Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Desarrollo de un entorno de realidad virtual para la inmersión en obras de arte

Autor: Juan Jesús Ruiz Toscano

Tutores: Irene Fondón García
María Auxiliadora Sarmiento Vega

Dpto. Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Proyecto Fin de Carrera
Ingeniería de Telecomunicaciones

Desarrollo de un entorno de realidad virtual para la inmersión en obras de arte

Autor:

Juan Jesús Ruiz Toscano

Tutores:

Irene Fondón García

Profesora Titular

María Auxiliadora Sarmiento Vega

Profesora Titular

Dpto. de Teoría de la Señal y Comunicaciones

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2021

Proyecto Trabajo Final de Grado: Mondrian desde dentro. Una experiencia inmersiva e interactiva de realidad virtual en el arte

Autor: Juan Jesús Ruiz Toscano

Tutores: Irene Fondón García

María Auxiliadora Sarmiento Vega

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

A los que creyeron

A los que confiaron

A los que quiero

A los que dudaron

AGRADECIMIENTOS

Este trabajo es la culminación de un crecimiento personal de varios años de preparación, dónde se han forjado las correspondientes competencias académicas al mismo tiempo que mi propia personalidad crecía, llegando a límites nunca antes imaginados.

Soy como soy gracias a personas a mi alrededor con las que he podido contar y alcanzar la propuesta realizada en esta experiencia. Todas ellas han sido parte del producto final, el cual se adentra en la tecnología y el arte a partes iguales. Siendo conscientes o no, estas personas han dejado huellas que, incluso en algunos casos, se siguen manteniendo tras muchos años y que, por ellos mismos, con sus propios nombres, se han merecido todas mis gratitudes. Es por ello por lo que en esta sección quiero agradecerles la implicación que han tenido conmigo durante todo este tiempo. De corazón, gracias.

A Irene y Auxi. Porque han confiado ciegamente desde un primer momento en un proyecto de esta envergadura. Por abrirme las puertas a grandes logros y formar parte de este gran equipo. Y dentro del mismo Departamento, a Ramón, Iván, Sergio, Susana, Begoña, Carmen y Juan Antonio, porque el aprendizaje va más allá de asignaturas concretas. Seguiremos viéndonos, y ojalá sea con nuevos proyectos.

A Inma y Evaristo. A él, por sus explicaciones en el arte, haciendo que floreciera aún más mi interés por este campo. A ella, por aquella frase que un momento concreto me dijo y ha sido uno de los motivos principales a lo hora de hacer este proyecto final. Sin ambos, el arte no hubiera aparecido.

A Patricio, porque de principio a fin hemos sufrido juntos, pero como sabemos, aún quedan muchos junios por vivir. A Álvaro que, aunque de forma tardía, conocerlo fue una gran sorpresa. Las críticas nunca fueron tan finas ni con tanto disimulo. A Lucía, Celia y Mario, porque en los últimos años fueron indispensables y sin ellos, nada hubiera sido lo mismo. Quedan muchas cervezas por compartir. A Carlos, que empezamos hablando de aficiones muy nuestras y hemos acabado compartiendo conceptos. Muchas fotos y bullas nos quedan aún. Y por supuesto, a Dani e Ignacio, a mi tertulia de telecos. Amigos y compañeros que me llevo allá donde vaya. Extraordinarias personas de ambientes extraordinarios.

No puedo dejar pasar la oportunidad de recordar a mi otra cuadrilla, la que siempre va de frente. En especial a Enrique y Gonzalo, por la confianza, las ganas e ilusiones. Porque las Juntas desde las sombras siempre se han llevado bien. A Alejandro, Alberto y Jaime, porque todo tiene un principio y sin vosotros nada hubiera salido. A Pedro, por las grandes oportunidades que aún hoy me sigue dando, siempre dispuesto a enseñar lo mejor de sí. Y al resto de miembros, con los que quedan muchas tardes, presentaciones y vivencias que compartir.

A los que me cuidan. A Antonia, a Domingo y Nicolás. Os quiero, nada más que decir. Bien lo sabéis. Y a Rosario, porque soy quien soy también gracias a ella.

A Luis, porque ser compadre y hermano no es incompatible. Que lo que han unido las circunstancias no lo separa nadie.

Y, por último, los más importantes. A quienes me dieron la vida y me cuidan cada día, Manolo y Bartolina; y a Marta, que hoy es quien me da la vida y con quien comparto mi mayor proyecto. A Bartolina, porque aquí somos luchadores y se demuestra hasta el final. A Manolo por sus ganas de aprender y enseñar, por ser de las personas más buenas que he conocido en la vida, aunque algunos se quieran aprovechar de ello. A Marta, porque los tesoros no son de oro, sino de vida e historias. Os quiero como nadie jamás lo ha hecho.

A todos, gracias. No sería quien soy ni habría conseguido nada sin estar rodeado de los mejores. Cada uno de ellos, y los que faltan, pero que siempre tengo en mente, son los nombres sobre los que se sustenta este trabajo final de grado. Sin ellos, no hubiera sido posible. Ahora vienen nuevos retos y metas. Sólo espero que me acompañéis en el camino y poder teneros a mi lado por muchos años más. Siempre de frente.

RESUMEN

Teniendo en cuenta investigaciones anteriores en el campo de la realidad virtual aplicada a los museos, se puede observar que existen diversas exposiciones virtuales donde el usuario tan solo se encuentra envuelto por el entorno, sin la capacidad de poder interactuar con otros objetos. Aunque existen distintos trabajos interactivos, ninguno de ellos es también inmersivo con gafas de realidad virtual. Debido a ello, se ha decidido crear una aplicación que combine interacción e inmersión en un entorno virtual y cuyo tema central sea el arte.

El fin principal de este proyecto es que el usuario forme parte de una pintura y pueda interactuar con ella. Se ha decidido hacer con una pintura de Mondrian, en concreto la obra “Composition A”. Se ha elegido esta pintura por su sencillez en el montaje, ya que cada cuadrado modelado en un espacio tridimensional, se convierten en un cubo. Teniendo el modelo completo en 3D, cada cubo es capaz de realizar diferentes acciones. Aprovechando este hecho, el usuario va a poder conocer más aún acerca del autor, su biografía, estilo artístico que usa en sus pinturas, formas de vida durante su época y algunos modos de interacción con el fin de conocer y jugar con el arte.

El objetivo es aprender acerca del arte al mismo tiempo que el usuario se divierte en un entorno de realidad virtual, aprovechando todas las posibilidades que esta tecnología ofrece. Finalmente, se ha realizado una experiencia como nuevo modo de aprendizaje para saber quién fue Mondrian y cómo cambió la concepción del arte.

ABSTRACT

During the past ages, museums have been related to scientific advances. At the same time new research have been appearing, artists and creators have included all of these new devices. Nowadays one of the most important elements which can be found is virtual reality.

Attending to previous works in virtual reality applied to museums, we can see that there were many virtual expositions where the user just is involved by an environment, without being able to interact to other objects. Although many interactive works of art exist, no one of them are also immersive with virtual glasses.

For this work we propose to take part into a painting and interact with it. It was decided to do it in a Mondrian's painting, specifically "Composition A". Due to this painting, as most of authors' paintings, is made of squares. Those ones, when are created in a three-dimensional space, become in cubes. Having the whole 3D model, each cube can make different actions. Taking advantage to this task, the user is going to know about the painter, his biography, style of art used in his paints, ways of leaving during his life and some ways of interaction to understand and play with art.

The main purpose of this project is to learn about art at the same time the user enjoys. We wanted to introduce virtual reality, with all the chances this technology offers, as immersion and interaction, to art. Finally, we did this experience as a new way of learning who was Mondrian and how he had changed art.

ÍNDICE

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

En este trabajo, se procede a explicar una aplicación desarrollada en realidad virtual (o VR en inglés) que introducirá al usuario a una pintura de Piet Mondrian.

La idea es atraer a un mayor número de personas al arte haciendo que sientan que forman parte de él. Para ello, la posibilidad que nos facilita esta tecnología de interacción e inmersión es de gran ayuda. Con respecto a la inmersión, la persona se encontrará en el interior de una sala a oscuras sin techo, donde el cielo está repleto de estrellas. En la zona central, pegado a una de las paredes, se encuentra ubicada la pintura "Composition A" del mencionado autor. A pesar de que más elementos conforman la sala, no serán vistos hasta que el usuario avance en las sucesivas escenas que constituyen la experiencia. Por otro lado, se ha incluido gran cantidad de objetos interactivos. Esta colaboración entre usuario y aplicación será necesaria, ya que sin ella no serán posibles los cambios de escena para la experiencia completa.

Aprovechando ambos conceptos mencionados, se explicará brevemente la biografía, historia y conceptos artísticos del autor, así como diferentes ideas en las que el usuario podrá aprender jugando de forma interactiva, haciendo que la experiencia completa sea más atractiva para conocer el arte.

Repasando las aplicaciones de realidad virtual ya existentes, se puede apreciar que, a pesar de la presencia de varias experiencias en museos, la gran mayoría están únicamente enfocadas a la inmersión, dejando de lado la interacción con el entorno. Pese a que es una propuesta interesante, hace que el usuario no absorba del todo lo que se quiere exponer, ya que la interacción con el entorno es inexistente. Así pues, esta propuesta viene de añadir interacción a la inmersión ya existente, llevando la realidad virtual en el arte más allá.

Este trabajo engloba dos caminos diferentes. El primero de ellos, hacer una aplicación lo más accesible para todos. La idea es que cualquiera pueda usarla sin problemas. La segunda motivación es el fin cultural, divulgativo y educativo que la propia experiencia ofrece. La posibilidad de poder usarlo en cualquier lugar, facilitando la enseñanza y llevándola a límites aún desconocidos, es quizás el mayor atractivo de esta aplicación.

A continuación, se procede a explicar la evolución histórica que han experimentado tanto los museos como la realidad virtual, en qué punto actual se encuentra el conjunto de ambas y cómo han servido de referencia para el proyecto realizado. Posteriormente, se explicarán los programas usados, qué se ha realizado en cada uno de ellos, resolviendo los problemas a medida que iban apareciendo y las conclusiones a las que se ha llegado. Por último, se mostrarán los resultados de una prueba de calidad que se ha realizado en distintas personas con el fin de evaluar la experiencia y los posibles futuros proyectos que se podrían realizar tomando este como referencia.

2 EVOLUCIÓN HISTÓRICA

2.1. Museos:

Comenzaremos analizando el estado del arte de los museos, de dónde vienen, en qué punto se encuentran y hacia dónde se dirigen.

El principal fin de los museos procede del coleccionismo de un conjunto de objetos sujetos a la protección especial con la finalidad de exponerlos, ya fuera a individuos públicos o privados. Así pues, el museo tiende a ser una exposición completa donde sus elementos tienen relación entre sí, ya sea pertenecientes a épocas o contextos similares [1, 2].

Prestando atención a la propia historia, es necesario comenzar por la antigua Babilonia. En esta situación geográfica e histórica, existen referencias al palacio del Rey Nabucodonosor II (605-502 a.C.), el cual se conocía como "gabinete de maravillas para la humanidad". Este palacio destacaba por la gran cantidad de piezas que formaban parte del tesoro del rey. Piezas de alto valor artístico, de diversas procedencias y mostradas como motivo de fuerza y poder, ya que los mencionados tesoros provenían de los botines de guerra que el ejército del rey conseguía de sus asaltos y victorias [3, 4, 5].

Posteriormente, se debe continuar con el Antiguo Egipto. Se han descubierto objetos que se guardaban con los faraones en el interior de las tumbas, en las zonas más ocultas de las pirámides, con el fin de dotar de la máxima supervivencia a los gobernantes mencionados. Y, aunque el fin principal no era de conservación ni exposición pública, sí era entendido como un modelo de exposición a los Dioses. Además, hay que destacar la importancia que tuvo la antigua ciudad de Alejandría, afincada al norte del país, en la zona más occidental del delta del Nilo. En esta ciudad se origina el primer "museo", el cual fue organizado por Ptolomeo I Soter (367-283 a.C.) y su hijo Ptolomeo II Filadelfo (308-246 a.C.). Según las investigaciones, este museo se encontraría unido a la conocida Biblioteca de Alejandría. Debido a ello, fue punto neurálgico y zona de encuentro de poetas, artistas y sabios, siguiendo el modelo de las escuelas atenienses de la época. Como consecuencia directa de estos encuentros, eran tratados muchos temas de raciocinio, culturales y artísticos, dotando al lugar de importantes corrientes de pensamiento que calarían profundamente en la sociedad del momento [6, 7, 8].

Avanzando en la historia, se llega a la Grecia del siglo V a.C. En dicho punto, las obras de arte se exponían sobre los llamados *mouseion*, los cuales estaban consagrados en honor a las musas, que se consideraban protectoras de las Artes y Ciencias. Se construyeron los templos (*thesaurus*), lugar dónde se recibían los exvotos y tesoros. Estos edificios serían los orígenes de las primeras pinacotecas públicas y abiertas a los ciudadanos, que serían usadas para dar mayor difusión a las corrientes filosóficas y artísticas que imperaban en el lugar [9].

Durante la época de la antigua Roma, los museos eran usados como colecciones privadas con los botines de las guerras, que posteriormente se expusieron al público. Se empezó a concebir la filosofía de "museo al aire libre". Basándose esta idea, proliferaron mercados de arte, falsificaciones y restauraciones. Surge la concepción de colección artística como inversión de capital, prestigio político y coleccionismo, así como un afán por la protección del arte.

El Emperador Octavio Augusto (63 a.C. - 14 d.C.) reagrupa las colecciones particulares para disfrute público y dictamina leyes, con el fin de proteger el patrimonio que se expone. En esta época destaca

la difusión cultural mediante exposiciones con pinturas de hazañas de los grandes estrategas y héroes romanos.

Con la correspondiente evolución histórica, en la Edad Media el cristianismo es el gran precursor del arte. La religión hace uso del arte con intenciones pedagógicas y morales, como medio de enseñanza para una sociedad mayoritariamente analfabeta. La difusión de mensajes, enseñanzas y liturgias mediante imágenes da la posibilidad al clero de ofrecer un mayor entendimiento y formación a los fieles; y a la población de una mejor comprensión de lo que se exponía. Durante tiempo, los templos son usados como museos públicos, en los cuales las principales obras devocionales serían relicarios de distintos santos. Aunque no de manera sentimental, otras obras de arte serían altamente valoradas materialmente y, en especial, simbólicas, como fueron los tesoros profanos causados por los saqueos durante las cruzadas. Se interrumpe la exposición pública en recintos civiles; se desestima el *museion*, entendidos como un avance de las ciencias y las letras y se reduce el gusto por las colecciones privadas, las cuales disminuyen. Este hecho hace que se retrase la formación de museos como instituciones, más aún desde los gobiernos locales.

A partir del renacimiento, el obispo humanista Paolo Giovio acuñó el término "*museum*" para describir las colecciones y edificios que albergaban los diferentes elementos artísticos. En el Palazzo de Giardino de Sabbioneta, en la ciudad italiana de Mantua, se encontraban las mejores colecciones privadas del momento [10].

Exposiciones privadas de estatuas, bajos relieves y bustos fueron los grandes condicionantes de la concepción moderna que tenemos de los museos, asemejándolos a galerías. Destacan especialmente el Palacio de los Uffizi, el cual recogía colecciones de los Médicis coordinadas por Vasari y el Antiquarium o Sala de Antigüedades de la antigua Residencia de Múnich, el cual era el anterior palacio real de los Reyes de Baviera [11].

Durante esta época empiezan las primeras medidas contra la exportación ilegal de obras de arte. Todo ello debido a la gran valoración histórica, artística y cultural que la sociedad tenía sobre las obras de arte. Durante el renacimiento, encontramos el primer coleccionista moderno documentado, el duque francés de Berry.

El hecho de coleccionismo y preservación durante esta época dio lugar a un incremento de interés científico y pedagógico. Motivado por ello, la aristocracia cortesana, la Iglesia y la burguesía culta trataron de acaparar la mayoría de las obras de arte, donde buscaban lo bello y lo pintoresco. Este motivo hizo florecer un aumento en cantidad y calidad de las obras, de la mano del propio mecenazgo de las sociedades ya mencionadas. Con el incremento de calidad en el arte floreció la aparición de críticos de arte y catálogos como guías. Aunque se desarrollaron distintos aspectos artísticos, destacan dos principalmente: el primero de ellos, la "*naturalia*", donde el arte provenía de la propia naturaleza; y la "*artificialia*", cuyo origen estaba en la mano del hombre. Estos serían los dos principales caminos que tomaría el arte renacentista.

En el año 1565 se escribe el primer tratado de museología escrito por el médico holandés Samuel von Quiccheberg. Este tratado recogería la idea de museo multidisciplinar, combinando la *naturalia* y *artificialia*.

Durante la Edad Moderna, como consecuencia de la Revolución Francesa, se crea el primer museo de carácter público: el Palacio Real del Louvre, el cual reinterpretaría la definición de museo y sería referente mundial hasta la actualidad, junto con el Palacio de Cristal, en Londres.

Esto motivó el interés por exponer mayor variedad y estilos de contenidos, ya que siempre se había mantenido distancia y arrogancia frente al público. Público que, a partir de esta época, empezaría a tomar conciencia y formar parte, principalmente con el fin de educar y conservar el patrimonio. En los

palacios franceses surgen galerías artísticas, que tenían un esquema de presentación lineal. En Italia aparecen los museos de reproducciones con fines didácticos.

Se pueden observar museos con tres tipologías museísticas diferentes: museos de arte; museos de ciencias naturales; y museos arqueológicos. Van originándose nuevos museos de Bellas Artes dónde se exponían las obras y colecciones artísticas de los alumnos, así como la creación de sociedades científicas, las cuales formarían mayoritariamente parte de la burguesía. Durante este período se abren el Museo Británico, Ermitage y Vaticano.

Finalmente, se expone la época que más nos interesa en referencia a este proyecto. Durante la época contemporánea, en el siglo XIX Goethe, en Alemania, define las ideas sobre la expansión y la función de los objetos. Comienza todo a abrirse finalmente a lo público. A partir de entonces, se entienden los museos como lugares donde admirar obras de arte y se desarrollan en dos ámbitos. El primero, relacionado con el arte, se focaliza en que la vida del hombre varía según las circunstancias y el estado de ánimo que lo envuelve. Por otro lado, desatacan los museos en sí mismos, donde se pueden comprobar distintas variedades de artes diferentes, tanto en técnica, como ejecución o temática. Los museos se adaptan como una realidad viva y entienden que lo artístico no mueve. Para ello, se da mayor libertad y la existencia pasa por los museos. Se centra en una cultura humanista dónde lo más importante es la relación entre la obra de arte y la idea del hombre. Durante los años 50 la sociedad sufre un cambio cultural en la manera en que se entienden los museos. En el año 1974 se definen los estatutos para los museos. Los mismos serán sin fines lucrativos al servicio de la sociedad, la cual adquiere, conserva, comunica y presenta con diversos fines. Entre ellos están el estudio, la educación y el deleite basados en testimonios materiales del hombre y su medio. Así mismo, incluyen institutos de conservación y galerías de exposiciones de archivos y bibliotecas. En 1983 el Comité Internacional de los Museos (ICOM) incluye los parques naturales, arqueológicos e históricos, así como centros científicos y planetarios en la categoría de museos.

Ha existido una evolución, entendiéndose como muesos-mercados que ofertan productos culturales para el gran público. Se toma esta idea como “renovarse o morir”.

Gracias a los nuevos medios técnicos, humanos, materiales y financieros, y promovidos por la colaboración con el capital privado de los patrocinios y mecenazgos culturales, se consigue el crecimiento de estas instituciones, tomándolas empresas colaboradoras. En este punto las obras de arte se ven como instrumentos financieros.

Con la influencia que cobran los museos, la política museística se conforma a través de la suma de varios aspectos, como son el contenido (colecciones), continente (edificio), personal interno (especialistas, administrativos, técnicos, subalternos) y externo (público). De esta manera, todo este conjunto conforma la concepción de museo en la época contemporánea.

Con motivo de la variedad de arte que existen, se descentralizan en pequeños museos especializados según los distintos dominios de la historia y las tecnologías, siempre acorde a la evolución de la sociedad.

En la actualidad, los museos modernos tienen en mente otros aspectos, como son las emociones, la complejidad, la intención durante el proceso o distintas formas de visualización de los objetos. También son aceptadas las copias, a veces incluso alterando la original y reinterpretándolas, dotándoles de un enfoque más informal, comunicativo e inconformista. La idea es innovar, siendo más creativo y popular.

Los propios autores actuales se cuestionaron las obras de arte, tanto del pasado como de nuestros tiempos, enfocándose sobre todo en los modos de presentación. Se separan las obras de las paredes, dotándolas de mayor dinamismo y profundidad. Existe una relación directa entre la importancia histórica y la estética.

Con el avance de la ciencia y la sociedad, se adaptan nuevos medios y tecnologías, dando lugar a un aumento de la investigación y la difusión.

Finalmente, con motivo de la búsqueda entre emociones y sensaciones, teniendo siempre presente la relación del museo con el territorio, las nuevas tecnologías son aplicadas al arte, iniciando un nuevo camino que aún se está descubriendo y experimentando.

2.2. Realidad Virtual:

Atendiendo a los avances diarios, es visible que el futuro de la realidad virtual es mucho más cercano de lo imaginado. En este apartado se explica la historia y los elementos que componen esta tecnología.

Si se atiende a las definiciones que dieron los expertos, se puede definir como un dispositivo que intenta simular de forma lo más realista posible el mundo real y su interacción. Para ello, se necesita principalmente de dos factores: una correcta inmersión en el mundo virtual y la presencia dentro de este mundo. La inmersión es la sensación que provoca el entorno que rodea al usuario, que es la que se produce cuando giramos la cabeza con el dispositivo. Por otro lado, la presencia es la sensación de estar presente en el entorno mencionado, como puede ser el saltar físicamente los escalones. Para conseguir una mejor inmersión, se puede recurrir a la estereoscopía, la cual dota de mayor profundidad, proporcionando a cada ojo una imagen diferente [12, 13].

La realidad virtual se encarga de generar digitalmente un entorno tridimensional en el que el usuario se sienta presente y sea capaz de interactuar de manera intuitiva y en "tiempo real" con los distintos objetos que forman parte de él. Los mencionados objetos tridimensionales presentas sus propiedades intrínsecas como son la gravedad o la fricción, así como mantener la posición u orientación dentro del entorno.

Existen distintos dispositivos que generan entornos de realidad virtual. El primero de ellos son las gafas móviles, en lo que tan sólo se tiene una carcasa que sirve como sujeción de los propios dispositivos móviles, por lo que únicamente se puede tener un entorno inmersivo, en el que las limitaciones de calidad proceden del terminal móvil. Por otro lado, se pueden encontrar cascos que permiten tener inmersión e interacción gracias a mandos externos. Se dividen en dos tipos, aquellas que requieren de un cable y conexión a ordenador y en segundo lugar las autónomas, que pueden valerse por sí mismas y no necesitan de elementos externos a los que conectarse. Las limitaciones de calidad suelen estar mejor preparadas para entornos virtuales mejor que los móviles. Sin embargo, únicamente se explicarán la usadas, las Oculus Rift S, que necesitan de cable y ordenador para su funcionamiento. Cabe destacar que las autónomas también podrían ser usadas en el proyecto [14, 15, 16].

Gracias al correcto uso de algunas características como son la resolución y fidelidad de la imagen, las propiedades de los objetos y escenarios, reacciones de objetos, interactividad o respuesta sensorial, la cual ayuda a mostar representaciones anatómicas como las manos, se puede conseguir la mejor experiencia.

Una vez explicado el fin principal de la realidad virtual, se procede a mostrar el funcionamiento y los componentes que conforman los dispositivos. Hay que entender cómo se crean las imágenes inmersivas que interesan. Y es que el funcionamiento en estos dispositivos, a grandes rasgos, es sencillo. La idea es colocar una pantalla frente a cada ojo, siguiendo el mismo modelo de las antiguas gafas con cristal rojo y azul que se usaban en las pantallas de cine. Sin embargo, en este caso lo que se proyecta en ambas pantallas es lo mismo. De esta manera se consigue tener un casco que represente imágenes estereoscópicas tridimensionales Gracias a los sensores de las propias gafas, es posible mover la cabeza y tener visión 360º [17].

En la actualidad, la mayoría de los cascos de realidad virtual llevan incorporados paneles AMOLED, los cuales tienen una tasa de refresco alta (capacidad de los dispositivos para reflejar los números de fotogramas en una pantalla), que suelen tener como mínimo 60Hz, aunque existen dispositivos con hasta 90 o 120, facilitando la inmersión del usuario al tener mejor sensación de movimiento. Otro

punto para tener en cuenta es el campo de visión, que es la extensión del mundo, en forma de ángulo, que el ser humano puede observar. El ojo humano abarca hasta los 114º de campo de visión de media, sin embargo, las gafas de realidad virtual se quedan en un ángulo menor, entre 100 y 110º para los visores de más calidad [18].

Con respecto a la sensación de movimiento, es necesario para el usuario que la orientación de la cabeza con respecto a la cabeza virtual sea lo más realista posible. Para ello se recurre a sensores como el acelerómetro, el giroscopio o el magnetómetro. Además de ello, y la mencionada tasa de refresco, es necesario tener en cuenta la latencia. Este es el retraso entre el movimiento que ocurre al capturar el dispositivo la señal y el instante en que se percibe el resultado por el usuario. A más alta latencia, peor resultado. Por ello, se establece una latencia máxima de 20ms, aunque los dispositivos tienen menor. Estos 20ms, según distintos experimentos, es la latencia máxima que deben tener los equipos para que el usuario tenga una correcta y cómoda experiencia [19].

También es necesario destacar un efecto para evitar como es el efecto rejilla. El mismo se produce por tener la vista colocada cerca de una pantalla. Esta cercanía hace que se puedan observar los píxeles como si fuera la propia matriz de puntos. Este aspecto es importante a la hora de diseñar los dispositivos y los cascos, pero el problema está siendo corregido por la mayoría de las compañías.

2.2.1. Evolución histórica de la realidad virtual

Para iniciar el contexto histórico, se debe recurrir a Jaron Lanier, director de VPL *Research*, en el año 1989. Su intención era simular realidades alternativas que a la vez fueran interactivas, basadas en mundos digitales. Atendiendo a lo comentado en los puntos anteriores, esta tecnología se explica como una "simulación interactiva por computador desde el punto de vista del participante, en la cual se sustituye o se aumenta la información sensorial que recibe". Esta definición dada por A. Rowell en el año 2009, argumenta perfectamente el fin principal de la realidad virtual como tecnología [20].

Con el fin de ofrecer un recorrido de la historia al completo, hay que recurrir a los años 50s, y es que, con el origen de la Segunda Guerra Mundial, EE. UU. buscaba una forma de mejorar y agilizar el entrenamiento de los pilotos mediante simulaciones de vuelo. Posteriormente, en los años 60s la comunidad científica siguió experimentando con esta tecnología en ámbitos técnicos, científicos y de ocio, buscando la relación entre hombre y ordenador.

Es importante mencionar la figura de Ivan Sutherland, profesor del MIT y pionero en el campo de la computación gráfica. En 1965 crea el primer sistema para sentir el juego. Posteriormente, en 1968 crea el primero casco de realidad virtual o HMD (*Head Mounted Display*), llamado "La Espada de Damocles", el cual constituía un sistema conectado a un teclado. De esta manera, se conseguía la orientación de la cabeza y la inclusión de pantallas estereoscópicas [21, 22, 23].

Entre las décadas de 1970 y 1990, se crea el proyecto Aspen *Movie Map*, desarrollado por el MIT, el cual representa virtualmente la ciudad de Aspen. Con respecto a las tecnologías, se abaratan los precios, encontrando una mejor relación entre coste y beneficio. Durante esta época comienza realmente la primera generación inmersiva, enfocados en centros de investigación e industria. Los dispositivos son cada vez más pequeños, potentes y asequibles para la mayoría de público.

En la última década del siglo XX y primera del XXI, ocurren distintos eventos que revolucionan esta tecnología. Nintendo y Sega crean sus propios cascos. Son las pioneras en el uso de la realidad virtual como entretenimiento. Se pasa de la idea de uso militar a uso civil.

Con respecto a los simuladores de vuelo, Thomas Furness consigue gran cantidad de avances en esta materia. Así, en 1982 desarrolla el sistema VCASS, que consiste en un simulador de vuelo avanzado.

En 1984, el Centro de Investigación Ames de la Nasa crea el proyecto VIVED, que evalúa el potencial de los sistemas para entrenar a futuros astronautas. El crecimiento de la tecnología y los avances dan lugar a la expansión y crecimiento en ámbitos como los videojuegos. Aprovechando la inmersión de las gafas, se empiezan desarrollando sobre todo juegos de terror.

A partir del siglo XXI, grandes empresas como Google, HP, Samsung o Microsoft empiezan a desarrollar nuevos dispositivos. Esto hace que el mercado sufre una gran expansión en la tecnología y, por ende, un gran avance en su desarrollo. Destaca Oculus, que en el año 2016 comercializa el modelo Rift. Dicho modelo será el que dé el pistoletazo actual en la realidad virtual.

Hoy en día la realidad virtual es usada en campos como los videojuegos, la educación, entretenimiento, industria, cultura, fines militares o medicina. Esta última es la que actualmente mayor crecimiento está teniendo y dónde más focos hay.

Para este proyecto, se ha hecho uso de las Oculus Rift S, las cuales se procede a explicar a continuación.

2.2.2. Historia de Oculus

Para hablar de Oculus, es necesario hablar del ingeniero Palmer Luckey. El mismo era miembro activo de distintos foros de discusión de realidad virtual en abril de 2009. Al mismo tiempo, el fundador de id Software, John Carmack, investiga de forma paralela las propuestas que Luckey se encontraba haciendo. Carmack crea un prototipo basado en las investigaciones de Palmer que se presenta en el E3 de 2012, el cual incluía un acelerómetro, pantalla LCD de 5,6 pulgadas, FOV horizontal de 90º y FOV vertical de 110º.

En 2014 se crea el último prototipo del primer casco, conocido como *Crescent Bay Prototype*, el cual mejoraba la ergonomía, la resolución; se añadían auriculares y micrófonos; y permitía movimientos 360º con la inclusión de dos pantallas. La pantalla estaba compuesta por tecnología OLED de 2160x1200 píxeles, a 90Hz y con 110º de FOV. El casco también incluía distintos sensores, como son acelerómetro, giroscopio, magnetómetro y cámara de rastreo [24]. Finalmente, el dispositivo que salió a la venta incluía también dos mandos, conocidos como Oculus *Touch*, que dotaban al usuario de movimientos más naturales. Estos mandos fueron creados en 2016. El funcionamiento es sencillo de comprender, y es que permite tener 6 grados de libertad que se consiguen con un sensor de movimiento a escala de la habitación, haciendo de esta manera que sea más inmersivo. Los primeros prototipos tenían un seguimiento de manos que se captaban mediante sensores colocados en la habitación. Esta era conocida como "*Constellation*". Esa tecnología se fue desarrollando hasta la actualidad, llegando a la tecnología Oculus *Insight*. La misma, consiste en varias luces led en la parte superior de los mandos, que emiten rayos infrarrojos. Estos rayos son captados mediante las seis cámaras que llevan el casco. Una vez capturados, el casco crea una representación tridimensional del entorno y procesa las luces creando un mapa de movimiento de los propios mandos. De esta manera, las gafas son capaces de continuar el movimiento de las manos, interpretar sus movimientos e incluirlos en el entorno virtual que el usuario esté observando sin necesidad de sensores externos [25].

En este proyecto, como se ha comentado anteriormente, se ha hecho uso de las Oculus Rift S. Para usar estas gafas, son necesarios unos requisitos mínimos que permitan hacer funcionarlas correctamente. Para ello se necesita un ordenador, preferiblemente bastante potente que facilite tasas de refresco altas. Así, como requisitos mínimos se necesitan Intel i3-6100/AMD Ryzen 3 1200,

FX4350, 8GB de RAM, Windows 10, Nvidia GTX 1050 Ti/AMD Radeon RX 470, *Display* Port 1.2, USB 3.0 [26].

Las Rift S presenta dos pantallas LCD a 80Hz y con 2560x1440 píxeles, repartidos en 1280x1140 píxeles en cada ojo. Hay que decir que se usan pantallas LED debido a su consumo más bajo y con mejores eficacias. Además, estas gafas dan la posibilidad de representar parte del mundo exterior haciendo uso de las cámaras estereoscópicas que incorporan. Esta tecnología se conoce como *Passthrough*. Sin embargo, aunque esto es posible, este modelo en concreto no tiene tan desarrollado el uso de estas cámaras. Otros modelos, tanto de la misma marca como de otras, han conseguido mejores resultados al respecto.

Una vez explicada la tecnología que envuelve a la realidad virtual, así como todo el contexto histórico de los museos, es el momento de relacionarlos ambos, entendiendo todo el proceso conjunto de ambos elementos. Saber cómo ha evolucionado el arte con las nuevas tecnologías, al mismo tiempo que ha crecido la inmersión en ella, cobra gran importancia en el proyecto.

3 ESTADO DEL ARTE

3.1. Museos y Realidad Virtual. Influencias

Habiendo conocido los antecedentes históricos de ambos mundos, se puede hacer una recopilación de diversos usos que se han realizado en conjunto y explicar cuáles han sido las referencias usadas en este proyecto. El potencial que ofrece esta tecnología en el arte aún está por experimentar y desarrollar. Las posibilidades que se pueden conseguir son inimaginables, pudiendo tener una experiencia inmersiva al mismo tiempo que interactiva. Gracias a ello, se pueden entender las obras desde dentro, modificarlas, alterarlas y aprender más de ellas. Así, tanto artistas como público pueden llevar el arte al siguiente nivel y estirarla a límites insospechados.

Para conocer el fin principal de este proyecto, podría entenderse como una experiencia educativa que difundir el arte y mostrarlo al público de una manera diferente, tomando parte de obras concretas y adentrando al usuario en distintos cuadros. Por tanto, se debe conocer cuál es la situación actual de la realidad virtual en la educación.

Según algunas investigaciones [27], la realidad virtual aporta diversos beneficios en la docencia, tanto en adultos como niños. Hay que recordar que las aplicaciones de realidad virtual siguen la filosofía de los videojuegos, la cual permite crear entornos virtuales de interacción entre el usuario y el mencionado entorno. Así, se puede hacer uso de la gamificación en las aulas o salas de los museos. La gamificación consiste en una técnica de aprendizaje que se aprovecha de los juegos dentro del ámbito educativo. Esto se debe a diversas razones [28], y es que mejora las habilidades a la hora de resolver problemas, ayuda al desarrollo de habilidades cognitivas, mejora la concentración y retención del usuario, otorga a la persona de nuevas formas de pensamiento, añaden un componente divertido proveniente de la experiencia, así como facilitar la interiorización de mejor manera de la información que el usuario experimenta. Esto se puede entender atendiendo a la Figura 1.

Figura 1: Gamificación en juegos [27]

Como ya se ha comentado, es posible entenderlo como un videojuego. En este caso, similar a diversas ideas ya existentes [29], sería necesario proponer puzles o distintos elementos con los que el usuario pueda interactuar. Este tipo de juegos son conocidos como serious games [30]. Los serious games son

aquellos juegos digitales que son usados para educación, formación o aprendizaje. Toman un rol más serio, dónde el fin no es el entretenimiento, sino aprender haciendo uso de experiencias divertidas, como se muestra en la Figura 2.

Figura 2: Puzles en Serious Games [29]

De esta manera, se han tenido en cuenta algunas aplicaciones educativas dónde se usa la realidad virtual. En estos casos, la propuesta era crear entornos virtuales de aulas o laboratorios que permiten a la persona interactuar con objetos y aprender distintas materias como física, matemáticas o química [31]. Otras materias que se pueden impartir usando la realidad virtual es la historia, arte (con su respectiva historia), arquitectura (en la Figura 3), conservación o bellas artes, ya que se pueden recrear edificios, esculturas, pinturas o catedrales [32]. De esta forma, el alumno es capaz de conocer las dimensiones, contexto, autoría o técnicas de las obras, así como aprender de su ejecución o elementos que las componen.

Figura 3: Realidad virtual en arquitectura [32]

Esta función de enseñanza artística gracias a un entorno virtual fue la primera idea que se tomó a la hora de desarrollar este proyecto. Gracias a las referencias encontradas, se pudo tener un contexto educativo que ayudara a entender qué elementos eran necesarios ser incluidos para el consumidor. Se llegó a la conclusión de tener una sala que integrara al sujeto con la pintura y los museos. Por tanto, se van a mostrar qué referencias fueron tomadas en cuenta a la hora de mostrar la inmersión que proporciona la realidad virtual en los museos con casos reales.

La primera referencia que se va a contar es aquella en la que se muestra las catacumbas de San Senatore, en el municipio italiano de Albano Laziale [33]. Se muestra un ejemplo en la Figura 4.

Figura 4: Catacumbas de San Senatore [33]

En este caso, mediante una técnica conocida como fotogrametría, la cual mediante varias fotografías se origina una nube de puntos que posteriormente origina un objeto tridimensional de forma virtual, se consiguió un entorno similar a la cueva real. Posteriormente, se hizo uso de dicho objeto para crear un entorno donde el usuario pudiera experimentar la sensación de inmersión y conocer las pinturas del lugar. De forma similar, se han realizado distintos entornos virtuales de temática industrial, como se ha podido comprobar en algún artículo [34] y se refleja en la Figura 5.

Figura 5: Realidad virtual en ambientes industriales [34]

Con respecto a entornos virtuales relacionados con el arte, encontramos distintas propuestas como son la creación de salas con obras de arte en sus paredes [35], visible en la Figura 6, además de diferentes experiencias donde se exponen brochazos de pintura volando alrededor del usuario [36].

Figura 6: Salas de arte virtual [35]

Todo ello para otorgar al público de un acercamiento al arte aprovechando las nuevas tecnologías y experiencias de usuario. Sin embargo, las referencias más interesantes las encontramos en aquellas experiencias en que se han buscado mayor realismo y que el usuario se sintiera inmerso en un museo [37] o en una propia pintura. Cabe destacar las experiencias propuestas por distintos museos de bellas artes de gran importancia mundial, como son el Museo Nacional del Prado de Madrid [38], el cual

representa una exposición virtual de su sala 39; el Museo Nacional Thyssen-Bornemisza de Madrid realizado varias de sus exposiciones de forma virtual accesible a cualquier público [39]; el British Museum ha recopilado varias de sus salas en entornos virtuales que nos permiten conocer culturas de distintas partes del mundo, recopilando elementos todos los continentes [40]; o la interesante propuesta del MUVA (Museo virtual de arte), el cual ha creado un museo completamente virtual y se presentan diversas exposiciones en su interior. El mayor interés que ofrece es que tanto el entorno como las propias exposiciones, son completamente virtuales [41]. Como muestra, se puede ver en la Figura 7.

Figura 7: Museo en realidad virtual [37]

Tras la visión expuesta de distintos museos virtuales, se puede dar un paso más allá y profundizar en varias propuestas de inmersión que ya existen. Se empezará por la exposición itinerante llamada "Sumérgete en los Impresionistas", que se inició en mayo del año 2021. En este caso, gracias al uso de diversos proyectores, se reflejan experiencias en pantallas que se encuentran alrededor de los usuarios. [42]. En este caso, gracias al uso de diversos proyectores, se reflejan experiencias en pantallas que se encuentran alrededor de los usuarios. De forma similar, se puede hacer referencia al Museo Inmersivo de Tokio [43, 44], el cual experimenta con esta tecnología llevándola hasta los límites y creando experiencias más complejas. Finalmente, relacionado con entornos inmersivos, hay que mencionar la propuesta del museo del Louvre de París [45], la cual muestra una visión distinta de la Gioconda donde se puede contemplar de forma tridimensional, así como un hipotético entorno que acogería la modelo de la obra, lo que resulta interesante ya que conjuga una inspiración real con un entorno inventado que conecta muy bien con la escena. Este último se puede observar en la Figura 8.

