Universidad Zaragoza
1542

Trabajo Fin de Grado

Videojuego de realidad mixta adaptado a unas gafas de realidad virtual de bajo coste

Autor

Daniel Enrique Rodero Domingo

Director

Rubén Béjar Hernández

Escuela de Ingeniería y Arquitectura – Universidad de Zaragoza

2015

Quiero dedicarle este trabajo a mis padres por su apoyo y confianza,
a mis compañeros por todos los momentos que hemos pasado y
al profesor Rubén Béjar por su ayuda y colaboración.

Resumen

En 1994 Paul Milgram y Fumio Kishino definieron el concepto de realidad mixta como cualquier espacio entre los extremos del continuo de la virtualidad. Este continuo de la virtualidad se extiende desde el mundo completamente real hasta el entorno completamente virtual, encontrándose entre medio de estos la realidad aumentada y realidad virtual. En términos más generales, la realidad mixta consiste en combinar mundos virtuales con el mundo real a tiempo real. En los últimos años, lo que era un sueño se ha convertido en realidad gracias a las revoluciones tecnológicas que el hombre ha sido capaz de lograr. Esto unido a las numerosas investigaciones realizadas ha dado lugar a una serie de aplicaciones, casi todas ellas en la industria del entretenimiento y las artes, aunque también ha logrado expandirse hacia el mundo de los negocios y la educación.

Este trabajo pretende investigar en esta nueva tecnología y hacer que personas sin unos grandes recursos, en cuanto a material tecnológico se refiere, puedan disfrutar de ella. Por ello se ha creado una aplicación que mezcla la realidad mixta mediante el uso de una herramienta que hoy en día posee todo el mundo, como es un teléfono móvil, y un marco de gafas de bajo coste que puede encontrarse en muchos lugares de Internet e incluso puede fabricarse uno mismo.

La aplicación desarrollada es un juego capaz de captar el mundo real a través de la cámara del teléfono móvil y combinarlo con elementos virtuales insertados mediante un entorno de desarrollo específico. El resultado es una recreación de la escena de la película *"King Kong"* donde el simio que se encuentra en lo alto del Empire State Building debe defenderse del ataque de los aviones que pretenden acabar con él. Las técnicas implementadas van desde la captura de vídeo mediante el teléfono y de los movimientos de cabeza que el usuario puede realizar como si se tratase del simio, hasta la funcionalidad de cada uno de los modos de juego con todas las partes que esto conlleva (aparición de enemigos, gestión de elementos del juego como por ejemplo el tiempo, etc). Asimismo se han realizado una serie de pantallas compuestas totalmente por elementos gráficos que ayudan al usuario mientras está jugando. En definitiva, el objetivo es investigar en el alcance de la realidad mixta y en la inmersión por parte del jugador en un mundo en el que se encuentra familiarizado, pero que tiene elementos virtuales superpuestos y con los cuales puede interactuar.

La metodología empleada para la realización del proyecto se ha basado en la utilización de prototipos iterativos cada vez más complejos. Estos prototipos han ido incrementando y creciendo en complejidad conforme el trabajo progresaba y se creaban nuevas escenas y/o pantallas. Simultáneamente se ha ido elaborando la documentación del sistema como en cualquier proyecto software.

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Daniel Enrique Rodero Domingo,

con nº de DNI 73007489S en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Videojuego de realidad mixta adaptado a unas gafas de realidad virtual de bajo coste,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 25 de Junio de 2015

Fdo: Daniel Enrique Rodero Domingo

Índice

Índice de figuras

1. Introducción

El presente documento tiene como objetivo recopilar toda la información referente a la realización de este Trabajo Fin de Grado (TFG) titulado *Videojuego de realidad mixta adaptado a unas gafas de realidad virtual de bajo coste*. A continuación se introduce al lector en el mundo de la realidad virtual y la realidad mixta explicando los objetivos planteados en el proyecto. Posteriormente se explica la organización del resto del documento.

1.1. Objetivo, contexto y alcance

En los últimos años han aparecido los términos de *realidad mixta* y *realidad virtual*. La primera de ellas hace referencia a la combinación de mundos virtuales con el mundo real y todo ello a tiempo real. Todo esto permite crear nuevos espacios en los que la interacción, tanto de objetos y/o personas reales como virtuales, es posible. En definitiva, se puede considerar como una mezcla entre la realidad, realidad aumentada y realidad virtual. Como se puede observar, la realidad virtual está presente de algún modo en la realidad mixta, pero, ¿qué es esta realidad virtual?. Podría decirse que es un entorno de escenas u objetos de apariencia real, generado mediante tecnología informática, que crea en el usuario la sensación de estar inmerso en él. Añadir que dicho entorno es contemplado generalmente por el usuario a través de un dispositivo conocido como gafas o casco de realidad virtual, y que puede ir acompañado por otros dispositivos para su interacción. En este TFG se abordará el diseño y desarrollo de un videojuego para móvil adaptando su uso a unas gafas de realidad virtual. Éste presentará un entorno de realidad mixta, donde el fondo será el entorno real del usuario y los elementos que aparezcan serán objetos 3D virtuales que se superpondrán sobre este fondo. Dicho videojuego se inspirará en la escena de la película *"King Kong"* situada en el Empire State Building donde el simio tiene que defenderse de los aviones enemigos que pretenden acabar con él. Como se puede observar, se pretende tomar el mundo que puede ser generado mediante realidad mixta, a través de la cámara del móvil y la aparición de nuevos objetos 3D, y crear una inmersión del usuario adaptando esta tecnología a unas gafas de realidad virtual de bajo coste.

1.2. Estructura del documento

El presente documento se compone de 6 capítulos. El primero de ellos pone en situación al lector y se introduce cual es el objetivo del TFG. En el capítulo 2 se profundiza en dicho objetivo y se sugieren algunas alternativas con las que se podría haber trabajado en el proyecto. El diseño

de la solución es presentado en el capítulo 3 donde se muestra la arquitectura definitiva de la aplicación y cómo interactuan cada una de sus partes. El proceso de trabajo seguido se detalla en el capítulo 4, y por último en los capítulos 5 y 6 se muestras los resultados y las conclusiones obtenidas junto con las líneas de trabajo futuras y la valoración personal del trabajo realizado.

Añadir que la memoria contiene una serie de anexos que complementan la información descrita en el párrafo anterior:

- Anexo **A**. Requisitos del sistema: En este anexo se detallan los requisitos que debe satisfacer la aplicación creada, tanto funcionales como no funcionales.
- Anexo **B**. Dispositivos empleados para probar el proyecto: En este anexo se describen cada uno de los dispositivos hardware empleados para realizar las pruebas de la aplicación creada.
- Anexo **C**. Diseño del Sistema: En este anexo se especifican detalles sobre el diseño de la aplicación implementada.
- Anexo **D**. Instrucciones para subir una aplicación a *Google Play*: En este anexo se detallan los pasos que un usuario debe seguir para poder subir su aplicación a la store de *Android*.
- Anexo **E**. Manual de usuario: En este anexo se detalla un manual el cual un usuario debe seguir para el correcto manejo de la aplicación.

2. El problema

Como se ha introducido en el capítulo anterior el problema o tema que se aborda es la implementación de un juego basado en realidad mixta que es visualizado en un dispositivo, unas gafas en este caso, de realidad virtual.

2.1. Descripción

Desde hace tiempo las personas han querido emular el cine de ciencia ficción y uno de los temas más destacados es la realidad virtual, donde Matrix, Desafío Total o Gamer son algunas de las numerosas películas donde se aborda. Hay que decir que esta tendencia tecnológica nunca llegó a ser el gran salto que todos esperábamos y que más bien se quedó en una curiosidad con contadas aplicaciones reales. Pero con el paso del tiempo, la llegada de la realidad aumentada, el empujón de Google Glass y la creciente popularidad de las Oculus Rift, parece que esa tecnología del futuro-pasado llamada Realidad Virtual vuelve a tener una oportunidad.

Todo el mundo cuando escucha cosas sobre la realidad virtual solo ve los aspectos positivos, pero ¿qué inconvenientes tiene?. Al igual que los simuladores de vuelo utilizados por los pilotos en el entrenamiento, la realidad virtual tiene el potencial de presentar desajustes entre las señales de movimiento físicas y visuales. Este desajuste conocido como "enfermedad simulador"puede producir nauseas, y esto se produce cuando tus ojos creen que estás en movimiento, pero tu cuerpo no lo está.

Como solución aparece el término realidad mixta donde el usuario no se ve envuelto en su totalidad por un ambiente completamente virtual, sino que se encuentra en un entorno más familiar como es el mundo real, pero visualizado a través de un dispositivo. De este modo se produce una inmersión en un mundo que no es creado digital o virtualmente pero que sí se puede interactuar con elementos de dicho mundo.

En definitiva, lo que se pretende con este TFG es investigar en el alcance de la realidad mixta y en la inmersión por parte del usuario en un mundo en el que se encuentra familiarizado, pero que tiene elementos virtuales superpuestos y con los cuales puede interactuar. En consecuencia se ha realizado un juego como bien se ha definido anteriormente, recreando la escena de la película *"King Kong"* donde éste deberá defenderse del ataque de los aviones enemigos. Para ello se van a implementar dos modalidades de juego, Arcade y Survival. En el primer modo, el jugador tendrá que destruir los enemigos que van apareciendo por oleadas en el tiempo indicado. El jugador ganará puntos conforme los vaya destruyendo y terminará el juego cuando acabe con

todas las oleadas, o no pueda acabar con todos enemigos en una oleada concreta. Por otro lado, en el segundo modo, el jugador deberá destruir cuantos enemigos pueda antes de que estos le quiten todas las vidas que posee. El juego acaba cuando el jugador destruye a todos los enemigos que aparecen, o se queda sin vidas. A parte de esto, se incluirán diferentes pantalla de ayuda para el usuario y se podrán ver las tres mejores puntuaciones de cada modo de juego.

2.2. Requisitos

La aplicación creada debe cumplir con una serie de requisitos. Dichos requisitos engloban tanto las tareas que deben poder llegar a realizarse con nuestra aplicación, como las posibles restricciones impuestas a nivel de implementación y de hardware. Haciendo un rápido resumen, los requisitos funcionales que son impuestos en nuestro sistema tienen que ver con todas las opciones que debe ofrecer nuestro juego, ya sea a nivel de jugabilidad (modos de juego) como a nivel informativo (puntuaciones y ayuda). Por otro lado, los requisitos no funcionales abarcan desde la creación de diversas clases que serán usadas durante la ejecución del juego (headtracking, pantalla dividida, etc.) hasta restricciones en la aplicación como el uso de un sistema operativo determinado o que el dispositivo móvil utilizado posea ciertas tecnologías (giróscopo, bluetooth, etc.)

Para poder observar estos requisitos de manera más detalla ir al Anexo **A**.

2.3. Alternativas tecnológicas existentes

En consecuencia al boom generado, se han ampliado los campos de investigaciones de este sector informático, dando lugar a la aparición de diversas tecnologías tanto a nivel de software, como de hardware.

Así pues podemos apreciar una amplia variedad de dispositivos hardware que para lo que nos concierne, haría referencia al marco de las gafas virtuales empleadas. En este aspecto cabe destacar que en el TFG realizado, y como en su propio título se puede leer, se busca una tecnología de bajo coste. Debido a esta restricción el pionero en el campo de cascos o gafas de realidad virtual *Oculues Rift* queda totalmente descartado. En detrimento, otros dispositivos que podrían haber sido empleados en el TGF serían las gafas de Realidad Virtual *ColorCross* o las *Google Cardboard*.

Por otro lado, a nivel de software para el desarrollo de aplicaciones de esta índole también han aparecido muchos competidores. La diferencia principal entre estos motores de juegos es la licencia, es decir, el tener que pagar o no. En consecuencia *Unity* se ha convertido en el motor de juego más famoso para todos los programadores y no sólo por su versión gratuita, sino porque incluso los programadores no interesados en la creación de juegos pueden utilizarlo. *Epic* y *Valve* son algunas de las alternativas presentes, pero hasta hace pocos meses estos motores de juegos eran de pago.

En último lugar mencionar que hay diversos plugin que son ofrecidos por empresas con renombre como por ejemplo *Android* o *Durovis*, las cuales ayudan a la creación de aplicaciones

(mediante APIs), centrándose sobre todo en el *headtracking* o movimiento de la cabeza, lo que facilita mucho las cosas.

