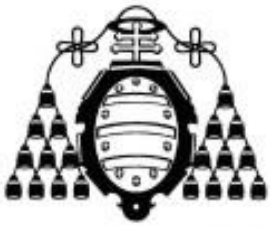
UNIVERSIDAD DE OVIEDO



ESCUELA DE INGENIERÍA INFORMÁTICA

TRABAJO FIN DE GRADO

SINTETIZADOR EN REALIDAD VIRTUAL

TUTORES: CARLOS MENCÍA CASCALLANA

RAÚL MENCÍA CASCALLANA

AUTOR: JESÚS ATORRASAGASTI GARCÍA

Agradecimientos

En primer lugar, agradecer a mi familia el apoyo incondicional que me han dado durante todos estos años y darme la oportunidad de poder estudiar esta carrera.

A todos los amigos de siempre, Iván, Tascón, Inés, Ana, Paula, María, Klaus, Jose, Juan y muchos otros, por estar ahí siempre que hizo falta.

A todos los buenos amigos que he hecho durante mi paso por la Escuela, en especial a Jorge, Damián, Sergio, Julián, Ana y Lorena.

A los muy buenos profesores del Grado, por impartir sus asignaturas con verdadera pasión e interés por el éxito de sus alumnos.

Y, por último y en especial, a Carlos Mencía, por ser un excelente profesor, por dirigir este TFG y aguantar todos estos años mis idas y venidas.

Resumen

Este proyecto tiene como objetivo desarrollar y presentar una aplicación que combine tecnologías ya existentes referentes al mundo multimedia (efectos visuales y generación de música) con otras relativamente innovadoras del mundo del hardware y software (realidad virtual). En concreto, el resultado final es un motor de generación de ondas de sonido (sintetizador) cuya interfaz es un sistema de realidad virtual de nueva generación.

Mediante el uso del seguimiento de manos incorporado en el visor, el usuario es capaz de generar música e interactuar con el sistema sin necesidad de utilizar mandos o controladores adicionales. Con gestos simples, una interfaz intuitiva y una curva de aprendizaje relativamente asequible, se pueden obtener resultados interesantes incluso en usuarios poco experimentados en cualquiera de los campos en los que se basa este proyecto.

El uso de la realidad virtual sumerge al usuario en un entorno futurista completamente inmersivo en el que puede visualizar la música creada.

El sintetizador incluye diversas funcionalidades, como la modificación de parámetros de la onda generada, selección de varios instrumentos y grabación y gestión de pistas.

Palabras Clave

Realidad Virtual, Hand Tracking, Sintetizador, Oculus, Interacción persona-máquina

Abstract

This project aims to present an application that combines existing technologies from the multimedia world (visual effects and music generation) with other relatively innovative technologies from the hardware and software world (virtual reality). Specifically, the final result is a sound wave generation engine (synthesizer) whose interface is a state-of-the-art virtual reality system.

Through the use of hand tracking built into the viewer, the users are able to generate music and interact with the system without the need for additional knobs or controllers. With simple gestures, an intuitive interface and a relatively steep learning curve, interesting results can be obtained even by inexperienced users in any of the fields on which this project is based.

The use of virtual reality immerses the users in a fully immersive futuristic environment in which they can visualize the created music.

The synthesizer includes several functionalities, such as modifying parameters of the generated waveform, selecting various instruments, and recording and managing tracks.

Keywords

Virtual Reality, Hand Tracking, Synthesizer, Oculus, Human-computer interaction

Índice General

Índice de Figuras

Índice de Tablas

Capítulo 1. Memoria del Proyecto

1.1 Motivación del proyecto

Este proyecto está motivado por el interés personal del autor en investigar y aprender sobre el desarrollo de aplicaciones y juegos enfocados al entretenimiento en el paradigma de la realidad virtual.

Si bien el concepto de realidad virtual ha existido durante varias décadas, es a partir de este momento (década de 2020) cuando las grandes compañías de software y hardware están enfocando una cantidad significativa de recursos en el desarrollo de estos sistemas.

El autor posee un gran interés en esta tecnología y considera que el desarrollo de aplicaciones de entretenimiento que habiliten a nuevos usuarios para explorar e interactuar de manera novedosa puede suponer un incremento en la adopción de este tipo de dispositivos para un uso casual y/o continuado.

La ambición última del proyecto es demostrar que, con las herramientas escogidas (software gratuito y hardware relativamente poco costoso), es posible desarrollar una aplicación que sea atractiva para el usuario consumidor medio de aplicaciones de entretenimiento o videojuegos, al que le pueda despertar interés sobre el mundo de la creación musical para aficionados mediante el uso de las nuevas tecnologías.

1.2 Resumen de los objetivos del proyecto

El proyecto tiene como objetivo permitir al usuario interactuar de distintas maneras con un entorno virtual generando diversos sonidos utilizando “hand tracking” (seguimiento de manos) para controlar el instrumento seleccionado.

La aplicación resultante será estará formada principalmente por un entorno 3D inmersivo con la que el usuario podrá cambiar de instrumentos, modificar parámetros de los sonidos y experimentar con la creación de audio en tiempo real.

El usuario también podrá grabar sus propias creaciones y hacer diferentes acciones de gestión sobre ellas, como reproducirlas y pararlas, renombrarlas o eliminarlas. Además, estas pistas podrán ser accesibles desde un ordenador para ser utilizadas en aplicaciones externas.

1.3 Alcance del proyecto

Se define como alcance final del proyecto una aplicación para dispositivos de realidad virtual en la cual se integran diversos plugins, como el de generación de audio o el de integración con el hardware.

Demás de los objetivos de desarrollo del proyecto, se esperan obtener observaciones interesantes sobre el uso de aplicaciones de realidad virtual en distintos tipos de usuarios.

1.4 Estructura del documento

Este documento se estructura de la siguiente manera:

Capítulo 1. Memoria del proyecto: Resumen de la motivación, objetivos y alcance de este proyecto.

Capítulo 2. Introducción: Se describe la justificación y objetivos del proyecto en detalle y se realiza una comparación de alternativas en cuanto a las herramientas de desarrollo y productos similares.

Capítulo 3. Aspectos teóricos: Explicación conceptual de los conceptos tecnológicos más importantes a tener en cuenta para comprender el desarrollo del proyecto.

Capítulo 4. Planificación del Proyecto y Resumen de Presupuestos :Breve explicación sobre la planificación y presupuesto del proyecto.

Capítulo 5. Análisis: Fase de análisis del proyecto, en la que se redacta la especificación del sistema a partir de la cual se obtienen los requisitos, casos de uso, diagrama preliminar de clases y una especificación preliminar del plan de pruebas.

Capítulo 6. Diseño: Diagramas de clases definitivos, descripción de la arquitectura del sistema, diseño de interfaces y especificación técnica del plan de pruebas.

Capítulo 7. Implementación del sistema: Se describen los problemas encontrados durante el desarrollo del sistema, así como una mención a las herramientas y lenguaje utilizados, junto con una descripción detallada de las clases y de partes importantes del código del sistema.

Capítulo 8. Desarrollo de las pruebas: Resultados de las pruebas especificadas durante la fase de diseño, así como observaciones y correcciones realizadas.

Capítulo 9. Manuales del sistema: Manuales de instalación, ejecución y de usuario.

Capítulo 10. Conclusiones y ampliaciones: Se resumen las conclusiones obtenidas tras terminar el proyecto y se describen algunas de las ampliaciones posibles.

Capítulo 11. Presupuesto: Descripción detallada del presupuesto interno y de cliente.

Capítulo 12. Referencias y Bibliografía: Referencias a fuentes consultadas para la redacción e implementación del proyecto.

Capítulo 13. Apéndices: Glosario, contenido de anexos e índice alfabético.

Capítulo 2. Introducción

2.1 Justificación del Proyecto

La Realidad Virtual (RV) es uno de los campos en los que el software ha estado creciendo a pasos agigantados durante los últimos años. La idea de acceder a un entorno virtual de un modo completamente inmersivo, en comparación con los sistemas "2D" mayoritariamente utilizados hasta la fecha, abre infinidad de posibilidades en cuanto a casos de uso, aplicaciones y nuevas maneras de interacción con los ordenadores.

Más recientemente, se ha hablado de la creación de un "metaverso" que vendría a sustituir lo que actualmente conocemos como identidad digital. Se propone como una extensión del "yo físico", en un universo virtual, en el cual podemos interactuar con distintas aplicaciones o con otros usuarios.

Si el objetivo de este metaverso es extender las posibilidades de la realidad física, debería hacerlo en todas sus facetas: redes sociales, productividad, trabajo, ocio, entretenimiento… Y por tanto debe existir contenido que satisfaga todas estas necesidades.

Aquí es donde este proyecto entra en juego: si bien los músicos y artistas tienen opción de crear música de manera analógica utilizando instrumentos y software tradicionales, también deben tener su espacio en este universo virtual para crear y experimentar con nuevas tecnologías.

Figura 2.1 Gafas de realidad virtual (Oculus Quest)

Si bien ya existen otras soluciones en el mercado, muchas de ellas se centran en instrumentos específicos y generalmente no son interoperables entre sí. Además, dependen exclusivamente del uso de los controladores incluidos con los dispositivos, con lo que conlleva cierta curva de aprendizaje para los nuevos usuarios.

Este proyecto tiene como objetivo ofrecer una rápida introducción a la creación musical experimental utilizando los nuevos paradigmas de interacción persona-máquina con una curva

de aprendizaje baja, así como la utilización de formatos de audio comunes y plugins de código abierto.

2.2 Objetivos del Proyecto

Existe una serie de objetivos dentro de este proyecto; si bien la mayoría de ellos son relacionados con la aplicación resultante, otros lo son con el desarrollo de la aplicación en sí, a modo de investigación sobre los nuevos paradigmas que se aplican.

Reproducción de sonidos generados por un sintetizador virtual

El proyecto utilizará un motor de generación de sonido (un sintetizador de audio) en el cual se podrán modificar las ondas de sonido generadas a través de diferentes parámetros.

Se concibe como uno de los objetivos principales del proyecto.

Variedad de instrumentos a elegir

Se permite al usuario utilizar diferentes instrumentos virtuales que generan sonidos diferentes en función del cual esté seleccionado.

Guardado y reproducción de pistas

Desde la aplicación se pueden hacer guardados de las pistas generadas, para ser reproducidas en un futuro o exportadas para su uso en otras aplicaciones.

Aprendizaje sobre el desarrollo de una aplicación en Realidad Virtual con Hand Tracking

El desarrollo de una aplicación en realidad virtual ofrece la posibilidad de conocer de primera mano los desafíos y características que se presentan durante las diferentes fases de un proyecto de este tipo.

2.3 Estudio de la Situación Actual

En el mundo de las aplicaciones enfocadas hacia la producción musical para el gran público existen multitud de opciones, tanto para aquellos que prefieren utilizarlas como simplemente una forma más de entretenimiento, como para los músicos aficionados y profesionales que las utilizan para explorar ideas que les ayuden en tareas creativas y de composición.

La gran mayoría de aplicaciones existentes actualmente son utilizadas en plataformas de escritorio o móvil, es decir, poseen una interfaz 2D sobre la cual, de manera más o menos realista, los usuarios interactúan con instrumentos y generan pistas y sonidos de distintos tipos.

Este tipo de aplicaciones suelen tener una interfaz generalmente estándar, en la cual se pueden diferenciar claramente distintas partes, correspondientes a un selector de instrumentos, una zona de gestión de pistas, interfaces para “tocar” los instrumentos, así como diversos puntos de

información: volumen, pulsos por minuto (BPM), clave, entre otros. Como ejemplo de interfaz de este tipo, se puede ver el software GarageBand en la Figura 2.1.

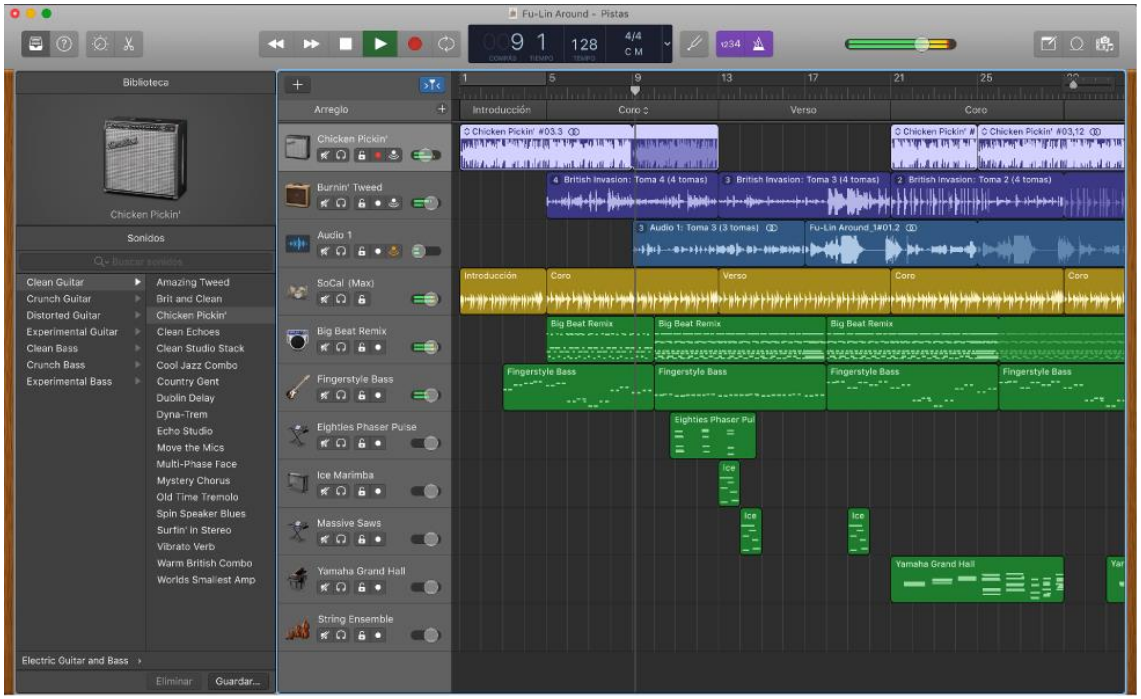


Figura 2.2 Interfaz del software GarageBand (Mac OS)

Si bien, como mencionamos, este tipo de aplicaciones suelen ser de escritorio o para dispositivos móviles, se pueden encontrar alternativas más sencillas como juegos musicales o aplicaciones que simulan tocar instrumentos, sin incluir todo lo relacionado con la producción musical como puede ser la gestión de pistas o un secuenciador MIDI. Apps de este tipo se pueden encontrar con facilidad en las tiendas de aplicaciones de iOS o de Android.

Para desarrollar aplicaciones de escritorio y móvil se pueden utilizar infinidad de lenguajes y frameworks, como podrían ser Flutter, Ionic, React o Xamarin. Sin embargo, y dado que el objetivo del proyecto es hacer un desarrollo para dispositivos de realidad virtual, hay que hacer uso de los frameworks de desarrollo y ejecución estándar en estas plataformas. En este caso, es interesante utilizar un motor de videojuegos, por la facilidad de desarrollo y despliegue de este tipo de aplicaciones en dispositivos de realidad virtual.

Además, los fabricantes de dispositivos de realidad virtual suelen proporcionar SDKs (kits de desarrollo) propietarios para facilitar la integración de aplicaciones desarrolladas en estos motores en sus sistemas.

La aplicación resultante del proyecto compartirá funcionalidades con estas aplicaciones ya mencionadas. Sin embargo, la manera en la que el usuario aprenderá a utilizarlas a través de esta interfaz será sustancialmente distinta: se traslada la interacción desde una pantalla, en dos dimensiones y utilizando ratón y teclado, a poner al usuario en el centro de la acción y utilizando sus propias manos para interactuar con el sistema.

2.3.1 Evaluación de Alternativas

2.3.1.1 Software de desarrollo

Podríamos diferenciar los dos mayores motores de videojuegos en este momento, Unity y Unreal Engine. Ambos tienen extensa documentación en Internet, así como gran cantidad de foros y recursos en los que basarse a la hora de desarrollar.

2.3.1.1.1 Unity

El motor Unity **[U03] [W01]** es uno de los motores de videojuegos más conocidos de la industria de desarrollo de videojuegos y otros tipos de aplicaciones 3D. Destaca por su facilidad de aprendizaje (se programa en C#), tiene una interfaz sencilla y fácil de utilizar, pero también es un software potente en cuanto a la cantidad de paquetes extra que se pueden instalar. Suele ser el software de referencia a la hora de empezar a programar videojuegos.

2.3.1.1.2 Unreal Engine

Unreal Engine **[UE01] [W02]** es un motor de videojuegos que destaca por el potencial de calidad gráfica que se puede obtener desarrollando en él. Proporciona una gran cantidad de opciones gráficas, iluminación, postprocesado y renderizado 3D. También es un motor ampliamente usado por la comunidad, aunque presenta una curva de aprendizaje mayor para nuevos usuarios.

2.3.1.1.3 Alternativa seleccionada

Se eligió Unity en favor de Unreal Engine como plataforma de desarrollo principalmente por su cantidad de documentación relativa a integración y desarrollo de aplicaciones en realidad virtual y por su licencia de uso gratuita. Otro factor para tener en cuenta es que el alumno ya posee cierta experiencia con el motor, lo que facilita las fases iniciales del desarrollo.

2.3.1.2 Plataformas de hardware

Como se plantea en los objetivos del proyecto, es necesario elegir una plataforma de hardware (en forma de gafas de realidad virtual) sobre la que se ejecutará la aplicación. Existen distintos segmentos y rangos de precios para este tipo de dispositivos, cada uno enfocado a un tipo de usuario diferente. Se pueden diferenciar en gafas para el consumidor más casual, jugadores de videojuegos más exigentes o usuarios profesionales de esta tecnología.

2.3.1.2.1 Oculus Quest

Las gafas Oculus Quest son el dispositivo de realidad virtual más conocido y mundialmente cuando se habla de realidad virtual standalone (independiente, que no requiere estar conectado a un PC). Tienen un precio asequible para el público general y una tienda de aplicaciones con gran cantidad de contenido.

A la hora de desarrollar sobre esta plataforma, se pueden utilizar tanto como un dispositivo Android (standalone) como un dispositivo PCVR (se conectan a un ordenador, que es quien se encarga del procesamiento, y se utilizan las gafas como display), lo que facilita mucho el desarrollo y reduce en gran medida los tiempos de compilación y entre iteraciones del proyecto.

Tres años después de su lanzamiento oficial, reciben actualizaciones regularmente que incluyen nuevas funcionalidades en su software, como el hand tracking, o la posibilidad de acceder a las cámaras de seguimiento externas. En 2021, se lanzó una revisión de hardware llamada Oculus Quest 2 con diversas mejoras del sistema y componentes a un coste más reducido.

En la fecha de redacción de este documento (junio 2022), es necesario tener una cuenta de Facebook para utilizarlas, pero se ha anunciado que a partir de agosto de 2022 ya no será necesario y comenzará una migración hacia cuentas de Meta (el nuevo nombre de la compañía.)

Figura 2.3 Oculus Quest (primera generación)

2.3.1.2.2 Pico Neo 3 Pro

Las gafas Pico Neo 3 Pro son un dispositivo similar a las Oculus Quest 2: presentan prestaciones similares, pero con un sistema algo menos pulido y más enfocado al público empresarial. Tienen un coste algo más elevado y tanto la documentación como la comunidad de usuarios desarrolladores son más reducidas. No disponen de funcionalidades de hand tracking, pero tampoco requieren de una cuenta de ningún tipo para empezar a utilizarlas.

Figura 2.4 Pico Neo 3 Pro

2.3.1.2.3 Dispositivos PCVR

Dentro de la gama de productos de PCVR (conectados a un ordenador), existen diversas opciones más enfocadas a un público profesional o de usuarios más expertos y aficionados a los videojuegos más exigentes gráficamente en realidad virtual.

Por lo general, estas gafas son más caras que su contraparte standalone pero ofrecen prestaciones muy superiores en cuanto al seguimiento del usuario en el entorno, pantallas, lentes y calidad de fabricación. Como su nombre indica, requieren de un PC potente que pueda ejecutar estas aplicaciones más exigentes gráficamente, con lo que el uso de este tipo de gafas resulta significativamente más caro de inicio.

Figura 2.5 HTC Vive

2.3.1.2.4 Leap Motion Controller

El dispositivo Leap Motion Controller es un dispositivo de reducido tamaño que integra una serie de cámaras infrarrojas que son capaces de capturar y mapear las manos del usuario a modo de controladores en distintos dispositivos (PC o realidad virtual).

Inicialmente se lanzó a través de una campaña de mecenazgo online como un periférico novedoso para ordenadores. Sin embargo, y debido a su popularidad, la empresa cambió su modelo de mercado, mejorando su producto para facilitar su integración con dispositivos de realidad virtual a modo de accesorio.

Figura 2.6 Leap Motion Controller en unas Pico Neo 3 Pro

2.3.1.2.5 Alternativa seleccionada

A la hora de desarrollar el proyecto, el alumno ya posee un dispositivo Oculus Quest de primera generación. Sumado a que ya está familiarizado con el entorno de desarrollo y el dispositivo proporciona todas las características necesarias para cumplir con los objetivos del proyecto, esta fue la alternativa de hardware escogida.

2.3.1.3 Aplicaciones similares

2.3.1.3.1 GarageBand

GarageBand es el software de producción de audio gratuito incluido en todos los sistemas del ecosistema Apple (Mac OS, iOS). Ofrece una gran cantidad de instrumentos y de opciones de producción. Es utilizado tanto por aficionados como por profesionales para composición rápida de música.

Como ventajas principales se puede destacar que es uno de los softwares más reconocidos del mercado en cuanto a facilidad de uso y curva de aprendizaje, y que puede utilizarse para composiciones complejas si es necesario.

La mayor desventaja de este software es que está únicamente disponible en el ecosistema Apple como hemos comentado antes: esto puede ser una barrera de entrada, sobre todo por el precio, para el usuario final. Además, diversas opciones, como el generador de ritmos o la simulación de instrumentos de cuerda, están algo limitadas.

Figura 2.7 Interfaz del software GarageBand (iPad)

2.3.1.3.2 TribeXR

TribeXR es una de las aplicaciones de música en realidad virtual más conocidas. Presenta una premisa muy simple: enseñar al usuario a hacer de DJ a través de la realidad virtual, sin tener que adquirir todo el hardware necesario.

Es uno de los softwares de este tipo más completos, ya que simula perfectamente una mesa de mezclas completamente idéntica a una *Pioneer* real, con lo cual si el usuario ya sabe utilizar este tipo de hardware sólo necesitará acostumbrarse a los controles para poder hacer sus mezclas en realidad virtual. Además, incluye un "instructor virtual" que guía al usuario en caso de que no tenga experiencia.

Como desventaja, se podría decir que no es un software de creación musical en sí, ya que sólo permite hacer mezclas como haría un DJ en el mundo real, aunque no deja de ser una propuesta única y muy interesante.

Figura 2.8 TribeXR

2.3.1.3.3 Virtuoso

Virtuoso es una de las aplicaciones más recientes y completas para producir música en realidad virtual. Permite al usuario tener su propio "estudio musical" con variedad de instrumentos reales y nuevos creados específicamente para esta aplicación.

Cuenta con un looper (secuenciador) y funcionalidades como sincronizador de tempo y de un asistente para mantener la clave. De las distintas alternativas, esta es la que más se asemeja al objetivo del proyecto: sin embargo, está basada en input por controladores, no por hand tracking.

Figura 2.9 Virtuoso

Capítulo 3. Aspectos Teóricos

3.1 Sintetizador musical

Un sintetizador musical es un instrumento electrónico, que genera sonido mediante señales eléctricas y lo reproduce a través de altavoces o auriculares. También existen sintetizadores virtuales, completamente software, que generan señales digitales en lugar de sonidos analógicos.

Los sintetizadores son instrumentos generalmente modulares, en los que, tradicionalmente, se conectan distintos tipos de efectos a un teclado que es quien lanza las notas. Estos efectos modifican la señal de distintos tipos: pueden distorsionarla subiendo la ganancia, crear un efecto de vibrato, simular otros instrumentos o incluso voces.

Los sintetizadores comenzaron a desarrollarse en la época de los años 20, comenzando por el Theremin (creado por Léon Theremin). Este instrumento tiene la particularidad de que no hace falta tocarlo físicamente para conseguir sonidos a través de él. Posee dos antenas que son capaces de detectar la posición de las manos del intérprete, que controla la frecuencia y el volumen de sonido generado con cada una de ellas.

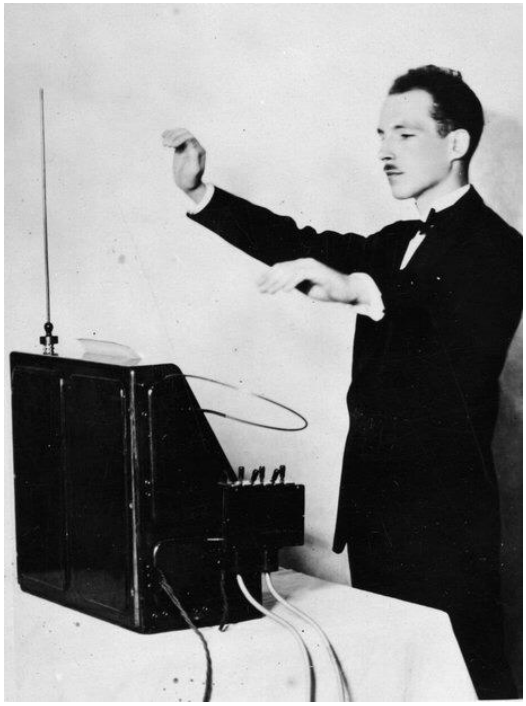


Figura 3.1 Theremin

Posteriormente, este instrumento fue evolucionando a través de los años hasta que, alrededor de la década de los 70 llegó a ser lo que conocemos hoy como sintetizador, un aparato altamente modificable que permitía a los intérpretes generar sonido como hasta entonces no se había hecho **[BV01] [DS18]**.

Infinidad de artistas han utilizado y utilizan hoy en día sintetizadores para producir música. Es un instrumento altamente versátil y que tampoco requiere un alto presupuesto ni formación musical específica para empezar a utilizar.

Figura 3.2 Moog Minimoog

Algunos de los módulos más conocidos y utilizados en sintetizadores son el envelope, el bitcrush, o el oscilador. Algunos conceptos sobre estos módulos se utilizarán durante el desarrollo del proyecto por lo que conviene hacer una breve descripción de cada uno de ellos:

- **Oscillator**: es uno de los módulos más básicos de un sintetizador. Es aquel que produce una nota (frecuencia) con una forma de onda cíclica específica. Existen distintas formas de onda: sinoidal, triangular, diente de sierra, o cuadrada. Se pueden combinar distintos osciladores para crear efectos de sonido particulares.
- **Envelope**: controla el volumen de un parámetro (por lo general, el volumen) de un sonido en un espacio de tiempo. Es un módulo especialmente versátil ya que con sus cuatro componentes clásicos ofrece infinidad de distintos tipos de sonidos: attack (tiempo que toma la nota en alcanzar el máximo nivel), decay (tiempo que toma en alcanzar el nivel de sustain), sustain (nivel al cual se mantiene la nota) y release (el tiempo que tarda en decaer desde sustain a cero) **[IL01]**.
- **Bitcrush o bitcrusher**: produce distorsión en un sonido reduciendo la "resolución" de un sonido digital. Esto puede generar desde sonidos con un efecto más cálido (simulando instrumentos analógicos) o fuertemente distorsionado (lo que se suele llamar "lo-fi").

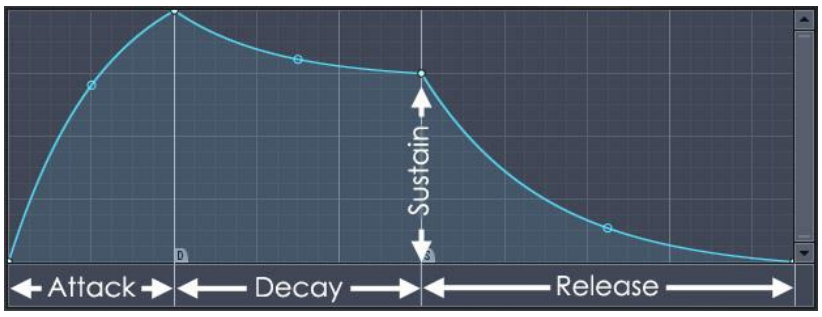


Figura 3.3 Representación gráfica de un módulo Envelope [IL01]

3.2 Realidad Virtual

La realidad virtual (RV) es uno de los conceptos de las tecnologías inmersivas de los que más se lleva hablando durante décadas en el campo de la informática. Presenta la idea de, a través de un hardware específico, sumergir al usuario en un entorno virtual sin que este sea necesariamente gráfica o físicamente realista. En cierto modo, se engaña a los sentidos (vista, oído e incluso tacto en ciertas situaciones) de la persona para que esta piense que realmente "está ahí" **[Boas13]**.

Si bien la tecnología lleva en estudio y desarrollo durante décadas, con numerosos prototipos y dispositivos rudimentarios a lo largo de los años, debido a la incrementada potencia de los últimos procesadores para dispositivos portátiles, las grandes empresas han decidido dedicar gran cantidad de recursos al desarrollo de esta tecnología.

En 1968, un grupo de ingenieros del MIT, encabezados por Ivan Sutherland, presentan "**The Sword of Damocles" [XC16]**, un rudimentario sistema que permitía seguir los movimientos del usuario a través de un casco que, sujeto por un brazo mecánico colgado del techo, proyectaba imágenes de baja resolución en dos pantallas de tubo de rayos catódicos.

Figura 3.4 The Sword Of Damocles [XC16]

Este sistema (proyectar una imagen distinta en cada ojo) prevalece hasta nuestros días. Si bien era un prototipo, permitía lo que hoy se conoce tracking en seis grados de libertad. Como se puede ver en la Figura 3.2, el tracking en tres grados de libertad permite únicamente el seguimiento rotacional de la cabeza del usuario: podrá mirar a su alrededor, pero no moverse dentro del contenido. En cambio, los seis grados de libertad proporcionan un nivel adicional de inmersión permitiendo no sólo la rotación sino también el desplazamiento en las tres dimensiones del espacio.

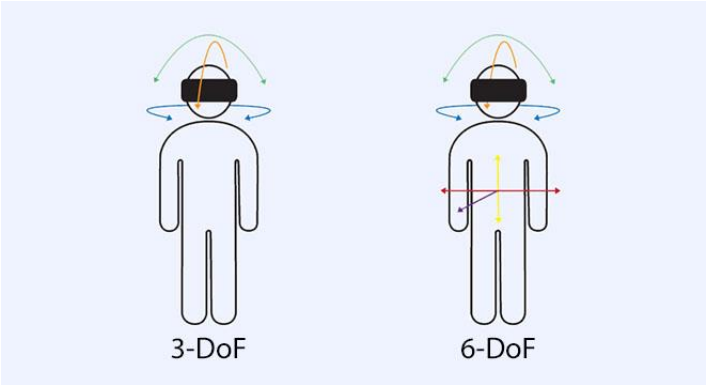


Figura 3.5 Comparativa entre tres y seis grados de libertad

A lo largo de las posteriores décadas, hubo numerosos intentos de incrementar el uso de la realidad virtual en el mercado de consumo, pero ninguno llegó a buen puerto debido a las limitaciones técnicas y el elevado precio de estos aparatos. El más famoso fue el Virtual Boy, consola de videojuegos lanzada por Nintendo en el año 1995. Sus mayores limitaciones fueron el propio tamaño de la consola, una pantalla pobre (monocromática y a poca resolución) y un efecto estereoscópico que no llegó a convencer a los usuarios.

Durante la última década, gracias al avance de la tecnología, han aparecido varios sistemas que ofrecen capacidades de inmersión mucho más interesantes para el público general. La potencia de los procesadores y tarjetas gráficas, junto con la disponibilidad de ordenadores de alto rendimiento para el público general han propiciado que muchos usuarios, tanto aficionados a los videojuegos como empresas dedicadas a formación o simulación se decanten por estos dispositivos.

En 2012, Palmer Luckey presenta su "Oculus Rift DK1" [Harley20] una unidad prototipo de lo que sería el primer casco de realidad virtual de consumo con aceptación masiva, el "Oculus Rift CV1". El Rift CV1 ofrecía tres grados de libertad, un sistema de auriculares integrados y una pantalla por ojo 1080x1200 a 90Hz.

Figura 3.6 Palmer Luckey con las Oculus DK1 (izquierda) – Oculus Rift CV1 (derecha)

Posteriormente, la empresa Oculus fue adquirida por Facebook (ahora Meta) y el desarrollo de los productos de realidad virtual continuó la línea establecida, con el Rift S, Go y finalmente Oculus Quest.

3.3 Oculus Quest

El dispositivo de realidad virtual Oculus Quest es uno de los más modernos y potentes actualmente. Fue lanzado en mayo de 2019, cuando Oculus ya era parte del grupo Facebook. Se presentó como una versión completamente mejorada de lo que hasta la fecha se conocía como un casco de realidad virtual: ofrecía libertad total de movimientos ya que no dependía de estar conectado a un ordenador para funcionar, pues utilizaba un sistema operativo Android modificado, con las cámaras integradas era capaz de ofrecer tracking en seis grados de libertad, el casco era ligero y cómodo, contenía altavoces de calidad integrados y todo ello a un precio muy asequible. Sus especificaciones pueden comprobarse en la Tabla 3.1.

Pantalla	2 pantallas OLED 1440x1600 (72Hz)
Procesador	Qualcomm Snapdragon 835
Gráficos	Adreno 540
RAM	4 GB
Tracking	Inside-out a través de 4 cámaras posicionales
Almacenamiento	64 o 128 GB
Precio	449€ (64GB) / 549€ (128GB)

Tabla 3.1 Especificaciones técnicas de las Oculus Quest

En octubre de 2020 se presentó una versión revisada, las Oculus Quest 2. Ofrecían mayor confort gracias al menor peso del dispositivo. También incrementó su potencia gracias al procesador Qualcomm XR2 y memoria RAM aumentada (6GB). Se revisó la pantalla por un panel único LCD con resolución de 1832 x 1920 por cada ojo, y se redujo el precio a 349€ por el modelo más básico (que pasó a ser de 128 GB).

Figura 3.7 Dispositivos Quest y Quest 2

Se eligió este dispositivo como plataforma de desarrollo porque cumple todas las características requeridas para el proyecto (bajo coste, facilidad de desarrollo, hand tracking). Además, fue una razón de peso que el desarrollador ya fuera poseedor de uno de estos dispositivos (de primera generación).

3.4 Hand tracking

Dentro del mundo de la computación, el hand tracking (seguimiento de manos) es un concepto que engloba las diferentes técnicas utilizadas para reconocer, mapear y representar la posición y rotación en tiempo y espacio de manos humanas reales en un entorno virtual.

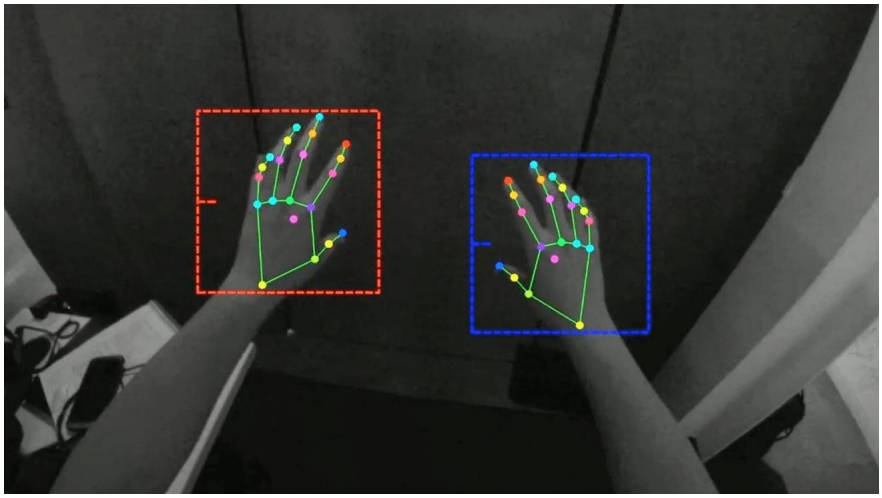


Figura 3.8 Hand tracking

Existen dos maneras principales de llevar a cabo esta técnica: con y sin sensores de movimiento, cada uno con sus respectivas ventajas e inconvenientes:

Los sistemas **con sensores de movimiento** (guantes especializados o dispositivos similares) son capaces de ofrecer un seguimiento muy preciso de distintos parámetros, por ejemplo: la inercia de los movimientos o la posición y rotación a través de periféricos con reconocimiento óptico de patrones y marcadores (de un modo similar al que se utiliza para hacer el seguimiento de los controladores en realidad virtual). Sin embargo, estos dispositivos suelen tener un coste relativamente elevado y generalmente no son completamente ergonómicos, lo que dificulta su uso generalizado. Sí que pueden tener usos específicos como el de la captura de movimientos (MoCap) en entornos de cine y producción virtual, en los que el uso de trajes que utilizan esta tecnología es habitual.

