



**UNIVERSIDAD NACIONAL AUTÓNOMA DE
MÉXICO**



FACULTAD DE INGENIERÍA

**Computación Gráfica e Interacción Humano
computadora.**

**Manual Técnico
Proyecto Final**

Grupo: 04

Alumna:

Cortés López Maricela

315326237

Profesor: Ing. Carlos Aldair Román Balbuena

Semestre: 2021-2

Fecha de entrega: 10/07/2021



MANUAL TÉCNICO.

OBJETIVO:

El alumno deberá aplicar y demostrar los conocimientos adquiridos durante todo el curso.

Este proyecto final tiene como finalidad que sean aplicados de manera conjunta todos los conocimientos adquiridos a lo largo del semestre en el Laboratorio de Computación Gráfica e Interacción Humano-Computadora en una aplicación, para visualizar la gran utilidad que tiene el conocer y emplear las diferentes herramientas que complementan a OpenGL para la creación de entornos virtuales

DESCRIPCIÓN DEL PROYECTO:

El trabajo consiste en realizar dos cuartos, los cuales contenga 5 elementos diferentes. Dichos elementos van a ser recreados en Maya y 1 tiene que ser modelado en OpenGL usando primitivas. De estos 5 objetos, 3 deben tener una animación sencilla y 2 deben tener una animación más compleja. El cuarto debe tener estructura tanto interna como externa, por lo que fuera del cuarto debe representarse la fachada de una casa entendiendo que es la parte exterior del cuarto.

Para este proyecto se tomó como referencia la casa de la película “UP una aventura de altura”, es por ello por lo que se emplearon imágenes de referencia para poder diseñar todo el escenario, mismas que se encuentran a continuación.

ALCANCE DEL PROYECTO:

Este informe contiene la información sobre la entrega que corresponde al funcionamiento completo del proyecto. Su principal objetivo es el uso de lo aprendido en el semestre Para lograr el objetivo se necesita modelar objetos en 3D, texturizar los objetos y realizar animaciones

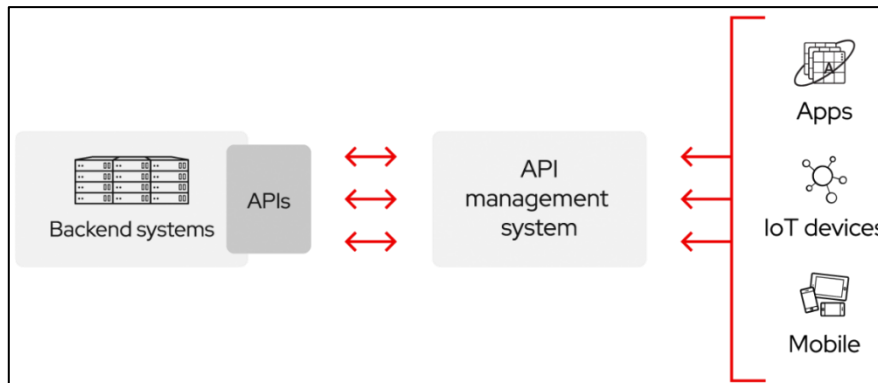
INTRODUCCIÓN.

Una API es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones. API significa interfaz de programación de aplicaciones.

Las API permiten que sus productos y servicios se comuniquen con otros, sin necesidad de saber cómo están implementados. Esto simplifica el desarrollo de las aplicaciones y permite ahorrar tiempo y dinero. Las API le otorgan flexibilidad;



simplifican el diseño, la administración y el uso de las aplicaciones, y proporcionan oportunidades de innovación, lo cual es ideal al momento de diseñar herramientas y productos nuevos (o de gestionar los actuales).



¿Qué es OpenGL?

Se considera principalmente como una API que nos proporciona un gran conjunto de funciones que podemos usar para manipular gráficos e imágenes 3D y 2D. Sin embargo, por sí solo no es simplemente una API, sino una especificación, desarrollada y mantenida por el Grupo Khronos. También se usa para el desarrollo de videojuegos, donde compete con Direct3D en la plataforma de Microsoft.

Especifica exactamente como debería ser el resultado de cada función y cómo debería funcionar. Dado que la especificación OpenGL no proporciona detalles de implementación, las versiones desarrolladas reales pueden tener implementaciones diferentes, siempre y cuando sus resultados cumplan con la especificación (y, por lo tanto, sean los mismos para el usuario).

Los que desarrollan las bibliotecas OpenGL suelen ser los fabricantes de tarjetas gráficas. Cada GPU admite versiones específicas de esta API, que son las versiones de OpenGL desarrolladas específicamente para esa tarjeta gráfica. Por ejemplo, cuando se utiliza un sistema de una compañía como Apple, mantienen la biblioteca OpenGL y, bajo Linux, existe una combinación de versiones de proveedores gráficos y adaptaciones para estas bibliotecas. Esto puede significar que cada vez que OpenGL muestra un comportamiento extraño que no debería, es muy probable que sea culpa de los fabricantes de tarjetas gráficas. Cada vez que hay un error en la implementación, generalmente se resuelve actualizando los controladores de la tarjeta gráfica. Esos controladores incluyen las versiones más recientes que admite su GPU.