Figura 8: Mona Lisa VR [45]

A continuación, se procede a explicar otro de los puntos importantes del proyecto. Y es que para que el usuario se sienta identificado con la escena donde se encuentra, es necesario que en ciertos

momentos existan elementos de dinamismo. En este caso, varios aspectos se han tenido en cuenta como referencias que posteriormente se han analizado para adaptarlos de la mejor manera al fin buscado. El primer modelo que se ha tomado ha sido la propuesta conocida como “The Night Café”, donde se enseña una experiencia que da a conocer el mundo de Van Gogh a través de algunas de sus obras conectadas entre sí [46]. La propuesta más interesante de esta escena viene del propio entorno, ya que todos sus elementos toman referencia del estilo pictórico del autor. Se muestra un interesante enfoque narrativo en el conjunto de los distintos momentos y expande las referencias del autor superando el espacio del lienzo [47]. Así, se puede muestra en la Figura 9.

Figura 9: The Night Café [46]

Siguiendo un modelo similar al anterior propuesto, se puede experimentar la experiencia inmersiva “Dreams of Dalí” [48]. En este caso se sumerge a la persona en el mundo interior de la figura de Dalí, recorriendo una reconstrucción nueva de la obra del pintor, en concreto, la conocida como “Reminiscencia arqueológica del Ángelus de Millet”. Así, se propone un camino guiado que va mostrando un posible entorno imaginado por pintor, atravesando el escenario y criaturas que lo componen.

Por último, con respecto al dinamismo en entornos inmersivos, se ha tenido en cuenta la aplicación Disney Movies VR [49]. En ella, los Studios Disney proponen una sucesión de experiencias de video en 360º. Se reproducen algunas escenas de películas de la compañía formando parte de su entorno. De esta forma, el usuario puede sentir y ver lo que experimentaría el protagonista de la correspondiente escena. Esta aplicación será de interés en el proyecto, ya que gracias a ella se adaptará la idea de vídeo dentro de la experiencia.

Una vez explicada la sensación inmersiva en distintos proyectos, es el momento de mostrar las puramente interactivas. Por lo tanto, en este caso, no se encontrarán motivos de inmersión al respecto. Empezaré hablando de una aplicación encargada en mostrar en Realidad Aumentada (técnica que consiste en integrar objetos virtuales en entornos reales) distintos cuadros y monumentos cuyo fin es el de propagación y enseñanza de arquitectura [50]. La interacción viene por el hecho en que se pueden mostrar los paneles explicativos, así como los distintos datos de los objetos que se muestran. Esta aplicación se presenta en la Figura 10.

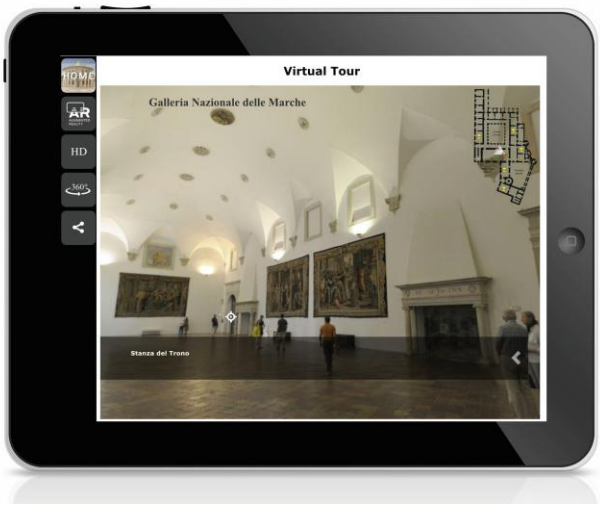

Figura 10: Aplicación de realidad aumentada [50]

Otra propuesta importante que se ha tenido en cuenta a lo largo del desarrollo del proyecto, son diversas exposiciones que se han realizado en el Centro Andaluz de Arte Contemporáneo de Sevilla [51]. Este centro, antiguamente un monasterio cartujano, es hoy día un museo con innovadoras propuestas y exposiciones altamente interactivas, desde la opción de "tocar" burbujas virtuales a través de la sombra del usuario, a pintar en pantallas gracias a luces. Debido a la propuesta que toma el proyecto que nos ocupa, este centro, así como algunas de sus exposiciones, se han tenido muy presente en todo momento.

Tras haber explicado los fines educativos de la realidad virtual en museos y escuelas, la inmersión que ofrece y la capacidad interactiva que pueden presentar algunas exposiciones, es el momento de unirlo todo y llegar a los puntos de mayor relevancia para este trabajo. La capacidad de aprendizaje y estimulación cognitivas que ofrecen las experiencias interactivas e inmersivas convierten este modelo en una de sus mayores propuestas. Por ello, se procede a mostrar las referencias y en qué puntos se encuentran ambas capacidades en conjunto.

En primer lugar, hay que comentar una investigación en la que se ha creado un ambiente virtual que combina arquitectura con elementos industriales [52]. La diferencia con otros aspectos industriales anteriormente comentados es que en este caso se da la posibilidad de interactuar con varios de los elementos que aparecen en la escena. Para ello, se hace uso del mando de la consola Wii, el cual se usa como puntero virtual, mostrado en la Figura 11. Así, es posible cambiar los parámetros de la maquinaria que se puede encontrar en su interior y de esta manera experimentar con los distintos valores.

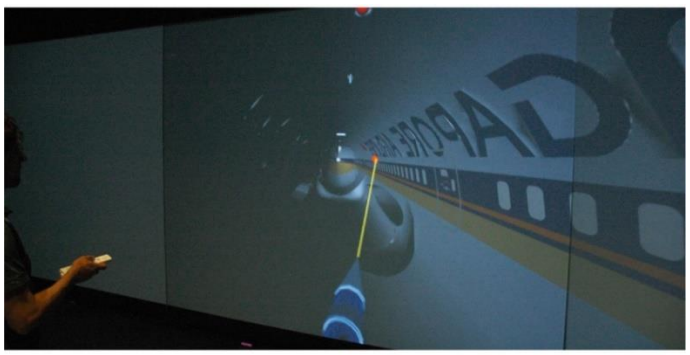

Figura 11: Controladores en realidad virtual [52]

Otra propuesta interesante es aplicar la realidad virtual interactiva a la arquitectura. Se pueden hacer uso de diversos artículos para experimentar con ello, como puede ser el mismo en el que se recrea la Cartuja de Miraflores, en la ciudad de Burgos [53]. En este caso, se ha creado virtualmente un entorno real como son el monasterio y los alrededores que le rodean. Esta aplicación da la posibilidad de poder abrir paneles explicativos, así como recreaciones holográficas dentro del propio entorno.

Uno de los aspectos más interesantes que se pueden encontrar en los museos es el hecho de que exista algún tipo de guía que complemente lo visto con la experiencia. Así, se ha tomado como referencia una propuesta de audioguía implementada en una aplicación [54].

Figura 12: Audioguía para museos [54]

Se da la posibilidad de recorrer varias salas en un museo virtual, en el que la imagen virtual de una persona acompaña en todo momento con sus explicaciones, como se muestra en la Figura 12. Esta idea se ha tenido muy en cuenta, ya que cobra gran importancia en el proyecto final. Aprovechando la oportunidad de añadir sonidos en la escena, así como la opción de interactuar con instrumentos dentro del entorno [55] (visible en la Figura 13), es interesante el uso que se le podría otorgar en este ámbito. El interés en incluir música ambiente, explicación de una audioguía o la interacción sonora con varios objetos, viene de aporta elementos dotados de mayor viveza a las escenas.

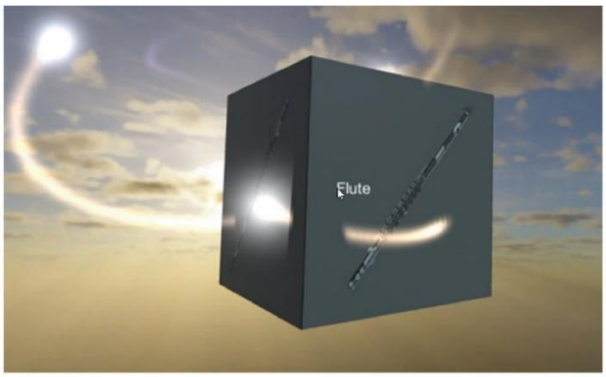

Figura 13: Experiencias musicales [55]

Como se está comentando, la interacción en este tipo de experiencias es muy importante, ya que el usuario es capaz de sentir mejor la sensación inmersiva. Un punto que se ha tenido en cuenta al respecto ha sido la posibilidad de distracción y rehabilitación de aspectos fisiológicos y psíquicos. Esto viene de la idea propuesta por un artículo concreto [56] que se puede observar en la Figura 14.

Figura 14: Rehabilitación con realidad virtual [56]

Habiendo explicado la evolución de la interacción e inmersión, es el momento de uno de los aspectos más interesantes de los mismos. Así, se puede observar la posibilidad de construir un robot dentro del entorno virtual [57] o interactuar con objetos de distinta temática. Un ejemplo de ello se aprecia en la Figura 15.

Figura 15: Creación de robot en realidad virtual [57]

Sin embargo, lo que más interesa es poder añadir estos aspectos a los museos. Esas referencias se pueden encontrar en varios artículos donde se ha intentado incluir distintos experimentos científicos con el fin de que el usuario pueda conocerlos de cerca [27, 58]. Estas experiencias son consideradas como museos, ya sean científicos o artísticos. Disponer de distintas salas agrupadas por temáticas y poder interactuar con sus cuerpos, provoca sensaciones muy completas en el usuario que lo experimenta. En la Figura 16, se recrea una de estas salas, la cual podría equipararse a un parque de atracciones de temática científica.

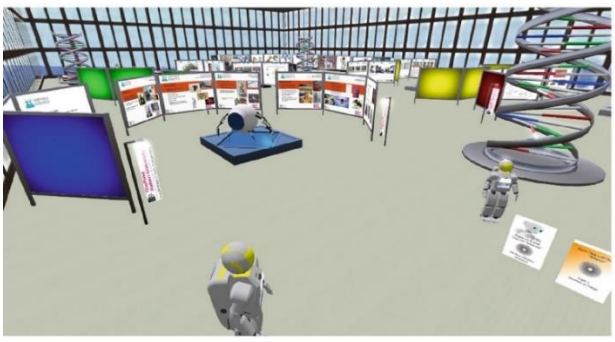

Figura 16: Museo futurista [58]

Finalmente, es necesario comentar algunas aplicaciones que han influenciado mayoritariamente el proyecto completo. En este caso, es necesario comenzar por una de las primeras experiencias que la mayoría de público siente la primera vez que prueba las gafas usadas (Oculus Rift S, mencionadas anteriormente). En concreto, la aplicación "Oculus: primer contacto" [59]. La misma, muestra el interior de una caravana en la que un robot acompaña a la persona en todo momento. Este robot va facilitando distinto cartuchos que, al introducirlos en una máquina, da la opción de poder trabajar con los mandos y conocer el funcionamiento de sus botones. Una imagen de esta aplicación se puede visualizar en la Figura 17. Esta fue la primera influencia que se ha tenido a la hora de desarrollar el proyecto, intentando conseguir una experiencia igual de inmersiva.

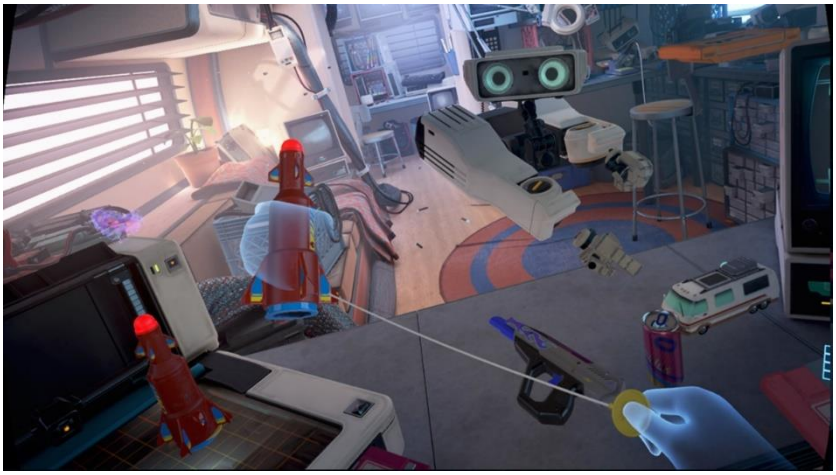

Figura 17: Oculus Primer Contacto [59]

La siguiente aplicación que se tuvo en cuenta fue un juego llamado Kaisuo [60]. Este consiste en un juego montado en una sucesión de puzles, al estilo "scape room", ambientado en la cultura japonesa. Es interesante la propuesta de este videojuego, ya que combina perfectamente la inmersión, la interacción, música ambiente y escena. El conjunto de todos ellos dota al usuario de una breve, pero intensa experiencia donde al accionar un elemento, se activa otro y así se va desarrollando el motivo completo.

Posteriormente, se explica uno de los desarrollos más complejos analizados y estudiados. En este caso, hablamos de una práctica en realidad virtual en la que es posible pintar en el aire del entorno virtual. La misma, conocida como Tilt Brush [61], lleva los límites de la pintura a entornos hasta el momento solo imaginados. La capacidad de dibujar en el aire adentra en la pintura inmersiva, ya que es posible contemplar aquello dibujado desde cualquiera perspectiva. Este diseño, desarrollado por Google, ha sido utilizado en juegos, cortos de animación o entornos para otras experiencias de realidad virtual. Un ejemplo realizado por el artista Wesley Allsbrook se puede muestra en la Figura 18.

Figura 18: Tilt Brush [61]

Enlazando con el punto anterior, hay que mencionar una plataforma nativa de HTC conocida como HTC Vive Arts [62]. Esta sección de la empresa es la encargada de almacenar gran cantidad de trabajos en los que se combinan los dispositivos de la marca y el arte. Por tanto, se puede entender como un museo virtual de experiencias y elementos usados en sus dispositivos. La filosofía de esta sección es la preservación del patrimonio artístico y cultural de entornos virtuales HTC, aplicando técnicas novedosas, así como contribuir con la expansión del conocimiento y diversión de la cultura, tanto en los museos como hogares.

Por último, como combinación de todo lo explicado, se va a explicar una de las aplicaciones más completas en cuanto a inmersión e interacción se refiere. Me refiero a The Lab [63], desarrollada por la plataforma Steam. Aunque se puede considerar un videojuego, va más allá, ya que esta ofrece una completísima experiencia interactiva de diversas temáticas. Y es que se permite que el usuario se adentre en un laboratorio que permite movimiento por distintos mundos, dando la opción al usuario de tener distintas sensaciones en cada uno de ellos. Así, se puede mover en la cima más alta de una montaña, explorar una llanura acompañados de un perro robot, sentirse en la edad media lanzando flechas a los enemigos o luchar en escenas más futuristas contra robots o rayos láser. La sensación inmersiva que ofrece este producto, así como la cantidad de momentos que se pueden encontrar, hace que esta sea una de las experiencias más completas actualmente en la realidad virtual. Es una gran propuesta que ha servido de inspiración durante la creación de este proyecto.

Figura 19: The Lab [63]

Llegados a este punto, se puede dar por concluido el estado del arte de los museos y la realidad virtual, así como las influencias que han otorgado en conjunto ambos elementos, teniendo en cuenta aplicaciones, motivaciones en museos o experiencias inmersivas. A continuación, se procederá a explicar cuál es la propuesta realizada frente a lo ya existente, qué elementos nuevos se incluyen y de qué manera se ha alcanzado. La interacción y la inmersión, como se ha comentado anteriormente, es el principal fin, tratando de encontrar nuevos puntos de vista del arte y la pintura, capaces de ampliar nuevos horizontes.

3.2. Conclusiones del Estado del arte y aportaciones

Como se ha podido comprobar, existen multitud de aplicaciones donde el arte y la realidad virtual están presentes. Desde la capacidad de pintar en el aire, concepto que se tomará muy en cuenta para este proyecto, hasta visitas virtuales en salas completas de museos. Se ofrece la posibilidad de formar parte de los cuadros que a diario son expuestos en salas de museos, haciendo que el usuario viva desde dentro las distintas intenciones que los autores querían conseguir.

Respecto a estos desarrollos, aún quedan grandes caminos que recorrer, sin embargo, el interés que está suscitando en especialistas en bellas artes, artistas o creadores, sumado a los perfiles técnicos encargados en realizar estas aplicaciones, hace que existan experiencias más elaboradas y atractivas enfocadas al gran público.

Se ha querido aportar un punto de vista diferente a este mundo, empezando con esta pintura en concreto. Y es que, debido a los elementos que contienen, así como las infinitas posibilidades que ofrece tanto de añadir como de mejora, hace que el público se encuentre con uno de los grandes avances actuales en lo referido a arte, cultura y enseñanza. Desde la interacción con mundos reales hasta mundos donde tan sólo existen las mentes de sus creadores.

4 METODOLOGÍA

En este apartado se procede a explicar los programas que se han usado en la realización de este proyecto: 3DS Max y Unity; las partes que lo componen y el procedimiento que se ha llevado a cabo mientras se desarrollaba hasta conseguir la aplicación final.

4.1. 3DS Max

Este es un programa perteneciente a Autodesk que se emplea para modelar, texturizar, animar y renderizar elementos tridimensionales con el fin de poder se incluidos en campos como animación, videojuegos o simulaciones. Gracias a este programa, se pueden conseguir objetos en 3D de gran calidad profesional. Con respecto a este proyecto, se ha usado simplemente para la creación del cuadro y montaje de algunas partes en objetos muy concretos.

Los conocimientos de este programa han sido ampliamente adquiridos durante la asignatura Técnicas de Animación 3D que se enseña en el Grado de Ingeniería de las Tecnologías de las Telecomunicaciones [64].

4.1.1. Presentación

En este punto, se procede a explicar la interfaz del programa, comentando las partes que lo componen. Consta decir que se ha realizado en la versión para estudiantes de 3DS Max, la cual es gratuita siempre que se justifique el que usuario cursa algún estudio.

Comienzo mostrando la ventana principal, la cual se muestra en la Figura 20:

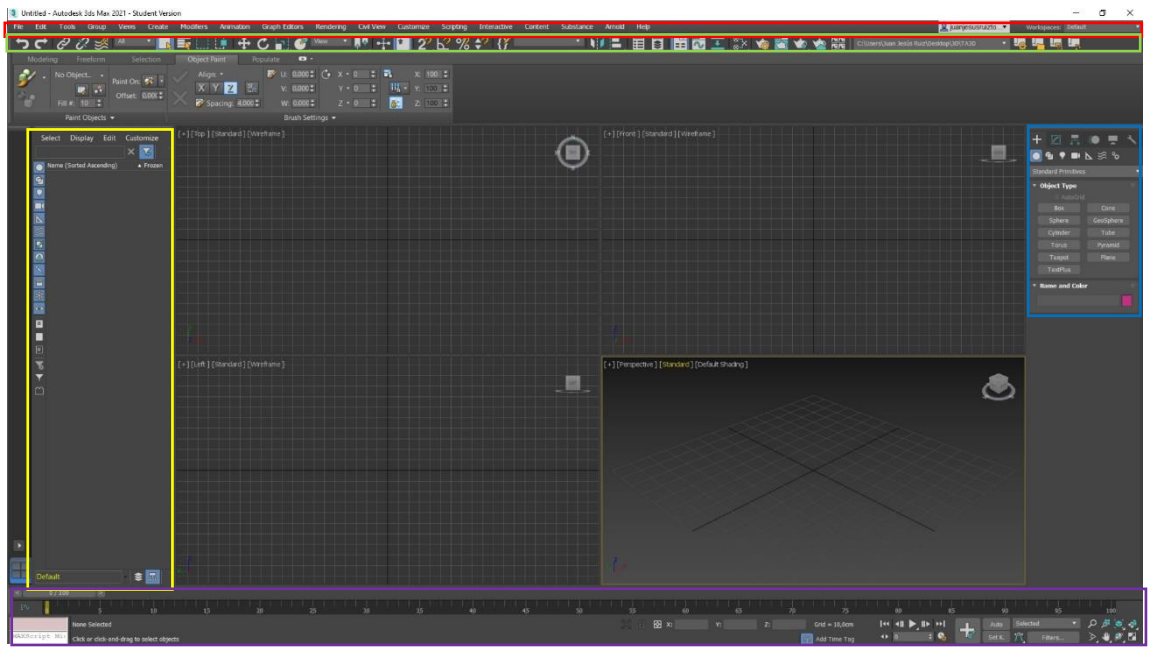

Figura 20: Interfaz 3DS Max [65]

En la parte central de la ventana principal, se presentan cuatro ventanas con mallas. Esta zona es conocida como *viewport* o *ventana gráfica* (aunque se usa más el término en inglés), y es la encargada de mostrar la visualización del trabajo que se esté realizando. Se presenta en cuatro perspectivas diferentes, como son frontal, lateral, alzada y otra basada en la escala y volumen completos del objeto. Aun así, las perspectivas de todas estas ventanas se pueden modificar para poder ver la que más interesa en cada momento. Estas opciones de visualización se encuentran en la parte superior izquierda de cada ventana (entre corchetes), pudiendo cambiar la perspectiva, el mallado del objeto o el ángulo de visión. Por contra, en la parte superior derecha, se puede observar un cubo tridimensional que permite rotar el objeto y ver aquellas secciones que interesan. Sin embargo, para hacer esto, se hace uso de la rueda central del ratón más la tecla *Alt* del teclado, ya que es más intuitivo y cómodo. En la zona baja izquierda de la ventana, unos ejes muestran la posición de referencia del objeto.

En la parte superior de la ventana principal, señalada de rojo en la Figura 20, se encuentra la barra de menú con distintas opciones.

Se van a enumerar explicando el uso de cada una de ellas:

a) *File*: esta ventana da la posibilidad de crear nuevos proyectos o importar otros existentes.
b) *Edit*: permite modificar lo que se tenga montado. Además, se puede cambiar la dimensión de los objetos, aunque existen accesos directos justo bajo el menú que se está explicando.
c) *Tools*: herramientas de trabajo del programa. Se pueden hacer uso de estas herramientas en los modificadores.
d) *Group*: permite seleccionar y agrupar los objetos como más convenga, facilitando el trabajo de la persona encargada, teniendo un proyecto más limpio.
e) *View*: da la opción de modificar el *viewport* y las perspectivas.
f) *Create*: muestran los objetos que se pueden crear, como pueden ser líneas, objetos, partículas o líquidos. Esto se puede hacer también en el panel derecho que se puede ver en la imagen.
g) *Modifiers*: este aplica los modificadores que se le pueden aplicar a los objetos. Al igual que anteriormente, se pueden encontrar en la ventana derecha de la interfaz.
h) *Animations*: añade modificadores de animaciones aplicables a personajes y objetos.
i) *Graph editors*: da la opción de editar principalmente animaciones.
j) *Rendering*: se muestran las opciones de renderizado, ya sean tipo de iluminación, fondos… Hay que tener en cuenta si se renderiza vídeo o imágenes. En este caso, se ha usado Arnold para dotar de un mejor realismo los elementos creados.
k) *Civil view*: vista civil del proyecto.
l) *Customize*: permite personalizar la interfaz acorde a la comodidad del usuario.
m) *Scripting*: permite controlar parámetros y controladores mediante programas que realicen distintas funciones.
n) *Interactive*: da interactividad.
o) *Content*: se encuentran las librerías del programa.
p) *Arnold*: información respecto al tipo de renderizado. Esta ventana aparece debido a que se ha seleccionado este renderizado concreto.
q) *Help*: se puede buscar el funcionamiento de ciertas dudas que el usuario tenga acerca del programa.

Bajo el menú ya explicado, remarcado con color verde en la Figura 20, existen distintos botones que permiten modificar diversos parámetros como son el tamaño, rotación o tipo de unión entre objetos. Estos paneles son muy interesantes para los usuarios de este programa, ya que facilita el trabajo a la

hora de cambiar las especificaciones de los objetos. Cabe destacar que estos botones se pueden activar mediante comandos de teclado, facilitando y dando mayor velocidad en el flujo de trabajo del desarrollador.

En esta zona, encontramos un botón concreto que nos permite alterar la apariencia de los objetos a través de los materiales. Para encontrar este botón, se debe buscar aquel que tiene dos círculos que convergen en un cuadrado. Es el que se encuentra a la izquierda del botón que tiene una tetera con un engranaje amarillo sobre ella. Al pulsar el botón o apretar la tecla *M* en el teclado se abre un panel que permite cambiar la apariencia de todos los objetos, teniendo la posibilidad de tener el mismo material para todos los objetos o uno diferente para cada elemento, dependiendo de las necesidades de la aplicación. También es posible que dentro de un mismo material se puedan tener distintos aspectos, pudiendo dotar los objetos de mayor realismo.

A continuación, explico la ventana lateral izquierda de la imagen, de color amarillo en la Figura 20. Esta ventana muestra los objetos y grupos añadidos a la escena. Se pueden marcar y desmarcar los botones correspondientes a los objetos, los cuales se harán visibles o no, respectivamente. Es visible en la Figura 21.

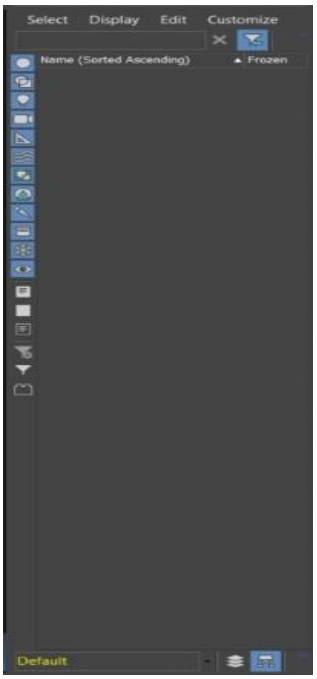

Figura 21: Ventana lateral izquierda 3DS Max

Por debajo del *viewport*, en color morado en la Figura 20, se ve una línea de tiempos y distintos botones. Estos hacen referencia a los parámetros que se pueden controlar con respecto a la animación, como pueden ser pausar, ejecutar, modificar los *keyframes* (estos son marcadores que se le añaden a los *frames* en la línea de tiempos para colocar las posiciones que interesan en el tiempo adecuado) o cambiar el límite temporal de la animación. Esta área es muy importante para la corrección y adición de animaciones. La Figura 22 muestra la zona de animación.

Figura 22: Línea de tiempos 3DS Max

La ventana que se muestra en el lado derecho, junto al *viewport* y de color azul en la Figura 20, es la encargada de crear los objetos, modificarlos, cambiar la herencia o detalles de las animaciones. Esta ventana se puede ver en la Figura 23. A pesar de que los cambios se pueden hacer directamente desde las pestañas del menú, el hecho de tener esta ventana flotante junto al *viewport* hace que el flujo de trabajo sea más rápido y fluido, y es que se puede entender como un acceso rápido para las modificaciones de los objetos.

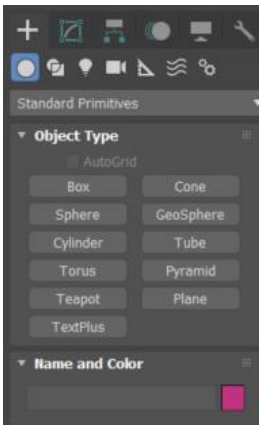

Figura 23: Ventana lateral derecha 3DS Max

4.1.2. Trabajo realizado en 3DS Max

Se procede a mostrar lo realizado en este programa. Dado que se recurrió a una pintura de Piet Mondrian, el modelado de los objetos fue muy sencillo, ya que la pintura empleada está formada por cuadrados. La obra realizada es “Composition A”, la cual se presenta en la Figura 24.

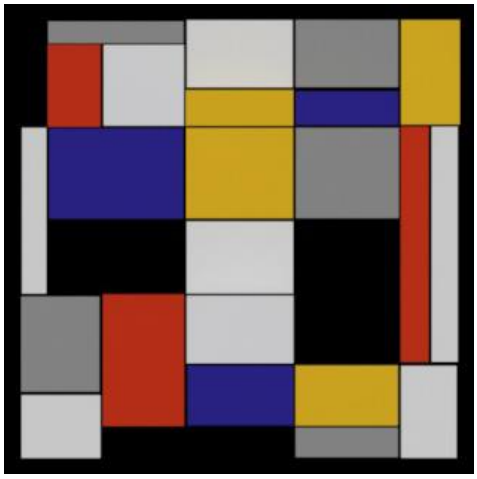

Figura 24: Representación digital del cuadro “Composition A”, Piet Mondrian

Ya que la idea era conseguir un cuadro tridimensional e inmersivo, era necesaria la pintura original como modelo [66, 67], que se encuentra en la *Galleria Nazionale d’Arte Moderna e Contemporanea*, en la ciudad italiana de Roma. Para realizar el modelado del cuadro, se empezó añadiendo un plano vertical a la escena con la imagen del cuadro. Así, se tiene un “fondo” que es usado como plantilla a la hora de construir proporcionalmente todos los cuadrados. Esto se muestra en la Figura 25.

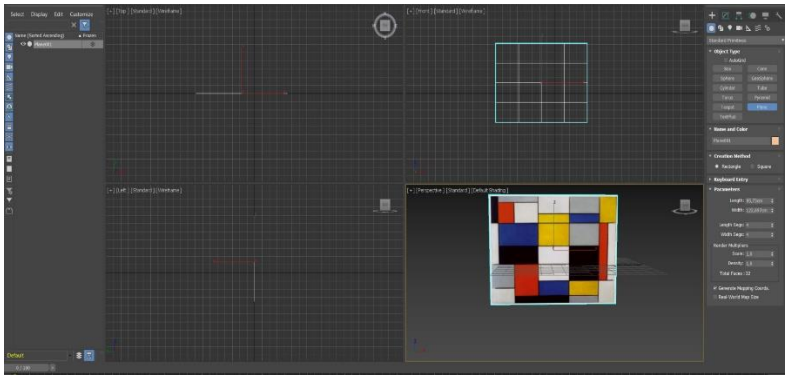

Figura 25: Plantilla de referencia

Una vez conseguido el plano de referencia, en la ventana derecha se seleccionó la pestaña “*Box*” y se fueron creando los distintos cubos, montando de esta manera el cuadro original. Debido a que la pintura real es en dos dimensiones, se pudo tener profundidades imaginarias y colocarlas de la manera que mejor interesasen. Por ello mismo, se decidió dotar a cada cubo una profundidad distinta, dotando a la escena de mayor sensación inmersiva. Posteriormente, en al panel de materiales y se asignó a cada cubo su correspondiente color. Esto se muestra en la Figura 26. Para ello, se hizo uso de la herramienta “cuentagotas” y se tomaron los colores de la forma más realista posible para posteriormente se aplicados. En la siguiente imagen, en concreto en la Figura 27, se puede ver el resultado final del cuadro desde la perspectiva lateral, pudiendo entender mejor la propuesta de distintas perspectivas que anteriormente mencionadas. Para ello se ha añadido un fondo celeste que facilite un contraste y todos los elementos se puedan observar de la mejor manera. Así, es visible la

idea que se quería conseguir a la hora del modelado, teniendo en cuenta no solo la altura o posición, sino también el fondo de cada objeto y el tamaño que ocupa en el espacio.

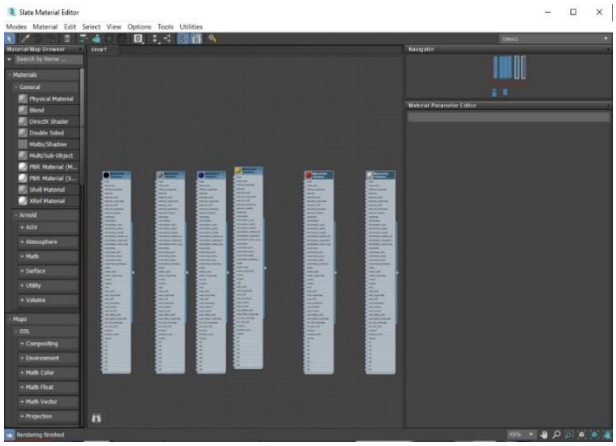

Figura 26: Materiales asignados

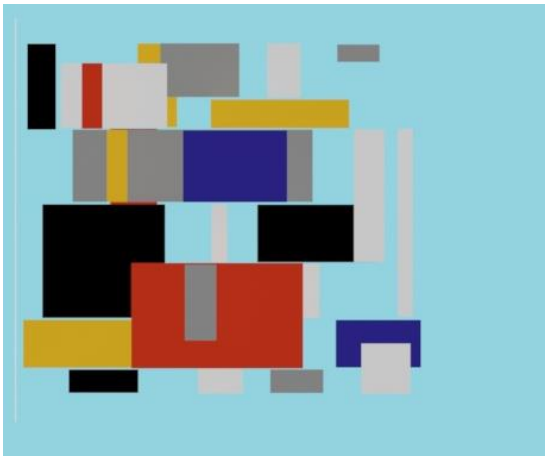

Figura 27: Lateral del cuadro en 3D

Finalmente, se ha decidido añadir una pared trasera que ayudara al espectador a darle sensación de que se encontraba frente a una pintura. Para ello, se colocó un plano vertical, similar al que se había tomado como plantilla al fondo del cuadro. A diferencia de la plantilla, se tomó un color plano para el objeto entero. Posteriormente, se seleccionó la opción *Convert to Editable Poly*, como se muestra en la Figura 28, que permite modificar los parámetros del objeto.

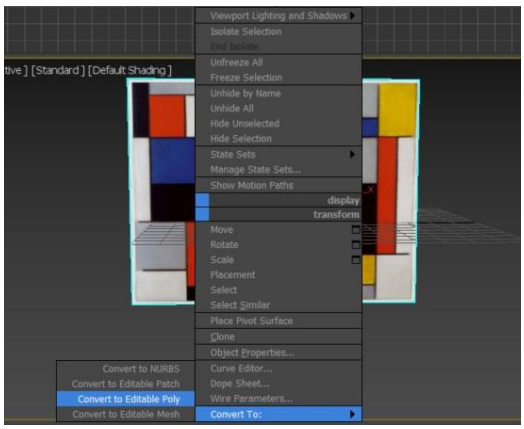

Figura 28: Panel para convertir objeto

De esta manera, se pudo acceder a cada cara de la malla que compone el plano. En la ventana derecha se accedió a las propiedades del objeto y se marcó la opción de polígonos (*polygon*). Con este botón marcado, se seleccionaron los polígonos del plano que interesaban, como son los que eran tapados por los cubos desde una perspectiva frontal. Una vez estaban todos seleccionados, se bajó hasta la sección de *Surface Properties* y en *Set ID* le asignamos el valor 2. Todo esto se puede ver de forma gráfica en la Figura 29.

Figura 29: Valor de ID asignado al polígono

De forma similar, se seleccionaron los polígonos restantes del plano, los cuales corresponden con los bordes de este y que son visibles por los cubos desde la vista frontal de la escena y, siguiendo el mismo procedimiento, se le asignó valor de *Set ID* 1. Tras realizar estas acciones, se consigue un plano con sus correspondientes polígonos separados por distintos valores de ID. Habiendo separado la ID de los polígonos, se asignaron los colores al fondo del cuadro. Para ello, se hizo uso de la librería conocida como *Multi/Sub* Object, que se encarga de dotar distintos elementos de un objeto de diferentes colores. Así, al 1, correspondiente a los polígonos con ID 1, se le estableció el color blanco. Se hizo lo mismo con el 2 y el color negro, como se puede comprobar en la Figura 30.

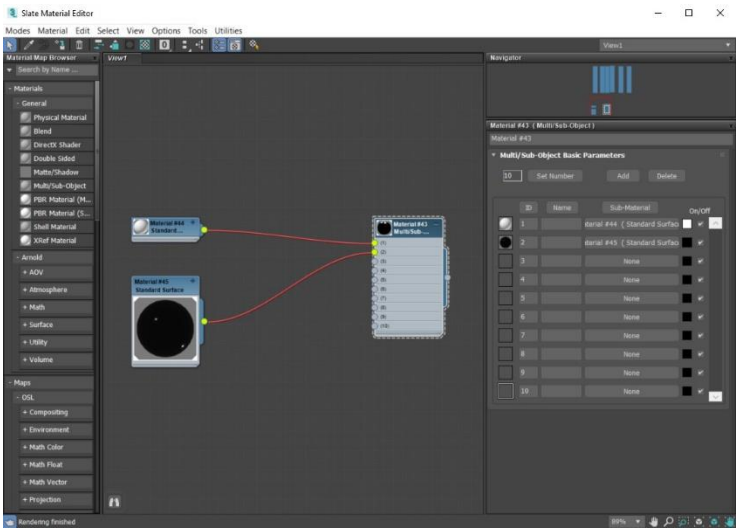

Figura 30: Materiales *Multi/Sub* en la pared de fondo

Tras todo ello, se llegó al resultado final, donde se presenta el cuadro modelado en formato tridimensional con una pared que también será usada en el proyecto final. Al igual que se ha comentado anteriormente, se ha añadido un plano por detrás, con el fin de conseguir una mejor visualización. En la Figura 31 se pueden ver todos los elementos, la pared trasera y la conexión que existe entre todos los elementos.

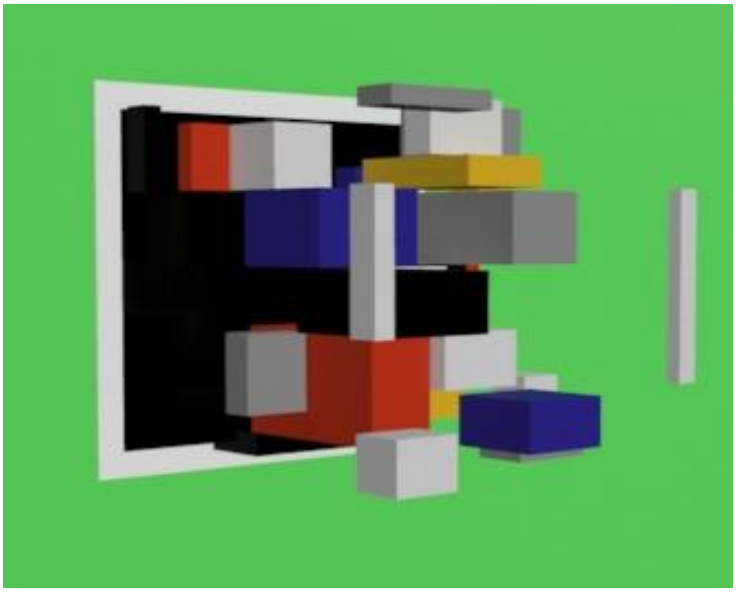

Figura 31: Modelado completo del cuadro tridimensional

4.2. Unity

Este programa es un motor de juegos que se emplea en el desarrollo de juegos, visualizaciones y animaciones 3D para diseñadores, artistas y desarrolladores en tiempo real. Ofrece soporte a diferentes plataformas, como son PC, IOS, Android o gran variedad de consolas [68]. Con el fin de realizar aplicaciones interactivas e inmersivas, este programa soporta paquetes que ayudan a los desarrolladores en la creación de experiencias de realidad virtual, aumentada o mixta. Además, ofrece una gran calidad visual, haciendo que el cliente pueda tener una mejor experiencia.

Unity es un programa creado por la compañía danesa Unity Technologies, fundada en 2004 por David Helgason, Nicholas Francis y Joachim Ante. Principalmente el programa fue creado como motor de juegos de su propia compañía, cuyo fin era crear su primer videojuego. Sin embargo, este juego no tuvo éxito alguno, pero detectaron el gran potencial del motor de juegos que habían implementado y que se encontraban usando. A partir de este momento, deciden comercializarlo y centrarse en mejorar y añadir herramientas de desarrollo, facilitando el trabajo de todos los profesionales.

