Escuela Politécnica Superior de Jaén
Grado en Ingeniería Informática

Universidad de Jaén
Escuela Politécnica Superior
de Jaén

TRABAJO FIN DE GRADO

PROTOTIPO DE APLICACIÓN PARA DOCENCIA DE INFORMÁTICA GRÁFICA MEDIANTE PROBLEMAS INTERACTIVOS

Alumno

Manuel Tejada García

Tutor

Francisco de Asís Conde Rodríguez
(Departamento de Informática)

Julio, 2019

Universidad de Jaén

Departamento de Informática

Don Francisco de Asís Conde Rodríguez, tutor del Trabajo Fin de Grado titulado: '**Prototipo de aplicación para docencia de informática gráfica mediante problemas interactivos**', que presenta Don Manuel Tejada García, otorga el visto bueno para su entrega y defensa en la Escuela Politécnica Superior de Jaén.

Jaén, Julio de 2019

El alumno: El tutor:

Manuel Tejada García Francisco de Asís Conde Rodríguez

Agradecimientos

A mi padre y mi madre por el apoyo incondicional durante estos años.
A mi hermano David Tejada García por ser desde siempre un apoyo fundamental en todas mis metas.
A mi primo Youssef Djebari García por mostrarme la informática y porque sin el yo no hubiese escogido este camino.
A mi pareja Alba Milla por ayudarme incondicionalmente incluso no entendiendo de informática.
A mi amigo Guillermo porque empezamos haciendo practicas pero me llevo de estos años a un compañero de por vida.
A Francisco Roca por no darse por vencido y ayudarme a comprender el analisis matemático.
A mi tutor Francisco de Asís Conde Rodríguez por todo el apoyo y dedicación en este trabajo.
A Manuel Gracia Vega por darme ese empujon que necesitaba en mi tercer año.

FICHA DEL TRABAJO FIN DE TÍTULO	
Titulación	Grado en Ingeniería Informática
Modalidad	Proyecto de Ingeniería
Especialidad (solo TFG)	Sin especialidad
Mención (solo TFG)	Sistemas Gráficos
Idioma	Español
Tipo	Específico
TFT en equipo	No
Autor/a	Manuel Tejada García
Fecha de asignación	24/05/2018
Descripción corta	En este Trabajo Fin de Grado, se va a desarrollar un prototipo de Aplicación para el apoyo docente de asignaturas de Informática Gráfica. El prototipo planteará a los estudiantes algunos problemas clásicos de Informática Gráfica. Se mostrarán visualmente mediante escenas 3D junto con explicaciones del problema. El estudiante podrá interactivamente explorar distintas soluciones y ver los resultados que se obtienen. El prototipo mostrará explicaciones de por qué se obtienen esos resultados.

NORMAS APLICADAS EN ESTE DOCUMENTO	
LOCALES	
TFT-UJA:2017	Normativa de Trabajos Fin de Grado, Fin de Máster y otros Trabajos Fin de Título de la Universidad de Jaén (Normativa marco UJA aprobada en Consejo de Gobierno)
TFT-EPSJ:2017	Normativa sobre Trabajos Fin de Grado y Fin de Máster en la Escuela Politécnica Superior de Jaén (Normativa EPSJ aprobada en Junta de Escuela)
TFT-EPSJ	Criterios de evaluación y normas de estilo para TFG y TFM de la Escuela Politécnica Superior de Jaén
NACIONALES E INTERNACIONALES	
ISO 2145:1978	Documentación - Numeración de divisiones y subdivisiones en documentos escritos
UNE 50132:1994	Traducción de la ISO 2145
APA 6ª edición	Estilo de referencias y citas de APA (American Psychological Association)

NORMAS UTILIZADAS COMO BASE O REFERENCIA	
NACIONALES	
UNE 157001:2014	Criterios generales para la elaboración formal de los documentos que constituyen un proyecto técnico
UNE 157801:2007	Criterios generales para la elaboración de proyectos de sistemas de información
*Estas normas se han utilizado **como base o referencia** para la inclusión de algunos contenidos y definiciones sobre elaboración de proyectos, entendiendo como **proyecto** la documentación consensuada entre una empresa y un cliente, que da lugar al perfeccionamiento de un contrato para la elaboración de una obra o la prestación de un servicio. Por consiguiente, no debe esperarse la aplicación de estas normas en cuanto a la completitud de los contenidos ni a la organización de los mismos.*	

Contenido

Índice de ilustraciones

Índice de tablas

1 ESPECIFICACIÓN DEL TRABAJO

1.1 Introducción

La idea original de este proyecto era generar un proyecto gráfico donde dar al usuario varias opciones a elegir y dependiendo de la elección se generase una respuesta, no sabiamos como enfocarlo para diferenciar esta idea de una novela grafica y consultando con el tutor decidimos enfocarlo sobre una asignatura (programación de aplicaciones graficas), lo que queremos conseguir con este proyecto es simplificar el entendimiento de la materia con unos problemas sencillos los cuales pueda manipular, para poder ver de forma visual (y escrita) partes clave de las unidades impartidas en clase, para porder localizar sus problemas con dicha asignatura.

Esta aplicación no será un sustitutivo de las clases tradicionales, sino más bien será una ayuda complementaria para facilitar la comprensión de los temas más visuales de la asignatura.

1.2 Objetivos del trabajo

El objetivo principal de este trabajo fin de grado es aportar una herramienta para ayudar a los alumnos en la comprensión de los temas complejos de entender de la asignatura de programación de aplicaciones gráficas.

Con esta herramienta queremos que el alumno pueda visualizar los puntos más importantes de la asignatura a la vez que leen una pequeña explicación (sea del tema en general o de algo concreto), en algunas escenas mostraremos pseudo código dado que creemos que muchos alumnos suelen tener mayor problema en llevar acabo los objetivos que en comprenderlos.

La razón de este trabajo fin de grado es que vimos que muchos alumnos de mi año y de anteriores años, no obtenían los resultados deseados en ella, al ser una asignatura que tiene un primer contacto algo fuerte.

Por eso mismo, creemos que creando esta herramienta podemos facilitar su entendimiento y/o ayudar a la ejecución de puntos clave de la asignatura.

1.3 Antecedentes y estado del arte

Navegando por internet puedes encontrar algunas herramientas que ayudan y/o facilitan la comprobación de alguna parte de la asignatura como puede ser, comprobar si un shader funciona correctamente.

En este apartado vamos a hablar de las herramientas conocidas por nosotros que nos dieron paso a esta idea.

1.3.1 BKcore

Si entramos en http://shdr.bkcore.com/ podemos observar una página web que nos muestra un objeto pintado con un shader de ejemplo, es decir, nos permite escribir un shader en su editor y ver si funciona o no.

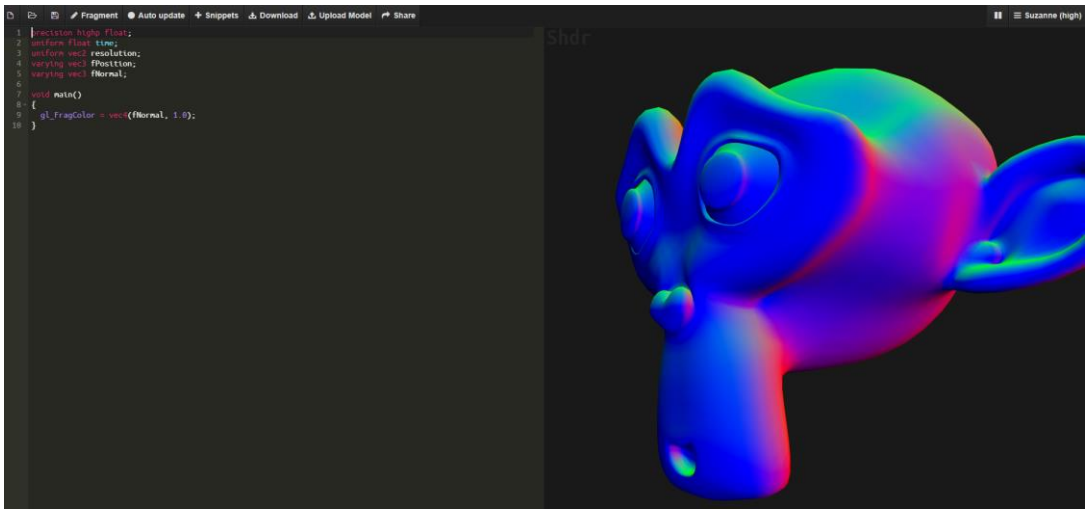


Ilustración 1.3.1

Como podemos observar en la imagen, su editor nos permite editar el fichero, guardarlo, abrir un shader nuestro, incluso compartirlo, pero en ningún momento esta herramienta nos enseña que es un shader y como empezar a trabajar con ellos.

Esta herramienta es muy potente, pero necesitamos antes de usarla tener unos conocimientos básicos que no nos aporta.

1.3.2 Kickjs

Si entramos en este enlace:

http://www.kickjs.org/example/shader_editor/shader_editor.html

podemos observar un editor para comprobar también shader que hagamos en ese momento o que tengamos ya escritos, esta herramienta interactiva, igual que BKcore, nos permite observar si nuestro shader funciona o si estamos fallando en algo, pero seguimos necesitando unos conocimientos previos, ya que, por si solo no nos enseña a programar shader.

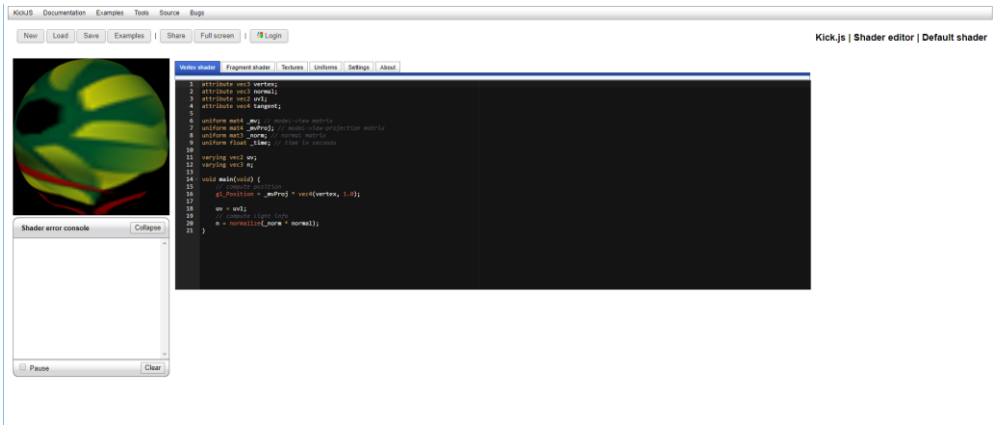


Ilustración 1.3.2

1.3.3 Nuestra herramienta

Viendo estos ejemplos se nos ocurrió que sería muy interesante una herramienta que permitiese al alumno comprender los temas más importante de la asignatura de Programación de Aplicaciones Gráficas, una herramienta que en nuestro primer contacto definimos como un libro interactivo, es decir, una herramienta que al igual que un libro didáctico te permita leer y comprender un tema específico, pero a la vez te permita en algunos apartados modificar la escena que se esta viendo, además, mover la cámara de la escena, para ver con todo detalle la escena.

Nuestro objetivo, a diferencia con los anteriores ejemplos, no se centran en enseñar que es un shader, más bien, los shader serían un tema que se podría añadir a nuestra herramienta, pero nuestra idea es crear una aplicación que ayude a nuestros usuarios (normalmente los usuarios de nuestra aplicación son alumnos, por lo que a partir de ahora nos referiremos a ellos como alumnos) estudiar de un modo alternativo

el temario de esta asignatura, intentando de este modo, que los alumnos puedan comprender la asignatura de un modo más ameno.

1.4 Descripción de la situación de partida

1.4.1 Descripción del entorno actual

Para entender el entorno actual hemos contactado con los usuarios finales los alumnos, y le hemos propuesto una encuesta. Adjuntamos unas ilustraciones con las preguntas de la encuesta (La encuesta original esta disponible en este enlace), para poder ver la encuesta en la web es necesario acceder con una cuenta de la universidad de Jaén, decidi poner este requisito para evitar respuestas falsas y/o de personas que no estaban previstas en el estudio.

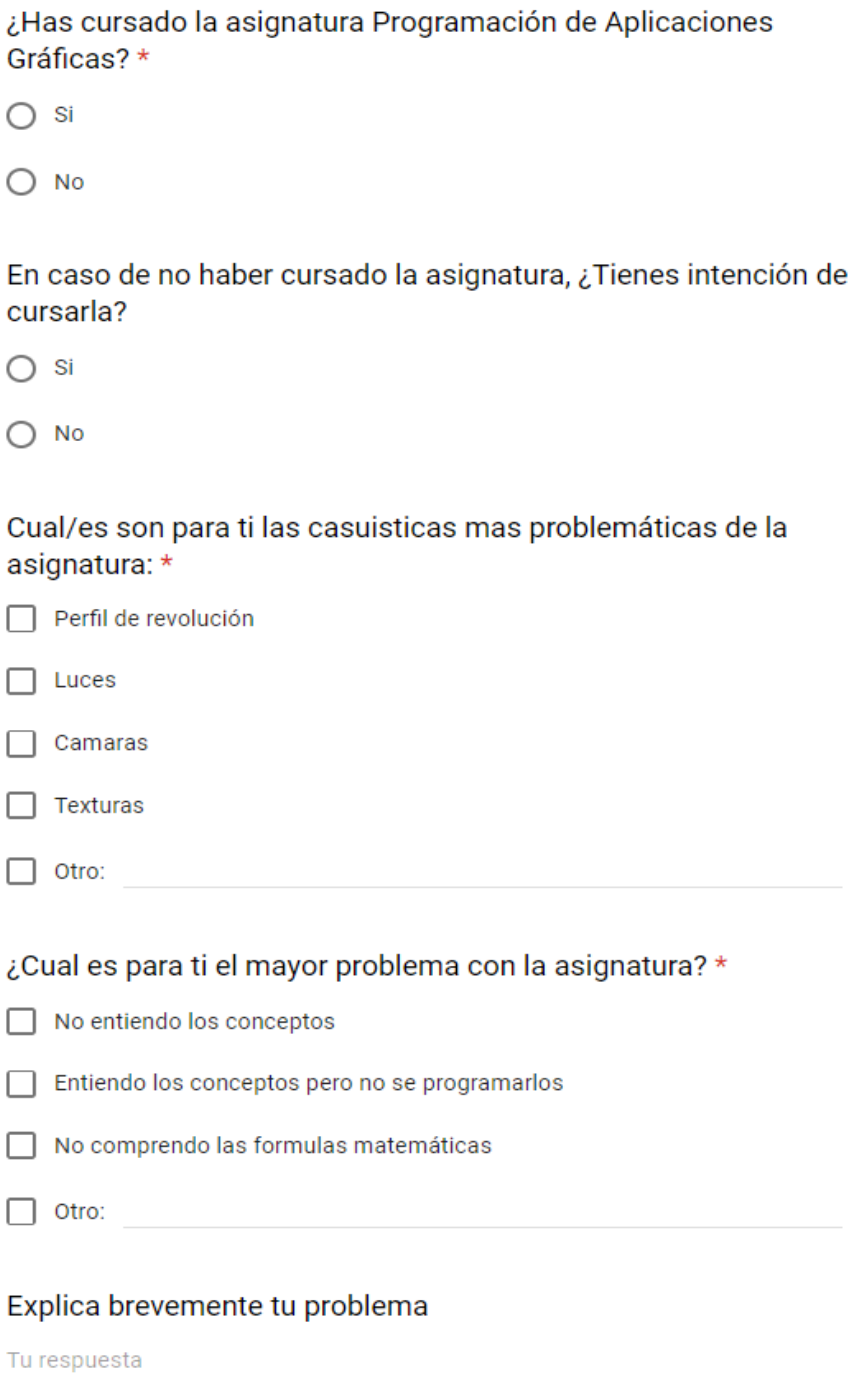


Ilustración 1.4.1

1.4.1.1 Resultados de la encuesta

En este apartado queremos recopilar los datos que nos han facilitado los alumnos, esta encuesta inicial fue contestada por 6 personas y sabemos que no es una muestra significativa si tenemos en cuenta que esta asignatura es optativa y suelen cursarla unas 25 personas tenemos más o menos un 25% de lo que sería una

clase en un curso lectivo, y al ser todo anteriores alumnos de la asignatura creemos que si es una muestra realista.

