UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Desarrollo de manos virtuales como NFT para el metaverso

Trabajo Fin de Grado

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

AUTOR/A: Bruixola Martínez, Alba

Tutor/a: Cerdá Boluda, Joaquín

CURSO ACADÉMICO: 2022/2023

Resumen

El proyecto presentado en la siguiente memoria consiste en el desarrollo de un metaverso utilizando técnicas de diseño 3D.

Para llevar a cabo el proyecto se han utilizado el software de código abierto Blender 3D, para el modelado, y Unity 3D, para la programación de la interactividad con la sala. Además, se ha utilizado el Leap Motion Controller, sensor de movimiento capaz de realizar el hand-tracking de unas manos reales y replicar estos movimientos en unas manos digitalizadas. Se plantea el uso del estándar WebXR, para lograr una conectividad multiplataforma, que posibilita el acceso a la sala virtual desde un dispositivo de realidad virtual o un ordenador. Al mismo tiempo, se utiliza el diseño 3D animado para la creación de un filtro de Instagram haciendo uso de la realidad aumentada. Con el fin de rentabilizar el proyecto, se encapsula el producto como un NFT, lo que posibilita su venta.

Con este proyecto se pretende mostrar el potencial de la realidad extendida, en qué consiste y cómo puede acercarse al público en general, que en la actualidad es un gran desconocedor de este campo.

Palabras Clave: Realidad extendida, realidad virtual, realidad aumentada, metaverso, hand-tracking.

Resum

El projecte presentat a la memòria següent consisteix en el desenvolupament d'un metavers utilitzant técniques de disseny 3D.

Per a dur a terme el projecte s'han fet servir el programari de codi obert Blender 3D, per al modelatge, i Unity 3D, per a la programació de la interactivitat amb la sala. A més, s'ha utilitzat el Leap Motion Controller, sensor de moviment capaç de fer el hand-tracking d'unes mans reals i replicar aquests moviments en unes mans digitalitzades. Es planteja l'ús de l'estàndard WebXR per aconseguir una connectivitat multiplataforma que possibilita l'accès a la sala virtual des d'un dispositiu de realitat virtual o un ordinador. Alhora, s'utilitza el disseny 3D animat per a la creació d'un filtre d'Instagram fent ús de la realitat augmentada. Per tal de rendibilitzar el projecte s'encapsula el producte com un NFT, la cual cosa en possibilita la venda.

Amb aquest projecte es pretén mostrar el potencial de la realitat estesa, en què consisteix i com es pot acostar al públic en general, que actualment és un gran desconeixedor d'aquest camp.

Paraules Clau: Realitat estesa, realitat virtual, realitat augmentada, metavers, hand-tracking.

Abstract

The project presented in the following memory consists of developing a metaverse using 3D design techniques.

To accomplish the project, the open code software Blender 3D for modelling and Unity 3D for programming the interactivity to the room has been used. In addition, the Leap Motion Controller, a movement sensor capable of making the hand-tracking of real hands and registrate these movements on digital hands. The use of WebXR standard is proposed in order to achieve multiplatform connectivity, enabling access to the virtual room from a VR device or a computer. At the same time, animated 3D design is used to create an Instagram filter using augmented reality. To make the project profitable, the product is encapsulated as an NFT, which makes it possible to sell.

This project aims to show the potential of extended reality, what it is used for, and how it can be brought closer to the general public, which is currently largely unaware of this field.

Keywords: extended reality, virtual reality, augmented reality, metaverse, hand-tracking.

Agradecimientos

En mi primer lugar, quiero dar las gracias a mi familia, mis padres Maite y Jose, mi hermana Elena y mi pareja Rares; y a mis amigos, los de la infancia y los que me llevo después de esta etapa en la universidad. Por estar a mi lado a lo largo de toda mi carrera y animarme para que siga siempre superándome.

En segundo lugar, quiero agradecer a mi tutor Joaquín Cerdá, por servirme de inspiración y por la gran cantidad de conocimientos que me ha transmitido.

Sin olvidar, a todo el equipo que ha hecho que esto sea posible.

A todos ellos, mil gracias.

Índice

Lista de figuras

Lista de siglas y acrónimos

3D: 3 Dimensiones

UPV: Universidad Politécnica de Valencia

ETSIT: Escuela Técnica Superior de Ingenieros de Telecomunicación

VR: Virtual Reality

AR: Augmented Reality

MR: Mixed Reality

XR: Extended Reality

TFG: Trabajo Final de Grado

NFT: Non- Fungible Token

ETH: Ethereum

APK: Android Application Package

PC: Personal Computer

HDR: High Dynamic Range

OMS: Organización Mundial de la Salud

FBX: FilmBox

SDK: Software Development Kit

PBR: Physically Basic Rendering

I3M: Instituto Imagen Molecular

IDF: Insitituto de Diseño para la Fabricación y Producción

DIMC: Departamento Ingenieria Mecánica y Construcción

GPS: Global Positioning System

Capítulo 1. Introducción

En este trabajo se ha desarrollado un proyecto de Realidad Extendida y se procederá en él a definir los conceptos que engloba la realización de un proyecto de estas características.

Asimismo, se definen conceptos clave y se presentan los distintos tipos de realidades existentes para dar a conocer un campo poco explorado hasta el momento, el cual ha sufrido un fuerte impulso en el año 2020, durante la pandemia de la covid-19; viéndose influenciado por el comunicado de la OMS, en el que recomendó los videojuegos como vía de escape para reducir el estrés que producía el confinamiento y mejorar la salud mental.

Además, pretende servir de guía de inicio para crear una experiencia XR, tecnología que está en pleno auge de desarrollo. Es todo un reto que afrontar, ya que pueden aparecer nuevos problemas de incompatibilidad y configuración. Se explicará el proceso desde la realización del modelo 3D hasta la publicación en un servidor web. Se estudiarán los softwares necesarios para el modelado y programación de este mundo virtual, la creación y venta de los NFTs y de un efecto AR para la aplicación Instagram. Para posteriormente, analizar los resultados obtenidos y valorar si las decisiones tomadas han sido las correctas.

1.1 Motivación

La idea de realizar este Trabajo de Fin de Grado surge durante la realización de mis prácticas en la empresa Metric Salad. En el mes de mayo entra una propuesta para llevar a cabo un proyecto de Realidad Extendida para la marca Ana Locking.

Ana González Rodriguez, es diseñadora de moda y directora creativa de la marca Ana Locking, fundada en Madrid en el año 2008 [1]. Durante su pasó como jurado en el programa “Drag Race España” lució una colección de uñas de su marca. Tras ver la acogida que estas tuvieron en Twitter, quisieron llevarlas más allá. Para lograrlo se propuso digitalizarlas e incluirlas en el Metaverso y así conseguir darles mayor visibilidad.

Debido al plazo de entrega tan apurado de este proyecto, finales de junio, y al hecho de que algunas de sus exigencias no se han hecho antes en el departamento de Realidad Virtual (VR) de la empresa, se decide colaborar con Joaquín Cerdá Boluda, profesor de la ETSIT del equipo de investigación I3M de la UPV y Marta C. Mora Aguilar, profesora del equipo de investigación DIMC en la Universidad Jaume I. Los cuales estaban realizando un trabajo de investigación acerca de hand-tracking y escaneado digital de objetos, respectivamente. También han colaborado en este proyecto los equipos de investigación IDF y Atenea de la UPV.

Personalmente, el campo de estudio de la VR siempre ha llamado mi atención. Es una tecnología que se encuentra en pleno auge, en los últimos años han surgido diversos dispositivos de VR y ha aumentado notablemente el número de empresas y universidades que apuestan por investigar su desarrollo.

Debido a que aún está en desarrollo hay ciertas configuraciones que la tecnología actual no permite realizar. Sin embargo, esta situación no detiene a los desarrolladores de XR, sino todo lo contrario. Se ha creado una comunidad magnífica en la que todo el mundo se ayuda y comparte sus avances. Al fin y al cabo, a todos les interesa que la tecnología avance lo más rápido posible. Por ello, creo que es el momento ideal para ampliar mis conocimientos sobre esta tecnología y las posibles aplicaciones que se le pueden dar.

1.2 Objetivos

El principal objetivo de este trabajo es elaborar una guía de inicio para nuevos desarrolladores; con el propósito de ofrecer la información necesaria, recogida en un mismo documento, para la creación de experiencias XR.

En la realización de este proyecto se han abordado otros objetivos como: la investigación acerca de VR, AR, XR, metaversos y NFT; una propuesta de modelo de negocio con la finalidad de rentabilizar este tipo de proyectos; una solución que acerque la VR al mayor número de personas posible para cambiar la visión que se tiene sobre ésta, demostrando que existen aplicaciones más allá del mundo de los videojuegos.

1.3 Planificación

El desarrollo del TFG cuenta con tres etapas claramente diferenciadas y separadas en el tiempo, ya que se ha compaginado con la realización de otros proyectos laborales y personales.

- Documentación: Marzo – Mayo

Durante los primeros meses en la empresa realicé tareas de investigación sobre las posibilidades actuales que ofrecía la Realidad Virtual, las aplicaciones en los diferentes campos, las nuevas herramientas que hay en el mercado y las que se espera que salgan en los próximos años.

También hice un estudio acerca de las plataformas de Metaverso más exitosas. De este modo, pudimos analizar cuáles son los aspectos del mercado actual que pueden explotarse más, que cosas se podían ofrecer en nuestros servidores y cuales la tecnología actual todavía no permite. En este estudio también se analizaron las diferentes *blockchains* sobre las que trabajan los metaversos. Se analizó la posibilidad de realizar NFTs sobre Open Sea y mostrarlos en nuestro mundo virtual.

Hasta el momento las aplicaciones que se habían realizado en la empresa para gafas VR, eran mediante APK. Así que, se estudió la manera de hacerlo a través de un servidor web, con el uso de WebXR.

- Proyecto: Mayo – Junio

Durante estos dos meses se crea y configura el proyecto 3D, durante los meses anteriores se habían realizado pruebas para asegurar un correcto funcionamiento de un proyecto de esta envergadura. Se realizan las tareas mencionadas en el “Capítulo 3. Metodología”.

- Evento: 4 de Julio

Para publicitar la colección de uñas en el mundo virtual se realizó un evento en Madrid en el Hyatt Centric Hotel de la Gran Vía.

Se invitó a periodistas y diferentes personas influyentes de este campo a asistir a dicho evento.

Este evento consistió en:

Al entrar, mientras se esperaba a ser atendido se podía presenciar un espectáculo audiovisual en colaboración con SoundCool.

Después, se prepararon tres stands en los cuales se podía probar la sala y ver las uñas desde las gafas de VR. Allí había diferentes personas de la empresa para explicar el funcionamiento de las gafas, ya que para muchas personas era la primera vez que las utilizaban, y la finalidad de la sala.

En primer lugar, se explicaba a la gente como podía moverse por la sala. Una vez este paso estaba controlado, se les explicaba la distribución de la sala y se les pedía que se acercarán a las manos y vieran tan de cerca como quisieran las diferentes uñas. Por último, mientras disfrutaban descubriendo la sala se les explicaba brevemente como se había realizado y las oportunidades que el Metaverso nos trae.

Tras haber presenciado esta prueba todos tenían alguna duda ya sea acerca del diseño o algún aspecto técnico, así que iban a otra sala en la cual podían resolver sus dudas y hablar con Ana Locking, diseñadora de las uñas, y Nuria Lloret, CEO de Metric Salad.

- Redacción documento de memoria del presente TFG: Agosto – Actualidad

Se ha redactado el manual con diferentes notas y apuntes tomados a lo largo de la realización del proyecto. Posteriormente, se ha continuado con la investigación con el objetivo de entender ciertos errores de comportamiento de las diferentes plataformas que se han visto durante el desarrollo del proyecto.

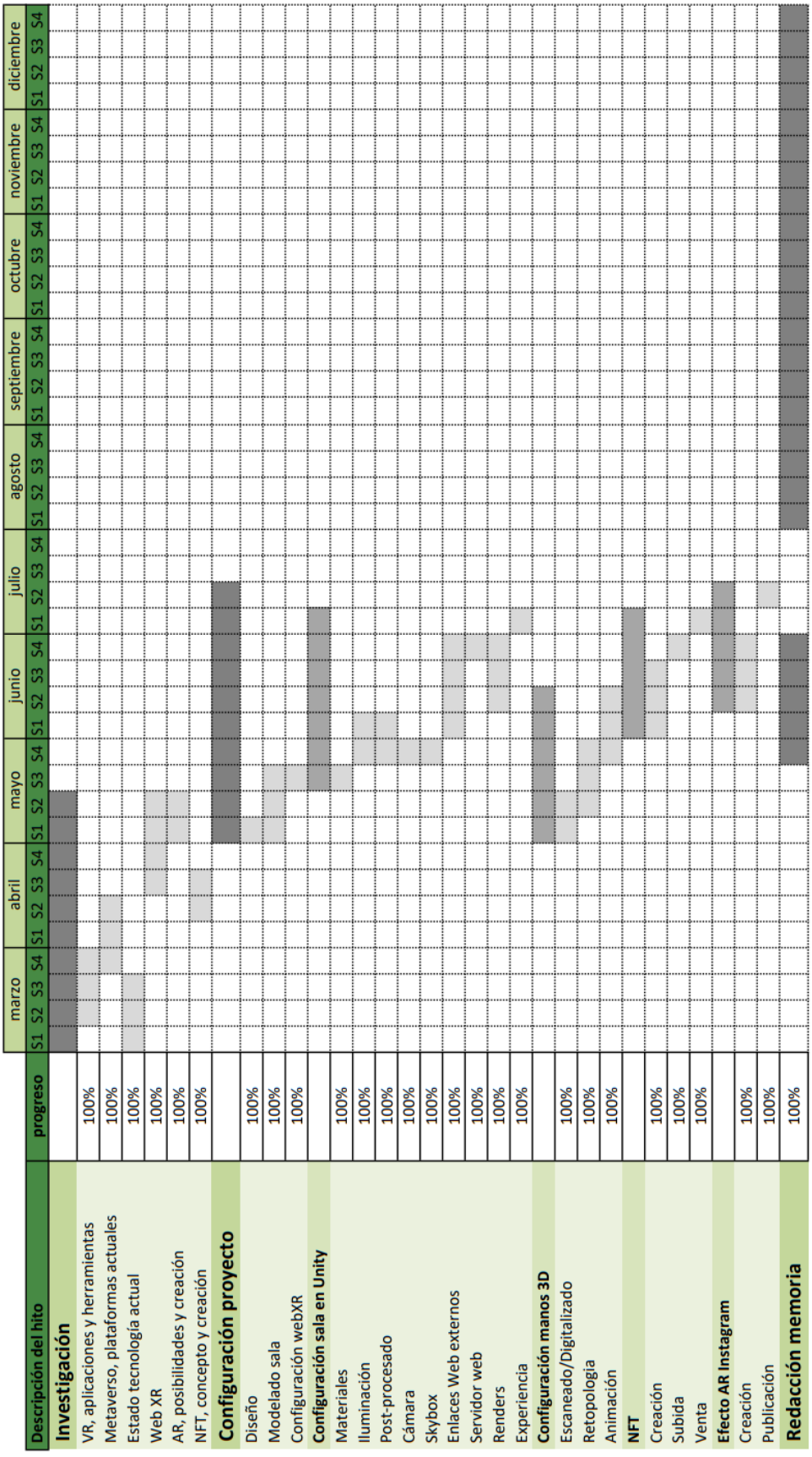

Figura 12. Gráfico Gantt para la Planificación

Capítulo 2. Estudio previo

El primer paso, previo a la realización de este proyecto fue la búsqueda de información sobre el tema, con el fin de ampliar los conocimientos en este campo tecnológico.

Existen ciertos conceptos que es necesario entender antes de comenzar con la explicación del proyecto. En este capítulo se pretende sentar las bases de la Realidad Extendida, así como de las aplicaciones y plataformas que permiten su desarrollo. Se presenta también de forma breve la historia de estas bases, ya que "Aquellos que no pueden recordar el pasado están condenados a repetirlo" (George Santayana, 1905).

2.1 Conocimiento previo necesario

Para poder ofrecer una buena experiencia XR se deben conocer y aplicar los siguientes conceptos:

- **Mundo virtual:** Es el espacio en el que se desarrolla la experiencia XR, incluye el contenido con el cual el usuario interactúa.
- **Interactividad:** En términos informáticos, interactivo, se dice de aquel sistema que permite interacción, a modo de diálogo, entre la máquina y el usuario. (Oxford Languages, Agosto 2022). Para poder ofrecer interactividad se necesita una interfaz, código informático que permite la comunicación usuario-máquina.
- **Inmersión:** Acción de introducir a alguien en un ambiente determinado, en este caso se simula un ambiente tridimensional usando la tecnología que el usuario percibe a través de estímulos sensoriales. Se busca replicar el modo en el que el cerebro se comporta en el mundo real (Slater, 2003)
- **Retroalimentación sensorial:** se produce cuando al realizar una interacción con el mundo virtual, esta ocasiona una acción en respuesta que puede ser percibida por nuestros sentidos. Principalmente de carácter visual o auditivo.

2.2 Hand-Tracking

Tecnología que captura el movimiento de las manos, para el rastreo de manos o manipulación de objetos. Tras décadas de investigación en inteligencia artificial en el procesamiento de imagen, se ha logrado replicar la complejidad y delicadeza de los movimientos de las manos, ya que permite reconocer el posicionamiento y la rotación de los dedos del usuario en tiempo real.

Esto ofrece la oportunidad de una experiencia más realista a los usuarios que lo utilizan y ayuda a la inmersión en la realidad virtual, campo donde se utiliza el Hand-Tracking.

2.3 Realidad Virtual

Es una tecnología que simula una interacción con un mundo virtual en tiempo real, mediante múltiples canales sensoriales del ser humano: la vista, el oído, el tacto, el olfato y el gusto (Burdea, 1993). El objetivo es conseguir sensación de realidad, en el mundo virtual creado mediante un sistema informático, engañando a nuestros sentidos [2]. Para ello se utiliza un motor gráfico capaz de renderizar imágenes hiperrealistas con la información de la geometría, texturas e iluminación en fotogramas 2D. Este proceso se calcula para dos puntos diferentes cada vez, uno para cada ojo, y de este modo, se consigue el efecto de tridimensionalidad. Se necesitan dispositivos potentes ya

que, para que el usuario no note el cambio entre fotogramas, se recomienda que la tasa de fotogramas por segundo sea superior a 72 en todo momento.

El origen de la VR data del siglo XX, con descubrimientos como el simulador de vuelo "Blue Box" o el dispositivo de inmersividad "Sensorama"; pero, no es hasta 2012, cuando Palmer Luckey construye el primer prototipo de gafas VR tal y como las conocemos hoy en día. En 2014, esta patente fue comprada por Meta y se comercializó como Oculus Rift. En la actualidad, hay una amplia gama de productos VR.

La VR tiene múltiples usos, en el ámbito médico, tratando diferentes patologías, ayudando a la movilidad de personas con discapacidades, los estudiantes pueden realizar prácticas sin peligrar la vida de los pacientes; en el ámbito de defensa, en el entrenamiento militar y en las labores humanitarias que realizan hoy en día los ejércitos; en la arquitectura, para mostrar a los clientes los resultados antes de que comience la obra; en la educación, las metodologías mediante las que se interiorizan los conceptos son juegos y actividades; en los deportes, recreando entrenamientos y jugadas; en ingeniería, aplicada a maquetas, proyectos, edificaciones, automóviles, etc.; en los viajes, se puede visitar cualquier lugar del mundo sin salir de casa; en los videojuegos, participando en combates, disparando, corriendo con un fórmula uno, etc.; y en muchos ámbitos más [3].