El acceso a una interfaz sencilla, aunque completa, sumado al hecho de poder incluir elementos tridimensionales en el espacio, hace que su uso se expanda en multitud de compañías y trabajos. Destaca el auge conseguido en 2008 gracias al iPhone, y es que la plataforma añadió gran cantidad de aplicaciones en los dispositivos cuyo motor de juego usado era el que aquí comentamos. Para ello, este programa hace uso del lenguaje de programación C#.

El programa es gratuito siempre y cuando no se supere la cantidad de 100 millones de dólares estadounidenses de ingresos o fondos recaudados en los 12 meses anteriores [69].

4.2.1. Presentación

Una vez iniciada la aplicación Hub, se nos muestra el panel de la Figura 32. Este Hub es una herramienta de Unity que permite gestionar los inicios del programa y permite elegir el tipo de aplicación a desarrollar, facilitando algunos componentes de forma que sea óptima la ejecución. Esta herramienta se muestra en la Figura 32.

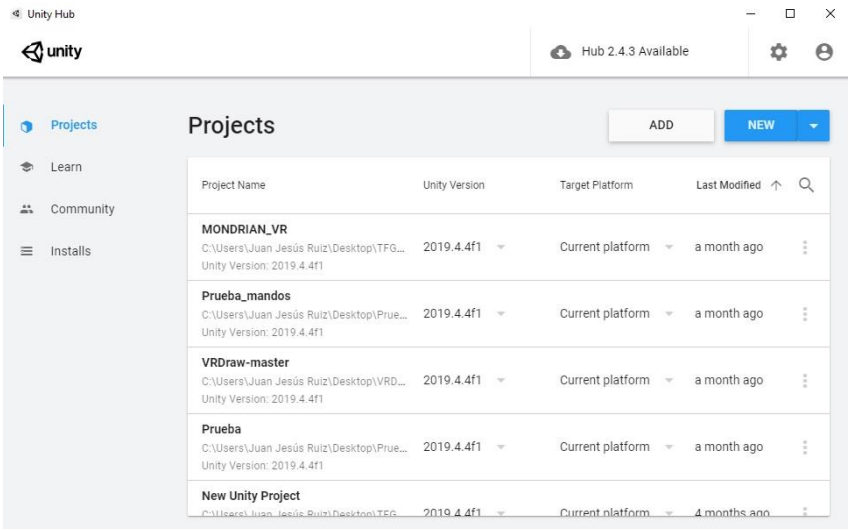

Figura 32: Inicio Unity

Se procedió a crear un proyecto tipo 3D, como se ve en la Figura 33.

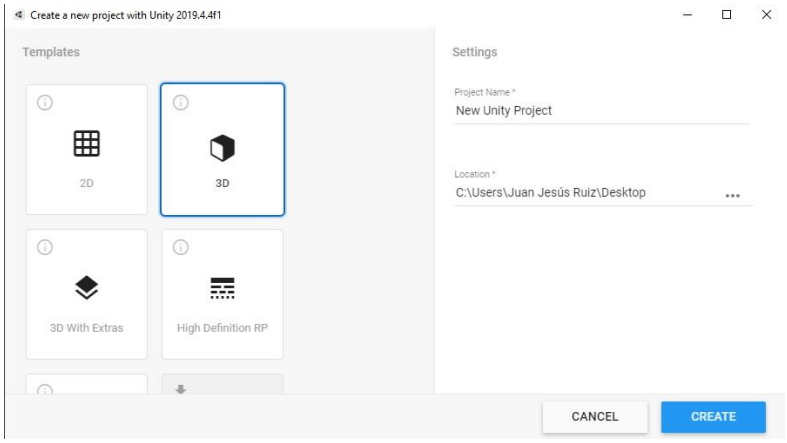

Figura 33: Selección del tipo de aplicación

Al proyecto se le asignó el nombre de “MONDRIAN_VR”, fácilmente identificable en todo momento.

Al igual que con 3DS Max, se va a explicar la interfaz de este programa y qué se puede encontrar en sus pestañas y ventanas. Tomamos como referencia la Figura 34.

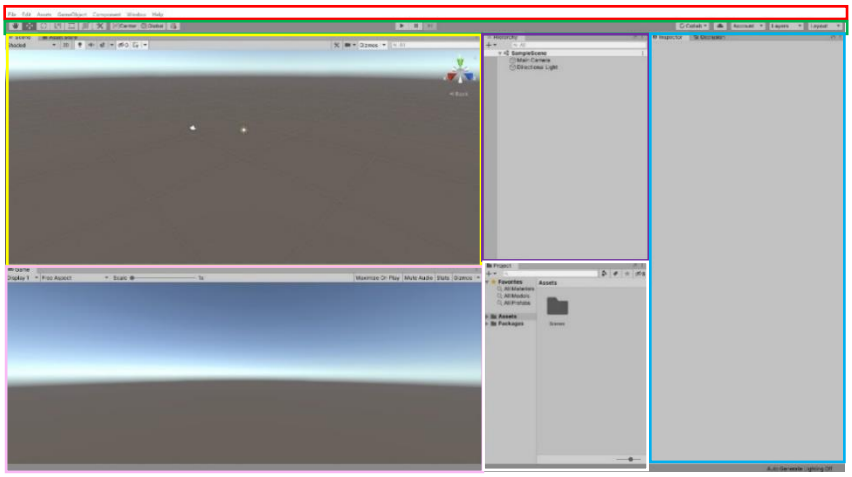

Figura 34: Interfaz de Unity

Para ello, se empieza por la parte superior, remarcada de rojo. En ella hay un menú con una sucesión de distintas pestañas. Estas son:

a) *File*: se da la opción de abrir, crear, guardar o implementar la aplicación final a desarrollar.
b) *Edit*: permite editar, copiar o pegar elementos dentro de la escena. Además, se puede ejecutar la aplicación que estemos creando.
c) *Assets*: aquí se encuentran las carpetas que se pueden ver en la zona media baja, a la derecha de la Figura 34. Es una pestaña que permite explorar los archivos y tenerlos perfectamente ubicados.
d) *GameObject*: esta pestaña permite crear los elementos que vayan a usarse, sean objetos 2D o 3D, luces, cámaras, efectos o el resto de los aspectos necesarios. Sin embargo, por comodidad se suele hacer uso del *click* derecho del ratón, ya que facilita el trabajo a la hora crear nuevos objetos por ser más rápido.
e) *Component*: los objetos de la escena pueden estar unidos a diversos modificadores que cambian sus parámetros. Estos modificadores se encuentran en esta pestaña, así como en la

ventana derecha llamada Inspector. Se suele hacer uso de esta ventana, la cual tiene un manejo más cómodo para el desarrollador.

f) *Window*: permite ajustar los parámetros de visualización de la pantalla de inicio.
g) *Help*: ofrece ayudas de manuales, foros o similares respecto al programa.

Todas estas pestañas están mostradas en la Figura 35.

Figura 35: Menú superior Unity

Bajo este menú se ubican distintos botones, correspondientes a la Figura 36.

Figura 36: Botones superiores Unity

Gracias a los botones de la zona izquierda, se puede cambiar la posición, rotación, escala o pivote de los objetos. En la zona central, se encuentran el menú de play y pausa del videojuego. En la zona derecha, permiten el acceso a los servicios de Unity Cloud, visibilidad de las capas y menú de layout [70].

Posteriormente, bajo la barra de herramientas explicada, encontramos tres zonas distintas. Comenzaremos la explicación empezando por la zona izquierda, de color amarillo. En este punto se encuentra la *Scene View* o *View Port*. De forma similar a 3DS Max, en esta zona se ubican los elementos y objetos que se vayan añadiendo a la escena. Por defecto aparecen una cámara y luz que actuará como sol del entorno. A pesar de ello, pueden ser eliminadas o modificar sus parámetros si así conviene en el desarrollo. En esta área se pueden mostrar dos aspectos bien distintos. En el primero de ellos, nombrado como *Scene*, estarían los elementos ya descritos, mientras que en la pestaña *Asset* Store se muestra el acceso a la tienda de Unity, que da la opción de incluir complementos o librerías que interesen. Los botones de la escena hacen referencia a aspectos como el tipo de visionado, luz o audio. En este proyecto no se tendrán en cuenta.

Con respecto a la ventana de la propia escena, además de la luz y la cámara mencionadas, existe una malla que indica como referencia en qué punto se encuentra el suelo. También se puede ver en la parte superior una zona iluminada de color celeste, y es que en este caso lo que se pretende imitar es la iluminación real del cielo. En la esquina superior derecha, un eje con tres colores muestra la posición local de la escena, que puede coincidir o no con la relativa de los objetos incluidos. Se puede ver la escena completa en la Figura 37.

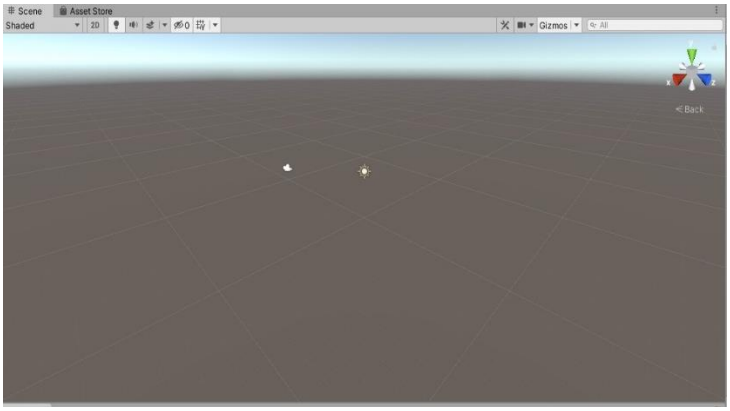

Figura 37: *Scene View*

Junto a la escena, en la parte derecha, se encuentra la ventana de *Hierarchy*. En la Figura 34 se muestra en color morado. Aquí estarán todos los elementos que pertenezcan a la escena. Se puede modificar su jerarquía y hacer que un objeto dependa de otro (siendo hijo del padre) o hacerlos visibles o no dentro de la escena. Se puede resumir como una lista de los objetos presentes y la relación entre ellos. En la Figura 38 se muestra la ventana de *Hierarchy*.

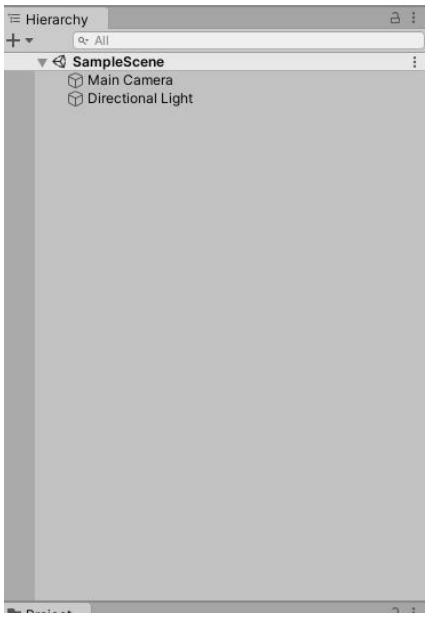

Figura 38: Panel *Hierarchy*

Siguiendo por la zona derecha, se muestra el panel de *Inspector* y *Occlusion* [71], visible en la Figura 39. En este, de color celeste, aparecerán todas las propiedades del objeto u objetos seleccionados. Permite modificar todos los parámetros de estos, pudiendo adaptarlos a los intereses buscados.

Figura 39: *Inspector*

A continuación, se presenta la ventana de *Game*, que se ha dispuesto con el color rosa. Aquí se muestra el juego propiamente a través de la cámara. Así, aquello que enfoque la cámara, ya sea controlada por el programador o por el usuario, será mostrado en esta ventana. Los botones incluidos hacen referencia a la relación de aspecto de la imagen, la cámara con que se verá, escala o estado del audio, teniendo en cuenta si está activo o no. Todos ellos se pueden ver en la Figura 40.

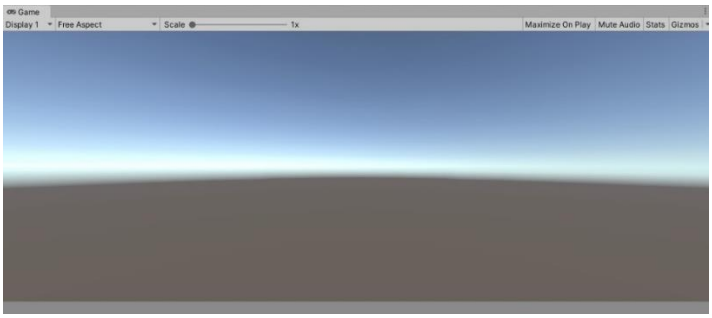

Figura 40: Ventana *Game*

Por último, con el color blanco, se puede ver la ventana Project, dónde se muestra el explorador de archivos con sus respectivas carpetas y elementos externos que se vayan usando, como pueden ser ficheros de texto, objetos 3D externos, copias de objetos en las escenas o materiales. Esto se puede ver en la Figura 41.

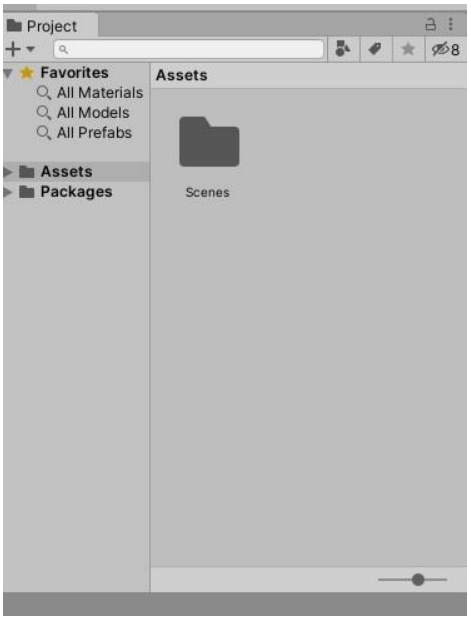

Figura 41: Ventana *Project*

4.2.2. Trabajo realizado en Unity

Dado que Unity no ofrece soporte para realidad virtual de forma predeterminada, es necesario asignar en el programa las herramientas de Oculus. Con la ayuda de distintas propuestas [72, 73, 74], se ha incluido lo necesario para tener un entorno de realidad virtual compatible con Unity, como pueden ser las manos virtuales o los paquetes de funcionamiento propios de Oculus.

Se han usado los SDKs de Oculus, correspondientes a la Oculus *Integration* [75] y Oculus *Avatar* [76]. El paquete de integración también puede ser descargado a través de la propia *Asset Store* de Unity, como se puede ver en la Figura 42. Estos paquetes dan la posibilidad tener interacción con el ecosistema Oculus sin ningún tipo de problemas. Este dará las herramientas necesarias para conseguir la inmersión, interacción y las distintas acciones que mejor se adecuen a la aplicación a desarrollar, consiguiendo una mejor experiencia de usuario. Además, gracias a las manos virtuales (avatar) desarrolladas por la compañía, junto con los SDKs anteriormente mencionados, se consigue que el usuario sienta mayor inmersión en la escena, pudiendo interactuar con sus propias manos en lugar de los modelos de los mandos.

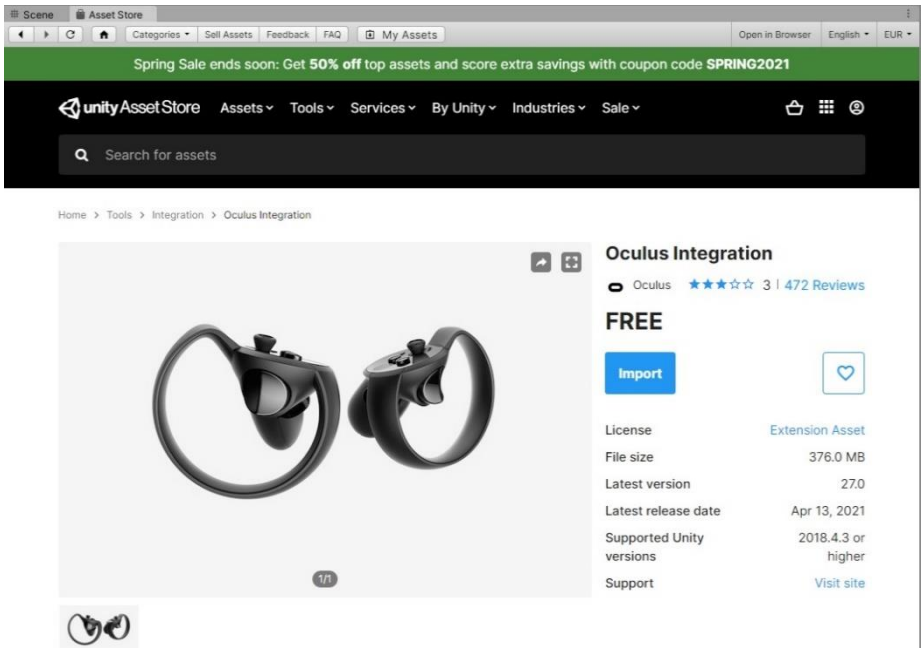

Figura 42: *Oculus Integration Package*

Se han tomado como referencia distintos tutoriales [77, 78] encontrados para implementar correctamente los paquetes descargados.

Tras haber importado todos los ficheros en el entorno de Unity es necesario activar ciertos elementos de forma que el programa "entienda" que se van a trabajar con aplicaciones de realidades extendidas. Para ello, en la pestaña superior *Edit*, se selecciona *Project Settings*, como se indica en la propia web de Unity [79]. Así se puede ver en la Figura 43.

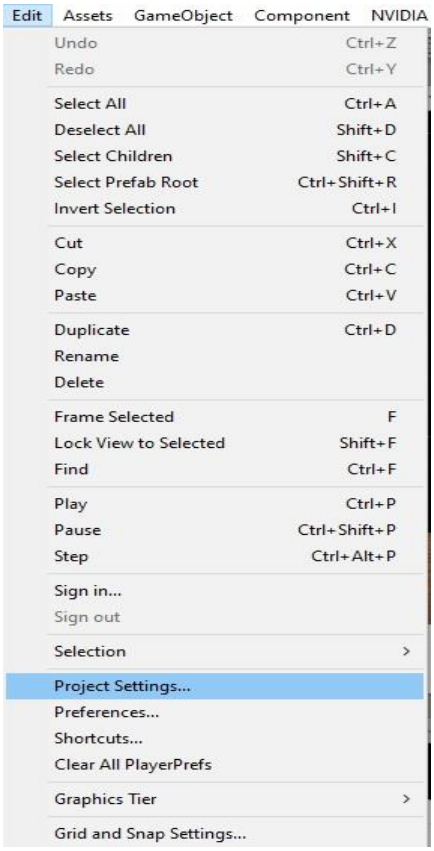

Figura 43: *Project Settings*

Se abrirá una ventana donde se pueden cambiar distintos parámetros del programa. En concreto, la pestaña *Player*, dónde aprovecharemos para asignarle un nombre a la aplicación, la autoría y la versión. Aunque se podría realizar en este momento, al final del desarrollo se asignará también un icono como imagen de la aplicación, que en este caso será el cuadro "Composition A", cuya imagen ya usamos anteriormente. La Figura 44 nos muestra la ventana.

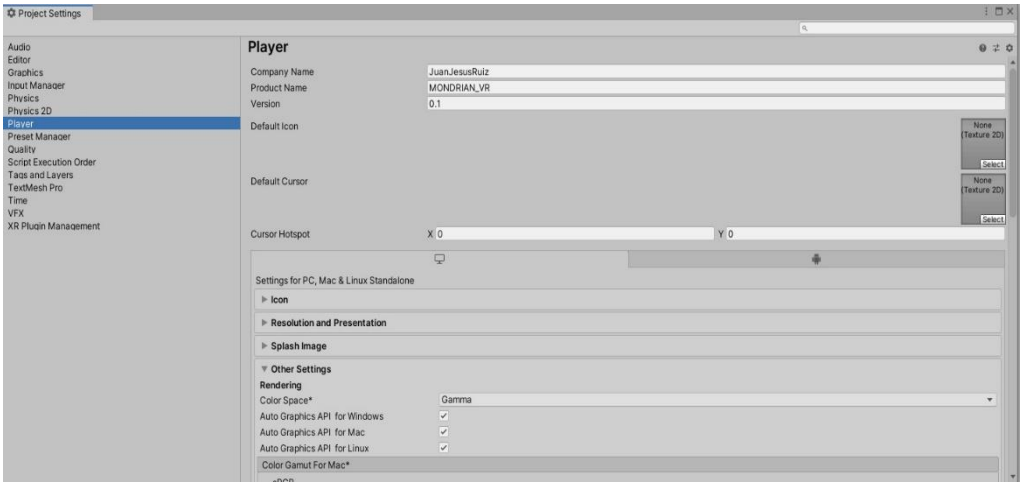

Figura 44: *Player*

Una vez abierta la pestaña *Player*, hay que bajar hasta encontrar la sección llamada *XR Settings*, la cual permite habilitar en Unity el uso de aplicaciones para realidades extendidas (virtual, aumentada y mixta), pudiendo elegir el ecosistema que se va a usar. Se marca el cuadro correspondiente a Virtual *Reality Supported* y los dos que forman parte del entorno Oculus, como se ve en la Figura 45.

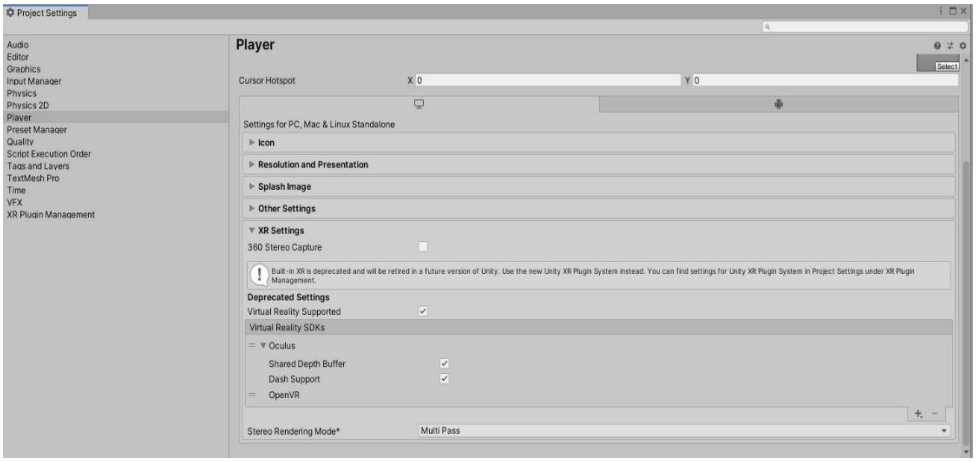

Figura 45: *XR Settings*

Al quitar esta ventana se puede ver en los *Assets* de Unity una carpeta llamada Oculus. En ella, se encuentran todos los elementos que se han importado [80], así como todos los elementos necesarios para poder visualizar la imagen dentro de los dispositivos *HMD*.

Para conseguir la sensación completa de inmersión es necesario realizar algunos ajustes. Para ello, se crea un plano, que se usará como suelo ya que, sin el mismo, el usuario caería infinitamente y no se podría mover correctamente. Además, es necesario eliminar la cámara que viene por defecto en Unity. Tras ello, en la carpeta Oculus > *VR* > *Prefabs* se añade el objeto llamado *OVRPlayerController* en la escena. Este objeto permite que el usuario se pueda mover en el entorno. Para conseguir la sensación completa de inmersión, son necesarias que las manos descargadas sean incluidas en el programa. Esto es posible conseguirlo añadiendo el *LocalAvatar* que se encuentra en la carpeta Oculus > *Avatar* > *Content* > *Prefabs*, y se coloca como hijo de *OVRPlayerController* > *OVRCameraRig* > *TrackingSpace*, anteriormente añadido. El resultado se puede ver en la Figura 46.

Figura 46: Avatar y cámara en Unity

Una vez conseguido el entorno de realidad virtual, se procede a importar el cuadro que anteriormente fue modelado en 3DS Max. Así, se puede tener siguiendo los pasos correctos descritos en la web de Unity [81]. Se puede ver el resultado en las Figuras 47, 48 y 49. Estas son fotografías del cuadro en el entorno que se había creado visto desde diferentes perspectivas.

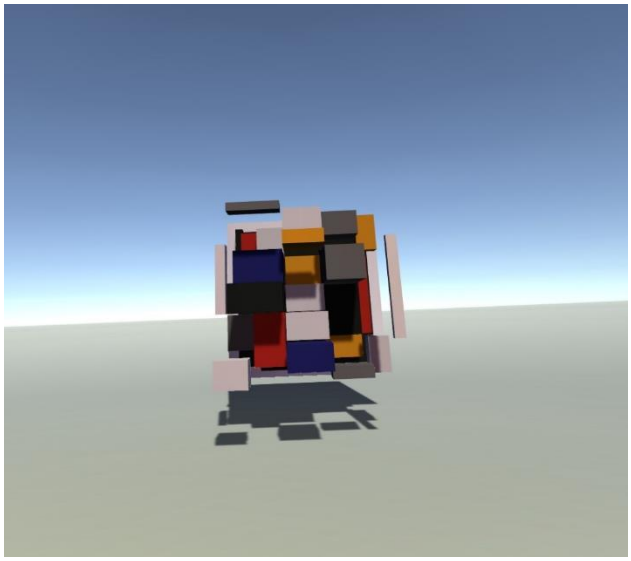

Figura 47: Frontal del cuadro

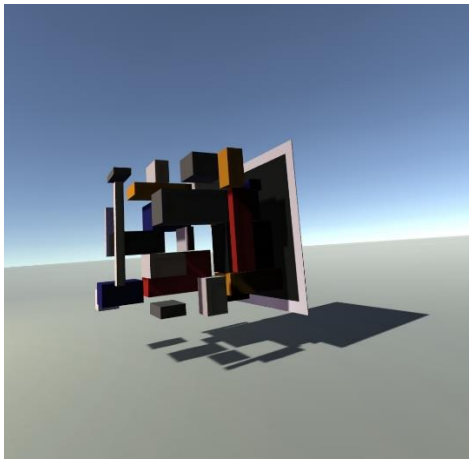

Figura 48: Lateral derecho del cuadro

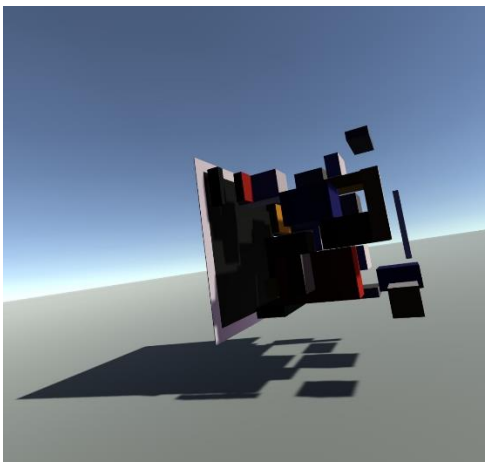

Figura 49: Lateral izquierdo del cuadro

Como se puede ver, gracias al montaje del cuadro, el entorno o la propia luz del sol, que consigue sombras realistas del cuadro, hacen que el sujeto sea capaz de vivir el ambiente virtual como algo real con lo que se puede interactuar.

Antes de explicar en profundidad lo ya mencionado, es necesario comentar ciertos aspectos que no interfieren en la secuencia de escenas, como son la colisión, posición, materiales de los objetos y comunicación entre ficheros. El primero de ellos, es añadir un modificador llamado *Box Collider*. El objeto que lleve asignado este modificador permite que otros objetos colisionen con él. Es la forma que tiene Unity para que los objetos no atraviesen entre sí y actúen según el contacto sufrido [82, 83, 84]. Esto se ha podido conseguir siguiendo el orden propuesto en varios tutoriales [85, 86]. Así, se ha podido añadir correctamente este modificador de forma que los cubos y distintos objetos puedan interactuar entre sí, tanto pertenecientes al usuario como formando parte de los elementos que aparecen en el entorno propios de las propias explicaciones. Este parámetro, como la gran mayoría de Unity, es modificable mediante línea de comandos en un fichero de texto [87].

El segundo es disponer de distintas opciones siempre controladas, como son encontrar objetos en la escena mediante código [88, 89]. También es posible cambiar la posición de los objetos [90].

En tercer lugar, es necesario tener una correcta comunicación entre todos los objetos, teniendo siempre presente la existencia de variables globales que ayudarán a controlar correctamente los tiempos e interacciones entre elementos [91, 92, 93]. Estas variables se almacenan en el fichero GlobalVar. Gracias a lo ya comentado, así como las distintas funciones que ofrece Unity, es posible controlar los objetos a partir de código [94, 95].

El cuarto aspecto que destacar es la manera de alterar e incluir los materiales, ya que durante todo el proyecto esto se ha tenido en cuenta, haciendo que la apariencia sea lo más atractiva posible [96], dando la posibilidad de activar o desactivar la visibilidad de los objetos [97, 98]. Esto haría referencia puramente a la apariencia. Otra idea que se ha tenido en cuenta en relación con los materiales era cambiar la transparencia de los objetos [99, 100], así como poder cambiar los colores por código [101].

Por último, es necesario mencionar las formas de incluir audios, relacionarlos con objetos [102], reproducir y pausarlos de forma correcta [103, 104].

Para analizar el contexto del autor y planificar las diferentes escenas se han usado distintas referencias. Así, es posible encontrar referencias en webs [105], documentos [106] o vídeos [107, 108]. Durante todas estas referencias, es posible encontrar gran cantidad de información acerca de la biografía y estilos artísticos desarrollados por el autor. De esta manera, se ha podido mantener un orden lógico de forma clara y lo más cercano a la realidad del pintor. Además, en una escena concreta, relacionada con las influencias que la música influyó en el autor, se ha tomado la información principalmente de dos enlaces [109, 110].

Finalmente, en otra de las escenas que a continuación se procede a desarrollar, se realizó una muy breve exposición con cuatro obras del propio autor. La mayoría de estas pinturas fueron tomadas también de otro enlace [111].

Las escenas están a oscuras, aunque levemente iluminadas. En una sección final posterior a la explicación de las escenas se mostrarán en más detalle.

Respecto a la audioguía, se valoraron distintas opciones, hasta que finalmente se eligió una que fuera lo más natural posible. Para la realización de esta audioguía, se decidió usar una aplicación web que permite transcribir texto a habla y se adecuaba a lo buscado [112].

A partir de este punto, se va a explicar qué se ha realizado en cada una de las escenas.

4.2.2.1. Escena 1

En la primera escena se da la bienvenida y se explican las instrucciones al usuario mediante una audioguía, buscando sorprender y llamar la atención acerca del entorno donde se encuentra. La persona es capaz de moverse con los joysticks de los mandos. Se menciona también el cuadro con el que va a interactuar, haciendo que la persona sea consciente desde un primer momento de la existencia real de esta pintura. Este audio suena una vez iniciada la experiencia, aunque se ejecuta tan solo una vez, excepto al llegar al final de la experiencia y volver al principio. En el entorno, el usuario aparece en el interior de una sala a oscuras, con el cuadro iluminado al fondo. Esta luz imita un foco permanente en todas las escenas, de forma que el cuadro siempre queda completamente visible. Se puede ver el entorno creado en la Figura 50.

Figura 50: Cuadro inicial en la escena

Los parámetros del foco mencionado se pueden ver en la Figura 51.

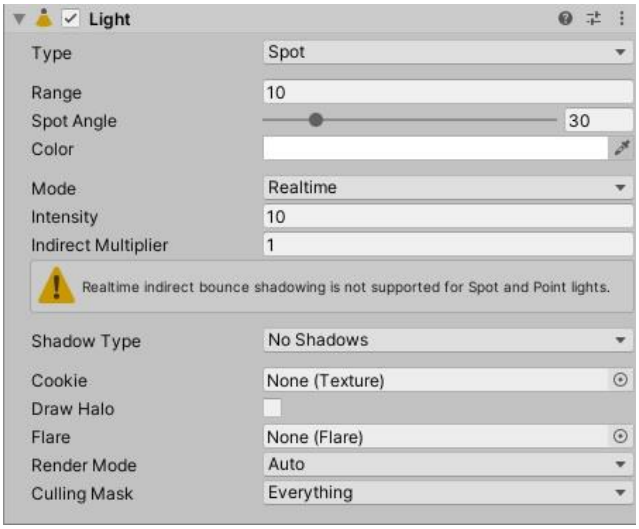

Figura 51: Parámetros del foco

Para dar mayor realismo a la escena se ha añadido una pared de ladrillos alrededor del cuadro, imitando una habitación. De esta manera se ha evitado tener un espacio infinito.

En todas las escenas se mostrarán un cubo verde, los cuales permiten que se inicien las siguientes escenas. Para separar cada escena de las demás, es importante destacar que cada una de ellas empieza al iniciarse su correspondiente audio de explicación. Así pues, los cubos verdes son los encargados de iniciar esos audios, activar los nuevos cubos de acciones y desactivar [113, 114] los pertenecientes a la escena anterior. Se decidió que para organizar de mejor manera la escena, los cubos verdes fueran aquellos de color gris o negro, tan sólo uno en cada escena.

En lo referido al cubo iluminado de verde, se ha añadido un modificador con un audio diferente al de la introducción, que se activará en la siguiente escena al tocar el cubo. Los parámetros correspondientes al mencionado audio se pueden ver en la Figura 52.

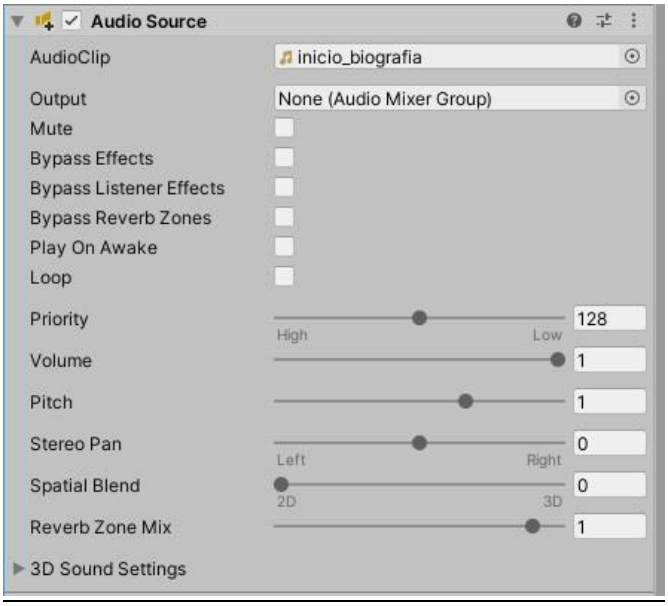

Figura 52: Audio Escena 1

Para conseguir la correspondiente interacción con el cubo y así poder pasar a la siguiente escena, es necesario prepararlo mediante líneas de código, como se puede ver en la Figura 53.

Figura 53: Script Escena 1

Como se puede ver en la Figura anterior, se han asignado elementos a los correspondientes campos. El primero de ellos, un AudioSource. Posteriormente, la pared que acompaña al cuadro se ha relacionado con el Objetoanterior del fichero; el grupo de objetos llamado Mondrian_joven asignado a Cuadro 1 y, finalmente, la luz_joven se relacionó con el Point 1.

4.2.2.2. Escena 2

En esta escena se empieza a interactuar con los primeros elementos externos al cuadro. Con respecto al audio activado al tocar el cubo verde en la escena anterior, se explican los inicios de Mondrian, incidiendo en su lugar de nacimiento, profesión previa a la pintura e influencias artísticas provenientes de la familia.

Con el fin de mostrar estos inicios en la vida del autor, se presenta un plano con la foto de este en la zona trasera de la habitación donde se ubica el cuadro. Este plano, llamado Mondrian_joven, es iluminado por una luz que se enciende tan solo durante esta escena. Esto se pude ver en la Figura 54.

Figura 54: Imagen de Mondrian [113]

Una vez el usuario toca la foto, ocurren dos eventos distintos. El primero de ellos es la aparición de una gran pantalla, realizado por un plano, que reproduce un vídeo donde se pueden ver imágenes de Ámsterdam [116], lugar de nacimiento del pintor en las fechas correspondiente a su nacimiento. El vídeo se muestra en la Figura 55. Estos vídeos se pueden reproducir de forma correcta siguiendo los pasos indicados [117, 118]. El plano donde se reproduce este vídeo se realizó en 3DS MAX, dándole forma curva para abarcar la mayoría de campo visual en la escena. El gran tamaño de este fue asignado directamente en Unity.

Figura 55: Vídeo de Ámsterdam [116]

Además de la pantalla con el vídeo, aparece una bandera con los colores de Holanda [119] con el nombre bandera_holanda. Esto se añadió al modelado de una enseña que previamente fue descargada [120]. Al igual que el resto de los objetos, a esta bandera le acompaña una luz que tan solo será encendida durante esta escena. La imagen de la bandera se puede ver en la Figura 56.

Figura 56: Bandera de Holanda

El plano con la foto del autor lleva asignado un Script llamado juventud. Se muestra en la Figura 57.

Figura 57: Asignación Script juventud

Como se puede ver, el objeto correspondiente a Bandera1 lleva asignado el objeto pCyllinder29. Para Bandera2, pCyllinder30, y para Bandera3, pPlane2. Todos estos objetos conforman el distinto modelado de la bandera. En este Script será usado para hacerlos visibles o no. El objeto Fondo_video1 tiene asignado el elemento fondo_videos1, el cual corresponde con el objeto el cual cumple la función de pantalla a la hora de mostrar el vídeo. Este objeto lleva asignado un modificador de Box Collider, el cual se ha relacionado con Collider1 del archivo de texto. Por otro lado, tanto Point1 como Point2 son luces para iluminar tanto la bandera como el gran plano, nombradas Luz_Holanda y Luz_fondo_videos1, respectivamente.

Respecto al plano de vídeo, en la Figura 58 se puede ver el modificador Video Player que permite que la reproducción se realice, con el vídeo correcto y de la mejor manera. También se ha desmarcado el cuadro Play On Awake, evitando entonces que se reproduzca el vídeo al iniciar la aplicación, ya que lo que interesaba era realizarlo al tocar el plano con la foto. Sí se han dejado marcados los tres siguientes marcadores, haciendo que se pueda tener un vídeo repetitivo y ejecutándose desde el principio.

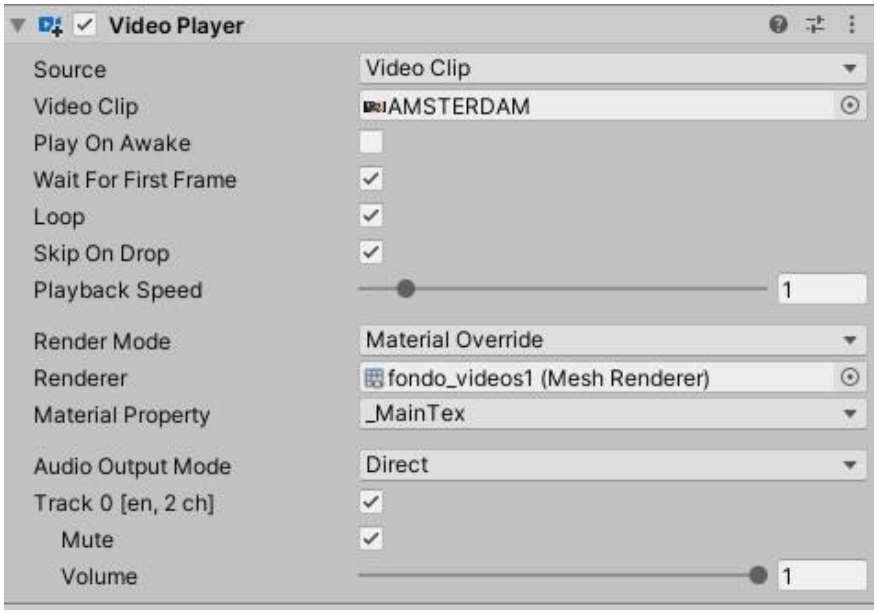

Figura 58: Video Player plano 1

Tanto el objeto Mondrian_joven como bandera_holanda han sido juntados en otro elemento que se ha llamado juventud. Este corresponde tan solo a una agrupación de ambos, entendiéndolo como una carpeta donde juntar todos los elementos con un fin similar. Esto se ha realizado para tener todos los elementos mejor ordenados.

