Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Desarrollo de un entorno de realidad virtual para la evocación de serenidad en experimentos de detección de emociones

Autor: Fernando Cárdenas Baña
Tutoras: Irene Fondón García
María Auxiliadora Sarmiento Vega

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

Sevilla, 2013

Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Desarrollo de un entorno de realidad virtual para la evocación de serenidad en experimentos de detección de emociones

Autor:
Fernando Cárdenas Baña

Tutoras:
Irene Fondón García, Profesora Titular de Universidad
María Auxiliadora Sarmiento Vega, Profesora Contratada Doctora

Dpto. de Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2020

Proyecto Fin de Carrera: Desarrollo de un entorno de realidad virtual para la evocación de serenidad en experimentos de detección de emociones

Autor: Fernando Cárdenas Baña

Tutoras: 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, 2020

El Secretario del Tribunal

A mi familia

A mis maestros

Agradecimientos

En primer lugar, expresar mi enorme agradecimiento a las tutoras de este trabajo de fin de grado, Irene Fondón García y María Auxiliadora Sarmiento Vega, por el apoyo brindado, tanto por el tutelaje de este trabajo como por la innovación y dedicación implicadas en las asignaturas impartidas. La visión de ambas de lo que un ingeniero de telecomunicaciones puede llegar a hacer fue en gran medida lo que me hizo decantar por esta especialidad del grado.

Agradecer a mis padres, Fernando y María Auxiliadora, quienes, de forma incondicional, me han apoyado sin descanso durante el transcurso de la carrera, instándome a nunca rendirme y otorgándome todas las facilidades que han estado al alcance de sus manos para que, a día de hoy, haya podido llegar aquí.

A mi hermana, Ana, por el vínculo tan fuerte por el que estamos unidos, por su habilidad para hacerme reír y despejar las nubes de mi mente cuando en esta sólo había fórmulas y números.

A Alba, quien, de un modo u otro, ha permanecido a mi lado desde el principio de este tan largo camino, recordándome que, con tiempo, no hay reto que no pueda superarse.

A mis amigos, y compañeros del grado, Antonio, Emilio y Manuel, por la ayuda prestada, por luchar codo con codo contra cada asignatura y por la incondicional amistad que surgió a raíz de ello.

Por último, y no por ello menos importante, al resto de mis familiares y amigos, los cuales son demasiados para nombrar aquí. Gracias por todo el cariño y afecto ofrecido.

Fernando Cárdenas Baña

Sevilla, 2020

Resumen

El objetivo de este trabajo es la creación y simulación de un entorno de realidad virtual para la evocación en un individuo de la emoción serenidad. Para ello se han empleado las herramientas ofrecidas por el motor de juegos Unity, el software 3ds Max y las gafas de realidad virtual Oculus Rift S.

La generación de emociones mediante el uso de aplicaciones implementadas en realidad virtual se ha estudiado desde sus orígenes y, gracias a los avances tecnológicos de hoy día, este campo de estudio ha experimentado un gran auge.

La inmersión controlada de los usuarios dentro de un entorno virtual permite exponerlos a diferentes situaciones para estudiar sus reacciones. Tomando medidas de las señales cerebrales de los sujetos bajo estudio, se puede llegar a entender mejor el comportamiento de los seres humanos.

Para este proyecto, se espera que el entorno generado evoque en el individuo la emoción anteriormente mencionada, la serenidad, la cual fue definida por el psicólogo Robert Plutchik como una emoción secundaria derivada de la emoción básica de la alegría [1].

El entorno generado, está compuesto por eventos y elementos que pueden interactuar con el usuario, amén de una misión principal, la cual, una vez cumplida, pondrá fin a la simulación del entorno.

Abstract

The objective of this work is the creation and simulation of a virtual reality environment for the evocation in an individual of the emotion serenity. For this purpose, we have used the tools offered by the Unity game engine, 3ds Max software and virtual reality glasses Oculus Rift S.

The generation of emotions through the use of applications implemented in virtual reality has been studied since its origins and, thanks to today's technological advances, this field of study has experienced a great boom.

The controlled immersion of users within a virtual environment allows them to be exposed to different situations to study their reactions. By taking measurements of the brain signals of the subjects under study, the behaviour of human beings can be better understood.

For this project, the generated environment is expected to evoke in the individual the aforementioned emotion, serenity, which was defined by psychologist Robert Plutchik as a secondary emotion derived from the basic emotion of joy. [1].

This environment is made up of events and elements which can be interacted with by the user, as well as a main mission, which, once completed, will end the simulation of the environment.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

Why shouldn't people be able to teleport wherever they want?

- Palmer Luckey -

La primera de las tareas que se debe realizar para el correcto entendimiento de este trabajo fin de grado es la definición del concepto de **Realidad Virtual**. Aun siendo un concepto muy amplio y existiendo discrepancias entre los distintos autores, podemos afirmar que la realidad virtual es, de manera general, un entorno simulado, con una enorme cantidad de datos interactivos, el cual ha sido generado por ordenador y que implica a todos los sentidos, siendo manipulable, explorable y visualizable en "tiempo real" y dando la sensación de presencia en el entorno informático [2].

El objetivo con el que se ha elaborado este trabajo es la total inmersión del sujeto bajo estudio en el entorno virtual generado mediante el software **Unity** para la evocación de la emoción secundaria "serenidad".

Con el fin de evocar la emoción anteriormente mencionada se ha creado un entorno "relajante", el cual carece de cualquier estímulo que pudiese provocar alerta, terror, frustración u otros tipos de incitativos que pudiesen sacar al sujeto del estado de calma en el que se le ha introducido.

El entorno generado para el desarrollo de este trabajo es totalmente *inmersivo*, es decir, es un ambiente tridimensional preparado para que el sujeto, mediante unas gafas de realidad virtual y un periférico controlador para cada una de sus manos, viva la experiencia de adentrarse en un mundo ficticio, interactivo en "tiempo real", colmado de estímulos auditivos y visuales que tienen como fin transmitir al sujeto la sensación de estar en plena naturaleza, carente de cualquier peligro, pudiendo explorarla en su totalidad a voluntad.

Aunque dicho entorno podría ser simulado de manera *no inmersiva*, es decir, simular dicho entorno a través de una ventana de escritorio, optamos por descartar dicha opción debido a que la experiencia *inmersiva* que logramos gracias al uso de las gafas de realidad virtual Oculus Rift S es mucho más real y vívida, haciendo más disfrutable la exploración a lo largo del entorno virtual generado.

En resumen, los objetivos perseguidos en la realización de este proyecto son:

- Generar un entorno virtual completamente funcional y realista.
- Generar en 3ds Max objetos para su integración en el entorno.
- Integrar en el entorno dichos objetos de forma que puedan interaccionar con el usuario.
- Generar scripts que modifiquen el comportamiento de los objetos integrados para la interacción con el usuario.

2 EMOCIONES

VR is a way to escape the real world into something more fantastic. It has the potential to be the most social technology of all time.

- Palmer Luckey -

El objetivo de este trabajo de fin de grado es el desarrollo de un entorno virtual capaz de evocar una emoción concreta en un individuo. Sin embargo, antes de comenzar dicha tarea, es primordial dar una respuesta a la pregunta "¿Qué es una emoción?".

El científico inglés Charles Darwin fue de las primeras personas en dar una respuesta a dicha pregunta en su libro *"The Expression of the Emotions in Man and Animals"* (1872) [3]. En él, presenta sus ideas acerca de las expresiones emocionales como evoluciones adaptativas que tienen dos propósitos principales: comunicación y supervivencia. Estas ideas fueron aceptadas por la mayor parte de la comunidad de biólogos, constituyendo las bases de la etología contemporánea. No obstante, las ideas expuestas por Darwin no tuvieron un gran impacto entre el colectivo de psicólogos, dando lugar a un largo periodo de tiempo en el que las contribuciones psicológicas en este área de estudio fueron muy escasas [4].

Es por tanto posible definir las emociones como reacciones físicas y psicológicas las cuales representan modos de adaptación de los individuos ante la presencia de objetos, lugares, personas, recuerdos o sucesos significativos. El objetivo es modificar o establecer una conducta en el individuo con el fin de adaptarse a las circunstancias del entorno en el cual está experimentando dicha emoción [3]. Hay varias fases o componentes que son posibles distinguir en la producción de las emociones:

- Recepción del estímulo que desencadena el proceso (objeto, persona, lugar, proceso o recuerdo).
- Reacción psicobiológica o expresión del sentimiento. Conductualmente, las emociones sirven para establecer un comportamiento adecuado a nuestro entorno. Fisiológicamente, sirven para organizar rápidamente las respuestas de distintos sistemas biológicos. Cognitivamente, las emociones alteran la atención, hacen subir de rango ciertas conductas y respuestas del individuo, y activan redes asociativas relevantes en la memoria.
- Percepción de las emociones. Los sentimientos emocionales aparecen con la percepción de lo que nuestro cuerpo hace mientras se manifiesta la emoción, junto con percepciones del estado de nuestra mente durante ese mismo período de tiempo [5].

2.1 Sistema Límbico

El sistema límbico es el responsable de las respuestas emocionales y espontáneas que experimentan los seres humanos. Los límites de las diferentes partes del encéfalo que forman parte del conjunto del sistema límbico son algo difusos, pero es totalmente correcto afirmar que dichas partes están conectadas entre sí y entre sus funciones se encuentran la aparición de los estados emocionales, los cuales tienen su principal base neurológica en esta red de neuronas [6].

Estos cambios en nuestro cuerpo se producen de manera física y, de algún modo, son reconocidos por nosotros mismos, es decir, podemos diferencias nuestras propias emociones.

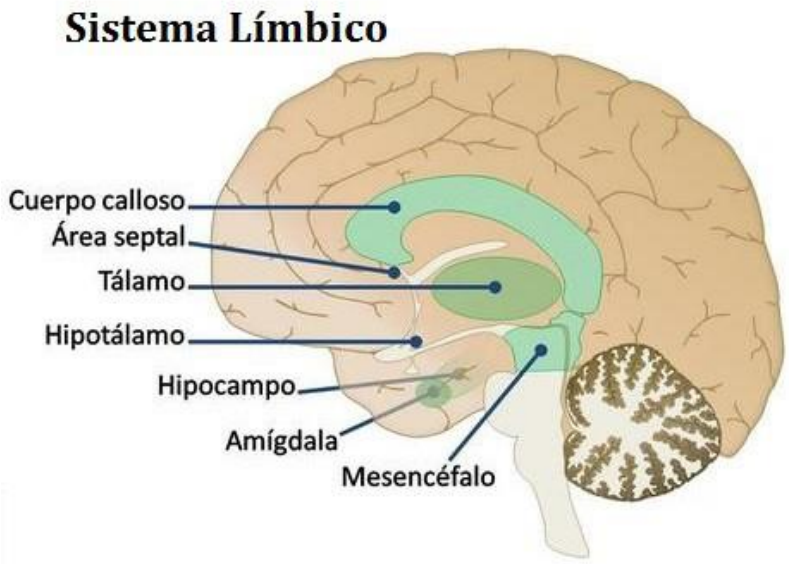

Figura 2-1. Sistema Límbico. [7]

Aunque el sistema límbico no es exactamente una región anatómicamente exacta del encéfalo, sino que es una red de neuronas distribuidas por el cerebro y que se mezclan con muchas estructuras diferentes, varios estudios afirman que el sistema límbico debe al menos estar formado por: el hipotálamo, la amígdala, el hipocampo y la corteza límbica [6].

2.1.1 Tálamo

El tálamo es, en esencia, un conjunto de sustancia gris (cuerpos de neuronas) formado por dos estructuras encefálicas con forma de huevo que se encuentran por debajo de la corteza cerebral. Interviene en una gran cantidad de procesos mentales que dan forma a nuestra manera de percibir las cosas y de actuar sobre el entorno que nos rodea, incluso no siendo nosotros mismos conscientes de ello [8].

Dentro de este conjunto de grupos neuronales es posible diferenciar entre tres núcleos:

- **Núcleos de conexión específica.** Mandan información sensorial a zonas concretas de la corteza cerebral especializadas en trabajar con datos provenientes de un sentido específico.
- **Núcleos de conexión inespecífica.** Mandan información a zonas muy amplias de la corteza cerebral,

sin discriminar por especializaciones.

- **Núcleos de asociación.** Forman parte de un circuito de información que comunica la corteza cerebral con estructuras subcorticales [8].

El tálamo tiene varias funciones y entre ella se encuentra la regulación de emociones, ya que, como hemos comentado, interactúa con vías neuronales que participan directamente en la aparición de estados emocionales. Además, recibe información del hipotálamo, el cual interviene directamente en la regulación de las emociones y de la segregación de distintas hormonas en el torrente sanguíneo [8].

2.1.2 Hipotálamo

El hipotálamo es, junto al tálamo, una de las partes de la estructura cerebral conocida como diencéfalo. Se encuentra en el centro del encéfalo de los seres humanos, justo por debajo de la corteza cerebral y por encima del tronco del encéfalo.

Esta estructura cerebral juega un papel muy importante en la regulación de la temperatura corporal, del sueño, de impulsos sexuales, del hambre, la sed y de los estados de ánimo. La puesta en marcha y coordinación de los procesos que nos permiten sobrevivir y responder de manera adecuada a los estímulos del medio que nos rodea son llevados a cabo por el hipotálamo.

Al actuar de puente entre el cerebro y el sistema endocrino, coordina todo lo que se realiza a través del sistema nervioso autónomo, en otras palabras, envía órdenes a distintas partes del cuerpo para que estas se adapten a la situación.

Debido a su conexión con la glándula pituitaria y, por consecuente, con el sistema endocrino y todas las partes del cuerpo en las que se liberan todo tipo de hormonas, el hipotálamo es una de las zonas del diencéfalo con más involucración en la regulación de las emociones [8].

2.1.3 Hipocampo

Localizado en la cara interior de los lóbulos temporales, muy cerca del tálamo y las amígdalas, encontramos el hipocampo. El hipocampo tiene una función muy importante en procesos mentales relacionados con la memoria, ya sean experiencias, informaciones abstractas o recuerdos.

Se encuentra encuadrado en lo que se conoce como corteza del lóbulo límbico o arquicorteza, la cual es una de las partes de la corteza cerebral más antiguas ya que apareció muy pronto en la línea evolutiva que ha llevado a la aparición del ser humano.

Es el encargado tanto del aprendizaje como de la consolidación de nuestra memoria. Permite que la memoria a corto plazo se consolide para dar lugar a memoria a largo plazo. Además, al hablar de memoria, no solo nos estamos refiriendo a las experiencias a corto o largo plazo vividas por un individuo, sino también a los sentimientos asociados a dichos recuerdos. Este último hecho puede explicar por qué el hipocampo es una de las partes más antigua de la corteza cerebral, ya que el asociamiento de sentimientos a los recuerdos de experiencias vividas es una gran herramienta tanto de aprendizaje como de supervivencia [6].

2.1.4 Amígdala

Las amígdalas cerebrales, cuya forma es similar a una almendra, están situadas al lado de cada hipocampo. Se trata de estructuras subcorticales situadas en la parte interna del lóbulo temporal medial. La principal función llevada a cabo por las amígdalas es la integración de las emociones mediante patrones de respuestas para

dichas emociones, lo que provoca una respuesta a nivel fisiológico o una respuesta conductual. Es por tanto el principal núcleo de control de las emociones y sentimientos en el cerebro [6][9].

Las amígdalas gestionan de manera precisa la emisión de respuestas emocionales, a nivel consciente e inconsciente, siendo una de las funciones más relevantes que posee. Permite asociar a las experiencias vividas sensaciones de aversión o gratificación, contribuyendo así al desarrollo del aprendizaje emocional, permitiendo la elección de estrategias a aplicar ante la presencia de estímulos, así como detectar las situaciones en las que estas estrategias son aplicables. Además de las reacciones emocionales producidas debido a las conexiones que posee, también permite la inhibición de ciertas conductas gracias a la vinculación con el lóbulo frontal [9].

Otras funciones que desempeña esta estructural cerebral son:

- La gestión del miedo y reacción de lucha o huida, lo cual es una pieza clave en la supervivencia del ser humano.
- La memoria, ya que la amígdala también afecta a la estructuración de los recuerdos debido a la asociación de los mismos con estados emocionales.
- La regulación de la conducta sexual.
- La agresividad, la cual se ve potenciada frente a una híper estimulación del complejo amigdalino, o reducida la reacción de agresividad y autodefensa ante un mal funcionamiento de la amígdala.
- El control de la ingesta, ya que un mal funcionamiento de la amígdala daría lugar a fallo en la percepción de la saciación [9].

2.1.5 Corteza orbitofrontal

La corteza orbitofrontal se encuentra en los límites del sistema límbico, formando parte de la corteza prefrontal. Su función no es otra que desechar los impulsos irracionales que llegan por parte del sistema límbico. También está implicada en el establecimiento de lo que denominamos personalidad [6].

2.2 Funciones de las emociones

Tras haber expuesto el sistema límbico y las distintas partes que lo componen, las cuales dan lugar a la generación y gestión de las emociones, es importante preguntarse para qué sirven dichas emociones y qué funciones desempeñan, ya sean positivas o negativas.

Estas funciones pueden ser de tres tipos:

- **Adaptativa.** Cada emoción, negativa o positiva, tiene una función determinada. Facilitan el ajuste del organismo a nuevas condiciones ambientales.
- **Motivacional.** Potencian y dirigen conductas mediante la atracción-repulsión de las mismas. Así, ante información variada e incompleta, como la resolución de decisiones difíciles, nos guiaremos mediante el uso de nuestras emociones, acercándonos a aquello que pueda evaluarse como agradable y repudiando lo que nos resulte desagradable o aversivo.
- **Comunicativa.** Puede ser intrapersonal, como fuente de información, o interpersonal, en una dimensión social, comunicando sentimientos e intenciones, influyendo en la conducta de otros y potenciando las relaciones [10].

2.3 Tipos de emociones y su clasificación

Aunque no es posible establecer una clasificación totalmente precisa acerca de cómo se agrupan y se relacionan entre sí las emociones, son muchos los teóricos que coinciden en que existen dos grandes grupos de emociones:

- **Emociones primarias.** También conocidas como emociones básicas, son aquellas que experimentamos directamente como respuesta a un estímulo. El psicólogo Robert Plutchik definió ocho emociones básicas: éxtasis, admiración, terror, asombro, pena, odio, furia y vigilancia. Todas ellas constituyen procesos cuya finalidad es la supervivencia y adaptación al medio donde se produce el estímulo, sin importar la procedencia o cultura en la que se haya desarrollado el individuo [1].
- **Emociones secundarias.** Son un grupo de emociones derivadas de las emociones primarias. Estas emociones son causa directa de normas sociales y normas morales [1]. Así, por ejemplo, se podría decir que una persona que se ha asustado de un ruido estruendoso ha experimentado una emoción secundaria, susto, a raíz de las emociones primarias de terror y asombro.

Además, ya sea una emoción primaria o secundaria la que afecte al comportamiento del sujeto, éstas pueden ser clasificadas como emociones positivas o emociones negativas. Las emociones positivas se conocen también como emociones saludables. Afectan positivamente al bienestar del individuo, favoreciendo la manera en que piensa, razona y actúa.

Las emociones negativas, por el contrario, afectan negativamente al bienestar de los individuos, llegando a provocar el deseo de evitarlas. No obstante, en pequeñas cantidades y con baja intensidad no son perjudiciales y nos permiten recordar las consecuencias de ciertas conductas gracias a nuestra memoria emocional [11].

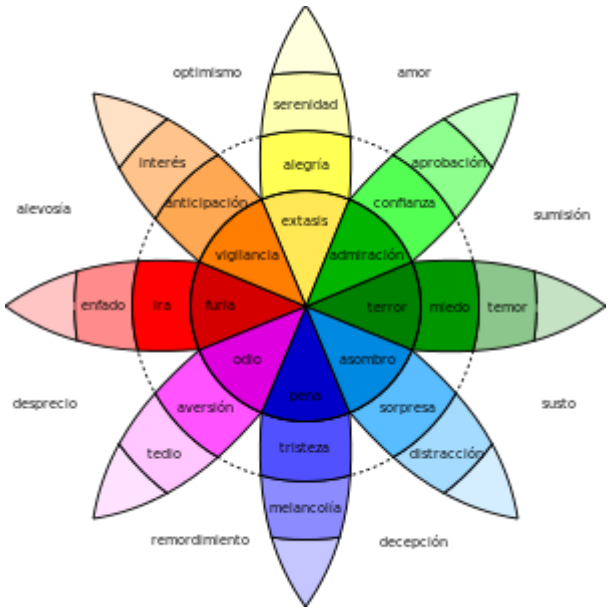

Figura 2-2. Rueda de las emociones de Plutchik. [12]

Una vez comentadas los distintos tipos de emociones existentes y los grupos en los que se encuentran clasificadas, podemos dar paso a la clasificación de las mismas mediante sistemas automáticos.

Estos sistemas de clasificación automáticos reciben señales fisiológicas provenientes del sistema nervioso central y, a su vez, del sistema nervioso autónomo. La técnica más usada a día de hoy para la obtención de las señales es el electroencefalograma, ya que las emociones se producen en el cerebro [13].

El electroencefalograma no es más que una serie de electrodos conectados alrededor de la cabeza del individuo del que se van a obtener las señales. Es un método no invasivo que hace uso de la corriente iónica de los potenciales postsinápticos para medir cualquier fluctuación que se produzca en la actividad cerebral. Para que la medición sea lo más exacta posible, es necesario que dichos electrodos estén lo más cerca posible del cuero cabelludo, por lo que son preferibles las mediciones en individuos con la cabeza rapada o personas alopécicas [13].

3 ESTADO DEL ARTE

There are as many applications for VR as you can think of; it's restricted by your imagination.

- John Goddard -

En este capítulo haremos un recorrido por los orígenes de la realidad virtual y las distintas experiencias en este campo que se han llevado a cabo hasta ahora, centrándonos especialmente en aquellas cuyo propósito sea el mismo que el de este trabajo fin de grado.

3.1 Historia

Es difícil establecer el momento exacto en el que el término "realidad virtual" se acuñó por primera vez, no obstante, muchos autores coinciden en que cuando hablamos de realidad virtual, es necesario remontarse a la década de 1960.

Uno de los principales impulsores de los ordenadores gráficos fue Ivan Edward Sutherland, cuya tesis *"Sketchpad: a man-machine graphical communication system"*, publicada en 1963, manifestó que era posible crear gráficos interactivos mediante el uso de ordenadores desarrollados para tal propósito [14].

Dicha tesis estuvo bien fundamentada, pues en 1959 Sutherland estuvo trabajando con un ordenador conocido como TX-2, desarrollado por los laboratorios Lincoln. Disponía de un panel de interruptores, un lápiz óptico y una pantalla de nueve pulgadas sobre la que Sutherland podía dibujar planos directamente gracias al uso de dicho lápiz. Esta pantalla está considerada la primera interfaz gráfica de la historia, incluso cuando aún no había sido definido tal término [15].