Sin nada más que aclarar adjuntamos los resultados de la encuesta:

Ilustración 1.4.1.1

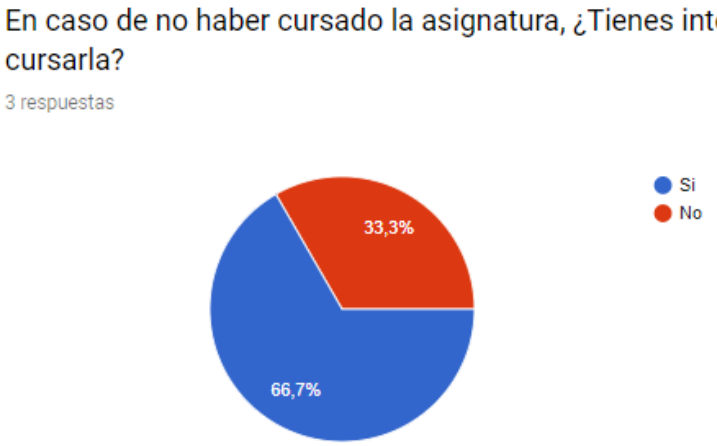


Ilustración 1.4.1.2

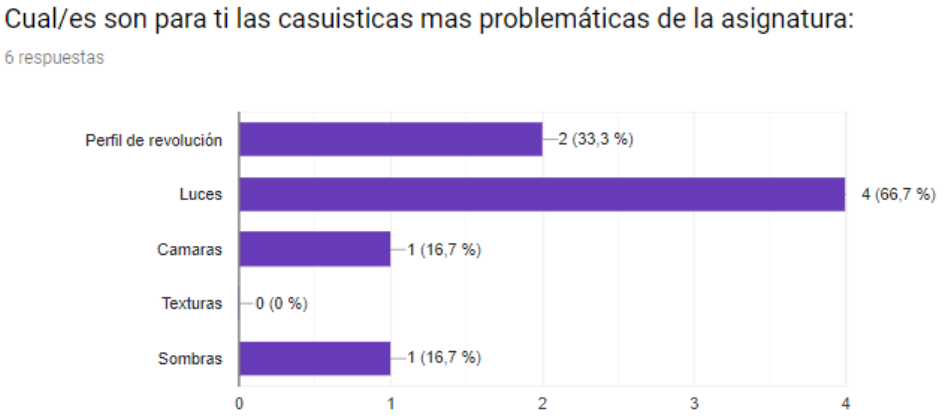


Ilustración 1.4.1.3

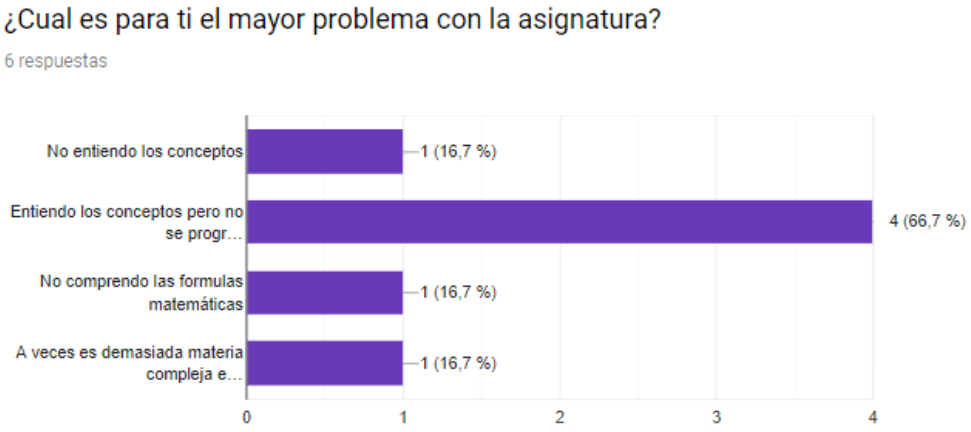


Ilustración 1.4.1.4

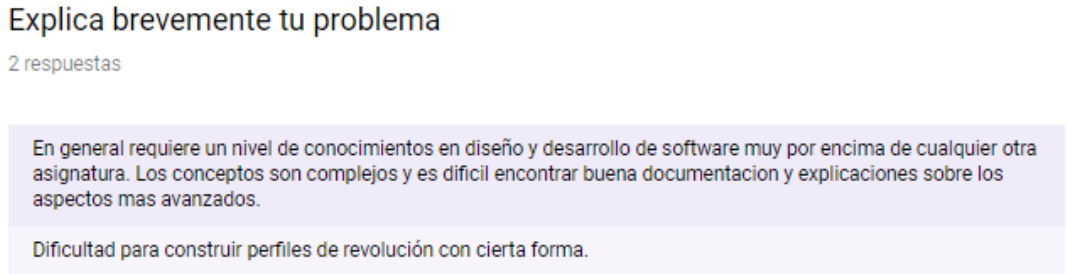


Ilustración 1.4.1.5

Dado que los alumnos creen que sus mayores obstáculos en la asignatura son los temas de perfil de revolución y luces, creemos que nuestro trabajo en este prototipo se debe centrar en dichos temas, además, intentaremos buscar una forma interactiva de mostrar qué se intenta conseguir en esos temas y ayudar a llegar a esos resultados programándolos ellos mismos.

1.4.2 Resumen de las deficiencias y carencias identificadas

Dentro de la propia asignatura vimos que los alumnos no sabían detectar sus propias dudas sobre el temario, por ejemplo, algo que creo que es muy común en todas las asignaturas, es que los alumnos no preguntan las dudas ya sea por falta de asimilación o por factores externos a la docencia, pero en cuanto se ponen a estudiar buscan alternativas donde buscar las dudas que le han surgido.

Por eso mismo, decidimos dar un paso con esta aplicación, donde podrán ver el temario de un modo más ameno al método tradicional, creemos que esta herramienta de ningún modo debe o pueda reemplazar las clases ni las tutorías, pero puede ser un complemento muy potente para asimilar los conceptos o incluso de entrada previa al temario.

Con esta nueva herramienta software que planteamos creemos que podemos suplir este tipo de carencias.

También creemos que es una buena herramienta inicial antes de usar herramientas de comprobación como las vistas anteriormente.

1.5 Requisitos iniciales

Los requisitos iniciales para esta aplicación son:

- Debe ser fácil de usar.
- Debe tener una parte teórica.
- Debe tener interacción con el usuario.
- Debería mostrar ejemplos dinámicos.
- Debe dejar que el usuario mueva la escena.

1.6 Alcance

Una vez finalizado nuestro trabajo se obtendrá:

- Prototipo de la aplicación
- Guía de usuario
- Manual de instalación
- Encuesta inicial

1.7 Hipótesis y restricciones

El TFG se define como una asignatura de 12 créditos, lo que supone que la duración total del proyecto será de 300 horas, incluyendo todas las etapas del ciclo de vida, con la excepción del mantenimiento. Por consiguiente, la principal restricción aplicable es la limitación de la duración del trabajo.

Como debe durar aproximadamente 300 horas, hemos estimado que la duración ideal sería 3 meses, que sería el trabajo a tiempo parcial de un ingeniero informático.

Dado que nuestra aplicación está pensada para uso universitario hemos decidido que sea software libre por lo que debemos evitar coste elevados en su producción, por lo que es recomendable usar recursos y aplicaciones externas que tengan un tipo de licencia similar en medida de lo posible.

1.8 Estudio de alternativas y viabilidad

Hemos tenido en cuenta diferentes alternativas para el desarrollo de este prototipo, pero todas tenían inconvenientes más grandes que las ventajas que aportaban, en los siguientes subapartados se puede ver con más detalles las alternativas estudiadas para este proyecto.

1.8.1 Primera alternativa Qt

La alternativa más clara era usar Qt por ser una herramienta muy potente para este tipo de aplicaciones, pero tenía varios inconvenientes bajo nuestro criterio, sus inconvenientes son:

- La mezcla de código OpenGL nativo con las clases de WT para OpenGL es muy problemática.
- La adaptación del código ya desarrollado.
- La adaptación del entorno del trabajo para usar la nueva librería.
- El tamaño en disco del proyecto sería mucho mayor.

No todo eran inconvenientes como es obvio, también podíamos haber decidido usar Qt por las ventajas que aporta, como crear interfaces más complejas con menor esfuerzo, pero al ser una aplicación didáctica preferíamos centrar nuestro esfuerzo en unas escenas y/o explicaciones más elaboradas que en la propia interfaz de la aplicación.

1.8.2 Segunda alternativa WebGL

La otra alternativa que estudiamos, fue realizar el proyecto en WebGL, una herramienta muy potente orientada para proyectos web, pero suponía inconvenientes muy grandes:

- Adaptar nuestro código.
- La necesidad de un Servidor web para alojar el proyecto.
- Las comprobaciones en busca de errores son más complejas.

Sin embargo, esta alternativa nos daba unas ventajas únicas como:

- Funcionaría en todos los sistemas.
- No ocupa espacio en el disco del usuario final.
- Los alumnos pueden consultar la herramienta en cualquier lugar (en clase, en su casa, en la biblioteca, …).

1.9 Descripción de la solución propuesta

Después de pensarlo durante días y estudiar con detenimiento las distintas alternativas, hemos decidido seguir el proyecto con las mismas herramientas que se usa para las prácticas de la asignatura, de esto modo, además de presentarles un prototipo a los alumnos, también le mostramos un incentivo y no es otro que hacerles una demostración de que con las herramientas que van a usar se puede montar una aplicación potente, rápida y lo mejor de todo, muy ligera.

Por eso hemos decidido usar para el desarrollo de este prototipo usar C++ con OpenGL y las librerías GLEW y GLUT.

Las razones por la que hemos decidido esta solución son:

- La experiencia personal en este lenguaje y bibliotecas para la realización del proyecto.
- Con estas librerías el proyecto ocupara poco espacio en el disco.
- Creemos que el alumno asimilara mejor los conceptos si los ejemplos de código se asemejan más a lo que ellos utilizaran.
- La eficiencia del programa en cuanto rapidez y espacio necesario

Una vez finalizado el proyecto ocupa unos 6,13MB la carpeta donde se aloja todo lo necesario para su ejecución.

1.10 Tecnologías utilizadas

En este apartado se describen las tecnologías utilizadas (lenguajes de programación, herramientas web, programas de terceros, librerías de programación, …), vamos a enumerar todo lo usado para la realización del prototipo de nuestra aplicación:

1. **C++:** Lenguaje de programación usado en nuestro prototipo, no es multiplataforma, pero como la mayoría de equipos de la universidad son Windows no nos supone gran problema.
2. **OpenGL (Core profile):** Librería Grafica usada en la aplicación, decidimos usarla por varias razones, el software base está escrito en este lenguaje, el alumno comprenderá más fácil los ejemplos de programación y yo personalmente he trabajado más con esta librería que con cualquier otra librería gráfica.
3. **GLEW:** Librería de extensión a OpenGL que ayuda y facilita la realización de aplicaciones graficas con dicha librería.
4. **GLUT:** Librería de extensión a OpenGL que nos aporta funciones para el manejo de ventanas e interacciones con teclado y ratón, muy útil en nuestro proyecto puesto que queremos escenas donde el usuario tenga una enseñanza dinámica.
5. **Google Forms:** Herramienta elegida para crear el formulario para iniciar un pequeño estudio de lo que debería mostrar la aplicación, decidimos usar esta herramienta web para facilitar la búsqueda de respuesta y para evitar el coste en papel de dichas encuestas.
6. **GIMP:** Herramienta para edición y creación de imágenes, usada para la creación de las distintas texturas usadas en la aplicación, escogida por encima de Photoshop por ser un software libre y de este modo ahorrar costes en la elaboración del prototipo.
7. **Microsoft Word:** Procesador de texto usado para los borradores de las explicaciones que se presentarían en el proyecto y para la creación de esta memoria.

8. **Visual Studio 2015:** Entorno de desarrollo escogido para la creación del prototipo, escogido por su interfaz conocida por el alumno y por la facilidad de incorporar las librerías ya mencionadas.

1.11 Metodología de desarrollo de software

Para desarrollar este proyecto hemos decidido optar por una metodología de desarrollo ágil, en concreto scrum, creemos que es la mejor forma de desarrollar este tipo de software, ya que para avanzar a una etapa superior necesitas las anteriores en un rendimiento pleno y/o estable, hemos decidido tener cada sprint por semana adaptando así el tiempo de trabajo optimo a 12 sprint (aproximadamente 3 meses).

1.11.1 Scrum con 1 sola persona

Originalmente la metodología de desarrollo scrum está pensada para un equipo de desarrollo, pero hemos pensado que en este caso puede adaptarse y puede ofrecer buenos resultados con una sola persona, básicamente lo que nos convence de este estilo de desarrollo es que dividimos los problemas en problemas más pequeños para abarcar mejor una solución final.

También nos permite adaptarnos a los problemas que vayan surgiendo y con ello marcarnos objetivos a corto plazo más realistas, y gracias a que necesitaremos realizar pequeños estudios diarios, podemos saber que necesitamos priorizar y de este modo encontrar un prototipo realista a nuestros recursos y tiempo.

Por último, personalmente, creo que es una manera muy efectiva de trabajar porque en todo momento sabes cuáles son tus hitos o metas semanales, además un desarrollo ágil creo que se adapta muy bien a mí, porque tener un objetivo tan lejano suele ser más abrumador que marcarse pequeños objetivos para sentir un avance y una satisfacción cada vez que termina una etapa.

1.12 Estimación del tamaño y esfuerzo

Ya que el presente proyecto es un TFG, no existen restricciones de tipo económico, sino de tipo temporal (un número aproximado de horas). Por consiguiente, los cálculos de tamaño del proyecto están supeditados el tiempo disponible. En cuanto al esfuerzo, se dispone de tan un solo efectivo (la persona autora del trabajo).

Se va a usar la metodología básica de SCRUM para estimar el esfuerzo las pautas de esfuerzo.

1.13 Planificación temporal

Actividad	Inicio	Final
Preparaciones previas	25/03/2019	01/04/2019
Iteración 1	01/04/2019	08/04/2019
Iteración 2	08/04/2019	15/04/2019
Iteración 3	15/04/2019	22/04/2019
Iteración 4	22/04/2019	29/04/2019
Iteración 5	29/04/2019	06/05/2019
Iteración 6	06/05/2019	13/05/2019
Iteración 7	13/05/2019	20/05/2019
Iteración 8	20/05/2019	27/05/2019
Iteración 9	27/05/2019	03/06/2019
Iteración 10	10/06/2019	17/06/2019
Iteración 11	17/06/2019	24/06/2019
Iteración 12	24/06/2019	01/07/2019

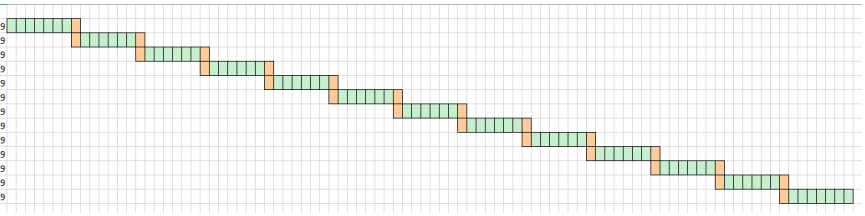


Ilustración 1.13

1.14 Presupuesto

Recurso	Tiempo de uso	Coste global	Coste Parcial
Visual Studio Community 2015	3 meses	0,00€	0,00€
Windows 10 pro	3 meses	259,00€	10,80€
GIMP	3 meses	0,00€	0,00€
Licencia Microsoft Office	3 meses	7€/mes	21,00€
Portatil msi	3 meses	970 €	48,60€
Ratón óptico	3 meses	12 €	1,50€
Encuestas e impresión	1 mes	0,00€	0,00€
1 ingeniero informático	3 meses	21.000,00€	2.625,00€
Costes indirectos	3 meses	Suplemento del 10%	270,74€
		Coste Total del prototipo:	2977,30€

Tabla 1.14.1

2 DISEÑO INICIAL

2.1 Especificaciones del sistema

2.1.1 Requisitos no funcionales

En este apartado vamos a enumerar los aspectos no funcionales mas importantes que se ven reflejados en la aplicación y algunos que podrían mejorarse:

- **Facilidad de uso:** Para facilitar la experiencia del usuario y el uso de nuestra aplicación hemos decidido que puedas seleccionar escenas con el botón izquierdo del ratón o con los números (ahora mismo 1 o 2, pero más adelante tantos como numero de temas), creemos que ambas posibilidades son buenas porque aunque una asignatura puede tener más de 9 temas, no

tiene por qué mostrarse en una aplicación para el apoyo de la enseñanza de forma interactiva todos los temas, por ejemplo, temas como la introducción pueden ser obviados en este tipo de aplicación, también si el usuario necesita pulsar una tecla para ver de forma iterativa el proceso de una escena, se le explicará en dicha escena, por lo que afirmo sin miedo a equivocarme que es una aplicación fácil de usar.