Finalmente, en esta escena se presenta otro cubo de escena verde. Al igual en el anterior, permite cambiar a la siguiente, que reproduce el siguiente audio de la experiencia. En la Figura 59 se puede ver el modificador de audio que se ha asignado.

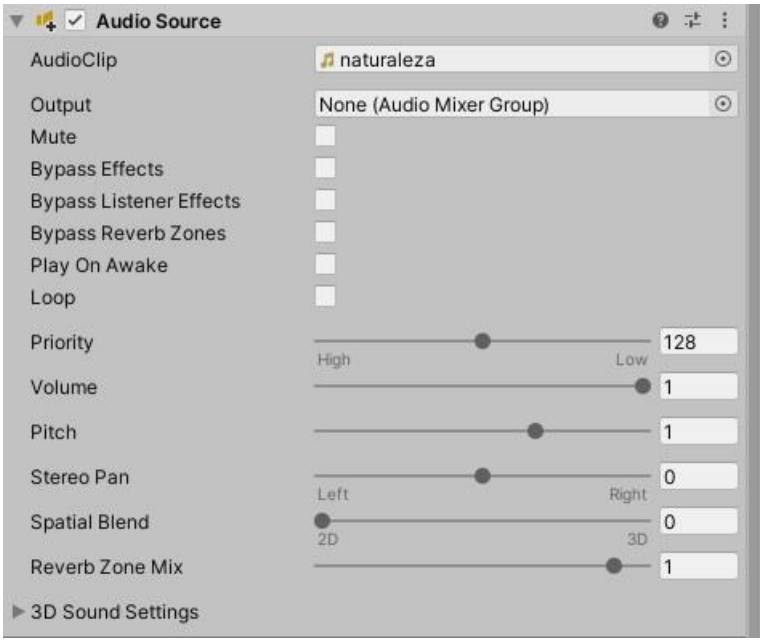

Figura 59: Audio Escena 2

Al igual que el resto de los cubos verdes, se necesita un Script para realizar lo comentado anteriormente. Se muestra en la Figura 60 el correspondiente modificador.

Figura 60: Script Escena 2

Igualmente, no se asigna ningún audio al campo de AudioSource ya que esto se hará mediante línea de código. A Objetoanterior se le ha asignado el cubo Negro_Escena1. De forma similar, a Luz1, luz_joven. Por último, Cuadro1 lleva asociado el objeto Mondrian_joven, de forma que posteriormente se procederá a activar o desactivar los elementos, según convenga e interese en la aplicación.

4.2.2.3. Escena 3

Durante esta escena se explican los inicios de Mondrian en la pintura. Debido a la influencia de la pintura holandesa de la época, el pintor comienza realizando bodegones, paisajes y elementos naturales.

Al pulsar el cubo aparecen mariposas que conectan con la intención del pintor y su relación con la naturaleza. A pesar de que en un principio se eligieron varios animales distintos, en última instancia se decidió que aparecieran tan solo mariposas alrededor. Estas mariposas, cuyos modelos fueron descargados [121], se puede ver en la Figura 61.

Figura 61: Modelado de mariposa [121]

Estas aparecen al tocar un determinado cubo azul que se ilumina. Este cubo lleva asociado un Script llamado Crea_Mariposas que hace que aparezcan en tres puntos diferentes del espacio. Los objetos a los que se hacen referencia en el fichero de texto se pueden ver en la Figura 62.

Figura 62: Script Crea_Mariposas

Como se puede ver, se ha asignado el prefab del objeto descargado llamado SCharacter_Butterfly 1. Un prefab es un Asset, normalmente realizado a partir de un objeto de la escena, que mantiene los modificadores y propiedades del objeto. Posteriormente se puede eliminar el objeto, sin que interfiera en la ejecución de la aplicación e incluir los prefabs de forma que interese. Esto se puede ver explicado en diversos enlaces [122, 123]. Una vez conseguido el prefab de este objeto, que además lleva incluido el movimiento de las alas mientras vuela, es necesario hacer que las mismas aparezcan en los lugares donde interesan. Para conseguir estos puntos, se decidió colocar tres objetos vacíos, llamados empty GameObject, alrededor del cuadro, de manera que estos estuviesen bien colocados en la escena y las mariposas estuvieran integradas de la mejor manera. Estas posiciones han sido nombradas como Butterfly1, Butterfl2 y Butterfly3, que se han agrupado en un paquete llamado Mariposas en la ventana de Hierarchy.

Una vez obtenidos los puntos se decidió que aparecieran cuatro mariposas en cada uno de ellos, de manera que en total surgen doce en la escena, el cual es un número bastante adecuado a lo que se quería conseguir.

Para hacer que los insectos aparecieran, era necesario instanciarlos. Esta acción consiste en hacer que un objeto de Unity aparezca de forma automática según los parámetros establecidos [124, 125]. Como se puede ver en diferentes ficheros, necesitan establecerse algunos parámetros como posición, objeto o rotación [126, 127]. Esto ha sido usado en el Script Butterflyvlieg.

Con respecto al movimiento de las mariposas, se buscó alguno que fuera el más realista posible [128, 129]. Sin embargo, se adaptaron algunas velocidades del fichero, con el fin de hacer que la animación fuera más acorde a la realidad. Además, en el momento en que las mariposas tocaban un cubo o la mano del usuario, estas las atravesaban, por lo que la sensación de interacción era muy imprecisa. Para corregir este fallo, se decidió controlar la dirección hacia la que se movían las mariposas. Para ello, mediante código, establecí un volumen alrededor del cubo y que, de forma aleatoria, se eligiera un punto del mencionado volumen [130, 131]. Al chocar con cualquier otro elemento de la escena, se ha configurado para que las mariposas recalculen la posición establecida, buscando de esta manera otra diferente. De esa manera, el movimiento de vuelo es más realista y comparable a los que se pueden encontrar en la naturaleza. Una vez explicado todo esto, se pueden seguir los pasos para instanciar un objeto en posiciones aleatorias [132]. El resultado final de todo lo realizado en las mariposas se puede ver en la Figura 63.

Figura 63: Movimiento de mariposas

Una vez instanciadas las mariposas es necesario que en la siguiente escena estas sean eliminadas, de forma que no ocurran colisiones ni elementos extraños durante el resto de la ejecución. Es por ello por lo que también se preparó mediante Script que en la siguiente escena los objetos que no pertenecieran a ella fueran destruidos.

Al igual que en el resto de las escenas, el cubo verde es el indicado para cambiar a la siguiente. Esto se realiza mediante el fichero Escena 3, que se muestra en la Figura 64.

Figura 64: Script Escena 3

Como se puede ver, en Objetoanterior está asignado el objeto Negro_Escena_2 que se añadió en Unity. El elemento Cubismo se relaciona con Cuadro 1 y Luz_Cubismo con Luz 1. Con estos elementos es posible activar y desactivar las distintas opciones a realizar.

El cubo verde también lleva otro AudioSource, que se activará al tocarlo. El campo de audio se puede ver en la Figura 65.

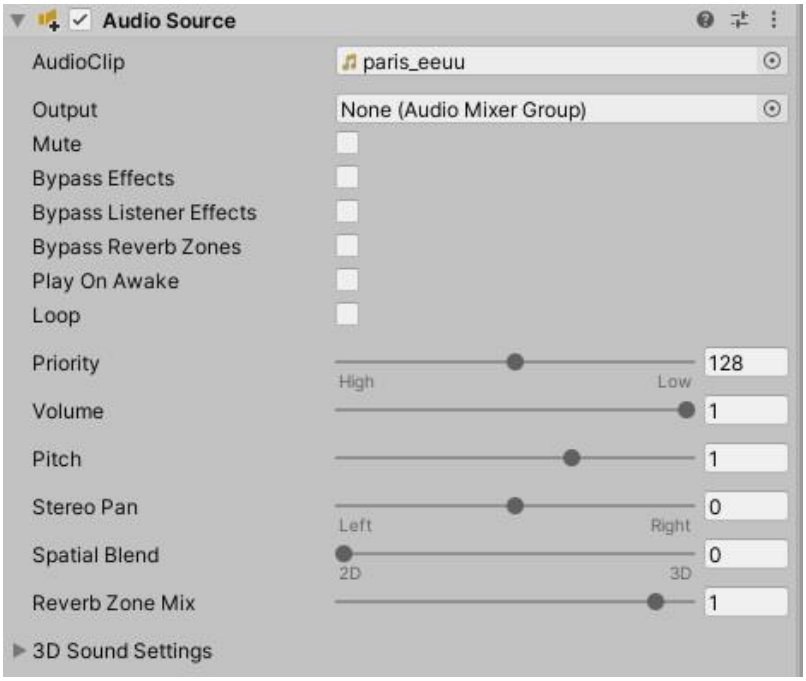

Figura 65: Audio Escena 3

Cabe destacar el dinamismo que esta escena ofrece, consiguiendo que la inmersión sea más atractiva para el usuario. Y es que esta fue una de las escenas más complicadas de realizar, ya que el incluir algún elemento de Inteligencia Artificial de forma realista, como son los movimientos de las mariposas variando las velocidades, tiempos de esperas, posiciones finales e interacción con otros objetos, hicieron que el desarrollo fuera más complejo.

4.2.2.4. Escena 4

Durante esta escena la audioguía explica la importancia de dos grandes viajes que realizó el pintor. El primero de ellos, hacia París, donde aprendió y fue influenciado por el cubismo. El segundo viaje fue el realizado hacia Estados Unidos, dónde se consagró como gran artista y pasó los últimos años de su vida.

De forma similar a la Escena 2, se decidió incluir un plano que al tocarlo hiciera aparecer un gran plano donde se mostrará un vídeo, similar a la pantalla de un cine. Además, con motivo de estos viajes mencionados, se propuso incluir de nuevo otra bandera, en este caso con los colores de Estados Unidos [133], así como dos modelados diferentes. Uno de ellos de la Torre Eiffel [134], visible en la Figura 66, mientras que el otro de un avión [135] que otorgará de mayor dinamismo a la escena.

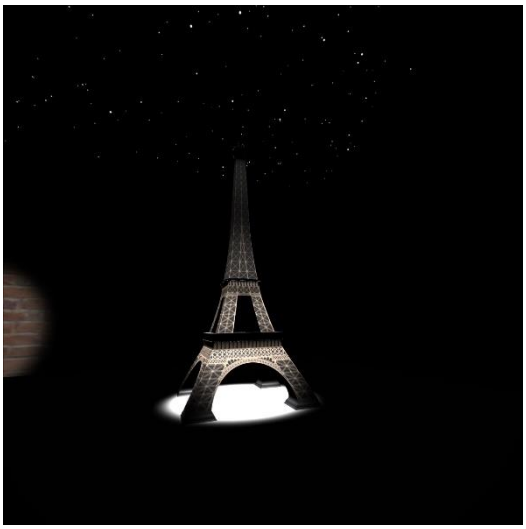

Figura 66: Modelo de la Torre Eiffel [134]

Respecto al plano a tocar, se eligió hacerlo con una imagen de las Señoritas de Avignon [136], de Picasso, como muestra de cubismo. Se puede ver en la Figura 67, al igual que el plano de vídeo una vez activado el mismo. En esta pantalla se pueden ver imágenes de las calles de Nueva York en fechas similares a las del autor en este país [137].

Figura 67: Imagen cubista y vídeo de EE. UU. [136, 137]

Para conseguir que esto ocurra, se ha decidido recurrir de nuevo a un Script, en este caso llamado Cubismo, que se muestra en la Figura 68.

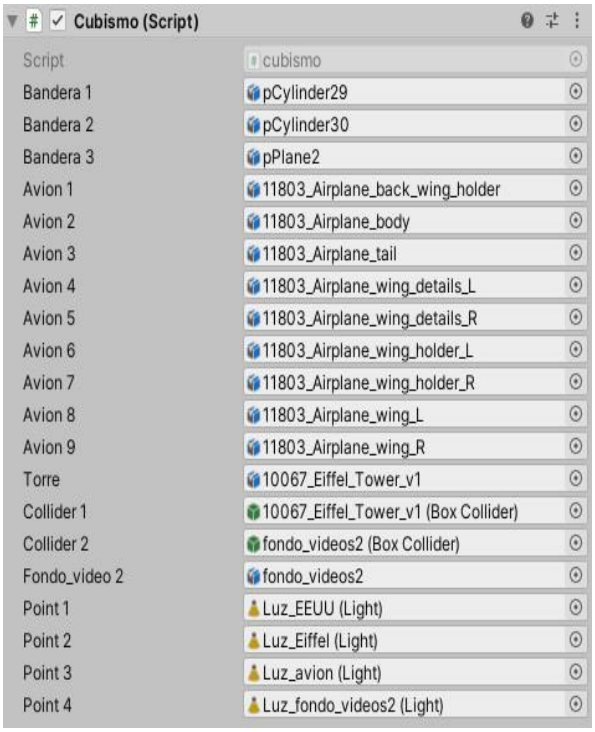

Figura 68: Script Cubismo

Como se puede ver en la imagen, se han asignado gran cantidad de elementos a los correspondientes del Script. Bandera1, Bandera2 y Bandera3 llevan asignados los objetos correspondientes al modelo de la bandera, como son pCylinder29, pCylinder30 y pPlane2 (este último objeto lleva asignado el dibujo de la bandera de Estados Unidos). Por otro lado, se pueden ver los objetos Avión1, Avión2, Avión3, Avión4, Avión5, Avión6, Avión7, Avión8 y Avión9, que se relacionan con las distintas partes que conforman el modelado del avión. Además de ellos es posible ver que el objeto Torre tiene asignado el modelado de la Torre Eiffel descargado. Para Collider1 y Collider2 se decidió que fueran los Box Collider pertenecientes a la Torre Eiffel y el plano de vídeo (fondo_videos2), con el nombre de Fondo_video2, de manera que se pudiera activar o desactivar según convenga para la acción. Finalmente, en la Figura 71 se puede ver los elementos Point1, Point2, Point3, Point4, que pertenecen a las luces asignadas a la bandera, Torre, avión y pantalla de vídeo, respectivamente. Para tener un mejor orden, el objeto del vídeo de la Escena 2, así como el de esta escena, han sido juntados en un mismo paquete llamado Fondos_videos.

Para realizar el movimiento del avión se decidió que siguiera una trayectoria en el aire alrededor del cuadro, moviéndose desde la Torre Eiffel hasta la bandera de Estados Unidos. Para ello, se establecieron seis puntos alrededor del espacio, que mediante programación se ha preparado para que el modelado del avión se mueva de un punto al otro. Los puntos, con los nombres Direccion1, Direccion2, Direccion3, Direccion4, Direccion5 y Direccion6 se han agrupado en un paquete llamado Direcciones_Avion. Para conseguir el resultado buscado, se ha decidido que se prepare todo el desarrollo en un Script llamado Vuelo. Este fichero de texto se ha asignado a un objeto llamado Pivote, que es el encargado de dar dirección al avión, cuyo modelo se ha establecido como hijo del objeto mencionado. Así pues, gracias al Script Vuelo el pivote guiará la dirección del avión hacia los seis puntos que se encuentran en el espacio. Por último, se ha proyectado sobre el avión el cuadro de

Mondrian “Composition with red, yellow, blue and black”, que tiene características similares a “Composition A”. En la Figura 69 se puede ver el cuadro usado.

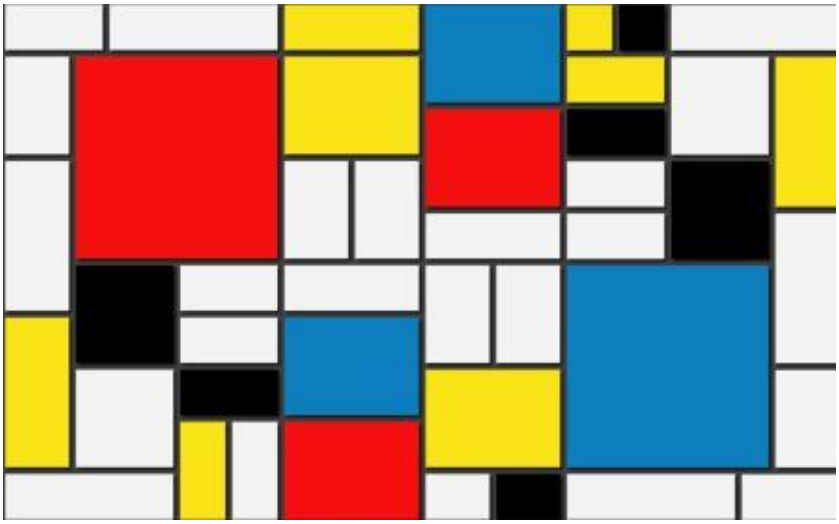

Figura 69: Material del avión

Como se puede ver en la Figura 70, el impacto que causa el modelado de la escena es uno de los grandes atractivos que ofrece la experiencia.

Figura 70: Avión [135]

Sin embargo, al igual que en el caso anterior, el hecho de tener elementos estáticos y tan alejados del usuario hacía que la experiencia de inmersión perdiera el efecto buscado. Es por ello por lo que finalmente se decidió colocar los elementos con la disposición que se ha mostrado en esta sección.

Para terminar con la escena, como ocurre con las demás, es necesario mostrar los modificadores correspondientes al cubo verde de cambio de escena. En este caso, el AudioSource, en la Figura 71, lleva asignado el audio música_notas.

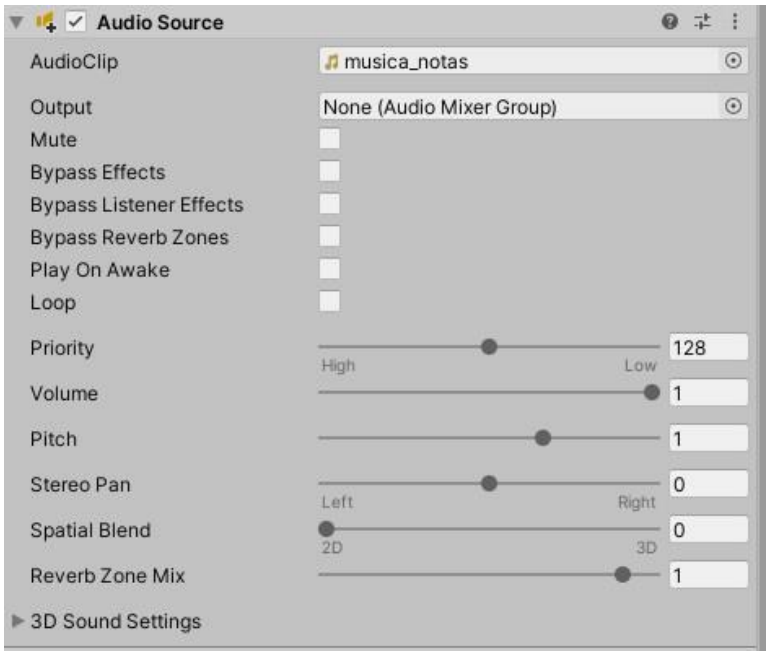

Figura 71: Audio Escena 4

Respecto al Script correspondiente al mencionado cubo, se le ha asignado el que lleva el nombre Escena 4. En la Figura 72 se pueden ver los elementos que lo componen.

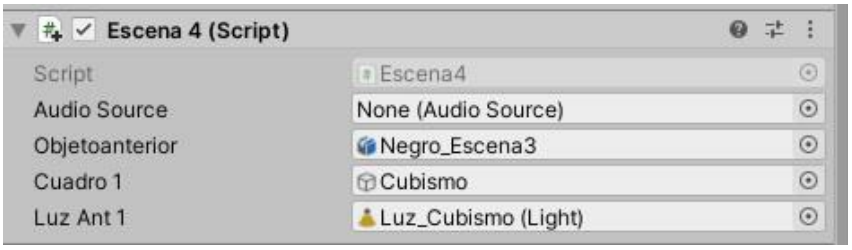

Figura 72: Script Escena 4

Así, se puede ver que en este caso Objetoanterior es el elemento Negro_Escena3. Por otro lado, Cuadro1 lleva asociado el paquete Cubismo y Luz Ant 1 la luz con el objeto Luz_Cubismo. Este fichero será el encargado de desactivar los elementos de esta escena y activar los de la siguiente.

4.2.2.5. Escena 5

Durante esta escena se muestra la unión entre la música y Mondrian. Las principales referencias de la música en el autor provenían de su propio padre. Sin embargo, la gran influencia de esta se produce al viajar a Estados Unidos. Los ritmos marcados del jazz se muestran en sus pinturas con las líneas verticales y horizontales, simulando de esta manera los pulsos de las partituras. Observando atentamente los colores de la pintura, se pensó en una estrategia clara. Como se puede observar en la Figura 73, aún no se habían usado los cubos blancos.

Figura 73: Cubos blancos

Por ello mismo se decidió que sonara la escala de Do Mayor. Cualquier escala musical se compone de ocho notas, al igual que el número de cubos blancos que existen. Pese a que un principio se decidió que la escala fuera cromática, es decir, con todos los semitonos [138], finalmente la elección fue usar sólo la escala mayor. Los audios de las notas para realizar estas acciones fueron descargadas [139]. Así pues, al tocar los cubos blancos se reproducen estas notas.

En esta escena se puede ver un inicio donde desarrollar la creatividad del usuario, ya que es posible crear pequeñas melodías con las mencionadas notas. Sin embargo, el fin principal no es la creación de música, sino el poder interactuar con elementos a través de los mandos y que se produzca una reacción sonora que cause un impacto en la persona.

Para conseguir correctamente la acción comentada, fue necesario realizar un Script que ejecutara estos audios. A este fichero se le llamó CollisionAudio, que se va a mostrar en la Figura 74.

Figura 74: CollisionAudio

Como se puede ver en la Figura anterior, no se ha asignado ningún objeto perteneciente a la escena. Esto es debido a que, dado que tan sólo se aplican los sonidos, no es necesario alterar ningún otro elemento. Para conseguirlo fue necesario incluir un modificador de AudioSource como en la Figura 75.

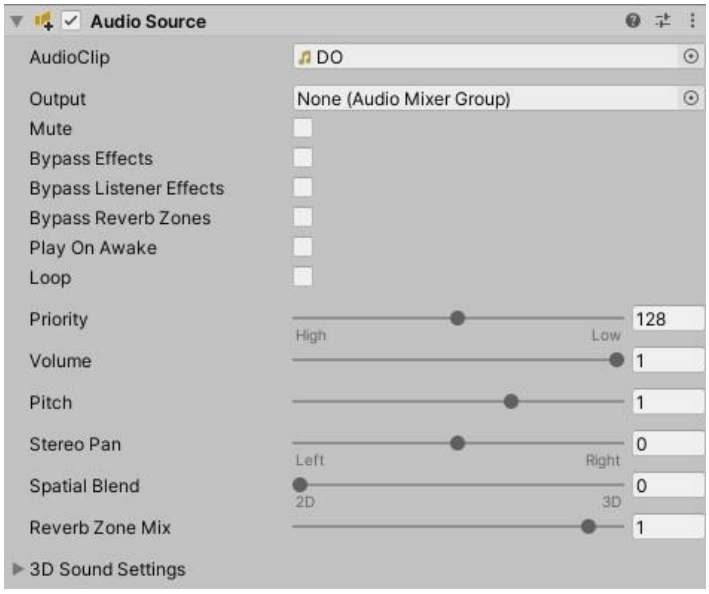

Figura 75: AudioSource de notas

Como se puede ver en el campo AudioClip, se ha asignado el archivo de audio DO. Siguiendo el mismo procedimiento que este cubo, se ha construido de forma similar para el resto de los blancos, asignando el resto de los audios de las notas a los otros cubos.

De la misma manera que en el resto de las escenas, se prepararon los cubos para que estos sonaran tan sólo en este escenario y que se mantuvieran en silencio al tocar los cubos en cualquiera de las otras escenas. Para conseguir esto, se hizo mediante el cubo verde de cambio de escena. Al igual que para los cubos blancos, el correspondiente verde lleva asignado un modificador de AudioSource que al tocarlo inicia el audio de la siguiente escena. Esto se puede ver en la Figura 76.

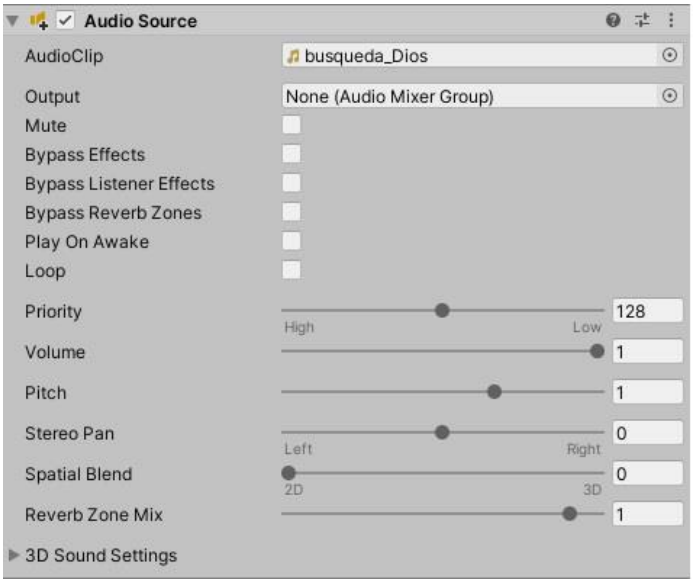

Figura 76: Audio Escena 5

El Script que se ha usado para realizar las determinadas acciones se ha nombrado Escena 5. Este puede verse en la Figura 77.

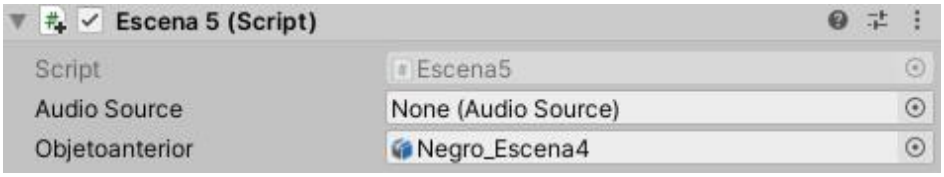

Figura 77: Script Escena 5

En este caso al objeto Objetoanterior se le ha asignado el elemento Negro_Escena4.

Esta escena, pese a no ser de una gran complejidad, es la primera sensación inmersiva e interactiva relacionada con el arte. A pesar de tener relación con la historia del autor, el usuario puede llevar la experiencia a un nivel más creativo gracias a los audios de las notas.

4.2.2.6. Escena 6

Prestando atención a la biografía del autor se puede ver la gran importancia que tuvo Dios en su vida. El pintor estuvo fuertemente vinculado a la búsqueda de Dios debido a que su padre era un sacerdote calvinista. Este hecho provocó que su pintura se desarrollara hacia una pintura más abstracta. Según el pintor, el mundo de la abstracción pertenecía al mundo de Dios, mientras que lo tangible era consecuencia de la realidad. Debido a ello, y habiendo estado influenciado por el cubismo durante su estancia en París, desarrolla su propio estilo artístico, alcanzando su faceta más reconocida en la actualidad. El abstractismo basado en los cuadrados es la manera en la que el autor fundamenta su búsqueda de Dios y su respectiva teología.

Durante esta escena, se tuvo en cuenta esta idea para desarrollar la experiencia de la mejor manera. Para ello se quiso usar la luz del sol como si fuera Dios. Mediante Scripts en distintos cubos, se consiguió la propuesta inicial, que era cambiar la intensidad de la mencionada luz.

Para ello primeramente es necesario saber el tipo de luz que se ha usado [140]. Una vez conocida su categoría se puede adaptar de la mejor manera a los intereses que se requieran para la experiencia. Este caso se ha decidido modificar cuatro intensidades diferentes de la luz del sol [141], pudiendo así tener la luz de madrugada, noche, mañana y medio día y poder asemejar esta relación de la luz con Dios. Se decidió que la relación del cuadro con la luz se hiciera mediante los cubos amarillos, como se ve en la Figura 78.

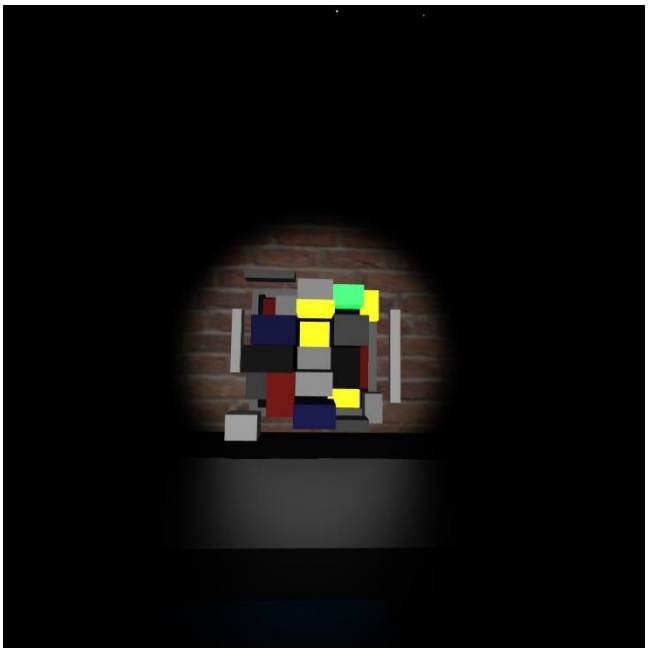

Figura 78: Cubos de luz del sol

Los porcentajes asignados a madrugada, noche, mañana y medio día, han sido 0%, 30%, 70% y 100% [142, 143] de la intensidad inicial, respectivamente.

Dado que la experiencia completa tiene como concepto la oscuridad y las luces acompañan al usuario en todo momento, se decidió que el cielo en ningún momento tuviera la claridad de la mañana, como en un principio se tenía. Esto se puede ver en la Figura 79.

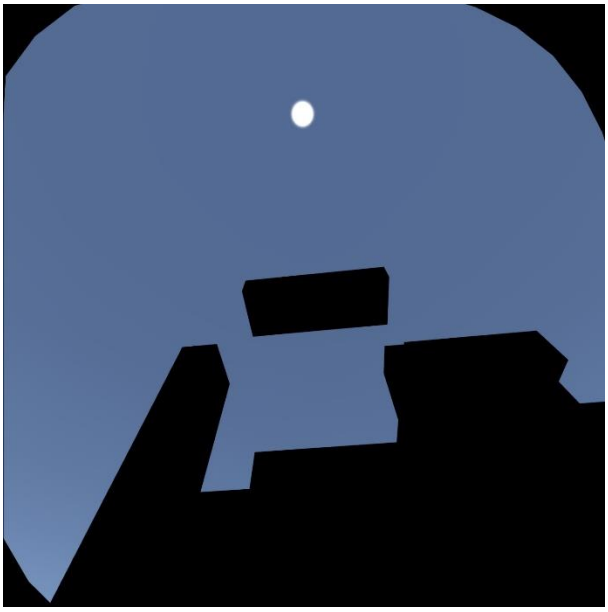

Figura 79: Luz de día

Sin embargo, como se ha comentado anteriormente, esta idea fue descartada, con el fin de tener un mejor resultado en la persona. El efecto más interesante conseguido durante esta escena se muestra en las Figuras 80 y 81.

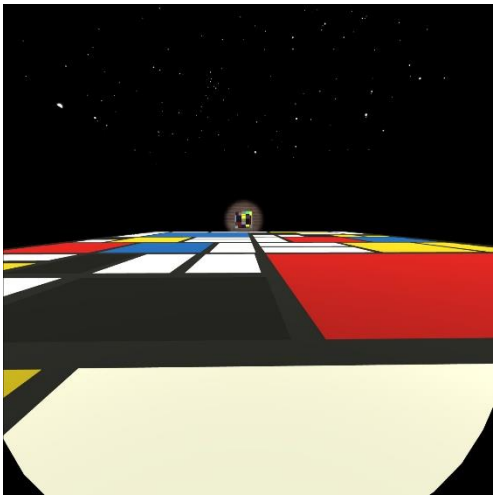

Figura 80: Luz de noche I

Figura 81: Luz de noche II

Es necesario destacar el gran impacto que causa sobre el usuario que interactúa en la experiencia. La luminosidad de la escena, en concreto del suelo, en contraste con la oscuridad del resto de la escena, hace que destaque sobre los ojos de la persona. Con respecto al suelo comentado, como se puede ver en las Figuras anteriores, corresponde con otra pintura diferente del mismo autor. Prestando atención a la disposición de los colores y cuadrados, se puede ver que es la misma imagen de la Figura 69, el cuadro usado como material en el avión. Esto se hizo de forma muy sencilla, y es que tan solo se asignó esta imagen como material a un plano, con el tamaño adecuado, de forma que la habitación tuviera las dimensiones correctas. El plano fue realizado en 3DS MAX, siendo de esa manera más sencillo organizar correctamente la disposición y colores para que fueran realistas respecto al cuadro original. Posteriormente el objeto se importó a Unity como se ha preparado con elementos anteriores.

Dado que cada cubo se usa para cambiar la intensidad de la luz, era necesario activar y desactivar cada uno según los intereses [144, 145]. En la Figura 82 se puede ver el modificador usado para la luz del sol.

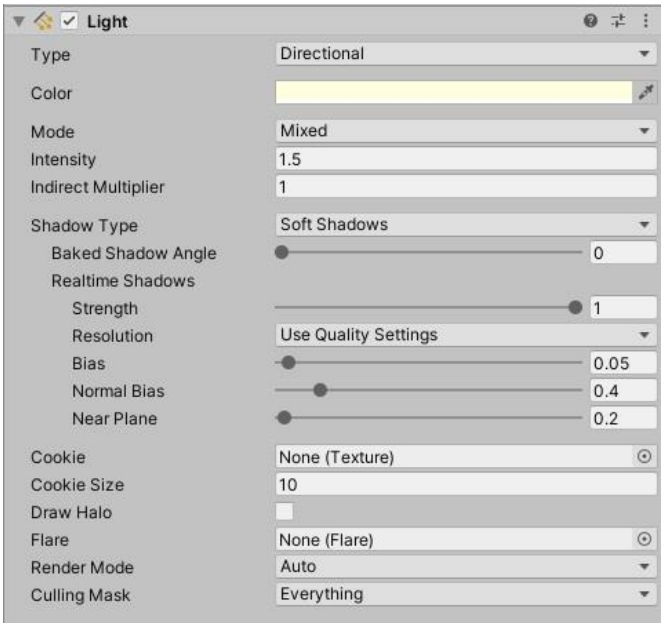

Figura 82: Luz del sol

Respecto a los cubos amarillos, se necesitaba de un fichero de texto que permitiera modificar las intensidades al tocarlos. En la Figura 83 se muestra el Script, llamado MidNight, usado en uno de los cubos.

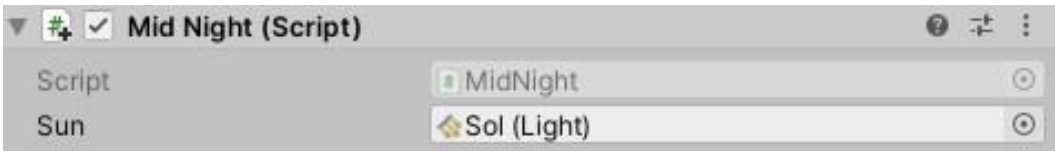

Figura 83: MidNight

Como se puede ver, al objeto Sun se le ha asignado el objeto correspondiente a la luz principal, con el nombre de Sol. De la misma manera que con este cubo amarillo, se han usado los Scripts Morning, Evening y Day en los otros tres. Todos ellos siguen la misma estructura, usando la luz Sol dependiendo de la intención buscada. Cada fichero es el encargado de ofrecer un valor de intensidad diferente.

El último cubo por explicar es el de color verde, que permite cambiar a la siguiente escena. De nuevo, se le asigna un AudioSource que sonará una vez active el cubo mencionado. Se muestra en la Figura 94.

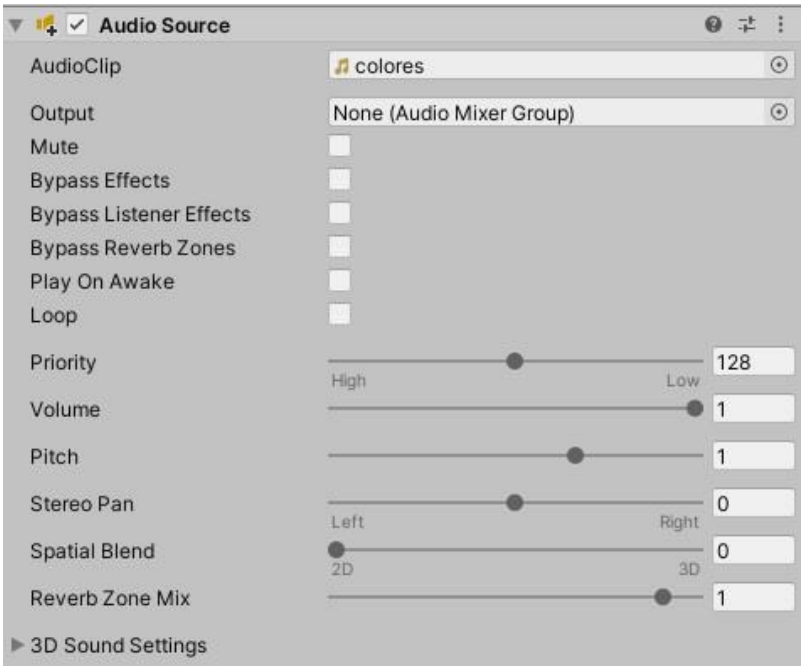

Figura 84: Audio Escena 6

Como se puede ver, lleva asignado el audio colores, que previamente se guardó.

El Script que realizará las acciones del cubo verde, siguiendo el orden de los anteriores, fue nombrado Escena 6. En la Figura 85 se muestra el uso dado en el objeto.

Figura 85: Script Escena 6

El elemento Luzsol lleva asignada la luz Sol de la escena de Unity, ya que en la siguiente escena esta luz seguirá manteniendo la intensidad inicial de la aplicación. Al igual que en anteriores ficheros, el campo Audio Source está vacío porque se hace uso del clip mediante líneas de código. Finalmente, Objetoanterior lleva asignado el objeto Gris_Escena5, haciendo que este cubo se desactive en un escenario diferente al suyo propio.

4.2.2.7. Escena 7

Esta escena se relaciona directamente con la anterior. La búsqueda de Dios que anteriormente se ha explicado fue el fin principal de las pinturas de Mondrian. Para alcanzar esa idea, recurrió a colores monocromáticos, en concreto gris, blanco y negro; y los colores primarios de la pintura, como son el azul, rojo y amarillo. Es por este motivo por lo que se usan estos colores en todas sus obras ya que, con el color y el cubismo, enfocado en líneas sencillas, se muestra la teología presente siempre en todas sus pinturas.

Se estuvieron evaluando distintas opciones que encajaran de la mejor manera con el conjunto de la aplicación. Finalmente se llegó a la conclusión de aprovechar la interacción con luces de colores, pudiendo ser modificadas mediante líneas de código [146, 147]. Dado que hay tres cubos rojos, se decidió que hubiera cuatro colores diferentes (rojo, azul y amarillo) y un cuarto que sería común entre todos, como es el blanco. De esta manera, al tocar los cubos se encenderían las correspondientes luces de colores.

Para realizar estas acciones se realizaron mediante tres ficheros de texto diferentes: Luz_Roja, Luz_Amarilla y Luz_Azul. En la Figura 86 se muestra el primero de estos ficheros.

