UNIVERSIDAD NACIONAL

AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA E INTERACCIÓN HUMANO COMPUTADORA

PROFESOR: CARLOS ALDAIR ROMAN BALBUENA

SEMESTRE 2025-2

PROYECTO FINAL:

Manual De Usuario y Técnico

ALUMNOS:

VERANO PERALTA MARÍA FERNANDA

GRUPO: 05

FECHA DE ENTREGA: 20/05/2025

ÍNDICE

MANUAL DE USUARIO - PROYECTO FINAL

Este manual te guiará paso a paso para ejecutar y utilizar el programa "Proyecto Final", un entorno 3D basado en OpenGL con iluminación, cámara en tercera persona y modelos 3D. Asegúrate de seguir las instrucciones cuidadosamente para disfrutar de la experiencia.

REQUISITOS PREVIOS

- Sistema operativo: Windows (compatible con las bibliotecas incluidas).
- Archivos necesarios: Asegúrate de tener todos los archivos de la carpeta ejecutable antes de comenzar.

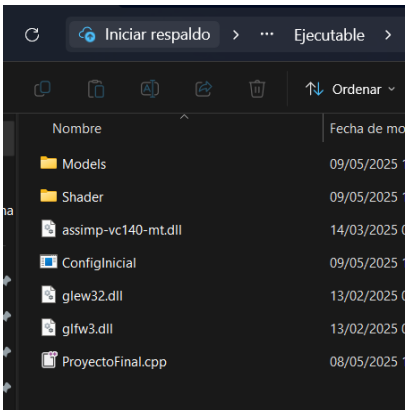


Imagen 1. Archivos necesarios en la carpeta “Ejecutable”

ESTRUCTURA DE LA CARPETA EJECUTABLE

Al abrir la carpeta ejecutable, encontrarás los siguientes elementos:

- Models: Carpeta que contiene los archivos de modelos 3D (por ejemplo, casafinal.obj y snoopy.obj) necesarios para renderizar la casa y el personaje.
- Shader: Carpeta con los archivos de shaders (lighting.vs, lighting.frag, lamp.vs, lamp.frag) que controlan la iluminación y el renderizado.
- assimp-vc140-mt.dll: Biblioteca para cargar modelos 3D (necesaria para el programa).
- ConfigInicial: Archivo de configuración (puede contener ajustes iniciales del programa).
- glew32.dll: Biblioteca para extensiones de OpenGL (requerida para el renderizado).
- glfw3.dll: Biblioteca para la gestión de ventanas y eventos (esencial para la interfaz).
- ProyectoFinal.cpp: Archivo fuente del código (no es necesario ejecutarlo directamente, pero está incluido como referencia).

Nota: No modifiques ni elimines ningún archivo o carpeta, ya que el programa depende de todos ellos para funcionar correctamente.

CÓMO EJECUTAR EL PROGRAMA

1. Abrir la Carpeta Ejecutable:

   Navega hasta la carpeta que contiene los archivos mencionados utilizando el explorador de archivos de Windows.

   Asegúrate de que todos los archivos y carpetas estén presentes.

2. Ejecutar el Programa:

   Busca un archivo ejecutable (.exe) dentro de la carpeta. Si no ves un .exe, significa que necesitas compilar el código ProyectoFinal.cpp con un compilador compatible (como Visual Studio) usando las bibliotecas incluidas (GLEW, GLFW, Assimp, etc.). Para un usuario final, asume que un .exe ya está proporcionado (por ejemplo, ProyectoFinal.exe).

   Haz doble clic en el archivo .exe (por ejemplo, ProyectoFinal.exe) para iniciar el programa.

   Si aparece una advertencia de seguridad de Windows, haz clic en "Ejecutar de todos modos" para continuar.

Imagen 2. Archivo .exe

3. Primer Lanzamiento:

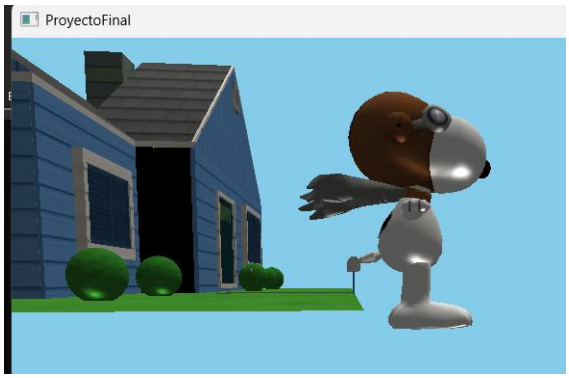
   Al ejecutar el programa, se abrirá una ventana titulada "Fuentes de luz" con dimensiones de 800x600 píxeles.

   Verás una escena 3D con una casa, un personaje (Snoopy) y fuentes de luz representadas como cubos pequeños.

Imagen 3. Escena 3D

CONTROLES DEL PROGRAMA

El programa permite interactuar con la cámara, el personaje y las luces utilizando el teclado y el ratón. A continuación, se detallan los controles:



Movimiento del Personaje

- El personaje (Snoopy) se desplaza en un plano 2D (X, Z) con una altura fija (Y = 0.8).
- **P:** Abre o cierra la puerta del cuarto principal.
- **O:** Abre o cierra la puerta del cuarto de Odin.
- **M:** Abre o cierra la puerta del ropero 1.
- **N:** Abre o cierra la puerta del ropero 2.
- **B:** Abre y cierra todas las ventanas de la casa.
- **C:** Activa o detiene la vibración de la copa.
- **I:** Mueve la manecilla de las horas del reloj (se recomienda presionar junto con la tecla U).
- **U:** Mueve la manecilla de los minutos del reloj (se recomienda presionar junto con la tecla I).
- **R:** Enciende o apaga la televisión.
- **W:** Mueve a Snoopy hacia adelante (aumenta la coordenada Z).
- **S:** Mueve a Snoopy hacia atrás (disminuye la coordenada Z).
- **A:** Mueve a Snoopy hacia la izquierda (disminuye la coordenada X).
- **D:** Mueve a Snoopy hacia la derecha (aumenta la coordenada X).
- **F:** Abre o cierra la puerta del cuarto de Fernanda.
- **V:** Abre o cierra las ventanas.
- **L:** Hace que el pájaro vuele en círculos.
- **(Automático):** El humo de la tetera se muestra de forma constante, sin necesidad de presionar una tecla.

Control de la Cámara (Tercera Persona)

- Ratón: Mueve el ratón para rotar la cámara alrededor del personaje.
- Movimiento horizontal (izquierda/derecha): Gira la cámara en el eje Y (yaw).
- Movimiento vertical (arriba/abajo): Inclina la cámara en el eje X (pitch), con un límite de 89° hacia arriba y 5° hacia abajo para evitar vistas invertidas o demasiado bajas.
- La cámara mantiene una distancia fija de 6 unidades del personaje y se ajusta automáticamente para no bajar por debajo de Y = 1.0.

Control de la Luz Puntual

- T: Mueve la primera luz a la derecha (aumenta X).
- G: Mueve la primera luz a la izquierda (disminuye X).
- F: Eleva la primera luz (aumenta Y).
- H: Baja la primera luz (disminuye Y).
- U: Mueve la primera luz hacia adelante (disminuye Z).
- J: Mueve la primera luz hacia atrás (aumenta Z).
- La luz tiene un color dinámico que pulsa basado en funciones seno, activado/desactivado con la tecla Espacio.

Otras Funciones

- Espacio: Alterna la activación de la primera luz puntual. Cuando está activa, el color cambia (amarillo); cuando está desactivada, se apaga.
- Esc: Cierra el programa y termina la ejecución.

CARACTERÍSTICAS DE LA ESCENA

- Modelos 3D: La escena incluye una casa (casafinal.obj) y un personaje (snoopy.obj) renderizados con iluminación.
- Iluminación:
    - Luz direccional: Simula una luz global (como el sol) con dirección (-0.2, -1.0, -0.3).
    - Luces puntuales: Cuatro luces, de las cuales solo la primera es funcional y animada; las otras están desactivadas.
    - Luz de foco: Sigue la cámara, simulando una linterna con un cono de 12° de apertura.
    - Transparencias: La casa soporta canal alfa para efectos de transparencia.
    - Cámara en tercera persona: La cámara orbita alrededor del personaje, ajustándose dinámicamente.

SOLUCIÓN DE PROBLEMAS

- El programa no se abre:
    - Verifica que todos los archivos .dll (glew32.dll, glfw3.dll, assimp-vc140-mt.dll) estén en la misma carpeta que el .exe.
    - Asegúrate de tener permisos de ejecución en la carpeta.
- La escena no se renderiza correctamente:
    - Confirma que los archivos en las carpetas Models y Shader no hayan sido movidos o eliminados.
    - Reinstala los archivos desde la fuente original si es necesario.
- El movimiento es lento o errático:
    - Cierra otras aplicaciones para liberar recursos del sistema.

CIERRE DEL PROGRAMA

- Presiona la tecla Esc para cerrar la ventana y terminar el programa de manera segura.
- Alternativamente, haz clic en el botón "Cerrar" (X) de la ventana.

NOTAS ADICIONALES

- Este programa está diseñado para fines educativos o de demostración. No modifiques los archivos a menos que tengas conocimientos avanzados de OpenGL y C++.
- Si deseas personalizar la escena (por ejemplo, cambiar modelos o shaders), necesitarás recompilar el código con un entorno de desarrollo como Visual Studio.

¡Disfruta explorando la escena 3D y experimentando con los controles!