Los sistemas **sin sensores de movimiento** basan su implementación en el procesamiento de imágenes en tiempo real utilizando técnicas de visión artificial y algoritmos de Deep Learning **[Melax13].**
A través de distintas "fases" (sustracción de fondo, segmentación de la mano, localización de puntos de interés), el sistema es capaz de extraer un modelo de mano completo únicamente con una imagen de entrada. Posteriormente, se puede analizar el modelo obtenido para obtener

más información relevante, por ejemplo: qué mano es (izquierda o derecha), si está realizando algún gesto o si se está viendo la palma o el reverso de la mano.

El uso de reconocimiento de gestos en realidad virtual está principalmente enfocado a reemplazar los controladores incluidos con los dispositivos. De esta manera, se busca una interacción más natural con el sistema, ya que generalmente los nuevos usuarios tienden a dedicar cierto tiempo a obtener soltura con los controladores.

Los gestos reconocidos en este tipo de sistemas más comunes son:

- *Pinch*: juntar dedos índice y pulgar. Se suele utilizar para seleccionar y apuntar en elementos pequeños en interfaces, sustituyendo a un puntero o ratón.
- *Grab*: cerrando el puño completamente, se suele utilizar para coger y arrastrar objetos virtuales en entornos inmersivos.
- Otros: *pointer* (apuntar con el dedo índice), *thumbs up* (pulgar arriba), *bloom* (juntar todas las puntas de los dedos y abrir la mano).

A finales del año 2019, Oculus añadió esta característica a su sistema operativo utilizando las cámaras monocromáticas del hardware, que hasta ahora solo se utilizaban para el posicionamiento espacial del usuario.

3.5 Motor de videojuegos Unity

El motor Unity es una de las herramientas de desarrollo de videojuegos y aplicaciones 3D más potentes del mercado. La fácil curva de aprendizaje, su versión gratuita para pequeños desarrolladores y la comunidad online han sido los factores más relevantes a la hora de hacer de Unity uno de los motores de referencia.

Si bien existen otras opciones (por ejemplo, Unreal Engine) igualmente válidas a la hora de desarrollar aplicaciones para dispositivos de realidad virtual, es Unity la plataforma que tiene más material de aprendizaje en este ámbito. Además, el SDK de Oculus es fácilmente integrable en el entorno, y tiene una base de código realmente sólida que hacen del desarrollo de aplicaciones una tarea poco problemática.

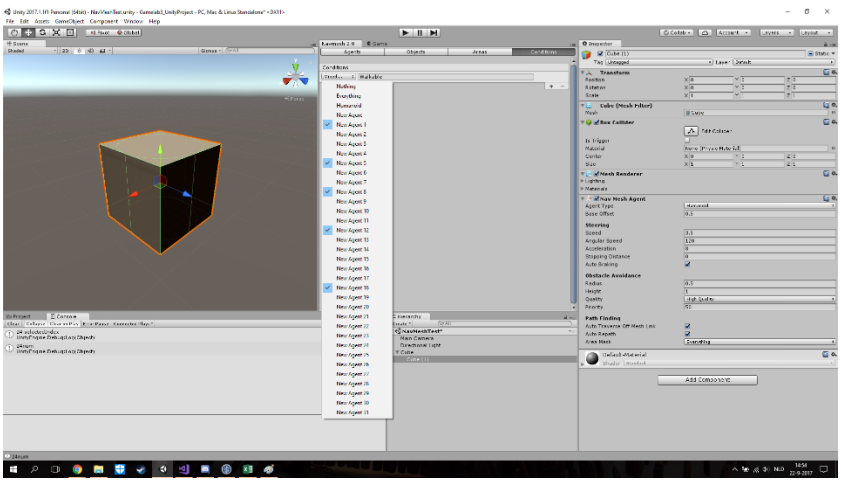


Figura 3.9 Interfaz del motor Unity

Capítulo 4. Planificación del Proyecto y Resumen de Presupuestos

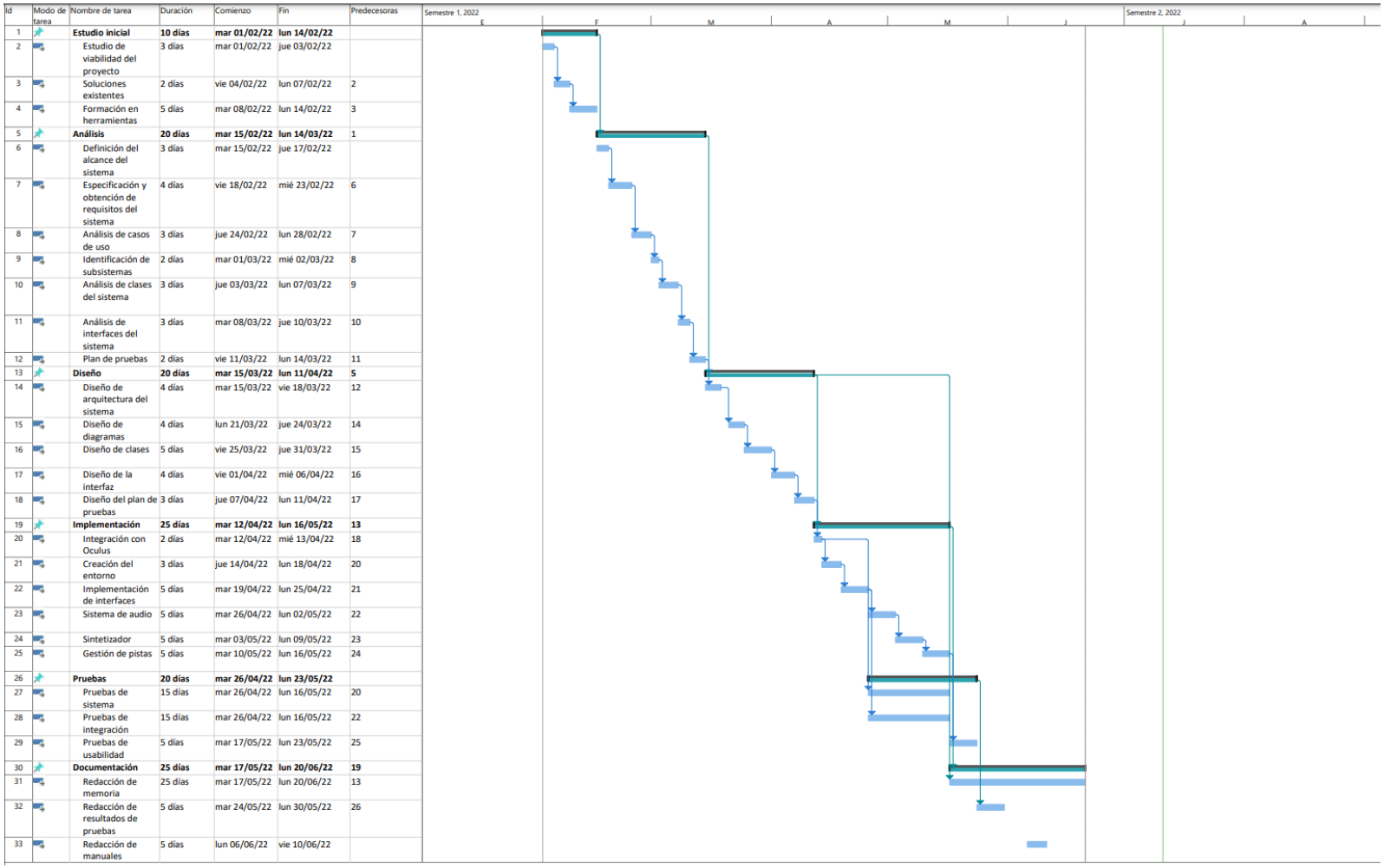
4.1 Planificación

El proyecto fue inicialmente planificado para ser realizado en el transcurso de un semestre (finales de enero a finales de junio de 2022).

Para el desarrollo del software, se planeó un desarrollo en cascada con diferentes fases: análisis, diseño, implementación, pruebas y documentación.

Se estimó un tiempo de desarrollo semanal total de 15 horas, dedicando 3 horas semanales extra en aquellas que se ejecutaran tareas de implementación de código para dedicar a las pruebas.

En la figura 4.1 se puede observar el diagrama de Gantt utilizado para la planificación del proyecto.



Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1		**Estudio inicial**	**10 días**	**mar 01/02/22**	**lun 14/02/22**	
2		Estudio de viabilidad del proyecto	3 días	mar 01/02/22	jue 03/02/22	
3		Soluciones existentes	2 días	vie 04/02/22	lun 07/02/22	2
4		Formación en herramientas	5 días	mar 08/02/22	lun 14/02/22	3
5		**Análisis**	**20 días**	**mar 15/02/22**	**lun 14/03/22**	1
6		Definición del alcance del sistema	3 días	mar 15/02/22	jue 17/02/22	
7		Especificación y obtención de requisitos del sistema	4 días	vie 18/02/22	mié 23/02/22	6
8		Análisis de casos de uso	3 días	jue 24/02/22	lun 28/02/22	7
9		Identificación de subsistemas	2 días	mar 01/03/22	mié 02/03/22	8
10		Análisis de clases del sistema	3 días	jue 03/03/22	lun 07/03/22	9
11		Análisis de interfaces del sistema	3 días	mar 08/03/22	jue 10/03/22	10
12		Plan de pruebas	2 días	vie 11/03/22	lun 14/03/22	11
13		**Diseño**	**20 días**	**mar 15/03/22**	**lun 11/04/22**	5
14		Diseño de arquitectura del sistema	4 días	mar 15/03/22	vie 18/03/22	12
15		Diseño de diagramas	4 días	lun 21/03/22	jue 24/03/22	14
16		Diseño de clases	5 días	vie 25/03/22	jue 31/03/22	15
17		Diseño de la interfaz	4 días	vie 01/04/22	mié 06/04/22	16
18		Diseño del plan de pruebas	3 días	jue 07/04/22	lun 11/04/22	17
19		**Implementación**	**25 días**	**mar 12/04/22**	**lun 16/05/22**	13
20		Integración con Oculus	2 días	mar 12/04/22	mié 13/04/22	18
21		Creación del entorno	3 días	jue 14/04/22	lun 18/04/22	20
22		Implementación de interfaces	5 días	mar 19/04/22	lun 25/04/22	21
23		Sistema de audio	5 días	mar 26/04/22	lun 02/05/22	22
24		Sintetizador	5 días	mar 03/05/22	lun 09/05/22	23
25		Gestión de pistas	5 días	mar 10/05/22	lun 16/05/22	24
26		**Pruebas**	**20 días**	**mar 26/04/22**	**lun 23/05/22**	
27		Pruebas de sistema	15 días	mar 26/04/22	lun 16/05/22	20
28		Pruebas de integración	15 días	mar 26/04/22	lun 16/05/22	22
29		Pruebas de usabilidad	5 días	mar 17/05/22	lun 23/05/22	25
30		**Documentación**	**25 días**	**mar 17/05/22**	**lun 20/06/22**	19
31		Redacción de memoria	25 días	mar 17/05/22	lun 20/06/22	13
32		Redacción de resultados de pruebas	5 días	mar 24/05/22	lun 30/05/22	26
33		Redacción de manuales	5 días	lun 06/06/22	vie 10/06/22	

Figura 4.1 Diagrama de Gantt del proyecto

4.2 Resumen del Presupuesto

Si bien este proyecto se planteó como algo experimental/de investigación, no se estableció un presupuesto inicial con el que debiera contarse para ser llevado a cabo.

De igual manera, resulta interesante elaborar un resumen que sirva para tener en cuenta cuántas horas de trabajo y coste supondría reproducir un proyecto similar en el futuro.

Se incluye un desglose de horas y costes más detallado en el capítulo 11 (Presupuesto).

Código	Concepto	Coste
1	**Análisis**	1.768,27 €
2	**Diseño**	1.768,27 €
3	**Implementación**	3.536,55 €
4	**Pruebas**	1.503,03 €
	TOTAL	**8.576,13 €**

Tabla 4.1 Resumen del presupuesto

Capítulo 5. Análisis

Este apartado contendrá toda la especificación de requisitos y toda la documentación del análisis de la aplicación, a partir de la cual se elaborará posteriormente el diseño.

5.1 Definición del Sistema

5.1.1 Determinación del Alcance del Sistema

La naturaleza del proyecto hace que el alcance del propio sistema esté claramente definido: desarrollar una aplicación independiente (que no sea parte de un sistema más complejo) para la plataforma Oculus Quest.

Esta aplicación tiene varias funcionalidades, todas ellas con el objetivo de fomentar la creatividad de los usuarios a través de la música mientras interactúan con nuevas tecnologías. El usuario puede interactuar con distintos elementos de la interfaz para experimentar con la síntesis de audio, utilizando el hand tracking en realidad virtual.

Además, puede grabar y reproducir pistas guardadas anteriormente que se almacenarán como ficheros dentro del sistema de archivos local, por lo que pueden ser exportadas a, por ejemplo, un software de producción de audio en un PC.

Ahondando más específicamente en qué se va a desarrollar, la implementación principal será la una interfaz en realidad virtual, asemejándose como sería la de un videojuego o aplicación de entretenimiento, que permitirá interactuar con distintos subsistemas (sonido y gestión de pistas) dentro de una misma aplicación.

Se parte de una base de código ya existente (explicado en más detalle en la sección 7.4.2) que modela un sintetizador sencillo. Implementar un motor de audio desde cero implicaría un desarrollo extra no planificado, por lo que se utilizó este código de partida, que requirió de diversas modificaciones, para conseguir una representación básica de un sintetizador que se pudiera utilizar en la aplicación.

También se desarrollará un sistema de persistencia sencillo que permita realizar las gestiones de archivos mencionadas anteriormente.

5.2 Requisitos del Sistema

5.2.1 Especificación del sistema

En esta sección se facilita una especificación detallada de las principales funcionalidades del sistema. A partir de ella, se obtendrán sus requisitos y posteriormente, casos de uso.

5.2.1.1 Idea principal

La idea inicial del proyecto se propone como una aplicación que reúna distintos componentes relacionados con la producción musical a nivel aficionado y permitan al usuario experimentar con ellos sin tener conocimientos relacionados con la materia.

Se plantea la experiencia del sistema como una aplicación similar a las existentes en las App Stores de los dispositivos móviles para tocar el piano o la batería, en las cuales el usuario no necesita hacer un registro ni iniciar sesión en el sistema; la experiencia se muestra directamente delante de sus ojos e invita a interactuar con la interfaz proporcionando un feedback inmediato a sus acciones. No es necesario configurar nada ni seguir un tutorial extenso que guíe por las distintas partes de la aplicación.

Como se ha mencionado, las interfaces deben invitar al usuario a interactuar con ellas. Estas interfaces forman parte del entorno (lo que se conoce como interfaz diegética **[Salomoni16], [Salomoni17]**): existen en el espacio y el usuario puede interactuar directamente con ellas, sin otro método de entrada que no sean sus manos. Esto proporciona una sensación de inmersión que no se obtiene si se diseña para ser utilizada con mandos u otro tipo de controladores, ya que los distintos botones y palancas requieren de cierto aprendizaje.

5.2.1.2 Instrumentos

Se plantean tres instrumentos inicialmente: el theremín, el piano y la batería. Cada uno de estos tres instrumentos da una dimensión distinta a la experiencia del usuario.
Por una parte, el piano es el instrumento más “común” y es una fácil introducción al sistema (pulso una tecla, suena una nota). La batería es un instrumento al cual no todo el mundo puede tener acceso (por razones de espacio o ruido), pero interactuar con ella es completamente intuitivo, ya que únicamente requiere de golpear los “pads”. Por último, el theremín es quizá el más desconocido de los instrumentos, ya que no es común utilizar uno de ellos sin tener que tocarlo físicamente. Aquí se da una experiencia al usuario a la que raramente podrá tener ocasión de acceder fuera del sistema. Todos estos instrumentos deben ser fácilmente reconocibles y sencillos de usar (bien sea por su aspecto o por la descripción o explicaciones que se den dentro de la aplicación), y debe ser sencillo también cambiar entre unos y otros sin interrumpir demasiado la experiencia del usuario.

Para aumentar la inmersión del usuario en el sistema al usar el theremín (ya que no es un instrumento “visible”), se propone que exista algún tipo de visualización de las frecuencias producidas por el instrumento en el entorno. Por ejemplo, barras de sonido en 3D.

En los instrumentos en los que se haga uso del sintetizador (piano y theremín) el usuario debe ser capaz de utilizar funcionalidades avanzadas para modificar algunos de los módulos para cambiar el sonido generado. En el piano, debe poder cambiar el bitcrush y el modulador, mientras que en el theremín con una mano modificará la nota que se va a tocar (frecuencia) y con la otra el multiplicador de la modulación (que se mantendrá fija).

5.2.1.3 Gestión de pistas

Por otro lado, y para dotar al sistema de cierta profundidad, se plantea también el diseño e implementación de un gestor de pistas. Este gestor de pistas debe ser capaz de realizar las acciones más básicas de un sistema de persistencia (añadir archivos, renombrarlos, eliminarlos) y de reproducción de música (reproducir, pausar, reanudar) debe ser robusto para evitar caer en malas experiencias de usuario. Del mismo modo que los instrumentos, esta interfaz también debe ser diegética para facilitar la experiencia de usuario. Además, utilizará conceptos ya conocidos por la mayoría, representando las funciones disponibles con iconos e imágenes familiares. Por tanto, el usuario debe poder reproducir, pausar, reanudar, grabar, modificar y eliminar pistas directamente desde esta interfaz.

Si bien podría resultar interesante no imponer limitaciones al sistema de gestión de pistas, se decide establecer inicialmente un límite de tres pistas como máximo en el sistema para facilitar el diseño, implementación y experiencia de la aplicación. También se pueden añadir pistas al sistema de forma externa, accediendo al almacenamiento interno del dispositivo por USB.

Cualquier acción no permitida en el sistema (excepciones, errores) debe ser propiamente mostrada al usuario.

5.2.1.4 Usabilidad, entorno y rendimiento

El sistema debe funcionar en un dispositivo Oculus Quest, que es de precio reducido e incluye la funcionalidad de hand tracking que se busca utilizar en esta aplicación. El hand tracking, junto con el reconocimiento gestual, será utilizado para interactuar con la interfaz, diferenciar entre la mano dominante y la no dominante

Además, la experiencia debe ser cómoda para el usuario, por lo que se espera que no canse demasiado la vista ni produzca mareos por mucho movimiento o un framerate bajo.

Los textos de la aplicación se mostrarán en inglés, pudiendo ampliarse a otros idiomas en un futuro.

5.2.2 Obtención de los Requisitos del Sistema

Se obtienen los requisitos del sistema a través de la especificación desarrollada tras elaborar la idea principal del proyecto por parte del alumno. No existe un stakeholder externo o cliente que defina unos requisitos de inicio, pero como parte del proyecto, se intenta elaborar un listado lo más exhaustivo posible.

5.2.2.1 Requisitos funcionales

5.2.2.1.1 Uso del sistema de audio

RS1: El usuario debe poder generar diferentes sonidos por medio de interfaces que representen instrumentos musicales.

RS1.1: El usuario debe poder utilizar el sintetizador de la aplicación por medio de la interfaz para reproducir ondas de sonido melódicas.

RS1.2: El usuario debe poder utilizar una de las interfaces de usuario para generar sonidos de instrumento de percusión.

RS2: El usuario debe poder cambiar entre los instrumentos disponibles en cualquier momento.

RS2.1: El usuario podrá cambiar el instrumento activado al piano.

RS2.1.1: Para tocar el piano, el usuario pulsará las teclas con las manos.

RS2.2: El usuario podrá cambiar el instrumento activado al theremín.

RS2.2.1: Para tocar el theremín, el usuario realizará el gesto “pinch” con las manos.

RS2.2.2: El usuario podrá ver representadas las frecuencias del sonido generado en un elemento del entorno 3D que simule un ecualizador de barras.

RS2.2.3: Se mostrará un mensaje de instrucción que explique al usuario cómo utilizar el instrumento.

RS2.3: El usuario podrá cambiar el instrumento activado a la batería.

RS2.3.1: Para tocar la batería, el usuario pulsará los pads con las manos.

RS3: El usuario podrá modificar parámetros del sintetizador en función de qué instrumento esté activado.

RS3.1: Si el instrumento activado es el piano, podrá modificar los módulos de bitcrush y de modulación del sintetizador.

RS3.1.1: La cantidad de bitcrush podrá variar entre los valores MIN_BITCRUSH y MAX_BITCRUSH.

RS3.1.2: La modulación podrá variar en valores MIN_MODULACION y MAX_MODULACION.

RS3.2: Si el instrumento activado es el theremín, podrá modificar el valor del multiplicador y la frecuencia del sintetizador.

RS3.2.1: El multiplicador podrá variar entre los valores MIN_MULTIPLICADOR y MAX_MULTIPLICADOR.

RS3.2.1.1: El usuario modificará este valor con MANO_NO_DOMINANTE

RS3.2.2: La frecuencia podrá variar entre los valores MIN_FRECUENCIA y MAX_FRECUENCIA.

RS3.2.2.1: El usuario modificará este valor con MANO_DOMINANTE

5.2.2.1.2 Gestión de pistas

RP1: El sistema debe cargar automáticamente las pistas del sistema de archivos.

RP1.1: El sistema podrá cargar un máximo de MAX_PISTAS pistas del sistema de archivos.

RP1.1.1: Si existen más de MAX_PISTAS pistas en el sistema de archivos, se mostrará un mensaje advirtiendo al usuario de que existen más pistas de las admitidas.

RP1.2: Las pistas deben estar en formato .wav para ser cargadas.

RP1.2.1: Si existen archivos con un formato diferente a .wav en RUTA_PISTAS, no se cargarán.

RP1.3: El sistema podrá admitir la carga de archivos externa a RUTA_PISTAS a través de la conexión USB del dispositivo.

RP2: El usuario debe poder grabar una pista de sonido y guardarla como un fichero en el almacenamiento del sistema.

RP2.1: El usuario podrá guardar pistas hasta un máximo de MAX_PISTAS al mismo tiempo.

RP2.1.1: Si el usuario intenta grabar una pista más de MAX_PISTAS, se mostrará un mensaje advirtiendo de que ha llegado al límite de pistas disponibles y no se permitirá que se inicie la grabación.

RP2.1.2: Las pistas se guardarán con un nombre predeterminado: NOMBRE_PISTA añadiendo NUM_PISTAS. Por ejemplo (Track_1, Track_2).

RP2.1.3: Las pistas se guardarán en formato .wav.

RP2.1.4: Las pistas se guardarán en una ruta establecida dentro del almacenamiento interno: RUTA_PISTAS

RP3: El usuario tendrá diversas opciones de reproducción de pistas de sonido.

RP3.1: El usuario podrá reproducir (Play) una pista seleccionada.

RP3.1.1: Únicamente se podrá reproducir una pista de cada vez.

RP3.2: El usuario podrá pausar (Pause) una pista seleccionada.

RP3.3: El usuario podrá reiniciar (Stop) una pista seleccionada.

RP3.4: El usuario podrá reanudar una pista que haya sido pausada o parada.

RP4: El usuario podrá modificar el nombre de las pistas de sonido.

RP4.1: El nuevo nombre de la pista tendrá un número de caracteres entre MIN_CARACTERES y MAX_CARACTERES.

RP4.2: El nombre de la pista estará formado por caracteres alfanuméricos.

RP4.3: Si el nuevo nombre de una pista ya existe en el sistema, se cancelará la acción y se mostrará un mensaje de advertencia al usuario.

RP4.4: Cambiar el nombre a una pista modificará también el nombre del fichero .wav del sistema que la representa.

RP5: El usuario podrá eliminar una pista del sistema.

RP5.1: Al eliminar una pista, el sistema también borrará el fichero .wav que la representa dentro del sistema de archivos.

5.2.2.2 Requisitos no funcionales

5.2.2.2.1 Requisitos tecnológicos

RNF1: La aplicación debe funcionar en dispositivos Oculus Quest (primera generación).

RNF2: El dispositivo Oculus Quest debe poseer capacidades de hand tracking.

RNF2.1: La funcionalidad de hand tracking está incluida en versiones del sistema operativo a partir de "V12". El sistema debe de tener como mínimo esta versión instalada.

RNF3: El dispositivo Oculus Quest debe tener activado el modo desarrollador para ofrecer la posibilidad de instalar y ejecutar aplicaciones no aprobadas por el fabricante.

RNF3.1: Para activar el modo desarrollador es necesario una cuenta de Facebook y vincular el dispositivo en la aplicación Oculus.

5.2.2.2.2 Requisitos de rendimiento

RNF4: El sistema debe funcionar a un framerate mínimo de MIN_FRAMERATE un 95% del tiempo.

5.2.2.2.3 Requisitos de usabilidad

RNF5: El sistema debe presentar una interfaz agradable e intuitiva para nuevos usuarios.

RNF5.1: La aplicación debe ser estática, es decir, no requiere de desplazamiento por parte del usuario más allá de acercarse a los botones de la interfaz.

RNF6: Los textos del sistema se mostrarán en inglés.

5.2.2.3 Referencia de constantes en la especificación de requisitos

En la siguiente tabla se presentan los valores de los campos numéricos obtenidos en la especificación de requisitos.

CONSTANTE	VALOR
MIN_BITCRUSH	1
MAX_BITCRUSH	16
MIN_MODULACION	0
MAX_MODULACION	1
MIN_MULTIPLICADOR	1
MAX_MULTIPLICADOR	24
MIN_FRECUENCIA	440Hz
MAX_FRECUENCIA	16 KHz
MANO_NO_DOMINANTE	Mano izquierda
MANO_DOMINANTE	Mano derecha
MAX_PISTAS	3
RUTA_PISTAS	/Android/data/com.jatorrasagasti.SynthVR/data/tracks
NOMBRE_PISTAS	Track_
NUM_PISTAS	0
MIN_CARACTERES	1
MAX_CARACTERES	10
MIN_FRAMERATE	60

Tabla 5.1 Referencia de constantes en la especificación de requisitos

5.2.3 Identificación de Actores del Sistema

En este proyecto se identifica claramente un único actor del sistema, en este caso, el usuario que interactúa con él.

5.2.4 Especificación de Casos de Uso

Se dividen los diagramas de casos de uso en dos secciones: uso del sistema de audio (Figura 5.1) y gestión de pistas (Figura 5.2).

5.2.4.1 Uso del sistema de audio

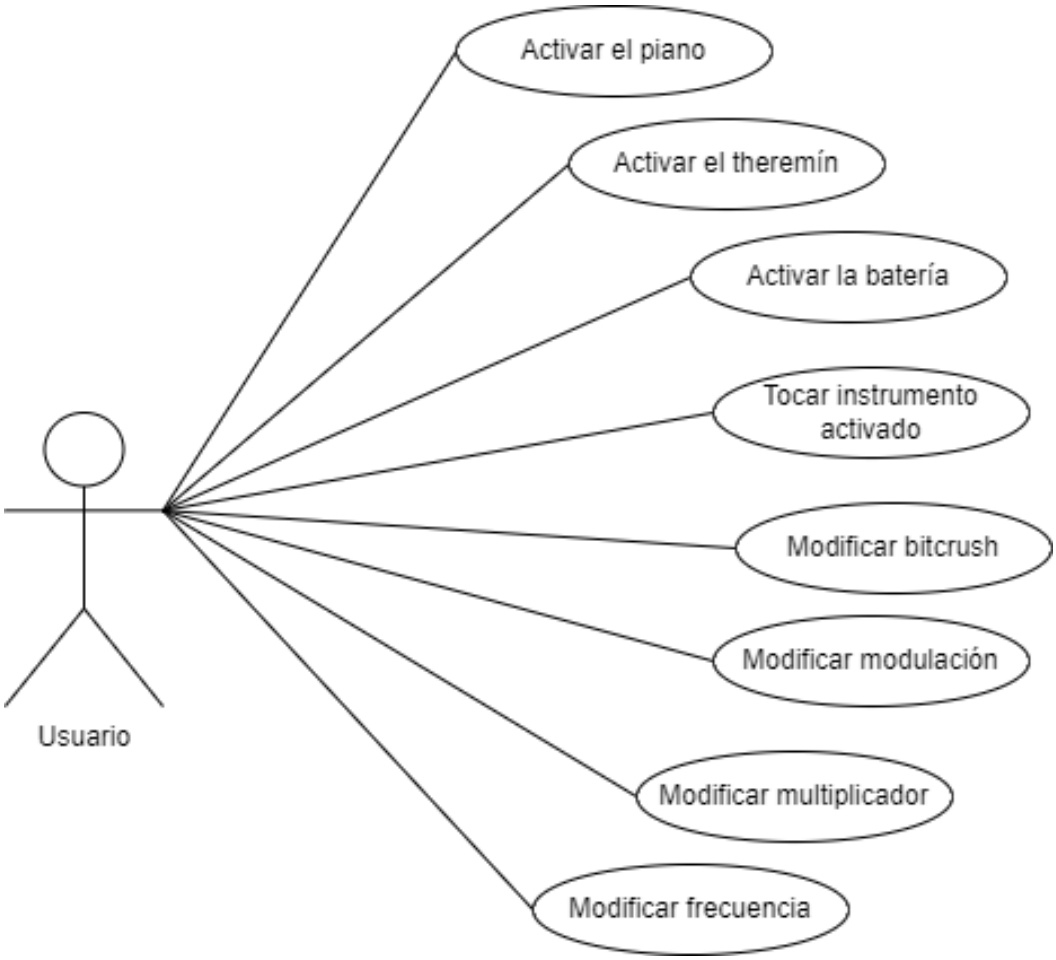


Figura 5.1 Casos de uso relativos al uso del sistema de audio

A continuación, se describe brevemente cada uno de los casos de uso relativos al uso del sistema de audio:

Nombre del Caso de Uso
CUA01 – Activar el piano
Descripción
El usuario activará el piano utilizando el elemento de la interfaz correspondiente. Se desactivará el instrumento que esté activado en ese momento. Aparecerán los modificadores del sintetizador correspondientes al lado del usuario.

Nombre del Caso de Uso
CUA02 – Activar el theremín
Descripción
El usuario activará el theremín utilizando el elemento de la interfaz correspondiente. Se desactivará el instrumento que esté activado en ese momento. Aparecerá un elemento informativo de la interfaz que explique cómo usar el theremín.

Nombre del Caso de Uso
CUA03 – Activar la batería
Descripción
El usuario activará la batería utilizando el elemento de la interfaz correspondiente. Se desactivará el instrumento que esté activado en ese momento. Se desactivarán todos los modificadores del sintetizador de la escena en caso de que estén activados.

Nombre del Caso de Uso
CUA04 – Tocar el instrumento activado
Descripción
El usuario activará cualquiera de los instrumentos disponibles a través del elemento de interfaz correspondiente. Una vez activado el instrumento, el usuario debe poder generar sonido a través de la interfaz y métodos de entrada propios de este.

Nombre del Caso de Uso
CUA05 – Modificar bitcrush
Descripción
El usuario activará el instrumento piano. Aparecerán los dos modificadores del sintetizador correspondientes (bitcrush y modulación). Arrastrando el deslizador arriba o abajo, se actualizarán los valores del modificador “bitcrush” dependiendo de su posición en el espacio.

Nombre del Caso de Uso
CUA06 – Modificar modulación
Descripción
El usuario activará el instrumento piano. Aparecerán los dos modificadores del sintetizador correspondientes (bitcrush y modulación).

Arrastrando el deslizador arriba o abajo, se actualizarán los valores del modificador "modulación" del oscilador dependiendo de su posición en el espacio.

Nombre del Caso de Uso
CUA07 – Modificar multiplicador
Descripción
El usuario activará el instrumento theremín. Dependiendo de la posición de su mano no dominante, y si está realizando el gesto "pinch", el usuario modificará el valor del módulo "multiplicador" del oscilador.

Nombre del Caso de Uso
CUA08 – Modificar frecuencia
Descripción
El usuario activará el instrumento theremín. Dependiendo de la posición de su mano dominante, se actualizará el valor del campo "frecuencia" del sintetizador.

5.2.4.2 Gestión de pistas

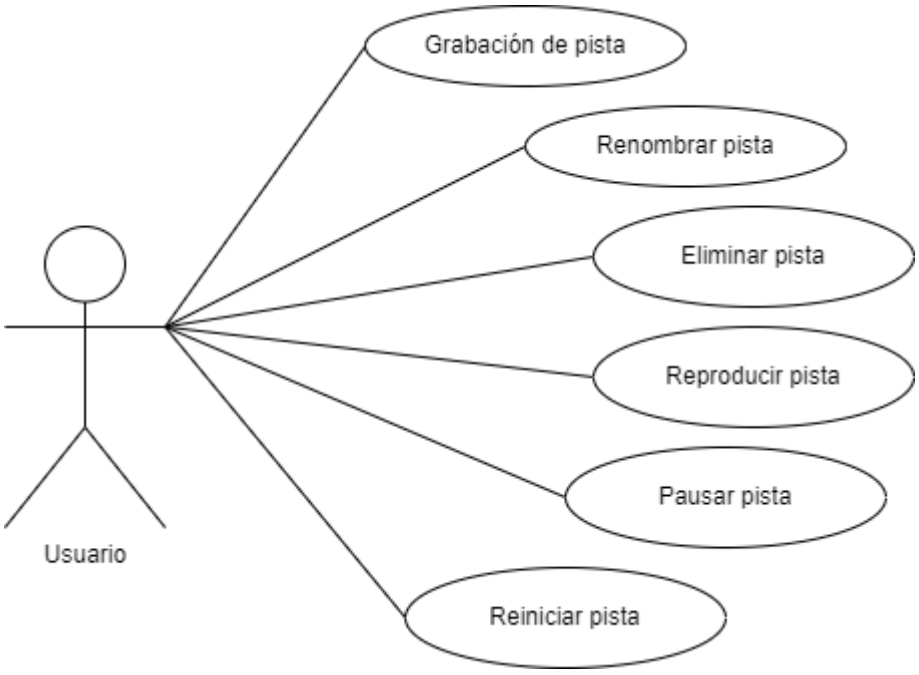


Figura 5.2 Casos de uso relativos a la gestión de pistas

A continuación, se describe brevemente cada uno de los casos de uso relativos a la gestión de pistas:

Nombre del Caso de Uso
CUP01 – Grabación de pista
Descripción
El usuario podrá grabar el sonido reproducido por el instrumento activado en cada momento. Al terminar la grabación, se genera un archivo de audio siguiendo un formato de nombre predeterminado y se actualiza el listado de pistas de la interfaz.
Excepción
Si se excede el número máximo de pistas, se informará al usuario y no se permitirá grabar ninguna más hasta que se elimine una de las existentes. No se permitirá la grabación de una pista si hay alguna en reproducción en el sistema.

Nombre del Caso de Uso
CUP02 – Renombrar pista
Descripción
El usuario podrá cambiar el nombre de cualquiera de las pistas. El cambio de nombre ocurre tanto en la interfaz del sistema como en el sistema de archivos.
Excepción
Si el nombre ya existe, se informará al usuario y la modificación no se realizará. Si el nombre excede el número máximo de caracteres o es menor que el límite establecido, se informará al usuario y la modificación no se realizará. No se permitirá la modificación de la pista si hay alguna en reproducción en el sistema.

