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:	5
Título de la práctica:	Adaptación y carga de modelos		
Fecha de entrega:	29 de septiembre de 2025		

Introducción: (descripción inicial sobre la práctica)

Esta práctica se centra en la integración de modelos tridimensionales que se cargan forma externa, es decir archivos ajenos a los que realizamos nosotros como usuarios. El uso de modelos es un muy útil, ya que, si tenemos la necesidad de utilizar un modelo que no tenemos y que debemos utilizar, podemos utilizar modelos que otra persona ya creo. Además de que existen distintas paginas en las que se publican modelos 3D gratuitos que podemos utilizar en nuestros proyectos.

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

El proceso de asignar a cada vértice un conjunto de coordenadas (u,v) que indican qué parte de una imagen se debe dibujarse sobre esa porción del modelo; esto se implementa con librerías adicionales como SOIL2 o stb_image para la carga de imágenes.

La Cámara Sintética y Transformaciones, donde la matriz de Vista (obtenida de la clase Camera) y la matriz de Proyección se utilizan para posicionar la vista del usuario, mientras que la matriz de Modelo se aplica para rotar, escalar y trasladar el objeto cargado.

El código C++ está creado para la aplicación de las transformaciones y la integración de las texturas. Los objetivos teóricos que se persiguen son las transformaciones matriciales permiten la manipulación del objeto y la interacción del usuario en la escena 3D a través de la cámara artificial con distintos modelos cargados.

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

La organización de los archivos “shader.h”, “camera.h”, “modelLoading.frag”, “modelLoading.vs” y “Carga de Modelos.cpp” dentro de la solución quedo de la siguiente forma:

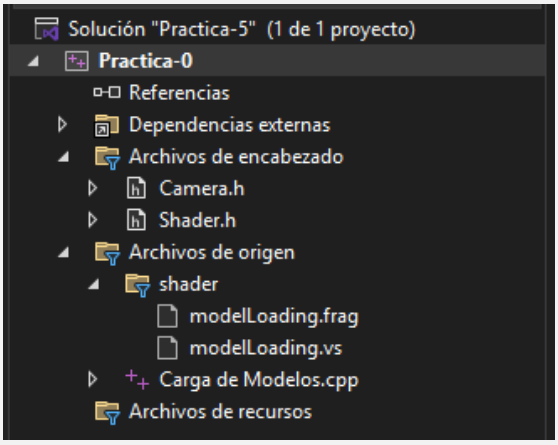


Imagen 1.1 - Ubicación de los archivos utilizados en la práctica.

Camera.h, crea la cámara sintética, es decir simulara una vista en la que el usuario se puede mover dentro del entorno creado. Shader.h se encarga de procesar la información que contiene cada uno de los vértices, como su color y la posición.

modelLoading.frag, su función principal es determinar el color final que tendrá el píxel correspondiente en la pantalla. modelLoading.vs, calcula la posición final de un vértice en el espacio de la pantalla. además, realiza las transformaciones geométricas esenciales.

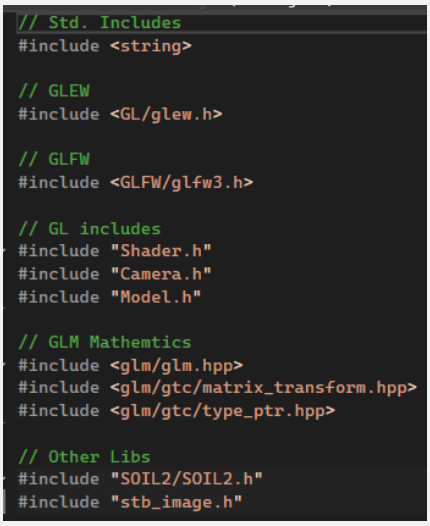
```
#version 330 core
out vec4 FragColor;

in vec2 TexCoords;

uniform sampler2D texture_diffuse1;

void main()
{

  vec4   texColor= texture(texture_diffuse1, TexCoords);
    if(texColor.a < 0.1)
        discard;
    FragColor = texColor;
}
```

Imagen 1.2 – código de modelLoading.

```
1   #version 330 core
2   layout (location = 0) in vec3 aPos;
3   layout (location = 1) in vec3 aNormal;
4   layout (location = 2) in vec2 aTexCoords;
5
6   out vec2 TexCoords;
7
8   uniform mat4 model;
9   uniform mat4 view;
10  uniform mat4 projection;
11
12  void main()
13  {
14      TexCoords = aTexCoords;
15      gl_Position = projection * view * model * vec4(aPos, 1.0);
16  }
```

Imagen 1.3 – código de modelLoading.