3. Diseño de la solución

En este capítulo se va a detallar el diseño realizado durante este TFG tras haber realizado un análisis previo. En la Sección 3.1 se mencionan las tecnologías y herramientas empleadas, mientras que en las Secciones 3.2 y 3.3 se explica la arquitectura de la aplicación, indicando el papel que desempeña cada una de sus partes y como interactuan entre ellas. Así pues dentro de esta ultima sección, el Apartado 3.3.1 incluye una descripción de las clases del sistema, es decir, la parte estática del mismo, mientras que lo concerniente a la parte dinámica del sistema queda plasmado en el Anexo **C**.

3.1. Tecnologías y herramientas utilizadas

A continuación se muestran las diversas tecnologías y herramientas las cuales se han empleado en la realización del proyecto. Dichas tecnologías y herramientas han sido agrupadas a nivel de hardware y a nivel de software. En el primer caso se ha hecho uso de un teléfono móvil, un mini teclado bluetooth y un marco de gafas de realidad virtual (Ver Anexo **B** para más información sobre los dispositivos usados para probar la aplicación). Por otro lado, en el segundo caso, se ha empleado el entorno de desarrollo *Unity* y el lenguaje de programación C# para confeccionar y crear la aplicación de nuestro proyecto. Ambas herramientas son detalladas a continuación.

3.1.1. Unity

Unity es un framework o plataforma de desarrollo flexible y poderosa para crear juegos y experiencias interactivas 3D y 2D multiplataforma, en otras palabras, un motor de vieojuegos. Fue creado por Unity Technologies y ofrece un ecosistema completo para todo aquel que busque desarrollar un negocio a partir de la creación de contenido de alta gama y conectarse con sus jugadores y clientes más fieles y entusiastas.

Se optó por este entorno de desarrollo principalmente por su amplia comunidad de usuarios y debido a que ofrece una versión del mismo totalmente gratuita, en contraposición a sus competidores como *Valve* y *Epic* que han sido mencionados en la Sección 2.3.

3.1.2. C#

Es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que posteriormente fue aprobado como un estándar. C# es simple y eficaz, y permite desarrollar aplicaciones rápidamente y mantener la expresividad y elegancia de los lenguajes de estilo de C.

La elección de este lenguaje de programación fue debido a que era uno de los lenguajes ofrecidos por el entorno de desarrollo *Unity*, además de UnityScript inspirado en ECMAScript y Boo inspirado en Python. Como bien se ha mencionado en el párrafo anterior está orientado a objetos al igual que Java, lenguaje que más domino debido a que ha sido el que más he usado en mi carrera, comparte esta característica y por lo tanto se ha optado por él.

3.2. Mapa de navegación

La primera idea sobre el juego fue la de un funcionamiento básico, algo rápido pero eficaz, donde el usuario pudiera experimentar con esta tecnología. Una vez llevada a cabo la implementación de esta primera idea se observó que los usuarios que jugaban por primera vez al juego y que usaban esta tecnología andaban un poco perdidos. En consecuencia se crearon unas pantallas previas de ayuda dando como resultado la Figura 3.1 mostrada a continuación, en la cual, se puede observar con más precisión el funcionamiento y navegación dentro del juego.

El juego arranca con una pantalla de *Unity* y es debido a que se ha usado la versión gratuita del mismo y en consecuencia esta pantalla no se puede quitar. A continuación aparecen dos pantallas informativas donde se explica al jugador que debe hacer antes de comenzar a jugar. Una de ellas indica como debe ponerse a punto el marco de gafas y el móvil, y en la otra se muestran los controles que serán usados durante el juego. Cuando esta información ha quedado clara se da un pequeño tiempo al usuario para preparase y mientras se muestra una pantalla donde se indica que se esta cargando el juego. A partir de este instante el jugador ya se ha colocado las gafas con el dispositivo móvil en su interior y ha conectado el teclado bluetooth para comunicarse con el juego, apareciendo así un menú donde el usuario podrá elegir qué desea hacer. Este menú muestra cinco opciones donde las dos primeras permiten jugar a dos modos de juego, Arcade y Survival. La tercera y cuarta son informativas en las cuales se puede leer de qué trata cada modo de juego y lo que se debe hacer, y de las tres mejores puntuaciones de cada uno de ellos. Por último se da la opción de salir del juego ya que la inaccesibilidad al móvil, debido a que se encuentra en el interior del marco de gafas, nos hace imposible darle al botón de volver que ofrece éste. Esta pequeña explicación hace referencia al funcionamiento principal del juego y las pantallas que vamos a encontrarnos si todo va bien. En caso de que haya algún problema, como la falta de alguna de las tecnologías que son usadas en el teléfono móvil, como por ejemplo el giroscopio, se mostrará una pantalla donde se informará del error y se indicará que hacer para salir del juego.

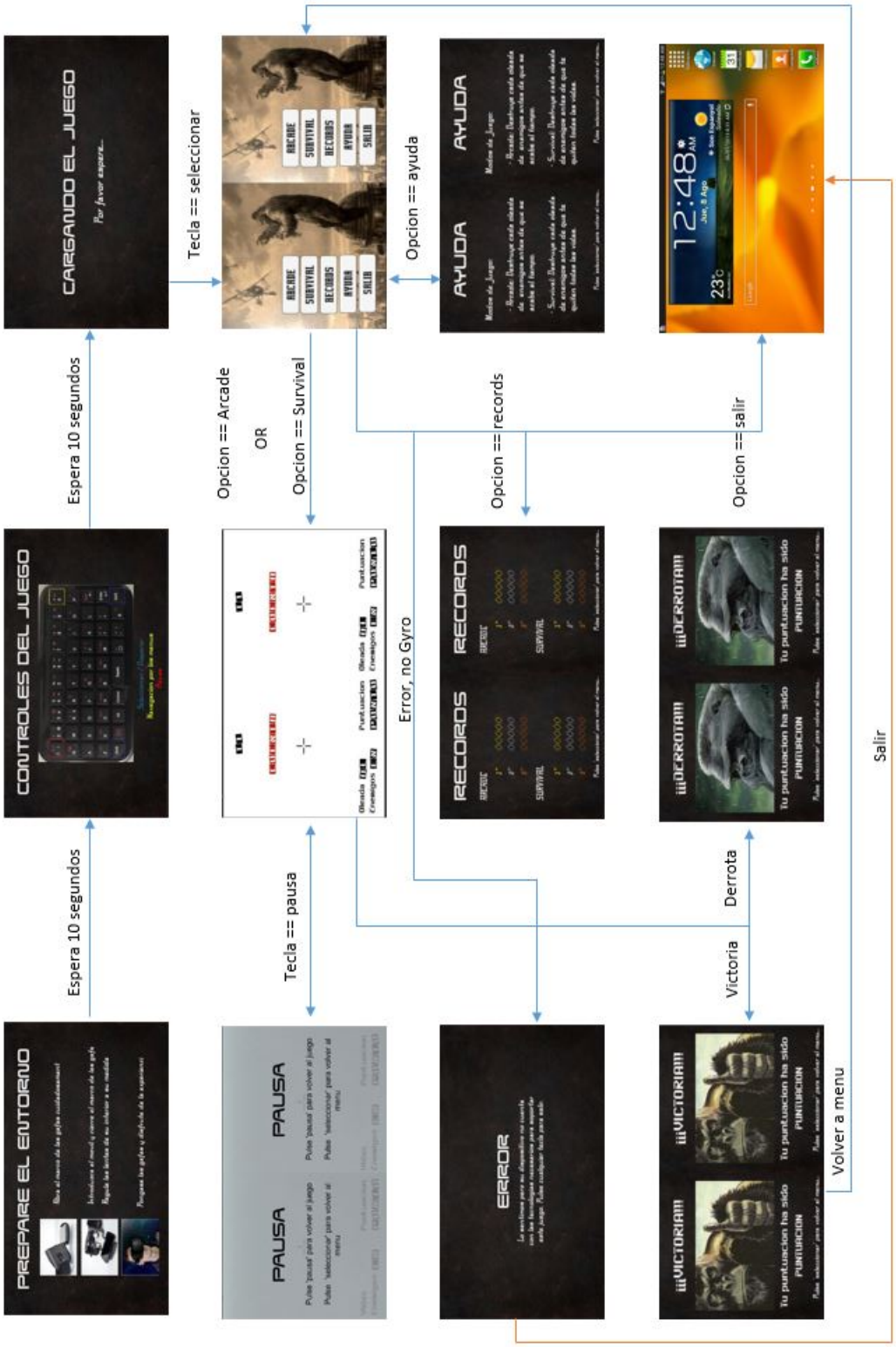

Figura 3.1: Mapa de navegación donde se puede observar las pantallas que hay en el juego con la navegación que puede realizarse en función de la opción seleccionada por el jugador.

3.3. Modelo de la aplicación

En primer lugar, y para facilitar la comprensión de la estructura y arquitectura de la aplicación creada, se muestra un pequeño diagrama de paquetes (ver Figura 3.2) donde todo queda englobado en dos *namespace* llamados *gui* y *controlador*. Como sus propios nombres indican, en cada uno de ellos se encuentran las escenas creadas que hacen referencia a todo lo relacionado con la *GUI* (Interfaz gráfica de usuario) del juego y a su funcionalidad o control respectivamente. Añadir que se ha representado también *MonoBehaviour* ya que es la clase base que automáticamente extiende *Unity*.

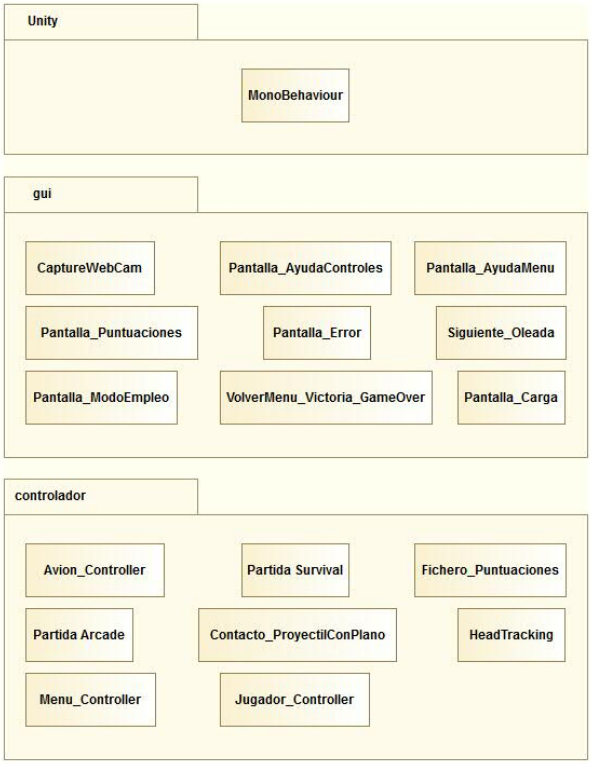

Figura 3.2: Diagrama de paquetes que representa cada una de las clases implementadas bajo el mismo *namespace*.

3.3.1. Diagrama de clases

El juego creado se compone de 18 clases, las cuales han sido implementadas desde cero, y que están asociadas con otros tantos elementos que también han sido creados aunque estos con ayuda del entorno de desarrollo utilizado. Hay que destacar que no todas las clases creadas contienen una parte vital de la funcionalidad del juego, sino que más bien estas clases se centran en mostrar información de manera gráfica, y por consiguiente la parte que verdaderamente ha sido implementada está en la escena creada con *Unity*. Esto sucede en la mayoría de juegos del mercado, puesto que hasta que no se llega a lo que es el juego en sí, lo que aparece son meras pantallas con información y animaciones.