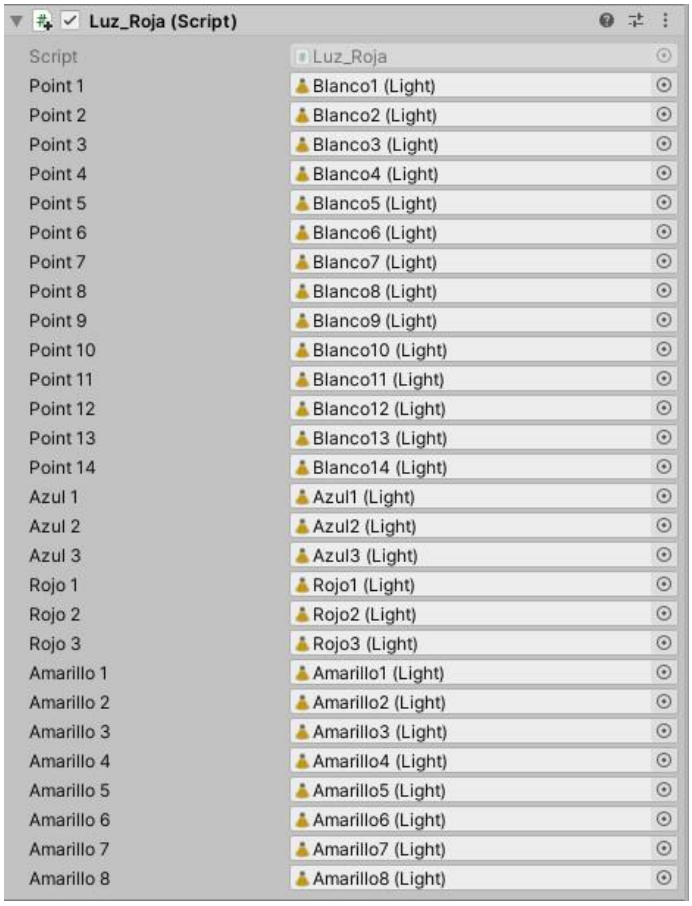

Figura 86: Script Luz_Roja

Como se puede ver en la imagen, todos los objetos del Script se corresponden con luces. Los otros dos ficheros cumplen la misma función que este. La idea que se quería conseguir era que al tocar cualquier determinado cubo se encendiera el color que va en el nombre del fichero. Si volvía a tocarse ese cubo, las luces cambian a blanco, y así sucesivamente. Lo mismo ocurre con los otros cubos rojos, cambiando los colores de las luces.

Se decidió adaptar el número de luces, así como las intensidades y rangos de estas buscando una intención concreta. La propuesta que se hizo fue la de colocar cada luz con el mismo tamaño que los cuadrados del suelo. Debido a que estos cuadrados tienen tamaños diferentes, fue necesario adaptar cada luz de forma individual, de manera que compartieran las mismas dimensiones. En la Figura 87 se muestra los parámetros de una de las luces usadas, aunque es necesario tener en cuenta que cada luz tiene su propio color, rango e intensidad.

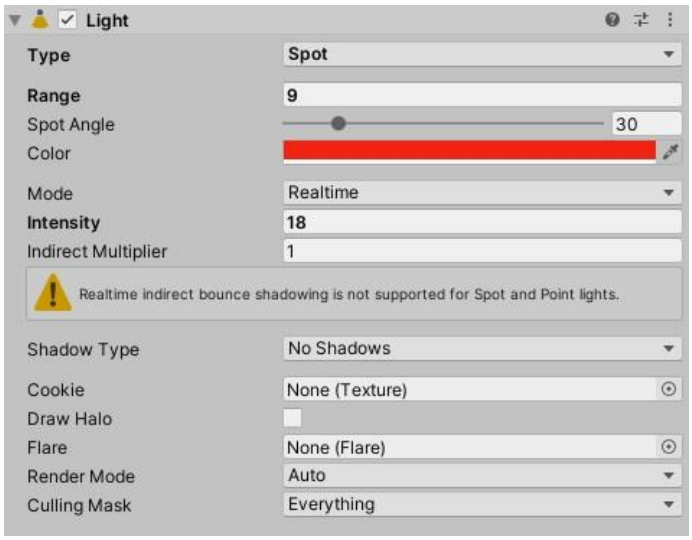

Figura 87: Luz de color

Se han usado 14 luces diferentes, 3 rojas, 3 azules y 8 amarillas que han sido colocadas sobre los cuadrados de esos colores del suelo, mencionado en la Escena 6. De forma similar, en las mismas posiciones, se han usado otras 14 luces para el color blanco. De esa manera, la combinación entre luces siempre es correcta y se mantiene. Todas ellas se han agrupado en un solo paquete llamado Luces_del_suelo.

El resultado final de las luces adaptadas al entorno queda de esta manera más acorde a la experiencia que se ha realizado. Las luces azules se pueden ver en las Figuras 88 y 89.

Figura 88: Luces azules

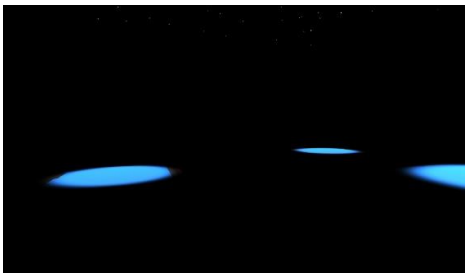

Figura 89: Luces blancas

De la misma forma se puede observar en las Figuras 90 y 91 para el color rojo y amarillo.

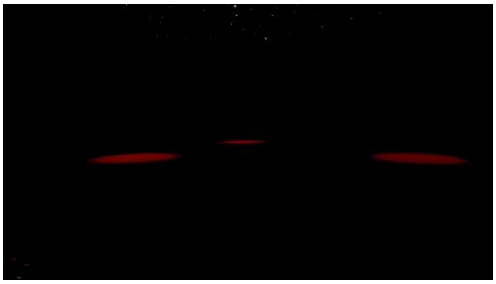

Figura 90: Luces rojas

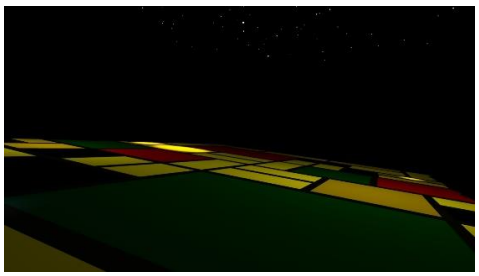

Figura 91: Luces amarillas

En esta escena, al igual que en la anterior, se ha experimentado con los sentimientos del autor y las referencias que tuvo en cuenta a la hora de construir su pintura. La búsqueda de Dios a través de los colores y la luz, así como las líneas verticales y horizontales, es visible en todo momento en sus obras.

Respecto al cubo verde, el modificador AudioSource lleva asociado el audio recopilacion_obras, como se puede ver en la Figura 92.

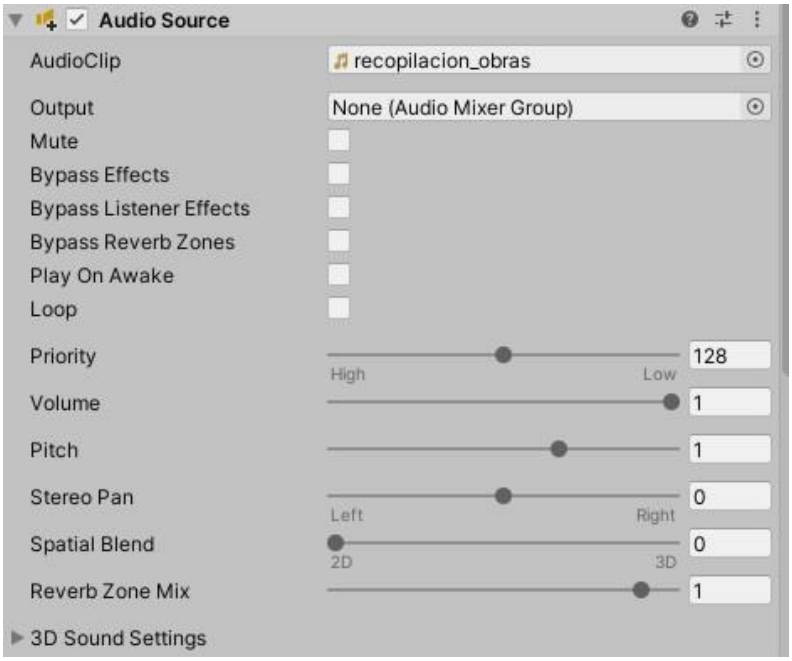

Figura 92: Audio Escena 7

En lo referente al Script asociado a este cubo, se puede ver en la Figura 93.

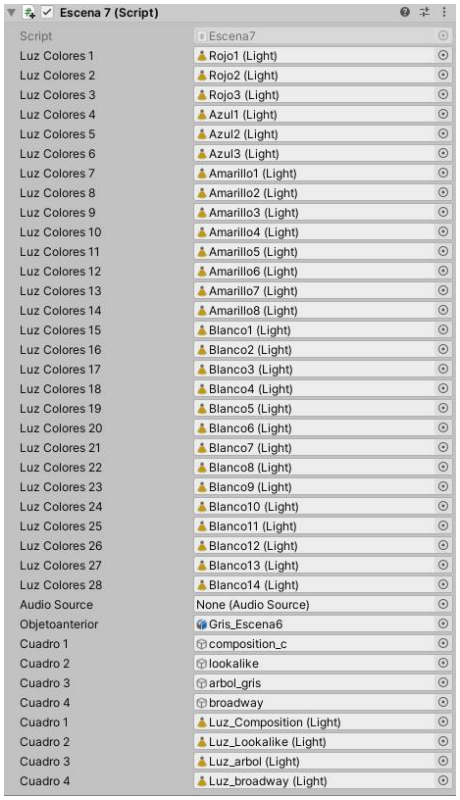

Figura 93: Script Escena 7

En este caso se pueden ver todas las luces de colores, ya que en la siguiente escena estas serán apagadas. Objetoanterior se relaciona con el objeto Gris_Escena6. Por otro lado, se activan cuatro objetos y sus respectivas luces. Estos objetos llevan el nombre composition_c, lookalike, árbol_gris y broadway. De forma similar, las luces llevan los nombres Luz_Composition, Luz_Lookalike, Luz_arbol y Luz_broadway.

4.2.2.8. Escena 8

Esta es la última escena dónde se repasará la vida y obras del autor. Las posteriores estarán relacionadas con el arte general. Para ello, una vez explicada toda la biografía, estilos artísticos e influencias en su pintura es el momento de realizar una pequeña muestra de algunas de sus obras. Dado que esta experiencia está estrechamente relacionada con el arte y los museos, era interesante dejar constancia de una muy breve exposición de sus pinturas.

Para conseguir este objetivo se eligieron las obras Composition C, Lookalike, El Árbol Gris y Broadway Boogie-Woogie [148]. Al igual que se ha realizado en escenas anteriores, estos cuadros fueron asignados a diferentes planos.

Se quiso que las pinturas rodearan y acompañaran al usuario en todo momento. A pesar de que la idea es sencilla, la ejecución resultó más complicada de lo esperado. Ello se debe a que ocurrían dos problemas al mismo tiempo. El primero de ellos era que si se preparaban las imágenes para acompañar al autor también se producían rotaciones alrededor del usuario, de manera que tan sólo se podían ver dos de las cuatro pinturas. El segundo problema se encontraba al intentar no alterar esa rotación. En ese caso la mayoría del conjunto de pinturas se situaba en el exterior de la habitación, haciendo imposible su visualización. Llegados a este punto, intentando solucionar ambos aspectos, hizo que casi se descartara esta escena. Sin embargo, tras pensar mucho acerca de estos elementos se dio con la solución.

Primeramente, era necesario tener una correcta colocación de todos los planos. Se colocaron todas las imágenes alrededor de la persona, a una distancia adecuada. Una vez se tenían las posiciones buscadas, se agruparon todas las pinturas en un paquete llamado Pinturas. A este paquete se le asoció un Script llamado Traslación_cuadros que permite acompañar al usuario. Esto se puede ver en la Figura 94.

Figura 94: Traslación de pinturas

Este fichero hace que se cumpla la rotación de los objetos y se pueda trasladar correctamente con el movimiento de la persona. Finalmente, la solución que se pensó fue calcular la posición actual de la persona en cada frame mediante un elemento vacío hijo del OVRPlayerController. Mediante la función Update de Unity se calcula un offset restando las posiciones x, y y z actuales a las posiciones x, y y z que previamente fueron guardadas en una variable. De esta manera, se tiene un valor para cada eje x, y y z. Por último, se necesita sumar estos offset a la correspondiente posición local del conjunto de pinturas y se actualiza la posición actual de las pinturas. Ejecutando este fichero de forma iterativa se consigue que las pinturas acompañen correctamente a la persona sin que roten con los mismos ángulos que la visión del sujeto. El resultado final se puede ver en la Figura 95.

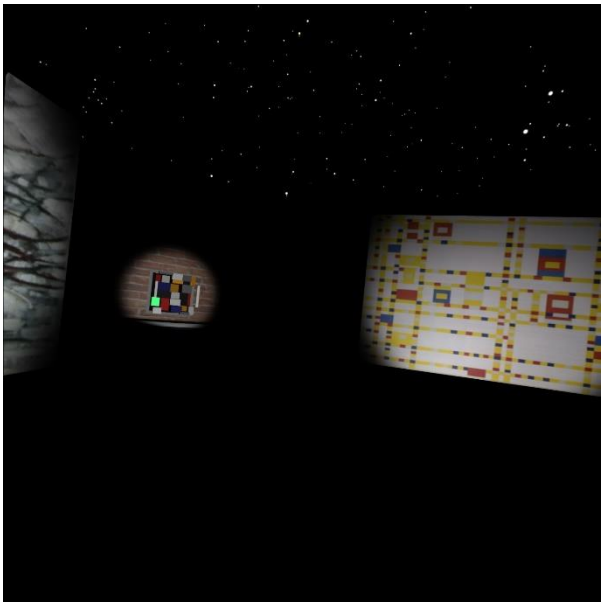

Figura 95: Exposición de pinturas del autor

Para el cubo verde se han añadido los mismos elementos que en los anteriores escenarios, como son un AudioSource y un Script para controlar los objetos aparecidos en esta escena y activar los de la siguiente. En la Figura 96 se puede ver el archivo de audio, con el nombre tilt_brush, asignado al cubo verde.

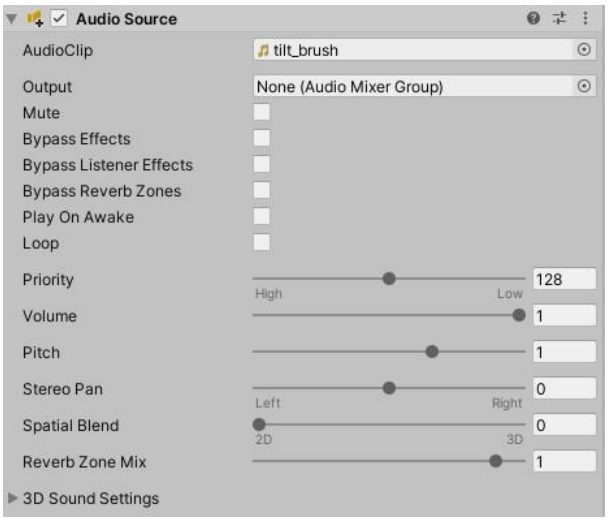

Figura 96: Audio Escena 8

Para ejecutar estas acciones era necesario crear un Script, en este caso llamado Escena 8, que deshabilitara los elementos actuales y activar los siguientes. Esto se ve en la Figura 97.

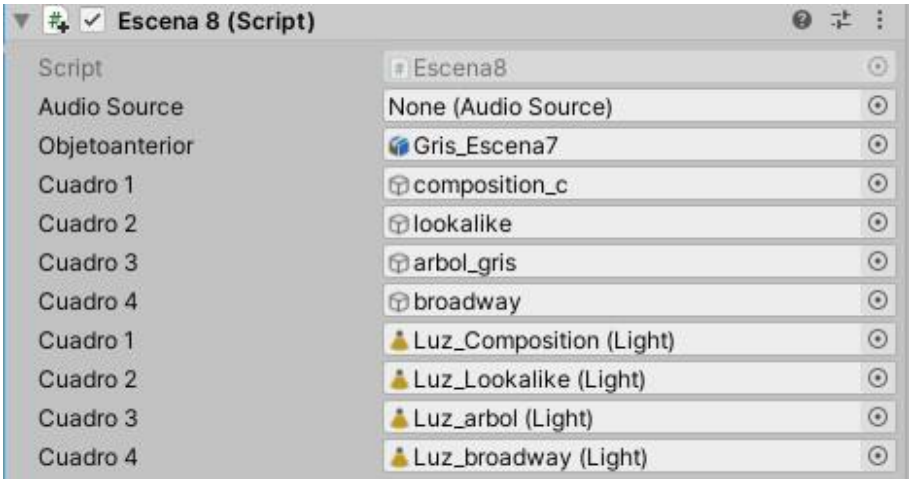

Figura 97: Script Escena 8

El Objetoanterior se ha relacionado con el objeto Gris_Escena7 y al igual que en la escena anterior, el resto de los elementos son estas cuatro pinturas y las luces que las acompañan. De esta manera, es posible cerrar la escena.

4.2.2.9. Escena 9

Para la actual escena se quiso desarrollar uno de los aspectos más interesantes y completos de la experiencia. En las anteriores escenas se han expuesto las referencias al pintor, por lo que se pensó que esta que compete estuviera más relacionada con el arte en general que con el propio autor, entendiéndose como un juego dónde usuarios sin experiencia, así como artistas, pudieran desarrollar sus dotes artísticas. El fin principal no era mostrar y enseñar todo lo relacionado con el arte, sino más bien poder interactuar con él. Es por ello por lo que se creó un escenario donde se pudiera pintar en el aire, siguiendo la forma usada en la aplicación Tilt Brush.

Crear esta escena fue la propuesta más complicada de todo el desarrollo, principalmente todo lo relacionado con la programación. Para conseguir el resultado buscado, se recurrió a gran cantidad de información de enlaces y vídeos [149, 150]. Esta debía crearse de la mejor manera, siempre adaptada a las circunstancias de mi aplicación. Es por ello por lo que, a pesar de estar basado en Tilt Brush, se decidió que se pudiera pintar en el aire tan sólo con líneas rectas y puntos con los cuatro colores que han acompañado la experiencia en todo momento (rojo, azul, amarillo y blanco), de la manera más parecida a las pinturas de Mondrian. Para conseguir las líneas rectas fue necesario adaptar los ficheros de texto, tomando como referencia algunos enlaces [151].

El desarrollo de esta escena se realizó tomando como partida ficheros descargados [152, 153] que posteriormente se han adaptado a esta experiencia. Para ello se colocaron dos luces, con el nombre Luz_Draw1 y Luz_Draw2, en la parte trasera de la habitación de forma que se pudieran visualizar de mejor manera las pinturas realizadas en el aire. Estas luces se agruparon en el paquete Luces_Draw.

Para poder pintar correctamente en el aire se creó el Script Draw_Lines, que fue añadido como modificador en los objetos LeftHandAnchor y RightHandAnchor pertenecientes al objeto OVRPlayerController. El uso de este fichero en la mano izquierda se puede ver en la Figura 98.

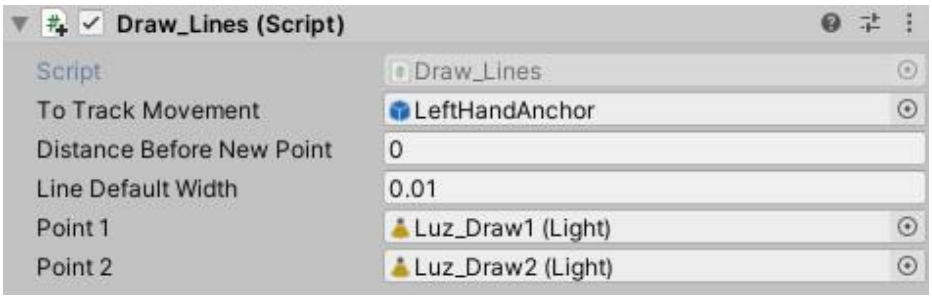

Figura 98: Draw_Lines

Se ha creado un objeto llamado ToTrackMovement al que se le ha asignado LeftHandAnchor de la escena. Se ha usado el mismo objeto para la mano derecha con el objeto RightHandAnchor. A continuación, se puede ver otro parámetro llamado DistanceBeforeNewPoint, el cual calcula cuántos puntos se necesitan tras haber pintado en el aire para volver a pintar. Ya que se quería que las líneas fueran continuas, se estableció este valor en 0. El posterior parámetro, llamado LineDefaultWidth establece el ancho de las líneas y puntos. Tras haber realizado diversas pruebas, se decidió que tuviera un valor de 0.01. Finalmente, los otros dos campos que se pueden ver corresponden a Point1 y Point2, relacionados con las dos luces que se han mencionado anteriormente. En este punto del desarrollo al mover las manos se creaban pinturas en el aire y seguían el movimiento de las manos, pero sin poder controlar el color ni la desactivación de la pintura.

Para darle solución, se decidió que se pintara con los gatillos delanteros de los mandos, usando en el Script los comandos OVRInput.GetDown(OVRInput.Button.PrimaryIndexTrigger) para la mano

izquierda y OVRInput.GetDown(OVRInput.Button.SecondaryIndexTrigger) para la mano derecha. Si se pulsa en uno sólo de los mandos, se pueden dibujar puntos en el aire. Si se pulsan los dos *triggers* a la vez, aparecerá una línea entre ambas manos. De esta manera se puede controlar la pintura a realizar. Teniendo en cuenta esta idea se propuso aprovechar los *triggers* laterales, correspondientes a los dedos corazón de ambas manos, de forma independiente para borrar cualquier pintura que se encontrara en el aire. Esto se puede hacer con cualquiera de los mandos de la misma manera que los *triggers* delanteros con los códigos OVRInput.GetDown(OVRInput.Button.PrimaryHandTrigger) y OVRInput.GetDown(OVRInput.Button.SecondaryHandTrigger). Así, el usuario es capaz de pintar y borrar pintando en el aire. El último paso que quedaba por realizar era el cambio de colores. Esto se hizo con los cuatro botones superiores, dos en cada mando, que mediante código se realizó con los comandos OVRInput.GetDown(OVRInput.Button.One), OVRInput.GetDown(OVRInput.Button.Two), OVRInput.GetDown(OVRInput.Button.Three) y OVRInput.GetDown(OVRInput.Button.Four), asignándoles un color a cada uno de ellos. Para realizar la programación de forma correcta, se buscaron los correspondientes enlaces [154]. También fue necesario desactivar la pintura cuando el usuario se encontrase en una escena diferente a la actual [155].

Con respecto al cubo verde de cambio de escena, nuevamente se recurre a un modificador AudioSource y un Script, en este caso con el nombre Escena 9. Para el campo de audio se le ha asignado el clip fin, como se puede ver en la Figura 99.

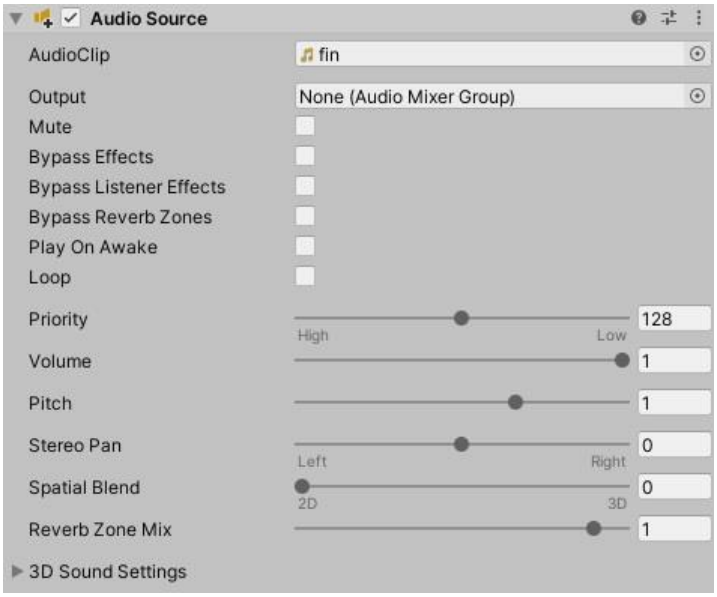

Figura 99: Audio Escena 9

En referencia al fichero de texto, se hizo uso de los mismos nombres usados en los anteriores Scripts. Así, el campo de audio se encuentra vacío y el objeto Objetoanterior lleva asignado el cubo Gris_Escena8. Esto se puede ver en la Figura 100.

Figura 100: Script Escena 9

Tras haber realizado toda la programación, la cual ha sido la más compleja de todo el proceso se consiguió construir la escena pensada. De esta manera, las líneas y puntos con los cuatro colores han sido desarrollados satisfactoriamente. Se puede ver en la Figura 101.

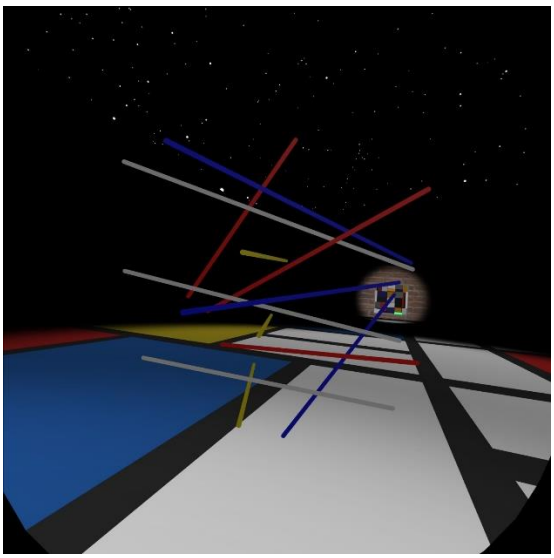

Figura 101: Pinturas en el aire

A la vista de los resultados anteriores, es posible ver las distintas perspectivas que ofrece esta técnica, así como todas las posibilidades que ofrece. Aunque cualquier usuario puede interactuar con esta experiencia, los artistas pueden enfocarla de una mejor manera y llevar el arte a otro límite. Estas perspectivas se pueden ver en la Figura 102.

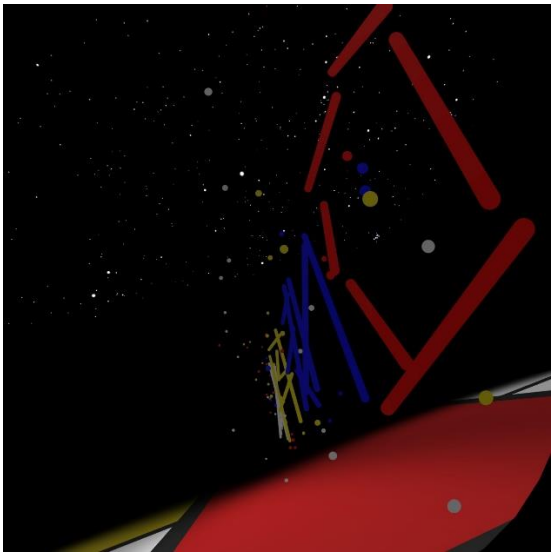

Figura 102: Perspectiva de pinturas en el aire

En la siguiente imagen se puede ver la muestra realizada en esta escena como homenaje a Mondrian. El autor ha sido la gran referencia de toda la experiencia y es por ello por lo que se ha tenido presente en este escenario concreto. Por ello mismo, como se puede ver en la Figura 103, se enseña una pequeña muestra del alcance de esta tecnología y lo que se podría conseguir en manos de grandes artistas y creadores.

Figura 103: Mondrian a pintura

Esta es quizás la escena más completa de la experiencia, ya permite tener visible en todo momento el cuadro del autor, inspiración de todo el conjunto, así como la posibilidad de jugar con el arte de la mejor manera posible. La inmersión, interacción, pintura y arte son visibles en todo momento gracias al trabajo realizado en este desarrollo tan complejo.

4.2.2.10. Escena 10

Llegados a este punto, se ha llegado a la última escena del desarrollo, que se puede ver en la Figura 104.

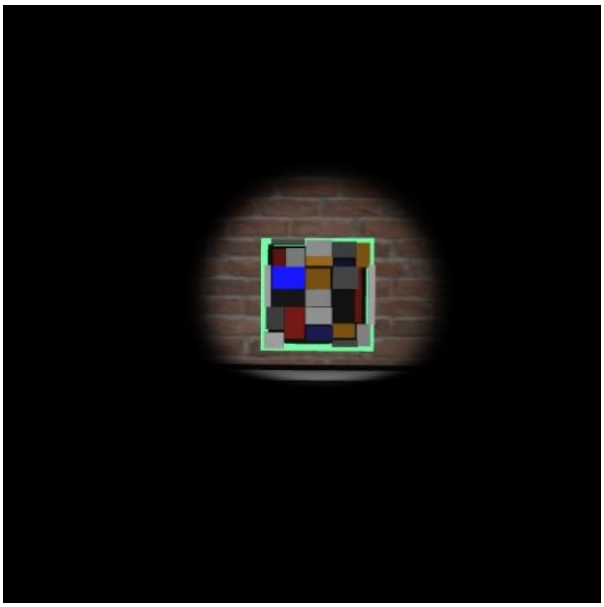

Figura 104: Escena final

En ella la audioguía agradece al usuario haber completado la experiencia y haber podido usar la aplicación. También se mencionan palabras textuales del pintor, haciendo este más cercano al usuario. Dado que es el final de la experiencia, se decidió aprovechar uno de los cubos azules de la pintura con el fin de cerrar del todo la sensación del usuario. Así, se propuso finalmente crear fuegos artificiales por encima de la habitación. Los fuegos se realizaron siguiendo la estructura de un vídeo encontrado [156]. Estos fuegos necesitan ser producidos mediante partículas [157, 158]. Para hacer un correcto uso de estos sistemas, fue necesario saber el funcionamiento de las partículas. Para ello se buscaron en varias fuentes [159, 160], intentando siempre que el resultado final fuera el mejor. Para conseguir la mejor experiencia de usuario, se necesitaba también que el movimiento estuviera acompañado del audio [161]. Al igual que todo lo relacionado con la experiencia, los fuegos artificiales son de colores rojos, amarillos y azules y aparecen en las posiciones designadas con objetos vacíos que se llaman Fuegos_rojos, Fuegos_azules y Fuegos_amarillos, que fueron agrupados todos en el paquete Fuegos.

En la Figura 105 se muestra el sistema de partículas usado para crear los fuegos rojos, llamado Rocket1. De igual manera se han usado para los otros colores, en sus respectivas posiciones. Hay que destacar que para que estos aparecieran desde cero, era necesario que fueran guardados como prefabs. Como se puede ver en los parámetros, fue necesario cambiar algunos valores [162] con el fin de adaptar las partículas a la experiencia.

Figura 105: Partículas de fuegos rojos

El resto de los colores siguen la misma estructura. El resultado final de los fuegos se puede ver en la Figura 106.

Figura 106: Fuegos artificiales

Con respecto al audio anteriormente comentado, se estableció mediante línea de comandos con un Script llamado Sound Explosion. El mismo se puede ver en la Figura 107.

Figura 107: Sonido de los fuegos

Este fichero contiene dos ficheros de audio, Fuegos_inicio y Fuegos_fin. El primero evalúa si se ha lanzado algún sistema de partículas y activa el primer audio. Posteriormente se detecta si esas partículas llegan a cero y, cuando ocurre, desactiva el audio anterior y ejecuta el segundo. Esto se realiza de forma repetitiva cada vez que un fuego es lanzado y estalla.

Para conseguir que los fuegos fueran activados y desactivados [163, 164], fue necesario crear un Script que se usara en el cubo azul. Esto se puede ver en la Figura 108.

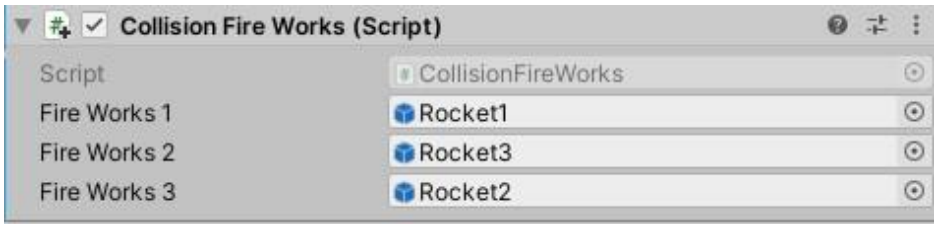

Figura 108: Script Fuegos Artificiales

Como se puede ver, se han asignado a los elementos FireWorks1, FireWorks2 y FireWorks3 los objetos pertenecientes a los sistemas de partículas creados. De esta manera, al tocar el correspondiente cubo azul, se pueden conseguir el efecto deseado. Esta es finalmente la última acción que se realiza en la experiencia.

En las Figuras 109 y 110 se puede ver el resultado completo de la Escena 10.

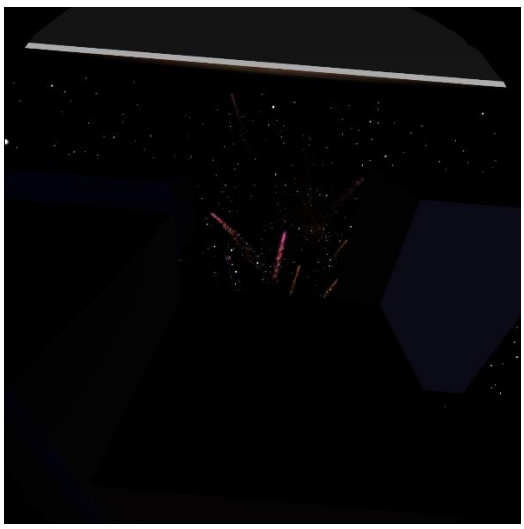

Figura 109: Fuegos artificiales I

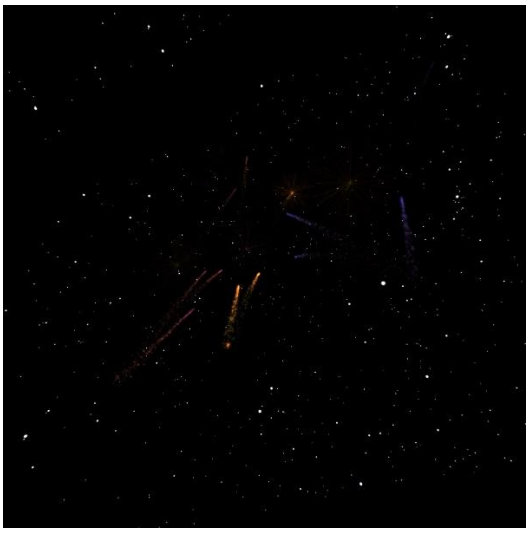

Figura 110: Fuegos artificiales II

Dado que es una aplicación educativa, se da la opción de poder iniciar la aplicación desde el principio. Esto se realiza de forma similar a los cubos verdes de las escenas anteriores. La diferencia se encuentra en que se realiza con la pared del fondo, que acompaña a la pintura y en esta escena se ilumina de verde. En la Figura 111 se puede ver el AudioSource usado como modificador.

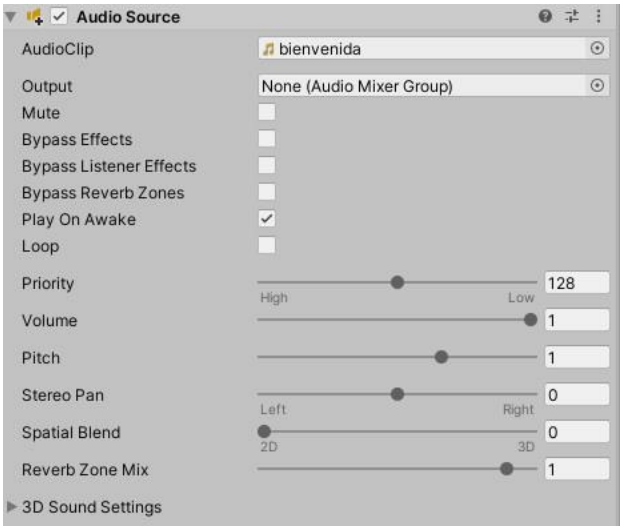

Figura 111: Audio Escena 10

Como se puede ver, el clip de audio, llamado bienvenida, es el que se activó en la Escena 1, dando de nuevo la bienvenida a la persona.

Para realizar esto, se hizo uso del Script Escena 10, que se puede ver en la Figura 112.

Figura 112: Script Escena 10

En este caso, el Objetoanterior es el cubo Gris_escena9.

Se ha conseguido desarrollar la experiencia completa, con todas las escenas de forma correcta y explicando la teoría del autor y el arte general. La inmersión conseguida en todo momento, así como la interacción con todos los elementos hace que esta experiencia sea una propuesta interesante para el arte. Llegados a este punto, ya ha sido desarrollada la mayor parte del trabajo, consiguiendo resultados muy satisfactorios y dotando a la aplicación de una calidad muy cuidada.

A partir de este punto queda cerrada la explicación referente a las distintas escenas. Se va a explicar una sección más dentro del trabajo de Unity que corresponde con los últimos detalles añadidos, puliendo y añadiendo los últimos objetos para tener en cuenta dentro de la experiencia.

4.2.2.11. Últimos detalles en Unity

Para conseguir el mejor resultado en la experiencia, el último tramo del desarrollo se enfocó en pulir los detalles y dar una mejor apariencia a toda la escena.

Lo primero que se hizo fue colocar correctamente las paredes con ladrillos de manera que resultara estética en la escena. Fueron colocadas dos líneas de muros, ya que por las limitaciones del programa el usuario podía atravesar con las gafas los ladrillos y ver tras ellos. Así, teniendo dos líneas de muros en cada lado, no se vería el espacio infinito que se tenía en un principio. La pared de ladrillos se puede ver en la Figura 113. La oscuridad de la escena se debe a la ausencia de luz que en este momento no existían.

Figura 113: Pared de ladrillos

En estos muros fueron añadidos dos modificadores. El primero de ellos, los respectivos Box Collider que se han usado en anteriores objetos y ya se explicó su función. El segundo modificador fue el llamado Rigidbody. Este se encarga de otorgar al objeto de físicas naturales que se pueden encontrar en el mundo real, como es el caso de la gravedad. En la Figura 114 se muestra el modificador de uno de los muros. Para el resto, se cumplen los mismos parámetros.

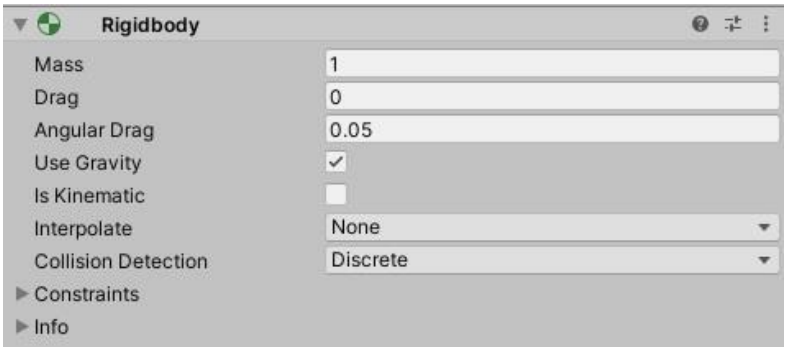

Figura 114: Rigidbody

A la vista de la Figura 113, así como las anteriores donde se podían ver las escenas, se puede ver que el cielo está a oscuras y repleto de estrellas. Para ello se descargó [165] una imagen preparada con las

dimensiones correctas que permitieran adaptarla al Skybox de Unity [166]. De esta forma, el entorno general de la experiencia estaría completamente desarrollado.