Nombre del Caso de Uso
CUP03 – Eliminar pista
Descripción
El usuario podrá eliminar pistas de audio generadas previamente o que ya se encuentren en el listado de pistas (pueden ser introducidas por el sistema de archivos). La pista desaparecerá de la lista y será eliminada del sistema de archivos. No se permitirá la eliminación de la pista si hay alguna en reproducción en el sistema.

Nombre del Caso de Uso
CUP04 – Reproducir pista
Descripción
El usuario podrá reproducir pistas de audio generadas previamente o que ya se encuentren en el listado de pistas (pueden ser introducidas por el sistema de archivos). El sistema solo permitirá la reproducción de una pista a la vez.

Nombre del Caso de Uso
CUP05 – Pausar pista
Descripción
El usuario podrá pausar una pista de audio en reproducción. Cuando se reanude la reproducción, la pista continuará en el punto en el que se pausó.

Nombre del Caso de Uso
CUP06 – Reiniciar pista
Descripción
El usuario podrá parar una pista de audio en reproducción. Cuando se reanude la reproducción, la pista comenzará de nuevo desde el principio.

5.3 Identificación de los Subsistemas en la Fase de Análisis

5.3.1 Descripción de los Subsistemas

5.3.1.1 Interfaz/Entorno

Si bien el entorno puede no parecer un subsistema en un principio, en esta aplicación (y debido al funcionamiento inherente del motor Unity), gran parte de la lógica que se ejecuta por interacción del usuario está fuertemente acoplada a la interfaz (o programada sobre ella directamente). Los elementos del entorno contienen identificadores y propiedades que proporcionan información sobre qué elementos activar o desactivar, qué nota tiene que tocar el sintetizador, qué sonido de batería tiene que sonar a cada momento o hace llamadas a los métodos del gestor de pistas, por lo que también interactúa con el sistema de persistencia.

5.3.1.2 Oculus XR Plugin

El SDK de Oculus proporciona las interfaces y componentes Unity necesarios para una correcta integración de la aplicación en realidad virtual con dispositivos Oculus. Es el encargado de gestionar la posición del usuario en el entorno, el hand tracking y los eventos asociados.

5.3.1.3 Audio

El subsistema Audio es el subsistema encargado de toda la parte de generación de sonido por código. Recibe los eventos generados por los componentes de los instrumentos de la interfaz, genera el sonido solicitado y lo reproduce a través de componentes diferentes de esta. Comprende las clases y componentes relacionados con los módulos y modificadores de los instrumentos.

5.3.1.4 Gestor de pistas

El gestor de pistas es el encargado de manejar todo lo relacionado con el sistema de persistencia y reproducción y gestión de pistas en tiempo de ejecución. Tiene las funcionalidades de crear, modificar y eliminar ficheros y pistas, así como reproducirlas, pararlas o reiniciarlas.

5.3.2 Descripción de los Interfaces entre Subsistemas

Dada la naturaleza del proyecto, y que va a ser desarrollado en el entorno Unity, todos los subsistemas e interfaces son implementados como clases en C#. La aplicación no tiene ningún tipo de funcionalidad online por lo que toda comunicación se realiza localmente. Tampoco hay conexiones con bases de datos externas.

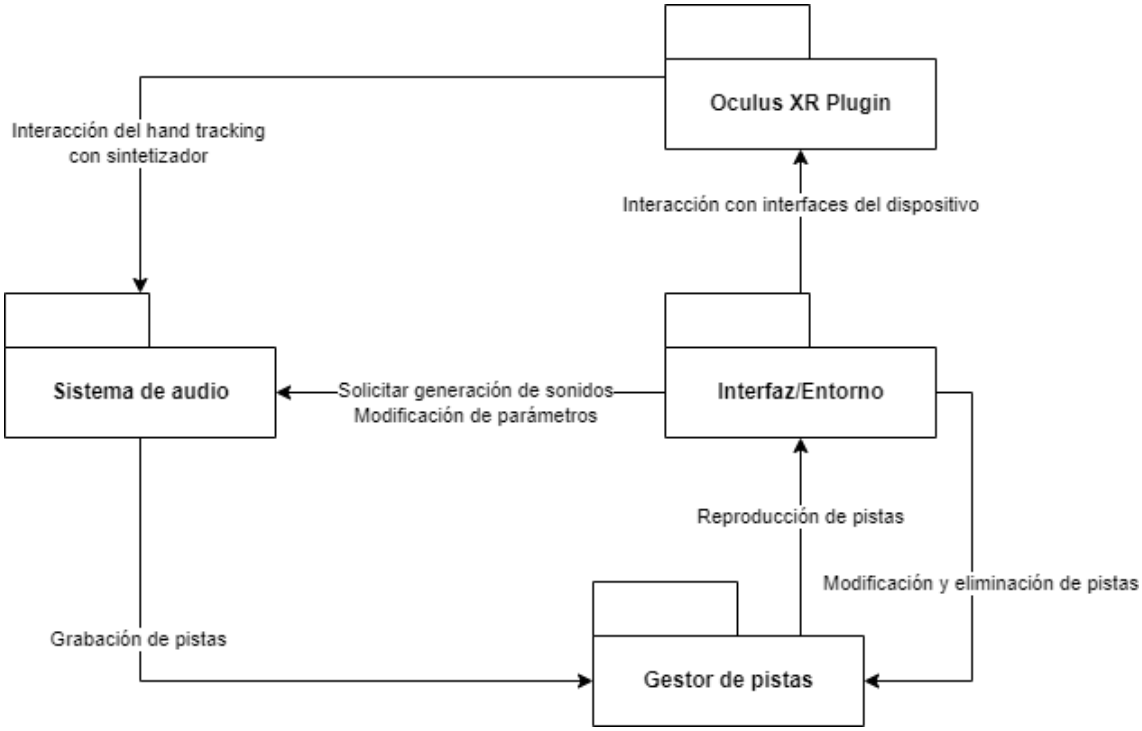


Figura 5.3 Diagrama preliminar de dependencia de subsistemas

En el diagrama mostrado en la Figura 5.3, se describen las relaciones que existen entre los distintos subsistemas: el entorno tiene un papel principal en la aplicación, ya que la interfaz de usuario se comunica con todos los demás aspectos de la aplicación. El Oculus XR Plugin se relaciona directamente con la interfaz ya que es el método de entrada único obtenido a través del dispositivo y el hand tracking. Por último, el motor de audio y el gestor de pistas están fuertemente relacionados ya que tienen funciones complementarias y relacionadas con la reproducción, grabación y gestión del audio.

5.4 Diagrama de Clases Preliminar del Análisis

A continuación, se identificarán posibles clases de los subsistemas descritos en la sección 5.3.

5.4.1 Diagramas de Clases

Se separarán los diagramas por cada subsistema, haciendo referencia a las interfaces entre distintos subsistemas. Estos diagramas se muestran en las figuras desde la Figura 5.4 hasta la Figura 5.6.

No se hará un diagrama del paquete Oculus XR Plugin puesto que es un componente externo y ya desarrollado, aunque sí se referenciará cuando sea necesario

5.4.1.1 Subsistema Interfaz/Entorno

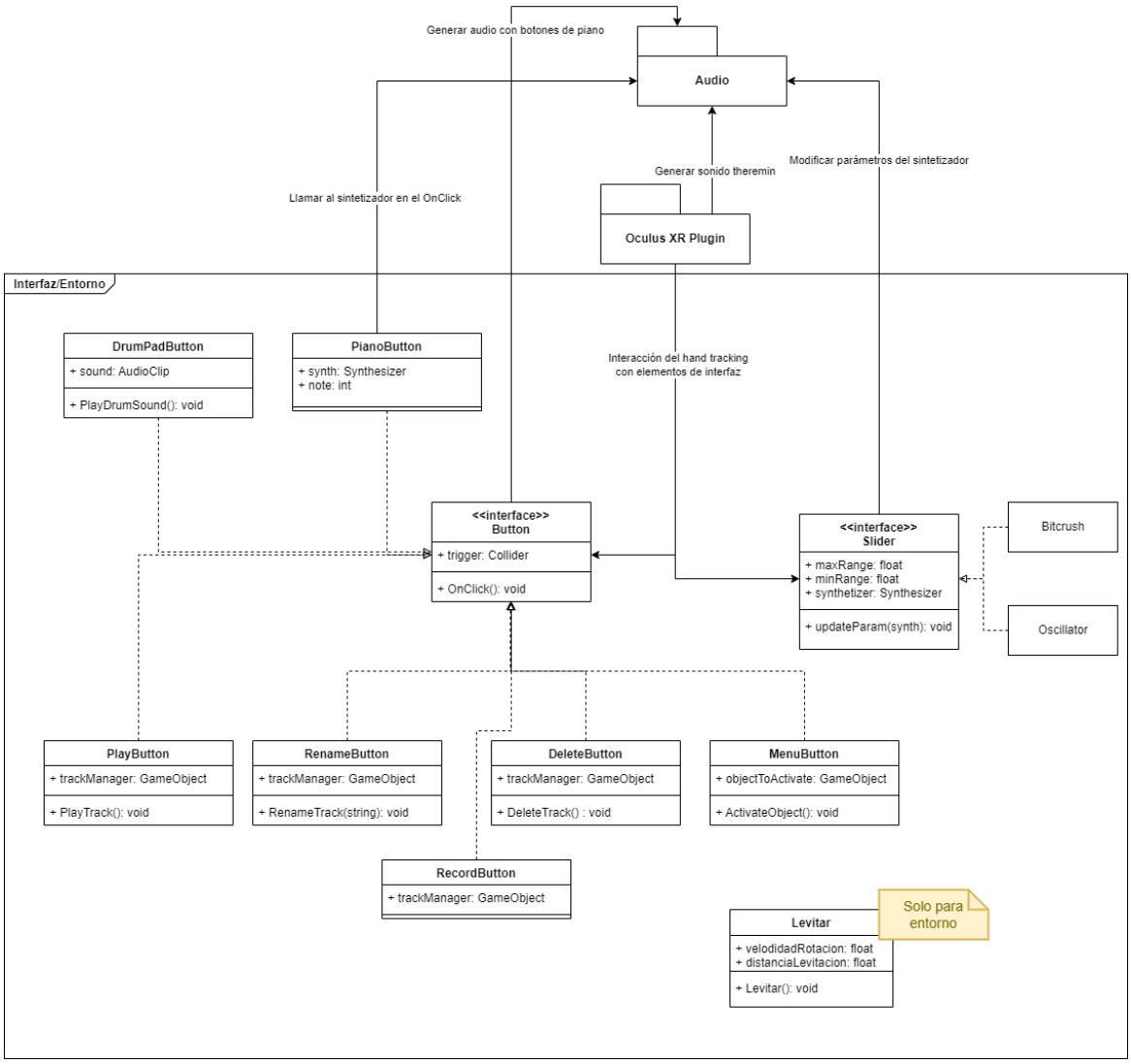


Figura 5.4 Diagrama de clases preliminar del subsistema Interfaz/Entorno

5.4.1.2 Subsistema Audio

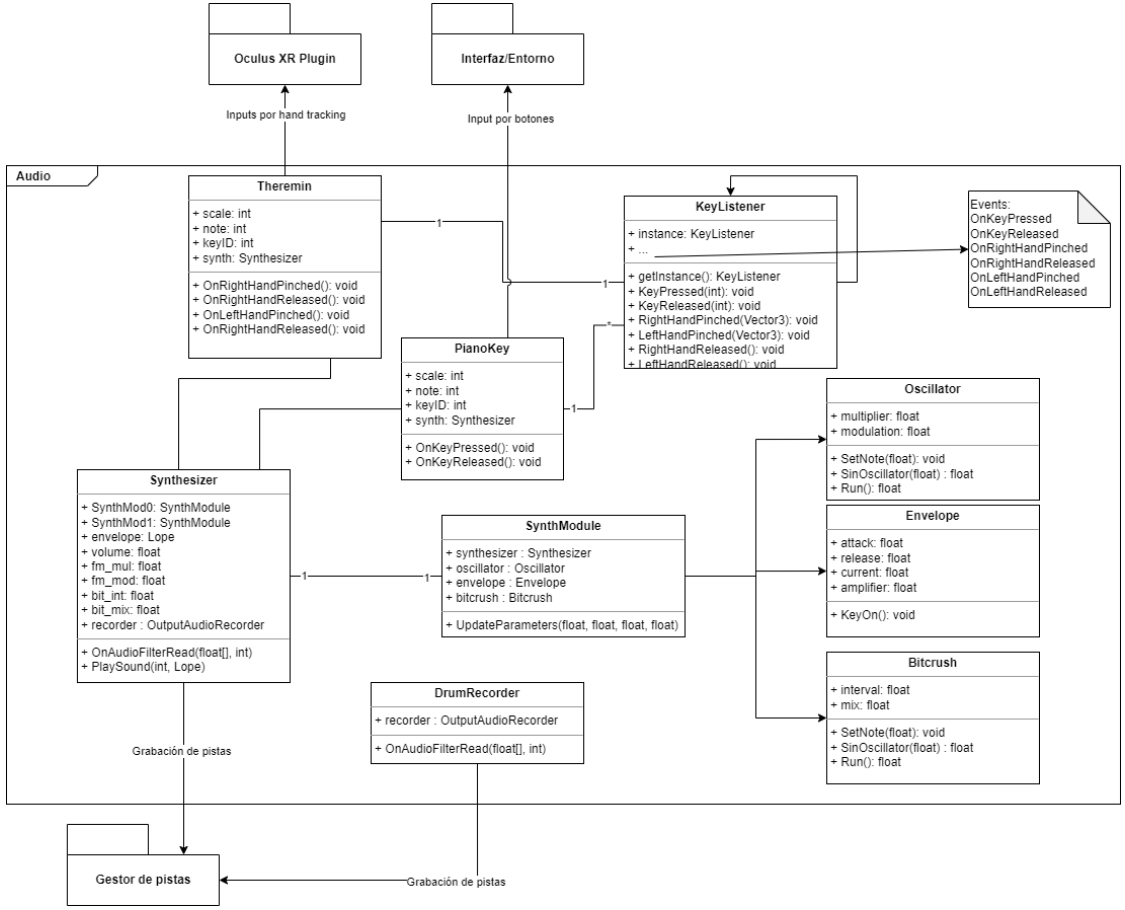


Figura 5.5 Diagrama de clases preliminar del subsistema Audio

5.4.1.3 Subsistema Gestor de pistas

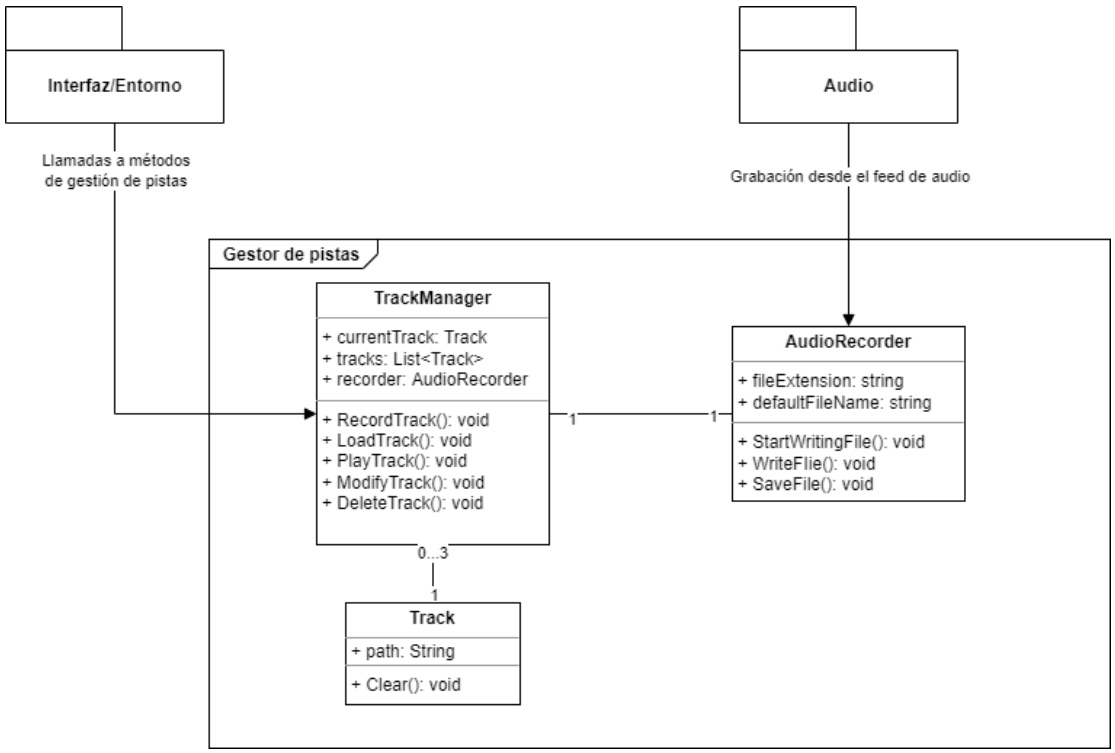


Figura 5.6 Diagrama de clases preliminar del subsistema Gestor de pistas

5.4.2 Descripción de las Clases

5.4.2.1 Interfaz/Entorno

Nombre de la Clase
Button
Descripción
El botón será una de las interfaces principales del sistema. Desde estos botones se podrán activar distintos objetos, simular teclas de piano o pads de batería. Se generarán varias clases derivadas que implementen distintos comportamientos dependiendo del escenario de uso.
Responsabilidades
Proporcionar una interfaz de interacción con distintos componentes del sistema.
Atributos Propuestos
Trigger: Punto de contacto a partir del cual se registra que se ha pulsado el botón.
Métodos Propuestos
OnClick: Método a implementar por las distintas clases derivadas que se ejecutará cuando se alcance el objeto trigger por parte del botón.

Nombre de la Clase
PlayButton
Descripción
Botón utilizado para reproducir una pista.
Responsabilidades
Comunicarse con el gestor de pistas para reproducir una pista de audio.
Atributos Propuestos
TrackManager: Referencia al gestor de pistas.
Métodos Propuestos
PlayTrack: Lanzado dentro de OnClick, envía la señal de reproducción al gestor de pistas.

Nombre de la Clase
RenameButton
Descripción
Botón utilizado para renombrar una pista.
Responsabilidades
Comunicarse con el gestor de pistas para renombrar una pista de audio.
Atributos Propuestos
TrackManager: Referencia al gestor de pistas.
Métodos Propuestos
RenameTrack: Lanzado dentro de OnClick, envía la señal de cambio de nombre al gestor de pistas.

Nombre de la Clase
DeleteButton
Descripción
Botón utilizado para eliminar una pista.
Responsabilidades

Comunicarse con el gestor de pistas para eliminar una pista de audio.
Atributos Propuestos
TrackManager: Referencia al gestor de pistas.
Métodos Propuestos
RenameTrack: Lanzado dentro de OnClick, envía la señal de eliminar pista al gestor de pistas.

Nombre de la Clase
RecordButton
Descripción
Botón utilizado para grabar una pista.
Responsabilidades
Comunicarse con el gestor de pistas para grabar una pista de audio.
Atributos Propuestos
TrackManager: Referencia al gestor de pistas.
Métodos Propuestos
OnClick: Lanzar señal al gestor de pistas para comenzar/terminar una grabación.

Nombre de la Clase
DrumPadButton
Descripción
Botón que simula un pad de batería. Habrá varios con distintos sonidos en la escena.
Responsabilidades
Lanzar un clip de audio que simule un sonido de batería.
Atributos Propuestos
Sound: Clip de audio para el pad de batería.
Métodos Propuestos
PlayDrumSound: Reproduce el clip de audio de batería.

Nombre de la Clase
PianoButton
Descripción
Botón que simula una tecla de piano. Habrá varias con distintos tonos en la escena.
Responsabilidades
Lanzar una señal al sintetizador con una nota específica.
Atributos Propuestos
Synthetizer: Referencia al sintetizador. **Note:** Nota asignada a la tecla.
Métodos Propuestos
OnClick: Envía la nota específica del botón al sintetizador.

Nombre de la Clase
Slider
Descripción
Los sliders permiten al usuario modificar valores del sintetizador entre dos límites. Se generarán dos clases derivadas que implementen distintos comportamientos dependiendo del escenario de uso.
Responsabilidades

Proporcionar una interfaz de interacción con distintos componentes del sistema.
Atributos Propuestos
MaxRange: Valor máximo que se le puede dar al parámetro asignado al slider. **MinRange:** Valor mínimo que se le puede dar al parámetro asignado al slider. **Synthetizer:** Referencia al sintetizador.
Métodos Propuestos
UpdateParams: Actualiza el sintetizador con los cambios en los parámetros

En las siguientes clases no se define ningún método nuevo. Implementan directamente UpdateParams de la clase Slider.

Nombre de la Clase
BitcrushSlider
Descripción
Modifica el módulo bitcrush en el sintetizador.
Responsabilidades
Modificar el valor de downsample del bitcrush a través de la interfaz.

Nombre de la Clase
OscillatorSlider
Descripción
Modifica el módulo oscilador en el sintetizador.
Responsabilidades
Modificar el valor de multiplicador del oscilador a través de la interfaz.

Nombre de la Clase
Levitar
Descripción
Provoca que un objeto en el entorno levite girando sobre sí mismo.
Responsabilidades
Modificar la posición y rotación del objeto continuamente.
Atributos Propuestos
VelocidadRotacion: Valor de velocidad de rotación del objeto alrededor del eje Y. **DistanciaLevitacion:** Distancia máxima a la que flotará el objeto desde su origen.
Métodos Propuestos
Levitar: Modifica la posición y rotación del objeto.

5.4.2.2 Audio

Nombre de la Clase
Synthesizer
Descripción
El sintetizador es una de las clases principales del sistema: tiene todas las funcionalidades relacionadas con la generación de sonido y sus propiedades. Controla el volumen, las notas reproducidas y los distintos tipos de modulación del sonido.
Responsabilidades
Reproducir (a través de una fuente de audio) el sonido con los parámetros establecidos.
Atributos Propuestos
SynthModule0: Referencia al primer módulo de modificación de sonido acoplado al sintetizador. **SynthModule1**: Referencia al segundo módulo de modificación de sonido acoplado al sintetizador. **Envelope:** Módulo que controla la evolución de la amplitud de la onda en el tiempo. **Volume:** Atributo que representa el volumen del sintetizador. **Fm_mul:** Valor del multiplicador del oscilador. **Fm_mod:** Valor del modulador del oscilador. **Bit_int:** Valor del intervalo del bitcrush. **Bit_mix:** Valor de la mezcla del bitcrush. **Recorder:** Referencia a la clase encargada de grabación de pistas.
Métodos Propuestos
OnAudioFilterRead: Captura y filtra la onda recibida por el componente Envelope y le aplica los módulos asignados al sintetizador. En caso de estar grabando, envía el flujo de datos al grabador de audio. **PlaySound**: Simula el pulsado de una tecla en un sintetizador. Se envía la nota deseada al Envelope.

Nombre de la Clase
SynthModule
Descripción
Los módulos se añaden al sintetizador para modificar la onda de audio generada ajustando valores de los distintos modificadores.
Responsabilidades
Generar y modificar la notas producidas por el componente Envelope de distintas maneras.
Atributos Propuestos
Synthesizer: Referencia al sintetizador. **Oscillator:** Referencia al componente Oscillator. **Envelope:** Referencia al componente Envelope. **Bitcrush:** Referencia al componente Bitcrush.
Métodos Propuestos
UpdateParameters: Actualiza los parámetros del módulo (atributos) con los valores solicitados.

Nombre de la Clase
Oscillator
Descripción
Componente que se puede añadir a un módulo para generar un efecto de oscilación de tipo sinoidal.
Responsabilidades
Modificar la nota producida por el componente Envelope con un efecto de oscilación
Atributos Propuestos
Multiplier: Referencia al sintetizador. **Modulation:** Referencia al componente Oscillator.
Métodos Propuestos
SetNote: Genera la nota solicitada. **SinOscillator**: Genera la onda sinoidal. **Run**: Aplica la nota y la modula con la onda sinoidal generada.

Nombre de la Clase
Envelope
Descripción
Componente que debe añadirse a un módulo de sintetizador para poder generar notas musicales. Controla el ataque, caída, sustain y release de la nota.
Responsabilidades
Generar una nota musical con distintos parámetros relacionados a cómo se comporta la nota con el tiempo.
Atributos Propuestos
Multiplier: Referencia al sintetizador. **Modulation:** Referencia al componente Oscillator.
Métodos Propuestos
SetNote: Genera la nota solicitada. **SinOscillator**: Genera la onda sinoidal. **Run**: Aplica la nota y la modula con la onda sinoidal generada.

Nombre de la Clase
Bitcrush
Descripción
Componente que se puede añadir a un módulo para generar un efecto de distorsión de tipo bitcrush.
Responsabilidades
Modificar la nota producida por el componente Envelope con un efecto de bitcrush.
Atributos Propuestos
Interval: Atributo que reduce (downsample) la señal recibida. **Mix:** Nivel de mezcla del efecto.
Métodos Propuestos
Run: Aplica la nota de entrada y la modula con la distorsión bitcrush generada.

Nombre de la Clase
KeyListener
Descripción
Clase encargada de gestionar a los eventos generados por los instrumentos piano y theremin. Invoca a las implementaciones relacionadas con las notas generadas en cada instrumento.
Responsabilidades
Definir las acciones a las que se suscriben los instrumentos y enviar las notas generadas a los componentes generadores de audio correspondientes.
Atributos Propuestos
KeyListener: Singleton de la clase. **OnKeyPressed:** Evento que se lanza cuando se pulsa una tecla. **OnKeyReleased:** Evento que se lanza cuando se suelta una tecla. **OnKeyPushed:** Evento que se lanza cuando se mantiene pulsada una tecla. **OnRightHandPinched:** Evento que se lanza cuando se hace el gesto pinch con la mano derecha. **OnRightHandReleased:** Evento que se lanza cuando se deja de hacer el gesto pinch con la mano derecha. **OnLeftHandPinched:** Evento que se lanza cuando se hace el gesto pinch con la mano izquierda. **OnLeftHandReleased:** Evento que se lanza cuando se deja de hacer el gesto pinch con la mano izquierda.
Métodos Propuestos
KeyPressed: Invoca la implementación de la clase suscrita al evento **OnKeyPressed.** **KeyPushed**: Invoca la implementación de la clase suscrita al evento **OnKeyPushed.** **KeyReleased**: Invoca la implementación de la clase suscrita al evento **OnKeyReleased.** **RightHandPinched**: Invoca la implementación de la clase suscrita al evento **OnRightHandPinched.** **RightHandReleased**: Invoca la implementación de la clase suscrita al evento **OnRightHandReleased.** **LeftHandPinched**: Invoca la implementación de la clase suscrita al evento **OnLeftHandPinched.** **LeftHandReleased**: Invoca la implementación de la clase suscrita al evento **OnLeftHandReleased.**

Nombre de la Clase
PianoKey
Descripción
Clase que modela una tecla de piano. Contiene atributos como la escala o la nota tocada actualmente.
Responsabilidades
Simular una tecla de piano y lanzar el evento correspondiente cuando se toca una nota.
Atributos Propuestos
Scale: Escala en la que va a escogerse la nota. **Note:** Nota musical que se va a tocar dentro de una escala. **KeyID:** Identificador de la tecla de piano. **Synthesizer:** Referencia al sintetizador.
Métodos Propuestos
KeyPressed: Lanza una nota al sintetizador. Es la implementación del evento OnKeyPressed y OnKeyPushed. **KeyReleased**: Deja de enviar una nota al sintetizador. Implementa el evento OnKeyReleased.
Nombre de la Clase

Theremin
Descripción
Clase que modela el theremin. Contiene atributos como la escala o la nota tocada actualmente.
Responsabilidades
Modificar la nota producida por el componente Envelope con un efecto de bitcrush.
Atributos Propuestos
Scale: Escala en la que va a escogerse la nota. **Note:** Nota musical que se va a tocar dentro de una escala.
Métodos Propuestos
KeyPressed: Lanza una nota al sintetizador. Es la implementación del evento OnKeyPressed y OnKeyPushed. **KeyReleased**: Deja de enviar una nota al sintetizador. Implementa el evento OnKeyReleased.

Nombre de la Clase
DrumRecorder
Descripción
Componente que permite la grabación de las pistas de percusión.
Responsabilidades
Enviar al grabador de audio del gestor de pistas el flujo de datos del instrumento de percusión.
Atributos Propuestos
Recorder: Referencia al grabador de audio.
Métodos Propuestos
OnAudioFilterRead: Captura el audio enviado por el sistema y lo envía al grabador de audio.

5.4.2.3 Gestor de pistas

Nombre de la Clase
TrackManager
Descripción
Clase que permite realizar distintas operaciones sobre las pistas de audio: grabar, reproducir, cargar, pausar o hacer stop (volver al principio)
Responsabilidades
Gestionar las pistas en forma de ficheros de audio.
Atributos Propuestos
CurrentTrack: Pista que está sonando actualmente. **Tracks**: Lista de pistas cargadas. **Recorder**: Referencia al grabador de audio.
Métodos Propuestos
RecordTrack: Graba una pista de audio. **LoadTrack**: Carga una pista del sistema de ficheros. **PlayTrack**: Reproduce/reanuda una pista de audio. **ModifyTrack**: Renombra una pista de audio. **PauseTrack:** Pausa una pista de audio. **StopTrack**: Reinicia una pista de audio. **DeleteTrack**: Elimina una pista de audio.

Nombre de la Clase
Track
Descripción
Clase que modela una pista de audio.
Responsabilidades
Representa una pista y guarda una referencia a su ruta en el sistema de archivos.
Atributos Propuestos
Path: Ruta en la que se almacena el fichero.
Métodos Propuestos
Ninguno .

Nombre de la Clase
AudioRecorder
Descripción
Clase que realiza todas las operaciones de guardado de archivos .wav en tiempo real.
Responsabilidades
Escribir en el sistema de archivos en el formato seleccionado, manejar la correcta creación del archivo.
Atributos Propuestos
FileExtension: Extensión del archivo. **DefaultFileName**: Nombre de archivo por defecto.
Métodos Propuestos
StartWritingFile: Inicia la grabación, crea el archivo inicial. **WriteFile**: Escribe sobre el archivo en cada frame. **SaveFile**: Termina la grabación y guarda el archivo.

5.5 Análisis de Casos de Uso y Escenarios

5.5.1 Casos de uso

5.5.1.1 Uso del sintetizador

CUA01. Activar el piano	
Precondiciones	-
Poscondiciones	El instrumento “piano” está activado.
Actores	Usuario
Descripción	El usuario: 1. Activa el panel de instrumentos. 2. Selecciona el instrumento piano desde el panel. 3. Se desactiva el instrumento activo en ese momento si lo hubiera y se activa el piano.
Notas	-

CUA02. Activar el theremín	
Precondiciones	-
Poscondiciones	El instrumento “theremín” está activado.
Actores	Usuario
Descripción	El usuario: 1. Activa el panel de instrumentos. 2. Selecciona el instrumento piano desde el panel. 3. Se desactiva el instrumento activo en ese momento si lo hubiera y se activa el theremín.
Notas	-

CUA03. Activar la batería	
Precondiciones	-
Poscondiciones	El instrumento “batería” está activado.
Actores	Usuario
Descripción	El usuario: 1. Activa el panel de instrumentos. 2. Selecciona el instrumento piano desde el panel. 3. Se desactiva el instrumento activo en ese momento si lo hubiera y se activa la batería.
Notas	-

CUA04. Tocar el instrumento activado	
Precondiciones	Debe haberse cumplido cualquiera de los casos CUS01, CUS02 o CUS03.
Poscondiciones	Se genera un sonido en el sistema.
Actores	Usuario
Descripción	El usuario: 1. Se coloca frente al instrumento activado en ese momento. 2. Interactúa con la interfaz correspondiente para tocar el instrumento .
Variaciones (escenarios secundarios)	• **Instrumento “Piano”**: El usuario interactúa con las teclas del piano físicamente (empujando los botones). • **Instrumento “Batería”**: El usuario interactúa con los pads de la batería físicamente (empujando los botones). • **Instrumento “Theremin”**: El usuario interactúa con el theremin utilizando el hand tracking (haciendo el gesto “pinch” con la mano derecha).
Notas	-

CUA05. Modificar oscilador	
Precondiciones	Debe haberse cumplido cualquiera de los casos CUS01 o CUS02.
Poscondiciones	Se modifica el parámetro oscilador del sintetizador.
Actores	Usuario

Descripción	El usuario: 1. Se coloca frente al panel de modificadores. 2. Interactúa con el elemento de interfaz correspondiente y se modifica el parámetro oscilador.
Variaciones (escenarios secundarios)	• **Si el instrumento "Piano" está activado**: El usuario interactúa con el deslizador del piano físicamente (empujándolo de arriba a abajo). • **Instrumento "Theremin"**: El usuario modifica el parámetro utilizando el hand tracking (haciendo el gesto "pinch" con la mano izquierda).
Notas	-

CUA06. Modificar bitcrush	
Precondiciones	Debe haberse cumplido el caso CUS01
Poscondiciones	Se modifica el parámetro bitcrush del sintetizador
Actores	Usuario
Descripción	El usuario: 1. Se coloca frente al panel de modificadores 2. Interactúa con el deslizador correspondiente y se modifica el parámetro bitcrush
Notas	-

CUA07. Modificar multiplicador	
Precondiciones	Debe haberse cumplido el caso CUS02.
Poscondiciones	Se modifica el parámetro multiplicador del módulo oscilador del sintetizador.
Actores	Usuario
Descripción	El usuario: 1. El usuario interactúa con el theremin utilizando el hand tracking (haciendo el gesto "pinch" con la mano izquierda). 2. La el multiplicador del oscilador se modifica dependiendo de la posición de la mano.
Notas	-

CUA08. Modificar frecuencia	
Precondiciones	Debe haberse cumplido el caso CUS02.
Poscondiciones	Se modifica el parámetro frecuencia del sintetizador.
Actores	Usuario
Descripción	El usuario: 1. El usuario interactúa con el theremin utilizando el hand tracking (haciendo el gesto "pinch" con la mano derecha). 2. La frecuencia del theremín se modifica dependiendo de la posición de la mano.
Notas	-

5.5.1.2 Gestión de pistas

CUP01. Grabación de pista	
Precondiciones	Debe de haberse cumplido cualquiera de los casos CUS01, CUS02, o CUS03. No debe haber ninguna pista en reproducción en ese momento.
Poscondiciones	Se crea una nueva pista de audio en el sistema de ficheros.
Actores	Usuario
Descripción	El usuario: 1. Activa el panel de gestión de pistas. 2. Presiona el botón de grabación. 3. Interactúa con el instrumento elegido y se genera un sonido (CUS04). 4. Presiona el botón de grabación de nuevo. 5. Se genera una nueva pista de audio.
Excepciones	• **Número máximo de pistas alcanzado**: no se permitirá la grabación de una nueva pista. Se mostrará un mensaje de advertencia al usuario.
Notas	-

CUP02. Renombrar pista	
Precondiciones	Debe haber al menos una pista cargada en el sistema. No debe haber ninguna pista en reproducción en ese momento.
Poscondiciones	La pista selecionada habrá cambiado de nombre de fichero.
Actores	Usuario
Descripción	El usuario: 1. Activa el panel de gestión de pistas. 2. Pulsa el botón de "renombrar" correspondiente a una pista en particular. 3. Aparece un teclado y el usuario escribe el nuevo nombre. 4. Al pulsar "enter" en el teclado virtual, el nombre del fichero habrá cambiado al nuevo.
Excepciones	• **Número de caracteres no válido**: no se permitirá renombrar la pista. Se mostrará un mensaje de advertencia al usuario.