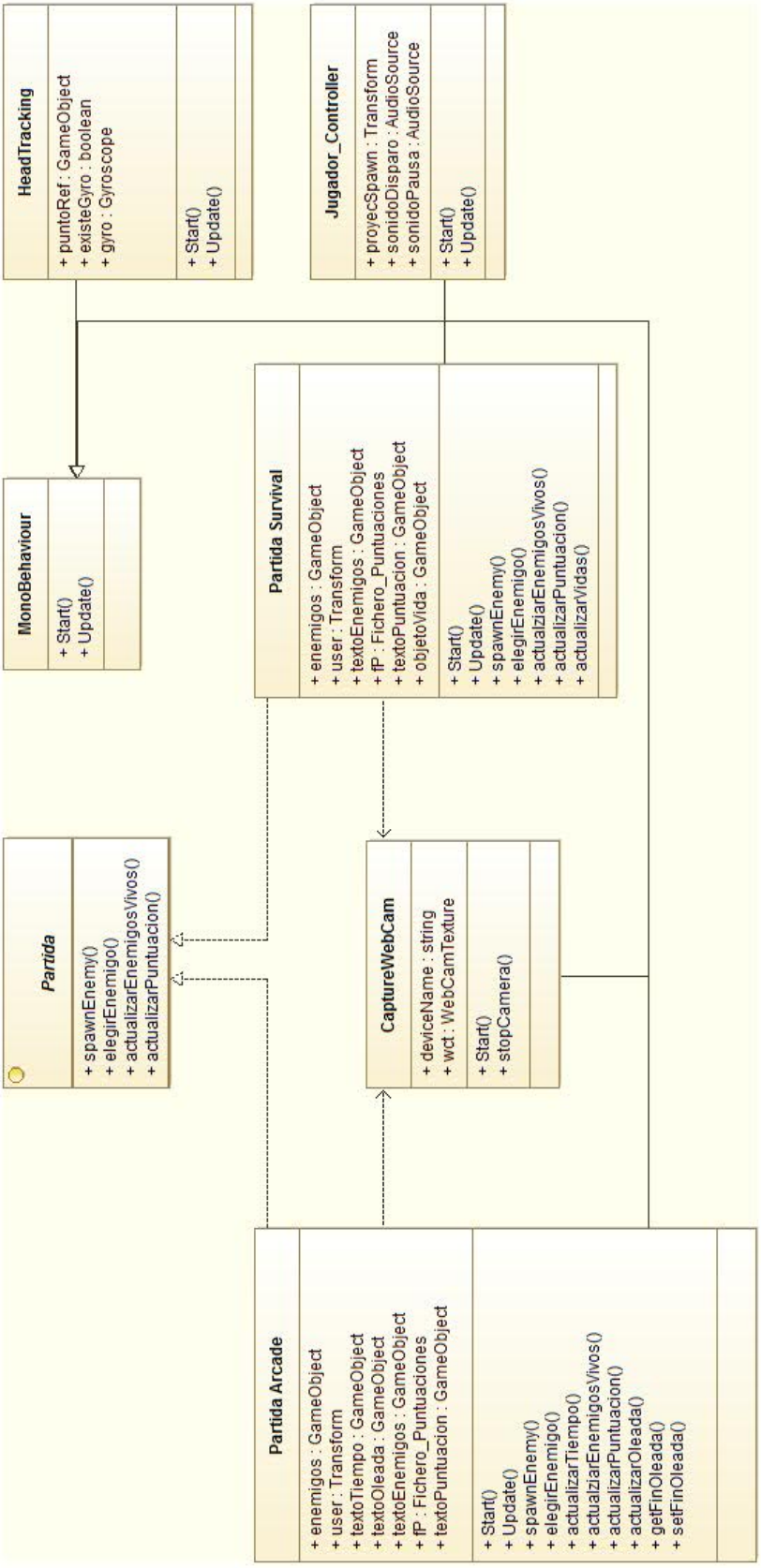

Figura 3.3: Diagrama de clases que representa cada una de las clases implementadas que hacen referencia a la funcionalidad principal del juego y la relación que hay entre ellas.

Para simplificar y centrarnos en las clases que verdaderamente contienen la información principal del juego, se ha creado un diagrama (ver Figura 3.3) donde queda plasmado su funcionamiento. Para ello las clases que aparecen son las propias de una partida dependiendo del modo de juego en el que el jugador se encuentra y los elementos a controlar en cada uno. A continuación se desglosan estos dos apartados y se explicará que función tiene cada una de ellas.

- Partida: Interfaz creada para la representación de las partidas, la cual ofrece las funciones básicas ha realizar en ella (crear enemigos, actualizar puntuación y actualizar contador de enemigos).
- Partida Arcade: Clase que hereda de Partida y que se encarga de la gestión del modo de juego Arcade. En ella se tratan todos los aspectos que ocurren durante la partida implementando varios métodos más aparte de los heredados.
- Partida Survival: Clase que hereda de Partida y que se encarga de la gestión del modo de juego Survival. En ella se tratan todos los aspectos que ocurren durante la partida implementando varios métodos más aparte de los heredados.
- Jugador Controller: Clase encargada de gestionar todas las acciones que puede realizar el jugador durante una partida (pausar el juego, disparar un proyectil).
- HeadTracking: Clase encargada del control de los movimientos de la cabeza del jugador y que son trasladados al control de la cámara del dispositivo móvil mediante el giroscopio que posee. Dicha clase está ejecutándose constantemente cada frame mientras la partida esta activa.
- CaptureWebCam: Clase encargada de la captura de vídeo mediante la cámara del dispositivo móvil empleado. Dicha captura es plasmada en una textura previamente asignada y en la cual se proyectará para su visualización.
- MonoBehaviour: Clase base de la que derivan todos los scripts o clases implementadas. Esta clase ofrece una serie de funciones ya creadas, de las cuales, la usadas para este proyecto son *Start()*, *Update()* y *OnTriggerEnter()*. La primera de ellas se ejecuta en el primer frame, lo que nos permitirá programar el comportamiento en el mismo. La segunda de ellas, se ejecuta en todos los frames, lo que nos permite programar las actualizaciones en el resto de ellos. y por último, la tercera, nos permite especificar el comportamiento cuando el objeto que lleva asociado el script entra dentro del área definida por un trigger.

4. Desarrollo del proyecto

En este capítulo se va a detallar el tipo de proceso de trabajo seguido, así como la planificación que ha sido llevada a cabo (ver Secciones 4.1 y 4.2).

4.1. Tipo de proceso de trabajo

La realización de este proyecto ha seguido un desarrollo iterativo y creciente, donde todo el peso del mismo forma un conjunto de tareas agrupadas en pequeñas etapas repetitivas. Dicho modelo consta de diversas etapas de desarrollo en cada incremento, las cuales se inician con el análisis y finalizan con la instauración y aprobación del sistema. Una vez queda claro el modelo de desarrollo seguido para este TFG se va a explicar en primer lugar el conjunto de tareas que han sido realizadas. Por un lado tenemos el aprendizaje de las tecnologías que han sido empleadas para la elaboración del proyecto, el cual ha sido un poco tedioso debido a la escasa experiencia previa (se ha seguido en muchos aspecto la metodología de prueba y error). En lo referente a la creación del juego (tras un primer análisis y boceto en sucio de como queríamos que quedase) y cómo bien se ha comentado en la Sección 3.2 lo primero que se realizó fue un juego con una funcionalidad básica, por lo que el primer paso fue crear una escena simple donde se implementaría el *headtracking*, es decir, la detección de movimientos de la cabeza del usuario. Tras esta tarea se creó otra nueva escena con el menú del juego y la implementación necesaria para seleccionar la opción que el usuario desease realizar. Por último se crearon cada uno de los modos de juego por separado con su respectiva funcionalidad y una nueva pantalla donde se mostraban las puntuaciones. Cuando esta primera etapa concluyó, se comprobó la correcta funcionalidad del sistema corrigiendo posibles errores y se analizaron posibles mejoras para su posterior realización. Cuando todo fue correctamente verificado se pasó a crear una serie de nuevas escenas antes de llegar al menú del juego ya que se creyó conveniente algunas aclaraciones y/o ayudas para el usuario. Para finalizar se refinó y mejoró el juego incluyendo animaciones, sonidos, efectos visuales, etc. y se llevó a cabo la revisión final del sistema.

4.2. Planificación

A continuación se plasma de manera gráfica en la Figura 4.1 el conjunto de actividades desarrolladas, así como su duración, de una manera aproximada. Este TFG comenzó a mediado de febrero y se ha finalizado a finales de junio, usando tanto días laborales como festivos.

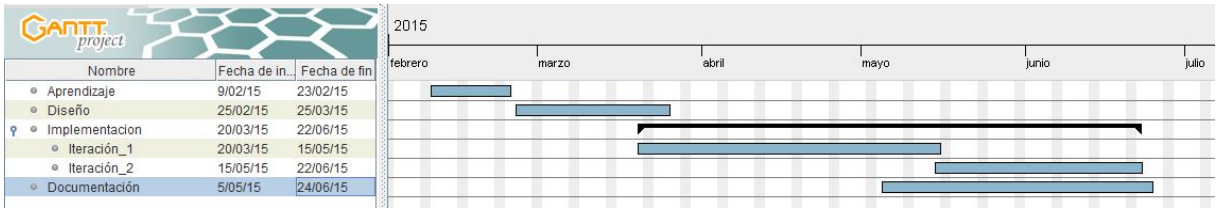

Figura 4.1: Diagrama de Gantt con la duración del proyecto y las tareas que se han realizado.

Como se puede observar se han desglosado las tareas que se han realizado durante el transcurso del TFG. Así pues durante la etapa de aprendizaje se hizo una toma de contacto con el entorno de desarrollo *Unity* y el lenguaje de programacion C# mediante tutoriales y guías encontradas en Internet. Posteriormente se llevó a cabo un análisis y diseño de la aplicación a crear, para tener un modelo que seguir en la posterior etapa de implementación. Por último se documentó todas las tareas realizadas en una memoria.

Finalmente el número de horas empleadas para la realización de este TFG ha sido de 334, las cuales han sido representadas en la Figura 4.2 atendiendo al desglose de tareas comentadas anteriormente.

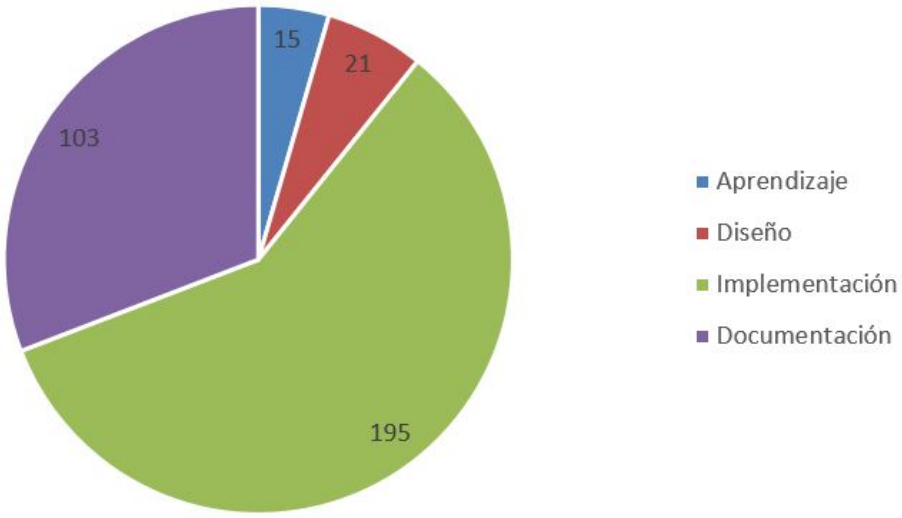

Figura 4.2: Gráfica con el número de horas invertidas en cada una de las partes de las que se ha compuesto este TFG.

5. Resultados

En este capítulo se van a detallar los resultados obtenidos, explicando los prototipos realizados. En el primer prototipo o primera iteración (Sección 5.1) se van a indicar todos los pasos realizados desde cero además de los problemas que se han encontrado y las soluciones que han sido adoptadas. Por otro lado en el segundo prototipo o segunda iteración (Sección 5.2) se mencionan los cambios realizados tanto a nivel de arreglo como de mejora. Por último, en la Sección 5.3 se detallan las pruebas que han sido realizadas para comprobar el correcto funcionamiento y calidad de la aplicación creada.