2.4 Realidad Aumentada

La AR es una tecnología que permite albergar elementos virtuales en el mundo real, visibles a través de la pantalla de algún dispositivo con cámara. Esto se consigue con imágenes superpuestas del entorno virtual en el real, haciendo que el virtual no lo modifique, pero si añada información a éste.

Este término fue acuñado por Boeing Tom Caudell en 1990, que pensó que el uso de unas gafas que mostraran información de las piezas de su fábrica ayudaría a sus trabajadores [4]. Esta idea tiene origen en los avances de la realidad virtual desde la creación del dispositivo "Sensorama", se popularizó con la aparición del juego "Pokemon Go" en el año 2016. Y en la actualidad podemos encontrarla en nuestro día a día.

Los usos de la realidad aumentada son muy variados, en el ámbito de la conducción puede mejorar la eficacia de los dispositivos de navegación o ayudar con diferentes alertas en tiempo real; en el ámbito médico, dando a los profesionales herramientas más rápidas y efectivas para el desempeño de su trabajo; en la enseñanza, posibilita el aprendizaje interactivo, un dibujo puede "cobrar vida" y salir del papel; y muchos más.

2.5 Realidad Mixta

Es una tecnología que combina VR y AR, en la cual se está presente en un entorno virtual y real simultáneamente. De este modo se puede interactuar con objetos reales y virtuales al mismo tiempo, ya que se reconocen mutuamente. Los objetos virtuales reflejan cambios en el entorno como lo haría cualquier objeto real de ese mismo espacio (SSVAR, 2021).

El término se introdujo en un documento de 1994 de Paul Milgram y Fumio Kishino llamado "A Taxonomy of Mixed Reality Visual Displays"; en éste se exploraba la categorización de pantallas visuales [5].

La Realidad Mixta se puede utilizar en el ámbito médico, con la creación de unas gafas que diferencian en tiempo real las células cancerígenas de las sanas; logística, ofrece la posibilidad de agilizar operaciones de embalaje y localización de productos por medio de recorridos guiados por marcadores; en la compra online, aumenta el interés del consumidor mediante una experiencia de compra personalizada; en la industria, en control de calidad mediante la integración con modelos de visión artificial en cualquier punto en la cadena de producción; y muchos más.

2.6 Realidad Extendida

La realidad extendida engloba los tres tipos de realidad vistos anteriormente (VR, AR y MR) en un solo termino; mediante la interacción humano-máquina. Su objetivo es combinar estas para darlas a conocer y aprovechar las ventajas de cada una de ellas.

El término se introdujo también en el documento de 1994 de Paul Milgram y Fumio Kishino llamado "A Taxonomy of Mixed Reality Visual Displays" [5].

En ésta se recogen todos los ejemplos de usos mencionados anteriormente.

2.7 Metaverso

El concepto de metaverso todavía no está completamente definido. Se va fijando, a medida que se crea el propio metaverso.

Lo que sí podemos decir es que el metaverso es un acrónimo compuesto por el prefijo "meta" - que en griego significa "más allá" o "después" - y "verso" - que se refiere al "universo"-. Es decir, un universo más allá de lo que conocemos, haciendo uso de elementos 3D con los que se puede interactuar para generar inmersividad.

Aunque este término se ha popularizado tras la presentación de Mark Zuckerberg, CEO y fundador de Meta, en octubre de 2021, lo cierto es que fue Neal Stephenson, novelista americano, quien utilizó este término por primera vez en su libro de ciencia ficción "Snow Crash" en 1992 para referirse a una versión de internet.

Zuckerberg asegura que estará disponible en 5 o 10 años y se podrá acceder desde dispositivos de VR y AR. Meta está trabajando en la creación de un metaverso abierto, el cual, unirá el resto de plataformas de esta compañía [6].

En este momento, existen gran variedad de empresas que trabajan en el desarrollo de plataformas para el metaverso, alrededor de todo el mundo. Algunos ejemplos son: Roblox, Fornite, Decentraland, Sandbox, Somnium Space, Virtway, Shiba…

Los aspectos en los que se está trabajando y caracterizan el metaverso son:

- Espacios interactivos en los cuales el usuario pueda interactuar con objetos del espacio o con otros usuarios.
- Combina la creación de gemelos digitales con la creación de espacios totalmente únicos que no existen en la realidad.
- Sigue las leyes de la física del mundo real.
- Puede ser centralizado o descentralizado. La diferencia entre estas cualidades se basa en el órgano de control de la plataforma. El metaverso centralizado es dirigido por una única empresa o persona, en cambio, el metaverso descentralizado es controlado por todos los usuarios.

- Incorpora economías virtuales. Puede incluir en el interior de una misma plataforma un mecanismo de compraventa basado en la tecnología blockchain, cómo ocurre con los NFTs.
- Uso de avatares personalizables para conseguir que el usuario se sienta identificado con su alter ego.
- Dispositivos para el acceso. Existen diferentes opciones gafas de VR o AR, así como los dispositivos convencionales como el smartphone, la tablet o el PC. Esto «enturbia las aguas conceptuales» (Girvan, 2018), ya que al hablar del metaverso pensamos en ir más allá de lo que conocemos y con ello lo que viene a nuestra mente es una experiencia inmersiva.

2.8 Problemática del estudio

La tecnología XR está en pleno desarrollo, por ello todavía presenta diferentes problemas con los que hay que lidiar y que en ocasiones nos limitan a la hora de crear una experiencia XR.

A continuación, se mencionan algunos de estos problemas, que surgieron durante la realización del proyecto.

2.8.1 Interacción

"En la industria, los desarrolladores de XR se enfrentan hoy a un desafío: la XR es muy compleja y la interacción con el usuario es casi imprevisible; el nivel de interactividad y realismo de estos sistemas es cada vez mayor, lo que hace que sean muy difíciles y costosos de probar. Normalmente, la evaluación y análisis de estos sistemas se hace de forma manual, con herramientas que sólo funcionan para escenarios de prueba simples. Así, nuestro objetivo es automatizar este análisis, con una herramienta avanzada de testeo que permita responder a estos desafíos y contribuya a garantizar la mejor experiencia de usuario posible", apunta Beatriz Marín, investigadora del instituto VRAIN de la UPV en el artículo "Noticia UPV: Inteligencia Artificial para asegurar la calidad de los juegos, mundos virtuales y simuladores de Realidad Extendida | Universidad Politécnica de Valencia." (s. f.). UPV. http://www.upv.es/noticias-upv/noticia-13552-proyecto-iv4xr-es.html | Proyecto IV4XR del 27/04/22.

2.8.2 Dispositivos

En la actualidad, para poder disfrutar una experiencia XR satisfactoria es necesario que los dispositivos tengan unas altas especificaciones y buena conexión a internet para su correcto funcionamiento. La mayoría de los dispositivos móviles todavía no están preparados para soportar esta tecnología; esto nos limita a la hora de crear la experiencia y todavía debe mejorarse.

Lo ideal es que el proyecto esté lo más optimizado posible para asegurar una experiencia aceptable para todos los usuarios. Esto ha sido algo que se ha tenido en cuenta en la realización de nuestro trabajo y en lo que se ha invertido mucho tiempo y recursos.

2.8.3 Compatibilidad

Hace varios años que Unity soporta diferentes entornos de XR, pero esta solución no es perfecta, aún tiene diversas cosas a mejorar, por ejemplo:

- Para cada dispositivo de VR existía un plugin específico, con lo que la creación de una experiencia VR necesitaba de la creación de un proyecto diferente para cada dispositivo [7].
  Esta situación ha mejorado y ahora hay agrupaciones de dispositivos según si se trata de PC, Android (incluye apk para gafas VR), webGL…
- Desarrollar una aplicación en la que la misma experiencia soporte dispositivos VR y otros que no lo sean era algo realmente costoso y que presentaba muchos problemas [7].
  Aunque el plugin que encontramos, de webXR, te permite realizar esto, sin necesidad de realizar una configuración excesivamente complicada. Pero el coste es prescindir de algunas herramientas que no son soportadas en este tipo de experiencias VR.
- Algunas versiones de plugins pueden ser incompatibles con versiones anteriores de otros plugins o programas con los que funcionaba bien; ya que no todas se adaptan y quizás se necesiten realizar cambios [7]. Por ejemplo, en este proyecto nos ocurrió que MRTK para webXR no funciona con las nuevas versiones de Unity.

2.8.4 Desconocimiento general

Uno de los principales problemas en el campo de la tecnología XR es el desconocimiento general de la gente. La gran mayoría de la población es desconocedora de las posibilidades que ofrecen la VR, AR y XR en nuestro día a día. Este sector de la población presenta diversos prejuicios como, la idea de que pueda producirse daño físico a causa de la desorientación producida por el aislamiento del entorno real; que se desmiente ya que, las gafas VR incorporan diversas cámaras, que activan el sistema guardián cuando se detecta un objeto con riesgo de colisión o que el usuario ha salido de la zona de juego. También existe la opinión de que es perjudicial para la salud mental de los usuarios, ignorando que muchos psicólogos utilizan esta herramienta en terapias para curar trastornos mentales. Generalmente las personas lo asocian al mundo de los videojuegos, y relacionan todas las aplicaciones con ello, la realidad es que ofrece posibilidades en muchos otros campos como se explica anteriormente en el capítulo de investigación. Además, se habla de efectos nocivos en la salud a largo plazo, hecho que no está de momento contrastado ya que la tecnología es reciente; en plataformas abiertas, como el metaverso, no es posible controlar el tipo de contenido al que pueden acceder los usuarios, la realidad es que las plataformas controlan y censuran todo tipo de contenido hiriente o malsonante; problemas de ciberseguridad y desconfianza.

Capítulo 3. Metodología

En este proyecto se ha seguido una metodología de investigación bibliográfica y experimento, en la cual en primer lugar se han estudiado los manuales de los diferentes softwares y hadwares que se pensaban utilizar, así como la situación actual de la tecnología utilizada; y posteriormente se ha realizado la parte práctica del mismo.

Podría decirse que este TFG es un "Manual de Buenas Prácticas" según la experiencia adquirida realizándolo, en el que se explica paso a paso cómo se realiza un proyecto de estas características y los conocimientos necesarios para ello. De este modo, otra persona que esté iniciándose en este campo puede seguir estos consejos para llevar a cabo proyectos 3D.

3.1 Hardware

Se presentan los dispositivos que se han utilizado en el desarrollo de este proyecto. Así como otras posibles opciones, más recientes, que podrían resultar interesantes en este campo.

3.1.1 Leap Motion Controller

Es un dispositivo utilizado para el *hand-tracking*, en concreto, para la obtención de datos sobre el movimiento de manos en tiempo real. Leap Motion Controller hace que la interacción con el contenido digital sea natural y sin esfuerzo [8].

Figura 2. Dispositivo Leap Motion Controller https://www.ultraleap.com/product/leap-motion-controller/

Figura 3. Partes del Leap Motion Controller
https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf

Es un elemento pequeño y ligero que está formado por dos cámaras de infrarrojos de 640x240 píxeles. Esto le permite captar el movimiento de las manos en una zona de 60 cm o más, que se extiende desde el dispositivo en un campo de visión típico de 140x120° [8]. Detecta ambas manos, todos los dedos y hasta el antebrazo del usuario.

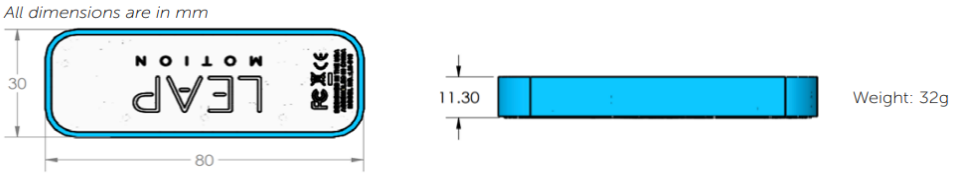

Figura 4. Dimensiones del Leap Motion Controller
https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf

Para ello, incorpora unos LEDs que iluminan las manos mediante luces infrarrojas, que no son perceptibles por el ojo humano. Las cámaras tienen una latencia de movimiento por debajo del umbral humano de percepción [8]. Gracias a esto se consigue un resultado realmente fluido.

Puede distinguir hasta 26 puntos diferentes de cada mano, incluso si están ocultos por otra parte de la mano. Aunque si en algún movimiento muchos puntos se solapan, puede ocasionar problemas. Esto nos ocurrió cuando estábamos grabando el chasquido, por lo que hizo falta probar desde diversas perspectivas.

Figura 5. Leap Motion Controller en funcionamiento https://www.ultraleap.com/product/leap-motion-controller/

La primera versión de este dispositivo salió al mercado en el año 2012, pero no fue hasta el año 2015, cuando salió la primera versión orientada a gafas VR. Con el accesorio *VR Developer Mount* se permite conectar el dispositivo con gafas VR de forma sencilla y segura [8].

Existen diferentes *plugins* para utilizar el Leap Motion en programas como Unity o Unreal.

Existen diferentes aplicaciones: Interfaces públicas sin contacto, experiencias VR/AR, robótica, terapia y salud, educación, simuladores, diseño en arquitectura o ingeniería…

3.1.2 Oculus Quest 2

Es el modelo de gafas VR que se ha usado para desarrollar, probar y hacer la demostración de este proyecto en los stands de la presentación en Madrid. El motivo de esta elección se debe, a que cuando se hizo el estudio de investigación en Metric Salad acerca de la VR, se decidió utilizar estas gafas por su versatilidad y alta compatibilidad con los programas de VR.

Figura 6. Dispositivo Oculus Quest 2 https://uploadvr.com/oculus-quest-2-128-model-on-sale/

Las gafas VR, son un visor de VR con pantallas en el interior, que consiguen engañar a tu cerebro para lograr dar la impresión de estar viendo una escena 3D. Estas gafas vienen acompañadas de dos controladores, que nos permiten interactuar con el espacio de manera más sencilla, sobre todo si el usuario no está familiarizado con el mundo de la realidad virtual. Las gafas incluyen sonido 3D, pero, también tienen un *jack* por si se quiere conectar unos auriculares de mayor calidad que pueden mejorar notablemente la experiencia [9].

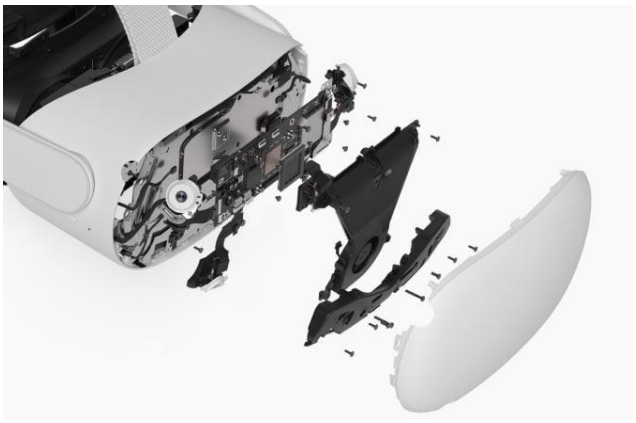

Figura 7. Partes del Oculus Quest 2. https://www.meta.com/es/quest/products/quest-2/

Además, en la parte frontal se incluyen cuatro cámaras de posición. Estás sirven para que el software de las gafas detecte nuestra posición en la sala y nos avisa si nos hemos desplazado mucho y hay peligro de colisión. Esto se controla mediante el sistema guardián que se configura cada vez que encendemos las gafas [9]. Las lentes también tienen diferentes posiciones y se pueden ajustar según la distancia a la que están separados nuestros ojos. Ofrecen una buena resolución gracias a estar formadas por paneles LCD [9]. La correa de sujeción es ajustable, con el fin de que puedan ser utilizadas por personas de todas las edades. Son más ligeras que las versiones anteriores, Oculus Quest, otro aspecto que mejora la experiencia.

Otra de sus ventajas, es que pueden ejecutarse sin necesidad de estar conectadas al ordenador o a la batería, al contrario que el modelo previo, Oculus Rift.

Al iniciarlas nos pide enlazarlas con una cuenta de Facebook, o a una cuenta Oculus enlazada a un correo, ya que estas gafas ahora pertenecen a Meta. Además, Oculus en su web, cuando creas un perfil de desarrollador, te da la opción de compartir pantalla en el ordenador, siempre que las gafas y el ordenador estén conectados a la misma red wifi. También ofrece la posibilidad de hacer esto mismo y suministrar alimentación mientras se hace uso de ellas, utilizando un cable de conexión, Oculus Link, al ordenador.

La primera vez que pruebas unas gafas de VR, puedes sufrir un poco de mareo y desorientación, pero, sólo es cuestión de utilizarlas algunas veces más, para que tu cerebro se acostumbre a la experiencia.

3.1.3 Anet Handy Sense

Es un escáner de mano con cámara 3D, con diseño compacto y ligero, que permite trabajar con un ordenador portátil allí donde se necesites.

Figura 8. Dispositivo Anet Handy Sense https://it3d.com/wp-content/uploads/download-files/manual/es/Manual-de-usuario-HandySense-(es).pdf

Utiliza una tecnología de luz estructurada infrarroja de doble cámara, segura para la piel y los ojos. Con un sistema sencillo de utilizar, en el cual sólo se necesita pulsar un botón. Trabaja a una velocidad de 10 fps. Permite exportar los resultados en ficheros de diferentes formatos: .Obj, .Stl, .3mf, .Asc, .Ply. [10].

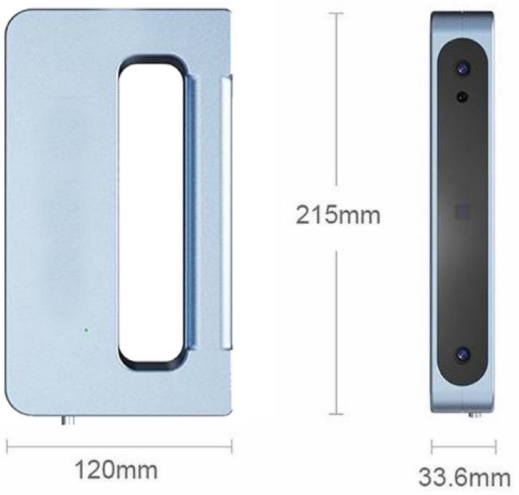

Figura 9. Dimensiones del Anet Handy Sense https://it3d.com/escaneres-3d/portatil/anet-handysense/

3.2 Software

Se presentan de forma breve las aplicaciones que se han utilizado para el desarrollo de este proyecto.

3.2.1 Blender

Blender es un software gratuito y de código abierto; que puede crear visualizaciones 3D, así como imágenes fijas, animaciones 3D y tomas VFX.

Se ha popularizado debido a la combinación de varios factores: un uso intuitivo que ayuda a los principiantes a realizar tareas sencillas como cambiar la posición, el tamaño y la orientación de un objeto; y la amplia complejidad técnica adicional en cuanto a animaciones, mapas UV, sombreadores, mallas y modificadores. Otra gran ventaja que nos ofrece este software es la compatibilidad con otros programas de creación y motores de videojuegos, como Unreal Engine o Unity.

Tiene sus orígenes en 1988 cuando Ton Roosendaal cofundó NeoGeo, estudio de animación 3D, que trabajó en el desarrollo de este software. En 1998, Ton pensó que Blender debía comercializarse para que artistas de todo el mundo pudieran usarlo, para ello fundó NaN, compañía Not a Number. Inicialmente tuvo gran existo y muchos inversores [11].