- **Eficiencia:** Sin lugar a dudas es uno de los puntos fuertes de esta aplicación, pues tiene una respuesta muy rápida con los resultados que pide el usuario, su navegación entre escenas y el menú es muy fluida por lo que se puede afirmar que es una aplicación muy eficiente en cuanto a ejecución se refiere.

  Por otro lado, en cuestión a espacio que ocuparemos en nuestros equipos, solo cabe recordar que este prototipo ocupa 6.34MB en disco, es un tamaño que podría considerarse diminuto en cualquier dispositivo de hoy en día por eso mismo en un ordenador este tamaño puede considerarse que es una aplicación muy eficiente para lo que puede llegar a ofrecer.

- **Seguridad:** En nuestro prototipo admitimos que no hemos tenido muy en cuenta la seguridad pero también creemos que no es necesario en este tipo de aplicaciones, porque al ser una aplicación en la que ni tratamos datos sensibles ni personales, es más, no guardamos datos de nuestros usuarios finales, lo único que se podría destrozar es la experiencia y la enseñanza de los alumnos, lo cual es contraproducente, pero no estaría de más en futuras iteraciones para la realización final del proyecto, pensar en un método para evitar posibles cambios en los textos, creo que con una comprobación inicial sería suficiente para tener una aplicación segura y robusta.

- **Privacidad:** Como se ha dicho en el punto de seguridad, esta aplicación no guarda datos de los usuarios finales por lo tanto no tenemos que proteger sus datos, se podría decir que el uso de la aplicación es anónimo porque no se pide un usuario y contraseña para su uso.

2.1.2 Requisitos funcionales

- La aplicación debe visualizar objetos 3D con mallas de triángulos junto a una breve explicación.
- El usuario debe tener la posibilidad de modificar partes de la escena para ver el resultado en los objetos.
- El usuario debe poder usar la aplicación de forma fluida, sin tener un gran periodo de aprendizaje sobre la misma.

2.2 Análisis y diseño del sistema

2.2.1 Diseño arquitectónico del sistema

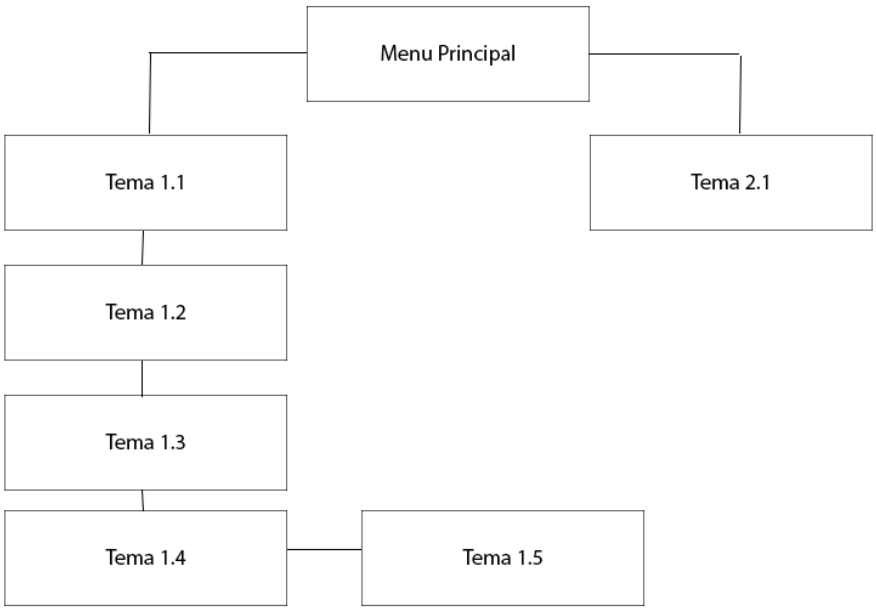


Ilustración 2.2.1

2.2.2 Diagrama de clases

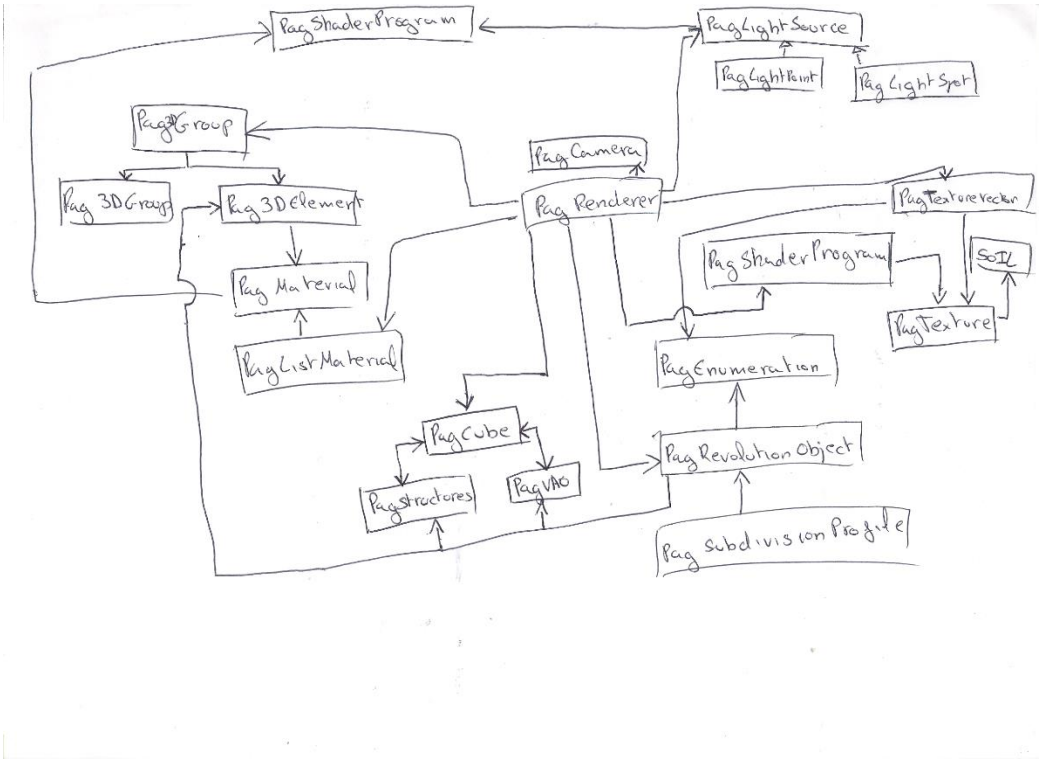


Ilustración 2.2.2

2.2.3 Diagrama de caso de uso

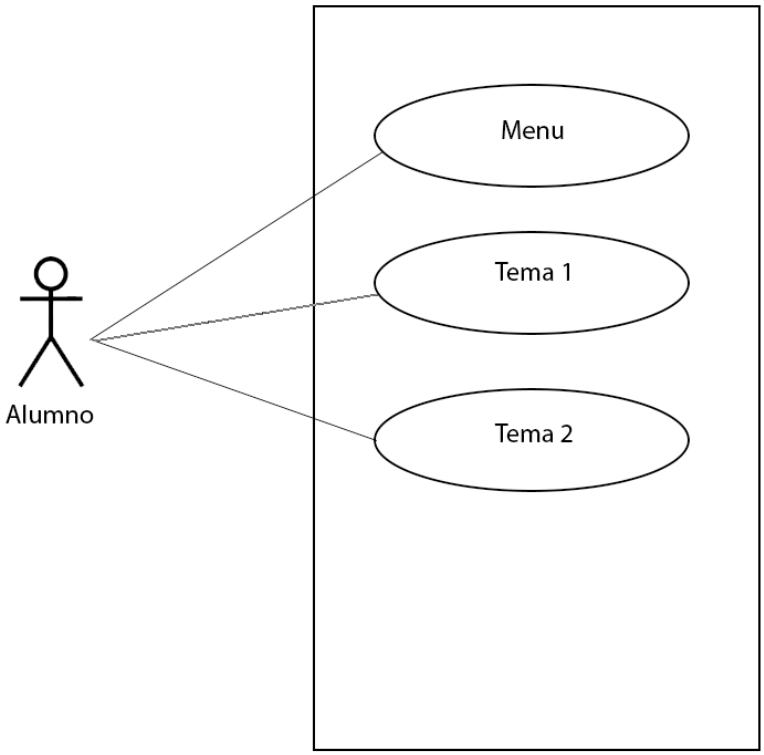


Ilustración 2.2.3

2.2.4 Diagrama de secuencia

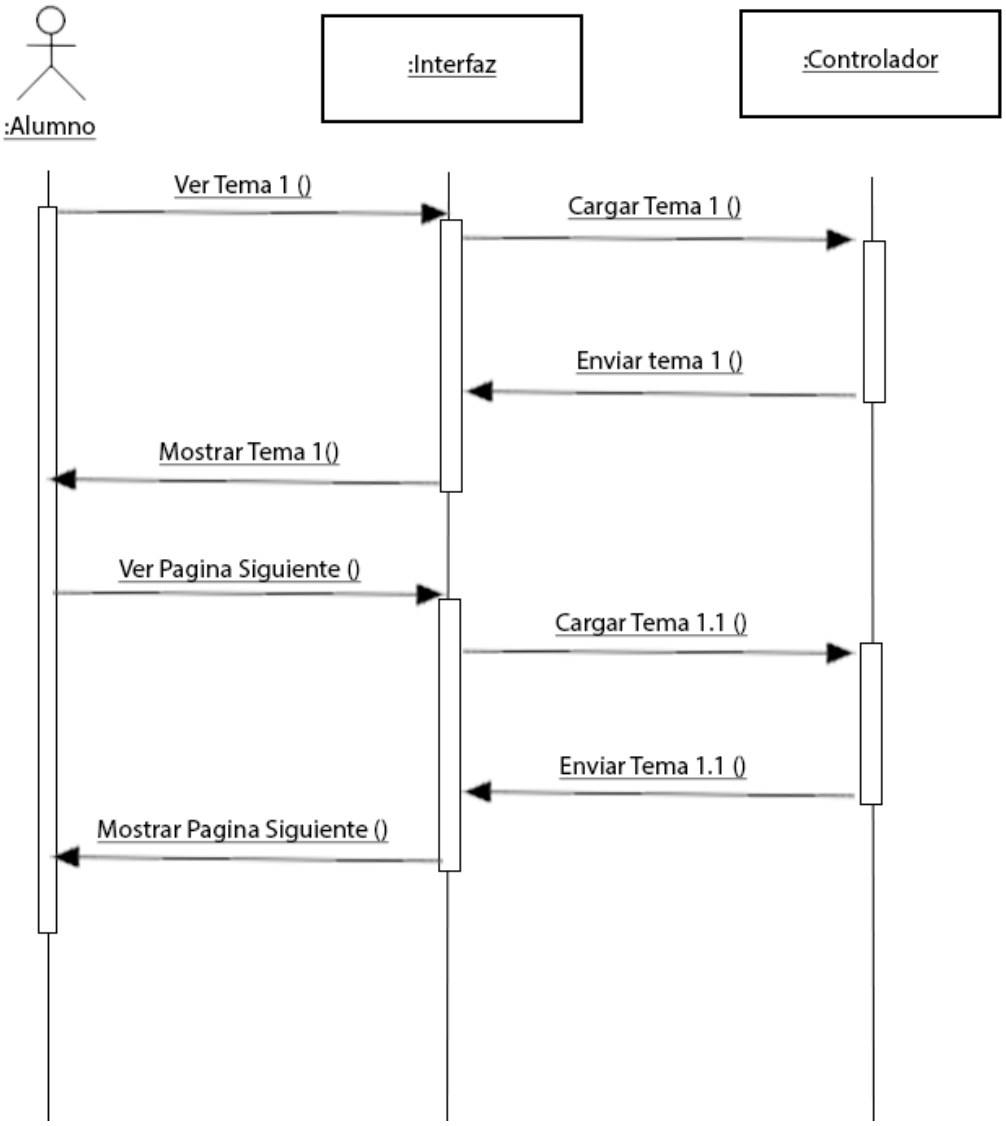


Ilustración 2.2.4

2.2.5 StoryBoard

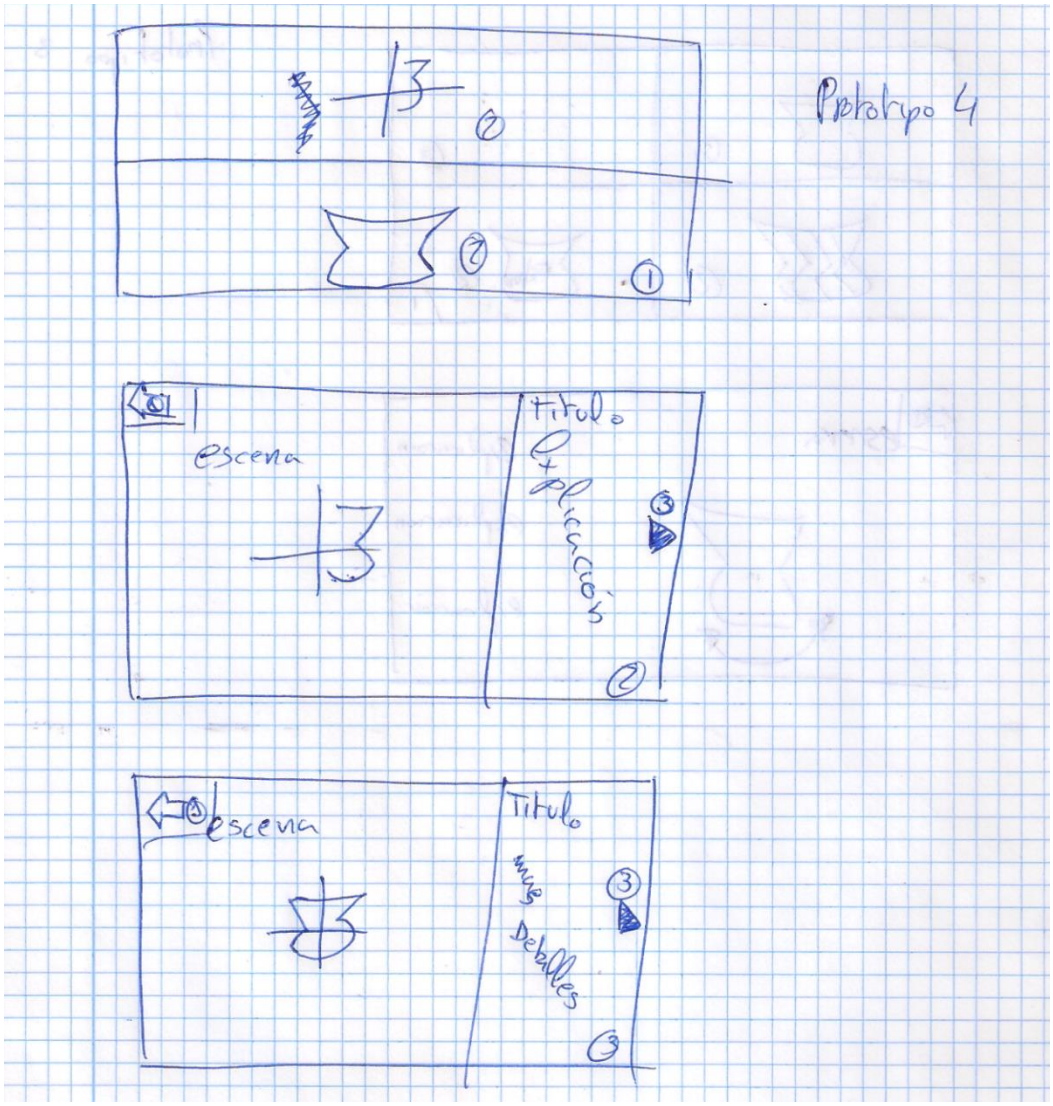


Ilustración 2.2.5

3 DESARROLLO

3.1 Primera Iteración

En esta semana se elaboraron todos los storyboards salvo el número 4 y se investigó la forma de elaborar el menú del storyboard número 3.