5.1. Primera iteración

La primera tarea que se abordó, una vez estudiado el entorno de desarrollo y el lenguaje de programación elegidos, fue la de crear el entorno que daría pie a todas las escenas del juego. Con crear el entorno se quiere decir que se crean las cámaras que tomarán las imágenes del juego (captura del móvil y objetos virtuales) y el plano donde se visualizará lo captado desde el teléfono. La captura de vídeo a través del dispositivo móvil y plasmarla en un plano con una textura auto-iluminada fue una tarea fácil, lo que no se puede decir lo mismo de las cámaras que iban a funcionar como los ojos del jugador. Para este asunto se trataron varias opciones en lo referente al número de cámaras a utilizar y la relación entre las distancias de estas y el plano de captura. La idea inicial y la que al final ha sido implantada fue la de usar dos únicas cámaras mediante las cuales capturar todos los elementos que aparecen en la escena. Se llegó a tantear la posibilidad de usar cuatro cámaras, dos para todo lo referente a elementos virtuales insertados en la escena y otras dos para visualizar el plano de captura, pero se vio innecesario ya que cuantas menos se utilizasen para realizar la misma función mucho mejor. En cuanto al tema de conjuntar plano y cámaras para que la relación de los objetos, en cuanto a tamaño se refiere, se viesen de manera natural dio algún que otro quebradero de cabeza ya que en un principio se intentó crear una relación entre las unidades que usa *Unity* y las que conocemos del mundo real. Se intentaron varias aproximaciones como la de una unidad de *Unity* igual a un metro en la realidad, etc. pero no llegó a ningún puerto. Al final se realizó una aproximación a ojo y se hizo que los elementos que apareciesen en la escena tuviesen un tamaño determinado nada que ver con las unidades del mundo real, pero que no resultasen molestos o extraños para el jugador. Por último y para poder crear el efecto 3D de los objetos virtuales que aparecían en la escena, se separaron las cámaras que capturaban la misma una respecto de la otra (0.1 en medidas de *Unity*), dando como resultado final la Figura 5.1.

(a) Pantalla partida Arcade (b) Pantalla partida Survival

Figura 5.1: Pantalla donde se puede observar como se vería el juego en el teléfono móvil. Mediante esta técnica el jugador tendrá una visión 3D de los elementos virtuales.

Cuando el entorno principal de las escenas (cámara y plano de captura) fue creado, se implementó un código propio para el tratamiento de los movimientos de la cabeza o *headtracking*. Como bien se comentó en la Sección 2.3 existen diversos *plugin* ofrecidos por las empresas que facilitan esta tarea pero se consideró como un requisito de nuestro juego que no se usarían y que sería obligación del estudiante crear una clase que lo gestionase. Tras analizar varias formas de hacerlo e informarnos de las herramientas que poseía un teléfono móvil, se llegó a la conclusión que el giroscopio, ya que era capaz de percibir la rotación, era mejor opción que el acelerómetro. Cuando se tuvo claro que herramienta se iba a usar, se llevaron a cabo diversas pruebas y al final el método que se uso para captar los movimientos que el jugador realizaba fue la obtener la diferencia entre dos posiciones. La primera de ellas era la que capturaba la posición del teléfono móvil en la pantalla de carga del juego, y la segunda era la que se tomaba dónde se encontraba el móvil en el momento de actualización del frame. De este modo la operación realizada era llevada a cabo en cada uno de los ejes del giroscopio y se aplicaban al objeto que contenía las cámaras de la escena. Como resultado, la visión del jugador rotaba como lo haría en el mundo real (ver Figura 5.2).

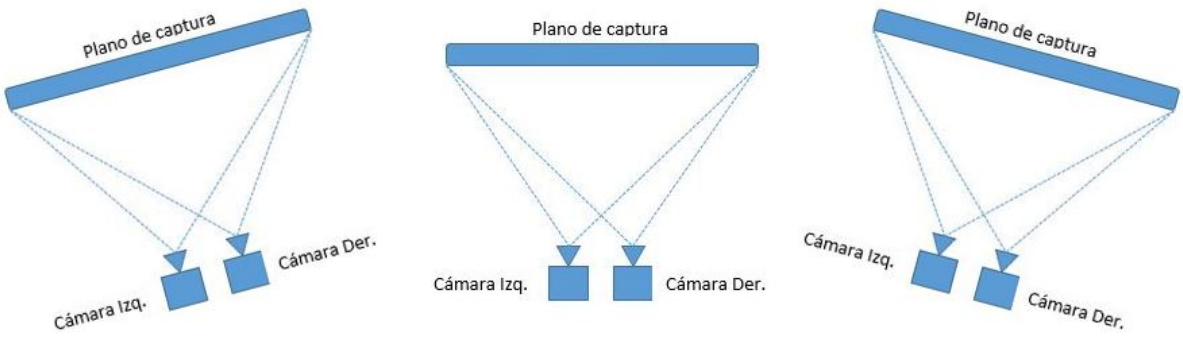

Figura 5.2: *Headtraking*. Entorno que muestra como es una escena básica donde se puede apreciar las cámaras que capturan todos los elementos que habrá en ella y el plano donde se proyectará el vídeo tomado por el teléfono móvil.

Una vez la estructura de cada una de las escenas del juego se terminó, se planteó la pri-

mera idea del juego a crear. Esta idea (ver Figura 5.3) fue la de construir un juego con un funcionamiento básico, algo rápido pero eficaz, donde el usuario pudiera experimentar con esta tecnología. De este modo, la idea principal partía con la carga de la pantalla de *Unity* (imposible de quitar debido a que se usa la edición gratuita) que posteriormente daría lugar al menú del juego. Tras él, el usuario podría elegir el modo de juego que desease jugar, una pantalla con las puntuaciones de cada uno y un botón para finalizar y salir ya que el móvil quedaría inaccesible en el marco de las gafas.

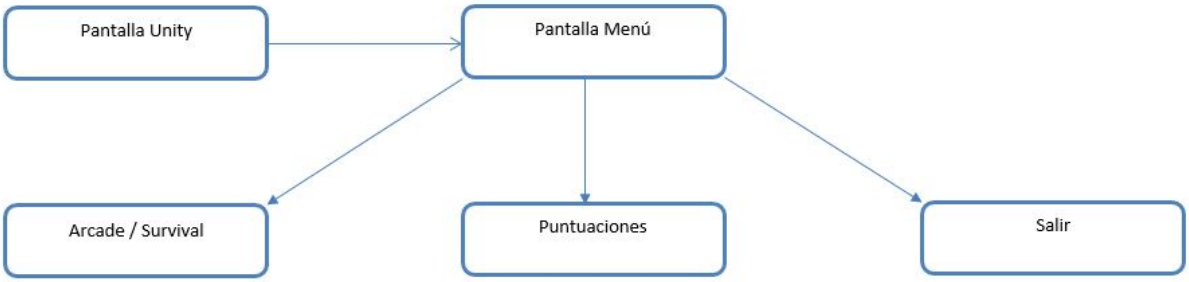

Figura 5.3: Boceto del juego donde se muestra cómo era la estructura inicial del juego creado donde se puede observar una funcionalidad muy básica y como se navega entre las escenas que hay.

Al final y como se explicó en la Sección 3.2, se vio necesario la creación de una serie de escenas adicionales para ayudar al jugador a saber que hacer en cada momento. Así pues, la estructura del juego creció con tres escenas adicionales desde el comienzo del juego hasta la aparición del menú, y se incluyó una nueva opción de ayuda en éste.

Con estas nuevas escenas y con el propio menú surgió una duda, ¿qué pasa si el juego es lanzado en un dispositivo con una resolución de pantalla diferente?. En una primera aproximación se empezó a utilizar la *GUI* que ofrecía *Unity*, mediante la cual, se podía crear e insertar elementos gráficos que posteriormente se configuraban si se quería desde la clase donde se utilizaban. Los resultados no fueron malos aunque era un trabajo laborioso refinar y perfeccionar cada elemento de modo que todo quedase perfecto sin atender a una única resolución. Por ello se buscó una alternativa y se encontró; con la llegada de la versión 4.6, *Unity* ofrece un nuevo sistema de interfaz de usuario (*UI*) para diseñar interfaces para un juego o aplicación. Con esta nueva herramienta, únicamente hay que configurar y crear una escena y automáticamente *Unity* la ajusta a la resolución del dispositivo donde es lanzada. Cabe destacar que sólo nos ayuda a la hora de ajustar los elementos gráficos a la resolución del dispositivo, pues el crear cada uno de ellos es igual a la herramienta utilizada en primer lugar.

Por último se llevo a cabo la implementación de cada uno de los modos de juego con sus consiguientes contratiempos que fueron solventados con éxito. Uno de ellos fue la creación de la interfaz de juego, y otro la elección de los objetos que representarían a los enemigos (aviones en este caso) y el lugar donde estos serían instanciados en la escena y cómo se moverían.

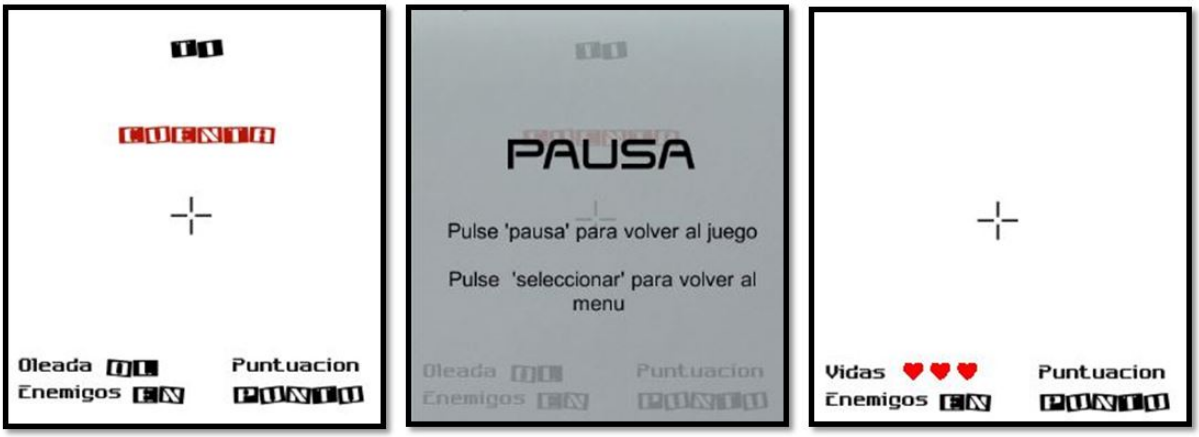

Figura 5.4: Interfaces de modo de juego Arcade, pausa y modo de juego Survival de izquierda a derecha respectivamente.

Para crear la interfaz (ver Figura 5.4) se optó por utilizar un nuevo plano, un componente "imagen"de la UI ofrecida por el framework *Unity* en este caso, el cual es situado justo delante del plano de captura donde se proyecta el vídeo tomado por el teléfono móvil. A su vez este plano lleva integrado una serie de componentes UI referentes a todos los elementos informativos de la partida: puntuación, enemigos, oleadas, tiempo, etc. Añadir que para el estado de pausa (ver Figura 5.4) se utilizó el mismo método que para la interfaz, solo que este plano tiene un fondo difuso y de un color más tupido. A continuación, en la Figura 5.5 se muestra como quedaría la escena en cuanto a los planos mencionados.

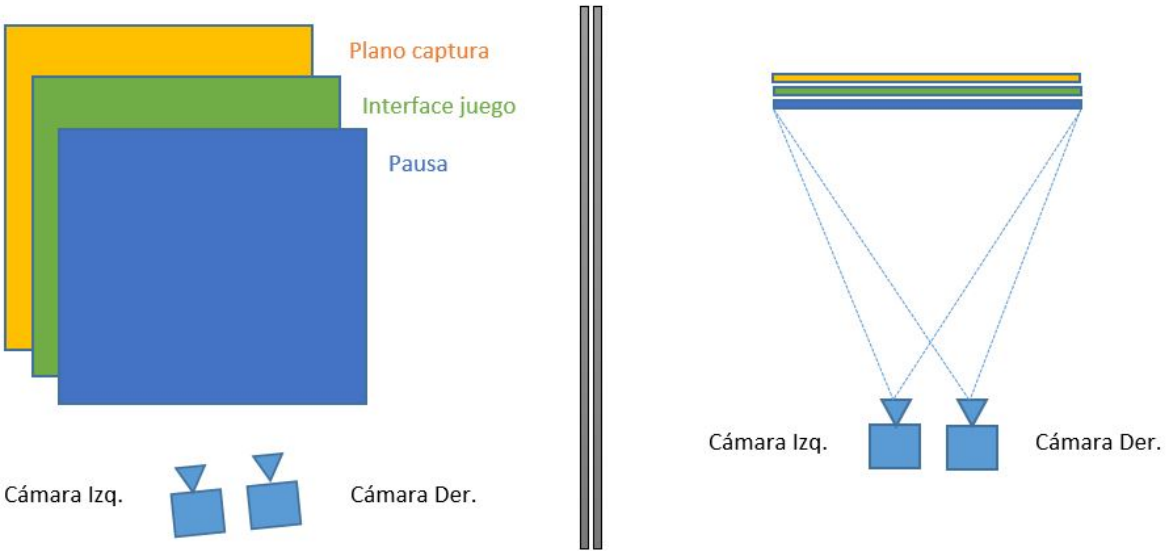

Figura 5.5: Representación de la situación de cada una de las interfaces creadas dentro de la escena. Indicar que la distancia entre ellas en la realidad es mínima pero se han separado un poco más en la imagen para una entendimiento y visión más clara.