En 2001, las ventas no fueron las esperadas, ya que la realidad de la época es que el mercado de medios 3D interactivos era reducido. Finalmente, esto ocasionó el cierre de NaN; lo que no frenó a Ton, quien creo una fundación sin ánimo de lucro, “Blender Foundation”, con el objetivo de seguir desarrollando el software y publicarlo como código abierto. Creó la campaña “Free Blender” y en apenas 7 semanas, ya había recaudado el dinero para comprar los derechos de Blender, 100.000€ [11].

3.2.2 Unity

Este software es un motor gráfico de videojuegos multiplataforma.

Cuando se habla de motor gráfico o motor de videojuegos se hace referencia al uso de rutinas de programación para el diseño, creación y funcionamiento de un videojuego. Ofrece infinidad de funcionalidades, de las cuales, nos centraremos principalmente en: escenarios 3D, *renders* gráficos, motor físico que simula colisiones, animaciones, *scripts*...

Existen 4 menús principales, que se deben conocer antes de empezar a trabajar con este software.

- **Project:** Desde el cual se puede acceder y gestionar los elementos del proyecto, Assets. En el lado izquierdo de este menú, se muestra la estructura de carpetas del proyecto. [12] Para incluir archivos desde nuestro ordenador al proyecto, podemos hacerlo arrastrándolos sobre este menú.

- **Vista Escena o de juego (Game):** En la parte central de la pantalla hay un *sandbox*, es decir, un entorno de pruebas interactivo.
  Desde la vista escena pueden seleccionarse y modificarse los elementos del proyecto, ya sean entornos, objetos, iluminación, cámaras o el propio jugador [12].
  Desde la vista de juego no se puede modificar la escena. Es una representación de la visualización del proyecto en el tipo de dispositivo correspondiente, corresponde a un renderizado de la cámara principal [12].

- **Jerarquía:** Menú en el cual se muestran ordenados los elementos de la escena siguiendo un sistema padre-hijo; en el cual un hijo hereda las transformaciones de su padre.

- **Inspector:** Menú en el cual se muestran las propiedades del GameObject seleccionado. En él se pueden ver y modificar todos los componentes y materiales de los que está compuesto el GameObject.
  Si añadimos un componente *script*, las variables públicas del GameObject podrán ser modificadas desde este menú.

3.2.3 Spark AR

Aplicación de Meta creada por Dan Moller, *creative strategist* y AR *specialist* [13]. De uso gratuito, en la cual únicamente es necesario registrarse y conectarlo con una cuenta de Facebook vinculada a una cuenta de Instagram. Este software está destinado a la creación de efectos AR para Instagram, Facebook y Messenger. Los usuarios y empresas pueden crear, publicar y compartir efectos.

Ofrece una gran variedad de opciones para personalizar los diseños. Permite importar objetos 3D, archivos de audio y paquetes de *scripts*; así como utilizar los archivos gratuitos que ofrece en su biblioteca. Además, ofrece la opción Scripting que permite añadir lógica e interactividad, más allá de las opciones predeterminadas, haciendo uso del lenguaje de programación JavaScript. En este proyecto no ha sido necesario, pero se estudiará la posibilidad de utilizarlo en futuros proyectos [13].

De este modo, se puede animar objetos estáticos, crear texturas y materiales, modificar/distorsionar la máscara facial, interactuar con los elementos del efecto… Incluye una función de seguimiento que es capaz de detectar un cuerpo o mano, con rastreo facial de alta fidelidad.

3.2.4 Leap Motion

Software que recibe la información del sensor y se encarga de analizar los datos sobre la posición y movimiento de los brazos, manos y dedos.

Una vez instalado, para comenzar a usarlo es necesario conectar el controlador de Leap Motion mediante la conexión USB de nuestro ordenador.

Contiene un modelo interno de la estructura de una mano que compara con las imágenes que recibe del dispositivo para ofrecer datos lo más fiables posibles, y nos muestra en tiempo real los puntos de referencia que identifica en la mano.

Para hacer uso de este software podemos instalar SDK de algún motor de videojuegos, como es el caso de Unity, o utilizar la aplicación "Airspace" que incluye gran variedad de juegos en los cuales podemos utilizar movimientos o gestos de nuestras manos para interactuar [8].

3.2.5 Open Sea

Es un *marketplace* descentralizado enfocado a la compraventa de activos digitales, los NFTs, basado en la tecnología blockchain. La plataforma se encarga de gestionar los *smart contracts* sobre las redes blockchain como Ethereum, Polygon y Solana, bajo los estándares ERC-721 y ERC-1155 [14].

Fue fundado en el año 2017 por Devin Finzer, como plataforma de mercado de arte digital [14]. Se ha popularizado gracias a su interfaz sencilla y fácil de usar, superando los 500 mil usuarios y los 40 millones de NFTs registrados.

3.2.6 Metamask

Metamask es una criptocartera de Ethereum, para el almacenamiento de tokens del estándar ERC-20 y NFTs. Es un monedero activo, lo que significa que para su funcionamiento necesita ejecutarse en un dispositivo conectado a la red. Permite tener diferentes cuentas, cada una de ellas con una dirección diferente, en una misma criptocartera.

Existen 2 formatos diferentes, aplicación y extensión web. La extensión web, de navegadores como Chrome o Firefox, permite la interacción con *smart contracts* y aplicaciones descentralizadas, de forma fácil y rápida.

3.3 Configuración del proyecto

A continuación, se explica paso a paso el desarrollo del proyecto. Se detalla el proceso de creación y configuración de un mundo vitual, haciendo uso de las herramientas Blender y Unity; la creación y venta de un NFT, en Open Sea; la creación y publicación de un efecto de Instagram, gracias a Spark AR.

3.3.1 Diseño

En esta primera fase se intercambian ideas y se definen los objetivos del proyecto. Para ello, se organiza una reunión con el cliente en la que se realiza un *brainstorming*, definiendo la línea de diseño que se va a seguir.

En este caso, fueron los diseñadores del equipo de Ana Locking, quienes definieron el estilo que se seguiría en el proyecto y en los entregables. Tenían una idea muy clara de lo que querían transmitir con este proyecto y cómo hacerlo. Para transmitir su estrategia de proyecto se utilizaron diferentes medios visuales.

El equipo de desarrollo se encarga de evaluar las limitaciones que el VR supone y cuál es la mejor estrategia para llevar a cabo las ideas del diseño.

3.3.2 Modelado 3D de la sala

Para realizar el modelado de la sala se ha utilizado Blender.

En primer lugar, se realiza el modelo 3D de la sala siguiendo las imágenes de referencia.

Se crea la geometría necesaria utilizando elementos básicos como cubos, media esfera y para la estructura principal de la sala, prisma hexagonal, se utiliza el cilindro y se reduce el número de vértices a 6.

Para las escaleras, utilizando un cubo se crea un único escalón y se multiplica para crear toda la escalera haciendo uso del modificador *Array* con la opción *Fixed Count*. La cual genera un número específico de copias que se define con la variable *Count* [11]. Este modificador crea una matriz de copias del objeto, es decir, se duplica ese mismo escalón un número determinado de veces. Pero, para que se coloquen en la posición esperada simulando una escalera, se debe seleccionar *relative offset* y modificar los valores del *offset* en el vector (x, y, z) para imitar ese desplazamiento.

Tanto con el uso de este modificador, como en el resto de los casos que veremos posteriormente, es importante que en la sección del modificador se seleccione *Apply* para que se aplique en el modelo cuando lo exportemos a Unity. En caso contrario, a pesar de que podremos ver dicha modificación en el proyecto de Blender, ésta no se verá en el *fbx* del mismo.

Para la creación de las salas secundarias en las que se mostrarán las manos con las uñas, se ha utilizado el modificador *Bevel*, que permite crear esquinas redondeadas en la geometría [11]. Esta herramienta, solo se aplica sobre los bordes seleccionados, en este caso, las aristas. Se debe cambiar el apartado *Segments* y aumentarlo para darle el efecto de esquinas redondeadas. Otra forma de realizarlo es utilizar el atajo “ctrl + B” para aplicar el modificador y la rueda del ratón para aumentar/disminuir el número de segmentos.

Para cohesionar estas salas con la sala principal el mejor método es utilizar el modificador *Boolean*, el cual, realiza operaciones booleanas sobre las mallas de dos objetos. Existen tres tipos diferentes de operaciones: Intersección, únicamente los puntos que están en ambas mallas se mantienen; Unión, la malla destino se suma a la malla modificada; y Diferencia, la malla destino se resta a la malla modificada. En este caso utilizamos este último.

Aunque en Blender los planos son *double-sided* en Unity no, lo que implica que los planos solo son visibles desde una de sus caras. Por ello, es importante que se comprueben las normales antes de pasar el proyecto a Unity. Para ello, una de las opciones que ofrece Blender es utilizar el *viewport overlays Face Orientation* que muestra en azul las caras que serán visibles y en rojo las que no. Otra forma de realizar este proceso es activar el *viewport overlays Normals* y elegir la

opción *Faces*, la cual genera segmentos azules que aparecen en cada una de las caras siguiendo la dirección de las normales, como se muestra en la siguiente figura.

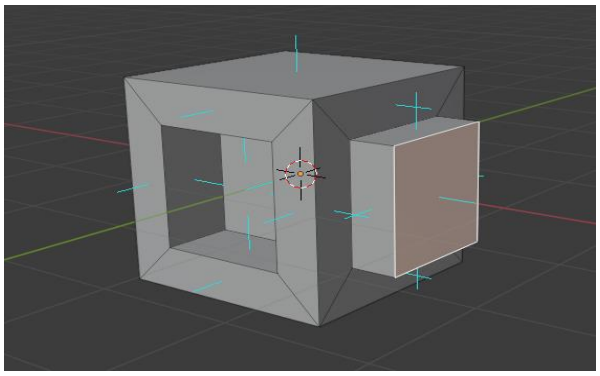

Figura 130. Visualización de normales en un ejemplo de modelo 3D

Si alguna de ellas no es correcta se puede cambiar desde el modo *Edit Mode*, se seleccionan las caras con orientación errónea y se les aplica la modificación desde el menú *Mesh → Normals → Flip*.

Desde Unity, las normales tienen diferentes limitaciones; por ejemplo, nos indican las caras que serán visibles y las que generarán sombras.

En el caso de esta sala, si queremos que las paredes sean visibles desde el interior de la misma y además generen sombras con la luz del sol será necesario darles relieve a las paredes y que estas no sean un plano.

En el centro de la sala se propuso colocar una escultura de un puño que se creó para un desfile de Ana Locking con anterioridad. Era un modelo industrial con alto nivel de detalles, eso implica miles de triángulos en una sola figura. Para poder utilizar este modelo en un espacio de VR, es necesaio optimizarlo. Para ello, se corrigió el modelo a partir del archivo *.fbx* desde Blender usando el modificador *Decimate*. El cual permite reducir el número de polígonos de la malla sin afectar significativamente a su forma. Es útil para objetos de modelado complejo con mucho detalle, esculpidos, digitalizados y/o en los que se han utilizado modificadores de subdivisión [11]. Para reducir el número de caras, se modifica la variable *Ratio* desde “1.00”, cuando la malla no cambia, hasta “0.00”, cuando se eliminan todas las caras.

Para crear la base de la peana se han utilizado diversos modificadores, como el *Boolean* o *Array*. Pero, en este caso, para los elementos decorativos de la base, el *Array* se eligió el tipo *Fit Curve*, con el cual se puede configurar para que se sitúen alrededor de ésta seleccionando como referencia un elemento *Curve → Circle* del tamaño de la peana.

3.3.3 Web XR y MRTK

Utilizamos Web XR, ya que es un formato que combina la posibilidad de ver un mismo proyecto en un servidor web desde el ordenador o desde un dispositivo VR o AR, como pueden ser las gafas de realidad virtual Oculus Quest 2.

Para configurarlo es necesario seguir los siguientes pasos:

En primer lugar, descargamos e instalamos el paquete *Oculus XR plugin* desde *Package Manager → Unity Registry*. El cual, proporciona un soporte de entrada y visualización de los dispositivos Oculus y ofrece compatibilidad con el editor integrado para OpenXR.

A continuación, es necesario seleccionar, tal y como se muestra en la siguiente figura, la opción de Oculus en PC y Android (las gafas se consideran Android) desde *Edit → Project Settings → XR plugin*.

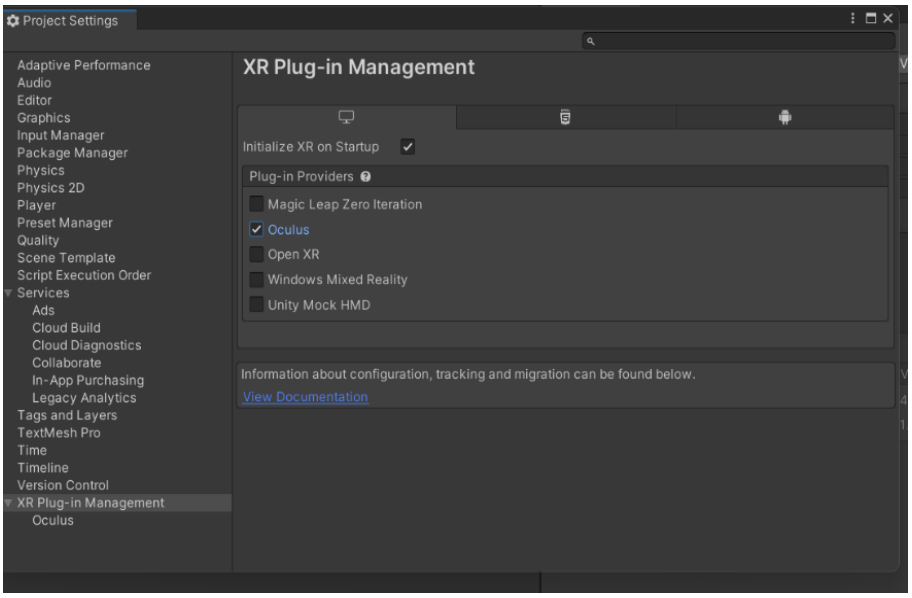

Figura 114. Configuración del XR Plug-in

En cuanto al modo de añadir el MRTK al proyecto se debe descargar la aplicación que ofrece Microsoft y configurarlo con las características de éste que se requieran para nuestros proyectos.

Mixed Reality Toolkit es un proyecto de Microsoft que incluye un conjunto de características y configuraciones con el fin de facilitar y acelerar el desarrollo de aplicaciones VR y acelerar este campo de estudio. Incorpora las interacciones espaciales y con la interfaz de usuario (UI). Otra de sus ventajas es que existen diferentes versiones compatibles con una amplia gama de plataformas [15].

Para importar el MRTK al proyecto se debe instalar la aplicación y tras ejecutarla nos aparece una ventana que nos pide que indiquemos el *path* del proyecto previamente creado en Unity. Al continuar, aparece el menú de características dentro del cual se debe analizar que te ofrece cada uno de los paquetes y que se va a utilizar.

En nuestro caso, las configuraciones y *prefabs* que nos interesan se encuentran en *Foundation* y *Examples*. Instalamos la versión 2.7.3 que probamos en ese momento y era compatible con WebXR, en caso de utilizar una nueva versión se debe comprobar la compatibilidad.

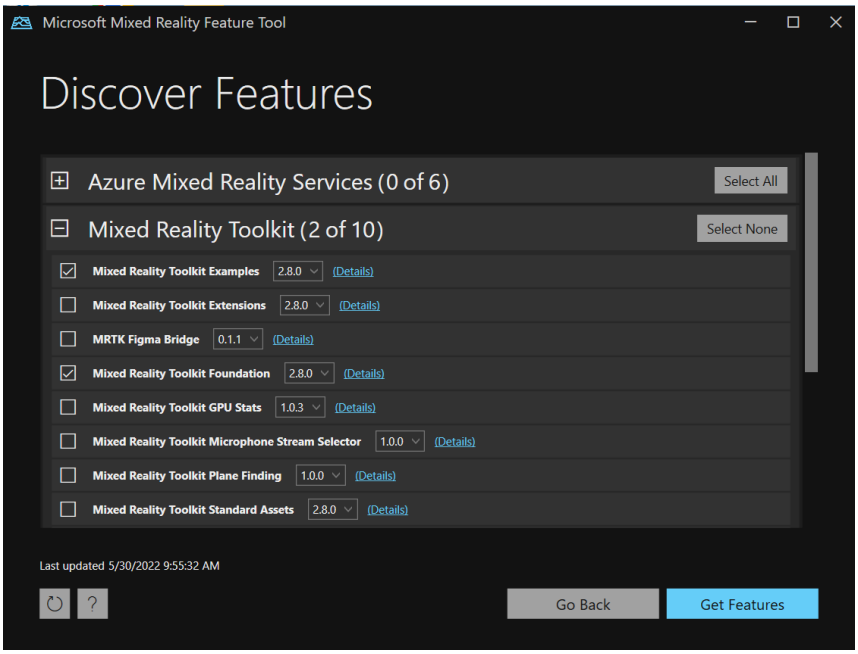

Figura 152. Configuración de MRTK

Además, se debe importar *TMP Essentials* si es la primera vez que añades MRTK al proyecto, ya que algunas de sus configuraciones necesitan acceder a los recursos del *TextMesh Pro*.

Existen diferentes escenas de ejemplo, en este caso elegimos importar la *Demostración - HandTracking* desde Unity utilizando el menú *Windows → Package Manager → In Project → Mixed Reality Toolkit Examples*.

De este modo, se añaden los siguientes componentes a la escena:

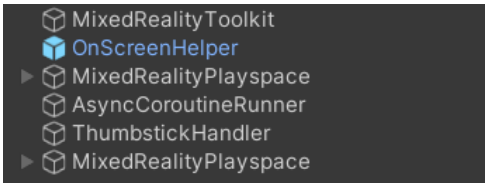

Figura 163. Componentes del MRTK en la escena

De los cuales se modificarán y configurarán los siguientes:

- **MixedRealityToolkit**: Clase encargada de coordinar el funcionamiento del proyecto. Proporciona un registro que gestiona los servicios disponibles del proyecto y un perfil de configuración [15]. Dicho perfil, contiene diferentes elementos que pueden configurarse en función de nuestras necesidades.
- **MixedRealityPlayspace**: Hace referencia a una clase estática que se encarga de definir las transformaciones del juego [15] y en el cual, se incluye la cámara principal con los controles para el movimiento de ésta.
- **OnScreenHelper**: Muestra en la esquina superior izquierda los comandos para interactuar con el espacio desde XR. Pero, para que no moleste a la visión, se ha desactivado.

Existe otro modo de configurar la escena de este modo a partir de un proyecto en blanco, en lugar de utilizar la Demo de plantilla. Para ello, en un nuevo proyecto se crea una nueva escena. Para agregar MRTK a esta escena se va al nuevo menú *Mixed Reality → Toolkit → Add to Scene and Configure*.

A continuación, dentro del *Mixed Reality Toolkit* tenemos las configuraciones del perfil de realidad mixta, que podemos modificar a nuestro antojo en función de las necesidades del proyecto.

Después, instalamos el paquete *webXR.mrtk*, versión de WebXR compatible con MRTK. Este plugin lo encontramos en GitHub y está certificado por los creadores de Unity.

En este caso, únicamente es necesario cambiar el perfil por defecto de la configuración a *SimpleWebXRConfigurationProfile*.