Años después, en 1968, Sutherland publicó otra tesis titulada "*A head-mounted three dimensional display*", tesis que sentó las bases para el desarrollo de los cascos de realidad virtual que usamos hoy día.

En el año 1966, Ivan Sutherland y Bob Sproull, un estudiante colaborador, utilizaron y modificaron un casco desarrollado por la compañía Bell Helicopter, el cual poseía una cámara de visión nocturna para permitir a los pilotos aterrizar en la oscuridad. Sutherland y Sproull reemplazaron dicha cámara por un conjunto de imágenes generadas mediante el uso de un ordenador. El escenario virtual que crearon consistía en una habitación con cuatro paredes con los puntos cardinales anotados sobre las mismas. Estando "dentro" de dicha habitación, el usuario podía mover la cabeza y mirar en las distintas direcciones [16].

Sin embargo, hasta este momento no se había aún acuñado el término de realidad virtual. Es en la década de 1980 cuando Jaron Lanier hizo referencia por primera vez a este término. De la mano de Lanier han venido algunas de las innovaciones técnicas y de los grandes avances que han ocurrido en este campo hasta día de

hoy.

En el año 1994 la compañía *Silicon Graphics* desarrolló un lenguaje al que llamaron VRML 1.0. Este lenguaje estaba destinado al desarrollo de aplicaciones de realidad virtual en Internet, o dicho de otra manera, a la creación de mundos virtuales tridimensionales cuyos objetos son interactivos [17].

Había tres razones las cuales hacían de VRML 1.0 un lenguaje muy interesante:

- Era extensible, es decir, podía ser ampliado para realizar mejoras y modificaciones.
- Su ejecución era totalmente independiente de la plataforma en la que fuese ejecutada el visualizador.
- Trabajaba de manera muy eficiente aún con conexiones lentas.

Pronto los mundos virtuales estáticos dejaron de ser suficiente para los usuarios, por lo que se necesitaba que los objetos tuviesen su propio comportamiento y que el usuario pudiera interaccionar con dichos objetos.

La compañía *Silicon Graphics* lanza VRML 2.0 en el año 1996. Esta nueva versión del lenguaje permitía a los usuarios interactuar con el mundo virtual en el cual se veían inmersos. Entre sus nuevas mejoras destacan las siguientes:

- A cada objeto puede asociarse un *script*, en lenguaje VRML u otro lenguaje, el cual definirá el comportamiento de dicho objeto de cara a la interacción con el mundo y con el usuario.
- Se añaden efectos de fondo, sonidos tridimensionales y demás factores ambientales los cuales hacen más real y vívida la experiencia dentro del entorno virtual.

En el año 1999 el empresario Philip Rosedale, fundador de Linden Lab, centró sus estudios en el desarrollo de Hardware orientado a la realidad virtual. Como fruto de su trabajo nació *Second Life*, un famoso mundo virtual 3D, donde es posible crear un avatar e interactuar en línea con otros jugadores. Dicho programa cuenta con su propia herramienta para la creación de objetos 3D a partir de figuras geométricas simples.

En 2007 Google lanzó *Google Street View*, un servicio que muestra vistas panorámicas de un inmenso número de lugares de nuestro planeta. Además, desde el año 2010, puede ser usado en modo estereoscópico 3D.

Finalmente, esta tecnología experimentó un gran auge en el año 2010 debido al empleo de los productos de realidad virtual enfocados al turismo, a los videojuegos y a las películas con realidad aumentada. Fue durante este año cuando Palmer Luckey diseñó el primer prototipo de Oculus Rift.

3.2 Experiencias en realidad virtual

Podemos afirmar sin temor a equivocarnos que la realidad virtual es una tecnología emergente con muy pocos años desde su nacimiento hasta día de hoy. No obstante, esta tecnología ha permitido abrir campos de investigación en sectores tales como la educación, tratamientos psicológicos o el entretenimiento.

A continuación, trataremos algunos de los experimentos que se han llevado a cabo en los campos anteriormente mencionados de mano de esta tecnología.

3.2.1 Educación

La preparación de los estudiantes para enfrentarse a un mundo globalizado es muy compleja y costosa y, en algunos casos, resulta inalcanzable. Los estudiantes deben aprender no sólo teoría, sino también práctica y es aquí donde surgen los inconvenientes, ya sean de carácter económico, debido a la falta de recursos materiales, o en el ámbito de la seguridad humana al no tener el equipamiento adecuado para evitar accidentes u otros inconvenientes.

Gracias a la realidad virtual los estudiantes pueden involucrarse de manera multisensorial en entornos virtuales donde podrán experimentar, tomar decisiones y explorar varias soluciones de entre todos los caminos posibles para la resolución de problemas. En un entorno virtual donde todo suceso posible está controlado, no importa cuántas veces falle un alumno hasta llevar a cabo la realización de la tarea, pues podrá repetir tantas veces sea necesario hasta adquirir las habilidades, destrezas y conocimientos que le permitan resolver dicha posible situación en la vida real.

- En la Universidad de Washington se ha creado un "cadáver virtual" con el que los estudiantes u otros profesionales de la materia pueden realizar prácticas y adquirir conocimientos mediante la manipulación de dicho "cadáver" en lugar del tradicional empleo de un verdadero cadáver humano [18].
- En Canadá se ha desarrollado el sistema *Mandala*, el cual permite a estudiantes de danza practicar y aprender movimientos de baile además de desarrollar su habilidad musical mediante el empleo de instrumentos "virtuales" [19].
- Las prácticas de vuelo, ya sea para pilotos militares o comerciales, también han sido implementadas en multitud de lugares mediante tecnologías de realidad virtual, las cuales permiten a los pilotos aprender y practicar su habilidad de vuelo, amén de medir sus tiempos de reacción sin el peligro que supone estar pilotando un verdadero avión [20].
- Las herramientas de realidad virtual también permite a estudiantes y profesionales del mundo de la arquitectura y de la ingeniería beneficiarse de herramientas tales como **Autocad**, **3ds Max**, **3DStudio**... y similares, las cuales facilitan la labor tanto del aprendizaje como del desarrollo de grandes proyectos.

3.2.2 Entretenimiento

La realidad virtual permite la creación de un entorno virtual el cual, presentado a nuestros sentidos, permite experimentar sensaciones como si realmente estuviéramos en el entorno simulado y evocásemos las mismas sensaciones que las que evocaríamos de estar en él realmente.

Llevado al mundo del entretenimiento, esto podría traducirse como que sería posible realizar visitas turísticas a museos, ciudades, monumentos… De una forma mucho más asequible económicamente y más rápida de lo que podríamos hacerla visitando dichos lugares realmente. Pero no solo el entretenimiento es aplicable al sector turístico. También es posible evocar las emociones que se manifestarían en el cerebro humano en otras circunstancias, como contemplar una obra de arte gracias a las texturas y colores de los que se compone, las emociones que experimentaría en una persecución de automóviles con la correspondiente liberación de adrenalina o la evocación de las emociones que experimentaría de estar entre una gran multitud de personas que componen el público de un concierto.

- Existen un inmenso número de videojuegos con soporte para la realidad virtual. Entre los más famosos destacan *The Elder Scroll V: Skyrim (2011)*, *SuperHOT* (2016), *Half-Life: Alyx* (2020), *Lies Beneath* (2020) o *Saints and Sinners* (2020). Si el videojuego por sí mismo consiste en un mundo virtual en el que el jugador experimenta la vida de un personaje ficticio, la aplicación de la realidad virtual para la inmersión del jugador en el entorno virtual lo involucra hasta el punto de "convertirse" en el propio personaje, el verdadero protagonista de la aventura.

- La cultura también se ha visto beneficiada del uso de la realidad aumentada para poder acercarse aún más al usuario. Muchos museos como el Museo Arqueológico Nacional (Madrid), el Museo Británico (Londres), el Museo Vaticano (Ciudad del Vaticano), el Museo Nacional de Historia Natural del Instituto Smithsoniano (Washington D.C.) o el Museo Hermitage (San Petersburgo) son ejemplos de lugares que han sabido aprovechar el potencial de la realidad virtual para llegar a un público más amplio.
- En el mundo cinematográfico también podemos encontrar algunos ejemplos de películas que pueden ser disfrutadas en realidad virtual, como son *Invasion!*, *Leggenda 360* y *My Africa*. Además, la aparición de los cines 4DX, cuyas salas emulan efectos atmosféricos y cuentan con asientos con efectos de movimiento, son un aliciente a la propulsión de esta tecnología para la innovación cinematográfica [21].

3.2.3 Tratamientos psicológicos

La realidad virtual permite una *inmersión total* en una simulación de la realidad en la que el usuario puede interactuar con el mundo virtual, de una forma similar a como interactúa con el mundo real.

En el campo de la Psicología Clínica, la utilización de esta tecnología implica que podemos disponer de simulaciones de la realidad para llevar a cabo tratamientos psicológicos. La ventaja de esta nueva herramienta es que cada uno de los elementos, y eventos, que suceden en la misma están bajo el control total del terapeuta.

La realidad virtual, aplicada a la terapia psicológica, ofrece un control total sobre los eventos que ocurren en el mundo virtual, por lo que podemos asegurar al paciente que todo cuanto ocurra en dicho entorno será lo que nosotros queramos que ocurra. Con esta idea se pretende que el paciente explore y experimente sin consecuencias directas gracias a que se encuentra en un ambiente protegido. La finalidad de dicha tarea es que el paciente pueda aplicar en la realidad las destrezas aprendidas en el ambiente protegido [22].

- Para el tratamiento del estrés postraumático, el grupo de la Dra. C. Botella, en la Universidad Jaime I de Castellón, desarrolló un sistema de realidad virtual, *EMMA*, el cual estaba orientado a adultos y su fin era ayudar a superar al paciente este trastorno sin necesidad de realizar la terapia cognitivo-conductual, la cual es una técnica de exposición considerada altamente aversiva y dolorosa. Actualmente hay un sistema derivado de *EMMA* denominado *EMMA-Infancia*, el cual fue desarrollado con la intención de ayudar a menores a superar los trastornos derivados del estrés postraumático al que hayan estado sometidos [23].
- Estudios clínicos han demostrado que pacientes con esquizofrenia han experimentado una mejora de los dominios cognitivos al incluir en su tratamiento varias sesiones de un juego basado en realidad virtual. Los resultados mostraron que los pacientes habían sufrido una leve mejora en su velocidad de procesamiento, su memoria de trabajo, la capacidad de estar alerta, mejoraron su aprendizaje verbal e incrementaron su capacidad de resolución de problemas [24].
- Para el tratamiento de la agorafobia también se ha hecho uso de herramientas de realidad virtual. Los pacientes expuestos a un entorno virtual generado específicamente para un escenario fóbico han demostrado beneficiarse en mayor medida que aquellos pacientes que han sido tratados exclusivamente con psicofármacos [25].

Un ejemplo de los estudios que respaldan la terapia psicológica podría ser el llevado a cabo por la Universidad de Valencia, España. El equipo de investigadores de la Universidad, valiéndose del Software mencionado anteriormente, EMMA, desarrolló tres módulos virtuales en los que se introducirían a sujetos con estrés

postraumático para observar cómo reaccionarían ante los distintos estímulos existentes en función de los traumas a tratar.

Para el caso concreto de un niño de 13 años que sufría de estrés postraumático se realizaron un total de 47 sesiones de una hora aproximadamente, de las cuales 19 sesiones se realizaron valiéndose del software EMMA-Infancia. Se trabajó respetando el ritmo del menor, se estableció una buena alianza terapéutica y se utilizó la RV para facilitar la expresión emocional, lo que favoreció la buena evolución del caso. En EMMA-Infancia se trabajó el reconocimiento y expresión emocional, tanto de emociones positivas como negativas, la toma de conciencia, el libro de la vida y especialmente la exposición a las situaciones traumáticas vividas: en el pasado con su familia de origen (maltrato físico, separación de sus padres y hermanos) y en el momento de la intervención (ausencia de visitas, no contacto con hermanos, abandono de compañeros y educadores del centro, problemas de convivencia con compañeros, educadores y profesores, salidas puntuales con voluntarios, expectativas con familia amiga, etc.) [23].

En cada sesión, el menor era informado acerca del tema que era necesario abordar y trabajar y se le pedía que representase en el sistema de realidad virtual lo que dicho tema significaba para él, valiéndose para ello de colores, imágenes, objetos y música.

En la siguiente tabla se recogen los resultados obtenidos del menor en cada uno de los módulos:

Tabla 3–1 Resultados de la intervención en EMMA-Infancia [23].

	Resultado
Módulo 1	-Rechaza técnica de relajación. No conoce esa sensación. No le sirve. Le cuesta respirar. -Fuerte bloqueo emocional.
Módulo 2	- Alegría: dificultad para reconocer situaciones en las que siente esta emoción. -Tristeza: evita hablar de esta emoción. -Miedo: no reconoce sentir miedo, sólo cuando vivía con familia. Ninguna persona le transmite seguridad. -Enfado: dificultad para reconocerlo, expresarlo y controlarlo. -Culpa: culpable por maltrato de padre y responsable del cuidado de los hermanos. -Afecto: pide afecto con llamadas de atención de los adultos. Protege a hermano y no recibe agradecimiento, se siente traicionado. -Empatía: no permite que nadie le ayude.
Módulo 3	-Resistencia a hablar sobre temas del pasado, especialmente cuando se trabajaron emociones negativas como la culpa y el libro de la vida. -Pesadillas sobre temas familiares.

Se realizaron tres evaluaciones: antes de la intervención, antes de la elaboración del trauma y al finalizar la intervención. Los resultados arrojaron una disminución en las puntuaciones tanto en sintomatología general evaluada a través de los educadores del centro, así como en los auto-informes del menor, en ansiedad, depresión, estrés postraumático e inadaptación. Las conclusiones obtenidas de dichos resultados fueron que el menor había sido capaz de reconocer más fácilmente sus experiencias traumáticas, que el proceso de

elaboración del trauma había mejorado notablemente y que los procesos de autocontrol y atención en el menor también mejoraron significativamente. Por tanto, queda demostrado que una herramienta clínica de técnicas de intervención cognitivo comportamentales como puede ser EMMA-Infancia (o EMMA a secas, la cual está destinada a adultos) es eficaz en el tratamiento de individuos con problemas relacionados a la gestión de emociones [23].

3.3 Experiencias en realidad virtual para la generación de emociones

La tarea principal de este trabajo fin de grado es la generación de emociones específicas en un sujeto a través de un entorno virtual. Es una tarea complicada que ha sido llevada a cabo por numerosos estudios, los cuales han arrojado como resultado que si un sujeto es introducido en un mundo virtual diseñado exclusivamente para la evocación de una emoción concreta, como podría ser la ansiedad, el terror o la relajación, el sujeto experimentará dichas emociones respectivamente.

Uno de los estudios, también llevado a cabo por la Universidad de Valencia, en conjunto con otras Universidades e instituciones, consistió en un grupo de 40 sujetos inmersos en un entorno virtual, concretamente un parque, tanto haciendo uso de gafas de realidad virtual como haciendo uso solo de un ordenador. Dicho entorno cambiaba progresivamente en función de la emoción que se estuviese evocando en el sujeto (relajación o alegría). Para construir estos dos entornos emocionales distintos se hicieron variaciones en la música, imágenes estáticas, clips de vídeo y recuerdos autobiográficos. La apariencia inicial del entorno era la misma para todos los usuarios, pero esta cambiaba en función de la condición emocional en la que estuviese inmerso el sujeto. Tras explorar el parque y todos los elementos que encontrasen en él, se les pedía que "creasen" un recuerdo similar a lo que habían experimentado y vivido al estar dentro del parque virtual. [26].

Todos los individuos fueron evaluados de manera individual y los resultados obtenidos fueron los recogidos en la siguiente tabla:

Tabla 3–2 Media y desviación estándar para medida de emociones [27].

Medidas	**Grupo alegría**				**Grupo relajación**			
	Monoscópico		**Estereoscópico**		**Monoscópico**		**Estereoscópico**	
	Media	**Desviación estándar**	**Media**	**Desviación estándar**	**Media**	**Desviación estándar**	**Media**	**Desviación estándar**
Alegría	Antes: 4.7 Después: 5.3	Antes: 0.32 Después: 0.28	Antes: 5.0 Después: 5.3	Antes: 0.32 Después: 0.28	Antes: 4.7 Después: 5.0	Antes: 0.32 Después: 0.28	Antes: 5.1 Después: 5.2	Antes: 0.32 Después: 0.28
Tristeza	Antes: 2.6 Después: 1.7	Antes: 0.39 Después: 0.27	Antes: 2.1 Después: 1.5	Antes: 0.39 Después: 0.27	Antes: 2.4 Después: 1.7	Antes: 0.39 Después: 0.27	Antes: 2.1 Después: 1.7	Antes: 0.39 Después: 0.27
Ansiedad	Antes: 2.6 Después:	Antes: 0.50 Después:	Antes: 2.0 Después:	Antes: 0.50 Después:	Antes: 2.4 Después:	Antes: 0.50 Después:	Antes: 2.6 Después:	Antes: 0.50 Después:

	1.4	0.21	1.2	0.21	1.3	0.21	1.4	0.21
Relajación	Antes: 4.6 Después: 5.5	Antes: 0.38 Después: 0.23	Antes: 5.1 Después: 5.8	Antes: 0.38 Después: 0.23	Antes: 4.6 Después: 5.9	Antes: 0.38 Después: 0.23	Antes: 4.5 Después: 5.2	Antes: 0.38 Después: 0.23
Efectos positivos	Antes: 29.5 Después: 31.8	Antes: 2.03 Después: 2.40	Antes: 29.6 Después: 31.7	Antes: 2.03 Después: 2.40	Antes: 29.1 Después: 30.6	Antes: 2.03 Después: 2.40	Antes: 33.6 Después: 34.1	Antes: 2.03 Después: 2.40
Efectos negativos	Antes: 14.5 Después: 8.1	Antes: 1.51 Después: 2.03	Antes: 12.3 Después: 8.9	Antes: 1.51 Después: 2.03	Antes: 15.3 Después: 8.7	Antes: 1.51 Después: 2.03	Antes: 13.7 Después: 9.9	Antes: 1.51 Después: 2.03

El resultado no solo arroja una mejora de las emociones con efecto positivo frente a la reducción de la presencia de las emociones con efectos negativos, sino que, además, también demostró que no había variaciones entre aquellos sujetos inmersos en un entorno virtual estereoscópico y los que estuvieron inmersos en un entorno virtual monoscópico [27].

Otro estudio llevado a cabo en el ámbito académico por la Universidad de Warwick, en el año 2018, exponía los efectos de la realidad virtual en el rendimiento y las emociones de los estudiantes [28]. Para este estudio se eligió a un grupo de 99 personas cuya edad promedio era de 19 años. Los participantes realizaron una prueba de conocimientos sobre una materia concreta antes y después del experimento. Se crearon 3 grupos en función del formato en el que los participantes aprenderían el contenido de dicha materia [28]:

- **Grupo realidad virtual:** Utilizaron unas gafas de VR HTC Vive que permitían a los estudiantes ver e interactuar con el modelo 3D al que acompañaba un texto descriptivo. Podían navegar por una habitación virtual, rotar, ampliar o destacar partes del modelo 3D y acceder a un menú virtual en el que seleccionar las diferentes partes de la célula y la información asociada. No había audio.
- **Grupo tradicional:** Se realizaron capturas de pantalla de los modelos 3D y los textos que se mostraban en las HTC Vive y se formatearon en un documento PDF. Este documento se mostraba en la pantalla de un ordenador. No había audio.
- **Grupo vídeo:** Se realizó una grabación de vídeo en 2D a partir del contenido generado en las HTC Vive, con los mismos gráficos y textos informativos. Los estudiantes podían pausar, adelantar o retroceder en el vídeo. El vídeo se mostraba en la pantalla de un ordenador. No había audio.

De este modo, los 3 grupos tendrían acceso exactamente a la misma información.

Los participantes del grupo de realidad virtual mostraron un aumento de las emociones positivas (interés, sorpresa, diversión, o euforia) respecto a los estudiantes del grupo tradicional o de vídeo. También mostraron una mayor implicación en el proceso de aprendizaje que los otros dos grupos [28].

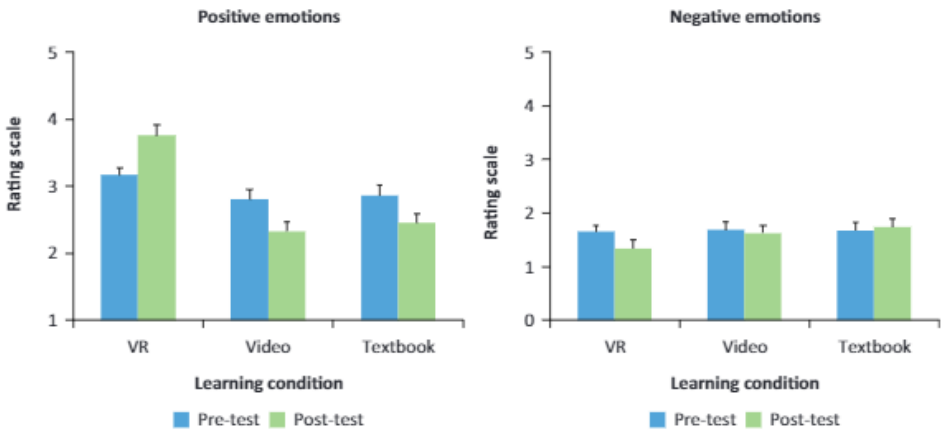

Figura 3-1. Niveles medios de emociones positivas y negativas de los participantes [28].

Tras la evaluación posterior al experimento, los resultados arrojaron que el rendimiento del grupo de realidad virtual era claramente superior al de los otros 2 grupos, siendo los porcentajes de mejora en la materia los siguientes [28]:

- **Grupo realidad virtual**: 28.5%
- **Grupo tradicional**: 24.9%
- **Grupo vídeo**: 16.1%

Un estudio llevado a cabo por el Instituto Tecnológico de Kyushu [29], en el año 2017, consistió en tomar medidas sobre la activación de emociones en un grupo de sujetos. Dichos participantes se veían expuestos a 5 tipos distintos de escenas cuyo objetivo era evocar, respectivamente, felicidad, relajación, depresión, angustia y miedo. El visor de realidad virtual Oculus Rift DK2 fue el elegido para la inmersión de los usuarios dentro del entorno 3D [29].

Tras pasar 60 segundos observando cada escena, los usuarios debían hacer un reporte acerca de las emociones experimentadas durante la visión de las mismas [29]. Posteriormente, 10 de los sujetos que habían ya participado en el experimento, volvieron a ser introducidos dentro de los entornos virtuales para registrar sus pulsaciones, su frecuencia respiratoria, la conductancia de la piel y la temperatura, es decir, para registrar las señales biológicas experimentadas dentro de los entornos.