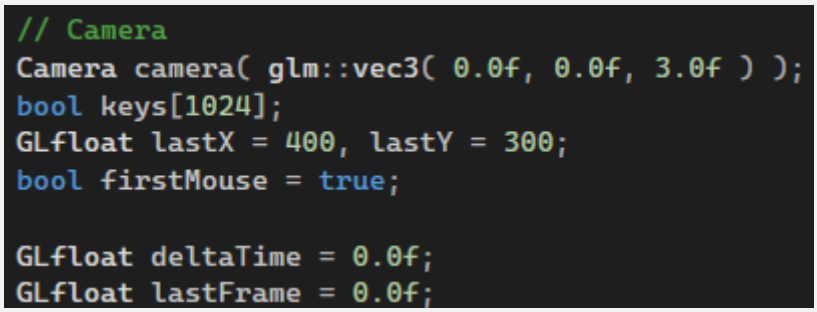
Uso de bibliotecas, GLFW, para ventana y eventos. GLEW para funciones de OpenGL. GLM para funciones matemáticas 3D. SOIL2 para carga de texturas y Model.h: Clase personalizada para cargar modelos 3D



```
// Std. Includes
#include <string>

// GLEW
#include <GL/glew.h>

// GLFW
#include <GLFW/glfw3.h>

// GL includes
#include "Shader.h"
#include "Camera.h"
#include "Model.h"

// GLM Mathemtics
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <glm/gtc/type_ptr.hpp>

// Other Libs
#include "SOIL2/SOIL2.h"
#include "stb_image.h"
```

Imagen 2.1 – código para utilizar las bibliotecas necesarias.

Utilización de la cámara sintética, maneja posición, orientación y matrices de vista.



```
// Camera
Camera camera( glm::vec3( 0.0f, 0.0f, 3.0f ) );
bool keys[1024];
GLfloat lastX = 400, lastY = 300;
bool firstMouse = true;

GLfloat deltaTime = 0.0f;
GLfloat lastFrame = 0.0f;
```

Imagen 2.2 – código para utilizar la cámara sintética.

Una vez que tenemos los códigos cargados, podemos ejecutar el código, nos mostrara una ventana de color gris, ya que no tenemos ningún modelo 3D creado, y mucho menos un modelo cargado en el código.

