



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA
INGENIERÍA EN COMPUTACIÓN
COMPUTACIÓN GRÁFICA E INTERACCIÓN
HUMANO COMPUTADORA



Proyecto 2 Final
Manual de Usuario
Computación Gráfica e Interacción Humano
Computadora

Alumno. 319079485

Profesor. ING. CARLOS ALDAIR ROMAN BALBUENA

Grupo 5

Semestre 2026-1

Fecha de entrega: 25 de Noviembre 2025

Índice





1. Diagrama de Gantt	Pag 1-2
2. Manual de Usuario	Pag 3-6
2.1 Objetivos del proyecto	Pag 3
2.2 Tecnologías Utilizadas	Pag 3
2.3 Requisitos del sistema	Pag 4
2.4 Guía de Interacción y controles	Pag 5
2.2.1 Controles Modo Libre	Pag 6
2.2.2 Controles Modo Recorrido	Pag 6
2.2.3 Interacciones Automaticas	Pag 7
5.- Repositorio GitHub	Pag 8
5.- Video demostrativo	Pag 8

Introducción

Este proyecto consiste en el desarrollo de una simulación 3D interactiva construida con C++ y OpenGL 3.3, donde se recrea una casa steampunk caminante equipada con un sistema de animaciones jerárquicas, objetos interactivos por proximidad, un dron autónomo y un recorrido cinematográfico reproducible mediante keyframes.

El objetivo principal es demostrar el aprendizaje de técnicas de computación gráfica: modelado 3D, texturizado, iluminación, animación por interpolación, transformaciones jerárquicas, cámara sintética y manejo de recursos en GPU.

Durante la ejecución, el usuario puede:

-  Explorar libremente el entorno post-apocalíptico.
-  Activar animaciones de la casa mediante teclas.
-  Interactuar con objetos que reaccionan a la proximidad.
-  Ejecutar un recorrido automático previamente grabado.

La escena incluye un skybox de ambiente desértico, un dron con vuelo autónomo circular, puertas con apertura dependiente de distancia, una antena parabólica giratoria y cuatro patas mecánicas articuladas con movimientos.

La implementación del proyecto involucra conceptos de gráficos por computadora en tres dimensiones.

El pipeline de renderizado comienza con la definición de geometría tridimensional mediante vértices, normales y coordenadas de textura almacenados en buffers de la GPU. Los shaders programables procesan esta información: el vertex shader transforma las coordenadas de los vértices del espacio local al espacio de pantalla mediante matrices de modelo, vista y proyección, mientras que el fragment shader calcula el color final de cada píxel aplicando modelos de iluminación Phong y mapeado de texturas. El sistema de iluminación implementa tres tipos de fuentes luminosas: luz direccional que simula el sol con rayos paralelos, luces puntuales que emiten radialmente desde posiciones específicas con atenuación cuadrática, y una luz focal tipo linterna con ángulo de apertura definido. Cada luz contribuye componentes ambientales, difusas y especulares al color final de las superficies según su material y orientación respecto a la fuente.

El proyecto utiliza la biblioteca GLFW para gestión de ventanas y eventos de entrada, GLEW para acceder a extensiones modernas de OpenGL, GLM para operaciones matriciales y vectoriales, SOIL2 para carga de imágenes como texturas, y un parser personalizado para archivos .obj que extrae la geometría de los modelos tridimensionales.

Objetivos

El objetivo principal de este proyecto es desarrollar una escena 3D interactiva en tiempo real utilizando C++ y la API gráfica OpenGL.

1.- Diagrama de Gantt	10/10	15/10	17/10	20/10	21/10	22/10	23/10	24/10	25/10	26/10	27/10	28/10	29/10	30/10	31/10	01/11	02/11	03/11	05/11	06/11	07/11	08/11	09/11	10/11	11/11	12/11	13/11	14/11
Fase 1: Planificación y Setup																												
1.1 Elección de referencia y objetos																												
1.2. Configuración del Entorno (VS, Git)																												
1.3. Setup de Librerías (GLFW, GLEW, GLM)																												
Fase 2: Entorno Básico																												
2.1. Carga de Shaders (Clases)																												
2.2 Modelado de casa fachada																												
2.3 Modelado interior de la casa																												
2.4 Modelado objetos de la casa																												
2.5 Implantación de texturas																												
Fase 3: Creación de escenario																												
3.1 Implementación cámara sintética																												
3.2 Carga de modelos y texturas																												
3.3 Iluminación Básica																												
3.4 Implementación de skybox																												
Fase 4: Animación																												

2.- Manual de Usuario




2.1 Objetivos del Proyecto

Bienvenido al proyecto "Proyecto Final". Este es un entorno 3D interactivo en tiempo real que simula una casa-robot caminante, con estilo inspirado en "steampunk" en un paisaje post-apocalíptico. El objetivo de esta simulación es que se pueda explorar libremente el entorno de una casa caminante.