MANUAL TÉCNICO - PROYECTO FINAL

DESCRIPCIÓN DEL PROYECTO

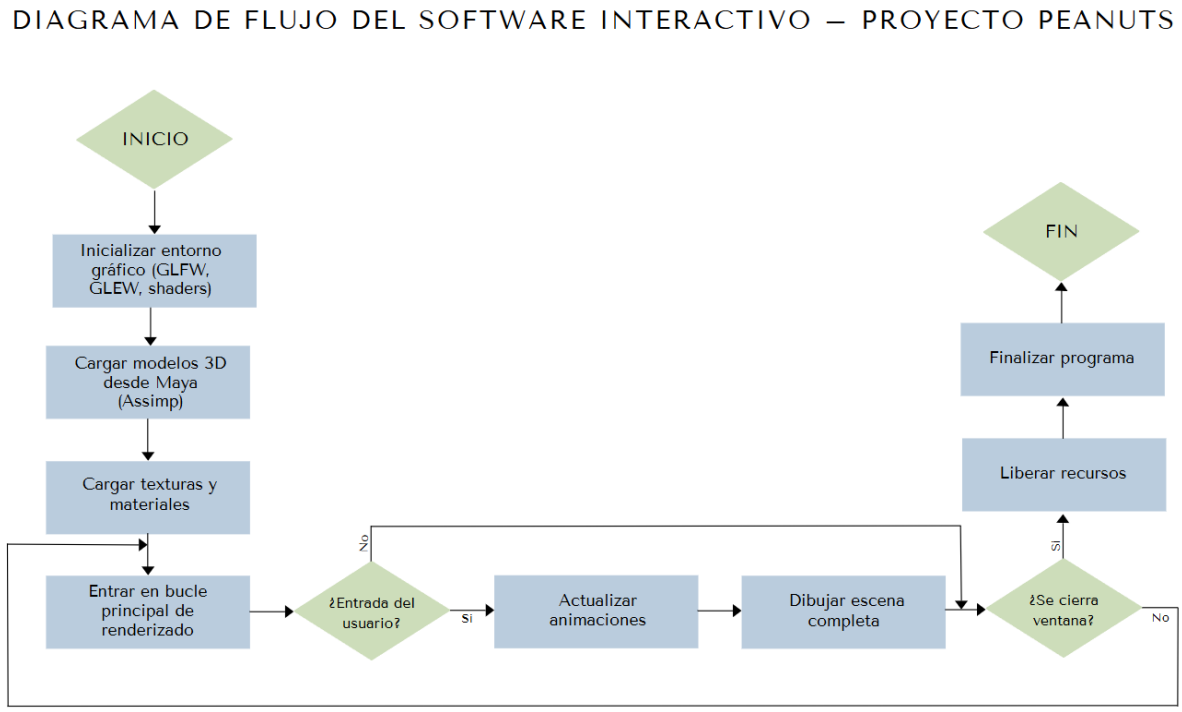
Este proyecto consiste en la recreación tridimensional de una fachada ficticia inspirada en la casa de Charlie Brown, del universo Peanuts. Fue desarrollado en OpenGL con C++ y emplea modelos creados en Autodesk Maya, los cuales fueron exportados e integrados usando la librería Assimp. La escena contiene tres habitaciones distintas, cada una con al menos cinco objetos modelados, texturizados y algunos animados, así como interacciones que le aportan dinamismo y coherencia al entorno.

OBJETIVOS

Los objetivos de este proyecto serán:

- ➔ Modelar e implementar una escena 3D completa y funcional basada en una referencia ficticia.
- ➔ Aplicar herramientas y técnicas de modelado 3D profesional con Autodesk Maya.
- ➔ Programar animaciones e interacciones usando OpenGL y C++.
- ➔ Documentar adecuadamente el desarrollo y uso del software.

DIAGRAMA DE FLUJO DEL SOFTWARE



El diagrama representa el flujo de ejecución de un software interactivo 3D, desde su inicio hasta el cierre. Comienza con la inicialización del entorno gráfico, utilizando herramientas como GLFW, GLEW y shaders, necesarias para crear una ventana de renderizado y controlar gráficos en tiempo real.

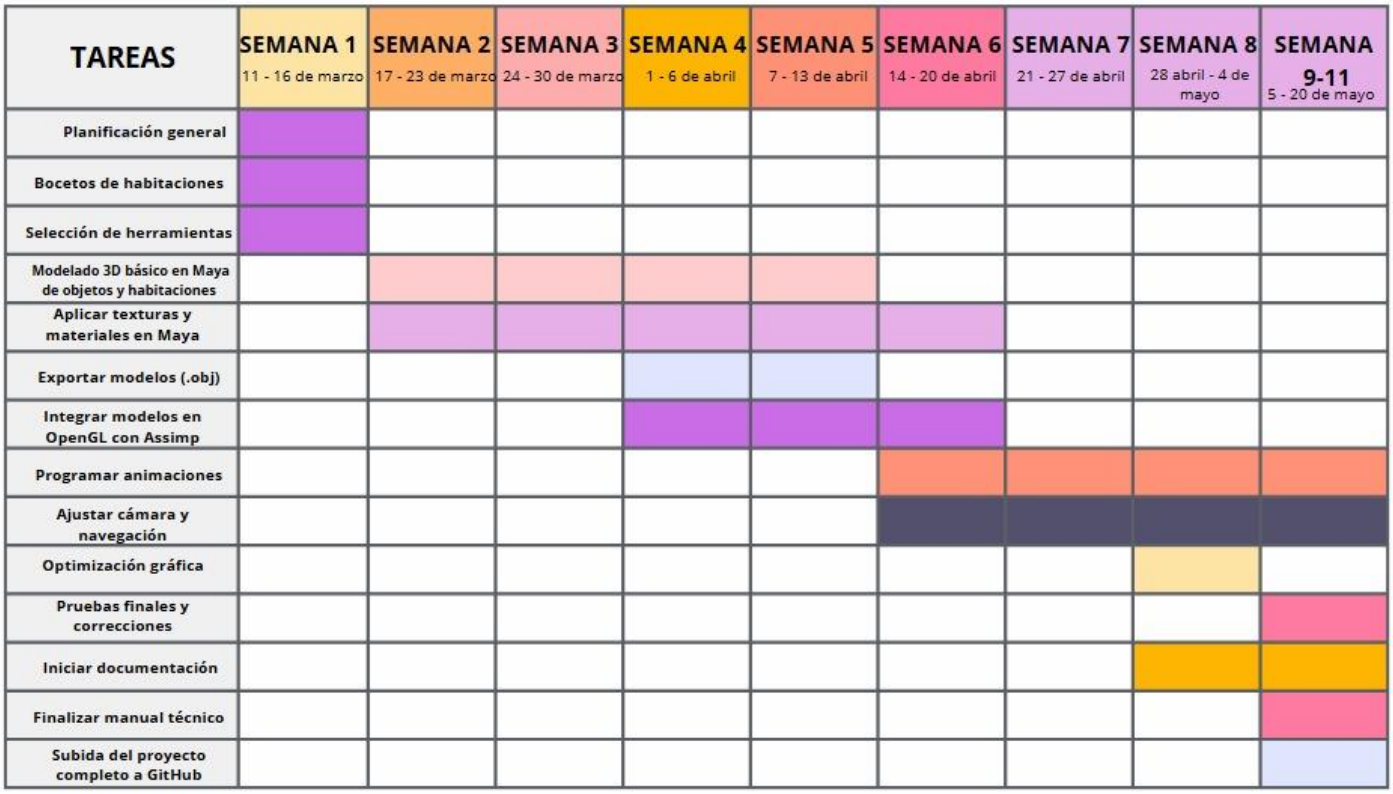
Posteriormente, se cargan los modelos 3D previamente creados en Maya, a través de la librería Assimp, seguida por la carga de texturas y materiales que definen la apariencia visual de dichos modelos.

Una vez completada esta fase de preparación, el programa entra en el bucle principal de renderizado, que se ejecuta continuamente mientras la ventana esté activa. Dentro de este bucle, el sistema verifica si hay entrada del usuario (por ejemplo, teclas, clics o movimientos). Si la hay, se actualizan las animaciones o elementos interactivos en pantalla.

Después de esto, se dibuja la escena completa, es decir, se genera visualmente el entorno con sus modelos, iluminación y animaciones. Al final de cada ciclo, el programa revisa si el usuario ha cerrado la ventana. Si no se ha cerrado, el bucle se repite. Si se cierra, el software libera los recursos utilizados (memoria, texturas, buffers) y finaliza el programa correctamente.

DIAGRAMA DE GANTT

Este cronograma muestra la planificación de tareas clave para el desarrollo de un proyecto de software interactivo en 3D, distribuido en un periodo de nueve semanas, desde el 11 de marzo hasta el 20 de mayo.



ALCANCE DEL PROYECTO

- ➔ Tres habitaciones con ambientaciones distintas.
- ➔ Al menos 15 objetos modelados en Maya.
- ➔ Animaciones implementadas:
  - ◆ Recorrido de Snoopy a través del proyecto
- ➔ Exportación de modelos en formato compatible (.obj).
- ➔ Renderizado e interacción en tiempo real usando OpenGL.
- ➔ 4 animaciones:
  - ◆ Humo de la tetera

- Pájaro que se está estático en uno del arbusto de la entrada de la casa y al presionar “L” da vueltas sobre el mismo eje.
- Abrir y cerrar todas las ventanas
- Abrir y cerrar la puerta de una de las habitaciones.