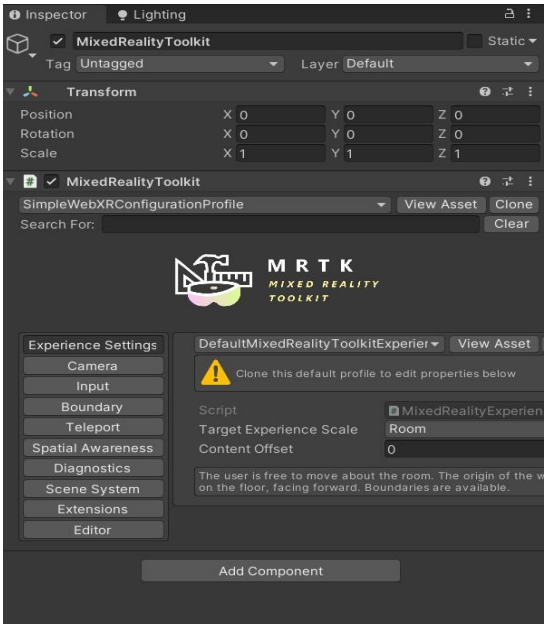

Figura 174. Perfil de MRTK

Por último, cambiamos el *Build Setting* a Web GL desde *Project Setting → Player*; esperamos que cargue la nueva configuración y tenemos el proyecto listo sobre el que se puede empezar a trabajar.

3.3.4 Configuración de la sala

Importamos el *fbx* de la sala modelado en Blender y sobre él, seleccionamos *Unpack prefab*, para poder modificarlo.

3.3.4.1 Materiales

Para este proyecto, hemos elegido añadir los materiales y texturas desde Unity, aunque también pueden añadirse desde Blender y después exportarse a Unity.

Para crear materiales desde Unity se selecciona *Create → Material*.

Se puede editar desde el menú inspector, con el material seleccionado, ajustando la configuración según las necesidades.

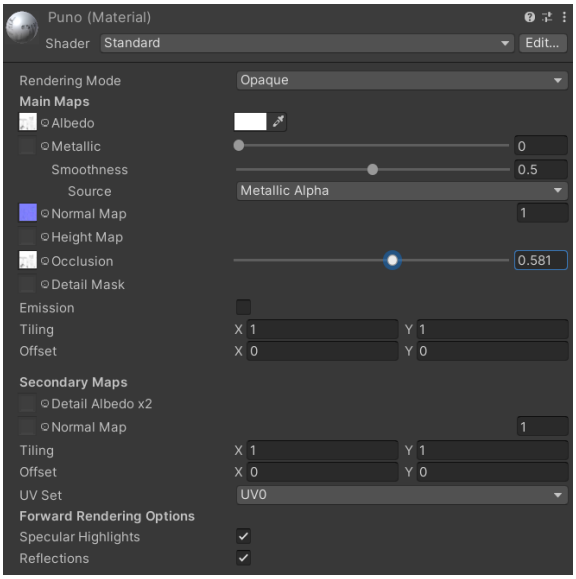

Figura 185. Ejemplo de la configuración de un material

Algunas de las opciones configurables son:

- El modo de *Shader*, dentro del cual existen infinidad de opciones de tipos de materiales.
- El *Rendering Mode*, *Opaque* (opaco), *Transparent* (transparente), *Fade* (desvanecer) o *CutOut* (transparencia recorte, para png).
- El Color Base del objeto, puede elegirse un color desde el propio menú en la paleta de color, o se puede importar una textura y añadirla en el campo *Albedo*.
- El *Metallic*, modifica la reflectividad. Para que pueda reflejar los elementos de la sala, como un espejo, es necesario añadirle un *Reflection Probe*. Se mide en valores de 0 a 1. Puede añadirse un mapa metálico, que indique las partes del objeto sobre las que hará efecto, o en caso contrario, todo el objeto sufrirá el mismo efecto.
- El *Smoothnes*, modifica la suavidad. La forma en la que se distribuyen los rayos de luz sobre la superficie, la difusión. Se mide en valores de 0 a 1. Puede añadirse un mapa de suavidad, que indique las partes del objeto sobre las que hará efecto, o en caso contrario, todo el objeto sufrirá el mismo efecto.
- El *Normal Map*, mapa de relieve. Permite simular pequeños detalles, sin aumentar la geometría del objeto.
- El *Heigth Map*, mapa de paralelaje. Consigue simular protuberancias que parecen sobresalir de objeto. Mejores resultados que el *Normal Map*, pero más costoso en temas de rendimiento.
- El *Occlusion*, mapa que delimita como afecta la luz indirecta al objeto.
- Decidir si el material será Emisivo, y en tal caso, el color y la intensidad de este.
- Configuración de la textura en el objeto: *Tilling*, para la escala, y *Offset*, para la posición. Teniendo en cuenta que esta se ve directamente influenciada por el mapa de UV del objeto sobre el que se aplica.
- Además de las opciones de renderizado: reflexiones o si le afectan los rayos de la iluminación.

Para lograr el efecto de los materiales metálicos, se añade un componente a la escena del tipo *Reflection Probe*.

Su funcionamiento es similar al de una cámara, que captura la visión de los elementos y espacio a su alrededor y la almacena en una esfera [12]. La imagen generada podrá ser usada posteriormente por objetos con materiales reflectantes.

Podemos encontrar este componente en el menú *GameObject → Light → Reflection Probe*.

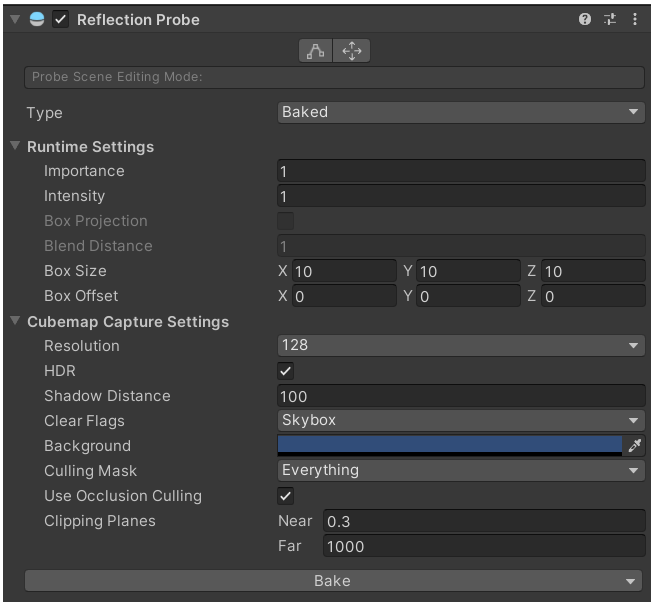

Figura 196. Configuración del Reflection Probe

Este proceso puede ser: *Baked*, *Mixed* o *Realtime*; dependiendo si se quiere que la información se calcule una vez y los cambios no le afecten, o se vaya recalculando todo el tiempo y sea más fiel. Este último requiere mayor capacidad del procesador del dispositivo para estar calculando esta reflexión en tiempo real. Se selecciona el área de efecto del *Reflection Probe*, todos los elementos dentro de este espacio con materiales reflectantes se verán afectados por este.

3.3.4.2 Iluminación

La iluminación de una escena es muy importante al igual que en cine o en fotografía. La luz es el elemento que crea el ambiente. Establecer una combinación de fuentes de luz nos sirve para representar las propiedades naturales, como la luz del día. Nos ayuda a conseguir acabados más realistas, pero a costa de la optimización del proyecto. Esto tiene solución, si realizamos un *Baked* de las luces, éstas estarán optimizadas. Hay que tener en cuenta que, si un objeto cambia de posición en la escena su sombra no lo hará, ya que Unity precalcula toda la ruta de la luz y genera mapas de luces.

Existen diferentes tipos de luces en Unity:

- **Point light:** Es una luz que se ubica en un punto de la escena y emite luz en todas las direcciones por igual. La intensidad de la luz disminuye con la distancia desde este punto, llegando a cero en un rango especificado por el creador de la escena. La intensidad de la luz es inversamente proporcional al cuadrado de la distancia desde la fuente [12].
  Funciona como una bombilla, por ello es el tipo de luz que más hemos utilizado para simular la luz artificial de las salas. Se han colocado cuatro de ellas en la sala central y una en cada sala pequeña de las manos, cambiando el color de las bombillas para que coincida con el color de la sala.

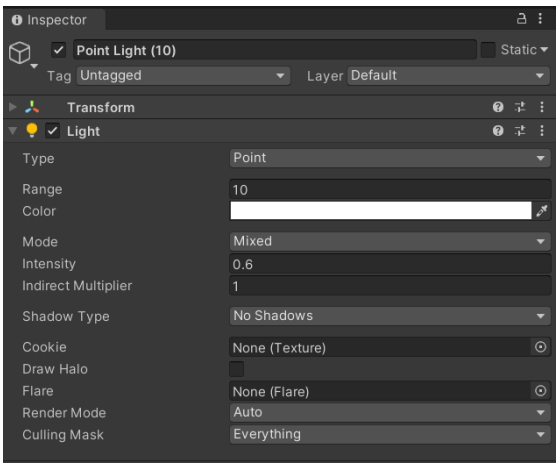

Figura 207. Configuración de la iluminación del Point Light

- **Directional light:** Es una luz que se encuentra extremadamente alejada de la escena, por lo que emite luz en una sola dirección a toda la escena, sin importar donde se coloque, solamente influye su orientación. La distancia de la luz desde el objeto destino no está definida, por lo que no disminuye e ilumina todos los objetos por igual [12].
  Todas las escenas que son creadas en Unity incluyen una luz direccional de forma predeterminada, pero ésta puede ser eliminada o modificada. Funciona como la luz del sol. En general, se coloca una de éstas en la escena. Pero, en este caso, no se ha añadido ya que el skybox que hemos elegido incluía su propia Directional Light animada.

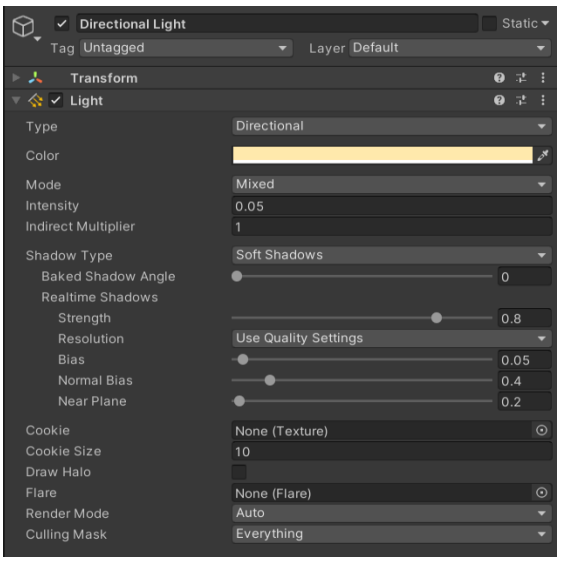

Figura 218. Configuración de la iluminación del Directional Light

- **Spot light:** Es una luz que se encuentra en un punto de la escena y emite en una dirección en forma de cono con un ángulo especificado por el creador de la escena. En este proyecto, la luz también disminuye en los bordes del cono respecto al centro de este [12]. Funciona como un foco. Se ha colocado una de éstas, con luz blanca, sobre cada mano, para una mejor iluminación de las uñas.

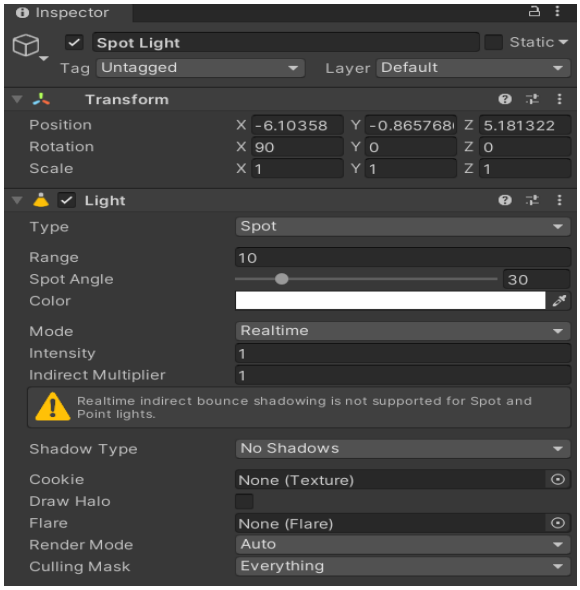

Figura 229. Configuración de la iluminación del Spot Light

- **Area light:** Es una luz que se emite en todas las direcciones uniformemente, a lo largo de la superficie definida por un rectángulo o disco en la escena, pero sólo desde un lado del rectángulo o disco [12]. En este caso no se ha utilizado iluminación de este tipo, debido a que perjudica notablemente la optimización del proyecto. Este tipo de iluminación requiere un gran uso del procesador.

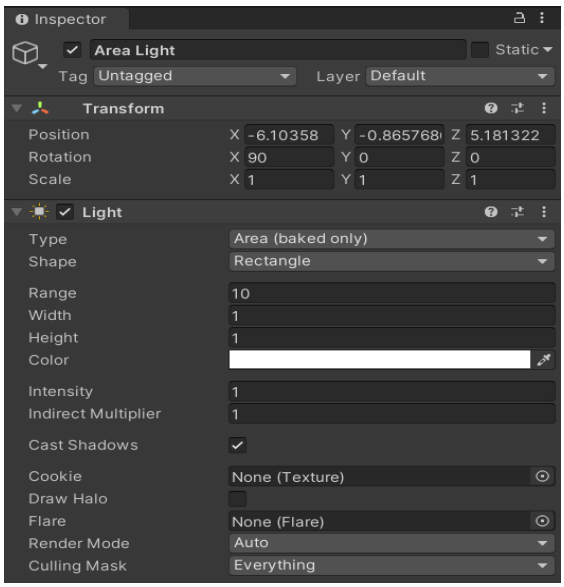

Figura 20. Configuración de la iluminación del Area Light

3.3.4.3 Post-Procesado

Es necesario descargar del Asset Store de Unity el paquete *Post Processing* e instalarlo desde el *Package Manager*.

Una vez se ha importado este paquete al proyecto, desde el menú *Project* se selecciona Create → *Post-Processing Profile.*

Se añade un *Empty GameObject* a la escena, con el componente *Post-Processing Volume* configurado como se muestra en la siguiente figura:

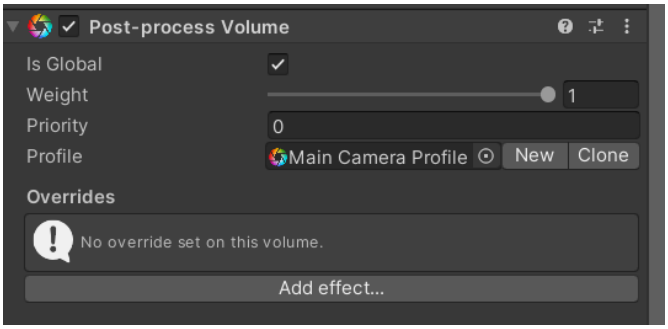

Figura 223. Configuración del Post-Process Volume

Desde este componente se puede añadir efectos de post-procesado para lograr los acabados necesarios. Para lograr el efecto brillo emisivo de los leds, se añade: *Bloom* y se modifica el campo *Intesity*, fuerza del filtro, y *Threshold*, umbral de pixeles bajo el efecto de este brillo.

Desde el *Main Camera* se añade el componente *Post-Processing Layer* de acuerdo con la siguiente configuración:

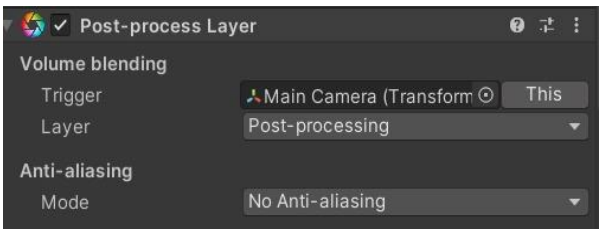

Figura 22. Configuración del Post-process Layer

3.3.4.4 Cámara

Debe haber un componente de cámara y estar habilitado del tipo *MainCamera*, solo lectura, en cada proyecto.

Está es la cámara principal y nos permite movernos por el espacio. Se incluye desde el MRTK, dentro del *PlaySpace*, por lo que no es necesario configurar los comandos para moverse por el espacio, ya viene programado.

Para mover la cámara desde el ordenador se utiliza:

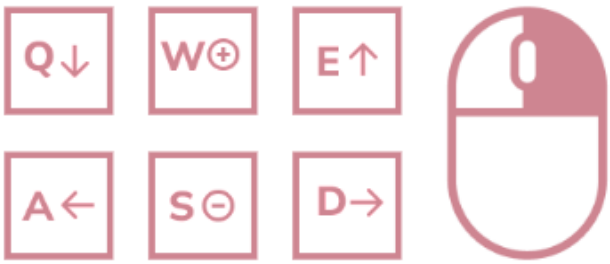

Figura 23. Comandos para mover la cámara

- Con la Tecla Q, la cámara se desplaza hacia abajo.
- Con la Tecla E, la cámara se desplaza hacia arriba.
- Con la Tecla A, la cámara se desplaza hacia la izquierda.
- Con la Tecla D, la cámara se desplaza hacia la derecha.
- Con la Tecla W, la cámara se desplaza hacia delante.
- Con la Tecla S, la cámara se desplaza hacia detrás.

También funciona con las flechas del teclado.

Por último, si se mantiene pulsado el botón derecho del ratón, puede rotarse la cámara a su antojo, moviendo éste.

Desde las gafas se utilizan los mandos para moverse por el espacio.

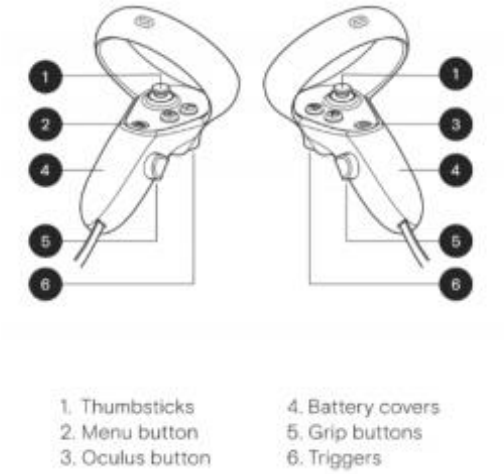

Figura 24. Comandos de los mandos de Oculus Quest 2 https://fisiovr.es/wp-content/uploads/2021/02/MANUAL-DE-USO.pdf

Manteniendo presionados los *joysticks* o *thumbsticks* la cámara se desplaza por el espacio en la dirección en la que el usuario, con las gafas VR puestas, está mirando.

Para interactuar con el entorno y pulsar los botones de las uñas que enlazan con la página de Open Sea para ver los NFTs, se utilizan los *Triggers*.

3.3.4.5 Skybox

Es una envoltura alrededor de la escena, que simula el mundo más allá de su geometría [12]. Puede utilizarse un color o una textura que puede simular un paisaje.

El funcionamiento de este para su renderizado se compone de un *mesh* formado por un cubo en el que se combinan 6 texturas, una por cada una de las caras, de forma que simulen una esfera. Para implementarlo es necesario crear un material e incorporarlo a la escena.

En este caso, el *Skybox* que se ha incorporado es HDR y dinámico, es decir, la iluminación de éste afecta a toda la escena y se modifica automáticamente, dependiendo de la hora, siguiendo el ciclo del sol. Se ha usado un *Skybox* del Asset Store llamado Stylized Sky. Del cual se modificó el *script*, adjuntado en el Anexo B, para que variara únicamente dependiendo de la hora. Se accede a la hora del ordenador al entrar a la sala y automáticamente, se selecciona el color del cielo y la rotación del *Directional Light*.