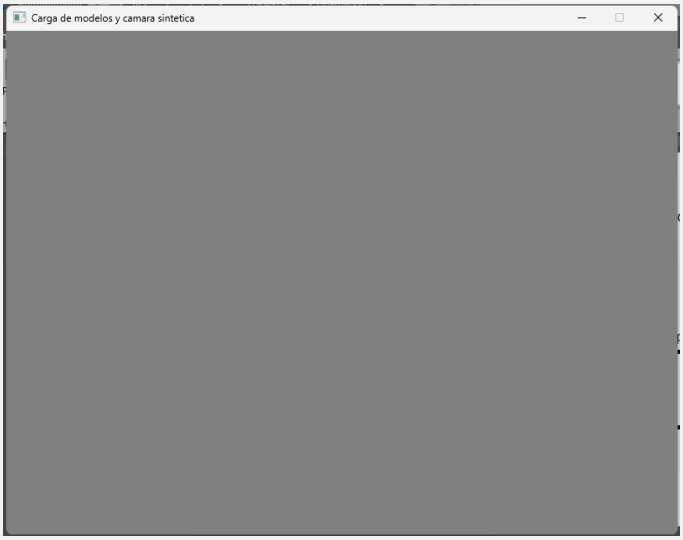
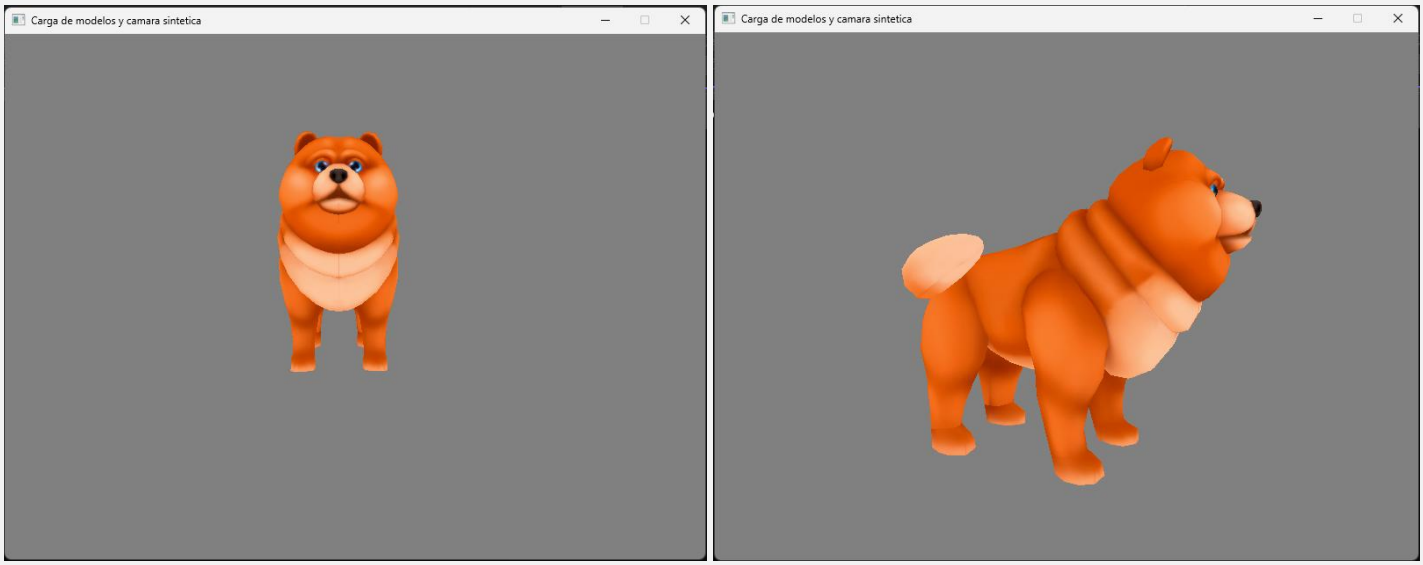


Imagen 2.3 – ventana visualizada en la primera ejecución.

Para cargar el primer modelo agregamos la línea `Model dog((char*)"Models/RedDog.obj");`
En este indicamos que el modelo se llamada “dog” y agregamos la ruta en la que se encuentra el archivo del modelo 3D que vamos a cargar.

Para poder observar ya el modelo utilizamos las operaciones de transformación, como la escala y traslación. Además de las líneas de Código que dibujaran el modelo 3D.

```
glm::mat4 model2(1);
model2 = glm::translate(model2, glm::vec3(0.0f, 0.0f, 0.0f));
model2 = glm::scale(model2, glm::vec3(2.0f, 2.0f, 2.0f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE,
glm::value_ptr(model2));
dog.Draw(shader);
```