LIMITANTES

- ➔ Recursos visuales limitados a fines académicos.
- ➔ No había conocimiento previo para utilizar Git y GitHub.
- ➔ Peso restringido de los modelos (< 100 MB cada uno).
- ➔ No se permite recrear escenarios reales (UNAM, empresas).
- ➔ El desarrollo se realizó en plataformas compatibles con Windows.
- ➔ El proyecto puede no correr de forma fluida en computadoras con GPU integrada o sin soporte completo de OpenGL 3.3 o superior.
- ➔ La animación de Snoopy es activada por teclas; no hay detección de colisiones.
- ➔ No se utilizaron simulaciones físicas avanzadas.
- ➔ El proyecto no incluye efectos de sonido o música de fondo.
- ➔ El programa fue desarrollado y probado en Windows. Su ejecución en otros sistemas operativos como macOS o Linux podría requerir ajustes.

METODOLOGÍA DE SOFTWARE APLICADA

Para el desarrollo de este proyecto se aplicó una metodología **incremental y colaborativa**, adaptada a los tiempos establecidos por el calendario académico. Esta metodología permitió construir el proyecto en fases bien definidas, facilitando la integración progresiva de modelos, funcionalidades y animaciones, así como la asignación de tareas entre los dos integrantes del equipo. El proceso de desarrollo se dividió en las siguientes etapas:

Planificación inicial

- ➔ Se definió el alcance, la temática (casa de Charlie Brown) y se identificaron los elementos clave a modelar y animar.

  Durante la fase inicial del proyecto se estableció que el entorno interactivo estaría inspirado en el universo de Peanuts, específicamente en la casa de Charlie Brown. El alcance abarcó desde el modelado de interiores y exteriores básicos hasta la integración de elementos animados dentro de un entorno navegable. Se determinaron los elementos esenciales a incluir, tales como: la cama, la televisión, la mesa, los libros, la maceta, la tetera, el bowl de frutas, el sofá y el personaje de Snoopy, quien sería el foco de animación principal. Se tomó como referencia tanto la estética original de los dibujos como una adaptación en estilo low poly, con un enfoque visual caricaturesco. Se priorizó que cada objeto tuviera un propósito visual o interactivo, y que en conjunto recrearan un espacio coherente, reconocible y funcional para el usuario.

- ➔ Se acordó la distribución de tareas entre los integrantes (modelado, programación, integración y documentación).

  Con base en las habilidades y preferencias de los miembros del equipo, se asignaron responsabilidades específicas para asegurar una ejecución eficiente. El trabajo se dividió en cuatro grandes áreas: modelado, programación, integración y documentación.

Modelado: Odin fue el principal encargado de esta área, desarrollando la mayoría de los objetos 3D en Maya. Esto incluyó el diseño de la estructura interior de la casa, mobiliario, objetos decorativos y la optimización de las mallas para su exportación.

Programación: Fernanda asumió principalmente la responsabilidad del desarrollo del entorno en OpenGL. Se encargó de implementar la carga de modelos con Assimp, la gestión del renderizado, la interacción del usuario y, especialmente, de programar la animación de Snoopy.

Integración: Ambos integrantes colaboraron en esta etapa, colocando los modelos en el entorno interactivo, asegurando que cada objeto tuviera la escala, posición y orientación adecuadas. También se coordinó el ajuste de iluminación, cámara y navegación.

Documentación: Odin y Fernanda compartieron la elaboración del manual técnico, la redacción de la documentación del proyecto, y la producción del video explicativo. También gestionaron la preparación final del repositorio para su entrega y publicación en GitHub.

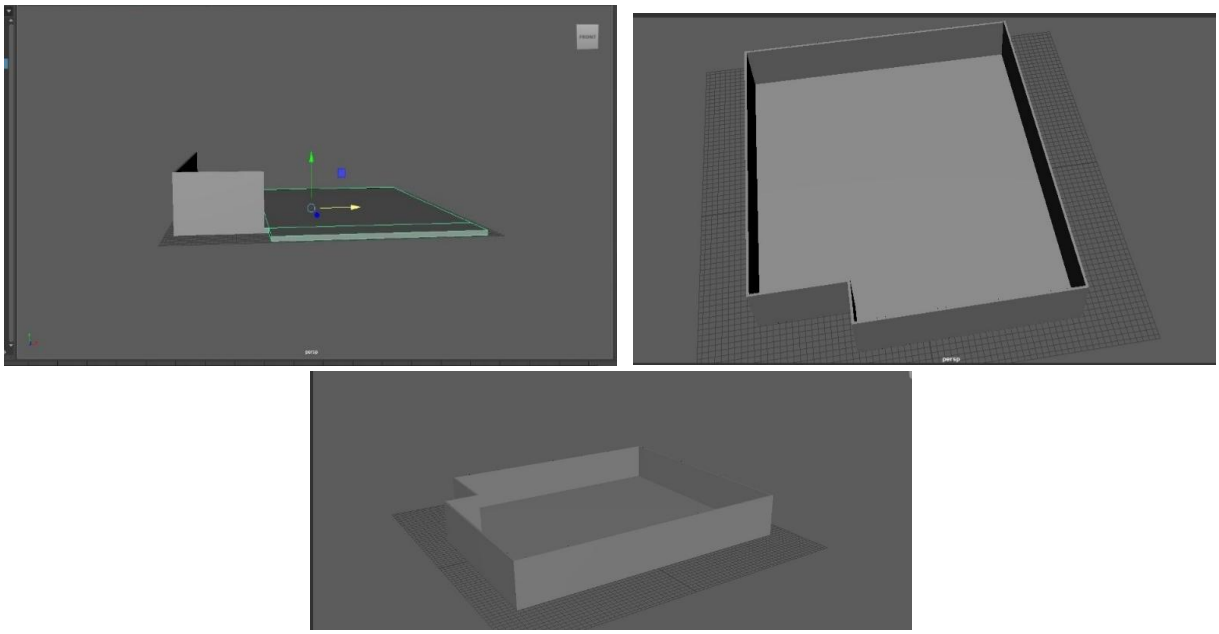
Diseño y modelado 3D

➔ Cada habitación y objeto fue diseñado en Autodesk Maya, siguiendo bocetos previos de referencia.

En este punto mencionaremos cómo fue que creamos cada uno de nuestros objetos comenzando con la casa.

Casa

La casa fue realizada principalmente mediante formar cúbicas que vienen de opción dentro de Maya, en este caso, nuestra casa se conforma por 6 paredes con vistas al exterior que le dan forma a la fachada del proyecto.

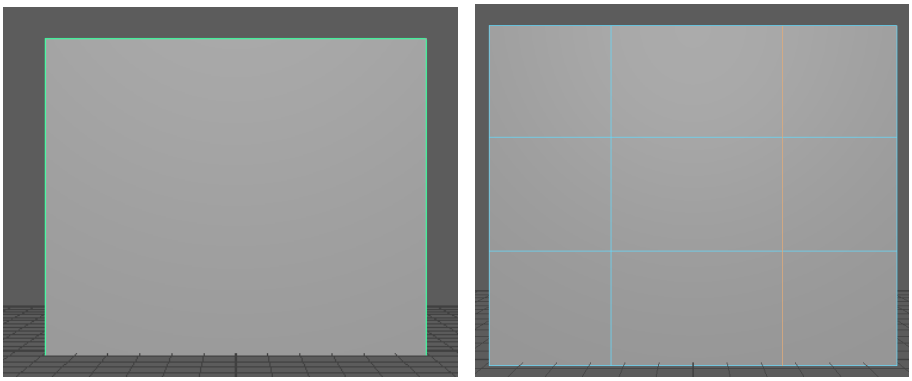


Imágenes iniciales de la estructura de la casa

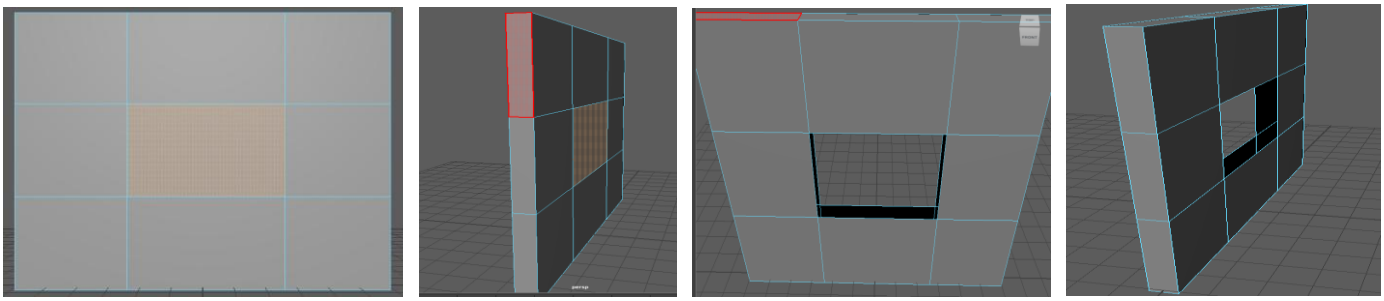
A estas paredes se les aplicaron diversas herramientas para poder crear los marcos de las ventanas y puertas. Los pasos a seguir para poder crear los marcos fueron los siguientes:

1. Seleccionar el objeto en este caso una pared, una vez seleccionado activar mediante el click derecho la opción Edges. Una vez estando en Edges con la tecla shift derecho y el click derecho activar la opción Insert Edge Loop Tool lo cual te permitirá agregar aristas (que después se convertirán en

caras) al cuerpo que observamos, dibujamos los edges de acuerdo con nuestras necesidades ya sea una ventana o una puerta.

