UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

EJERCICIOS DE CLASE Nº 03

NOMBRE COMPLETO: Aguilar Pérez José Ramón

Nº de Cuenta: 317515048

GRUPO DE LABORATORIO: 02

GRUPO DE TEORÍA: 04

SEMESTRE 2025-2

FECHA DE ENTREGA LÍMITE: 25/febrero/2025

CALIFICACIÓN: _____

EJERCICIOS DE SESIÓN:

1. **Actividades realizadas.**

- **Instanciar primitivas geométricas para recrear el dibujo de la práctica pasada en 3D, se requiere que exista un piso; la casa tiene una ventana azul circular justo en medio de la pared trasera, 2 ventanas verdes en cada pared lateral iguales a las de la pared frontal y solo puerta en la pared frontal.**

Partiendo del código proporcionado, se fueron agregando las figuras correspondientes para generar la casa. Primero, con ayuda de un cubo y la función de escalamiento, se generó el piso donde se empezaría a dibujar la casa.

```
//Suelo de la figura
model = glm::mat4(1.0f);
color=glm::vec3(0.4f,0.4f,0.4f);
model = glm::translate(model, glm::vec3(0.0f,0.0f, -3.0f));
model = glm::scale(model, glm::vec3(3.0f, 0.05f, 3.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar e
meshList[0]->RenderMesh();
```

A partir de las coordenadas del suelo, se fueron generando las demás figuras que conforman la casa. Para la estructura general, las ventanas y la puerta se utilizaron cubos. Para el techo se utilizó una pirámide cuadrada. Para los árboles se usaron cilindros para los troncos y conos para el follaje. Finalmente, para la ventana circular de la parte trasera se empleó una esfera. Es importante señalar que estas figuras se realizaron con ayuda de las funcionas *RenderMesh()* y *RendeerMeshGeometry(),* las cuales reciben la figuras de la lista meshList. La esfera es un caso particular, para esta figura se utiliza la función *sp.render()*.

```
//Cubo rojo de la casa
model = glm::mat4(1.0f);
color = glm::vec3(1.0f, 0.0f, 0.0f);
model = glm::translate(model, glm::vec3(0.0f, 0.45f, -3.0f));
model = glm::scale(model, glm::vec3(0.8f, 0.9f, 0.8f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model)); /
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el
meshList[0]->RenderMesh();

//Piramide del techo de la casa
model = glm::mat4(1.0f);
color = glm::vec3(0.0f, 0.0f, 1.0f);
model = glm::translate(model, glm::vec3(0.0f, 1.05f, -3.0f));
model = glm::scale(model, glm::vec3(1.0f, 0.4f, 1.0f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model)); /
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar el
meshList[4]->RenderMesh();
```

```
//Cubos de las ventanas y puerta verde de la casa
//Ventana derecha
model = glm::mat4(1.0f);
color = glm::vec3(0.0f, 1.0f, 0.0f);
model = glm::translate(model, glm::vec3(0.18f, 0.63f, -2.61f));
model = glm::scale(model, glm::vec3(0.25f, 0.25f, 0.05f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar e
meshList[0]->RenderMesh();
//Ventana izquierda
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(-0.18f, 0.63f, -2.61f));
model = glm::scale(model, glm::vec3(0.25f, 0.25f, 0.05f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
//glUniform3fv(uniformColor, 1, glm::value_ptr(color));  //para cambia
meshList[0]->RenderMesh();
//Puerta
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(0.0f, 0.141f, -2.61f));
model = glm::scale(model, glm::vec3(0.25f, 0.25f, 0.05f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
//glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambia
meshList[0]->RenderMesh();
```

```
//Cilindros para los troncos
//Tronco derecho
model = glm::mat4(1.0f);
color = glm::vec3(0.478f, 0.255f, 0.067f);
model = glm::translate(model, glm::vec3(0.8f, 0.155f, -3.0f));
model = glm::scale(model, glm::vec3(0.10f, 0.25f, 0.10f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar e
meshList[2]->RenderMeshGeometry();
//Tronco izquierdo
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(-0.8f, 0.155f, -3.0f));
model = glm::scale(model, glm::vec3(0.10f, 0.25f, 0.10f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
//glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar
meshList[2]->RenderMeshGeometry();

//Conos para los arbolitos
//Arbol derecho
model = glm::mat4(1.0f);
color = glm::vec3(0.0f, 0.5f, 0.0f);
model = glm::translate(model, glm::vec3(0.8f, 0.46f, -3.0f));
model = glm::scale(model, glm::vec3(0.10f, 0.4f, 0.10f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color));  //para cambiar
meshList[3]->RenderMeshGeometry();
//Arbol izquierdo
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(-0.8f, 0.46f, -3.0f));
model = glm::scale(model, glm::vec3(0.10f, 0.4f, 0.10f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
meshList[3]->RenderMeshGeometry();
```

```
//Lado derecho de la casa
//Ventana derecha
model = glm::mat4(1.0f);
color = glm::vec3(0.0f, 1.0f, 0.0f);
model = glm::translate(model, glm::vec3(0.39f, 0.63f, -3.2f));
model = glm::scale(model, glm::vec3(0.05f, 0.25f, 0.25f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar e
meshList[0]->RenderMesh();
//Ventana izquierda
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(0.39f, 0.63f, -2.8f));
model = glm::scale(model, glm::vec3(0.05f, 0.25f, 0.25f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
meshList[0]->RenderMesh();

//Lado izquierdo de la casa
//Ventana derecha
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(-0.39f, 0.63f, -3.2f));
model = glm::scale(model, glm::vec3(0.05f, 0.25f, 0.25f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
meshList[0]->RenderMesh();
//Ventana izquierda
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(-0.39f, 0.63f, -2.8f));
model = glm::scale(model, glm::vec3(0.05f, 0.25f, 0.25f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
meshList[0]->RenderMesh();
```