Los resultados arrojaron que las emociones descritas por los participantes se correspondían con las reacciones biológicas que experimentaban durante el transcurso del experimento [29].

De los ejemplos expuestos es posible extraer como conclusión que es posible la evocación controlada de emociones de distintos sujetos en un entorno virtual diseñado específicamente para ello, siendo los experimentos realizados de manera *inmersiva* aquellos cuyos resultados son ligeramente más favorables, tanto para potenciar las emociones positivas como para reducir el impacto de las negativas.

4 UNITY

There are decades of innovations ahead. We're at the very beginning, where it's just at the stage where we can bring in consumers [but] there's so much further to go from there.

- Brendan Iribe -

En este apartado se expondrán las principales características de **Unity**, ya que ha sido el software empleado para el desarrollo del entorno virtual generado para este proyecto de fin de carrera.

4.1 Introducción

Unity es un *motor de videojuego*, o simplemente *motor de juego*, multiplataforma desarrollado por la empresa *Unity Technologies*.

El término *motor de videojuego* o *motor de juego* no es más que una serie de rutinas de programación que permiten llevar a cabo el diseño, creación y funcionamiento de un videojuego. Típicamente está compuesto de un motor gráfico para las renderizaciones, ya sean 2D o 3D, un motor físico que emule las leyes de la física y herramientas varias para animación, *scripting*, inteligencia artificial, reproducción de sonidos, escenarios gráficos, soporte para lenguaje por secuencia de comandos, redes, retransmisión y gestión de memoria [30].

El contenido generado puede ser exportado a todas las plataformas principales de realidad virtual, realidad aumentada, dispositivos móviles, consolas, ordenadores, etc. Además, es posible testear dicho contenido antes de ser exportado ya que Unity posee una capacidad de renderizado en tiempo real con gran fidelidad visual.

Actualmente Unity está disponible como plataforma de desarrollo para los sistemas *Microsoft Windows*, *Mac OS* y *Linux*.

4.2 Historia

La primera versión de Unity fue lanzada en el año 2005 durante la Conferencia Mundial de Desarrolladores de Apple, y aunque fue exclusivamente diseñado para funcionar en *Mac*, obtuvo el suficiente éxito como para continuar el desarrollo del motor y de sus herramientas.

En el año 2003 fue lanzada la versión *Unity 3.0*, la cual estaba centrada en la introducción de herramientas que los estudios de alta gama solían emplear, además de proporcionar paquetes asequibles con algunas limitaciones a estudios independientes.

En 2012 salió a la luz *Unity 4.0*, el cual agregó compatibilidad con DirectX11 y Adobe Flash.

Con la intención de que el desarrollo de videojuegos fuese universalmente accesible, en el año 2015 la compañía lanzó al mercado *Unity 5.0*, que ofrecía iluminación global en tiempo real, vistas previas de mapas de luz, Unity Cloud, un nuevo sistema de audio y el motor de física Nvidia PhysX 3.3. Dicho lanzamiento fue bastante polémico debido al alto volumen de videojuegos que se produjeron en un periodo muy corto por desarrolladores con nula experiencia.

En diciembre del año 2016, *Unity Technologies* anunció que cambiaría el sistema de numeración de las versiones del software, las cuales se basarían en secuencia al año de lanzamiento para la alineación del control de versiones.

4.3 Asset Store

En noviembre de 2010 se lanzó el Unity Asset Store, recurso disponible en el editor de Unity.

Asset Store es una tienda virtual en la que es posible acceder a una amplia gama de paquetes de distintas categorías, incluyendo modelos 3D, texturas y materiales, sistemas de partículas, música y efectos de sonido, tutoriales y proyectos, paquetes de scripts, extensiones para el editor y servicios en línea [31].

Algunos de estos paquetes son totalmente gratuitos, por lo que usuarios que se sumerjan en el mundo del desarrollo de videojuegos por primera vez pueden probar y experimentar con dichos paquetes de forma totalmente legal, realizando modificaciones sobre los mismos o manteniéndolos tales como sus desarrolladores los diseñaron.

La totalidad de los paquetes utilizados para este proyecto de fin de carrera han sido obtenidos en la Asset Store de forma gratuita.

4.4 Licencias

Las principales licencias que podemos encontrar en la web de Unity son:

- **Unity Personal:** ofrece todas las prestaciones del motor con la única restricción de compilar con una pantalla al inicio del ejecutable con el icono y la leyenda "Made with Unity". Además, si los proyectos exportados a ejecutables generan unos ingresos de 100 mil dólares, será obligatorio adquirir una licencia de pago.
- **Unity Plus:** enfocado a desarrolladores móviles. Si los proyectos exportados a ejecutables generan unos ingresos de 200 mil dólares, será obligatorio adquirir una licencia de **Pro** o **Enterprise**. El precio de suscripción mensual es de 35 dólares y conlleva un periodo determinado de compromiso (usualmente 1 año).
- **Unity Pro:** cuenta con todas las herramientas de Unity. Sin tope de ingresos, acceso a todos los servicios de Unity y hasta 200 usuarios simultáneos con Unity Multiplayer. El precio es de 125 dólares al mes.
- **Unity Enterprise:** es necesario de una empresa de al menos 10 usuarios para contratar esta licencia.

El precio es de 200 dólares al mes por usuario.

Características principales

4.5

Unity puede usarse en conjunto con algunos programas, como podrían ser *Blender*, *3ds Max*, *Maya, Softimage, ZBrush, Cinema4D, Cheetah3D, Adobe Photoshop y Adobe Firework*. Estos programas permiten crear objetos que pueden ser exportados para su uso en Unity, y, además, los cambios realizados a dichos objetos dentro del entorno de Unity actualizarán todas las instancias del objeto sin necesidad de que vuelva a ser importado manualmente.

Unity utiliza distintos *motores gráficos* en función del sistema operativo en el que se esté ejecutando. El más usado es *OpenGL*, ya que es válido en Windows, Mac y Linux, pero también podemos encontrar exclusivo para Windows el motor *Direct3D* o para Android e iOS el motor *OpenGL ES*. Todos tienen soporte para mapeado de relieve, mapeado de reflejos, mapeado por paralaje, oclusión ambiental en espacio de pantalla, sombras dinámicas utilizando mapas de sombras, render a textura y efectos de post-procesamiento de pantalla completa.

Posee soporte integrado para *Nvidia*, el motor de física *PhysX*. Además, Unity puede detectar la tarjeta de vídeo en uso para adaptar algunas características del programa para una mayor compatibilidad. Un ejemplo en el que ocurre esto es en el empleo de sombras o *Shaders*.

El desarrollo de scripts se realiza mediante *Mono*, que no es más que una implementación en código abierto de *.NET Framework*. A partir de la versión *Unity 3.0* se añadió una personalización de *MonoDevelop* que permitía la depuración de scripts. Actualmente, el principal lenguaje utilizado para el desarrollo de scripts en Unity es *C#*.

4.6 Iniciación a Unity

Tal y como hemos descrito en el apartado de licencias, Unity es un software de uso gratuito siempre que el objetivo del contenido generado no tenga fines comerciales. Para descargarlo, basta con visitar la web de Unity y descargar la aplicación Unity Hub.

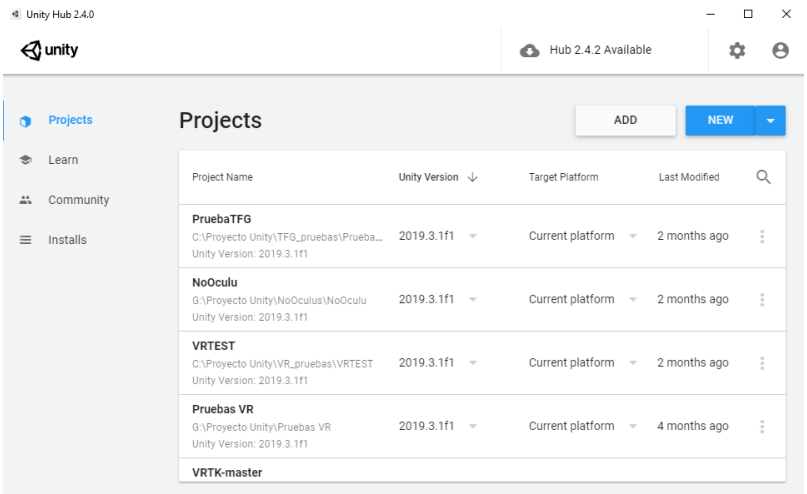

Figura 4-1. Hub de Unity.

Desde el Hub de Unity tendremos acceso a los proyectos, actualizaciones, módulos y versiones de Unity con las que deseemos trabajar y ajustes básicos, como el idioma de la interfaz o la ruta del equipo en la que se almacenarán los proyectos.

Para crear un proyecto desde la interfaz de Unity Hub nos situaremos en la pestaña *Projects* y pulsaremos el botón *New*. Se abrirá una ventana en la cual podremos seleccionar el tipo de proyecto que queremos desarrollar, el nombre que le otorgaremos y la carpeta en la que deseamos crear el nuevo proyecto.

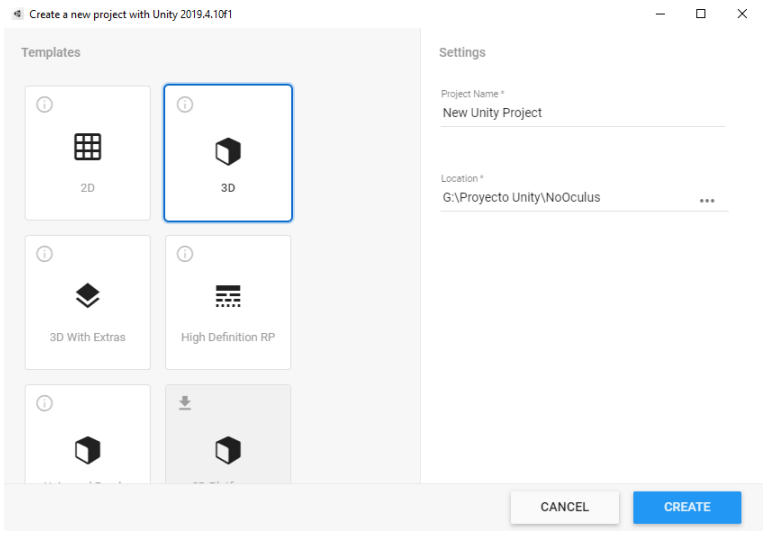

Figura 4-2. Creación de proyecto.

La pestaña *Learn* ofrece una amplia gama de proyectos y tutoriales, los cuales se pueden descargar para ampliar conocimientos acerca de la herramienta o simplemente apreciar el trabajo de otros usuarios/empresas.

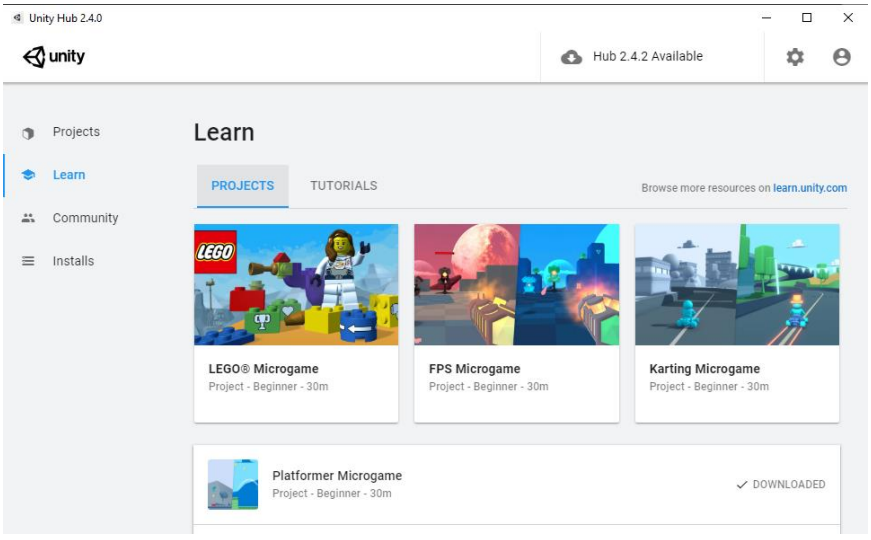

Figura 4-3. Tutoriales y proyectos.

En la pestaña *Installs* podremos encontrar las distintas versiones del software Unity instaladas en el equipo. Además, ofrece la posibilidad de obtener las versiones más recientes del software Unity, recordándote en cada acceso a la herramienta si hay una nueva versión disponible para su descarga.

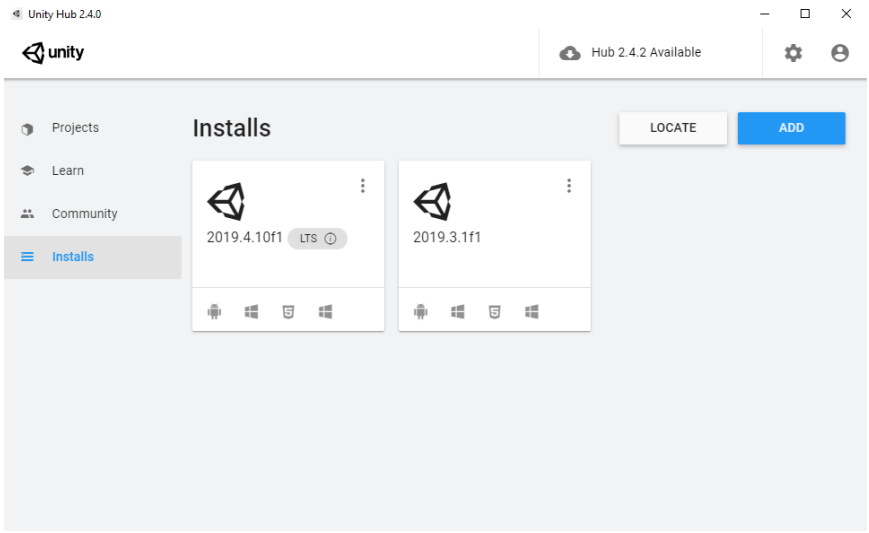

Figura 4-4. Versiones de Unity instaladas.

Por último, en la pestaña *Community*, aún en fase Beta, encontramos distintos foros y blogs en los que es posible recibir ayuda en directo, hacer preguntas a otros usuarios o resolver dudas de los mismos.

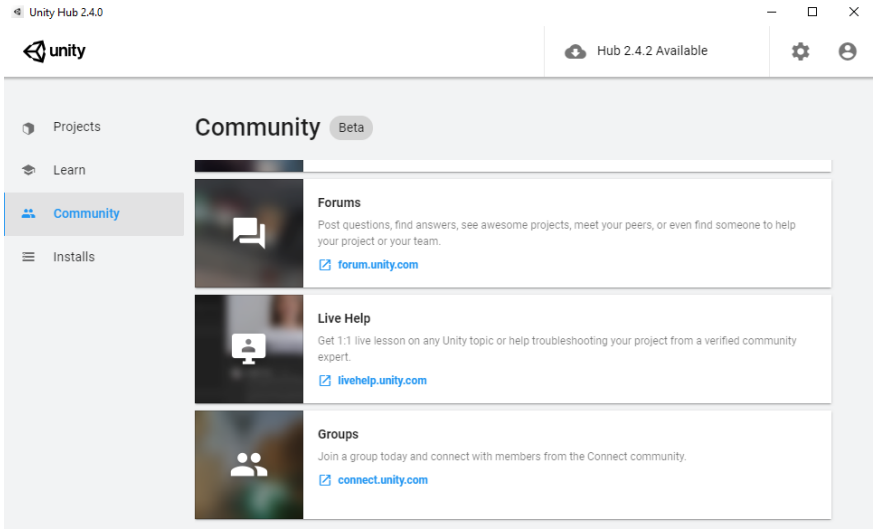

Figura 4-5. Blogs y foros de Unity.

Una vez hemos explicado las características del Hub de Unity y cómo crear un proyecto, vamos a proceder a explicar los distintos elementos que componen la interfaz visual dentro de un proyecto en Unity.

4.7 Entorno gráfico

El aspecto que presenta la ventana del entorno gráfico de Unity es el de la siguiente imagen:

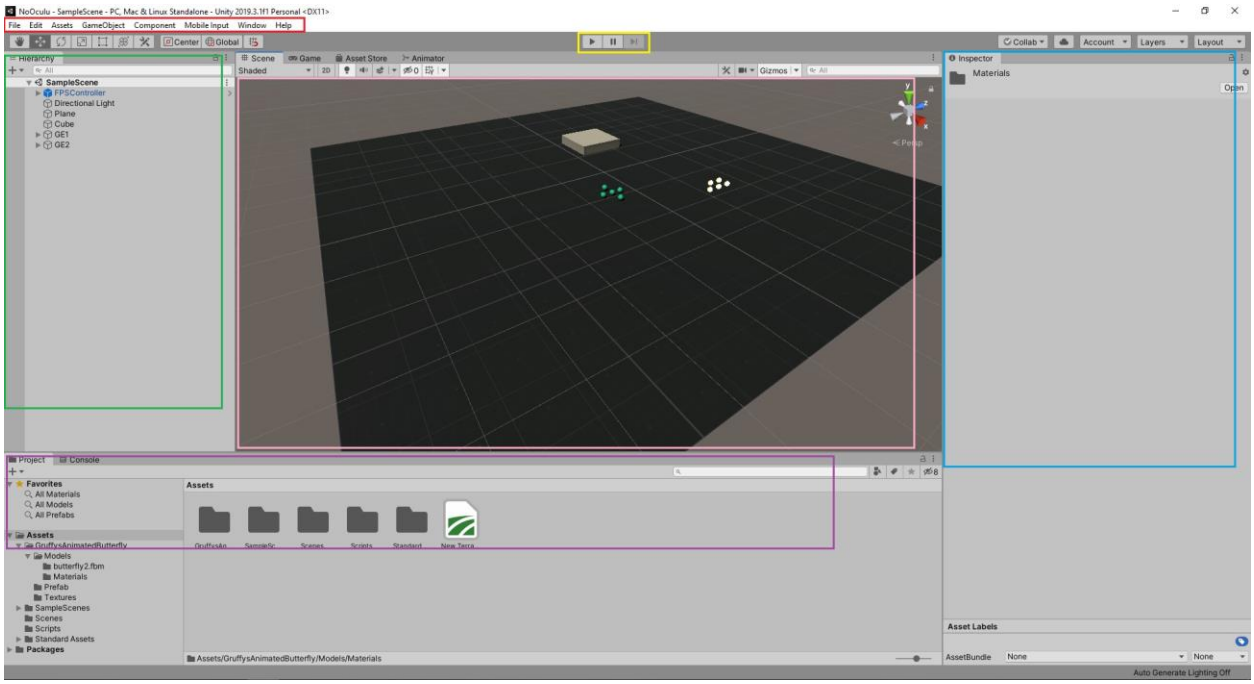

Figura 4-6. Entorno de Unity.

Como se puede observar en la imagen, he decidido realizar la explicación sobre un proyecto ya creado el cual contiene un plano, varias esferas y un cubo. La intención de esto es poder observar con mayor claridad el verdadero aspecto que tendría un proyecto "real". Además, para facilitar la explicación, separaremos las distintas partes de la ventana (en la imagen los distintos rectángulos de colores) y trataremos cada una de ellas individualmente.

4.7.1 Barra de menú

Se corresponde con la zona recuadrada en rojo de la figura 4.6. Procedemos a explicarlas en el orden que aparecen:

- **File.** Desde esta ventana desplegable podremos crear, abrir o guardar escenas y proyectos. También podemos encontrar aquí todas las opciones disponibles para exportar el proyecto generado.
- **Edit.** En esta ventana desplegable encontramos opciones de edición, como seleccionar todos los elementos de la escena, duplicarlos, renombrarlos, copiar y pegar, etc. También podremos fijar aquí las características del proyecto.
- **Assets.** Cada objeto importado en la escena genera una carpeta y desde esta ventana tenemos varias opciones para poder tratar con las mismas.
- **GameObject.** Lista desplegable de opciones para la creación de objetos en 2D, 3D, cámaras, luces, etc.
- **Component.** Lista desplegable de modificadores que pueden aplicarse a los distintos objetos integrados en la escena del proyecto.
- **Mobile Input.** Esta ventana solo muestra las opciones de *Enable* o *Disable*. Permanecerá en *Disable* a menos que el dispositivo con el que se interactúe con el entorno sea un teléfono móvil.
- **Window.** Permite modificar el aspecto de la pantalla de inicio y características de la luz y

renderizado.

- **Help.** Ayuda acerca del uso del programa, de la licencia, reporte de bugs, etc.

4.7.2 Explorador de archivos

Se corresponde con la zona recuadrada en morado en la figura 4.6. Aquí podremos encontrar todos los Assets que hayamos importado al proyecto, estén incluidos o no en la escena. También será el lugar desde el que podamos acceder a la carpeta en la que almacenemos los distintos *scripts* para modificar el comportamiento de los objetos y generar eventos, además de los distintos materiales que hayamos creado o importado.

Además, en esta zona también encontraremos la consola de depuración y la línea temporal de animaciones, desde la cual tendremos la posibilidad de realizar modificaciones a placer.

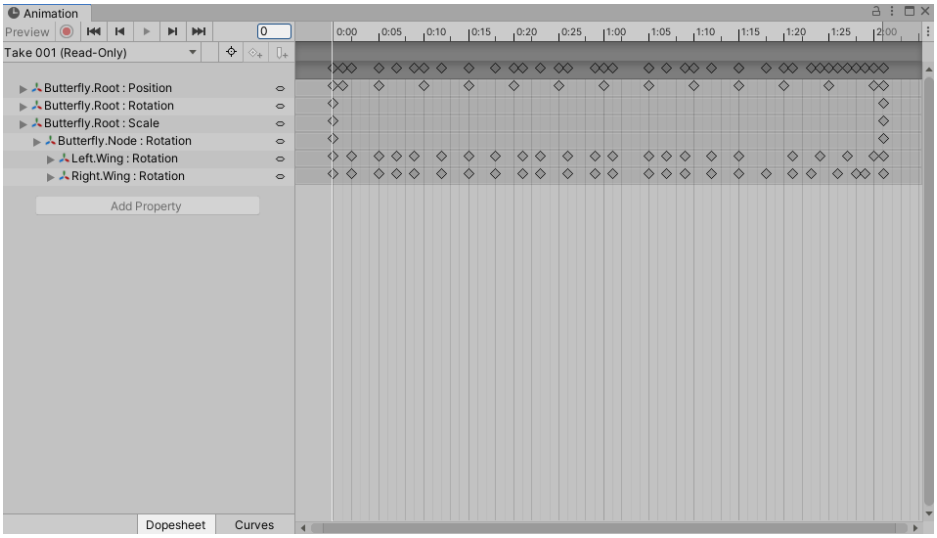

Figura 4-7. Línea temporal de animaciones.