Los principales objetivos de OpenGL son:

- Reducir la complejidad de la interfaz con las diferentes tarjetas gráficas, presentando al programador una API única y uniforme.
- Ocultar las diferentes capacidades de las diversas plataformas de hardware, requiriendo que todas las implementaciones soporten el conjunto completo de características de OpenGL (utilizando emulación por software si fuese necesario)

La operación básica de OpenGL es aceptar acciones primitivas tales como puntos, líneas y polígonos, y convertirlas en píxeles. Este proceso es realizado por un pipeline gráfico conocida como la Máquina de estados de OpenGL. La mayor parte de los comandos de OpenGL emiten operaciones primitivas a la segmentación de la gráfica. Hasta la aparición de la versión 2.0, cada etapa de la segmentación se ejecutaba en una función establecida, resultando poco configurable

HERRAMIENTAS NECESARIAS.

Software.

1. Sistema operativo Windows 10.
2. Software especializado en modelado Autodesk Maya.
3. Software especializado en modelado Autodesk 3ds Max.
4. Visual Studio 2019.
5. C++ lenguaje de programación.

Hardware.

1. Laptop
2. Mouse
3. Teclado

LIMITANTES.

1. El primer problema presentado dentro del proyecto fue toda la carga de los .obj ya que al paso del tiempo mientras fue agregando más diseños el tiempo de ejecución se hacía más tardado.
2. El segundo problema fue al momento de crear el archivo ejecutable .exe ya que, el que genera en automático el programa al momento de probarlo y directamente funcionaba correctamente, pero al subirlo a GitHub y probarlo no cargaba los modelos .obj por lo que tuve que buscar otras alternativas.
3. El tercer problema que tuve fue subir todo el programa a la carpeta de GitHub ya que había cosas que directamente no se podían subir, esto me tardo incluso dos días poderlo solucionar, especialmente porque nunca había utilizado esta plataforma.



COTIZACIÓN DEL PROYECTO.



PRESUPUESTO

Presupuesto horas trabajadas

	Sem 1	Sem 2	Sem 3	Total	
1 Análisis Factibilidad, Planes y Requisitos	9	0	0	9	5%
2 Diseño	10	0	0	10	5%
3 Programación	10	20	25	55	40%
4 Documentación	0	0	12	12	8%
5 Pruebas individuales	0	5	15	20	24%
6 Integración y pruebas	0	0	5	5	16%
7 Revisión proyecto	0	0	5	5	3%
Total Horas programación	29	35	125	189	100%
	15%	19%	66%	100%	

Hora de Programación = \$400

Presupuesto en Pesos sobre horas trabajadas

	Sem 1	Sem 2	Sem 3	Total	
1 Análisis Factibilidad, Planes y Requisitos	3,600	0	0	3,600	31%
2 Diseño	4,000	0	0	4,000	34%
3 Programación	4,000	8,000	10,000	22,000	48%
4 Documentación	0	0	4,800	4,800	11%
5 Pruebas individuales	0	2,000	6,000	8,000	18%
6 Integración y pruebas	0	0	2,000	2,000	4%
7 Revisión proyecto	0	0	2,000	2,000	0%
Horas	11,600	14,000	24,800	45,600	100%
	25%	31%	54%	111%	

Presupuesto en Insumos (costos)

	Sem 1	Sem 2	Sem 3	Total	
8 Infraestructura (SW)	6,000	0	0	6,000	36%
9 Costos Fijos (Luz, Agua, servicios, renta)	3,500	3,500	3,500	10,500	64%
10 Transportes	0	0	0	0	0%
Total Costos (insumos)	9,500	3,500	3,500	16,500	100%
	58%	21%	21%	100%	

Presupuesto Total

	Sem 1	Sem 2	Sem 3	Total
1 Análisis Factibilidad, Planes y Requisitos	3,600	0	0	3,600
2 Diseño	4,000	0	0	4,000
3 Programación	4,000	8,000	10,000	22,000
4 Documentación	0	0	4,800	4,800
5 Pruebas individuales	0	2,000	6,000	8,000
6 Integración y pruebas	0	0	2,000	2,000
7 Revisión proyecto	0	0	2,000	2,000
8 Infraestructura (SW)	6,000	0	0	6,000
9 Costos Fijos (Luz, Agua, servicios, renta)	3,500	3,500	3,500	10,500
10 Transportes	0	0	0	0
Total Presupuesto	21,100	13,500	28,300	54,900
	38%	25%	52%	115%