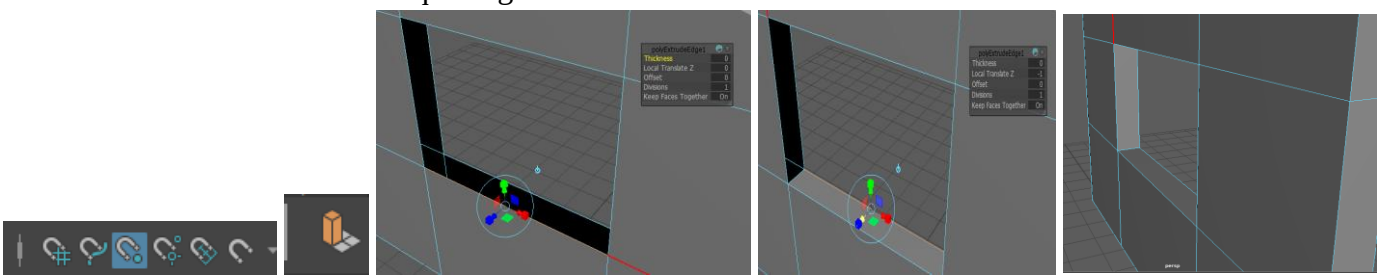


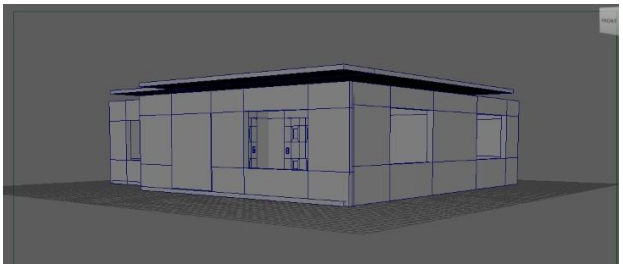
2. Una vez que ya tenemos los nuevos edges dibujados a corde a nuestras necesidades, lo que hicimos fue utilizar la herramienta seleccionar y posteriormente activamos la opción de Face con click derecho sobre el objeto. Seleccionamos las caras que no nos sirvan, en este caso hay que recordar que los cuerpos tienen cierto número de profundidad por lo cual los cuerpos tienen un espacio vacío dentro de ellos y una vez seleccionadas las caras que deseamos eliminar, presionamos la tecla de borrado y veremos ahora un hueco sobre el cuerpo y la parte negra que se ve es el espacio vacío que contiene nuestra pared debido a que el cuerpo tiene profundidad.



Paredes con el hueco de los marcos de la ventana.

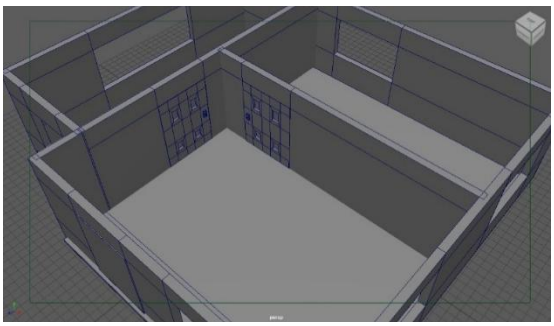
3. Como ya se mencionó, se queda la pared con el hueco y se puede ver el vacío del cuerpo. Para poder arreglar esto, haremos uso de la herramienta Extrude la cual es una herramienta que funciona para crear geometría a partir de una cara, arista o vértice de un objeto 3D existente. En este caso lo haremos a partir de una arista por lo cual entramos en modo Edge y seleccionamos una arista del marco de la ventana. Una vez seleccionada la arista seleccionamos de la barra de herramienta el Extrude y se pondrá un pivote en medio de la arista seleccionada la cual de momento no la moveremos. Seleccionamos de la barra de herramientas Snap To Points la cual sirve para alinear un objeto, vértice, arista o cara directamente sobre el punto (vértice) de otro objeto, facilitando una colocación precisa para que no queden huecos y sea uniforme al objeto general y repetimos esto con todos los marcos que hagamos.



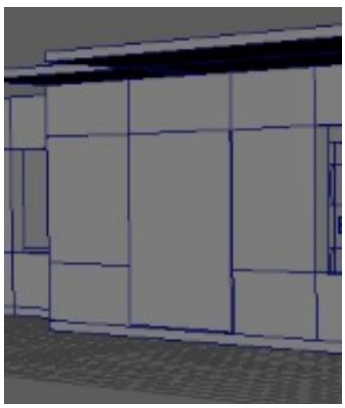
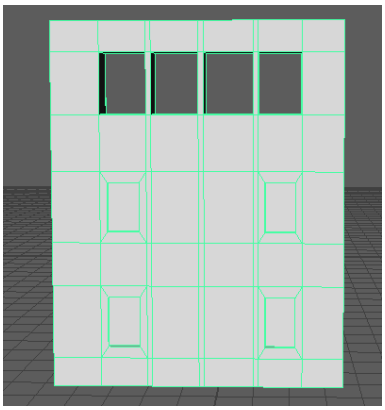


Además, se agregó el techo mediante una calca exacta hecha con Ctrl+D y el ajustamos la posición para que quede donde deseamos, esto solo funcionará para ser la base de nuestros cuerpos que serán después el tejado de la fachada.

4. Una vez que tuvimos los marcos de ventanas y puertas, decidimos crear las paredes internas, como tal el único patrón de distribución al espacio fue que en su mayor parte los 3 cuartos que tiene nuestra casa contaran con el mismo o similar espacio, aunque no con la misma forma. De tal manera que quedó de la siguiente manera la distribución.



5. En la imagen anterior podemos ver que ya contábamos con puertas, las puertas fueron realizadas mediante una forma cúbica de Maya a la cual primeramente escalamos para que entrara en el hueco de las paredes donde dejamos los marcos, una vez que ya entraban, procedimos a utilizar la herramienta Insert Edge Loop Tool para agregarle aristas al cuerpo y así formar más caras en el cuerpo. Para así luego mediante Extrude darle los 4 detalles de profundidad que tiene una puerta. En el caso de la puerta está formada por 4 cuerpos, un cubo con eliminación de caras y con aplicación de la herramienta Extrude e Insert Edge Loop Tool para hacer los detalles, otro cubo escalado de tal forma que simule la base de una chapa, un cilindro que es parte de la chapa y una esfera modificada en sus vértices para que se achate y sea parte de la agarradera de la chapa.

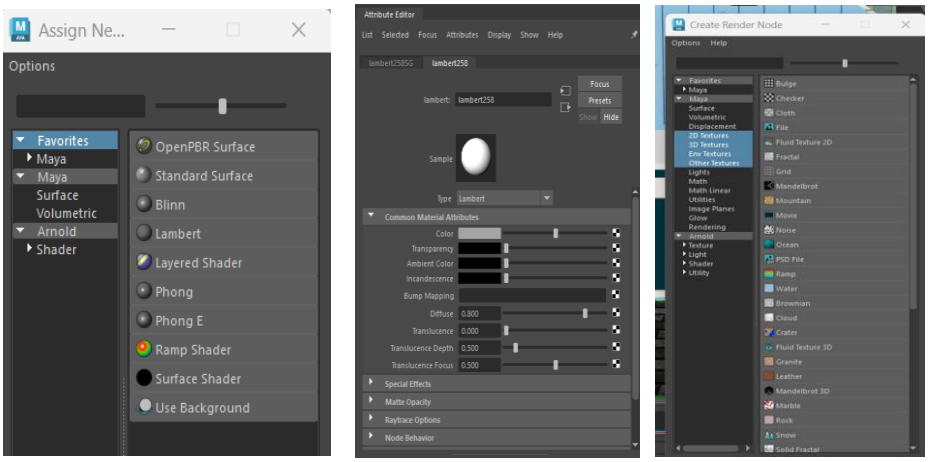



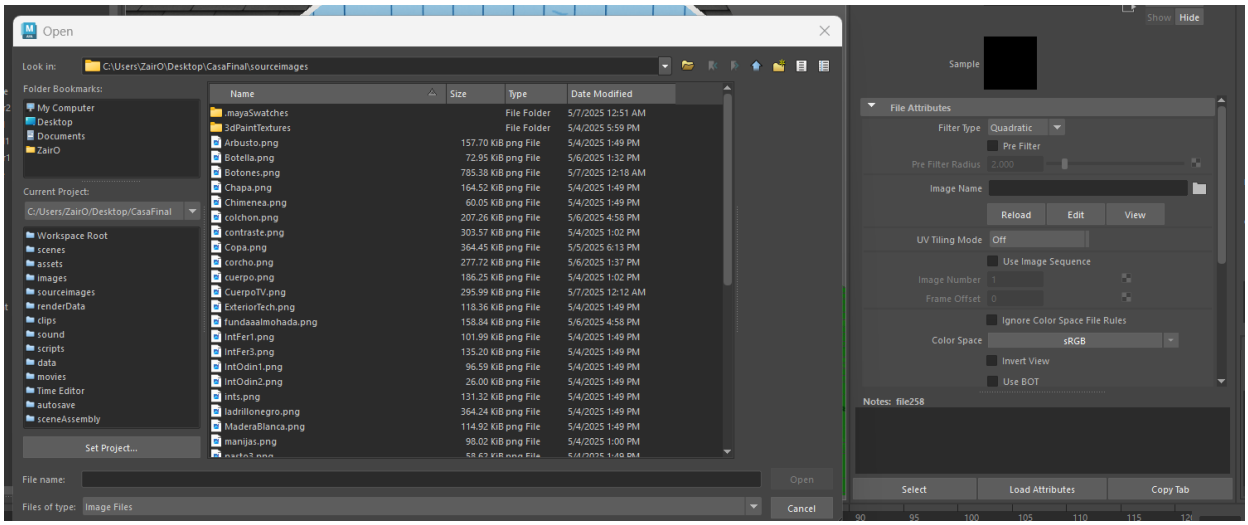

6. Una parte esencial del proyecto fue la asignación de texturas, para ello recibimos apoyo de inteligencia artificial para la creación de estas. El promt la mayor parte de las veces fue *"Necesito que me ayudes a generar una textura similar a la imagen de referencia que sea de 512x512 px y sobre todo que sea -tileable- y en formato png"* aunque no siempre se cumplió una similitud al 100%. En algunos casos los promts también eran de recomendaciones para poder combinar bien los colores dentro de los cuartos. Es importante resaltar que para la mayoría de los objetos se utilizó el material Lambert para poder asignar la textura como lo pueden ser en las paredes, puertas, suelo,

techo y algunos objetos complementarios de la casa. El material “Blinn” lo usamos para los vidrios de las ventanas y para algunos objetos como lo pudieron ser la botella y las copas de vidrio.