4.7.3 Jerarquía

Se corresponde con la zona recuadrada en verde en la figura 4.6. En esta sección tendremos acceso a todos los objetos incluidos en la escena del proyecto. Desde aquí podremos crear nuevos objetos, duplicar existentes, renombrarlos, crear dependencia entre los mismos, etc.

4.7.4 Inspector

Se corresponde con la zona recuadrada en azul en la figura 4.6. Una vez hayamos creado un objeto o seleccionemos uno ya existente, aparecerán en esta sección todos los parámetros que constituyen al objeto. Además, existe la opción de añadir nuevos atributos a objetos ya creados para modificar sus características, por ejemplo, la apariencia, mediante el uso de materiales, o su comportamiento, mediante *scripts* u otros atributos que harían que el objeto esté afectado por leyes físicas.

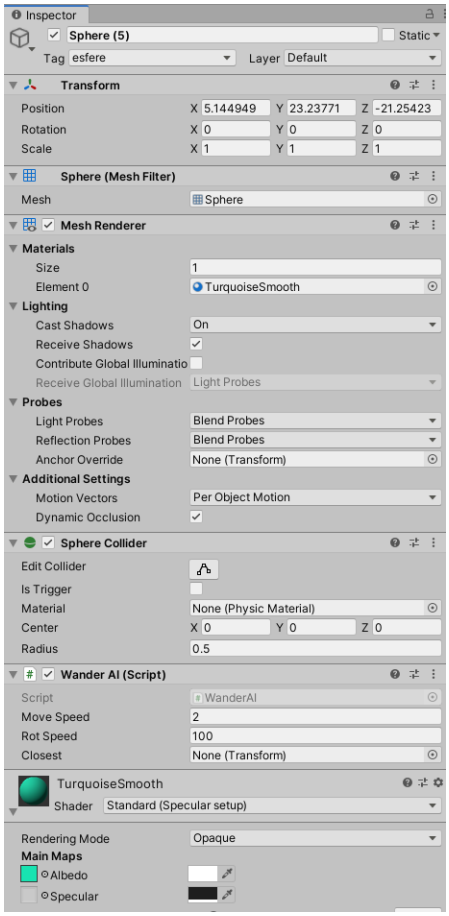

Figura 4-8. Ejemplo de parámetros de una esfera.

4.7.5 Escena

Se corresponde con la zona recuadrada en rosa de la figura 4.6. La escena o *view port* es donde se representa gráficamente el resultado del entorno creado. Por defecto, el entorno se generará con una cámara y un sol como fuente de luz de la escena. Es posible modificar la dirección y posición de ambos, aunque carece de sentido modificar la dirección y posición del sol pues ilumina en todas direcciones, no un punto fijo concreto.

Además, justo en la parte superior del *view port* encontramos 4 pestañas las cuales tienen distintas funciones:

- **Scene.** Es la escena por defecto. Desde aquí podremos seleccionar y modificar los elementos de la escena y todo el entorno generado.
- **Game.** Nos muestra la vista de la cámara, es decir, lo que veremos en el momento de iniciar la simulación.
- **Asset Store.** Nos lleva a la tienda de assets de Unity.
- **Animator.** En esta pestaña podremos generar las animaciones que deseemos para nuestra escena, además de modificar el comportamiento de animaciones previamente generadas.

4.7.6 Controles de simulación

Se corresponde con la zona recuadrada en amarillo de la figura 4.6. Cuenta con 3 botones que permiten iniciar, pausar y detener la simulación. Al iniciar la simulación (botón *Play*), el entorno nos dirigirá a la ventana *Game* (definida anteriormente) y podremos interactuar con el entorno generado mediante las herramientas que hayamos definido para ello. En el caso de este proyecto, la interacción con el entorno se realiza a través de las gafas de realidad virtual Oculus Rift S.

4.8 Adecuación del proyecto a la realidad virtual

Tras haber creado un proyecto en Unity según el proceso explicado anteriormente, tendremos que configurarlo de tal manera que sea posible su uso mediante un sistema de realidad virtual.

Para ello, en la *barra de menú*, dentro de *Edit*, seleccionamos *Project Settings*. Se abrirá una ventana para configurar los distintos parámetros del proyecto. Nos dirigiremos a *Player* y marcaremos la opción *Virtual Reality Supported*, dentro de *XR Settings*.

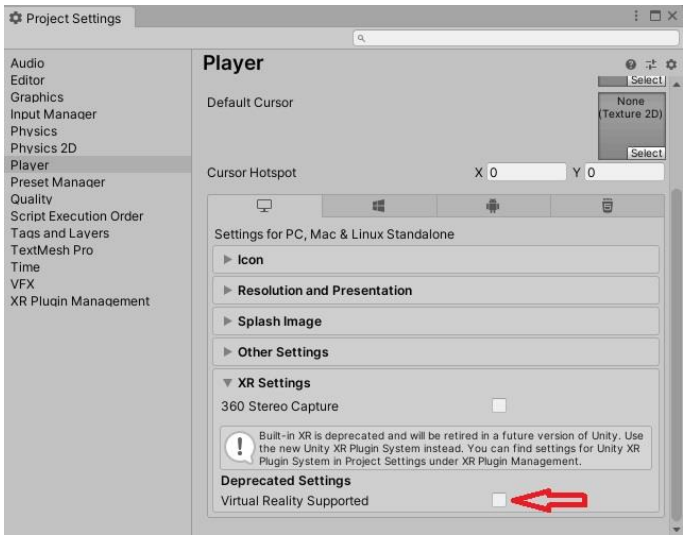

Figura 4-9. Parámetros del proyecto.

Una vez ha sido implementada la realidad virtual dentro de las características del proyecto, tendremos que asignar el SDK, en este caso Oculus, para que permita el uso de las gafas de realidad virtual a la hora de simular el entorno. Para ello, configuraremos los parámetros que se han desplegado dentro de *XR Settings* tras haber marcado la opción anteriormente mencionada.

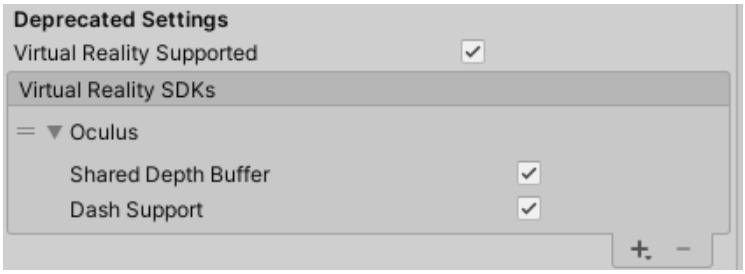

Figura 4-10. SDK Oculus.

Por último, será necesario descargar e importar un *asset* para obtener las funciones prediseñadas de Oculus para su uso en Unity. Estas funciones permitirán al usuario poder agarrar objetos con los periféricos, establecer una altura para el avatar en función de la propia altura del individuo, tener el control de la cámara a través de la detección de los movimientos de la cabeza, etc. Dicho *asset* lo obtendremos de la *Asset Store*, como ya explicaremos más adelante.

5 3DS MAX

The screen is a window through which one sees a virtual world. The challenge is to make that world look real, act real, sound real, feel real.

- Ivan Sutherland -

Este capítulo estará dedicado al uso del software **3ds Max** para la integración de objetos creados mediante el uso de las herramientas que ofrece este programa y su posterior integración en Unity. Para este proyecto de fin de carrera, los objetos que han sido diseñados a través de este software son los únicos objetos dentro del entorno virtual que son directamente interactuables a través de las "manos" del usuario.

5.1 Introducción

En la asignatura "Técnicas de Animación 3D", impartida por la profesora Irene García Fondón, se forma a los alumnos para la creación y animación de objetos 3D mediante el uso de las diversas herramientas que ofrece el software de Autodesk, **3ds Max**. Al igual que pasa con Unity, el software de Autodesk, 3ds Max, es gratuito siempre que su uso no tenga fines comerciales.

Aunque es cierto que Unity cuenta con cientos de paquetes de gran calidad y gratuitos para descargar y usar por parte del usuario, es casi una obligación el reflejar todas las horas dedicadas al aprendizaje de esta herramienta en una de las asignaturas más innovadoras e interesantes de esta carrera.

El software 3ds Max permite modelar, crear materiales, asignarlos a objetos, animar dichos objetos y finalmente renderizarlos. Es una herramienta indispensable para la creación de elementos en 3D, sobre los cuales el artista tendrá total control.

5.2 Iniciación a 3ds Max

Tal y como hemos mencionado anteriormente, 3ds Max es un software de uso gratuito siempre que el objetivo del contenido generado no tenga fines comerciales. Para descargarlo, basta con visitar la web de Autodesk y descargar la aplicación en la versión que se encuentre. Para este trabajo de proyecto de fin de grado ha sido empleada la versión 3ds Max 2019.

Tal y como hemos hecho para explicar el entorno de Unity, mostraremos la ventana del entorno gráfico principal de 3ds Max y la dividiremos en secciones para facilitar su explicación.

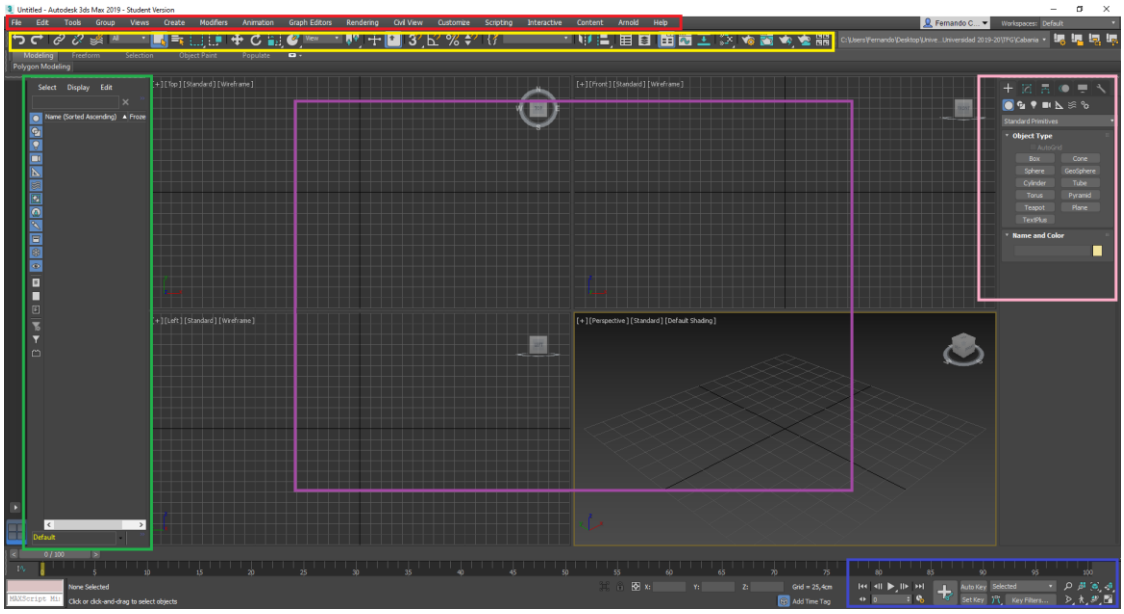

Figura 5-1. Entorno de 3ds Max.

5.2.1 Barra de menú

Se corresponde con la zona recuadrada en rojo de la figura 5.1. Procedemos a explicarlas en el orden que aparecen:

- **File.** Desde esta ventana desplegable podremos crear, abrir o guardar escenas y proyectos. También podemos encontrar aquí todas las opciones disponibles para exportar el proyecto generado.
- **Edit.** En esta ventana desplegable encontramos opciones de edición, como seleccionar todos los elementos de la escena, duplicarlos, renombrarlos, copiar y pegar, etc.
- **Tools.** Herramientas básicas para la modificación de los objetos en la escena.
- **Group.** Herramienta que permite agrupar varios objetos de la escena en un único objeto. Es útil para agrupar elementos que tienen características iguales o similares, por ejemplo, agrupar todos los árboles de la escena en un solo objeto denominado “bosque”.
- **View.** Ofrece distintas opciones para la modificación del *view port*.
- **Create.** En esta ventana desplegable encontramos una lista muy intuitiva para la creación de cualquier tipo de objeto disponible en 3ds Max, desde los más simples, como un plano, a los más complejos, como son los líquidos.
- **Modifiers.** Aquí se encuentran los distintos modificadores disponibles para la alteración de los objetos creados.
- **Animations.** En esta ventana se encuentran todas las opciones disponibles para la edición y animación de los elementos incorporados en la escena.
- **Graph Editors.** Distintas opciones para el acceso gráfico a la edición de la animación y objetos.

- **Rendering.** En esta ventana se encuentran todas las opciones disponibles para la renderización, ya sea imagen o vídeo, de los elementos de la escena. Es posible elegir la vista del *view port* que desea renderizarse y el tipo de renderizado deseado (*Arnold* ha sido el más empleado por los estudiantes de la asignatura, debido a su calidad respecto al renderizado *Scanline Renderer*). Además, ofrece la posibilidad de configurar los parámetros del tipo de renderizado que empleemos, lo cual se traduce en que es posible adaptar la calidad del renderizado en función de los recursos que pueden consumirse.
- **Civil view.** Es una vista civil, empleada en otro tipo de proyectos.
- **Customize.** Aquí encontramos herramientas para la personalización de la interfaz del usuario de 3ds Max.
- **Scripting.** Contiene la herramienta para la creación de *scripts*. En 3ds Max los *scripts* suelen emplearse para automatizar o acelerar distintas funciones, como construir objetos de una manera determinada, influenciar en los comportamientos de una animación, etc.
- **Interactive.** Esta pestaña nos dirige a la web de Autodesk. Sirve para descargar un motor de juego usado para la creación de contenido en realidad virtual.
- **Content.** Aquí se encuentra la librería de 3ds Max.
- **Arnold.** Nos aparece en la barra de menú debido a que el renderizado que empleamos es de tipo Arnold. Podemos encontrar información acerca de este tipo de render.
- **Help.** Ayuda acerca del uso del programa, de la licencia, reporte de bugs, etc.

5.2.2 Accesos rápidos

Se corresponde con la zona recuadrada en amarillo de la figura 5.1. En esta sección se encuentran accesos rápidos para hacer uso de las distintas herramientas del programa. Es posible configurarla desde la pestaña **Customize** explicada anteriormente. Entre los más usados se encuentran aquellos que permiten mover el objeto, rotarlo, escalar su tamaño, deshacer y rehacer acciones, etc.

5.2.3 Jerarquía

Se corresponde con la zona recuadrada en verde en la figura 5.1. En esta sección tendremos acceso a todos los objetos incluidos en la escena del proyecto. Nos facilitará la selección de objetos en la escena, identificación de los mismos, agrupar varios objetos en un mismo grupo, ocultar/mostrar elementos de la escena a placer, etc.

5.2.4 Escena

Se corresponde con la zona recuadrada en morado en la figura 5.1. La escena o *view port* es donde se representa gráficamente el resultado del entorno creado desde diferentes perspectivas. Podemos configurar dichas perspectivas a voluntad, siendo alzado, planta y perfil las que nos aparecen por defecto, además de una vista 3D de la escena. En la esquina superior izquierda de cada una de las vistas encontraremos, entre corchetes, las distintas opciones disponibles para la personalización de las mismas.

En la esquina superior derecha de cada vista aparece un cubo con los puntos cardinales, el cual podremos usar para rotar el ángulo de visión de la vista en cuestión, aunque no es una herramienta demasiado útil, ya que podremos lograr el mismo resultado, de manera mucho más cómoda y eficiente, manteniendo la tecla *Alt* y el botón central del ratón sobre la vista que queremos rotar.

5.2.5 Panel de creación de objetos

Se corresponde con la zona recuadrada en rosa en la figura 5.1. En esta sección encontramos un menú con los distintos objetos que podemos crear en 3ds Max y los modificadores que pueden aplicarse al objeto una vez creado. También nos permite crear otro tipo de elementos como luces o cámaras para la renderización. Su uso, aunque no tan intuitivo como el de la pestaña ***Create***, hace más dinámica la creación y modificación de los objetos.

5.2.6 Controles de animación

Se corresponde con la zona recuadrada en azul en la figura 5.1. En esta sección encontramos una línea temporal y varios botones que permiten interactuar con la animación. En la línea temporal se reflejarán las distintas animaciones mediante cuadrados de distintos colores, en función de la modificación realizada a la escena en dicho fotograma (escalado, desplazamiento, rotación, etc). Los botones nos permitirán fijar la escala usada en la línea temporal, avanzar fotograma a fotograma, avanzar de forma continua, detener la animación, etc.

5.3 Materiales

Para la creación de materiales, nos situaremos en cualquiera de las vistas del *view port* y presionaremos la tecla *M*. Se abrirá una ventana como la indicada en la figura siguiente.

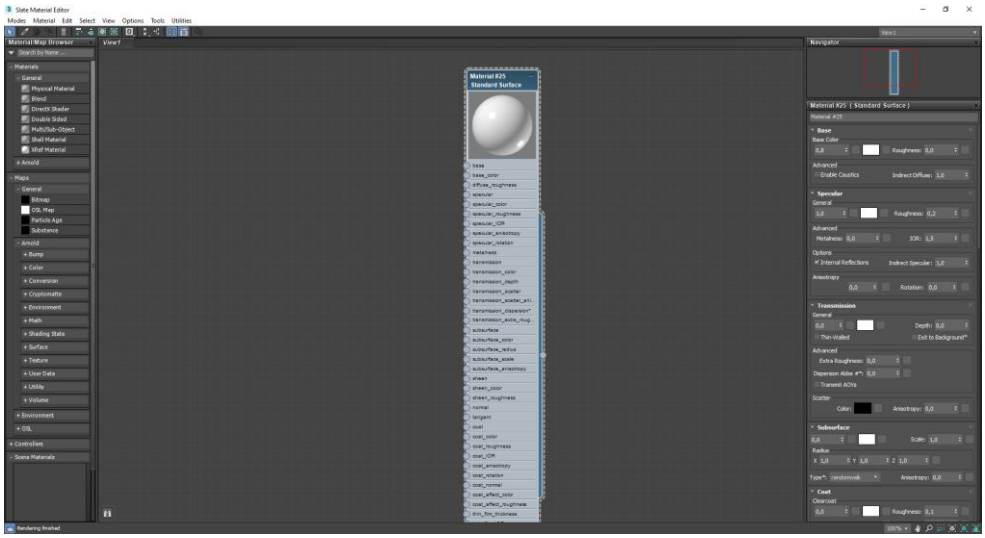

Figura 5-2. Ventana de materiales.

En el lateral derecho de la figura 5.2 podremos observar un panel con los distintos parámetros que componen el material básico creado como ejemplo, el cual se encuentra en el centro de la imagen. Mediante la alteración de los valores de dichos parámetros podremos modificar el aspecto del material que se está editando.

Es posible visualizar el resultado en la esfera, blanca para este ejemplo, situada en el centro de la imagen. Además, es posible añadir al material imágenes o *mapas*, los cuales otorgarían al material el aspecto de la textura de la imagen que se esté asociando.

Una vez creado el material, seleccionaremos en el *view port* el objeto al que queramos asociarlo y aplicamos el material a la selección actual.

Para este proyecto de fin de grado no se ha creado ningún material en 3ds Max, pues no son exportables a Unity, ya que este usa sus propios materiales.

5.4 Resultado

Una vez haya finalizado el proceso de creación de un objeto en 3ds Max, se exportará el modelo en formato *fbx* para su posterior integración en Unity. Para ello, seleccionamos *File->Export->Export…* en la barra del menú y elegimos el formato *fbx*.

Los objetos creados para este proyecto, de los cuales hablaremos más adelante, son del tipo *Low Poly*, es decir, tienen un número relativamente pequeño de polígonos y eso hace que su calidad visual no sea demasiado buena, pero son ideales para aplicaciones en las que estos objetos intervienen en "tiempo real" ya que sobrecargan menos la GPU, razón por la cual se ha elegido su diseño para la posterior integración en el entorno virtual.

6 OCULUS RIFT S

I really do think VR is now one of the most exciting things that can be done in this whole sector of consumer electronic entertainment stuff.

- John Carmack -

El uso de unas gafas de realidad virtual, en el caso de este proyecto de fin de carrera el modelo *Rift S* de Oculus, acerca al usuario a experimentar de una manera más vívida el entorno virtual en el que se ve sumergido. Se expondrá a lo largo de este capítulo el mantenimiento de las Oculus, los requisitos, los elementos de los que está compuesto este sistema de realidad virtual y el software necesario para su uso mediante un ordenador.

6.1 Componentes

A continuación, se muestran los distintos elementos que forman parte de la totalidad del visor de realidad virtual Oculus Rift S:

- **Diadema:** cinta conectada al visor de realidad virtual. Permite mantener sujeta y firmes las gafas en la cabeza.
- **Visor:** sistema a través del cual se reproducirán las imágenes creadas por ordenador.
- **Interfaz facial:** goma transpirable y acolchada para una mayor comodidad.
- **Cable óptico:** cable de 5 metros que conecta las gafas a un Puerto USB 3.0 del ordenador.
- **Controladores Touch:** son los mandos que simulan ser las manos del usuario. Permiten la interacción del usuario con el mundo virtual.
- **Lentes graduadas (opcional):** personalizables y ligeras, con filtro antibrillo de alta calidad.

Figura 6-1. Casco de realidad virtual. [32]

Figura 6-2. Controladores Touch. [33]

Figura 6-3. Diadema. [34]

Figura 6-4. Interfaz facial. [35]

Figura 6-5. Cable Óptico. [36]

Figura 6-6. Lentes graduables. [37]

Esta información ha sido obtenida a través de la web oficial de Oculus [38].

6.2 Requisitos

A continuación, se muestran los requisitos mínimos y los recomendados del ordenador necesario para poder hacer uso de estas gafas:

Tabla 6–1 Requisitos Oculus Rift S

	Requisitos Mínimos	**Requisitos Recomendados**
Tarjeta Gráfica	NVIDIA GTX 1050 Ti/AMD Radeon RX 470 o superior	NVIDIA GTX 1060/AMD Radeon RX 480 o superior
Tarjeta Gráfica Alternativa	NVIDIA GTX 960/AMD Radeon R9 290 o superior	NVIDIA GTX 970/AMD Radeon R9 290 o superior
CPU	Intel i3-6100/AMD Ryzen 3 1200, FX4350 o superior	Intel i5-4590/AMD Ryzen 5 1500X o superior
Memoria	8 GB de RAM o más	8 GB de RAM o más
Salida de video	DisplayPortTM 1.2 / Mini DisplayPort (con adaptador incluido en la caja)	DisplayPortTM 1.2 / Mini DisplayPort (con adaptador incluido en la caja)
Puertos USB	1 puerto USB 3.0	1 puerto USB 3.0
Sistema operativo	Windows 10	Windows 10

Esta información ha sido obtenida a través de la web oficial de Oculus [38].