Para esta primera etapa pensamos en varias opciones, una de ellas fue implementar este proyecto con Qt ya que es una librería grafica muy potente y aporta un gestor de ventanas excelente, pero su mayor desventaja era que al ser un recurso tan potente también es muy costoso en cuanto a tamaño de los proyectos se refiere, También bajo mi punto de vista como las practicas se realizan con GLUT y GLEW en cierto modo, estamos demostrando al alumno que todo lo propuesto en esta aplicación lo pueden hacer sin problema y al darnos cuenta de este importantísimo detalle descartamos cualquier otra opción encontrada como cambiar de lenguaje.

Por ejemplo, ahora que está terminado el prototipo, podemos observar que todo lo necesario para ejecutar la aplicación apenas llega a 6MB lo que no supone ningún esfuerzo para ningún ordenador, creo (si se me permite), que, gracias al uso de estas librerías, hemos conseguido una aplicación ligera, rápida y adaptable a cualquier equipo.

Por lo que decimos seguir con C++, con el uso de las librerías GLUT y GLEW, gracias a lo visto en dicha asignatura decidimos recurrir a usar múltiples viewport para fabricar el menú y sus escenas, antes de llegar a la segunda semana me dio tiempo a implementarlo, pero solo visualmente por lo que decidí poner las siguientes metas para la segunda semana:

-Implementación de una escena de prueba, incluyendo un texto.

-Adaptación del menú para hacer que el tamaño de este fuese dinámico respecto al tamaño de la ventana de la aplicación.

3.1.1 Primer Storyboard

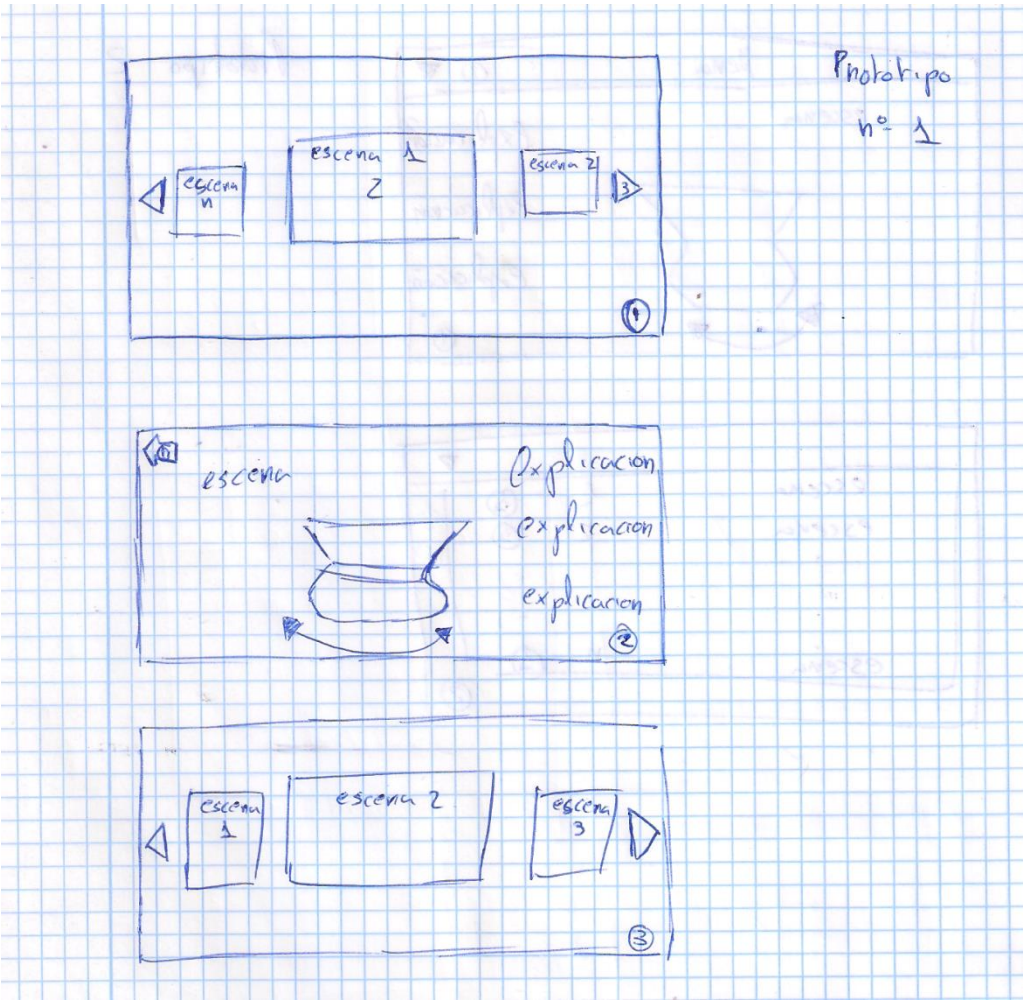


Ilustración 3.1.1

En este primer storyboard, busque un diseño bonito, de fácil entendimiento ya que me basaba en otras aplicaciones que había visto para intentar facilitar el uso de la aplicación, pero muy costosa en tiempo, ya que decidí hacer la aplicación solo con C++ y OpenGL, lo que supone que llegar a una interfaz tan elaborada me haría perder tiempo para las partes más técnicas.

3.1.2 Segundo Storyboard

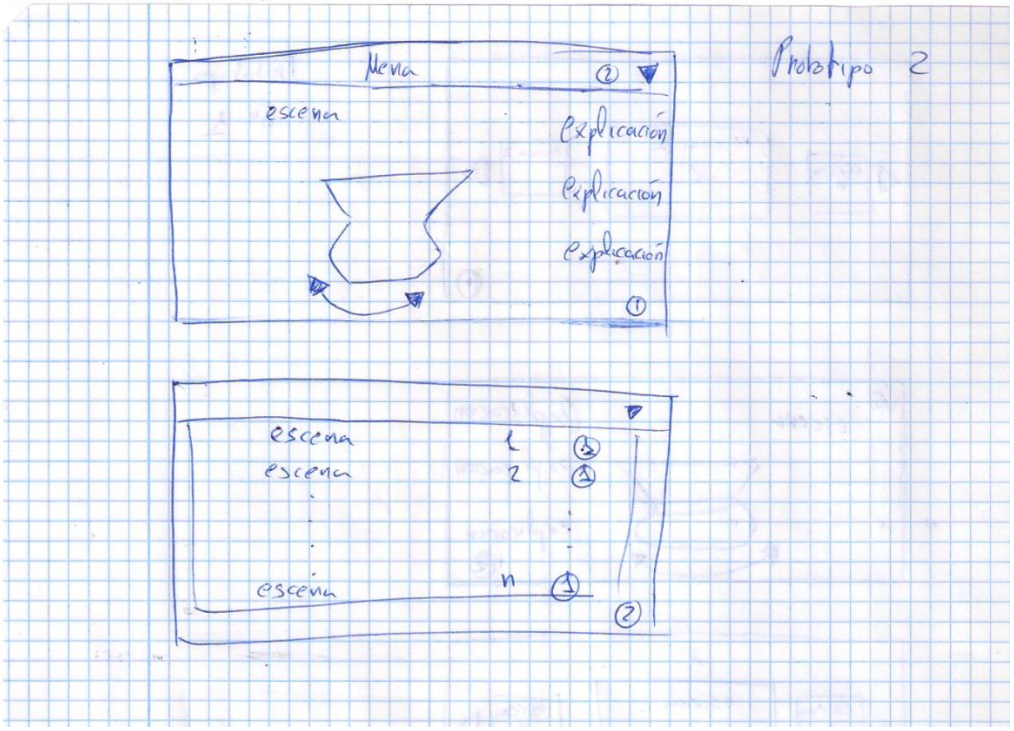


Ilustración 3.1.2

En este segundo storyboard, busque simplificar el primero, pero seguir buscando algo similar a lo ya visto en otras aplicaciones, el mayor problema que vi en este prototipo fue que antes de seleccionar la escena no se tenía una vista previa de la escena y creo que para un alumno que no sepa que busca puede ser algo lioso, por lo que decidí buscar un formato que le permitiese mirar una escena previa antes de visualizarlas.

3.1.3 Tercer Storyboard

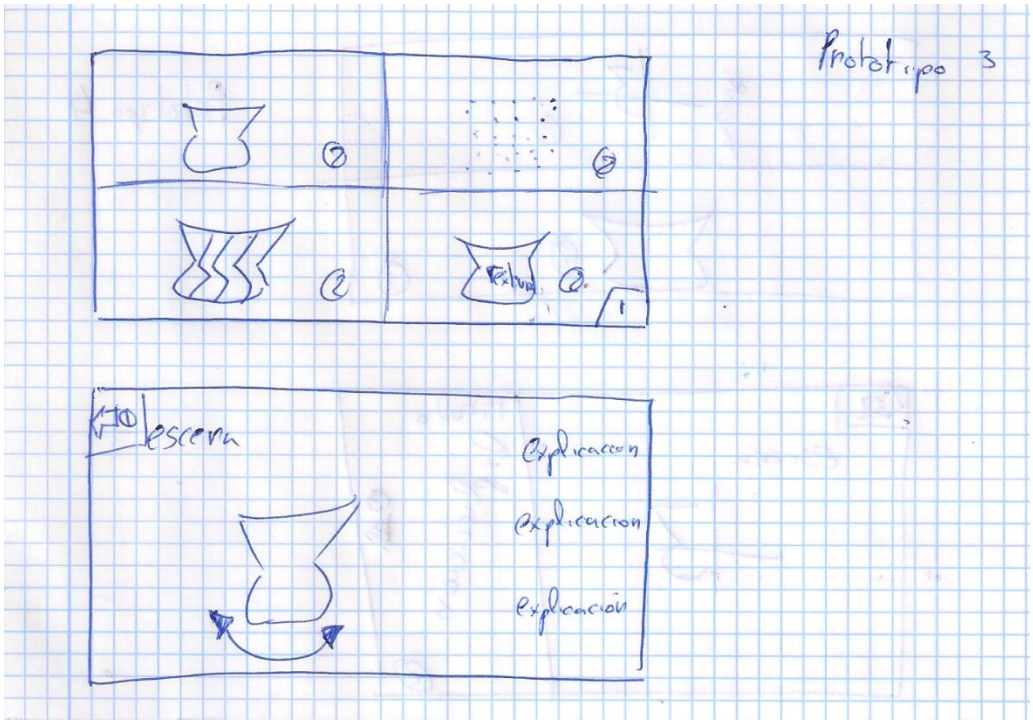


Ilustración 3.1.3

Storyboard casi definitivo, ya que encontré lo que considero el punto medio entre complejidad y usabilidad del usuario, la idea con este prototipo era mostrar 4 escenas y una vez pulsamos en una explicar lo que debería comprender el alumno.

El problema que vimos una vez empezó el desarrollo es que teníamos poco sitio donde explicar temas que normalmente suelen ser complejos, por lo que necesitábamos más hueco donde exponer la idea principal de cada tema y desarrollarlos.

3.2 Segunda Iteración

Intente implementar añadiendo el texto desde OpenGL a la escena, pero se volvía muy engorroso y no encontraba ninguna forma de poder hacer, aunque fuese un poco, dinámica esta integración, ya que para cada escena el texto cambia.

Hablando con el tutor decidí implementar dentro de cada escena dos viewport, en el primero mostraríamos la escena que deseamos que vea (y en algunos casos que interactúe), y en el segundo viewport hacer un objeto rectangular donde pegar una textura la cual será el texto que corresponde a la escena.

Por culpa de este imprevisto, no se pudo completar durante esta semana la implementación de una escena de prueba, pero se modificó el código para evitar fallos de refresco, para ello restructuramos la función `keyPressCallback`, en la cual teníamos el código necesario para dibujar la escena según se pulsase una u otra tecla, y se pusieron solo variables booleanas, gracias a esto en esta función solo dejamos marcado que y como debe dibujarse la escena, y en la función `refreshCallback` que como su nombre indica es la indicada para redibujar los objetos de la escena, pasamos y adaptamos el código que había en la función anteriormente mencionada, con este cambio conseguimos además de corregir el fallo por el cual no se refrescaba la escena, un mejor entendimiento del código.

También adaptamos los valores de los viewport del menú para que fuesen dinámicos y que cambien en función al tamaño de la ventana.

Para la siguiente iteración se propusieron los siguientes apartados:

-Implementación de una escena de prueba, con dos viewport y una textura con texto de prueba.

Adjuntamos a este momento como se ve el menú:

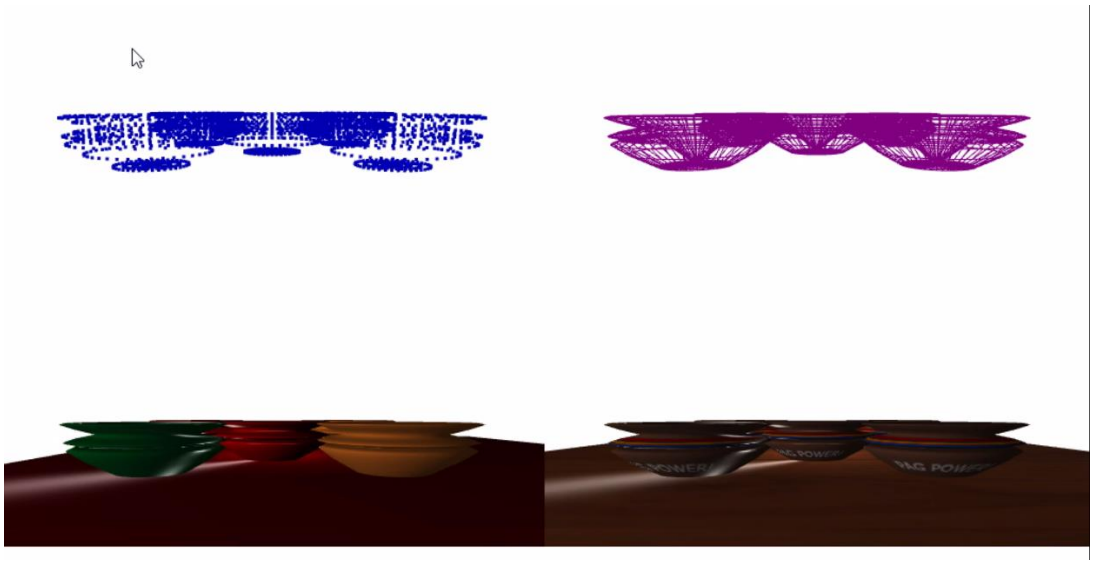


Ilustración 3.2

3.3 Tercera Iteración

En esta iteración he conseguido implementar una escena de prueba con su texto, el problema encontrado ha sido que al usar la misma cámara para todos los viewport, cuando el usuario final quiera mover la escena para ver los detalles y comprenderla, también mueve el texto dejándolo ilegible.

He implementado la transición entre el menú y la escena de prueba, por lo que ahora cada escena tiene visualmente lo que debe mostrar, faltaría el texto correspondiente a cada escena, Esta transición se puede hacer tanto con eventos del botón izquierdo del ratón o con los números del 1 al 4, los cuales cada uno corresponde con las escenas del menú.

Se ha implementado una salida de la escena para volver al menú en la esquina superior izquierda con el botón izquierdo del ratón o bien con la tecla 0 del teclado.

Para la siguiente iteración se han propuesto los siguientes objetivos:

-Implementar dos cámaras, una para las escenas y otra para el texto.

-Solo si se consigue el primer objetivo, empezar a redactar de forma amigable para el usuario final la lección del primer tema.

3.4 Cuarta Iteración

Se han implementado dos cámaras, una fija para el texto (cuando es necesaria) y otra para ver las escenas, las cuales comparten cámara para poder observar los detalles de estas.