Para poder asignar una textura en un objeto seguimos lo siguientes pasos:

a) Seleccionar el objeto o la cara a la cual deseamos asignar la textura.
b) Con click derecho seleccionar la opción Assing New Material / New Material.
c) Seleccionar el material de acuerdo con las necesidades. Y una vez seleccionado se abrirá el panel lateral de Atribute Editor, en color escogimos algún color que tuviera el objeto o bien en el cuadrito blanco con negro, se presiona y abre un menú donde podemos seleccionar diferentes tipos de material y nosotros siempre elegíamos la opción de File la cual debe seleccionar Image Name y agregar el archivo mediante el explorador de archivos.





d) Es importante utilizar la herramienta UV en Maya, ya que permite definir cómo se aplicará una textura 2D sobre un objeto 3D. El proceso de mapeado UV genera coordenadas que conectan cada punto del modelo con su equivalente en la imagen de textura, asegurando que se visualice correctamente.

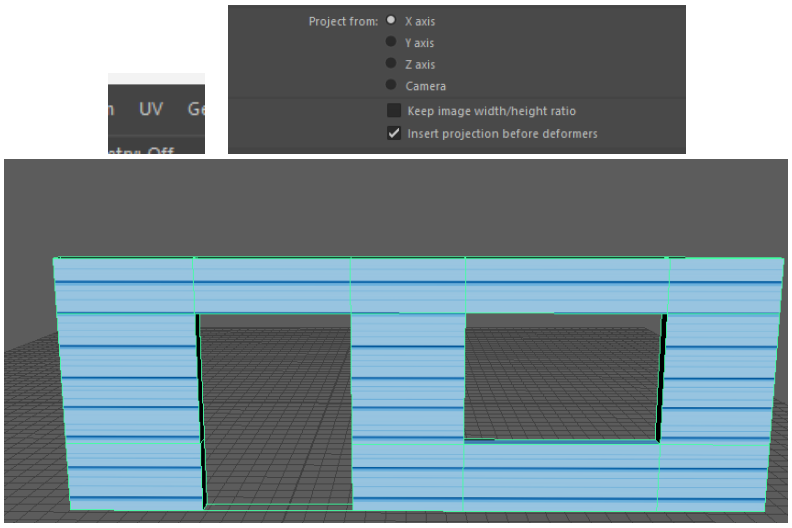
Para ello, es necesario aplicar una de las siguientes proyecciones según la geometría del objeto:

Cylindrical (cilíndrica): Ideal para objetos con forma tubular, como botellas o columnas. La proyección se realiza envolviendo la textura alrededor del eje principal del objeto.

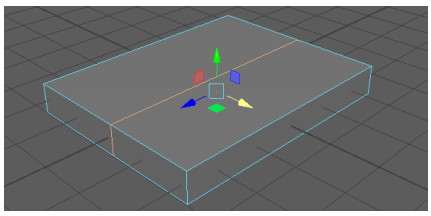
Spherical (esférica): Útil para esferas u objetos redondeados, permite una distribución uniforme de la textura desde el centro hacia afuera.

Planar (plana): Adecuada para superficies planas o ligeramente curvas. Se puede proyectar desde un eje específico (X, Y o Z) o desde la cámara activa, según convenga.

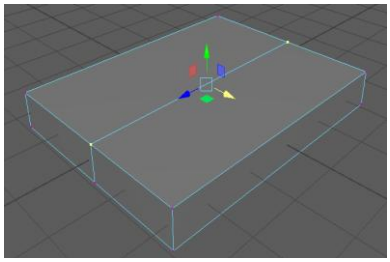
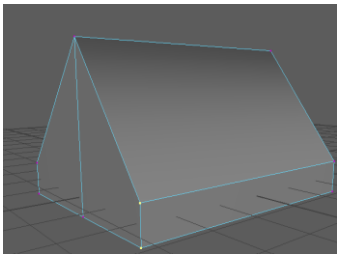
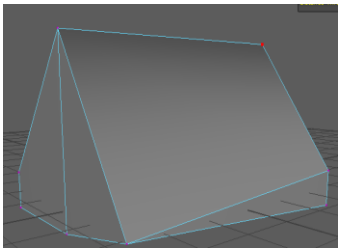
En el caso de objetos cilíndricos o esféricos, se recomienda usar Cylindrical o Spherical respectivamente, proyectándolos desde el cuerpo del objeto. Si el objeto es más plano o irregular, se puede usar Planar, eligiendo el eje que mejor se adapte a la orientación del objeto o utilizando una proyección desde la cámara.

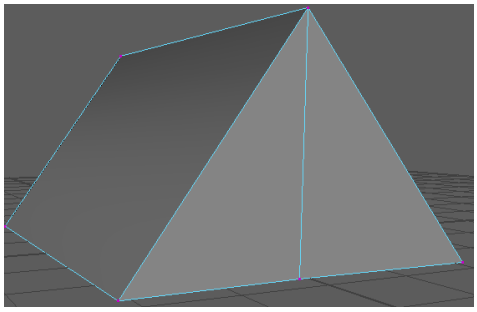
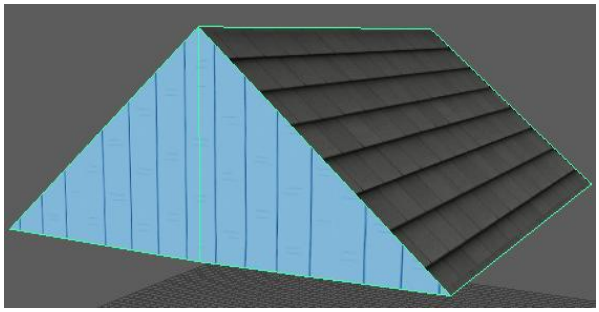


7. Para poder realizar la construcción del techo, ocupamos 2 objetos de forma cúbica, los ajustamos de acuerdo con la división de caras que tiene la base del techo que hicimos en el punto 3, en este caso la base contiene 2 divisiones por lo cual una vez teniendo los 2 cubos ajustados encima de las caras de la base, procedimos a usar Insert Edge Loop Tool y agregamos solo una arista de esta forma:

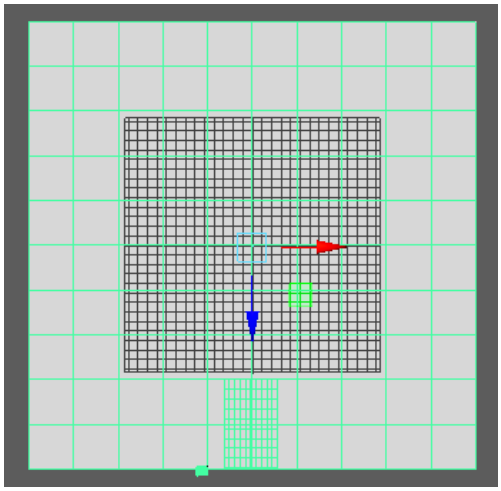
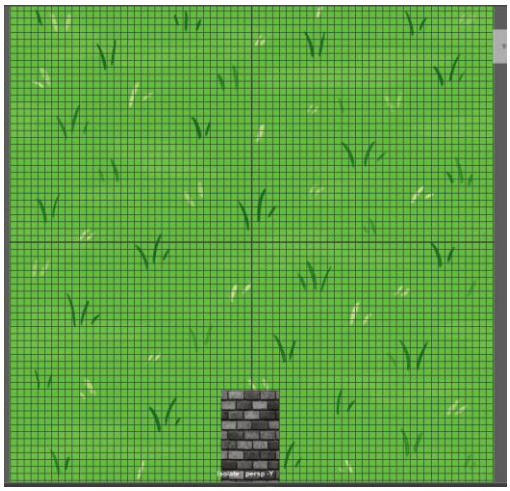


Esto lo que nos permite es poder levantar los vértices del Edge agregado y poder hacer la forma de prisma triangular la cual se requiere para el diseño de nuestro tejado. Para ello fue necesario ocupar la herramienta de Vertex con click derecho. Una vez estando ya dentro de vertex seleccionamos los vértices de en medio y con la herramienta de Move Tool aumentar en el eje Y. Para poder eliminar esa forma de los laterales lo que haremos es seleccionar los vértices laterales de modo que se haga un par, una vez seleccionados utilizamos la herramienta de Merge Vertices y aplicamos esto en las 4 parejas laterales de vértices, esto lo que hace es juntar ambos vértices y convertirlos solo en 1. Por últimos agregamos la textura correspondiente a cada una de las caras.