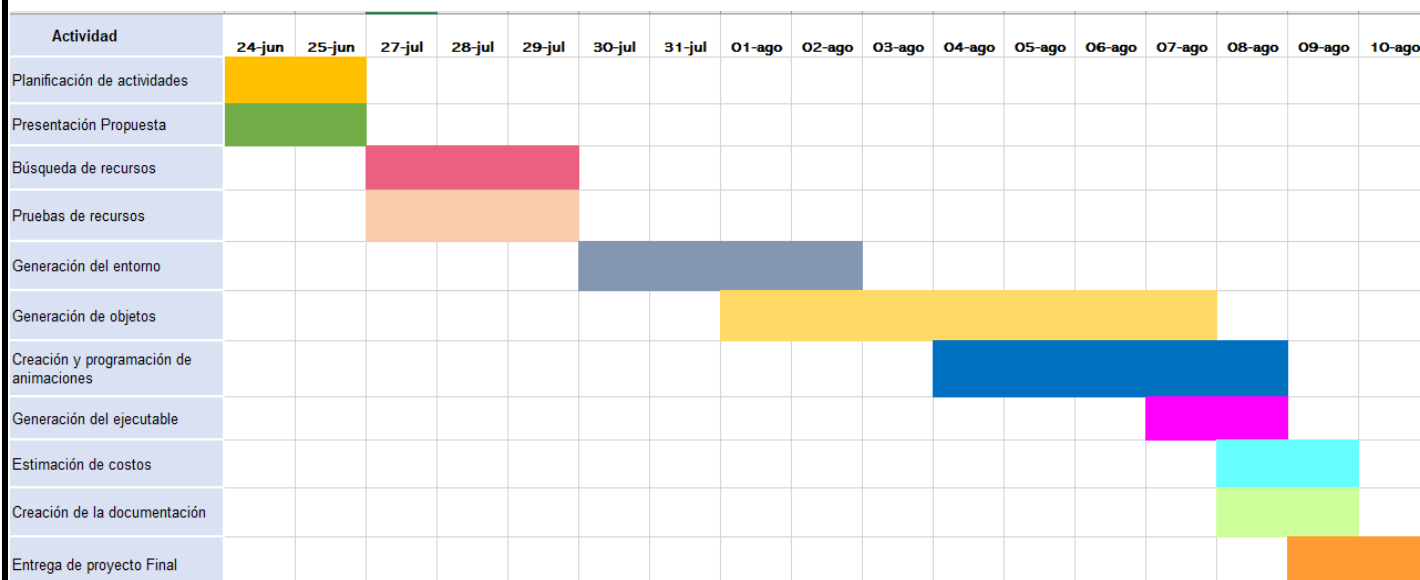


ORGANIZACIÓN:



Actividad	Fecha de Inicio	Duración en días	Fecha de término.
Planificación de actividades	24 junio 2021	1	25 junio 2021
Presentación Propuesta	24 junio 2021	1	25 junio 2021
Búsqueda de recursos	27 julio 2021	2	29 julio 2021
Pruebas de recursos	27 julio 2021	2	29 julio 2021
Generación del entorno	30 julio 2021	3	2 agosto 201
Generación de objetos	1 agosto 2021	7	7 agosto 2021
Creación y programación de animaciones	4 agosto 2021	4	8 agosto 2021
Generación del ejecutable	7 agosto 2021	1	8 agosto 2021
Estimación de costos	8 agosto 2021	1	9 agosto 2021
Creación de la documentación	8 agosto 2021	1	9 agosto 2021
Entrega de proyecto Final	09 agosto 2021	1	10 agosto 2021

A continuación, se muestra el diagrama de Gantt de la planeación y elaboración que tuvo el proyecto.





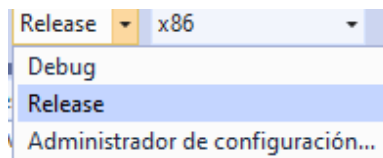
DESARROLLO



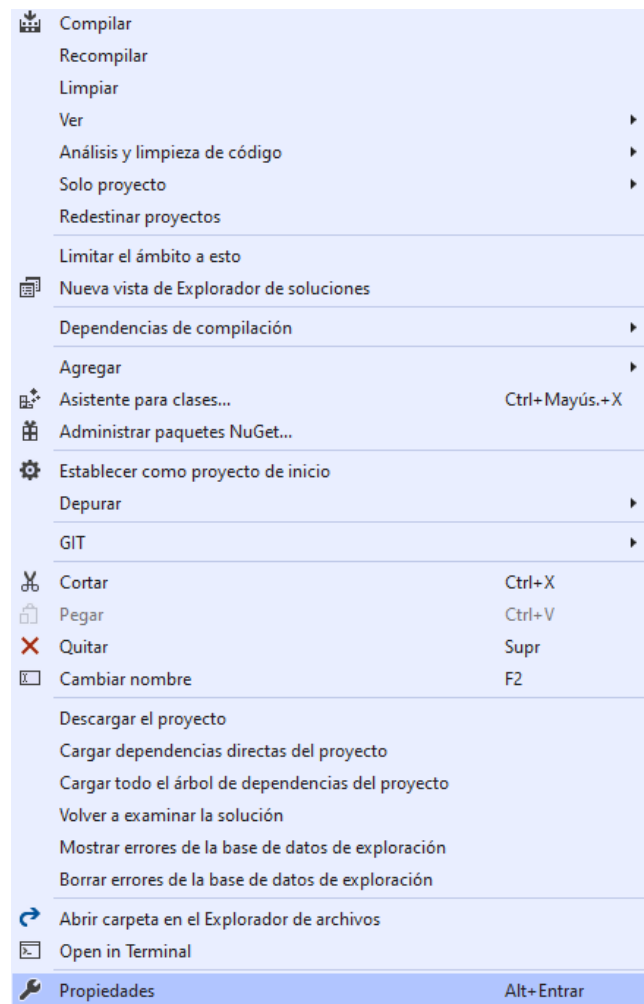
En el proyecto se utilizó como código base, el correspondiente a la Práctica 11 del curso. Ya que a mi parecer es el más completo comparándolo con prácticas anteriores, antes de tener dicho código, se utilizaba el código de la Práctica 9, con éste se hicieron los avances pedidos en días anteriores. Lo que se agregó fue el dibujo de los modelos, sus transformaciones y sus animaciones.

BIBLIOTECAS UTILIZADAS.

El primer paso es poner la configuración en Release



Como segundo paso, es ir a las propiedades del programa.