El problema derivado de la elección de un modelo de avión para la representación de los enemigos era su tamaño en cuanto a espacio en bytes se refiere, es decir, objetos con un bajo

número de polígonos. El juego ha sido creado para ser lanzado en un teléfono móvil y no un ordenador o una videoconsola de última generación, por lo que los recursos consumidos mientras está en ejecución deben ser los mínimos posibles. En consecuencia se eligieron tres modelos diferentes de aviones antiguos (ver Figura 5.6) los cuales ocupan una quinta parte que otros modelos empleados durante el proyecto como explosiones o disparos ya que estos últimos tienen una vida muy fugaz durante el juego, todo lo contrario que los aviones.

Figura 5.6: Modelo de los aviones utilizados para representar los enemigos que aparecen en el juego.

En cuanto al tema de cómo los enemigos iban a ser instanciados y cómo se deberían mover había que analizarlo dependiendo del modo de juego donde nos encontrásemos. En el primer modo de juego (ver Figura E.7) y dado que el funcionamiento de éste era eliminar todas las oleadas de enemigos que apareciesen dentro de un tiempo estipulado, había que hacer que los aviones saliesen en diferentes posiciones para que no resultase monótono el estar esperando que apareciesen en el mismo sitio para destruirlos, y había que hacer que estos se moviesen alrededor del jugador. Así pues se estipuló un rango en los *ejes x,y,z* con unos valores *random* como *spawners*, o puntos de partida, donde los enemigos son creados y mediante la función *RotateAround* se hacía que estos girasen entorno a una referencia que en este caso eran las cámaras que hacen de ojos del jugador. Por último y para hacer el juego un poco más entretenido, mediante otro valor *random* se estipulaba hacia que lado iban a girar los aviones y así hacer un poco más difícil para el jugador eliminarlos.

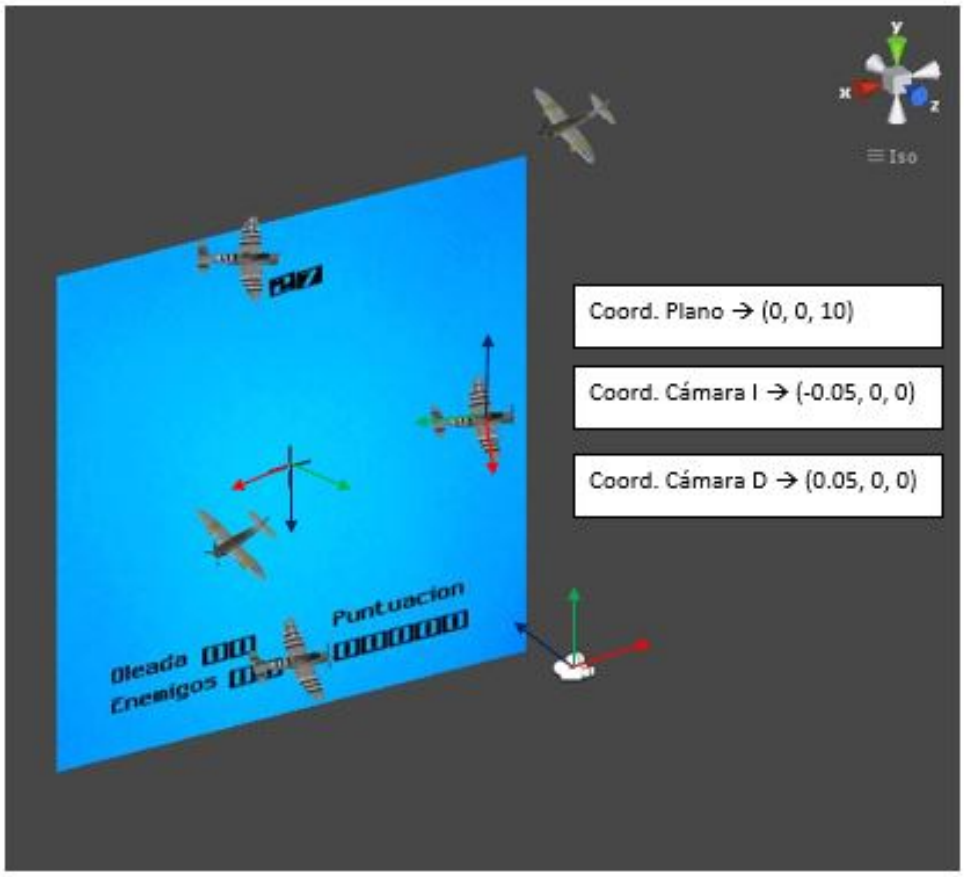

Figura 5.7: Muestra el lugar donde son instanciados los aviones en el modo de juego Arcade, así como el resto de elementos que hay en la escena junto con sus ejes y coordenadas.

En el segundo modo de juego (ver Figura E.8) la cosa cambiaba puesto que ahora la dinámica era destruir todos los enemigos que se acercaban hacia el jugador antes de quedarse sin vidas. La forma de instanciar los aviones era parecida al modo anterior pero con dos matices. Uno de ellos era que ahora un valor en uno de sus ejes era constante (*eje z*) debido a que los aviones se acercaban a ti y la distancia más alejada debía de ser constante. Esto era debido a que si estaba más alejada no se apreciaba nada y si estaba más cerca no le daba tiempo al jugador a eliminar el avión. En cuanto al movimiento se realizó un cálculo obteniendo la distancia normalizada entre el origen del avión donde era instanciado, y el destino. De este modo, usando este valor junto a la función *Translate*, hacía que el avión se moviese en dirección al jugador con la posición correcta del mismo.

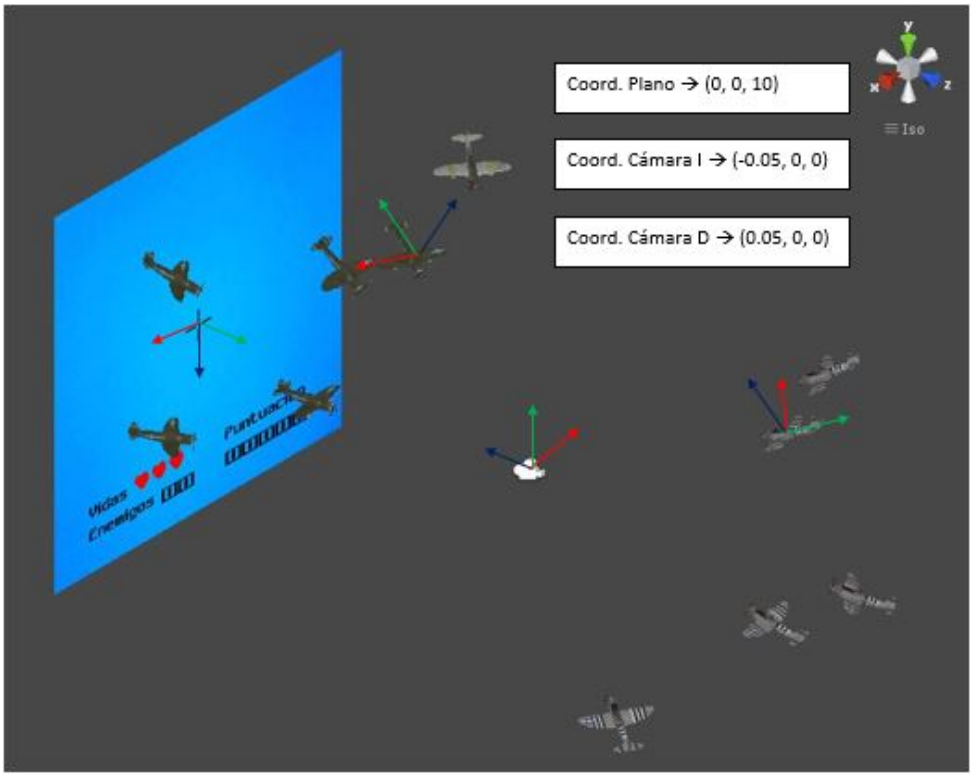

Figura 5.8: Muestra el lugar donde son instanciados los aviones en el modo de juego Survival, así como el resto de elementos que hay en la escena junto con sus ejes y coordenadas.

Dentro de la implemenatación de los modos de juego apareció una última cuestión referente a las puntuaciones que se obtenían. Dicha cuestión hacia referencia a su modo de almacenamiento pues era necesario tener un lugar donde se almacenasen las tres mejores puntuaciones y que no fuesen borradas cada vez que un jugador accedía al juego. En consecuencia se barajaron dos opciones: un fichero o una base de datos. *Android* ofrece ambas posibilidades pero dado que únicamente se van almacenar seis datos referentes a las tres mejores puntuaciones de cada uno de los modos de juego, se consideró que una base de datos era innecesaria en cuanto a funcionalidad y comodidad se refiere. De este modo se ha empleado un fichero el cual se almacena en una ruta específica del teléfono que es persistente, es decir, el fichero no será borrado a no ser que el usuario acceda a él y lo elimine.

Una vez que todo este proceso finalizó, se llevaron a cabo una serie de pruebas para comprobar su correcto funcionamiento.

5.2. Segunda iteración

Como era de esperar el trabajo realizado tenía diversos errores, en su mayoría condiciones mal gestionadas en el código, los cuales fueron solventados con más o menos prontitud. Tras un pequeño período de pruebas y errores, se consiguió tener el juego totalmente operativo en cuanto a la funcionalidad inicial se refería. Fue a partir de este momento cuando todas las tareas

que se realizaron fueron de mejora y refinamiento de lo que se había realizado. En un primer lugar se reestructuró el código creando algunos *namespaces* que agrupaban las clases referentes a la parte gráfica y a "controladores"del juego. Por otro lado se creó una interface *Partida* de la cual heredan *Partida_Arcade* y *Partida_Survival*, y se crearon dos nuevas clases para el control de todo lo que hace referencia a los aviones y el fichero que almacena las puntuaciones. En definitiva, el código quedó mucho más claro donde con un simple vistazo se podía intuir cual era su función sin abrir cada una de ellas.

Figura 5.9: Muestra dos imágenes del menú del juego donde se puede observar como la opción seleccionada es la segunda debido al *focus* realizado sobre dicha opción.

Tras esto se creó un menú más atractivo (ver Figura 5.9) donde cada una de las opciones tenía un *focus* el cual hacía que el botón seleccionado cambiase su fondo de blanco a rojo. Posteriormente se añadieron mejoras en temas de animación y sonido. Se mejoró visualmente el disparo realizado durante el juego mediante un sistema de partículas (ver Figura 5.10) y se añadieron animaciones para las explosiones cuando los enemigos eran destruidos. Por otro lado se incluyó música durante el menú y pantallas informativas, además de dar sonido a las teclas del teclado cuando son pulsadas.

Figura 5.10: Muestra una serie de imágenes durante una partida donde se ve la animación de eliminar a un enemigo. Primero se fija el avión en la cruceta que se ve en la interfaz del juego, para después realizar el disparo y acabar con el enemigo que explota.

Por último se llevaron a cabo una serie de pruebas tanto a nivel de configuración que tuvieron como objetivo el ajuste de cada uno de los modos de juego en cuanto a velocidades de los enemigos, tiempos y número de enemigos, como a nivel de validación por parte de una serie de sujetos de prueba.

5.3. Pruebas realizadas

Para la realización de las pruebas se han empleado los dispositivos comentados en el Anexo **B**. Dichas pruebas podrían catalogarse en dos tipos: el primero de ellos se centra más en la configuración de todos los elementos de cada uno de los modos de juego para que haya una buena relación entre ellos dando como resultado una mejor experiencia para el jugador. En cuanto al segundo tipo se basa en realizar un test a una serie de sujetos con el fin de comprobar algunos aspectos tales como la sencillez de la aplicación, el éxito de la misma y posibles sugerencias para mejorarlo.