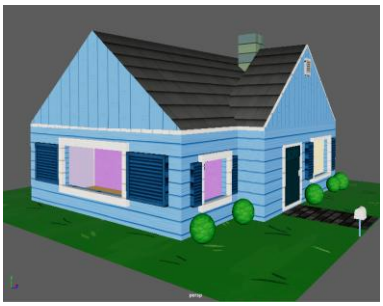
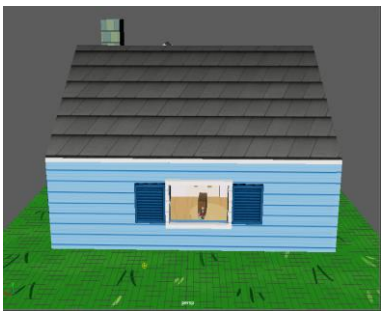
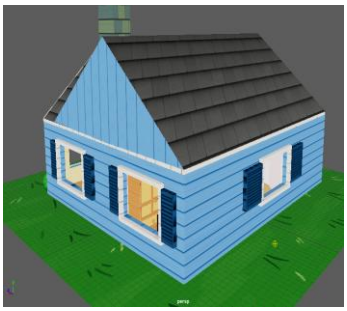




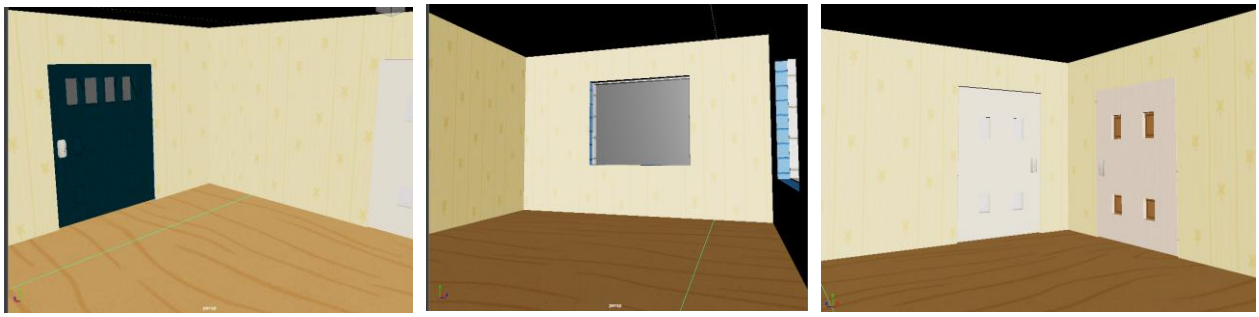
8. Ahora, procedemos a ponerle suelo a nuestro proyecto, para esto ocupamos dos planos de los cuales tenemos disponibles en el software, uno será para el pasto y el otro para el camino de ladrillos, una vez agregados en las posiciones finales, se les pone la textura.

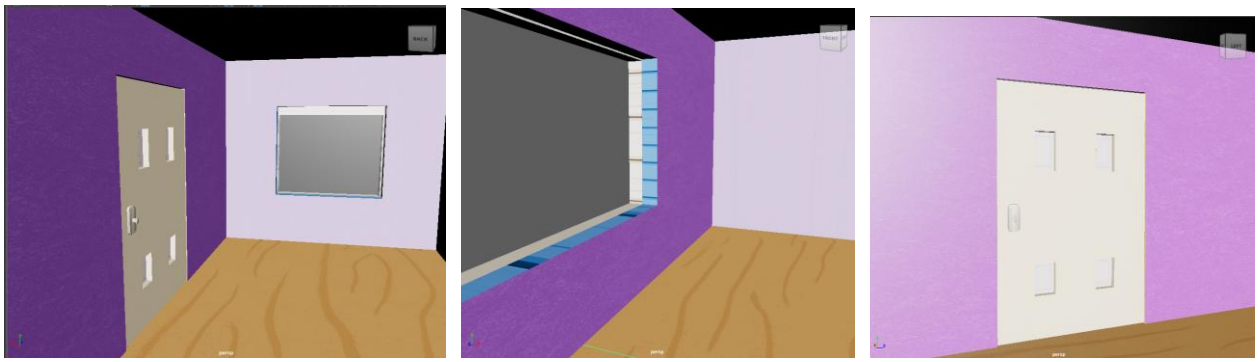
9. Después de agregar todas las texturas a la fachada y al interior, nuestra casa se ve se la siguiente manera:

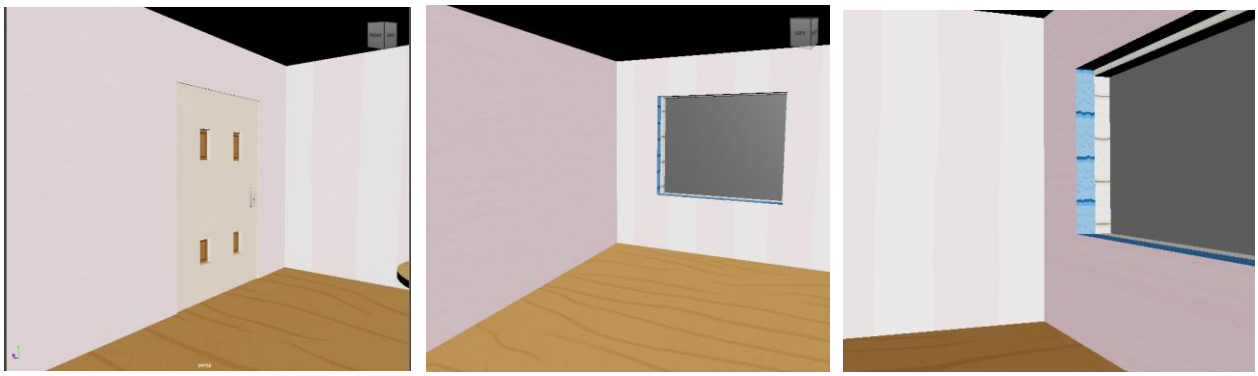
Interior cuarto 1:



Interior cuarto 2:

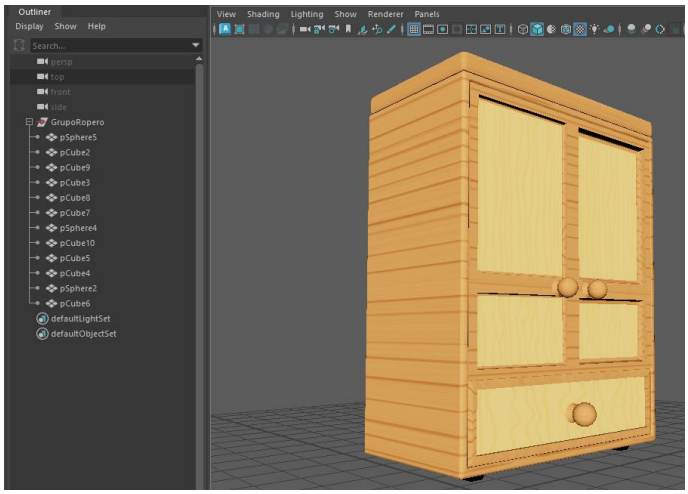


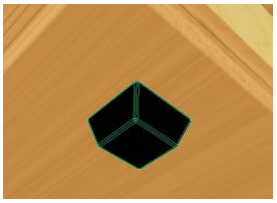
Interior cuarto 3:



Los objetos que se crearon para poder ambientar nuestra fachada fueron los siguientes:

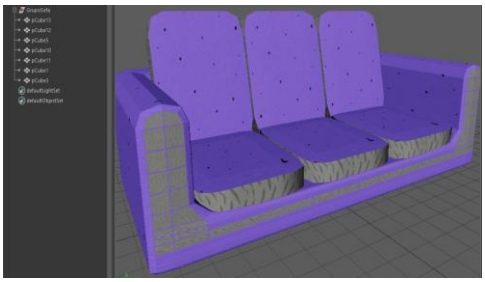
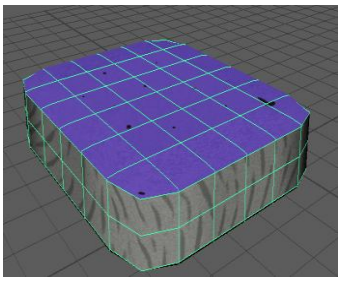
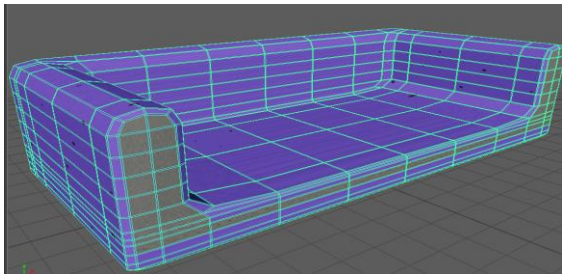
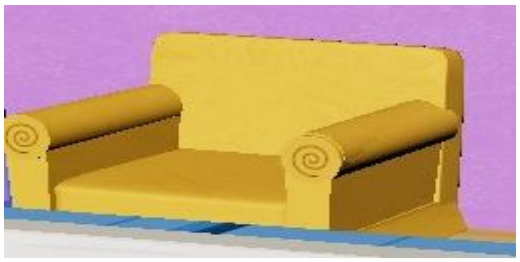
Ropero:
El ropero está construido en su mayoría por cubos y 3 esferas que son las manijas de los cajones. Las puertas del ropero y el cajón están planeadas para animarse en un futuro por lo cual decidimos que sea objetos independientes, estas mismas también se les aplico la herramienta de extrude para dar ese efecto de "profundidad". Las patas son cubos a los cuales se les aplicó la herramienta Insert Edge Loop Tool para que optaran esa forma. También este objeto es texturizado con un total de 3 texturas.





Sofá:

Fue construido por 6 cubos, el principal es la base del sofá Se agregaron divisiones en altura, ancho y profundidad para permitir modelado más detallado. Se usaron herramientas como Insert Edge Loop Tool para definir bordes y detalles curvos. La forma curva de los descansabrazos se logró con Extrude, ajustes de vértices. Para los cojines se aplicaron divisiones para redondear los bordes.

Mesa:

Esta escena esta formada por diversos objetos. La televisión esta construida con 2 cubos independientes, el cubo que contiene la textura será animado en un futuro por eso se mantuvo independiente a la caja de la televisión. Ambos objetos se les agrega edges mediante Insert Edge Loop Tool para que cambiaran su forma a una que no se vea completamente cuadrada.