	• **Nombre de pista ya existente**: no se permitirá renombrar la pista. Se mostrará un mensaje de advertencia al usuario.
CU03. Eliminar pista	
Precondiciones	Debe haber al menos una pista cargada en el sistema. No debe haber ninguna pista en reproducción en ese momento.
Poscondiciones	La pista seleccionada será eliminada de la aplicación y del sistema de archivos.
Actores	Usuario
Descripción	El usuario: 1. Activa el panel de gestión de pistas. 2. Pulsa el botón de "eliminar" correspondiente a una pista en particular. 3. La pista seleccionada se eliminará de la aplicación y del sistema de ficheros.
Notas	-

CU04. Reproducir pista	
Precondiciones	Debe haber al menos una pista cargada en el sistema. No debe haber ninguna pista en reproducción en ese momento.
Poscondiciones	La pista seleccionada se reproducirá una vez.
Actores	Usuario
Descripción	El usuario: 1. Activa el panel de gestión de pistas. 2. Pulsa el botón de "reproducir" correspondiente a una pista en particular. 3. La pista seleccionada se reproducirá una vez a través de los componentes de reproducción de audio.
Notas	-

CU05. Pausar pista	
Precondiciones	Debe haber una pista en reproducción en ese momento.
Poscondiciones	La pista seleccionada se pausará.
Actores	Usuario
Descripción	El usuario: 1. Activa el panel de gestión de pistas. 2. Pulsa el botón de "pausa" correspondiente a una pista en particular. 3. La pista seleccionada se pausará. Una vez se reanude la reproducción pulsando en el botón, la pista continuará en el punto en el que se pausó.
Notas	-

CU06. Reiniciar pista	
Precondiciones	Debe haber una pista en reproducción en ese momento.
Poscondiciones	La pista seleccionada se parará.
Actores	Usuario

Descripción	El usuario: 1. Activa el panel de gestión de pistas. 2. Pulsa el botón de “reiniciar” correspondiente a una pista en particular. 3. La pista seleccionada se pausará. Una vez se reanude la reproducción pulsando en el botón, la pista continuará desde el principio.
Notas	-

5.6 Análisis de Interfaces de Usuario

5.6.1 Descripción de la Interfaz

La interfaz está diseñada en un entorno 3D inmersivo. Como se puede ver en la Figura 5.5, se sitúa al jugador en una zona "pedestal" desde el cual tiene acceso a todos los botones y elementos de la interfaz con los que va a interactuar.

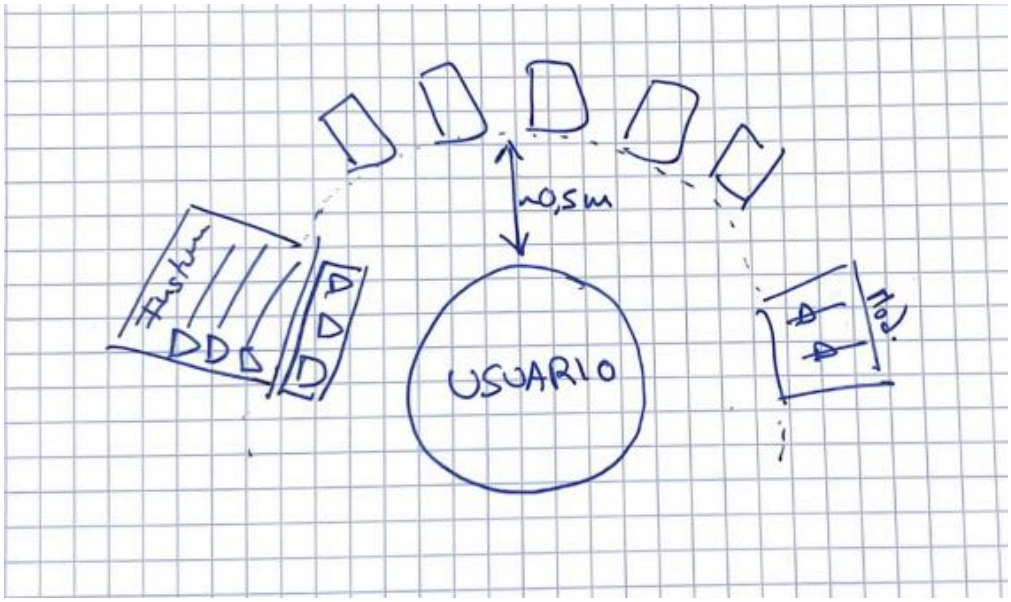


Figura 5.7 Boceto de la interfaz

Teniendo en cuenta que la aplicación está diseñada para ser utilizada únicamente con hand tracking, se procura que todos los elementos de interacción sean intuitivos de usar y estén a una distancia cercana al usuario (aproximadamente 0,5m y a la altura del torso, lo suficientemente cerca como para alcanzarlos cómodamente con las manos)

5.6.1.1 Selección de instrumentos y gestión de pistas

Siguiendo el boceto de la Figura 5.6, en la zona izquierda del pedestal existe un panel que permite seleccionar los distintos instrumentos. Presionando cada uno de los botones se desactiva el instrumento activado actualmente y se activa el instrumento seleccionado.

Del mismo modo, bajo este panel, se puede seleccionar la opción "gestor de pistas" que activará dicho panel. En este panel se podrá ver el listado de pistas guardadas, cada una de ellas mostrando su nombre y las distintas opciones disponibles a realizar sobre ellas (reproducir, modificar, eliminar).

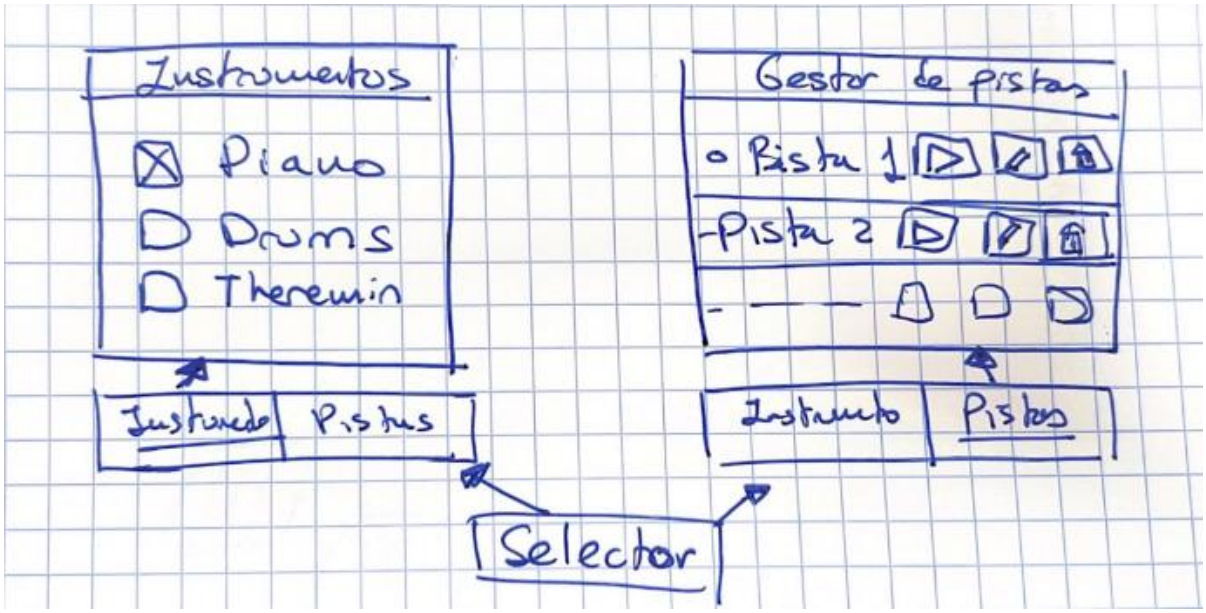


Figura 5.8 Boceto de la interfaz de selección de instrumentos y selección de pistas

5.6.1.2 Parámetros del sintetizador

Estos dos deslizadores se desplazan en su eje local Y a través de la interacción directa (con las manos) del usuario. La posición de cada uno de los deslizadores aplicará una cantidad de modificación a un parámetro específico del sintetizador.

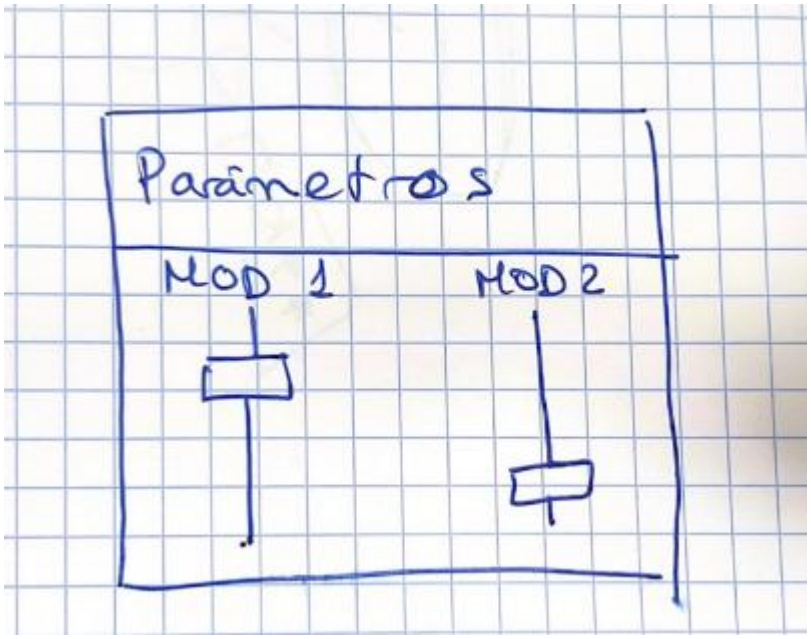


Figura 5.9 Boceto de la interfaz de modificación de parámetros

5.6.2 Comportamiento de la Interfaz

5.6.2.1 Botones y deslizadores

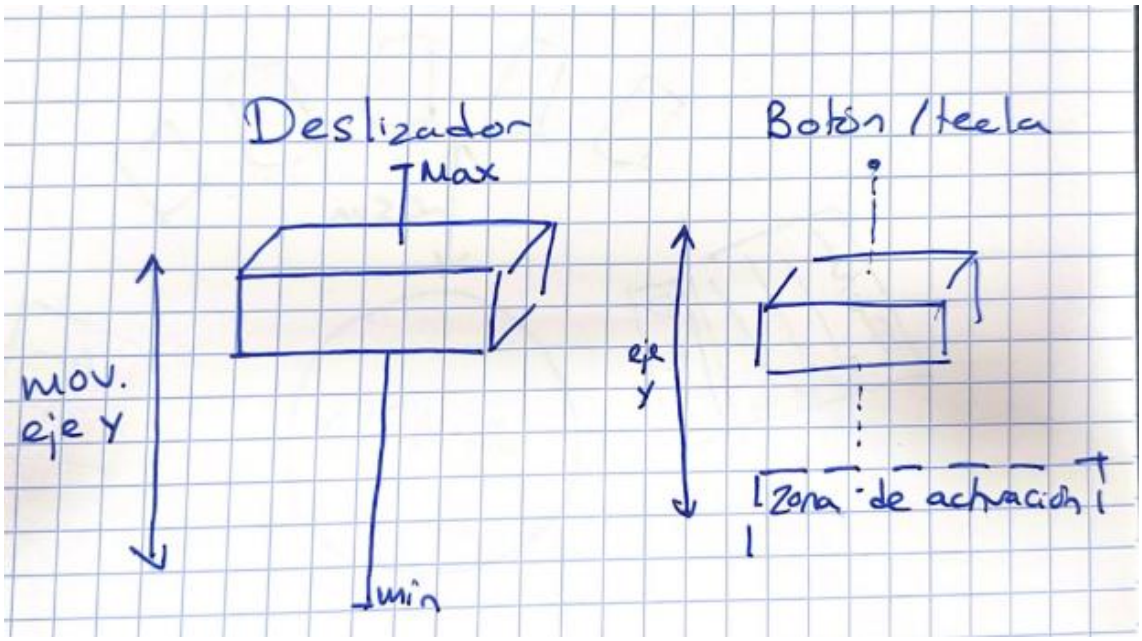


Figura 5.10 Boceto del funcionamiento de botones y deslizadores

Los dos modos de input principales que se van a usar en la aplicación serán a través de botones y deslizadores. El usuario utilizará sus manos para accionarlos de manera física (presionando o arrastrando, respectivamente).

- **Los deslizadores** tendrán una restricción que limita su movimiento al eje Y local (es decir, solo pueden moverse de arriba abajo, independientemente de que el usuario los empuje en otras direcciones). Además, estarán limitado en dos valores extremos.
- **Los botones** simulan el funcionamiento de un botón real. También están limitados en su movimiento al eje Y local, pero no tienen un valor máximo o mínimo restringido. Sin embargo, dentro de cada objeto "Botón" existe una "zona de activación" que lanzará un evento cada vez que el botón colisione físicamente con esta zona.

5.7 Especificación del Plan de Pruebas

Debido a la propia naturaleza del proyecto y de cómo se plantea su implementación, el plan de pruebas intentará seguir las directrices "estándar" cuando sea posible. Sin embargo, se espera que prácticamente toda la fase de testeo de la aplicación sea de forma manual, es decir, sin test unitarios o automatizaciones.

5.7.1 Pruebas unitarias

El motor Unity, en principio, no está pensado para ser capaz de llevar a cabo pruebas unitarias. Una aplicación Unity está compuesta en su gran mayoría por clases que heredan de la clase MonoBehavior: esta clase permite que cualquier script que herede pueda ser utilizado en el editor y añadido a cualquier GameObject de la aplicación, permitiendo al desarrollador acceder a campos y otras funcionalidades del script directamente desde el editor, lo que facilita el desarrollo de la aplicación y reduce en gran medida el tiempo de desarrollo.
El inconveniente que esto presenta se reduce a que todo aquel script que herede de MonoBehavior estará fuertemente acoplado a Unity en sí, y no podrá ser ejecutado fuera del software.

Por otra parte, existen paquetes y plugins que permiten realizar test en pequeñas clases y porciones de código; sin embargo, no es lo habitual, ya que muchos de los resultados a testear dependen principalmente del input del usuario, que por lo general es impredecible.

De este modo cabría la posibilidad de hacer pequeños Unit Test para porciones de código reducidas y simples, aunque conllevaría el desarrollo de una capa de lógica extra que separara los componentes Unity y la lógica.

Añadiendo aún más a la problemática de Unity, hay que tener en cuenta que la totalidad de acciones que el usuario puede realizar deben hacerse en realidad virtual utilizando la funcionalidad de hand tracking, lo que es virtualmente imposible de replicar en este tipo de prueba. En este caso, el sistema requeriría de alimentar de forma artificial el software con inputs de este tipo, algo que está completamente fuera del alcance del sistema y mucho menos de un plan de pruebas de lógica sencillas.

Por las razones expuestas, se decide no realizar pruebas unitarias ya que la estimación del esfuerzo a realizar en comparación con los resultados que podrían obtenerse nos dice que sería un trabajo poco eficiente.

5.7.2 Pruebas de integración

Por otro lado, la mayoría de las pruebas que se realizarán en el proyecto serán de integración entre los distintos componentes y subsistemas. Estas pruebas se harán manualmente y comprobarán el correcto funcionamiento de las clases y funcionalidades de acuerdo con los casos de uso definidos anteriormente (apartado 5.5.1, Casos de uso)

Caso de Uso CUS01 – Activar el piano

Identificador	Prueba	Procedimiento	Resultado Esperado
PI01	Activar piano	El usuario localiza el botón de activación de instrumentos. Presiona el botón "Piano".	Se activa el objeto Piano. Si hay otro instrumento activo en la escena, se desactiva.

Caso de Uso CUS02 – Activar el theremín

Identificador	Prueba	Procedimiento	Resultado Esperado
PI02	Activar theremín	El usuario localiza el botón de activación de instrumentos. Presiona el botón "Theremin".	Se activa el objeto Theremin. Si hay otro instrumento activo en la escena, se desactiva.

Caso de Uso CUS03 – Activar la batería

Identificador	Prueba	Procedimiento	Resultado Esperado
PI03	Activar batería	El usuario localiza el botón de activación de instrumentos. Presiona el botón "Drums".	Se activa el objeto Batería. Si hay otro instrumento activo en la escena, se desactiva.

Caso de Uso CUS04 – Tocar instrumento activado

Identificador	Prueba	Procedimiento	Resultado Esperado
PI04.1	Tocar instrumento Piano	El usuario ha activado el Piano previamente. Para tocar una nota, pulsa con la mano en cualquiera de las teclas del piano.	Se produce un sonido específico para cada tecla del piano.
PI04.2	Tocar instrumento Theremín	El usuario ha activado el Theremín previamente. Para tocar una nota, realiza el gesto "pinch" con su mano derecha.	Se produce un sonido con una frecuencia en particular dependiendo de la posición de la mano en el espacio.
PI04.3	Tocar instrumento Batería	El usuario ha activado la Batería previamente. Para tocar una nota, pulsa con la mano en	Se produce un sonido específico para cada pad de batería.

		cualquiera de los pads de batería.	

Caso de Uso CUS05 – Modificar oscilador

Identificador	Prueba	Procedimiento	Resultado Esperado
PI05.1	Modificar valor del oscilador (piano)	El usuario ha activado el Piano previamente. El usuario localiza el deslizador correspondiente al modificador del oscilador. Lo empuja con la mano verticalmente hacia arriba o hacia abajo.	El valor multiplicador del oscilador varía dependiendo de la posición en la que se encuentre.

Caso de Uso CUS06 – Modificar bitcrush

Identificador	Prueba	Procedimiento	Resultado Esperado
PI06	Modificar valor del bitcrush	El usuario ha activado el Piano previamente. El usuario localiza el deslizador correspondiente al modificador del bitcrush. Lo empuja con la mano verticalmente hacia arriba o hacia abajo.	El valor de downsample del bitcrush varía dependiendo de la posición en la que se encuentre.

Caso de Uso CUA07 – Modificar multiplicador

Identificador	Prueba	Procedimiento	Resultado Esperado
PI07	Modificar valor del multiplicador	El usuario ha activado el Theremín previamente. Para modificar el valor del oscilador, realiza el gesto “pinch” con su mano izquierda (juntando dedos índice y pulgar) y mueve la mano en el eje Y.	El valor multiplicador del oscilador varía dependiendo de la posición en la que se encuentre.

Caso de Uso CUS08 – Modificar frecuencia

Identificador	Prueba	Procedimiento	Resultado Esperado
PI08	Modificar valor del bitcrush	El usuario ha activado el Theremín previamente. Para tocar una nota, realiza el gesto “pinch” con su mano derecha y	El valor de la frecuencia del theremín cambia dependiendo de la posición en la que se encuentre.

		mueve la mano en los ejes X e Y.	

Caso de Uso CUP01 – Grabación de pista

Identificador	Prueba	Procedimiento	Resultado Esperado
PI07.1	Grabar una nueva pista	El usuario ha activado el gestor de pistas previamente. El usuario localiza y pulsa el botón "grabar". Toca los instrumentos durante un tiempo y vuelve a pulsar el botón grabar.	Se crea una nueva pista en el sistema de archivos.
PI07.2	Grabar una nueva pista (límite de pistas alcanzado)	El usuario ha activado el gestor de pistas previamente. Debe haberse alcanzado el número máximo de pistas. El usuario localiza el botón "grabar" y lo pulsa.	No se comienza a realizar la grabación y se muestra un mensaje de advertencia al usuario.

Caso de Uso CUP02 – Renombrar pista

Identificador	Prueba	Procedimiento	Resultado Esperado
PI08.1	Renombrar pista	El usuario ha activado el gestor de pistas previamente. El usuario localiza y pulsa el botón "renombrar" de una pista. Aparece un teclado sobre el cual el usuario escribe el nuevo nombre. Pulsa "Enter" en el teclado.	Se renombra la pista en la aplicación y en el sistema de archivos.
PI08.2	Renombrar pista (nombre en uso)	El usuario ha activado el gestor de pistas previamente. El usuario localiza y pulsa el botón "renombrar" de una pista. Aparece un teclado sobre el cual el usuario escribe el nuevo nombre (idéntico a uno ya existente en el sistema). Pulsa "Enter" en el teclado.	No se renombra la pista y se muestra un mensaje de advertencia al usuario.
PI08.3	Renombrar pista (número de	El usuario ha activado el gestor de pistas previamente.	No se renombra la pista y se muestra un mensaje de advertencia al usuario.

	caracteres menor que el mínimo)	El usuario localiza y pulsa el botón "renombrar" de una pista. Aparece un teclado y el usuario no escribe nada. Pulsa "Enter" en el teclado.	
PI08.4	Renombrar pista (número de caracteres mayor que el máximo)	El usuario ha activado el gestor de pistas previamente. El usuario localiza y pulsa el botón "renombrar" de una pista. Aparece un teclado sobre el cual el usuario escribe el nuevo nombre (superando el límite de caracteres establecido) Pulsa "Enter" en el teclado.	No se renombra la pista y se muestra un mensaje de advertencia al usuario.

Caso de Uso CUP03 – Eliminar pista

Identificador	Prueba	Procedimiento	Resultado Esperado
PI09	Eliminar pista	El usuario ha activado el gestor de pistas previamente. No debe haber ninguna pista en reproducción. El usuario localiza y pulsa el botón "eliminar" de una pista.	Se elimina la pista seleccionada.

Caso de Uso CUP04 – Reproducir pista

Identificador	Prueba	Procedimiento	Resultado Esperado
PI10	Reproducir pista	El usuario ha activado el gestor de pistas previamente. No debe haber ninguna pista en reproducción. El usuario localiza y pulsa el botón "reproducir" de una pista.	Se reproduce la pista seleccionada.

Caso de Uso CUP05 – Pausar pista

Identificador	Prueba	Procedimiento	Resultado Esperado
PI11	Pausar pista	El usuario ha activado el gestor de pistas previamente.	Se pausa la pista seleccionada.

		No debe haber ninguna pista en reproducción. El usuario localiza y pulsa el botón "pausar" de una pista.	

Caso de Uso CUP06 – Reiniciar pista			
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**
PI12	Reiniciar pista	El usuario ha activado el gestor de pistas previamente. No debe haber ninguna pista en reproducción. El usuario localiza y pulsa el botón "reiniciar" de una pista.	Se pausa y reinicia la pista seleccionada.

5.7.3 Pruebas de sistema

Las pruebas de sistema que se realicen comprobarán el correcto funcionamiento y comunicación de la aplicación Unity con los componentes que están fuera del desarrollo propio de la aplicación.

Estos componentes se localizan en el subsistema Oculus XR Plugin, que es aquel que permite a la aplicación Unity comunicarse con el dispositivo Oculus, enviando y recibiendo información sobre el tracking del entorno, barrera de juego, posicionamiento espacial y hand tracking.

Prueba tracking del entorno			
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**
PS01	Tracking del entorno correcto	El usuario debe colocarse las Oculus Quest y comprobar que el sistema de tracking funciona correctamente.	No se producen saltos ni temblores en el entorno virtual

Prueba barrera de juego			
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**
PS02	Comprobación de la barrera de juego	El usuario debe colocarse las Oculus Quest y caminar físicamente por el entorno.	Al acercarse al límite de juego, debe aparecer una barrera virtual que indique que el usuario está saliendo de la zona segura.

Prueba posicionamiento espacial			
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**

PS03	Prueba de posicionamiento espacial	El usuario debe colocarse las Oculus Quest y caminar físicamente por el entorno, girar la cabeza y moverse en el sitio.	El sistema debe representar con fidelidad los movimientos del usuario, sin latencia, y con una correspondencia real entre el movimiento físico y el virtual.

Prueba hand tracking			
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**
PS04	Prueba de sistema de hand tracking	El usuario debe colocarse las Oculus Quest y colocar las manos delante de su cara.	El sistema debe representar las manos del usuario en 3D delante de él. El sistema debe responder ante el movimiento y rotación de los dedos y manos.

5.7.4 Pruebas de código

El propio entorno Unity y su compilador integrado son capaces de identificar partes de código muerto, así como de lanzar diferentes advertencias (código obsoleto, variables sin usar).

Cada vez que se realiza un cambio en el código del proyecto, Unity recompila únicamente las clases modificadas en caso de que hubiera cambios. Esto resulta muy eficiente a la hora de desarrollar en este entorno ya que no es necesario volver a compilar toda la base de código cada vez que se quiere ejecutar o probar algo dentro del editor.

5.7.5 Pruebas de usabilidad

La naturaleza del proyecto, con un objetivo de cierto grado de estudio de este nuevo paradigma, indica que se deberían realizar diferentes pruebas en cuanto a la usabilidad de la aplicación.

Debido a que se introducen muchas mecánicas novedosas para la gran mayoría de los usuarios, es interesante conocer el grado de familiarización que tienen con entornos virtuales (2D, 3D, inmersivos, y realidad virtual, en ese orden) así como de distintas formas de interacción que hayan tenido con sistemas informáticos.

Una vez conocida esta información, será de interés evaluar con qué facilidad se desenvuelven los usuarios interactuando con el sistema: si la aplicación es sencilla de utilizar, si el sistema guía correctamente a nuevos usuarios, si da feedback coherente o si ocurre algún tipo de inconsistencia o de problema a la hora de utilizar el sistema que el usuario no se espere o no entienda.

Se confeccionarán cuestionarios y se realizarán sesiones de prueba a varios usuarios. Después de la sesión, responderán a una serie de preguntas de este tipo y se utilizarán los resultados de estas pruebas para redactar la conclusión del proyecto.

Capítulo 6. Diseño del Sistema

6.1 Arquitectura del Sistema

6.1.1 Diagrama de Paquetes

Debido a la naturaleza del proyecto, los paquetes del sistema estarán estrechamente basados en los subsistemas descritos en el apartado 5.3.1 (Descripción de los Subsistemas). Dentro de los paquetes aparecerán pequeñas distinciones entre los distintos elementos, aunque las funcionalidades de las clases dentro de cada paquete están relacionadas entre sí.

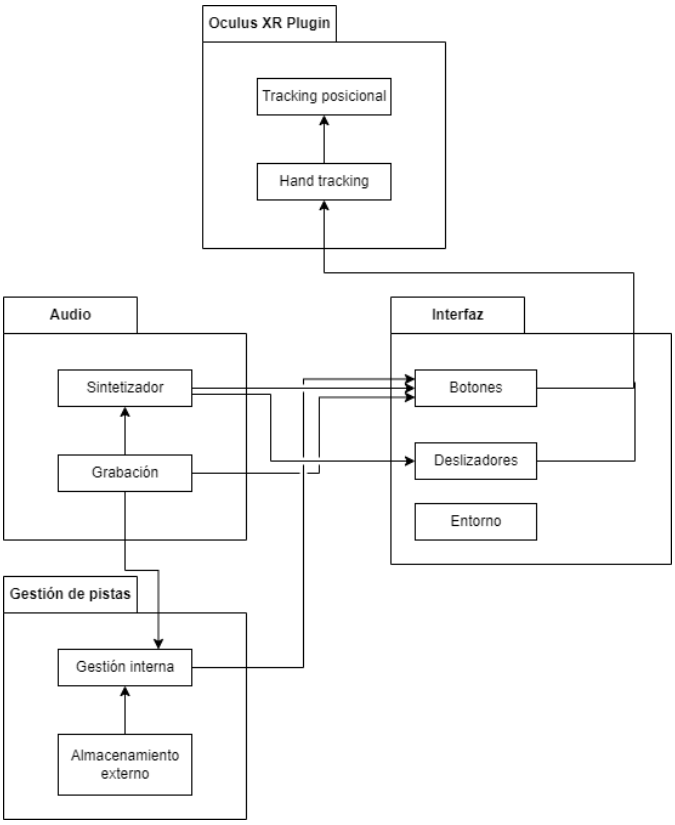


Figura 6.1 Diagrama de dependencia entre paquetes

En la Figura 6.1 se puede ver como aparecen pequeños "sub-paquetes" que diferencian distintos componentes dentro de cada paquete. Por ejemplo, en el paquete interfaz, se diferencian claramente los botones y los deslizadores, cada uno con una función distinta y específica para cada caso de uso.

Por otro lado, separamos el paquete de audio entre "Sintetizador" y "Grabación", cada uno con una función explícita; del mismo modo ocurre en la gestión de pistas, en la cual podemos diferenciar la gestión interna de las pistas (dentro de la aplicación) y la gestión del sistema de almacenamiento en el dispositivo.

Por último, es interesante diferenciar los distintos paquetes en el sistema Oculus, que permiten clarificar qué partes del plugin se van a utilizar. En este caso la interfaz está fuertemente relacionada con el hand tracking, que a su vez requiere del paquete de input y tracking posicional del dispositivo.

6.1.2 Diagrama de Componentes

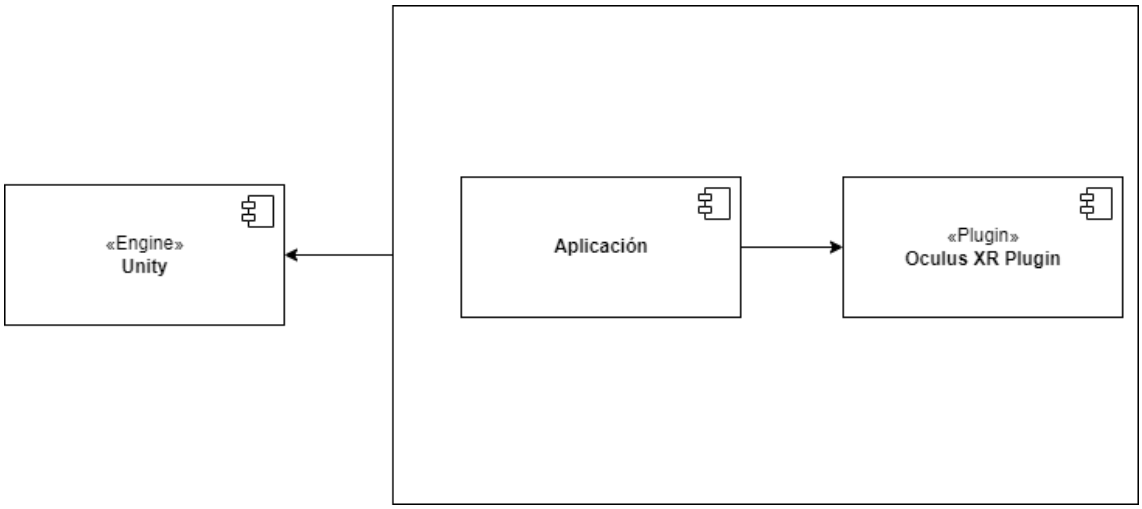


Figura 6.2 Diagrama de dependencia entre componentes

Como podemos observar en la Figura 6.2, los componentes del proyecto son claramente diferenciables. Por un lado, tenemos la aplicación (incluye todo el proyecto desarrollado) que se relaciona con el componente Oculus XR Plugin a través de una API.

Por otro lado, todo el proyecto es dependiente del motor Unity, ya que es el encargado de ejecutar la totalidad del código.

6.2 Diseño de Clases

6.2.1 Diagramas de Clases

Se separan los diagramas de clases por paquetes de código desde la Figura 6.3 hasta la Figura 6.5.