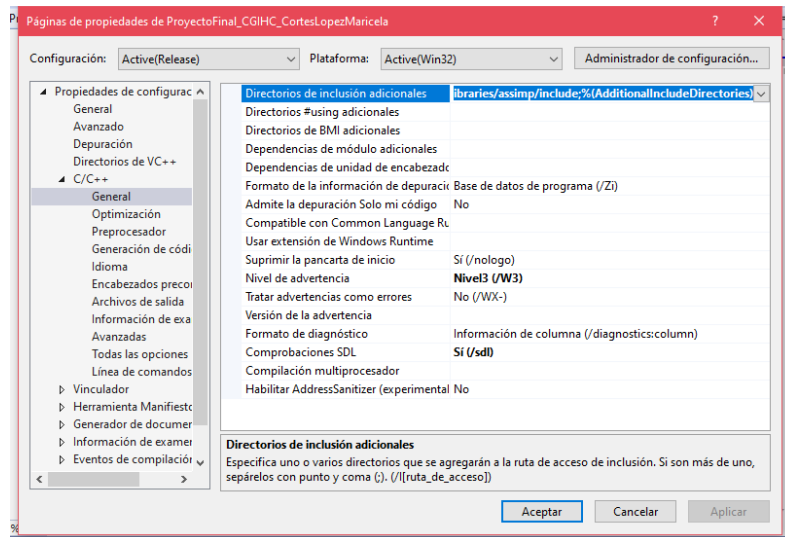


Tercer paso es poner las bibliotecas en cada una de las configuraciones



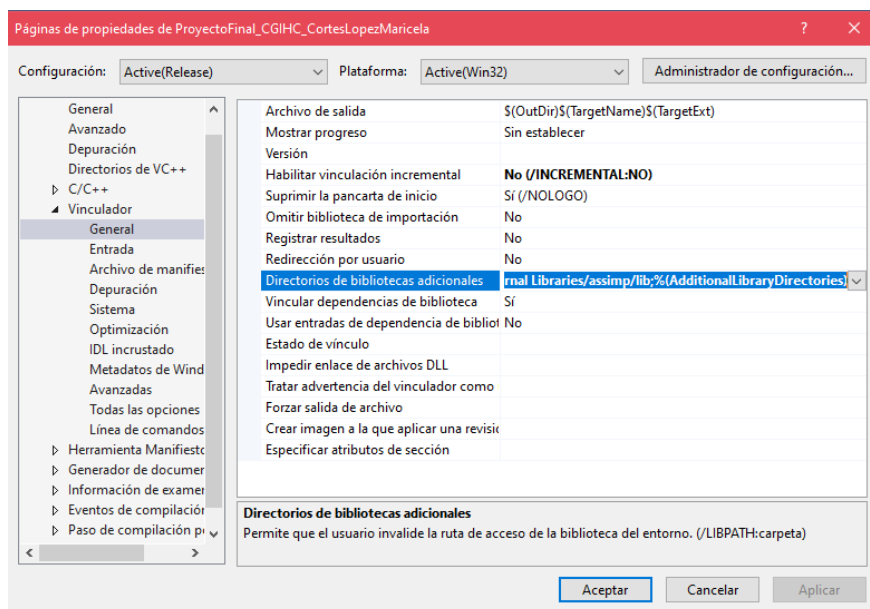
Para C/C++ General:

\$(SolutionDir)/External Libraries/GLEW/include;\$(SolutionDir)/External Libraries/GLFW/include;\$(SolutionDir)/External Libraries/glm;\$(SolutionDir)/External Libraries/assimp/include;% (AdditionalIncludeDirectories)



Para Vinculador General

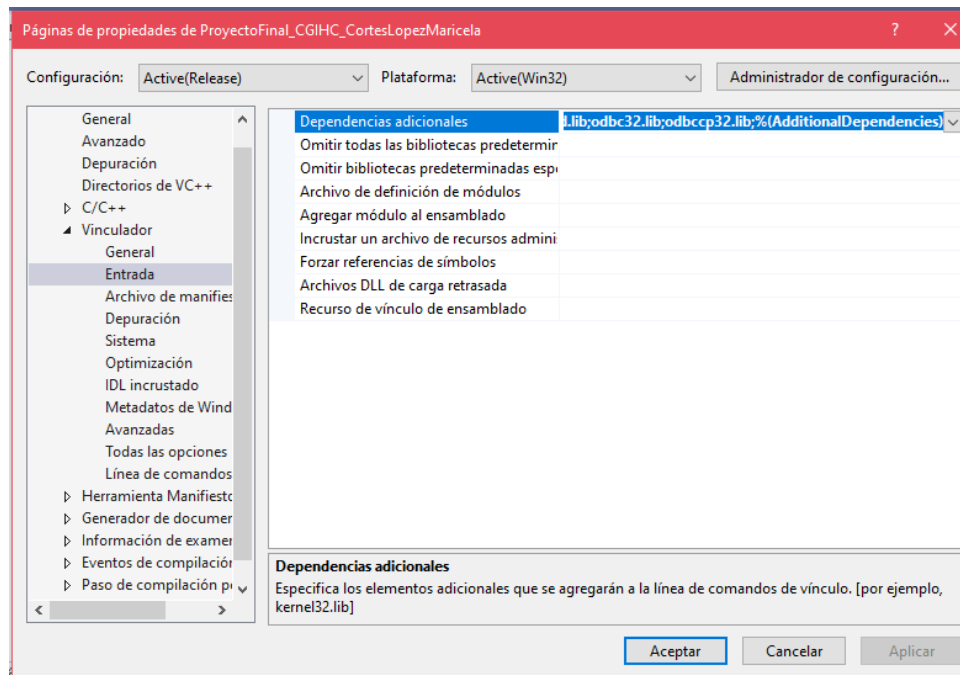
\$(SolutionDir)/External Libraries/GLEW/lib/Release/Win32;\$(SolutionDir)/External Libraries/GLFW/lib-vc2015;\$(SolutionDir)/External Libraries/glm;\$(SolutionDir)/External Libraries/SOIL2/lib;\$(SolutionDir)/External Libraries/assimp/lib;% (AdditionalLibraryDirectories)