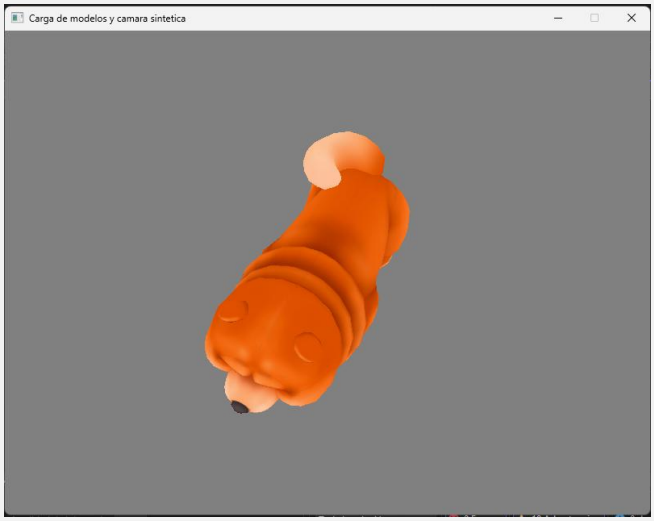
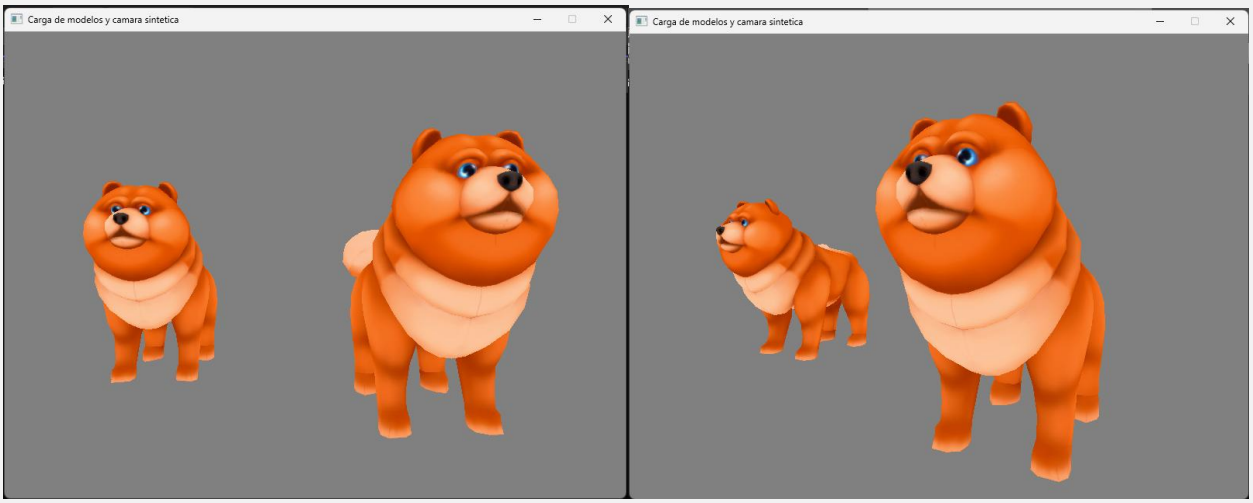


Imagen 2.3 – Visualización del modelo cargado a través de la cámara sintética.

Para agregar el mismo modelo 2 veces realizamos el Código de las operaciones de transformación, para duplicar el perro utilizamos el siguiente Código:

```
glm::mat4 model3(1);
model3 = glm::translate(model3, glm::vec3(2.0f, 0.0f, 0.0f));
model3 = glm::scale(model3, glm::vec3(3.0f, 3.0f, 3.0f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE,
glm::value_ptr(model3));
dog.Draw(shader);
```



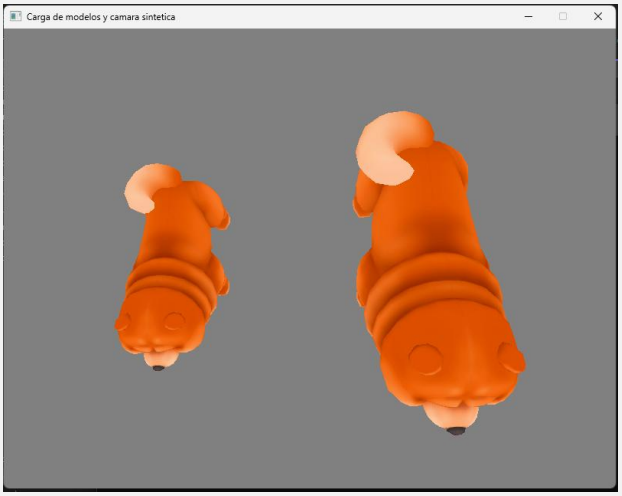


Imagen 2.4 – Visualización del modelo duplicado a través de la cámara sintética.

Por último, cargamos otro modelo 3D, que encontramos en una pagina web, en este caso decidimos descargar una casa de madera para perros.