Para las copas fueron modeladas en Maya utilizando un cilindro poligonal como base. A partir de la cara superior, se aplicaron múltiples operaciones de Extrude seguidas de ajustes con Scale para formar la silueta del objeto. Se generaron secciones como la base ancha, el tallo delgado y el cáliz, modificando progresivamente el tamaño de cada extrusión. Finalmente, se insertaron edge loops para definir mejor los bordes y mantener la forma al aplicar suavizado. Este método permitió construir una copa simétrica y detallada mediante modelado poligonal directo.

Con la botella se modeló a partir de un único cilindro poligonal, utilizando la herramienta Extrude para construir toda su forma, incluyendo el corcho. A partir de la base, se realizaron extrusiones hacia arriba para definir el cuerpo principal de la botella, ajustando el Scale en cada paso para generar las transiciones entre las secciones cilíndricas rectas y el estrechamiento del cuello. En la parte superior, se continuó extruyendo para

formar el cuello y luego el corcho, ampliando ligeramente la última extrusión para crear el borde ancho que lo distingue. Todo esto se logró sin separar geometrías, manteniendo la continuidad del objeto en un solo mesh. Finalmente, se aplicaron materiales y texturas a diferentes secciones del modelo utilizando mapeo UV para simular la etiqueta navideña y la apariencia del corcho. Este enfoque permitió una forma coherente y eficiente sin necesidad de añadir objetos adicionales.

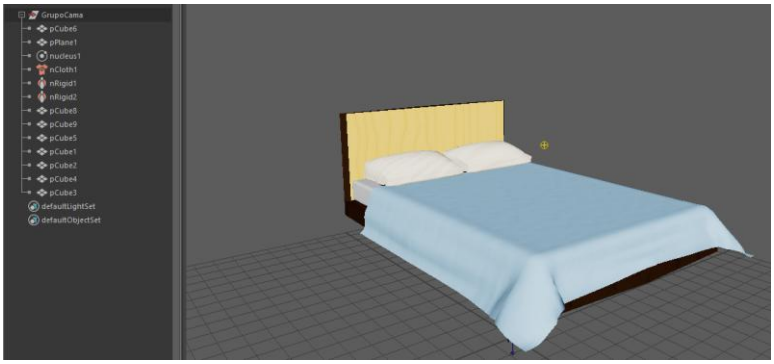
La mesa fue modelada en Maya a partir de geometrías básicas. La superficie de la mesa se construyó utilizando un polyCylinder, al cual se le ajustó la altura para hacerlo delgado y se le aumentaron las subdivisiones radiales para mejorar la definición del borde. Posteriormente, se aplicó una textura de madera clara para simular el acabado superior.

Las patas de la mesa se modelaron a partir de pCubes, que fueron escalados de forma no uniforme para lograr una forma trapezoidal (más ancha en la parte superior y más estrecha en la base). Se duplicaron las patas y se posicionaron en los cuatro extremos del cilindro para mantener simetría y soporte. Finalmente, se asignó un material de madera oscura a las patas para diferenciar visualmente los elementos de la mesa.

Cama:
La estructura principal de la cama (colchón, cabecera y base) se construyó con pCubes, escalados y posicionados para definir el volumen general. Las almohadas también se realizaron con cubos suavizados o con subdivisiones para darles una apariencia mullida.

La sábana fue creada a partir de un Plane, el cual se posicionó encima del colchón. Para simular el comportamiento realista de la tela, a este plano se le aplicó el sistema de simulación nCloth, transformándolo en una malla dinámica. Se ajustaron sus atributos físicos para darle un comportamiento de caída natural sobre la cama. Las colisiones se configuraron utilizando nRigid sobre el colchón y almohadas, de modo que la tela pudiera adaptarse correctamente a la forma de estos objetos.

Finalmente, se utilizó la línea de tiempo de la escena para reproducir la simulación. Tras algunos fotogramas, cuando la sábana alcanzó una posición natural y estable, se detuvo la reproducción y se aplicó el estado de la simulación como pose final, dando como resultado una sábana que se ve suavemente caída y adaptada al colchón.



Casa Snoopy:

La casita fue creada usando únicamente cubos, sin necesidad de otras primitivas. Para formar la estructura principal (paredes y base), se utilizó un cubo escalado en los ejes X, Y y Z para darle la forma de un bloque rectangular. El techo a dos aguas se construyó también a partir de cubos rotados en un ángulo y posicionados simétricamente sobre la parte superior de la estructura, simulando el típico techo inclinado.

Mesas rectangulares:

Cada mesa fue construida de forma sencilla utilizando primitivas básicas:

- Tablero superior: Un pCube escalado en los ejes X y Z para formar la superficie rectangular de la mesa.
- Patas: También son pCubes, escalados de forma no uniforme (más altos en Y y más delgados en X/Z) para darles forma de patas cónicas o trapezoidales. Se duplicaron y posicionaron en las esquinas de la tabla.

Los libros fueron modelados con cubos simples (pCubes) escalados para representar la forma rectangular de un libro cerrado. Se duplicaron y rotaron ligeramente para generar una composición natural (uno encima del otro, con ángulo).

El televisor también se modeló con pCubes: Un cubo grande para el cuerpo principal. Otro más delgado como pantalla.

A diferencia de los anteriores, la tetera y el bowl no fueron hechos con cubos. Se utilizaron primitivas más específicas:

- La tetera fue creada con la forma predefinida teapot.
- El bowl fue generado con un polySphere, al cual se le eliminaron las caras superiores y se aplicó Extrude hacia adentro para darle profundidad.
- Las naranjas son esferas (pSpheres) colocadas dentro del bowl.



Maceta:

Se utilizó un polyCylinder como base. Se escaló en la parte superior para hacerla más ancha que la base, creando una forma troncocónica (como una maceta real). Posteriormente, se seleccionaron las caras superiores y se aplicó un Extrude hacia adentro con “Offset”, y luego un Extrude hacia abajo para crear el hueco interno de la maceta. Las hojas fueron duplicadas y rotadas para que salieran del centro en distintas direcciones.

Cada hoja fue probablemente creada a partir de pCube muy delgado, al cual se le ajustaron los vértices para dar curvatura y forma de hoja puntiaguda.

➔ Los modelos fueron exportados en formato .obj cuidando la geometría, escalado y compatibilidad con OpenGL. Px

Una vez completado el modelado de los objetos en Autodesk Maya, se procedió a la exportación de cada uno en formato .obj, el cual es ampliamente compatible con motores gráficos y librerías como Assimp y OpenGL. Durante este proceso se prestó especial atención a varios aspectos técnicos clave:

Geometría optimizada: Se verificó que cada modelo tuviera una malla limpia y sin errores, evitando caras invertidas, normales mal orientadas, polígonos no manifold o innecesarios. También se aplicaron combinaciones de objetos cuando fue necesario para simplificar la estructura jerárquica del archivo.

Escalado uniforme: Se ajustó la escala global de los modelos para que fueran coherentes entre sí y adecuados al entorno general. Esto evitó problemas de visualización y permitió que al integrarlos en OpenGL no requirieran transformaciones excesivas.

Pivotes y orientación: Se centraron los pivotes de los objetos y se alinearon correctamente respecto al eje Y (arriba) para asegurar una integración directa y sin desajustes en la escena renderizada.

Compatibilidad con OpenGL: Antes de exportar, se congelaron las transformaciones y se eliminaron los historiales de construcción en Maya para evitar inconsistencias. Además, se exportaron sin materiales embebidos, ya que en OpenGL se cargan mediante shaders y texturas separadas. Se garantizó que los nombres de los archivos fueran claros y consistentes para facilitar su carga mediante código.

Desarrollo incremental del software

➔ Se inició con la base gráfica del proyecto usando C++ y OpenGL, incorporando bibliotecas como GLFW, GLEW y Assimp.

Una vez definidos los modelos y su formato de exportación, se procedió a desarrollar la estructura base del entorno gráfico utilizando el lenguaje de programación C++ en conjunto con OpenGL para el renderizado en tiempo real. Se integraron tres bibliotecas fundamentales para facilitar este proceso:

GLFW fue utilizada para la creación de ventanas, gestión del contexto de OpenGL y el manejo de eventos de entrada del usuario (teclado y mouse). Esto permitió definir la resolución,

habilitar el modo interactivo y capturar acciones del usuario para mover la cámara o activar funciones dentro de la escena.

GLEW (OpenGL Extension Wrangler) se incorporó para acceder a las extensiones modernas de OpenGL, asegurando la compatibilidad con shaders personalizados y funcionalidades avanzadas como VBOs (Vertex Buffer Objects) y VAOs (Vertex Array Objects).

Assimp (Open Asset Import Library) se empleó para cargar los modelos en formato .obj exportados desde Maya. Esta biblioteca facilitó la lectura de geometrías, jerarquías y materiales, convirtiéndolos en estructuras aptas para renderizado en OpenGL.

Con estas herramientas se estableció un entorno gráfico estable, modular y escalable, sirviendo como la base sobre la cual se irían incorporando los elementos del proyecto.