Para Vinculador Entrada

soil2-debug.lib;assimp-vc140-
mt.lib;opengl32.lib;glew32.lib;glfw3.lib;kernel32.lib;user32.lib;gdi32.lib;winspool.l
ib;comdlg32.lib;advapi32.lib;shell32.lib;ole32.lib;oleaut32.lib;uuid.lib;odbc32.lib;
odbccp32.lib;% (AdditionalDependencies)



DIRECTORIO DE VARIABLES:

Nombre del archivo de cabecera.	Descripción.
Camara.h	Archivo de cabecera para el manejo de la cámara
Mesh.h	Le envía el objeto a model.h
Model.h	Archivo para modelar y texturizar un objeto
Shader.h	Manejo de paquetes de instrucciones y lo mandará al fragment shader
Stb_image	Archivo de encabezado
Texture.h	Archivo de cabecera para el mapeo de las texturas



DIRECCIONAMIENTO DE DATOS.



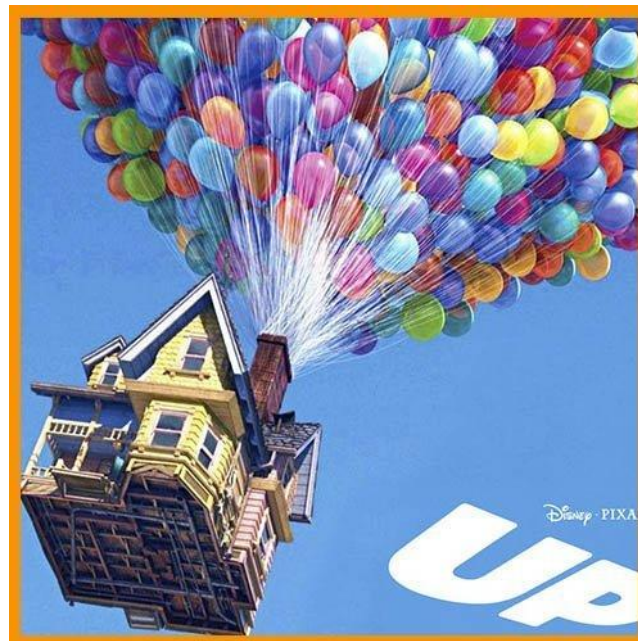
Nombre	Tipo de dato	Descripción
KeyCallback	void	Devolución de llamada de tecla
Devolución de llamada del mouse	void	Devolución de llamada del mouse
DoMovement	void	Función para hacer los movimientos
Animación	void	Función para crear las animaciones de los objetos
WIDTH	const	Medida del ancho de la pantalla
HEIGHT	const	Medida del alto de la pantalla
SCREEN_WIDTH	int	Ancho de pantalla
SCREEN_HEIGHT	int	Alto de pantalla
posX; posY; posZ; incX; incY; incZ;	float float float float float float	Variable para PosicionX Variable para PosicionY Variable para PosicionZ Variable para IncrementoX Variable para IncrementoY Variable para IncrementoZ
rotCarl;	float	Variable para la rotación de Carl
rotGlobo;	float	Variable para la rotación de los globos
movCarl;	float	Variable para el movimiento del carl
movGlobo;	float	Variable para el movimiento de los globos
Play	bool	True activa la animación de los frames y false la desactiva
PlayIndex	int	Índice de la animación.
pointLightPositions	vec	Posiciones de las luces
saveFrame	void	Guardado de los frames
ResetElements	void	Restablecer elementos



Animación	void	Función para crear las animaciones de los objetos
KeyCallback	void	Función de llamado para activar las animaciones cada vez que se presiona una tecla
MouseCallback	void	Devolución de llamada del mouse
DoMovement	void	Mueve / altera las posiciones de la cámara según la entrada del usuario

MODELADO DE OBJETOS 3D:

Para este paso es necesario utilizar un software a nuestro gusto (Maya) para poder recrear un objeto lo más parecido posible a su imagen de referencia



Objetos Creados:

1. Fachada
2. Globos



Objetos Creados:

3. Sillon Carl
4. Sillon Ellie.
5. Banquito
6. Alfombra.
7. Mesa de noche.
8. Lampara de mesa.
9. Libro en la mesa.
10. Librero
11. Chimenea.
12. Foto o pintura en la pared.





Objetos Creados:

1. Foto o pintura en la pared
2. Cuna
3. Juguetes colgados
4. Mesita
5. Ventana

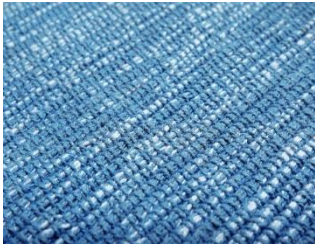
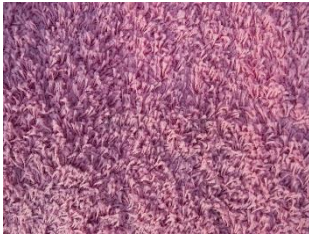

TEXTURIZACIÓN DE OBJETOS:



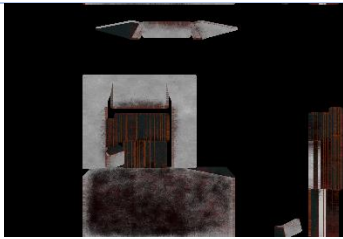



Aquí lo complicado es elegir las imágenes correctas para el texturizado, ya que algunas pueden salir borrosas, no cargan los canales RGB en OpenGL o simplemente no cargan y mandan una excepción en el código.