La siguiente idea que se quiso conseguir fue la de destacar correctamente los cubos activos en las correspondientes escenas. Como se puede ver en todas las Figuras de escena que mostraban la pintura en los distintos niveles, los cubos que influían en los escenarios se encontraban más iluminados que los demás. En un principio se pensó en colocar un foco frente a cada uno de ellos, sin embargo, el hecho de que aparecieran sombras en las zonas traseras de esos cubos, así como que no se pudiera conseguir iluminar el área frontal de estos fueron motivos más que suficientes para entender que el resultado no era agradable a la vista. Para solucionar ese problema y encontrar un sentido acorde a la filosofía del proyecto, finalmente se decidió que cada cubo emitiera su propia luz cuando estuviera activo, con la salvedad de los cubos negros y grises, que tendrían el color verde que destacaría en el conjunto como cambio de escena. Esto se realizó programando mediante código. Primero se buscó la forma de realizarlo [167, 168], y posteriormente se adaptó el texto a los colores buscados [169]. De esta manera, los cubos que realizan acciones se harían más visibles y sencillos para que el usuario interactuase con ellos.

Siguiendo la misma idea que en el párrafo anterior, se decidió incluir una acción más con el fin de destacar los cubos iluminados. Para ello se pensó en hacer que esos cubos vibraran, más rápido en los verdes y a menor ritmo en los demás. Teniendo en cuenta que el resto de los colores diferentes de los verdes tenían la misma frecuencia, es interesante ver que todos ellos se mueven al unísono, indicando que todos ellos forman parte de un mismo conjunto. Para ello fue necesario buscar los comandos necesarios para realizarlo [170], alterando la velocidad según la conveniencia con el proyecto. Así se conseguía tener todas las escenas de la manera más atractiva e intuitiva para la interacción.

Durante las pruebas de la experiencia completa se descubrieron algunos fallos en diversos sistemas ya que existían colisión entre ellos. Tras analizar los distintos objetos y Scripts relacionados, se descubrió que algunos parámetros debían ser programados de forma que se establecieran al iniciarse el juego, antes de que el usuario se encontrara dentro. Para solucionarlo, se hizo uso de la función Awake de Unity [171], la cual es la encargada de ejecutar comandos antes de comenzar la experiencia. Gracias a ella, se consiguieron las distintas ideas buscadas.

Posteriormente, se decidió mejorar la apariencia de las manos. Como se ha podido ver en las Figuras dónde aparecen las manos, estas se encontraban flotando en el aire. Para que el usuario sintiera las manos virtuales como las suyas propias se pensó en añadir unos brazos virtuales que acompañaran las manos. Tras varias propuestas, como podría ser el modelado de un brazo robótico o similar, finalmente se decidió prescindir de modelados externos. Con el fin de conseguir un dinamismo constante en todas las escenas, así como un cierto colorido, se designó que los brazos estuvieran formados por pequeños confetis formados por los cuatro colores principales que conforman la experiencia: rojo, amarillo, azul y blanco. La única manera de conseguir esto era con partículas que acompañarían a las manos, por lo que estas serían padres de los sistemas realizados [172]. Teniendo cuatro colores, la propuesta más interesante era usar dos colores para una mano y los otros dos para la otra. Así pues, teniendo en cuenta que el amarillo y el azul son colores complementarios, se juntaron estos dos en la mano izquierda y el blanco y rojo en la derecha, consiguiendo un contraste más visible para el sujeto. El sistema usado en la mano izquierda, dónde los parámetros tuvieron que ser modificados para que resultaran más acordes a la experiencia, se puede ver en la Figura 115. Usando los mismos parámetros, con los otros colores, se ha realizado para la derecha.

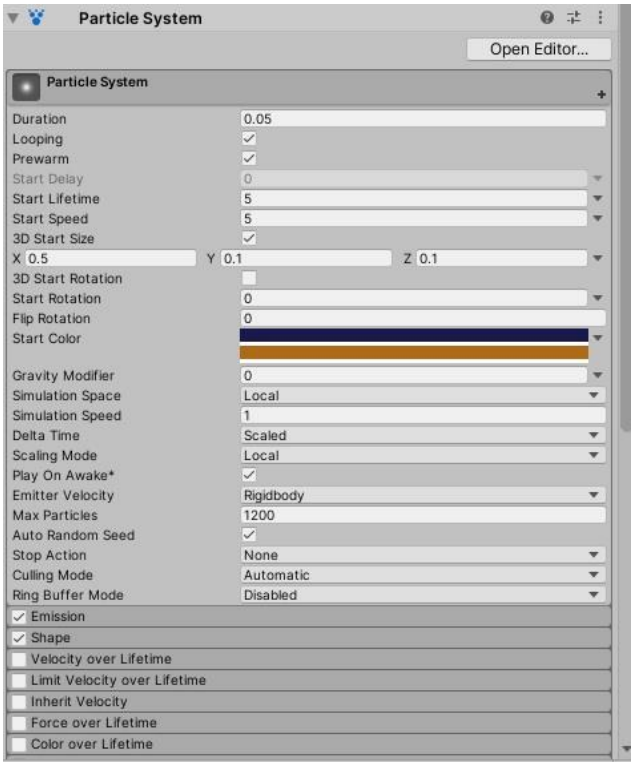

Figura 115: Sistema de partículas de manos

Las primeras pruebas se realizaron en un entorno diferente dónde solo hubiera un plano y fuese día, con el fin de colocar el sistema de partículas correctamente. Esto puede ver en la Figura 116.

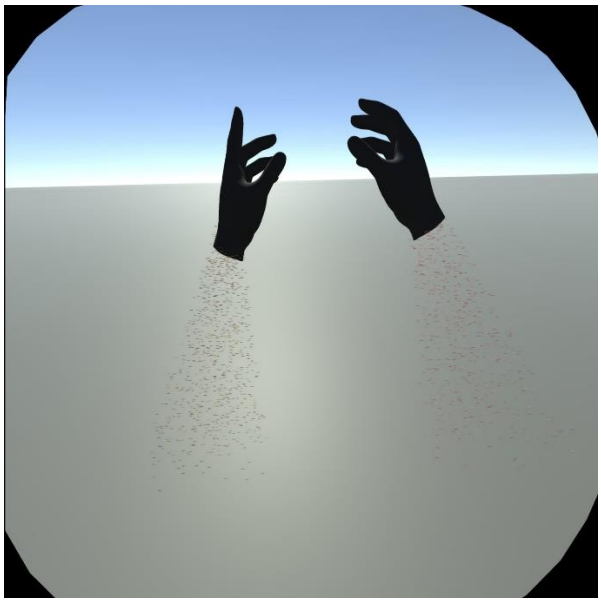

Figura 116: Partículas en las manos

Tras comprobar que el resultado coincidía con lo que se quería conseguir, se integró en la aplicación final, acompañado de todos los elementos que se han explicado con anterioridad. Esto se puede ver en la Figura 117.

Figura 117: Manos finales

Es interesante comprobar que el propio usuario se toca sus brazos teniendo la sensación de que las partículas que ve a través de las pantallas forman parte de su cuerpo.

Una vez desarrollados todos estos elementos se decidió incluir una música de fondo que tuviera relación con la experiencia. Esto se tuvo en cuenta por dos principales razones. La primera de ella venía de la concepción de videojuego que esta aplicación podía tener. Pese a tener un fin educativo y cultural, el concepto juego siempre se tuvo presente, como se ha podido comprobar en la Escena 9. Teniendo en mente esta idea, es conocida la importancia de una buena música de fondo acorde con lo que se ve. La segunda razón fueron los silencios que se producían tras los audios de las escenas. Una vez terminaba la audioguía, el usuario se encontraba en una habitación sin ningún tipo de sonido, lo que hacía que la sensación de inmersión se perdiera en gran medida. Para conseguir una música relacionada con la vida de Mondrian se pensó en dos estilos diferentes. El primero de ellos, el Jazz. Este estilo musical influenció de manera directa la pintura del autor, en concreto durante sus últimos años en Estados Unidos, los cuales fueron los más importantes desde un punto de vista artístico. Sin embargo, a pesar de buscar en diferentes lugares, ninguna de las melodías encontradas encajaba bien con el resto de los elementos. Se detectó que una música tranquila, que no distrajera del entorno ni solapara los audios de las explicaciones, sería la mejor opción para incluir. Como el estilo de música mencionado no funciona, se decide buscar según una segunda idea como es París.

Como se comentó en apartados anteriores, es durante su época en París cuando el autor se siente influenciado por el cubismo, técnica que le acompañaría hasta el final de sus días. Debido a ello, y tras realizar varias búsquedas de música francesa, se decide recurrir a la banda sonora de la película Amélie. Prestando atención de varias de las piezas que se interpretan se puede comprobar que la mayoría de ellas se adecuan a lo que se estaba buscando. Se decide entonces que la música de fondo sería Comptine d'un autre ete o La valse d'Amélie, ambas del compositor Yann Tiersen. Tras varias consultas, se decide usar el vals debido a que la otra música tiene unas subidas y bajadas de tonos que en ciertos momentos despistarían al usuario. Sabiendo la música que finalmente se iba a utilizar, se procede a analizar la interpretación de distintos músicos, con distintos instrumentos, de esta obra. Tras ello, se decide usar una interpretación del Vals d'Amélie a piano [173]. Con esta melodía la experiencia consigue que el usuario se sienta más tranquilo y pueda prestar más atención a lo que se

desarrolla a su alrededor. Para que sonara en la experiencia, se añadió un modificador AudioSource a OVRPlayerController, de manera que el sonido de la música siempre se encontraría junto al usuario.

Finalmente, referente a esta sección, es necesario comentar los parámetros de las luces. Para evitar que ciertas zonas aparecieran oscuras o se mostraran sombras sobre objetos que destacaban, fue necesario asignar gran cantidad de luces que se fueran activando y desactivando según les correspondieran en las escenas. A lo largo de toda la explicación se han mostrado algunos de estos parámetros, por lo que no serán explicados de nuevo. Como se pueden ver en las distintas Figuras de las luces, los parámetros y tipos de luces se han ido variando según se fuera demandando para el proyecto. Se han ido distribuyendo a lo largo de la habitación de forma que no se superpusieran unas con otras ni iluminarán más allá de los límites que se querían tener. Pese a no ser de gran complejidad, sí hay que mencionar que fue uno de los trabajos más entretenidos durante la creación completa, ya que cada cubo, cuadrado o elemento se comportaba de una manera diferente y con distintas dimensiones.

A continuación, se hará un repaso de las luces usadas dependiendo del elemento al que acompaña. En la Figura 118 se tiene una luz, llamada Luz_avion, que es hija del objeto Avion de la escena. Es la encargada de mantener en todo momento iluminado el avión en la escena correspondiente a los viajes del pintor.

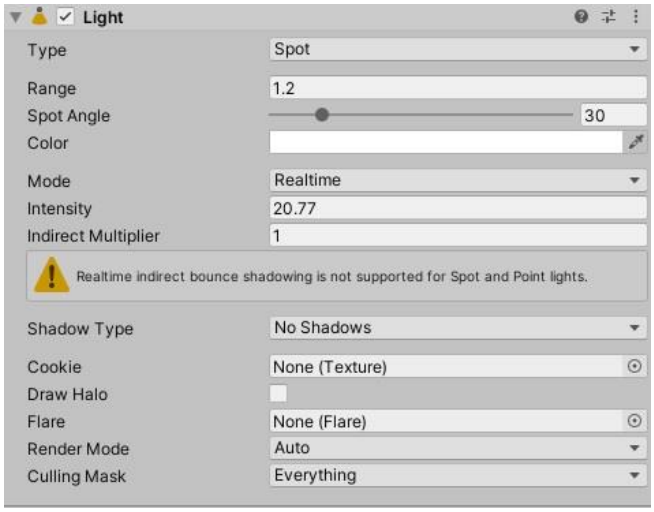

Figura 118: Luz del avión

En esta misma escena, al igual que en la Escena 2, se han asignado luces tanto al plano que se podía tocar (en este caso Cubismo y en la Escena 2 Mondrian_joven) como a los grandes planos dónde se reproducían los vídeos y las banderas colocadas en la pared. En este caso, también la Torre Eiffel fue iluminada por una luz específica.

Respecto a la Escena 8, dónde se mostraba una breve exposición con pinturas del artista, se colocó una luz frente a cada cuadro, de manera que estos siempre fueran visibles independientemente de la posición del usuario.

En el siguiente escenario, la Escena 9, se habilitaron dos luces más alejadas del cuadro donde se vieran bien los colores pintados en el aire. Es por ello por lo que aparece una escena más iluminada que invita a la persona a acercarse y experimentar en ese lugar.

Finalmente, se colocaron luces en cada una de las posiciones de activación de los fuegos artificiales de la Escena 10. Es curioso que, pese a ser fuegos artificiales y estar montados a partir de partículas,

la ausencia de luces en estos puntos hacía que las explosiones y estelas no fueran visibles. Para corregir este fallo, se colocaron las mencionadas luces apuntando hacia arriba, consiguiendo de esa manera que todo el proceso desde inicio hasta la desaparición del sistema quedara visible para el usuario. La imposibilidad de ver estas partículas resultaba chocante en la persona, ya que se producían los sonidos asignados, pero no se podía contemplar el resultado final construido. Los parámetros de una de estas luces se muestran en la Figura 119. En las otras dos luces se mantienen esos mismos números.

Figura 119: Luz de fuegos artificiales

Tras explicar estas últimas luces, se ha llegado al final de lo realizado en el programa Unity. Como se comentó en un principio, ha sido lo que más tiempo, dedicación y esfuerzo ha llevado. La complejidad de los elementos, la interacción entre ellos y la propia con el usuario ha supuesto un gran reto en el desarrollo final de la experiencia. Sin embargo, pese a todo, se han conseguido realizar correctamente todas las ideas propuestas con el fin de incrementar el interés del usuario en Mondrian y, más aún, en la pintura y arte en general.

En la siguiente sección se mostrarán las líneas de código de los distintos ficheros de texto usados, utilizando el lenguaje de programación C# y realizados en la aplicación Visual Studio, que han permitido en todo momento mantener la interacción entre objetos y la correcta inmersión del usuario en las escenas.

4.2.3. Códigos realizados en C#

4.2.3.1. Butterflyvlieg

En este fichero se ha programado el movimiento de las mariposas, teniendo especial cuidado en la velocidad de movimiento, la dirección aleatoria hacia la que se deben dirigir y la posibilidad de detectar otros elementos y de esa manera volver a buscar un destino nuevo.

```
using UnityEngine;
using System.Collections;
public class Butterflyvlieg : MonoBehaviour
{
    float delaytimer;
    Vector3 pos;
    private GameObject butterfly;
    void Start()
    {
        getNewPosition(); // get initial targetpos
    }

    void Update()
    {
        delaytimer += Time.deltaTime*0.2f;

        if (delaytimer > Random.Range(0.7f,1.7f)) // time to wait
        {
            getNewPosition(); //get new position every 1 second
            delaytimer = 0f; // reset timer
        }
        transform.position = Vector3.MoveTowards(transform.position, pos, .01f);        //Direccion por
        cuadro
    }

    private void Awake()
    {
        butterfly = GameObject.Find("SCharacter_Butterfly");
    }

    private void OnTriggerEnter(Collider other)
    {
        if (other.gameObject)
        {
            getNewPosition();               //Choca con cubo, busca nueva posicion
        }
    }
    void getNewPosition()
    {
        float x = Random.Range(-2.3f, 1f);
        float y = Random.Range(-0.925f, 1.545f);
        float z = Random.Range(-120.5f, -118.5f);

        pos = new Vector3(x, y, z);
    }
}
```

4.2.3.2. CollisionAudio

Este código permite activar los sonidos correspondientes a un determinado cubo. Se ha usado el fichero en cada uno de ellos, cambiando el clip de audio según la conveniencia. Además, se ha programado el movimiento de vibración de los cubos de interés.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CollisionAudio : MonoBehaviour
{
    private GameObject leftHand, rightHand;
    public AudioSource AudioSource;

    private float speed = 10.0f;
    private float amount = 0.005f;
    Vector3 initialposition;

    // Start is called before the first frame update
    void Start()
    {
        AudioSource = GetComponent<AudioSource>();
    }

    // Update is called once per frame
    void Update()
    {
        if (GlobalVar.Escena == 5f)
        {
         transform.position = new Vector3(initialposition.x + (Mathf.Sin(Time.time * speed) * amount),
        initialposition.y   +   (Mathf.Sin(Time.time   *   speed)   *   amount),   initialposition.z   +
        (Mathf.Sin(Time.time * speed) * amount));
        }
    }

    private void Awake()
    {
        leftHand = GameObject.Find("LeftHandAnchor");
        rightHand = GameObject.Find("RightHandAnchor");

        initialposition.x = transform.position.x;
        initialposition.y = transform.position.y;
        initialposition.z = transform.position.z;
    }

    private void OnTriggerEnter(Collider other)
    {
        if((other.gameObject == leftHand || rightHand) && GlobalVar.Activado == 2f)
        {
            AudioSource.Play(0);
        }
    }
}
```

4.2.3.3. CollisionFireWorks

El fichero se encarga de activar y desactivar los fuegos artificiales de la última escena. Busca los objetos vacíos para posteriormente instanciar los fuegos previamente preparados. Al pulsar de nuevo el cubo de activación, este fichero destruye las partículas presentes en la escena, de forma que se puede encender y apagar los fuegos artificiales.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CollisionFireWorks : MonoBehaviour
{
    private GameObject leftHand, rightHand;
    public GameObject FireWorks1;
    public GameObject FireWorks2;
    public GameObject FireWorks3;
    private GameObject rojos, azules, amarillos;
    private static float bandera;
    private GameObject vacio1;
    private GameObject vacio2;
    private GameObject vacio3;

    private float speed = 10.0f;
    private float amount = 0.005f;
    Vector3 initialposition;

    // Start is called before the first frame update
    void Start()
    {
        bandera = 1f;
    }

    // Update is called once per frame
    void Update()
    {
        if (GlobalVar.fuegos == 0f)
        {
            Destroy(rojos);
            Destroy(azules);
            Destroy(amarillos);

        }


        if (GlobalVar.Escena == 10f)
        {
         transform.position = new Vector3(initialposition.x + (Mathf.Sin(Time.time * speed) * amount),
        initialposition.y   +   (Mathf.Sin(Time.time   *   speed)   *   amount),   initialposition.z   +
        (Mathf.Sin(Time.time * speed) * amount));
        }

    }

    private void Awake()
    {
        leftHand = GameObject.Find("LeftHandAnchor");
        rightHand = GameObject.Find("RightHandAnchor");
        vacio1 = GameObject.Find("Fuegos_rojos");
        vacio2 = GameObject.Find("Fuegos_azules");
        vacio3 = GameObject.Find("Fuegos_amarillos");

        initialposition.x = transform.position.x;
        initialposition.y = transform.position.y;
        initialposition.z = transform.position.z;
    }


    private void OnTriggerEnter(Collider other)
    {
```

```
        if ((other.gameObject == leftHand || rightHand) && GlobalVar.Activado == 6f)
        {
            if (bandera == 1f)
            {
                rojos = Instantiate(FireWorks1, vacio1.transform.position, vacio1.transform.rotation);
                azules = Instantiate(FireWorks2, vacio2.transform.position, vacio2.transform.rotation);
                amarillos = Instantiate(FireWorks3, vacio3.transform.position,
                vacio3.transform.rotation);

                bandera = 0f;
            }
            else
            {
                Destroy(rojos);
                Destroy(azules);
                Destroy(amarillos);

                bandera = 1f;
            }
        }
    }
}
```

4.2.3.4. Crea_Mariposas

De igual manera que con el fichero anterior, este es el encargado de hacer aparecer las mariposas. Si anteriormente se explicó el script correspondiente al movimiento de los insectos, en este caso, la idea es únicamente crearlos en los lugares correspondientes.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Crea_Mariposas : MonoBehaviour
{
    public GameObject prefab;
    private GameObject vacio1, vacio2, vacio3;
    private GameObject leftHand, rightHand;
    private static float bandera;
    private GameObject mariposa1, mariposa2, mariposa3, mariposa4, mariposa5, mariposa6, mariposa7,
mariposa8, mariposa9, mariposa10, mariposa11, mariposa12;

    private float speed = 10.0f;
    private float amount = 0.005f;
    Vector3 initialposition;

    // Start is called before the first frame update
    void Start()
    {
        bandera = 1f;
    }

    private void Awake()
    {
        leftHand = GameObject.Find("LeftHandAnchor");
        rightHand = GameObject.Find("RightHandAnchor");
        vacio1 = GameObject.Find("Butterfly1");
        vacio2 = GameObject.Find("Butterfly2");
        vacio3 = GameObject.Find("Butterfly3");

        initialposition.x = transform.position.x;
        initialposition.y = transform.position.y;
        initialposition.z = transform.position.z;
    }

    // Update is called once per frame
    void Update()
    {
        if (GlobalVar.mariposa == 0f)
        {
            Destroy(mariposa1);
            Destroy(mariposa2);
            Destroy(mariposa3);
            Destroy(mariposa4);
            Destroy(mariposa5);
            Destroy(mariposa6);
            Destroy(mariposa7);
            Destroy(mariposa8);
            Destroy(mariposa9);
            Destroy(mariposa10);
            Destroy(mariposa11);
            Destroy(mariposa12);
        }
        if(GlobalVar.Escena == 3f)
        {
         transform.position = new Vector3(initialposition.x + (Mathf.Sin(Time.time * speed) * amount),
        initialposition.y   +   (Mathf.Sin(Time.time   *   speed)   *   amount),   initialposition.z   +
        (Mathf.Sin(Time.time * speed) * amount));
        }

    }
    private void OnTriggerEnter(Collider other)
    {
```

```
        if ((other.gameObject == leftHand || rightHand) && GlobalVar.Activado == 1f)
        {
            if(bandera ==1f)
            {
                mariposa1 = Instantiate(prefab, vacio1.transform.position, vacio1.transform.rotation);
                mariposa2 = Instantiate(prefab, vacio1.transform.position, vacio1.transform.rotation);
                mariposa3 = Instantiate(prefab, vacio1.transform.position, vacio1.transform.rotation);
                mariposa4 = Instantiate(prefab, vacio1.transform.position, vacio1.transform.rotation);
                mariposa5 = Instantiate(prefab, vacio2.transform.position, vacio2.transform.rotation);
                mariposa6 = Instantiate(prefab, vacio2.transform.position, vacio2.transform.rotation);
                mariposa7 = Instantiate(prefab, vacio2.transform.position, vacio2.transform.rotation);
                mariposa8 = Instantiate(prefab, vacio2.transform.position, vacio2.transform.rotation);
                mariposa9 = Instantiate(prefab, vacio3.transform.position, vacio3.transform.rotation);
                mariposa10 = Instantiate(prefab, vacio3.transform.position, vacio3.transform.rotation);
                mariposa11 = Instantiate(prefab, vacio3.transform.position, vacio3.transform.rotation);
                mariposa12 = Instantiate(prefab, vacio3.transform.position, vacio3.transform.rotation);


                bandera = 0f;

            }
            else
            {
                Destroy(mariposa1);
                Destroy(mariposa2);
                Destroy(mariposa3);
                Destroy(mariposa4);
                Destroy(mariposa5);
                Destroy(mariposa6);
                Destroy(mariposa7);
                Destroy(mariposa8);
                Destroy(mariposa9);
                Destroy(mariposa10);
                Destroy(mariposa11);
                Destroy(mariposa12);
                bandera = 1f;
            }
        }
    }
}
```

4.2.3.5. Cubismo

Es el encargado de activar la bandera y el vídeo de las calles de Estados Unidos, las luces de esa escena y los materiales de la Torre Eiffel y el avión. El cuadro con la imagen cubista detecta el momento en que cualquiera de las manos interactúa con él y sirve como sensor de activación. El fichero Juventud sigue la misma lógica que este, activando el primer vídeo de la aplicación y la bandera de Holanda.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Video;

public class cubismo : MonoBehaviour
{
    private GameObject leftHand, rightHand;
    float intensityMultiplier = 1f;
    float intensityMultiplieroff = 0f;

    public GameObject bandera1;
    public GameObject bandera2;
    public GameObject bandera3;

    public GameObject avion1;
    public GameObject avion2;
    public GameObject avion3;
    public GameObject avion4;
    public GameObject avion5;
    public GameObject avion6;
    public GameObject avion7;
    public GameObject avion8;
    public GameObject avion9;

    public GameObject torre;

    public Collider collider1;
    public Collider collider2;

    public GameObject fondo_video2;
    private VideoPlayer videoPlayer;


    public Light point1;
    float pointInitialIntensity1;
    public Light point2;
    float pointInitialIntensity2;
    public Light point3;
    float pointInitialIntensity3;
    public Light point4;
    float pointInitialIntensity4;

    // Start is called before the first frame update
    void Start()
    {
        point1.intensity = pointInitialIntensity1 * intensityMultiplieroff;
        point2.intensity = pointInitialIntensity2 * intensityMultiplieroff;
        point3.intensity = pointInitialIntensity3 * intensityMultiplieroff;
        point4.intensity = pointInitialIntensity4 * intensityMultiplieroff;
    }

    private void Awake()
    {
        leftHand = GameObject.Find("LeftHandAnchor");
        rightHand = GameObject.Find("RightHandAnchor");

        pointInitialIntensity1 = point1.intensity;
        pointInitialIntensity2 = point2.intensity;
        pointInitialIntensity3 = point3.intensity;
        pointInitialIntensity4 = point4.intensity;
```

```
        bandera1.GetComponent<Renderer>().enabled = false;
        bandera2.GetComponent<Renderer>().enabled = false;
        bandera3.GetComponent<Renderer>().enabled = false;

        avion1.GetComponent<Renderer>().enabled = false;
        avion2.GetComponent<Renderer>().enabled = false;
        avion3.GetComponent<Renderer>().enabled = false;
        avion4.GetComponent<Renderer>().enabled = false;
        avion5.GetComponent<Renderer>().enabled = false;
        avion6.GetComponent<Renderer>().enabled = false;
        avion7.GetComponent<Renderer>().enabled = false;
        avion8.GetComponent<Renderer>().enabled = false;
        avion9.GetComponent<Renderer>().enabled = false;

        torre.GetComponent<Renderer>().enabled = false;

        fondo_video2 = GameObject.Find("fondo_videos2");
        fondo_video2.GetComponent<Renderer>().enabled = false;
        videoPlayer = fondo_video2.GetComponent<VideoPlayer>();

    }
    // Update is called once per frame
    void Update()
    {
        if (GlobalVar.cubismo == 2f)
        {
            point1.intensity = pointInitialIntensity1 * intensityMultiplieroff;
            point2.intensity = pointInitialIntensity2 * intensityMultiplieroff;
            point3.intensity = pointInitialIntensity3 * intensityMultiplieroff;
            point4.intensity = pointInitialIntensity4 * intensityMultiplieroff;

            bandera1.GetComponent<Renderer>().enabled = false;
            bandera2.GetComponent<Renderer>().enabled = false;
            bandera3.GetComponent<Renderer>().enabled = false;
            torre.GetComponent<Renderer>().enabled = false;

            collider1.isTrigger = true;
            collider2.isTrigger = true;

            avion1.GetComponent<Renderer>().enabled = false;
            avion2.GetComponent<Renderer>().enabled = false;
            avion3.GetComponent<Renderer>().enabled = false;
            avion4.GetComponent<Renderer>().enabled = false;
            avion5.GetComponent<Renderer>().enabled = false;
            avion6.GetComponent<Renderer>().enabled = false;
            avion7.GetComponent<Renderer>().enabled = false;
            avion8.GetComponent<Renderer>().enabled = false;
            avion9.GetComponent<Renderer>().enabled = false;

            fondo_video2.GetComponent<Renderer>().enabled = false;
            videoPlayer.Stop();
        }
    }

    private void OnTriggerEnter(Collider other)
    {
        if ((other.gameObject == leftHand || rightHand) && GlobalVar.cubismo == 1f)
        {
            point1.intensity = pointInitialIntensity1 * intensityMultiplier;
            point2.intensity = pointInitialIntensity2 * intensityMultiplier;
            point3.intensity = pointInitialIntensity3 * intensityMultiplier;
            point4.intensity = pointInitialIntensity4 * intensityMultiplier;

            bandera1.GetComponent<Renderer>().enabled = true;
            bandera2.GetComponent<Renderer>().enabled = true;
            bandera3.GetComponent<Renderer>().enabled = true;
            torre.GetComponent<Renderer>().enabled = true;

            collider1.isTrigger = false;
            collider2.isTrigger = false;

            avion1.GetComponent<Renderer>().enabled = true;
            avion2.GetComponent<Renderer>().enabled = true;
            avion3.GetComponent<Renderer>().enabled = true;
```

```
            avion4.GetComponent<Renderer>().enabled = true;
            avion5.GetComponent<Renderer>().enabled = true;
            avion6.GetComponent<Renderer>().enabled = true;
            avion7.GetComponent<Renderer>().enabled = true;
            avion8.GetComponent<Renderer>().enabled = true;
            avion9.GetComponent<Renderer>().enabled = true;

            fondo_video2.GetComponent<Renderer>().enabled = true;
            videoPlayer.Play();
        }
    }
}
```

4.2.3.6. Day

Código de texto encargado de modificar la luz del sol al 100%. De forma similar, se han usado los ficheros Evening, MidNight y Morning para asignarle valores del 30%, 0% y 70% respectivamente.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Day : MonoBehaviour
{
    public Light sun;
    float sunInitialIntensity;
    float intensityMultiplieroff = 0f;
    float intensityMultiplier = 1f;
    private GameObject leftHand, rightHand;

    private float speed = 10.0f;
    private float amount = 0.005f;
    Vector3 initialposition;

    // Start is called before the first frame update
    void Start()
    {
        sun.intensity = intensityMultiplieroff;
    }

    // Update is called once per frame
    void Update()
    {
        if (GlobalVar.Escena == 6f)
        {
         transform.position = new Vector3(initialposition.x + (Mathf.Sin(Time.time * speed) * amount),
        initialposition.y   +   (Mathf.Sin(Time.time   *   speed)   *   amount),   initialposition.z   +
        (Mathf.Sin(Time.time * speed) * amount));
        }
    }

    private void Awake()
    {
        leftHand = GameObject.Find("LeftHandAnchor");
        rightHand = GameObject.Find("RightHandAnchor");
        sunInitialIntensity = sun.intensity;

        initialposition.x = transform.position.x;
        initialposition.y = transform.position.y;
        initialposition.z = transform.position.z;
    }

    private void OnTriggerEnter(Collider other)
    {
        if ((other.gameObject == leftHand || rightHand) && GlobalVar.Activado == 3f)
        {
            sun.transform.localRotation = Quaternion.Euler(90, 170, 0);
            intensityMultiplier = 0.7f;
            sun.intensity = sunInitialIntensity * intensityMultiplier;

            GlobalVar.VarMid = 0f;
            GlobalVar.VarMorning = 0f;
            GlobalVar.VarDay = 1f;
            GlobalVar.VarEvening = 0f;
        }
    }
}
```

4.2.3.7. Draw_Lines

Este es quizás uno de los ficheros más complejos del desarrollo, ya que es el encargado de pintar en el aire formando puntos y líneas rectas, además de evaluar el botón pulsado para poder decidir el color. Pese a todo, se ha conseguido programar correctamente.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Draw_Lines : MonoBehaviour
{
    private Color fadecolor = Color.white;

    public GameObject ToTrackMovement;
    private Vector3 PointDistanceR = Vector3.zero;
    private Vector3 PointDistanceL = Vector3.zero;
    public float DistanceBeforeNewPoint = 0f;
    public float lineDefaultWidth = 0.01f;
    private int positionCount = 0;
    private List<LineRenderer> lines = new List<LineRenderer>();
    private LineRenderer currentLineRender;

    float intensityMultiplier = 1f;
    float intensityMultiplieroff = 0f;
    public Light point1;
    float pointInitialIntensity1;
    public Light point2;
    float pointInitialIntensity2;
    // Start is called before the first frame update
    void Start()
    {
        point1.intensity = pointInitialIntensity1 * intensityMultiplieroff;
        point2.intensity = pointInitialIntensity2 * intensityMultiplieroff;
    }

    private void Awake()
    {
        pointInitialIntensity1 = point1.intensity;
        pointInitialIntensity2 = point2.intensity;
    }

    // Update is called once per frame
    void Update()
    {
        if (GlobalVar.Escena == 9f)
        {
            point1.intensity = pointInitialIntensity1 * intensityMultiplier;
            point2.intensity = pointInitialIntensity2 * intensityMultiplier;

            if (OVRInput.GetDown(OVRInput.Button.PrimaryHandTrigger))
            {
                Destroylines();
            }
            if (OVRInput.GetDown(OVRInput.Button.SecondaryHandTrigger))
            {
                Destroylines();
            }

            if (OVRInput.GetDown(OVRInput.Button.PrimaryIndexTrigger))
            {
                UpdateLine(0);
            }

            if (OVRInput.GetUp(OVRInput.Button.PrimaryIndexTrigger))
            {
                AddNewLineRenderer(0);

            }
```

```
        if (OVRInput.GetDown(OVRInput.Button.SecondaryIndexTrigger))
        {
            UpdateLine(1);
        }

        if (OVRInput.GetUp(OVRInput.Button.SecondaryIndexTrigger))
        {
            AddNewLineRenderer(1);
        }

        if (OVRInput.GetDown(OVRInput.Button.One))
        {
            fadecolor = Color.red;
        }

        if (OVRInput.GetDown(OVRInput.Button.Two))
        {
            fadecolor = Color.blue;
        }

        if (OVRInput.GetDown(OVRInput.Button.Three))
        {
            fadecolor = Color.white;
        }

        if (OVRInput.GetDown(OVRInput.Button.Four))
        {
            fadecolor = Color.yellow;
        }
    }
    else
    {
        point1.intensity = pointInitialIntensity1 * intensityMultiplieroff;
        point2.intensity = pointInitialIntensity2 * intensityMultiplieroff;
        Destroylines();
    }


}

void UpdateLine(float manos)
{
    if (manos == 0f)
    {
        if (PointDistanceL == null)
        {
            PointDistanceL = GameObject.Find("LeftHandAnchor").transform.position;
        }

        if     (PointDistanceL     !=     null     &&     Mathf.Abs(Vector3.Distance(PointDistanceL,
        GameObject.Find("LeftHandAnchor").transform.position)) >= DistanceBeforeNewPoint)
        {
            Vector3     dirL     =     (GameObject.Find("LeftHandAnchor").transform.position     -
            Camera.main.transform.position).normalized;
            PointDistanceL = GameObject.Find("LeftHandAnchor").transform.position;
            AddPoint(PointDistanceL, dirL);
        }
    }
    else
    {
        if (PointDistanceR == null)
        {
            PointDistanceR = GameObject.Find("RightHandAnchor").transform.position;
        }

        if     (PointDistanceR     !=     null     &&     Mathf.Abs(Vector3.Distance(PointDistanceR,
        GameObject.Find("RightHandAnchor").transform.position)) >= DistanceBeforeNewPoint)
        {
            Vector3     dirR     =     (GameObject.Find("RightHandAnchor").transform.position     -
            Camera.main.transform.position).normalized;
            PointDistanceR = GameObject.Find("RightHandAnchor").transform.position;
            AddPoint(PointDistanceR, dirR);
        }
    }
```

```
    }

    void AddPoint(Vector3 position, Vector3 direction)
    {
        currentLineRender.SetPosition(positionCount, position);
        positionCount++;
        currentLineRender.positionCount = positionCount + 1;
        currentLineRender.SetPosition(positionCount, position);
    }

    void Destroylines()
    {
        foreach (LineRenderer p in lines)
        {
            Destroy(p);
        }
    }

    void AddNewLineRenderer(float manos)
    {
        if (manos == 0f)
        {
            positionCount = 0;
            GameObject go = new GameObject($"LineRenderer_");
            go.transform.parent = GameObject.Find("LeftHandAnchor").transform.parent;
            go.transform.position = GameObject.Find("LeftHandAnchor").transform.position;
            LineRenderer goLineRenderer = go.AddComponent<LineRenderer>();
            goLineRenderer.startWidth = lineDefaultWidth;
            goLineRenderer.endWidth = lineDefaultWidth;
            goLineRenderer.useWorldSpace = true;
            goLineRenderer.material.color = fadecolor;
            goLineRenderer.positionCount = 1;
            goLineRenderer.numCapVertices = 90;
            goLineRenderer.SetPosition(0, GameObject.Find("LeftHandAnchor").transform.position);

            currentLineRender = goLineRenderer;
            lines.Add(goLineRenderer);
        }
        else
        {
            positionCount = 0;
            GameObject go = new GameObject($"LineRenderer_");
            go.transform.parent = GameObject.Find("RightHandAnchor").transform.parent;
            go.transform.position = GameObject.Find("RightHandAnchor").transform.position;
            LineRenderer goLineRenderer = go.AddComponent<LineRenderer>();
            goLineRenderer.startWidth = lineDefaultWidth;
            goLineRenderer.endWidth = lineDefaultWidth;
            goLineRenderer.useWorldSpace = true;
            goLineRenderer.material.color = fadecolor;
            goLineRenderer.positionCount = 1;
            goLineRenderer.numCapVertices = 90;
            goLineRenderer.SetPosition(0, GameObject.Find("RightHandAnchor").transform.position);

            currentLineRender = goLineRenderer;
            lines.Add(goLineRenderer);
        }
    }
}
```

4.2.3.8. Escena1

Fichero de texto encargado de activar y desactivar los primeros elementos. Activa los elementos correspondientes a la escena actual y anula los de la escena anterior, que en este caso sería el último escenario de la aplicación. De forma similar se ha realizado para el resto de escena, con los nombres de los ficheros correspondientes a la escena indicada, teniendo especial cuidado en los elementos que debían funcionar y apagarse dentro del cuadro, así como los correspondientes elementos externos que pudieran ocurrir en cada situación. Tan sólo se mostrará el fichero de esta escena, sabiendo que las demás siguen el mismo patrón con las adaptaciones necesarias.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Escena1 : MonoBehaviour
{
    private GameObject leftHand, rightHand;
    float intensityMultiplier = 1f;
    float intensityMultiplieroff = 0f;
    public AudioSource AudioSource;

    public GameObject objetoanterior;
    private AudioSource audioanterior;

    public GameObject cuadro1;

    public Light point1;
    float pointInitialIntensity1;


    private float speed = 30.0f;
    private float amount = 0.005f;
    Vector3 initialposition;

    private Color color = new Color(5f / 255f, 75f / 255f, 18f / 255f);

    // Start is called before the first frame update
    void Start()
    {
        AudioSource = GetComponent<AudioSource>();

        point1.intensity = pointInitialIntensity1 * intensityMultiplieroff;

        audioanterior = objetoanterior.GetComponent<AudioSource>();

    }

    private void Awake()
    {
        leftHand = GameObject.Find("LeftHandAnchor");
        rightHand = GameObject.Find("RightHandAnchor");


        pointInitialIntensity1 = point1.intensity;

        cuadro1.GetComponent<Renderer>().enabled = false;

        initialposition.x = transform.position.x;
        initialposition.y = transform.position.y;
        initialposition.z = transform.position.z;
    }

    // Update is called once per frame
    void Update()
    {

        if(GlobalVar.Escena == 1f)
        {
```

```
            transform.position = new Vector3(initialposition.x + (Mathf.Sin(Time.time * speed) *
amount), initialposition.y + (Mathf.Sin(Time.time * speed) * amount), initialposition.z +
(Mathf.Sin(Time.time * speed) * amount));
            this.GetComponent<Renderer>().material.SetColor("_EmissionColor", color * 4f);
            this.GetComponent<Renderer>().material.EnableKeyword("_EMISSION");
        }
        else{
            this.GetComponent<Renderer>().material.DisableKeyword("_EMISSION");
        }
    }
    private void OnTriggerEnter(Collider other)
    {
        if ((other.gameObject == leftHand || rightHand) && GlobalVar.Escena == 1f)
        {
            AudioSource.Play(0);
            GlobalVar.Escena = 2f;
            GlobalVar.Activado = 0f;

            audioanterior.Stop();

            point1.intensity = pointInitialIntensity1 * intensityMultiplier;

            cuadro1.GetComponent<Renderer>().enabled = true;