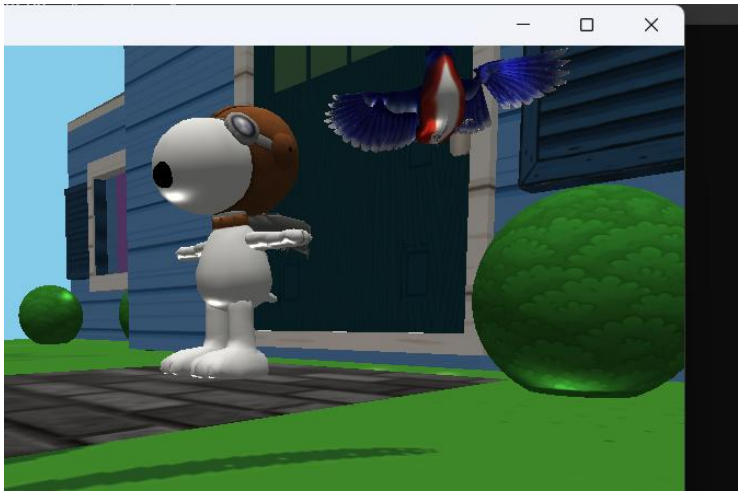
➔ Se integraron progresivamente los modelos exportados, añadiendo texturas, iluminación y animaciones.
A partir de la base gráfica ya funcional, se comenzó la integración de los modelos 3D previamente creados y exportados desde Maya. Esta fase se realizó de manera progresiva, agregando los elementos uno por uno dentro del entorno visual para mantener el control sobre su posicionamiento, escalado y apariencia. Durante la integración:

Se asociaron texturas y materiales a cada modelo, cargándolos como imágenes externas (normalmente en formato .png o .jpg) y mapeándolos correctamente mediante coordenadas UV previamente definidas en Maya.

Se utilizaron shaders en GLSL para manejar los materiales, aplicando iluminación básica con modelos como Phong o Blinn-Phong, lo que permitió simular brillo especular, difuso y ambiental, mejorando el realismo visual de los objetos.

La iluminación se configuró utilizando luces direccionales y, en algunos casos, luces puntuales o spotlights para destacar zonas específicas del entorno.

Además, se implementaron animaciones básicas mediante transformaciones geométricas programadas en tiempo real (traslaciones y rotaciones controladas por teclado), lo que permitió simular efectos como el movimiento del humo de la tetera, la apertura de puertas y ventanas, y la rotación de un pájaro alrededor de un eje. Estas animaciones se controlan dentro del ciclo de renderizado principal, asegurando fluidez y sincronización con la lógica de interacción del usuario.



Este proceso aseguró una presentación visual coherente y atractiva, facilitando además la detección temprana de errores o desfases en los modelos.

Pruebas continuas

- ➔ A lo largo del desarrollo, se realizaron pruebas constantes para verificar la carga correcta de modelos, ajustes visuales y funcionamiento de animaciones.
- ➔ Se priorizó la corrección de errores de carga, iluminación defectuosa o problemas de interacción.

Optimización y documentación

- ➔ Se optimizaron los modelos para mejorar el rendimiento sin comprometer la calidad visual.
- ➔ Se elaboró una documentación técnica que detalla tanto el proceso como el uso del software, incluyendo un video explicativo.

Control de versiones y colaboración

- ➔ Se utilizó Git y GitHub para organizar y respaldar el código fuente, los modelos y recursos del proyecto.
- ➔ Esto permitió un flujo de trabajo coordinado y facilitó el seguimiento del progreso.

Esta metodología permitió un equilibrio entre la libertad creativa (diseño artístico) y el enfoque técnico (programación y documentación), asegurando que el resultado fuera funcional, visualmente coherente y ejecutable en tiempo real.

DOCUMENTACIÓN DEL CÓDIGO

A continuación, mencionaremos los archivos de los cuales consta nuestro proyecto y daremos una explicación de estos.

Descripción General del Archivo Principal (ProyectoFinal.cpp)

Este archivo constituye el núcleo de la aplicación, desarrollada en **C++ con OpenGL**, y es responsable de:

- → Inicializar el entorno gráfico.
- → Cargar shaders, texturas y modelos 3D (casita, objetos, personaje Snoopy).
- → Gestionar entradas del usuario (teclado y ratón).
- → Renderizar la escena 3D de forma interactiva.

Bibliotecas Utilizadas

- → **GLFW**: Manejo de ventana, teclado y ratón.
- → **GLEW**: Acceso a extensiones modernas de OpenGL.
- → **GLM**: Operaciones matemáticas (vectores, matrices, transformaciones).
- → **SOIL2** / **stb_image.h**: Carga de texturas.
- → **Shader.h**, **Camera.h**, **Model.h**: Manejo personalizado de shaders, cámara en 3D y modelos .obj (con Assimp).

Prototipos de Funciones

- → KeyCallback(...): Captura y gestiona eventos del teclado.
- → MouseCallback(...): Controla el movimiento del ratón.
- → DoMovement(): Ejecuta desplazamientos según teclas presionadas.

Configuración Inicial

- → Dimensiones de la ventana: 800x600.
- → Posición de cámara: vista en tercera persona con offset relativo a playerPosition.
- → Se crea y valida la ventana OpenGL, se activan los contextos, y se asignan las funciones de entrada.

Configuración de Iluminación y Proyección

- → Iluminación direccional estilo Phong.
- → Cuatro luces puntuales y una spotlight controlada por la cámara.
- → Matriz de proyección en perspectiva para simulación 3D.

Animaciones Específicas Implementadas

1. Humo de la Tetera

- → Se implementa con un sistema básico de partículas.
- → Cada partícula (una esfera) tiene posición, tiempo, velocidad y escala.
- → Suben desde origenHumo, oscilan en X y Z, crecen y luego reinician su posición.
- → El efecto visual se logra mediante blending con canal alfa.

2. Animación del Pájaro (tecla "L")

- → Inicialmente estático sobre un arbusto (posición arbustoBase).
- → Al presionar **"L"**, gira en órbita usando cos() y sin() con un radioVuelo definido.
- → Se incrementa su altura ligeramente para simular vuelo.
- → Se detiene si se presiona nuevamente la tecla.

3. Abrir/Cerrar Todas las Ventanas (tecla "B")

- → Cada ventana es un modelo separado (vd1, vf1, vi1, vt1, etc.).
- → Las posiciones se modifican directamente (X o Z) para simular apertura.
- → Usa la variable estadoVentanas para alternar el estado.
- → Se controla el toggle con la variable bPresionado.

4. Animación de la Puerta de una Habitación (por ejemplo, puerta principal)

→ Al presionar la tecla **"P"**, se activa una animación progresiva de rotación (rotacionPuertaPrincipal).
→ La puerta gira alrededor de un pivote definido mediante glm::translate() y glm::rotate().
→ El movimiento es suave y reversible.

Otros Componentes Importantes
Shaders

→ **Core.frag/Core.vs**: Renderizado básico con textura.
→ **Lighting.frag/Lighting.vs**: Shaders con iluminación Phong para materiales realistas.
→ **Lamp.frag/Lamp.vs**: Visualización de luces como cubos blancos.
→ **ModelLoading.frag/ModelLoading.vs**: Renderizado de modelos con canal alfa.

Archivos Personalizados

→ **Shader.h**: Encapsula carga, compilación y uso de shaders.
→ **Camera.h**: Maneja posición, dirección y matrices de vista.
→ **Model.h**: Usa Assimp para cargar modelos .obj con materiales y texturas.

Ciclo Principal de Renderizado (Game Loop)

→ Calcula deltaTime para movimiento independiente del framerate.
→ Ejecuta DoMovement() según entradas.
→ Limpia buffers y activa el blending para transparencias.
→ Configura luces, material y matrices de transformación.
→ Renderiza modelos (casa, personaje, objetos).
→ Visualiza luces como cubos.
→ Dibuja partículas de humo.
→ Dibuja pájaro animado.

CONCLUSIONES

➔ General

El desarrollo de este proyecto me permitió consolidar conocimientos adquiridos en modelado 3D, programación gráfica con OpenGL y trabajo en equipo, integrando de manera coherente cada una de las fases que involucra un entorno interactivo. Desde la construcción de objetos en Maya —como muebles, accesorios y personajes— hasta su exportación en formato .obj, texturizado y carga dentro del entorno de ejecución y animaciones, se logró un flujo completo y funcional de diseño a implementación.

Desarrollar este proyecto fue una experiencia muy enriquecedora, ya que pude aplicar de forma práctica los conocimientos adquiridos sobre computación gráfica, manejo de modelos 3D y programación con OpenGL. Modelar en Maya y luego exportar e integrar esos modelos en una escena interactiva en OpenGL fue un reto técnico que me permitió comprender mejor el flujo completo de trabajo en una aplicación gráfica. Además, el trabajo colaborativo me ayudó a mejorar mis habilidades de comunicación y organización.

El uso de bibliotecas como GLFW, GLEW y Assimp fue clave para estructurar el motor base del proyecto, y mediante shaders personalizados se consiguió simular iluminación direccional, puntual y tipo spotlight, lo cual aportó realismo a la escena. También se implementó una cámara en tercera persona, movimiento de personaje y luces dinámicas, logrando una interacción fluida y visualmente agradable.

REFERENCIAS

- Ing. Espinoza Urzúa, E. Curso de Computación Gráfica Semestre 2025-2 .
- Angel, E., & Shreiner, D. (2012). Interactive Computer Graphics: A Top-Down Approach with Shader-Based OpenGL (6th ed.). Pearson Education.
- Assimp. (n.d.). Open Asset Import Library (Assimp). Recuperado de https://www.assimp.org/
- Autodesk. (n.d.). Maya - 3D Modeling and Animation Software. Recuperado de https://www.autodesk.com/products/maya/overview
- LearnOpenGL. (n.d.). LearnOpenGL. Recuperado de https://learnopengl.com/
- GLFW. (n.d.). Graphics Library Framework. Recuperado de https://www.glfw.org/
- GLEW. (n.d.). The OpenGL Extension Wrangler Library. Recuperado de http://glew.sourceforge.net/
- GitHub Docs. (n.d.). Managing repositories. Recuperado de https://docs.github.com/en/repositories