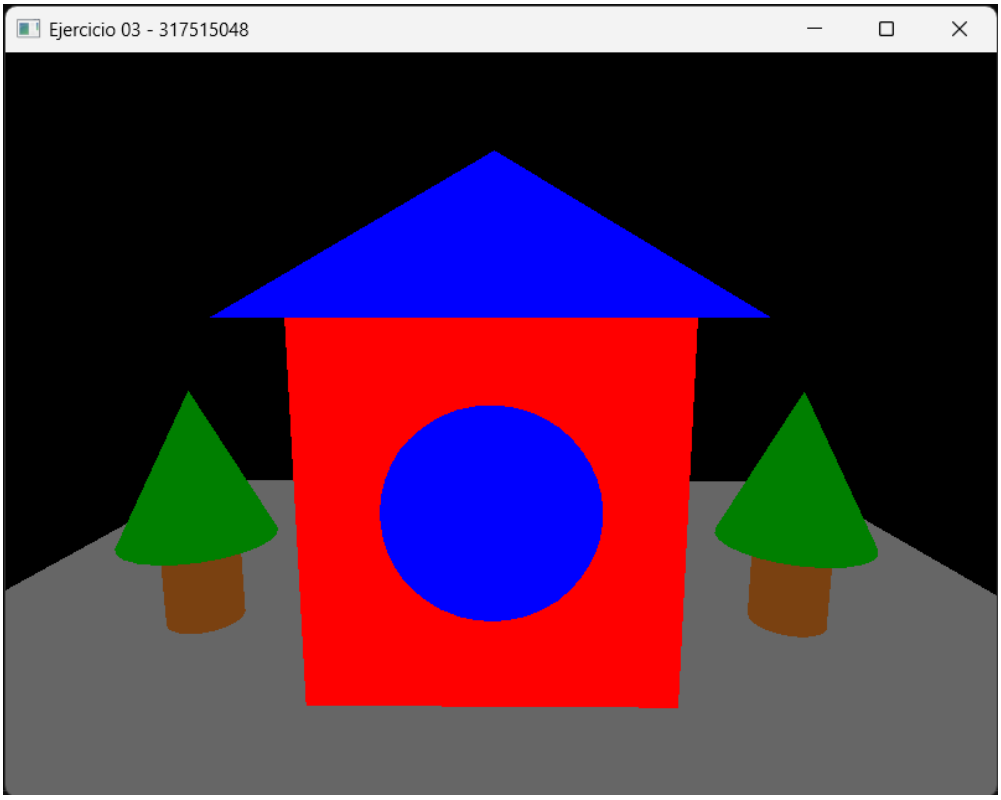
```
//Lado trasero de la cada
//Ventana circular (?)
model = glm::mat4(1.0f);
color = glm::vec3(0.0f, 0.0f, 1.0f);
model = glm::translate(model, glm::vec3(0.0f, 0.45f, -3.38f));
model = glm::scale(model, glm::vec3(0.25f, 0.25f, 0.05f));
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
glUniform3fv(uniformColor, 1, glm::value_ptr(color)); //para cambiar e
sp.render();
```

Por último, para cambiar los colores entre elementos de la casa se modificó la variable *color=glm::vec3(R,G,B)* y se añadió la función *glUniform3fv()* para así aplicar el color que fuese necesario.

Frente de la casa.

Lado izquierdo de la casa.

Lado derecho de la casa.



Lado trasero de la casa.

2. Problemas presentados.

No se presentaron problemas a la hora de realizar el ejercicio.

3. Conclusión

a. Los ejercicios de la clase: Complejidad, explicación

Los ejercicios solicitados tuvieron una complejidad razonable, pero se llevaron a cabo sin complicaciones gracias a que la explicación sobre la generación de las figuras necesarias fue clara y precisa. Además, fue de gran utilidad la explicación sobre el uso de *Uniform* para cambiar el color entre figuras.

b. Comentarios generales: Faltó explicar a detalle, ir más lento en alguna explicación, otros comentarios y sugerencias.

La explicación general del funcionamiento del programa me pareció buena, ya que fue preciso y claro sobre cómo se generan las figuras a través de cálculos de triángulos. Lo más difícil de la practica fue agarrarle la onda al funcionamiento de la cámara, ya que el espacio de trabajo del mouse es pequeño, por lo que fue necesario tener paciencia para adaptarse a este ambiente.