            GlobalVar.juventud = 1f;
        }
    }
}
```

4.2.3.9. GlobalVar

Este script es el encargado de almacenar las variables globales que son usadas en los distintos ficheros, todo ello con el fin de tener comunicación entre ellos.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class GlobalVar : MonoBehaviour
{
    // Start is called before the first frame update

    public static float Bandera = 1f;
    public static float VarColor = 0f;
    public static float VarMid = 0f;
    public static float VarMorning = 0f;
    public static float VarDay = 0f;
    public static float VarEvening = 0f;
    public static float Escena = 1f;
    public static float Activado = 0f;
    public static float mariposa = 0f;
    public static float fuegos = 0f;
    public static float juventud = 0f;
    public static float cubismo = 0f;
}
```

4.2.3.10. Luz_Roja

Este código permite encender las luces rojas sobre el suelo en la escena correspondiente a las luces de colores. Se ha programado de manera que al tocar el cubo se encienda la luz de color. Si se vuelve a pulsar, cambia a blanca, y así sucesivamente. Esto se ha preparado para que al cambiar de color se siga manteniendo el mismo comportamiento. Se ha creado otros dos ficheros llamados Luz_Amarilla y Luz_Azul siguiendo la misma estructura de programación y adaptados a las luces de colores de los nombres.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Luz_Roja : MonoBehaviour
{
    float intensityMultiplier = 1f;
    float intensityMultiplieroff = 0f;
    public Light point1;
    float pointInitialIntensity1;
    public Light point2;
    float pointInitialIntensity2;
    public Light point3;
    float pointInitialIntensity3;
    public Light point4;
    float pointInitialIntensity4;
    public Light point5;
    float pointInitialIntensity5;
    public Light point6;
    float pointInitialIntensity6;
    public Light point7;
    float pointInitialIntensity7;
    public Light point8;
    float pointInitialIntensity8;
    public Light point9;
    float pointInitialIntensity9;
    public Light point10;
    float pointInitialIntensity10;
    public Light point11;
    float pointInitialIntensity11;
    public Light point12;
    float pointInitialIntensity12;
    public Light point13;
    float pointInitialIntensity13;
    public Light point14;
    float pointInitialIntensity14;
    private GameObject leftHand, rightHand;
    public Light Azul1;
    float AzulInitialIntensity1;
    public Light Azul2;
    float AzulInitialIntensity2;
    public Light Azul3;
    float AzulInitialIntensity3;

    public Light Rojo1;
    float RojoInitialIntensity1;
    public Light Rojo2;
    float RojoInitialIntensity2;
    public Light Rojo3;
    float RojoInitialIntensity3;

    public Light Amarillo1;
    float AmarilloInitialIntensity1;
    public Light Amarillo2;
    float AmarilloInitialIntensity2;
    public Light Amarillo3;
    float AmarilloInitialIntensity3;
    public Light Amarillo4;
    float AmarilloInitialIntensity4;
    public Light Amarillo5;
```

```
float AmarilloInitialIntensity5;
public Light Amarillo6;
float AmarilloInitialIntensity6;
public Light Amarillo7;
float AmarilloInitialIntensity7;
public Light Amarillo8;
float AmarilloInitialIntensity8;

private float speed = 10.0f;
private float amount = 0.005f;
Vector3 initialposition;

// Start is called before the first frame update
void Start()
{
    point1.intensity = pointInitialIntensity1 * intensityMultiplieroff;
    point2.intensity = pointInitialIntensity2 * intensityMultiplieroff;
    point3.intensity = pointInitialIntensity3 * intensityMultiplieroff;
    point4.intensity = pointInitialIntensity4 * intensityMultiplieroff;
    point5.intensity = pointInitialIntensity5 * intensityMultiplieroff;
    point6.intensity = pointInitialIntensity6 * intensityMultiplieroff;
    point7.intensity = pointInitialIntensity7 * intensityMultiplieroff;
    point8.intensity = pointInitialIntensity8 * intensityMultiplieroff;
    point9.intensity = pointInitialIntensity9 * intensityMultiplieroff;
    point10.intensity = pointInitialIntensity10 * intensityMultiplieroff;
    point11.intensity = pointInitialIntensity11 * intensityMultiplieroff;
    point12.intensity = pointInitialIntensity12 * intensityMultiplieroff;
    point13.intensity = pointInitialIntensity13 * intensityMultiplieroff;
    point14.intensity = pointInitialIntensity14 * intensityMultiplieroff;

    Rojo1.intensity = RojoInitialIntensity1 * intensityMultiplieroff;
    Rojo2.intensity = RojoInitialIntensity2 * intensityMultiplieroff;
    Rojo3.intensity = RojoInitialIntensity3 * intensityMultiplieroff;

    Azul1.intensity = AzulInitialIntensity1 * intensityMultiplieroff;
    Azul2.intensity = AzulInitialIntensity2 * intensityMultiplieroff;
    Azul3.intensity = AzulInitialIntensity3 * intensityMultiplieroff;

    Amarillo1.intensity = AmarilloInitialIntensity1 * intensityMultiplieroff;
    Amarillo2.intensity = AmarilloInitialIntensity2 * intensityMultiplieroff;
    Amarillo3.intensity = AmarilloInitialIntensity3 * intensityMultiplieroff;
    Amarillo4.intensity = AmarilloInitialIntensity4 * intensityMultiplieroff;
    Amarillo5.intensity = AmarilloInitialIntensity5 * intensityMultiplieroff;
    Amarillo6.intensity = AmarilloInitialIntensity6 * intensityMultiplieroff;
    Amarillo7.intensity = AmarilloInitialIntensity7 * intensityMultiplieroff;
    Amarillo8.intensity = AmarilloInitialIntensity8 * intensityMultiplieroff;
}

// Update is called once per frame
void Update()
{
    if (GlobalVar.Escena == 7f)
    {
    transform.position = new Vector3(initialposition.x + (Mathf.Sin(Time.time * speed) * amount),
    initialposition.y   +   (Mathf.Sin(Time.time   *   speed)   *   amount),   initialposition.z   +
    (Mathf.Sin(Time.time * speed) * amount));
    }
}

private void Awake()
{
    leftHand = GameObject.Find("LeftHandAnchor");
    rightHand = GameObject.Find("RightHandAnchor");

    pointInitialIntensity1 = point1.intensity;
    pointInitialIntensity2 = point2.intensity;
    pointInitialIntensity3 = point3.intensity;
    pointInitialIntensity4 = point4.intensity;
    pointInitialIntensity5 = point5.intensity;
    pointInitialIntensity6 = point6.intensity;
    pointInitialIntensity7 = point7.intensity;
    pointInitialIntensity8 = point8.intensity;
    pointInitialIntensity9 = point9.intensity;
    pointInitialIntensity10 = point10.intensity;
```

```
        pointInitialIntensity11 = point11.intensity;
        pointInitialIntensity12 = point12.intensity;
        pointInitialIntensity13 = point13.intensity;
        pointInitialIntensity14 = point14.intensity;

        AzulInitialIntensity1 = Azul1.intensity;
        AzulInitialIntensity2 = Azul2.intensity;
        AzulInitialIntensity3 = Azul3.intensity;

        RojoInitialIntensity1 = Rojo1.intensity;
        RojoInitialIntensity2 = Rojo2.intensity;
        RojoInitialIntensity3 = Rojo3.intensity;

        AmarilloInitialIntensity1 = Amarillo1.intensity;
        AmarilloInitialIntensity2 = Amarillo2.intensity;
        AmarilloInitialIntensity3 = Amarillo3.intensity;
        AmarilloInitialIntensity4 = Amarillo4.intensity;
        AmarilloInitialIntensity5 = Amarillo5.intensity;
        AmarilloInitialIntensity6 = Amarillo6.intensity;
        AmarilloInitialIntensity7 = Amarillo7.intensity;
        AmarilloInitialIntensity8 = Amarillo8.intensity;

        initialposition.x = transform.position.x;
        initialposition.y = transform.position.y;
        initialposition.z = transform.position.z;
    }

    private void OnTriggerEnter(Collider other)
    {
        if ((other.gameObject == leftHand || rightHand) && GlobalVar.Escena == 7f)
        {
            if (GlobalVar.VarMid == 1f)
            {
                intensityMultiplier = 2f;
            }
            else if (GlobalVar.VarMorning == 1f)
            {
                intensityMultiplier = 0.7f;
            }
            else if (GlobalVar.VarDay == 1f)
            {
                intensityMultiplier = 0f;
            }
            else
            {
                intensityMultiplier = 1f;
            }

            if (GlobalVar.VarColor != 1f)
            {
                point1.intensity = pointInitialIntensity1 * intensityMultiplieroff;
                point2.intensity = pointInitialIntensity2 * intensityMultiplieroff;
                point3.intensity = pointInitialIntensity3 * intensityMultiplieroff;
                point4.intensity = pointInitialIntensity4 * intensityMultiplieroff;
                point5.intensity = pointInitialIntensity5 * intensityMultiplieroff;
                point6.intensity = pointInitialIntensity6 * intensityMultiplieroff;
                point7.intensity = pointInitialIntensity7 * intensityMultiplieroff;
                point8.intensity = pointInitialIntensity8 * intensityMultiplieroff;
                point9.intensity = pointInitialIntensity9 * intensityMultiplieroff;
                point10.intensity = pointInitialIntensity10 * intensityMultiplieroff;
                point11.intensity = pointInitialIntensity11 * intensityMultiplieroff;
                point12.intensity = pointInitialIntensity12 * intensityMultiplieroff;
                point13.intensity = pointInitialIntensity13 * intensityMultiplieroff;
                point14.intensity = pointInitialIntensity14 * intensityMultiplieroff;

                Rojo1.intensity = RojoInitialIntensity1 * intensityMultiplier;
                Rojo2.intensity = RojoInitialIntensity2 * intensityMultiplier;
                Rojo3.intensity = RojoInitialIntensity3 * intensityMultiplier;

                Azul1.intensity = AzulInitialIntensity1 * intensityMultiplieroff;
                Azul2.intensity = AzulInitialIntensity2 * intensityMultiplieroff;
                Azul3.intensity = AzulInitialIntensity3 * intensityMultiplieroff;

                Amarillo1.intensity = AmarilloInitialIntensity1 * intensityMultiplieroff;
```

```
        Amarillo2.intensity = AmarilloInitialIntensity2 * intensityMultiplieroff;
        Amarillo3.intensity = AmarilloInitialIntensity3 * intensityMultiplieroff;
        Amarillo4.intensity = AmarilloInitialIntensity4 * intensityMultiplieroff;
        Amarillo5.intensity = AmarilloInitialIntensity5 * intensityMultiplieroff;
        Amarillo6.intensity = AmarilloInitialIntensity6 * intensityMultiplieroff;
        Amarillo7.intensity = AmarilloInitialIntensity7 * intensityMultiplieroff;
        Amarillo8.intensity = AmarilloInitialIntensity8 * intensityMultiplieroff;

        GlobalVar.VarColor = 1f;
        GlobalVar.Bandera = 1f;
    }
    else
    {
        if (GlobalVar.Bandera == 1f)
        {
            point1.intensity = pointInitialIntensity1 * intensityMultiplier;
            point2.intensity = pointInitialIntensity2 * intensityMultiplier;
            point3.intensity = pointInitialIntensity3 * intensityMultiplier;
            point4.intensity = pointInitialIntensity4 * intensityMultiplieroff;
            point5.intensity = pointInitialIntensity5 * intensityMultiplieroff;
            point6.intensity = pointInitialIntensity6 * intensityMultiplieroff;
            point7.intensity = pointInitialIntensity7 * intensityMultiplieroff;
            point8.intensity = pointInitialIntensity8 * intensityMultiplieroff;
            point9.intensity = pointInitialIntensity9 * intensityMultiplieroff;
            point10.intensity = pointInitialIntensity10 * intensityMultiplieroff;
            point11.intensity = pointInitialIntensity11 * intensityMultiplieroff;
            point12.intensity = pointInitialIntensity12 * intensityMultiplieroff;
            point13.intensity = pointInitialIntensity13 * intensityMultiplieroff;
            point14.intensity = pointInitialIntensity14 * intensityMultiplieroff;

            Rojo1.intensity = RojoInitialIntensity1 * intensityMultiplieroff;
            Rojo2.intensity = RojoInitialIntensity2 * intensityMultiplieroff;
            Rojo3.intensity = RojoInitialIntensity3 * intensityMultiplieroff;

            GlobalVar.Bandera = 0f;
        }
        else
        {
            point1.intensity = pointInitialIntensity1 * intensityMultiplieroff;
            point2.intensity = pointInitialIntensity2 * intensityMultiplieroff;
            point3.intensity = pointInitialIntensity3 * intensityMultiplieroff;
            point4.intensity = pointInitialIntensity4 * intensityMultiplieroff;
            point5.intensity = pointInitialIntensity5 * intensityMultiplieroff;
            point6.intensity = pointInitialIntensity6 * intensityMultiplieroff;
            point7.intensity = pointInitialIntensity7 * intensityMultiplieroff;
            point8.intensity = pointInitialIntensity8 * intensityMultiplieroff;
            point9.intensity = pointInitialIntensity9 * intensityMultiplieroff;
            point10.intensity = pointInitialIntensity10 * intensityMultiplieroff;
            point11.intensity = pointInitialIntensity11 * intensityMultiplieroff;
            point12.intensity = pointInitialIntensity12 * intensityMultiplieroff;
            point13.intensity = pointInitialIntensity13 * intensityMultiplieroff;
            point14.intensity = pointInitialIntensity14 * intensityMultiplieroff;

            Rojo1.intensity = RojoInitialIntensity1 * intensityMultiplier;
            Rojo2.intensity = RojoInitialIntensity2 * intensityMultiplier;
            Rojo3.intensity = RojoInitialIntensity3 * intensityMultiplier;

            GlobalVar.Bandera = 1f;
        }

        Azul1.intensity = AzulInitialIntensity1 * intensityMultiplieroff;
        Azul2.intensity = AzulInitialIntensity2 * intensityMultiplieroff;
        Azul3.intensity = AzulInitialIntensity3 * intensityMultiplieroff;

        Amarillo1.intensity = AmarilloInitialIntensity1 * intensityMultiplieroff;
        Amarillo2.intensity = AmarilloInitialIntensity2 * intensityMultiplieroff;
        Amarillo3.intensity = AmarilloInitialIntensity3 * intensityMultiplieroff;
        Amarillo4.intensity = AmarilloInitialIntensity4 * intensityMultiplieroff;
        Amarillo5.intensity = AmarilloInitialIntensity5 * intensityMultiplieroff;
        Amarillo6.intensity = AmarilloInitialIntensity6 * intensityMultiplieroff;
        Amarillo7.intensity = AmarilloInitialIntensity7 * intensityMultiplieroff;
        Amarillo8.intensity = AmarilloInitialIntensity8 * intensityMultiplieroff;

        GlobalVar.VarColor = 1f;
```

```
            }
        }
    }
}
```

4.2.3.11. SoundExplosion

Código utilizado para asignar sonido a las explosiones de los fuegos artificiales. Así, en primera instancia se evalúa cuándo se crea un sistema de partículas nuevo y activa el audio de las estelas. Posteriormente, en cada frame, se comprueba si ese sistema tiene 0 partículas en ese instante, indicando el fin del determinado fuego. Una vez se alcanza ese valor, se activa otro audio que corresponde con la explosión. De esta manera se ha programado el sonido completo de los fuegos artificiales.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class SoundExplosion : MonoBehaviour
{
    // Start is called before the first frame update
    private ParticleSystem _parentParticleSystem;

    private int _currentNumberOfParticles = 0;

    public AudioClip BornSound;
    public AudioClip DieSound;
    void Start()
    {
        _parentParticleSystem = this.GetComponent<ParticleSystem>();
    }

    // Update is called once per frame
    void Update()
    {
        var amount = Mathf.Abs(_currentNumberOfParticles - _parentParticleSystem.particleCount);
        if (_parentParticleSystem.particleCount < _currentNumberOfParticles)
        {
            //Die sound
            AudioSource audio = GetComponent<AudioSource>();
            audio.clip = DieSound;
            audio.Play();
        }

        if (_parentParticleSystem.particleCount > _currentNumberOfParticles)
        {
            //Born sound
            AudioSource audio = GetComponent<AudioSource>();
            audio.clip = BornSound;
            audio.Play();
        }

        _currentNumberOfParticles = _parentParticleSystem.particleCount;
    }
}
```

4.2.3.12. Traslacion_cuadros

Este fichero, junto al correspondiente de la pintura en el aire, ha sido uno de los más complejos de realizar. No tanto por la programación como por la ocurrencia. Este el encargado de hacer que las pinturas de muestra acompañen al usuario en traslación, pero no en rotación. Para ello se pensó en calcular la distancia que el usuario recorre en cada frame en los tres ejes x, y y z. Esos tres valores se guardan en tres variables distintas que posteriormente se suma a la posición local del conjunto de cuadros. De esta manera, la distancia recorrida por los cuadros es la misma que recorre en cada frame el usuario, consiguiendo que el conjunto acompañe a la persona en todo momento.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class traslacion_cuadros : MonoBehaviour
{
    private GameObject persona;
    private float offsetx, offsety, offsetz;
    Vector3 positionnow;
    Vector3 initialposition;

    // Start is called before the first frame update
    void Start()
    {
    }

    private void Awake()
    {
        persona = GameObject.Find("Seguimiento");
        initialposition = persona.transform.position;
    }

    // Update is called once per frame
    void Update()
    {
        positionnow = persona.transform.position;

        offsetx = positionnow.x - initialposition.x;
        offsety = positionnow.y - initialposition.y;
        offsetz = positionnow.z - initialposition.z;

        this.transform.position = new Vector3(transform.position.x + offsetx, transform.position.y +
        offsety, transform.position.z + offsetz);
        initialposition = positionnow;
    }
}
```

4.2.3.13. Vuelo

Programa creado para controlar el movimiento que realizar el avión. Este script se ha asignado a un elemento vacío, colocado en la parte delantera del objeto y actuando como padre del avión. Se han establecido seis objetos vacíos en el espacio de la habitación, de forma que, con este programa, se va buscando de forma ordenada las posiciones de esos objetos en cada frame. Una vez alcanzado el primero, el avión automáticamente busca el segundo, y así sucesivamente.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Vuelo : MonoBehaviour
{
    private Vector3 pos;
    private float flag;

    private Vector3 pos1, pos2, pos3, pos4, pos5, pos6;

    private Quaternion rotacion_inicial;
    private Quaternion rotacionsig;
    private float offsetx, offsety, offsetz;

    // Start is called before the first frame update
    void Start()
    {
        this.transform.localRotation = rotacion_inicial;
        getNewPosition();
    }


    private void Awake()
    {
        flag = 1f;

        pos1 = GameObject.Find("Direccion1").transform.position;
        pos2 = GameObject.Find("Direccion2").transform.position;
        pos3 = GameObject.Find("Direccion3").transform.position;
        pos4 = GameObject.Find("Direccion4").transform.position;
        pos5 = GameObject.Find("Direccion5").transform.position;
        pos6 = GameObject.Find("Direccion6").transform.position;

        rotacion_inicial.x = 90;
        rotacion_inicial.y = 90;
        rotacion_inicial.z = -90;
    }


    // Update is called once per frame
    void Update()
    {
        this.transform.position = Vector3.MoveTowards(transform.position, pos, .01f);       //Direccion
        por cuadro
        this.transform.LookAt(pos);
        if (transform.position == pos1)
        {
            flag = 2f;
        }
        if (transform.position == pos2)
        {
            flag = 3f;
        }
        if (transform.position == pos3)
        {
            flag = 4f;
        }
        if (transform.position == pos4)
        {
            flag = 5f;
        }
        if (transform.position == pos5)
```

```
        {
            flag = 6f;
        }
        if (transform.position == pos6)
        {
            flag = 1f;
        }
        getNewPosition();

    }

    void getNewPosition()
    {
        if (flag == 1f)
        {
            pos = GameObject.Find("Direccion1").transform.position;
            transform.LookAt(pos1);
        }
        else if (flag == 2f)
        {
            pos = GameObject.Find("Direccion2").transform.position;
            transform.LookAt(pos2);
        }
        else if (flag == 3f)
        {
            pos = GameObject.Find("Direccion3").transform.position;
            transform.LookAt(pos3);
        }
        else if (flag == 4f)
        {
            pos = GameObject.Find("Direccion4").transform.position;
            transform.LookAt(pos4);
        }
        else if (flag == 5f)
        {
            pos = GameObject.Find("Direccion5").transform.position;
            transform.LookAt(pos5);
        }
        else if (flag == 6f)
        {
            pos = GameObject.Find("Direccion6").transform.position;
            transform.LookAt(pos6);
        }

    }
}
```

5 PRUEBAS DE CALIDAD

Tras realizar todo el desarrollo, era necesario que personas externas a la creación del proyecto probaran la experiencia completa. Para ello, se pidió a algunos conocidos, en concreto cinco, que experimentaran la aplicación y respondieran un cuestionario con el fin de conocer qué mejorar y qué tal había sido la experiencia de usuario.

Dado que este trabajo explora el mundo del arte a través de la realidad virtual, se necesita usar un casco que permita interactuar con esta tecnología. Como se ha explicado al principio, estos cascos tienen contacto estrecho con los ojos y la nariz. Teniendo esto en cuenta, es necesario comentar que sólo han podido probar la experiencia con un número menor de personas del que en un primer momento se había pensado, ya que la situación sanitaria producida por el virus SARS-CoV-2 y la pandemia ocurrida hacían más difícil tener una gran cantidad de resultados que poder evaluar. Pese a ello, estas personas han tenido todas las protecciones pertinentes y se ha primado la seguridad y salud del usuario antes que la evaluación de la propia experiencia.

Para conocer la sensación general de las personas que han realizado esta experiencia, se ha propuesto un pequeño cuestionario de once sencillas preguntas. Con ellas se ha querido conocer la opinión acerca de la aplicación desarrollada, pero también tener una noción de las edades, experiencias previas, sensación de mareo o facilidad de uso del usuario. Así, se han podido corregir pequeños defectos detectados y estos resultados se pueden estudiar para posibles futuras actualizaciones que se pueda desarrollar en la experiencia. A continuación, se mostrarán las preguntas del cuestionario con las respuestas en forma de gráfico, de manera que sea más de interpretar.

- Pregunta 1: ¿A qué rango de edad perteneces?

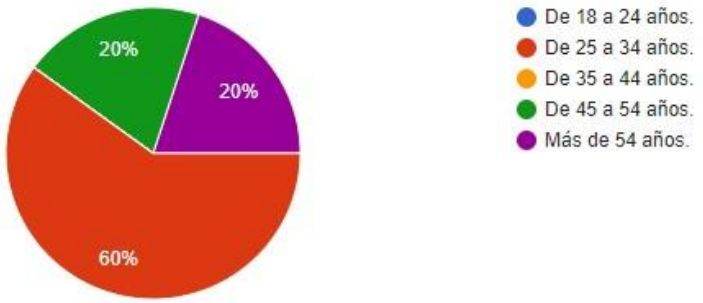

Figura 120: Respuestas pregunta 1

Como se puede ver en la Figura 120, el rango de edades de las personas que han participado es muy amplio. Es necesario comentar que, debido al número reducido de pruebas, en comparación con las que se había pensado en un primer momento, el rango de edades comprendidos entre los 18-24 y 35-44 no se muestra en la gráfica. Sin embargo, fácilmente se podrían haber encontrado usuarios con estas edades y haber probado la experiencia.

Pregunta 2: ¿Tienes conocimientos previos de realidad vritual?

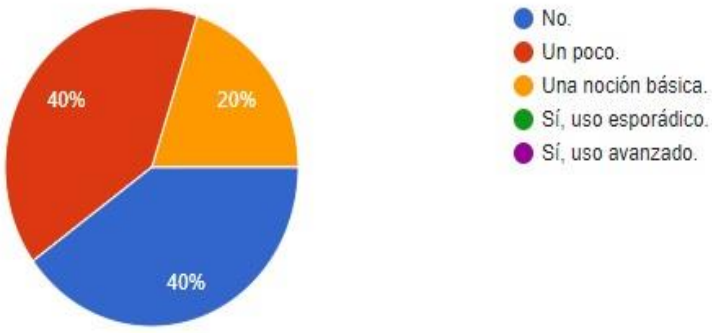

Figura 121: Respuestas pregunta 2

En el caso de la Figura 121, casi la mayoría de las personas tenían algunos conocimientos previos de la realidad virtual, pero ninguno de los usuarios tenía acceso continuo a esta tecnología ni había probado aplicaciones previas a esta experiencia.

- Pregunta 3: ¿Te ha gustado la experiencia?

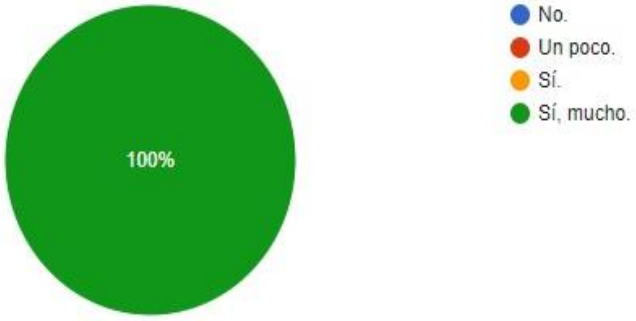

Figura 122: Respuestas pregunta 3

En la totalidad de las respuestas la experiencia ha resultado satisfactoria, como se puede comprobar con el porcentaje de la Figura 122.

- Pregunta 4: ¿Te has mareado usándola?

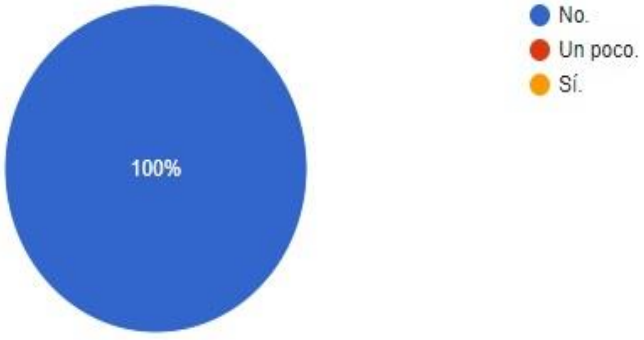

Figura 123: Respuestas pregunta 4

Una manera de entender el correcto uso de los mandos y el entorno creado se puede comprobar en la Figura 123, ya que la relación entre lo que ve el sujeto, la interacción de los mandos y la combinación de elementos, si todos ellos están mal configurados, aumentan la sensación de mareo. A la vista de los resultados, dónde el 100% de las personas que han probado la experiencia, no han sufrido mareos, se puede afirmar que el diseño se ha realizado correctamente.

- Pregunta 5: ¿Qué tal el control de los mandos?

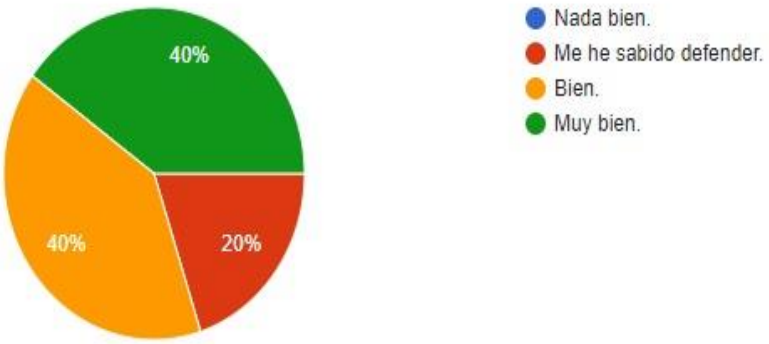

Figura 124: Respuestas pregunta 5

Como se ve en la Figura 124, se ha tenido diversidad de opiniones respecto a la facilidad de uso de los mandos. El motivo de ello es principalmente la falta de costumbre de los usuarios en estos entornos.

- Pregunta 6: ¿Pagarías, aunque fuera muy barato, por esta aplicación?

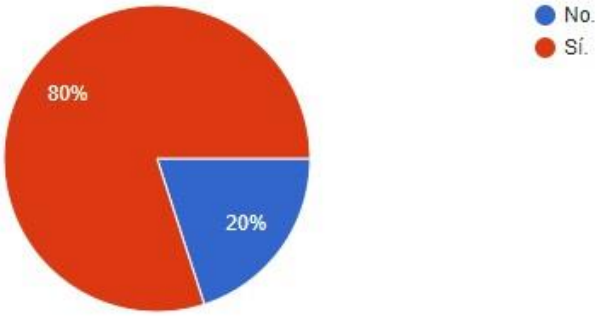

Figura 125: Respuestas pregunta 6

El éxito de la aplicación queda constatado en la Figura 125, dónde la mayoría de las personas pagarían un precio económico por esta experiencia.

- Pregunta 7: ¿Te ha parecido fácil?

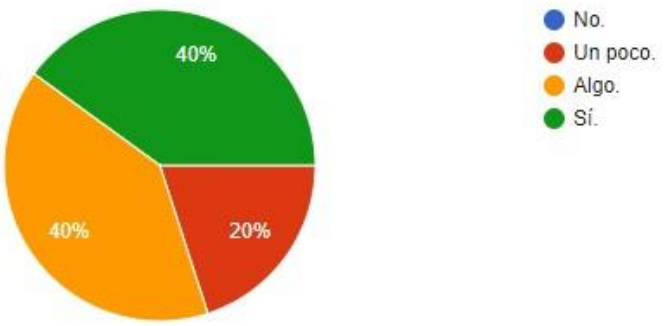

Figura 126: Respuestas pregunta 7

Como se ve en la Figura 126, está muy ligada con las respuestas de la pregunta 5, ya que el uso de los mandos y su respectivo conocimiento son de gran importancia en la aplicación.

- Pregunta 8: Dentro de la experiencia, ¿te ha parecido un entorno espacioso?

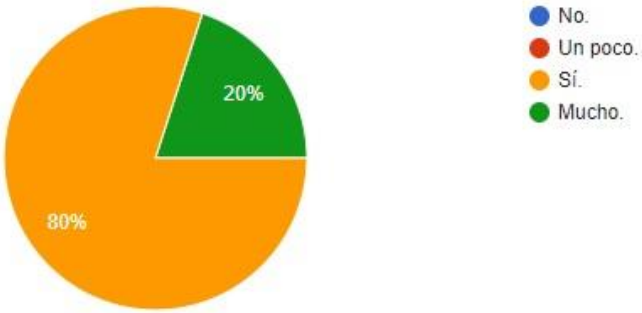

Figura 127: Respuestas pregunta 8

El fin principal, además de conseguir una experiencia inmersiva e interactiva dónde el usuario se sintiera atraído por el arte, era conseguir que la persona también se sintiera libre y cómoda en el entorno, evitando situaciones claustrofóbicas. En las respuestas de la Figura 127 se puede comprobar que el trabajo realizado ha sido correcto.

- Pregunta 9: ¿Gracias a esta experiencia ha aumentado tu interés en el arte?

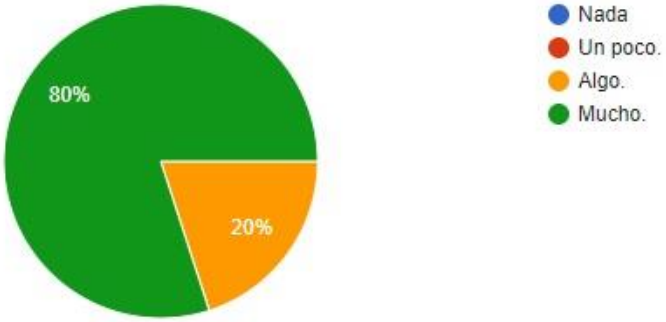

Figura 128: Respuestas pregunta 9

A la vista de las respuestas mostradas en la Figura 128, el interés general en el arte ha aumentado en todos los usuarios. Esto muestra la predisposición general del público hacia este tipo de experiencias.

- Pregunta 10: ¿Gracias a esta experiencia ha aumentado tu interés en el pintor?

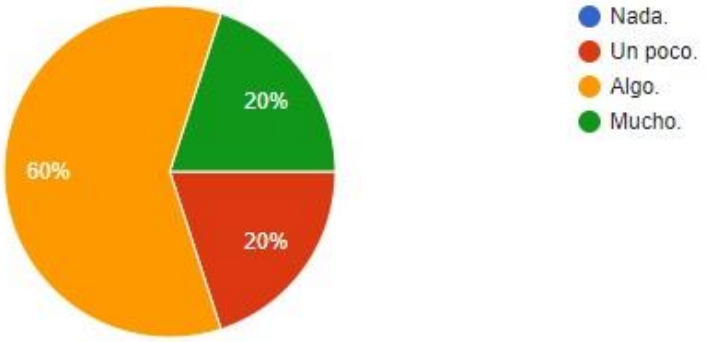

Figura 129: Respuestas pregunta 10

Como se puede ver en la Figura 129, pese a aumentar el interés de los usuarios en la figura del pintor, no llega a los mismos números respecto al interés acerca del arte en general. Estos datos podrían servir en futuras actualizaciones, con el fin de conseguir una mayor atracción en el público.

- Pregunta 11: ¿Cómo valoras esta experiencia en comparación con la visión por ordenador tradicional?

Figura 130: Respuestas pregunta 11

El gran interés que suscita esta tecnología se puede ver en los resultados de la Figura 130. Como se ha podido comprobar, la mayoría de las respuestas del cuestionario han sido positivas y es por ello por lo que el resultado final de esta experiencia en realidad virtual es satisfactorio. En el siguiente punto se procede a mostrar las conclusiones respecto al desarrollo de la aplicación, mostrando una perspectiva general del proyecto y todo lo que se ha conseguido.

6 CONCLUSIONES

Como se ha estado comentando desde un principio, la idea era crear una aplicación de realidad virtual, tanto interactiva como inmersiva, teniendo siempre presente el arte. Con la mente puesta en este objetivo, se ha conseguido desarrollar lo que se quería mostrar.

Dado que la intención era atraer al público al aprendizaje del arte y la cultural, en vista a los porcentajes obtenidos en las distintas pruebas, se puede afirmar que los resultados han sido satisfactorios. Se ha conseguido una experiencia muy completa, donde todo aquel que lo ha probado, se ha encontrado con un entorno interesante y que realmente se ha sentido atraído a lo que se mostraba.

La aplicación ha sufrido distintos cambios a medida que se iban incluyendo las distintas escenas y se iban haciendo las pruebas correspondientes. Así, se han añadido elementos como el avión, otorgando dinamismo a la escena; las pantallas de vídeo; la importancia de la música y la inclusión de una escala con los cubos del mismo color; cambios de luces; añadir fuegos artificiales o permitir pintar en el aire. Todo esto ha conseguido que en el usuario emerjan las sensaciones buscadas, entendiendo que el resultado de la aplicación creada es efectivo.

Es necesario comentar algunos errores encontrados, como son las manos en ciertos instantes o las luces de colores. Estos fallos, producidos por el propio programa, sería interesante tenerlos en cuenta en futuras actualizaciones y buscar las posibles soluciones.

Como se ha comentado, a pesar de pequeños fallos encontrados, la mayoría de la experiencia ha sido correctísima. El hecho más interesante viene de que personas que han probado la experiencia y, aún a sabiendas de que no les gustan los videojuegos, han sentido atracción por este mundo tras haber “jugado” con la experiencia. Profesores de colegios, ingenieros o estudiantes relacionados con bellas artes o historia del arte han probado esta aplicación, donde se puede ver la variedad de profesionales que la han experimentado, y todos ellos han acogido de la mejor manera las posibilidades que se les ofrecía. En todos estos casos, se han sentido cómodos, ya sea en el entorno con lo que veían o sin sufrir mareos, así como en las aplicaciones didácticas de aprendizaje artístico que ofrece.

7 LÍNEAS FUTURAS

Tras haber finalizado este proyecto, hay algunos comentarios que se deben hacer al respecto.

Durante esta sección, se explican los posibles trabajos futuros que puede introducir esta aplicación. Los principales campos que pueden encontrar interesante esta experiencia son la educación, bellas artes o preservación cultural.

En primer lugar, es importante remarcar la existencia de distintos caminos por dónde mejorar este trabajo. El primero de ellos, en lo referente a qué fue programado. Para próximas actualizaciones, se pueden desarrollar diferentes elementos. Y es que, aunque existe una audioguía que explica la experiencia completa, la misma se encuentra tan solo en español, por lo que añadir otros idiomas podría introducir la experiencia en otros países, aumentando de esta manera la atracción por el arte y el interés de la realidad virtual en el mencionado campo. Partiendo de la base de estos audios, otra propuesta sería incluir pequeños juegos o interacciones con los sonidos, haciendo que el usuario se encuentre con una experiencia aún más completa.

Es necesario comentar el hecho de que algunos elementos necesitan mejoras, como pueden ser los cubos verdes que se encuentran muy cercanos a otros, haciendo que el usuario en algunas ocasiones se pueda llegar a saltar alguna escena. Esto puede solucionarse cambiando determinados cubos por otros, haciendo que el orden de actuación varíe. Tras las diversas pruebas realizadas, se ha detectado que, aunque las explicaciones son claras y añadir más audio distraería al usuario de la sensación de inmersión, es necesario que se acompañe de instrucciones en todo momento, ya que en ciertas ocasiones algunos elementos son ignorados, salvo que el conocedor de la aplicación mencione algo al respecto. Todo ello, con el fin de conseguir un uso correcto. Una posible solución, podría ser añadir placas explicativas, similares a los que se encuentran en los museos, mostrando las instrucciones necesarias para conocer completamente el entorno. Sería interesante también corregir los pequeños fallos correspondientes al programa que en determinados casos se presentan, como son los referidos a las luces de colores en su respectiva escena, la interacción en ciertos puntos de las manos con el entorno o el efecto de vibración que se produce en ciertos momentos, debido mayoritariamente a la capacidad de ejecución del hardware. Aunque, como se ha comentado, estos fallos dependen más de los propios dispositivos que de la aplicación realizada.

Dado que el fin principal de este proyecto es atraer la atención tanto de artistas como público, gran cantidad de ideas o elementos pueden ser incluidos. Con las nuevas formas de pintar que la aplicación ofrece, los artistas pueden ser capaces de llevar estas nuevas formas de arte hasta el límite. Además, con las explicaciones incluidas, el público común es capaz de entender mejor las explicaciones artísticas que se muestran. Siguiendo este propósito, esta aplicación puede formar parte de un “scape room” más grande relacionado con el arte, el cual podría ser un punto para tener muy en cuenta.

Por otro lado, esta experiencia puede formar parte de un proyecto mucho mayor. Tomando la misma idea que se ha realizado, en la que se muestra el entorno inmersivo de una pintura y se dé la posibilidad de interactuar con ella, es extensible a otras pinturas u obras de arte, siendo del mismo autor o explorando nuevos artistas. El número de obras que se pueden hacer de forma tridimensional es infinito. Todo ello se puede almacenar en un gran museo virtual, el cual podría tener sus propios tours virtuales separando por autorías, estilos, fechas o intenciones artísticas. El público sería capaz

de ver obras reales junto a aquellas que tan sólo existen de forma virtual. Este museo puede ser formado siguiendo el orden cronológico de sus obras y dar una muestra cercana de la historia del arte.

Por supuesto, todo esto debería ser realizado de forma eficiente, la cual puede ser otra línea de investigación y otorgaría de grandes mejoras a los proyectos futuros. Todas estas ideas necesitan ser desarrolladas de forma correcta y aprovechando de la mejor manera los recursos disponibles. Existen muchas y variadas propuestas relacionadas con esta área y muchos desarrollos aún por hacer. Aún queda mucho camino por mostrar e investigar en conjunto en lo referente al arte, la cultura y las realidades extendidas.