3.3.4.6 Enlaces con una Web Externa

Unity WebGL tiene un sistema de seguridad que bloquea de forma predeterminada las ventanas/pestañas emergentes del navegador. Sólo se abren si está claro que la acción que activa el evento es directamente responsabilidad del usuario. Si la acción se produce mediante la interacción con un objeto, se procesa dentro de un *Player Loop* y el navegador bloquea la nueva ventana/pestaña [12].

En este proyecto, se quieren añadir unos botones como interfaz, que enlacen nuestra sala virtual con el *marketplace* Open Sea donde se venden los NFTs de las manos. Para solucionar la limitación comentada anteriormente, se instala el plugin *OpenWindows*.jsilb que se muestra en el Anexo A y se modifica el código *NewTab* que acompaña a éste para que cada evento nos lleve a la página indicada. Este script debe añadirse al objeto con el que se va a interactuar.

Después, desde el proyecto de Unity se añade el *prefab* de MRTK *PressableRoundButton*, que es compatible con diferentes instrucciones de entrada del usuario. Esta interacción puede ser por *colliders*, voz, *presionables*…

En nuestro caso, se quiere que al interactuar con el botón, se abra una nueva pestaña con el enlace para conectarlo, sin sacarnos de la experiencia inmersiva. Desde el Inspector tenemos dos formas de lograr el resultado esperado, mediante alguno de los eventos de MRTK (*Begin*, *End*, *Pressed*, *Released*…) usando los *Colliders* o mediante el menú interacción de MRTK. Debido a que debe ser compatible con gafas VR y con el ordenador, se selecciona el evento *OnClick()* del menú *Interactable*, el cual se añade en el *GameObject* del objeto y se añade la función del *NewTab* correspondiente a dicho botón.

El resto de los ajustes pueden dejarse en sus valores predeterminados.

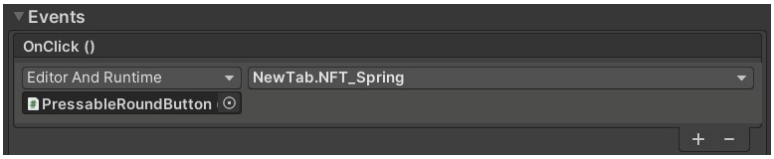

Figura 25. Configuración de la interacción con un botón

Además, se puede observar cómo al acercar el puntero de los controladores al botón presionable aparece un efecto de luz sutil. Son las llamadas luces de proximidad que configura MRTK con el método *ProximityLight.Pulse*.

3.3.5 Configuración de las manos 3D

El objetivo de este proyecto es mostrar las uñas digitalizadas y crear una experiencia XR entorno a ellas y su historia.

Para mostrar las uñas, se digitalizan unas manos 3D. Para conseguirlo, se debe seguir el siguiente proceso.

3.3.5.1 Escaneado/ Digitalizado

En este caso se ha utilizado el dispositivo Escáner 3D Anet HandySense, presentado en el apartado 3.1 Hardware.

Aunque posteriormente a la realización del proyecto se han probado diferentes aplicaciones móviles gratuitas que ofrecen muy buenos resultados. Como es el caso de KIRI Engine o LUMA, que utilizan una tecnología basada en la fotogrametría.

Para realizar el escaneado, se debe conectar el escáner mediante el puerto USB al ordenador y hacer uso del software como visualizador de resultados. Al crear un nuevo proyecto, se debe seleccionar: el nombre del proyecto, el modo de textura, el método de alineación y la resolución.

Se gira el dispositivo lentamente alrededor del objeto, en este caso la mano de los modelos, para tomar datos 360º de dicho objeto. Es conveniente colocar el escáner en diferentes perspectivas para que la información que recoja sea lo más fiable posible.

Desde el software, se recoge una nube de puntos, que se unen en una malla, para formar un modelo 3D. Permite ver si hay partes del modelo en las que falta información o si hay puntos muy distanciados en los cuales se ha escaneado otro elemento accidentalmente. Si alguna de estas situaciones se ha producido, desde el propio software o desde otro editor 3D, se debe limpiar el modelo eliminando los puntos de ruido y rellenando los agujeros.

Tras este proceso, se obtiene un modelo a partir de los diferentes puntos que se han escaneado, los cuales se unen creando una malla de triángulos.

Figura 26. Modelo de la mano Escaneado antes de limpiarlo

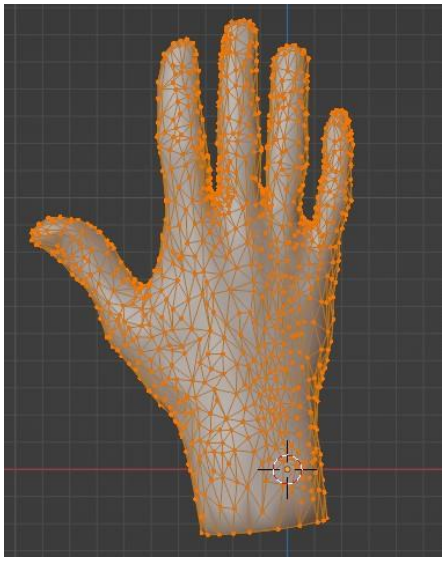

Figura 27. Modelo de la mano tras limpiarlo y aplicarle un modificador Decimate

3.3.5.2 Retopología

Es un proceso digital que consiste en la reconstrucción de la superficie de un modelo 3D a partir de la malla de polígonos que tenemos. El objetivo es reducir el número de polígonos todo lo posible para optimizar el objeto 3D.

Existen 2 tipos:

- **Retopología automática**: será un programa, por ejemplo ZBrush, quien analice la malla y reestructure los polígonos de la manera más eficiente. Este proceso es útil en muchas ocasiones, debido a su elevada rapidez, pero no en nuestro caso. Cuando se desea realizar una animación con dicho objeto 3D, no es recomendable utilizar esta herramienta porque los criterios de retopología pueden no ofrecer la mejor solución para su funcionamiento.

- **Retopología manual:** será el artista quien recoloque los polígonos uno por uno, esto ocasiona que el proceso sea significativamente más lento, a la par que preciso.

En este caso a partir del modelo digitalizado con una malla de miles de triángulos, se quiere obtener una malla de *quads* más optimizada en forma de anillos, *loops*. Para ello, se realiza este proceso de forma manual para los dos escáneres de las manos digitalizadas, presentadas en el apartado anterior.

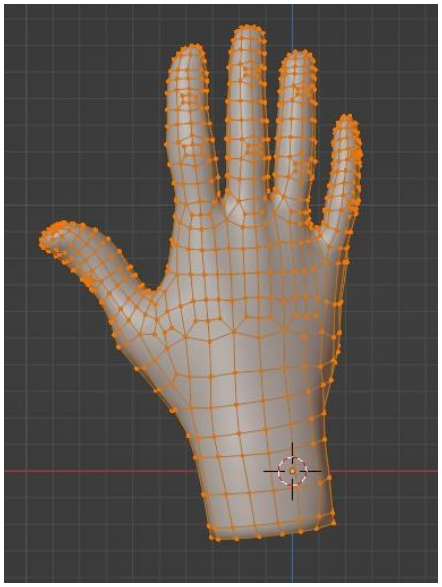

Figura 28. Modelo de la mano tras realizar la retopología

Se prepara el modelo 3D para el proceso de animación, para ello es necesario realizar el *rigging*. Proceso que consiste en crear una estructura de huesos y puntos de movimiento dentro de la geometría, en este caso, de las articulaciones de la mano.

Para formar este esqueleto se puede añadir *Armature* → *Single Bone*. Podemos crear el esqueleto hueso a hueso, si los enlazamos. El elemento *Bone* tiene una estructura de tres partes diferenciadas: *Root* (cabeza), *Body* (cuerpo) y *Tip* (cola). Además, puede modificarse el área de efecto de los mismos, con el fin de que al animar el objeto la malla no se rompa.

3.3.5.3 Animación

En primer lugar, debemos instalar el software de seguimiento de manos de Leap Motion.

En un nuevo proyecto de Unity, con el MRTK instalado y configurado del mismo modo que se explica en el apartado 3.3.3 WebXR, se descarga el SDK de Leap Motion para Unity, herramienta que nos permite conectar aplicaciones de Unity a datos de seguimiento manual [8]. Puede instalarse desde el siguiente enlace: https://developer.leapmotion.com/unity

Existen diferentes opciones para la instalación: a través de OpenUPM o descargando los archivos del paquete de Unity e importándolos en el proyecto.

En nuestro caso se escoge el segundo método; para ello, se configura del siguiente modo:

En primer lugar, se debe instalar el software de seguimiento de manos Ultraleap. En el momento de la realización de este proyecto, la última versión de software es la 5.3.1, pero esta no funcionó correctamente en las pruebas, así que finalmente se ha realizado el proyecto con la versión 5.2.0.

A continuación, se descarga el paquete *Tracking.Unity* con los módulos *Core, Interaction Engine* y *Hands Module,* y se importa en el proyecto de Unity. Se indica que el SDK se diseñó para trabajar bajo Unity 2019.4 LTS o 2020.3 LTS [8], pero se ha probado para la versión 2021.2.11f1 con buenos resultados. Se ha cargado uno de los ejemplos del paquete *Examples* y se ha ejecutado el proyecto con el fin de probar su funcionamiento.

Posteriormente, utilizando el ejemplo *Capsule Hands* como base, se ha modificado para que se adapte a las necesidades del proyecto. Contiene 3 *prefabs*:

- **Service providers:** existen tres tipos, para XR, *Desktop* y *Screentop*. *Desktop* y *Screentop* recurren al mismo componente, *Leap Service Provider,* y sólo cambian el *Edit Time Pose* y la *Tracking Optimization* (dentro de las opciones avanzadas). Por su parte, el Service Provider XR usa el componente Leap XR Service Provider y requiere una referencia a la cámara que se debe usar. En este caso se utiliza el Desktop [8].

- **HandModels**: incluye *Capsule Hand Left* y *Capsule Hand Right,* cada uno de estos con el componente *Capsule Hand* configurado a izquierda y derecha, respectivamente. De éste se debe cambiar el modelo de mano por el creado en Blender. Para este caso en concreto, el modelo se debe exportar con el vector Forward enlazado con el eje Y, y el vector Up enlazado con el eje x.

- **ExampleAssets:** su utilidad está vinculada a la visualización [8]. Incluye un *canvas* con logo y *link* a la documentación, un suelo y el sistema de eventos.

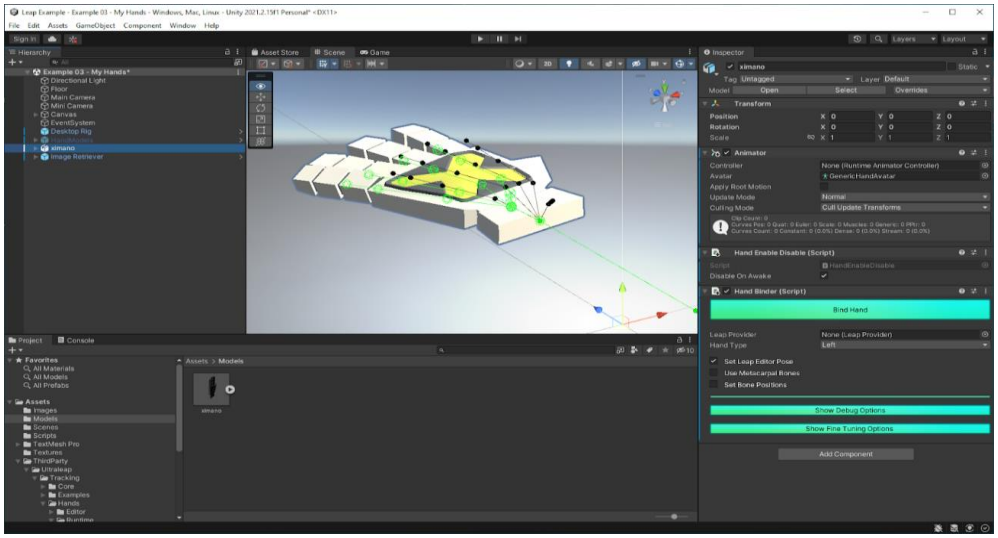

Figura 29. Ejemplo de la configuración de una escena para grabar la animación de manos

El SDK incluye la interfaz entre Unity y el software de *hand-tracking* Ultraleap. Permite la representación física de manos y controles en escenas de realidad virtual incluyendo funciones de agarre, lanzamiento, retroalimentación de colisión y detección de proximidad; y, lo utilizado en este caso, un conjunto de scripts que permite vincular los datos del dispositivo con nuestros modelos 3D [8].

Para poder conectar el Leap Motion con Unity y que desde la escena puedan reconocerse y reproducirse las animaciones, es necesario cambiar la configuración de entrada del menú del *Mixed Reality Toolkit*. Para poder realizar modificaciones es necesario clonar el perfil.

A continuación, se añade un proveedor de datos en el apartado *Input Data Providers* del tipo: *Microsoft.MixedReality.Toolkit.LeapMotion.Input*.

Una vez la configuración de Unity está lista, se importa el modelo. *fbx* de la mano. El cual, incluye un *rigging* (esqueleto) con cada una de las articulaciones que reconoce el software Leap Motion. Para cada estilo de uñas de la diseñadora se han creado modelos 3D de la uña postiza de cada dedo y se han nombrado con números del 1 al 5.

Se debe colocar la uña en el último punto del dedo correspondiente, para que se sitúe en la posición correcta, pero sin intervenir en el proceso de animación. Es un punto de fijación, pero no tiene movimiento de articulación. Es importante que las uñas se importen por separado de la mano y se añadan después, debido a que al vincular el modelo con el software Leap Motion, este compara nuestro modelo con un modelo de una mano estándar que tiene como función predeterminada. Si las manos están unidas al dedo como un único objeto, el software no reconocerá nuestra mano correctamente y se producirán errores en el seguimiento del movimiento de la mano. A continuación, se muestra un ejemplo de la jerarquía que debe seguirse en esta configuración.

Figura 30. Ejemplo de estructura del esqueleto de la mano

Desde el *Inspector* de Unity se deben añadir los siguientes componentes al *GameObject* de la mano:

- **El *script Hand Binder***: Permite enlazar nuestro modelo con el software de Leap Motion. Se debe indicar si el modelo corresponde a la mano derecha o izquierda y el resto del proceso con el cual se vincula cada punto de la mano con el punto correspondiente del modelo puede realizarse de forma manual o automática, al pulsar el botón *Auto Bin*.

- **El *script Hand Enable Disable*:** Sirve para configurar el correcto funcionamiento del *script Hand Binder*, evita alertas por colisiones consigo mismo.

- ***Animator***: Permite reproducir grabaciones de movimiento grabadas previamente. Útil para comprobar si las uñas han atravesado el modelo de la mano en algún momento. Se debe seleccionar la mano del avatar genérica como modelo de referencia.

Figura 31. Configuración en Unity para animar la mano con Leap Motion

Una vez tenemos la escena de Unity configurada y conectada con la aplicación Leap Motion, basta con darle al *play* y desde el modo *Game* podremos ver como el modelo 3D de las manos que se han digitalizado junto a las uñas sigue el movimiento que la cámara del Leap capta en tiempo real de nuestra mano.

Si queremos grabar este movimiento en forma de animación para incluirlo en el proyecto del espacio virtual, se debe instalar el paquete *Unity Recorder* que permite grabar/generar renders. Una vez instalado desde el menú *window → general → recorder → recorder window*, aparece una ventana en la que se puede seleccionar el formato que se quiere generar: *movie*, *image sequence*, *audio*, *fbx* o *clip* animado.

Seleccionamos la opción *.fbx* para, de este modo, exportar el modelo 3D con la animación integrada en el modelo y configuramos la grabadora tal y como se muestra en la siguiente figura:

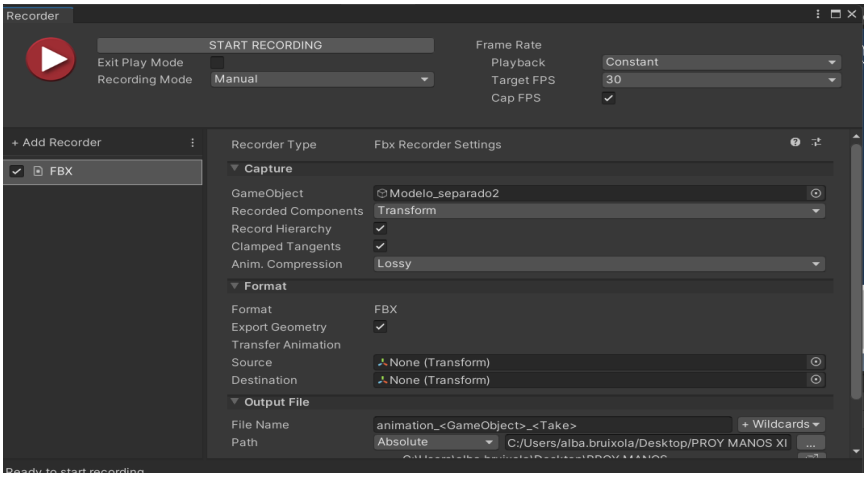

Figura 32. Configuración del Windows Recorder

Se debe indicar que objeto es el que se quiere capturar, de este modo, sólo el GameObject indicado y todo aquello que esté por debajo en su jerarquía se exportarán.

3.3.6 Subir al servidor

Para conseguir alcanzar una cota mayor de público y que sea más accesible, el proyecto se muestra a través de una página web. El hecho de tener que descargarse una aplicación para verlo, reduce significativamente el público dispuesto a consumir el contenido.

El Build que se ha creado desde Unity, se sube a través de File Zilla. Se deben incluir todos los archivos de la carpeta de Build y Template Data, al igual que el Index.