Tras realizar varias pruebas sobre posibles mejoras en la configuración en el prototipo inicial se llevaron a cabo una serie de cambios sobre los valores concernientes al número de aviones que aparecían y su frecuencia de aparición, y sobre todo en su velocidad. Esto fue debido a que tras superar varios niveles el movimiento de cada avión se incrementaba demasiado y era imposible ya no solo de destruir, sino de seguir. A continuación se muestran estos cambios con los valores iniciales y los finales respectivamente:

- Modo de juego Arcade.

  Aviones iniciales 5 e incremento de 5 más por oleada – Aviones iniciales 5 e incremento de 1 más por oleada.

  Velocidad inicial de 10 e incremento de 2 unidades por oleada – Velocidad inicial de 7 e incremento de 1 unidad por oleada.

- Modo de juego Survival.

  Velocidad inicial de 5 e incremento de 2 unidades por oleada con un factor de reducción de 15 – Velocidad inicial de 5 e incremento de 2 unidades por oleada con un factor de reducción de 20.

  Tiempo de aparición de un avión de 0.5 segundos – Tiempo de aparición de un avión de 1.5 segundos.

Cuando el juego estuvo listo, se llevo a cabo un test de prueba con varios sujetos que se ofrecieron voluntarios. Este grupo estaba formado por dos personas jóvenes y dos personas de mediana edad a los cuales se les entregaron los dispositivos que se necesitan para jugar y se les indicó donde estaba la aplicación en el teléfono móvil. Para el test no se les dieron ninguna instrucción ni idea de como debían usar todo el material disponible, con la salvedad de una pequeña descripción del juego y los propios materiales. El objetivo de estas pruebas era contemplar a cada uno de los sujetos y ver si eran capaces por ellos mismos, junto con la ayuda que ofrecía la aplicación creada, de poder llegar a jugar sin ningún contratiempo. Los resultados eran los esperados pues las personas jóvenes no tuvieron ningún problema y llegaron a jugar con total normalidad, mientras que las personas de mediana edad se encontraban algo más perdidas aunque una de ellas, con algo más de tiempo, logró emular a los sujetos más jóvenes. La conclusión obtenida fue que los jóvenes están más acostumbrados a las nuevas tecnologías y se adaptan a ellas mucho más rápido que las personas mayores. Esto unido a que la aplicación creada va dirigida más hacia este tipo de usuarios hace creer que la información que se ofrece es más que suficiente para poder jugar sin ningún tipo de problema. No obstante se aconseja leer previamente el manual de usuario (Ver Anexo **E**) antes de acceder a la aplicación.

Tras realizar el test se les formularon a los sujetos tres preguntas muy sencillas:

- ¿Qué le ha parecido el juego?
- ¿A encontrado alguna dificultad a la hora de usar todos los dispositivos que se requieren?
- ¿Cambiaría alguna cosa, o añadiría algo nuevo?

Haciendo un resumen de las respuestas ofrecidas, se llegó a la conclusión de que el juego gustaba ya que prácticamente ningún sujeto había podido disfrutar del uso de esta tecnología antes y le parecía algo innovador. Hay que añadir que varios de ellos se quejaron un poco debido a que la captura de vídeo se veía algo borrosa y eso hacia que en algunas ocasiones el sujeto se marease. Este dato ya se había contemplado y es debido al auto enfocado del dispositivo móvil con el que se llevaron a cabo las pruebas y el cual era imposible de quitar, por tanto se trata de un problema aislado y el cual no tiene que ver con la aplicación creada. Por último, alguno de ellos sugirieron algunas mejoras visuales durante el transcurso del juego, pero esto se plantea como trabajo futuro, ya que para llevar a cabo estas mejoras, se necesita la obtención de licencias que a la larga (muchos modelos) resulta algo costoso en cuanto a dinero se refiere.

6. Conclusiones

En este trabajo se ha conseguido implementar una aplicación para un teléfono móvil, un juego en nuestro caso, y podemos concluir que se han alcanzado los objetivos propuestos al comienzo del mismo (ver Sección 1.1). Para conseguirlo se han estudiado las tecnologías existentes y se han elegido aquellas que más se ajustaban a las necesidades de un juego de realidad mixta. Se han realizado dos modalidades de juego con una jugabilidad diferente en cada caso, en los cuales aparecen una serie de enemigos a los que hay que destruir pero cuyos objetivos a cumplir varían. Por otro lado se han creado una serie de pantallas con elementos gráficos donde se muestra al jugador información importante sobre como jugar y los estados en los que el juego se encuentra. Todo ello ha sido modelado mediante un entorno de desarrollo llamado *Unity* el cual, pese a dar algunos quebraderos de cabeza como ya se ha comentado, ha resultado de mucha utilidad. Por último indicar que el sistema final satisface con los requisitos del proyecto mencionados en la Sección A.1, tanto con los funcionales como los no funcionales.

La finalidad de prácticamente todas las aplicaciones móviles creadas es la comercialización o, en su defecto, difundirla para que todo el mundo pueda disfrutar de ella. En nuestro caso el medio por el cual se puede dar a conocer nuestro juego es *Google Play*, la plataforma de distribución digital de aplicaciones móviles para los dispositivos con sistema operativo *Android*. En las últimas convenciones realizadas centradas en videojuegos, como por ejemplo la famosa "E3" (*Electronic Entertainment Expo*), se ha visto el auge que esta teniendo está nueva tecnología y por ello se ha creído oportuno llevar una investigación del mercado actual, donde se han observado el potencial que puede llegar a tener nuestra aplicación o posibles competencias.

Figura 6.1: Foto tomada durante el evento E3 de 2015 donde se puede observar a un grupo de jóvenes jugando utilizando unas gafas de RV.

Se ha observado que ya existen numerosas aplicaciones que requieren del uso de unas "Google Cardboard" (marco de gafas creado por *Google* al igual que la store de *Android*) o marco de gafas de realidad virtual similar, casi todas ellas enfocadas a los videojuegos y a la visualización de paisajes tanto reales como virtuales, aunque hay algunas más de estas últimas. Hay que destacar que estas aplicaciones no se encuentran entre las más descargadas o mejor valoradas, y se cree que una de las causas de ésto es que no todo el mundo posee un marco de gafas de realidad virtual o en su defecto un teléfono móvil capaz de soportar esta tecnología.

En su defecto, se ha encontrado un top 5 de las mejores aplicaciones para "Cardboard":

- Roller coaster VR: Simulador de un montaña rusa.
- Cardboard: Simulador de la ya famosa versión de Google Earth.
- VR Cinema for Carboard: Conversor de videos en formato MP4 en una vista de pantalla dividida para simular la realidad virtual.
- Orbulus: Simulador de panoramas de 360 grados.
- Lamper VR: Juego de plataformas donde se debe ir recogiendo diferentes objetos y esquivar obstáculos.

(a) App Roller coaster (b) App Cardboard

Figura 6.2: Ejemplo de dos aplicaciones que podemos encontrar en *Google Play* y que requieren del uso de un marco de gafas de realidad virtual

Una vez se ha observado que el mercado para estas aplicaciones tienen salida se comprobó si existe algún competidor. Tras indagar en diversos lugares, no se observó ningún juego que mezcle el mundo real y el virtual, únicamente se vieron juegos creados completamente por ordenador. En consecuencia no se puede llevar a cabo una aproximación en cuanto a precio o descargas potenciales que pueda tener la aplicación creada. Hay que destacar que la mayoría de los juegos que hay en *Google Play* son completamente gratis, y ésto unido a las posibles mejoras que podrían llevarse sobre el juego creado (debido mayormente a la obtención de licencias para diferentes fines) hacen prever que una primera versión del mismo en caso de que se subiese a la store fuese totalmente gratuita.

Como bien se ha comentado en el párrafo anterior existe un margen de mejora del juego creado en cuanto a temas económicos se refiere (obtener mejores modelos de objetos virtuales,

etc.), pero aún con todo, la aplicación podría ser subida sin ningún problema y observar el recibimiento que podría tener un juego totalmente nuevo a lo que los usuarios han visto. Ver Anexo **D** para más información de cómo subir una aplicación a *Google Play*.

6.1. Líneas de trabajo futuras

Durante la realización del trabajo aparecieron una serie de temas que podrían ser interesantes para un futuro. Así pues estos temas se centran en la interacción con el mundo real, ya que durante la realización de este trabajo, este mundo es un mero fondo que no tiene ninguna interacción con los elementos virtuales. Por consiguiente sería buena opción mejorar la aplicación creada añadiendo control de los objetos virtuales utilizando objetos reales. También resaltar que podrían mejorarse la interfaz del juego si se obtuviesen las licencias necesarias para modelos de objetos más complejos dándole un toque más profesional. Añadir que con la llegada de la última versión *Unity* se encuentra disponible un nuevo soporte completo para dispositivos de realidad virtual como Oculus Rift y por tanto se podría integrar nuestro juego a estos dispositivos. Por otro lado, podría seguirse investigando en nuevas aplicaciones de realidad mixta dejando de lado el mundo de los juegos y centrándose en otros campos como la arquitectura y el arte, o incluso la educación.

6.2. Valoración personal

La realización de este trabajo fin de grado a nivel personal ha sido muy gratificante, donde he podido expandir mis conocimientos, aunque sea solo un poco, en un campo cada vez más extendido como es el caso de la realidad virtual, aumentada y mixta. La elección de un entorno de desarrollo como *Unity* me ha permitido adquirir unos conocimientos muy valiosos para mi futuro de cara al mundo laboral. Ésto unido a la implementación de la aplicación en un nuevo lenguaje de programación, el cual me era desconocido, como es C#, ha hecho que la experiencia haya sido muy constructiva desde el punto de vista profesional. En definitiva, este trabajo me ha permitido poner el broche final a mi formación académica y a una etapa muy importante en mi vida.

Bibliografía

[1] Unity
http://docs.unity3d.com/Manual
http://unity3d.com/es/learn/tutorials/modules

[2] Epic Games
http://epicgames.com/technology

[3] Valve
http://www.valvesoftware.com/company

[4] C#
http://es.wikipedia.org/wiki/C_Sharp
https://msdn.microsoft.com/es-es/library/ms228602%28v=vs.90%29.aspx

[5] Durovis Open-Dive
http://www.durovis.com/opendive.html

[6] Oculus Rift
https://www.oculus.com/en-us

[7] Google Glass
https://es.wikipedia.org/wiki/Google_Glass

[8] Headtracking
http://peted.azurewebsites.net/windows-phone-mobile-vr-gyro-head-tracking-in-unity3d
http://blog.heyworks.com/how-to-write-gyroscope-controller-with-unity3d

[9] Sonidos para videojuegos
http://www.flashkit.com

[10] Assets o modelos para videojuegos
http://tf3dm.com
https://www.yobi3d.com

[11] Comparativa de entornos de desarrollo
http://www.i-programmer.info/news/144-graphics-and-games/
8353-game-engines-for-free.html

[12] Realidad virtual
http://es.wikipedia.org/wiki/Realidad_virtual

[13] Realidad aumentada
http://es.wikipedia.org/wiki/Realidad_aumentada

[14] Realidad virtual
http://es.wikipedia.org/wiki/Realidad_mixta

[15] Realidad virtual VS Realidad mixta
http://www.google.com/design/spec-vr/designing-for-google-cardboard/a-new-dimension.html#

[16] Comparativa de SO en teléfonos móviles
https://blog.uchceu.es/informatica/ranking-de-sistemas-operativos-mas-usados-para-201

[17] Aplicaciones de Google Play con Google Cardboard
https://play.google.com/store/apps/collection/promotion_3001011_cardboard_featured_apps?hl=es_419
http://www.xatakandroid.com/aplicaciones-android/las-mejores-aplicaciones-para-cardbo

[18] Preguntas y respuestas sobre programación
http://forum.unity3d.com
http://stackoverflow.com

NOTA: Las direcciones de internet que aparecen en esta bibliografía están revisadas y accedidas a fecha 15/06/2015.

A. Requisitos del sistema

En este anexo se van a detallar el conjunto de requisitos que el sistema debe de satisfacer. Dichos requisitos se han catalogado como funcionales, que definen una función del sistema, y no funcionales, que son usados para juzgar la operación de un sistema en lugar de sus comportamientos específicos.