REFERENCIAS

[1] Hernández Hernández, Francisca. Evolución del concepto de museo. Revista. [Internet]. Madrid, España. Facultad de Prehistoria. Universidad Complutense. Vol. 2. Nº 1. 1992. Disponible en: http://esferapublica.org/museo.pdf

[2] Pintor Alonso, María del Pilar. Gestión y Conservación de los fondos museísticos. Revista de estudios campogibraltareños [Internet]. Algeciras, España. Almoraima 39. 2009. Disponible en: http://institutoecg.es/wp-content/uploads/2019/01/38_PPINTOR.pdf

[3] Silva Soto, Natalia. Embalaje interno para obras de arte. Unidades de almacenamiento. [Proyecto de grado]. [Internet]. Bogotá, Colombia. Facultad de arquitectura. Departamento de diseño industrial. 2009. Disponible en: https://repositorio.uniandes.edu.co/bitstream/handle/1992/14329/u402097.pdf?sequence=1

[4] Espacio Visual Europa (EVE). Breve Historia de los Museos. [Internet]. España. 2020. Disponible en: https://evemuseografia.com/2015/11/30/breve-historia-de-los-museos/

[5] Huguet, Guiomar. Así serían los jardines colgantes de Babilonia en la actualidad. [Internet]. España. National Geographic. 2019. Disponible en: https://historia.nationalgeographic.com.es/a/asi-serian-jardines-colgantes-babilonia-actualidad_14213/2

[6] Fernández Abad, Francisco Javier. El Serapeo o Serapeum: Templo, Biblioteca y Centro de Investigaciones Científicas. Universidad Complutense de Madrid. España. 2008.

[7] Serantes, Arantxa. Los sabios y los reyes: la biblioteca de Alejandría. Simposio de estudios humanísticos. [Internet]. A Coruña, España. 2006. Disponible en: https://ruc.udc.es/dspace/bitstream/handle/2183/12769/CC-88_art_7.pdf

[8] Villarello Reza, Rosamaría. Reseña de "El incendio de Alejandría" de Luminet, Jean-Pierre. [Internet]. México. Biblioteca Universitaria. Vol. 13. Nº 2. Pp 270-277. 2010. Disponible en: https://www.redalyc.org/pdf/285/28520822014.pdf

[9] Hernández, Sachie. La evolución de los museos y su adaptación. Revista. [Internet]. Cuba. Unesco. Museo nacional de Bellas Artes. 2012. Disponible en: http://www.lacult.unesco.org/docc/evolucion_museos.pdf

[10] El Rebobinador. De mausoleo a espacio de investigación: la evolución histórica del concepto de museo. [Internet]. España. MASDEARTE.COM. Disponible en: https://masdearte.com/especiales/de-mausoleo-espacio-de-investigacion-la-evolucion-historica-del-concepto-de-museo/

[11] Residencia de Múnich. Wikipedia. [Internet]. Disponible en: https://es.wikipedia.org/wiki/Residencia_de_M%C3%BAnich

[12] Navarro, Fernando; Martínez, Antonio; Martínez, José M. Realidad Virtual y Realidad Aumentada. Desarrollo de aplicaciones. [Libro]. [Internet]. España. Ra-Ma. 2018. Disponible en: https://elibro--net.us.debiblio.com/es/ereader/bibliotecaus/106518

[13] Luque Ordóñez, Javier. Realidad Virtual y Realidad Aumentada. [Revista digital]. [Internet]. Madrid, España. Acta. 2020. Disponible en: https://www.acta.es/medios/articulos/ciencias_y_tecnologia/063001.pdf

[14] Marqués Gómez, José Luis. Aplicación móvil para entrenamiento cognitivo con gafas de realidad aumentada y neurofeedback. [TFG]. [Internet]. Valladolid, España. Escuela técnica superior de ingenieros de telecomunicaciones. Universidad de Valladolid. 2019. Disponible en: https://core.ac.uk/download/pdf/250406429.pdf

[15] Gadget Lab. Así funciona la realidad virtual. [Internet]. España. Revista Gadget. Disponible en: http://www.revista-gadget.es/reportaje/asi-funciona-la-realidad-virtual/

[16] Rodríguez de Luis, Eva. Guía de compra de gafas de realidad virtual: 11 modelos para todas las expectativas, necesidades y presupuestos. [Internet]. España. Xataka. 2021. Disponible en: https://www.xataka.com/seleccion/guia-compra-gafas-realidad-virtual-16-modelos-para-todas-expectativas-necesidades-presupuestos

[17] ¿Cómo funcionan las gafas de realidad virtual? [Internet]. España. Tokio School. 2020. Disponible en: https://www.tokioschool.com/noticias/como-funcionan-gafas-realidad-virtual/

[18] Gadget Lab. Así funciona la realidad virtual. [Internet]. España. Revista Gadget. Disponible en: http://www.revista-gadget.es/reportaje/asi-funciona-la-realidad-virtual/

[19] Prado Álvarez, Danaisy. Creación de Interfaces de Realidad Virtual Móvil en Unity 3D y Validación Experimental de sus Requisitos de Red. [Internet]. Valencia, España. Universidad Politécnica de Valencia. 2016. Disponible en: https://riunet.upv.es/bitstream/handle/10251/112997/Danaisy%20Prado%20Alvarez%20TFM.pdf?sequence=1

[20] Del Río Martín, Beatriz. Los docentes de español como lengua extranjera y su relación con la realidad virtual. [TFM]. [Internet]. Jaén, España. Universidad de Jaén. Departamento de Didáctica de la Lengua y Literatura. 2018. Disponible en: http://tauja.ujaen.es/bitstream/10953.1/8751/1/delRoMartnBeatriz_TFM_1718.pdf

[21] Sacristán, Alejandro. Realidad virtual + Internet 3D. [Internet]. España. Artfutura. Disponible en: https://www.artfutura.org/v3/realidad-virtual-internet-3d/

[22] Casco de realidad virtual. Wikipedia. [Internet]. Disponible en: https://es.wikipedia.org/wiki/Casco_de_realidad_virtual#Tipos

[23] Ojanguren Álvarez, Marina. Realidad virtual en la ingeniería civil. Virtualización de una obra. [TFM]. [Internet]. Cantabria, España. Universidad de Cantabria. Escuela Técnica Superior de Ingenieros de Caminos, Canales y Puertos. 2016. Disponible en: https://repositorio.unican.es/xmlui/bitstream/handle/10902/10018/Ojanguren.pdf?sequence=1&isAllowed=y

[24] Fernández, Yúbal. 14 sensores que encontrarás en tu móvil: cómo funcionan y para qué sirven. [Internet]. España. Xataka. 2019. Disponible en: https://www.xataka.com/basics/sensores-que-encontraras-tu-movil-como-funcionan-sirven#:~:text=Aceler%C3%B3metro,-El%20aceler%C3%B3metro%20es&text=El%20aceler%C3%B3metro%20del%20m%C3%B3vil%20consta,lo%20hace%20y%20su%20orientaci%C3%B3n

[25] Gglassday. ASÍ funciona el TRACKING en OCULUS QUEST. [Internet]. España. Youtube. 2019. Disponible en: https://www.youtube.com/watch?v=9yMgtk7indo

[26] Bendala José, Jorge. Desarrollo de un entorno de realidad virtual para la inmersión en experimentos de detección de emociones. [TFG]. [Internet]. Sevilla, España. Universidad de Sevilla. Escuela Técnica Superior de Ingeniería. Departamento de Teoría de la Señal y Comunicaciones. 2020. Disponible en: https://biblus.us.es/bibing/proyectos/abreproy/93020/fichero/TFG-3020+BENDALA+JOS%C3%89%2C+JORGE.pdf

[27] Kose, Ahmet; Tepljakov, Aleksei; Petlenkov, Eduard. Towards Assisting Interactive reality. Interactive Reality for Education, Data Analysis and Industry. [Artículo]. Estonia. Tallinn University of Technology. Department of Computer Systems. AVR 2018. Vol. 2. Pág. 569. 2018.

[28] Martínez Navarro, Gema. Tecnologías y nuevas tendencias en educación: aprender jugando. El caso de Kahoot. [Artículo]. Madrid, España. Universidad Complutense de Madrid. 2017.

[29] Iacono, Saverio; Zolezzi, Daniele; Vercelli, Gianni. Virtual Reality Arcade Game in Game-Based Learning for Cultural Heritage. [Artículo]. Italia. Università degli Studi di Genova. AVR 2018. Vol. 2. Pág. 383. 2018.

[30] Romero, Margarida y Turpo Gebera, Osbaldo. Serious Games para el desarrollo de las competencias del siglo XXI. [Revista]. [Internet]. Salamanca y Barcelona, España. Universidad Autónoma de Barcelona y Universidad de Salamanca. RED. Revista de Educación a Distancia. 2012. Disponible en: https://revistas.um.es/red/article/view/233511/179431

[31] Hansen, Anders; Bundgaard Larsen, Kirstine; Høgh Nielsen, Helene; Kalinov Skolov, Miroslav; Kraus, Martin. Asymmetrical Multiplayer Versus Single Player: Effects on Game Experience in a Virtual Reality Edutainment Game. [Artículo]. Dinamarca. Aalborg University. AVR 2020. Vol. 1. Pág. 22. 2020.

[32] Checa, David; Gatto, Carola; Cisterino, Doriana; Tommaso De Paolis, Lucio; Bustillo, Andrés. A Framework for Educational and Training Immersive Virtual Reality Experiences. [Artículo]. Italia y España. Universidad de Burgos y University of Salento. Departamento Ingeniería Informática y Department of Engineering for Innovation. AVR 2020. Vol. 2. Pág. 220. 2020.

[33] Bazzurri, Fabrizio; Picardello, Massimo A. Optimization Techniques for Photogrammetry Applied to Cultural Heritage and the Action of Transformation Groups. [Artículo]. Italia. University of Rome. Department of Mathematics. AVR 2018. Vol. 2. Pág. 332. 2018.

[34] Checa, David; Alaguero, Mario; Bustillo, Andrés. Industrial Heritage Seen Through the Lens of a Virtual Reality Experience. [Artículo]. Burgos, España. Universidad de Burgos. Departamento de Historia y Geografía y Departamento de Ingeniería Civil. AVR 2017. Vol. 1. Pág. 116. 2017.

[35] K. Y. Chan, Leith; Sum Geran Yuen, Kit; Y. K. Lau, Henry. Immersive Learning Environment for Visual Arts. [Artículo]. China. The University of Hong Kong y HKSKH Bishop Hall Secondary School. Department of Industrial and Manufacturing. AVR 2020. Vol. 2. Pág. 231. 2016.C

[36] Mack, Kevin. Blortasia. Virtual Reality Art by Kevin Mack. [Internet]. EE. UU. Blortasia. Disponible en: http://www.shapespacevr.com/blortasia.html

[37] Settembrini, Francesco; Angelini, Maria Giuseppa. Virtual Museum, the Exploration of Cultural Heritage with Virtual reality Techniques. [Artículo]. Italia. Politecnico di Bari. AVR 2018. Vol. 2. Pág. 392. 2018.

[38] Experiencia gamificada de la sala 39. Recreación virtual. [Internet]. España. Museo del Prado. Disponible en: https://www.museodelprado.es/recurso/experiencia-gamificada-de-la-sala-39/360332b6-adbe-fef0-480b-350c1fd8dff5?searchMeta=realidad%20virtual

[39] Visitas virtuales inmersivas. [Internet]. España. Museo Nacional Thyssen-Bornemisza. Disponible en: https://www.museothyssen.org/thyssenmultimedia/visitas-virtuales

[40] Galleries. [Internet]. Reino Unido. The British Museum. Disponible en: https://www.britishmuseum.org/collection/galleries

[41] Ramírez Martín, Abraham. MUVA. Museo virtual de arte. [Internet]. España. Binarybox Studios. Disponible en: https://www.binaryboxstudios.com/museo-virtual-de-arte/

[42] Sumérgete en los impresionistas. [Internet]. España. Art apart. Disponible en: https://impresionistasexpo.com/

[43] MORI Building DIGITAL ART MUSEUM. [Internet]. Japón. TeamLab y Borderless. Disponible en: https://borderless.teamlab.art/

[44] TeamLab Planets. [Internet]. Japón. TeamLab. Disponible en: https://planets.teamlab.art/tokyo/

[45] La Gioconda en casa gracias a la realidad virtual. [Internet]. Francia. Museo del Louvre. 2021. Disponible en: https://www.louvre.fr/es/programacion/vida-del-museo/la-gioconda-en-casa-gracias-a-la-realidad-virtual

[46] The Night Café. [Internet]. Borrowed Light Studios. 2015. Disponible en: http://www.borrowedlightvr.com/the-night-cafe/

[47] Valcheff García, Fernando. Explorando un universo pictórico: "The Night Café: a VR tribute to Vincent van Gogh". [Internet]. España. El Cuaderno. 2020. Disponible en: https://elcuadernodigital.com/2020/07/31/explorando-un-universo-pictorico-the-night-cafe-a-vr-tribute-to-vincent-van-gogh/

[48] Dreams of Dalí. [Internet]. EE. UU. Y Rusia. The Dalí. Salvador Dalí Museum y Goodby Silverstein & partners. Disponible en: http://thedali.org/dreams-of-dali-2/

[49] Disney Movies VR: Full Immersion, Virtual Reality. [Internet]. EE. UU. Disney Studios. Disponible en: https://www.disneymoviesvr.com/

[50] Pierdicca, Roberto; Frontoni, Emanuele; Zingaretti, Primo; Sturari, Mirco; Clini, Paolo; Quattrini, Ramona. Advanced Interaction with Paintings by Augmented Reality and High Resolution Visualization: A Real Case Exhibition. [Artículo]. Italia. Università Politecnica Delle

Marche. DII-Dipartimento di Ingegneria Dell' Informazione y DICEA-Dipartimento di Ingegneria Civile Edile E Dell' Architettura. AVR 2015. Pág 38. 2015.

[51] Centro Andaluz de Arte Contemporáneo. [Internet]. España. Junta de Andalucía. Disponible en: http://www.caac.es/

[52] Chionna, Francesco; Cirillo, Piero; Palmieri, Vito; Bellone, Mauro. A Proposed Hardware-Software Architecture for Virtual Reality in Industrial Applications. [Artículo]. Italia y Suecia. Consorzio CETMA y Department of Applied. Department of Applied Mechanics. AVR 2015. Pág. 287. 2015.

[53] Checa, David; Gatto, Carola; Cisterino, Doriana; Tommaso De Paolis, Lucio; Bustillo, Andrés. A Framework for Educational and Training Immersive Virtual Reality Experiences. [Artículo]. Italia y España. Universidad de Burgos y University of Salento. Departamento Ingeniería Informática y Department of Engineering for Innovation. AVR 2020. Vol. 2. Pág. 220. 2020.

[54] Carrozzino, Marcello; Colombo, Marianna; Tecchia, Franco; Evangelista, Chiara; Bergamasco, Massimo. Comparing Different Storytelling Approaches for Virtual Guides in Digital Immersive Museums. [Artículo]. Italia. Scuola Superiore Sant'Anna y Università di Pisa. Perceptual Robotics Laboratory, Institute of Communication, Information and Perception Technologies y Deparment of Filologia, Letteratura e Lingüística. AVR 2018. Vol. 2. Pág. 292. 2018.

[55] Zhang, Yanxiang; Elieisar, Clayton; Sourou Fangbemi, Abassin. Interactive 3D Symphony in VR Space. [Artículo]. China. University of Science and Technology of China. Department of Communication of Science and Technology y School of Software Engineering. AVR 2017. Vol. 1. Pág. 270. 2017.

[56] Pruna, Edwin; Tigse, Jenny; Chuquitarco, Alexandra; Escobar, Ivón; Pilatásig, Marco; Galarza, Eddie Daniel. Immersive Virtual System Based on Games for Children's Fine Motor Rehabilitation. Ecuador. Universidad de las Fuerzas Armadas ESPE. AVR 2018. Vol. 2. Pág. 30. 2018.

[57] Sisto, Maria; wenk, Nicolas; Ouerhani, Nabil; Gobron, Stéphane. A Study of Transitional Virtual Environments. [Artículo]. Suiza. Image Processing and Computer Graphics Group e Interaction Technology Group. HE-Arc. HES-SO. AVR 2017. Vol. 1. Pág. 35. 2017.

[58] Nisiotis, Louis; Alboul, Lyuba; Beer, Martin. Virtual Museums as a New Type of Cyber-Physical-Social System. [Artículo]. UK. Sheffield Hallam University. AVR 2019. Vol. 2. Pág. 256. 2019.

[59] Oculus First Contact. [Internet]. EE. UU. Oculus. 2016. Disponible en: https://www.oculus.com/experiences/rift/1217155751659625/?locale=es_ES

[60] Kaisuo. [Internet]. EE. UU. Kaisuogame. 2018. Disponible en: https://www.kaisuogame.com/

[61] Tilt Brush. [Internet]. EE. UU. Google. 2016. Disponible en: https://www.tiltbrush.com/

[62] VIVE Arts. [Internet]. EE. UU. HTC. 2011. Disponible en: https://arts.vive.com/us/

[63] The Lab. [Internet]. EE. UU. Valve. 2016. Disponible en: https://store.steampowered.com/app/450390/The_Lab/?l=spanish

[64] Fondón García I. Técnicas de Animación 3D. Modelado básico I (memoria de prácticas). Sevilla; 2020.

[65] La nueva interface de usuario en 3DS MAX 2017. [Internet]. Lynda.com. 2017. Disponible en: https://www.lynda.com/3ds-Max-tutorials/nueva-interface-usuario-3ds-Max-2017/516945/516952-4.html

[66] Bianchini, Riccardo. GNAM - Galleria Nazionale d'Arte Moderna e Contemporanea - Rome. [Internet]. Roma, Italia. Inexhibit. 2019. Disponible en: https://www.inexhibit.com/mymuseum/galleria-nazionale-arte-moderna-e-contemporanea-rome/

[67] La Galleria Nazionale. [Internet]. Roma, Italia. Galleria Nazionale de Roma. Disponible en: https://lagallerianazionale.com/en/

[68] Cerón Cardona, Alejandro y Bedoya Herrera, Paola Andrea. Manual básico de Unity 3D como apoyo al desarrollo turístico nacional. [Internet]. Pereira, Colombia. Universidad Tecnológica de Pereira. Facultad de Ingeniería de Sistemas y Computación. 2014. Disponible en: http://repositorio.utp.edu.co/dspace/bitstream/handle/11059/4637/006686C416.pdf?sequence=1&isAllowed=y

[69] Unity Personal. [Internet]. EE. UU. Unity Store. Disponible en: https://store.unity.com/es/products/unity-personal

[70] Aprendiendo la Interfaz. [Internet]. EE. UU. Unity Documentation. Disponible en: https://docs.unity3d.com/es/530/Manual/LearningtheInterface.html

[71] Occlusion Culling (eliminación selectiva). [Internet]. EE. UU. Unity Documentation. 2018. Disponible en: https://docs.unity3d.com/es/2018.4/Manual/OcclusionCulling.html

[72] Valem. Unity Tutorial: VR, Oculus Avatar and Grabbing Object setup IN 5 MINUTES. [Internet]. Francia. Youtube. 2017. Disponible en: https://www.youtube.com/watch?v=sxvKGVDmYfY

[73] Valem. Introduction to VR in Unity – PART 1: VR SETUP. [Internet]. Francia. Youtube. 2020. Disponible en: https://www.youtube.com/watch?v=gGYtahQjmWQ

[74] Become A Better Programmer. How to Set up XR Interaction Toolkit with Oculus Quest in Unity. [Internet]. EE. UU. Youtube. 2020. Disponible en: https://www.youtube.com/watch?v=ZaVD1oVHQD8

[75] Oculus Integration SDK. [Internet]. EE. UU. Oculus Developers. 2021. Disponible en: https://developer.oculus.com/downloads/package/unity-integration/

[76] Oculus Avatar SDK. [Internet]. EE. UU. Oculus Developers. 2020. Disponible en: https://developer.oculus.com/downloads/package/oculus-avatar-sdk/

[77] Import Oculus Integration Package. [Internet]. EE. UU. Oculus for developers. Disponible en: https://developer.oculus.com/documentation/unity/unity-import/

[78] Oculus. Comenzando (Windows). [Internet]. EE. UU. Unity Documentation. Disponible en: https://docs.unity3d.com/es/530/Manual/VRDevices-Oculus.html

[79] Configuring your Unity Project for XR. [Internet]. EE. UU. Unity Documentation. 2020. Disponible en: https://docs.unity3d.com/Manual/configuring-project-for-xr.html

[80] Understand Oculus integration Package Components. [Internet]. EE. UU. Oculus for developers. Disponible en: https://developer.oculus.com/documentation/unity/unity-utilities-overview/#documentation/unity-import/

[81] Importing Objects From 3D Studio MAX. [Internet]. EE. UU. Unity Documentation. 2017. Disponible en: https://docs.unity3d.com/2017.4/Documentation/Manual/HOWTO-ImportObjectMax.html

[82] Colliders. [Internet]. EE. UU. Unity Documentation. 2018. Disponible: https://docs.unity3d.com/es/2018.4/Manual/CollidersOverview.html

[83] ¿Cómo agregar Box Collider? [Internet]. Tutoriales de aplicaciones y video juegos. Aprende a crear apps, juegos y más! Disponible en: http://appgametutoriales.com/como-agregar-box-collider/

[84] Collider. [Internet]. EE. UU. Unity Documentation. Disponible en: https://docs.unity3d.com/ScriptReference/Collider.html

[85] codigofacilito. 25.- Curso Unity – Colisiones entre objetos. [Internet]. México. Youtube. 2015. Disponible en: https://www.youtube.com/watch?v=5K5Fbhd1Qo8

[86] Nereu, Arturo. Básico 07: Detección de Colisiones en Unity 3D. [Internet]. EE. UU. Youtube. 2013. Disponible en: https://www.youtube.com/watch?v=iCB6lcZrrT8

[87] Icradin. How to disable collider trigger on collision enter? [Internet]. EE. UU. Unity. 2013. Disponible en: https://forum.unity.com/threads/how-to-disable-collider-trigger-on-collision-enter.427891/

[88] GameObject.Find. [Internet]. EE. UU. Unity Documentation. Disponible en: https://docs.unity3d.com/es/530/ScriptReference/GameObject.Find.html

[89] Josh. Detect a gameObject's tag via OnCollisionEnter(). [Internet]. EE. UU. Unity. 2011. Disponible en: https://answers.unity.com/questions/154433/detect-a-gameobjects-tag-via-oncollisionenter.html

[90] Ashky. How can I set the position and size of a sprite from inside a script? [Internet]. EE. UU. Unity. 2014. Disponible en: https://answers.unity.com/questions/726936/how-can-i-set-the-position-and-size-of-a-sprite-fr.html

[91] Creando y usando scripts. [Internet]. EE. UU. Unity Documentation. 2018. Disponible en: https://docs.unity3d.com/es/2018.4/Manual/CreatingAndUsingScripts.html

[92] Escuela de Videojuegos. Acceder a variables y funciones entre Scripts | Tutorial Unity 5. [Internet]. Youtube. 2017. Disponible en: https://www.youtube.com/watch?v=u1dLdXBTBB8

[93] Lester, James. Accesing Global Variables from Another Script – C# Unity Tutorial. [Internet]. UK. Youtube. 2019. Disponible en: https://www.youtube.com/watch?v=VjAhAlk6OgY

[94] Controlando GameObjects utilizando Componentes. [Internet]. EE. UU. Unity Documentation. Disponible en: https://docs.unity3d.com/es/530/Manual/ControllingGameObjectsComponents.html

[95] UNITY Operaciones sobre objetos 3D. [Internet]. España. Wikipedia. Disponible en: https://wiki.cifprodolfoucha.es/index.php?title=UNITY_Operaciones_sobre_objetos_3D

[96] Creating and Applying Materials. [Internet]. EE. UU. Unity. Disponible en: https://docs.unity3d.com/Packages/com.unity.probuilder@4.0/manual/workflow-materials.html

[97] David 3. Show and Hide a prefab or GameObject. [Internet]. EE. UU. Unity. 2010. Disponible en: https://answers.unity.com/questions/14165/show-and-hide-a-prefab-or-gameobject.html

[98] ButterGames – Desarrollo de videojuegos. Activar o Desactivar un objeto | como crear tus juegos en Unity3D. [Internet]. Youtube. 2017. Disponible en: https://www.youtube.com/watch?v=gfaM-Ww82X8

[99] Akari. How to make game object transparent in Unity. [Internet]. Stack Over Net. 2013. Disponible en: https://stackovernet.xyz/es/q/5319593

[100] How to change the transparency of an object in Unity c# scripting? [Internet]. Game Development. Stack Exchange. 2015. Disponible en: https://gamedev.stackexchange.com/questions/105378/how-to-change-the-transparency-of-an-object-in-unity-c-scripting

[101] TechnoRazor. How do I change the transparency of objects? [Internet]. EE. UU. Unity. 2014. Disponible en: https://answers.unity.com/questions/863381/how-do-i-change-the-transparency-of-objects.html

[102] Aldonaletto. How to stop other game object sound? [Internet]. EE. UU. Unity. 2011. Disponible en: https://answers.unity.com/questions/183363/how-to-stop-other-game-object-sound.html

[103] AudioSource.Play. [Internet]. EE. UU. Unity Documentation. Disponible en: https://docs.unity3d.com/ScriptReference/AudioSource.Play.html

[104] AudioSource.clip. [Internet]. EE. UU. Unity Documentation. Disponible en: https://docs.unity3d.com/ScriptReference/AudioSource-clip.html

[105] Tiposdearte.com. Piet Mondrian Biografía Corta – técnicas y obras. [Internet]. Costa Rica. Tipos de arte. Disponible en: https://tiposdearte.com/biografias/piet-mondrian

[106] Pitts Rembet, Virginia. Piet Mondrian. [Internet]. EE. UU. Parstone International. 2016. Disponible en: https://ebookcentral--proquest--com.us.debiblio.com/lib/uses/detail.action?pq-origsite=primo&docID=5320977

[107] Armonización por el arte. Mondrian. Obra y BIOGRAFíA. [Internet]. Argentina. Youtube. 2020. Disponible en: https://www.youtube.com/watch?v=A9cPQjnrlq0

[108] El diablo de los números. Piet Mondrian (Mr. Boogie – Woogie Man) Documental en español. [Internet]. Youtube. 2020. Disponible en: https://www.youtube.com/watch?v=n7PFYA7OwDs

[109] Aracil, Alfredo. El neoplasticismo en la música. [Internet]. España. El País. 1982. Disponible en: https://elpais.com/diario/1982/03/25/cultura/385858808_850215.html

[110] Ciclo de música para una exposición Mondrian. [Internet]. Madrid, España. Fundación Juan March. 1982. Disponible en: https://recursos.march.es/culturales/documentos/conciertos/cc225.pdf

[111] JLPM, Piet Mondrian, curiosidades y sus obras más destacadas. [Internet]. Lienzos, cuadros y arte. Artículos sobre pintores, obras, lienzos, cuadros, óleos, pinturas, y exposiciones de pintores, artistas y arte en general. Disponible en: https://cuadrosylienzos.blogspot.com/2012/10/piet-mondrian-curiosidades-y-obras-destacadas.html

[112] Texto a voz español gratis. [Internet]. Text to speech robot. Disponible en: http://texttospeechrobot.com/tts/es/texto-a-voz/

[113] Piet Mondrian. [Internet]. EE. UU. Y Francia. WahooArt.com. Disponible en: https://es.wahooart.com/A55A04/W.nsf/O/BRUE-8LT565

[114] Denis Shiryaev. [60 fps] A Trip Through the Streets of Amsterdam, 1922. [Internet]. Polonia. Youtube. 2020. Disponible en: https://www.youtube.com/watch?v=6tykGHGhC00

[115] Play video in unity 3D. [Internet]. India. Gyanendu Shekhar's Blog. 2020. Disponible en: http://gyanendushekhar.com/2020/03/15/play-video-in-unity-3d/

[116] Video Player component. [Internet]. EE. UU. Unity documentation. 2020. Disponible en: https://docs.unity3d.com/Manual/class-VideoPlayer.html

[117] TeleAdhesivo. Nederland – Pegatinas. [Internet]. Pinterest. Disponible en: https://www.pinterest.es/pin/266064290468620243/

[118] Mons, Marc. Bandera modelo 3d. [Internet]. Barcelona, España. Turbo Squid. 2015. Disponible en: https://www.turbosquid.com/es/3d-models/flag-max-free/974692

[119] Khos85. Script to break mesh into smaller pieces? [Internet]. EE. UU. Unity. 2015. Disponible en: https://answers.unity.com/questions/1006318/script-to-break-mesh-into-smaller-pieces.html

[120] Robertbu. How to toggle off / on raycast on a Gameobject. [Internet]. EE. UU. Unity. 2013. Disponible en: https://answers.unity.com/questions/512843/how-to-toggle-off-on-raycast-on-a-gameobject.html

[121] Bringer, Acorn. Simplistic Low Poly Nature. [Internet]. Brasil. Unity Asset Store. 2018. Disponible en: https://assetstore.unity.com/packages/3d/environments/simplistic-low-poly-nature-93894#content

[122] Creating Prefabs. [Internet]. EE. UU. Unity Documentation. 2020. Disponible en: https://docs.unity3d.com/Manual/CreatingPrefabs.html

[123] Fernández González, Ángel D. Unity Prefabs. [Internet]. España. Wikipedia. 2018. Disponible en: https://wiki.cifprodolfoucha.es/index.php?title=Unity_Prefabs

[124] Creando y destruyendo GameObjects. [Internet]. EE. UU. Unity Documentation. 2018. Disponible en: https://docs.unity3d.com/es/2018.4/Manual/CreateDestroyObjects.html

[125] Object.Instantiate. [Internet]. EE. UU. Unity Documentation. 2016. Disponible en: https://docs.unity3d.com/es/530/ScriptReference/Object.Instantiate.html

[126] Renaissance Coders. Unity C# Random Object Instantiation. [Internet]. Youtube. 2017. Disponible en: https://www.youtube.com/watch?v=kmU7d4Sqblk

[127] Michael Dowell. Unity how to instantiate prefabs. [Internet]. Youtube. 2016. Disponible en: https://www.youtube.com/watch?v=tz2fRF2GnqY

[128] LuukArts. Script for butterfly roaming. [Internet]. EE. UU. Unity. 2017. Disponible en: https://forum.unity.com/threads/script-for-butterfly-roaming.508849/

[129] Fafase. Random insect movement. [Internet]. EE. UU. Unity. 2012. Disponible en: https://answers.unity.com/questions/308017/random-insect-movement.html

[130] Random.Range. [Internet]. EE. UU. Unity Documentation. 2020. Disponible en: https://docs.unity3d.com/ScriptReference/Random.Range.html

[131] Random.Range. [Internet]. EE. UU. Unity Documentation. 2016. Disponible en: https://docs.unity3d.com/es/530/ScriptReference/Random.Range.html

[132] Arcane Energy. Random Object Spawner in Unity. [Internet]. Youtube. 2019. Disponible en: https://www.youtube.com/watch?v=WgKxnqfuOmk

[133] Saber es práctico. ¿Cómo es la bandera de los Estados Unidos? (con imagen). [Internet]. Saber es práctico. 2012. Disponible en: https://www.saberespractico.com/curiosidades/como-es-la-bandera-de-los-estados-unidos-con-imagen/

[134] Printable_models. Eiffel Tower V1. [Internet]. Free3D. 2018. Disponible en: https://free3d.com/3d-model/-eiffel-tower-v1--470573.html

[135] Printable_models. Airplane V1. [Internet]. Free3D. 2018. Disponible en: https://free3d.com/3d-model/airplane-v1--79106.html

[136] Griegoromano. El Cubismo. [Internet]. Vanguardias artísticas del siglo XX. 2015. Disponible en: https://vanguardiaartisticasigloxx.wordpress.com/2015/08/07/el-cubismo/

[137] Denis Shiryaev. [4k, 60 fps] A Trip Through New York City in 1911. [Internet]. Polonia. Youtube. 2020. Disponible en: https://www.youtube.com/watch?v=hZ1OgQL9_Cw

[138] C-sharp note. [Internet]. Basic Music Theory. 2021. Disponible en: https://www.basicmusictheory.com/c-sharp-note

[139] Pacway. PACWAY's latest sounds. [Internet]. Free sound. 2018. Disponible en: https://freesound.org/people/PACWAY/

[140] Tipos de luz. [Internet]. EE. UU. Unity Documentation. 2018. Disponible en: https://docs.unity3d.com/es/2018.4/Manual/Lighting.html

[141] Light. [Internet]. EE. UU. Unity Documentation. 2020. Disponible en: https://docs.unity3d.com/ScriptReference/Light.html

[142] Ealbinu. CambiarLuz. [Internet]. México. GitHub Gist. 2013. Disponible en: https://gist.github.com/ealbinu/5441529

[143] 57. Clase light (II). [Internet]. España. Tutorial de scripts para Unity 3d. 2011. Disponible en: http://unityscripts.blogspot.com/2011/10/57-clase-light-ii.html

[144] Administrador. Script Unity – Encender o apagar lámpara. [Internet]. Ecuador. Unity3D - Script. 2013. Disponible en: http://unity3d-script.blogspot.com/2013/12/script-unity-encender-o-apagar-lampara.html

[145] Delgado, José. ¿Cómo puedo encender un objeto que está apagado al iniciar el juego en Unity? [Internet]. España. Stack Overflow. 2019. Disponible en: https://es.stackoverflow.com/questions/259541/c%C3%B3mo-puedo-encender-un-objeto-que-est%C3%A1-apagado-al-iniciar-el-juego-en-unity

[146] Eno-Khaon. Change Color in C# with RGB Values? [Internet]. EE. UU. Unity. 2018. Disponible en: https://answers.unity.com/questions/1211937/change-color-in-c-with-rgb-values.html

[147] Hexagonius. How do you get a color from another game object. [Internet]. EE. UU. Unity. 2017. Disponible en: https://answers.unity.com/questions/1313161/how-do-you-get-a-color-from-another-game-object.html

[148] Viloria, Ignacio. Piet Mondrian: del apocalipsis al boogie-woogie. [Internet]. España. Líneas sobre arte. 2016. Disponible en: https://lineassobrearte.com/2016/08/12/piet-mondrian-del-apocalipsis-al-boogie-woogie/

[149] FusedVR. Apple ARKit Tutorial: How to Build Tilt Brush Paint Demo in Unity. [Internet]. Youtube. 2017. Disponible en: https://www.youtube.com/watch?v=Kj8mWHkBTTw

[150] FusedVR. Unity VR Tutorial: How to Build Tilt Brush From Scratch. [Internet]. Youtube. 2016. Disponible en: https://www.youtube.com/watch?v=eMJATZI0A7c

[151] Lleon79. How do I replace “SetVertexCount”? [Internet]. EE. UU. Unity. 2017. Disponible en: https://answers.unity.com/questions/1413044/how-do-i-replace-setvertexcount.html

[152] Dilmer Valecillos. Oculus Quest Development – VR Draw Open Source Project Editor Options. [Internet]. EE. UU. Youtube. 2019. Disponible en: https://www.youtube.com/watch?v=73ZWhQfxATI

[153] Dilmer Valecillos. Oculus Quest Development – An Intro To My Oculus VR Open Source Project VRDraw! [Internet]. EE. UU. Youtube. 2019. Disponible en: https://www.youtube.com/watch?v=SYY0BYP0ncE

[154] Map Controllers. [Internet]. EE. UU. Oculus for developers. Disponible en: https://developer.oculus.com/documentation/unity/unity-ovrinput/

[155] Takatok. UI Raycast Target Enabled by Default Issue. [Internet]. EE. UU. Unity. 2016. Disponible en: https://forum.unity.com/threads/ui-raycast-target-enabled-by-default-issue.429475/

[156] Random Art Attack. Unity 3D Particle Sub Emitters: Making Fireworks. [Internet]. Youtube. 2017. Disponible en: https://www.youtube.com/watch?v=nlLow7DbKkY

[157] ParticleSystem. [Internet]. EE. UU. Unity Documentation. 2020. Disponible en: https://docs.unity3d.com/ScriptReference/ParticleSystem.html?_ga=2.92177114.1740463657.1613040849-1059153934.1612700180

[158] ParticleSystem.GetParticles. [Internet]. EE. UU. Unity Documentation. 2020. Disponible en: https://docs.unity3d.com/ScriptReference/ParticleSystem.GetParticles.html?_ga=2.92177114.1740463657.1613040849-1059153934.1612700180

[159] Brackeys. Everything to know about the PARTICLE SYSTEM. [Internet]. Dinamarca. Youtube. 2018. Disponible en: https://www.youtube.com/watch?v=FEA1wTMJAR0

[160] Don Pachi. Tutorial: Como crear efectos de partículas en Unity. [Internet]. Perú. Youtube. 2019. Disponible en: https://www.youtube.com/watch?v=BxMefgiaGCs

[161] USA TODAY Life. 12 Minute Fireworks Show with Sound 4K UHD 60FPS | GRATEFUL. [Internet]. EE. UU. Youtube. 2019. Disponible en: https://www.youtube.com/watch?v=E7gAmFmKBv0

[162] Rf Mvs. ¿Cómo activar sistema de partículas en Unity? [Internet]. Stack overflow. 2018. Disponible en: https://es.stackoverflow.com/questions/179249/c%C3%B3mo-activar-sistema-de-part%C3%ADculas-en-unity

[163] HolBol. Turning the Particle System on and off. [Internet]. EE. UU. Unity. 2010. Disponible en: https://answers.unity.com/questions/37875/turning-the-particle-system-on-and-off.html

[164] Z0hann. Toggle particle system on or off. [Internet]. EE. UU. Unity. 2019. Disponible en: https://answers.unity.com/questions/1642426/toggle-particle-system-on-or-off.html

[165] Dallimore, Geoff. Real Stars Skybox Lite. [Internet]. EE. UU. Unity Asset Store. 2018. Disponible en: https://assetstore.unity.com/packages/3d/environments/sci-fi/real-stars-skybox-lite-116333

[166] Skybox. [Internet]. EE. UU. Unity Documentation. 2017. Disponible en: https://docs.unity3d.com/560/Documentation/Manual/class-Skybox.html

[167] PresidentPorpoise. Disable and enable emission property of a material via script? [Internet]. EE. UU. Unity. 2016. Disponible en: https://forum.unity.com/threads/disable-and-enable-emission-property-of-a-material-via-script.445016/

[168] Code_monkeee. How do I check if a material's Emission property is currently enabled? [Internet]. EE. UU. Reddit. 2017. Disponible en: https://www.reddit.com/r/Unity3D/comments/6enoij/how_do_i_check_if_a_materials_emission_property/

[169] HarpSeal1. Setting emission scale in script. [Internet]. EE. UU. Unity. 2013. Disponible en: https://forum.unity.com/threads/setting-emission-scale-in-script.297525/

[170] VintagePoet82. Trying to make object shake via script (I'm not a coder, got this online). How to fix this error? "Cannot modify the return value of 'Transform.position' because it is not a variable". [Internet]. EE. UU. Reddit. 2020. Disponible en: https://www.reddit.com/r/Unity3D/comments/fo6ax0/trying_to_make_object_shake_via_script_im_not_a/

[171] MonoBehaviour.Awake(). [Internet]. EE. UU. Unity Documentation. 2020. Disponible en: https://docs.unity3d.com/ScriptReference/MonoBehaviour.Awake.html

[172] Blizzy. How do you make particle system follow a game object? [Internet]. EE. UU. Unity. 2015. Disponible en: https://forum.unity.com/threads/how-do-you-make-particle-system-follow-a-game-object.324273/

[173] Rousseau. Yann Tiersen – La vlase d'Amélie. [Internet]. Youtube. 2019. Disponible en: https://www.youtube.com/watch?v=uj9BihmugmI