Se ha pensado como explicar la primera escena (perfil de revolución y subdivisión de polilíneas) y adaptarlo al tamaño original de la aplicación para que sea legible y este es el resultado:

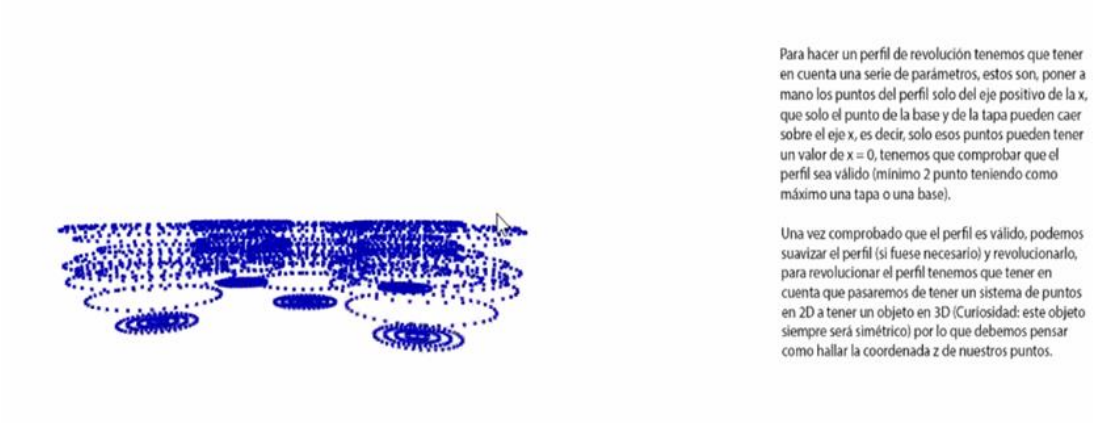


Ilustración 3.4

3.5 Quinta Iteración

He mostrado al tutor los avances y hemos llegado a la conclusión de que no es suficiente con una escena para cada tema, por lo que se ha decidido implementar un número similar de escenas, pero con menos temas, para que se comprendan y sirvan como un apoyo más sólido a los usuarios finales.

Por lo tanto, en esta iteración, hemos desarrollado un nuevo storyboard, una vez planteado y revisado, se han dado los primeros pasos en la modificación de la aplicación, viendo que el prototipo se ha ido complicando de manera significativa, se ha decidido hacer un github para tener una copia de respalda para las futuras actualizaciones.

Para la siguiente iteración se plantean los siguientes objetivos:

-Poner un método para pasar entre escenas.

-Buscar un método para no tener que retocar todo el rato el código cuando se añada algo a la escena (una textura, un objeto, una luz, etc.).

3.5.1 Cuarto Storyboard

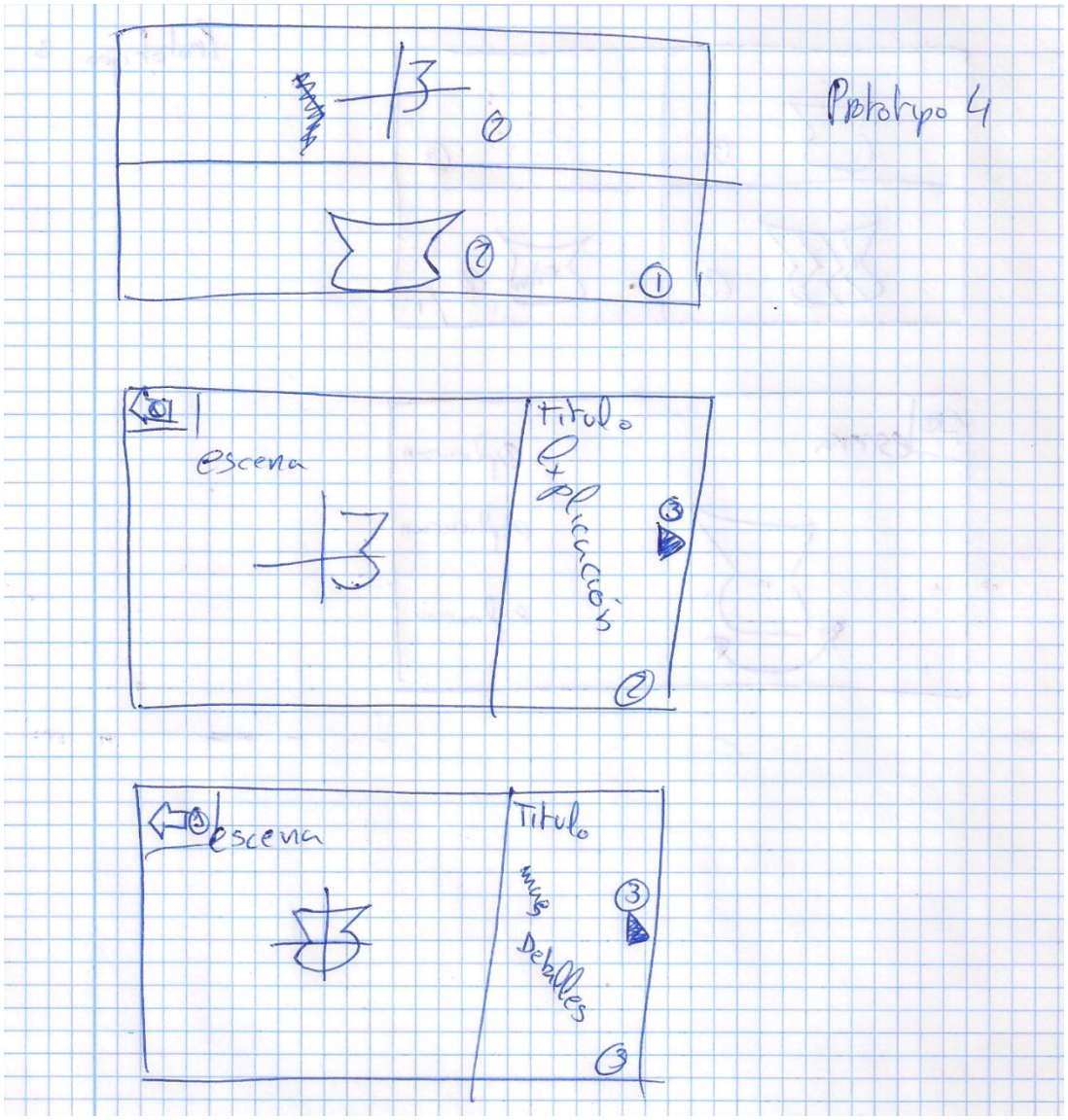


Ilustración 3.5.1

En este último storyboard, decidimos simplificar nuestro prototipo de aplicación a dos temas, dado que dentro de cada tema tenemos entre 3 y 5 escena donde explicamos cada tema, por lo tanto, estamos doblando el número de escena que íbamos a tener al principio.

Lo que más nos gusta de este prototipo es que, siguiendo la sencillez que íbamos buscando, podemos desarrollar explicaciones más compleja y mayor interacción con el usuario.

3.6 Sexta Iteración

Hemos adaptado las texturas ya creadas para poner un símbolo que recuerda a pasar página, he decidido coger uno que se asemeje a las demás aplicaciones que he visto, para que el usuario final no necesite un aprendizaje previo, ahora cuando pulsas con el botón izquierdo del ratón a la zona derecha de la ventana de la aplicación pasará a la siguiente escena del tema, como se planteó en el storyboard.

El segundo objetivo marcado se ha conseguido, ya que revisando el código varias cosas podían agilizarse y automatizarse creando unos bucles según el número de objetos que se desean dibujar y añadiendo las texturas del texto como otra escena aparte, con esto hemos podido guardar el número de textura que deseamos poner y así automatizar el proceso, por lo tanto, cada vez que pulsamos para cambiar de escena el contador de la textura de los texto aumentara en uno, por lo que es muy importante desarrollar todo el tema seguido, para no que todas las texturas estén en orden.

Adjuntamos en este momento dos imágenes para mostrar un ejemplo de cambio de escena dentro de un mismo tema.

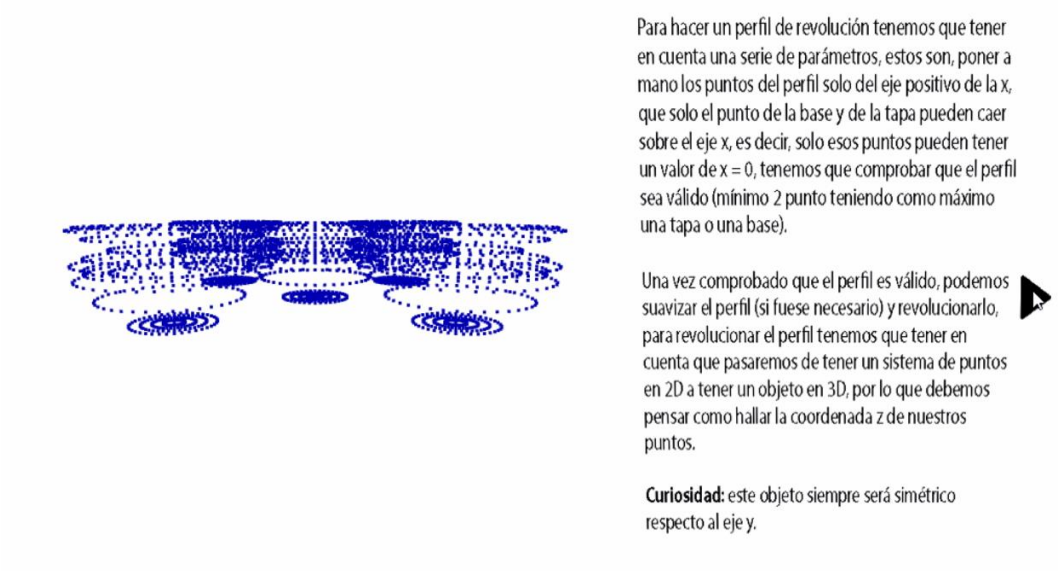


Ilustración 3.6.1

Ilustración 3.6.2

Los objetivos para la siguiente iteración son:

-Crear una segunda escena para el tema1.

-Hacer que dicha escena sea dinámica para facilitar el aprendizaje del usuario.

3.7 Séptima Iteración

He realizado la textura para la segunda escena del tema uno, en esta escena intentamos demostrarle de forma visual al alumno como se revoluciona un perfil según el número de slices que se usen, por eso le hemos puesto un texto ameno pero explicativo de cómo se consigue revolucionar el perfil y una tecla (en concreto la tecla D) que al pulsarla aumentara en 1 el número de slices del objeto presentado, al ser un perfil dibujado como una nube de puntos se puede observar perfectamente como con cada aumento de sclices se redibuja el objeto haciendo amena la enseñanza.

Por si no me he explicado bien adjunto dos imágenes de la escena, uno habiendo pulsado la tecla D una sola vez (slices = 2) y otra con la tecla D pulsada seis veces (slices = 7).

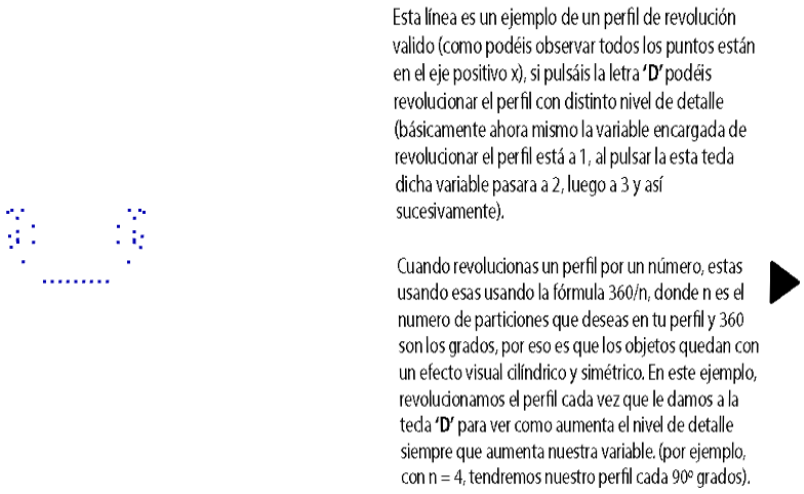


Ilustración 3.7.1

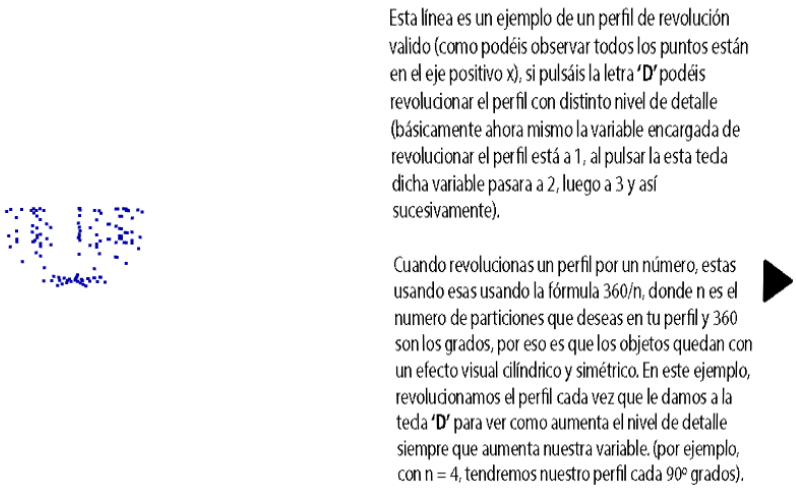


Ilustración 3.7.2

Los objetivos para la siguiente iteración son:

-Hacer la siguiente escena del primer tema

-Buscar una forma de explicar la parte de programación que conlleva revolucionar un perfil dado que es el problema más pedido por la encuesta.

3.8 Octava Iteración

En esta iteración conseguimos hacer la escena relativamente pronto, ya que decidimos adaptar la enseñanza con nuestro código, generando de esta forma un pseudo código más cercano a C++, lo que queremos conseguir con esto es que sea más fácil para el alumno el entendimiento de la parte de programación de la asignatura.

Adjuntamos una foto de la escena para entender cuál es nuestra intención con este tipo de escenas:

Supongo que os estáis preguntando, ¿Cómo pasamos de un perfil 2D a una serie de puntos en 3D? La respuesta es simple cuando revolucionamos un perfil, tenemos que calcular las nuevas coordenadas del eje x y del z, para esto ¡usamos los valores de x que ya tenemos!

veamos un ejemplo (simplificado) del código:

```
for (int i = 0; i < perfil.size(); i++) //calculo de los nuevos puntos
{
    for (int s = 0; s <= slices; s++)
    {
        a = s * delta * π / 180;
        x = perfil[i].x * cos(a)
        z = perfil[i].x * -sin(a));
        //faltaría hacer un nuevo punto con esta nueva x , la y que ya
        tenemos y la nueva z
    }
}
```

Donde delta es 360/n, y la posición del eje y no varia en nuestros puntos, por lo que con este bucle podrás conseguir los puntos que has visto en nuestro ejemplo (si haces el mismo perfil).

Ilustración 3.8

Una vez visto el resultado, le planteamos un poco de teoría, y le ponemos el ejemplo con código, buscando de este modo que se sienta dentro de la enseñanza virtual que le estamos aportando, gracias a haber mostrado primero una escena más dinámica con la que puede interactuar, esperamos conseguir que esta escena sea mucho más interesante para él, ya que conseguirá entender como se ha generado el objeto de la escena.

Los objetivos para la siguiente iteración son:

-Hacer una escena dinámica para ayudar a la comprensión de como generar la subdivisión de un perfil.

3.9 Novena Iteración

He realizado una escena similar a la escena dinámica de como revolucionar un perfil, pero para la explicación de cómo subdividir los puntos de un perfil para tener más detalle, al ser un tema más extenso que el de revolucionar un perfil, hemos tenido que quitar algunas partes por el bien de buscar resumen completo pero ameno en su lectura.

Para mostrar la diferencia que tiene en el perfil, hemos implementado un zoom a la cámara con el botón derecho del ratón (herramienta que podrán usar los alumnos), para que vean la diferencia entre subdividir el perfil o no y que puedan comprobar que cada vez que subdividen el perfil aumenta el detalle.

Adjuntamos dos imágenes para facilitar la comprensión de la escena, la primera sin subdividir el perfil y la segunda subdividiéndolo 2 veces:

Una vez sabemos revolucionar un perfil, nos falta aprender como subdividir los puntos, ¿Qué conseguimos subdividiendo el perfil? La respuesta es muy sencilla, un perfil mucho más detallado y con eso un objeto más detallado, gracias a que suavizaremos las curvas del perfil, dicho de otro modo, evitaremos picos pronunciados.

Si presionas la letra **'S'** puedes ir observando cómo vamos ganando detalle en nuestro objeto, originalmente nuestro perfil estaba con la variable de subdivisión a 1 y conforme presiones avanzara a 2, luego a 3 y así sucesivamente.

