicar las distintas iluminaciones a distintos modelos, ya que el desarrollo de mi proyecto final, que se será el desarrollo de una casa, la iluminación dentro de la casa y en el exterior serán distintas.

Bibliografía consultada:

Interactivas y Computación Gráfica, T. [@ArturoVMS]. (s/f). *Fuentes de Luz en OpenGL: Luz Direccional, Puntual y Reflector [Tutorial Completo]* [[Object Object]]. Youtube. Recuperado el 26 de octubre de 2025, de https://www.youtube.com/watch?v=NaJQbtOTxe4

Dog House Free - Download Free 3D model by donnichols. (2020, septiembre 12). https://sketchfab.com/3dmodels/dog-house-free-fc9e3897b3564f36be62748aaf46adb5