6.3 Medidas de seguridad

6.3.1 Obligatorias

Antes de comenzar a usar el casco de realidad virtual es necesario preparar el entorno que se encuentra en los alrededores. Es necesario cerciorarse de que no hay ningún objeto (silla, mesa,…) con el que se pueda impactar o resbalar durante el uso de las gafas.

Es aconsejable garantizarse 1 metro de radio despejado de objetos en los alrededores, no obstante, mientras más espacio libre haya, mejor [39].

6.3.2 Recomendables

Configuración del "sistema guardián": es posible crear una barrera que aparecerá en la realidad virtual para delimitar el área de juego previamente establecida. No obstante, esta barrera es solo una alerta, ya que el sistema no puede evitar que el usuario salga de ella.

También es posible establecer una restricción del contenido al que no se desea acceder, ya sea porque dicho contenido fue clasificado por edades (IARC[1]) o porque sencillamente no sea del agrado del consumidor [39].

6.4 Controlador

Antes de su uso, es necesario comprobar que la tarjeta gráfica y los puertos USB están actualizados con los controladores en la última versión que dispone el fabricante. En la página web de Oculus podemos encontrar una serie de instrucciones sobre cómo mantener actualizados dichos controladores.

El siguiente paso será instalar la aplicación de Oculus en el ordenador que se haya dispuesto para usar junto al casco de realidad virtual. De nuevo, en la web de Oculus podemos encontrar una serie de instrucciones para descargar dicho software y los pasos a seguir para instalarlo.

6.5 Mantenimiento

Es importante cuidar los productos de Oculus y limpiarlos con frecuencia [39].

A continuación, se detallará como limpiar varios de los componentes de los que está formado Oculus Rift S:

- **Controladores Touch:** se recomienda usar un paño seco para limpiar la parte exterior del componente. Para limpiar la superficie y los botones se usarán toallitas antibacterianas no abrasivas.
- **Auriculares:** se recomienda usar un paño seco para su limpieza.
- **Visor:** se recomienda usar un paño seco para limpiar la parte exterior del componente. Para limpiar las correas y la espuma de la interfaz para el rostro se usarán toallitas antibacterianas no abrasivas.
- **Lentes del visor:** se recomienda usar un paño seco para la limpieza de las lentes. Se harán movimientos circulares desde el centro de estas hasta el exterior.
- **Sensor:** se recomienda usar un paño seco para la limpieza de las lentes. Se harán movimientos circulares desde el centro de estas hasta el exterior. Para limpiar el soporte se usará un paño seco o toallitas antibacterianas no abrasivas.

Para mantener en buen estado Oculus Rift S habrá que tener en cuenta las siguientes consideraciones:

- Evitar que quede expuesto directamente a la luz solar.
- No utilizar objetos afilados cerca del Rift S.
- No dejar el Rift S en ubicaciones con elevada temperatura.
- No dejar el Rift S cerca de mascotas o niños pequeños.
- No comer, beber ni fumar cerca del Rift S.
- Asegurar la zona en la que se va a usar para evitar posibles colisiones.

[1] International Age Rating Coalition

6.5.1 Prevenciones COVID-19

Debido a las circunstancias especiales del año en el cual se ha desarrollado este proyecto de fin de Carrera, ha sido necesario tomar precauciones adicionales durante el uso de las Oculus Rift S.

Además del mantenimiento mencionado en el apartado 6.5 se han tenido en cuenta las siguientes consideraciones:

- Uso de antifaz higiénico, o máscara de ojos desechable, para evitar el contacto de la piel de la cara y la interfaz facial.
- Desinfección con gel hidroalcohólico de las manos para el uso de los controladores *touch*.
- En ningún caso las Oculus Rift S han sido usadas por personas que puedan tener enfermedades contagiosas.
- Desinfección, en la medida de lo posible según indicaciones del fabricante, de los distintos elementos por los que están formado el sistema de realidad virtual.
- Depositar las gafas en un lugar limpio y desinfectado mientras no estén en uso.

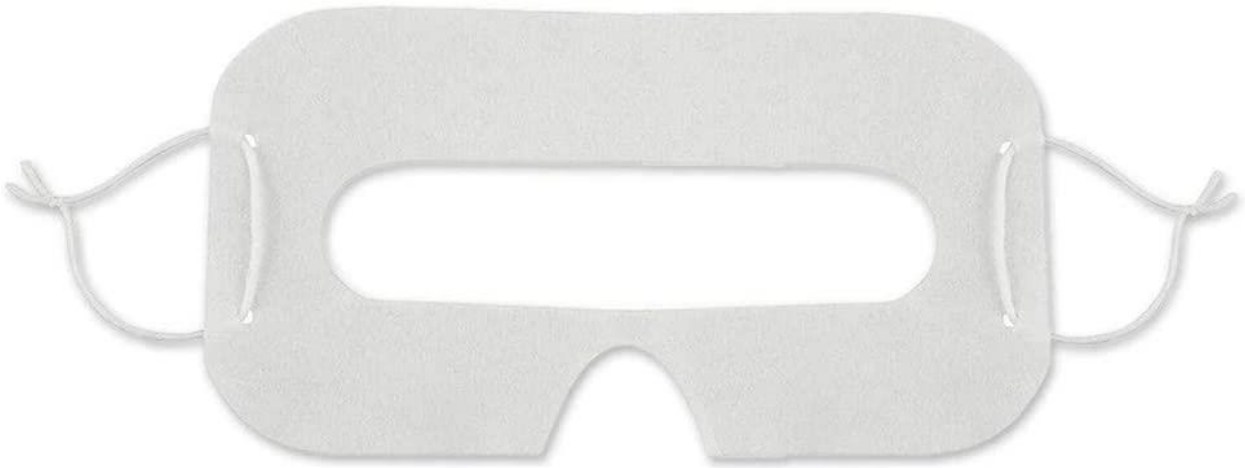

Figura 6-7. Antifaz Higiénico. [40]

7 ENTORNO GENERADO

Virtual reality is the first step in a grand adventure into the landscape of the imagination.

- Frank Biocca -

El siguiente capítulo tratará sobre la creación del entorno de realidad virtual que se ha generado para este proyecto de fin de carrera, de los pasos que se han seguido para su creación, los elementos que se incorporaron a la escena, el código empleado para las interacciones del usuario con el mundo y las futuras líneas en las que podría avanzar el proyecto.

7.1 Introducción

El entorno de realidad virtual generado mediante Unity tiene, como ya se ha mencionado varias veces a lo largo de esta memoria, la intención de evocar en el usuario la emoción secundaria de la serenidad. Para ello, el entorno virtual creado cuenta con el aspecto de un bosque situado en una isla desierta, la cual está llena de vegetación, agua, animales y paisaje montañoso, además de sonidos ambientales asociados a distintos eventos.

El objetivo de este capítulo será explicar, con la mayor fidelidad posible, el proceso de creación del entorno virtual. Ya hemos explicado cómo se crea un proyecto y se habilitan los parámetros correspondientes para que sea compatible con la realidad virtual, por lo que tomaremos como punto de partida el primero de los elementos que se integró en la escena.

7.2 Terreno y entorno

Unity trae sus propias herramientas para la creación de terrenos, no obstante, para este proyecto se ha usado el paquete *Terrain Tools Sample Asset Pack* [41], el cual incorpora nuevas funciones para la creación de terrenos, mejora las ya existentes e incluye un paquete de texturas.

Los pasos seguidos para la creación del terreno en este proyecto fueron los siguientes:

- En *Hierarchy* hacer click derecho y seleccionar *3D Object->Terrain*.
- Seleccionar el objeto creado y crear "vecindades" mediante *Create Neighbor Terrains*. Para el terreno en cuestión se crearon ocho "vecindades", siendo la primera la central.
- Dentro de *Paint Terrain->Raise or Lower Terrain* se encuentran diversas herramientas para modificar geográficamente nuestro terreno.

Para dar forma al terreno sobre el que se situarían todos los demás elementos, se deformaron las ocho "vecindades" creadas, disminuyendo su altura con respecto al terreno central y creando una conexión en forma de rampa entre estas ocho "vecindades" y dicho terreno, la cual simula ser la orilla de la playa.

En cuanto al terreno central, fue deformado con el propósito de que no fuese una llanura vacía. Para ello, se usaron herramientas de elevación irregular del terreno y se creó también un socavón para ser usado como lago.

Todas las herramientas mencionadas para la deformación del terreno se encuentran en el apartado *Brushes* dentro de *Paint Terrain->Raise or Lower Terrain*.

7.2.1 Montañas

Para la creación de las montañas que se encuentran en el terreno del proyecto se empleó una combinación de diferentes *Brushes*. Sería imposible replicar de manera exacta los pasos seguidos para la elaboración de las mismas, no obstante, se podría conseguir un resultado similar mediante el empleo de los mismos *Brushes* que se usaron para este proyecto, los cuales son: *builtin_brush_1, builtin_brush_2, builtin_brush_4, builtin_brush_6, builtin_brush_8, terrainstamp_canyon01, terrainstamp_canyon02 y terrainstamp_mountain06*.

Hay varios aspectos a tener en cuenta a la hora de usar uno de estos "pinceles":

- Al seleccionar un pincel y posicionarlo sobre el terreno, si se hace clic derecho y se arrastra el ratón, el terreno se elevará.
- En el caso de estar pulsando *Shift* y hacer clic izquierdo, al mover el ratón el terreno decrecerá.
- Se pueden combinar varios pinceles usando para ello un nuevo modelo sobre la parte del terreno en la que se actuó con el anterior.

Hay dos parámetros que pueden ser modificados en todos los *Brushes*:

- **Brush size:** determina el tamaño del área de efecto del pincel.
- **Opacity:** determina la "fuerza" con la que se aplicará el efecto del pincel.

Una vez creadas las montañas se procede a aplicar texturas para otorgarles un aspecto realista. Dentro de *Paint Terrain->Paint Texture* se encuentran las distintas texturas que incluye el paquete que se ha usado para la creación del terreno en este proyecto.

Para otorgar a las montañas el aspecto realista del que hablábamos anteriormente, se han usado las texturas *Rock* y *Snow*, "pintando" el terreno con el ratón y hacienda uso de los parámetros Brush size y Opacity para los detalles que requerían de mayor precisión.

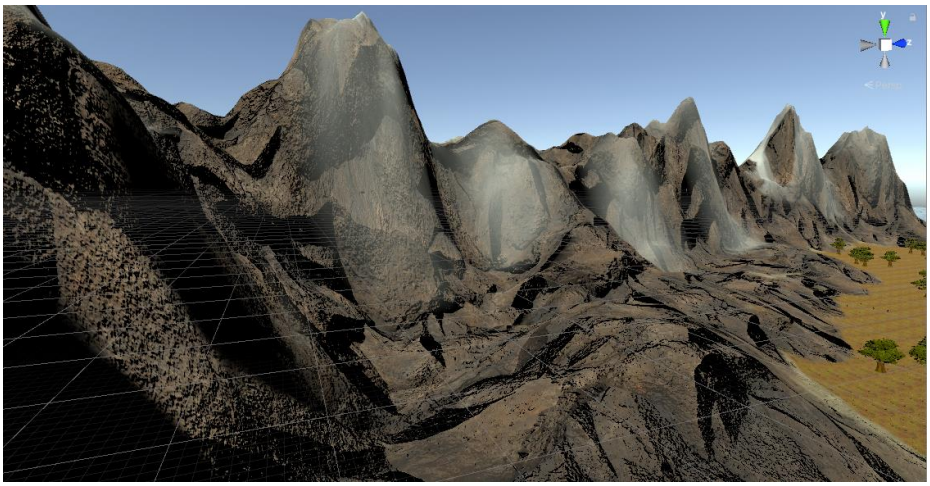

Figura 7-1. Montañas.

7.2.2 Lago

De forma similar a la creación de las montañas, la creación del lago fue un hundimiento del terreno mediante el uso del pincel *Paint Terrain->Raise or Lower Terrain ->builtin_brush_1*.

Dicho lago goza de poca profundidad, ya que es posible para el usuario acceder a su interior y sumergirse en él. Si se incrementase la profundidad, podría suponer un problema para el usuario a la hora de intentar salir de este.

Para otorgar al lago un aspecto realista se usó la textura *Paint Terrain->Paint Texture->Dirt*, también incluida entre las texturas del paquete en uso.

Figura 7-2. Lago vacío.

7.2.3 Árboles

El modelo de árbol que se ha usado en este proyecto ha sido obtenido del paquete *Fantasy Forest Environment - Free Demo* [42].

Para integrarlo en el entorno seleccionamos el terreno principal, *Paint Trees->Edit Trees…->Add Tree* y

seleccionamos el modelo que ha sido importado anteriormente, por defecto *tree_1*.

Al seleccionar el árbol añadido aparecerá un área para "dibujar" los árboles en el terreno, como si de un pincel se tratase. Los parámetros que se pueden configurar para ello son:

- **Brush size:** determina el tamaño del área de efecto del pincel.
- **Tree density:** determina la cantidad de árboles que habrá dentro del área de efecto.
- **Tree height:** determina la altura de los árboles que aparecerán. Es posible activar la casilla **Random?** para que dicha altura varíe entre los árboles que aparezcan.
- **Lock width to height:** fija el ancho del árbol del mismo tamaño que la altura.
- **Random tree rotation:** los árboles poseen distinta orientación de manera aleatoria.
- **Color variation:** solo funciona si el *Shader* es compatible. Aplica una pequeña variación de color entre los árboles creados.

La configuración de los parámetros anteriores, en especial *Tree height* y *Tree density*, fue de suma importancia debido a que si el parámetro *Tree height* era demasiado elevado, los árboles adquirían una altura irrealista con respecto al jugador, mientras que un valor elevado en el parámetro *Tree density* provocaba un efecto de superposición entre los distintos árboles que se creaban en la escena.

Figura 7-3. Árboles.

7.2.4 Flores

El paquete importado al proyecto para la obtención de los distintos modelos de flores es *Grass Flowers Pack Free* [43], el cual cuenta con doce modelos distintos de flora.

Para integrarlo en el entorno seleccionamos el terreno principal, *Paint Details->Edit Details…->Add Grass Texture* y seleccionamos los doce modelos que han sido importados anteriormente.

Al seleccionar los distintos modelos de flora añadidos aparecerá un área para "dibujarlos" sobre el terreno, como si de un pincel se tratase. Los parámetros que se pueden configurar para ello son:

- **Brush size:** determina el tamaño del área de efecto del pincel.

- **Opacity:** determina la "fuerza" con la que se aplicará el efecto del pincel.
- **Target Strenght:** determina la cantidad de flores que aparecerán bajo el área de efecto del pincel.

Para la creación de este entorno se ha hecho uso de los doce modelos importados mediante el paquete anteriormente mencionado. Aunque no es posible modificar la altura de los distintos modelos de flores, es importante ajustar los parámetros *Opacity* y *Target Strenght* para evitar superposiciones y efectos no deseados en la incorporación del conjunto floral al entorno.

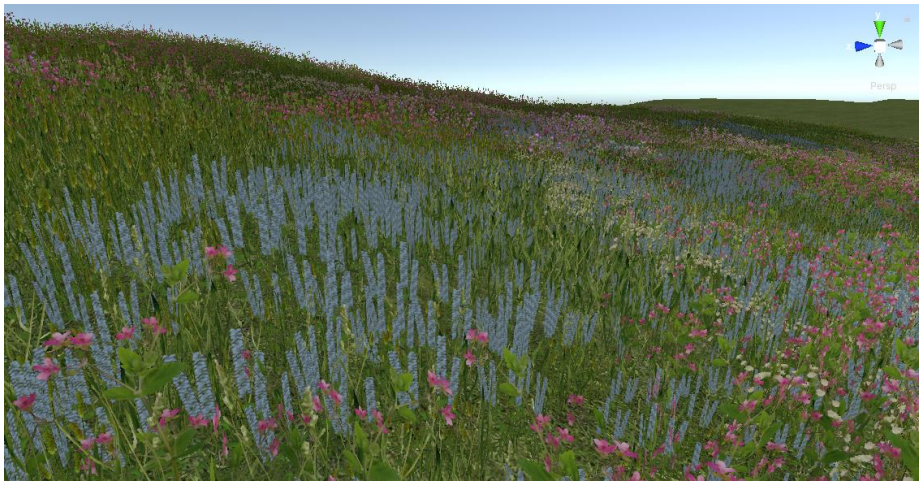

Figura 7-4. Flora.

7.2.5 Playa

Como se ha mencionado anteriormente, las ocho "vecindades" que forman la playa fueron deformadas para disminuir su altura con respecto al terreno central. Además, como también se ha mencionado ya, la conexión del terreno entre estas ocho "vecindades" y el terreno central es una rampa que simula ser la orilla del mar.

La playa es bastante extensa, especialmente la zona que queda bajo el agua. Es posible adentrarse mar adentro y recorrer la playa por el fondo, y, pese a ser bastante amplia, es posible llegar al "final del mapa", por lo que el usuario caería al vacío durante una milésima de segundo, tras la cual el entorno se reiniciaría gracias al script *RestartGame.cs*.

Por último, para otorgar a la playa un aspecto realista se usó la textura *Paint Terrain->Paint Texture->Sand*, también incluida entre las texturas del paquete *Terrain Tools Sample Asset Pack* [41].

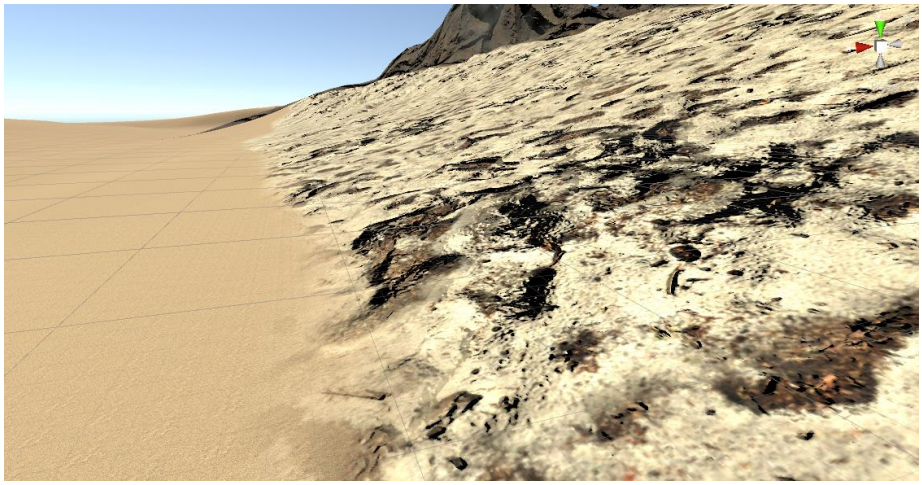

Figura 7-5. Playa.

Al resto del terreno, aquel sobre el que se han situado los árboles, flora y demás elementos, se le aplicó la textura *Paint Terrain->Paint Texture->Moss*, también incluida entre las texturas del paquete *Terrain Tools Sample Asset Pack* [41], además de ligeras distorsiones mediante el empleo de los pinceles disponibles para conferir al terreno leves irregularidades, evitando así que sea completamente plano y ofrezca mayor realismo visual.

Figura 7-6. Terreno genérico.

7.3 Animales

Para evitar que en el entorno se pudiese experimentar la sensación de soledad se han importado cuatro modelos distintos de animales, los cuales otorgan al entorno la sensación de estar "vivo". Todos ellos son gratuitos y han sido descargados e importados de la propia Asset Store de Unity.

Todos los modelos cuentan con animación y con texturas, creada por los desarrolladores de los mismos. El comportamiento de algunos de los modelos fue modificado mediante el uso de un script, de los cuales hablaremos más adelante.

7.3.1 Conejos

El modelo de conejo que se ha usado en este proyecto ha sido obtenido del paquete *White Rabbit* [44].

Una vez importado, dentro de la carpeta *Rabbits->Demo->Prefabs* se encuentra el objeto *Demo White Rabbit*, el cual puede ser arrastrado a la escena para ser integrado en el entorno.

No obstante, cuando el objeto se acaba de incorporar en la escena, es completamente estático y no está correctamente dimensionado con respecto el tamaño que un conejo real debería tener acorde a una persona. Para ello, el primer paso que damos será el redimensionado del objeto mediante *Scale*. El siguiente paso será eliminar el *script* que trae el objeto por defecto para controlar su comportamiento y añadimos el que hemos generado nosotros, *Conejo.cs*, el cual se explicará más adelante. Finalmente, para que el objeto sea completamente funcional, entraremos en *DemoAnimator* y eliminaremos todos los estados de la animación, salvo *Run*. Conectaremos *Entry* y *Any State* a *Run* mediante flechas, usando para ello el ratón.

El resultado de seguir estos pasos es un objeto *Conejo* correctamente dimensionado y capaz de desplazarse por el entorno, pero con el problema de que es capaz de atravesar otros objetos del mismo entorno. Para evitar esto añadimos el atributo *Capsule Collider* al objeto *Conejo*, activando así la colisión con el resto de los elementos de la escena. Finalmente, creamos una etiqueta, *Tag*, para cada uno de los conejos que se han incorporado al entorno, la cual será usada por *Conejo.cs* para detectar la distancia a la que se encuentran con respecto al jugador.

Figura 7-7. Conejo.

Este proyecto incluye un total de nueve conejos, todos creados con los mismos parámetros, con la única salvedad de la etiqueta, la cual es distinta para cada uno de ellos.

7.3.2 Mariposas

El modelo de mariposa que se ha usado en este proyecto ha sido obtenido del paquete *Butterfly with Animations* [45].

Una vez importado, dentro de la carpeta *GruffysAnimatedButterfly->Prefab* se encuentra el objeto *butterfly2*, el cual puede ser arrastrado a la escena para ser integrado en el entorno.

Al igual que para el objeto de tipo *Conejo*, fue necesario realizar una serie de modificaciones para su uso en el proyecto. La primera fue añadirle el script *Wander AI.cs* para modificar el comportamiento de la mariposa, el cual explicaremos más adelante. La segunda consistió en la alteración del archivo de textura *butterfly2*, situado

en la carpeta *GruffysAnimatedButterfly->Models->butterfly2.fbm*, para obtener mariposas de distintos colores.

La modificación de este archivo de imagen se realizó mediante el editor de fotos de Windows 10, aplicando distintos filtros hasta obtener cinco imágenes de distintos colores, concretamente el rojo, el verde, el amarillo, el morado y el azul. Tras obtener las cinco imágenes distintas, se copian en la carpeta *GruffysAnimatedButterfly->Textures* y se seleccionamos en *Texture Type* la opción *Sprite (2D and UI)*. Estableceremos el valor de *Max Size* en 1024 y crearemos cinco materiales del tipo *Mobile/Particles/Multiply* para asociarles las imágenes creadas mediante el editor de Windows 10. Por último, seleccionamos la mariposa a la que vamos a aplicar el material generado y en *Butterfly Mesh->Materials->Element 0* arrastramos uno de los cinco materiales que hemos creado previamente.