Nombre de archivo	Tamaño...	Tipo de ...	Última modificación	Permisos	Propietar...
..					
standard_index.html	369	Microsof...	28/09/2022 11:36:28	0754	5045 5007
robots.txt	14	Docume...	28/09/2022 10:50:01	0754	5045 5007
index.html	3.368	Microsof...	06/10/2022 12:31:04	0644	5045 5007
favicon.ico	7.358	Icono	28/09/2022 10:50:01	0754	5045 5007
.htaccess	142	Archivo ...	28/09/2022 11:33:36	0644	0 0
videos		Carpeta ...	06/10/2022 9:31:39	0755	5045 5007
TemplateData		Carpeta ...	04/10/2022 11:33:53	0755	5045 5007
stats		Carpeta ...	06/10/2022 0:02:09	0755	5045 5007
error		Carpeta ...	28/09/2022 10:50:01	0755	5045 5007
Build		Carpeta ...	06/10/2022 12:30:11	0755	5045 5007

Figura 33. Ejemplo e la estructura de un servidor web

Para poner enlazar el contenido de Unity con otras páginas web, alojadas en un servidor Apache, se debe añadir un fichero ".htaccess" con las siguientes líneas de código:

```
<IfModule mod_mime.c>
    AddEncoding gzip .unityweb
    AddEncoding br .unityweb
    AddType application/wasm .wasm
</IfModule>
```

3.3.7 Resultados

A continuación, se muestran diferentes imágenes de la sala finalizada. De este modo se puede observar el resultado del proyecto 3D que sea usado de ejemplo en este TFG. Para una vista detallada del proyecto puede visitarse el enlace web del mundo virrual a través de un ordenador accediendo al siguiente enlace web, https://analocking.metricsalad.com/metaverso.html

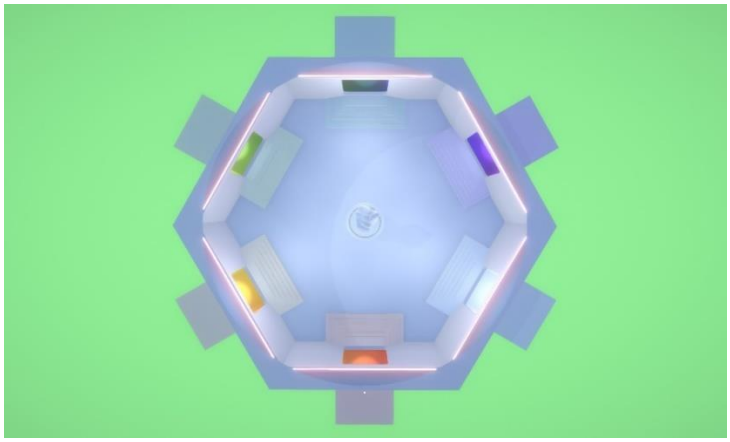

Figura 34. Vista superior de la sala

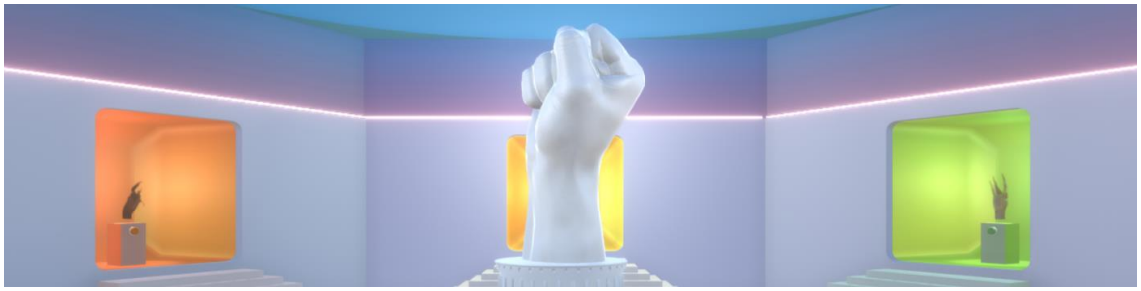

Figura 35. Vista de la sala

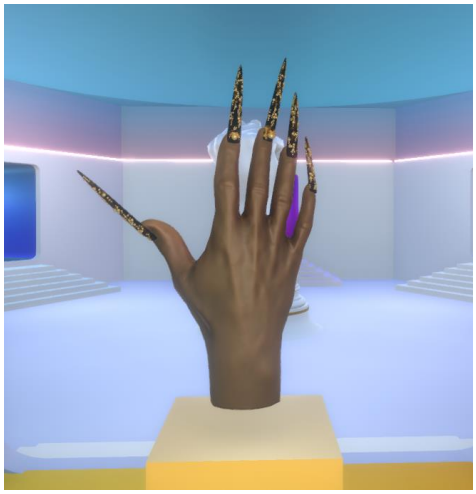

Figura 36. Vista de una de las manos en la sala

3.3.7.1 Renders

El concepto "*render*" se refiere a un tipo de representación gráfica, que puede ser una imagen o un video generado a través de un software especializado. El objetivo es crear un efecto óptico realista para mostrar un concepto o proyecto de forma digital dándole texturas, iluminación y profundidad.

En este proyecto se han creado *renders* tanto desde Unity como desde Blender.

Unity se ha utilizado para los *renders* de la sala o de las manos animadas, mientras que Blender se ha utilizado para obtener mejores imágenes de las uñas para la publicidad e incluso la creación de los NFTs.

Para realizar *renders* desde Unity, se debe instalar el paquete *Unity Recorder* que permite grabar/generar renders. Una vez instalado desde el menú *Window → General → Recorder → Recorder Window*, aparece una ventana en la que se puede seleccionar el formato que se quiere generar; video, secuencia de imagenes, audio, fbx o animación.

En este caso se elige imagen o video que se configurará: la cámara que quiere usarse para la grabación de dicho video/imagen la *Resolution* y el *Aspect Ratio* (p.e 16:9) del *render*, y el formato y calidad del archivo. Además, también pueden elegirse los fps del video.

Si se quiere realizar un video de la perspectiva de la cámara moviéndose por el espacio será necesario generar una cinemática de la misma. Para ello, desde el menú *Animation* se crea un nuevo perfil de animación asignado a la cámara y se seleccionan las propiedades que se quieran modificar de la cámara, es decir, posición y rotación. En él se marcan *Keyframe*, que son los fotogramas claves. Después, Unity recalcula el resto de los fotogramas para generar la secuencia completa.

Para realizar *renders* desde Blender, se debe preparar la escena y configurar la cámara.

Se ha modificado la iluminación para obtener resultados en los cuales las uñas parecen muy realistas y apenas se aprecie diferencia con el modelo real. Se juega con la iluminación utilizando *Area Light* si se quiere iluminar todo el modelo o diferentes *Point Lights* si se quiere simular el efecto que producen las sombras en un objeto real.

Para generar las imágenes o videos de alta calidad en Blender no es necesario instalar ningún *Add-on*. Se configura la cámara con los valores de *Resolution, Aspect Ratio, Frame Rate* (en el caso del video), *File Format, Compression* y *Color Depth*. Además, en el menú de render se escoge la tecnología que se quiera utilizar, *Eevee* o *Cycles*.

Eevee es un motor PBR en tiempo real, que consigue un resultado realista en un periodo de tiempo menor que *Cycles*. *Cycles* consigue calcular el comportamiento de la luz y reflejos físicamente más correctos, mientras que *Eevee* hace uso de técnicas de aproximación no tan exactas, de ese modo reduce significativamente el tiempo de *render* [11].

Con el fin de obtener los resultados deseados se pueden configurar diferentes menús de la cámara de Blender; como es el caso de *Sampling, Ambient Occlusion, Bloom, Depth Of Field, Shadows* y muchos más.

Tanto en Unity como en Blender, la resolución se adapta al canal con el que vayan a ser distribuidos y el formato necesario. Puede ser para prensa, publicaciones para redes sociales (p.e instagram), historias, televisión…

3.3.7.2 Experiencia

Con el fin de acercar a más gente la posibilidad de visualizar la sala desde las gafas VR y como estrategia de publicidad, en el evento que se organizó en Madrid para la presentación del proyecto y del que he hablado en el capítulo 1, se organizó una experiencia inmersiva. En esta experiencia, el usuario utilizaba una de las gafas Oculus Quest 2 que había en los *stands* para visualizar la sala. Para poder mejorar la experiencia, en cada uno de los *stands* había una persona cualificada para guiar al usuario. Para lograrlo el personal necesitaba ver lo que el usuario veía desde las gafas, es decir, se debía transmitir la imagen de las gafas a un ordenador. Para poder compartir la pantalla es importante que el ordenador y las gafas estén conectadas a la misma red wifi. Una vez conectadas, desde las gafas, abrir menú de Oculus pulsando botón del controlador que se muestra en la figura:

Figura 37. Botón del menú Oculus https://liukin.es/como-tomar-una-captura-de-pantalla-en-oculus-quest-2/

Desde el menú de Oculus, seleccionar la app de "Compartir pantalla", flecha blanca sobre fondo rosa, y debe seleccionarse la primera opción llamada "Transmitir". Entonces se muestran los dispositivos disponibles, de entre los cuales se selecciona "Ordenador".

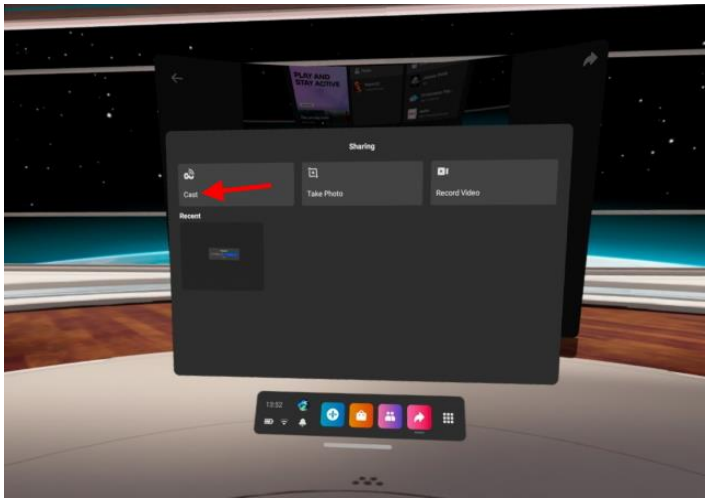

Figura 38. Configuración del menú transmitir https://viralitrix.com/como-transmitir-oculus-quest-2-a-la-tv-2022/

Desde el ordenador: se abre en el navegador la web de Oculus, https://www.oculus.com/casting.

Si se han seguido todos los pasos correctamente, ya se han conectado. En tal caso, aparece un punto rojo en la esquina superior derecha de la pantalla de las gafas y ya será visible desde el ordenador.

3.3.8 NFT

Las siglas en inglés corresponden a *Non-Fungible Token*.

Los *tokens* son unidades que se utilizan en un modelo de negocio y a las que se les asigna un valor [16]. Para poder explicar el término "No Fungible" es necesario primero conocer el significado de "Bien Fungible" [16]. Son bienes a los cuales se les puede dar un valor en función de sus características y pueden intercambiarse por elemento similar de mismo valor. Pero estos bienes se consumen cuando se utilizan, por ejemplo, el dinero. Por el contrario, un "Bien No Fungible" no se consume al ser utilizado ni puede ser sustituido por otro, ya que son únicos; un ejemplo de estos es una pintura o escultura.

El valor de un NFT se basa en el activo digital que garantiza su originalidad, es único e insustituible. Muchos de ellos son en realidad piezas de arte digital acuñadas mediante una tecnología *blockchain*, que desarrollan una trazabilidad segura y fiable con toda la información de su creación y los derechos de autor.

Debido a que los elementos que se suben a la red deben estar estandarizados, los NFTs se pueden transferir rápida y fácilmente entre diferentes aplicaciones.

3.3.8.1 Usos

Tiene diferentes usos. A continuación, se presentan ejemplos que distan del uso principal.

En algunos casos se utiliza como certificado digital de verificación de una obra física, para demostrar su autenticidad. El autor de la obra se lo transfiere al cliente cuando éste compra la pieza física.

A algunos consumidores de arte digital les gusta la posibilidad que los NFT ofrecen de poder mostrar la pieza que adquieres a todo el mundo, mientras que si compras una pieza de arte física y la cuelgas en tu salón solo podrás mostrársela a aquellos que invites a tu casa.

3.3.8.2 Funcionamiento

Su funcionamiento se define bajo la red de tecnología *blockchain*, al igual que las criptomonedas.

Para cada NFT se crea un *smart contract*, certificado digital de autenticidad, que incluye un conjunto de metadatos para garantizarlo. En él se registran el valor de partida, la información de las adquisiciones o transacciones, así como la información del autor. De este modo, se registra el valor que se le adjudica al NFT en cada una de las transacciones y las direcciones de cada uno de los propietarios.

3.3.8.3 Creación

Para el proceso de creación de un NFT en Open Sea se debe conectar una *Wallet*, en este caso se ha elegido MetaMask por ser la opción más popular.

Se debe crear una cuenta de MetaMask e instalar la extensión de la billetera en el navegador web desde https://metamask.io/. Este *plugin* facilita la interacción con aplicaciones de la *blockchain* de Ethereum y permite almacenar artículos digitales.

Metamask incorpora un alto nivel de seguridad; al crear la cuenta, se genera una frase semilla de 12 palabras y una contraseña. Cuando se inicie sesión en un nuevo dispositivo será necesario introducir la frase semilla y posteriormente, en cada proceso que se haga y para acceder a la cuenta de Open Sea, será necesario introducir la contraseña y firmar.

Para, posteriormente, poder subir el modelo de las manos digitalizadas como un NFT, se deben tener en cuenta diferentes ajustes durante el proceso de creación.

En primer lugar, se debe concretar el formato del NFT, puede ser una imagen, vídeo, audio o un modelo 3D. En este caso se apuesta por el modelo 3D, que permite ofrecer una experiencia más interactiva que un simple vídeo. Para ello, es necesario exportar el modelo en .glb o .gltf, formatos 3D interactivos que incluyen animaciones.

Esto nos permite interactuar con el modelo 3D desde el visualizador de OpenSea, que funciona como un motor gráfico de render. Y es aquí donde hay que tener en cuenta las limitaciones de este motor gráfico. Tras investigar acerca del tema, se realiza una lista con las características más importantes que Open Sea no permite importar:

- Efectos/Ajustes de la cámara
- Fondo de la escena
- Luces (solo permite exportar las *Point light*)
- Partículas
- Deformaciones de malla
- Nodos
- Para los formatos elegidos sólo pueden usarse los mapas: *Base Color*/*Diffuse*, *Metallic*, *Roughness*, *Normal map*, *Emissive*, *Alpha*

Una vez se conocen las limitaciones del NFT, estas son las directrices que se deben seguir en la creación del modelo 3D:

- Corregir las normales
- Mantener la cantidad de polígonos optimizada
- Se deben aplicar los modificadores, sino no aparecerán en el render final
- Aplicar modificaciones de posición, rotación y escala
- Corregir UVs, para que no se superpongan, haciendo uso del Smart UV Project
- Reducir el peso de las texturas que se importan al modelo
- Se permiten 3 tipos de animaciones:
    - Keyframe: se añaden indicadores de posición, rotación y escala a lo largo del tiempo, marcando algunos *frames* de referencia y dejando que Blender calcule el resto.
    - Bones: Se crea el esqueleto o *riggy body* y se anima simulando la movilidad física del individuo.
    - ShapeKeys: modificaciones de la malla *bakeadas*, es decir, ya calculadas y precargadas.

3.3.8.4 Subida a Open Sea

Para iniciar el trámite de la subida de un NFT en la página de Open Sea, se selecciona la opción "*Create*" o "Crear".

En primer lugar se debe tener claro el tipo de fichero que se convertirá en NFT, los formatos soportados son .jpg, .png, .gif, .svg, .mp4, .webm, .mp3, .wav, .ogg, .glb y .gltf con un peso máximo de 100 MB.

Con el fin de añadir la información necesaria para el comprador, existen los siguientes campos a rellenar:

- **Archivo**: se sube el fichero que se convertirá a NFT. Si es un archivo 3D también nos pedirá una imagen para previsualizar el NFT desde el menú.
- **Nombre**: para identificar el NFT.
- **Enlace externo**: url de una web que pueda aportar más información y/o fiabilidad acerca del NFT y del creador.
- **Descripción**: breve reseña del NFT.
- **Colección**: se selecciona a cuál pertenece, debe crearse con anterioridad.
- **Propiedades**: en caso de crear una colección en la cual los NFTs tendrán diferentes características pueden regularse éstas, modificando las cualidades de niveles, estatus…
- **Contenido desbloqueable**: En este campo puede añadirse contenido que solo podrá ver el propietario del NFT.
- **Número de copias del NFT**
- **Red Blockchain**: red sobre la que se crea el NFT. Ethereum es la más popular, pero en este caso se utiliza Polygon.
- **Congelar los datos**: si se activa esta opción, el contenido del NFT "se congela" bloqueando cualquier cambio futuro; de este modo se verifica que el creador no podrá realizar modificaciones en la pieza. Es importante asegurarse de que toda la información y contenido es correcto antes de realizar este paso, ya que es irreversible.

Al crear una colección se rellena la siguiente información:

- **Imagen, imagen de fondo e imagen de banner**.
- **Url**: nombre del enlace que nos permitirá localizar la colección. Debe empezar por: https://opensea.io/collection/ y se rellena, generalmente, con el nombre de la colección a continuación.
- **Descripción**: breve reseña.
- **Categoría**: menú desplegable en cuál puede elegirse una de las opciones entre arte, deportes, coleccionables, utilidades…
- ***Urls* de diferentes sitios web**: youtube, discord, twitter, instagram, telegram…
- ***Royalties***: tasas de creador para todos los NFTs de la colección. Esta tasa indica el porcentaje del precio de venta que recibirá el creador en cada una de las ventas del NFT entre usuarios. El valor que puede recibir esta tasa oscila entre 0 y 10%. En caso de querer añadir un valor se recomienda que este no sobrepase un 2,5%.
- Dirección de la criptocartera en la que se ingresarán los beneficios. Dirección hexadecimal que aparece en la extensión de nuestra criptocartera.
- **Red blockchain** sobre la que se realizarán las transacciones de los NFTs de dicha colección

Si queremos observar el resultado final antes de publicarlo y que sea visible, se puede utilizar el visor de Windows o https://testnets.opensea.io/es, un visor de prueba que ofrece OpenSea.

3.3.8.5 Venta

Al entrar en el menú del NFT, se selecciona la opción "*SELL*" o "Vender". Dependiendo de la red *Blockchain* se pueden establecer diferentes parámetros, en este caso, al utilizar Polygon, a fecha junio de 2022, solo nos permite escoger:

- Precio de venta en ETH
- Duración de la oferta en el mercado, permitiendo un máximo de 6 meses.

En ese mismo menú, puede verse la comisión por el servicio de OpenSea de un 2,5%, ya que es la plataforma, quien se encarga de crear los *smart contracts*. Esta tasa corre a cargo del comprador.

En el caso de trabajar con Ethereum o con Polygon, a partir de septiembre de 2022, la plataforma ofrece tres tipos diferentes de venta:

- ***Set Price* (Venta Directa)**: esta opción es la que se ha comentado anteriormente, se debe fijar el precio final de venta. Puede programarse una fecha y hora futura a partir de la cual el *token* se listará y estará disponible en el mercado para su compra. Pero este será visible desde el momento en el que se sube. Si, por el contrario, nos interesa mantener el token en privado y que solo aquellos que tengan el enlace de la venta puedan acceder a ellos, se les debe seleccionar como Privados.

- ***Highest Bid* (Subasta):** se establece un precio mínimo de venta y una fecha de expiración de la subasta. La subasta puede cerrarse por dos motivos diferentes: el periodo de venta ha expirado y quien haya apostado más se queda el NFT, o porque se ha superado el precio de reserva y el NFT se vende automáticamente.

- ***Bundle* (Bulto)**: en este caso, se realiza una venta agrupada, de dos o más NFTs, de forma conjunta.

También puede reservarse para un comprador en concreto, indicando la dirección de la criptocartera del comprador.

3.3.9 Efecto AR de Instagram

Para la parte de Realidad Aumentada del proyecto se utiliza Spark AR Studio.

Para este trabajo creamos un nuevo proyecto desde cero, aunque Spark AR ofrece algunas plantillas dependiendo de la funcionalidad que quiera dársele al filtro. Se ha preferido añadir únicamente los componentes que vayamos a utilizar, para optimizar al máximo el proyecto.

3.3.9.1 Seguimiento del rostro

Para conseguir detectar el movimiento del rostro del usuario que utiliza el filtro y poder hacer que el objeto 3D que se añada siga dicho movimiento.

En consecuencia, se selecciona *Add Object* y se añade un *Face Mesh* (Malla facial) que es un modelo en 3D de una cara. Se selecciona la malla facial y en el Inspector debe pulsarse el "+" que

aparece junto a la opción Materiales, para añadir un nuevo material. En la configuración de este nuevo material se escoge la opción *Retouching* en el *Shader type*.

Es una opción predeterminada que incluye la *app* en la que se utiliza un *Skin Smoothing*, se regula para suavizar la textura de la cara y reducir imperfecciones. Se aplica un 30% únicamente para que quede natural.