6.2.1.1 Entorno

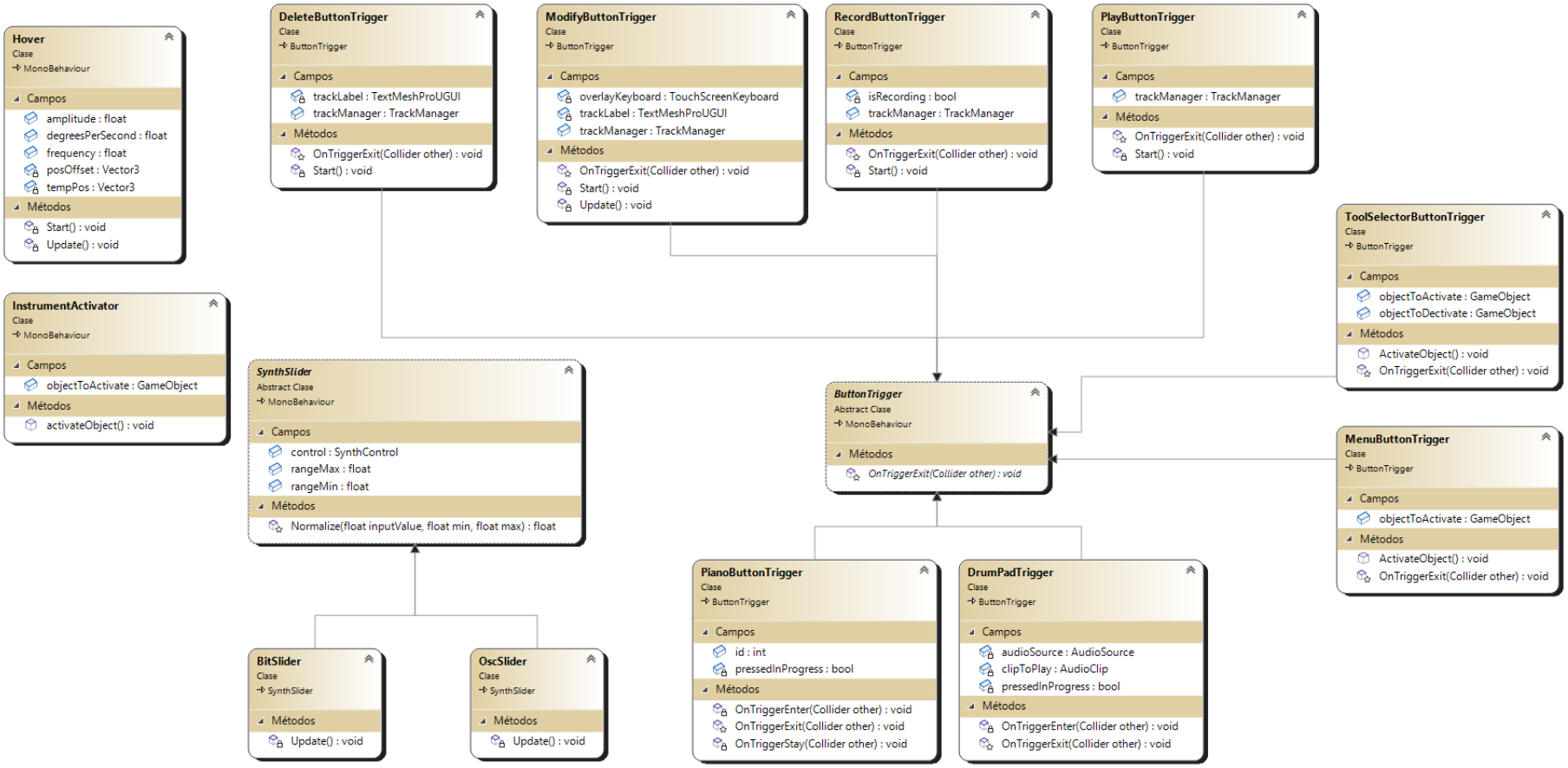


Figura 6.3 Diagrama de clases del paquete Entorno

6.2.1.2 Audio

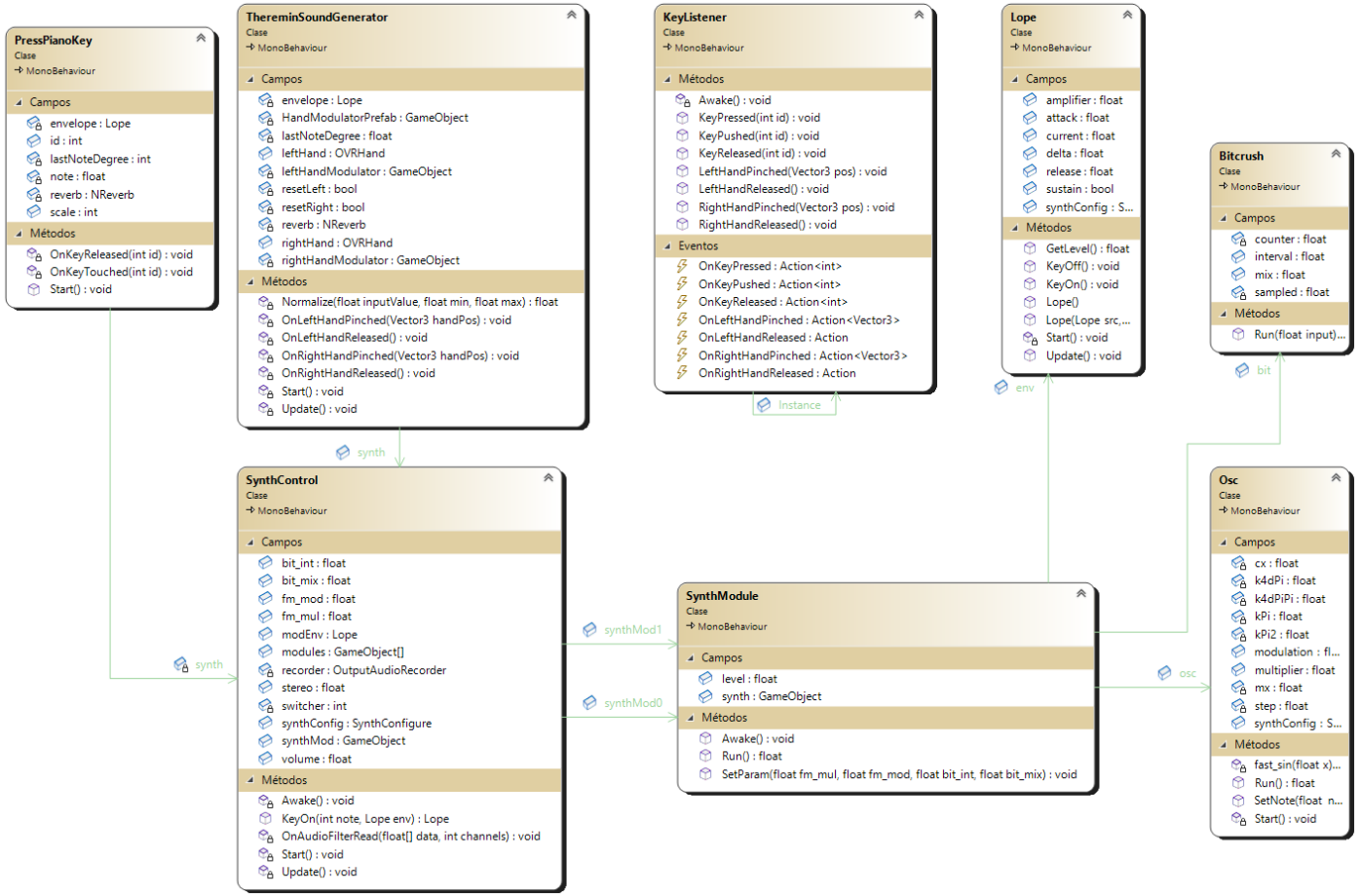


Figura 6.4 Diagrama de clases del paquete Audio

6.2.1.3 Gestor de pistas

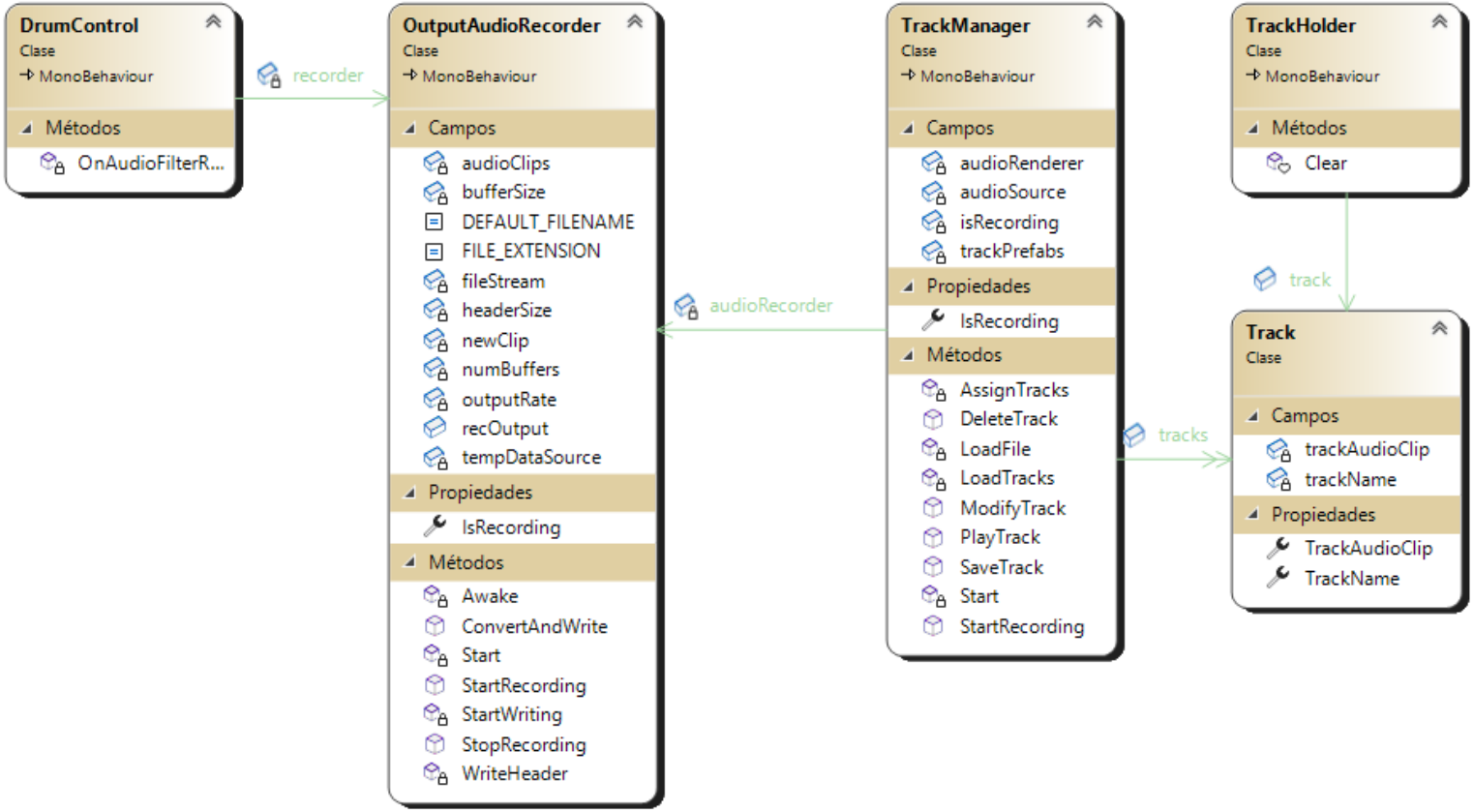


Figura 6.5 Diagrama de clases del paquete Gestor de pistas

6.3 Diseño de la Interfaz

El diseño de la interfaz tiene un papel principal en el proyecto. Como se explicó en apartados anteriores, el grueso de la aplicación consiste en desarrollar un entorno 3D inmersivo, diseñado específicamente para utilizar las manos como método de entrada principal.

6.3.1 Entorno

El entorno ha sido diseñado directamente desde el motor Unity, utilizando paquetes de assets gratuitos (modelos 3D) de terceros y también componentes nativos del motor. Se ha diseñado con una perspectiva de realidad virtual en mente, con lo que se han tenido en cuenta los siguientes aspectos en especial:

- Entorno no muy luminoso: al colocar una pantalla a escasos centímetros de los ojos del usuario, resulta mucho más cómodo para la vista del jugador tener un entorno oscuro con detalles coloridos (objetos interactuables) que destaquen en contraposición al resto del entorno.
- Distancia usuario-interfaz: se ha de tener en cuenta que en esta aplicación no se usarán mandos ni ningún otro tipo de herramienta que pueda facilitar al usuario interactuar a distancia con objetos del entorno. Por lo tanto, todos los elementos de la interfaz se han colocado a una distancia cómoda para su interacción (alrededor de 0,5 metros). No es una medida perfecta para todos los usuarios, pero cubre la gran mayoría de los tipos de usuario que utilizarán la aplicación.
- Sencillez de menús e interfaces: Dentro de la aplicación, se primará que haya la menor cantidad de menús posible, para facilitar al usuario el uso intuitivo de estos. Al estar las interfaces “ocupando” un espacio real dentro del entorno, no se puede dedicar una gran parte de este a menús o pantallas “2D” que entorpecerían la experiencia.

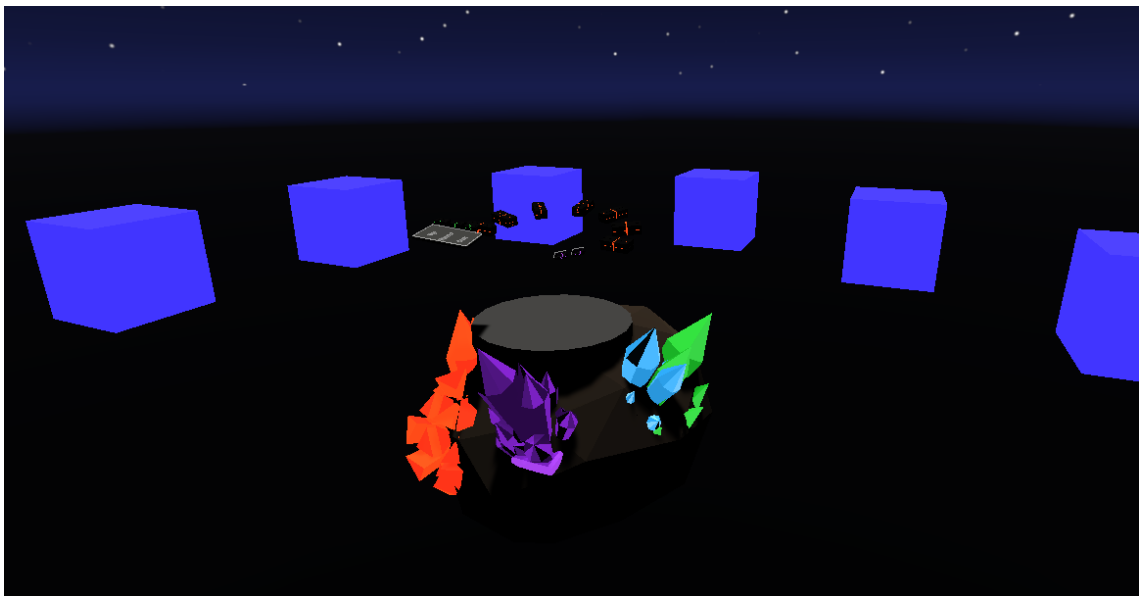


Figura 6.6 Entorno virtual de la aplicación

6.3.2 Pedestal

Toda la acción que ocurra dentro de la aplicación sucederá alrededor de lo que se ha denominado "pedestal". El pedestal se representa como una plataforma que "flota" en el espacio, como si fuera un meteorito, sobre la que el usuario está de pie interactuando con las interfaces.

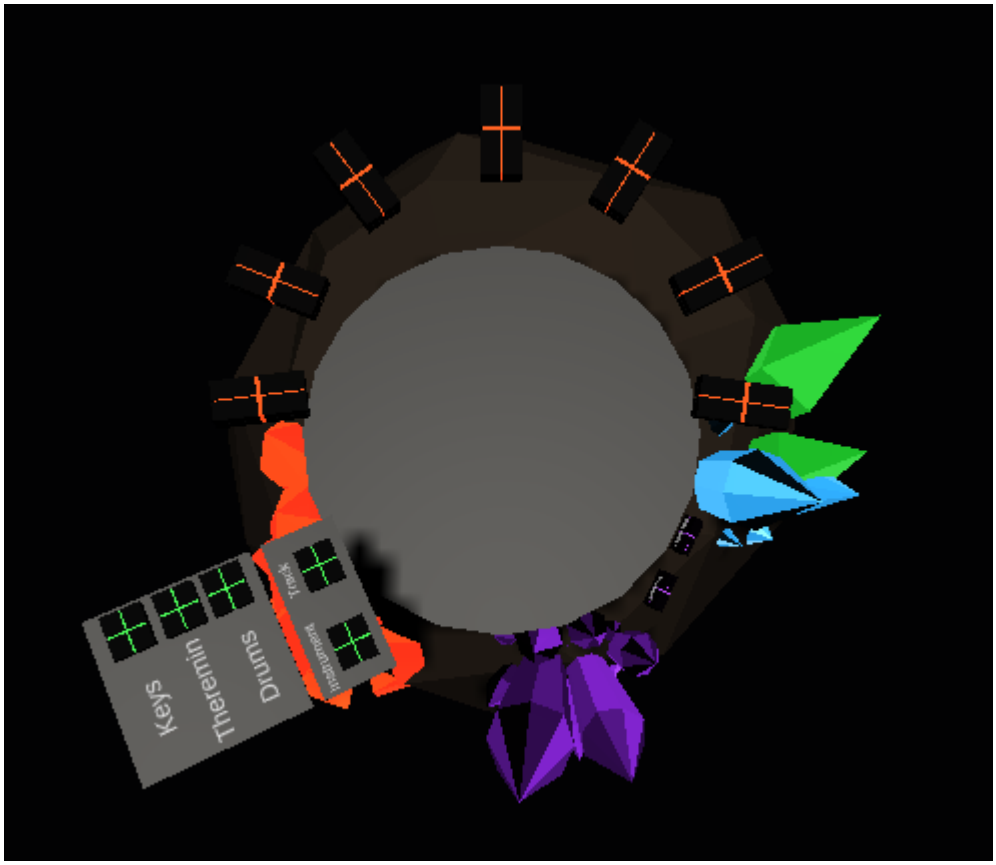


Figura 6.7 Vista cenital del pedestal

6.3.3 Selección de instrumentos y gestión de pistas

Al lado izquierdo del usuario, según su posición y orientación inicial, se presenta un panel que contiene los nombres de los diferentes instrumentos a elegir. Debajo de este panel existen dos botones que permiten alternar entre este panel y el de gestión de pistas.

Si el usuario pulsa cada uno de los botones respectivamente, el panel activado cambiará por el seleccionado por el usuario, por lo que no es posible activar ambos a la vez.

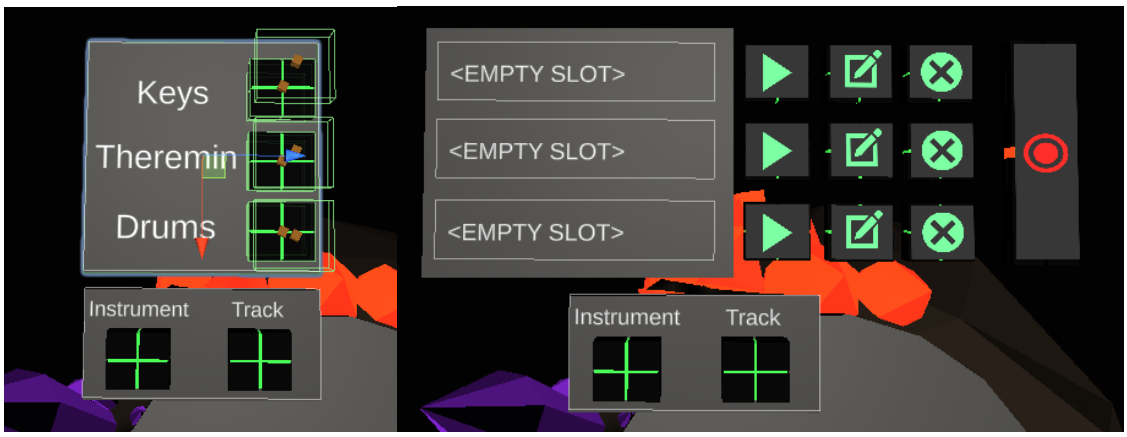


Figura 6.8 Vista cenital de la interfaz principal (modo "Instrumentos" y modo "Pistas")

En el **modo instrumentos**, aparecen los tres instrumentos disponibles en una lista vertical. Pulsar cualquiera de los botones adyacentes a los nombres de los instrumentos desactivará el instrumento actual y activará el seleccionado.

En el **modo pistas**, aparecen las tres casillas (slots) de pista disponibles. Al lado de cada pista existen tres botones:

- Play/pause: permite reproducir y parar la pista seleccionada.
- Modificar: permite renombrar la pista seleccionada.
- Eliminar: permite eliminar la pista seleccionada.

Además, en la parte derecha aparece el botón de grabación. Si existe un hueco libre en la lista de pistas, el botón permite grabar una pista nueva.

Cuando existe algún tipo de error (al guardar un archivo o renombrarlo) aparece un mensaje de este tipo explicando el error al usuario durante unos segundos. El texto explicativo variará en función del fallo que haya ocurrido.

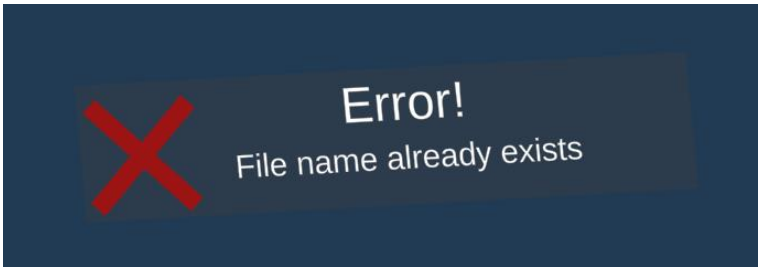


Figura 6.9 Mensaje de advertencia de ejemplo

6.3.4 Instrumentos

Al ser seleccionados, los instrumentos (visibles) aparecen alrededor del pedestal. Se han hecho los botones de los diferentes instrumentos de distintos colores para que sean diferenciables con el entorno. En el caso del piano, los botones son rojos, y en el de la batería, azules.

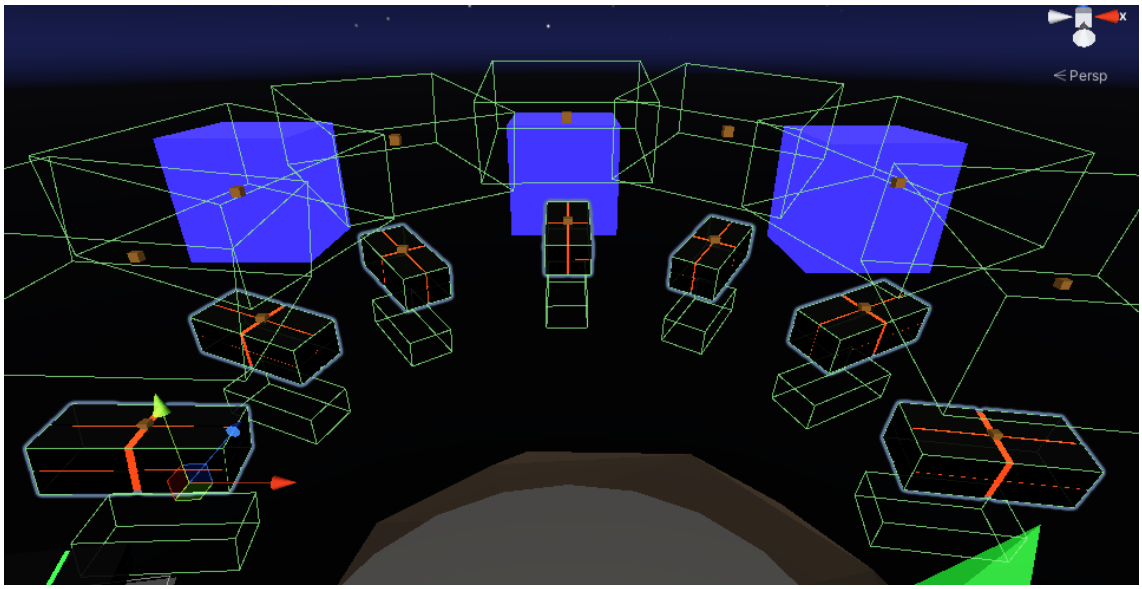


Figura 6.10 Instrumento Piano

Las teclas del piano aparecen rodeando al usuario en semicírculo alrededor del pedestal.

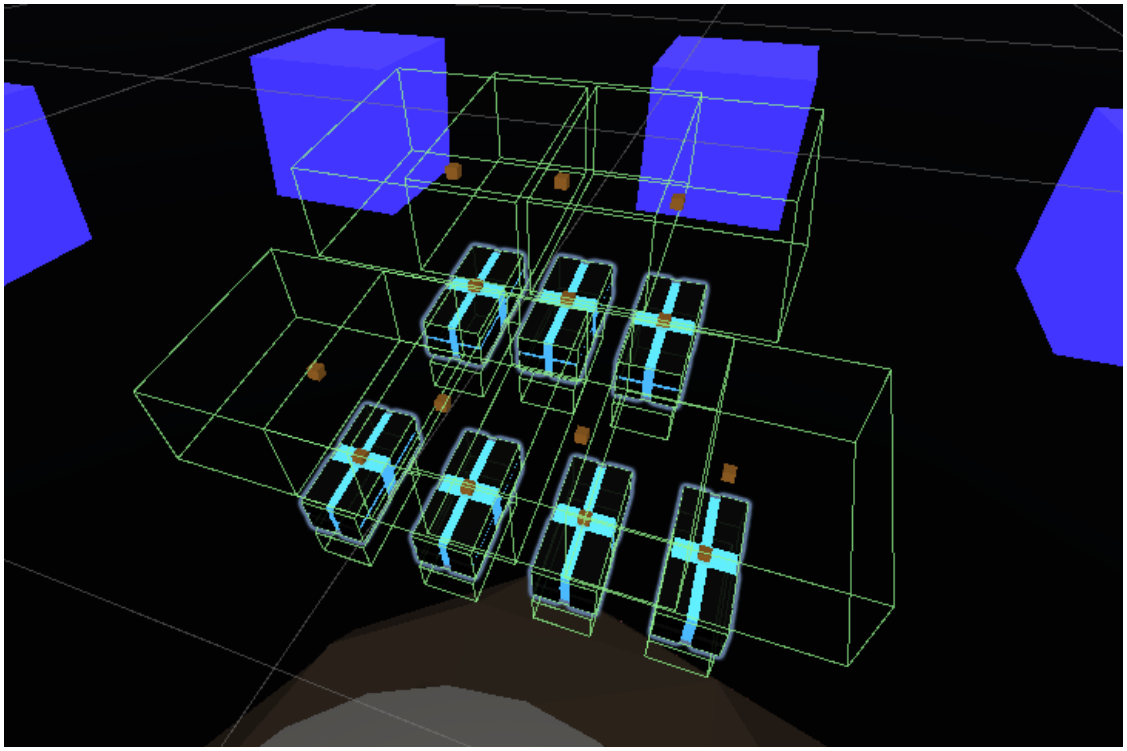


Figura 6.11 Instrumento batería

Los pads de batería aparecen en un espacio más reducido, en la parte frontal del pedestal.

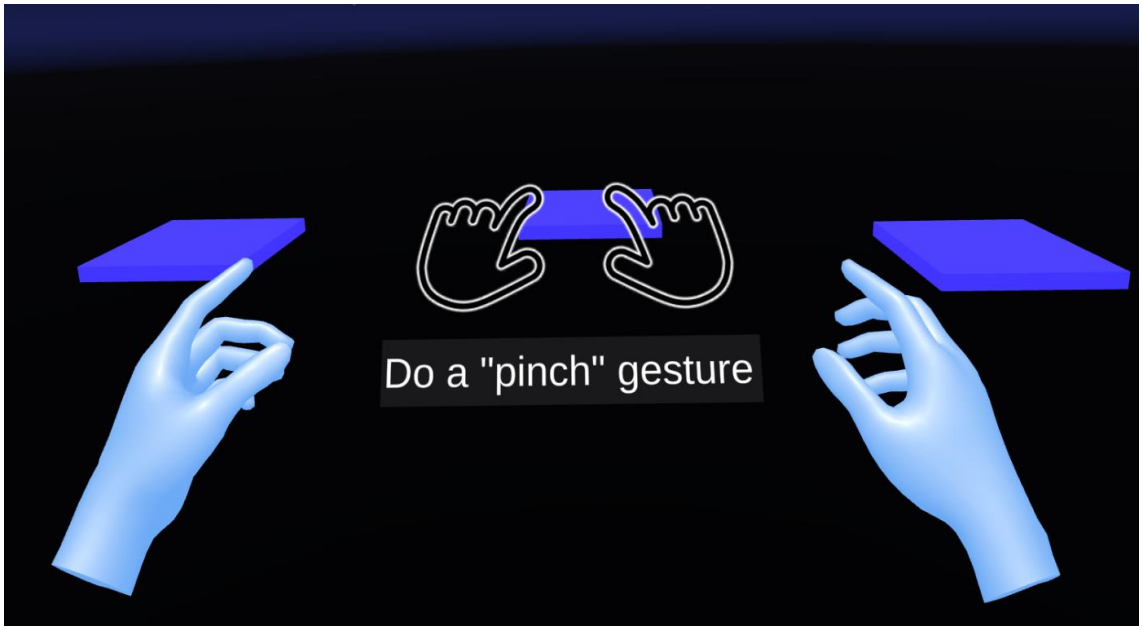


Figura 6.12 Instrumento Theremín

Cuando se selecciona el theremin, aparece un indicador de cómo realizar el gesto “pinch” (juntando el dedo índice y el pulgar) con ambas manos.

6.3.5 Diseño de botones y deslizadores

El diseño de los botones (Figura 6.13) y deslizadores (Figura 6.14) ha sido realizado para ser lo más fiel a la realidad posible y siguiendo los bocetos realizados en la fase de análisis.

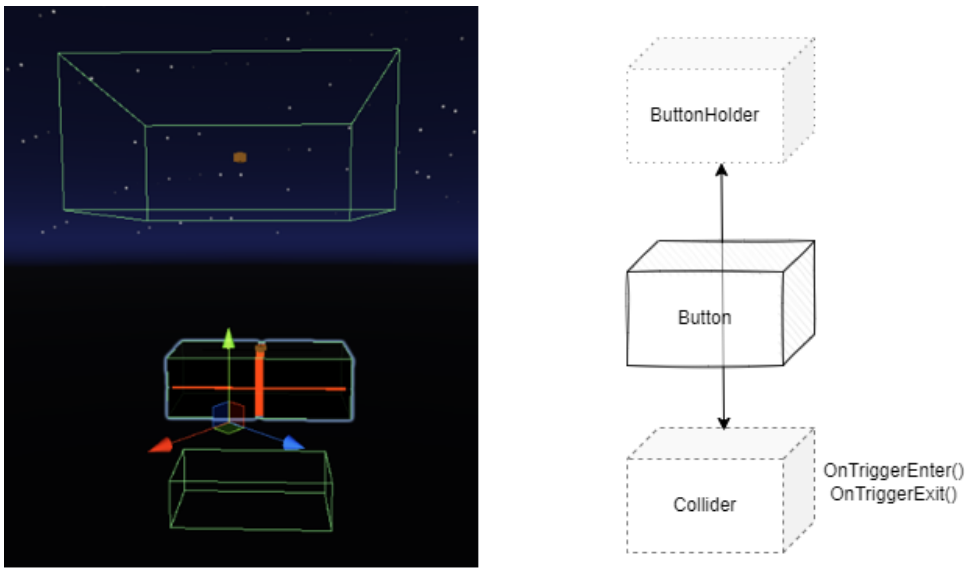


Figura 6.13 Funcionamiento de un botón

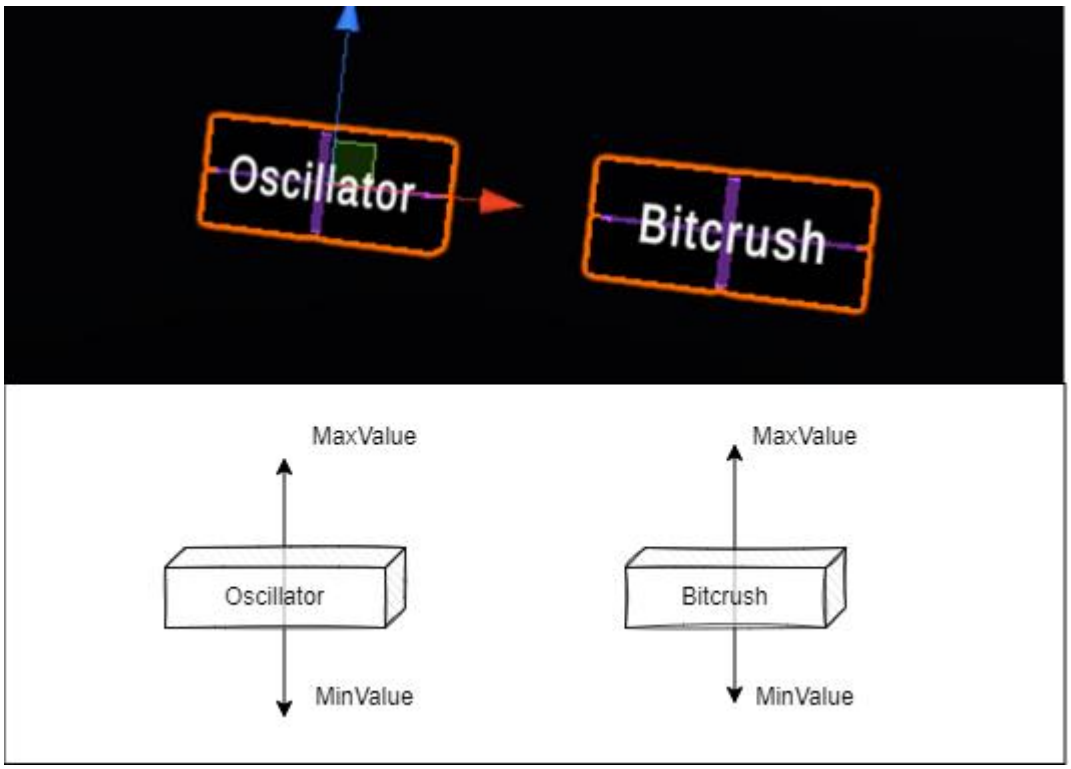


Figura 6.14 Funcionamiento de los deslizadores

6.4 Especificación Técnica del Plan de Pruebas

De acuerdo con la especificación del plan de pruebas detallada en el apartado 5.7, se realizarán pruebas de integración, sistema y usabilidad, restando así las pruebas unitarias que por alcance del proyecto se decidió no desarrollar.

6.4.1 Pruebas de Integración y Sistema

Como se indicó en el apartado 5.7.2 y en el apartado 5.7.3 las pruebas de integración y sistema se realizarán directamente sobre la aplicación para comprobar el correcto funcionamiento de los casos de uso establecidos.

Debido a la naturaleza del proyecto, estas pruebas se realizarán manualmente: el desarrollador conectará las gafas VR al ordenador y desde el propio editor de Unity podrá obtener los inputs necesarios por parte del dispositivo y probar las funcionalidades deseadas.

También, y siempre que se amplíe de manera significativa alguna parte de la aplicación, se creará una .apk (Paquete de Aplicación de Android) para su despliegue directo en el sistema Android y comprobar que no existen cuellos de botella ni incompatibilidades entre lo probado en el editor y lo que será la aplicación final.

Se tomará nota de todos los resultados inesperados o no consistentes para implementar correcciones en fases futuras de implementación.

En el apartado 8 (Desarrollo de las pruebas) se listarán las pruebas de los apartados 5.7.2 y 5.7.3 junto con los resultados obtenidos y las anotaciones oportunas en caso de que a partir de las pruebas se hubiera obtenido información relevante.

6.4.2 Pruebas de Usabilidad

Como se planificó anteriormente, durante la fase de pruebas de la aplicación se llevará a cabo una sesión de pruebas de usabilidad con cada uno de los usuarios “Tester” de la aplicación.

Las pruebas se basarán en darles una ligera explicación de cómo funciona la aplicación, con aclaraciones sobre cómo interactuar con las interfaces para darles una idea principal de lo que se van a encontrar.

Después de que el usuario pruebe la aplicación, rellenará dos cuestionarios: el primero, relacionado con su experiencia en general con los sistemas informáticos y el segundo será sobre su experiencia con la aplicación del proyecto. De este modo, aunque la muestra de usuarios sea pequeña, podrá ser interesante ver si existe algún tipo de correlación entre experiencia en general con aplicaciones o videojuegos y experiencias inmersivas.

Por otra parte, las pruebas de usabilidad pueden dar pistas de cómo mejorar la aplicación en estos términos: adaptando las interfaces, ofreciendo más ayuda, etc.

6.4.2.1 Introducción a la aplicación

Para introducir a los usuarios a la aplicación desarrollada, se hará una breve explicación de cómo funciona y qué esperar una vez dentro de ella. Como "guion" principal a la hora de hacer la introducción, se deberán explicar de manera sencilla los siguientes puntos:

- Qué es la realidad virtual, en qué se basa y qué permite el dispositivo.
- El objetivo de la prueba: cómo de intuitivo y fácil de usar es el diseño del sistema.
- Qué va a ver el usuario cuando se ponga las gafas de realidad virtual.
- Cómo interactuar con la interfaz y los instrumentos.

Una vez hecha esta pequeña introducción (que no debería durar más de 5 minutos) se le colocan las gafas al usuario y se le permite interactuar libremente con el sistema (proporcionando asistencia si fuera necesario). Se intentará que el usuario pruebe todas las funcionalidades del sistema, pero si no es capaz de hacerlo por sí mismo, también será algo para tener en cuenta.

6.4.2.2 Cuestionarios

Una vez terminada la prueba (que debería durar alrededor de unos 15 minutos), se pide al usuario que rellene los siguientes cuestionarios:

6.4.2.2.1 Experiencia con sistemas informáticos

Del 1 al 5, indique su respuesta:

Uso el ordenador o smartphone …				
1 (Nunca)	2 (Muy ocasionalmente)	3 (De vez en cuando)	4 (Varias veces por semana)	5 (Todos los días)
Tengo experiencia con …				
1 (Ningún programa. Siempre necesito ayuda)	2 (Navegador y correo)	3 (Algún programa de ofimática)	4 (Instalación de programas, aplicaciones de trabajo)	5 (Cualquier tipo de software, soy programador)
He jugado a videojuegos…				
1 (Nunca)	2 (Muy ocasionalmente)	3 (De vez en cuando)	4 (Varias veces por semana)	5 (Todos los días)
He utilizado algún controlador distinto a un ratón y teclado (se pueden marcar varias opciones)				
1 (Ninguno)	2 (Mando de consola)	3 (Smartphone)	4 (Joystick/Volante)	5 (Realidad Virtual)

6.4.2.2.2 Experiencia con la aplicación

Del 1 al 5, indique su respuesta:

Valoración general				
1 (No me ha gustado)	2 (Tiene bastante que mejorar)	3 (Regular)	4 (Me ha gustado)	5 (Me ha encantado)
Usaría este tipo de aplicación como entretenimiento				
1 (No)	2 (Lo veo poco útil)	3 (Está entretenido, pero no lo usaría)	4 (Sí, es interesante, lo usaría)	5 (Sí, me ha encantado y lo usaría a menudo)
Tengo experiencia musical …				
1 (No me gusta la música)	2 (Me gusta, pero no sé tocar instrumentos)	3 (Escucho y toco algún instrumento)	4 (Toco varios instrumentos)	5 (Soy músico profesional / productor)
La interfaz me ha parecido…				
1 (Nada intuitiva)	2 (Muy poco intuitiva)	3 (Mejorable)	4 (Buena)	5 (Muy fácil de usar)
He podido experimentar con… (se pueden marcar varias opciones)				
1 (Instrumentos - Piano)	2 (Instrumentos - Batería)	3 (Instrumentos - Theremín)	4 (Grabación de pistas)	5 (Gestión de pistas)
Me han gustado las opciones que ofrece				
1 (Muy pobre)	2 (Añadiría más)	3 (Está bien)	4 (Podría tener alguna pequeña mejora)	5 (No añadiría nada)
Esta aplicación ha cambiado mi opinión sobre la realidad virtual				
1 (No)	2 (Apenas)	3 (Me quedo como estaba)	4 (Ha cambiado ligeramente)	5 (Ha cambiado radicalmente)
Sobre la pregunta anterior: ¿es su opinión actual favorable o desfavorable?				
Observaciones, comentarios o sugerencias:				

6.4.2.2.3 Cuestionario para el responsable de las pruebas

Aspecto Observado	Notas
¿Se desorienta el usuario en una primera instancia al colocarse las gafas?	
¿Le cuesta al usuario aprender a usar la interfaz?	
¿Entiende el usuario el sistema de reconocimiento de gestos/hand tracking?	
¿Puede el usuario volver a centrarse si se pierde dentro de la escena?	
¿Qué comentarios hace el usuario en voz alta al probar la aplicación?	
¿Necesita ayuda el usuario en algún momento?	
Errores leves cometidos	
Errores graves cometidos	

6.4.3 Pruebas de Rendimiento

Durante el desarrollo de videojuegos y aplicaciones 3D, y especialmente en aquellas que utilizan realidad virtual, es importante mantener un framerate (número de cuadros por segundo) lo más elevado posible para dar una sensación de fluidez durante su uso.