En este proyecto se han incluido noventa mariposas agrupadas en conjuntos de cinco, todas ellas con la etiqueta *Mariposa*.

Figura 7-8. Conjunto Mariposas.

7.3.3 Peces

El modelo de pez que se ha usado en este proyecto ha sido obtenido del paquete *#NVJOB Simple Boids (Flocks of Birds, Fish and Insects)* [46].

Una vez importado, dentro de la carpeta *#NVJOB Boids->Simple Boids->Prefabs->Fish* se encuentra el objeto *Fish 1 Pref*, el cual puede ser arrastrado a la escena para ser integrado en el entorno.

En este proyecto se han incluido nueve peces, todos ellos dentro del lago mencionado en el apartado 7.2.2. Mediante *Size* modificamos el tamaño de los peces para que sea un tamaño acorde con respecto al jugador. Cuentan todos ellos con un *Shader* por defecto, el cual les confiere, además del color base, una serie de sombras y reflejos que otorgan realismo al objeto y permite diferencias las escamas del mismo. Por último, este objeto también cuenta con el parámetro *Capsule Collider* para evitar las colisiones, tanto con el jugador como con cualquier objeto rígido que pueda estar incorporado en el entorno.

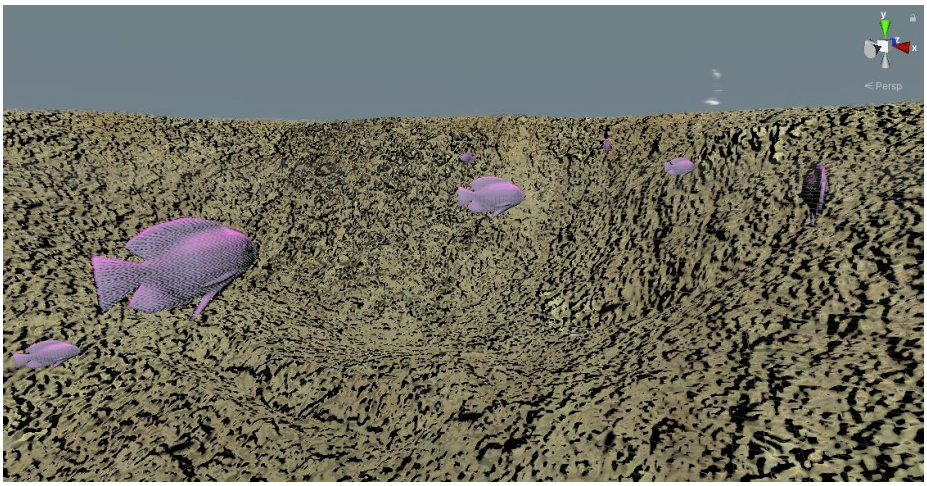

Figura 7-9. Peces.

Aunque la intención en primera instancia fue también añadir un bloque de peces en el mar mediante el objeto *NVBoids Fish*, situado en la carpeta *#NVJOB Boids->Simple Boids->Prefabs->Fish*, se desechó dicha opción ya que el movimiento de los peces era demasiado difícil de controlar, llegando estos a atravesar el terreno creado o a "nadar por el aire".

7.3.4 Pájaros

El modelo de pájaros que se ha usado en este proyecto ha sido obtenido del paquete *#NVJOB Simple Boids (Flocks of Birds, Fish and Insects)* [46].

Una vez importado, dentro de la carpeta *#NVJOB Boids->Simple Boids->Prefabs->Bird* se encuentra el objeto *NVBoids Bird*, el cual puede ser arrastrado a la escena para ser integrado en el entorno.

Los parámetros de los que dispone el bloque importado son los mostrados en la figura 7.10. Dichos parámetros fueron configurados mediante prueba y error, ya que la única forma posible de observar el resultado obtenido tras establecer los valores en los distintos campos era la iniciación de la simulación del entorno.

Una vez obtenido el resultado visual deseado, clonamos un total de quince veces el objeto con los parámetros establecidos y los repartimos por todo el entorno a una altura elevada.

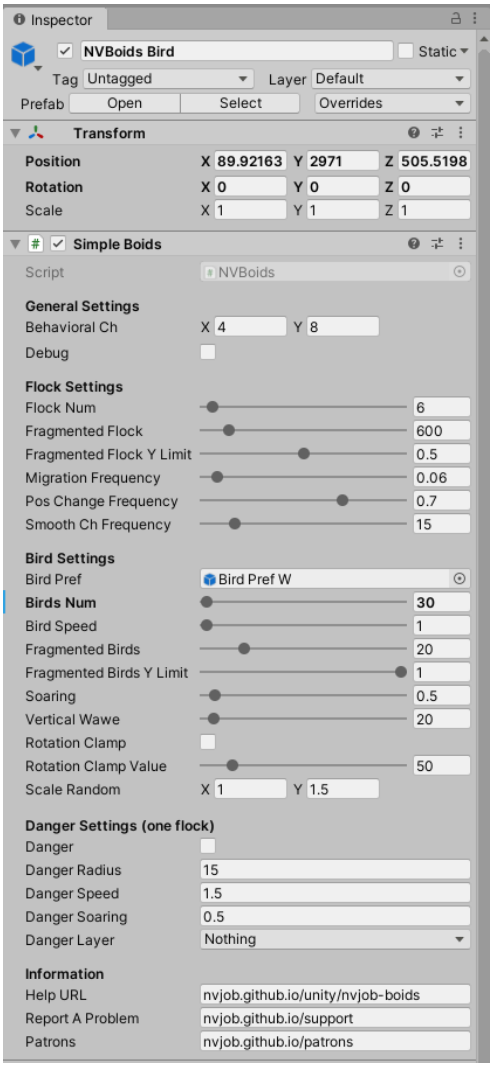

Figura 7-10. Parámetros para pájaros.

Figura 7-11. Pájaros.

7.4 Agua

Para la creación del agua del mar y la del lago se hizo uso del paquete *Standard Assets (for Unity 2018.4)* [47]. Dicho paquete es gratuito y ha sido descargado e importado de la propia Asset Store de Unity.

Todos los modelos cuentan con animación y con texturas, creada por los desarrolladores de los mismos.

7.4.1 Agua del lago

El modelo de agua que se ha usado en el lago ha sido obtenido del paquete *Standard Assets (for Unity 2018.4)* [47].

Una vez importado, dentro de la carpeta *Standard Assets->Enviroment->Water->Water4->Prefabs* se encuentra el archivo *Water4Simple*, el cual, una vez arrastrado a la escena, fue escalado y situado debidamente en el emplazamiento del lago. Dicho objeto cuenta con una serie de *scripts* asociados para controlar la ondulación de la superficie del agua, el reflejo y la luz que incide sobre la superficie. Al ser un lago, configuramos dichos *scripts* para que la ondulación del agua fuese suave y reflejase correctamente el entorno en función del punto desde el que se mire. En la figura 7.12 se puede observar el reflejo del cielo y de varios árboles sobre la superficie acuática.

Al objeto de agua creado se le asignó la etiqueta *pista* para posteriormente aplicar eventos de interacción con el usuario mediante el uso de scripts, sobre lo cual se hablará más adelante.

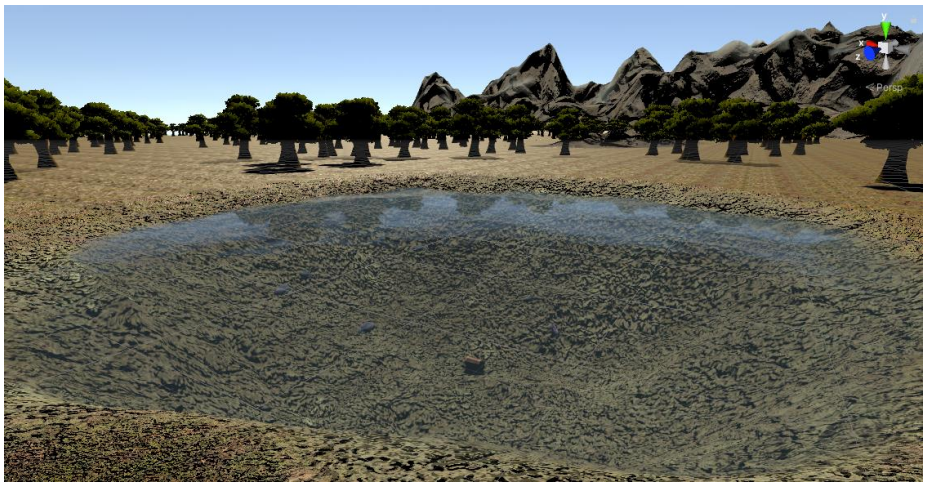

Figura 7-12. Agua del lago.

7.4.2 Agua del mar

El modelo de agua que se ha usado en el mar ha sido obtenido del paquete *Standard Assets (for Unity 2018.4)* [47].

Una vez importado, dentro de la carpeta *Standard Assets->Enviroment->Water->Water4->Prefabs* se encuentra el archivo *Water4Advanced*, el cual, una vez arrastrado a la escena, fue escalado y configurado sus parámetros.

Escalar dicho objeto provocó una pérdida de calidad en los bordes del mismo, dando lugar a que las olas del agua se apreciasen de forma poco realista. No obstante, era necesario escalar dicho objeto ya que la integración

en el proyecto de muchos de ellos provocaba un cuelgue de Unity, debido a la cantidad de información que la tarjeta gráfica tenía que procesar. Al igual que para el objeto del agua del lago, este objeto cuenta con una serie de *scripts* asociados para controlar el tamaño de las olas, la fuerza con la que rompen, la espuma que genera el mar, los reflejos en el agua, etc. Una vez configurados todos los parámetros para obtener el resultado deseado, clonamos el objeto resultante un total de ocho veces y lo situamos alrededor del entorno generado para rodearlo completamente por agua y conferir el aspecto de una verdadera isla.

Figura 7-13. Agua del mar.

En las siguientes dos fotos se muestran los parámetros elegidos para la configuración del agua del lago y del mar.

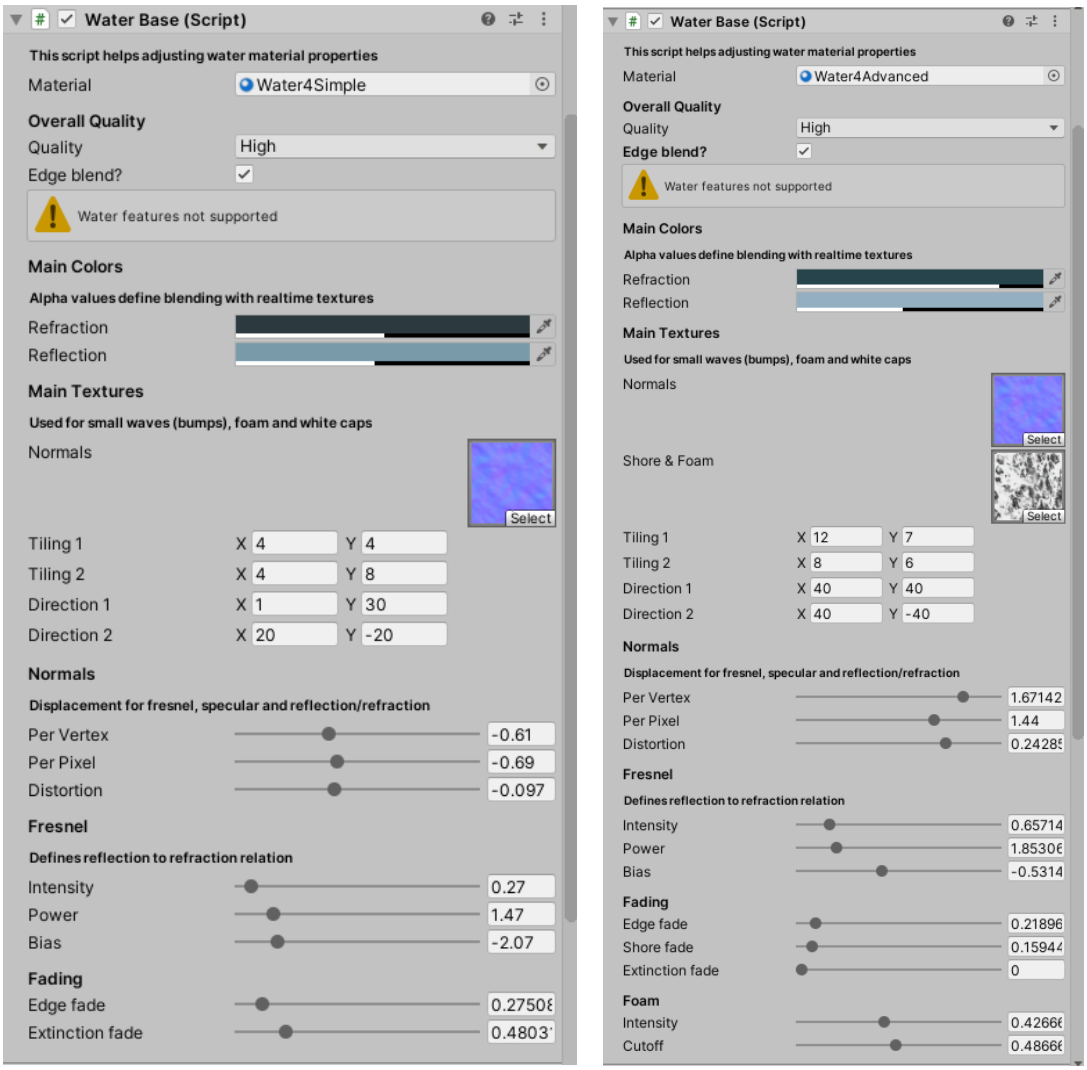

Figura 7-14. Parámetros del lago y del mar.

7.5 Objetos interactuables

Los objetos creados para ser interactuables por el usuario son los objetos tipo “rama” y tipo “piedra”. Para importarlos a Unity solo es necesario arrastrar el archivo **fbx** que se generó tras el proceso de exportación, explicado en los apartados anteriores, a una carpeta que se cree en el proyecto.

7.5.1 Piedras

En Unity se han importado tres modelos distintos de rocas, no obstante todas ellas han sido diseñadas a partir de solo una, dando distintos valores a los parámetros seleccionables para que adquiriesen distintas formas.

El proceso de creación de una roca se ha llevado a cabo en 3DS Max. Empezamos creando un elemento tipo *GeoSphere* en el *view port*. Dicho elemento puede ser encontrado en el panel de creación de objetos. Abriremos los modificadores del objeto creado y aumentaremos el número de segmentos a un valor aceptable, no demasiado alto para evitar tiempo de procesado a la GPU. A continuación, aplicamos el modificador

Displace e indicamos que el objeto es de tipo esférico y le aplicamos el *map* "Celullar". Modificando el parámetro *Strenght* distorsionaremos la forma inicial que poseía el objeto.

Dentro de los parámetros del material, en *Cell Characteristics*, seleccionamos *Circular* y aumentamos el valor de *Size*. El resultado de seguir estos pasos será la creación de una roca *Low Poly* como las mostradas en la figura 7.15.

Para crear los tres tipos distintos de rocas, bastó con hacer copias del objeto creado y modificar el valor del parámetro *Strenght* para alterar la forma del objeto inicial.

Una vez importados en Unity los tres modelos de piedras que se han creado para este proyecto, su integración en el entorno virtual se hará arrastrando a la escena cualquiera de los modelos. Unity reconocerá el tipo de objeto que se ha añadido a la escena y le otorgará los atributos correspondientes.

Además de los parámetros que Unity configura de manera automática, para la correcta interacción con el usuario, necesitaremos añadir a los objetos importados el parámetro *Box Collider* y marcar la casilla *Is Trigger*, lo cual nos servirá para la detección de eventos. Añadimos también el parámetro *Mesh Collider* y marcamos la casilla *Convex*, lo que creará un área de colisión adaptada al contorno del objeto en cuestión. Para que el objeto esté sujeto a las leyes de la gravedad, le añadimos el atributo *Rigidbody* y, finalmente, le asociamos *OVR Grabbable.cs*, un *script* propio de Oculus que nos permitirá la interacción con el objeto mediante las manos del jugador.

Al realizar todos estos cambios, los objetos estarán preparados para ser manipulados en tiempo real mediante el uso de los controladores touch de Oculus. Pueden ser agarrados, lanzados, intercambiarlo entre las manos, utilizados para empujar o golpear otros objetos, etc.

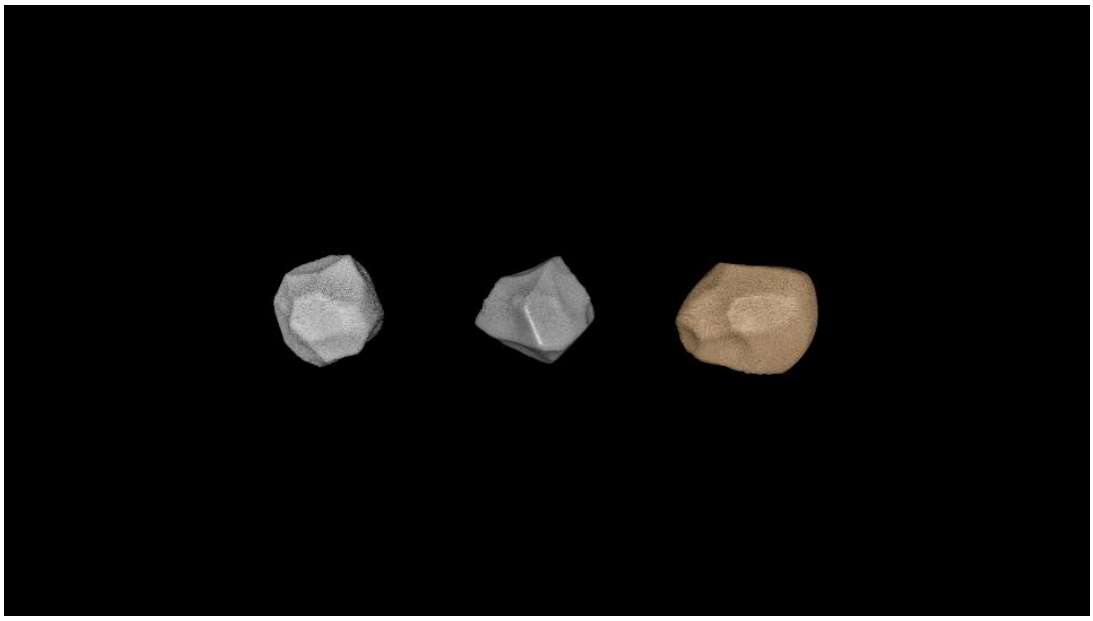

Figura 7-15. Tres modelos de rocas.

7.5.2 Rama

El otro de los objetos importados al entorno virtual, en este caso un único modelo, es un palo o rama con algunos esquejes.

El proceso de creación de la rama se ha llevado a cabo en 3DS Max. Empezamos creando un elemento tipo *Cylinder* en el *view port*. Dicho elemento puede ser encontrado en el panel de creación de objetos. Accederemos al menú de modificadores, aumentamos el número de segmentos a un valor aceptable, tal y

como explicamos antes, y convertimos el objeto en un *Editable Poly*. Una vez convertido, dentro de los modificadores del *Editable Poly*, seleccionamos *Border* y, seleccionando el borde del cilindro, reduciremos su escala para conferir al extremo una forma similar a la punta de una rama. De nuevo en los modificadores de *Editable Poly* seleccionamos *Polygon* y, seleccionando distintos grupos de polígonos para reescalarlos y desplazarlos levemente, conferiremos al cilindro el aspecto de una rama.

Una vez hemos hecho esto, haremos copias del objeto original y las escalaremos para reducir su tamaño considerablemente, tal y como podría ser el esqueje de una rama. De nuevo, mediante los modificadores de *Editable Poly* y *Polygon*, modificaremos dichos esquejes para hacerlos más realistas.

La herramienta *Select and Place*, situada en los accesos rápidos, nos permitirá colocar dichos esquejes en el objeto original a voluntad. Por último, realizaremos un *Attach* de la rama con los esquejes creados para generar así un único objeto como el mostrado en la figura 7.16.

Una vez importado en Unity el modelo de rama que se ha creado para este proyecto, su integración en el entorno virtual se hará arrastrando a la escena dicho modelo. Unity reconocerá el tipo de objeto que se ha añadido a la escena y le otorgará los atributos correspondientes.

Además de los parámetros que Unity configura de manera automática, para la correcta interacción con el usuario, necesitaremos añadir a los objetos importados el parámetro *Box Collider* y marcar la casilla *Is Trigger*, lo cual nos servirá para la detección de eventos. Añadimos también el parámetro *Mesh Collider* y marcamos la casilla *Convex*, lo que creará un área de colisión adaptada al contorno del objeto en cuestión. Para que el objeto esté sujeto a las leyes de la gravedad, le añadimos el atributo *Rigidbody* y, finalmente, le asociamos *OVR Grabbable.cs*, un *script* propio de Oculus que nos permitirá la interacción con el objeto mediante las manos del jugador.

Al igual que mencionamos para las piedras, tras aplicar todos estos cambios, los objetos *Rama* podrán ser agarrados, lanzados, intercambiarlo entre las manos, utilizados para empujar o golpear otros objetos, etc.

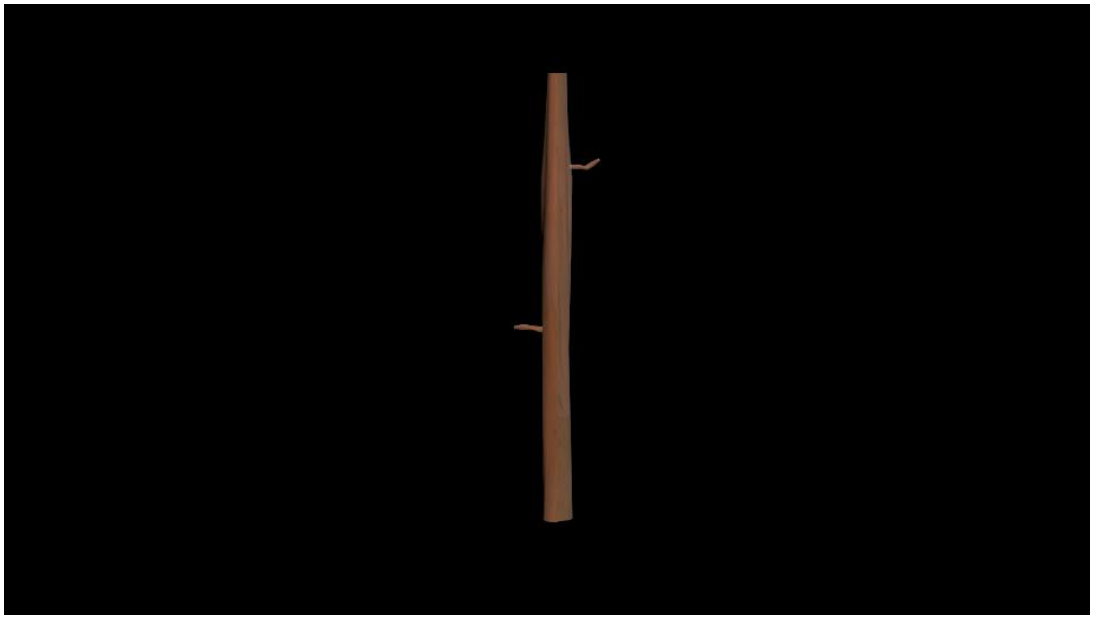

Figura 7-16. Modelo de rama.