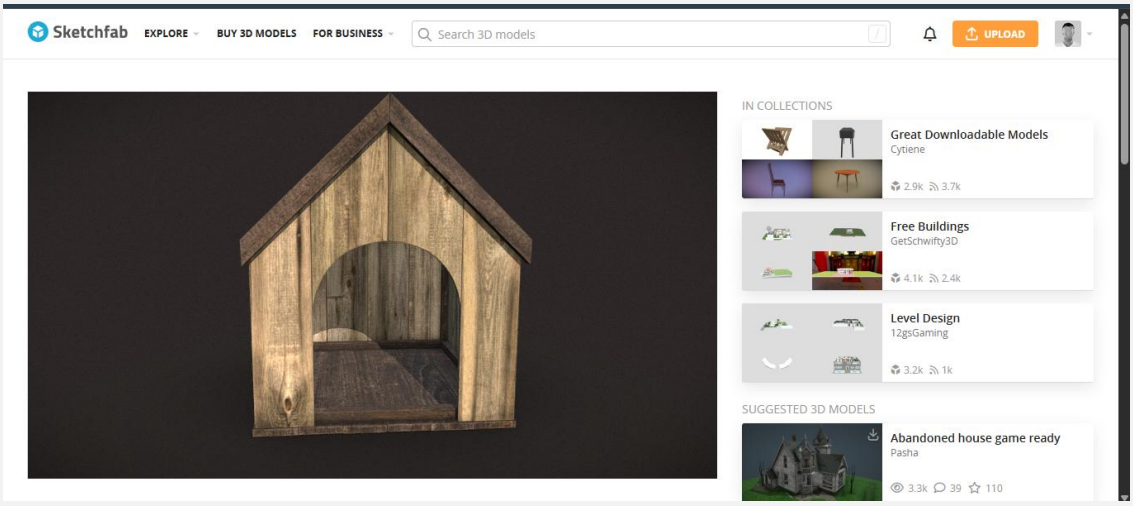


Imagen 3.1 – pagina de donde descargamos el modelo.

Al descargar la carpeta del modelo 3d, nos dio 6 archivos, 1 que es del modelo 3d de tipo ".fbx", y otros 5 de tipo png para las texturas de la casa.

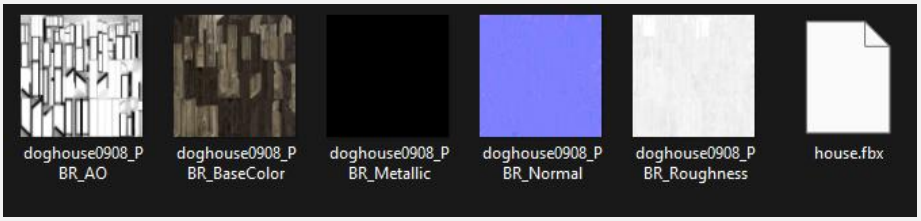


Imagen 3.2 – archivos utilizados para el modelo de la casa cargada.

Para poder ver el nuevo modelo cargado utilizamos la línea de Código:

```
Model house((char*)"Models/house.fbx");
```

Para poder observar ya el modelo utilizamos las operaciones de transformación, como escala, traslación y rotación. La rotación fue utilizada porque, la casa venia cargada al revés, ya que la puerta de entrada estaba abajo. Por último, agregamos las líneas de Código que dibujaran el modelo 3D, el Código utilizado fue el siguiente:

```
glm::mat4 model(1);
model = glm::rotate(model, glm::radians(rotacion), glm::vec3(-100.0f, 0.0f, 0.0f));
model = glm::scale(model, glm::vec3(2.5f, 2.5f, 2.5f));
model = glm::translate(model, glm::vec3(-1.0f, 0.0f, -0.35f));
glUniformMatrix4fv(glGetUniformLocation(shader.Program, "model"), 1, GL_FALSE,
glm::value_ptr(model));
house.Draw(shader);
```