Podrá moverse por el escenario, interactuar con objetos que reaccionan a la presencia y activar un recorrido cinematográfico automatizado para ver la escena en acción, todo renderizado en tiempo real.

2.2. Tecnologías Utilizadas

Este proyecto fue construido usando un conjunto de herramientas para el desarrollo de gráficos por computadora:

-  Software de Modelado 3D: Blender. Utilizado para crear, editar y ensamblar todos los modelos 3D y exportarlos (.obj) que se usan en la escena (la casa, el dron, las patas, etc.).
-  Lenguaje de Programación: C++. Es el lenguaje principal sobre el que corre toda la lógica del programa.
-  API Gráfica: OpenGL (Open Graphics Library). Es la interfaz que permite al programa comunicarse con la tarjeta gráfica (GPU) para dibujar todos los gráficos 3D.

Librerías externas

-  Librería GLFW: Se utiliza para crear la ventana de la aplicación, recibir las entradas del teclado (W, A, S, D, P, L, etc.) y del mouse.
-  Librería GLEW: Permite al código acceder a las funciones modernas de OpenGL.
-  Librería GLM (OpenGL Mathematics): Se usa para todas las operaciones matemáticas de vectores y matrices (trasladar, rotar y escalar los objetos 3D).



Librería Assimp (Open Asset Import Library): Utilizada para cargar los modelos 3D (los archivos .obj de la casa, el dron, las patas, etc.) y traducirlos a un formato que OpenGL entienda.



Librerías stb_image.h y SOIL2: Se utilizan para cargar los archivos de imagen (como .jpg o .png) que se usan como texturas para los modelos y el skybox.

2.3 Requisitos de sistema

Este proyecto es un visor/ demo desarrollado en C/C++ con OpenGL (GLFW + GLAD + GLM) y ejecutable en Windows. A continuación, se listan los requisitos mínimos y recomendados tanto para ejecutar y compilar el código.

Mínimos (para abrir la escena y recorrido a calidad básica)

- Sistema operativo: Windows 10 (86 bits)
- CPU: Intel/AMD 64 bits con soporte SSE2
- RAM: 4 GB
- GPU: Compatible con OpenGL 3.3 (por ejemplo: Intel HD 4000, NVIDIA GeForce serie 600, AMD GCN 1ª gen) o gráficos integrados de CPU
- VRAM: 1 GB
- Almacenamiento: 500 MB libres (binario + assets)
- Pantalla: 1280 × 720
- Entradas: Teclado y mouse
- Drivers: Controlador gráfico actualizado del fabricante (Intel/NVIDIA/AMD)

Recomendados (para iluminación y animación más fluidas)

- CPU: 4 núcleos (Intel i5/Ryzen 5 o superior)
- RAM: 8 GB
- GPU: OpenGL 4.1+ (ej. NVIDIA GTX 1050/1650, AMD RX 560/570, Intel UHD 620+) tarjeta de video dedicada
- VRAM: 2 GB
- Almacenamiento: 1 GB libres
- Pantalla: 1920 × 1080 (o superior)

2.4. Guía de Interacción y Controles

Para ejecutar el programa, descargue el archivo comprimido .rar titulado “319079485”, descomprima el archivo y después abra la carpeta y ejecute “Proyecto Final Computación Grafica.exe”, espere a que cargue y finalmente se mostrara la escena.

El programa tiene dos modos:



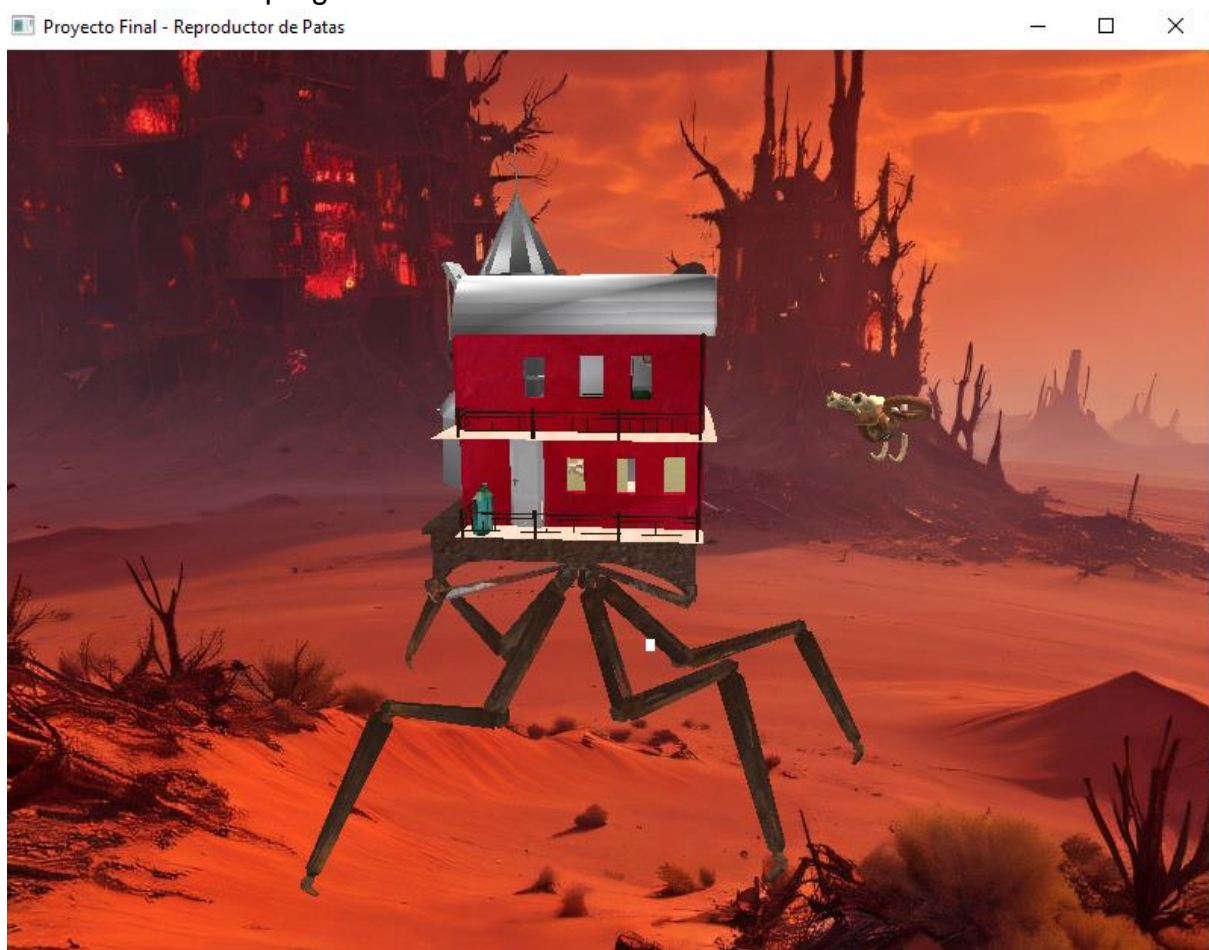
Modo Libre (control manual)



Modo Recorrido (reproducción automática).

El programa iniciará en modo libre (control manual), podrá navegar libremente por la escena.

Vista de inicio del programa:



Podrá comenzar el modo recorrido apretando la Tecla “L”



2.2.1 Controles del Modo Libre

En este modo, se tiene control total sobre la cámara.

Acción	Tecla
Mover la cámara	
Avanzar	W o Flecha Arriba
Retroceder	S o Flecha Abajo
Moverse a la izquierda	A o Flecha Izquierda
Moverse a la derecha	D o Flecha Derecha
Rotar cámara	Mouse
Animaciones	
Comenzar recorrido automático	L
Luz central animada	Espacio
Animación patas metálicas	P
Ocultar pared del segundo piso	F
Salir del programa	
Salir	ESC

Activar Animación de Patas: Presione la tecla P. Las cuatro patas de la casa comenzarán a moverse en su ciclo de animación. Presione P de nuevo para detenerlas.

Ocultar Pared: Presione la tecla F. La pared frontal del segundo piso de la casa desaparecerá, permitiendo ver el interior. Presione F de nuevo para que vuelva a aparecer.



2.2.2 Controles del Modo Recorrido

Al presionar la tecla correspondiente (L), la cámara tomará control automático y comenzará un recorrido pregrabado por la escena (cargado desde camera_anim.txt).

Al mismo tiempo, la animación de las cuatro patas se activará (igual que al presionar P).

Durante este modo, los controles de W, A, S, D y el mouse se desactivarán.

Presione L en cualquier momento para detener el recorrido de la cámara, devolviendo el control manual.

2.2.3 Interacciones Automáticas (Modo Libre)

Mientras se explora en modo libre, la escena reaccionará a la presencia de la cámara:

Objeto	Reacción	Sin reacción cámara lejos	Con reacción cámara cerca
Puerta Principal	Si se acerca la cámara a la puerta principal de la casa, esta se abrirá suavemente. Al alejarse, se cerrará.		
Plato (Dish)	Si se vuela con la cámara hasta el techo y se acerca a la antena (plato) oxidado, esta se activará y comenzará a girar.		

Repositorio GitHub:

https://github.com/soeil1/319079485_ProyectoFinalTeoria_GPO05.git

Para visualizar el funcionamiento completo de la casa y todas sus características implementadas, se preparó un video demostrativo que muestra la navegación el entorno, la activación de las animaciones de cada parte

Video demostrativo:

<https://youtu.be/0YKs-a2w2YA>