A.1. Requisitos funcionales

- RF-1 La aplicación deberá ofrecer un modo de juego de tipo Arcade.
- RF-2 La aplicación deberá ofrecer un modo de juego de tipo Survival.
- RF-3 La aplicación deberá mostrar las tres mejores puntuaciones para cada uno de los modos de juego.
- RF-4 La aplicación deberá mostrar una pantalla de ayuda para el usuario.

A.2. Requisitos no funcionales

- RNF-1 La aplicación deberá gestionar la pantalla partida del dispositivo para la simulación del efecto 3D, puesto que cada ojo verá una de las mitades del mismo.
- RNF-2 La aplicación deberá gestionar los movimientos de cabeza que el usuario realice, es decir, captar los movimientos con las herramientas ofrecidas por el dispositivo móvil utilizado y actuar en consecuencia, con las gafas de realidad virtual.
- RNF-3 La aplicación deberá gestionar la captura de la cámara del dispositivo empleado para la captura de vídeo del entorno que rodea al usuario.
- RNF-5 La aplicación deberá ofrecer la opción de salir del juego ya que será imposible el acceso al móvil mientras el usuario este jugando.
- RNF-6 Se requerirá un dispositivo móvil, un teclado bluetooth y un marco de gafas para introducir el dispositivo móvil.
- RNF-7 La aplicación funcionará al menos en dispositivos con un sistema operativo *Android*.

- RNF-8 La aplicación utilizará como lenguaje el español.
- RNF-9 El teléfono móvil deberá disponer de giroscopio.

B. Dispositivos empleados para probar el proyecto

En este anexo se detallan las tecnologías empleadas para realizar las pruebas de nuestro proyecto.

B.1. Marco de gafas de Realidad Virtual (Durovis-OpenDive)

Un marco de gafas de Realidad Virtual(RV) es un dispositivo que simula un casco de RV pero sin la pantalla donde se proyectan las imágenes. Debido a su falta de pantalla interna, este marco de gafas ofrece la opción de introducir un teléfono móvil que hará de la misma. Este dispositivo esta diseñado para que el desarrollo de imágenes 2D comunes se puedan convertir en 3D, dando así, un efecto de inmersión más realista.

Figura B.1: Durovis Open-Dive.Marco de gafas de Realidad Virtual empleado durante la realización del proyecto.

Para la realización de este proyecto se ha optado por Durovis-OpenDive (ver Figura B.1) como marco de gafas de Realidad Virtual debido a que en calidad-precio era la que más se ajustaba. Mejoraba la calidad de los marcos de cartón que se veían quizá demasiado frágiles y se ajustaban a las necesidades entre el resto de opciones en su versión de plástico.

B.2. Teléfono Móvil (BQ Aquaris)

Se trata de un aparato indispensable en la actualidad, sin embargo su auge ha sido un fenómeno muy reciente. Fue creado para satisfacer la demanda de comunicación móvil dentro de un espectro de radiofrecuencia limitado. El sistema debe acomodar miles de usuarios dentro de un espectro reducido para transmitir cada vez más información. La actual demanda no solo de la transmisión de voz, sino de imágenes, vídeos y acceso a Internet han hecho que sigan aumentando los requisitos del sistema.

Figura B.2: BQ aquaris E5.Teléfono móvil empleado durante la realización del proyecto.

Para la realización de este proyecto se ha optado por un smartphone BQ Aquaris (ver Figura B.2) como teléfono móvil. Dicha elección ha sido principalmente por su Sistema Operativo (SO). En un principio se barajaron dos opciones: *Android* y *Windows Phone*. Esto se debía principalmente al coste que hoy en día tienen estos smartphones frente a su gran competidor *IOS* de Apple. También se tuvo en consideración algunas estadísticas sobre el uso de cada sistema operativo por parte de la gente, dando como vencedor a *Android* frente al resto de opciones. Por último se tomó en consideración la comodidad para programar ya que en *Android* el SDK es totalmente gratuito y en caso de querer publicar tu aplicación para el resto de personas es mucho mas cómodo que con *IOS*.

B.3. Mini Teclado Bluetooth (Ol-link LL-AT-2)

Como su propio nombre indica se trata de un teclado más pequeño de lo normal que se conecta y comunica con su dispositivo principal a través de bluetooth. Estos dispositivos se utilizan generalmente con otros dispositivos portátiles, como tablets o teléfonos móviles. La popularidad de los teclados bluetooth ha llegado recientemente, en 2011, con el auge de los dispositivos portátiles.

Figura B.3: Ol-link LL-AT-2.Teclado bluetooth empleado durante la realización del proyecto.

Para la realización de este proyecto se ha optado por Ol-link LL-AT-2 (ver Figura B.3) como teclado bluetooth. Se bajaron dos ideas a la hora de interactuar con el teléfono móvil, ya que este sería inaccesible una vez se introdujese en el marco de gafas de RV. La primera de ellas y por la que se ha optado al final ha sido un mini teclado bluetooth ya que un joystick o mando podría haber problemas para conectarlo con el dispositivo. Así pues mediante un teclado se podría sincronizar con el teléfono fácilmente y además ofrece más posibilidades que un mando, como por ejemplo el poder escribir sin la necesidad de visualizar un teclado en la pantalla e ir eligiendo letras una a una.

C. Diagramas de actividad

A continuación se detalla de una manera más técnica la parte dinámica de nuestro juego. Esta tarea ha sido un poco difícil de plasmar en un diagrama debido a que *Unity* es un framework en el que las llamadas a las clases que aparecen en una escena es desconocido. Con esto se quiere decir que *Unity* tiene un orden de ejecución de sus funciones algo arbitrario. Si que es cierto que existe una jerarquía entre ellos, el método *Start()* se ejecuta antes que el método *Update()* por ejemplo, pero en lo referente a dos métodos *Update()* de diferentes clases, el orden de ejecución lo decide el propio *Unity*. Así pues, se ha optado por utilizar un diagrama de actividad ya que ilustra la naturaleza dinámica de un sistema mediante el modelado del flujo ocurrente de actividad en actividad, donde cada una de ellas representa una operación en alguna clase del sistema y que resulta en un cambio en el estado del mismo.

Como bien se ha comentado en el párrafo anterior, el no saber el orden de ejecución de los métodos de cada una de las clases que intervienen en la escena, ha hecho que se tenga que hacer una aproximación de cómo se ejecutaría todo si se hiciese de una manera secuencial. Como resultado se ha intentado representar una serie de estados, en este caso cada una de las actividades que aparecen en el diagrama, agrupando varias partes del código implementado para obtener una representación a más alto nivel sin entrar a cada una de las condiciones existentes dentro del mismo código.

El funcionamiento de ambos modos parece similar en algunos aspectos, pero si se miran bien los diagramas existentes de cada uno, se puede apreciar que hay muchas variaciones. En el modo de juego Arcade (ver Figura C.1) se sigue un patrón o bucle centrado en los enemigos vivos por cada oleada, el cual debe tener en cuenta diversos factores como son el tiempo y los enemigos vivos entre otros, mientras que en el modo de juego Survival (ver Figura C.2), todo gira entorno a las vidas que el jugador posee y como éste las pierde. Por otro lado, la forma de gestionar la dificultad varía entre cada modo teniendo en cuenta diferentes causas como el fin de una oleada o el tiempo que lleva el jugador eliminando enemigos. Por otro lado, lo que ambos modos de juego comparten es la forma de instanciar enemigos en la escena aunque la forma de moverse de los mismos, entorno al jugador varia (vuelo en círculos alrededor del jugador en Arcade o vuelo hacia el jugador en Survival).

Figura C.1: Diagrama de actividad de una partida Arcade que representa los flujos de trabajo paso a paso. En este diagrama se puede observar el proceso que sigue una partida en el modo de juego Arcade donde el jugador debe completar todas las oleadas, eliminando los enemigos que van apareciendo en cada una de ellas.

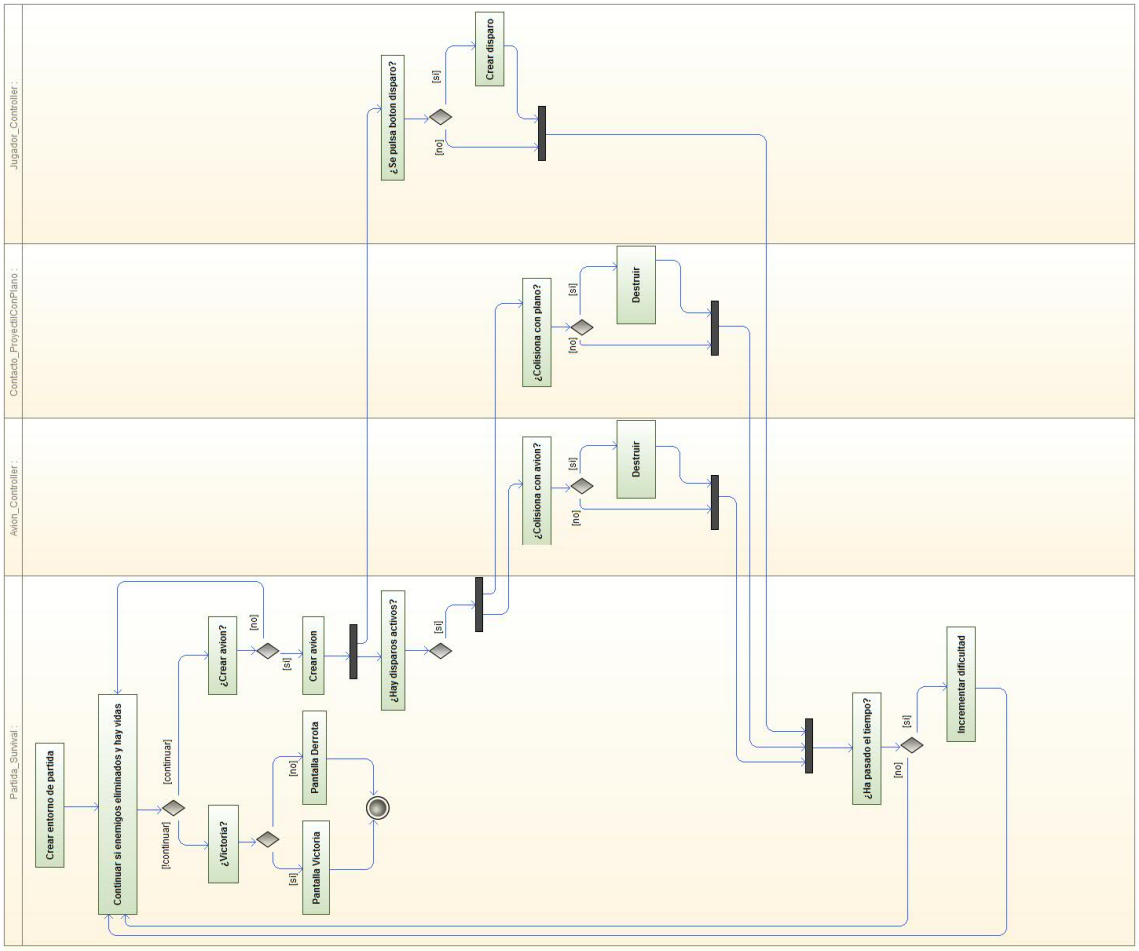

Figura C.2: Diagrama de actividad de una partida Survival que representa los flujos de trabajo paso a paso. En este diagrama se puede observar el proceso que sigue una partida en el modo de juego Survival donde el jugador debe eliminar los enemigos que van apareciendo y se acercan hacia él antes de que se quede sin vidas.

D. Instrucciones para subir una aplicación a Google Play

En este anexo se detallan los pasos que deben ser seguidos para subir una aplicación a *Google Play*.

D.1. Pasos previos

Lo primero que se debe hacer antes de querer subir cualquier aplicación a la store de *Android* es obtener una cuenta de desarrollador. Para obtener dicha cuenta hay que registrarse e iniciar sesión con una cuenta de Google, después aceptar el acuerdo para desarrolladores y pagar 25 dólares (22 euros aproximadamente) como cuota de registro, y por último rellenar la información personal de la cuenta. Además de todos pasos es conveniente consultar los países de distribución en los que se puede vender y distribuir aplicaciones para evitar posibles problemas en un futuro.