Para lograr que el seguimiento facial funcione, se debe añadir el componente *Face Tracker* (Seguimiento facial) que al combinarse con el *Face mesh* permite que se puedan seguir los movimientos de la cara en el espacio y las expresiones faciales. Se añade desde el panel de *Assets* (Recursos) y aparece un eje en el centro del rostro que nos permite observar cómo se sigue el movimiento de éste. Se selecciona el *Face mesh* añadido anteriormente como cara a seguir.

Por ende, se sigue la siguiente estructura:

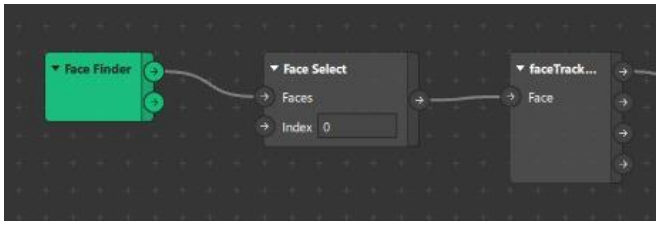

Figura 39. Configuración del seguimiento del rostro

En primer lugar, se añade un *Face Finder* (Buscador de caras) que indica las diferentes caras que la cámara encuentra. A continuación, se enlaza con el *Face Select* (Selección de cara) para indicar que cara se selecciona de todas las que nos indica el nodo anterior y que tenemos a su salida. Finalmente, localiza y sigue el movimiento de la cara seleccionada anteriormente. A su salida representa la posición con coordenadas 3D, rotación y tamaño.

3.3.9.2 Elemento AR

Podemos añadir el objeto 3D que se ha modelado en Blender, en este caso la mano con las uñas. El objeto debe exportarse en formato .fbx desde Blender y marcando la casilla de incluir la animación.

Ante nuestra idea inicial surge un problema, que nos lleva a cambiar el plan establecido; el peso del objeto, debido a su elevado nivel de detalle, impide que se puedan incluir diferentes manos o uñas en un mismo filtro, ya que se sobrepasarían las limitaciones que nos impone Spark AR. El filtro, para usarse en Instagram, tiene un límite de peso de 4MB.

Así que la idea inicial de crear un solo filtro, en el que se incluyeran todas las uñas y las versiones masculina y femenina de la mano, con sus diferentes texturas, tuvo que ser descartada. Finalmente, se optó por crear un filtro para cada mano de las representadas en la sala (cada uña con la combinación de mano masculina o femenina de la sala) e incluir en esta todas las texturas de la piel, con el fin de que puedan usarlo personas de diferentes razas.

3.3.9.3 Cambio de textura

En este caso, queremos que el usuario pueda interactuar con el efecto y pueda elegir el color de la piel de las manos del filtro que mejor se adapte a su tono de piel o que prefiera, a su elección. Para ello, añadimos unos botones en la parte inferior de la pantalla, de manera que cuando el usuario los clique cambie el color de la textura de las manos. Dichos botones sólo son visibles durante la grabación del efecto, pero no en el video final.

Para configurarlo se deben añadir y configurar los siguientes nodos en el "*Path editor*" (Editor de nodos).

- **Picker UI**

Es un nodo de interfaz de usuario que nos permite añadir hasta 10 texturas de nuestra elección y mostrarlas en los iconos de los botones en la pantalla del dispositivo del usuario [13]. Cuando el usuario presiona alguno de estos botones se actualiza el nodo y cambia el elemento que se muestra a la salida de dicho nodo enlazándolo con la entrada seleccionada en este caso.

Es muy importante que las texturas que se incluyan en el nodo para ser usadas en los iconos de los botones estén en modo "*Non compression*" (descomprimidas), para que pueda leerlas correctamente, sino dará error. Para ello se debe configurar del siguiente modo:

En el Inspector, dentro del panel de recursos, se selecciona "Textura", dentro de esa opción, el menú de "*Compression*" y en el tipo de dispositivo (iOS, Android, Older Android) se selecciona la opción "*None*" [13].

Su estructura es la siguiente [13]:

Entradas:

- Visible: muestra u oculta los botones
- *Start index*: nos indica qué opción (botón) está seleccionada de forma predeterminada cuando se abre el filtro.
- Texturas 1-10: Son las diferentes opciones de texturas que se pueden incluir.

Salidas:

- *Selected Option Index*: nos indica el índice de la opción seleccionada (0-10), es decir, el número del botón.
- *Selected Texture*: nos indica la textura del botón que ha seleccionado el usuario que usa el filtro.

En este caso, se utiliza la salida "*Selected Option Index*", aunque lo lógico parezca utilizar la salida "*Selected Texture*". Esto se debe a que la textura de los botones no es la misma que la de la mano, y si usáramos ésta no se conseguiría el resultado deseado. Para los botones se ha usado un pequeño cuadrado de un solo color liso, sin detalle, ya que está más optimizado a pesar de que no pueda comprimirse esta textura.

La textura de la mano si se ha podido comprimir y se añade usando el siguiente nodo.

- **Option Picker:**

Este nodo se usa como selector de opciones para adjudicar valores a un objeto, material u otro elemento [13].

Su estructura es la siguiente [13]:

Entradas:

- Opción: índice numérico (0-31), que indica qué opción se mostrará a la salida

En el interior del nodo, se puede seleccionar el tipo de contenido que se quiere añadir a él. Existen diferentes tipos de datos: señal booleana, escalar, pulso, punto 3D, punto 2D, color, progreso, vector2, vector3, vector4, matriz2, matriz3, matriz4 y textura 2D. Es el último de los mencionados, el que usaremos en este caso. Aquí se añaden las texturas de la mano 3D que hemos importado desde Blender. La salida se actualiza cambiando el campo de textura de este apartado según indica el índice de la entrada.

Salidas:

- Salida: muestra el valor que se ha incluido en el interior de las casillas de texturas del nodo.

Por último, conectamos la salida del Option Picker con el campo de textura material de la mano. En este caso, sólo cambiamos el "Base Color", los demás mapas de texturas son iguales.

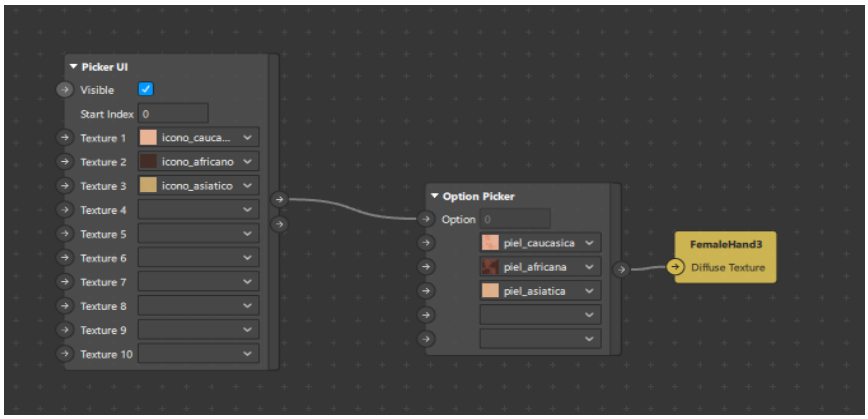

Figura 40. Configuración de los botones para la elección de la textura

3.3.9.4 Iluminación

Para la correcta visualización del objeto 3D en el filtro, es necesario el uso de luces en el proyecto AR. A continuación, se explica cuales se han utilizado.

- **Ambient Light**

Es la fuente de luz ambiente. Esta iluminación afecta a toda la sala por igual y no genera sombras. Sólo permite modificar dos parámetros, el color o la intensidad. Ésta última puede modificarse desde el menú Inspector en valores porcentaje o desde el código en valores del 0,0 al 1,0.

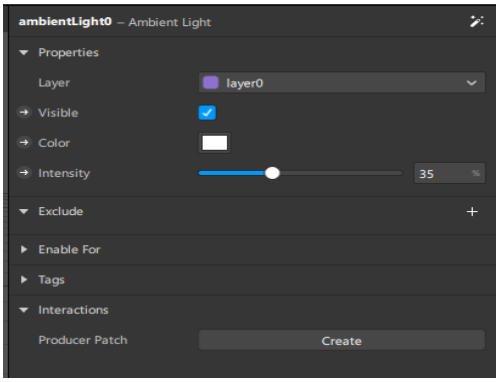

Figura 41. Configuración del Ambient Light

- **Directional Light**

Es la fuente de luz direccional. Funciona como una fuente de luz similar al sol, con una dirección determinada y por ello, genera sombras. Además de los parámetros color e intensidad, en este tipo de luces también pueden modificarse las transformaciones de posición y rotación en el espacio 3D.

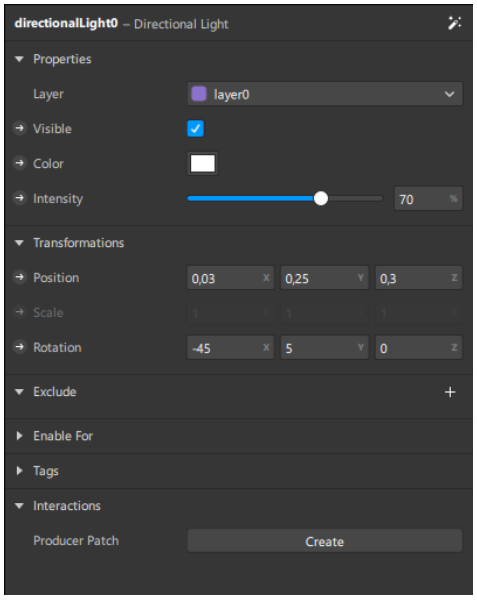

Figura 42. Configuración del Directional Light.

3.3.9.5 Animación

Si queremos que el objeto 3D que hemos añadido al efecto de Instagram tenga movimiento, se puede configurar desde la propia app usando los *Keyframes*, variando su posición y almacenandolos, o desde una app externa e importando el archivo de la animación, en este último caso.

Para nuestro filtro, la animación se ha grabado usando el Leap Motion como se ha explicado anteriormente. Se ha incluido en el modelo *fbx* y al importarlo, también se ha importado la animación. En Spark AR aparece como *recorder_clip*.

Llegados a este punto, ahora se debe configurar correctamente la animación. Para ello, desde el menú *Playback Controllers* se selecciona *animationPlaybackController → Playback Controller → Animation Playback Controller*. En el interior de este componente se añade el archivo *recorder_clip* en el campo *Animation Clip.*

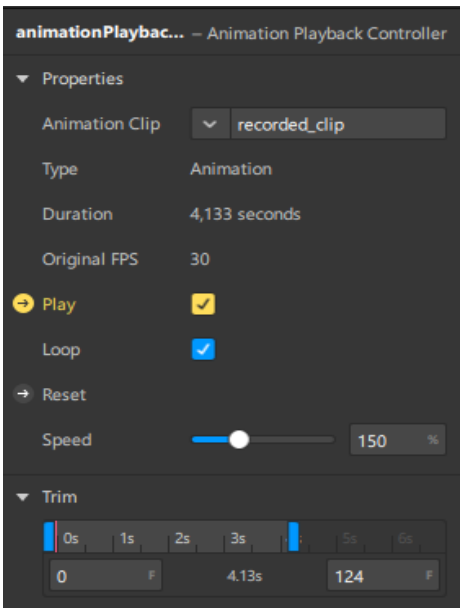

Figura 43. Configuración del animation playback controller

En este caso se busca que la animación se active cuando el usuario mantenga la boca abierta, simulando la expresión de WOW. Para ello, se utiliza la estructura que se ha visto anteriormente, en el apartado 3.3.9.1 Seguimiento del rostro. Se conecta la salida *face* del *Face Tracker* con la entrada *face* del *Mouth Open*, y así se lee el movimiento de la boca de la cara seleccionada. Funciona como una señal booleana, cuando la boca del usuario está abierta una cuarta parte o más, se activa la salida *Mouth Open*, en caso contrario, se activa *Mouth Openness* [13]. El grado de apertura de la boca se mide por la separación de los labios. En este caso, nos interesa que la animación se inicie cuando la cámara detecte que el usuario ha abierto la boca, así que conectamos la señal *Enable* del *Animation Playback Controller* con la salida *Mouth Open*. De este modo, cuando la salida vale "1", la animación se activa; y en caso contrario, si vale "0" la animación se detiene.

Figura 44. Configuración para que la animación se active al abrir la boca

3.3.9.6 Agregar instrucción

Si se desea añadir un texto que indique la acción que hay que realizar para activar el filtro, al abrirlo en el dispositivo, se debe agregar una instrucción. Este proceso consta de los siguientes pasos:

Se selecciona "Device" en el menú de la izquierda:

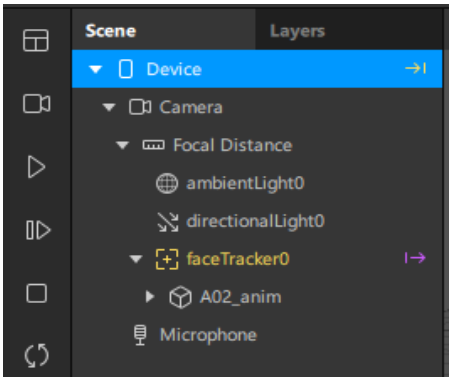

Figura 45. Menú de jerarquía

Al realizar esta acción, aparece en el Inspector (a la derecha de la pantalla) el siguiente menú:

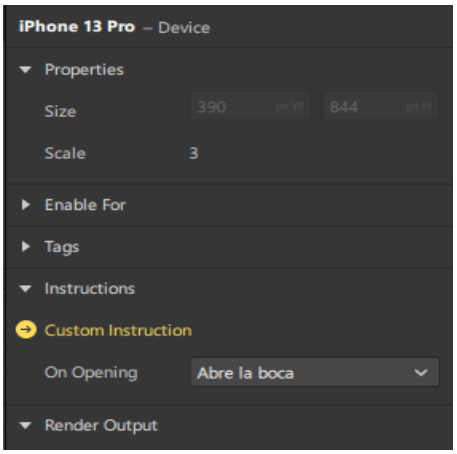

Figura 46. Configuración del dispositivo

Dentro de él, se selecciona la opción *Custom instruction* y se elige la instrucción que encaje con nuestro proyecto. En este caso sería: "Abre la boca", nos aparecen muchas opciones. Una vez se ha seleccionado la instrucción, en el *Edit Path* aparece la configuración que se muestra a continuación:

Figura 47. Configuración de instrucción

En ella, podemos indicar cuánto tiempo queremos que la instrucción aparezca en la pantalla, para ello se usa un comparador *Less Than* (menor que).

En él, se pueden introducir dos valores para compararlos dentro de este bloque. En este caso, la entrada1 se conecta con el bloque *Runtime*, señal de reloj que mide e indica el tiempo (en segundos) desde que el usuario abre el filtro en el dispositivo, y la entrada2, para la que se selecciona el valor numérico 5 segundos.

Continuamente se actualiza el dato *Runtime* y se realiza la comparación entre ambas entradas. La salida de este bloque es una señal booleana, que indica "1" (*true*) si el valor de la entrada1 es menor que el de la entrada2 o "0" (*false*) en caso contrario.

La salida de este bloque se conecta con la entrada *Enable* (Habilitar) del bloque *Device* que se ha explicado anteriormente. De esta forma, cuando la salida indica "1", se activa este bloque y la frase "Abre la boca" será visible en la pantalla del usuario, aunque no en el video final. Cuando la salida pase a ser "0", la frase dejará de estar visible.

En este caso, he elegido que sólo aparezca 5 segundos desde que se abre el filtro.

3.3.9.7 Publicación

Una vez se ha terminado el filtro, se debe publicar para que pueda ser usado por otros usuarios desde Instagram. Para ello, se utiliza Spark AR Hub, web que permite a los creadores publicar y administrar efectos AR para las plataformas de Meta.

En el proceso de publicar un nuevo efecto AR, es necesario rellenar los siguientes campos:

- Nombre.
- Archivo del efecto creado en formato .arexport
- Plataformas en las que podrá ser utilizado. En este caso, se elige únicamente la opción Instagram, pero podrían añadirse Facebook y Messenger.
- Propietario. Cuenta de Facebook con la que desde Spark AR Hub se podrá administrar el efecto
- Editor. Perfil, en este caso de Instagram, en el que parecerá el efecto en la plataforma.
- Categorías. Pueden elegirse hasta cuatro, permiten organizar el efecto y ayudan a mostrar el mismo a personas con búsquedas relacionadas con las categorías. Para este efecto se seleccionan Envolventes, AR del mundo, Causas y *Selfies*.
- Palabras clave. Palabras que describen el efecto, permiten encontrar el efecto introduciendo cualquiera de ellas en el campo de búsqueda. Se seleccionan: Ana Locking, mes del orgullo, *pride*, *pride day*, orgullo, lgtb, uñas, *nails*, mano.
- Video demostración. Video que muestra el funcionamiento del efecto.
- Icono. Miniatura del efecto.
- Colaboradores.
- Fecha de publicación.

Una vez esté creado el filtro se envía para que sea aprobado por la comunidad de Meta. Este proceso puede tardar un máximo de 10 días, aunque en general, responden en 24-48h.

Capítulo 4. Conclusiones

Sin duda alguna creo que todo el trabajo de investigación que he hecho para este proyecto, la resolución de problemas en un breve período de tiempo y la experiencia en este campo de conocimiento me ha ayudado a crecer como ingeniera, y me capacita para desarrollar otros proyectos de igual o mayor envergadura en el futuro.

Debido a lo extenso y variado del proyecto, se realizan las conclusiones de cada uno de los apartados de forma individual

4.1 La aceptación pública del mundo virtual

En general, la gente que probó la sala con las gafas de VR en el evento de presentación en Madrid, quedó encantada con el resultado. Esto se debe a que, aunque la tecnología no esté totalmente desarrollada y todavía le quede mucho camino por delante, los resultados que pueden obtenerse a día de hoy ya son realmente realistas a la par de novedosos. Esto genera en el público que consume estos contenidos por primera vez, el llamado efecto "wow". El nombre de efecto "wow" surge de la onomatopeya característica de asombro, cuando no somos capaces de procesar toda la información recibida y nos quedamos "sin palabras".

Gracias a esto conseguimos captar la atención del público rápidamente y que empiecen con una visión positiva, esta experiencia. Pero después, se necesita crear contenidos realistas y atractivos visualmente para mantener el interés del usuario. Para cumplir con esto, se realizó una amplia sala colorida con temática LGBT+ claramente definida y unos valores que se querían transmitir. En este caso, se quería que la sala fuera un showroom para enseñar las piezas de arte, manos digitalizadas con las uñas de la colección, a modo de museo virtual.

Pero, para conseguir una experiencia VR completa es necesario añadir algo de interactividad. La prioridad en este caso era realizar este proceso de manera elegante, alejándonos de la idea de videojuego de animación que tienen algunas personas cuando se habla de VR. Para ello, los elementos interactivos que se han añadido son el *skybox* que depende de la hora, las animaciones de las manos y los botones que nos enlazan con la página web de OpenSea. El hecho de que haya objetos en movimiento, las manos, en la sala ayuda a mejorar las experiencias. Permitiendo que el usuario se sienta más sumergido en la realidad virtual.

Otro de los grandes prejuicios que la gente tiene con la Realidad Virtual que debemos afrontar es el miedo a colisionar con algo, esto ocasiona que el usuario esté tenso durante la experiencia. Oculus tiene un sistema guardián incorporado que nos avisa si nos salimos del límite seguro que establecemos al iniciar las gafas y que cuenta con un sistema de cámaras. Estas se activan y nos muestra la realidad por pantalla si algún objeto se acerca mucho al usuario, a fin de facilitar a nuevos usuarios superar la inicial inestabilidad sensorial que producen. De este modo se mejora la sensación de seguridad y se mueven mejor por el espacio. También favorece al efecto de mareo que sufren algunos usuarios que prueban estos dispositivos por primera vez, ya que al cerebro le cuesta un poco procesar que se está moviendo por este espacio virtual como si fuese una realidad.