Texturas Utilizadas.


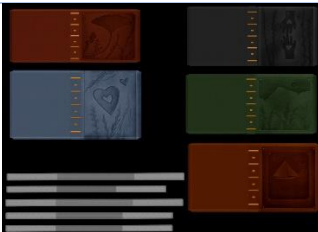

Sillón Carl:			
Sillón Ellie:			
Globos:			
Ventana:			



Alfombra:			
------------------	---	--	---

Chimenea:			
Banquito:			Pintura: 

Lampara:			Mesita: 
-----------------	---	---	--

Librero:			Personaje: 
-----------------	---	---	---

Libro:			Mantel: 
---------------	---	---	--



ANIMACIÓN:

En la parte final, se debe elegir qué elementos pueden realizar una animación y cuales es mejor dejar de forma estática.



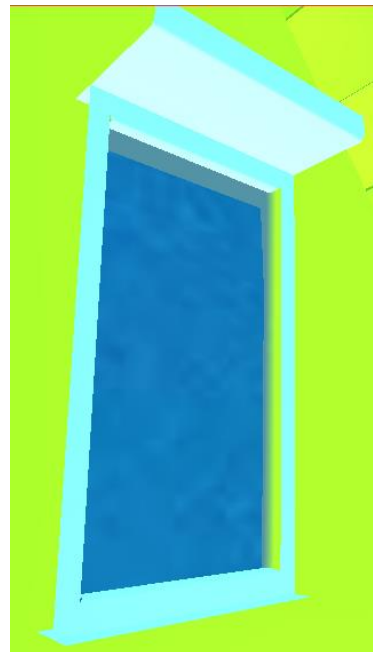
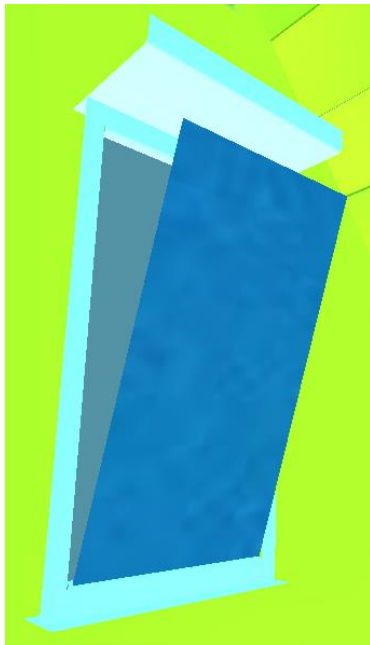
Letra m: abre cajón.

Letra n: cierra cajón lentamente.



Letra c: abre ventana.

Letra v: cierra ventana lentamente.



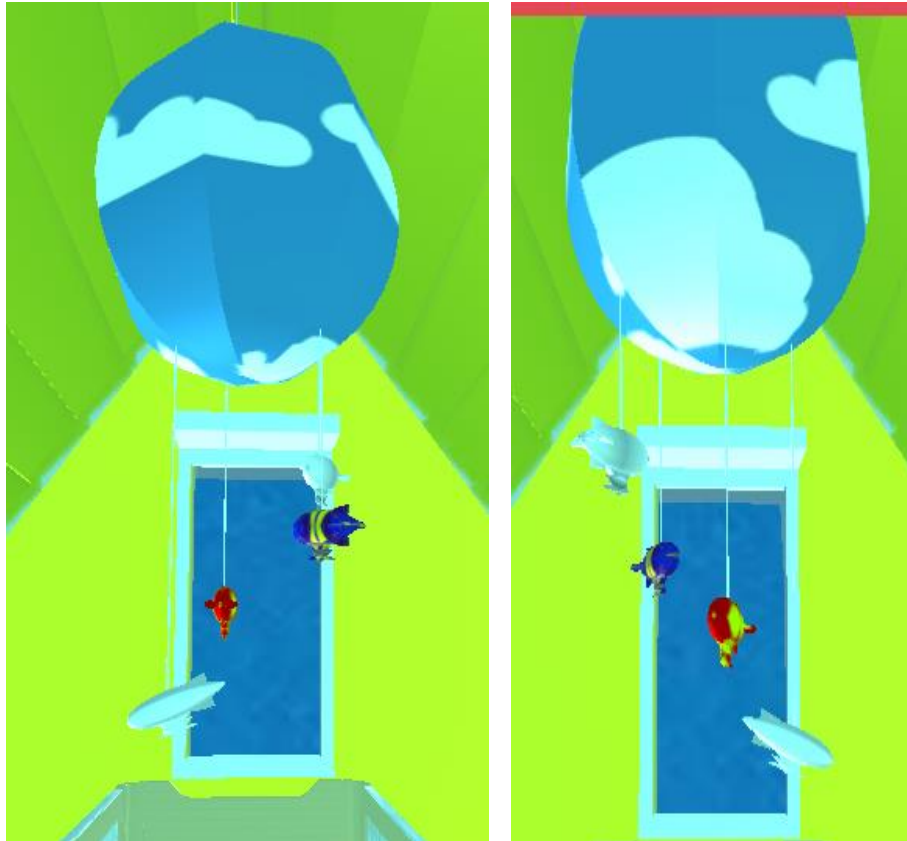


Activación automática: Esta animación la cual es la de flotar se iniciará en automático sin necesidad de activar alguna letra, sin embargo, con.