De hecho, cuando se desarrolla para RV, se suele establecer un mínimo de 72 Hz o frames por segundo (FPS) como estándar para evitar mareo al usar el dispositivo, ya que si es menor puede existir cierta descompensación entre lo que el usuario está viendo y su movimiento en el mundo real. Esto es lo que se conoce en inglés como “motion sickness” (cinetosis) y afecta sobre todo a aquellos usuarios que están empezando a utilizar sistemas de realidad virtual **[Chattha20]**.

Existen multitud herramientas para monitorizar sistemas de realidad virtual en tiempo real: al fin y al cabo, su sistema operativo está basado en Android y como tal, permite el acceso de este tipo de aplicaciones de depuración.

6.4.3.1 Unity Profiler

El motor Unity proporciona una herramienta de depuración que, de forma nativa, es capaz de acceder al proceso en ejecución en las gafas y desglosar en tiempo real la carga de CPU, GPU o de memoria que se está consumiendo por parte de la aplicación **[OC01].** Si bien es una herramienta muy útil y fácil de usar, es necesario considerar varios puntos:

- El propio “profiling” puede añadir cierta carga extra de procesamiento al sistema: se recomienda tenerlo en cuenta a la hora de hacer pruebas cuando el sistema es aún pequeño, y hacer mediciones de forma relativa en lugar de tener en cuenta los valores

absolutos (por ejemplo, ver si la carga de CPU aumenta un % elevado) y hacer distintas mediciones para comparar resultados.

- El propio sistema Oculus puede añadir ruido a las mediciones debido a los procesos que están funcionando en el segundo plano. Se recomienda, igualmente, hacer distintas mediciones y comparar, como en el caso anterior.
- El sistema Oculus soporta distintas velocidades de procesador para un mejor rendimiento de la batería. Por tanto, el rendimiento puede mejorar o empeorar dependiendo de la carga y se recomienda tenerlo en cuenta a la hora de hacer este tipo de pruebas. También existe la posibilidad de desactivar esta funcionalidad.

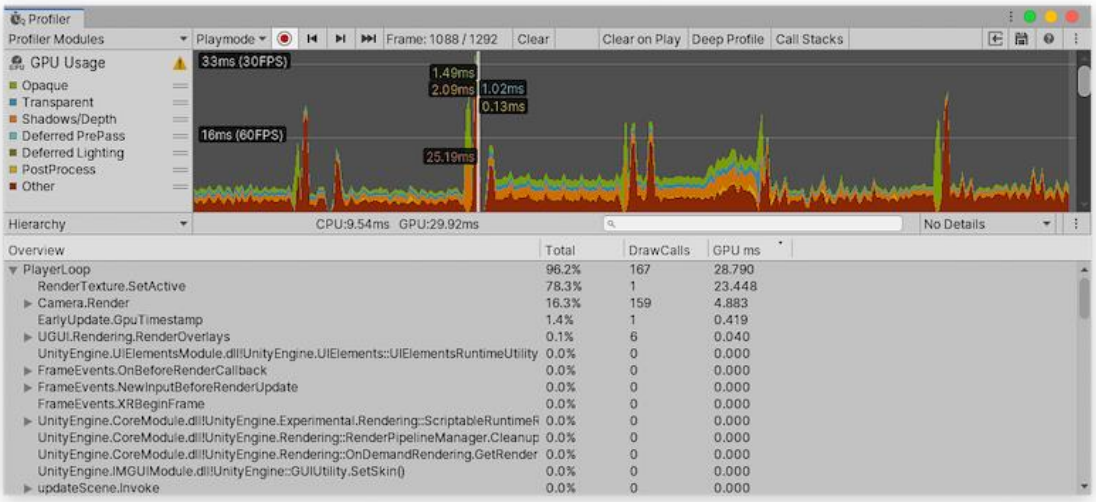


Figura 6.15 Ventana del Unity Profiler

6.4.3.2 OVR Metrics Tool

OVR Metrics Tool es una herramienta proporcionada por Oculus para proveer al desarrollador de métricas de rendimiento en tiempo real. Al ser una aplicación desarrollada por Oculus permite visualizar estas métricas dentro del dispositivo a modo de overlay en cualquier aplicación **[OC02].**

También dispone de una funcionalidad que crea un informe tras una sesión “grabada” y puede ser fácilmente exportado como un archivo CSV.

Proporciona métricas interesantes que el Unity Profiler por su naturaleza no puede mostrar, ya que esta herramienta tiene acceso a interfaces propias del sistema, como puede ser la batería del dispositivo, temperatura, u otros procesos que se estén ejecutando en segundo plano.

Figura 6.16 OVR Metrics Tool en modo Overlay

Capítulo 7. Implementación del Sistema

7.1 Lenguajes de Programación

Se utiliza el lenguaje C# para el desarrollo de los scripts. C# es un lenguaje de programación orientado a objetos moderno, utilizado en la mayoría de las aplicaciones Unity por su integración con el entorno de desarrollo.

Unity utiliza la versión 7.3 de C# (.NET 4.6).

7.2 Estándares y Normas Seguidos

Se siguen las convenciones de código de C# **[MS01].** El propio software (Visual Studio) es capaz de detectar si se están siguiendo estas convenciones. Algunos ejemplos:

- Notación de mayúsculas Pascal (PascalCasing) cuando se nombra una clase o struct.
- Notación Camel (camelCase) cuando se nombran campos *internal* o *private*, y se añade el prefijo "_".
- Indentado de cuatro caracteres, tabulaciones guardadas como espacios.

7.3 Herramientas y Programas Usados para el Desarrollo

7.3.1 Unity

El motor Unity fue utilizado para el desarrollo del videojuego en su versión 2019.4.18f1. Dentro de los tipos de licencia disponibles, se utilizó la versión personal (gratuita).

7.3.2 Visual Studio

Se utilizaron las versiones 2019 y 2022 del editor Visual Studio (Community) para el desarrollo. Este IDE proporciona herramientas de completado de código, estilo de código y depuración.

7.3.3 Doxygen

Para crear la documentación asociada al detalle de clases se ha utilizado este software en su versión 1.9.4.

7.4 Creación del Sistema

7.4.1 Problemas Encontrados

7.4.1.1 Desarrollo del motor de audio

Si bien no era un objetivo del proyecto en sí, sí que se contemplaba el desarrollo de un motor de audio sencillo, con las funcionalidades mínimas para poder ser utilizado como parte del proyecto.

Durante las fases de estudio y análisis se identificaron posibles dificultades a la hora de programar un paquete de scripts que modelaran un sintetizador completamente virtual, ya que las opciones y funcionalidades a la hora del desarrollo son muy elevadas (podría ser un proyecto aparte en sí mismo). Se comprendió que el desarrollo de un motor de audio desde cero era efectivamente una tarea mucho más costosa en tiempo y por esta razón se buscaron alternativas. Finalmente, se escogió una base de código ya existente con licencia MIT **[JZ01]** sobre la que adaptaría parte de su código para su uso en proyecto.

Este proyecto, desarrollado para una versión muy anterior de Unity (el primer commit data de hace 8 años) contenía las funcionalidades básicas de un sintetizador y otras más avanzadas. Se decidió por restricciones de tiempo y planificación utilizar este código como base y readaptarlo en su mayoría, tanto como para la nueva versión de Unity como para ser usado en una interfaz en realidad virtual. Además, muchas de las funciones avanzadas no se utilizan en este proyecto, pero podrían ser readaptadas también en posibles ampliaciones.

Se explica con más detalle cómo fue la adaptación de este código en la sección 7.4.2.6.

7.4.1.2 Consistencia de las pruebas en realidad virtual

Como se previó en las fases de análisis y desarrollo, el hecho de poder realizar pruebas unitarias para este proyecto era algo relativamente complicado debido a la propia naturaleza de la aplicación. Como la mayoría de las funcionalidades e interacciones dependen de la actuación directa del usuario a través del sistema de hand tracking con la interfaz, no hubiera resultado eficiente dedicar tiempo y recursos a desarrollar pruebas unitarias para probar estas interacciones.

Durante la fase de implementación del sistema se iba probando manualmente cada una de las funcionalidades, como podría ser el reconocimiento gestual para lanzar sonidos o el hecho de apretar un botón con la mano. No hubiera sido óptimo desarrollar una base de código paralela que simulara toda la interacción manual, cuando las pruebas unitarias representaban estas sencillas acciones.

Por lo tanto, se decidió desechar la idea de hacer test de tipo unitario para esta aplicación, ya que las pruebas se realizaban continuamente durante el desarrollo.

7.4.1.3 Escritura de archivos en una ruta diferente a la predeterminada

Debido a limitaciones del sistema Oculus, no es posible guardar archivos (en este caso, pistas de audio) fuera de la ruta predeterminada de la aplicación. Los archivos se deben guardar bajo la ruta *Application.persistentDataPath*, que en el caso de esta aplicación apunta a */Android/data/com.jatorrasagasti.synthvr/*

7.4.1.4 Problemas con el input de voz

Si bien utilizar el micrófono incorporado como fuente de sonido era una opción que se tuvo en cuenta a la hora de desarrollar la aplicación, durante la implementación del sistema surgieron diferentes problemas: la interfaz de acceso al micrófono de Android no funcionaba como esperado mientras se probaba la aplicación conectada a un PC. Además, la latencia a la hora de hablar/reproducir dentro de la aplicación hacía que el uso fuera molesto para el usuario y no permitía articular frases correctamente. Por estas y otras razones de implementación menores, se decidió descartar esta funcionalidad del sistema.

7.4.2 Implementación del sistema en Unity

7.4.2.1 Estructura del entorno

La implementación del sistema comenzó con los trabajos relativos a la creación del entorno en el cual el usuario iba a utilizar el sistema. Siguiendo los bocetos iniciales y el diseño establecido, se creó un objeto Escena en Unity (llamado SynthVRScene) que sería aquel que comprendiera todas las características del sistema. No se planteó crear nuevas escenas para distintas funcionalidades, ya que por la naturaleza del sistema no se consideró necesario.

Para representar los distintos elementos del entorno se utilizaron una serie de “assets” gratuitos y elementos del motor Unity que facilitaron no tener que diseñar iconos o modelos 3D para el sistema. Dentro de la jerarquía de objetos de la escena, se sitúan en la sección “Environment”.

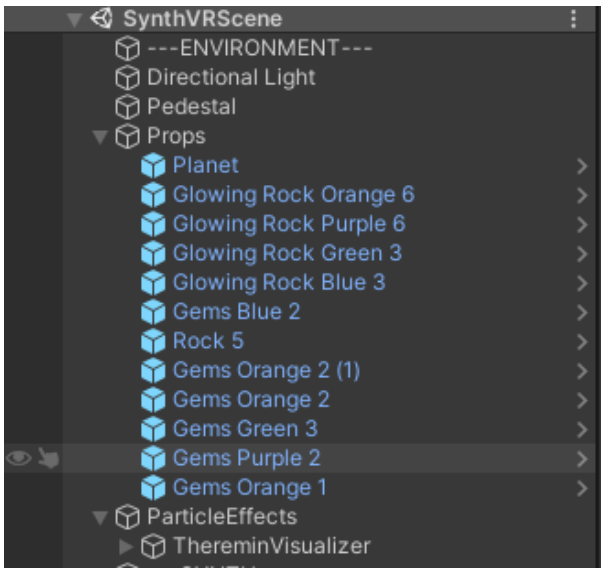


Figura 7.1 Jerarquía de objetos de la escena SynthVR

Algunos de estos elementos son el pedestal (zona central de la escena), el objeto "Directional Light" que produce iluminación en la escena, o los distintas rocas o cristales decorativos de la escena. Alrededor del pedestal, algunas de estas rocas flotan y rotan en el aire gracias al script "Hover", creando un efecto de estar en el espacio.

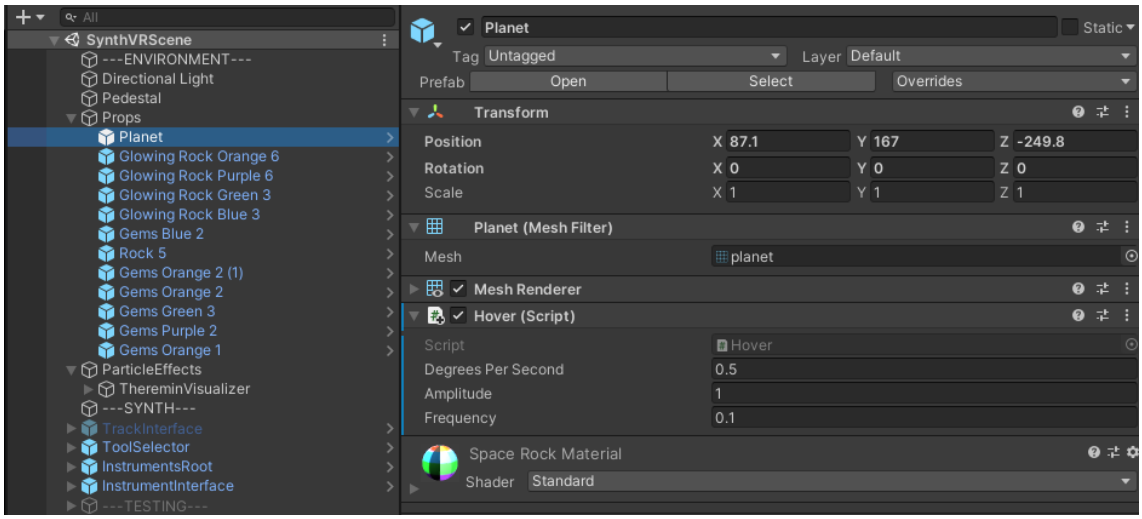


Figura 7.2 Configuración de script Hover

Para los efectos de iluminación y el "Skybox" de la aplicación, se utiliza un elemento de Unity denominado "Directional Light" (luz direccional) que proyecta un único punto de luz con un color, intensidad y sombras en una dirección. Este tipo de iluminación suele utilizarse en escenas exteriores con iluminación estática, donde no es necesario generar grandes cantidades de sombras dinámicas en tiempo real.

El skybox (cielo) de la aplicación tiene la intención de simular un cielo estrellado, o una escena del espacio exterior. Por lo tanto, se utilizó un shader (script para controlar la renderización de ciertos objetos) que permitiera representar este tipo de escena **[Yoda19].** La manera de utilizar

este shader como skybox es aplicándolo a un Material, que será después colocado como "Skybox Material" en las opciones de iluminación del proyecto.

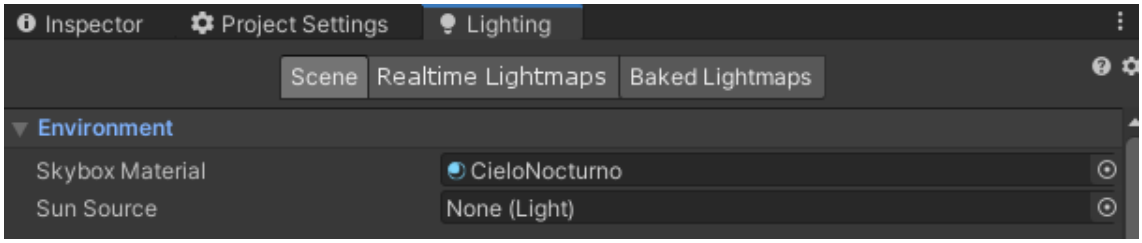


Figura 7.3 Configuración del skybox

Dentro de los elementos encontrados en esta sección, existen también los que se utilizan para visualizar las frecuencias generadas por el theremín a través del sintetizador. Estos objetos se encuentran en la jerarquía bajo el objeto "ParticleEffects". Cada uno de estos objetos "parametric-cube" tiene asignado un script que lee las frecuencias del espectro audible generadas por el sintetizador y escoge las que están dentro de la banda asignada en el script. Dependiendo del volumen de la banda, los cubos crecerán o encogerán su tamaño.

7.4.2.2 Cámara y eventos

Si bien en todo proyecto Unity es indispensable utilizar un objeto de tipo Cámara, ya que es el componente del motor que se encarga presentar al usuario todo el sistema gráfico de la aplicación.

En un proyecto de escritorio (por ejemplo, un videojuego) el usuario tiene cierto sobre la cámara (puede hacer zoom, desplazarse, rotar alrededor de un personaje o mirar alrededor), con unos límites establecidos. Sin embargo, cuando se desarrolla para la realidad virtual hay que tener en cuenta ciertos factores: la cámara estará directamente acoplada al dispositivo, por lo que todo movimiento o rotación de esta dependerá del movimiento real del usuario.

Para una integración de este tipo, los fabricantes de dispositivos proveen a los desarrolladores con plugins gratuitos que permiten facilitar la implementación de la mayoría de las funcionalidades. En este caso, como el dispositivo utilizado son unas Oculus Quest, se utiliza el Oculus XR Plugin. Este plugin ofrece distintos scripts y prefabs (objetos prefabricados) que se encargan de establecer una base con la cual el desarrollador puede comenzar a trabajar. En este caso, estos scripts se llaman OVR Camera Rig (controla la rotación y posición de la cámara) y OVR Manager. El OVR Manager es la interfaz principal con los componentes de hardware. Como se puede ver en la figura 7.4, permite elegir entre diversas opciones relativas al nivel de detalle de renderizado, la posición inicial del usuario en la escena, si se quiere utilizar hand tracking o no, entre otros.

Para este proyecto, se eligieron las siguientes opciones, dejando las demás en sus valores predeterminados:

- **Tracking Origin**: Eye Level (la altura de la cámara es la misma que la del usuarioen el mundo real)

- **Use Recommended MSAA Level** (Multi-Sampling AntiAliasing): para un mejor rendimiento
- **Focus Aware:** Requerido para utilizar el teclado del sistema
- **Requires System Keyboard:** Activa el teclado del sistema cuando es necesario
- **Hand Tracking Support:** Soporte para hand tracking activado

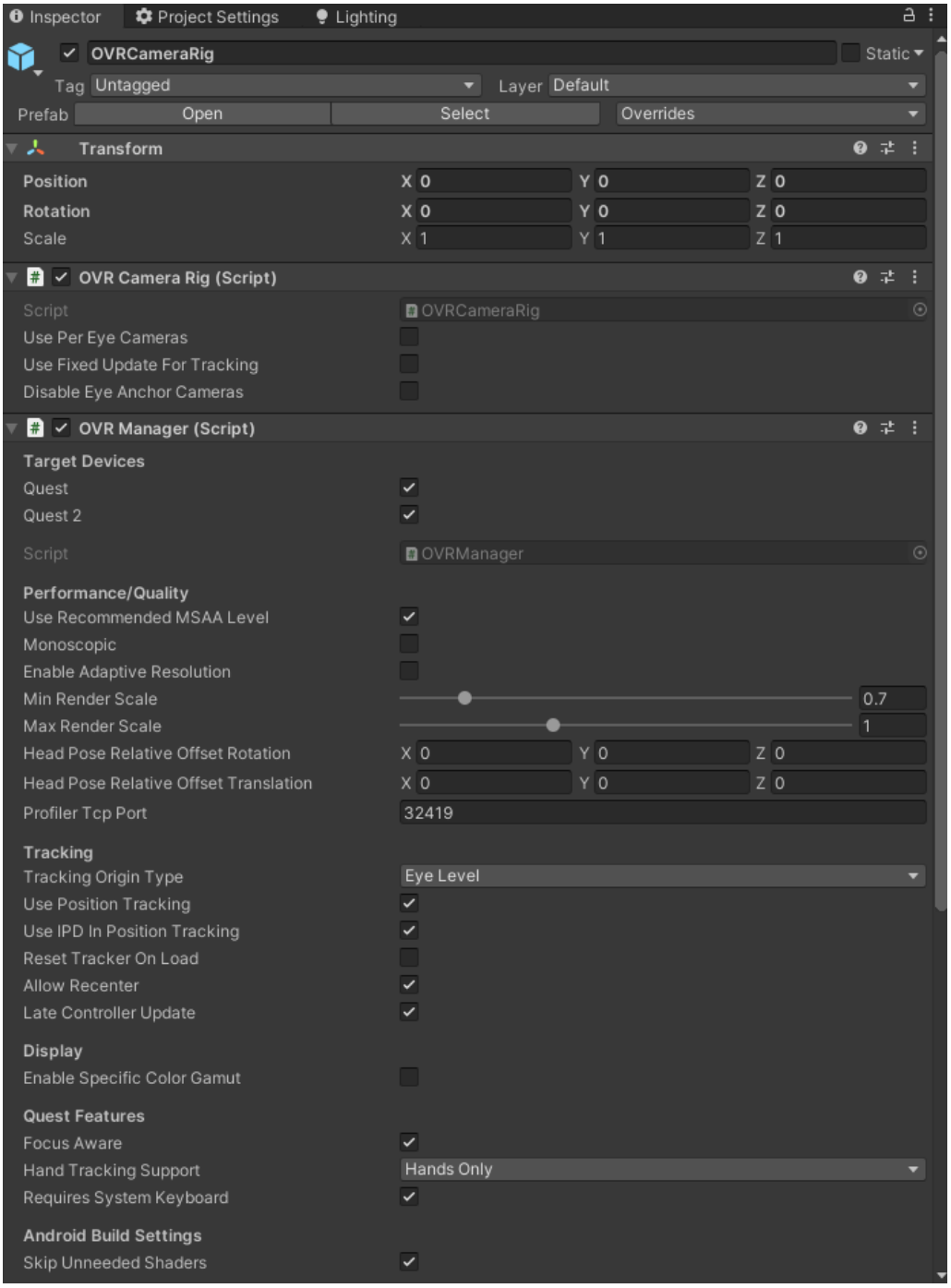


Figura 7.4 Configuración del OVR Camera Rig

Además, para utilizar los modelos 3D de manos incluidos en el paquete Oculus, es necesario añadir los prefabs correspondientes al sistema Camera Rig: un objeto OVRHandPrefab para cada una de las manos, escogiendo cuál sería la izquierda o la derecha.

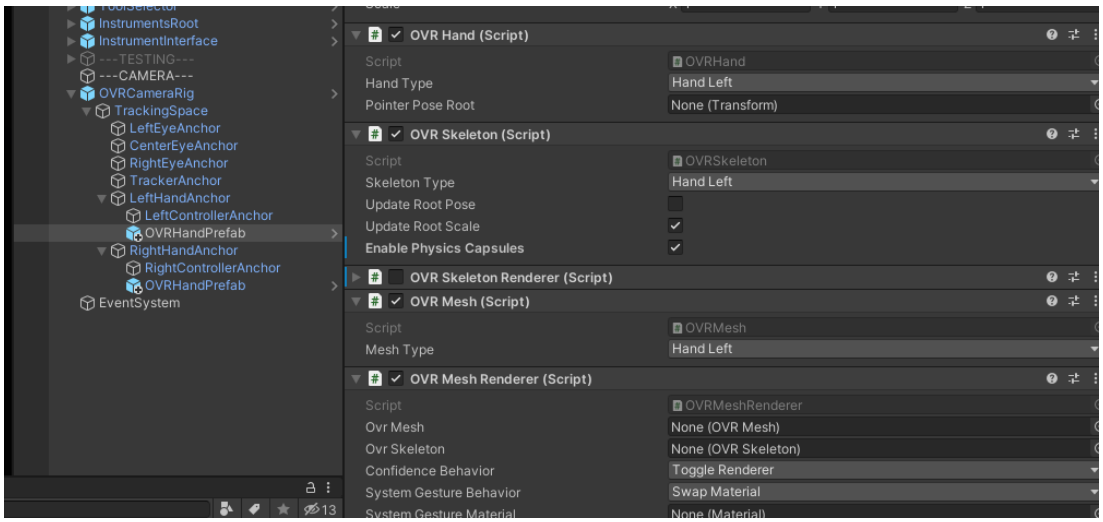


Figura 7.5 Configuración de OVR Hand Prefab

Por otra parte, en una escena Unity siempre es necesario añadir un objeto EventSystem si se quieren enviar eventos a objetos a la aplicación, por lo que se añade también dentro de esta sección de objetos.

7.4.2.3 Instrumentos

Los objetos de la escena relacionados con los instrumentos se dividen en dos objetos contenedores distintos:

- **InstrumentInterface:** Objeto que representa la interfaz sobre la cual se elige el instrumento que se va a activar en cada momento.
  Está compuesto por un panel y tres botones, cada uno de los cuales relacionado con un instrumento a través del script "Menu Button Trigger", que tiene el campo asociado "Object To Activate". Este campo tiene una referencia dentro de la escena a un instrumento.
- **InstrumentsRoot:** Objeto contenedor dentro del cual se encuentran los instrumentos. No tiene más función que agregar todos ellos en un mismo lugar para facilitar el desarrollo. Dentro de este objeto están:
  - **NotePicker:** Representa el instrumento piano. Está compuesto por siete botones que representan las teclas del piano, así como un objeto SynthController y los deslizadores de los parámetros especificados en los requisitos.
    Cada uno de los botones tiene asignado un script llamado "Press Piano Key" que, teniendo una referencia al SynthController, le envía una nota específica para que la reproduzca. Asimismo, cada uno de los deslizadores tiene un script específico (Osc Slider o Bit Slider) para modificar los parámetros correspondientes.
  - **ThereminPlayer:** Este objeto representa el instrumento theremín. Está compuesto por un SynthController y un objeto de interfaz llamado "ThereminTutorial" que muestra el gesto a realizar para utilizar el instrumento.

Posee un script "Theremin Sound Generator" que haciendo referencia al SynthController y dependiendo de la posición de las manos mientras se hace el gesto pinch, envía un sonido específico al sintetizador para que lo reproduzca.

- **DrumKit:** Representa la batería. Está compuesto por ocho pads (botones) que utilizan un script "Drum Pad Trigger" para reproducir diferentes sonidos de batería.

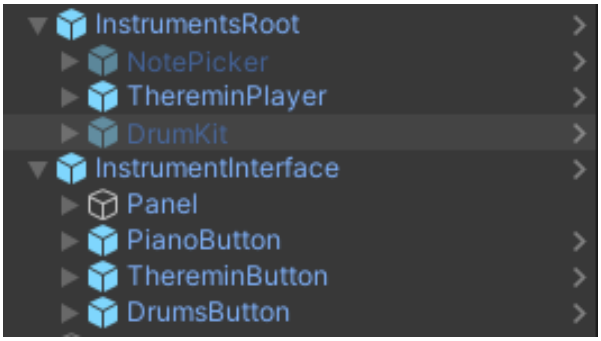


Figura 7.6 Jerarquía de objetos de los instrumentos

7.4.2.4 Gestor de pistas

El gestor de pistas está compuesto por un panel que contiene tres casillas para pistas, cada una de ellas con sus correspondientes botones de reproducción y gestión. Además, existe un botón común para grabar nuevas pistas.

Cada una de las casillas de pista está representada por un prefab personalizado llamado TrackPrefab. Están compuestas por cuatro objetos:

- **PlayButton**: Controla la reproducción de una pista específica. En su estado inicial, cuando se pulsa, se reproduce la pista. Si se vuelve a pulsar, la pista se pausa y se puede reanudar en el mismo punto. Si se mantiene pulsado, la pista se para y al reanudarse volverá a comenzar desde el principio. Utiliza el script "PlayButtonTrigger".
- **ModifyButton:** Gestiona el renombrado de una pista específica. Cuando se pulsa, lanza una llamada al método del sistema TouchScreenKeyboard.Open(), que invoca el teclado de sistema Oculus (por lo que no funciona mientras la aplicación se ejecuta en el ordenador). Utiliza el script "ModifyButtonTrigger".
- **DeleteButton:** Elimina una pista específica. Cuando se pulsa, hace una llamada al TrackManager para que elimine la pista. Utiliza el script "DeleteButtonTrigger".
- **TrackName:** Elemento de interfaz de tipo campo de texto que representa el nombre de la pista asignada a cada casilla. Para un mayor control sobre este texto, se utiliza el componente TextUI del paquete gratuito TextMeshPro.

El botón "RecordButton" hace una referencia al objeto TrackManager para que inicie o termine la grabación de una pista cada vez que es pulsado. Utiliza el script "RecordButtonTrigger".

Además, cada TrackPrefab utiliza un TrackHolder que hace referencia a una pista única en el sistema.

7.4.2.5 Selector de herramientas

El selector de herramientas es un objeto complementario al gestor de pistas y al panel de selección de instrumentos. Permite cambiar entre uno y otro utilizando el script "Tool Selector Button Trigger" en cada uno de los botones (InstrumentsButton y TrackManagerButton).

7.4.2.6 Adaptación del código del sintetizador

Como se ha comentado en capítulos anteriores, el desarrollo desde cero de un sintetizador de audio, no se contempló como un objetivo principal dentro del alcance del proyecto por la complejidad que su implementación suponía en cuanto a horas de trabajo y conocimientos específicos de este tipo de sistemas. Por lo tanto, se utilizó una base de código ya existente con licencia MIT de un proyecto open-source.

Este código había sido desarrollado para un pequeño juego musical (Unity Synthesizer in C#), por lo que contenía muchos componentes extras que no eran necesarios para su uso en este proyecto.

Si bien las pruebas realizadas inicialmente con este código resultaron satisfactorias, fue necesario adaptar parte del código para hacerlo usable en este proyecto. En un primer momento, se hizo un análisis de los componentes del sistema para identificar cuáles serían de utilidad en el proyecto.

Para comenzar, el proyecto existente era un proyecto 2D (una aplicación de escritorio y móvil), en la cual los inputs procedentes del ratón pinchaban en un diferentes "baldosas" para generar notas específicas, que además generaban ritmos de forma aleatoria al ser pulsadas.

Por lo tanto, se identificaron dos sistemas distintos, uno que generaba sonido y otro que generaba ritmos. La parte del código interesante para el proyecto era la encargada de generar sonidos (sintetizador): en este caso, un objeto Synth era el responsable de contener todos los scripts y módulos relacionados. Sin embargo, y debido a que a fin de cuentas es un proyecto para Unity, toda la lógica estaba fuertemente acoplada a la interfaz. Una de las tareas principales de esta adaptación del código fue desacoplar estos módulos.

La clase principal a través de la cual se interactuaba con el sintetizador era la clase "TouchKey": un script derivado de MonoBehavior que recibía toques en la interfaz (a través del sistema RayCast) y los utilizaba para lanzar notas al sintetizador.

```
. . .

if (Input.GetMouseButton(0) || Input.GetMouseButtonDown(0) || Input.GetMouseButtonUp(0))
{

        touchesOld = new GameObject[touchList.Count];
        touchList.CopyTo(touchesOld);
        touchList.Clear();

        RaycastHit2D hit =
Physics2D.Raycast(gameCam.ScreenToWorldPoint(Input.mousePosition), Vector2.zero);

                GameObject recipient = hit.transform.gameObject;
                var tDetails = recipient.GetComponent<TileDetails>();
                reverb.wetMix = hit.transform.position.x / 24;
                if(hit.collider!=null) {
                        touchList.Add(recipient);
                        if (Input.GetMouseButtonDown(0)) {
                                lastNoteDegree = scale + (int)tDetails.note;
                                envelope = synth.KeyOn(lastNoteDegree, envelope);
                                tDetails.OnTouchDown();
                        }
                        if (Input.GetMouseButtonUp(0)) {
                                envelope.KeyOff();
                                tDetails.OnTouchExit();
                        }
                        if (Input.GetMouseButton(0)) {
                                lastNoteDegree = scale + (int)tDetails.note;
                                envelope = synth.KeyOn(lastNoteDegree, envelope);
                                tDetails.OnTouchStay();

                        }
. . .
```

Evidentemente, este código no era útil para el proyecto ya que dependía en gran manera del sistema de cámaras, Raycast e input genérico de Unity. Por otro lado, la lógica relacionada con enviar notas al sintetizador se puede ver en la porción de código anterior dentro de las declaraciones if (Input…), invocando al método SynthControl.KeyOn(), con la nota deseada y pasando el módulo envelope como parámetro.

Habiendo identificado el código a adaptar, se planteó crear un sistema similar para los instrumentos melódicos (piano y theremín), por lo que habría que crear clases que enviaran este tipo de llamadas a los métodos del sintetizador.

En el caso del piano, se investiga sobre cómo crear un sistema de eventos en Unity. La idea inicial es que haya una clase “KeyListener” que se encargue de estar pendiente de si se lanzan eventos desde las teclas de piano. Estos eventos corresponden a los que se lanzan cuando se presionan o se sueltan las teclas. KeyListener define los eventos OnKeyPressed, OnKeyPushed y OnKeyReleased.

Después, se crea una clase PianoButtonTrigger. Esta clase se comporta de manera similar al resto de botones: cuando el botón (en este caso tecla) alcanza un límite, se invoca un método. En este caso se lanza un evento de la clase KeyListener, el correspondiente a si se ha alcanzado el límite (cuando se pulsa la tecla) o si ha vuelto a su posición original (cuando se suelta).

Cada una de las teclas contiene un componente derivado de MonoBehavior llamado PressPianoKey. Este componente tiene como parámetros la nota y la escala de la nota que se va a enviar, y se suscribe a los métodos de la clase KeyListener. Aquí es donde se utiliza la sintaxis vista anteriormente (en el código base) y se envía la nota deseada al sintetizador.

```
public void Start() {
        . . .

        KeyListener.Instance.OnKeyPressed += OnKeyTouched;
        KeyListener.Instance.OnKeyPushed += OnKeyTouched;
        KeyListener.Instance.OnKeyReleased += OnKeyReleased;

    }

    /// <summary>
    /// Evento que se lanza cuando se toca la nota en el piano.
    /// Se lanza una nota al sintetizador.
    /// </summary>
    /// <param name="id">Id de la nota que se toca</param>
    private void OnKeyTouched(int id) {
        if (id == this.id) {
            lastNoteDegree = scale + (int) note;
            envelope = synth.KeyOn(lastNoteDegree, envelope);
        }
    }

    /// <summary>
    /// Evento que se lanza cuando se suelta la nota del piano
    /// </summary>
    /// <param name="id">Id de la nota que se toca</param>
    private void OnKeyReleased(int id) {
        if (id == this.id) {
            envelope.KeyOff();
        }
    }
```

Habiendo desarrollado ya la clase KeyListener, desarrollar un sistema similar al de las teclas de piano para el theremin resultó relativamente sencillo: la mayor dificultad residía en detectar cuándo las manos estaban realizando el gesto pinch para invocar un nuevo evento definido en KeyListener.

La manera más sencilla de comprobar este tipo de gestos es a través de los métodos proporcionados por el paquete Oculus XR Plugin. Cada una de las manos estaba representada en código con la clase OVRHand, que proporciona el método OVRHand.GetFingerIsPinching.

Cuando el theremín está activado en la escena, se comprueba en cada frame si el método anterior está dando valores booleanos true o false. En caso afirmativo, se lanza el evento RightHandPinched o LeftHandPinched (dependiendo de qué mano esté haciendo el gesto, o las dos a la vez) utilizando la posición de la mano como parámetro.