En este evento vimos como muchas personas se apartaban e intentaban esquivar los objetos de la sala con miedo de chocar contra ellos, o incluso alargaban sus brazos para tocar las uñas a pesar de que realmente frente a ellas solo había "aire". Es curioso cómo se consigue engañar al cerebro con estos dispositivos y por un instante, te olvidas de que no estás viendo el "mundo real".

4.2 NFT

Se lanzó la colección de NFTs como prueba de estudio de mercado. El objetivo era conocer la mejor forma de realizar este proceso en proyectos futuros y analizar la respuesta del público ante dicho nicho de mercado. El resultado no fue el esperado, pero nos ayudó a analizar los fallos y en futuros proyectos realizar las modificaciones necesarias para lograr unos resultados más óptimos.

En primer lugar, no se hizo un buen plan de marketing para la publicidad de la venta. Al subir el NFT a OpenSea, éste ya te avisa que la plataforma no se hace responsable de la publicidad de los NFTs. Es el propio usuario quien debe encargarse.

Las notas de prensa que se publicaron y entradas en blog se centraron más en el mundo virtual que en la venta de estos NFTs y el alcance que tuvieron se vió resentido.

Como se ha comentado en el trabajo, existen dos cadenas de bloques principales para crear un NFT, Ethereum y Polygon. Se realizó un estudio para conocer cual cumplía los requisitos para el éxito de este proyecto.

Ethereum es la cadena de bloques más popular, en la que se produce la mayor parte de NFTs que se lanzan al mercado. Por ello existe mayor número de colecciones en ella y se encuentran los artistas más conocidos. Al realizar estas transacciones se pagan unas tarifas de gas que suelen ser muy elevadas debidas a la congestión de la red. Aunque esto es un inconveniente, también ayuda a controlar la red para que haya menos spam. Ofrece la función de subastar los NFTs, algo que no está disponible en Polygon.

Polygon es un protocolo construido para conectar redes blockchain compatibles con Ethereum, surgió como solución de segunda capa para liberar la congestión de Ethereum. Utiliza cadenas laterales, lo que la hace más rápida. Las tarifas de gas son mucho menores, incluso cero en algunos casos por lo que nos ofrece una solución más económica. El inconveniente que esta nos puede presentar es que no es tan conocida y mucha gente desconfía de su fiabilidad.

En este caso se consideró mejor opción usar Polygon, porque se abaratan costes y, además, es menos perjudicial para el medioambiente. Esto se debe a su plan de marketing, basado en la sostenibilidad y en reducir la huella de carbono. Esta iniciativa está recogida en un manifiesto llamado “A Smart Contract with the Planet Earth” donde la plataforma explica que, por cada transacción o movimiento en su red, van a comprar el equivalente estimado en huellas de carbono por su creación o distribución.

El problema principal fue que no se consiguió despertar el interés en el público mediante el marketing ni lograr eliminar las dudas de la fiabilidad de los NFTs para potenciales clientes.

Esto último se debe a la combinación de varios factores: lo comentado acerca de Polygon y el hecho de que ni la artista, Ana Locking, ni el creador del NFT, la empresa MetricSalad, eran ya conocidos en este mundo. Sin embargo, el comenzar este proceso ayudará a MetricSalad a elaborarse esa fama.

A todo esto se le suma la situación de las criptomonedas en ese momento. Recordemos que el proyecto se lanza a principios de Julio de 2022 y el Bitcoin se desplomó a principios de junio de ese mismo año. Esto hizo que se produjera una caída generalizada de todas las criptomonedas y la gente no estuviera tan predispuesta a comprar NFTs.

Otro elemento que pudo afectar a los resultados que ha tenido la venta de NFTs, es el porcentaje elegido como royalties del creador. En este caso, como había muchas personas involucradas que habían invertido dinero y tiempo en la creación de estos, se eligió un 10% a repartir, según convenio, entre las partes. En mi opinión, esta cifra tan elevada frenó la compra por parte de muchos usuarios. Debido a que si sumamos esa cifra a la que te cobra Open Sea por la transacción, es muy difícil sacarle beneficio en una futura reventa, y esto más lo arriesgado de un proyecto tan novedoso, puede asustar a algunos compradores.

4.3 Efecto AR de Instagram

Es una forma sencilla y rápida de incluir elementos AR al proyecto. De este modo se consigue acercar el proyecto a un público más joven y concienciado con la causa, de modo visual y divertido.

El target de edad principal de Instagram es el de entre 25 y 34 años, seguido muy de cerca por el rango de entre 16 y 24 años. 2/3 de las cuentas totales de Instagram tienen menos de 34 años según un estudio elaborado por The Social Media Family en abril de 2022.

Los filtros que se crearon han tenido muy buena acogida por los usuarios de esta plataforma. A continuación, se muestran las gráficas de uso e interacción con estos que nos proporciona Spark AR.

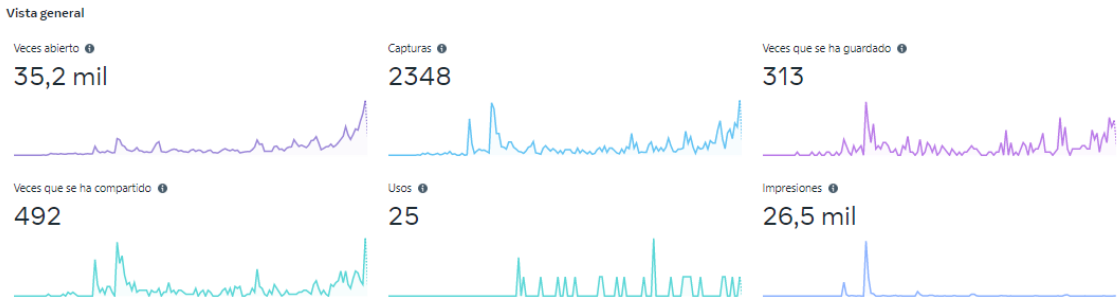

Figura 48. Estadísticas según la interacción del filtro de uñas: Spiders

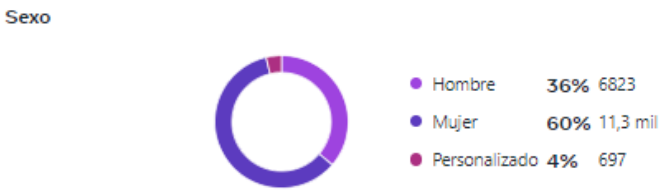

Figura 49. Estadísticas según el género del filtro de uñas: Spiders

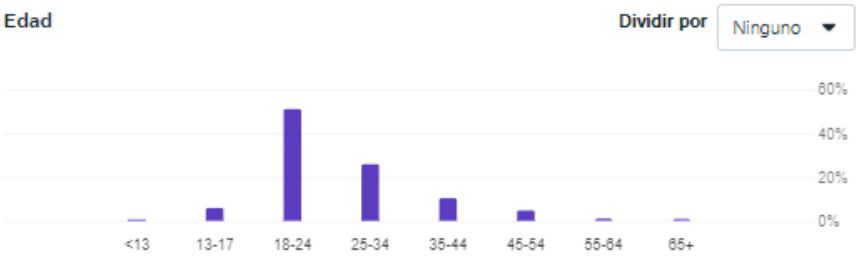

Figura 50. Estadísticas según la edad del filtro de uñas: Spiders

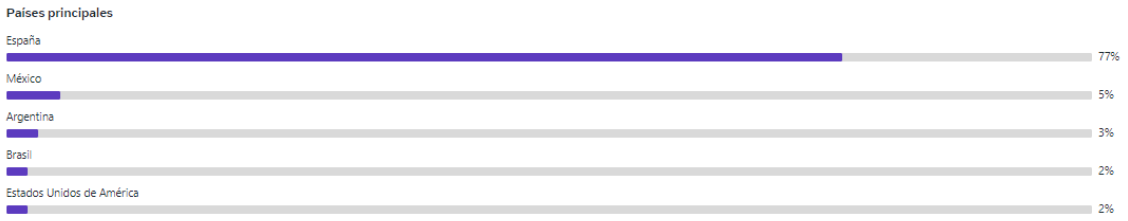

Figura 51. Estadísticas según el país del filtro de uñas: Spiders

Nombre	Impresiones	Veces abierto	Capturas	Veces que se ha compartido	Veces que se ha guardado	Usos	Tiempo medio que permanec...
Spring is coming	404	3563	119	20	19	-	7 s
Spiders from Mars	26 520	35 262	2348	492	313	-	9 s
White Doll	124	3864	136	18	15	-	4 s
Tartán Opulence	23 249	8448	507	75	48	-	6 s
Rose & Thorn	28 249	9070	601	139	108	-	10 s

Figura 52. Estadísticas generales de los filtros de uñas

Estos resultados superan las expectativas iniciales, ya que también era un proyecto piloto que se realizó por primera vez para este trabajo. Sobre todo, el filtro "Spiders from Mars". Analizando las gráficas podemos ver que ha tenido un gran alcance, pero no se ha compartido muchas veces. Esto puede deberse a que quizá los usuarios hayan accedido a él desde las historias de la cuenta de Ana Locking. También podemos observar información según el rango de edad, género o país de los usuarios.

Esto puede ser debido a que la propia Ana realizó videos utilizándolos en diferentes ocasiones y aparecen en el perfil de su Instagram que cuenta con 62000 seguidores. En general, la campaña publicitaria para los filtros fue más elaborada que para los NFTs.

Durante su realización aparecieron diferentes contratiempos pero, se afrontó el reto, y se consiguieron superar sin problemas. Entre estos problemas se encuentra la limitación de peso y el tipo de formato necesario. Para poder exportar el modelo 3D con la animación el fichero debe ser ".fbx" mientras que para el NFT debe ser ".glb" o "gltf".

Para ello, se exporta desde Unity, donde se ha grabado la animación, utilizando diferentes plug-ins.

En el caso del filtro podemos utilizar el *Windows Recorder* para exportar el fbx o un convertidor a "gltf" que hemos encontrado en github, para el caso del NFT.

Obviamente, cuando hablamos de que esta fue la primera vez que se realizaron las funciones de un filtro AR o de un NFT en la empresa no quiere decir que este proyecto se hiciera a ciegas.

Esto lleva un gran estudio detrás de varios meses y la realización de pruebas con elementos más simples y diferentes configuraciones. Pero, fue para el proyecto del Metaverso de Ana Locking la primera vez que se comercializó y se sacó al público.

Capítulo 5. Líneas futuras de investigación

Tras la realización de este proyecto, a pesar de haber abarcado numerosas soluciones; todavía son incontables los retos que quedan por desarrollar a cerca de esta tecnología.

En un futuro se espera ampliar los conocimientos en las siguientes áreas y ponerlos en funcionamiento:

- Encontrar el método para retransmitir videos en *streaming* en este tipo de proyectos. Debido a que, durante el proceso de investigación se ha estudiado el modo de incluir videos y sonido 3D; y no se ha encontrado información clara de como poder reproducir un *streaming* en WebXR.
- Ofrecer este tipo de experiencias en modo multijugador, para que los usuarios puedan interactuar entre sí. Incluyendo a la experiencia avatares, chat de texto y voz.
- Ofrecer una experiencia AR geolocalizada, que varie el contenido mostrado en función del uso del GPS. Configurar un proyecto similar a la tecnología utiliza en “Pokemon Go”, hace uso de la información del acelerómetro del teléfono móvil para conocer la información de posición y orientación.
- Ofrecer experiencias con mayor grado de interactividad con el medio. Para ello, se plantea estudiar las posibilidades que ofrece MRTK y el modo de implementarlas.

Bibliografia

[1] Web personal Ana Locking (s. f.). (2022, Septiembre) http://www.analocking.com/es/biografia.html

[2] Larijani, L. C. (1994). *Realidad virtual*. McGraw-Hill/Interamericana de España.

[3] Burdea, G.C., & Coiffet, P. (1996). Tecnologías de la realidad virtual. Paidós Ibérica.

[4] Martínez, J. M., Martínez, A., & Navarro, F. (2018). Realidad virtual y realidad aumentada. RA-MA Editorial.

[5] Milgram, P. & Kishino, F. (1994). A Taxonomy of Mixed Reality Visual Displays. IEICE Transactions on Information and Systems, 77, 1321-1329.

[6] Entrevista a Mark Zuckerberg. (2021, Octubre). The Metaverse and How We'll Build It Together. https://www.youtube.com/watch?v=Uvufun6xer8

[7] González Muñoz, C., Gracia Bandrés, M.A., Sanagustín Grasa, L., y Romero San Martín, D. (2015). TecsMedia: Análisis Motores gráficos y su aplicación en la industria.

[8] Digital worlds that feel human | Ultraleap. (2022, Octubre) (s. f.). https://www.ultraleap.com/

[9] Web Meta – Oculus Quest 2. (2022, Septiembre) https://store.facebook.com/es/quest/products/quest-2

[10] Escáner 3D Anet HandySense - IT3D Group. (2022, Noviembre) https://it3d.com/escaneres-3d/portatil/anet-handysense/

[11] Blender 3.3 Reference Manual - Blender Manual. (s. f.). (2022, Octubre) https://docs.blender.org/manual/en/latest/

[.11] Scolari, C. A. (2018). Las leyes de la interfaz: diseño, ecología, evolución, tecnología (Vol. 141). Gedisa.

[12] Technologies, U. (s. f.). Unity - Manual: Unity User Manual 2021.3 (LTS). (2022, Septiembre) https://docs.unity3d.com/Manual

[12] Ortega, P. (2022, Marzo). El metaverso llega a la Universidad de Stanford: éxito tras sus primeras clases. El Español.

[13] Spark AR – Manual. (s.f). (2022, Septiembre) https://sparkar.facebook.com/ar-studio/learn/

[13] Ball, M. (2021, Junio). Framework for the Metaverse.

[14] Open Sea – Acerca de (s. f.). https://opensea.io/

[.14] Ponencia de Andrew Price. (2018). The Next Leap: How A.I. will change the 3D industry. https://youtu.be/FlgLxSLsYWQ

[15] MRTK2-Unity Developer Documentation - MRTK 2. Microsoft Learn. (2022, Octubre) https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/?view=mrtkunity-2022-05.

[16] NFTs. Technology wewbsite, Xataka. (2022, Octubre) https://www.xataka.com/

[.16] Martins, D. (2021, Noviembre). Apple Stock: Metaverse Opportunity Is Unfolding.

[17] Stephenson, Neil. (1992). Snow Crash.

[19] Ortega-Rodríguez, P.J. (2022). *De la realidad extendida al metaverso: una reflexión crítica sobre las aportaciones a la educación.* Teoría de la Educación. Revista Interuniversitaria,

[23] Mullen, T. (2012). *Realidad aumentada: crea tus propias aplicaciones*. Anaya Multimedia.

[24] Manual de buenas prácticas de aplicación de la Realidad Aumentada en empresas. (2012). AIMME, Instituto Tecnológico Metalmecánico.

[26] Plataforma en la que se muestra el proyecto, Metaverso Ana Locking: https://analocking.metricsalad.com/metaverso.html

[26] Steptoe, W., Julier, S. and Steed, A., Presence and discernability in conventional and non-photorealistic immersive augmented reality, 2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR). (2014). pp. 213-218, doi: 10.1109/ISMAR.2014.6948430.

Anexos

A. Open URL, enlaces web

Scripts y librearías que se deben añadir al proyecto para poder enlazar una web externa con el proyecto, de modo que al ejecutar el código se abra una nueva pestaña en el navegador. Se modifica para enlazar el proyecto webXR con la página Open Sea.

OpenWindow:

```
mergeInto(LibraryManager.library, {
  OpenURLInExternalWindow: function (url) {
    window.open(Pointer_stringify(url), "_blank");
  }
});
```

NewTab:

```
using UnityEngine;
using System.Runtime.InteropServices;

public class NewTab : MonoBehaviour
{
    [DllImport("__Internal")]
    private static extern void OpenURLInExternalWindow(string url);

    public void NFT_Rose()
    {
OpenURLInExternalWindow("https://opensea.io/assets/matic/0x2953399124f0cbb46d2cbacd8a89cf0599974963/98424577951773872312668212779408287291793475123920534688580967957776425811978/");
    }

    public void NFT_Spring()
    {
OpenURLInExternalWindow("https://opensea.io/assets/matic/0x2953399124f0cbb46d2cbacd8a
```

```
89cf0599974963/98424577951773872312668212779408287291793475123920534688580967954477890928650/");
    }

    public void NFT_Spiders()
    {

OpenURLInExternalWindow("https://opensea.io/assets/matic/0x2953399124f0cbb46d2cbacd8a89cf0599974963/98424577951773872312668212779408287291793475123920534688580967955577402556417/");
    }

    public void NFT_Tartan()
    {

OpenURLInExternalWindow("https://opensea.io/assets/matic/0x2953399124f0cbb46d2cbacd8a89cf0599974963/98424577951773872312668212779408287291793475123920534688580967953378379300874/");
    }

    public void NFT_WhiteDoll()
    {

OpenURLInExternalWindow("https://opensea.io/assets/matic/0x2953399124f0cbb46d2cbacd8a89cf0599974963/98424577951773872312668212779408287291793475123920534688580967956676914184202/");
    }
}
```

B. Skybox

Script del Skybox llamado Stylized Sky que modifica el skybox y la iluminación en función de la franja horaria que nos encontramos. Se calculaba minuto a minuto y se ha modificado con el fin de lograr que cada franja de cambio sea de una hora. Y para que se adapte a la distribución de iluminación de la escena.

```
using UnityEngine;


[ExecuteAlways]
public class LightingManager : MonoBehaviour
{
   //Scene References
   [SerializeField] private Light DirectionalLight;
   [SerializeField] private LightingPreset Preset;
   //Variables
   [SerializeField, Range(0, 24)] public float TimeOfDay;



   private void Update()
   {
      if (Preset == null)
         return;


      if (Application.isPlaying)
      {
         //(Replace with a reference to the game time)
         TimeOfDay = System.DateTime.Now.Hour;
         TimeOfDay %= 24; //Modulus to ensure always between 0-24
         UpdateLighting(TimeOfDay / 24f);
      }
      else
      {
         UpdateLighting(TimeOfDay / 24f);
      }
   }
```

```
    private void UpdateLighting(float timePercent)
    {
        //Set ambient and fog
        RenderSettings.ambientLight = Preset.AmbientColor.Evaluate(timePercent);
        RenderSettings.fogColor = Preset.FogColor.Evaluate(timePercent);


        //If the directional light is set then rotate and set it's color, I actually rarely use the rotation because it casts tall shadows unless you clamp the value
        if (DirectionalLight != null)
        {
            DirectionalLight.color = Preset.DirectionalColor.Evaluate(timePercent);


            DirectionalLight.transform.localRotation = Quaternion.Euler(new Vector3((timePercent * 360f) - 90f, 170f, 0));
        }


    }




    //Try to find a directional light to use if we haven't set one
    private void OnValidate()
    {
        if (DirectionalLight != null)
            return;


        //Search for lighting tab sun
        if (RenderSettings.sun != null)
        {
            DirectionalLight = RenderSettings.sun;
        }
        //Search scene for light that fits criteria (directional)
        else
        {
            Light[] lights = GameObject.FindObjectsOfType<Light>();
            foreach (Light light in lights)
            {
```

```
            if (light.type == LightType.Directional)
            {
                DirectionalLight = light;
                return;
            }
        }
    }

}
```