Como podrás observar cada vez que aumentas el nivel de detalle también aumenta el número de puntos de nuestro objeto (originalmente aumentara el número de puntos del perfil), por lo que debemos buscar un punto medio entre detalle y complejidad del objeto.

Ilustración 3.9.1

Una vez sabemos revolucionar un perfil, nos falta aprender como subdividir los puntos, ¿Qué conseguimos subdividiendo el perfil? La respuesta es muy sencilla, un perfil mucho más detallado y con eso un objeto más detallado, gracias a que suavizaremos las curvas del perfil, dicho de otro modo, evitaremos picos pronunciados.

Si presionas la letra **'S'** puedes ir observando cómo vamos ganando detalle en nuestro objeto, originalmente nuestro perfil estaba con la variable de subdivisión a 1 y conforme presiones avanzara a 2, luego a 3 y así sucesivamente.

Como podrás observar cada vez que aumentas el nivel de detalle también aumenta el número de puntos de nuestro objeto (originalmente aumentara el número de puntos del perfil), por lo que debemos buscar un punto medio entre detalle y complejidad del objeto.

Ilustración 3.9.2

Como se puede observar el alumno podrá interactuar hasta coger el nivel de detalle que desee, gracias a esto podrá ver como cada vez (sobre todo si pulsa mucha veces la tecla S) tarda un poco más en calcular la subdivisión, en este caso no tendrá muchos problemas, pero creo que cualquier alumno se dará cuenta de que si en vez de tener una escena con un solo objeto, tiene un numero mucho mayor de objetos y todos tienen un nivel alto de detalle, será una escena lenta.

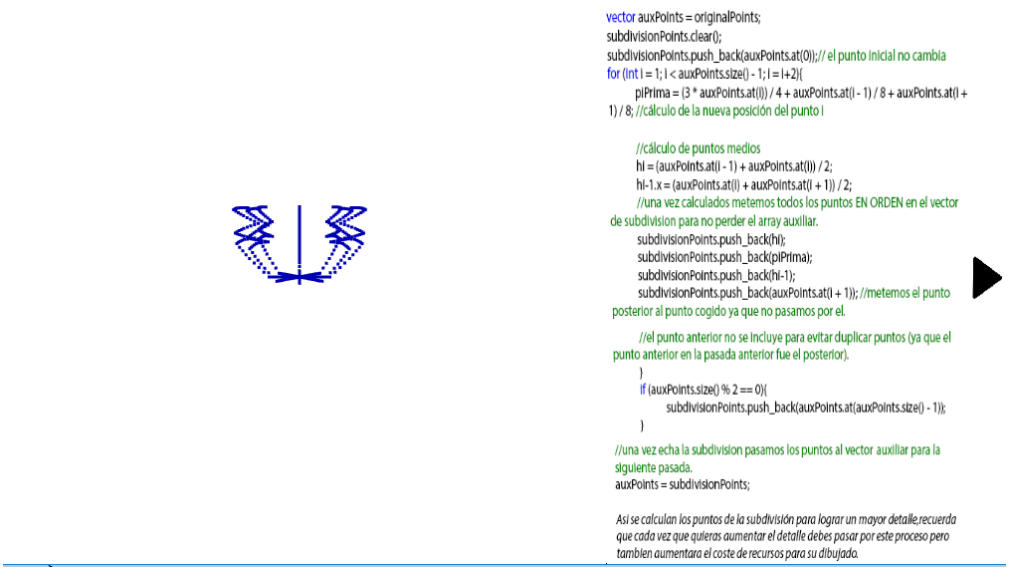
Los objetivos de la siguiente iteración son:

- hacer la última escena del tema uno, explicando cómo conseguir la subdivisión de un perfil en código
- Pensar una introducción para el tema 2 que estará enfocado en las luces.

3.10 Decima Iteración

En esta última escena del tema uno he visto complicado como poder resumir más el código, ya que al ser unas fórmulas más extensas, me era imposible quitar algo sin miedo a que no se comprendiese, por lo que se ha decidido por reducir un poco la letra para poder captar el código necesario para subdividir el perfil una vez y poner una nota al pie aclarando que se deberá hacer una pasada por ese código tantas veces como se quiera subdividir (dando a entender al alumno que necesitaran meter un bucle para poder automatizar la subdivisión).

Adjunto una imagen de la escena para ver el problema que he relatado en el párrafo anterior:



```
vector auxPoints = originalPoints;
subdivisionPoints.clear();
subdivisionPoints.push_back(auxPoints.at(0));// el punto inicial no cambia
for (int i = 1; i < auxPoints.size() - 1; i = i+2){
      piPrima = (3 * auxPoints.at(i)) / 4 + auxPoints.at(i - 1) / 8 + auxPoints.at(i +
1) / 8; //cálculo de la nueva posición del punto i

      //cálculo de puntos medios
      hi = (auxPoints.at(i - 1) + auxPoints.at(i)) / 2;
      hi-1.x = (auxPoints.at(i) + auxPoints.at(i + 1)) / 2;
      //una vez calculados metemos todos los puntos EN ORDEN en el vector
de subdivision para no perder el array auxiliar.
      subdivisionPoints.push_back(hi);
      subdivisionPoints.push_back(piPrima);
      subdivisionPoints.push_back(hi-1);
      subdivisionPoints.push_back(auxPoints.at(i + 1)); //metemos el punto
posterior al punto cogido ya que no pasamos por el.

      //el punto anterior no se incluye para evitar duplicar puntos (ya que el
punto anterior en la pasada anterior fue el posterior).
      }
      if (auxPoints.size() % 2 == 0){
            subdivisionPoints.push_back(auxPoints.at(auxPoints.size() - 1));
      }
//una vez echa la subdivision pasamos los puntos al vector auxiliar para la
siguiente pasada.
auxPoints = subdivisionPoints;

Así se calculan los puntos de la subdivisión para lograr un mayor detalle,recuerda
que cada vez que quieras aumentar el detalle debes pasar por este proceso pero
tambien aumentara el coste de recursos para su dibujado.
```

Ilustración 3.10

También como otra solución se podría aumentar las dimensiones por defecto de la ventana lo cual sería un cambio menor que no afectaría a la realización de la aplicación.

Como meta del otro objetivo, he pensado que, al ser un tema más visual, empezar con una escena dinámica que te permita tocar los distintos tipos de luces que se van a explicar, para que el alumno pueda ver desde inicio la importancia de una buena iluminación, al ser una escena con distinto tipo de interacción, será algo más compleja de llevar a cabo, pero el resultado puede ser muy positivo y llamativo para la introducción del tema.

Por lo tanto, el objetivo para la siguiente iteración es:

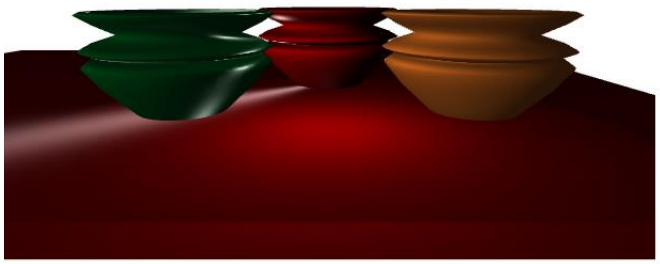
-Hacer una escena dinámica para la introducción del tema 2.

3.11 Undécima Iteración

Para esta iteración hemos montado una escena con colores sólidos, la cual cambiara con el objeto creado por el alumno si ha visto previamente el tema 1, he decidido usar colores sólidos y no una textura por varios motivos, el primero de ellos es que creo que se aprecian mejor las luces, el segundo porque en este momento de la signatura aún no se han visto la aplicación de texturas a un objeto y bajo mi punto de vista hacer un resultado similar al que el alumno obtendrá es beneficioso para que el alumno no se frustre, ya que si le mostráramos una escena más elaborada, cuando el consiguiese montar su escena puede pensar que se ha dejado algo sin hacer.

Vamos a adjuntar varias fotos de esta escena, ya que como tiene más interacción con el usuario de lo que estamos acostumbrado en las anteriores, creo necesario mostrar las distintas casuísticas que podrá experimentar el alumno con la aplicación.

Escena principal la cual vera al pinchar desde el menú a este tema:



Si quieres ver lo importante que son las luces solo pulsa la letra **'R'.**

Si has pulsado la tecla has podido comprobar que parece que se ha quedado a "oscuras", pues así de importante son las luces en nuestra escena y en este tema veremos 2 tipo de luces, la luz ambiente y la luz de tipo foco.

Puedes "jugar" con este tipo de luces en este momento, si pulsáis la tecla **'F'** veréis como se añade mas o menos en el centro una luz de tipo foco y si pulsáis la tecla **'G'** veréis el efecto de una luz de tipo puntual, es decir, una luz que viene de un punto en nuestro mundo.

¡Pero cuidado con darle muchas veces! Como consejo os diré (aunque, más adelante se comprenderá mejor) que las luces para estar a un nivel normal deben sumar todas 1 hablando en tanto por 1, pero si le dais mas de una vez a estas teclas podéis aplicar mas de lo necesario (en nuestro ejemplo estan a 0.5 pero si poneis muchas se apilan por lo que sumaran y sobrepasaran dicho valor), como podéis comprobar veremos como la escena se "quema" cada vez más, pero no os preocupéis haced toda las pruebas que queráis y si queréis restablecer las luces solo tenéis que pulsar la tecla **'R'**.

Ilustración 3.11.1

Resultado que obtendrá si pulsa la tecla R, como quita todas las luces, verá cómo se “pierde toda la escena” cuando en verdad es el resultado de quitar todas las luces, al ser el fondo blanco parece que se ha borrado, pero en verdad lo más similar es que ha apagado todas las luces de la habitación:

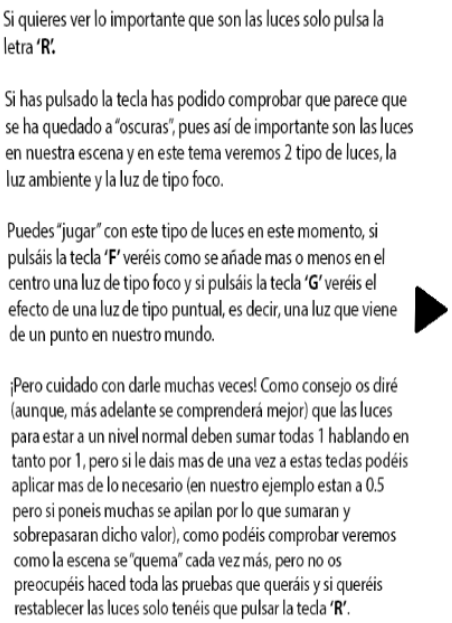


Ilustración 3.11.2

Escena pulsando primero la F, donde añadirá una luz tipo foco en el centro de la mesa, gracias a esto, como podrá ver la escena no se ha borrado pero vera la importancia de donde están las fuentes de iluminación y a donde iluminan, porque, aunque se pueden entrever los colores que tiene cada vasija, solo se ve su contorno y justo el lateral más cercano a la luz que acabamos de poner:

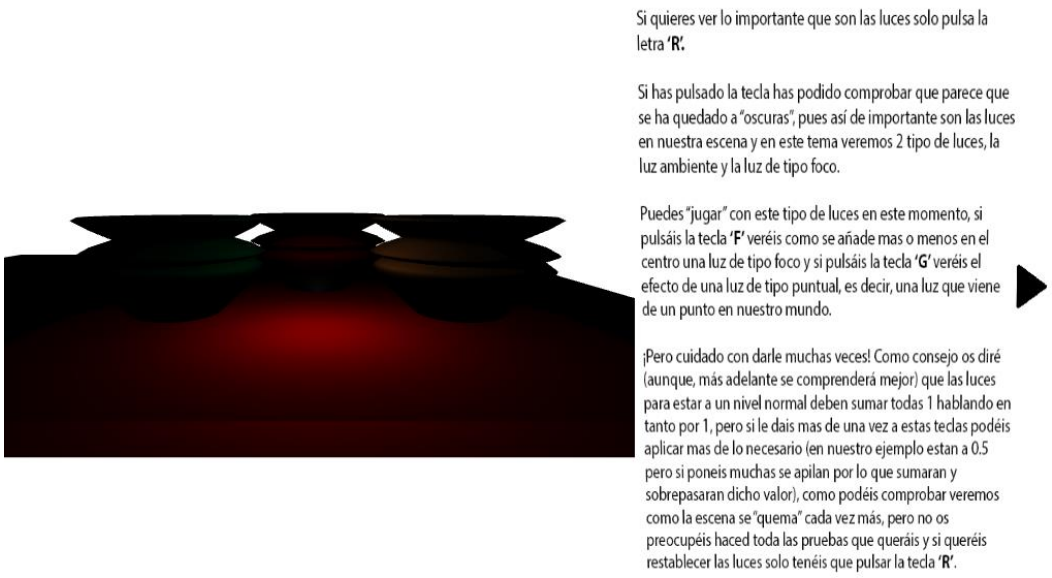


Ilustración 3.11.3

Escena pulsando la primero la ‘G’ de primeras vera una gran iluminación, pero esperamos que se dé cuenta de que falta una luz que antes estaba (obviamente como

solo tenemos dos luces, hablamos de la luz de tipo foco), con esta luz podrá observar la iluminación y el haz de una luz puntual, pero vera que la mesa, no tiene la iluminación de antes:

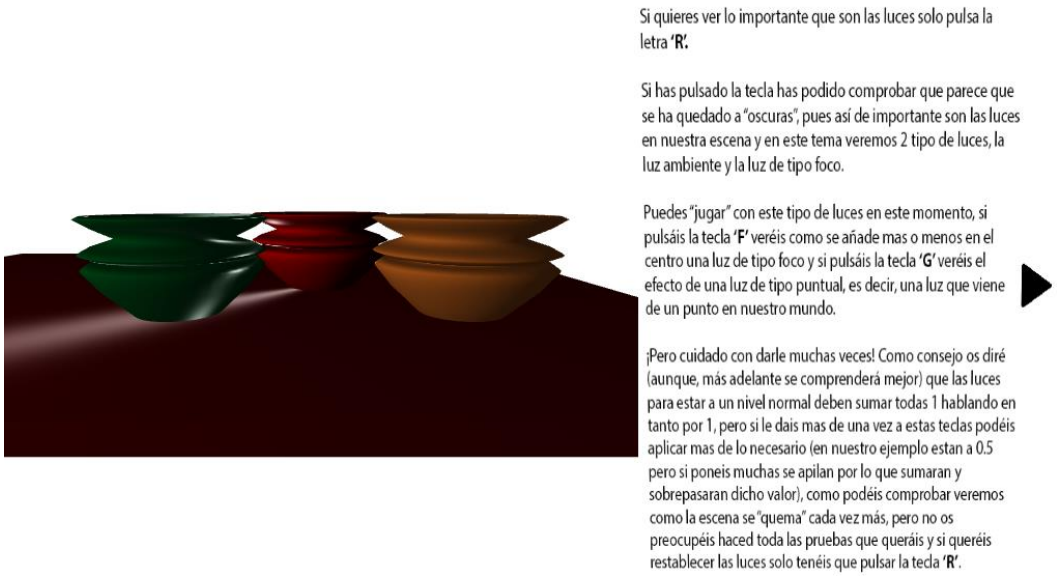


Ilustración 3.11.4

El resultado que obtendrá si pulsa solo una vez la luz tipo foco y la luz tipo puntual, es el resultado inicial como se podía esperar.

Pero lo más interesante bajo mi punto de vista en esta escena es la posibilidad de que queme la escena si pulsa repetidamente las luces, ya que es algo que normalmente te explican en clase pero al no experimentar el resultado que da, de primeras puede parecerle al alumno que una escena excesivamente saturada es el resultado ideal, pero gracias a esta escena conforme vea que va en aumento esta sensación de excesiva luz, esperamos que llegue a comprender que lo óptimo con la iluminación es encontrar un resultado que sería natural en el mundo real.

Ejemplo de una escena quemada dando repetidas veces a la luz de tipo foco y sin luz puntual, hemos dado para este ejemplo unas seis veces a este tipo de luz:

Ilustración 3.11.5

Ejemplo de una escena quemada solo con tipo puntual, también hemos dado a este tipo de luz unas siete veces (sabemos que es una exageración, pero este tipo de pruebas les suele gustar a los alumnos):

Ilustración 3.11.6

Y bueno no solo se puede quemar la escena con un tipo de luz, lo bueno de este tipo de interacción con la aplicación es que podrán combinar ambas luces tantas veces como quieran obteniendo resultados poco deseados, pero que pueden ser interesantes para comprender que es lo que no se desea obtener en una escena.