De este modo, la posición de la mano es utilizada como en un theremín real: dependiendo de la coordenada X o Y de la mano derecha en el espacio, cambiará de escala y nota. Una diferencia interesante en cuanto al diseño de un theremín real es que en este caso, la mano izquierda se utiliza para modificar un parámetro de un módulo del sintetizador: en el instrumento real, se controla el volumen.

```
/// <summary>
/// Método que se lanza cuando se hace el gesto "pinch" con la mano derecha.
/// Genera un HandModulatorPrefab en la posición de la mano.
/// Lanza un sonido al sintetizador dependiendo del valor X e Y de la posición de la
mano.
/// </summary>
/// <param name="handPos">Posición global de la mano derecha</param>
private void OnRightHandPinched(Vector3 handPos) {

        if (!rightHandModulator) {
            rightHandModulator = Instantiate(HandModulatorPrefab, new Vector3(handPos.x,
handPos.y, handPos.z + 0.1f), Quaternion.identity, null);
        }
        rightHandModulator.transform.LookAt(Camera.main.transform);

        GameObject cursor =
rightHandModulator.transform.FindChildRecursive("Cursor").gameObject;

        cursor.transform.localPosition = handPos;
        var cursorPos = cursor.transform.localPosition;

        var scale_mod = Mathf.Round(Normalize(cursorPos.y, 0f, 8f));
        var note_mod = Mathf.Round(Normalize(cursorPos.x, 1f, 24f));

        lastNoteDegree = scale_mod + note_mod;
        envelope = synth.KeyOn((int)lastNoteDegree, envelope);
    }
```

7.4.3 Descripción Detallada de las Clases

Para facilitar la lectura del documento, se adjunta junto a los archivos del proyecto bajo el directorio “Documentación”.

Capítulo 8. Desarrollo de las Pruebas

8.1 Pruebas de Integración

A continuación, se muestran los resultados de las pruebas definidas en la especificación, distribuidas por casos de uso:

Caso de Uso CUA01 – Activar el piano				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI01	Activar piano	El usuario localiza el botón de activación de instrumentos. Presiona el botón "Piano".	Se activa el objeto Piano. Si hay otro instrumento activo en la escena, se desactiva.	Correcto

Caso de Uso CUA02 – Activar el theremín				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI02	Activar theremín	El usuario localiza el botón de activación de instrumentos. Presiona el botón "Theremin".	Se activa el objeto Theremin. Si hay otro instrumento activo en la escena, se desactiva.	Correcto

Caso de Uso CUA03 – Activar la batería				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI03	Activar batería	El usuario localiza el botón de activación de instrumentos. Presiona el botón "Drums".	Se activa el objeto Batería. Si hay otro instrumento activo en la escena, se desactiva.	Correcto

Caso de Uso CUA04 – Tocar instrumento activado				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI04.1	Tocar instrumento Piano	El usuario ha activado el Piano previamente. Para tocar una nota, pulsa con la mano	Se produce un sonido específico para cada tecla del piano.	Correcto

		en cualquiera de las teclas del piano.		
PI04.2	Tocar instrumento Theremín	El usuario ha activado el Theremín previamente. Para tocar una nota, realiza el gesto "pinch" con su mano derecha (juntando dedos índice y pulgar) y mueve la mano en los ejes X e Y.	Se produce un sonido con una frecuencia en particular dependiendo de la posición de la mano en el espacio.	Correcto
PI04.3	Tocar instrumento Batería	El usuario ha activado la Batería previamente. Para tocar una nota, pulsa con la mano en cualquiera de los pads de batería.	Se produce un sonido específico para cada pad de batería.	Correcto

Caso de Uso CUA05 – Modificar oscilador				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI05.1	Modificar valor del oscilador (piano)	El usuario ha activado el Piano previamente. El usuario localiza el deslizador correspondiente al modificador del oscilador. Lo empuja con la mano verticalmente hacia arriba o hacia abajo.	El valor multiplicador del oscilador varía dependiendo de la posición en la que se encuentre.	Correcto

Caso de Uso CUA06 – Modificar bitcrush				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI06	Modificar valor del bitcrush	El usuario ha activado el Piano previamente. El usuario localiza el deslizador correspondiente al modificador del bitcrush. Lo empuja con la mano verticalmente hacia arriba o hacia abajo.	El valor de downsample del bitcrush varía dependiendo de la posición en la que se encuentre.	Correcto

Caso de Uso CUA07 – Modificar multiplicador				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI07	Modificar valor multiplicador del oscilador	El usuario ha activado el Theremín previamente. Para modificar el valor del oscilador, realiza el gesto "pinch" con su mano izquierda (juntando dedos índice y pulgar) y mueve la mano en el eje Y.	El valor multiplicador del oscilador varía dependiendo de la posición en la que se encuentre.	Correcto

Caso de Uso CUA08 – Modificar frecuencia				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI08	Modificar valor de la frecuencia del sintetizador	El usuario ha activado el Theremín previamente. Para modificar el valor de la frecuencia, realiza el gesto "pinch" con su mano derecha (juntando dedos índice y pulgar) y mueve la mano en el eje Y.	El valor de la frecuencia varía dependiendo de la posición en la que se encuentre.	Correcto

Caso de Uso CUP01 – Grabación de pista				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI07.1	Grabar una nueva pista	El usuario ha activado el gestor de pistas previamente. El usuario localiza y pulsa el botón "grabar". Toca los instrumentos durante un tiempo y vuelve a pulsar el botón grabar.	Se crea una nueva pista en el sistema de archivos.	Correcto
PI07.2	Grabar una nueva pista (límite de pistas alcanzado)	El usuario ha activado el gestor de pistas previamente. Debe haberse alcanzado el número máximo de pistas. El usuario localiza el botón "grabar" y lo pulsa.	No se comienza a realizar la grabación y se muestra un mensaje de advertencia al usuario.	Correcto

Caso de Uso CUP02 – Renombrar pista

Identificador	Prueba	Procedimiento	Resultado Esperado	Resultado Obtenido
PI08.1	Renombrar pista	El usuario ha activado el gestor de pistas previamente. El usuario localiza y pulsa el botón "renombrar" de una pista. Aparece un teclado sobre el cual el usuario escribe el nuevo nombre. Pulsa "Enter" en el teclado.	Se renombra la pista en la aplicación y en el sistema de archivos.	Correcto
PI08.2	Renombrar pista (nombre en uso)	El usuario ha activado el gestor de pistas previamente. El usuario localiza y pulsa el botón "renombrar" de una pista. Aparece un teclado sobre el cual el usuario escribe el nuevo nombre (idéntico a uno ya existente en el sistema). Pulsa "Enter" en el teclado.	No se renombra la pista y se muestra un mensaje de advertencia al usuario.	Correcto
PI08.3	Renombrar pista (número de caracteres menor que el mínimo)	El usuario ha activado el gestor de pistas previamente. El usuario localiza y pulsa el botón "renombrar" de una pista. Aparece un teclado y el usuario no escribe nada. Pulsa "Enter" en el teclado.	No se renombra la pista y se muestra un mensaje de advertencia al usuario.	Correcto
PI08.4	Renombrar pista (número de caracteres mayor que el máximo)	El usuario ha activado el gestor de pistas previamente.	No se renombra la pista y se muestra un mensaje de advertencia al usuario.	Correcto

		El usuario localiza y pulsa el botón "renombrar" de una pista. Aparece un teclado sobre el cual el usuario escribe el nuevo nombre (superando el límite de caracteres establecido) Pulsa "Enter" en el teclado.		

Caso de Uso CUP03 – Eliminar pista				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI09	Eliminar pista	El usuario ha activado el gestor de pistas previamente. No debe haber ninguna pista en reproducción. El usuario localiza y pulsa el botón "eliminar" de una pista.	Se elimina la pista seleccionada.	Correcto

Caso de Uso CUP04 – Reproducir pista				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI10	Reproducir pista	El usuario ha activado el gestor de pistas previamente. No debe haber ninguna pista en reproducción. El usuario localiza y pulsa el botón "reproducir" de una pista.	Se reproduce la pista seleccionada.	Correcto

Caso de Uso CUP05 – Pausar pista				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI11	Pausar pista	El usuario ha activado el gestor de pistas previamente.	Se pausa la pista seleccionada.	Correcto

		No debe haber ninguna pista en reproducción. El usuario localiza y pulsa el botón "pausar" de una pista.		

Caso de Uso CUP06 – Reiniciar pista				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PI12	Reiniciar pista	El usuario ha activado el gestor de pistas previamente. No debe haber ninguna pista en reproducción. El usuario localiza y pulsa el botón "reiniciar" de una pista.	Se pausa y reinicia la pista seleccionada.	Correcto

8.2 Pruebas de Sistema

8.2.1 Contexto de las pruebas

Como se indicó en el apartado de definición, se probará si el funcionamiento del sistema Oculus es correcto y se comunica de forma adecuada con la aplicación.

Como entornos de pruebas, se utilizaron lugares cotidianos (un estudio, un salón) con suficientes puntos descriptores para facilitar el funcionamiento del sistema de tracking. La iluminación debe ser estática y sin puntos fuertes de luz que "cieguen" las cámaras, y las manos del usuario deben poder verse sin ningún problema.

8.2.2 Resultados

Prueba tracking del entorno				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PS01	Tracking del entorno correcto	El usuario debe colocarse las Oculus Quest y comprobar que el sistema de tracking funciona correctamente.	No se producen saltos ni temblores en el entorno virtual, ni aparece ningún mensaje de error.	Correcto

Prueba barrera de juego				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PS02	Comprobación de la barrera de juego	El usuario debe colocarse las Oculus Quest y caminar físicamente por el entorno.	Al acercarse al límite de juego, debe aparecer una barrera virtual que indique que el usuario está saliendo de la zona segura.	Correcto

Prueba posicionamiento espacial				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PS03	Prueba de posicionamiento espacial	El usuario debe colocarse las Oculus Quest y caminar físicamente por el entorno, girar la cabeza y moverse en el sitio.	El sistema debe representar con fidelidad los movimientos del usuario, sin latencia, y con una correspondencia real entre el movimiento físico y el virtual.	Correcto

Prueba hand tracking				
Identificador	**Prueba**	**Procedimiento**	**Resultado Esperado**	**Resultado Obtenido**
PS04	Prueba de sistema de hand tracking	El usuario debe colocarse las Oculus Quest y colocar las manos delante de su cara.	El sistema debe representar las manos del usuario en 3D delante de él. El sistema debe responder ante el movimiento y rotación de los dedos y manos.	Correcto

8.3 Pruebas de Usabilidad

8.3.1 Contexto de las pruebas

Estas pruebas se realizaron sobre 6 usuarios voluntarios (familiares y amigos del desarrollador del proyecto), procedentes de distintos trasfondos en cuanto a conocimiento y experiencia con la tecnología de entretenimiento de consumo.

8.3.2 Resultados

A continuación, se presentan los resultados de las pruebas de usabilidad. Se muestran las respuestas de los usuarios agrupadas por cada pregunta realizada.

8.3.2.1 Experiencia con sistemas informáticos

Uso el ordenador o smartphone …	U1	U2	U3	U4	U5	U6
1 - Nunca 2 - Muy ocasionalmente 3 - De vez en cuando 4 - Varias veces por semana 5 - Todos los días	4	4	5	5	5	2
Tengo experiencia con…						
1 – Ningún programa. Siempre necesito ayuda 2 – Navegador y correo 3 – Programas de ofimática 4 – Instalación de programas, aplicaciones de trabajo 5 – Cualquier tipo de software, soy programador	4	2	2	2	5	2
He jugado a videojuegos…						
1 – Nunca 2- Muy ocasionalmente 3 – De vez en cuando 4 – Varias veces por semana 5 – Todos los días	3	1	3	3	4	1
He utilizado algún controlador distinto a ratón y teclado (pueden elegirse varias opciones)						
1 – Ninguno 2 – Mando de consola 3 – Smartphone 4 – Joystick/Volante 5 – Realidad virtual	2,3,5	1	2	2	2,3,4,5	1

8.3.2.2 Experiencia con la aplicación

Valoración general	U1	U2	U3	U4	U5	U6
1 – No me ha gustado 2 – Tiene bastante que mejorar 3 – Regular 4 – Me ha gustado 5 – Me ha encantado	4	4	4	5	4	3
Usaría este tipo de aplicación como entretenimiento						
1 – No 2 – Lo veo poco útil 3 – Es entretenido, pero no lo usaría 4 – Sí, es interesante, lo usaría 5 – Sí, me ha encantado y lo usaría a menudo	3	3	4	5	4	2
Tengo experiencia musical …						
1 – No me gusta la música 2 – Me gusta, pero no sé tocar instrumentos 3 – Escucho y toco algún instrumento 4 – Toco varios instrumentos 5 – Soy músico profesional/productor	2	2	2	3	5	2
La interfaz me ha parecido…						
1 – Nada intuitiva 2 – Muy poco intuitiva 3 – Mejorable 4 – Buena 5 – Muy fácil de usar	4	3	4	4	4	3
He podido experimentar con… (se pueden marcar varias opciones)						
1 – Instrumentos (piano) 2 – Instrumentos (theremín) 3 – Instrumentos (batería) 4 – Grabación de pistas 5 – Gestión de pistas	*	1,2,3,5	*	*	*	1,2,3
Me han gustado las opciones que ofrece						
1 – Muy pobre 2 – Añadiría más 3 – Está bien 4 – Podría tener alguna mejora 5 – No añadiría nada	3	5	4	4	4	3
Esta aplicación ha cambiado mi opinión sobre la realidad virtual						
1 – No 2 – Apenas 3 – Me quedo como estaba 4 – Ha cambiado ligeramente 5 – Ha cambiado radicalmente	3	4	3	3	3	2

* Todas las opciones

Sobre la pregunta anterior: ¿es su opinión actual favorable o desfavorable?
• Usuario 1: Favorable • Usuario 2: Favorable • Usuario 3: Favorable • Usuario 4: Favorable • Usuario 5: Favorable • Usuario 6: Desfavorable
Observaciones, comentarios o sugerencias:
Usuario 1: Añadiría algún tipo de guía o tutorial para la parte de grabación de pistas, o algún mensaje que saliera al tocar los botones del gestor. Tampoco hay feedback cuando se pulsa un botón Usuario 2: El hand tracking a veces se activa solo (cuando una de las cámaras tiene una mano en el campo de visión) Usuarios 3 y 4: Sin comentarios Usuario 5: Como músico, la premisa parece muy interesante y puliendo algunos detalles tiene potencial. Usuario 6: Sin comentarios

8.3.2.3 Cuestionario para el responsable de las pruebas

Aspecto Observado	**Resultados**
¿Se desorienta el usuario en una primera instancia al colocarse las gafas?	• Usuario 1: No • Usuario 2: Ligeramente, tuvo que girarse para encontrar el panel de instrumentos • Usuario 3: No • Usuario 4: No • Usuario 5: No • Usuario 6: Sí, no entiende muy bien lo que está viendo
¿Le cuesta al usuario aprender a usar la interfaz?	• Usuario 1: No • Usuario 2: Ligeramente • Usuario 3: No • Usuario 4: No • Usuario 5: No • Usuario 6: Sí, hay que explicarle qué es cada cosa
¿Entiende el usuario el sistema de reconocimiento de gestos/hand tracking?	• Usuario 1: Sí, ya lo había utilizado • Usuario 2: Sí, intuitivamente • Usuario 3: Sí, intuitivamente • Usuario 4: Sí, intuitivamente • Usuario 5: Sí, ya lo había utilizado • Usuario 6: No en un principio
¿Puede el usuario volver a centrarse si se pierde dentro de la escena?	• Usuario 1: Sí • Usuario 2: Sí • Usuario 3: Sí • Usuario 4: Sí • Usuario 5: Sí

	• Usuario 6: Sí, con ayuda
¿Qué comentarios hace el usuario en voz alta al probar la aplicación?	• Usuario 1: Sugerencias de mejora. Intenta salir de la zona de juego. • Usuario 2: Agrado, pregunta algunas cosas • Usuario 3: No hace comentarios • Usuario 4: Agrado e interés • Usuario 5: Interés, intenta provocar fallos en el programa • Usuario 6: Confusión, no sabe muy bien qué hacer
¿Necesita ayuda el usuario en algún momento?	• Usuario 1: No • Usuario 2: Sí, con la gestión de pistas • Usuario 3: Sí, con la gestión de pistas • Usuario 4: No • Usuario 5: No • Usuario 6: Sí, durante toda la prueba
Errores leves cometidos	• Usuario 1: Intentar borrar una pista mientras sonaba • Usuario 2: Ninguno • Usuario 3: Ninguno • Usuario 4: Ninguno • Usuario 5: Intentar renombrar y borrar una pista mientras sonaba • Usuario 6: No moverse para tocar los botones, no mantener el gesto para hacer sonar el theremin
Errores graves cometidos	• Usuario 1: Ninguno • Usuario 2: Ninguno • Usuario 3: Ninguno • Usuario 4: Ninguno • Usuario 5: Ninguno • Usuario 6: Ninguno

8.3.3 Análisis de resultados de las pruebas

8.3.3.1 Facilidad de uso

Los usuarios, por lo general, aprendieron bastante rápido a utilizar los diferentes elementos de la interfaz. Tuvieron tiempo suficiente para probar la mayoría de las funcionalidades y les pareció relativamente sencillo utilizar la aplicación. Un usuario en particular, por diferencia de edad con el resto, tardó más tiempo en entender cómo funcionaba el propio sistema de hand tracking y cómo interactuar con la interfaz.

8.3.3.2 Funcionalidades

Por lo general, los usuarios vieron bastante intuitivo el sistema de interfaz e instrumentos. En cuanto a la gestión de pistas, algunos tardaron más en entenderlo, pero la mayoría hizo funcionar todos los sistemas sin importantes problemas. En caso del usuario de más edad, fue más complicado y sólo pudo utilizar los instrumentos.

8.3.3.3 Mejoras aplicadas

8.3.3.3.1 Feedback en botones

Debido a algunos comentarios por parte de los usuarios, que no estaban seguros de si habían pulsado o no algún botón (ya que no existe feedback háptico al no utilizar mandos), se añadió un sonido “clic” que se reproduce cada vez que se pulsa un botón.

8.3.3.3.2 Indicador de grabación en curso

Del mismo modo, a veces los usuarios decían que no estaba claro si había comenzado la grabación. Por lo tanto, se cambia el color del botón de grabación cuando hay una grabación en curso.

8.3.3.4 Valoración general

El resultado general de las pruebas es que los usuarios coinciden en que es algo bastante novedoso e innovador, y que tiene potencial para ser una aplicación de entretenimiento con uso si se añaden más funcionalidades e instrumentos.

Para casi todos los usuarios, la realidad virtual fue una experiencia nueva por lo que en un primer momento estuvieron bastante sorprendidos de cómo el sistema era capaz de reconocer sus manos y su movimiento en el entorno en tiempo real.

La valoración general es positiva, con lo que se puede concluir que el proyecto ha dado como resultado una aplicación satisfactoria para la mayoría de los usuarios considerados.

8.4 Pruebas de Rendimiento

Durante el desarrollo del proyecto, especialmente durante las fases de implementación se monitorizó constantemente el rendimiento de la aplicación utilizando las herramientas comentadas previamente (Unity Profiler y OVR Metrics Tool).

Además, si se ejecuta la aplicación desde el editor de Unity a través de Oculus Link (el software propietario de Oculus utiliza las gafas en modo "tethered", esto es, para utilizarlas en aplicaciones o juegos de PC) el sistema escribe logs indicando los FPS a los que está funcionando, lo cual facilita en gran medida el desarrollo de estas pruebas, ya que no hay que utilizar ningún software específico.

```
VrApi FPS=63/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=70/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=69/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=59/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=68/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=62/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=71/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=60/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=61/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=67/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=66/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=67/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=63/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
VrApi FPS=65/72,Prd=0ms,Tear=0,Early=0,Stale=0,VSnc=1,Lat=0,Fov=0,CPU4/GPU=2/3,1651/515MHz,OC=FF,TA=0/E0/0,SP=N/F/N,Mem=1804MHz,Free=934MB,
```

Figura 8.1 Resultado de las pruebas de rendimiento (Unity)

De todos modos, trabajar desde el Editor añade algo de carga de procesamiento extra, por lo que llegar al objetivo de un framerate exacto de 72 FPS no siempre se consigue, aunque a partir de 55-60 FPS la experiencia es buena, no existe tearing (desincronización entre la GPU y la pantalla) y la aplicación se nota fluida.

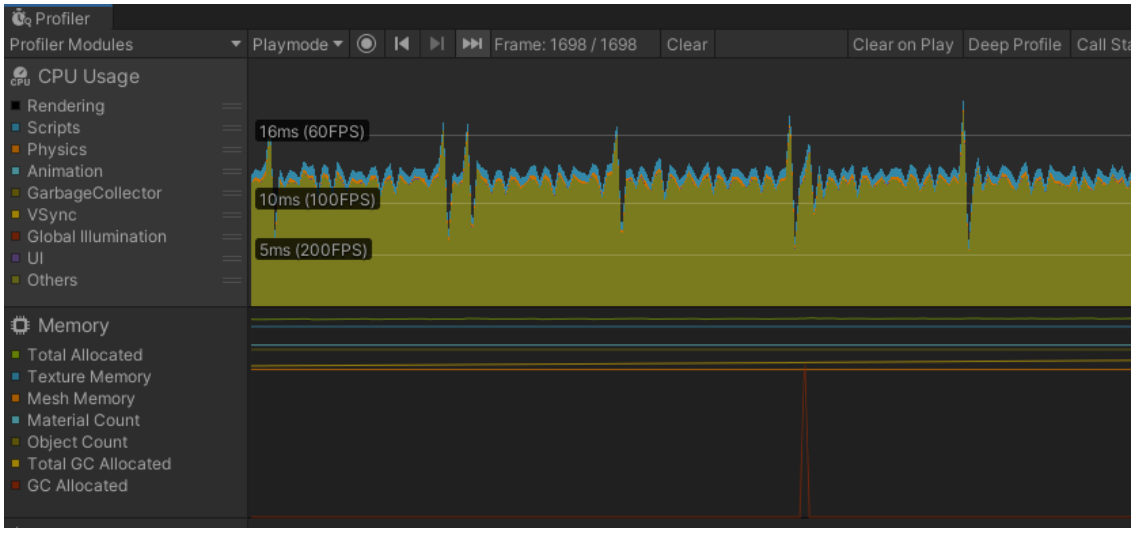


Figura 8.2 Resultado de las pruebas de rendimiento (Profiler)

Con el Unity Profiler se puede comprobar que rara vez los FPS bajan de 60, y si eso ocurre suele ser por llamadas periódicas que se hacen a métodos del Editor.

Si utilizamos la OVR Metrics Tool (directamente en el dispositivo) podemos comprobar que la aplicación se mantiene en 70-72 FPS estables cuando evitamos la carga de procesamiento extra que aparece si ejecutamos desde el PC.

Figura 8.3 Resultado de las pruebas de rendimiento (OVR Metrics Tool)

En la Figura 8.3 se puede apreciar un ligero pico de bajada de FPS, este se debe a que se llamó a la función del sistema que graba vídeo (a partir de la cual se obtuvo esta captura de pantalla).

Podemos concluir que el rendimiento del sistema es satisfactorio. Cumple con los requisitos especificados y con lo esperado en el plan de pruebas.

Capítulo 9. Manuales del Sistema

9.1 Manual de Instalación

Para instalar aplicaciones de manera no oficial (procedimiento conocido como “sideloading”) en un dispositivo Oculus Quest es necesario seguir los siguientes pasos:

9.1.1 Activar modo desarrollador

En primer lugar, para utilizar un dispositivo Oculus Quest es necesario tener una cuenta de Oculus (o Facebook si se han vinculado previamente) y acceder con ella a la aplicación Oculus para smartphones.

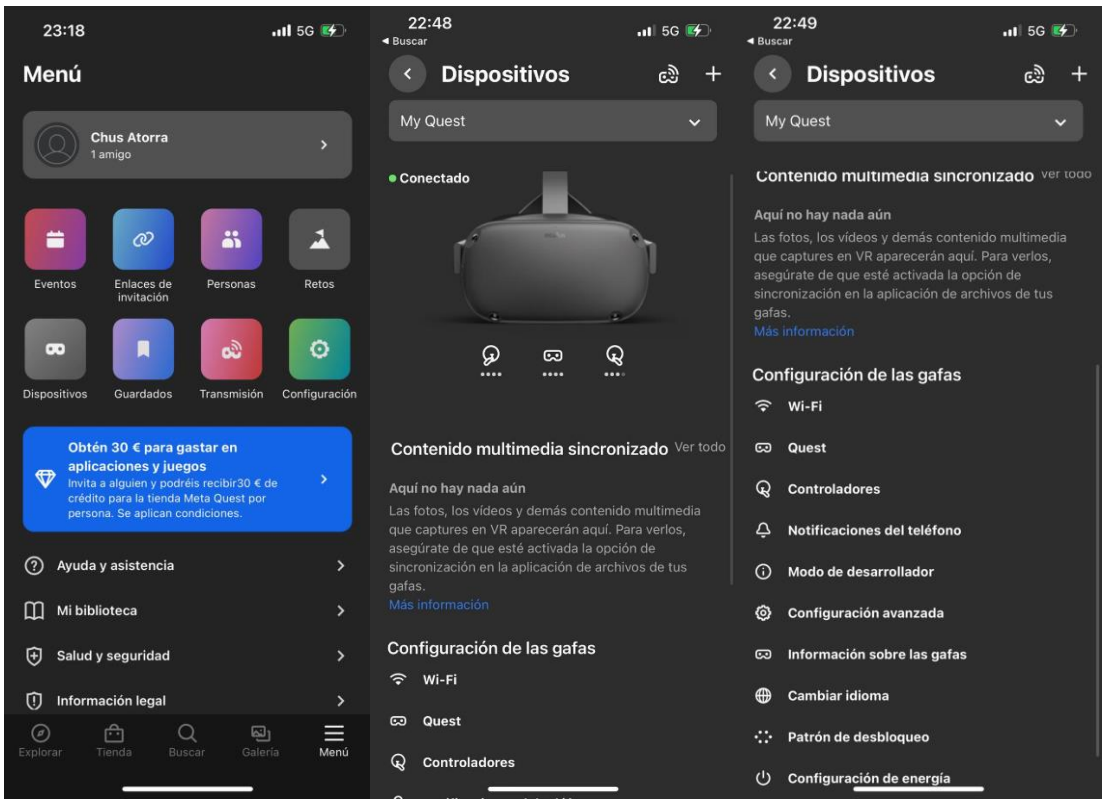


Figura 9.1 Activación del modo desarrollador

Desde esta aplicación, el usuario debe acceder a la sección “Dispositivos” y activar la opción “Modo de desarrollador”. Esta opción permite la conexión USB para depurar e instalar aplicaciones.

Una vez activada esta opción, conectar el dispositivo Oculus al ordenador por USB.

9.1.2 Instalación de la aplicación

9.1.2.1 Método 1: Instalación por adb (Android Debug Bridge)

Este método, aunque más rápido, requiere de cierto conocimiento en terminales de comandos. Es necesario instalar ADB (Android Debug Bridge) en el ordenador. [XD01]

Después, abrir la terminal en la ubicación en la que se haya extraído el .zip que contiene ADB y ejecutar el siguiente comando para comprobar que el dispositivo está conectado correctamente:

```
./adb devices
```

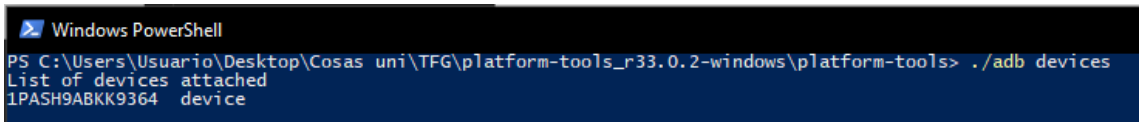


Figura 9.2 Comando adb devices

Si obtenemos un número de serie con el texto “device” después, significa que el dispositivo está conectado por ADB.

Después, para instalar la aplicación:

```
./adb install . \SynthVR.apk
```

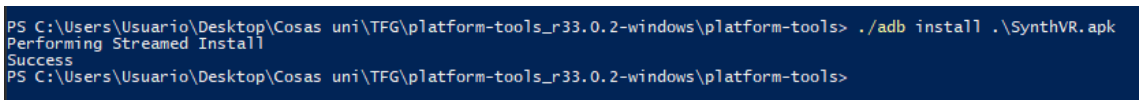


Figura 9.3 Comando adb install

Si recibimos el mensaje “Success” significa que la aplicación se ha instalado correctamente.

9.1.2.2 Método 2: Instalación por SideQuest

Para seguir este método, es necesario instalar SideQuest [SQ01].

SideQuest es un software de terceros que facilita la instalación de aplicaciones no oficiales en dispositivos Quest, además de poseer su propia tienda de apps alternativa y opciones de configuración del dispositivo avanzadas (generalmente ocultas al público general).

Instalar aplicaciones utilizando este software es sencillo. El usuario debe ir a la barra/menú superior y pinchar en el botón “Install APK file from folder on computer” (icono de una caja con una flecha):

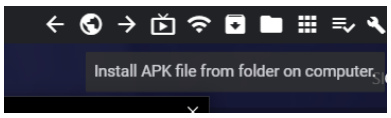


Figura 9.4 Botón "instalar APK" en SideQuest

Se abrirá un cuadro de búsqueda de archivos. El usuario debe navegar hacia donde se encuentre el archivo .apk a instalar, seleccionarlo y pulsar "Abrir" :

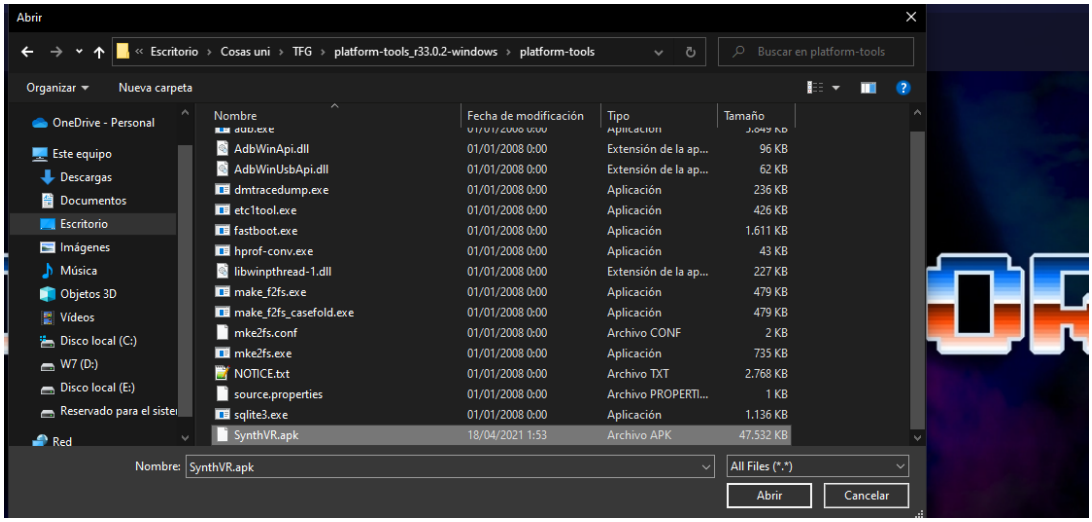


Figura 9.5 Búsqueda de ficheros en SideQuest

Si todo ha salido correctamente, aparecerá un mensaje diciendo: "APK installed OK!"

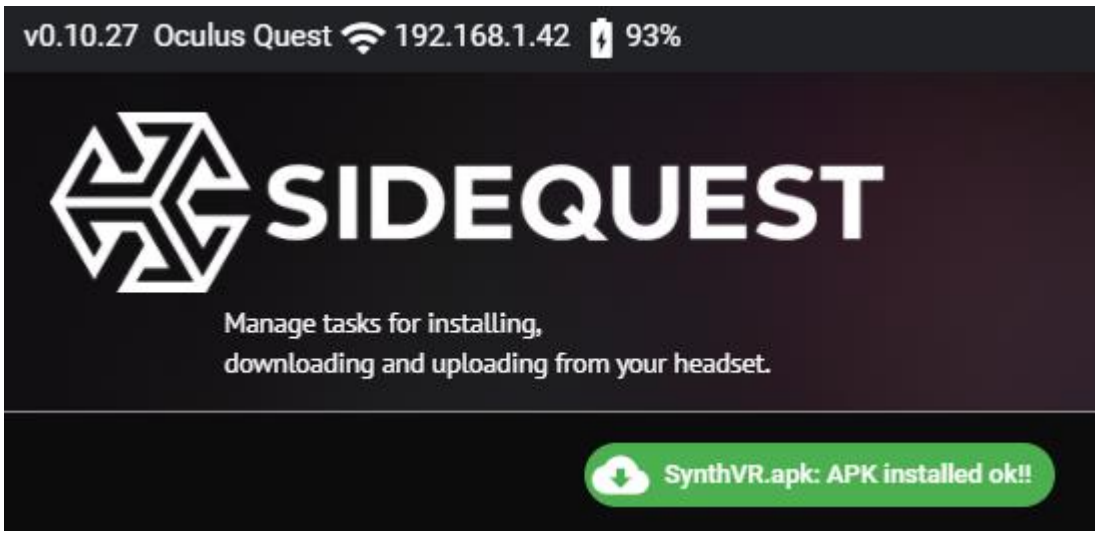


Figura 9.6 Mensaje de confirmación de SideQuest

9.2 Manual de Ejecución

Una vez instalada la aplicación, el usuario debe dirigirse a la librería de aplicaciones dentro del menú principal:

Figura 9.7 Menú principal del sistema Oculus

Después, en el dropdown superior derecho, seleccionar “orígenes desconocidos”:

Figura 9.8 Orígenes desconocidos en Oculus

Para abrir la aplicación, buscar la que tiene de nombre “SynthVR” y hacer clic directamente en ella:

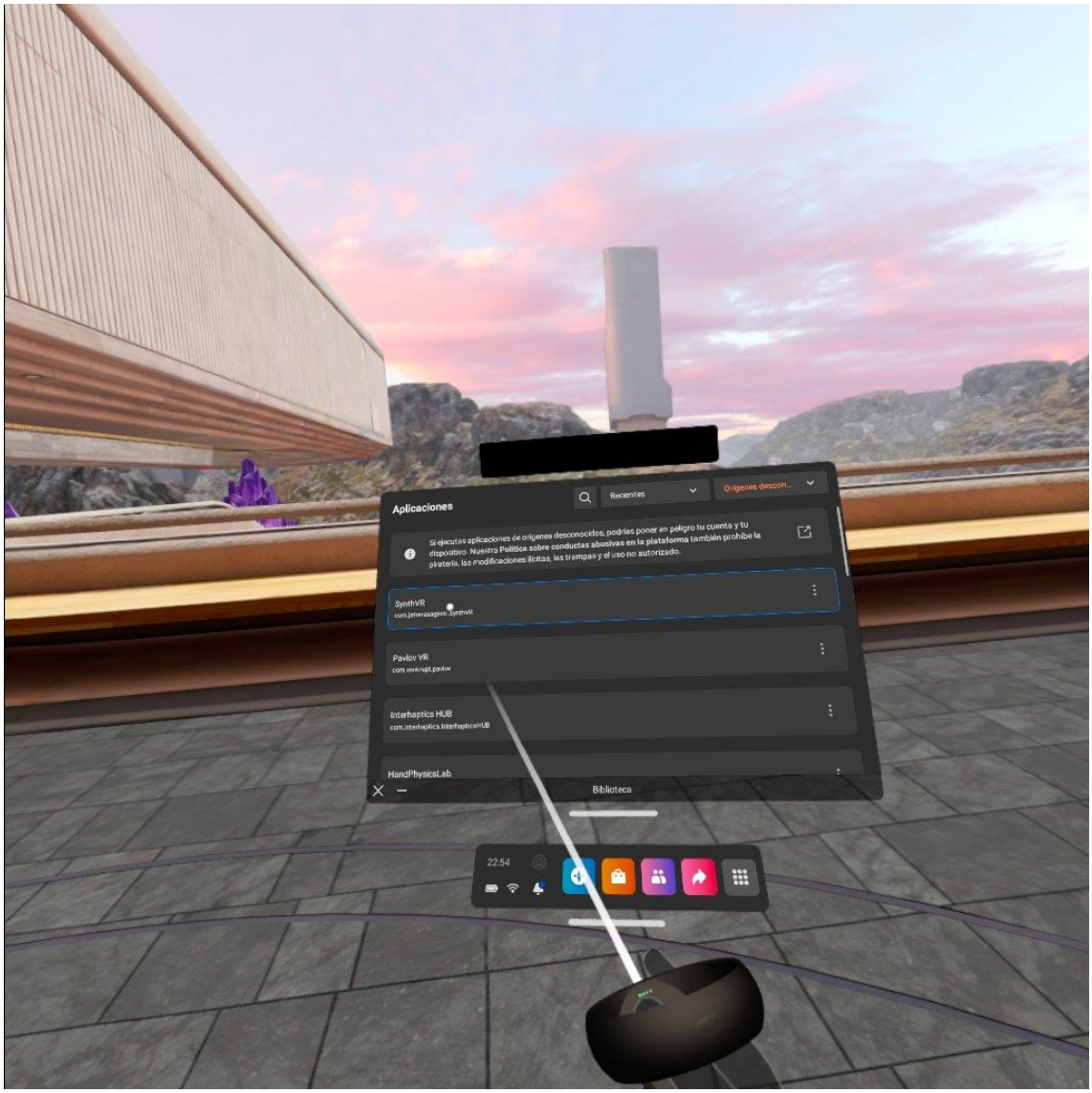


Figura 9.9 Ejecutar aplicación desde Orígenes desconocidos

9.3 Manual de Usuario

9.3.1 Interacción con la interfaz (controles)

Lo primero que el usuario va a encontrarse será un entorno con un cielo estrellado, varias rocas coloridas flotando a su alrededor, con un pedestal bajo sus pies y ciertos objetos y paneles a su alrededor.