7.6 Jugador

Para la creación del jugador se hizo uso del paquete *Oculus* [48]. Dicho paquete es gratuito y ha sido descargado e importado de la propia Asset Store de Unity.

Una vez importado, dentro de la carpeta *Oculus->VR->Prefabs* se encuentra el archivo *OVRPlayerController*, el cual, una vez arrastrado a la escena, fue situado en un emplazamiento designado para ser el lugar desde el que se inicia la inmersión al entorno.

Este objeto está compuesto por una gran cantidad de elementos, los cuales debemos configurar para su correcto funcionamiento. El primer paso será añadir el módulo *LocalAvatar* dentro de *OVRPlayerController->OVRCameraRig->TrackingSpace*. Tras añadir *LocalAvatar*, añadiremos los módulos *AvatarGrabberLeft* y *AvatarGrabberRight*, los cuales contienen un *script* denominado *OVR Grabber.cs*, el cual mejora el funcionamiento de los controladores Oculus Touch a la hora de agarrar objetos. Además, estos módulos contienen el modelo de las manos que nos permitirá visualizar en tiempo real los gestos y movimientos que hacemos con las mismas.

Para que la altura del visor dentro de la simulación se corresponda con la altura aproximada del jugador, dentro del módulo *OVRCameraRig* modificamos el valor de *Tracking Origin Type*, fijando su valor en *Floor Level*, lo cual medirá la distancia que hay entre el suelo y las gafas de realidad virtual. Además de fijar la altura a la que el visor se encontrará en la simulación, también nos permitirá agacharnos dentro de la misma si nos agachamos en la vida real.

Por último, es también de suma importancia marcar la opción *Dynamic Height* dentro del objeto *OVRPlayerController* para habilitar la rotación o desplazamiento de la cámara conforme a los movimientos de la cabeza del jugador.

Además, para las interacciones con el entorno, se emplea la etiqueta “jugador”, la cual se asoció a *OVRPlayerController*.

7.6.1 Filtro de visión

Para otorgar la sensación de que realmente el jugador se encuentra bajo agua al adentrarse en el lago o el mar se ha usado un objeto que ocupa toda la extensión del terreno, a nivel del agua. A dicho objeto le añadimos el parámetro *BoxCollider* y marcamos la casilla *Is Trigger*, lo cual nos servirá para la detección de eventos. A continuación añadir el parámetro *Post-process Volume* y creamos un filtro azul, el cual otorgará la sensación de estar sumergido bajo el agua. Para la configuración de los parámetros del filtro fue necesario realizar ensayos de prueba y error, pues solo es posible observar el resultado de la aplicación del filtro una vez se ha iniciado la simulación. Por último, añadimos el parámetro *Audio Source* a la lista de parámetros y dentro de *AudioClip* seleccionamos el archivo *Underwater*, el cual es un archivo de audio cuyos sonidos otorgan la sensación de estar realmente bajo el agua.

Este bloque reaccionará cuando el jugador haya “introducido la cabeza” bajo el agua del mar o del lago, tiñendo la visión de la cámara en tonos azulados y reproduciendo el archivo de audio mencionado. Dicho filtro de visión y la reproducción del sonido se detendrán una vez la cabeza del jugador se encuentre por encima del nivel del agua.

Figura 7-17. Bajo Agua.

7.7 Sonidos

A lo largo del recorrido por el entorno virtual, el jugador experimentará una serie de sonidos los cuales tienen la intención de hacer la inmersión al entorno más realista y también definir cuál es el cometido que debe cumplir para así poder abandonar este entorno.

Todos los archivos de sonidos utilizados a lo largo de este proyecto están libres de copyright.

7.7.1 Presentador inicio

Una vez el jugador tenga colocadas las gafas de realidad virtual y de comienzo la simulación del entorno, escuchará la voz de un hombre que le confiere la misión de encontrar un cofre allí abandonado. El archivo de audio ha sido grabado por Ricardo Manuel Olivares Bizcocho, el cual se ofreció a prestar su voz, de manera altruista, para este proyecto de fin de carrera.

Para incorporar el audio a la escena, creamos un bloque dentro del objeto *OVRPlayerController* y añadimos el parámetro *Audio Source*. Dentro de *AudioClip* seleccionamos el archivo *Rick Inicio*, el cual contiene el audio que confiere la misión al jugador. Para que dicho audio se reproduzca al inicio de la simulación, será necesario marcar la casilla *Play On Awake* dentro de *Audio Source*.

7.7.2 Presentador final

Una vez el jugador encuentra el cofre del tesoro, mencionado al inicio de la simulación, escuchará la voz de un hombre felicitándolo por haber cumplido satisfactoriamente la tarea. El archivo de audio ha sido grabado por Ricardo Manuel Olivares Bizcocho, el cual se ofreció a prestar su voz, de manera altruista, para este proyecto de fin de carrera.

Para incorporar el audio a la escena, creamos un bloque dentro del objeto *Tesoro* y añadimos el parámetro *Audio Source*. Dentro de *AudioClip* seleccionamos el archivo *Rick Final*, el cual contiene el audio que da fin a la misión del jugador. Este audio se reproduce en el momento en el que el cofre del tesoro detecta una colisión con el jugador gracias a un *script* que explicaremos más adelante.

7.7.3 Ambiente

Una vez el jugador tenga colocadas las gafas de realidad virtual y de comienzo la simulación del entorno, se reproducirá en bucle un archivo de audio con música procedente de un piano, la cual pretende ser relajante.

Para incorporar el audio a la escena, creamos un bloque dentro del objeto *OVRPlayerController* y añadimos el parámetro *Audio Source*. Dentro de *AudioClip* seleccionamos el archivo *Ambiente*, el cual contiene el audio con música relajante. Para que dicho audio se reproduzca al inicio de la simulación, será necesario marcar la casilla *Play On Awake* dentro de *Audio Source*.

7.7.4 Olas

Cuando el jugador entre en contacto con un objeto invisible situado en la orilla de la playa, se comenzará a reproducir un archivo de audio que contiene un sonido de las olas del mar. El bloque invisible tendrá el parámetro *Box Collider* y la casilla *Is Trigger* marcada, para la detección de eventos.

Para incorporar el audio a la escena, creamos un bloque dentro del objeto invisible mencionado y añadimos el parámetro *Audio Source*. Dentro de *AudioClip* seleccionamos el archivo *Olas*. El archivo de audio se reproducirá gracias a la detección de la colisión por parte de un *script* que explicaremos más adelante.

7.7.5 Bajo agua

Cuando el jugador entre en contacto con un objeto invisible situado en la superficie del agua, se comenzará a reproducir un archivo de audio que contiene un sonido de inmersión bajo agua. El bloque invisible tendrá el parámetro *Box Collider* y la casilla *Is Trigger* marcada, para la detección de eventos.

Para incorporar el audio a la escena, creamos un bloque dentro del objeto invisible mencionado y añadimos el parámetro *Audio Source*. Dentro de *AudioClip* seleccionamos el archivo *Underwater*. El archivo de audio se reproducirá gracias a la detección de la colisión por parte de un *script* que explicaremos más adelante.

7.7.6 Salto

En un inicio, cuando el jugador finalizaba la misión principal y daba paso a la animación del cofre, este comenzaba a dar saltos y se reproducía un archivo de audio que contenía un sonido onomatopéyico.

Para incorporar el audio a la escena, creamos un bloque dentro del objeto *Camera* y añadimos el parámetro *Audio Source*. Dentro de *AudioClip* seleccionamos el archivo *Boing*, el cual contiene el audio con el sonido onomatopéyico. Para que dicho audio se reprodujese al inicio de la simulación, fue necesario marcar la casilla *Play On Awake* dentro de *Audio Source*.

Dicha idea fue finalmente descartada debido a que rompía con la estética del entorno generado, ya que este pretende ser realista y evocar la emoción secundaria de la serenidad. Estos factores que dieron lugar a la eliminación de la animación de los saltos del cofre del tesoro y del sonido *Boing*.

7.7.7 Pájaros

Una vez el jugador tenga colocadas las gafas de realidad virtual y de comienzo la simulación del entorno, se reproducirá en bucle un archivo de sonido de graznidos de aves.

Para incorporar el audio a la escena, creamos un bloque dentro de *OVRPlayerController* y añadimos el parámetro *Audio Source*. Dentro de *AudioClip* seleccionamos el archivo *Pajaros*, el cual ha sido obtenido del paquete *#NVJOB Simple Boids (Flocks of Birds, Fish and Insects)* [46]. Para que dicho audio se reproduzca al inicio de la simulación, será necesario marcar la casilla *Play On Awake* dentro de *Audio Source*.

7.8 Tesoro

Para la creación del cofre del tesoro se hizo uso del paquete *Animated Cartoon Treasure Chest* [49]. Dicho paquete es gratuito y ha sido descargado e importado de la propia Asset Store de Unity. El modelo del cofre cuenta con animación y con texturas, creada por los desarrolladores del mismo.

Una vez importado, dentro de la carpeta *FS Loot Boxes->Assets->Prefabs* se encuentra el archivo *TreasureChestPrefab*, el cual fue arrastrado a la escena y colocado en el fondo del lago. Añadimos al objeto el parámetro *Box Collider*, para detectar las colisiones, y *Rigidbody*, para sujetarlo a las leyes físicas.

Entre los elementos que el fabricante de dicho paquete nos ofrece se encuentra *LootBox.cs*, el cual configuramos para que se ejecutase ante una colisión, *Open On Collision*, al detectar un objeto con la etiqueta "jugador". Además, tuvimos que modificar el código de dicho *script* para alterar su comportamiento, tal y como explicaremos más adelante.

Finalmente, añadimos al objeto *Tesoro* el parámetro *Audio Source*. Dentro de *AudioClip* seleccionamos el archivo *Rick Final*, el cual contiene el audio que da fin a la misión del jugador, tal y como explicamos previamente. Este audio se reproducirá gracias a las modificaciones efectuadas en el código *LootBox.cs*.

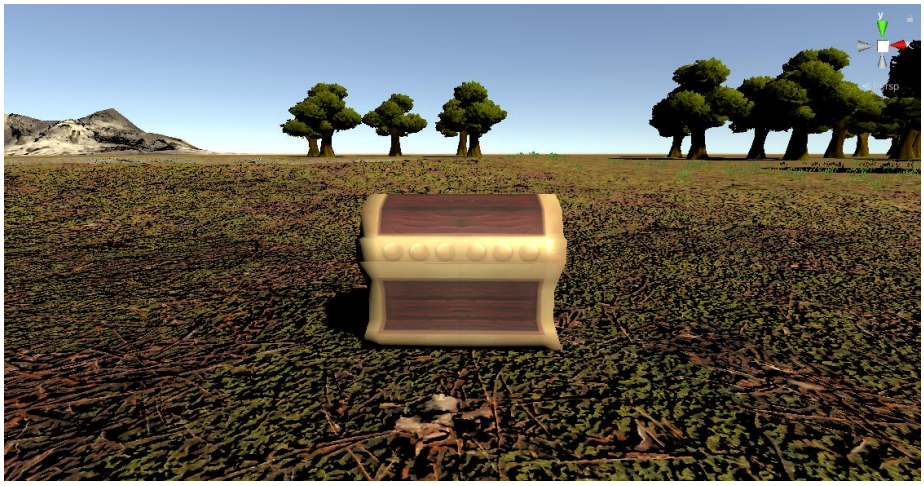

Figura 7-18. Tesoro.

7.9 Scripts

A lo largo de este proyecto ha sido necesario emplear, además de los que los propios paquetes incorporan, un total de ocho *scripts* para controlar el comportamiento de objetos o para activar y desactivar eventos. En este apartado mostraremos el código de todos los *scripts* que hemos desarrollado para el control de los eventos y explicaremos su funcionamiento.

7.9.1 BajoAgua.cs

El siguiente *script* detecta la entrada del jugador en el agua y comienza a reproducir el archivo *Underwater*. Además, incorpora una función que será llamada desde otro script, en caso de que el jugador salga del agua, la cual parará la música.

Para detectar la entrada del usuario en el agua hacemos uso de la función de *C#* llamada *OnTriggerEnter*, la cual, gracias a la caja invisible definida en apartados anteriores, detecta si el objeto que está entrando en contacto con el agua tiene la etiqueta "jugador" y, en caso de ser así, inicia la reproducción del audio *Underwater* mediante el uso de la función *Play()*.

La función definida en este *script* para ser llamada desde otro es *ParaMusica()*, la cual detendrá la reproducción del audio *Underwater* mediante el uso de la función *Stop()*.

El código puede verse reflejado en la siguiente imagen:

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

public class BajoAgua : MonoBehaviour
{
    AudioSource Underwater;
    int flag = 0;
    bool entrar = false;

    public void Start()
    {
        Underwater = GetComponent<AudioSource>();

    }

    public void OnTriggerEnter(Collider other)  //Suena al activar el trigger
    {
        if(other.gameObject.tag == "jugador"){
            flag = flag + 1;
            if(flag == 1 && entrar == false){
                Underwater.Play();
                entrar = true;
            }
        }
    }
    public void ParaMusica(){

        Underwater.Stop();
        flag = 0;
        entrar = false;
    }

}
```

Figura 7-19. Script BajoAgua.

7.9.2 BajoAguaStop.cs

El siguiente *script* detecta la salida del jugador del agua y detiene la reproducción del archivo *Underwater*, haciendo uso para ello de la función ParaMusica(), la cual se encuentra en el script BajoAgua.cs.

Para detectar la salida del usuario del agua hacemos uso de la función de *C#* llamada *OnTriggerEnter*. Lo ideal en esta situación habría sido usar la función *OnTriggerExit*, pero al existir problemas de compatibilidad con un jugador implementado en realidad virtual tuvimos que pensar una solución alternativa, la cual fue crear un bloque invisible justo encima del nivel del agua para que detecte el contacto de un objeto y, si la etiqueta del

mismo es “jugador”, se invocará a la función *ParaMusica()* situada en *BajoAgua.cs*, la cual hemos comentado previamente.

El código puede verse reflejado en la siguiente imagen:

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

public class BajoAguaStop : MonoBehaviour
{
    public BajoAgua music;
    void Start()
    {
        music = GameObject.FindObjectOfType(typeof(BajoAgua)) as BajoAgua;
    }

    void OnTriggerEnter(Collider other) //Suena al activar el trigger
    {
        if(other.gameObject.tag == "jugador"){

            music.ParaMusica();
        }
    }

}
```

Figura 7-20. Script BajoAguaStop.

7.9.3 Conejo.cs

El siguiente *script* confiere “inteligencia artificial” a los objetos de tipo Conejo que se encuentran en la escena. Continuamente actualiza la posición en la que se encuentran en la función *Update()*. Para ello, obtenemos el valor absoluto de la diferencia entre la posición del jugador y la posición de cada uno de los distintos conejos en la escena. Si esa diferencia es menor que un valor establecido, cada uno de los conejos para los que se ha cumplido esta condición intentarán “huir” aumentando su velocidad, gracias a *moveSpeed*, y avanzarán en dirección contraria al jugador, gracias a *transform.position*. Este cambio de dirección es gradual, ya que un giro de 180 grados con respecto al jugador en una milésima de segundo no se vería real.

Siempre que no se cumplan las condiciones anteriores, o para los conejos que no se haya cumplido la condición, mediante una función de *C#* para generar números aleatorios, *Random*, se determina durante cuantos segundos avanzarán, hacia donde rotarán y la velocidad con la que se desplazan. Todos estos cambios en el movimiento vienen de la mano de la función *transform.position*.

El código puede verse reflejado en la siguiente imagen:

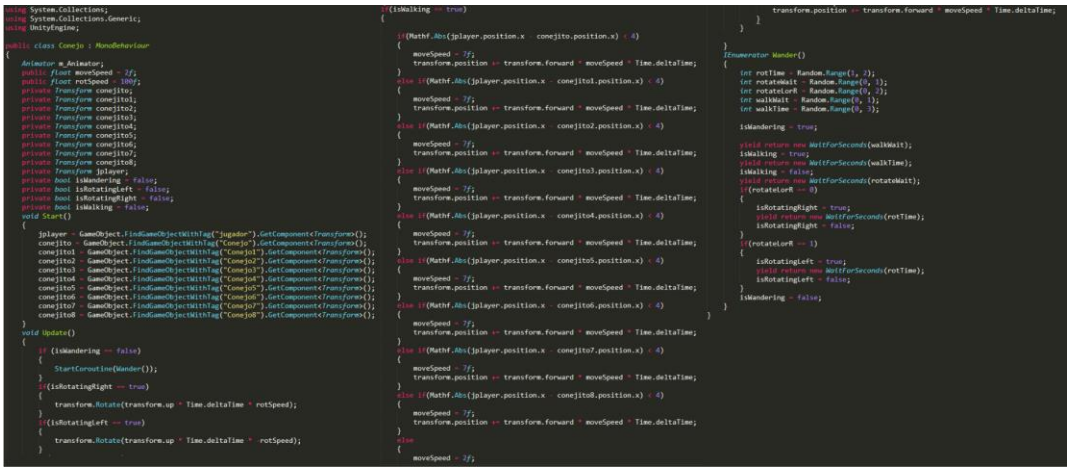

Figura 7-21. Script Conejo.

7.9.4 EndGame.cs

El siguiente *script* se ejecuta a la finalización de la simulación, justo cuando se produce el cambio de escena y el jugador se encuentra situado en frente del cofre del tesoro. Espera durante diez segundos tras su ejecución y finaliza con la simulación del entorno tras dicha espera.

Para contabilizar el tiempo que debe esperar antes de dar paso a la ejecución de la siguiente línea de código usamos la función *WaitForSeconds(10)*, siendo 10 el número de segundos que va a esperar. Una vez han pasado diez segundos, se ejecuta la siguiente línea de código, *Application.Quit()*, la cual da fin a la simulación.

El código puede verse reflejado en la siguiente imagen:

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

public class EndGame : MonoBehaviour
{
    IEnumerator Start()
    {
        yield return new WaitForSeconds(10);
        Application.Quit();
    }


}
```

Figura 7-22. Script EndGame.

7.9.5 RestartGame.cs

El siguiente *script* devuelve la simulación al punto inicial, reiniciando todos los sonidos, eventos y la posición del jugador. Se activa en caso de que el jugador caiga "al vacío".

Para detectar la caída del usuario "al vacío" hacemos uso de la función de *C#* llamada *OnTriggerEnter*, la cual, gracias a la caja invisible definida en apartados anteriores, detecta si el objeto que está saliendo de los límites establecidos en el entorno tiene la etiqueta "jugador" y, en caso de ser así, llama a la función *Application.LoadLevel(0)*. Al poner un 0 dentro de los paréntesis estamos indicando a Unity que queremos volver al estado inicial de la simulación.

El código puede verse reflejado en la siguiente imagen:

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

public class RestartGame : MonoBehaviour
{
public void Start() {

 }
public void OnTriggerEnter(Collider other)
      {
       if(other.gameObject.tag == "jugador"){
        Application.LoadLevel(0);
        }
      }
 }
```

Figura 7-23. Script RestartGame.

7.9.6 SoundOlas.cs

El siguiente *script* detecta cuando el jugador está cerca del mar y comienza a reproducir el archivo *Olas*. Además, incorpora una función que será llamada desde otro script, en caso de que el jugador se aleje del mar, la cual parará la música.

Para detectar la entrada del usuario en la orilla del mar hacemos uso de la función de *C#* llamada *OnTriggerEnter*, la cual, gracias a la caja invisible definida en apartados anteriores, detecta si el objeto que está entrando en contacto con la orilla del mar tiene la etiqueta "jugador" y, en caso de ser así, inicia la reproducción del audio *Olas* mediante el uso de la función *Play()*.

La función definida en este *script* para ser llamada desde otro es *ParaMusica()*, la cual detendrá la reproducción del audio *Olas* mediante el uso de la función *Stop()*.

El código puede verse reflejado en la siguiente imagen:

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

public class SoundOlas : MonoBehaviour
{
   AudioSource Olas;
   int flag = 0;
   bool entrar = false;
   public void Start()
    {
       Olas = GetComponent<AudioSource>();
    }

   public void OnTriggerEnter(Collider other) //Suena al activar el trigger
   {
       if(other.gameObject.tag == "jugador"){
          flag = flag + 1;
          if(flag == 1 && entrar == false){
              Olas.Play();
              entrar = true;
          }
       }

   }
   public void ParaMusica(){
       Olas.Stop();
       flag = 0;
       entrar = false;
   }
   }
```

Figura 7-24. Script SoundOlas.

7.9.7 SoundOlasStop.cs

El siguiente script detecta que el jugador se aleja del mar y detiene la reproducción del archivo *Olas*, haciendo uso para ello de la función ParaMusica(), la cual se encuentra en el script SoundOlas.cs.

Para detectar que el usuario se aleja de la orilla del mar hacemos uso de la función de *C#* llamada *OnTriggerEnter*. Lo ideal en esta situación habría sido usar la función *OnTriggerExit*, pero al existir problemas de compatibilidad con un jugador implementado en realidad virtual tuvimos que pensar una solución alternativa, la cual fue crear un bloque invisible justo al inicio de la orilla del mar para que detecte el contacto de un objeto y, si la etiqueta del mismo es "jugador", se invocará a la función *ParaMusica()* situada en *SoundOlas.cs*, la cual hemos comentado previamente.

El código puede verse reflejado en la siguiente imagen:

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

public class SoundOlasStop : MonoBehaviour
{
    public SoundOlas music;
    void Start()
    {
        music = GameObject.FindObjectOfType(typeof(SoundOlas)) as SoundOlas;
    }

    void OnTriggerEnter(Collider other) //Suena al activar el trigger
    {
        if(other.gameObject.tag == "jugador"){
        music.ParaMusica();
        }

    }

}
```

Figura 7-25. Script SoundOlasStop.

7.9.8 WanderAI.cs

El siguiente script confiere "inteligencia artificial" a los objetos de tipo Mariposa que se encuentran en la escena. Continuamente actualiza la posición en la que se encuentran en la función *Update()*. Para ello, llama a la función *FindClosestEnemy()*, la cual encuentra a todos los objetos del tipo *WanderAI* y calcula el valor absoluto de la posición de todos ellos con respecto al jugador. Si el valor calculado es menor que un valor previamente establecido, las mariposas que se vean afectadas por esta condición tratarán de "huir" de la misma forma que los conejos, aumentando su velocidad, gracias a *moveSpeed*, y avanzando en dirección contraria al jugador, gracias a *transform.position*. Este cambio de dirección es gradual, ya que un giro de 180 grados con respecto al jugador en una milésima de segundo no se vería real.