Para la siguiente iteración nuestro objetivo es muy sencillo, comprobar y en caso de ser negativo adaptar nuestra aplicación para daltónicos, ya que creemos que dejarlo para más adelante podría ser un error y se convertiría en algo mucho más costoso.

3.12 Duodécima Iteración

En esta iteración hemos comprobado que el color verde para determinados daltónicos no se diferencia demasiado del color rojo por lo que hemos decidido cambiar la vasija de este color a un color azul, que aun que hace un poco menos agradable la escena, ganamos que cualquier persona que use la aplicación sea capaz de distinguir todos los elementos de nuestra escena de ejemplo.

3.13 Pruebas finales

3.13.1 Pruebas de validación del sistema

Para este tipo de prueba debemos preguntarnos ¿Es esto lo que quería realmente el cliente? Para poder contestar a esta pregunta vamos a poner los objetivos que teníamos inicialmente y vamos a comprobar si se han cumplido o no.

- Aplicar técnicas de enseñanza alternativas para facilitar la comprensión de problemas clásicos de Informática Gráfica:

  Este apartado se ha cumplido, puesto que la mayoría de los temas desarrollados en este prototipo siempre hay alguna escena donde el usuario puede interactuar con los resultados clave de dicho tema, y una vez ha terminado de interactuar con esa escena se le presenta un ejemplo de lo que según la encuesta previa da mas problema al alumno, un ejemplo de código intentando con esto facilitar la tarea de llevar a la práctica lo aprendido teóricamente.

- Completar la formación adquirida sobre Informática Gráfica en el Grado, mediante el estudio e implementación de tecnologías avanzadas:

Este punto creemos que se ha conseguido, ya que, hemos enfocado el estudio de forma interactiva con ejemplos de lo que el usuario puede conseguir dominando los temas vistos en la aplicación, donde en todas las escenas, puede ver todos los detalles, jugando con las cámaras para ver los resultados obtenidos, además como ya hemos dicho alguna vez, esta aplicación se complementa muy bien si se usa para reforzar lo ya visto en la asignatura.

- Mostrar escenas 3D junto con explicaciones del problema:

  Este punto se ha cumplido, solo hace falta mirar los ejemplos de las iteraciones para comprobar que el punto fuerte de nuestro trabajo es este.

- El prototipo mostrará explicaciones de por qué se obtienen esos resultados:

  Creemos que el ejemplo que demuestra que este objetivo se ha cumplido son las escenas de código, las cuales muestran al alumno como conseguir estos resultados.

- Validar con estudiantes reales las mejoras en el aprendizaje que aporta el prototipo.

  Por desgracia, al terminar a estas alturas de curso el prototipo, solo hemos podido validar el prototipo con un estudiante, para esta prueba le hemos dejado la aplicación y el manual de usuario, después de probarla le hemos preguntado las siguientes cuestiones:
  Pregunta 1: ¿Es fácil navegar en la aplicación?

  El uso de la interfaz, gracias a los iconos, como la flecha, resulta intuitivo, y en la propia interfaz se van explicando las diferentes teclas que puedes usar y su efecto en la visualización de la escena. A pesar de carecer de un menú de ayuda en la propia aplicación, el manual puede complementar fácilmente cualquier duda que nos surja con su uso.
  Pregunta 2: Como esta aplicación está destinada al apoyo docente de la asignatura de programación de aplicaciones gráficas, ¿Crees que esta aplicación puede resolver las dudas básicas de estos paradigmas?

  Creo que puede ser una buena herramienta de apoyo para la docencia, gracias a que combina la explicación teórica con la visualización de los resultados, por ejemplo, en todos los aspectos del perfil de revolución, que

personalmente creo que puede ser algo complejo de entender sin ver "gráficamente" su utilidad, puede ayudar en gran medida al alumnado. Igualmente, en el aspecto de la iluminación, puede ayudar al alumno a manejarla mejor, pues en las primeras escenas es fácil que no se use correctamente, llegando a provocar el efecto de "quemado" en la escena.

Pregunta 3: Bajo tu punto de vista ¿Cómo mejorarías la aplicación?

Incluiría más temario. Por ejemplo, el tema del manejo de la cámara, fue una parte de la asignatura que me costó en su momento controlar, ya que hay que tener en cuenta muchos parámetros; o la colocación de texturas a los objetos. Posiblemente también incluiría algunas otras escenas (con otros objetos u otra disposición, por ejemplo), para que el alumno pueda ver la respuesta de los distintos aspectos que se explican, y les pueda ayudar a entender el funcionamiento. Otro punto que podría ser también interesante incluir, sería la inclusión de escenas con fallos y cómo se visualizan, para que puedan detectar fácilmente en sus propias escenas los errores que cometan y poder tener una idea de cómo solucionarlos.

Pregunta 4: si esta aplicación hubiese estado en su versión final, es decir, con todos los temas incorporados y explicados, en el año que cursaste la asignatura, ¿Crees que te hubiese ayudado ha alcanzar los objetivos de la asignatura?

Creo que sí, al comienzo de la asignatura hay una carga teórica importante para explicar los conceptos fundamentales de la asignatura, y algunos alumnos, entre los que me incluyo, nos costó entender un poco éstos conceptos hasta que no llegamos a implementarlos, por lo que una aplicación con éste propósito creo que agilizaría bastante la explicación teórica de los conceptos de la asignatura, gracias a poder ver el resultado al tiempo que se explica teóricamente su funcionamiento, y facilitaría la resolución de dudas gracias a que se puede interactuar en algunos aspectos, por ejemplo, el suavizado de un perfil de revolución, o aumentar la subdivisión de las polilíneas.

3.13.2 Validación de la usabilidad del sistema

Haciendo varias pruebas y buscando que sea eficiente para todos los usuarios decidimos en la última iteración comprobar que nuestra aplicación era usable para los daltónicos, por eso cambiamos el color verde (que se confundía con el rojo), por el color azul, que vimos que se diferenciaba bien (al menos en los dos tipos de daltonismo que hemos podido comprobar):

Imagen original:

Imagen de prueba para los daltónicos de tipo pronatopía:

Imagen de prueba para los daltónicos de tipo deuteratropia:

Con este cambio hemos intentado que la aplicación sea usable para todo el mundo, ya que los textos como son en blanco y negro no dan problema para los daltónicos, además en la explicación de los textos hemos intentado encontrar el punto medio entre ser un texto amable y sencillo pero lo más técnico posible, buscando de este modo el entendimiento de nuestros usuarios.

También hemos comprobado que se puede llegar desde cualquier estado de la aplicación al estado inicial con menos de 3 acciones (normalmente en 1) y que su comprensión es sencilla, aunque si se decide seguir con el proyecto se debería hacer una interfaz un poco más elaborada, pero sin dejar de lado un diseño universal como el que se está proyectando.

4 CONCLUSIONES Y TRABAJOS FUTUROS

Al ser un prototipo creemos que quedan muchos temas que añadir pero que hemos conseguido mostrar la esencia de la aplicación final, pero aun así queda un gran trabajo por hacer, vamos a dividir este punto en mejoras futuras y un presupuesto para la finalización de este proyecto.

Creemos que hemos completado con éxito los objetivos del trabajo fin de grado, porque hemos buscado una forma simple, pero a la vez técnica para ayudar a los alumnos a comprender la asignatura de una forma más amena, dado que pueden interaccionar con la aplicación y ver los puntos de programación que más se suelen complicar según la encuesta.

4.1 Trabajos futuros

Creo que para empezar como es obvio, lo más importante sería implementar los temas más complejos de la asignatura, todo ellos con sus correspondientes explicaciones y una escena que muestre lo más importante de esos temas.

Como propuestas para empezar seguiría con los siguientes temas:

- Añadir un par de aclaraciones en el tema de luces.
- Añadir un tema nuevo para el tema de cámaras virtuales y sus diferentes movimientos.
- Añadir un tema de texturas.

Creo que son los temas que más pidió los alumnos después de los realizados y sería muy positivo crear una primera aplicación con esos temas que engloban casi toda la asignatura.

Una vez realizados estos temas sería conveniente mejorar un poco la interfaz, aun siendo una interfaz clara y sencilla, para mi gusto se podría adornar un poco para que fuese más vistosa visualmente y de este modo más apetecible usar la aplicación.

Si una vez finalizada la primera versión final de la aplicación se quisiese seguir mejorando, añadiría casuísticas que no sean del todo correctas, pero puedan ser plausibles, por ejemplo, añadir movimientos de cámara que se pueden programar,

pero que no serían del todo correcto por generar un movimiento de cámara extraño o irreal, para que el alumno pueda ver de forma visual de este modo porque es tan importante saber qué y que debemos modificar en cada uno de los movimientos.

Por ultimo y en caso de que esta asignatura se convirtiese en una opción bilingüe sería muy interesante buscar un método de internacionalizar la aplicación al menos con los idiomas inglés y español de este modo se ampliaría mucho los usuarios potenciales de la aplicación, este último punto no se ha buscado en este prototipo porque como se explica al inicio esta asignatura en nuestra universidad (que es por ahora nuestro cliente potencial) no tiene esta asignatura en formato bilingüe.

4.2 Presupuesto Aplicación final

En este presupuesto solo hemos tenido en cuenta hasta los dos primeros apartados de trabajos futuros ya que con esos puntos finalizados creemos que se podría entender que la aplicación pasaría a la versión 1.0.

Recurso	Tiempo de uso	Coste global	Coste Parcial
Visual Studio Community 2015	1 año	0,00€	0,00€
Windows 10 pro	1 año	259,00€	51,80€
GIMP	1 año	0,00€	0,00€
Licencia Microsoft Office	1 año	7€/mes	84,00€
Portatil msi	1 año	970 €	242,50€
Ratón óptico	1 año	12 €	3€
1 ingeniero informático	1 año	21.000,00€	21.000,00€
Costes indirectos	1 año	Suplemento del 10%	2.138,30€
		Coste Total del prototipo:	23.519,60€

Tabla 4.2.1

5 APÉNDICES

5.1 Guía original del Trabajo Fin de Título

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

PROPUESTA DE TRABAJO DE FIN DE GRADO

GRADO EN:

Grado en Ingeniería Informática

ESPECIALIDAD:

Tecnologías de la Información

MENCIÓN:

Sistemas Gráficos

TÍTULO DEL TRABAJO:

Prototipo de aplicación para docencia de Informática Gráfica mediante problemas interactivos

Idioma:

Castellano

DESCRIPCION CORTA DEL TFG:

En este Trabajo Fin de Grado, se va a desarrollar un prototipo de Aplicación para el apoyo docente de asignaturas de Informática Gráfica.

El prototipo planteará a los estudiantes algunos problemas clásicos de Informática Gráfica. Se mostrarán visualmente mediante escenas 3D junto con explicaciones del problema. El estudiante podrá interactivamente explorar distintas soluciones y ver los resultados que se obtienen. El prototipo mostrará explicaciones de por qué se obtienen esos resultados.

Ilustración 5.1.1

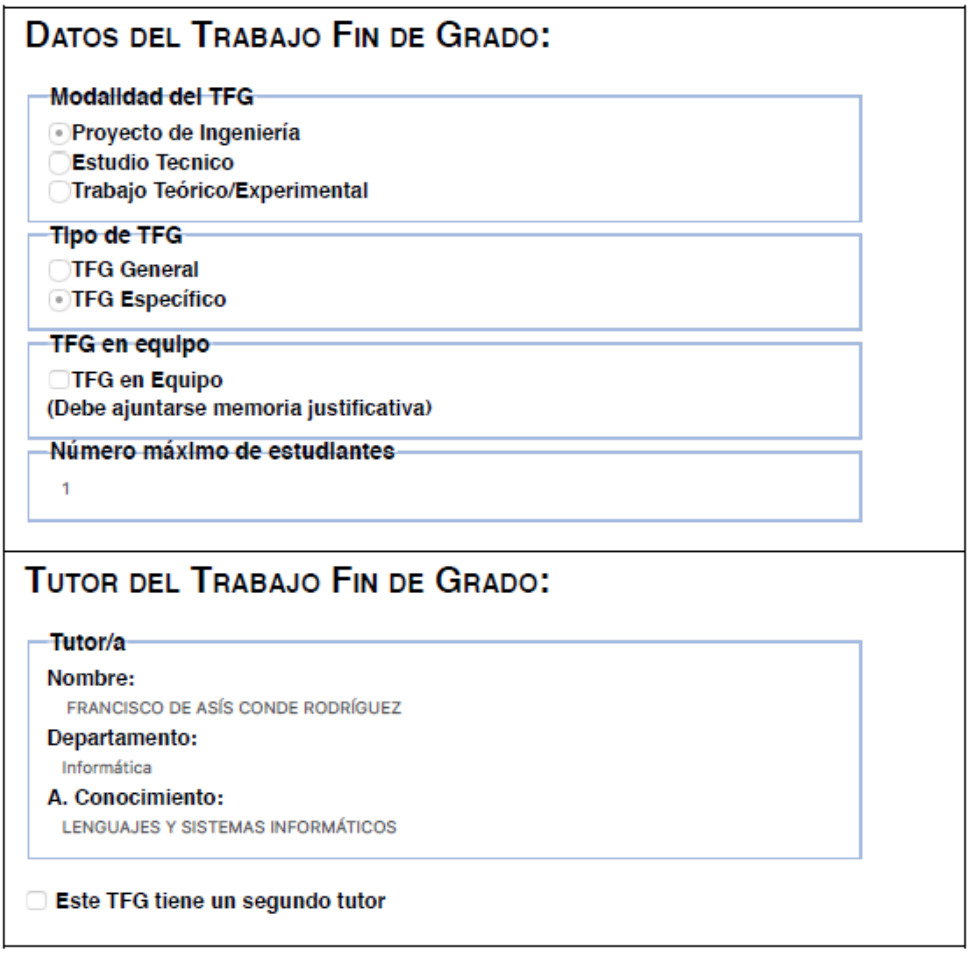
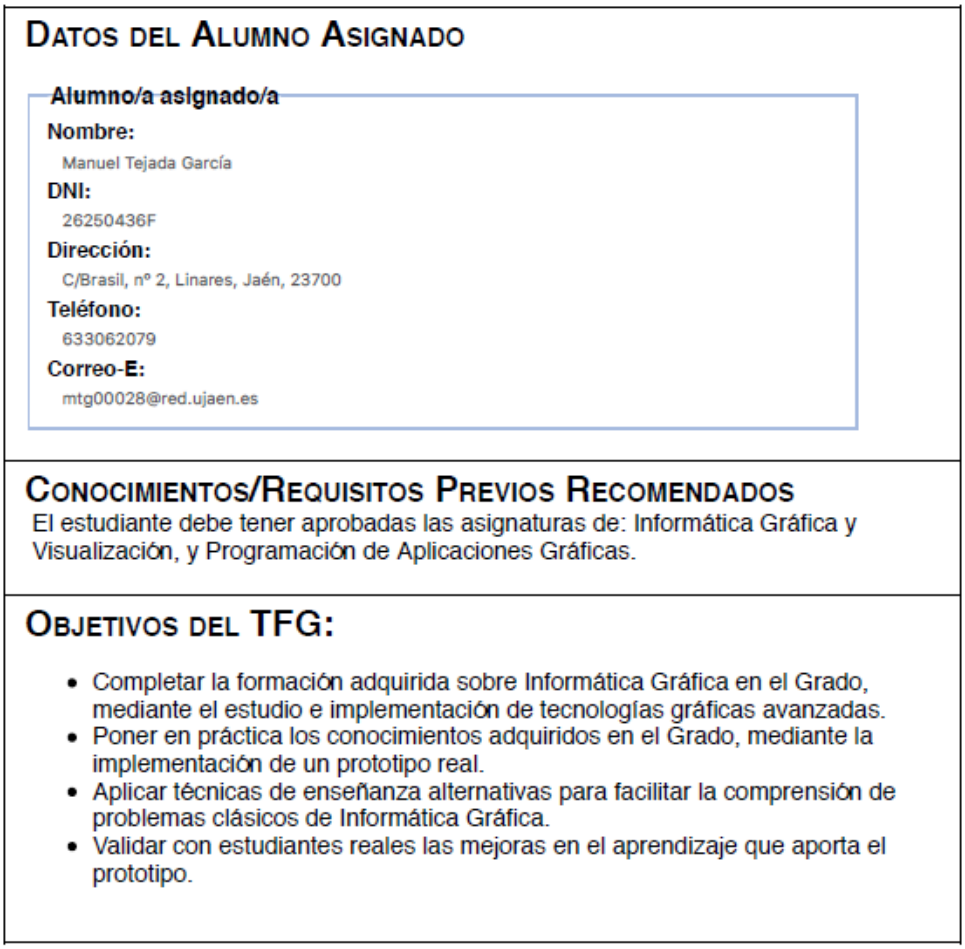