D.2. Subir aplicación a *Google Play*

Una vez se ha obtenido una cuenta de desarrollador ya se pueden subir aplicaciones a *Google Play* a través de la Consola de *Google Play* para desarrolladores. Los pasos a seguir son muy simples, en primer lugar hay que buscar el botón "añadir nueva aplicaciónz una vez seleccionada, se procede a subir el fichero APK y rellenar la ficha de Play Store para añadir la información de la aplicación. En cuanto a los archivos APK hay una serie de restricciones como su tamaño, el cual no debe sobrepasar los 50 MB ya que si esto sucede los recursos adicionales se almacenarán como archivos de expansión, o el número de archivos de los mismos ya que si un solo archivo APK es insuficiente para todos los dispositivos, se pueden subir varios con la misma ficha de la aplicación para las diferentes configuraciones. Por otro lado, la ficha de Play Store que hay que rellenar contiene la siguiente información:

- Idiomas y traducciones: en esta sección se indica el idioma de nuestra aplicación, inglés por defecto, y si se quieren añadir traducciones en otros idiomas o no.
- Detalles del producto: en esta sección se indica la información del producto, es decir, la información de nuestra aplicación. Los campos a rellenar son: título, descripción breve,

descripción completa y novedades de esta versión. Hay que destacar que la única restricción en todas ellas es el número de caracteres que se pueden utilizar.

- Categorización: en esta sección se indica el tipo de aplicación que se desea subir, aplicación o juego, y se le atribuye la categoría más adecuada (salud, entretenimiento, etc.).
- Recursos gráficos: en esta sección se suben, si el usuario lo desea, imágenes o vídeos sobre la aplicación para que los usuarios interesados puedan tener una primera impresión de como es.
- Datos de contacto: en esta sección se indican los recursos de asistencia para tu aplicación. Como mínimo debe aparecer un correo electrónico, pero se pueden incluir también, por ejemplo, teléfono o sitio web.

Cuando toda esta información ha sido rellenada la aplicación esta lista para subir a la store.

D.3. Comprobar estado de publicación

Una vez la aplicación ha sido subida a *Google Play* en la consola de desarrollo se puede observar el último estado de publicación de la misma. Si la aplicación subida es nueva, es decir, no es una actualización de una versión ya existente, podrán aparecer diferentes estados de la aplicación, desde que ha sido rechazada hasta que ha sido publicada, pasando por que la aplicación esta pendiente de publicar. Por el contrario si se trata de una nueva versión de la misma, sucederá algo similar a lo mencionado anteriormente pero con la actualización.

D.4. Precio y distribución

En la pagina "Precio y distribución", se podrá establecer si la aplicación subida será de pago o gratuita, seleccionar los países en los que estará disponible (acordarse de revisar los países de distribución) y si podrá decidir si se quiere distribuir la aplicación en otros dispositivos *Android*.

E. Manual de usuario

En este anexo se detalla un pequeño manual de usuario del juego desarrollado en este TFG.

E.1. Descripción de la aplicación

La aplicación creada, un juego en este caso, recrea la escena de la película *"King Kong"* donde el protagonista debe defenderse del ataque de los aviones enemigos que quieren acabar con él. En este juego estaremos en la piel del gorila en primera persona y deberemos destruir los aviones enemigos que aparecen a nuestro alrededor. A parte de las opciones que ofrece respecto a la funcionalidad del juego, se muestran diferentes pantallas informativas relevantes para el jugador.

Entre las opciones ofrecidas hay que destacar las siguientes:

- Modo de juego Arcade.
- Modo de juego Survival.
- Records.
- Ayuda.

En las siguientes páginas se detalla el uso y funcionamiento de la aplicación para un eficaz manejo.

E.2. Manual de usuario de la aplicación

E.2.1. Pantalla Unity

Es la pantalla inicial y se muestra nada más arrancar la aplicación. Esta pantalla contiene el logotipo del framework *Unity*.

Figura E.1: Pantalla inicial con el logotipo de *Unity*.

Tras unos tres segundo aparece la siguiente pantalla automáticamente.

E.2.2. Pantalla modo de empleo

Pantalla que aparece tras mostrar el logotipo de *Unity*. En ella aparecen una serie de instrucciones a seguir para el uso correcto del marco de gafas de realidad virtual y el teléfono móvil. Cabe destacar que no habrá que realizar estos pasos hasta que aparezca la pantalla de carga del juego donde se le indica al jugador que se coloque las gafas.

Figura E.2: Pantalla donde se da una serie de instrucciones al usuario para el correcto uso de las tecnologías que serán empleadas.

Tras diez segundos aparece la siguiente pantalla automáticamente.

E.2.3. Pantalla controles

Pantalla donde se muestran las teclas del teclado que serán usadas en el juego, ya sea para seleccionar opciones en los menús, como durante cualquier modo de juego.

A continuación se describe cada una de las teclas junto con su funcionalidad:

- Tecla "1": Tecla que sirve para pausar el juego en cualquier modo de juego donde el jugador se encuentra.
- Tecla "0": Tecla que sirve para navegar por el menú del juego, es decir, permite bajar opción a opción dentro del mismo.
- Tecla "Shift": Tecla que tiene dos funciones. La primera de ellas sirve para seleccionar la opción del menú que el jugador desee, y la segunda sirve para disparar dentro de cualquier modo de juego.

Figura E.3: Pantalla donde se muestra un teclado con las teclas que serán empleadas durante el juego.

Tras diez segundos aparece la siguiente pantalla automáticamente.

E.2.4. Pantalla carga

Pantalla en "dos fases". En la primera se informa al jugador que se esta cargando el juego y que debe esperar, mientras que en la segunda el juego se ha cargado y se indica al jugador que debe colocarse las gafas. En esta última fase también se le informa al jugador que adopte una postura natural y mire al frente, ya que será esta posición la que se tome como referencia a la hora de jugar.

Figura E.4: Pantalla que informa al jugador que el juego se está cargando.

Figura E.5: Pantalla que informa al jugador que el juego se ha cargado. En este momento se le indica que debe colocarse las gafas y pulsar una tecla determinada para continuar.

Al pulsar la tecla que se le indica al jugador, 'seleccionar', aparece la pantalla del menú.

E.2.5. Pantalla menú

Pantalla donde se muestra el menú del juego con las opciones que éste ofrece al jugador. Destacar que el color de fondo de las opciones cambiará a rojo cuando navegamos en ellas.

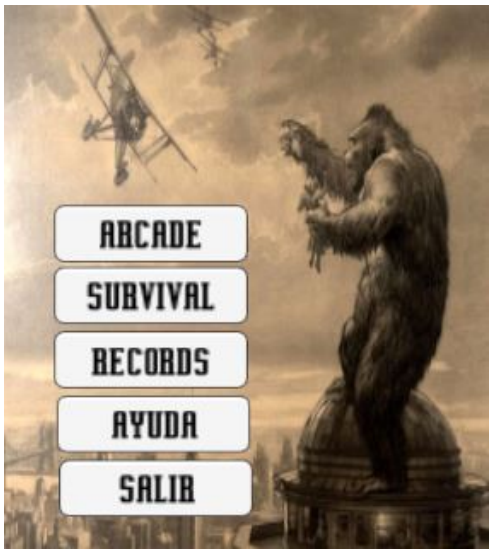

Figura E.6: Pantalla donde se muestra el menú del juego con las opciones que ofrece.

A continuación se describe cada una de las opciones que ofrece esta pantalla y los posibles errores que puedan ocurrir.

E.2.6. Pantalla modo de juego Arcade

Al seleccionar esta opción se muestra una pantalla en la cual aparece una captura del mundo real a través del teléfono móvil como si estuviésemos viendo nosotros mismos nuestro alrededor, y una interfaz de juego con el tiempo transcurrido, el número de las oleadas finalizadas, los enemigos y la puntuación que el jugador lleva. Será en este momento cuando tras informar al jugador que se aproxima una nueva oleada, aparezcan los enemigos los cuales habrá que destruir.

Figura E.7: Pantalla donde transcurrirá el modo de juego Arcade. En ella se puede ver una captura del mundo real que en este caso es un fondo azul, y una interfaz con la información del juego.

Al finalizar el modo de juego se mostrará una pantalla diferente dependiendo del estado de la partida: victoria o derrota.

E.2.7. Pantalla modo de juego Survival

Al igual que en el modo de juego Arcade, esta opción muestra una pantalla en la cual aparece una captura del mundo real a través del teléfono móvil como si estuviésemos viendo nosotros mismos nuestro alrededor, y una interfaz de juego con las vidas que le quedan al jugador, los enemigos y la puntuación que el jugador lleva. Una vez entrado en este modo de juego, el jugador deberá sobrevivir todo lo que pueda al ataque de los aviones enemigos que van hacia él.

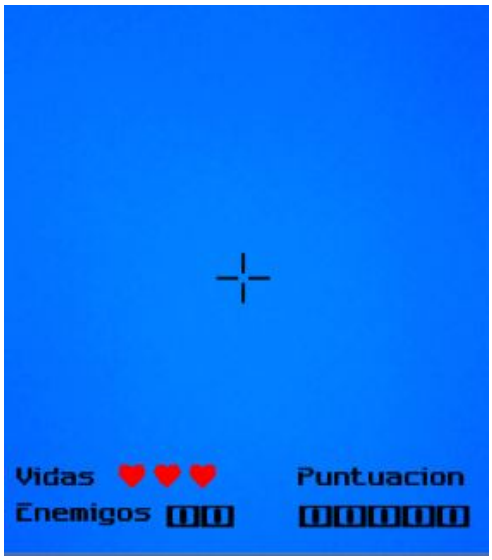

Figura E.8: Pantalla donde transcurrirá el modo de juego Survival. En ella se puede ver una captura del mundo real que en este caso es un fondo azul, y una interfaz con la información del juego.

Al finalizar el modo de juego se mostrará una pantalla diferente dependiendo del estado de la partida: victoria o derrota.

E.2.8. Pantalla records

Al seleccionar esta opción se muestra una pantalla en la cual aparecen los récords obtenidos por el jugador. En ella se podrán observar las tres mejores puntuaciones de cada uno de los modos de juego.

Figura E.9: Pantalla donde se podrán observar las tres mejores puntuaciones de cada uno de los modos de juego.

Al pulsar la tecla que se le indica al jugador, 'seleccionar', aparece la pantalla del menú.

E.2.9. Pantalla ayuda

Al seleccionar esta opción se muestra una pantalla en la cual aparecen unas pequeñas descripciones sobre cada uno de los modos de juego, para que el jugador sepa en todo momento a lo que se enfrenta y lo que debe hacer.

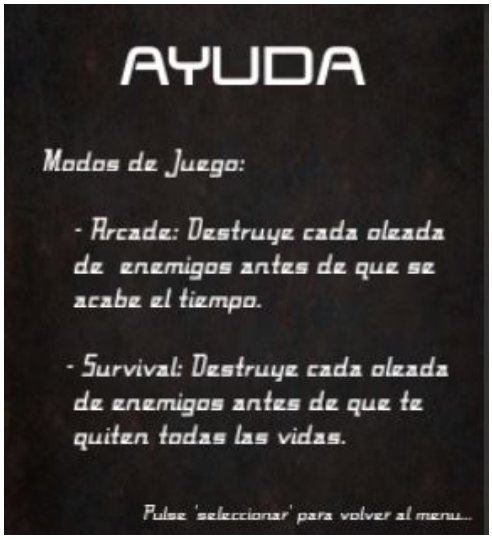

Figura E.10: Pantalla donde se informa al jugador sobre los objetivos a realizar en cada uno de los modos de juego.

Al pulsar la tecla que se le indica al jugador, 'seleccionar', aparece la pantalla del menú.

E.2.10. Pantalla error

Esta pantalla aparece al entrar en uno de los modos de juego que se ofrecen y se detecta que el teléfono móvil, el cual se esta empleando, no cumple con los requisitos que se esperan, en este caso, que no disponga de giróscopo.

Figura E.11: Pantalla donde se informa al jugador de que su teléfono móvil no dispone de las tecnologías que se esperaban y por lo tanto no puede ejecutarse.

Tras pulsar cualquiera tecla se sale del juego al menú del teléfono móvil.

E.2.11. Pantallas fin del juego

Esta pantalla aparece tras jugar cada uno de los modos de juego, Arcade y Survival, e informa al jugador del estado de la partida.

(a) Pantalla victoria (b) Pantalla derrota

Figura E.12: Pantalla donde se informa al jugador de que ha ganado o perdido respectivamente de izquierda a derecha junto con la puntuación obtenida.

Al pulsar la tecla que se le indica al jugador, 'seleccionar', aparece la pantalla del menú.