Siempre que no se cumplan las condiciones anteriores, o para los conejos que no se haya cumplido la condición, mediante una función de *C#* para generar números aleatorios, *Random*, se determina durante cuantos segundos avanzarán, hacia donde rotarán y la velocidad con la que se desplazan. Todos estos cambios en el movimiento vienen de la mano de la función *transform.position*.

El código puede verse reflejado en la siguiente imagen:

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

public class WanderAI : MonoBehaviour
{
    public float moveSpeed = 2f;
    public float rotSpeed = 100f;
    private Transform maripos;
    private Transform jplayer;
    public Transform closest;
    private bool isWandering = false;
    private bool isRotatingLeft = false;
    private bool isRotatingRight = false;
    private bool isWalking = false;
    void Start()
    {
        jplayer = GameObject.FindGameObjectWithTag("jugador").GetComponent<Transform>();
        maripos = GameObject.FindGameObjectWithTag("Mariposa").GetComponent<Transform>();
    }
    void Update()
    {
        FindClosestEnemy();

        if (isWandering == false)
        {
            StartCoroutine(Wander());
        }
        if(isRotatingRight == true)
        {
            transform.Rotate(transform.up * Time.deltaTime * rotSpeed);
        }
        if(isRotatingLeft == true)
        {
            transform.Rotate(transform.up * Time.deltaTime * -rotSpeed);
        }
        if(isWalking == true)
        {

            if(Mathf.Abs(jplayer.position.x - maripos.position.x) < 2)
            {
                moveSpeed = 6f;
                transform.position += transform.forward * moveSpeed * Time.deltaTime;
            }
            else
            {
                moveSpeed = 2f;
                transform.position += transform.forward * moveSpeed * Time.deltaTime;
            }
        }
    }
```

```
    void FindClosestEnemy()
    {
        WanderAI[] gos = FindObjectsOfType<WanderAI>();

        var distance = Mathf.Infinity;
        var position = transform.position;
        foreach (WanderAI go in gos)
        {
            var diff = (go.transform.position - position);
            var curDistance = diff.sqrMagnitude;
            if (curDistance < distance)
            {
                closest = go.transform;
                distance = curDistance;
            }
        }

        maripos = closest;

    }

    IEnumerator Wander()
    {
        int rotTime = Random.Range(1, 3);
        int rotateWait = Random.Range(1, 3);
        int rotateLorR = Random.Range(0, 3);
        int walkWait = Random.Range(1, 3);
        int walkTime = Random.Range(1, 5);

        isWandering = true;

        yield return new WaitForSeconds(walkWait);
        isWalking = true;
        yield return new WaitForSeconds(walkTime);
        isWalking = false;
        yield return new WaitForSeconds(rotateWait);
        if(rotateLorR == 1)
        {
            isRotatingRight = true;
            yield return new WaitForSeconds(rotTime);
            isRotatingRight = false;
        }
        if(rotateLorR == 2)
        {
            isRotatingLeft = true;
            yield return new WaitForSeconds(rotTime);
            isRotatingLeft = false;
        }
        isWandering = false;
    }
}
```

Figura 7-26. Script WanderAI.

7.9.9 LootBox.cs

Este script fue diseñado por el creador del paquete *Animated Cartoon Treasure Chest* [49]. Es importante mencionarlo aquí ya que, aunque no es uno de los ocho scripts originales de este proyecto, ha sido modificado ligeramente para alterar su comportamiento.

Cuando el script detecte la colisión de un objeto, gracias a la función *OnCollisionEnter,* comprobará si el objeto que ha colisionado tiene la etiqueta "jugador" y, en caso afirmativo, reproducirá el audio *Rick final* gracias a la función *Play()*, esperará durante cinco segundos, gracias a *WaitForSeconds(5)*, y después cargará la escena en la que, ya fuera del lago, se encuentran el jugador y el cofre, uno frente al otro. Este cambio de escena se consigue gracias a la función *LoadScene(1)*, siendo 1 el número de la escena que se desea simular.

La modificación del código ha sido la siguiente:

```
IEnumerator OnCollisionEnter(Collision collision)
{
    // OnCollisionMethod is not for OpenOnTouch method
    //if (openingMethod == OpeningMethods.OpenOnTouch) return;

    // check if the hitting object is our player
    if (collision.gameObject.tag == playerTag)
    {
        // if the method is OpenOnKeyPress, let's just flag the player as close
        if (openingMethod == OpeningMethods.OpenOnKeyPress) isPlayerAround = true;

        // otherwise, open the box.
        else Open();

            if(collision.gameObject.tag == "jugador"){
                flag = flag + 1;
                if(flag == 1){
                    Rick.Play();
                    yield return new WaitForSeconds(5);
                    SceneManager.LoadScene(1);

               //Debug.Log("Entro");
                    }

                }

    }
}
```

Figura 7-27. Script LootBox.

8 RESULTADO Y LÍNEAS FUTURAS

Virtual reality, all the A.I. work we do, all the robotics work we do - we're as close to realizing science fiction as it gets..

- Jensen Huang -

Tal y como se ha mencionado varias veces a lo largo de esta memoria, el objetivo de este proyecto fin de grado era la creación de un entorno virtual el cual fuese capaz de evocar en el usuario involucrado la emoción secundaria de la serenidad. En este capítulo se tratará el resultado obtenido y las líneas futuras.

8.1 Resultado

El objetivo establecido antes de la realización de este proyecto era la creación de un entorno virtual en el cual un usuario pudiese moverse libremente e interactuar con el entorno, evocando en el mismo la emoción de la serenidad debido a las características especiales del entorno creado.

A la vista del entorno virtual generado se puede afirmar que:

- El entorno virtual generado es totalmente funcional para el uso de gafas de realidad virtual.
- Todos los elementos que integran el entorno son de libre uso o han sido generados por el autor del mismo.
- Es interactuable con el usuario, mediante eventos u objetos.

Es por ello que se han alcanzado todos los objetivos propuestos para este trabajo fin de carrera. No obstante, existe un margen de mejora. El siguiente apartado tratará sobre posibles modificaciones y estudios a realizar.

Figura 8-1. Prueba entorno virtual.

8.2 Líneas futuras

A pesar de que la realidad virtual es una tecnología en auge y que cada vez encontramos más productos y servicios dependientes de esta, aún presenta problemas de compatibilidad con algunas funciones de los lenguajes de programación.

Durante el desarrollo del entorno virtual generado para este proyecto de fin de carrera se han usado diferentes funciones para la interacción con el usuario. Tres de estas funciones, *OnTriggerEnter*, *OnTriggerExit* y *OnTriggerStay*, no se comportan tal y como se espera ante la presencia de un jugador diseñado para realidad virtual, llamándose continuamente así mismas en lugar de una sola vez.

Esto ha provocado que a lo largo del proyecto se hayan tenido que crear scripts para detener sonidos en lugar de un solo script que los reproduzca y detenga, que no sea posible realizar una interacción con los objetos Conejo y Mariposa sin que estos pierdan el control o entren en bucle dando lugar al cuelgue de Unity o que algunos eventos no ocurran tal y como se espera debido a que no se ejecuta la función.

Concretamente hay dos puntos que no se han podido realizar debido a la dificultad, a la falta de tiempo y a los fallos obtenidos a lo largo de la realización del proyecto. La solución alternativa a estos fue modificar el script *Conejo.cs* y *WanderAI.cs* para hacer que conejos y mariposas huyesen del jugador al este acercarse. Si el proyecto hubiese seguido la línea original, estos puntos habrían sido:

- El conejo detectaría que el jugador tiene agarrado un objeto de tipo “Zanahoria” y se acercaría al jugador, deteniéndose ante este para comer hasta que la Zanahoria fuese soltada.
- Al estar cerca de un grupo de mariposas, si el jugador extiende la mano y estira el dedo índice, la mariposa más cercana detectaría el evento y se acercaría volando hasta el dedo para posarse en él.

Además, debido a las circunstancias especiales que hemos experimentado a lo largo del año 2020, no ha sido posible reunir a un grupo de sujetos para que, mediante un casco capaz de medir señales cerebrales, se constatase que realmente el entorno virtual generado es capaz de evocar la emoción buscada.

Medir las señales cerebrales en distintos individuos y comprobar si realmente el entorno generado evoca la emoción deseada es, sin lugar a dudas, fundamental para corroborar la correcta inducción de la emoción y es por ello la línea futura más importante.

REFERENCIAS

[1] Plutchik, R. y Kellerman, H. (ed.). *Theories of emotion.* Academic Press, 2013. [consulta: 15-11-2020]

[2] Mateus, S. P. y Giraldo, J. E. Diseño de un Modelo 3D del Politécnico Colombiano Jaime Isaza Cadavid con Realidad Virtual. En: *Información tecnológica* [en línea], 2012, vol. 23, no. 3, pp. 95-102 [consulta: 16-11-2020]. Disponible en: http://dx.doi.org/10.4067/S0718-07642012000300012

[3] Darwin, C; Prodger, P. *The expression of the emotions in man and animals*. Oxford University Press, USA, 1998. [consulta: 04-02-2021]. Disponible en: https://www.gla.ac.uk/myglasgow/library/files/special/exhibns/month/nov2009.html

[4] Plutchik, R. *The emotions*. University Press of America, 1991. [consulta: 03-02-2021]. Disponible en: https://positivepsychology.com/emotion-wheel/

[5] Rivera Arrizabalaga, A. *Arqueología de las emociones*. 2015. [consulta: 05-02-2021].

[6] Herculano-Houzel, S. *The Human Brain in Numbers: A Linearly Scaled-up Primate Brain.* Hum Neurosci, 2009. [consulta: 07-02-2021]. Disponible en: https://www.researchgate.net/publication/38091649_The_Human_Brain_in_Numbers_A_Linearly_Scaled-up_Primate_Brain

[7] Sistema límbico [Internet]. [consultado el 5 de diciembre de 2020]. Disponible en: https://pymstatic.com/3444/conversions/sistema-limbico-partes-default.jpg

[8] Sherman, S. M y Guillery, R. W. *Exploring the Thalamus*. Academic Press, 2000. [consulta: 08-02-2021]. Disponible en: https://brainmaster.com/software/pubs/brain/Exploring%20the%20Thalamus%20and%20Its%20Role%20in%20Cortical%20Function.pdf

[9] Kandel, E.R. y Schwartz, J.H. y Jessell, T.M. *Principios de neurociencia*. Cuarta edición. McGraw-Hill Interamericana. Madrid, 2001. [consulta: 11-02-2021]. Disponible en: https://doku.pub/documents/principios-de-neurociencia-eric-r-kandel-z06w5mpg2zqx

[10] *Bases neurobiológicas de las emociones* [Internet]. Neurowikia, 2019 [consulta: 12-02-2021]. Disponible en: http://www.neurowikia.es/content/bases-neurobiologicas-de-las-emociones

[11] Damasio, A. *En busca de Spinoza: neurobiología de la emoción y los sentimientos*, 2014. [consulta: 04-02-2021] Disponible en: https://gredos.org/Varios/Damasio%20Antonio%20-

%20En%20Busca%20De%20Spinoza.pdf

[12] Rueda de las Emociones de Plutchik [Internet]. [consultado el 5 de diciembre de 2020]. Disponible en: https://esp.6seconds.org/wp-content/uploads/2020/07/480px-Rueda-Plutchik.svg_.png

[13] García Domínguez, A.E. y Cruz Ramírez, N. y Coria Ávila, G. *Análisis de ondas cerebrales para determinar emociones a partir de estímulos visuales*, 2015. [consulta: 04-02-2021] Disponible en: https://cdigital.uv.mx/bitstream/handle/123456789/46639/GarciaDominguezAna.pdf?sequence=2&isAllowed=y

[14] Gutiérrez Maldonado, J. Aplicaciones de la realidad virtual en psicología clínica. En: *Aula médica psiquiatría* [en línea], 2002, vol. 4, no. 2, pp.92-126 [consulta: 18-11-2020]. Disponible en: www.ub.edu/personal/jgutierrez/realidadvirtual.pdf

[15] Sutherland, I. E. Sketchpad a man-machine graphical communication system. *Simulation* [en línea], 1964, vol. 2, no. 5, pp. R-3-R-20 [consulta: 21-11-2020]. Disponible en: https://doi.org/10.1177/003754976400200514

[16] Sutherland, I. E. A head-mounted three dimensional display. *Proceedings of the December 9-11, 1968, fall joint computer conference, part I* [en línea], 1968, pp. 757-764 [consulta: 22-11-2020]. Disponible en: https://doi.org/10.1145/1476589.1476686

[17] Hilera, J. R., Otón, S. y Martínez, J. Aplicación de la Realidad Virtual en la enseñanza a través de Internet. En: *Cuadernos de documentación multimedia* [en línea], 1999, vol. 8, pp. 25-35 [consulta: 24-11-2020]. Disponible en: https://revistas.ucm.es/index.php/CDMU/article/download/59110/4564456546539/0

[18] Campbell, B., Rosse, C. y Brinkley, J. F. The Virtual Anatomy Lab: a hands-on anatomy learning environment. *Studies in health technology and informatics*, 2001, pp. 85-87.

[19] Blodgett, A. T., et al. Moving beyond words: Exploring the use of an arts-based method in Aboriginal community sport research. *Qualitative research in sport, exercise and health* [en línea], 2013, vol. 5, no. 3, pp. 312-331 [consulta: 27-11-2020]. Disponible en: https://doi.org/10.1080/2159676X.2013.796490

[20] Magnusson Nählinder, S. *Flight simulator training: Assessing the potential* [en línea]. Doctoral thesis. Linköping University Electronic Press, 2009, p. 45 [consulta: 29-11-2020]. Disponible en: https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A209999&dswid=4014

[21] García Pérez, L. *Cinelo, un cine de experiencias* [en línea]. Trabajo Fin de Grado. Universidad Rovira i Virgili (URV), 2019. [Consulta: 1-12-2020]. Disponible en: http://hdl.handle.net/20.500.11797/TFG2027

[22] Arbona, C. B., García-Palacios, A. y Baños, R. M. *Realidad virtual y tratamientos psicológicos*. Editorial Médica, 2007.

[23] López-Soler, C., et al. Sistema de realidad virtual EMMA-Infancia en el tratamiento psicológico de un menor con estrés postraumático. En: *Revista de psicopatología y psicología clínica* [en línea], 2011, vol. 16, no. 3, pp. 189-206 [consulta: 3-12-2020]. Disponible en: https://doi.org/10.5944/rppc.vol.16.num.3.2011.10361

[24] López-Martín, O., et al. Efectividad de un programa de juego basado en realidad virtual para la mejora cognitiva en la esquizofrenia. En: *Gaceta Sanitaria* [en línea], 2016, vol. 30, no. 2, pp. 133-136 [consulta: 5-12-2020]. Disponible en: https://doi.org/10.1016/j.gaceta.2015.10.004

[25] Pitti, C. T., et al. Agorafobia: tratamientos combinados y realidad virtual. Datos preliminares. En: *Actas Españolas de Psiquiatría* [en línea], 2008, vol. 36, no 2, pp. 94-101 [consulta: 7-12-2020]. Disponible en: https://www.actaspsiquiatria.es/repositorio/9/50/ESP/9-50-ESP-94-101-512050.pdf

[26] Riva, G., et al. Affective interactions using virtual reality: The link between presence and emotions. *CyberPsychology & Behavior* [en línea], 2007, vol. 10, no. 1, pp. 45-56 [consulta: 9-12-2020]. Disponible en: https://doi.org/10.1089/cpb.2006.9993

[27] Baños, R. M., et al. Presence and emotions in virtual environments: The influence of stereoscopy. *CyberPsychology & Behavior* [en línea], 2008, vol. 11, no. 1, pp. 1-8 [consulta: 10-12-2020]. Disponible en: https://doi.org/10.1089/cpb.2007.9936

[28] Allcoat, D. & Von Mühlenen, A. Learning in virtual reality: Effects on performance, emotion and engagement. *Research in Learning Technology* [en línea], 2018, vol. 26 [consulta: 24-3-2021]. Disponible en: https://doi.org/10.25304/rlt.v26.2140

[29] K. Hidaka, H. Qin & J. Kobayashi, "Preliminary test of affective virtual reality scenes with head mount display for emotion elicitation experiment" 2017 17th International Conference on Control, Automation and Systems (ICCAS), Jeju, 2017, pp. 325-329, doi: 10.23919/ICCAS.2017.8204459. [consulta: 24-3-2021]. Disponible en: https://ieeexplore--ieee--org.us.debiblio.com/document/8204459

[30] Manrubia Pereira, A. M. El proceso productivo del videojuego: fases de producción. En: *Historia y comunicación social* [en línea], 2014, vol. 19, pp.791-805 [consulta: 12-12-2020]. Disponible en: http://www.academia.edu/download/45071602/VIDEOJUEGOS_PDF.pdf

[31] Unity Asset Store. *Unity Technologies*, 2018 [consulta: 14-12-2020]. Disponible en: https://assetstore.unity.com/publishers/1

[32] Oculus Rift S PC-Powered VR Gaming Headset [Internet]. [consultado el 5 de diciembre de 2020]. Disponible en: https://images-na.ssl-images-amazon.com/images/I/71byGcUN9iL._AC_SL1500_.jpg

[33] Oculus Touch Controller [Internet]. [consultado el 5 de diciembre de 2020]. Disponible en: https://images-na.ssl-images-amazon.com/images/I/71WtgfRInqL._AC_SL1500_.jpg

[34] Headband Adapter for Oculus Rift-S [Internet]. [consultado el 5 de diciembre de 2020]. Disponible en: https://scontent.oculuscdn.com/v/t64.5771-25/38974805_513384172528449_3972989587982123008_n.jpg?_nc_cat=104&ccb=1-3&_nc_sid=ad8a9d&_nc_ohc=jtRJShnnv4IAX-rh8GU&_nc_ht=scontent.oculuscdn.com&oh=922e13ae5f24ef529d6045829d7ee2dd&oe=6081931E

[35] Cover Facial Interface for Oculus [Internet]. [consultado el 5 de diciembre de 2020]. Disponible en: https://scontent.oculuscdn.com/v/t64.5771-25/38974654_2219429185039184_1044500655856680960_n.jpg?_nc_cat=108&ccb=1-3&_nc_sid=ad8a9d&_nc_ohc=vZT8jYG3j30AX_aVfT3&_nc_ht=scontent.oculuscdn.com&oh=b7e7e8165658cf420883bb1178793aa8&oe=60814BC2

[36] Optical Headset Cable for Oculus [Internet]. [consultado el 5 de diciembre de 2020]. Disponible en: https://scontent.fsvq1-2.fna.fbcdn.net/v/t39.2365-6/74795901_1284049065089757_475737756687505817 6_n.jpg?_nc_cat=105&ccb=1-3&_nc_sid=ad8a9d&_nc_ohc=YFyaq8f464MAX-RvGkd&_nc_ht=scontent.fsvq1-2.fna&oh=393d636e4a3c6d6554f2191bee410b7a&oe=60803A41

[37] Oculus Lens [Internet]. [consultado el 5 de diciembre de 2020]. Disponible en: https://scontent.fsvq1-1.fna.fbcdn.net/v/t39.2365-6/32739098_1189432057857913_5676087108897341440_n.jpg?_nc_cat=103&ccb=1-3&_nc_sid=ad8a9d&_nc_ohc=QBpjNzDmmEMAX8DG9ou&_nc_ht=scontent.fsvq1-1.fna&oh=f7c9ad39f7ba569247defbaf092d8ccf&oe=60808AB3

[38] Facebook Technologies, LLC. *Oculus.* [consulta: 5-12-2020]. Disponible en: https://www.oculus.com/rift-s/#rift-s-pc-requirements/

[39] Oculus Rift. *Manual de salud y seguridad de Oculus Rift S*. [Consulta: 16-12-2020]. Disponible en: https://www.oculus.com/safety-center/rift-s/?locale=es_ES

[40] Protective Hygiene Eye Mask for Oculus [Internet]. [consultado el 5 de diciembre de 2020]. Disponible en: https://images-na.ssl-images-amazon.com/images/I/41hy3w3y9FL._AC_SX355_.jpg

[41] UNITY TECHNOLOGIES, INC. *Terrain Tools Sample Asset Pack 1.0* [extension asset]. May 22, 2019 [consulta: 13-5-2020]. Disponible en: https://assetstore.unity.com/packages/2d/textures-materials/nature/terrain-tools-sample-asset-pack-145808

[42] UNITY TECHNOLOGIES, INC. *Fantasy Forest Environment - Free Demo 2.0* [extension asset]. Jul 23, 2019 [consulta: 20-5-2020]. Disponible en: https://assetstore.unity.com/packages/3d/environments/fantasy/fantasy-forest-environment-free-demo-35361#content

[43] UNITY TECHNOLOGIES, INC. *Grass Flowers Pack Free 1.0* [extension asset]. Feb 5, 2019 [consulta: 31-5-2020]. Disponible en:

https://assetstore.unity.com/packages/2d/textures-materials/grass-flowers-pack-free-138810

[44] UNITY TECHNOLOGIES, INC. *White Rabbit 1.0* [extension asset]. May 30, 2019 [consulta: 10-6-2020]. Disponible en: https://assetstore.unity.com/packages/3d/characters/animals/white-rabbit-138709

[45] UNITY TECHNOLOGIES, INC. *Butterfly with Animations 1.0* [extension asset]. [consulta: 15-6-2020]. Disponible en: https://assetstore.unity.com/packages/3d/characters/animals/insects/butterfly-with-animations-20985

[46] UNITY TECHNOLOGIES, INC. *Simple Boids (Flocks of Birds, Fish and Insects)1.1.1* [extension asset]. Apr 1, 2020 [consulta: 2-7-2020]. Disponible en: https://assetstore.unity.com/packages/3d/characters/animals/nvjob-simple-boids-flocks-of-birds-fish-and-insects-164188

[47] UNITY TECHNOLOGIES, INC. *Standard Assets 1.1.6* [extension asset]. Apr 8, 2020 [consulta: 18-7-2020]. Disponible en: https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-2018-4-32351

[48] UNITY TECHNOLOGIES, INC. *Oculus Integration 23.1* [extension asset]. Dec 22, 2020 [consulta: 1-6-2020]. Disponible en: https://assetstore.unity.com/packages/tools/integration/oculus-integration-82022

[49] UNITY TECHNOLOGIES, INC. *Animated Cartoon Treasure Chest 1.0* [extension asset]. Jun 17, 2019 [consulta: 23-10-2020]. Disponible en: https://assetstore.unity.com/packages/3d/environments/fantasy/animated-cartoon-treasure-chest-146757