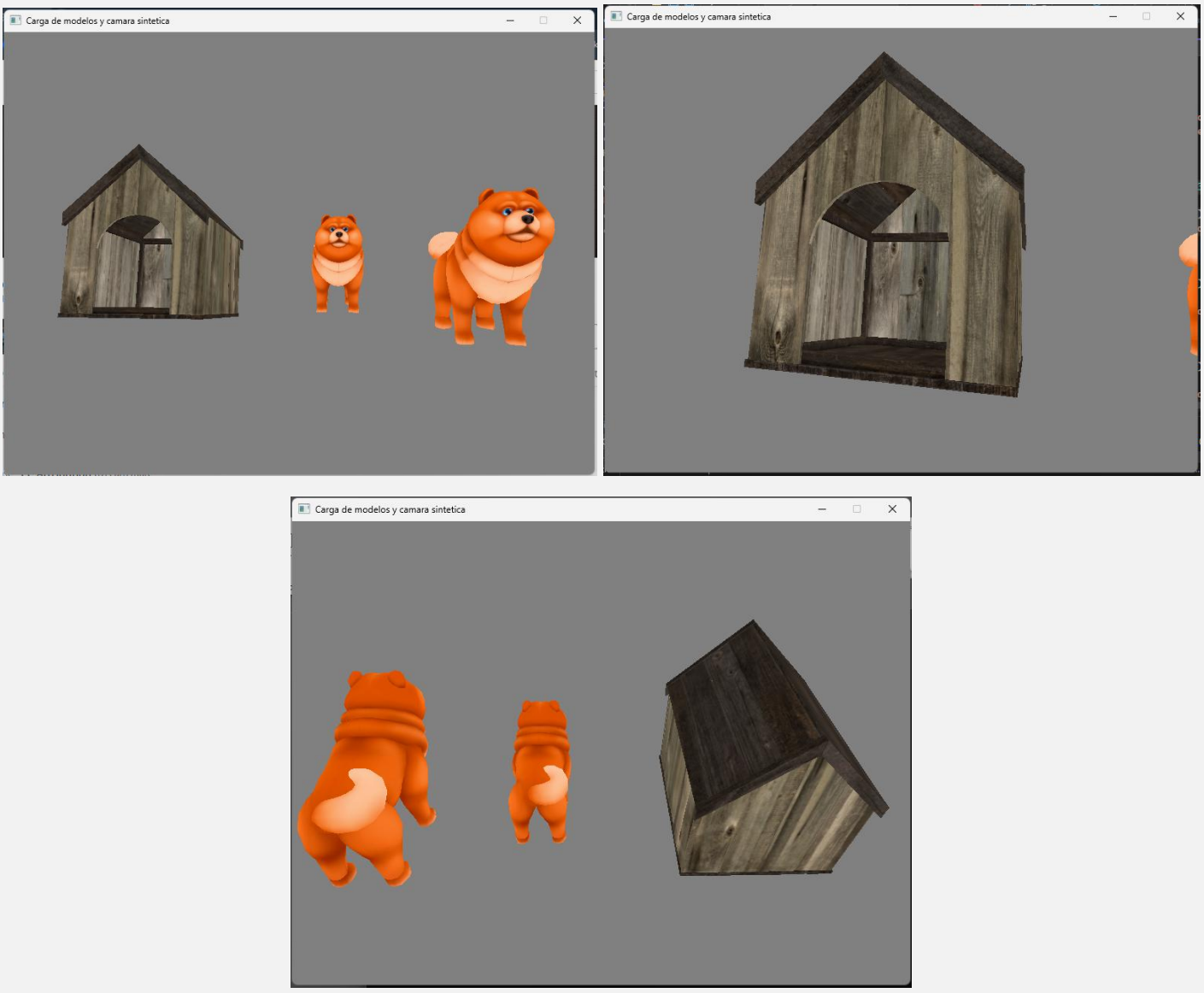


Imagen 3.3 – archivos utilizamos para el modelo de la casa cargada.

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

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

- Se logro entender el concepto de cámara sintética, además de aplicarlo dentro de las ventanas generadas.
- Se logro cargar un modelo 3d, además de poder dibujarlo utilizando las operaciones de transformación.
- Pudimos duplicar el mismo modelo 3d ya precargado en el código.
- Logramos agregar otro modelo 3d dentro de nuestra, adema de comprender los archivos necesarios para poder cargar modelo 3d externos.

Conclusiones:

En esta practica logre poder cargar un modelo 3d, que nos proporcionó el profesor, además de visualizarlo con la cámara sintética. Dentro de lo que no logre poder cargar fue algunos modelos 3d de la página de la que baje la casa del perro. Como primera opción descargue un trofeo, pero al cargarlo como el primer modelo, solo se mostraba de color negro. Al no poder visualizarlo como que quería, cambié el modelo por la casa de madera para perros que mostré en el reporte.

Bibliografía consultada:

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

Interactivas y Computación Gráfica, T. [@ArturoVMS]. (s/f). *Carga de Modelos 3D y Cámara Sintética en OpenGL* [[Object Object]]. Youtube. Recuperado el 28 de septiembre de 2025, de https://www.youtube.com/watch?v=iZu3z2E4qKs