Ilustración 5.1.2



Datos del Alumno Asignado

Alumno/a asignado/a

Nombre:
Manuel Tejada García

DNI:
26250436F

Dirección:
C/Brasil, nº 2, Linares, Jaén, 23700

Teléfono:
633062079

Correo-E:
mtg00028@red.ujaen.es

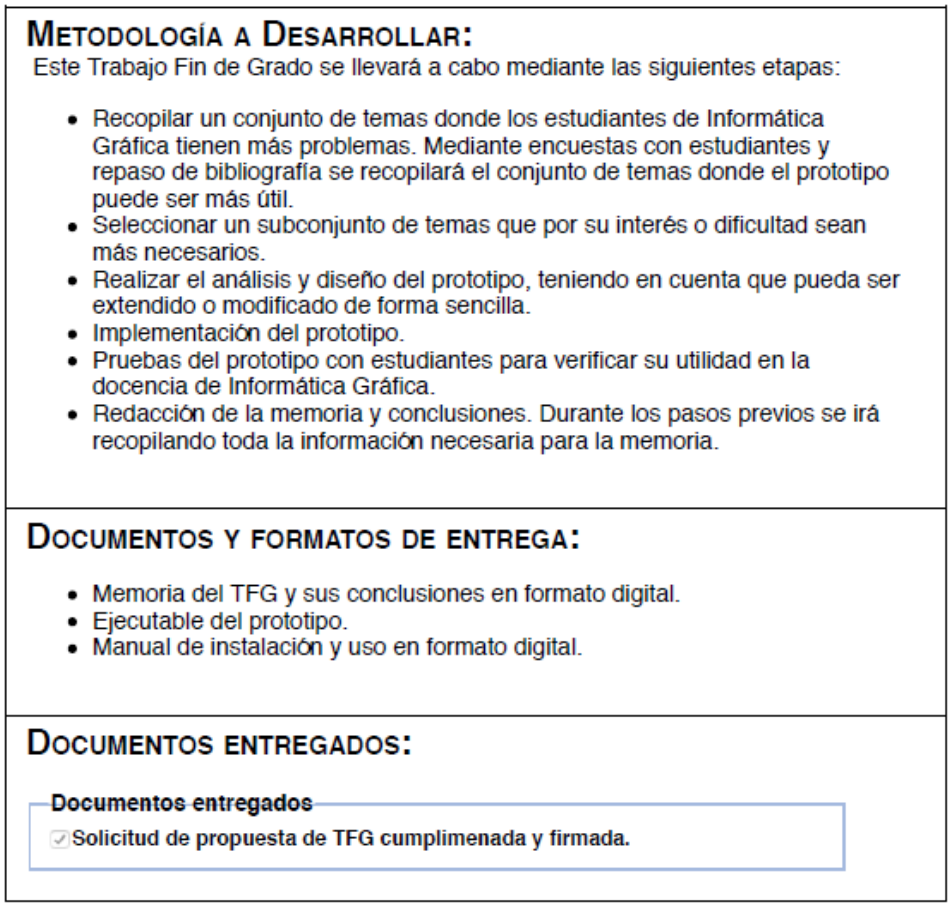
Conocimientos/Requisitos Previos Recomendados

El estudiante debe tener aprobadas las asignaturas de: Informática Gráfica y Visualización, y Programación de Aplicaciones Gráficas.

Objetivos del TFG:

- Completar la formación adquirida sobre Informática Gráfica en el Grado, mediante el estudio e implementación de tecnologías gráficas avanzadas.
- Poner en práctica los conocimientos adquiridos en el Grado, mediante la implementación de un prototipo real.
- Aplicar técnicas de enseñanza alternativas para facilitar la comprensión de problemas clásicos de Informática Gráfica.
- Validar con estudiantes reales las mejoras en el aprendizaje que aporta el prototipo.

Ilustración 5.1.3



METODOLOGÍA A DESARROLLAR:

Este Trabajo Fin de Grado se llevará a cabo mediante las siguientes etapas:

- Recopilar un conjunto de temas donde los estudiantes de Informática Gráfica tienen más problemas. Mediante encuestas con estudiantes y repaso de bibliografía se recopilará el conjunto de temas donde el prototipo puede ser más útil.
- Seleccionar un subconjunto de temas que por su interés o dificultad sean más necesarios.
- Realizar el análisis y diseño del prototipo, teniendo en cuenta que pueda ser extendido o modificado de forma sencilla.
- Implementación del prototipo.
- Pruebas del prototipo con estudiantes para verificar su utilidad en la docencia de Informática Gráfica.
- Redacción de la memoria y conclusiones. Durante los pasos previos se irá recopilando toda la información necesaria para la memoria.

DOCUMENTOS Y FORMATOS DE ENTREGA:

- Memoria del TFG y sus conclusiones en formato digital.
- Ejecutable del prototipo.
- Manual de instalación y uso en formato digital.

DOCUMENTOS ENTREGADOS:

Documentos entregados

☑ **Solicitud de propuesta de TFG cumplimenada y firmada.**

Ilustración 5.1.4

5.2 Instalación y configuración del sistema

Para instalar el prototipo de la aplicación deberemos descargar la carpeta debug facilitada con esta documentación.

Una vez obtenida la carpeta debemos descargar la carpeta glwrappers y ubicarla en el escritorio.

Dentro de esta carpeta deben estar las tres bibliotecas usadas en el proyecto.

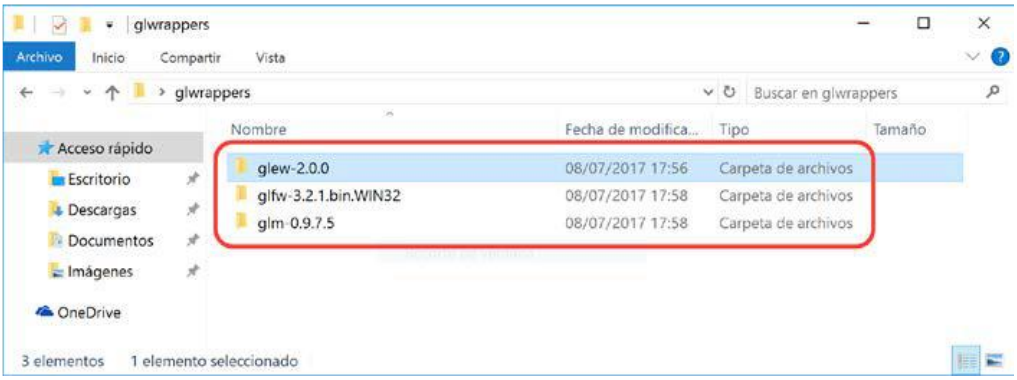


Ilustración 5.2.1

Una vez seguidos estos pasos podremos ejecutar el ejecutable de la primera carpeta y usar esta aplicación.

5.3 Manuales de usuario

Esta aplicación tiene como objetivo ayudar a comprender y facilitar la enseñanza de programación gráficas, para ello dispone de diferentes escenas con las cuales puedes interaccionar par su comprensión, dentro de ellas podrás ver un texto explicativo y que te anima a pulsar determinadas teclas para ver el avance de la escena, esta aplicación consta de varios tipos de escenas:

-Escenas dinámicas: son aquellas que te permiten interactuar con la escena para ver cómo se va construyendo paso a paso.

-Escenas introductorias: principalmente son las primeras escenas de cada tema, que como objetivo tienen darte a conocer un poco de que trata el tema y porque es tan importante.

-Escenas de código: En estas escenas podrás ver un ejemplo de la parte más importante de código de lo explicado previamente, con estas escenas esperamos

facilitar que tu como usuario final seas capaz de hacer tus propias escenas desde cero.

5.3.1 Funcionamiento

Para el funcionamiento de esta aplicación, es fundamental el uso del ratón.

Botón izquierdo del ratón:

- Pulsado en una de las escenas del menú, te lleva al temario relacionado con dicha escena.

- Pulsando en el filo derecho de la pantalla estando dentro de un tema te llevará a la siguiente página del tema, en caso de ser la última página te devuelve a la primera página de ese primer tema.

- Manteniendo pulsado el botón dentro de una escena puedes rotar la cámara en el eje y para explorar la escena.

Botón derecho del ratón:

- Manteniendo pulsado el botón dentro de una escena puedes acercar la cámara o alejarla para explorar la escena.

Teclas 1 y 2:

- Si pulsas alguna de estas teclas te llevara a la escena relacionada con esa tecla (la tecla numero 1 te lleva al primer tema y la tecla 2 a la segundo).

Tecla 0:

- Si pulsas esta tecla volverás al menú principal.

Teclas especiales:

- Solo funcionan en su escena concreta, y se explica que tecla se debe pulsar en cada escena que sea necesaria, por ejemplo, la tecla D, permite aumentar el número de slices a la hora de revolucionar un perfil, como ya he comentado, todas están explicadas en sus escenas y se comprenden mejor dentro de estas.

Tecla esc:

- Cierre de la aplicación, también puede cerrarse pulsando el aspa roja.

5.3.2 Estructura de la aplicación

Al abrir la aplicación, lo primero que veras es el menú:

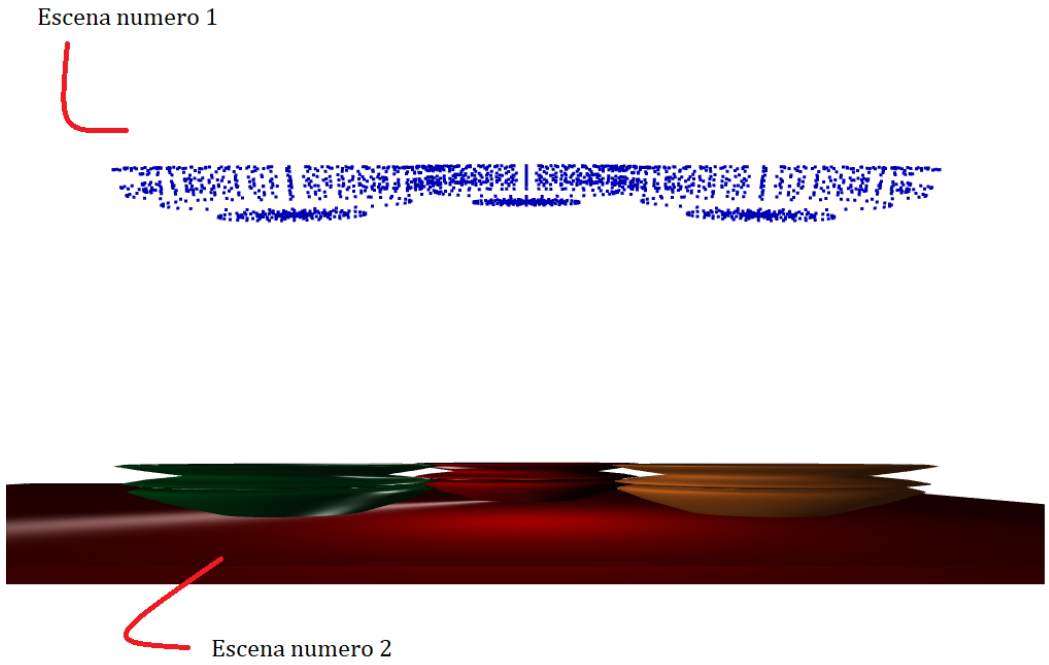


Ilustración 6.4.2.1

Al pinchar en una de las dos escenas (en un futuro, habrá más escenas seleccionables) o al pulsar la tecla 1 o 2 iras al tema relacionado con la escena mostrada.

Una vez dentro de la escena puedes o bien ver todo el tema e interactuar con él, o volver al menú con la tecla 0.

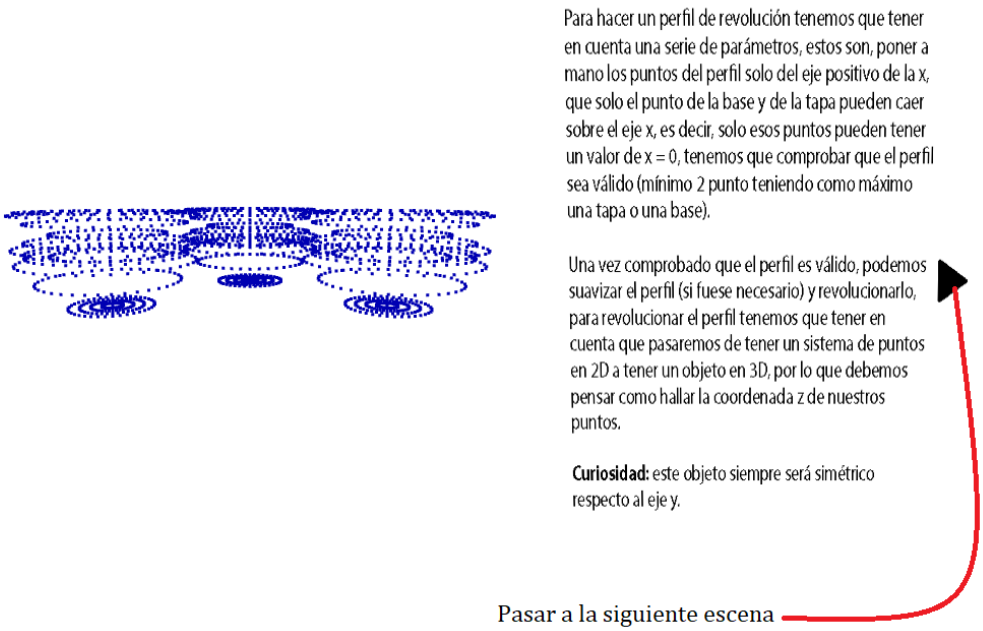


Ilustración 6.4.2.2

En las escenas que puedes interaccionar, puedes ver que tecla pulsar leyendo la explicación de la escena, además, la tecla a pulsar siempre aparecerá en negrita.

Si deseas salir de la aplicación puedes pulsar la tecla esc o pulsar el aspa con el botón izquierdo del ratón.

Ilustración 6.4.2.3

6 DEFINICIONES Y ABREVIATURAS

- **Slices:** Son las porciones en las que se divide nuestro objeto de revolución, contra más porciones, más pulido será nuestro objeto.
- **Escena:** Es el espacio utilizado para representar alguna cosa de forma visual, por ejemplo, en nuestra aplicación la escena sería el conjunto de los jarrones y la mesa.
- **Viewport:** Es un área donde podemos insertar lo que queremos representar de forma visual en nuestra aplicación, esta zona no tiene por qué ser del mismo tamaño que la ventana de nuestra aplicación, por ejemplo, en esta aplicación normalmente se usan dos viewports en cada escena.
- **Perfil de revolución:** Sucesión de puntos que al rotar 360º simularían un objeto curvo facilitando su creación virtual, ya que, se calculan los demás puntos de forma automática.
- **Texturas:** Imagen de mapa de bits usada para cubrir la superficie de los objetos virtuales para conseguir resultados más realistas.
- **Cámara Virtual:** Punto en nuestro universo virtual desde donde será vista nuestra escena.

7 BIBLIOGRAFÍA

Pressman, R. (2010). *Ingeniería del Software.* McGraw-Hill.

Angel, Edward (2012). *Interactive Computer Graphics 6th edition. Boston.*

Wesley, Addison (2014). *Computer graphics 3rd edition.* Upper Saddle River.

Rodríguez, Jacobo (2013). *GLSL essentials enrich your 3D scenes with the power of* GLSL!. Brimingham.

Lengyel, Eric (2012). *Mathematics for 3D game programming and computer graphics 3rd edition.* Boston.

Möller, Tomas (2008). *Real-time rendering 3rd edition.* Wellesley

Langetepe, Elmar (2006). *Geometric data structures for computer graphics.* Wellesley