Letra **o**: para la animación

Letra **i**: reactiva animación.

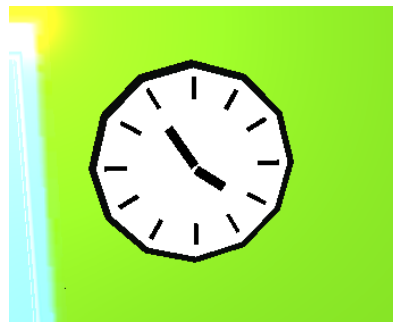
Letra **k**: activa una animación de giro.



Primer Piso: Sala

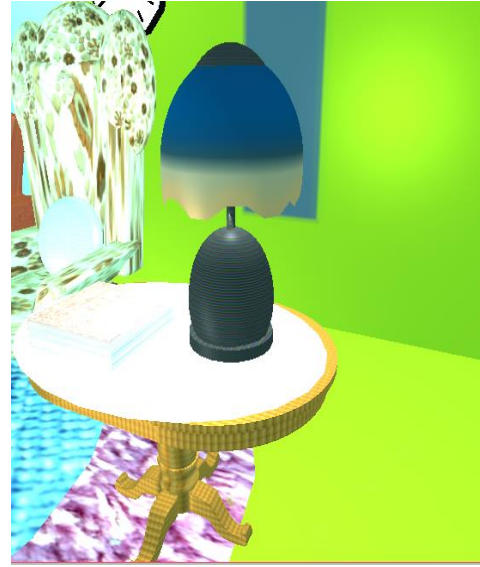
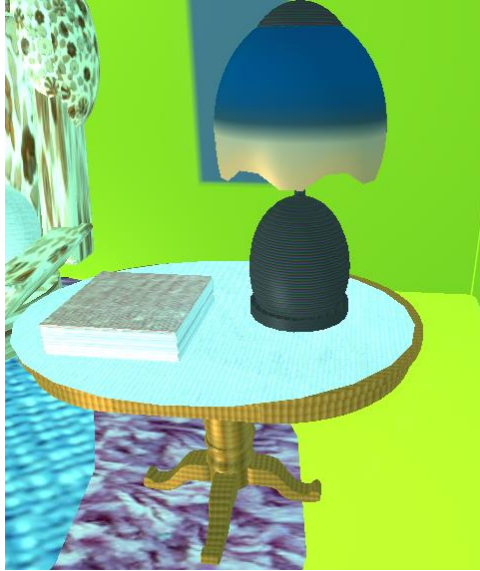
Activación Automática.

El reloj iniciará a girar de manera automática sin necesidad de activar alguna tecla.



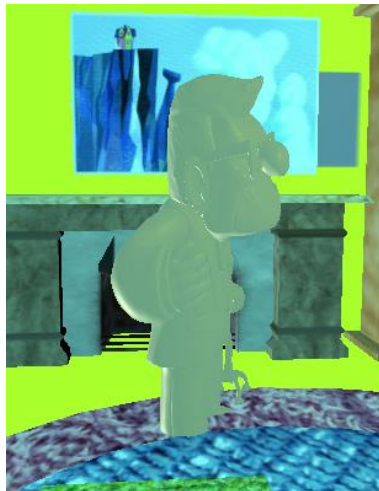


Letra **barra espaciadora**: Activa luces y apaga luces de la lampara.



Letra **L**: camina Carl

Letra **k**: reinicia caminata Carl



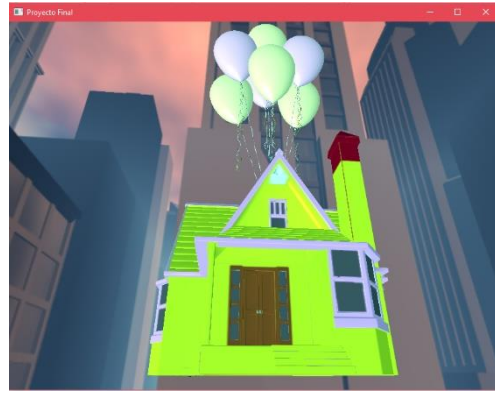
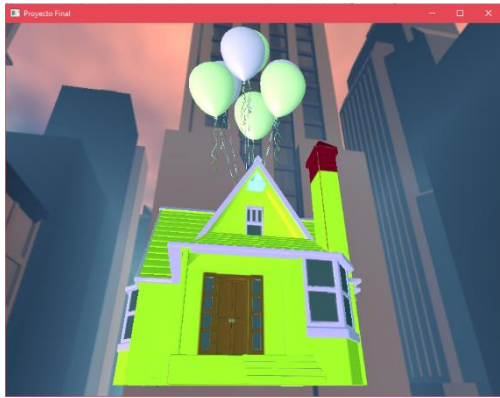
Exterior.

Activación automática: Esta animación la cual es la de flotar los globos se iniciará en automático sin necesidad de activar alguna letra, sin embargo, con.