Estos objetos son con los que el usuario va a interactuar mientras utilice la aplicación. Esta aplicación no es compatible con los controladores incluidos con el dispositivo Oculus: hace uso de la funcionalidad de hand tracking incluida en el sistema operativo. Esta funcionalidad utiliza las cámaras infrarrojas del dispositivo para localizar en tiempo real el comportamiento de las manos del usuario.

Por lo tanto, es necesario que las manos del usuario estén en el campo de visión de las cámaras de seguimiento del entorno y a una distancia no más de 1,5 m del dispositivo. De lo contrario, el seguimiento de las manos será inestable y provocará una mala experiencia de uso de la aplicación.

Cuando el usuario coloque las manos delante del dispositivo, estas aparecerán directamente representadas con un modelo 3D que será fiel a sus movimientos:

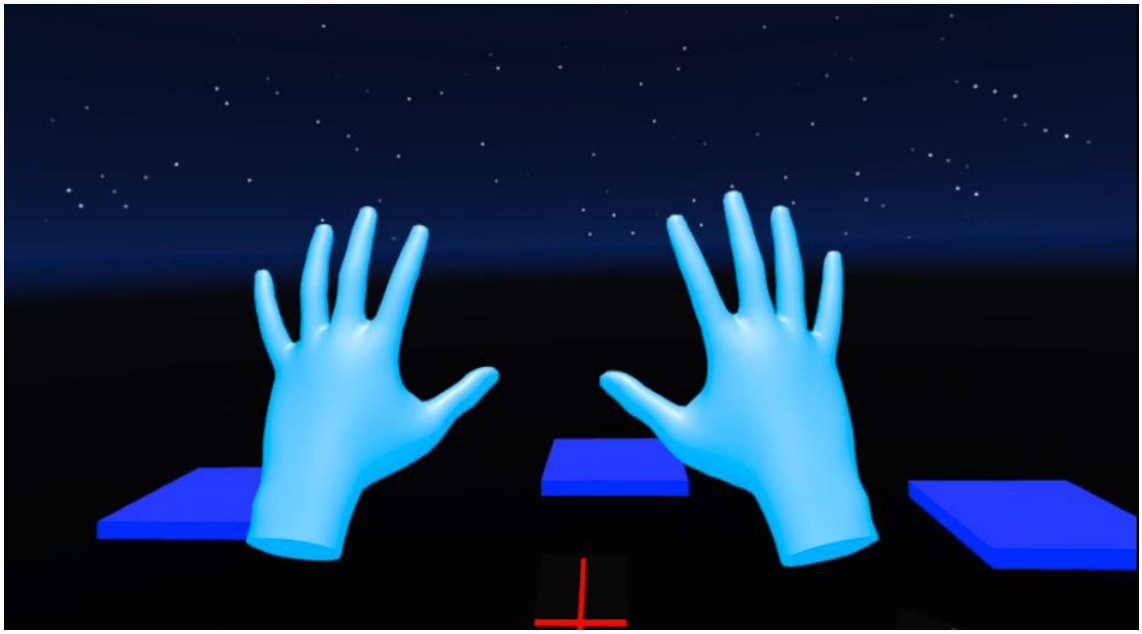


Figura 9.10 Hand tracking en el sistema

Para interactuar con las interfaces, el usuario simplemente debe hacer lo que haría en el mundo real: acercarse al objeto deseado (en este caso un botón o un deslizador) y utilizarlo físicamente. Esto se ilustra en Figura 9.11 y Figura 9.12

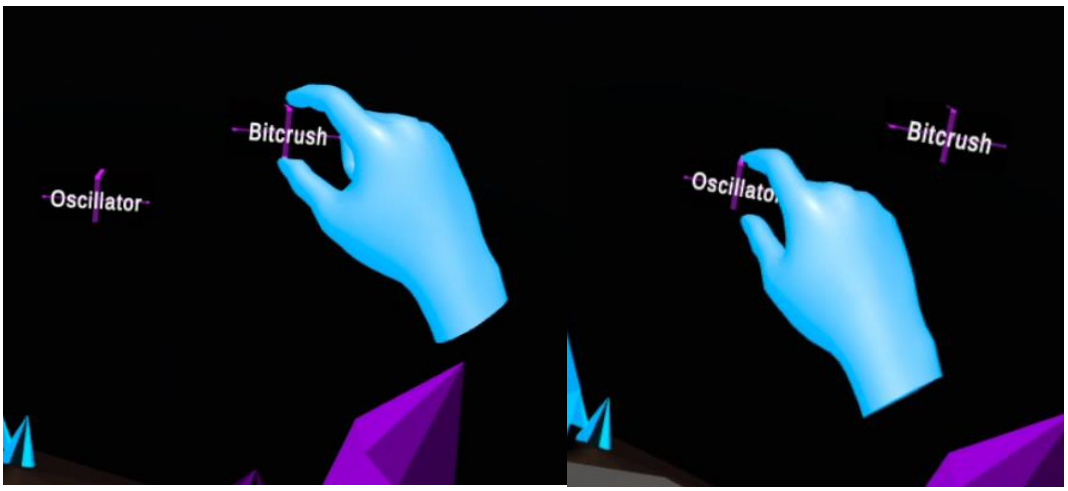


Figura 9.11 Interacción con deslizadores

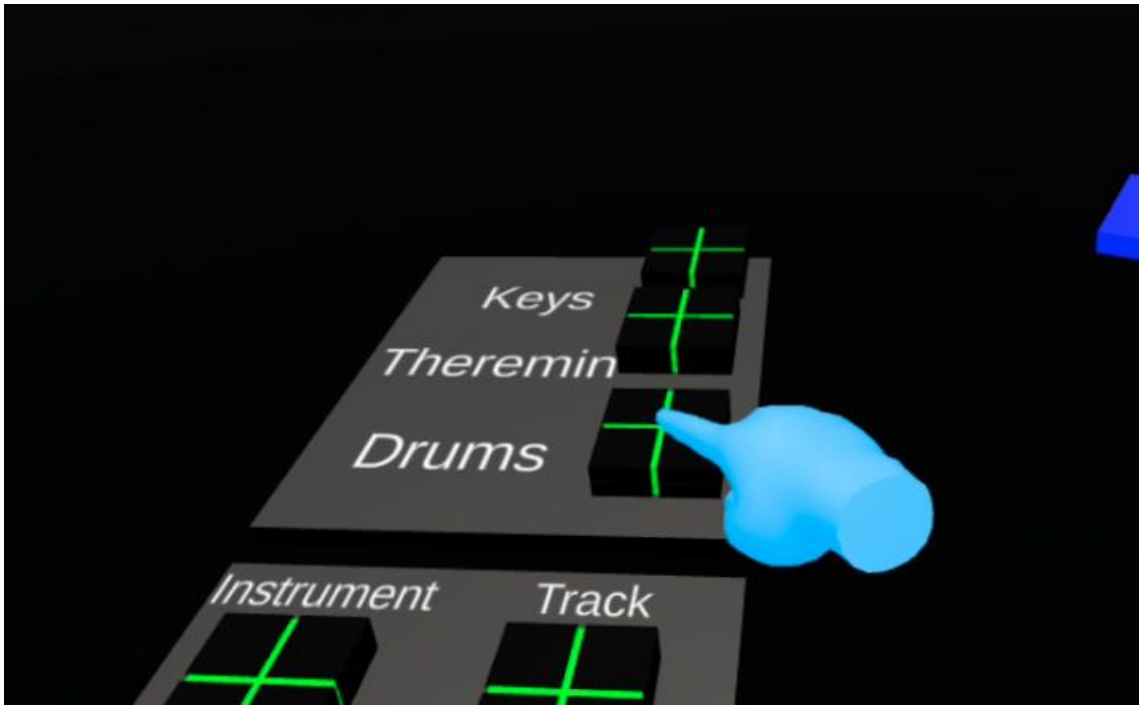


Figura 9.12 Interacción con botones de la interfaz

9.3.1.1 Funciones del sistema Oculus

Debido a que en el dispositivo no hay botones que permitan al usuario salir de la experiencia o realizar una grabación de vídeo, entre otras funciones del sistema Oculus, se hace uso de los comandos por gestos implementados desde el propio sistema operativo.

Para acceder a estos comandos, el usuario debe realizar el siguiente gesto: con la palma apuntando hacia sí mismo, juntar el dedo índice y pulgar durante unos segundos. Esto activará el menú contextual que permite activar estas funciones.

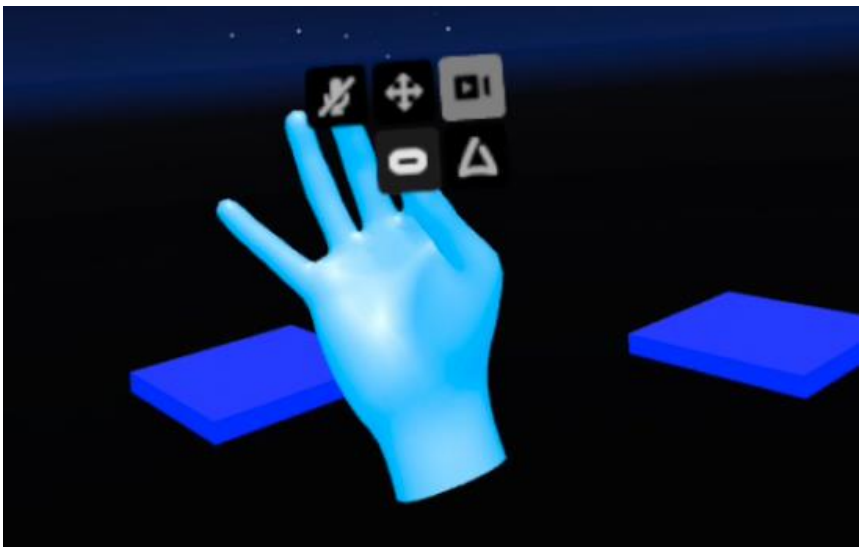


Figura 9.13 Activar el menú contextual del sistema Oculus

Manteniendo este gesto, el usuario debe desplazar la mano hacia el icono de la funcionalidad deseada, entre las que se encuentran:

- Botón Oculus: simula pulsar el botón "Menú" en los mandos. Permite salir de la aplicación y reanudarla, crear nuevas zonas de seguridad, etc.
- Desactivar micrófono: Desactiva el micrófono (no se utiliza en esta aplicación, pero el usuario podría estar haciendo una llamada por voz mientras tanto).
- Recentrar vista: Devuelve al usuario al punto de origen en la aplicación. Útil cuando el usuario camina y pierde la orientación o cuando se crea una nueva zona de seguridad.
- Asistente de voz: Activa la funcionalidad de comandos por voz del sistema operativo.
- Grabar vídeo: Comienza una grabación de vídeo de lo que está haciendo el usuario o la termina si ya hay una en curso.

9.3.2 Instrumentos

Para tocar los instrumentos, el usuario debe hacerlo como haría en un entorno real.

Para tocar el piano o la batería, simplemente pulsar las teclas o pads con la mano:

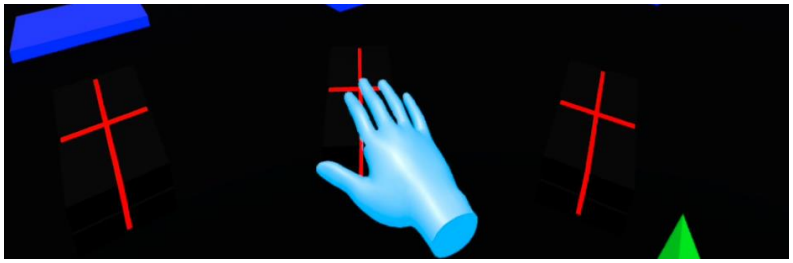


Figura 9.14 Tocar una tecla de piano

Es posible tocar más de un pad o botón a la vez.

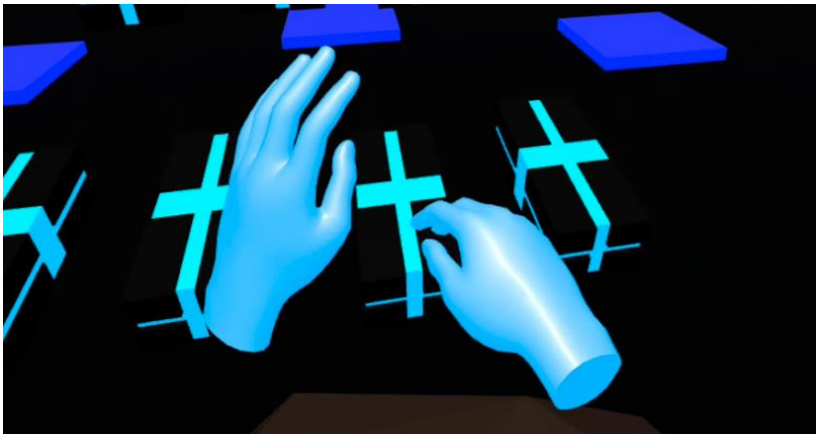


Figura 9.15 Tocar un pad de batería

Para utilizar el theremín, el usuario debe seguir el mensaje mostrado en la parte frontal del pedestal, que indica el gesto que hay que realizar (juntar dedos índice y pulgar).

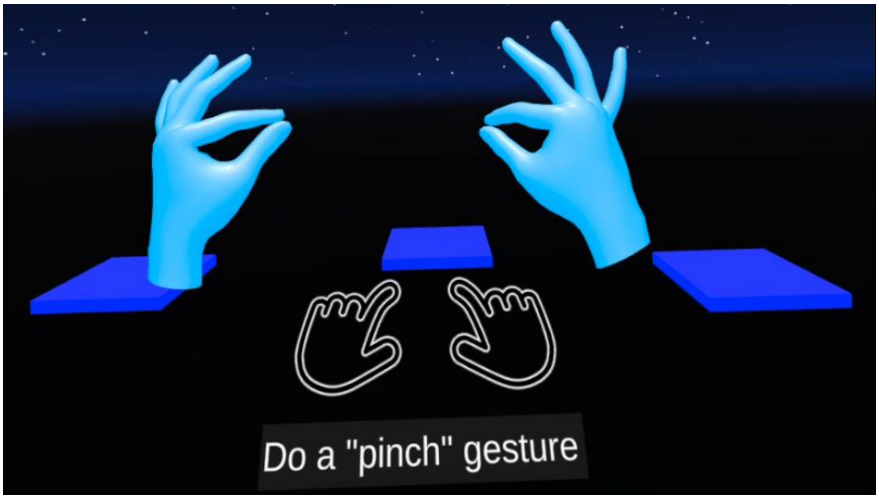


Figura 9.16 Gesto "pinch"

9.3.3 Gestor de pistas

Del mismo modo, para utilizar el gestor de pistas, hay que pulsar los botones correspondientes a cada función que se quiera usar.

- Play/pause/stop: primer botón (cambiará de icono dependiendo de si hay algo en reproducción).
- Renombrar pista: segundo botón (lápiz).
- Eliminar pista: tercer botón (cruz).
- Grabar pista: cuarto botón (botón grande con círculo rojo).

Figura 9.17 Interacción con el gestor de pistas

Para utilizar el teclado que se activa cuando se renombra una pista, la aplicación pierde el foco y el usuario puede interactuar con él a través de un puntero que se lanza desde la mano.
Con este puntero, el usuario apunta hacia la letra deseada y hace el gesto pinch con el dedo índice y pulgar. Para terminar de renombrar la pista, se debe pulsar el botón azul.

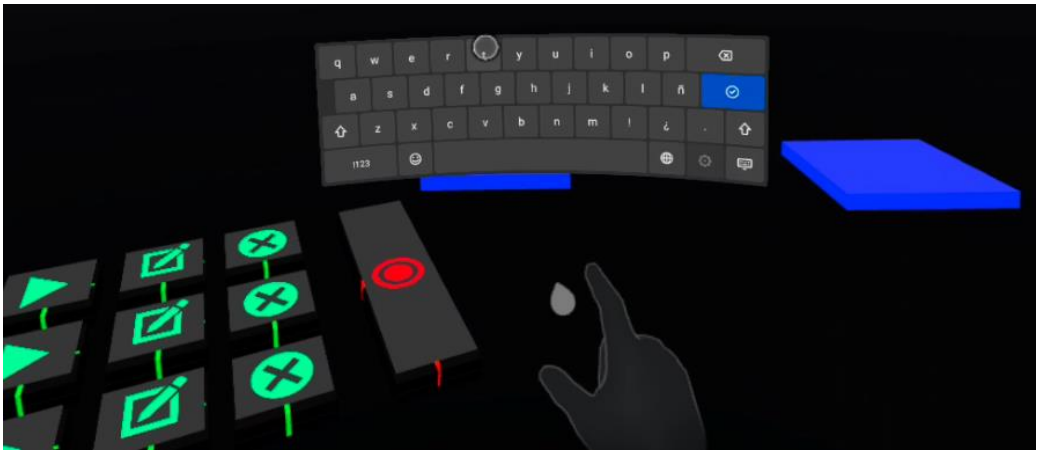


Figura 9.18 Interacción con el teclado

Capítulo 10. Conclusiones y Ampliaciones

10.1 Conclusiones

Por parte del autor, el desarrollo de este proyecto ha sido todo un reto cumplido de manera satisfactoria.

Si bien ya tenía ciertos conocimientos sobre este tipo de aplicaciones, sistemas y entornos, nunca había desarrollado un proyecto de este tipo, que pasó de ser un concepto poco definido al desarrollo de una aplicación de principio a fin, con unos requisitos claros establecidos, sus correspondientes fases de análisis, diseño, desarrollo, pruebas, presupuesto y planificación.

La tecnología Unity ya era conocida por el alumno, lo que facilitó el desarrollo inicial, aunque llegada la hora de implementar todas las interacciones con el sistema Oculus y su hand tracking, se presentaron numerosos obstáculos en el camino que finalmente fueron solventados. Fue necesario investigar sobre muchos conceptos relacionados con la generación de audio por código, las distintas interfaces de Unity para acceder al flujo de datos de audio, así como conceptos más abstractos y de software como la programación orientada a eventos en Unity (utilizada en el reconocimiento gestual y las interacciones con los instrumentos).

Las pruebas realizadas fueron útiles para comprender de primera mano cómo es desarrollar desde cero una aplicación que muchos usuarios podrían utilizar para acceder al mundo de la realidad virtual por primera vez. La oportunidad de probar en usuarios de distintos trasfondos de edad y familiarización con la tecnología proporciona perspectivas interesantes sobre cómo diseñar interfaces para estos sistemas y qué tipo de ayudas diferentes hay que ofrecer a los nuevos y experimentados usuarios.

El resultado final es una aplicación que, si bien es sencilla en concepto, abre muchas posibilidades de ampliación futuras, y que puede ser tomada como inspiración para futuros proyectos por los conocimientos adquiridos y plasmados en esta memoria.

10.2 Ampliaciones

A lo largo del desarrollo del proyecto, se fueron identificando distintas ampliaciones posibles:

10.2.1 Más instrumentos y sonidos disponibles

Una de las funciones a mejorar en el sistema es que la variedad de instrumentos y sonidos es limitada. Por esto, podría expandirse la lista de instrumentos y añadir algunos nuevos que simularan guitarras, instrumentos de viento, arpas… Las posibilidades del hand tracking son interesantes para este tipo de casos de uso.

Por otra parte, se podría expandir también la variedad de sonidos disponibles para cada instrumento. Por ejemplo, en lugar del piano que utiliza funciones del sintetizador, se podría dar la opción de simular un piano de cuerda o uno electrónico. Igualmente, con la batería, se podría dar la opción de utilizar distintos kits (clásica, de los 70, electrónica, jazz…)

10.2.2 Eliminar limitaciones del gestor de pistas

Se podría dedicar más tiempo de desarrollo a solventar los problemas encontrados durante el desarrollo del gestor de pistas. Se podría eliminar la limitación de tener únicamente tres pistas, se podrían reproducir varias a la vez o dar la posibilidad de realizar operaciones mientras se está reproduciendo alguna.

10.2.3 Compartir pistas

Ya que las Oculus Quest poseen conexión a internet, sería de interés dar la opción al usuario de compartir las pistas grabadas con otros usuarios, por ejemplo, utilizando las APIs públicas de redes sociales o de correo que están disponibles para ser utilizadas en Unity.

10.2.4 Sesiones colaborativas

Se podría hacer de la aplicación una plataforma de composición y producción de música colaborativa. Interpretar en directo con otros usuarios sería complicado debido a la latencia que existe con las tecnologías de red actuales, pero se podría plantear como un sistema de composición por turnos en el que los usuarios van añadiendo su interpretación a una pista única y compartida, una vez cada uno. Para integrar esta función colaborativa se podría usar los plugins Photon **[PE01]** o Normcore **[NC01]**.

10.2.5 Publicación en tiendas de aplicaciones

Como extensión de las pruebas de usabilidad, podría ser útil publicar la aplicación en tiendas alternativas que no tengan filtros de aceptación tan estrictos (como SideQuest o App Lab). Esto permitiría recoger feedback de usuarios que el desarrollador no conoce personalmente y, por

tanto, más sinceros y valiosos ya que los usuarios de estas tiendas suelen ser aficionados a conocer y probar aplicaciones conceptuales como esta.

Capítulo 11. Presupuesto

11.1 Presupuesto de costes

11.1.1 Costes directos (desarrollo)

El cálculo de los costes de desarrollo se hará en base al sueldo anual que tiene de media un desarrollador con experiencia en Unity en España. La media de salario es de 26.000 euros brutos anuales **[GD01]** que dan un coste de 12,50€ por hora.

Teniendo en cuenta el cronograma del proyecto, se estima un trabajo que durará aproximadamente 5 meses (24 semanas). Si se estima un desarrollo de 15 horas semanales, a las que se añadirán 3 horas semanales durante las fases de pruebas, se estima una cantidad de 411 horas totales.

Código	Concepto	Cantidad	Precio/Hora	Total
CD1	Estudio inicial	30 h	12,50 €	375,00 €
CD2	Análisis	60 h	12,50 €	750,00 €
CD3	Diseño	60 h	12,50 €	750,00 €
CD4	Implementación	120 h	12,50 €	1.500,00 €
CD5	Pruebas	51 h	12,50 €	637,50 €
CD6	Documentación	90 h	12,50 €	1.125,00 €
		411 h	**TOTAL**	5.137,50 €

Tabla 11.1 Costes directos

Por lo tanto, el coste directo por el desarrollo del proyecto se sitúa en los **5.137,50 €.**

11.1.2 Costes indirectos

Para calcular los costes indirectos del proyecto, se tendrá en cuenta tanto el material utilizado por el alumno para desarrollar el proyecto en su casa como los gastos que esto pueda derivar. Además, si bien las licencias utilizadas son gratuitas por su uso a través de una cuenta de la Universidad, en este presupuesto se tendrá en cuenta el precio oficial de cada una de ellas. El desglose total de costes indirectos se puede ver en la Tabla 11.2.

Código	Concepto	Cantidad	Esperanza de vida (meses)	Amortización	Precio	Total
CI1	Licencia uso Windows 10 Home	1	-	-	149€	149,00 €
CI2	Ordenador workstation	1	60	8,33	1300€	108,29 €
CI3	Tarifa de internet	5	-	-	30€	150,00 €
CI4	Consumo de electricidad	5	-	-	50€	250,00 €
CI5	Oculus Quest 1	1	36	13,89	449€	62,37€
CI6	Suite Office 365 Personal	1	12	41,66	69€	28,74€
CI7	Visual Studio	5	-	-	45€	225,00 €
CI8	Unity Engine	5	-	-	150€	750,00 €
						1.723,40 €

Tabla 11.2 Costes indirectos

11.1.3 Cálculo de beneficios

Se aplica un porcentaje de cálculo de beneficios sobre el coste total de un 20%. Procedemos a calcular el coste total (costes directos + indirectos):

Concepto	Coste
Costes directos	5.137,50 €
Costes indirectos	1.723,40 €
TOTAL	**6.860,90 €**

Tabla 11.3 Costes directos + indirectos

Concepto	Coste
Coste total	6.860,90 €
Porcentaje de beneficios (20%)	**1.715,23 €**

Tabla 11.4 Cálculo de beneficios

11.1.4 Resumen del presupuesto de costes

El coste total del proyecto asciende a **6.752,75€.**

Concepto	Coste
Costes directos	5.137,50 €
Costes indirectos	1.723,40 €
Beneficios	1.715,23 €
TOTAL	**8.576,13 €**

Tabla 11.5 Resumen del presupuesto de costes

El presupuesto de costes solo contaría como presupuesto interno: en caso de querer realizar un presupuesto de cliente habría que atribuirlos a partidas específicas que se facturarán al cliente.

En este caso, se facturarán las partidas de análisis, diseño, implementación y pruebas. El coste de estas partidas se obtiene multiplicando las horas dedicadas a cada una de ellas por el precio/hora establecido.

Para calcular el promedio por partida, se hará en función del coste de cada una, siguiendo la fórmula:

$$Ta = Cp + \left(\frac{Cp}{Isp}\right) * Ip$$

Donde:

- Ta= Total atribuido
- Cp = Coste partida
- Isp = Importe sobre el que promediar
- Ip = Importe a promediar

El coste total que se atribuirá al presupuesto de cliente se observa en la Tabla 11.7.

Concepto	Total	Coste por atribuir	Total atribuido
Análisis	750,00 €	1.018,27 €	1.768,27 €
Diseño	750,00 €	1.018,27 €	1.768,27 €
Implementación	1.500,00 €	2.036,55 €	3.536,55 €
Pruebas	637,50 €	865,53 €	1.503,03 €
		TOTAL	**8.576,13 €**

Tabla 11.6 Partidas a facturar al cliente

11.2 Presupuesto de cliente

El presupuesto de cliente está compuesto por las partidas que así se seleccionaron durante la creación del presupuesto de costes. Estas partidas son las correspondientes al desarrollo en sí de la aplicación (Análisis, Diseño, Implementación y Pruebas):

Código	Concepto	Coste
1	**Análisis**	1.768,27 €
2	**Diseño**	1.768,27 €
3	**Implementación**	3.536,55 €
4	**Pruebas**	1.503,03 €
	TOTAL	**8.576,13 €**

Tabla 11.7 Resumen de presupuesto de cliente

Durante la fase de análisis, se estimaron 411 horas de trabajo. Por lo tanto, se obtiene un precio/hora de trabajo de desarrollo de 20,86€/h.

La tabla 11.7 es la utilizada en el resumen de presupuesto del Apartado 4.2.

Capítulo 12. Referencias Bibliográficas

12.1 Libros y Artículos

[Salomoni16] P. Salomoni, C. Prandi, M. Roccetti, L. Casanova and L. Marchetti, "Assessing the efficacy of a diegetic game interface with Oculus Rift," 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC), 2016, pp. 387-392, doi: 10.1109/CCNC.2016.7444811.

[Salomoni17] Salomoni, P., Prandi, C., Roccetti, M. et al. Diegetic user interfaces for virtual environments with HMDs: a user experience study with oculus rift. J Multimodal User Interfaces 11, 173–184 (2017). https://doi.org/10.1007/s12193-016-0236-5

[Melax13] Stan Melax, Leonid Keselman, and Sterling Orsten. 2013. Dynamics based 3D skeletal hand tracking. In Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D '13). Association for Computing Machinery, New York, NY, USA, 184. https://doi.org/10.1145/2448196.2448232

[Harley20] Harley, Daniel. “Palmer Luckey and the Rise of Contemporary Virtual Reality.” Convergence, vol. 26, no. 5–6, Dec. 2020, pp. 1144–1158, doi:10.1177/1354856519860237.

[Boas13] Boas, Y. A. G. V. "Overview of virtual reality technologies." *Interactive Multimedia Conference*. Vol. 2013. 2013.

[Chattha20] U. A. Chattha, U. I. Janjua, F. Anwar, T. M. Madni, M. F. Cheema and S. I. Janjua, "Motion Sickness in Virtual Reality: An Empirical Evaluation," in IEEE Access, vol. 8, pp. 130486-130499, 2020, doi: 10.1109/ACCESS.2020.3007076.

12.2 Referencias en Internet

[BV01] TheBassValley "Historia del Sintetizador" https://thebassvalley.com/breve-historia-de-los-sintetizadores/ [Últ. vez consultado 8/06/2021]

[IL01] Envelope https://www.image-line.com/fl-studio-learning/fl-studio-online-manual/html/glossary_envelope.htm [Últ. vez consultado 9/06/2021]

[XC16] Xperimenta Cultura. "Historia de la Realidad Virtual" https://xperimentacultura.com/historia-de-la-realidad-virtual/ [Últ. vez consultado 8/06/2021]

[DS18] Deusens. "10 hitos en la historia de la VR" https://deusens.com/es/blog/hitos-historia-realidad-virtual [Últ. vez consultado 8/06/2021]

[Kipersztok19] Kipersztok, J. "Synth VST Essentials: Oscillators, Envelopes, and Filters" https://www.izotope.com/en/learn/soft-synth-essentials-osillators-envelopes-and-filters.html 2019. [Últ. vez consultado 8/06/2021]

[Yoda19] Yoda, N. "Unity Skybox Shaders" https://github.com/n-yoda/unity-skybox-shaders 2019. [Últ. vez consultado 15/06/2021]

[OC01] Unity Profiler Tool. https://developer.oculus.com/documentation/unity/unity-profiler-tool/?locale=es_LA [Últ. vez consultado 21/06/2021]

[OC02] OVR Metrics Tool. https://developer.oculus.com/downloads/package/ovr-metrics-tool/?locale=es_ES [Últ. vez consultado 21/06/2021]

[OC03] Página oficial Meta Quest https://www.oculus.com/experiences/quest/?locale=es_ES [Últ. vez consultado 5/03/2021]

[U01] Oculus XR Plugin https://docs.unity3d.com/Manual/com.unity.xr.oculus.html [Últ. vez consultado 7/03/2021]

[U02] About the Oculus XR Plugin https://docs.unity3d.com/Packages/com.unity.xr.oculus@3.0/manual/index.html [Últ. vez consultado 6/06/2021]

[U03] Página oficial de Unity https://unity.com/es [Últ. vez consultado 9/07/2021]

[UE01] Página oficial de Unreal Engine https://www.unrealengine.com/en-US [Últ. vez consultado 9/07/2021]

[JZ01] JZito, Unity Synthesizer in C# https://github.com/JZito/Unity-Synthesizer-in-C-Sharp

[YT01] Audio Visualization Tutorial (lista de reproducción) https://www.youtube.com/watch?v=5pmoP1ZOoNs&list=PL3POsQzaCw53p2tA6AWf7_AWgplskR0Vo

[W01] Unity (motor de videojuegos). Entrada de Wikipedia. https://es.wikipedia.org/wiki/Unity_(motor_de_videojuego) [Últ. vez consultado 9/07/2021]

[W02] Unreal Engine. Entrada de Wikipedia. https://es.wikipedia.org/wiki/Unreal_Engine [Últ. vez consultado 9/07/2021]

[W03] Oculus VR https://es.wikipedia.org/wiki/Oculus_VR [Últ. vez consultado 9/07/2021]

[PE01] Photon Engine https://www.photonengine.com/ [Últ. vez consultado 9/07/2021]

[NC01] Normcore https://normcore.io/ [Últ. vez consultado 9/07/2021]

[MS01] C# Coding Conventions https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/coding-style/coding-conventions [Últ. vez consultado 10/07/2021]

[XD01] Install ADB on Windows, macOS, and Linux https://www.xda-developers.com/install-adb-windows-macos-linux/ [Últ. vez consultado 10/07/2021]

[SQ01] Página oficial de SideQuest https://sidequestvr.com/ [Últ. vez consultado 10/07/2021]

[GD01] Salario medio de desarrollador Unity https://www.glassdoor.es/Sueldos/unity-developer-sueldo-SRCH_KO0,15.htm [Últ. vez consultado 10/07/2021]

Capítulo 13. Apéndices

13.1 Glosario y Diccionario de Datos

- **Realidad virtual (RV)**: Tecnología que simula entornos reales inmersivos. Es necesario un dispositivo o casco de realidad virtual para acceder a ella.
- **PCVR**: Modalidad de realidad virtual que requiere de conectar el dispositivo a un PC para que haga el procesamiento gráfico
- **RV Standalone**: Modalidad de realidad virtual que no requiere estar conectado a un PC, sino que es un sistema autocontenido.
- **Hand tracking**: Tecnología que permite realizar seguimiento en tiempo real de las manos humanas.
- **Sintetizador:** Instrumento musical que permite generar sonidos modificables en tiempo real.
- **Unity:** Motor de creación y ejecución de videojuegos y aplicaciones 3D.
- **Oculus:** Fabricante de gafas (dispositivos) de realidad virtual
- **Plugin:** Componente añadido a un sistema informático que permite extender sus funcionalidades
- **Motion sickness:** Sensación de mareo y náuseas que puede aparecer al utilizar dispositivos de realidad virtual. Su nombre clínico es cinetosis o mareo por movimiento.

13.2 Contenido adicional entregado

13.2.1 Contenidos

Se entregan como contenido adicional a esta memoria tres directorios:

Directorio	Contenido
/Proyecto	Contiene el código fuente y el proyecto Unity para ser abierto desde el entorno de desarrollo.
/Documentacion	Contiene la documentación Doxygen generada en html y este documento en formato .docx y .pdf.
/Aplicacion	Contiene el archivo .apk para ser instalado en un dispositivo Oculus Quest .

13.2.1.1 Proyecto

Contiene todo lo necesario para abrir y modificar el proyecto Unity en un editor.

Las carpetas principales dentro de este directorio son:

Directorio	Contenido
/Assets	Contiene el código fuente y el proyecto Unity para ser abierto desde el entorno de desarrollo.
/Assets/Audio	Clips de audio utilizados para diversas funciones de la interfaz.
/Assets/Oculus	Plugin de Oculus necesario para desplegar el proyecto en unas gafas de este fabricante.
/Assets/Prefabs	Directorio de prefabs utilizados en el proyecto.
/Assets/Scripts	Directorio que contiene varias subcarpetas donde se guardan los distintos scripts utilizados en el proyecto.
/ProjectSettings	Configuración del proyecto.
./CREDITS.TXT	Archivo de texto que acredita todos los assets, plugins y código externo utilizado a sus autores según la licencia relevante en cada caso.

13.2.1.2 Documentacion

Contiene esta memoria en formatos .docx y .pdf. También incluye un directorio /html:

Directorio	Contenido
/html	Contiene la documentación Doxygen generada automáticamente para una descripción detallada de las clases.

13.2.1.3 Aplicación

Contiene el archivo .apk listo para ser instalado en un dispositivo Oculus Quest.

13.2.2 Código Ejecutable e Instalación

En el directorio "Aplicación" se encuentra SynthVR.apk. Instalar en el dispositivo Oculus siguiendo los pasos descritos en el Manual de Instalación (9.1).

13.3 Índice Alfabético