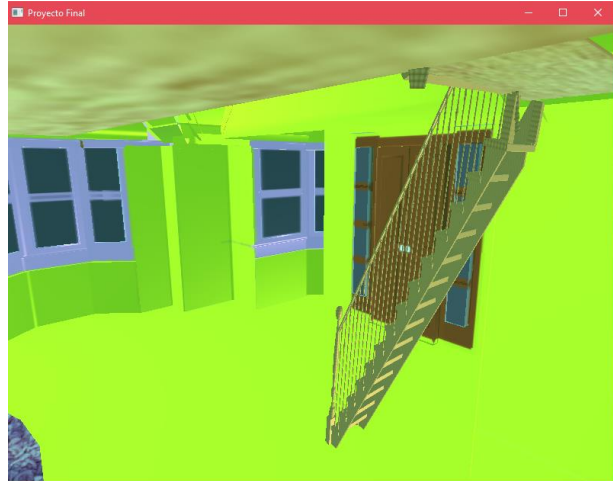
Letra **o**: para la animación

Letra **i**: reactiva animación.

Letra **k y L**: activa una animación de giro.

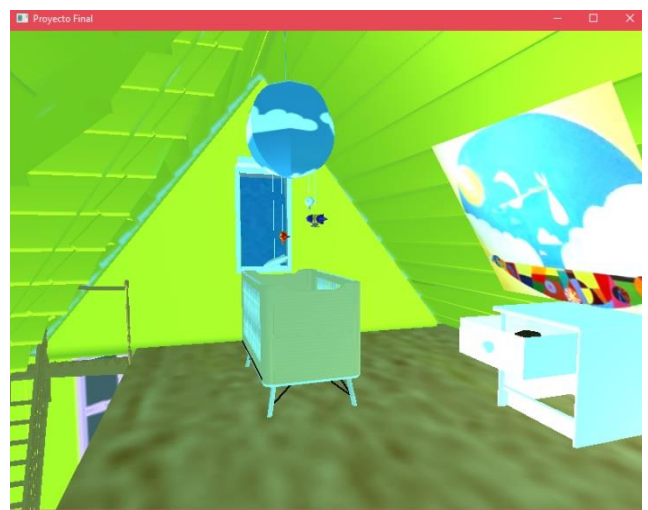
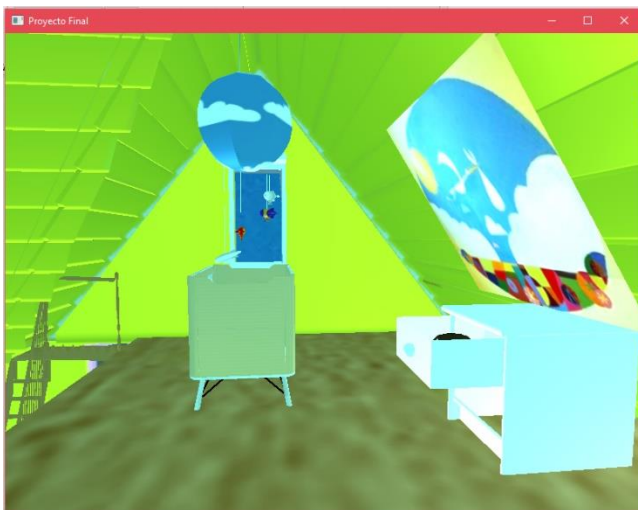


RESULTADOS OBTENIDOS PRIMER PISO:



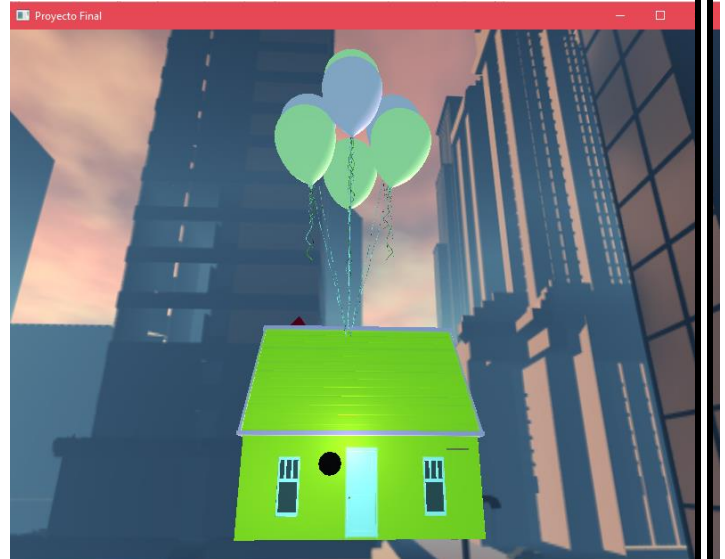


RESULTADOS OBTENIDOS SEGUNDO PISO:





RESULTADOS OBTENIDOS EXTERIORES:



REPOSITORIO EN GITHUB.

https://github.com/MaricelaCortesLopez/ProyectoFinal_CGIHC_GP04_2021_2.git



CONCLUSIONES.



Con la finalización de este trabajo fue posible integrar todos los conocimientos adquiridos a lo largo del curso, tanto en la materia de teoría como en el laboratorio, para construir una aplicación que virtualiza un escenario en 3 dimensiones. Para lograrlo fue necesario establecer las herramientas de software a implementar y posteriormente realizar una etapa de diseño, en la que se ubicó espacialmente el escenario para poder determinar su ubicación y poder cumplir la característica de realismo. Sin embargo, se presentaron varias dificultades en el desarrollo, pues se presentaron problemas en la carga de texturas y en la creación de primitivas geométricas de OpenGL, por lo que realicé una búsqueda en diversos foros de ayuda y en documentación de bibliotecas para poder solucionarlos de manera alternativa.